

MarketBrowser™

Function Reference Guide

L E A D I N G • M A R K E T • T E C H N O L O G I E S

One Kendall Square • Building 100 • Cambridge MA 02139 • (617) 494-4747 • Fax (617) 494-4788

Confidential. This document contains trade secrets of Leading Market Technologies, Inc.

Copyright 2010

SOFTWARE LICENSE AGREEMENT

1. NOTICE. THIS IS A LEGAL AGREEMENT BETWEEN YOU AND LEADING MARKET TECHNOLOGIES, INC ("LMT"). PLEASE READ THIS AGREEMENT CAREFULLY BEFORE INSTALLING OR USING LMT'S SOFTWARE, DOCUMENTATION, DATA OR SERVICES (THE "SOFTWARE"). BY OPENING THE SEALED MEDIA PACKAGE, IF ANY, DOWNLOADING, INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE WITH THE TERMS AND CONDITIONS OF THIS AGREEMENT, DO NOT INSTALL OR USE THE SOFTWARE AND PROMPTLY DELETE ALL COPIES OF THE VIEWER IN YOUR POSSESSION OR UNDER YOUR CONTROL. IF YOU HAVE ANY PHYSICAL MANIFESTATIONS OF THE SOFTWARE, THEN PROMPTLY RETURN THE MEDIA PACKAGE AND ALL ACCOMPANYING ITEMS (INCLUDING MATERIALS AND PACKAGING), ALONG WITH PROOF OF PAYMENT, TO US, AT THE ADDRESS BELOW, OR OUR AUTHORIZED DEALER FOR A FULL REFUND.

2. Permitted Uses. Subject to the terms and conditions of this Agreement, you are granted the following limited nonexclusive rights to use the Software:

(A)**Grant of Single User License.** You may use one copy of the Software installed on a single hard disk drive via a single terminal connected to a single computer CPU. You may not network the Software or otherwise use it on any other computer CPU or terminal.

(B)**Right to Copy.** You may make one copy of the Software solely for backup and archival purposes, provided that the original, whether on diskette or an electronic original, and each copy is kept in your possession and control, and provided you reproduce our copyright notice on the copy. You may not copy the documentation or other written materials included in the Software.

3. Prohibited Uses. You may not, without written permission from us:

(A) Use, copy, modify, merge, create derivative works of, or transfer copies of the Software or documentation except as provided in this Agreement;

(B) Use any backup or archival copies of the Software (or allow someone else to use such copies) for any purpose other than to replace the original copy in the event it is destroyed or becomes defective; or

(C) Disassemble, decompile or "unlock", reverse translate, reverse engineer, or in any manner decode the Software for any reason, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

4. Ownership and License. This agreement grants you only a license to use the Software. Title, ownership and intellectual property rights shall remain with us. We continue to own all copies of the Software. Your rights to use the Software are specified in this Agreement, and we retain all rights not expressly granted to you in this Agreement including, but not limited to, rights reserved to us or protected by patent, copyright, and trade secret laws and international treaty provisions. The license granted hereunder shall not be construed to confer any rights upon you by implication, estoppel or otherwise as to the Software or documentation not specifically set forth herein. Nothing in this Agreement constitutes a waiver of our rights under U.S. patent or copyright law or any other national, federal or state law.

5. Limited Warranty. We make the following limited warranties for a period of thirty (30) days from the date

you acquired the Software from us or our authorized dealer: (a) The media and documentation, if any, will be free from defects in materials and workmanship under normal use. (b) The Software in this package will materially conform to the documentation that accompanies it.

(A)Limited Remedy. Our entire liability and your sole and exclusive remedy for breach of the limited warranty set forth in this Section 5 shall be, at our option, either (a) return of the price actually paid for the Software or one hundred dollars (\$100), whichever is greater, or (b) replacement of the Software; provided, however, you (i) return all of the Software, and any copies thereof, and the documentation to us or to the authorized dealer from whom you acquired it, along with a dated proof of purchase; or (ii) certify in writing to us that all electronic copies of the Software and documentation and any archival copies thereof have been destroyed, and in either case specifying to us the problem in writing.

(B)WARRANTY DISCLAIMER. WE DO NOT WARRANT THAT THIS SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT ITS OPERATION WILL BE UNINTERRUPTED OR ERROR-FREE. EXCEPT FOR THE EXPRESS LIMITED WARRANTY SET FORTH IN THIS SECTION 5, THE SOFTWARE AND THE DOCUMENTATION ARE LICENSED “AS IS.” WE MAKE NO REPRESENTATIONS AND EXTEND NO WARRANTIES OF ANY KIND AND DISCLAIM ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT.

Some states do not allow the exclusion of implied warranties, so the above exclusion may not apply to you. This limited warranty gives you specific legal rights, and you may also have other legal rights, which vary from state to state.

6.LIMITATION OF LIABILITY. OUR LIABILITY TO YOU FOR ANY LOSSES SHALL BE LIMITED TO DIRECT DAMAGES, AND IN NO CASE SHALL EXCEED THE AMOUNT ORIGINALLY PAID FOR THE SOFTWARE OR ONE HUNDRED (\$100.00) DOLLARS, WHICHEVER IS GREATER. IN NO EVENT AND UNDER NO LEGAL THEORY, INCLUDING TORT, CONTRACT OR OTHERWISE, SHALL WE BE LIABLE TO YOU FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOSS OF PROFITS) EVEN IF WE HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN THE EVENT OF LITIGATION INVOLVING THE SOFTWARE, YOU AGREE THAT AFTER FINAL JUDGEMENT AND EXHAUSTION OF APPEALS THE NONPREVAILING PARTY SHALL PAY ALL REASONABLE COSTS OF LITIGATION OF THE PREVAILING PARTY, INCLUDING, BUT NOT LIMITED TO, REASONABLE ATTORNEY’S FEES.

Some jurisdictions do not allow these limitations or exclusions, so they may not apply to you.

7. Export Controls. None of the Software or underlying information or technology may be downloaded or otherwise exported or reexported (i) into (or to a national or resident of) Cuba, Iraq, Libya, Yugoslavia, North Korea, Iran, Syria or any other country to which the U.S. has embargoes goods; or (ii) to anyone on the U.S. Treasury Department’s list of Specially Designated Nationals or the U.S. Commerce Department’s Table of Denial Orders. By downloading or using the Software, you are agreeing to the foregoing and you are representing and warranting that you are not located in, under the control of, or a national or resident of any such country or on any such list. You agree that you will not directly or indirectly transfer the Software or documentation to any country to which such transfer would be prohibited by the U.S. Export Administration Act and the regulations issued thereunder. You further agree that should your country require registration of this license agreement you will obtain the appropriate licenses and registrations at your own expense.

8. Term and Termination. The term of this license shall be 99 years from the date of your purchase of this license, or any other term agreed to in a separate writing between you and LMT. This license and your rights to use this Software automatically terminate if you fail to comply with any provisions of this Agreement.

Upon termination, you will destroy all copies of the Software and documentation. Additionally, if the purchase of this license to use the Software includes any continuing payments or renewal fees, termination will also occur upon non-payment of said payments and fees without a requirement of notice of termination.

9. Confidentiality. You acknowledge and agree that to the extent that the Software discloses proprietary information of ours, such proprietary information constitutes valuable trade secrets of ours any may not be disclosed by you to any party at any time.

10. Miscellaneous Provisions. This Agreement will be governed by and construed in accordance with the substantive laws of the Commonwealth of Massachusetts. Any and all disputes arising hereunder shall be resolved only in courts located in the Commonwealth of Massachusetts and are subject to the exclusive jurisdiction of that court. The parties hereby irrevocably submit and consent to the exclusive jurisdiction of such courts and waive any defenses or objections thereto. The application of the United Nations Convention on Contracts for the International Sale of Goods, as amended, is expressly excluded. This is the entire agreement between us relating to the Software and supersedes any prior purchase orders, communications, advertising, or representations concerning the Software. If any provision of this Agreement is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. No change or modification of this Agreement will be valid unless it is in writing and is signed by a duly authorized Officer of our company. Computer records stored in reasonably secure conditions on the computer system of either party shall be accepted as evidence of communication, license agreement, and payments made between the parties.

Canadian Transactions. If you acquired this Software in Canada, France, or other French-speaking countries, you agree to the following:

The parties hereto have expressly required that the present Agreement and its Exhibits be drawn up in the English language. / Les parties au présent contrat ont expressément exigé que ce contrat ainsi que ses Annexes soient rédigées en langue anglaise.

If you have any questions about this Agreement, write to us at Leading Market Technologies, Inc., One Kendall Square, Building 100, Cambridge MA 02139 USA.

About This Manual

The listing of worksheet functions that appears starting below is organized by the type of operation that each function or command performs. The main body of the manual consists of an alphabetical listing of function definitions.

Table of Contents

Worksheet Function and Command Categories

Annotation

COMMENT	Sets the comment for the first series in a window
GETCOMMENT	Returns the comment for the first series in a window
GETSCOMMENT	Returns the comment for a window or variable
LEGCUR	Inserts a legend via cursor for all the series in the window
LEGEND	Inserts a legend at explicit x and y points
LINEANN.....	Draws an explicit line
LINECOPY	Copies a polyline created with LINECUR
LINECUR	Brings up a freehand line drawing cursor in a window
LINEDEL.....	Deletes a polyline created with LINECUR
LINEMOVE.....	Moves a polyline created with LINECUR
SETCOMMENT	Sets the comment for the first series in a window
SETVCOMMENT	Sets the comment field of a variable
TEXTANN	Draws a left-justified block of text
TEXTCUR	Brings up a freehand text annotation cursor in a window
TEXTDEL	Deletes a block of text created with TEXTCUR
TEXTEDIT	Edits text annotation
TEXTMOVE	Moves a block of text created with TEXTCUR
TLABEL	Labels the points in a series

Coordinate Manipulation

CLEARXLABEL.....	Clears the label for the x-axis
CLEARYLABEL.....	Clears the label for the y- axis. See CLEARXLABEL
DELAY	Delays a series by n number of points
DELTA	Displays the delta-x value (1/sample rate)
DTSETX	Sets the visible x-axis range between two dates and/or times
GETXL	Gets the leftmost x-coordinate
GETXR	Gets the rightmost x-coordinate. See GETXL
GETXTIC	Gets the x-axis tic interval. See GETXL
GETYB	Gets bottom y-coordinate. See GETXL
GETYT	Gets the top y-coordinate. See GETXL
GETYTIC	Gets the y-axis tic interval. See GETXL
LAG	Shifts a series right by n number of points along the x-axis
LEAD.....	Shifts a series left by n number of points along the x-axis
SETAORIX	Sets the orientation of the x-axis label
SETAORIY	Sets the orientation of the y-axis label. See SETAORIX
SETAROTX.....	Sets the rotation of the x-axis label
SETAROTY	Sets the rotation of the y-axis label. See SETAROTX
SETAVDEFX.....	Sets the vertical default rotation on the x-axis

SETAVDEFYSets the vertical default rotation on the y-axis
 SETDELTAChanges the delta-x value of series
 SETTICKSets the tick interval on the x- and y- axes
 SETTORIXSets the orientation of the tick label along the x-axis
 SETTORIYSets the orientation of the tick label along the y-axis
 SETTROTSets the rotation of the tick label along the x-axis
 SETTROTXYSets the rotation of the tick label along the y-axis
 SETTVDEFX.....Sets the vertical rotation of the tick label on the x-axis
 SETTVDEFY.....Sets the vertical rotation of the tick label on the y-axis
 SETXSpecifies the x-axis coordinate range
 SETXLABEL.....Sets the label along the x-axis
 SETXOFFSETSets the starting point of a series
 SETXY.....Specifies the overall coordinate range
 SETXLOGToggles the log scales for the x-axis
 SETYSpecifies the y-axis coordinate range
 SETYLABEL.....Sets the label along the y-axis. See SETXLABEL
 SETYLOGToggles the log scales for y-axis
 SPANX.....Restricts the scale along the x-axis to a range of the window
 SPANYRestricts the scale along the y-axis to a range of the window
 STAGGERXStaggers the x-axis scale display
 STAGGERY.....Staggers the y-axis scale display. See STAGGERX

Curve Fitting

LINREG.....Calculates the best linear fit
 LINREG2.....Performs a linear regression of two series
 POLYFITPerforms a least square fit
 POLYGRAPHGraphs polynomial coefficients
 SPLINECalculates cubic spline interpretation

Data Input/Output Functions

READA.....Reads an ASCII data file
 READAHISTReads tables of historical or intraday data
 READANYHISTReads in any file of ASCII historical data
 READBReads a binary data file
 READBHISTReads a binary file of historical data
 READDTReads an ASCII data file of dates and times
 READTABLEReads tables of ASCII data
 TOCONTINUOUSCopies an input series or matrix to a continuous time series
 TODISCRETECopies an input series or matrix to a discrete time series
 WRITEA.....Writes a series to an ASCII file
 WRITEAHISTWrites tables of historical intraday data to an ASCII file
 WRITEBWrites a series to a binary file
 WRITEBHISTWrites a table of historical intraday binary data to a file
 WRITETABLEWrites a table of data to an ASCII file

Data Reduction and Editing

CLIP.....Sets outliers to min and max y values
 CONCATConcatenates series (end to end)
 DECIMATELinearly removes points from a series

DELETEDeletes points from series when the corresponding point in the binary control series is non-zero

DTCONCATConcatenates two conforming series while respecting their timebase

EXTRACTCuts pieces of a series

GETPTDisplays the values of nth point

GETRAWPTReturns the value of the nth point, including NAs

INTERPOLATE.....Linearly adds points to a series

LOOKUPPicks selected points from a table

MERGESplices several series

NEGATE.....Multiplies a series by -1

REMOVE.....Removes points from a series

REPLICATEConcatenates a series with itself

REVERSE.....Reverses the order of points in a series

SETPTModifies the value of a single point in a series

Data Type Conversion

CARTESIANConverts input to Cartesian coordinates

CONJUGATECalculates the complex conjugate

IMAGINARYCalculates the real component of an imaginary series

MAGNITUDEReturns the magnitude component of a series

PHASE.....Calculates the phase angle of a complex expression

POLARConverts input to magnitude/phase form

REALFinds the real component of a complex series

Date and Time Manipulation

ADDBDAYAdds n valid business days to the base Julian date

BARCONVERT.....Converts the periodicity of trading bars

CONFORM.....Defines how MarketBrowser deals with different x offsets

DEFDATE.....Sets the default beginning date of a time series

DEFTIMESets the default beginning time for a time series

DEFHUNITSSets the default horizontal units for a worksheet

DATESTRReturns the corresponding index number of a date

GETDATEGets the system or series date

GETTIME.....Gets the system or series time. See GETDATE

JULDAYReturns the day of the week for a given Julian date

JULSTRCalculates a Julian date from a string

NAFILLReplaces NA values based on other known data points

PERCONVERTConverts historical data to a different periodicity

SETDATESets the beginning date for a series

SETHUNITS.....Sets the horizontal units

SETNAVALUE.....Replaces NAs with user-defined input

SETTIME.....Sets the beginning time for a series. See SETDATE

SETVDATESets the beginning date for a series contained in a variable

SETVHUNITSets the horizontal units of a variable

SETVTIME.....Sets the beginning time for a series contained in a variable. See SETVDATE

SETVUNITS.....Sets the vertical units

STRDATE.....Returns the string form of a date in a window

STRTODReturns the string form of the time in a window

STRJULConverts an integer Julian date to a string

TODSTRReturns the index number for a given date

DDE Interfacing

DDEADVISE.....Retrieves a series item from a DDE conversation when it changes
DDEEXECUTEExecutes a command in another application
DDEGETDATARetrieves a series item from a DDE conversation
DDEGETLINKRetrieves a DDE link name from the clipboard
DDEINITIATE.....Begins a DDE conversation
DDELINK.....Initiates a DDE conversation, then retrieves a series item whenever
the item changes
DDEPOKE.....Sends data to a DDE conversation in string form
DDEREQUESTRetrieves a string item from a DDE conversation
DDESTATUSReports the error status of the last DDE operation
DDETERMINATE.....Terminates a DDE conversation
DDEUNADVISE.....Ends a previous DDEADVISE operation
DDEUNLINK.....Ends a previous DDELINK operation

Digital Filter Functions

BANDPASSDesigns an FIR linear phase bandpass filter
BANDSTOPDesigns an FIR linear phase bandstop filter
BUTTERWORTH.....Designs an IIR Butterworth filter
CASCADE.....Filters a time domain input with an IIR filter
CHEBY1Designs an IIR Chebychev I filter
CHEBY2.....Designs an IIR Chebychev II filter
DIFFDESIGNDesigns an FIR differentiator
ELLIPTICDesigns an IIR Elliptical filter
FIREvaluates a FIR difference equation
HIGHPASSDesigns a FIR linear phase highpass filter
IIREvaluates an IIR difference equation
LOWPASSDesigns a FIR linear phase lowpass filter

Display Manipulation

BARSDisplays a series as bars
CANDLESTICK.....Displays a series as Japanese Candlesticks. See ERRORBAR
COLLAYOUTArranges windows by column
COLPOSReturns the last position of a crosshair cursor in a window
COMPRESSH.....Compresses a series horizontally
COMPRESSV.....Compresses a series vertically. See COMPRESSH
CURPOS.....Returns the last position of the point cursor
CURPOS2.....Returns the position of the second cursor
CURRENTFOCUS.....Returns the current focus in a window
CURSOROFFTurns the cross-hair cursor off
CURSORON.....Turns the cross-hair cursor on. See CURSOROFF
EQUIVOL.....Creates an equivolume plot of price and volume data
EXPANDH.....Expands a series horizontally
EXPANDV.....Expands a series vertically. See EXPANDH
FOCUS.....Sets the input focus for overlaid series
FREEZE.....Turns automatic real-time scaling of x- and y-axes on or off
GETDTFORMATReturns a window's date and time formatting style

SERCOLOR.....Specifies the series color
 SETCANVASResizes/locates the application frame
 SETCOLOR.....Sets the series color
 SETDTFORMAT.....Sets the date/time formatting in a window
 SETFORMATSets the display type for numerical values
 SETGCOLOR.....Sets the global color parameter
 SETLINESets the line style
 SETLINEWIDTH.....Sets the width of a line
 SETPLOTMETHODSpecifies the method used when drawing plots
 SETPLOTSTYLE.....Specifies the plot style
 SETPLOTTYPESpecifies the plot type
 SETSYMBOL.....Specifies the symbol used to mark data points
 SETWSIZESets the size of a window
 SETTICKSets the tick spacing on the axes
 SETVPLOTSTYLESets plotting style of a variable
 SETVHUNITSets the horizontal units of a variable
 SETVUNITS.....Sets the vertical units of a variable
 SETWLIKECopies attributes of one window to another
 SETWMARGINSets the plotting area for a given window
 SETWMARGINALL.....Sets the plotting area for all windows in the worksheet
 STEPSDisplays a series as steps
 STICKS.....Displays a series as vertical sticks
 SYNC.....Sets the sync mode that controls scaling and scrolling
 TICKFORM.....Display data points in tick chart form
 TILEArranges the screen into equal-sized windows
 TABLEVIEWSets the plotting style of a window to a table of numbers
 TOOLBAR.....Edits the properties of an MarketBrowser toolbar
 UNOVERPLOT.....Removes one or more overplotted series
 UNPOPWINDOW.....Unzooms a window
 WFSETSets the attributes of a waterfall plot
 WINBOXTurns on or off the display of the auto legend in the current window
 WINCOLOR.....Specifies the background window and series color
 WINFORMATSets a window's numeric formatting attributes
 ZOOM.....Expands the window size

File Manipulation

COPYFILECopies a file
 DELFILEDeletes a file
 DIREXISTSChecks to see if a directory exists
 FCLOSE.....Closes a file
 FCLOSEALL.....Closes all open files
 FFLUSH.....Flushes a buffer to the file
 FGETSGets a string from an open file
 FILEEXISTSChecks to see if a file exists
 FLOCATE.....Locates a file according to MarketBrowser's path logic
 FOPEN.....Opens a file
 FPUTSPuts a string in an open file
 FREADAReads ASCII data from an open file
 FREADB.....Reads binary data from an open file

FSEEKAdvances file pointer to specified byte in open file
 FTELLReturns the byte of file pointer in open file
 FWRITEA.....Writes ASCII data to an open file
 FWRITEB.....Writes binary data to an open file
 MKDIRCreates a new directory
 MOVEFILEMoves a file to a new location or name without copying it
 RMDIRRemoves a directory if it is empty

Formula Manipulation

ADDFORMAdds to a window formula without adding the expression in string form to the formula
 ADDWFORM.....Adds to a window formula while adding in string form to the formula
 CALC.....Sets automatic recalculation On/Off
 CASTCOMPLEXExplicitly returns an expression as a complex number
 CASTINTEGERExplicitly returns an expression as an integer. See CASTCOMPLEX
 CASTREAL.....Explicitly returns an expression as a real number. See CASTCOMPLEX
 CASTSERIESExplicitly returns an expression as a series. See CASTCOMPLEX
 CASTSTRING.....Explicitly returns an expression as a string. See CASTCOMPLEX
 CLEARClears any window
 CLEARALL.....Clears all windows
 CLEARDATA.....Clears data from windows without removing formulas
 COPYWIN.....Copies a window's formula and attributes to a target window
 DEFMACRODefines a macro string or scalar constant
 DEPEND.....Adds or removes a dependency between series/hot variables
 EVALEvaluates a string expression
 EVALNOMACROS.....Evaluates a string expression but suppresses macro substitution
 EVALTOSTREvaluates a string and returns its value as a string
 GETLABEL.....Returns the label of a specified window
 GETVFORMReturns the formula of a window or a variable
 GETVUNITSReturns the vertical units of a series
 GETWCOUNTReturns the number of a series in a window
 GETWFORMULAReturns the formula for a window in string form
 GETWNUM.....Returns the current window number
 GETXLABEL.....Returns the x-axis label of the window
 GETYLABEL.....Returns the y-axis label of the window. SEE GETXLABEL
 GOTOWINDOWMoves the cursor to a specified window
 IFEvaluates a conditional expression
 ISNAVALUE.....Creates a binary series based on NA values in a series
 LABEL.....Sets the label for a window
 MOVETOMoves the cursor to a specified window
 OFFReturns the integer 0
 ONReturns the integer 1
 NAVALUERepresents the null data value
 ONPLOT.....Evaluates formula at the time window is being redrawn
 PASSEvaluates a formula
 REFRESH.....Reevaluates a worksheet
 RMFORM.....Removes an expression from a window's formula
 RTDEPEND.....Causes a series to re-evaluate based on a real-time event or evaluation cycle

SERCOUNTCounts the number of series in a window or table
 SETVUNITS.....Sets the vertical units
 SETWFORMSets the formula for a window
 UPDATEUpdates each formula in worksheet window

Fourier Transform and Related Functions

AUTOCORPerforms a time domain auto-correlation
 CONVConvolve two series
 CONV2DConvolve two matrices
 CROSSCOR.....Performs a time domain cross-correlation
 DFTCalculates the Discrete Fourier Transform in real/imaginary form
 FFT.....Calculates the Fast Fourier Transform in Cartesian form
 FFTPCalculates the Fast Fourier Transform in magnitude/phase form
 HAMMINGMultiplies a series by a Hamming window
 HANNING.....Multiplies a series by a Hanning window
 IDFT.....Calculates the inverse discrete Fourier Transform
 IFFTCalculates the inverse Fourier Transform in Cartesian form
 IFFTP.....Calculates the inverse Fourier Transform in magnitude/phase form
 IMPULSEGenerates a discrete unit impulse series
 KAISERMultiplies a series by a Kaiser window
 PSDCalculates the power spectral density, Magnitude2
 SPECTRUMCalculates the magnitude of a normalized FFT

Generated Series

GEXPGenerates an exponential curve
 GHAMMING.....Generates Hamming window
 GHANNINGGenerates a Hanning window
 GKaiserGenerates a Kaiser window
 GLINEGenerates a line
 GLNGenerates a logarithmic (base e) series
 GLOGGenerates a logarithmic (base e) series. See GLN
 GLOG10Generates a logarithmic (base 10) series
 GNORMALGenerates a normal series
 GRANDOM.....Generates a random series
 GSERGenerates a real series
 GSQRT.....Generates a square root series
 GSQRWAVE.....Generates a square wave
 GTRIWAVEGenerates a triangle wave
 JNBessel function
 SEEDRANDSets the seed value for the random number generator
 YN.....Integer Bessel function

Logical Operators

AND&&.....Logical AND
 NOT!.....Logical NOT
 OR||Logical OR
 FLIPFLOPCombines two binary series into a “flipflop” output
 XORLogical XOR

Macro, Command File, and DLL Functions

CALL	Calls a command file n times
ALLMACROS	Lists all the macros, including those with underscores, in the worksheet
DEFMACRO	Defines a macro
DLBIND	Loads a shared library object file
DLRUN	Runs a function from a shared library object file
DLUNBIND	Unbinds all DLL functions from MarketBrowser
GETMACRO	Returns information on the make-up of a macro
ISDLFUNC	Determines whether or not a DLL function exists
LOAD	Loads a command file
MACREAD	Reads an external file of macro definitions
MACROS	Lists the current macros defined in a worksheet
MACWRITE	Writes the current macro list to an external file

Matrix Math

BALANCE	Balances a matrix
CROSSPROD	Calculates the matrix crossproduct of one or two conforming matrices
DET	Calculates a matrix determinant
DIAGONAL	Computes a matrix diagonal
EIGVAL	Calculates the Eigenvalues of a matrix
EIGVEC	Calculates the Eigenvectors of a matrix
HESS	Calculates the Hessenberg form of a matrix
INNERPROD	Calculates the matrix inner product
INVERSE	Calculates the inverse of a matrix
LLU	Computes a lower triangular matrix in permuted LU decomposition
LU	Calculates an LU decomposition matrix
MMULT	Multiplies two matrices
NBEIGVAL	Calculates the Eigenvalues without a balancing step
NBEIGVEC	Calculates the Eigenvectors without a balancing step
OUTERPROD	Calculates the outer product of two vectors
SCHUR	Generates the SCHUR form of a matrix
SVD	Calculates the singular value decomposition of a matrix
TRACE	Calculates the sum of the major diagonal of a matrix
ULU	Calculates an upper triangular matrix in LU decomposition
USCHUR	Calculates the SCHUR form of an input matrix

Menu Functions

ECHO	Prints text at the bottom of the screen
INPUT	Allows the user to input values to functions
MENUCLEAR	Clears menus from the screen
MENUFILE	Generates a pop-up menu at the worksheet level
MENUINCALLBACK	Determines if a function has been called as a result of a user interaction
MENULIST	Displays a specified menu from the screen
MENUPRINT	Reads a text menu and prints it to a file
MENUREPOP	Signals a request for a repaint of the current menu

MENURETURN.....Stops processing steps inside user-defined panels
 MESSAGE.....Displays a message in the native GUI
 PICKFILE.....Displays a native GUI system dialog box for selecting a file
 PICKLIST.....Displays a box from which a string can be picked or specified
 VIEWFILE.....Displays the contents of an ASCII file
 WAITCURSOR.....Turns the hourglass cursor on or off

Operating System Interface

BEEP.....Toggles beeper
 GETENV.....Returns an environment variable string
 GETPATH.....Returns the current working directory path
 GROUPID.....Returns the system or session-specific group id. See HOSTID
 GROUPNAME.....Returns the groupname as a string. See HOSTNAME
 HOSTID.....Returns the system or session-specific host id
 HOSTNAME.....Returns the system or session-specific hostname
 PATHCHAR.....Returns the path character of the current operating system
 PID.....Returns the system or session-specific process id. See HOSTID
 PUTENV.....Sets an environment string
 RUN.....Runs an external program from a worksheet
 USERID.....Returns the system specific user id as an integer. See HOSTID
 USERNAME.....Returns the current username as a string. See HOSTNAME
 WAITFILE.....Waits for a file to exist and be stable in size

Output

INFOPRINT.....Prints a window and series information box
 INFOPRINTALL.....Prints all windows, each with its series information box
 PLOT.....Plots a window
 PLOTALL.....Plots all windows - one per page
 PLOTWS.....Plots all windows on one page
 PREVIEWALLWIN.....Displays previews of how all the windows will be printed
 PREVIEWINFO.....Displays how the information box will be printed
 PREVIEWWIN.....Displays a preview of how a window will be printed
 PREVIEWWSWIN.....Displays a worksheet will be printed on a single page
 PRINT.....Prints a window
 PRINTALL.....Prints all windows, one per page
 PRINTOPT.....Selects elements of a worksheet to display or hide
 PRINTWS.....Prints all windows on one page
 PRNSCREEN.....Prints a snapshot of the screen
 PS.....Creates a PostScript file of a current window
 PSALL.....Creates a PostScript file of all windows, one per page
 SCREENOPT.....Selects screen elements to hide or display
 PSWS.....Creates a PostScript file of all windows on one page

Peak Analysis

FMAX.....Places the cursor on the maximum
 FMIN.....Places the cursor on the minimum
 FPEAK.....Places the cursor on the first peak
 FPEAKN.....Places the cursor on the next peak
 FPEAKP.....Places the cursor on the previous peak

FVALLPlaces the cursor on the first valley
 FVALLNPlaces the cursor on the next valley
 FVALLPFinds the previous valley
 GETPEAKFinds peaks of a series
 GETPTDisplays the values of nth point
 GETRAWPTReturns the value of the nth point, including NAs
 GETVALLEYFinds valleys of a series
 LEVELCROSSDetermines where series cross
 SETPTModifies the value of a single point in a series

Real-Time Functions

ABSVOLMONMonitors a real-time absolute volume instrument
 BARMONRegisters a data item for updating in a series
 CAPTURECollects and displays data in tic-by-tic graphs
 CUMVOLMONMonitors a real-time cumulative volume instrument
 EQUIVOLMONMonitors price and volume data as an equivolume plot
 MONITORMonitors data as a line plot
 PFMONMonitors data as a point and figure chart
 PRIORReturns the previous value of a scalar numeric hot variable
 QMODEGets the latest value for a requested symbol
 QPAGEReturns a “page” for a given quote
 QUOTEReturns a formatted "one-shot" quotation string
 QSTRINGReturns a formatted “one-shot” arbitrary string for a symbol
 RTAMENDMakes a copy of a historical price series, amended and/or
 extended by a currently ticking real-time value
 RTDEBUGSets debugging trace level for data service transactions
 RTDEPENDCauses a series to re-evaluate based on a real-time event or
 evaluation cycle
 RTHISTORYRetrieves historical data from a live data service
 RTHISTPRetrieves historical data of various periodicities from a live data
 service
 RTINTERVALSets the timing interval for timer-bound function
 RTLINKReturns the currently selected “Hot Link”
 RTNOBSSets the default number of observations in real-time window
 RTOFFDeactivates real-time processing
 RTONActivates real-time processing
 RTPMATRIXTabulates current values for a portfolio from the values of its
 components
 RTSUPPRESSSuppresses reevaluation of a window during real-time updates
 RTPOSTSends an arbitrary string to the real-time data interface
 RTQUOTERegisters a data item for event-driven updating
 RTRANGESets the date & time ranges for intraday & historical queries
 RTREADAReads in a file periodically, replacing or updating data
 RTSENDSends an arbitrary string to the real-time data interface
 RTSTATUSGets, sets, or clears a string describing the status of the Real Time
 Data Interface (RTDI)
 SETHIGHWATERSets the high-water mark for a real-time dependent series

Relational Operators

<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Series and Scalar Math

+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
ABS	Computes the absolute value of a series
AVGS	Computes the average of n series
CEILING	Finds the smallest integer greater than or equal to the input value
CURRENT	References a series in the current window
E	Returns Euler's number e
EXP	Raises e to a specified power
FLOOR	Finds the greatest integer less than or equal to the input value
GAMM	Executes the Gamma function
GAMMLN	Computes the natural log of the Gamma function
I	Provides the value of the imaginary number (the square root of -1)
INDEX	Normalizes a series to percentage terms
INTERPOSE	Applies the REDUCE function associatively
LN	Calculates the natural logarithm
LOG	Calculates the natural logarithm
LOG10	Calculates the common base logarithm (base 10)
PARTPROD	Calculates the partial product of a series
PHI	Macro. "The golden mean" $(-1+(5))/2$
PI	Macro. Approximates the value of Π
REDUCE	Applies an operator to all values
ROOTS	Generates n-complex roots of unity (series)
ROUNDUP	Finds the smallest integer greater than or equal to a value
RTHROOT	Generates the complex root (scalar)
SINC	Calculates the sine function of an expression
SQRT	Calculates the square root
SUMS	Sums n series
TRUNC	Finds the greatest integer less than or equal to the input value

Statistics and Calculus

AMPDIST	Calculates the amplitude distribution
AREA	Calculates the area under a curve
COLLENGTH	Calculates the length of each column in a table
COLMAX	Calculates the maximum value of each column in a table
COLMEAN	Calculates the mean value for each column in a table
COLMEDIAN	Calculates the median value for each column in a table

COLMIN.....Calculates the minimum value for each column in a table
COLNUMOBSVReturns the count of non-NA values for each column in a table
COLREDUCEApplies the REDUCE function to each column in a table
COLSTDEV.....Calculates the standard deviation of each column in a table
DERIVCalculates the derivative
ERFCalculates the error function
ERFCCalculates the complementary error function
ERRORBAR.....Adds error bars to a graph
GAMM.....Calculates a generalization of factorial of real numbers
GAMMA.....A numeric constant
GAMMPCalculates the incomplete Gamma function
GAMMQ.....Calculates the complementary incomplete Gamma function
HISTOGRAM.....Calculates the frequency of values in a series
INTCalculates the integer value of an expression
INTEGCalculates the integral
LDERIV.....Calculates the derivative using a left-to-right slope algorithm
LENGTH.....Calculates the length
LINREG.....Determines the best linear fit
LINREG2.....Performs a linear regression of two or more series
MAX.....Finds the maximum of a series
MEAN.....Calculates the mean
MEDIANCalculates the median
MIN.....Finds the minimum of a series
MOVAVG.....Smoothes a series by averaging around each point
MOVMAX.....Performs n-point moving maximum calculations for a series
MOVMINPerforms n-point moving minimum calculations for a series
NUMOBSVCounts how many observations in a series are not NA values
PARTSUMCalculates the partial sum of a series
RDERIV.....Calculates the derivative using a right-to-left slope algorithm
REALCalculates the real component of a complex series
RMSCalculates the root mean square
ROWREDUCE.....Applies the REDUCE function to each row of a table
SERCOUNTCounts the number of series in window or table
STATSDisplays the statistics STDEV and ERROR
STDEV.....Calculates the standard deviation
TOTAL.....Sums a series

String Manipulation

ANYFORMAT.....Formats up to three arguments of mixed type
CHARSTRReturns the ASCII integer representation of a single character
EVALTOSTRReturns an evaluated expression as a string
ISNUMBER.....Tests whether a string is a number
GETSTRPrompts the user for a string using the native GUI
LISTEDITCreates a graphical list editor for the creation and maintenance of
user-defined lists of information
NFORMATFormats a list of numbers
NUMSTR.....Converts a string into a number
PFORMATReturns a string formatted as a decimal or fraction
SFORMATFormats a list of strings

SPRINTF Produces an output string in the format of the C printf function
 STRCHAR Returns the character represented by a given ASCII integer
 STRCAT Concatenates two or more strings
 STRCMP Compares two strings in lexicographic order
 STRESCAPE Converts special “escape” characters in a string
 STREXTRACT Extracts a part of a string
 STRFILE Returns a text file as a string with embedded newlines
 STRFIND Finds a position within a string
 STRFLDSORT Sorts a list of strings
 STRGET Returns the nth substring
 STRLEN Returns the length of a string
 STRLIST Returns a list of strings as one string
 STRNUM Converts a number into a string
 STRREVERSE Reverses the order of characters in a string
 TOKENIZE Returns a string which is the nth token in a given string
 TOKENWRAP Converts a string into a form suitable for parsing by TOKENIZE
 TOLOWER Converts a string to lowercase
 TOUPPER Converts a string to uppercase. See TOLOWER

Table Manipulation

COL Extracts a column of a table
 COLADD Add columns of data to an existing matrix
 COLDEL Deletes columns of data from an existing matrix
 GETSERIES Returns a single series from a table
 GETVNUMCOLS Counts the number of columns in a data variable referenced by name
 GRADE Ranks table values
 NUMCOLS Calculates the number of columns in a table
 NUMITEMS Calculates the number of data items in a window
 NUMROWS Calculates the number of rows in a table
 RAVEL Creates a table from a single vector
 RAWROW Extracts a row from a table, with consideration for NA values
 REGION Extracts a rectangular region of a table
 REORDER Reorders a series based on rank values
 ROW Extracts a row of a table
 RTAMEND Makes a copy of a historical price series, amended and/or extended by a currently ticking real-time value
 RTPMATRIX Tabulates current values for a portfolio from the real-time values of its components
 SETMATRIX Treats data in a window as a two-dimensional matrix
 SETVITEMTYPE Sets an item type explicitly
 SORT Sorts a table
 TABLE Displays point values of one series
 TABLES Displays point values of n series
 TRANSPOSE Swaps the rows and columns of a table
 TRYGETSERIES Like GETSERIES function, but does not produce an error
 UNRAVEL Creates a single vector from multiple columns

Trig Functions

ACOS Calculates the arc-cosine of any expression

ACOSH.....Calculates the hyperbolic arc-cosine of an expression in radians
ACOT.....Calculates the arc-cotangent of any expression
ACOTH.....Calculates the hyperbolic arc-cotangent of any expression
ACSC.....Calculates the arc-cosecant of any expression
ACSCH.....Calculates the hyperbolic arc-cosecant of any expression
ANGLE.....Calculates the phase component of a complex expression
ASEC.....Returns the arc-secant of any expression
ASECH.....Calculates the hyperbolic arc-secant of any expression
ASIN.....Calculates the arc-sine of any expression
ASINH.....Calculates the hyperbolic arc-sine of any expression
ATAN.....Calculates the arc-tangent of any expression
ATANH.....Calculates the hyperbolic arc-tangent of any expression
COS.....Calculates the cosine of any expression
COSH.....Calculates the hyperbolic cosine of any expression
COT.....Calculates the cotangent of any expression
COTH.....Returns the hyperbolic cotangent of any expression
CSC.....Returns the cosecant of any expression
CSCH.....Returns the hyperbolic cosecant of any expression
SEC.....Calculates the secant of any expression
SECH.....Calculates the hyperbolic secant of any expression
SIN.....Calculates the sine of any expression
SINH.....Calculates the hyperbolic sine of any expression
TAN.....Calculates the tangent of any expression
TANH.....Calculates the hyperbolic tangent of any expression

Trig Generators

GACOS.....Generates an arc-cosine curve
GACOSH.....Generates a hyperbolic arc-cosine curve
GACOT.....Generates an arc-cotangent curve
GACOTH.....Generates a hyperbolic arc-cotangent curve
GACSCT.....Generates an arc-cosecant curve
GACSCH.....Generates a hyperbolic arc-cosecant curve
GASEC.....Generates an arc-secant curve
GASECH.....Generates a hyperbolic arc-secant curve
GASIN.....Generates an arc-sine curve
GASINH.....Generates a hyperbolic arc-sine curve
GATAN.....Generates an arc-tangent curve
GATANH.....Generates a hyperbolic arc-tangent curve
GCOS.....Generates a cosine curve
GCOSH.....Generates a hyperbolic cosine curve
GCOT.....Generates a cotangent curve
GCOTH.....Generates a hyperbolic cotangent curve
GCSC.....Generates a cosecant curve
GCSCH.....Generates a hyperbolic cosecant curve
GSEC.....Generates a secant curve
GSECH.....Generates a hyperbolic secant curve
GSIN.....Generates a sine curve
GSINC.....Generates a SINC function ($\sin(x)/x$)
GSINH.....Generates a hyperbolic sine curve

GTANGenerates a tangent curve
GTANHGenerates a hyperbolic tangent curve

Worksheet Control

ADDWINDOWAdds any number of windows to the worksheet
COLLAYOUTArranges windows by column
DISPLAYDisplays specified windows
DISPLAYALL.....Displays all windows
GETCONFReturns a string describing an item in the expo.cnf file
GETWKSATTRIBUTE.....Returns the value of items set with the SETWKSATTRIBUTE
function
GETWORKSHEETReturns the name of the current worksheet
GETWSNUMROWS.....Returns the largest number of rows in the current worksheet
GETWSNUMWINDOWS.....Returns the number of (visible) windows in the current worksheet
HIDE.....Hides a window
LAYOUTControls the layout of windows in the worksheet
LOADWORKSHEET.....Loads a worksheet
NEATEN.....Fills gaps between manually resized windows
NUMWINDOWSReturns the number of windows in the worksheet
REDRAWRedraws all the windows in a worksheet
REDRAWALLRedraws the entire MarketBrowser screen
REMOVEWINDOWRemoves any number of windows from a worksheet
ROWLAYOUTArranges windows by rows
SAVEWORKSHEET.....Saves a worksheet under a specified name
SETCONF.....Sets an item from the expo.cnf file to a specified value
SETWKSATTRIBUTE.....Sets various worksheet locking attributes
SETPRECISION.....Sets the number of significant digits to be displayed
STOPREFRESHInterrupts the refresh process or inquires if the refresh is interrupted
UNWIND.....Reverts MarketBrowser to a known state, that is, idle in a window
WINSTATUSReturns the status of the current window

XPL Functions

ALLFUNCTIONSLists all the functions defined in the worksheet, including those
preceded by an underscore
CHANGE.....Returns the difference between the current value of a hot variable
and the previous value
DELALLFUNCTIONSDeletes all the XPL functions associated with the current worksheet
DELALLVARIABLESDeletes all XPL variables defined in a worksheet
DELFUNCTIONDeletes a named XPL function in the current worksheet
DELVARIABLEDeletes an XPL variable defined in a worksheet
FUNCTIONSLists the XPL functions defined in the current worksheet
GETDATAREF.....Returns the name of the first series variable in a window
GETDATAVARNAMEReturns the name of the variable currently being assigned
GETLOCALVARIABLE.....Gets the value of a local variable
GETVARIABLE.....Gets the current value of a global variable
INRTEVALTests to see if a function has been called normally or as a result of a
real-time event or top of the minute processing
ISVAR.....Determines whether a variable of a given type exists
LISTEDIT.....Creates a graphical list editor for the creation and maintenance of

user-defined lists of information
 ONEXIT.....Runs specified functions when an XPL function is exited
 PCTCHANGEReturns the percentage change between the current and previous
 value of a hot variable
 PRIOR.....Returns the previous value of a scalar numeric hot variable
 SETLOCALVARIABLE.....Creates a local XPL variable and sets a value to it
 SETVARIABLE.....Creates a global XPL variable and assigns a value to it
 SETVHUNITSets the horizontal units of a variable
 SETVUNITS.....Sets the vertical units of a variable
 VALUETYPEReturns the type of a given XPL variable
 VARSLists the current values of the XPL variables in the worksheet
 VARWRITEWrites a list of XPL variables out to an external text file
 WAITCURSORTurns the hourglass cursor on or off
 WINSTATUSReturns the status of the current window
 XPLLOAD.....Reads in and compiles an ASCII file of XPL functions
 XPLREADReads in, but does not compile, an ASCII file of XPL functions
 XPLWRITE.....Writes the current XPL function list to an external file

XY Functions

XVALS.....Returns the x values from a window
 XYGenerates an XY plot in a window
 XYINTERP.....Linearly interpolates an XY series
 YVALS.....Returns the y values from a window

3D Graphics

CONTOURDisplays a matrix as a contour plot
 DENSITYDisplays a matrix as a density plot
 MOUSEROTATEActivates mouse-driven rotation of a PLOT3D graph
 PLOT3D.....Creates a 3D plot
 ROTATE3D.....Rotates a PLOT3D graph
 SETPALETTEDefines an ordered list of shading colors
 SETSHADINGSelects the range of shading colors
 SETWFORMSets the formula for a window
 SHADEWITH.....Shades 3-D objects with another object
 WATERFALLDisplays a table as a 3-D waterfall plot

MarketBrowser Functions

+ - * / ^ (ARITHMETIC OPERATORS)

PURPOSE: Add/subtract/multiply/divide/exponentiate two expressions.

FORMAT: <expr1> + <expr2>

<expr1> Any expression evaluating to an integer, real number, or series.

<expr2> Any expression evaluating to an integer, real number, or series.

RETURNS: If one or both of the expressions is a series, then a series results. The following is a list of type conversion rules:

integer + integer => integer

integer + real => real

integer + series => series

real + series => real series

EXAMPLES: 128 + 13.29

returns the integer 141.29

4.3 + (W1)

adds 4.3 to each element of the series from W1 and results in new series.

REMARKS: A division by zero produces a default value of 0, but your machine will beep a warning.
Additionally, the "+" operator can be used to concatenate strings rather than using STRCAT; for example:
_pdt_shortstr = _pdt_shortstr+STRESCAPE("\n\n")+ 'Would you like to see more information?';

SEE ALSO: CONFORM

< <= > >= == != (RELATIONAL OPERATORS)

PURPOSE: Establishes logical relationships between expressions.

FORMAT: <expr1> > <expr2>

<expr1> Any expression evaluating to an integer, real number, or series.

<expr2> Any expression evaluating to an integer, real number, or series.

RETURNS: Scalar or series with values of 1 or 0 (true or false). If one or both of the expressions is a series, then a series results.

EXAMPLES: 128 > 13.29
displays the scalar result 1.
W1 > 100.0
creates a series with the value 1 wherever the corresponding point in W1 exceeds 100, and the value 0 where is does not.

SEE ALSO: CONFORM

&& || ! AND OR NOT XOR (LOGICAL OPERATORS)

PURPOSE: Serve as logical operators.

FORMAT: <expr1> && <expr2>

FORMAT: AND(<expr1>,<expr2>)

<expr1> Any expression evaluating to an integer, real number, or series.

<expr2> Any expression evaluating to an integer, real number, or series.

RETURNS: Scalar or series with values of 1 or 0 (true or false). In the case of && returns a 1 (true) wherever both sides of the equation are non-zero, zero elsewhere. For ||, returns a 1 wherever either side of the equation is non-zero, returns zeros elsewhere. If one or both of the expressions is a series, then a series results.

SEE ALSO: CONFORM
FLIPFLOP

;(SEMI-COLON)

PURPOSE: Statement separator.

FORMAT: **statement1; statement2; statement3...**

RETURNS: Nothing.

EXAMPLE: If the formula in window 3 contains:

W1; OVERPLOT(W2)

then window 3 is filled with a copy of window 1 and an overplot of window 2, and any changes in these windows will cause a re-evaluation of window 3.

SEE ALSO: | (Vertical bar)

| (VERTICAL BAR, or 'PIPE')

- PURPOSE:** Statement separator that returns the expression of higher precedence (i.e., expressions evaluating to a series over those evaluating to scalars).
- FORMAT:** **statement1 | statement2 | statement3...**
- RETURNS:** Nothing.
- EXAMPLE:** W1; OVERPLOT(SETDELTA(W2,1) | W2)
The pipe evaluates the expressions both to the left and right of it, but returns only the value with the higher precedence, in this case, 'W2', which evaluates to a series.
- REMARKS:** Ordinarily, the semi-colon is the preferred statement separator. The pipe can be useful to resolve ambiguities, but is mainly preserved to maintain backwards compatibility. Note that the above example can also be written as: W1; OVERPLOT(W2); SETDELTA(W2,1) This form is preferable as it uses the semi-colon over the pipe.
- SEE ALSO:** ; (semi-colon)

ABS(expr)

- PURPOSE:** Produces the absolute value of any expression.
- expr** Any expression evaluating to a series, table, integer or real number.
- RETURNS:** A series, table or number.
- EXAMPLE:** ABS(MEAN(W7))
produces a scalar which is the absolute value of the mean of the series in window 7.

ABSVOLMON(symbol)

- PURPOSE:** Monitors a real-time instrument as absolute volume.
- symbol** String. A symbol and field name appropriate to your data feed, in quotes.
- RETURNS:** A series.
- EXAMPLE:** ABSVOLMON("IBM.VOL")
returns a series updating at even, user-defined intervals of the current volume of IBM.
- SEE ALSO:** CUMVOLMON
EQUIVOLMON
MONITOR

ADDBDAY(juldate, integer)

PURPOSE:	Adds n valid business days to the base Julian date.
juldate	A Julian date.
integer	The amount of business days you wish to add. Integer can be either positive or negative.
RETURNS:	The result, expressed as a Julian date.
EXAMPLE:	Given the date, Friday, October 5, 2007, the function: ADDBDAY(JULSTR("10/05/07"),2) returns 24754, the Julian date representation of Tuesday, October 9, 2007.
REMARKS:	If integer is negative, the dates are subtracted.
SEE ALSO:	JULDAY JULSTR STRJUL

ADDFORM(window, expression)

PURPOSE:	Adds additional statement(s) to a window's formula, without re-evaluating the window or adding the expression in string form to the formula.
window	(Optional.) Defaults to the current window.
expression	Any valid MarketBrowser statement, in quotes.
RETURNS:	The result of expression.
EXAMPLE:	Given a window with the following formula: W1: GRANDOM(100,1) the following statement: W2: ADDFORM(W1,"WINCOLOR(GREEN)") executes the WINCOLOR function in W1 and adds the expression to the compiled form of the formula. It does not add the expression in string form to the formula. It does not re-execute the GRANDOM function.
REMARKS:	Use ADDFORM in cases where you would use ADDWFORM, but do not want to add the expression to the formula as a string, for example when you are up to the 255 character length limit for an MarketBrowser formula. There is a 255 character limit to the expression supplied to ADDFORM. That is to say, even though ADDFORM can be applied repeatedly, each individual call is limited.
SEE ALSO:	ADDWFORM GETWFORM SETWFORM

ADDWFORM(window, formula)

- PURPOSE:** Adds to the window formula without causing complete re-evaluation.
- window** (Optional.) Defaults to the current window.
- formula** Any valid MarketBrowser statement, in quotes.
- RETURNS:** Depends on the contents of the formula.
- EXAMPLE:** If the contents of the current window (W3) is "DERIV(CURR)", and you type:
ADDWFORM(W3,"OVERPLOT(W2)")
MarketBrowser adds the overplot and sets the window's formula to "DERIV(CURR);OVERPLOT(W2)" without re-evaluating "DERIV(CURR)".
- REMARKS:** This function is useful when simply editing the formula line would cause an unwanted or lengthy recalculation. Shorthand ADDWF can also be used.
- SEE ALSO:** GETWFORMULA GETVFORM
SETWFORM

ADDWINDOW(n, r, c)

- PURPOSE:** Adds the indicated number of windows to the worksheet.
- n** An integer representing the number of windows to be added to the worksheet.
- r** (Optional.) An integer representing the number of windows to display per row after adding windows. The maximum allowed per row is 10. If a row value is specified, a column value must be specified also.
- c** (Optional.) An integer representing the number of windows to display per column after adding windows. The maximum allowed per column is 10. If a column value is specified, a row value must be specified also.
- REMARKS:** MarketBrowser will add the new windows after the active window.
- SEE ALSO:** LAYOUT
REMOVEWINDOW

ALLFUNCTIONS

- PURPOSE:** Lists the current values of all the XPL functions defined in the worksheet. This includes functions that start with an _ (underscore) character, unlike the FUNCTION command, which does not display any names starting with an _ (underscore).
- RETURNS:** Nothing; screen display only.
- SEE ALSO:** ALLMACROS
FUNCTIONS
MACROS

ALLMACROS

- PURPOSE:** Lists the current values of all the macros defined in the worksheet. This includes macros that start with an _ (underscore) character.
- RETURNS:** Nothing; screen display only.
- SEE ALSO:** MACROS
FUNCTIONS
ALLFUNCTIONS

AMPDIST(series, delta-y)

- PURPOSE:** Finds the amplitude distribution of a series.
- series** A series or table.
- delta-y** The y increment. The smaller the delta-y value, the greater the number of amplitude ranges that will be defined and hence, the greater the number of points in the resulting series.
- RETURNS:** A series or table.
- EXAMPLE:** If the series in window 3 contains the points (1.0, 1.5, 2.0, 2.5, 3.0) and you type:
AMPDIST(W3, 0.7)
The resulting series contains the points (2.0, 1.0, 2.0), with:
- 2 point values between 1.0 and 1.7
 - 1 point value between 1.7 and 2.4
 - 2 point values between 2.4 and 3.1

ANYFORMAT(string, val1, val2, val3)

PURPOSE:	Formats up to three arguments of mixed type.
string	A format control string, in quotes.
valn	Integer, real, or string, matching the control string.
RETURNS:	A string.
EXAMPLE:	ANYFORMAT("Max:%f(%s)", max, getdate) produces a string like Max: 32.7 (02/15/91).
REMARKS:	See any standard C/C++ language reference for further information.
SEE ALSO:	NFORMAT SFORMAT

AREA(series, start, length)

PURPOSE:	Calculates the area of any part of a series, using Simpson's Rule.
series	A series or table from any window or a generated series. Defaults to the current window.
start	(Optional.) Point defined as the start of the series section to be used. The default value is the first point.
length	(Optional.) Length of the series portion to be used; only valid when start has been specified. The default length is to the end of the series.
RETURNS:	A number.
REMARKS:	<ol style="list-style-type: none">1. MarketBrowser will calculate AREA correctly even if the defined start or length require points beyond the end of the series.2. The area below the origin on the y-axis is a negative area. If you want to include area below y=0 as positive area, take the absolute value of the series first, e.g. AREA(ABS(W1)).
SEE ALSO:	INTEG DERIV

AUTOCOR(series)

PURPOSE:	(A Macro.) Performs a time domain auto-correlation of a series.
series	A series or table.
RETURNS:	A series or table.
EXPA- SION:	CONV(Series, REVERSE(Series))/(2*SERSIZE(Series)),
EXAMPLES:	W1: GSIN(128, 1/128, 4.0) W2: AUTOCOR(W1) calculates the auto-correlation of a sine wave. W1: GRAND(128, 1/128) W2: AUTOCOR(W1) finds the auto-correlation of a random series.
REMARKS:	The AUTOCOR function is often used to indicate how "similar" a waveform is to itself. The auto-correlation of the above sine wave shows several distinct peaks, indicating that the series at time t is similar to the series at t+T. The auto-correlation of the random series shows only one distinct peak, indicating that the series is correlated at time = 0 (as are all series) and dissimilar elsewhere.
SEE ALSO:	CONV CONV2 CROSSCOR DFFT

AVGS(series1, ..., seriesn)

PURPOSE:	Creates a new series that is the arithmetic mean of any number of input series.
series1, ..., seriesn	One or more series or tables.
RETURNS:	A series or table.
EXAMPLES:	AVGS(W1, W2, W6, W7, W9) creates a new series by averaging the series in the listed windows (i.e. $(W1 + W2 + W6 + W7 + W9) / 5.0$). AVGS(W3..W8) averages windows 3 through 8 (i.e. $(W1 + W2 + W3 + W4 + W5 + W6 + W7 + W8) / 8.0$).
REMARKS:	If the input series are of different lengths, all series are padded with point values of 0.0 to the length of the longest series.

SEE ALSO: MEAN SUMS

AUTOFREEZE

PURPOSE: Requests that the current window be frozen automatically (view not affected by real-time updates) when it is magnified or scrolled by the user.

BALANCE(matrix)

PURPOSE: Verifies that the balancing step keeps the essential properties of a matrix.

matrix A real or complex square matrix.

RETURNS: A matrix.

EXAMPLE: $x = \begin{matrix} 1 & 8 & 3 \\ 3 & 5 & 2 \\ 1 & 3 & 4 \end{matrix}$

$BALANCE(x) = \begin{matrix} 1.0 & 4.0 & 1.5 \\ 6.0 & 5.0 & 2.0 \\ 2.0 & 3.0 & 4.0 \end{matrix}$

$x = \begin{matrix} 0+8i & 0 & 1+i \\ 0 & 1001 & 0+1.5i \\ 11.25 & 0+2i & 200 \end{matrix}$

REMARKS: EIGVAL and EIGVEC first perform a balancing step in which the rows and columns are transformed to have root mean squares as close as possible while leaving the Eigenvalues and Eigenvectors unchanged. In most cases, this improves the accuracy of EIGVAL and EIGVEC, but in some cases it does not. BALANCE can be used to check that relatively small matrix elements have not become unduly magnified by the balancing step. If they have, then NBEIGVAL and NBEIGVEC are likely to yield better results.

SEE ALSO: EIGVAL
EIGVEC
NBEIGVAL
NBEIGVEC

BANDPASS(order, rate, pb1, pb2, ripple, atten, sb1, sb2)

PURPOSE:	Designs an FIR linear phase bandpass filter.
order	(Optional.) The filter length. If specified, the order must be an integer. If not specified, MarketBrowser will automatically estimate the required filter order.
rate	A real number that specifies the sampling rate of the filter in Hertz.
pb1	A real number that specifies the first passband edge of the filter in Hertz.
pb2	A real number that specifies the second passband edge of the filter in Hertz.
ripple	(Optional.) A real number for the passband ripple in dB. The default value is 3 dB.
atten	(Optional.) A real number for the stopband attenuation in dB. The default value is 40 dB.
sb1	(Optional.) A real number that specifies the first stopband edge of the filter in Hertz. Defaults to: $pb1 - 0.05 * rate$.
sb2	(Optional.) A real number that specifies the last stopband edge of the filter in Hertz. Defaults to: $pb2 + 0.05 * rate$.

RETURNS: The time domain impulse response of the filter.

EXAMPLES: BANDPASS(1000.0, 200.0, 300.0)

creates a bandpass filter with a sampling rate of 1000 Hz, and the passband extends from 200 Hz to 300 Hz. The first stopband defaults to 150 Hz and the last stopband defaults to 350 Hz. The resulting filter is 24 points long, with a passband ripple of 3.0 dB and a stopband attenuation of 48 dB.

BANDPASS(74, 1000.0, 200.0, 300.0, 2.0, 50.0, 180.0, 320.0)

creates the same filter as above except the filter order is set to 74 points. The desired passband ripple is set to 2.0 dB and the desired stopband attenuation is 50.0 dB. The first stopband edge is 180.0 Hz and the last stopband edge is set to 320.0 Hz. The resulting passband ripple is 2.4 dB and the stopband attenuation increases to 59 dB. Note that this filter did not meet the desired passband ripple specification. To meet the specification, the filter order must be increased.

REMARKS: The band edges must lie between 0.0 and $rate/2$ Hz. Overlapping edges are not permitted. The resulting characteristics of the filter are written to an ASCII file names BPASSn.FIR, where n is the nth filter designed. This file can be displayed by using the BPASS macro. For example, to display the filter characteristic file named BPASS4.FIR, try BPASS(4).

Use the FIRMAG function to display the frequency response of the filter.

BANDSTOP(order, rate, sb1, sb2, ripple, attn, pb1, pb2)

PURPOSE:	Designs an FIR linear phase bandstop filter.
order	(Optional). The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.
rate	A real number that specifies the sampling rate of the filter in Hertz.
sb1	A real number that specifies the first stopband edge of the filter in Hertz.
sb2	A real number that specifies the second stopband edge of the filter in Hertz.
ripple	(Optional.) A real number for the passband ripple in dB. The default value is 3 dB.
attn	(Optional.) A real number for the stopband attenuation in dB. The default value is 40 dB.
pb1	(Optional.) A real number that specifies the first passband edge of the filter in Hertz. Defaults to: $sb1 - 0.05 * rate$.
pb2	(Optional.) A real number that specifies the last passband edge of the filter in Hertz. Defaults to: $sb2 + 0.05 * rate$.

RETURNS: The time domain impulse response of the filter.

EXAMPLES: BANDSTOP(1000.0, 200.0, 300.0)

creates a bandstop filter with a sampling rate of 1000 Hz, and the stopband extends from 200 Hz to 300 Hz. The first passband defaults to 150 Hz and the last stopband defaults to 350 Hz. The resulting filter is 25 points long, with a passband ripple of 1.87 dB and a stopband attenuation of 52 dB.

BANDSTOP(75, 1000.0, 200.0, 300.0, 2.0, 50.0, 180.0, 320.0)

creates the same filter as above except the filter order is set to 75 points. The desired passband ripple is set to 2.0 dB and the desired stopband attenuation is 50.0 dB. The first stopband edge is 180.0 Hz and the last stopband edge is set to 320.9 Hz. The resulting passband ripple is 2.0 dB and the stopband attenuation increases to 61.5 dB.

REMARKS: The band edges must lie between 0.0 and $rate/2$ Hz. Overlapping edges are not permitted. The resulting characteristics of the filter are written to an ASCII file named BSTOPn.FIR, where n is the nth filter designed. This file can be displayed by using the BSTOP macro. For example, to display the filter characteristic file named BSTOP4.FIR, try:

```
BSTOP(4)
```

Use the FIRMAG function to display the frequency response of the filter.

BARCONVERT(data, deltax, max_gap, vol_bars)

PURPOSE:	Converts the periodicity of trading bars/candlesticks (e.g. converts five minute bars to fifteen minute bars).
data	Source data; may be single regular series, trading bars/candlesticks, or XY ("tic by tic").
deltax	Delta X for output series (in seconds for time of day input data, days for daily data, etc.)
max_gap	Integer. When converting tick by tick data, ticks that are farther apart than max_gap will not generate bars in between. Used for "closing the gap" in overnight data, for example. Use -1 as a placeholder to indicate that gaps should not be closed.
vol_bars	(Optional.) Integer. What kind of bars to make. Options are: <ul style="list-style-type: none">• 1 - Makes volume bars (sums the ticks in each bar).• 0 - Makes CHLO bars (default).
RETURNS:	Trading bars/candlesticks.
EXAMPLE:	BARCONVERT(W3, 300) creates 5 minute trading bars/candlesticks out of the data in window 3.
REMARKS:	If the periodicity of the input data is not "time of day", spacing represents the spacing of the output series. Data may be a single series, a trading bar/candlestick, or XY ("tic by tic"). If the output periodicity is higher than the source data, BARCONVERT will fill in by carrying over current values, as appropriate.
SEE ALSO:	SETHUNITS PERCONVERT

BARMON(symbol, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, gap_2_end, interval, paint_tick, update, add_nas, inside, na_interp)

PURPOSE:	Registers an instrument for updating and maintains it as a series of close-high-low-open values. Data is displayed as trading bars or Japanese candlesticks. BARMON can collect data for a user-defined trading range, fine-tune the update frequency of the bars, and the automatically extract and clean historical data via MarketBrowser's API.
symbol	Valid market symbol and its field.
start_date	Quoted string of the form "mm/dd/yy", which represents the date from which to start extracting data from symbol. Use the "" character (a pair of empty quotes) to leave the default, which is the start date of symbol.
start_time	Quoted string of the form "hh:mm:ss", representing the time to start extracting data from symbol. To leave the default (the start time of symbol), use "" as a placeholder.
end_date	Quoted string of the form "mm/dd/yy" representing the date on which to stop extracting data from symbol. Use "" to leave the default, which is the end date of symbol.
end_time	Quoted string of the form "hh:mm:ss", representing the time to stop extracting data from symbol. To leave the default (the end time of symbol), use "" as a placeholder.
gap_1_start, gap_2_start	Quoted string of the form "hh:mm:ss", representing the beginning of a gap in the extraction of data from symbol. To leave the default (00:00:00), use "" as a placeholder.
gap_1_end, gap_2_end	Quoted string of the form "hh:mm:ss", representing the end of a gap in the extraction of data from symbol. To leave the default (23:59:59), use "" as a placeholder.
interval	(Optional.) Integer. Multiple of the underlying real-time interval (default 60 seconds). Defaults to 1.
paint_tick	(Optional.) Integer: 0 = OFF, 1 = ON. Paints a continuously updating bar at the end of the chart. The bar updates on every tick of the instrument; it does not cause any dependent studies to update. Defaults to 0, or OFF.
update	Integer. Determines how frequently the series, and any series dependent on it, updates. Options are: 0 update every interval 1 update every interval and real-time interval. Default. 2 every tick and real-time interval

- add_nas** (Optional). Integer. Type of NA processing. Options are:
- 0 Do not fill gaps with NAs (default).
 - 1 Fill all gaps with NAs
 - 2 Fill only valid gaps on business days inside of trading hours (represented by **gap_1_start/end** and **gap_2_start/end**) with NAs.
- inside** (Optional.) Integer. How to process gaps:
- 1 Keep data INSIDE (within) of **gap_1_start** or **gap_1_end** and **gap_2_start** or **gap_2_end** (default).
 - 0 Keep data OUTSIDE of **gap_1_start/gap_1_end** and **gap_2_start gap_2_end**.
- na_interp** (Optional.) Integer. Type of interpolation. Options are:
- 1 Perform linear interpolation through valid gaps
 - 0 Leave gaps (default).

RETURNS: A series updating in real-time.

EXAMPLE: `BARMON("IBM.LAST", "", "", "", "", "09:00:00", "17:00:00", "12:00:00", "13:00:00", 5, 1, 1, 1, 0, 0)`

Monitors IBM.LAST in 5 minute bars from 9 AM to 5 PM, with a gap between 12 PM and 1 PM. It paints a continuously updating bar at the end of the graph, and replaces any gaps with NAs. It does not interpolate gaps in the data.

REMARKS: BARMON collects data as trading bars or candlesticks depending on the value of the configuration variable `DEFAULT_BAR_STYLE`, where the option 4 represents trading bars, and 0-3 represent candlesticks.

BARMON will enable data extraction between specified start and end dates, and through gaps, only if the configuration variable `EXTRACT_RT_HISTORY` is set to 1, or TRUE.

NA filling and interpolation is only performed upon historical data. Use care with update option 2 when monitoring fast-ticking symbols.

SEE ALSO: CAPTURE
MONITOR
DTEXTTRACT

BARS

PURPOSE: Displays the data points of a series filled as bars rather than connecting the points with a continuous curve.

RETURNS: Nothing

SEE ALSO: LINES
PCTSTACK
POINTS
STEPS
STICKS
TABLEVIEW
TICKFORM

BEEP(OnOff)

PURPOSE: Turns the automatic error beeper on or off.

OnOff An integer. On = 1, Off = 0.

RETURNS: Nothing

BUTTERWORTH(type, order, rate, pb1, pb2, ripple, atten, sb1, sb2)

PURPOSE:	Designs an IIR digital Butterworth filter.
type	Integer filter type. 1 = Lowpass, 2 = Highpass, 3 = Bandpass, 4 = Bandstop
order	(Optional.) The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.
rate	A real number that specifies the sampling rate of the filter in Hertz.
pb1	A real number that specifies the first passband edge frequency of the filter in Hertz.
pb2	A real number that specifies the second passband edge frequency of the filter in Hertz.
ripple	(Optional.) A real number for the passband ripple in dB. The default value is 3 dB.
atten	(Optional.) A real number for the stopband attenuation in dB. The default value is 40 dB.
sb1	(Optional.) A real number that specifies the first stopband edge frequency of the filter in Hertz. Default values are (pb1 - rate * 0.05) for bandpass, (pb1 + rate * 0.05) for bandstop.
sb2	(Optional.) A real number that specifies the last stopband edge frequency of the filter in Hertz. Default values are (pb2 + rate * 0.05) for bandpass, (pb2 - rate * 0.05) for bandstop.
RETURNS:	The filter coefficients in cascade form.
EXAMPLES:	<p>BUTTERWORTH(1, 1000.0, 100.0) creates a Butterworth lowpass filter with a sampling rate of 1000 Hz, and a cutoff frequency of 100 Hz. The stopband edge frequency defaults to 150 Hz and the passband ripple defaults to 3 dB.</p> <p>BUTTERWORTH(1,1000.0, 100.0, 3.0, 50.0, 130.0) creates a similar filter to above except the stopband attenuation is set to 50 dB and the stopband edge is lowered to 130 Hz.</p> <p>BUTTERWORTH(3, 18, 1000.0, 200.0, 300.0) creates a Butterworth bandpass filter with a sampling rate of 1000 Hz, and the passband extends from 200 Hz to 300 Hz. The first stopband defaults to 150 Hz and the last stopband defaults to 350 Hz.</p> <p>BUTTERWORTH(3, 24, 1000.0, 200.0, 300.0, 2.0, 50.0, 180.0, 320.0) creates a similar filter to above except the order is set to 24 (121 coefficients), the desired passband ripple is set to 2.0 dB and the desired stopband attenuation is set to 50 dB. The first stopband edge is 180 Hz and the last stopband edge is set to 320 Hz.</p>
REMARKS:	The band edges must lie between 0.0 and rate/2 Hz. The cutoff frequency must be less than the stopband edge frequency.

BYTESWAP(series, datatype)

- PURPOSE:** Reverses the bytes in a series.
- series** A series or table.
- datatype** SBYTE, UBYTE, BYTE, SINT, UINT, LONG, FLOAT, or DOUBLE
- RETURNS:** A series or table.
- EXAMPLE:** BYTESWAP(W1, SINT)
converts W1 into signed 2 byte integers and then reverses the bytes. BYTESWAP can be very useful when reading foreign data files via READB.
- REMARKS:** The datatypes listed above are macros resolving to integers.

CALC(OnOff)

- PURPOSE:** Turns the automatic worksheet recalculation mode on or off.
- OnOff** An integer. 0 = OFF (manual recalculation); 1= ON (automatic mode).
- RETURNS:** Nothing.
- EXAMPLES:** CALC(1)
is the default mode and specifies automatic window recalculation.
- CALC(0)
specifies manual recalculation. In this mode you can enter window formulas without calculating series immediately. Once you type CALC(0), enter new formulas in the desired windows and then type CALC(1) to recalculate.
- SEE ALSO:** UPDATE
RTDEPEND

CALL(filename, n)

PURPOSE:	Calls a command file n times.
filename	The name of command file, in quotes.
n	(Optional.) An integer specifying the number of times to call the command file. Defaults to 1.
RETURNS:	Nothing.
EXAMPLE:	CALL("MYFILE.SCR", 2) executes MYFILE.SCR 2 times from within the current command file.
REMARKS:	CALL is useful for creating loops in a command file.
SEE ALSO:	LOAD

CAPTURE(symbol, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, gap_2_end, add_nas, inside, na_interp)

PURPOSE:	Registers an instrument for updating and collects every incoming tick into a real-time series. CAPTURE also extracts data for a user-specified trading period.
symbol	Valid market symbol and its field to be monitored.
start_date	Quoted string of the form "mm/dd/yy", which represents the date from which to start extracting data from symbol . Use the "" character (an empty pair of quotes) to leave the default, which is the start date of symbol .
start_time	Quoted string of the form "hh:mm:ss", representing the time to start extracting data from symbol . To leave the default (the start time of symbol), use "" as a placeholder.
end_date	Quoted string of the form "mm/dd/yy" representing the date on which to stop extracting data from symbol . Use "" to leave the default, which is the end date of symbol .
end_time	Quoted string of the form "hh:mm:ss", representing the time to stop extracting data from symbol . To leave the default (the end time of symbol), use "" as a placeholder.
gap_1_start, gap_2_start	Quoted string of the form "hh:mm:ss", representing the beginning of a gap in the extraction of data from symbol . To leave the default (00:00:00), use "" as a placeholder.

gap_1_end, Quoted string of the form "hh:mm:ss", representing the end of a gap in the extraction
gap_2_end of data from **symbol**. To leave the default (23:59:59), use "" as a placeholder.

add_nas (Optional.) Integer. Leave as default. No effect.
0 Do not fill gaps with NAs (default).

inside (Optional.) Integer. How to process gaps:
1 Keep data INSIDE (within) **gap_1_start** and **gap_1_end**, and **gap_2_start** and **gap_2_end** (default).
0 Keep data OUTSIDE of **gap_1_start** and **gap_1_end**, and **gap_2_start** and **gap_2_end**.

na_interp (Optional.) Integer. Leave as default. No effect.
0 Leave gaps (default).

RETURNS: A series updating in real-time.

EXAMPLE: CAPTURE("IBM.LAST","", "", "", "", "09:00:00", "12:30:00", "13:30:00",
"17:00:00",0,0,0)
captures all incoming tick data from 9 AM to 12:30 PM, and from 1:30 PM to 5 PM.

REMARKS: Extraction of historical data (and of real-time updates) is performed using the same mechanism as DTEXTTRACT. Data extraction is enabled only if the configuration variable EXTRACT_RT_HISTORY is set to 1, or TRUE. Capture cannot NA fill missing data, therefore add_nas and na_interp have no effect.

SEE ALSO: BARMON
MONITOR
DTEXTTRACT

CARTESIAN(expr)

- PURPOSE:** Converts an expression to real/imaginary form in Cartesian coordinates.
- expr** A series, table or number in integer, real/complex, or polar coordinate form.
- RETURNS:** A series, table or number.
- EXAMPLES:** CARTESIAN(GSIN(20, .05, 1.0))
creates a 1 Hz sine wave of 20 points spaced 0.05 seconds apart. The value of each point in the sine wave is a complex number in real/imaginary form.
- CARTESIAN(-1)
displays the complex scalar -1.0 + 0i.
- REMARKS:** Returns a complex value regardless of the input value. For series, CARTESIAN always returns a complex series.
- SEE ALSO:** POLAR
REAL
IMAGINARY
CONJUGATE
PHASE

CASCADE(series, iirseries)

- PURPOSE:** Filters a time domain input series with an IIR digital filter where the filter coefficients are represented in cascaded sections of second order stages.
- series** A series or table.
- iirseries** IIR filter coefficients in cascade form.
- RETURNS:** A series or table.
- EXAMPLE:** W1: BUTTERWORTH(1, 100.0, 10)
W2: GAIN(100, .01, 4.0) + GAIN(100, .01, 40)
W3: CASCADE(W2, W1)
removes the 40 Hz sine wave from the sum by filtering the series with a 10 Hz lowpass Butterworth IIR filter.

CASTCOMPLEX CASTINTEGER CASTREAL CASTSERIES CASTSTRING(expression)

PURPOSE: Explicitly returns a value of the said type.

expression Any valid MarketBrowser expression that returns a value.

RETURNS: A complex, integer, real, series, or string, as appropriate.

EXAMPLE: Given the following formula:
W1: GRANDOM(10,1,1,10)
where the series has a standard deviation 2.2594, the formula:
W2: GSER(CASTINTEGER(STDEV(W1)))
returns a series consisting of one value, 2.

SEE ALSO: EVAL
EVALTOSTR
PASS

CEILING(expr)

PURPOSE: Finds the smallest integer greater than or equal to the input value.

expr Any expression evaluating to a scalar, series, table, integer, or real or complex number.

RETURNS: A scalar, series, table or number.

EXAMPLES: CEILING(-2.4 + 7.2I)
returns -2.0 + 8.0i.
CEILING(W2)
creates a new series by applying CEILING to each series element of W2. The integer returned by CEILING is converted to a floating point value.

SEE ALSO: FLOOR

CHANGE(hotvar)

- PURPOSE:** Calculates the absolute difference between the current value of a scalar numeric hot variable and its previous value.
- hotvar** A hot variable reference.
- RETURNS:** The real difference between the hot variable's current value and its previous value.
- EXAMPLE:** Given the following hot variable:
mydata:= RTQUOTE("IBM.LAST")
CHANGE(mydata)
returns the difference between the two latest points.
- SEE ALSO:** PCTCHANGE
PRIOR

CHARSTR(string)

- PURPOSE:** Returns the ASCII integer representation of a single character.
- string** String. A single ASCII character.
- RETURNS:** An integer.
- SEE ALSO:** STRCHAR STRESCAPE
STRCAT STRNUM
NUMSTR

CHEBY1(type, order, rate, pb1, pb2, ripple, atten, sb1, sb2)

- PURPOSE:** Designs a digital IIR Chebyshev type I filter.
- type** An Integer. The filter type. 1 = Lowpass, 2 = Highpass, 3 = Bandpass, 4 = Bandstop.
- order** (Optional.) The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.
- rate** A real number that specifies the sampling rate of the filter in Hertz.
- pb1** A real number that specifies the first passband edge frequency of the filter in Hertz.
- pb2** A real number that specifies the second passband edge frequency of the filter in Hertz.
- ripple** A real number for the passband ripple in dB.

atten A real number for the stopband attenuation in dB.

sb1 (Optional). A real number that specifies the first stopband edge frequency of the filter in Hertz. Default values are (pb1 - rate * 0.05) for bandpass, (pb1 + rate * 0.05) for bandstop.

sb2 (Optional). A real number that specifies the last stopband edge frequency of the filter in Hertz. Default values are (pb2 + rate * 0.05) for bandpass, (pb2 - rate * 0.05) for bandstop.

RETURNS: The filter coefficients in cascade form.

EXAMPLES: CHEBY1(2, 1000.0, 100.0, 1.0, 40.0)

creates a Chebyshev I highpass filter with a sampling rate of 1000 Hz, and a cutoff frequency of 100 Hz. Its passband ripple is set to 1.0 dB. The stopband edge frequency defaults to 50 Hz. Its stopband attenuation is set to 40 dB.

CHEBY1(2, 1000.0, 200.0, 300.0, 2.0)

creates a similar filter to the above except the stopband attenuation is set to 50 dB and the stopband edge is increased to 70 Hz.

CHEBY1(3, 8, 1000.0, 200.0, 300.0, 2.0)

creates a Chebyshev I bandpass filter with a sampling rate of 1000 Hz, an order of 8 and the passband extends from 200 Hz to 300 Hz. Its passband ripple is set to 2 dB.

CHEBY1(3, 1000.0, 200.0, 300.0, 2.0, 60.0, 150.0, 350.0)

creates a Chebyshev I bandpass filter with a sampling rate of 1000 Hz, passband ripple of 2 dB, and the passband extends from 200 Hz to 300 Hz. The first stopband is set to 150 Hz and the last stopband is set to 350 Hz. The stopband attenuation is 60 dB.

CHEBY1(3, 1000.0, 200.0, 300.0, 2.0, 50.0, 180.0, 320.0)

creates a similar filter to the above except the filter's first stopband edge is set to 180 Hz, and the last stopband edge is set to 320 Hz. The desired stopband attenuation is set to 50 dB.

REMARKS: The band edges must lie between 0.0 and rate/2 Hz. The cutoff frequency must be less than the stopband edge frequency. A Chebyshev I filter has a non-zero ripple in the passband.

CHEBY2(type, order, rate, pb1, pb2, atten, ripple, sb1, sb2)

PURPOSE:	Designs a digital IIR Chebyshev type II filter.
type	Integer filter type. 1 = Lowpass, 2 = Highpass, 3 = Bandpass, 4 = Bandstop.
order	(Optional). The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.
rate	A real number that specifies the sampling rate of the filter in Hertz.
pb1	A real number that specifies the first passband edge frequency of the filter in Hertz.
pb2	A real number that specifies the second passband edge frequency of the filter in Hertz.
atten	A real number for the stopband attenuation in dB.
ripple	A real number for the passband ripple in dB. The default value is 3 dB.
sb1	(Optional). A real number that specifies the first stopband edge frequency of the filter in Hertz. Default values are (pb1 - rate * 0.05) for bandpass, (pb1 + rate * 0.05) for bandstop.
sb2	(Optional). A real number that specifies the last stopband edge frequency of the filter in Hertz. Default values are (pb2 + rate * 0.05) for bandpass, (pb - rate * 0.05) for bandstop.

RETURNS: The filter coefficients in cascade form.

EXAMPLES: CHEBY2(2, 4, 1000.0, 100.0, 40.0)

creates a Chebyshev II highpass filter with a sampling rate of 1000 Hz, order of 4 and a cutoff frequency of 100 Hz. The stopband attenuation is set to 40.0 dB.

CHEBY2(2, 1000.0, 100.0, 40.0)

creates a Chebyshev II highpass filter with a sampling rate of 1000 Hz, and a cutoff frequency of 200 Hz. Its stopband attenuation is set to 40.0 dB. The stopband edge frequency defaults to 50 Hz. Its stopband ripple defaults to 3.0 dB.

CHEBY2(2, 1000.0, 100.0, 40.0, 2.0, 70.0)

creates a similar filter to above except the stopband ripple is set to 2.0 dB and the stopband edge is increased to 70 Hz.

CHEBY2(3, 8, 1000.0, 200.0, 300.0, 40.0)

creates a Chebyshev II bandpass filter with a sampling rate of 1000 Hz, order of 8, and the passband extends from 200 Hz to 300 Hz. Its stopband attenuation is set to 40.0 dB.

CHEBY2(3, 1000.0, 200.0, 300.0, 40.0)

creates a Chebyshev II bandpass filter with a sampling rate of 1000 Hz, stopband attenuation of 30 dB, and the passband extends from 200 Hz to 300 Hz. The first stopband defaults to 150 Hz and the last stopband defaults to 350 Hz. The Stopband

ripple defaults to 3.0 dB.

CHEBY2(3, 1000.0, 200.0, 300.0, 40.0, 2.0, 180.0, 320.0)

creates a similar filter to above except the filter's first stopband edge is set to 180 Hz, and last stopband edge is set to 320 Hz. The desired stopband ripple is set to 2.0 dB.

REMARKS: The band edges must lie between 0.0 and rate/2 Hz. The cutoff frequency must be less than the stopband edge frequency.

A Chebyshev II filter has a non-zero ripple in the stopband.

CLEAR(window1,...,windown)

PURPOSE: Clears a series and its formula from any number of windows.

window1, ..., windown (Optional). Any window. Defaults to the current window.

RETURNS: Nothing.

EXAMPLES: CLEAR
clears the current window.
CLEAR(W3..W8)
clears windows 3 through 8.

REMARKS: A cleared window propagates throughout the entire worksheet. If W2 depends on W1 and W1 is cleared, MarketBrowser clears W2.

SEE ALSO: CLEARALL

CLEARALL

PURPOSE: Clears all series and formulas from every window in the worksheet. Be careful!

RETURNS: Nothing.

EXAMPLE: CLEARALL
Removes the series and formulas from every worksheet window.

REMARKS: Consider saving your worksheet before using CLEARALL.

SEE ALSO: CLEAR

CLEARDATA(window1,...,windown)

- PURPOSE:** Clears data from one or more windows without removing window formulas.
- window1, ..., windown** (Optional.) Window reference. Defaults to the current window.
- RETURNS:** Nothing.
- EXAMPLE:** CLEARDATA(W1, W5..W7)
clears the data in windows 1, 5, 6, and 7, but leaves the formulas intact.
- REMARKS:** CLEARDATA also stops the data updating process. CLEARDATA is useful when you want to save a worksheet without saving large amounts of accompanying data.
- SEE ALSO:** REFRESH
UPDATE

CLEARXLABEL CLEARYLABEL(window)

- PURPOSE:** Clears the x-axis or y-axis label, and resets the display to the horizontal (x-axis) or vertical (y-axis) units.
- window** (Optional.) Window reference. Defaults to the current window.
- EXAMPLE:** CLEARXLABEL clears the definition of the x-axis label in the current window, and displays the horizontal units.
- REMARKS:** Once the x-axis or y-axis label is cleared, the label assigned to the units will reappear as usual.
- SEE ALSO:** SETHUNITS
SETXLABEL
GETXLABEL

CLIP(series, maxthresh, minthresh)

PURPOSE:	Creates a copy of a series that has specified maximum and minimum y values.
series	A series or table.
maxthresh	(Optional). A real number, or expression resolving to a real number, setting the upper y limit. The default is the maximum of the series.
minthresh	(Optional). A real number, or expression resolving to a real number, setting the lower y limit. The default is the minimum of the series.
RETURNS:	A series or table.
EXAMPLES:	To select a horizontal slice from a series in window 3, type: CLIP(W3, 4.5, -3.0) This creates a new series, in the current window, with y values outside 4.5 and -3.0 set to their applicable thresholds. CLIP(W2, MAX(W2), -2.0) sets only a minthresh (of -2.0) because the maxthresh is the highest peak of the series in W2.
REMARKS:	If only one threshold argument is given, MarketBrowser assumes it is the maxthresh. If no threshold arguments are given, MarketBrowser will return an exact copy of the current series.
SEE ALSO:	MAX MIN

COL(table, column)

PURPOSE:	Extracts a column of data from a table.
table	A table.
column	An Integer. The number of the column to extract.
RETURNS:	A column of data as a series.
EXPAN- SION:	GETSER
EXAMPLE:	COL(W1, 6) extracts the sixth column of the table in window 1.
REMARKS:	The COL function may be used with any graph style, not only the table view, to extract a particular series of data from a multi-series set. To extract more than a single column at a time, use the REGION function.
SEE ALSO:	REGION ROW

SERCOUNT

COLADD(target, source, where)

PURPOSE:	Adds columns of data to an existing matrix. This is a faster alternative to RAVEL(), which copies the data and returns the result.
target	The matrix to manipulate, typically a variable
source	The source series or matrix to add
where	(Optional). The integer index indicating where to start adding. If present, must be ≥ 2 because we cannot replace the lead series in the target matrix with a new series. If not specified, source is added at the end of the target.
RETURNS:	An integer if a column is added, otherwise 0.
SEE ALSO:	COLDEL RAVEL

COLDEL(target, where, how many)

PURPOSE:	Deletes columns of data from an existing matrix.
target	The matrix to manipulate, typically a variable
where	(Optional). The integer index indicating where to start deleting. If present, must be ≥ 2 . If not specified, one column is deleted from the end of the target.
how many	(Optional). The integer index indicating the number of columns to delete.
RETURNS:	An integer if a column is deleted, otherwise 0.
SEE ALSO:	COLADD RAVEL

COLLAYOUT(int1,...,intn)

PURPOSE:	Sets how many columns the MarketBrowser screen is divided into, and in turn, how many windows the individual columns are divided into.
int	An integer between 1 and 10 specifying how many windows to be displayed in a column.
RETURNS:	A screen with a specified number of columns and windows per column.
EXAMPLE:	In a 9 window worksheet, COLLAYOUT(2,3,4) would give a screen with 3 even columns of 2, 3, and 4 windows, respectively.

REMARKS: COLLAYOUT can display no more than 10 windows per column.
 You will get an error message if the total number of windows specified as arguments exceeds the number of displayed windows in the worksheet.
 On the other hand, if you specify the layout for fewer windows than the displayed total, MarketBrowser will group the remaining windows into a single column. For example, in a 9 window worksheet, COLLAYOUT(2) will return a screen with 2 columns of 2 and 7 windows, respectively.

SEE ALSO: ROWLAYOUT
 NEATEN
 TILE

COLLENGTH(table)

PURPOSE: Applies the LENGTH function to each column in a table.

table A table.

RETURNS: A one row table with the same number of columns as the input table.

EXAMPLE: COLLENGTH(RAVEL(GSER(1, 2), GSER(3, 4, 5, 6)))
 produces a table with a single row 2, 4.

SEE ALSO: COLMAX
 COLMEAN
 COLMEDIAN
 COLMIN
 COLSTDEV
 LENGTH
 COLNUMOBSV

COLMAX(table)

PURPOSE: Applies the MAX function to each column in a table.

table A table.

RETURNS: A one row table with the same number of columns as the input table.

EXAMPLE: COLMAX(RAVEL(GSER(1, 3, 9), GSER(2, 1, 7)))
 produces a table with a single row 9, 7.

SEE ALSO: COLLENGTH COLMEAN
 COLMEDIAN COLMIN
 COLSTDEV MAX

COLMEAN(table)

- PURPOSE:** Applies the MEAN function to each column in a table.
- table** A table.
- RETURNS:** A one row table with the same number of columns as the input table.
- EXAMPLE:** COLMEAN(RAVEL(GSER(2, 4, 6), GSER(1, 3, 5)))
produces a table with a single row 4, 3.
- SEE ALSO:** COLLENGTH COLMAX
COLMEDIAN COLMIN
COLSTDEV

COLMEDIAN(table)

- PURPOSE:** Applies the MEDIAN function to each column of a table.
- table** A table.
- RETURNS:** A one row table with the same number of columns as the input table.
- EXAMPLE:** COLMEDIAN(RAVEL(GSER(1,2,3), GSER(2,3,4)))
produces a table with a single row 2, 3.
- SEE ALSO:** COLLENGTH COLMAX
COLMEAN COLMIN
COLSTDEV MEDIAN

COLMIN(table)

- PURPOSE:** Applies the MIN function to each column in a table.
- table** A table.
- RETURNS:** A one row table with the same number of columns as the input table.
- EXAMPLE:** COLMIN(RAVEL(GSER(1, 6, 9), GSER(9, 4, 7))
produces a table with the single row 1, 4.
- SEE ALSO:** COLLENGTH
COLMAX
COLMEAN
COLMEDIAN
COLSTDEVMIN

COLNUMOBSV(**table**)

PURPOSE: Applies the NUMOBSV function to each column in a table.

table A table.

RETURNS: A one row table with the same number of columns as the input table.

EXAMPLE: COLNUMOBSV(RAVEL(GSER(1, 2, NAVALUE, 4, 5, 6), 3))
produces a table with the single row 2, 3.

SEE ALSO: NUMOBSV
COLLENGTH

COLPOS(**window, item, cursor_num**)

PURPOSE: Returns the item or column number of the last position of the crosshair cursor in a window.

window (Optional). Window reference. Defaults to the current window

item (Optional). An integer specifying the item number (i.e. series) in the window. Defaults to 1.

cursor_num (Optional). An integer specifying the cursor number. 1 = First Cursor, 2 = Second Cursor. Defaults to 1.

RETURNS: The item or column number where the cursor was most recently placed. If the cursor was never activated in the given window, the column number returned is 0.

EXAMPLES: REGION(W1,1,LENGTH(W1),COLPOS(W1,1,1),NUMCOLS(W1)-
COLPOS(W1,1,1))
extracts a rectangular region from the table in window 1, starting from the item where the first cursor was last placed.

COLPOS(W1,1,2)
returns the item number of the second cursor for the first item in W1.

REMARKS: Changes in cursor position do not propagate through the worksheet. If you want to update a window dependent on a new cursor position, use the Line Editor ([F3]) to re-enter the line so the cursor position is reevaluated.

A series or an XY plot are considered an item.

MarketBrowser “remembers” cursors’ last position; that is, when you place a cursor on the series, MarketBrowser draws it at the most recent location (which may mean that the window is redrawn to display that x or y range). To disable this feature, set the configuration items CURSOR_MEMORY and CURSOR2_MEMORY to 0.

SEE ALSO: CURPOS NUMITEMS
CURSORON

COLREDUCE(table, op)

PURPOSE:	Applies the REDUCE function to each column of a table.	
table	A table.	
op	Quoted string containing the binary operator.	
RETURNS:	A table with one row and as many columns as the input table.	
EXAMPLE:	COLREDUCE(RAVEL(GSER(1,2,3), GSER(2,3,4)), "*") produces a table with the single row 6, 24.	
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR"	
SEE ALSO:	ROWREDUCE	REDUCE
	INTERPOSE	INNERPROD
	OUTERPROD	

COLSTDEV(table)

PURPOSE:	Applies the STDEV function to each column in a table.	
table	A table.	
RETURNS:	A one row table with the same number of columns as the input table.	
EXAMPLE:	COLSTDEV(RAVEL(GSER(1,2,3,4), 2) produces a table with a single row containing two values.	
SEE ALSO:	COLLENGTH	COLMAX
	COLMEAN	COLMEDIAN
	COLMIN	STDVEV

COMMENT(window, string)

PURPOSE:	Sets the comment for the first series in a window.	
window	(Optional). Defaults to the current window.	
string	Any text, in quotes.	
EXAMPLE:	COMMENT(W4, strcat("IBM as of ", getdate)) places "IBM as of 04/14/89" into the current comment field.	
SEE ALSO:	SETCOMMENT	GETCOMMENT
	GETSCOMMENT	GETHUNITS
	GETVUNITS	LABEL

COMPRESSH COMPRESSV(factor)

- PURPOSE:** Compresses a series in the current, activated window.
- factor** (Optional). A ratio of the current series dimension to the desired dimension. Default is 3/2, which makes the series appear 2/3 of its original size.
- RETURNS:** Nothing.
- REMARKS:** COMPRESSH compresses a series horizontally; COMPRESSV compresses a series vertically. To return the window to its original display, use:
- a) the reciprocal ratio of the argument you chose before,
 - b) EXPANDH or EXPANDV with the same argument, or
 - c) <CTRL>+<HOME>.
- Pressing <CTRL>+<←> or <CTRL>+<↓> when the current window is active runs the COMPRESSH/COMPRESSV function using the same default factor.
- SEE ALSO:** EXPANDH
EXPANDV

CONCAT(series1, ..., seriesn, inheritance)

- PURPOSE:** Concatenates series.
- series1, ..., seriesn** Any number of series or tables.
- inheritance** (*Optional*). Determines how attributes of the input series are inherited by the output. Possible values are:
- 1 attributes for each input series are added in successively (default),
 - 0 no inheritance,
 - 1 inherit attributes only from the first series,
 - 2 inherit only from the last series supplied.
- RETURNS:** A series whose length is the end-to-end sum of the input series.
- EXAMPLES:** CONCAT(W1,W2,W6)
creates a new series by appending the series in window 1, window 2 and window 6 end-to-end.
- CONCAT(W3..W8)
concatenates the series in windows 3 through 8.
- REMARKS:** CONCAT operates on any number of input series or tables.
- SEE ALSO:** EXTRACT MERGE
REPLICATE

CONFORM(type)

- PURPOSE:** Sets the type of date/time conformity performed by binary operators.
- type** The type of date/time conformity to perform; values are:
- 0 No automatic conformity.
 - 1 Union -- pad nonconforming observations with NAs
 - 2 Intersection -- trim all nonconforming observations.
- RETURNS:** An integer indicating the type of conformity currently in effect.
- EXAMPLE:** CONFORM(0)
turns off all conformity processing for binary operators.
- REMARKS:** CONFORM sets the default behavior of binary operations (such as + - * /) when they involve two series that do not conform exactly in their date/time ranges. By default, CONFORM creates an intersection of both series in time, with NAs where they do not have data in common.
- CONFORM() returns an integer indicating the type of conformity currently in effect.
- SEE ALSO:** ISNAVALUE ARITHMETIC OPERATORS
SETNAVALUE LOGICAL OPERATORS

CONJUGATE(expr)

- PURPOSE:** Calculates the complex conjugate of any series, table or number.
- expr** Any expression evaluating to a series, table, integer or a real/complex number.
- RETURNS:** A series, table.
- EXAMPLES:** CONJUGATE(3+4i)
results in 3-4i.
- CONJUGATE(POLAR(1 + i))
displays mag = 1.41421; angle = -0.7854, -45, which describes a complex number in polar coordinates where the magnitude is 1.41421 and the angle is -0.7854 radians, or -45 degrees.
- REMARKS:** The output is complex regardless of the input.
- SEE ALSO:** POLAR
CARTESIAN
REAL

CONTOUR(matrix, levels, InColor, labeling)

- PURPOSE:** Displays a matrix of data as a contour plot.
- matrix** A rectangular matrix of data to plot.
- levels** (Optional). An integer. The number of contour levels to draw (default 10).
- color** (Optional). An integer indicating whether color is on or off (default 0).
- labeling** (Optional). 4 or 8 real numbers representing end points of labeling axes (default 0).
- RETURNS:** A matrix, displayed as a contour plot by default.
- EXAMPLE:** CONTOUR(W1, 20, On, 20.0, 10.0, 20.0, 90.0)
shows the contents of W1 as a 20 level contour, with labels drawn wherever the contour lines cross an imaginary line with endpoints of (20.0, 10.0) and (20.0, 90.0), a vertical line near the left side of the plot. The background of the plot is filled according to the magnitude of the data, with colors as defined by the current shading scheme.
- REMARKS:** Up to two axes (a total of eight coordinates) can be specified for labeling.
- SEE ALSO:** SETSHADING SETPALETTE
SHADEWITH DENSITY

CONV(series1, series2, start, end)

- PURPOSE:** Convolves two series.
- series1, series2** Any two series.
- start** (Optional). The starting point. The default is the first point in each series.
- end** (Optional). The ending point. The default is the last point in each series.
- RETURNS:** A series or table.
- EXAMPLES:** CONV(W1, REVERSE(W1))
yields the auto-correlation of the series in W1.
CONV(W1, REVERSE(W1), 100, 300)
yields the same as the above example, except that only 200 values are calculated starting at the 100th point.
- REMARKS:** CONV does not use a Fourier Transform method to calculate convolution. By default, the resulting series contains SERSIZE(W1) + SERSIZE(W2) - 1 data point.
- SEE ALSO:** CONV2D CROSSCOR FFT AUTO

CONV2D(matrix1, matrix2, row1, col1, row2, col2)

PURPOSE:	Convolve two matrices.
matrix1, matrix2	Any two matrices.
row1	(Optional). The start row in matrix 1. The default is 1.
col1	(Optional). The start column in matrix 1. The default is 1.
row2	(Optional). The end row in matrix 1. The default is the number of image matrix rows.
col2	(Optional). The end column in matrix 1. The default is the number of image matrix columns.

RETURNS: A convolved matrix.

EXAMPLES: W1 contains an image matrix:

```
4   8   2
1   1   3
3   2   2
```

W2 contains a filter kernel matrix:

```
1   3
3   2
```

CONV2D(W1, W2) =

```
4   20   26   6
13  36   28   13
6   16   19   12
9   12   10   4
```

CONV2D(W1, W2, 2, 2, 4, 4) =

```
36   28   13
16   19   12
12   10   4
```

REMARKS: This is a two-dimensional linear convolution.

SEE ALSO: CONV
MOVAVG
AUTOCOR

COPYFILE(file1, file2, behavior)

PURPOSE:	Copies a file.	
file1	String. The existing source file.	
file2	String. The destination file.	
behavior	(Optional). Integer. 0 (default) - don't overwrite the destination file if it already exists; 1 - confirm before overwriting the destination file; 2 - overwrite the destination file without confirmation.	
RETURNS:	An error if file 1 does not exist, 1 if the copy is successful, 0 if it is not.	
EXAMPLE:	COPYFILE("c:\expo\data.dat", "c:\mydata\data.dat", 2)	
REMARKS:	If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).	
SEE ALSO:	GETPATH	DELFILE
	FILEEXISTS	MOVEFILE

COPYWIN(source_window, target_window, oldvar, newvar)

PURPOSE:	Copies the formula, annotations, and attributes from a source window to a target window, or replaces one variable for another. MarketBrowser reevaluates the window into which the contents were pasted after the copy.	
source_window	The window whose formula and attributes you wish to copy.	
target_window	(Optional). The window into which the copied selection is pasted. Defaults to the current window.	
oldvar	(Optional). String. A variable in a window which you would like to replace with newvar.	
newvar	(Optional). String. A variable that will replace all instances of oldvar when the window contents are copied into the target window.	
RETURNS:	Nothing.	
EXAMPLE:	Given the window and it's formula: W1: READAHIST("data.dat", "D", 2, 1); WINCOLOR(RED); TEXTANN(.1, .9, "My Data") The formula: W2: COPYWIN(W1, W3) copies the formula in W1 and pastes it into W3, then reevaluates W3.	
SEE ALSO:	GETDATAREF	

CROSSCOR(series, series)

PURPOSE: (A macro). Performs a time domain cross-correlation of a series.

series A series or table.

RETURNS: A series or table.

EXPANSION: $\text{CONV}(S1, \text{REVERSE}(S2)) / (\text{SERSIZE}(S1) + \text{SERSIZE}(S2))$

EXAMPLE: W1: GSIN(128, 1/128/ 4.0)
W2: GSIN(128, 1/128, 4.0)
W3: CROSSCOR(W1, W2)
performs a cross-correlation of a sine wave with a random series.

REMARKS: The cross-correlation function is often used to indicate how "similar" one waveform is to another. The cross-correlation of the above sine waves produces a waveform with several distinct peaks, indicating that the two series are very similar at each point in time where the peaks occur.

SEE ALSO: CONV FFT
AUTOCOR

CROSSPROD(matrix1, matrix2, make nu, inherit)

PURPOSE: Calculates the matrix crossproduct of one or two conforming matrices, that is, multiplies the transpose of a matrix by itself or another matrix. This routine is especially useful for covariance and correlation matrices.

matrix1 The matrix to manipulate, typically a variable. The matrix must be nfilled and rectangular.

matrix2 (Optional). The second matrix. The matrix must be nfilled and rectangular.

make nu (Optional). 0 - maintain horizontal units of original matrix, 1 - make horizontal units NU ("No Units"). Default is 1.

inherit (Optional). 0 - do not inherit attributes, 1 - inherit attributes, such as vertical units and comments from matrix2 (or matrix1 if matrix2 is not present). Default is 1.

RETURNS: A matrix.

REMARKS: If only matrix1 is supplied, MMULT(TRANSPPOSE(matrix1),matrix1) is calculated. If matrix1 and matrix2 are supplied, MMULT(TRANSPPOSE(matrix1),matrix2) is calculated.

SEE ALSO: MMULT
TRANSPPOSE

CUMVOLMON(**instrument**)

- PURPOSE:** Registers a volume data instrument to be updated as a real-time series of the day's cumulative volume.
- instrument** A cumulative volume instrument and its field, in quotes.
- RETURNS:** A series.
- EXAMPLE:** CUMVOLMON("IBM.VOL")
- REMARKS:** The series returned by CUMVOLMON will be a simple series whose value increases with time. Viewing volume as a cumulative total over the course of the day is only one way to monitor volume.
- SEE ALSO:** ABS
VOLMON
EQUIVOLMON
MONITOR

CURPOS(**window, item, cursor_num**)

- PURPOSE:** Returns the last position of the crosshair cursor in a window.
- window** (Optional). Window reference. Defaults to the current window.
- item** (Optional). An integer specifying the item number (i.e. series) in the window. Defaults to 1.
- cursor_num** (Optional). An integer specifying the cursor number. 1 = First Cursor, 2 = Second Cursor. Defaults to 1.
- RETURNS:** The point or index number in the specified series or item where the cursor was most recently placed. If the cursor was never activated in the given window, the point number returned is 0.
- EXAMPLES:** EXTRACT(W1,CURPOS(W1),10)
extracts 10 points from the series in window 1, starting from wherever the cursor was last placed.
- CURPOS(W1,1,2)
returns the index number of the second cursor location for the first series in window 1.

CURRENTFOCUS

- PURPOSE:** Returns the series from the current focus of the window.
- RETURNS:** A series.
- EXAMPLE:** Given the formula in W1:
Grandom(100,1); OVERLAY(GRANDOM(100,1,1,2)); FOCUS(2)
W1: CURRENTFOCUS
returns the second series in the window.
- SEE ALSO:** GETFOCUS
FOCUS
CURRENT

CURSOROFF, CURSORON

- PURPOSE:** Adds/removes the point cursor and its special features from the current, active window. CURSORON is equivalent to pressing the cursor button or the F9 key.
- RETURNS:** Nothing. Arrow keys become (in)active and the current point values (dis)appear.

DATESTR(window, date)

- PURPOSE:** Returns the index number of the data point corresponding to a given date.
- window** (Optional). A window reference. Defaults to the current window.
- date** A date string, in quotes.
- RETURNS:** An integer nearest the specified date.
- EXAMPLE:** DATESTR(W4, "4/14/87")
returns 72 if window 4 contains daily data starting on 1/1/87.
- SEE ALSO:** STRDATE

DDEADVISE(channel, type, overwrite, autoscale, item)

PURPOSE:	Automatically retrieves a series item from a DDE conversation whenever the item changes.
channel	An integer specifying the DDE channel number
type	(Optional). An integer specifying the type of data to retrieve (default 0 - ASCII)
overwrite	(Optional). An integer. 0: append new data to existing data, 1: overwrite existing data with new data (default 0).
autoscale	(Optional). An integer, 0: do not automatically scale the window to the range of the new data, 1: autoscale the window (default 1)
item	Quoted string. A string specifying the item to retrieve.

RETURNS: A series representing the value of the item requested.

EXAMPLE: CHAN = DDEINIT("Excel", "Sheet1")
DDEADVISE(CHAN, "R1C1:R100C1")

establishes a DDE conversation with Excel, returns the value of the cells in row 1, column 1 through row 100 column 1 as a series in the current window. Whenever a cell changes, the new series is appended to the existing series.

REMARKS: The optional type argument specifies the data type of the incoming data:

- ASCII
- ASCII comma or space delimited data (default)BYTE
- Binary signed byteUBYTE
- Binary unsigned byteSINT
- Binary signed 2 byte integerUINT
- Binary unsigned 2 byte integerLONG
- Binary signed 4 byte integerFLOAT
- Binary 4 byte single precision IEEE floatDOUBLE
- Binary 8 byte double precision IEEE float

If **overwrite** is set to 1, the new data overwrites the existing data. If **autoscale** is set to 0, the window scales do not automatically adjust to fit the range of the new data. Use DDEUNADVISE to terminate a DDEADVISE operation. DDEADVISE uses an explicit DDE channel number. DDELINK is similar to DDEADVISE but the channel number is managed internally.

SEE ALSO: DDEGETDATA
DDELINK
DDEUNADVISE

DDEEXECUTE(chan, command)

- PURPOSE:** Executes a command in another application.
- chan** An integer specifying DDE channel number.
- command** Quoted string that specifies the command to execute.
- RETURNS:** A 1 if successful, otherwise returns 0 indicating an error.
- EXAMPLE:** CHAN = DDEINIT("winword")
DDEEXECUTE(CHAN, '[Insert "This is a DDE string"']')
DDETERM(CHAN)
establishes a DDE conversation with Word, inserts the text This is a DDE string at the current cursor location, then terminates the conversation.
- REMARKS:** When MarketBrowser acts as the server, the command string can be any valid MarketBrowser command. For example:
Integ(W1)
- SEE ALSO:** DDELINK DDEPOKE
DDEREQUEST

DDEGETDATA(channel, type, item)

- PURPOSE:** Retrieves a series item from a DDE conversation.
- channel** An integer specifying DDE channel number.
- type** (Optional). An integer specifying the type of data to retrieve (default is ASCII).
- item** Quoted string specifying the item to retrieve.
- RETURNS:** A series representing the value of the requested item.
- EXAMPLE:** CHAN = DDEINIT("Excel", "Sheet1")
DDEGETDATA(CHAN, "R1C1:R100C1")
DDETERM(CHAN)
establishes a DDE conversation with Excel, returns the value of the cells in row 1 column 1 through row 100 column 1 as a series in the current window and then terminates the conversation.
- REMARKS:** The optional type argument specifies the data type of the incoming data and must be one of the following:
- ASCII - ASCII comma or space delimited data (default)
 - BYTE - Binary signed byte
 - UBYTE - Binary unsigned byte
 - SINT - Binary signed 2 byte integer
 - UNIT - Binary unsigned 2 byte integer

- LONG - Binary signed 4 byte integer
- FLOAT - Binary 4 byte single precision IEEE float
- DOUBLE - Binary 8 byte double precision IEEE float

DDEGETDATA always returns a series. Use DDEREQUEST to get a string.

DDEINITIATE(app, topic, item, server, autostart)

PURPOSE: Begins a DDE Conversation.

app Quoted string specifying the application name.

topic (Optional). Quoted string specifying the topic name.

item (Optional). Quoted string specifying the item name.

server (Optional). Quoted string specifying the name of the server executable.

autostart (Optional). Integer. 1: start server, 0: don't start (default)

RETURNS: A positive integer representing the channel number for subsequent DDE operations. If the DDE conversation can't be established, 0 is returned.

EXAMPLE: CHAN = DDEINIT("Excel", "Sheet1")

DDEREQUEST(CHAN, "R1C1")

DDETERM(CHAN)

establishes a DDE conversation with Excel, returns the value of the cell in row 1, column 1 as a string and then terminates the conversation. To automatically start Excel if it is not already running, try:

CHAN = DDEINIT("Excel", "Sheet1", "", "C:\excel\excel", 1)

REMARKS: The "app", optional "topic" and optional "item" strings can also be placed in one string of the following format: "app|topic|item".

For example:

DDEINITIATE("app|topic|item", "server", autostart)

CHAN = DDEINIT("Excel|Sheet1")

MarketBrowser supports the "**Commands**" and "**System**" topics when acting as a DDE server. The "**Commands**" topic is for normal interaction with MarketBrowser. The "**System**" topic supports the following "**items**":

- "SysItems" - all items under the System topic
- "Topics" - all topics
- "Formats" - supported Clipboard formats

SEE ALSO: DDELINK
DDETERMINATE

DDELINK(app, topic, item, server, autostart, startmode, type, overwrite, autoscale)

PURPOSE:	Retrieves a series item from a DDE conversation whenever the item changes. The DDE channel number is managed internally.
app	Quoted string specifying the application name.
topic	Quoted string specifying the topic name.
item	Quoted string specifying the item to retrieve.
server	Quoted string specifying the name of server executable.
autostart	(Optional). An integer, 1: start server, 0: don't start (default).
startmode	(Optional). An integer, 1: normal (default), 2:icon, 3: full size.
type	(Optional). An integer specifying the type of data to retrieve (default 0 - ASCII).
overwrite	(Optional). An integer. 0: append new data to existing data, 1: overwrite existing data with new data (default 0).
autoscale	(Optional). An integer, 0: do not automatically scale the window to the range of the new data, 1: autoscale the window (default 1).

RETURNS: A series representing the value of the item requested.

EXAMPLE: DDELINK("Excel", "Sheet1", "R1C1:R100C1")
establishes a DDE conversation with Excel, returns the value of the cells in row 1, column 1 through row 100 column 1 as a series in the current window. Whenever a cell changes, the new series is appended to the existing series.

REMARKS: Use DDEUNLINK to terminate a DDELINK operation. The "app", optional "topic" and optional "item" strings can also be placed in one string of the following format: "app|topic|item", that is, DDELINK("app|topic|item", "server", autostart, startmode, type, overwrite, autoscale). For example:

```
DDELINK("Excel|Sheet1!R1C1:R100C1")
```

DDELINK combines DDEINIT and DDEADVISE into one function.

See DDEINIT and DDEADVISE for a discussion of the optional arguments.

SEE ALSO: DDEADVISE
DDEGETDATA
DDEINITIATE
DDEUNLINK

DDEPOKE(channel, item, data)

PURPOSE: Sends data to a DDE conversation in string form.

channel An integer specifying the DDE channel number.

item Quoted string specifying the item of the data destination.

data A string, number or series - the data to send

RETURNS: A 1 if successful, otherwise returns 0 indicating an error.

EXAMPLES: CHAN = DDEINIT("Excel", "Sheet1")

DDEPOKE(CHAN, "R1C1", 12.7)

DDETERM(CHAN)

establishes a DDE conversation with Excel, sends the value 12.7 as a string to the cell in row 1, column 1 and then terminates the conversation.

DDEPOKE(CHAN, "R1C1:R100C1", W1*10)

sends the entire series of W1*10 as a CR-LF delimited string to the cells in row 1 column 1 through row 100 column 1

REMARKS: DDEPOKE always converts the data into an appropriate string format.

SEE ALSO: DDELINK
DDEREQUEST

DDEREQUEST(channel, item)

PURPOSE: Retrieves a string item from a DDE conversation.

channel An integer specifying the DDE channel number.

item A quoted string specifying the item to retrieve.

RETURNS: A string representing the value of the item requested.

EXAMPLE: CHAN = DDEINIT("Excel", "Sheet1")

DDEREQUEST(CHAN, "R1C1")

DDETERM(CHAN)

establishes a DDE conversation with Excel, returns the value of the cell in row 1, column 1 as a string and then terminates the conversation.

REMARKS: DDEREQUEST always returns a string. Use DDEGETDATA to obtain a series.

SEE ALSO: DDEGETDATA DDELINK
DDEPOKE

DDESTATUS

PURPOSE: Reports the error status of the last DDE operation.

RETURNS: A string indicating the status of the last DDE operation.

EXAMPLE: CHAN = DDEINIT("DummyAPP", "DummyTopic")
DDESTATUS

returns:

DDE STATUS: DMLERR_NO_CONV_ESTABLISHED

indicating the conversation could not be established.

REMARKS: The following DDE errors are reported:

- DMLERR_ADVACKTIMEOUT - A request for a synchronous advise operation has timed out.
- DMLERR_BUSY - The responding application is busy.
- DMLERR_DATAACKTIMEOUT - A request for a synchronous data operation has timed out.
- DMLERR_DLL_NOT_INITIALIZED - A DDE function was called before DDEINITIATE.
- DMLERR_DLL_USAGE - An application that is not a DDE server has attempted server operations.
- DMLERR_EXECACKTIMEOUT - A request for a synchronous execute operation has timed out.
- DMLERR_INVALIDPARAMETER - A parameter failed to be validated by the DDEML.
- DMLERR_LOW_MEMORY - An application has created a prolonged race condition where the server application outruns the client, causing large amounts of data to be consumed.
- DMLERR_MEMORY_ERROR - A memory allocation failed.
- DMLERR_NOTPROCESSED - An operation failed.
- DMLERR_NO_CONV_ESTABLISHED - A client's attempt to establish a conversation has failed.
- DMLERR_POKEACKTIMEOUT - A request for a synchronous poke transaction has failed.
- DMLERR_POSTMSG_FAILED - An internal call to the PostMessage function has failed.
- DMLERR_REENTRANCY - An application instance with a synchronous operation already in progress attempted to initiate another synchronous operation.
- DMLERR_SERVER_DIED - A server-side operation was attempted on a conversation that was terminated by the client, or the server terminated before completing an operation.
- DMLERR_SYS_ERROR - An internal error occurred in the DDEML.
- DMLERR_UNADVACKTIMEOUT - A request to end an advise operation has timed out.
- DMLERR_UNFOUND_QUEUE_ID - An invalid identifier was passed to a DDEML function.
- OK - No error.

DDETERMINATE(channel1, ..., channeln)

- PURPOSE:** Terminates a DDE Conversation.
- channeln** An integer list of DDE channel numbers returned by DDEINITIATE.
- RETURNS:** Channel number of the last terminated conversation.
- EXAMPLES:** CHAN = DDEINIT("Excel", "Sheet1")
DDEREQUEST(CHAN, "R1C1")
DDETERM(CHAN)
establishes a DDE conversation with Excel, returns the value of the cell in row 1, column 1 as a string and then terminates the conversation.
DDETERM(1, 3, 2)
terminates the conversations with channel numbers 1, 3 and 2.
- REMARKS:** All DDE conversations that were initiated by MarketBrowser are automatically terminated upon exit from MarketBrowser.
- SEE ALSO:** DDEINITIATE

DDEUNADVISE(channel, item)

- PURPOSE:** Ends a previous DDEADVISE operation.
- channel** An integer specifying the DDE channel number.
- item** Quoted string specifying the item.
- RETURNS:** A 1 if successful otherwise returns 0 indicating an error.
- EXAMPLE:** CHAN = DDEINIT("Excel", "Sheet1")
DDEADVISE(CHAN, "R1C1:R100C1")
perform other operations ...
DDEUNADVISE(CHAN, "R1C1:R100C1")
establishes a DDE conversation with Excel, returns the value of the cells in row 1, column 1 through row 100 column 1 as a series in the current window. Whenever a cell changes, the new series is appended to the existing series. Lastly, terminate the link.
- REMARKS:** DDEUNADVISE only terminates the advise operation, the DDE channel is still valid for other DDE operations. You can also compose a DDEUNADVISE statement with the following format:
DDEUNLINK("app|topic!item")
- SEE ALSO:** DDEADVISE DDELINK

DDEUNLINK(**app, topic, item**)

PURPOSE:	Ends a previous DDELINK operation.
app	Quoted string specifying the application name.
topic	Quoted string specifying the topic name
item	Quoted string specifying the item to retrieve
RETURNS:	A 1 if successful otherwise returns 0 indicating an error.
EXAMPLE:	<pre>DDELINK("Excel", "Sheet1", "R1C1:R100C1") perform other operations ... DDEUNLINK("Excel", "Sheet1", "R1C1:R100C1")</pre> establishes a DDE conversation with Excel, returns the value of the cells in row 1, column 1 through row 100 column 1 as a series in the current window. Whenever a cell changes, the new series is appended to the existing series. Lastly, terminate the link.
REMARKS:	The "app", optional "topic" and optional "item" strings can also be placed in one string of the following format: "app topic item". For example: <pre>DDEUNLINK("Excel Sheet1 R1C1:R100C1")</pre>
SEE ALSO:	DDEADVISE DDEGETDATA DDELINK

DECIMATE(**series, n, start_pt, blocksize**)

PURPOSE:	Linearly decimates (reduces) a series by a factor n.
series	A series or table.
n	An integer used to decimate the series.
start_pt	(Optional). Integer indicating where to begin decimation; defaults to 1.
blocksize	(Optional). Integer indicating how often to repeat the decimation; defaults to 1.
RETURNS:	A series or table.
EXAMPLES:	<pre>DECIMATE(W1,3)</pre> reduces the series in W1 by a factor of 3 and places the result in the current window. The new series consists of every third point from the W1 series. <pre>DECIMATE(EXTRACT(W2,10,LENGTH(W2)-10),4)</pre> decimates the series from window 2 by a factor of 4, starting from the 10th point of the series, and places the result in the current window.

DECIMATE(MOVMAX(W1,10),10)

calculates the 10 point moving maximum of non-overlapping blocks.

SEE ALSO: INTERPOLATE
MERGE
REMOVE

DEFDATE(date)

PURPOSE: Sets the default beginning date for any date-oriented series that has no date explicitly set.

date A calendar date, in quotes.

RETURNS: Nothing.

EXAMPLE: DEFDATE("12/01/82")
sets the origin of any series subsequently generated to December 1, 1982.

REMARKS: The default has effect only with series that have date oriented horizontal units (like DAYS or MONTHS).

SEE ALSO: GETDATE
SETDATE
DEFTIME
GETTIME
SETTIME

DEFHUNITS(unit)

PURPOSE: Sets the default horizontal units for the worksheet.

unit A string identifying the unit, in quotes.

RETURNS: Nothing.

EXAMPLE: DEFHUNITS("DAYS")
sets the default horizontal units of the worksheet to DAYS.

REMARKS: The default is applied only to series that have no horizontal units defined or inherited from another series (technically, horizontal units == "NU").

WRAPPER	EXPANSION
DAILY	DEFHUNITS('D')
WEEKLY	DEFHUNITS('WK')
MONTHLY	DEFHUNITS('MO')
QUARTERLY	DEFHUNITS('QTR')
YEARLY	DEFHUNITS('YR')

SEE ALSO: SETHUNITS GETHUNITS

DEFMACRO(name, expr, option, invisible, transient)

PURPOSE:	Creates or defines an MarketBrowser Macro.
name	Name of the macro, with optional arguments, in quotes
expr	Macro body: quoted string or expression evaluating to an integer, real, or complex number.
option	1 = no messages; 2 = enclose in single quotes; 3 = enclose in double quotes.
invisible	(Optional). If the macro name does not start with an underscore (_), determines if the macro is displayed in the list displayed with the MACROS command (Custom / Macros / List/Edit). Options are: 0 - Display the macros; 1 - Hide the macros.
transient	(Optional). If the macro name does not start with an underscore (_), determines if the macros is saved with a worksheet. Options are: 0 - Save the macros with the worksheet; 1 - Do not save the macro with the worksheet.
RETURNS:	Nothing.
EXAMPLES:	<p>If window 1 contains GLINE(10,1.0,1.0,0.0), then:</p> <pre>DEFMACRO("M1","MEAN(W1)")</pre> <p>creates a string macro MEAN(W1). This construction is identical to:</p> <pre>#DEFINE M1 MEAN(W1)</pre> <p>If the quotes are dropped from the second argument,</p> <pre>DEFMACRO("M2", MEAN(W1))</pre> <p>MarketBrowser evaluates the current value for MEAN(W1), which in this case is 4.5, and stores it as a macro scalar constant.</p> <p>To create a macro with arguments, the argument list must be included in the macro name, as in:</p> <pre>DEFMACRO("MYMAC(A,B,C)","A*(B+MAX(W1))+ C")</pre>
SEE ALSO:	GETMACRO EVAL

DEFTIME(time)

PURPOSE:	Sets the default beginning time for any time-of-day oriented series that has no start time explicitly set.
time	A valid time stamp, in quotes.
RETURNS:	Nothing.
EXAMPLE:	<pre>DEFTIME("9:00:00")</pre> <p>sets the origin of any time-of-day series subsequently generated to 9:00 am.</p>
REMARKS:	The default has effect only with series which have time-of-day horizontal units

(technically, horizontal units == "RT").

SEE ALSO: GETTIME SETTIME
 DEFDATE GETDATE
 SETDATE

DEG

PURPOSE: (A macro). Returns the degrees per radian ($360/2*\pi$).

RETURNS: A number.

**EXPAN-
SION:** 57.29577951308232087680

EXAMPLE: PI/4*DEG
 yields 45, the value of pi/4 in degrees.

SEE ALSO: LN(expr) PI
 E GAMMA
 PHI SETDEGREE

DELALLFUNCTIONS

PURPOSE: Deletes all XPL functions defined in the current MarketBrowser worksheet.

RETURNS: Nothing.

SEE ALSO: DELFUNCTION
 DELALLVARIABLES

DELALLVARIABLES

PURPOSE: Deletes all the XPL variables currently defined for the worksheet.

RETURNS: Nothing.

SEE ALSO: DELALLVAR (shortcut name) DELVARIABLE
 DELALLFUNCTIONS

DELAY(series, n)

PURPOSE:	Offsets a series by n number of points along the x-axis.
series	A series or table.
n	An integer number of points used to offset a series (positive or negative).
RETURNS:	A series or table.
EXAMPLE:	DELAY(W4,3) offsets the series in window 4 by 3 points and places the resulting series in the current window.
REMARKS:	DELAY effectively moves a series n points to the right by setting the first n points of the new series to the value of the first point of the series. The remaining values of the new series are set according to the following: $NEWSER(x)=OLDSER(x-n)$ In other words the nth point of the delayed series is equal to the first point of the original one. The delay amount can only be specified as an integer number of points. For example: DELAY(W1,TRUNC(4.2/DELTAX(W3)))
SEE ALSO:	LAG LEAD EXTRACT

DELETE(series, binseries)

PURPOSE:	Deletes points from a series when the corresponding point in the binary control series is non-zero.
series	A series or table.
binseries	Binary control series or table.
RETURNS:	A series or table.
EXAMPLES:	DELETE(W1, GSER(1, 0, 1)) deletes the first and third point from W1. DELETE(W1, W1>2.0) deletes all the points from W1 that are larger than 2.0.

DELFILE(file, behavior)

PURPOSE:	Deletes a file.
file	String. The file to delete.
behavior	(Optional). Integer. 0 (default) - don't confirm before deleting; 1 - confirm before deleting.
RETURNS:	An error if file1 does not exist, 1 if the deletion is successful, 0 if it is not.
EXAMPLE:	DELFILE("c:\expo\data.dat")
REMARKS:	If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).
SEE ALSO:	GETPATH COPYFILE DIREXISTS FILEEXISTS MKDIR MOVEFILE RMDIR

DELFUNCTION(function)

PURPOSE:	Deletes a named function from the list of currently defined functions in an MarketBrowser worksheet.
function	String that represents the name of the function to delete, optionally in quotes.
RETURNS:	Nothing.
REMARKS:	If function is not defined in the worksheet, MarketBrowser will return an error: Unknown Variable.
SEE ALSO:	DELFUN (shorthand) DELVARIABLE DELALLFUNCTIONS DELALLVARIABLES

DELVARIABLE(var)

- PURPOSE:** Deletes the specified XPL variable as defined in the worksheet.
- var** The variable name, optionally in quotes.
- RETURNS:** Nothing.
- EXAMPLE:** DELVARIABLE("myvar")
deletes the current variable "myvar".
DELVARIABLE(myvar)also deletes the current variable "myvar".
- SEE ALSO:** DELVAR (shortcut name)
DELALLVARIABLES
SETVARIABLE
GETVARIABLE

DELTAx(series)

- PURPOSE:** Returns the delta x increment of a series, i.e. the inverse of the sampling rate.
- series** (Optional). A series or table. Defaults to the current window.
- RETURNS:** A number.
- EXAMPLE:** DELTAX(GLINE(20,2,1,1))
returns 0.5, the inverse of the sampling rate.
- SEE ALSO:** SETDELTAx
RATE

DENSITY(matrix)

- PURPOSE:** Displays matrix data as a density plot.
- matrix** A rectangular matrix of data to plot.
- RETURNS:** A matrix, displayed as density plot.
- EXAMPLE:** DENSITY(RAVEL(GRAND(100,1),10))
fills a window with a randomly colored ten by ten checkerboard. Each square of the checkerboard is filled according to the magnitude of the data, with colors as defined by the current shading scheme.
- SEE ALSO:** SETSHADING
SETPALETTE
SHADEWITH
CONTOUR

DEPEND(target, series, addremove)

PURPOSE:	Adds or removes dependencies between a window or a hot variable.
target	(Optional). The series, in quotes, which you want to make dependent. Defaults to the current window.
series	The series, in quotes, to make target dependent on.
addremove	Integer flag. Whether to add or remove a dependency. <ul style="list-style-type: none">• 0 - Remove• 1 - Add
RETURNS:	Nothing.
EXAMPLE:	Given the following window formulas: W1: LMON("IBM.LAST") W2: LINREG(W1) the following command: W3: DEPEND("W2","W1",0) breaks the dependency of W2 on W1. In this case, W2 will no longer automatically recalculate every time W1 updates.
SEE ALSO:	RTDEPEND CALC

DERIV(series)

PURPOSE:	Returns the derivative of a series or table.
series	A series or table.
RETURNS:	A series or table.
EXAMPLE:	DERIV(W6) creates a new series from the contents of window 6 and places the result in the current window. The value of each point in the new series is the slope of the series in window 6 at that point.
REMARKS:	MarketBrowser calculates the derivative by taking points n, n-1, and n+1, finding the quadratic curve to fit those three points, and using the slope of the curve at point n as the derivative of point n.
SEE ALSO:	AREA INTEG LDERIV RDERIV

DET(matrix)

PURPOSE: Computes the determinant of a matrix.

matrix A real or complex square matrix.

RETURNS: A number.

EXAMPLE: $x = \begin{matrix} 1 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 12 \end{matrix}$

DET(x) = -15

DFT(series)

PURPOSE: Calculates the discrete Fourier Transform of any table or series expression in real/imaginary form.

series Any expression resolving to a series

RETURNS: A table or series.

REMARKS: The DFT produces the same result as an FFT. Although the DFT is a more straightforward method than the FFT is for calculating the discrete Fourier Transform, is also a much slower algorithm.

SEE ALSO: IDFT
FFT

DIAGONAL(series, n)

PURPOSE: Computes a matrix diagonal.

series Any series, multi-series table, or expression resulting in a series or table.

n (Optional). Integer. The diagonal number. Options are:
0 main diagonal (the default)
>0 above main diagonal
<0 below main diagonal.

RETURNS: Nothing

EXAMPLE: W1: GSER(1,2,3)
W2: DIAGONAL(W1)

Yields:

$\begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{matrix}$

SEE ALSO: MMULT

DIFF(order, rate, slope, fc)

- PURPOSE:** Designs an FIR differentiator.
- order** (Optional). The filter length. If specified, the order must be an integer value. The default filter order is 32.
- rate** A real number that specifies the sampling rate of the filter in Hertz.
- slope** (Optional). A real number that specifies the desired slope of the differentiator. The default value is 1.0.
- fc** (Optional). A real number that specifies the cutoff frequency of the differentiator in Hertz. The default value is rate/2.
- RETURNS:** The time domain impulse response of the differentiator.
- EXAMPLES:** DIFF(1000.0)
creates a 32 point differentiator with a default slope of 1.0 and a default cutoff frequency of 500.0 Hz. The resulting differentiator has a slope deviation of 0.006
- DIFF(16, 1000.0, 2.0, 200.0)
creates a similar filter except the order is set to 16, the desired slope is 2.0 and the differentiator band only extends to 200.0 Hz. The resulting differentiator has a deviation of 6.2E - 10 in the 0 to 200 Hz passband.
- REMARKS:** The band edges must lie between 0.0 and rate/2 Hz. The resulting characteristics of the filter are written to an ASCII file named DIFFn.FIR, where n is the nth filter designed. This file can be displayed by using the DIFFS macro. For example, to display the filter characteristic file named DIFF4.FIR, try: DIFFS(4)

DIREXISTS(directory)

- PURPOSE:** Checks to see if a directory exists.
- directory** String. The name of the directory to check.
- RETURNS:** 1 if the directory exists, 0 if it does not exist.
- EXAMPLE:** DIREXISTS("c:\expo\mydir")
- REMARKS:** If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).
- SEE ALSO:** GETPATH DELFILE
COPYFILE FILEEXISTS
MKDIR MOVEFILE
RMDIR

DISPLAY(window1,...,windown)

PURPOSE: Displays a specified set of windows from a worksheet.

**window1,...,
windown** List of windows to display.

RETURNS: Nothing.

EXAMPLE: If a worksheet contains 12 windows,
DISPLAY(W1..W4, W7, W11)
displays only windows 1, 2, 3, 4, 7, and 11. The hidden windows still automatically recalculate. To redisplay all of the windows, use DISPLAYALL.

SEE ALSO: DISPLAY
HIDE

DISPLAYALL

PURPOSE: Displays all windows from a worksheet.

RETURNS: Nothing.

EXAMPLE: DISPLAYALL
displays all windows.

SEE ALSO: DISPLAY
HIDE

DLBIND(libname, func1, ..., funcn)

PURPOSE:	DLBIND loads the shared library object file specified, and adds each named function from the list of names, func1 to funcn, to MarketBrowser's internal loaded library function list.
libname	String enclosed in quotes. The complete path to the shared library object file.
funcn	String enclosed in quotes. The name of the function in the shared library object file to load into MarketBrowser's internal loaded library function list.
RETURNS:	Nothing. May return an error message if not successful.
REMARKS:	DLBIND can be called multiple times for the same library without error. Attempting to load a function with the same name as an already loaded function will overwrite the original function. If libname is not a valid library name, DLBIND can fail catastrophically, causing MarketBrowser to exit. This is an operating system limitation and cannot be prevented by LMT.
SEE ALSO:	DLRUN DLUNBIND ISDLFUNC

DLNABS(x, y, color, style)

PURPOSE:	Draws a line from the current drawing cursor position to the indicated point.
x	x coordinate.
y	y coordinate.
color	Integer color parameter.
style	Integer style parameter. The style parameter controls how the line is drawn and takes the following values: 0 - No visible line 1 - Solid (default) 2 - Dashed 3 - Dotted
RETURNS:	Nothing.
REMARKS:	The coordinates are in "absolute" or "world" coordinates, which are those of the data displayed in the window. This function also relocates the drawing cursor to the endpoint of the line. Unlike LINECUR, this formula has no effect if executed when the window is activated. It must be part of the window formula to have an effect.
SEE ALSO:	DPTABS

DLRUN(funcname, ser1 ... sern, real1 ... realn, string1 ... stringn)

PURPOSE:	Runs the already loaded function specified by funcname with [0..n] series arguments, [0..n] real arguments, and [0..n] quoted string arguments.
funcname	Quoted string. The function to run from the shared object file loaded with DLBIND.
sern	Any series arguments that need to be passed to funcname.
realn	Any real arguments that need to be passed to funcname.
stringn	Any quoted string arguments that funcname requires.
RETURNS:	Whatever value the DLL function returns.
REMARKS:	A function can only be run via DLRUN once it has been bound in using DLBIND.
SEE ALSO:	DLBIND ISDLFUNC DLUNBIND

DLUNBIND

PURPOSE:	Unbinds all DLL functions from MarketBrowser.
RETURNS:	Returns 1 if successful in unloading the functions, otherwise 0.
SEE ALSO:	DLBIND DLRUN ISDLFUNC

DPTABS(x, y, color, show)

PURPOSE:	Draws a point in the current window at the indicated coordinates.
x	x coordinate.
y	y coordinate.
color	Integer color parameter
show	Parameter that determines if the point will be shown.
REMARKS:	The coordinates are in "absolute" or "world" coordinates, which are those of the data displayed in the window. The point is displayed only if "show" is set to 1. Set "show" to 0 for anchoring a line drawing for DLNABS. This function also relocates the drawing cursor to the indicated point. Unlike LINECUR, this formula has no effect if executed when the window is activated. It must be part of the window formula to have an effect.
SEE ALSO:	DLNABS

DTCONCAT(series1, series2)

- PURPOSE:** Concatenates two conforming series while respecting their timebase.
- series1** First input series.
- series2** Second input series. Must conform to series1, that is, it must have the same horizontal units and deltax (for example, both series must be historical 1 minute data not updating in real-time).
- RETURNS:** A series that contains all of series1 and all of the points in series2 that are after the last point in time of series 1.
- EXAMPLE:** W1: Contains daily historical closing data from 1/4/97 to 12/1/97.
W2: Contains daily historical closing data from 9/1/97 to 2/1/98.
W3: DTCONCAT(W1,W2) - This results in a series that contains data from 1/4/97 to 2/1/98 where points from 1/4/97 to 12/1/97 come from W1 and points from 12/2/97 to 2/1/98 come from W2.
- REMARKS:** The CONCAT function just appends one series to the end of another without regard for where points between the two may overlap in time or be missing in time. DTCONCAT considers the timebase when concatenating two series.
- SEE ALSO:** CONCAT

DTEXTRACT(source_win, add_nas, na_interp, inside, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, gap_2_end)

PURPOSE:	Extract a date and time range from intraday, daily, weekly, monthly, or yearly data.
source_win	Valid window or variable containing a series
add_nas	(Optional). Integer. Type of NA processing. Options are: <ul style="list-style-type: none">• 0 - Do not fill gaps with NAs (default).• 1 - Fill all gaps with NAs• 2 - Fill valid gaps on business days inside of trading hours with NAs (represented by gap_1_start/end and gap_2_start/end).
na_interp	(Optional). Integer. Type of interpolation. Options are: <ul style="list-style-type: none">• 1 - Perform linear interpolation through valid gaps• 0 - Leave gaps. (default)
inside	(Optional). Integer. How to process gaps: <ul style="list-style-type: none">• 1 - Keep data INSIDE (within) of gap_1_start or gap_1_end and gap_2_start or gap_2_end (default).• 0 - Keep data OUTSIDE of gap_1_start/gap_1_end and gap_2_start/gap_2_end.
start_date	Quoted string of the form 'mm/dd/yy', which represents the date from which to start extracting data from source_win .
start_time	Quoted string of the form 'hh:mm:ss', representing the time to start extracting data from source_win . To leave the default (the start time of source_win), use "" as a placeholder.
end_date	Quoted string of the form 'mm/dd/yy' representing the date on which to stop extracting data from source_win . Use "" to leave the default, which is the end date of source_win .
end_time	Quoted string of the form 'hh:mm:ss', representing the time to stop extracting data from source_win . To leave the default (the end time of source_win), use "" as a placeholder.

gap_1_start,
gap_2_start (Optional). Quoted string of the form 'hh:mm:ss', representing the beginning of a gap in the extraction of data from **source_win**. To leave the default (00:00:00), use "" as a placeholder.

gap_1_end,
gap_2_end (Optional). Quoted string of the form 'hh:mm:ss', representing the end of a gap in the extraction of data from **source_win**. To leave the default (23:59:59), use "" as a placeholder.

RETURNS: A series.

EXAMPLE: For the following window formula:

```
W1: GLINE(1000,1,1,1); SETDATE("1/1/96"); SETTIME("12:00:00");  
SETDELTA(300); SETHUNITS('RT')
```

```
W2: DTEXTRACT(W1, 2, 0, 0, "01/02/96", "09:00:00", "01/04/96", "17:00:00",  
"11:30:00", "12:00:00", "12:30:00", "13:00:00")
```

returns a series with data in it that falls between 01/02/96 and 01/04/96. Between the times of 11:30 am and 12:00 PM, and 12:30 and 1:00 PM, MarketBrowser fills the series with NA values.

REMARKS: For intraday data, data returned can be restricted to a pair of trading times, represented by gap_1_start/end and gap_2_start/end.

For daily and/or intraday data, interpolation can be performed between date and/or time gaps in the source data on valid business days. na_interp set to 1 allows linear interpolation between last close before gap and first open after gap for trading bar type data, or simply from close-to-close for line-type data.

If a start date is supplied, a start time and end time must also be supplied.

Finally, the add_nas parameter allows gaps to be displayed filled with NA values, when set to 1, or collapses gaps in intraday data to a series to prevent their display.

SEE ALSO: MONITOR
BARMON
CAPTURE
DTSETX

DTSETX(series, start_date, start_time, end_date, end_time)

- PURPOSE:** Sets the visible x-axis range to span between two given dates and/or times.
- series** (Optional). The input series. Defaults to the current window.
- start_date** String of the form mm/dd/yy, in quotes. To leave the default value (the entire range) use -1 in quotes.
- start_time** String of the form hh:mm:ss, in quotes. To leave the default value (the entire range) use -1 in quotes.
- end_date** String of the form mm/dd/yy, in quotes. To leave the default value (the entire range) use -1 in quotes.
- end_time** String of the form hh:mm:ss, in quotes. To leave the default value (the entire range) use -1 in quotes.
- RETURNS:** Nothing.
- EXAMPLE:** If W1 contains three days of intraday data, from 06/04/95 to 06/06/95, the function:
W2: DTSETX(W1,'06/05/95','12:00:00','06/05/95','14:00:00')
displays the data in W1 that falls between 12:00 PM and 2:00 PM on June 5, 1995. The underlying data is not affected, only the way it is displayed is. Note that no changes occur in W2; only W1 is affected by this call.
- SEE ALSO:** SETXY

DTTOINDEX(series, item, member, string)

- PURPOSE:** Matches a date and/or time string to an index into a series.
- series** (Optional). A valid window reference or variable. Defaults to the current window.
- item** (Optional). Which item in the window, i.e., which overplot or overlay.
- member** (Optional). Index into the item. Defaults to first member.
- date_time** A quoted string. The date and/or time for which to return an index. The string can take the form: "mm/dd/yy hh:mm:ss". In this case, a single string provides information on both the date and time. The date and the time are separated by a whitespace. Alternately, dates and times can be specified independently, as individual quoted strings: "mm/dd/yy", "hh:mm:ss"
- To specify both date and time independently, separate the two strings with a comma.
- RETURNS:** An integer, which is the index into the series.
- EXAMPLE:** Given the following window formula:
W1:RTHISTP('XR.X.BID','RT',2,4); OVERLAY(RTHISTP('IBM.BID','RT',2,4))
W2: DTTOINDEX(W1,2,3, "12:13:41")
- returns an integer that represents the index to the point in the second item, third column, that comes closest to the time 12:13:41.
- REMARKS:** The index into the series has as origin 1. If date_time is outside the time span of the series, DTTOINDEX returns the index to the point closest to the given time. It does not return an error.
- SEE ALSO:** INDEXTODT
DTTOVAL
STRDATE
DATESTR
JULDAY

DTTOVAL(series, item, member, date, time)

PURPOSE:	Given a date or time, returns the value of the series.
series	(Optional). A valid window or variable reference. Defaults to the current window.
item	(Optional). Integer. Window item number, i.e., which overplot or overlay. Defaults to the first item.
member	(Optional). Integer. The member, or column number, of the item. Defaults to the first member.
date	(Optional). A quoted string of form 'mm/dd/yy'.
time	(Optional). A quoted string of the form 'hh:mm:ss'.
RETURNS:	The value of the item and member at the given date and/or time.
EXAMPLE:	Given the following formula: W1: GLINE(100,1,1,1);SETDATE('1/1/95') W2: DTTOVAL(W1, '1/13/95') returns the value 10.0.
SEE ALSO:	INDEXTODT DTTOINDEX STRDATE DATESTR JULDAY

E

PURPOSE:	(A Macro). Returns Euler's number e ($\text{LN}(e) = 1$).
RETURNS:	A number.
EXPA- SION:	2.7182818284590452353602874
EXAMPLE:	E^3 displays 20.08553692.
SEE ALSO:	LN(expr) PI DEG GAMMA PHI SETDEGREE

ECHO(string, log)

- PURPOSE:** Prints text at the bottom of your screen.
- string** A string, enclosed in quotes.
- log** (Optional). If set to 1, the echo is repeated in the logfile.
- RETURNS:** A string.
- EXAMPLES:** ECHO("hi")
prints "hi" at the bottom of your screen.
- ECHO(STRCAT("MIN W1: ", STRNUM(MIN(W1))))
prints "Min W1: 3 " at the bottom of your screen, (with 3 representing the minimum value of window 1).

EDIT(series, string)

- PURPOSE:** Permits point-by-point editing of a series y-axis values.
- series** (Optional). A series or table. Use this argument to copy the contents of a window in an empty one.
- string** (Optional). The new name for the series. The default string is "*EDIT SERIES*". This must be in quotes.
- RETURNS:** Nothing, unless the series argument has been used to create a series.
- EXAMPLE:** The following formula is typed in window 3:
EDIT(W2)
This displays a table of points for the W2 series and puts a cursor on the first point. The arrow and Page keys move the cursor through the list of values.
To change a value, move the cursor to that point, type the new y-value at the bottom of the screen, and press <CTRL> then enter. To return the old value before pressing <CTRL>, press <ESC>.
To leave the EDIT function, press <ESC>. This will plot the new series in window 3.
- REMARKS:** It is generally more convenient to edit data directly in a window by switching to a tabular view, turning on the data value cursor, and entering new values directly.
- SEE ALSO:** PROTECT
EXTRACT
TABLE
TABLEVIEW

EIGVAL(matrix)

- PURPOSE:** Computes the Eigenvalues of a square matrix.

matrix A real or complex square matrix.

RETURNS: A series with as many rows as the input matrix. Each entry in the series is an Eigenvalue. The Eigenvalue in row n of EIGVAL corresponds to the Eigenvector in column n of EIGVEC.

EXAMPLE: x = 1 3 4
 5 6 7
 8 9 12

EIGVAL(x) = 19.964
 -1.4739
 0.50976

SEE ALSO: EIGVEC
BALANCE
NBEIGVAL
NBEIGVEC

EIGVEC(matrix)

PURPOSE: Computes the Eigenvectors of a square matrix.

matrix A real or complex matrix.

RETURNS: A square matrix of the same dimensions as the input matrix. Each column of the output matrix is an Eigenvector. The Eigenvector in column n of EIGVEC corresponds to the Eigenvalue in row n of EIGVAL.

EXAMPLE: x = 1 3 4
 5 6 7
 8 9 12

EIGVEC(x) = 0.25387 0.89628 0.046508
 0.50456 -0.27028 -0.80186
 0.82521 -0.35162 0.5957

SEE ALSO: BALANCE
EIGVAL
NBEIGVAL
NBEIGVEC

ELLIPTIC(type, order, rate, pb1, pb2, ripple, atten, sb1, sb2)

PURPOSE:	Designs a digital IIR Elliptical filter.
type	Integer filter type. 1 = Lowpass, 2 = Highpass, 3 = Bandpass, 4 = Bandstop.
order	(Optional). The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.
rate	A real number that specifies the sampling rate of the filter in Hertz.
pb1	A real number that specifies the first passband edge frequency of the filter in Hertz.
pb2	A real number that specifies the second passband edge frequency of the filter in Hertz.
ripple	A real number for the passband ripple in dB.
atten	A real number for the stopband attenuation in dB.
sb1	A real number that specifies the first stopband edge frequency of the filter in Hertz.
RETURNS:	The filter coefficients in cascade form.
EXAMPLES:	<p>ELLIPTIC(1, 4, 1000.0, 100.0, 2.0, 50.0, 200.0)</p> <p>creates an Elliptic lowpass filter with a sampling rate of 1000 Hz, order of 4, and a cutoff frequency of 100 Hz. Its passband ripple is set to 2.0 dB, and stopband attenuation is set to 50 dB. The desired stopband edge is 200 Hz.</p> <p>ELLIPTIC(1, 1000.0, 100.0, 2.0, 50.0, 200.0)</p> <p>creates a similar filter to above except the filter order is not specified.</p> <p>ELLIPTIC(3, 4, 1000.0, 220.0, 300.0, 2.0, 50.0, 180.0, 350.0)</p> <p>creates an Elliptic bandpass filter with a sampling rate of 1000 Hz. The first stopband edge is 180 Hz and last stopband edge is 350 Hz. The order is 4. The passband ripple is 2 dB and the stopband attenuation is 50 dB..</p> <p>ELLIPTIC(3, 1000.0, 220.0, 300.0, 2.0, 50.0, 300.0, 350.0)</p> <p>creates a similar filter to above except the filter's order is not specified.</p>
REMARKS:	The band edges must lie between 0.0 and rate/2 Hz. The cutoff frequency must be less than the stopband edge frequency.

ENTRY(matrix, row, col)

PURPOSE: (A Macro). Returns a single element from a matrix.

matrix A matrix or table

row An integer. The row index.

col An integer. The column index.

RETURNS: A number.

**EXPAN-
SION:** GETPT(GETSER(matrix, col), row)

EQUIVOL(price, volume, size)

PURPOSE: Creates an Equivolume plot of price and volume data.

price A price series.

volume A volume series.

size Integer. The number of observations per bar.

RETURNS: A series.

EXAMPLE: EQUIVOL(W1, W2, 10)

The data is shown as equivolume bars, where the top and bottom of the bars show the period high and low, and the width of the bar represents the period volume.

SEE ALSO: EQUIVOLMON
CANDLESTICK (a macro)

EQUIVOLMON(price, volume)

- PURPOSE:** Registers a price data item and a volume data item for updating in an accumulated data series.
- price** A valid price data identifier for your data service, in quotes.
- volume** A valid volume data identifier for your data service, in quotes.
- RETURNS:** A series that grows with time. The series may start out with no observations. It may be filled with earlier values, if available, from your data service.
- EXAMPLE:** EQUIVOLMON("IBM.N.LAST", "IBM.N.VOL")
causes the window to monitor price and volumes for IBM on the NYSE. The window is reevaluated periodically, as specified by the real-time interval that is set in the function RTINTERVAL.
The data is shown as equivolume bars, where the top and bottom of the bars show the period high and low, and the width of the bar represents the period volume.
- REMARKS:** The symbol naming convention depends on the data service being used.
If EQUIVOLMON is used while real-time updating is in effect, MarketBrowser automatically attempts to return an intra-day history, using the RTHISTORY function.
Note that the volume being monitored must be cumulative, rather than incremental.
- SEE ALSO:** ABSVOLMON
CUMVOLMON
EQUIVOL
RTINTERVAL
MONITOR
CANDLESTICK (a macro)

ERF(expr)

- PURPOSE:** Returns the error function of a series, table or number.
- expr** A series, table or number.
- RETURNS:** A series, table or number.
- EXAMPLES:** ERF(10)
displays 0.8427 at the bottom of the screen.
ERF(W1)
returns a series.
- SEE ALSO:** ERFC

ERFC(expr)

- PURPOSE:** Returns the complementary error function of a series, table or number.
- expr** A series, table or number.
- RETURNS:** A series, table or number.
- EXAMPLES:** ERFC(10)
displays 0.15729921 at the bottom of your screen.
ERFC(W1)
returns a series.
- SEE ALSO:** ERF

ERRORBAR(bartop, sticktop, stickbottom, barbottom, midpoint, tees)

- PURPOSE:** Displays four or five conforming series of data as errorbars/candlesticks.
- bartop** A series (or, rectangular matrix of four or more columns).
- sticktop** A series.
- stickbottom** A series.
- barbottom** A series.
- midpoint** A series.
- tees** An integer. On = 1; Off = 0. The default is 1.
- EXAMPLE:** If W1 is a series of 10 observations,
ERRORBAR(W1, 1.1*W1, 1.2*W1, 0.8*W1, 0.9*W1)
plots ten errorbars, with the background filled.
- REMARKS:** The ordering of the data series is important. The observations in **sticktop** should all be greater than the corresponding observations in **stickbottom**, and so on. When present, the **midpoint** series determines how the bars will be filled. Each observation of **midpoint** should range between the corresponding observations in **bartop** and **barbottom**. The region of the bar from **bartop** to **midpoint** is color filled. The region from **midpoint** to **barbottom** is background filled. If **midpoint** is equal to **bartop**, the bar is completely filled. If **midpoint** is equal to **barbottom**, the bar is completely empty. When **midpoint** is not supplied, the bars are filled when **bartop** exceeds **barbottom**, and empty when the converse is true. This has the effect of producing traditional Japanese trading candlesticks for data which is ordered as Close/High/Low/Open. All data series must have the same length. The **tees** option draws the tails of each errorbar as a "tee" rather than as a "wick" (single line).

EVAL(string)

PURPOSE: Evaluates any command.

string Any string which is a valid command.

RETURNS: Depends on the contents of the command string.

EXAMPLES: EVAL("1/W1; WINCOLOR(BLUE)")

works exactly as if it were typed in the command line; it puts the inverse of W1 into the current window and then sets the window color to blue.

DEFMACRO("TEST",1); EVAL("TEST")

displays "1" demonstrating that MarketBrowser performs macro substitutions and then executes the statements. If the reference to "TEST" in the second statement is not wrapped in EVAL, MarketBrowser does not handle the line because DEFMACRO has not yet created the "TEST" macro. With EVAL, "TEST" gets evaluated as the line is executed, not as it is typed in.

SEE ALSO: CAST
DEFMACRO
PASS
WHILE

EVALNOMACROS(string)

PURPOSE: Evaluates any command but suppresses macro substitution.

string Any valid command, in quotes.

RETURNS: Depends on the contents of the command string.

EXAMPLE: EVALNOMACROS("1/W1; WINCOLOR(BLUE)")

works exactly as if it were typed in the command line; it puts the inverse of W1 into the current window and then sets the window color to blue.

REMARKS: EVALNOMACROS works just like the EVAL function except it suppresses macro substitution. This is significant and useful only when EVAL() is getting called in a tight loop.

SEE ALSO: EVAL CAST
DEFMACRO PASS
WHILE

EVALTOSTR(string)

- PURPOSE:** Evaluates a string, and returns its value as a string.
- string** A string, enclosed in quotes.
- RETURNS:** A string.
- EXAMPLE:** #define mx evaltostr("max(W1)") menulist(mx)
pops up a menu containing the maximum value of W1.
- REMARKS:** If EVALTOSTR cannot cast the return value back as a string, it will not return a value. This would be the case, for example, with evaltostr('gser(1,2,3)')
- SEE ALSO:** EVAL PASS CASTSTRING

EXP(expr)

- PURPOSE:** Raises the constant e (=2.71828...) to a specified power.
- expr** A series, table, integer or real number.
- RETURNS:** A series, table or number.
- EXAMPLES:** EXP(W2)
creates a new series from the contents of window 2 and places the result in the current window. The value of each point in the new series is e raised to the value of the corresponding point in window 2.
- EXP(1)
displays 2.71828182...
- SEE ALSO:** LN(expr)

EXPANDH, EXPANDV(factor)

- PURPOSE:** Expands a series in the current, activated window.
- factor** (Optional). A ratio of the current series dimension to the desired dimension. Default is 2/3, which makes the series appear 3/2 of its original size.
- REMARKS:** EXPANDH expands a series horizontally; EXPANDV expands a series vertically. To return the window to its original display, use
- a) the reciprocal ratio of the argument you chose before,
 - b) COMPRESSH or COMPRESSV with the same argument, or
 - c) <CTRL>+<HOME>.
- Pressing <CTRL>+<-> > or <CTRL>+<↑> when the current window is active.
- SEE ALSO:** COMPRESSH COMPRESSV

EXPM(matrix)

- PURPOSE:** Calculates the exponential of a matrix.
- matrix** Window or variable reference that contains a square matrix.
- RETURNS:** A matrix.
- REMARKS:** The algorithm used to calculate the exponential of a matrix is the Padé Approximation Model, as described in *Matrix Computations*, by Gene H. Golub and Charles F. Van Loan, The Johns Hopkins University Press, London, 1989, second edition, pp. 555-558.
- SEE ALSO:** INNERPROD
EXP

EXTRACT(series, start, length, offset)

- PURPOSE:** Extracts any part of a series.
- series** A series or table.
- start** An integer. The starting point.
- length** An integer. The number of points to be extracted. A value of -1 causes all values from start through the end of the series to be extracted.
- offset** X offset of the resultant series. The default is 0.0.
- RETURNS:** A series or table.
- EXAMPLE:** EXTRACT(W5,100,-1)
places all points from window 5 except for the first 100 in the current window.
- REMARKS:** Start can be negative. If points are needed outside the ranges of the series, then it is padded with zeros.
- SEE ALSO:** LENGTH
CONCAT
REVERSE
DELAY
EDIT
IMPULSE
REMOVE
REPLICATE

FCLOSE(filename)

PURPOSE: Closes a file that was opened using FOPEN.

filename The name of the file to close, in quotes.

RETURNS: 1 if the file was successfully closed; otherwise 0.

EXAMPLE: FCLOSE("myfile")
closes "myfile", and returns a 1 in the status line if the file was successfully closed.

REMARKS: All files opened with FOPEN should be closed with the FCLOSE or FCLOSEALL functions prior to exiting from MarketBrowser.

SEE ALSO: FCLOSEALL FOPEN
FFLUSH FGETS
FPUTS FSEEK
FTELL FREADA
FREADB FWRITEA
FWRITEB

FCLOSEALL

PURPOSE: Closes all files that were opened using FOPEN.

RETURNS: 1 if all files were successfully closed; otherwise returns 0.

REMARKS: All files opened with FOPEN should be closed with the FCLOSE or FCLOSEALL functions prior to exiting from MarketBrowser.

SEE ALSO: FCLOSE
FOPEN

FFLUSH(filename)

PURPOSE: Clears the buffer of input from or output to the specified file.

filename The name of the file, in quotes.

RETURNS: 1 if successful, otherwise 0.

REMARKS: If the file was open for output, the remaining contents of the buffer are written to the file. Use with FOPEN and FCLOSE

SEE ALSO: FOPEN
FCLOSE

FFT(series)

PURPOSE: Calculates the Fast Fourier transform of a series or table in Cartesian (real/imaginary) form.

series A series or table.

RETURNS: A series or table.

EXAMPLES: Set up a four-window worksheet as:

W1: GSIN(125, 0.01, 1.0)

W2: GSIN(128, 0.01, 1.0)

W3: FFT(W1)

W4: FFT(W2)

Compare the speeds of the two FFTs. The 128 (a power of 2) point FFT should be considerably faster.

REMARKS: The FFT result is complex and MarketBrowser plots the real component of the resultant series. MarketBrowser uses a mixed radix FFT.

Series with lengths equal to a power of 2 are processed faster than series with lengths that are not. Use LENGTH to find if a series is a power of 2 points long. Use EXTRACT to tailor series to lengths such as 512 or 1024.

Use FFTP to get magnitude/phase output and SPECTRUM to get a normalized magnitude plot.

SEE ALSO:

AUTOCOR	CONV
CROSSCOR	SPECTRUM
GHAMMING	GHAMMING
GKAISER	FFTP
IFFT	PSD
DFT	

FFTP(series)

PURPOSE: Calculates the Fast Fourier Transform of a series in polar (magnitude/phase) form.

series Any expression evaluating to a series

RETURNS: A table or a series.

REMARKS: The result of FFTP is complex polar and MarketBrowser plots the magnitude of the resultant series. FFTP uses the same algorithm as the FFT but is slower because it calculates the magnitude/phase. Use SPECTRUM to get a normalized magnitude plot.

SEE ALSO:

FFT	FFTP
PSD	SPECTRUM

FGETS(filename)

PURPOSE: Returns the next line from the specified input file.

filename The name of the input file, in quotes.

RETURNS: A string.

EXAMPLES: If the file header.txt contains the text:

```
ASCII data file  
Interval 20
```

then the commands:

```
FOPEN("header.txt", 'r+')  
FGETS("header.txt")  
FCLOSE("header.txt")
```

display "ASCII data file" at the bottom of the screen before closing the file.

The commands:

```
FOPEN("header.txt", 'r+')  
DEFMACRO("str1", FGETS("header.txt"),2)  
DEFMACRO("str2", FGETS("header.txt"),2)  
FCLOSE("header.txt")
```

return macros str1 equal to "ASCII data file", and str2 equal to "Interval 20" and closes the file.

REMARKS: FGETS must be used in conjunction with FOPEN and FCLOSE. The first time FGETS is called, it returns the first line in the file; subsequent calls to FGETS return the line following the line that was previously returned. If there is a new line character at the end of a line, FGETS will also return that character. There is a buffer limit of 512 characters per line. Lines exceeding 512 characters will take multiple FGETS to read.

SEE ALSO: FOPEN
FCLOSE
FPUTS

FILEEXISTS(file)

PURPOSE:	Checks to see if a file exists.
file	String. The existing source file.
RETURNS:	1 if the file exists, 0 if it does not exist.
EXAMPLE:	FILEEXISTS("c:\expo\data.dat")
REMARKS:	If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).
SEE ALSO:	GETPATH FLOCATE DELFILE COPYFILE

FIR(series, coeff, initseries)

PURPOSE:	Evaluates a finite impulse response difference equation.					
series	A series or table.					
coeff	FIR coefficient series.					
initseries	(Optional). The initial conditions series. Defaults to ??/					
RETURNS:	A series or table.					
EXAMPLES:	<p>A FIR difference equation is of the form: $y(n) = b_0*x(n) + b_1*x(n-1) + b_2*x(n-2) + \dots + b_N*x(n-N)$</p> <p>For example, if $x(n) =$</p> <table><tr><td>1 for $n = 1$</td></tr><tr><td>2 for $n = 2$</td></tr><tr><td>1 for $n = 3$</td></tr><tr><td>1 for $n = 3$</td></tr><tr><td>0 for $n > 4$</td></tr></table> <p>and $y(n) = 0.8*x(n) + -2.0*x(n-1) + 10.0*x(n-2)$ you can find $y(n)$ with: FIR(GSER(1.0, 2.0, 1.0, 1.0), GSER(0.8, -2.0, 10.0))</p> <p>The resulting series contains the values: 0.8, -0.4, 6.8, 18.8</p> <p>To evaluate the same equation for 20 values of $x(n)$, try: FIR(EXTRACT(GSER(1.0, 2.0, 1.0, 1.0), 1, 20), GSER(0.8, -2.0, 10.0))</p> <p>If you add the initial conditions that $x(-1) = .5$ and $x(-2) = -0.2$ to the original equation, then you have: FIR(GSER(1.0, 2.0, 1.0, 1.0), GSER(0.8, -2.0, 10.0), GSER(0.5, -0.2))</p> <p>then the resulting series contains the values: -2.0, 4.6, 6.8, 18.8</p>	1 for $n = 1$	2 for $n = 2$	1 for $n = 3$	1 for $n = 3$	0 for $n > 4$
1 for $n = 1$						
2 for $n = 2$						
1 for $n = 3$						
1 for $n = 3$						
0 for $n > 4$						
REMARKS:	FIR performs a linear convolution between the input data and FIR coefficients. Unlike CONV, FIR accepts initial conditions and produces an output series containing the same number of samples as the input series.					

FLIPFLOP(onseries, offseries)

- PURPOSE:** Combines two binary series into a flipflop output, where each output point is a function of two input points and immediately prior output point.
- onseries** A binary series that flips the output to “on.”
- offseries** A binary series that flips the output to “off.”
- RETURNS:** A series.
- EXAMPLE:** FLIPFLOP(GSER(1,1,1,0),GSER(0,0,1,0))
returns a series 1,1,0,0
- REMARKS:** When an off and an on signal occur simultaneously, the output state switches. This function is also known as a “dual pad flipflop.” Use this function to implement and evaluate trading strategies where “onseries” and “offseries” are long and short indications, respectively.
- SEE ALSO:** AND OR XOR

FLOCATE(filename)

- PURPOSE:** Locates a file according to MarketBrowser’s path logic.
- filename** Name of file to be found, in quotes.
- RETURNS:** A string with the full path if the file is found; otherwise, returns an empty string.
- EXAMPLE:** FLOCATE(“winsock.dll”)
might return the string: “C:\WINNT\WINSOCK.DLL”

FLOOR(expr)

- PURPOSE:** Finds the greatest integer less than or equal to the input value.
- expr** Any expression evaluating to a scalar, series, table, integer, or real or complex number.
- RETURNS:** A scalar, series, table or number.
- EXAMPLES:** FLOOR(-3.4)
returns the scalar value -4.
- FLOOR(2.2 + 7.8I)
yields a value of 2.0 + 7.0i.
- FLOOR(W2)
creates a new series in the current window by applying FLOOR to each element of W2. The integer value returned by FLOOR is converted to a floating point value.
- SEE ALSO:** CEILING

FMAX(series)

PURPOSE: Places the cursor on the maximum value of the series in the current window.

series (Optional). Series in which to find maximum. Defaults to the current window.

EXAMPLE: If GSIN(100, 0.01) is in window 1, then:

FMAX

places the cursor on the 26th point of that sine wave where $y = 1.0$.

REMARKS: If there is more than one peak of the same height, FMAX finds the first.

SEE ALSO:

FPEAK	FPEAKN
FPEAKP	FVALL
FVALLN	FVALLP
FMIN	MAX

FMIN(series)

PURPOSE: Places the cursor on the minimum value of the series in the current window.

series (Optional). A series in which to find minimum. Defaults to the current window.

RETURNS: Nothing.

EXAMPLE: If GSIN(100, 0.01) is in window 1, then:

FMIN

places the cursor on the 76th point (where the y-value is -1.0).

REMARKS: If there is more than one valley of the same depth, FMIN will find the first one.

SEE ALSO:

FPEAK	FPEAKN
FPEAKP	FVALL
FVALLN	FVALLP
FMAX	MIN

FOCUS(window, OverlayNumber)

- PURPOSE:** Sets the input focus in an OVERLAY window, telling MarketBrowser which series should respond to any graphical manipulations.
- window** (Optional). A window reference. The default is the current window.
- Overlay Number** (Optional). Integer designating the series to "focus" on. The default is the first series in the current window.
- RETURNS:** Nothing.
- EXAMPLE:** FOCUS(2)
Assuming that the current window has an OVERLAY, this example causes the second series in the window to respond to visual manipulations such as scrolling and zooming, while leaving all other series in the window unchanged. These manipulations may be from commands, mouse interaction, or arrow keys.
- REMARKS:** Other series in the window may respond as well, depending on the SYNC setting.
- SEE ALSO:** OVERLAY
SYNC
SCALES

FOPEN(filename, mode)

PURPOSE:	Opens a file in a specified mode.
filename	The name of the file to open, in quotes.
mode	The file mode, in quotes. Valid options are: <ul style="list-style-type: none">• r - Open the file for read access. If the file does not exist, FOPEN fails.• w - Open the file for write access. If the file exists, the old contents are lost; if the file does not exist, FOPEN creates it.• a - Open the file for appending. If the file does not exist, it is created.• r+ - Open the file for read and write access. If the file exists, its old contents are lost; if it does not exist, FOPEN fails.• w+ - Open the file for read and write access. If the file exists, its old contents are lost; if it does not exist, it is created.• a+ - Open the file for read and append access; if the file does not exist, it is created.
RETURNS:	1 if the file is successfully opened; otherwise it returns 0.
EXAMPLE:	FOPEN("header.hdr", "w+") opens the file "header.hdr" for read and write access, and displays a 1 at the bottom of the screen if it is successfully opened.
REMARKS:	When using FOPEN, it is recommend that you use the FCLOSE or FCLOSEALL functions to close any files before exiting from MarketBrowser.
SEE ALSO:	FCLOSE FCLOSEALL FFLUSH FGETS FPUTS FSEEK FTELL FREADA FREADB FWRITEA FWRITEB

FPEAK(series, threshold, width)

PURPOSE:	Sets the cursor to the first series peak above a specified threshold.
threshold	A real number acting as the minimum above which the first peak will be found.
width	(Optional). An integer specifying the minimum number of points above the threshold. The default is 1.
RETURNS:	Nothing.
REMARKS:	FPEAK sets the cursor position to the maximum point of the peak but does not display point values nor does it provide a movable cursor. Use CURSORON to activate cursor.

FPEAKN(series, threshold, width)

PURPOSE:	Sets the cursor to the next peak above a specified threshold.
threshold	(Optional). A real number minimum above which peaks will be found. The default is last threshold value used.
width	(Optional). An integer minimum width above the threshold. The default is 1.
RETURNS:	Nothing.
REMARKS:	FPEAKN sets the cursor position to the maximum point of the peak but does not display point values nor does it provide a movable cursor. Use CURSORON to activate cursor.
SEE ALSO:	CURSORON FPEAK FPEAKP FVALL FVALLN FVALLP FMAX FMIN

FPEAKP(series, threshold, width)

PURPOSE:	Sets the cursor to the previous peak above a specified threshold.
series	(Optional). A series in which to find maximum. Defaults to current window.
threshold	(Optional). A real number acting as the minimum above which previous peak will be found, subject to width argument. The default is last threshold value used.
width	(Optional). An integer minimum width above threshold. The default is 1.
RETURNS:	Nothing.
REMARKS:	FPEAKP sets the cursor position to the maximum point of the peak but does not display point values nor does it provide an active cursor. Use CURSORON to activate cursor.
SEE ALSO:	CURSORON FPEAK FPEAKN FVALL FVALLN FVALLP FMAX FMIN

FPUTS(string, filename)

PURPOSE: Writes a specified string to a file.

string String to be written to file, in quotes.

filename Name of the file to which the string is to be written, in quotes.

RETURNS: 1 if the write is successful; otherwise it returns 0.

EXAMPLE:
FOPEN("header.txt", "w+")
FPUTS("IBM Daily Data\n", "header.txt")
FPUTS("Close High Low Open\n", "header.txt")
FCLOSE("header.txt")

creates a file header.txt that contains the following information: IBM Daily Data
Close High Low Open. If the file header.txt already exists, its former contents are
overwritten by the information shown above.

REMARKS: Must be used in conjunction with FOPEN and FCLOSE. Strings must end with `\n` in
order to be written properly. Strings may also contain other escape characters that
output non-printing characters. Valid escape characters include:

- `\n` - New line
- `\t` - Tab
- `\v` - Vertical tab
- `\b` - Backspace
- `\r` - Carriage return
- `\f` - Form feed
- `\a` - Bell
- `'` - Single quote
- `"` - Double quote
- `\` - Backslash
- `\ddd` - d is an octal digit ASCII character with corresponding octal ASCII code.

A backslash followed by any other character simply writes the character.

SEE ALSO: FOPEN
FCLOSE
FGETS

FREADA(filename, col)

PURPOSE: Reads an ASCII file and loads it directly into the current window.

filename The name of the file to read in, in quotes.

col (Optional). Column number. Defaults to the first column.

RETURNS: A scalar, series, or table.

EXAMPLE: FOPEN("ibm.chl", "r")
FREADA("ibm.chl", 2)
reads in the contents of the file "ibm.chl", starting at the second column.

REMARKS: FREADA is analogous to the READA function.

SEE ALSO: FOPEN
FCLOSE
FREADB

FREADB(filename, filetype)

PURPOSE: Reads a binary data file and loads it directly into the current window.

filename The name of the file to read in, in quotes.

filetype Binary format type. The file's format can be described either by its name, or by the corresponding code, as described below:

Name	Code	Data Type	Range
SBYTE	1	Signed Byte	-128 to +127
UBYTE	2	Unsigned Byte	0 to 255
BYTE	2	(same as UBYTE)	0 to 255
SINT	3	Signed Integer	-32768 to +32768
UINT	4	Unsigned Integer	0 to 65536
LONG	5	4-byte Signed Integer	-2,147,483,648 to +2,147,483,647
FLOAT	6	4-byte Floating Point	-10^{37} to $+10^{38}$ -10^{-37} to $+10^{-38}$
DOUBLE	7	8-byte Floating Point	-10^{307} to $+10^{308}$ -10^{-307} to $+10^{-308}$

RETURNS: A scalar, series, or table.

EXAMPLE: FOPEN("ibm.chl", "r")
FREADB("ibm.chl", SBYTE)

reads a file of signed integer CHLO data.

REMARKS: FREADB is analogous to the READB function.

SEE ALSO: FOPEN
FCLOSE
FREADA
FSEEK

FREEZE(window, flag)

PURPOSE: In a real-time window, turns automatic scaling and redrawing of x- and y-axes on or off.

window (Optional). Window reference. Defaults to the current window.

flag (Optional). Whether to freeze the window's scrolling (1), or have it revert to normal scrolling behavior (0)

RETURNS: Nothing.

REMARKS: If no arguments are present, FREEZE toggles the current window's setting.

SEE ALSO: PLOTMODE
SCALESON
SCALESOFF

FSEEK(filename, offset, origin)

PURPOSE: Moves the file pointer to the location indicated by the specified origin and modified by the specified offset.

filename The name of the file in which to move the pointer, in quotes.

offset Offset from origin, in bytes.

origin Byte location from which to calculate the offset

RETURNS: 1 if the pointer move is successful; otherwise nothing.

EXAMPLES: If the file "header.txt" contains the following text:
IBM Daily Data
Close High Low Open
then issuing the commands:
FOPEN("header.txt", "r+")
FSEEK("header.txt", 18, 0)
FGETS("header.txt")
displays the series "Close High Low Open" at the bottom of the screen.
FSEEK("header.txt", 0, 2)
moves the pointer to the end of the file.

REMARKS: You must use FOPEN to open a file before you can use FSEEK.

SEE ALSO: FOPEN
FCLOSE
FREADB
FTELL

FTELL(filename)

- PURPOSE:** Returns the current byte location of the file pointer, as measured from the beginning of the file.
- filename** The name of the file in which to locate the pointer, in quotes.
- RETURNS:** The location of the file pointer, or nothing if the search is unsuccessful.
- EXAMPLE:** FTELL("header.txt")
displays the location of the file pointer in the status line at the bottom of the screen.
- REMARKS:** To use the FTELL function, the file must be opened with the FOPEN function.
- SEE ALSO:** FOPEN
FCLOSE
FSEEK

FUNCTIONS

- PURPOSE:** Displays a list of all the functions currently defined the MarketBrowser worksheet.
- RETURNS:** Nothing; screen display only.
- SEE ALSO:** VARS

FVALL(series, threshold, width)

- PURPOSE:** Sets the cursor to the first series valley below a specified threshold.
- series** (Optional). A series in which to find minimum. Defaults to the current window.
- threshold** A real number maximum below which the first valley will be found.
- width** (Optional). An integer minimum width below threshold. The default is 1.
- RETURNS:** Nothing.
- REMARKS:** FVALL sets the cursor position to the minimum point of the valley but does not display point values nor does it provide a moveable cursor. Use CURSORON to activate cursor.
- SEE ALSO:** CURSORON
FPEAKN
FPEAKP
FPEAK
FVALLNFVALLP
FMAX
FMIN
MIN

FVALLN(series, threshold, width)

- PURPOSE:** Sets the cursor to the next valley below a specified threshold.
- series** (Optional). A series in which to find minimum. Defaults to current window.
- threshold** (Optional). A real number below which next valley will be found. The default is last threshold value used.
- width** (Optional). An integer minimum width below threshold. The default is 1.
- RETURNS:** Nothing.
- REMARKS:** FVALLN sets the cursor position to the minimum point of the valley but does not display point values nor does it provide a movable cursor. Use CURSORON to activate cursor.
- SEE ALSO:** CURSORON
FPEAK
FPEAKP
FVALL
FPEAKN
FVALLP
FMAX
FMIN

FVALLP(series, threshold, width)

- PURPOSE:** Sets the cursor to the previous valley below a specified threshold.
- series** (Optional). A series in which to find the minimum. Defaults to current window.
- threshold** (Optional). A real number acting as a maximum below which the previous valley will be found, subject to width argument.
- width** (Optional). An integer minimum width below threshold. The default is 1.
- RETURNS:** Nothing.
- REMARKS:** FVALLP sets the cursor position to the minimum point of the valley but does not display point values nor does it provide a movable cursor. Use CURSORON to activate cursor.
- SEE ALSO:** CURSORON
FPEAKN
FPEAK
FPEAKP
FVALL
FVALLNFMAX
FMIN

FWRITEA(filename, col)

- PURPOSE:** Writes a series as an ASCII file directly from the worksheet.
- filename** The name of the file, in quotes.
- col** (Optional). Column number from which to record. Defaults to the first column.
- RETURNS:** 1 if the write is successful, otherwise nothing.
- EXAMPLE:** FWRITEA("myfile.dat")
writes the data in the current window to an ASCII file name "myfile.dat".
- REMARKS:** After using FOPEN, FWRITEA is analogous to the WRITEA function.
- SEE ALSO:** FOPEN
FCLOSE
FWRITEB

FWRITEB(filename, filetype)

PURPOSE: Writes the series in the current window as a binary file.

filename The name of the file, in quotes.

filetype Binary format type. The file's format can be described either by it's name, or by the corresponding code, as described below:

Name	Code	Data Type	Range
SBYTE	1	Signed Byte	-128 to +127
UBYTE	2	Unsigned Byte	0 to 255
BYTE	2	(same as UBYTE)	0 to 255
SINT	3	Signed Integer	-32768 to +32768
UINT	4	Unsigned Integer	0 to 65536
LONG	5	4-byte Signed Integer	-2,147,483,648 to +2,147,483,647
FLOAT	6	4-byte Floating Point	-10^{37} to $+10^{38}$ -10^{-37} to $+10^{-38}$
DOUBLE	7	8-byte Floating Point	-10^{307} to $+10^{308}$ -10^{-307} to $+10^{-308}$

RETURNS: 1 if the write is successful, otherwise nothing.

EXAMPLE: `FWRITEB("myfile.dat", 1)`
writes the data in the current window to a binary file named "myfile.dat".

REMARKS: After using FOPEN, FWRITEB is analogous to the WRITEA function.

SEE ALSO: FOPEN
FCLOSE
FWRITEA

GAMM(expr)

- PURPOSE:** Executes the gamma function; a generalization of factorial for the domain of real numbers.
- expr** A series, table or number.
- RETURNS:** A series, table or number.
- EXAMPLE:** GAMM(GSER(1, 2, 3, 4))
returns series 1, 1, 2, 6. The gamma of n is the factorial of (n-1).
- REMARKS:** In practice, it is often advisable to use the natural log version of GAMM, GAMMLN, because GAMM will exceed the maximum floating point representation, for arguments greater than about 171.62.
- SEE ALSO:** GAMMA
GAMMLN

GAMMA

- PURPOSE:** (A macro). A numeric constant.
- RETURNS:** A number.
- EXPAN-
SION:** 0.57721566490153286061
- SEE ALSO:** LN(expr) PI
DEG E
PHI SETDEGREE
GAMM GAMMLN

GAMMLN(expr)

- PURPOSE:** Calculates the natural log of the gamma function. GAMMLN is often used in place of GAMM when the value of GAMM becomes very large.
- expr** A series, table or number.
- RETURNS:** A series, table or number.
- EXAMPLE:** GAMMLN(GSER(1, 2, 3))
returns a series; the natural log of GAMM.
- REMARKS:** In practice, it is often advisable to recast equations to use GANMLN instead of GAMM, as the GAMM value of arguments greater than 171.62 exceeds the maximum floating point representation.
- SEE ALSO:** GAMM

GAMMA

GAMMP(realnum, expr)

PURPOSE:	Calculates the incomplete gamma function of a series, table or number.
realnum	A real number.
expr	A real series, table or number.
RETURNS:	A series, table or number.
EXAMPLE:	$GAMMP(a, x) = 1 - GAMMQ(a, x)$
SEE ALSO:	GAMMQ

GAMMQ(realnum, expr)

PURPOSE:	Calculates the incomplete gamma function of a series, table or number.
realnum	A real number.
expr	A real series, table or number.
RETURNS:	A series, table or number.
EXAMPLE:	$GAMMQ(a, x) = 1 - GAMMP(a, x)$
SEE ALSO:	GAMMP

GENSTUDY(window, number, param1, param2, param3, string)

PURPOSE:	Generates a study on a window or variable.																																				
window	The window or variable from which to generate a study. Studies usually require that source data contain either trading bars or candlesticks.																																				
number	(Optional). Integer. The number of the study you wish to generate. Defaults to 0. Options are: <table><tr><td>0</td><td>'PERK'</td><td>Calculate %K series of a standard stochastic: $((cl-mn)/(mx-mn)) * 100$</td></tr><tr><td>1</td><td>'WILR'</td><td>Calculate Williams' %R: $((mx-cl)/(mn-mx)) * 100$</td></tr><tr><td>2</td><td>'ADOS'</td><td>Accumulation/Distribution Oscillator: $((cl-lo)-(hi-cl))/((hi-lo)*n) * 100$</td></tr><tr><td>3</td><td>'HLOS'</td><td>Hi-Lo Oscillator: $((hi-pcl)/\max((hi-pcl),(hi-lo),(pcl-lo))) * 100$</td></tr><tr><td>4</td><td>'MEDP'</td><td>Median Price (if n = 2): $(hi+lo)/n$</td></tr><tr><td>5</td><td>'HTPH'</td><td>(hi-phi)</td></tr><tr><td>6</td><td>'HTPC'</td><td>(hi-pcl)</td></tr><tr><td>7</td><td>'PHTC'</td><td>(phi-cl)</td></tr><tr><td>8</td><td>'LTPL'</td><td>(lo-plo)</td></tr><tr><td>9</td><td>'LTPC'</td><td>(lo-pcl)</td></tr><tr><td>10</td><td>'PLTC'</td><td>(plo-cl)</td></tr><tr><td>11</td><td>'PCTL'</td><td>(pcl-lo)</td></tr></table>	0	'PERK'	Calculate %K series of a standard stochastic: $((cl-mn)/(mx-mn)) * 100$	1	'WILR'	Calculate Williams' %R: $((mx-cl)/(mn-mx)) * 100$	2	'ADOS'	Accumulation/Distribution Oscillator: $((cl-lo)-(hi-cl))/((hi-lo)*n) * 100$	3	'HLOS'	Hi-Lo Oscillator: $((hi-pcl)/\max((hi-pcl),(hi-lo),(pcl-lo))) * 100$	4	'MEDP'	Median Price (if n = 2): $(hi+lo)/n$	5	'HTPH'	(hi-phi)	6	'HTPC'	(hi-pcl)	7	'PHTC'	(phi-cl)	8	'LTPL'	(lo-plo)	9	'LTPC'	(lo-pcl)	10	'PLTC'	(plo-cl)	11	'PCTL'	(pcl-lo)
0	'PERK'	Calculate %K series of a standard stochastic: $((cl-mn)/(mx-mn)) * 100$																																			
1	'WILR'	Calculate Williams' %R: $((mx-cl)/(mn-mx)) * 100$																																			
2	'ADOS'	Accumulation/Distribution Oscillator: $((cl-lo)-(hi-cl))/((hi-lo)*n) * 100$																																			
3	'HLOS'	Hi-Lo Oscillator: $((hi-pcl)/\max((hi-pcl),(hi-lo),(pcl-lo))) * 100$																																			
4	'MEDP'	Median Price (if n = 2): $(hi+lo)/n$																																			
5	'HTPH'	(hi-phi)																																			
6	'HTPC'	(hi-pcl)																																			
7	'PHTC'	(phi-cl)																																			
8	'LTPL'	(lo-plo)																																			
9	'LTPC'	(lo-pcl)																																			
10	'PLTC'	(plo-cl)																																			
11	'PCTL'	(pcl-lo)																																			

12 'PHTH' (phi-hi)
 13 'CTPC' (cl-pcl)
 14 'PCTC' (pcl-cl)
 15 'HTOL' (hi-lo)
 16 'PHPL' (phi-plo)
 17 'CTPCDPC' (cl - pcln)/pcln
 18 'TRUR' True Range: max (hi-pcl, hi-lo, pcl-lo)
 19 'TYPP' Typical Price: (hi + lo + cl) / 3
 20 'PERM' Percent Movement: ((hi-((phi+plo)/2))/((hi+lo)/2)) * 100
 21 'WGTC' Weighted Close Indicator: ((cl * 2) + hi + lo) / 4
 22 'SWIN' Swing Index: (see ATA manual)
 23 'KSWI' K Swing Index: (see ATA manual)
 24 'RSWI' R Swing Index: (see ATA manual)
 25 'MFIP' Positive Money Flow: if (phi + plo + pcl)/3 <= (hi + lo + cl)/3,
 then (hi + lo + cl)/3, else 0.
 26 'MFIN' Negative Money Flow: if (phi + plo + pcl)/3 > (hi + lo + cl)/3,
 then (hi + lo + cl)/3, else 0

param1 (Optional). Integer. First parameter of the study. Defaults to 1.
param2 (Optional). Integer. Second parameter of the study. Defaults to 1.
param3 (Optional). Integer. Third parameter of the study. Defaults to 1.
string (Optional). Quoted string indicating study name to perform. If provided, supersedes **number**.

EXAMPLE: W2:GENSTUDY(W1,1,'PERK')
 generates %K, not Williams' %R, as **string** supersedes **number**.

REMARKS: This function definition uses the following shorthand notation:

- cl - close values of trading bars
- hi - high values of trading bars
- lo - low values of trading bars
- op - open values of previous trading bars
- pop - open values of previous trading bar
- pcl - close values of previous trading bar
- phi - high values of previous trading bar
- plo - low values of previous trading bar
- pcln - close value of trading bar 'n' periods before
- mx - n-point moving maximum value (where 'n' is parameter1-3)
- mn - n-point moving minimum value (where 'n' is parameter1-3)
- n - parameter 1
- m - parameter 2
- k - parameter 3
- max(a,b,c) - maximum value of a, b, and c

GENSTUDY creates various simple studies on the data in a window or variable. It is primarily designed for internal use, but MarketBrowser users may find it helpful.

All studies GENSTUDY creates can be recreated using XPL, macros, or DLLs, but

GENSTUDY calculates these studies at a lower level much faster.

GETCOMMENT(window)

PURPOSE: Returns the comment string for the first series in a window.

window (Optional). Window reference. Defaults to the current window.

RETURNS: A string.

SEE ALSO: COMMENT
GETSCCOMMENT
SETCOMMENT
LEGEND
LEGCUR

GETCONF(item)

PURPOSE: (A Macro). Returns the current value of a configuration variable.

item A string enclosed in quotes that represents a configuration variable.

RETURNS: A string representing the current value of the requested configuration variable.

EXAMPLE: GETCONF("PLOT_STYLE")
returns the current value of the plot style configuration variable.

SEE ALSO: SETCONF

GETDATAREF(window, which_one)

PURPOSE: Returns the name of the first variable in a window which contains a series.

window (Optional). Window reference. Defaults to the current window.

which_one (Optional). Which unique data reference (origin 1) to return. Default to 1.

RETURNS: A string if successful, otherwise nothing.

EXAMPLE: =A = GRANDOM(200,2)
W1: =A;OVERPLOT(MOVAVG(A,20))
W2: =GETDATAREF(W1)
returns the string "A" in the status line.

SEE ALSO: COPYWIN
GETVARIABLE
GETVFORM

GETDATAVARNAME()

- PURPOSE:** Returns the name of the variable currently being assigned.
- RETURNS:** The name of the variable as a string.
- REMARKS:** This function is used by the data cache applications layer in MarketBrowser to set attributes of a variable's data as the variable is being assigned.

GETDATE, GETTIME(series)

- PURPOSE:** Return the system or series date or time.
- series** (Optional). A series or table; if not supplied, the system date/time is returned.
- RETURNS:** A string containing a date or time.
- SEE ALSO:** SETDATE, SETTIME
DEFDATE
DEFTIME

GETDTFORMAT(series, style)

- PURPOSE:** Returns the format of the date and time in the window.
- series** (Optional). String. A window or series reference, in quotes.
- style** Integer, where specifying 0 returns to short format of window, and specifying 1 returns the long format.
- RETURNS:** An integer representing the date/time format, as defined in the INDEXTODT function.
- EXAMPLE:** Given the formula:
W1: SETDTFORMAT(0, 8)
W2: GETDTFORMAT("W1", 0)
returns the integer 8, which corresponds to the date/time format hh:mm:ss (e.g., 11:59:59).
- REMARKS:** The short format refers to the manner in which dates and times are displayed along a window's time (x) axis, while the long format refers to dates and times displayed in cursor views, in status messages at the bottom of the screen, etc.
- SEE ALSO:** INDEXTODT
SETDTFORMAT

GETENV(env_var)

PURPOSE: Returns an environment variable string.

env_var The name of the environment variable, in quotes.

RETURNS: A string. May be of zero length if variable is not defined.

SEE ALSO: PUTENV
GETPATH
GETWORKSHEET
RUN

GETFOCUS(window)

PURPOSE: Returns an integer corresponding to the number of the curve in focus in the specified window.

window Window reference.

RETURNS: An integer.

EXAMPLE: W1: GSIN(100,.01);OVERLAY(GRAND(10,5),LRED)
W1: FOCUS(2);GETFOCUS(W1)
returns the value 2; the light red series of random noise is in focus.

REMARKS: GETFOCUS is useful when maneuvering in windows with many overlaid series.

SEE ALSO: OVERLAY
FOCUS
SCALES

GETGCOLOR(color_param)

PURPOSE: Returns the global MarketBrowser color as listed in the file dspcolor.

color_param Integer. Corresponds to the color-related parameters in the file dspcolor.

EXAMPLE: GETGCOLOR(2)
returns the color corresponding to the 2nd parameter (background color) in the file dspcolor.

SEE ALSO: dspcolor file
palette.mac file
SETGCOLOR

GETHIGHWATER(series)

PURPOSE: Returns the high-water mark of a series, which is the number of points in real-time dependent series that MarketBrowser uses to update the series at the end of a real-time interval.

series (Optional). Quoted string. A valid window or variable reference (variables must contain a series). Defaults to the current window.

RETURNS: Integer.

EXAMPLES: Given the following window formulas:

```
W1: BARMON("ABC")
```

```
W2: MOVAVG(W1,10)
```

```
then
```

```
GETHIGHWATER("W2")
```

```
returns 21
```

```
GETHIGHWATER("W1") returns 1.
```

REMARKS: The high-water mark is the number of points in real-time dependent series that MarketBrowser uses to update the series at the end of a real-time interval.

SEE ALSO: SETHIGHWATER

GETHUNITS(series)

PURPOSE: Returns the horizontal units of a series.

series A series or table, if not supplied. Defaults to the first series in the window.

RETURNS: A string or table.

SEE ALSO: GETVUNITS DEFHUNITS
 SETHUNITS SETVHUNIT
 GETXLABEL

GETITEMCOUNT(data)

PURPOSE: Returns the "item count" of the data. For the first series of a compound item, the count is the number of members of the item. For any subsidiary item members, the count is 1.

data (Optional.) A series or matrix reference (defaults to the current window).

GETITEMTYPE(data)

PURPOSE: Returns the “item type” of the data.

data (Optional.) A series of matrix reference (defaults to the current window).

RETURNS: An integer indicating the item type:
0: simple series
1: XY data
2: matrix
3: CHLO
5: equivolume

GETLABEL(window)

PURPOSE: Returns the label of the specified window.

window (Optional). A window reference. Defaults to the current window.

RETURNS: A string.

SEE ALSO: LABEL

GETLOCALVARIABLE(name)

PURPOSE: Gets the value of a local variable.

name The name of the local variable, optionally in quotes.

RETURNS: The assigned value of the local variable.

EXAMPLE: GETLOCALVARIABLE(“localvar”)
returns the value of the local variable “localvar”.

SEE ALSO: GETLOCALVAR (shortcut name)
SETLOCALVARIABLE
GETVARIABLE
SETVARIABLE

GETMACRO(macro_name, form)

PURPOSE: Returns information about the make-up of a macro.

macro_name A string. The name of the macro.

form (Optional). An integer representing the form in which you want the macro displayed; defaults to 0.

- 0 = body
- 1 = name
- 2 = arguments
- 3 = name + arguments
- 4 = name + body + arguments

RETURNS: A string.

EXAMPLES: GETMACRO("SLICE",0)

returns the string Col(transpose(m),n), which is the body of the macro.

GETMACRO("SLICE", 2)

returns the string (M,N) or the arguments.

SEE ALSO: DEFMACRO
MACWRITE
MACREAD

GETPATH

PURPOSE: Returns the current working directory path.

RETURNS: A string.

SEE ALSO: GETENV
GETWORKSHEET

GETPEAK(series, threshold, width, size, padmode, fillval)

PURPOSE:	Finds the peaks of a series and places the values in the resulting series.
series	A series or table from which to get the peaks.
threshold	Specifies the maximum values above which peaks will be accepted. The default is the minimum of series.
width	An integer that specifies the maximum number of points that comprise a peak.
size	Specifies the minimum acceptable peak to valley height of a peak. The default is 0.0
padmode	Determines whether the values between the peaks in the resulting series will be padded. 0 = do not pad, 2 = linear interpolation. The default is 1.
fillval	The padding value to use when padmode is on. The default = 0.

EXAMPLES: GETPEAK(W1)
finds all the peaks of window 1.

GETPEAK(W1,MIN(W1),1,0.0,1,0.0)
does the same thing. Note that because the default fillval is 0.0, it is possible that peaks and valleys of height 0.0 will be indistinguishable from the fillval. If this is a problem, set the fillval to be the minimum of the input series for GETPEAK and the maximum value of the input series for GETVALLEY.

For example, to uniquely find all the peaks of a series, type:

```
GETPEAK(W1, MIN(W1),1,0.0,1,MIN(W1))
```

REMARKS: The default values for the optional arguments suffice for most applications.

SEE ALSO: GETVALLEY
FPEAK
FVALL

GETPT(series, number)

PURPOSE:	Returns the value of the nth point of a series in any window.
series	(Optional). A series or table. Defaults to the current window.
number	Integer point in the series or table.
RETURNS:	A complex number.
EXAMPLE:	REAL(GETPT(LENGTH)) returns the real number which is the last value in the current window.
REMARKS:	If the value of the nth point is an NA value, then GETPT returns the value of the first preceding non-NA value.
SEE ALSO:	SETPT GETRAWPT

GETRAWPT(series, number)

PURPOSE:	Returns the value of the nth point of a series in any window, with equal consideration for NA values.
series	(Optional). A series or table. Defaults to the current window.
number	Integer. An index to a point in the series or table.
RETURNS:	The value of the requested point.
EXAMPLES:	Given the formula: W1: GSER(3,4,navalue) W2: GETRAWPT(w1,2) returns the integer 4, which is the value of the second point in W1. GETRAWPT(W1, LENGTH(W1)) returns the value NA. Compare this behavior to that of the GETPT function, which would return the value of the point preceding the NA value.
REMARKS:	GETRAWPT is analogous to the GETPT function, except when the value of the requested point is NA. In this case, it returns NA. GETPT, on the other hand, returns the value of the point preceding the NA value.
SEE ALSO:	GETPT

GETSCALES(window)

- PURPOSE:** Returns an integer corresponding to the scales used by the current focus of the specified window.
- window** (Optional). Window reference. Defaults to current window.
- RETURNS:** An integer.
- EXAMPLE:** W1: GSIN(100,.01);SCALES(16)
GETSCALES(W1)
returns the value 16; SCALES(16) yields x-axis on top with labels.
- REMARKS:** GETSCALES applies to the current focus of the specified window.
- SEE ALSO:** SCALES
OVERLAY
FOCUS

GETSCOMMENT(series)

- PURPOSE:** Returns the comment string for a window series or a variable.
- series** (Optional). Series or window reference. Defaults to the current window.
- RETURNS:** A string.
- REMARKS:** This function is analogous to the GETCOMMENT function, except that it returns comments for both windows and variables.
- SEE ALSO:** GETCOMMENT COMMENT
SETCOMMENT LEGEND
LEGCUR

GETSERIES(table, n)

- PURPOSE:** Return a single series from a table or matrix.
- table** Any compound data item; table, matrix, trading bars, candlesticks, etc.
- n** An integer. The number of the column to return.
- RETURNS:** A series.
- EXAMPLE:** GETSERIES(READTABLE("data"), 7)
returns the seventh column of data from the file "data".
- REMARKS:** Synonymous with the "COL" macro, which returns a column of data.

GETSTR(titlebar, prompt, defaultvalue)

PURPOSE:	Prompts the user for input strings using the native GUI.
titlebar	String enclosed in quotes.
prompt	String enclosed in quotes.
defaultvalue	String enclosed in quotes.
RETURNS:	The string that was typed into the entry field.
EXAMPLE:	GETSTR("Add Windows", "Number of Windows to Add:")
REMARKS:	The type of prompt box that appears is system dependent.
SEE ALSO:	MESSAGE PICKLIST PICKFILE

GETSYMBOL(series)

PURPOSE:	Returns an integer value which represents the symbol style for a given series.
series	(Optional). Window or series argument. Defaults to the current window.
RETURNS:	An integer.
EXAMPLE:	Given the formula: W1: GSER(1,2,3);SETSYMBOL(12) W2: GETSYMBOL(W1) returns the value 12.
SEE ALSO:	SETSYMBOL

GETTOOLBAR(attribute, toolbar, button)

- PURPOSE:** This function returns a string describing some attribute of the toolbar buttons. Depending on the attribute requested, the toolbar and button arguments may not be required and are thus optional. If they are relevant and not supplied, they are defaulted to 1.
- attribute** This argument can be any of:
- 1 current toolbar number
 - 2 max number of toolbars
 - 3 next free toolbar
 - 4 max number of buttons for this toolbar
 - 5 next free button for this toolbar
 - 6 OR of methods for this button
 - 7 color1 for this button
 - 8 color2 for this button
 - 9 keycode action for this button
 - 10 label for this button
 - 11 predefined command for this button
 - 12 user defined command for this button
 - 13 help/tip for this button
- toolbar** (Optional.) The toolbar that contains the button. Valid choices are: 1 - main worksheet toolbar; 2 - activated window toolbar; 3 - data cursor toolbar
- button** (Optional.) The button's location on the toolbar, counted from left, starting from 1.
- RETURNS:** A string. Note that some attributes of a button are integers and others are strings. This function returns its answers as strings always, so some answers will need to be explicitly cast to integers to be used as inputs to other functions.
- REMARKS:** The manifest constant string "*EOL*" is used to distinguish between a nonexistent string and a string of length zero.
- SEE ALSO:** TOOLBAR

GETVALLEY(series, threshold, width, size, padmode, fillval)

PURPOSE:	Finds the valleys of a series and places the values in the resulting series.
series	A series or table from which to get the valleys.
threshold	Specifies the maximum value below which valleys will be accepted. The default is the max of series.
width	An integer that specifies the maximum number of points that comprise a valley. The default is 1.
size	Specifies the minimum acceptable peak to valley height of a valley. The default is 0.0.
padmode	Determines whether the values between the valleys in the resulting series will be padded. 0 = do not pad, 2 = linear interpolation. The default is 1.
fillval	The padding value to use when padmode is on The default is 0.

EXAMPLES: GETVALLEY(W1)
finds all the valleys of window 1.

GETVALLEY(W1, MAX(W1), 1, 0.0, 1, 0.0)
does the same thing. Note that because the default fillval is 0.0, it is possible that peaks and valleys of height 0.0 will be indistinguishable from the fillval. If this is a problem, set the fillval to be the minimum of the input series for GETPEAK and the maximum value of the input series for GETVALLEY.

For example, to uniquely find all the valleys of a series, type:
GETVALLEY(W1, MAX(W1), 1, 0.0, 1, MAX(W1))

REMARKS: The default values for the optional arguments suffice for most applications.

SEE ALSO: GETPEAK
FPEAK
FVALL

GETVARIABLE(name)

- PURPOSE:** Gets the current value of a global variable
- name** The variable's name, optionally in quotes.
- RETURNS:** The current value of the named global variable.
- EXAMPLE:** GETVARIABLE("myvar")
returns the current value of the named global variable.
- SEE ALSO:** GETVAR (shortcut name)
SETVARIABLE
GETLOCALVARIABLE

GETVFORM(series)

- PURPOSE:** Returns a window or variable's formula as a string.
- series** (Optional). A window or variable reference. Defaults to the current window.
- RETURNS:** A string.
- REMARKS:** GETVFORM is analogous to the GETWFORMULA function, except that it can take a valid variable reference as an argument as well as a window reference.
- SEE ALSO:** GETWFORMULA
ADDWFORM
SETWFORM
GETWNUM

GETVNUMCOLS(varname)

- PURPOSE:** To count the number of columns in a data variable referenced by name.
- varname** The name of the variable of a series or matrix in which to count columns.
- RETURNS:** This function always succeeds, and gives the answer 0 if it cannot locate the variable or the variable does not contain a series or matrix.
- REMARKS:** GETVNUMCOLS("myvar") gives the same answer as NUMCOLS(myvar) but is more efficient.
- SEE ALSO** NUMCOLS

GETVUNITS(series)

- PURPOSE:** Returns the vertical units of a data series.
- series** (Optional). A series or table. Defaults to the current window.
- RETURNS:** A string.
- EXAMPLE:** GETVUNITS in a window with the formula GSER(1,2,3) returns "No Units".
- SEE ALSO:** SETVUNITS
SETVVUNIT
GETHUNITS
COMMENT
GETCOMMENT

GETWCOLOR(window, series_num)

- PURPOSE:** Returns the color of a window or its data series.
- window** (Optional). A window reference. Defaults to the current window.
- series_num** (Optional). A series number specifier. Defaults to the current window color.
- RETURNS:** An integer.
- EXAMPLES:** GETWCOLOR()
returns the window color of a current window.
GETWCOLOR(W4, 2)
returns the color of the second series in window 4.
- SEE ALSO:** WINCOLOR
SETCOLOR

GETWCOUNT(window)

- PURPOSE:** Returns the count of the number of a series in a window.
- window** (Optional). A window reference. Defaults to the current window.
- RETURNS:** An integer.
- SEE ALSO:** SERCOUNT

GETWFORMULA(series)

- PURPOSE:** Returns the formula for a variable or window in string form.
- window** (Optional). A window or variable reference. Defaults to the current window.
- RETURNS:** A string.
- SEE ALSO:** ADDWFORM
SETWFORM
GETWNUM

GETWINFORMAT(window, which_value)

- PURPOSE:** Gets the decimal or fractional format defined for a given window.
- window** (Optional). Window reference. Defaults to the current window.
- which_value** Integer value, specifying which particular fractional/decimal display parameter you are interested in. Choices are:
- 1 = format_method
 - 2 = reserved
 - 3 = denominator
 - 4 = reduce
 - 5 = trim
 - 6 = reserved
 - 7 = precision
- RETURNS:** The value of the specified format.
- EXAMPLE:** Given the formula:
W1: WINFORMAT(2,-1,16,0,1)
W2: GETWINFORMAT(W1, 3)
returns the integer value 16, which is the value of denominator.
- REMARKS:** Note that GETWINFORMAT returns the actual parameter stored with the window, rather than the effective value of the window. If, for example, the FRAC_DENOMINATOR setting for MarketBrowser is 8 and the window has no private setting for the fractional display denominator, GETWINFORMAT(3) returns -2 ("UNDEFINED") rather than 8.
- SEE ALSO:** WINFORMAT

GETWKSATTRIBUTE(attribute)

PURPOSE: Returns the value of items set with the SETWKSATTRIBUTE function.

attribute String enclosed in quotes. Choose from:

"WKSLOCK"
"READONLY"
"WKSHELP"
"WKSMENU"
"WINMENU"
"ADDREMOVE"
"HIDEDISPLAY"
"TILEARRANGE"
"MOUSERESIZE"
"CURSORINFO"
"OVERRIDE_REFRESH_POLICY"

RETURNS: A string.

REMARKS: "PASSWORD" is the one worksheet attribute that this function will not return a value for.

SEE ALSO: SETWKSATTRIBUTE

GETWNUM

PURPOSE: Returns the number of the current window.

RETURNS: An integer that represents the current window's number.

EXAMPLE: Given W1, which has the formula GSER(1,2,3)
STRCAT("W",STRNUM(GETWNUM),":", GETWFORMULA)
returns a string "W1: GSER(1,2,3)".

SEE ALSO: GETWFORMULA

GETWORKSHEET

PURPOSE: Returns the name of the current worksheet.

RETURNS: A string.

SEE ALSO: GETPATH
GETENV

GETWSNUMROWS

- PURPOSE:** Returns the largest number of rows in the worksheet.
- RETURNS:** An integer.
- REMARKS:** This function returns the largest number of windows that could be intersected by a vertical line drawn anywhere over the worksheet.

GETWSNUMWINDOWS(visible_only)

- PURPOSE:** Returns the number of (visible) windows in the current worksheet.
- visible_only** (Optional.) An integer. When set to 1, only visible windows are counted. If set to 0, all windows are counted, which is equivalent to the function NUMWINDOWS(). Defaults to 0.
- RETURNS:** An integer that represents the number of (visible) windows in the current worksheet.
- EXAMPLE:** Given a worksheet that has 50 windows but all but 9 are hidden:
GETWSNUMWINDOWS(1)
returns the integer 9.
- SEE ALSO:** NUMWINDOWS

GETXL, GETXR, GETYB, GETYT, GETXTIC, GETYTIC(window)

- PURPOSE:** Returns various window drawing parameters.
- window** (Optional). A window reference. Defaults to the current window.
- RETURNS:** A real number.
- REMARKS:**
- GETXL - Leftmost X coordinate
 - GETXR - Rightmost X coordinate
 - GETYB - Bottom Y coordinate
 - GETYT - Top Y coordinate
 - GETXTIC - X Axis tic interval
 - GETYTIC - Y Axis tic interval

GETXLABEL, GETYLABEL(window)

PURPOSE: Returns the x-axis or y-axis label.

window (Optional). Window reference. Defaults to the current window.

RETURNS: A string.

EXAMPLE: GETXLABEL(W3)
returns the x-axis label of window 3.

REMARKS: If no label has been set, GETXLABEL and GETYLABEL return the horizontal (x-axis) units and vertical (y-axis) units, respectively. In both interactive and printed output, an axis label, if set, will cover whichever units label has been defined for the axis.

SEE ALSO: GETHUNITS GETVUNITS
 SETXLABEL SETYLABEL
 CLEARXLABEL CLEARYLABEL

GEXP(points, spacing, factor, offset)

PURPOSE: Generates an exponential curve.

points An integer. The number of points in the series.

spacing Spacing between each point on the x-axis.

factor (Optional). The multiplicative factor used to expand or contract the series along the x-axis. The default is 1.

offset (Optional). The operand used to adjust the x position of the series. The default shift is 0.

EXAMPLE: GEXP(100, 0.1, 2, 5)
Creates an exponential curve shifted 5 points along the x-axis and compressed by a factor of 2. The first half of this series will look like you took every second point from a GEXP(100, 0.1) series.

GHAMMING(points, spacing, alpha)

PURPOSE: Generates a Hamming window.

points Number of points to generate.

spacing Spacing between points.

alpha (Optional). A scaling parameter. The default is 0.54.

RETURNS: A series.

EXAMPLES: GHAMMING(100, .01)

generates a 100-point Hamming window using the following formula:

$$\text{alpha} - (1 - \text{alpha}) * (\cos(2 * \pi * i * \text{spacing}) / (N - 1))$$

where alpha is the default value 0.54, i is some point, and N is 100, the number of points.

GHAMMING(100, .01, .5)

creates a similar window as above, except that alpha is now 0.50 and the formula used is:

$$0.5 * (1 - \cos(2 * \pi * i * \text{spacing}) / (N - 1))$$

A Hamming window with alpha = 0.5 is identical to a Hanning window.

REMARKS: Use the HAMMING macro command to automatically create and multiply a Hamming window with a series. For example: HAMMING(W1) will multiply window 1 with a Hamming window calculated to the same length and spacing as the series in W1.

Hamming, Hanning and Kaiser windows are useful in creating FIR filters and in preprocessing series for FFT calculations.

SEE ALSO: HAMMING
GHANNING
GKAISER
FFT
MACROS
SPECTRUM (macro)
PSD (macro)

GHANNING(points, spacing)

PURPOSE: Generates a Hanning window.

points Number of points to generate.

spacing Spacing between points.

RETURNS: A series.

EXAMPLE: GHANNING(100, .01)

creates a 100-point Hanning window with points spaced with an interval of 0.01 by the following formula:

$$0.5*(1 - \cos(2*\pi*i*spacing)/(N - 1))$$

where N is the number of points to generate.

REMARKS: Use the Hanning macro command to automatically create and multiply a Hanning window with a series. For example:

HANNING(W2)

will multiply window 2 with a Hanning window calculated to the same length and spacing as the series in W2.

SEE ALSO: HANNING
GHAMMING
GKAISER
FFT
MACROS
SPECTRUM (macro)
PSD (macro)

GKAISER(points, spacing, beta)

PURPOSE: Generates a Kaiser window.
points Number of points to generate.
spacing Spacing between points.
beta Optional scaling factor (default is 7.865).

RETURNS: A series.

EXAMPLE: GKAISER(100, .01)

generates a 100-point Kaiser window using the following formula:

$$\frac{I^*(\text{beta} * (((N-1)/2)^2 - (i * \text{spacing}) - (N-1)/2)^{0.5})}{i * (\text{beta} * N - 1) / 2}$$

where I is the modified zeroth order Bessel function. The default beta is 7.865 and N is the number of points to generate.

REMARKS: Use the KAISER macro command to automatically create and multiply a Kaiser window with a series.

For example:

KAISER(W3)

multiplies window 3 with a Kaiser window calculated to the same length and spacing as the series in W3.

Hamming, Hanning and Kaiser windows are useful in creating FIR filters and in preprocessing series for FFT calculations.

SEE ALSO: KAISER
GHAMMING
GHANNING
FFT
MACROS
SPECTRUM (macro)
PSD (macro)

GLINE(points, spacing, slope, y-intercept)

PURPOSE: Generates a line in accordance with the specified parameters.

points Number of points in the series.

spacing Spacing between each point on the x-axis.

slope The slope of the desired line.

y-intercept The point of intersection with the y-axis.

EXAMPLE: GLINE(100, 0.1, 4, 2)

This creates a line in the current window. The line is comprised of 100 points spaced 0.1 x-units apart. The equation of the line will be $y = 4x + 2$.

SEE ALSO: GSER
GRANDOM

GLN, GLOG(points, spacing, slope, intercept)

PURPOSE: Generates a natural logarithmic curve (base e) in accordance with the specified parameters.

points Number of points in the series.

spacing Spacing between each point on the x-axis.

slope (Optional). The slope of line used to create logged series. The default is a factor of 1.

intercept (Optional). The y-intercept of line used to create logged series. The default offset is 0.

EXAMPLES: GLN(100,1,2,5)

This creates a logarithmic series of 100 points spaced 1 unit apart.

E^GLN(100,1,2,5)

yields a straight line with a slope of z and y-intercept of 5.

REMARKS: The formula used to generate each point i in the series is as follows:

$LN(i * \text{spacing} * \text{factor} + \text{offset})$

SEE ALSO: GLOG10

GLOG10(points, spacing, slope, intercept)

- PURPOSE:** Generates a logarithmic curve (base 10) in accordance with the specified parameters.
- points** Number of points in the series.
- spacing** Spacing between each point on the x-axis.
- slope** (Optional). The slope of line used to create logged series. The default is a factor of 1.
- intercept** (Optional). The y-intercept of line used to create logged series. The default is 0.
- EXAMPLES:** GLOG10(100,1,2,5)
This creates a logarithmic series of 100 points spaced 1 unit apart.
10^GLOG10(100,1,2,5)
yields a straight line with a slope of 2 and y-intercept of 5.
- REMARKS:** The formula used to generate each point *i* in the series is as follows:
LOG10(*i**spacing*factor + offset)
- SEE ALSO:** GLN, GLOG

GNORMAL(points, spacing, mean, std)

- PURPOSE:** Creates a normally distributed random series.
- points** Number of points in the series.
- spacing** Spacing between each point on the x-axis.
- mean** (Optional.) Series mean. The default is 0.0.
- std** (Optional.) Series standard deviation. The default is 1.0.
- RETURNS:** A series.
- EXAMPLES:** GNORMAL(100, .01, 2.0, 3.0)
creates a 100 point, normally distributed random series with a mean of 2.0 and a standard deviation of 3.0.
GNORMAL(100, .01)
creates a 100 point random series with mean near 0.0 and standard deviation near 1.0.
- SEE ALSO:** GRANDOM

GOTOWINDOW(window)

- PURPOSE:** Makes a specified window the selected (current) window.
- window** Window reference
- RETURNS:** Nothing.
- EXAMPLE:** GOTOWINDOW(W3)
brings the worksheet cursor to window 3.
- REMARKS:** GOTOWINDOW will not work in an active window. In practice, this means GOTOWINDOW will generally not work in a nonempty window. Unlike MOVETO, GOTOWINDOW can move to hidden windows.
- SEE ALSO:** MOVETO

GRADE(series, order)

- PURPOSE:** Sorts a series in ascending or descending order and yields an output series that contains the indices of the sorted input series.
- series** Input series to grade.
- order** (Optional). Integer. 0 = descending; 1 = ascending. The default is 0.
- RETURNS:** A series.
- EXAMPLE:** GRADE(GSER(4,2,3,1,5))
returns a series containing the positional data series (5,1,3,2,4).
- SEE ALSO:** SORT
REORDER
LOOKUP

GRANDOM(points, spacing, range1, range2)

- PURPOSE:** Generates a random series based on a flat distribution. The optional range arguments lets you determine the output range.
- points** Number of points in series or table.
- spacing** Spacing between each point on the x-axis.
- range1** (Optional). The low end of the range. The default is 0.0
- range2** (Optional). High end of range. The default is 0.0
- RETURNS:** A series or table.
- EXAMPLE:** GRANDOM(100,0.1,1,10)
generates a series of 100 points from 1 to 10 every 0.1 x-units for ten x-units.
- REMARKS:** If only one range is given, range is [0, range] or [range,0], depending on whether range is positive or negative.
- SEE ALSO:** SEEDRAND

GRIDDASH, GRIDDOT, GRIDSOL, GRIDOFF

- PURPOSE:** Specifies that any grids shown in the current window will be dashed, dotted, or solid, respectively.
- RETURNS:** Nothing.
- REMARKS:** These functions set only the line style with which any grids are drawn.
- SEE ALSO:** GRIDH
GRIDV
GRIDHV

GRIDH, GRIDV, GRIDHV

- PURPOSE:** Sets the grid orientation horizontal and/or vertical.
- RETURNS:** Nothing.
- REMARKS:** Does not automatically set the grids to be visible. A visible line style must be set with a related grid function.
- SEE ALSO:** GRIDDASH
GRIDDOT
GRIDSOL
GRIDOFF

GSER(real1, ... , realn)

PURPOSE:	Generates a real series.
real1, ..., realn	A real number.
RETURNS:	A series.
EXAMPLE:	GSER(1, 2, 3, 2, 1) Generates a 5-point curve.
REMARKS:	The deltax if fixed at 1.0.
SEE ALSO:	GLINE

GSQRT(points, spacing, factor, offset)

PURPOSE:	Generates a square root curve based on a positive range of numbers in accordance with the specified parameters.
points	Number of points in the waveform.
spacing	Spacing between each point on the x-axis.
factor	(Optional). An operand to compress or expand the waveform along the x-axis. The default is a factor of 1.
offset	(Optional). An operand to adjust the x-axis position of the waveform. The default shift value is 0.
RETURNS:	A series.
EXAMPLE:	GSQRT(100, 0.1, 2, 5) creates a square root curve of 100 points spaced 0.1 units apart. This waveform will also be shifted by five units on the x-axis, and compressed by a factor of 2.
REMARKS:	The formula use to generate each point i in the waveform is as follows: $(i*\text{spacing}*factor + \text{offset})^{1/2}$

GSQRWAVE(points, spacing, frequency, phase)

PURPOSE:	Generates a square wave in accordance with the specified parameters.
points	Number of points in the waveform.
spacing	Spacing between each point on the x-axis.
frequency	(Optional). An operand, expressed in cycles per unit time, to adjust the frequency of the waveform. The default frequency is 1.
phase	(Optional). An operand to adjust the phase of the waveform, specified in radians. The default is 0.
RETURNS:	A series.
EXAMPLE:	GSQRWAVE(100, 0.1, 2, 5) creates a square wave of 100 points spaced 0.1 units apart. This waveform will also be shifted by five units on the x-axis. The values of this wave will be either at 1.0 or at zero.
REMARKS:	The square wave is generated by calculating: $S = \sin(i*\text{spacing}*frequency*2*PI + \text{phase})$ and returns a step function with a line at 1 when S is negative, and at 0 when S is positive.

GTRIWAVE(points, spacing, frequency, phase)

PURPOSE:	Generates a triangular wave in accordance with the specified parameters.
points	Number of points in the waveform.
spacing	Spacing between each point on the x-axis.
frequency	(Optional). An operand, expressed in cycles per unit time, to adjust the frequency of the waveform. The default frequency is 1.
phase	(Optional). An operand to adjust the phase of the waveform, specified in radians. The default is 0.
RETURNS:	A series.
EXAMPLE:	GTRIWAVE(100, 0.1, 2, 5) creates a triangular wave of 100 points spaced 0.1 units apart. This waveform will also be shifted by five units on the x-axis. The values of this will vary between 1.0 and zero.
REMARKS:	The triangular wave is generated by calculating: $S = \sin 2*(i*\text{spacing}*frequency*2*PI + \text{phase})$ and returns a line rising from 0 to 1 where S is positive, and a line falling from 1 to 0 where S is negative.

HAMMING(series)

- PURPOSE:** (A macro). Multiplies a series by a Hamming window.
- series** A series or a table.
- RETURNS:** A series or table.
- EXPAN-
SION:** GHAMMING(SERSIZE(S), DELTAX(S))* (S)
- EXAMPLE:** HAMMING(W2)
multiplies the series in window 2 by a Hamming window of the same length and spacing as the W2 series, and plots the resultant series in the current window.
- SEE ALSO:** HANNING
KAISER
GHAMMING

HANNING(series)

- PURPOSE:** (A macro). Multiplies a series by a Hanning window.
- series** A series or a table.
- RETURNS:** A series or table.
- EXPAN-
SION:** GHANNING(SERSIZE(S), DELTAX(S))* (S)
- EXAMPLE:** HANNING(W3)
multiplies the series in window 3 by a Hanning window of the same length and spacing as the W3 series, and plots the resultant series in the current window.
- SEE ALSO:** HAMMING
KAISER
GHANNING

HESS(matrix)

PURPOSE: Finds the Hessenberg form of a matrix.

matrix A real or complex square matrix.

RETURNS: A matrix.

EXAMPLE: $x = \begin{matrix} 1 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 12 \end{matrix}$

$\text{HESS}(x) = \begin{matrix} 1.0 & -4.982 & -0.424 \\ -9.434 & 17.506 & 1.809 \\ 0.0 & -0.19101 & 0.49438 \end{matrix}$

This example produces a Hessenberg matrix that is all zero below the first sub-diagonal.

HIDE

PURPOSE: Hides a window without tiling or resizing the windows surrounding it.

RETURNS: Nothing.

REMARKS: If the worksheet has a custom layout HIDE window will leave an empty space, otherwise it will retile the windows.

SEE ALSO: DISPLAY
DISPLAYALL

HIGHPASS(order, rate, fc, ripple, attn, fs)

PURPOSE: Designs a FIR linear phase highpass filter.

order (Optional). The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.

rate A real number that specifies the sampling rate of the filter in cycles per unit time.

fc A real number that specifies the cutoff frequency of the filter in Hertz.

ripple (Optional). A real number for the passband ripple in dB. The default value is 3 dB.

attn (Optional). A real number for the stopband attenuation in dB. The default value is 40 dB.

fs (Optional). A real number that specifies the stopband edge frequency of the filter in Hertz. The default value is $fc - rate * 0.05$.

RETURNS: The time domain impulse response of the filter.

EXAMPLES: HIGHPASS(1000.0, 100.0)
creates a HIGHPASS filter with a sampling rate of 1000 Hz, and a cutoff frequency of 100 Hz. The stopband edge frequency defaults to 50 Hz. The resulting filter is 25 points long, with a passband ripple of 1.8 dB and a stopband attenuation of 52 dB.

 HIGHPASS(1000.0, 100.0, 3.0, 50.0, 70.0)
creates a similar filter to the above except the stopband attenuation is set to 50 dB and the stopband edge is increased to 70 Hz. The resulting filter is 45 points long with a passband ripple of 2.27 dB and the stopband attenuation increases to 60 dB.

REMARKS: The band edges must lie between 0.0 and rate/2 Hz. The cutoff frequency must be less than the stopband edge frequency.

 The resulting characteristics of the filter are written to an ASCII file named HIGHPASSn.FIR, where n is the nth filter designed. This file can be displayed by using the HIPASS macro. For example, to display the filter characteristic file named HIPASS4.FIR, try HIPASS(4).

 FIR Highpass filters require an odd filter order. If you specify an even order, MarketBrowser increases the order by 1.

 Use the FIRMAG function to display the frequency response of the filter.

HISTOGRAM(series, bins)

PURPOSE: (A Macro). Calculates the frequency of values in a series.

series A series.

bins An integer. The number of bins.

RETURNS: A series.

**EXPAN-
SION:** AMPDIST(S, (MAX(S) - MIN(S))/((B)*10E5) + (MAX(s) - MIN(S))/(B)); BARS

SEE ALSO: AMPDIST

HOSTID, USERID, GROUPID, PID

PURPOSE: Returns system or session specific values.

RETURNS: Integer representation of corresponding operating system calls.

REMARKS: STRNUM(GETPID) is useful in generating unique file names.

HOSTNAME, GROUPNAME, USERNAME

PURPOSE: Returns the name assigned to the host computer.

RETURNS: A string as returned by the corresponding operating system call.

I

PURPOSE: Provides the value of the imaginary number (the square root of -1).

EXAMPLE: EXP(PI*I)
displays Mag=1.0000 Angle=3.1416, 180.0000
a 180-degree angle in polar coordinates notation and degree notation

SEE ALSO: DEG E GAMMA
PHI PIE SETDEGREE

IDFT(series)

PURPOSE: Calculates the inverse discrete Fourier Transform of any series or series expression in real/imaginary form.

series A series or table.

RETURNS: A series or table.

REMARKS: The IDFT produces the same result as an IFFT. Although the IDFT is a more straightforward method than the IFFT is for calculating the discrete Fourier Transform, it is also a much slower algorithm.

SEE ALSO: DFT IFFT

IF(cond, true, false)

PURPOSE: Evaluates a conditional expression. If the conditional expression is non-zero (TRUE), the “true” expression is evaluated. If the conditional expression is zero (FALSE), the optional “false” expression is evaluated. “True” and “false” can be any MarketBrowser expression.

cond Any MarketBrowser expression resolving to a number.

true Expression to evaluate if cond is non-zero (TRUE).

false Optional expression to evaluate if cond is zero (FALSE).

RETURNS: The result of “true” or “false”.

EXAMPLE: IF(MAX(W1) > MAX(W2), GSIN(100,.01,1.0), GRAND(200,1.0))

If the maximum value of W1 is greater than the maximum value of W2, then a sine wave is generated in the current window. If not, a random series is generated.

REMARKS: In MarketBrowser 2.x, enclose a string that you do not want evaluated in two sets of quotes, i.e., "hello". In MarketBrowser 3.0, only use one set of quotes around a string that you do not want evaluated, i.e., "hello".

SEE ALSO: The *MarketBrowser Developer's Guide* (especially Chapter 2 “Using XPL”)

IFFT(series)

PURPOSE: Calculates the inverse Fast Fourier transform of a series or series expression in Cartesian (real/imaginary) form.

series A series or table.

RETURNS: A series or table.

EXAMPLES: Set up a four-window worksheets as:

W1: GSIN(125, 0.01, 1.0)

W2: GSIN(128, 0.01, 1.0)

W3: IFFT(W1)

W4: IFFT(W2)

Compare the speeds of the two IFFTs. The 128 (power of 2) point IFFT should be considerably faster.

REMARKS: MarketBrowser uses a mixed radix IFFT. Series with lengths that are a power of 2 will be processed faster than other series. Use the SERSIZE function to find out if a series is a power of 2 points long. Use EXTRACT to tailor series to lengths such as 512 or 1024.

SEE ALSO: IFFTP
IDFT
FFT
FFTP
DFT

IFFTP(series)

PURPOSE: Calculates the inverse Fast Fourier transform of a series in polar (magnitude/phase) form.

series A series or table.

RETURNS: A series or table.

REMARKS: Uses the same algorithm as the IFFT but is slower because it calculates magnitude/phase.

SEE ALSO: IFFT
FFTP

IIR(series, coeffseries, initseries)

PURPOSE: Evaluates an infinite impulse response difference equation.

series A series or table.

coeffseries IIR coefficient series.

initseries Optional initial conditions series.

RETURNS: A series or table.

EXAMPLES: An IIR difference equation is of the form:

$$y(n) = x(n) + a1*y(n-1) + a2*y(n-2) + \dots + aN*y(n-N)$$

Note that IIR difference equations contain feedback.

For example, if $x(n) = 1$ for $n = 1$
2 for $n = 2$
1 for $n = 3$
1 for $n = 3$
0 for $n > 4$

and

$$y(n) = x(n) + 0.8*y(n-1) + -2.0*y(n-2) + 10.0*y(n-3)$$

we can find $y(n)$ with:

`IIR(GSER(1.0, 2.0, 1.0, 1.0), GSER(0.8, -2.0, 10.0))`

The resulting series contains the values:1.0, 2.8, 1.24, 6.39

To evaluate the same equation for 20 values of $x(n)$, try:

`IIR(EXTRACT(GSER(1.0, 2.0, 1.0, 1.0), 1, 10), GSER(0.8, -2.0, 10.0))`

If we add the initial conditions that $y(-1) = .5$ and $y(0) = -0.2$ to the original equation, then we have:

`IIR(GSER(1.0, 2.0, 1.0, 1.0), 1, 20), GSER(0.8, -2.0, 10.0), GSER(0.5, -0.2))`

Then the resulting series contains the values:1.8, 2.44, -0.65, 13.6

REMARKS: Because IIR difference equations contain feedback terms, it is possible to create an unstable system that results in math overflows.

The IIR function produces an output series containing the same number of samples as the input series.

IMAGINARY(expr)

- PURPOSE:** Calculates the real component of the imaginary part of an expression.
- expr** Any expression evaluating to a series, table, integer, real, or complex number.
- RETURNS:** A series, table or number, depending on the input.
- EXAMPLES:** `IMAGINARY(1)`
displays 0.0.
`IMAGINARY(3.0 + 4.0i)`
displays 4.0.
`IMAGINARY(GSIN(20, 0.05))`
returns a series with twenty zeros because a generated sine wave contains no imaginary component.
- REMARKS:** The series or number returned is real, not complex.
- SEE ALSO:** REAL
CARTESIAN
MAGNITUDE
PHASE

IMPULSE(start, length)

- PURPOSE:** (A macro). Generates a discrete unit impulse series.
- start** Point location of the impulse.
- length** Total length of the series.
- RETURNS:** A series or table.
- EXPAN-
SION:** `EXTRACT(GSER(1), 2 - START, LENGTH)`
- EXAMPLE:** `IMPULSE(10, 20)`
creates a series of 20 points where the 10th point of the series has a value of 1.0 and the other points are zero.
- SEE ALSO:** GSER
EXTRACT

INDEX(series)

PURPOSE: Normalizes a series to percentage terms.

series A series or table.

EXAMPLE: INDEX(GSER(3,4,3,5))

creates a new series with the values 100, 133, 100, 166. These preserve the shape of the data as relative values.

REMARKS: The function normalizes by dividing by the value of the first point. Thus if the first point is zero, the indexing is meaningless.

INDEXTODT(series, index, date_format)

PURPOSE: Returns the closest date and/or time of an index point in a series.

series (Optional). Window or valid variable. Defaults to the current window.

index Integer. Index to date and/or time.

date format Integer. Format in which the date and/or time is supplied. Options are:

1	mm/dd	01/15
2	mm/dd/yy	01/15/1998
3	mm-dd-yyyy	01-15-1998
4	mm/yy	01/98
5	yyq	98Q1
6	yy	98
7	yyyy	1998
8	hh:mm:ss	11:59:59
9	hh:mm	11:59
10	mm:sss	15:00s 59:59s
11	02/12/95 hh:mm:ss	01/15/1998 11:59:59
12	MMM	JAN
13	dd/mm	28/12
14	dd/mm/yy	28/12/98
15	dd/mm/yyyy	28-12-1998
16	dd/mm/yy hh:mm:ss	28/12/98 17:15:11
17	MMMdd	Jan 21
18	ddMMM	21 Jan
19)	mmmYY	Jan98

RETURNS: A string.

EXAMPLE: Given the following formula:

W1: GLINE(100,1,1,1);SETDATE(01/01/95)

W2: INDEXTODT(W1, 10, 2)

returns the string '01/13/1995'

SEE ALSO: DTTOINDEX DTTOVAL

INFOPRINT(windown, title)

- PURPOSE:** Expands a window to fill the bottom half of the screen, adds the series information box to the top half, and sends the entire image to the default printer.
- windown** (Optional). A window reference to a series. Defaults to the current window.
- title** (Optional). A string to be printed at the top of the series, in quotes. Defaults to the window formula.
- EXAMPLE:** INFOPRINT(W3,"CASH FLOWS")
prints the series from window 3 on the bottom half of a page and the series background info box on the top half. The series portion will be labeled CASH FLOWS.
- REMARKS:** After printing the specified window with info box, MarketBrowser returns to the original screen configuration.
- SEE ALSO:** INFOPRINTALL
PRINT
PRINTALL

INFOPRINTALL(title)

- PURPOSE:** Prints every series in the current worksheet. Expands each window to fill the bottom half of the screen, adds the series information box to the top half, and sends the entire image to the default printer.
- title** (Optional). A string to be printed at the top of each series, in quotes.
- RETURNS:** A printout of every series and its info box.
- EXAMPLE:** INFOPRINTALL("AVERAGE STOCK INDEX")
creates a print of every series and its info box from the current worksheet. Each series is titled "AVERAGE STOCK INDEX".
- REMARKS:** After printing each series, MarketBrowser returns to the original screen configuration.
- SEE ALSO:** INFOPRINT
PRINT
PRINTALL

INHSERSTYLE(window, OnOff)

- PURPOSE:** Causes the window to inherit/not inherit its plotting style(s) from its data series.
- window** (Optional). A window reference. Defaults to the current window.
- OnOff** An integer. 1 = ON; 0 = OFF;
- RETURNS:** Nothing.
- REMARKS:** If a window is toggled to display data as filled bars, INHSERSTYLE(1) would cause a line chart overplotted from another window to display as a line, because this is the inherited style. If INHSERSTYLE(0) is set, the overplot would show as another bar graph, because the window's own style now has precedence.
- SEE ALSO:** INHWINSTYLE
SETPLOTSTYLE

INHWINSTYLE(window, OnOff, sernum)

- PURPOSE:** Causes the series to inherit/not inherit its plotting style from the window.
- window** (Optional). A window reference. Defaults to the current window.
- OnOff** An integer. 1 = ON; 0 = OFF.
- sernum** Index (origin 1) specifying which data series to set.
- RETURNS:** Nothing.
- SEE ALSO:** INHSERSTYLE
SETPLOTSTYLE

INNERPROD(matrix1, matrix2, op1, op2)

PURPOSE:	Calculates the matrix inner (or "dot") product.
matrix1	A rectangular matrix.
matrix2	A matrix with as many rows as matrix1 has columns.
op1	Quoted string containing first binary operator.
op2	Quoted string containing second binary operator.
RETURNS:	Each entry of the output matrix is computed from a row of matrix1 and a column of matrix2. For an entry in the Ith row and the Jth column in the output matrix, the value is equivalent to REDUCE(ROW(matrix1,I) op2 COL(matrix2,J), op1). The number of columns in matrix1 must equal the number of rows in matrix2.
EXAMPLE:	INNERPROD(W1, W2, "+", "*") If W1 and W2 contain conforming matrices, this expression results in their "matrix multiplication".
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR".
SEE ALSO:	OUTERPROD MMULT REDUCE COLREDUCE ROWREDUCE INTERPOSE

INT(expr)

PURPOSE:	(A Macro). Calculates the integer value of an expression.
expr	A series, table or number.
RETURNS:	A series, table or number.
EXPAN- SION:	TRUNC(ARG)
EXAMPLES:	INT(5.73) displays 5. INT(W1) returns a new series by applying INT to each value of the W1 series.
SEE ALSO:	REAL

INTEG(series)

PURPOSE:	Calculates the integral of a series.
series	A series or table.
RETURNS:	A series or table.
EXAMPLE:	INTEG(W4) creates a new series from the contents of window 4 and places the result in the current window. The value of each point in the new series will be the integral of the corresponding point in window 4.
REMARKS:	The INTEG function uses Simpson's rule to compute the integral. This method fits a quadratic function to three points of the series for area calculations.
SEE ALSO:	AREA DERIV LDERIV RDERIV

INTERPOLATE(series, n)

PURPOSE:	Linearly interpolates (enlarges) a series by a factor n.
series	A series or table.
n	An integer point number by which to interpolate the series.
RETURNS:	A series or table.
EXAMPLES:	<p>INTERPOLATE(W1, 3)</p> <p>enlarges the series in window 1 by a factor of 3 and places the result in the current window. This new series is created by inserting two points between each point in the W1 series.</p> <p>INTERPOLATE(EXTRACT(W2,10,LENGTH(W2)-10),4)</p> <p>enlarges the series from window 2 by a factor of 4, starting from the 10th point of the series, and places the result in the current window.</p>
SEE ALSO:	DECIMATE

INTERPOSE(series, op)

PURPOSE:	Inserts an operator between every observation of a series and evaluates associatively, producing a vector of successive intermediate results.
series	A series or table.
op	Quoted string containing the binary operator.
RETURNS:	A series or table.
EXAMPLE:	<p>INTERPOSE(GSER(1,2,3), "*")</p> <p>Expands to the expression "(((1)*2)*3)", which evaluates associatively, producing one observation for each pair of parentheses, for a series result of 1,2,6.</p>
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR".
SEE ALSO:	REDUCEI INNERPROD PARTPROD OUTERPROD COLREDUCE ROWREDUCE MMULT

INVERSE(matrix)

PURPOSE: Computes the inverse of a matrix.

matrix A real or complex square matrix.

RETURNS: A matrix.

EXAMPLE:

x =	1	3	4
	5	6	7
	8	9	12

INVERSE(x) =	-0.6	0.0	0.2
	0.2667	1.3333	-0.8667
	0.2	-1.0	0.6

REMARKS: When x is badly scaled or nearly singular, no inverse of the matrix can be obtained.

SEE ALSO: MMULT

ISDLFUNC(funcname)

PURPOSE: Determines whether or not a DLL function exists.

funcname The name of the function to test.

RETURNS: Returns 1 if the DLL function specified in the quoted string funcname exists, otherwise returns 0.

SEE ALSO: DLBIND
DLRUN
DLUNBIND

ISMACRO(name)

PURPOSE: Determines whether or not a macro exists.

name The name of the macro to test in quotes.

RETURNS: Returns 1 if the macro specified exists, otherwise returns 0.

SEE ALSO: ISVAR

ISNAVALUE(series)

PURPOSE:	Returns data of the same shape as its input, with ones wherever the input is NA and zeros elsewhere.	
series	A series, table or real number.	
RETURNS:	A series, table or real number.	
SEE ALSO:	NAFILL	NAVALUE
	SETNAVALUE	CONFORM

ISNUMBER(argument)

PURPOSE:	Tests whether the input argument is a number.
argument	Any type of input argument
RETURNS:	If the input argument evaluates to a number, it returns a 1; otherwise, it returns a 0.
SEE ALSO:	STRNUM

ISRT(varname)

PURPOSE:	Tests whether a hot variable is source of real-time updating or not.
varname	A variable name or window reference, enclosed in quotes.
RETURNS:	<p>If given a variable name, returns 1 if:</p> <ol style="list-style-type: none">1) The variable is a real-time collector.2) The variable contains a series which has "painttick" set or has a nonzero highwater mark, because such a series must be downstream from a real-time collector.3) The variable contains a window reference which contains a series meeting #2 above. <p>If given a window reference, returns 1 if:</p> <ol style="list-style-type: none">1) The window is a real-time collector.2) The window contains a series meeting #2 above. <p>Returns 0 otherwise.</p>
REMARKS:	Note that with these conditions the function can give false negative answers but not false positives. This is because real-time causality cannot be traced backwards through variable assignments, XPL functions, etc.

ISVAR(variable, type)

PURPOSE:	Determines whether a string is an XPL variable defined in the worksheet. If the type argument is specified, it returns whether or not a variable of the said type exists.
variable	String optionally enclosed in quotes. The name of the variable for which you are testing.
type	(Optional). Integer. The variable's type. Options are: 1 - Global variable 2 - Local variable 3 - User-defined function 4 - Hot variable (real-time) 5 - Formal variable
RETURNS:	An integer. 1 = the variable exists; 0 = the variable does not exist.
EXAMPLE:	a:= MONITOR("IBM.N;TRD_PRC"); ISVAR("a", 4) returns the integer 1, which indicates that "a" is in fact a variable of type 4 (a hot variable).
REMARKS:	Type 3 (user-defined functions) checks to see if a function of the given name exists. Type 5 (formal variables) are arguments to an XPL function.
SEE ALSO:	VALUE TYPE FUNCTIONS VARS

ITEMCOL(window, item)

PURPOSE:	Returns the column number where a specified item begins, relative to the first column in the window.
window	(Optional). Window reference. Defaults to the current window.
item	(Optional). A positive integer. The index to the item in the window. Defaults to 1, or the first item.
RETURNS:	An integer.
EXAMPLE:	Given the formula: W1: RAVEL(GRANDOM(100,1),25); OVERLAY(GRANDOM(100,1,5,6)) W2: ITEMCOL(W1,2) returns 5, which states the second item starts in the fifth column.
REMARKS:	If the specified item is greater than the number of items in the window, ITEMCOL returns 0.
SEE ALSO:	COL ITEMCOUNT NUMITEMS SERCOUNT

ITEMCOUNT(window, item)

PURPOSE:	Returns the number of columns in an item.
window	(Optional). Window reference. Defaults to the current window.
item	(Optional). A positive integer. The index to the item in the window. Defaults to 1, or the first item.
RETURNS:	An integer.
EXAMPLE:	Given the window formula: W1: RAVEL(GRANDOM(100,1),4) W2: ITEMCOUNT(W1) returns 25, the number of columns in the item in W1.
REMARKS:	If the specified item is greater than the number of items in the window, ITEMCOUNT returns 0.
SEE ALSO:	COL ITEMCOL NUMITEMS SERCOUNT

JN(series, order)

PURPOSE:	Performs the Bessel function on a series or a number.
val	A series or table.
order	An integer order.
RETURNS:	A series or table.

JULDAY(juldate)

PURPOSE:	Returns the day of the week for a given Julian date.
juldate	A Julian date.
RETURNS:	An integer representing the day of the week associated with the date.
EXAMPLE:	JULDAY(JULSTR('01/13/95')) returns 5, or Friday.
REMARKS:	The integers corresponding to days of the week are: 0 - Sunday 1 - Monday 2 - Tuesday 3 - Wednesday 4 - Thursday 5 - Friday 6 - Saturday

SEE ALSO: JULSTR STRJUL
ADDBDAY

JULSTR(date)

PURPOSE: Calculates a Julian date from a date string.

date A valid date string, in quotes.

EXAMPLE: JULSTR('12/19/90') - JULSTR('12/14/90')
returns 5, the number of days between the two dates.

REMARKS: Useful in providing day counts for time-to-maturity and time to expiration calculations.

SEE ALSO: STRJUL
JULDAY
ADDBDAY

KAISER(series)

PURPOSE: (A macro). Multiplies a series by a Kaiser window.

series A series or table.

RETURNS: A series or table.

**EXPAN-
SION:** GKaiser(LENGTH(S), DELTAX(S))*(S)

EXAMPLE: KAISER(W2)
multiplies the series in window 2 by a Kaiser window, of the same length and spacing as the W2 series, and plots the resultant series in the current window.

SEE ALSO: HANNING
HAMMING
GKAISER

LABEL(window, string)

PURPOSE:	Sets the label for a window.
window	(Optional). A window reference. Defaults to the current window.
string	A text string, in quotes.
RETURNS:	Nothing.
EXAMPLE:	LABEL(W4, strcat("IBM as of ", getdate)) places "IBM as of 04/14/89" into the label of the current window.
SEE ALSO:	GETLABEL COMMENT

LAG(series, n)

PURPOSE:	Shifts a series right by n number of points along the x-axis.
series	A series or table.
n	An integer. The number of points used to offset the series.
RETURNS:	A series or table.
REMARKS:	When performing a technical study and especially any type of correlation, use DELAY rather than LAG to ensure correct results. For example, instead of using: LAG(w1,10) use: EXTRACT(DELAY(w1,10),11,-1)
SEE ALSO:	LEAD DELAY SETXOFFSET

LAYOUT(rownum, colnum)

PURPOSE:	(A macro). Controls the layout of windows in the worksheet.
rownum	An integer representing the number of windows to display per row. The maximum is 10.
colnum	An integer representing the number of windows to display per column. The maximum is 10.
EXPAN- SION:	ADDWINDOW(0, R, C)
SEE ALSO:	ADDWINDOW PRINTOPT SCREENOPT

LDERIV(series)

- PURPOSE:** Returns the derivative of a series or table using a left-to-right slope algorithm.
- series** A series or table.
- RETURNS:** A series or table.
- EXAMPLE:** LDERIV(W1)
creates a new series from the contents of window 1 and places the result in the current window. The value of each point in the new series is the slope of the series in window 1 at that point.
- REMARKS:** The formula used to compute derivatives with the LDERIV function for each point i is as follows:
$$\text{LDERIV}(i) = (\text{ser}\langle i \rangle - \text{ser}\langle i - 1 \rangle) / (\text{deltax})$$
The derivative of the first point is computed using the RDERIV method.
- SEE ALSO:** INTEG DERIV
RDERIV

LEAD(series, n)

- PURPOSE:** Shifts series left by n number of points along the x-axis.
- series** A series or table.
- n** An integer. The number of points used to offset the series.
- RETURNS:** A series or table.
- REMARKS:** When performing a technical study and especially any type of correlation, use DELAY rather than Lead to ensure correct results. For example, instead of using:
LEAD(w1,10)
use:
EXTRACT(DELAY(w1,-10),11,-1)
- SEE ALSO:** LAG
DELAY
SETXOFFSET

LEGCUR(target, fg_color, bg_color, font, box, margin_flag, focus)

PURPOSE:	Inserts a legend for all the series in the window.
target	Integer. Defines the manner in which the text should scroll with the series. <ul style="list-style-type: none">• 0 = PAPER - Moves with the series• 1 = GLASS - Remains in place as the series is scrolled• 2 = GLASS_WMARGIN - Glass-style plotting over the entirety of the window.• 3 = GLASS_WPMARGIN - Glass-style plotting over the windows entire vertical dimensions, but only over the horizontal dimensions of the plotting area.
fg_color	(Optional). Integer representing the text color. Defaults to the series' color, or -1.
bg_color	(Optional). Integer representing the background color. Defaults to the window's color, or -1.
font	(Optional). Integer. 0 = LARGE, 1 = SMALL. Defaults to 0.
box	(Optional). Integer. 0 = Off, 1 = On. Draws a box around the legend text. Defaults to 1.
margin_flag	(Optional). Integer. Margin to be adjusted. <ul style="list-style-type: none">• -1 = No option• 0 = TOP• 1 = RIGHT• 2 = BOTTOM• 3 = LEFT
focus	(Optional). Integer. In an window with overlays, the desired series.
RETURNS:	A legend associated with the various series in the window.
EXAMPLE:	LEGCUR(1, 12, -1, 0, 0, -1, 1) returns a legend in the current window that does not scroll with the worksheet, has light red text against a background the same color as the window, with small font, with the default margin style, and that has as focus the first series overlaid into the window.
REMARKS:	If the window in which you have inserted a legend with LEGCUR evaluates often, you might want to try using LEGEND instead. Because LEGCUR is a plot-time function, every time the window is reevaluated, LEGCUR inserts a new cursor in the window. Until you set the cursor's position, the window is frozen at that point.

LEGCUR derives the text it uses in the legend from the comments associated with each series in the window. You can manipulate comments using the GETCOMMENT and COMMENT functions.

The colors available to you for the text and background colors are pre-defined in palette.mac. Refer to this file for numbers associated with your desired color selections.

You can perform the standard editing functions on a legend with the TEXTMOVE, TEXTEDIT, and TEXTDELETE functions. These functions are easily accessed from the Drawing pull-down.

SEE ALSO: TEXTCUR LEGEND
 GETCOMMENT GETSCOMMENT
 COMMENT

LEGEND(x, y, target, fg_clr, bg_clr, font, box_flg, margin_flg, focus)

PURPOSE: Sets the attributes and location for a standard legend.

x, y The x and y coordinates of the point where you would like to place the legend.

target (Optional). An integer specifying the relationship of the text to the window. Defaults to 0.

- 0 = PAPER. Text on the “graph paper” in the window; within the coordinate system of the data.
- 1 = GLASS. Text within the plotting area of window.
- 2 = GLASS_WMARGIN. Text within the area of the entire window.
- 3 = GLASS_WPMARGIN. Text within the vertical dimensions of a window, and within the horizontal dimensions of the plotting area.
- 4 = GLASS_WSMARGIN. Text within the entire worksheet area. Bounded above by the toolbar.

fg_clr (Optional). An integer specifying the color of series in the window. Defaults to the color of the primary series.

bg_clr (Optional). An integer specifying the background color of the annotated text. Defaults to window’s color.

font (Optional). An integer specifying the font size. 0 = large font, 1 = small font. Defaults to 0.

box_flg (Optional). An integer specifying presence or absence of solid line box surrounding the text (with margin if legend_flg is ON, otherwise, no margin). 1 = ON; 0 = OFF. Defaults to 1.

margin_flg (Optional). An integer specifying margin to be adjusted. Defaults to -1.

- -1 = No Adjustment
- 0 = Top Margin
- 1 = Right Margin
- 2 = Bottom Margin

- 3 = Left Margin

focus (Optional). An integer specifying focus for PAPER annotations. Defaults to 1.

RETURNS: Nothing.

EXAMPLE: W1: GSIN(100,.01);SETSYMBOL(1,1,10,1); COMMENT (“Sine”)
 W2: GCOS(100,.01); SETSYMBOL(9,1,10,5); COMMENT
 (“Cosine”);LEGEND(.1,.8,1)

puts a legend in the bottom left corner of the window.

REMARKS: X and Y coordinate systems differ depending on whether your target is PAPER or GLASS. All GLASS coordinates are normalized to the specified rectangular regions in the worksheet, where the upper left corner is (0.0, 0.0) and the lower right corner is (1.0, 1.0). GLASS annotations “stick” to the window like the viewfinder in a camera. Paper coordinates, on the other hand, are taken from the x and y values of the series in the window. PAPER annotations scroll with the data.

The standard legend uses the comment field, retrieved via GETCOMMENT, to describe each item. To revise the legend, use the COMMENT command, followed by PON.

SEE ALSO:	LEGCUR	TEXTANN
	GETCOMMENT	GETSCOMMENT
	COMMENT	TEXTDEL
	TEXTEDIT	TEXTMOVE

LENGTH(series)

PURPOSE: Returns the length of a series or table.

series (Optional.) A series or table. Defaults to the current window.

RETURNS: A number.

EXAMPLE: LENGTH(GSER(4,5,6))
 returns 3, the length of the generated series.

SEE ALSO: COLLENGTH
 EXTRACT
 NUMOBSV

LEVELCROSS(series, level, edgedetect, edgeout)

PURPOSE: Creates a series with the value 1.0 (TRUE) where the input series crosses the level and 0.0 (FALSE) elsewhere.

series A series or table.

level Level crossing threshold.

edgedetect (Optional.) Crossing definition.

edgeout (Optional.) Output value alignment.

RETURNS: A series or table.

REMARKS: The optional EDGEDETECT parameter specifies the following edge detect methods:

- 0 - detect both rising and falling edges (default)
- 1 - detect rising edge only
- 2 - detect falling edge only

Because LEVELCROSS returns a regularly spaced series (i.e. an interval series), the actual crossing point may occur between two data points. The optional EDGEOUT parameter determines where the detected edge output will be placed. The output value will be placed to the left or right of the actual crossing point as specified below:

- 0 - left if input edge rising, right if falling (default)
- 1 - right on rising, left on falling
- 2 - right whether rising or falling
- 3 - left whether rising or falling

LEVELCROSS returns the exact crossing point if EDGEOUT is set to 4. In this case, LEVELCROSS returns an XY series, where X is the crossing location and Y is 1.0.

LINEANN(color, style, target, i, focus, inline, x1, y1,...xn, yn)

PURPOSE: Draws a polyline from the command line or a user-defined macro.

color (Optional). An integer or macro color name (e.g. RED) specifying line color. Defaults to the color of the primary series.

style (Optional). An integer specifying line style. Defaults to solid. Valid entries are:

- 1 = Solid
- 2 = Dashed
- 3 = Dotted

target (Optional). An integer specifying the relationship of the text to the window. Defaults to 0.

- 0 = PAPER. Text on the “graph paper” in the window; within the coordinate system of the data.
- 1 = GLASS. Text within the plotting area of window.

- 2 = GLASS_WMARGIN. Text within the area of the entire window.
- 3 = GLASS_WPMARGIN. Text within the vertical dimensions of a window, and within the horizontal dimensions of the plotting area.
- 4 = GLASS_WSMARGIN. Text within the entire worksheet area. Bounded above by the toolbar.

i (Optional). An integer that sets the line color to match the color of the selected overplot. Defaults to 1 (primary series).

focus (Optional). An integer specifying 1-based focus offset for PAPER annotations

infinite 0 - segment, 1 - infinite, 2 - ray

**x1, y1, ...
xn, yn** Real number coordinates representing endpoints of line segments.

RETURNS: Nothing.

EXAMPLE: GTRI(100,.01,2);LINEANN(purple, 2, 1, -1, 0.09, 0.04, 0.9, 0.5)
creates a purple dotted (2) line in “glass mode” (1). The color of the line is purple, and is unrelated to any overplots (-1); the line spans from coordinates (0.09, 0.04) to (0.9, 0.5).

REMARKS: X and Y coordinate systems differ depending on whether your target is PAPER or GLASS. All GLASS coordinates are normalized to the specified rectangular regions in the worksheet, where the upper left corner is (0.0, 0.0) and the lower right corner is (1.0, 1.0). GLASS annotations “stick” to the window like the viewfinder in a camera. Paper coordinates, on the other hand, are taken from the x and y values of the series in the window. PAPER annotations scroll with the data.

LINEANN is a plot-time function, and is therefore reevaluated with every PON redraw.

To use LINEANN from the command line, you must enclose a call to LINEANN() in a string passed to ADDWFORM() manually, or append it to the current window formula.

The rules for determining the drawing color used for LINEANN (or LINECUR) follow. If a color is supplied as the first argument, that color will be used to draw all polylines. If -1 is specified as the first argument, MarketBrowser checks the overplot index, the fourth argument. If an overplot index is specified, MarketBrowser draws all polylines using the color that corresponds to the index. If the color argument is -1 and the overplot index is -1 (or omitted), MarketBrowser draws the polylines using the color of the primary series.

SEE ALSO: LINECOPY
LINECUR
LINEDEL
LINEMOVE
ADDWFORM

LINECOPY

- PURPOSE:** Copies a polyline created with LINECUR.
- RETURNS:** Nothing.
- REMARKS:** LINECOPY places handles at each polyline point in the window. You may choose a polyline annotation by crossing the line with the mouse cursor while holding down the left mouse button. Or you can place the cursor directly on the line and "click and drag". Either way, a rubberband polyline then appears and moves with the mouse cursor until you release the left mouse button. Upon release, LINECOPY copies the chosen polyline annotation from its previous location, and replaces the rubberband line with the chosen polyline.
- You are not limited to the number of lines you can copy with LINECOPY. Press the right mouse button (or ESC) to indicate that you are finished.
- SEE ALSO:** LINEMOVE

LINECUR(color, style, target, i, focus inline)

- PURPOSE:** Brings up a freehand line drawing cursor in a window.
- color** (Optional). An integer or macro color name (e.g. RED) specifying line color. Defaults to the color of the primary series.
- style** (Optional). Integer specifying the line style of series in the window. Defaults to solid line. Valid entries are:
- 1 = Solid
 - 2 = Dashed
 - 3 = Dotted
- target** (Optional). An integer specifying the relationship of the text to the window. Defaults to PAPER, or 0.
- 0 = PAPER. Text on the "graph paper" in the window; within the coordinate system of the data.
 - 1 = GLASS. Text within the plotting area of window.
 - 2 = GLASS_WMARGIN. Text within the area of the entire window.
 - 3 = GLASS_WPMARGIN. Text within the vertical dimensions of a window, and within the horizontal dimensions of the plotting area.
 - 4 = GLASS_WSMARGIN. Text within the entire worksheet area. Bounded above by the toolbar.
- i** (Optional). An integer that sets the line color to match the color of the selected overplot. Defaults to 1 (primary series).
- focus** (Optional). An integer specifying 1-based focus offset for PAPER annotations.
- inline** 0 - segment, 1 - infinite, 2 - ray
- EXAMPLE:** LINECUR(-1, 3, 0, 2)

brings up a dotted crosshair cursor with the color of the second series (i.e. the first overplot).

REMARKS: When using the mouse with LINECUR, pressing the mouse button, the . character, or [-] will anchor the first line segment.

The . character (or the middle mouse button on some platforms) can be used at any time to drop the anchor without drawing a line from the prior position.

Using an overplot index lets you associate a polyline annotation with a specific overplot, by guaranteeing that the polyline color will be the same as the overplot color. When setting an index, use 1 to refer to the color of your primary series; use 2 to refer to the color of your first overplot; 3 for your second overplot, etc.

The rules for determining the drawing color used for LINECUR (or LINEANN) follow.

- If a color is supplied as the first argument, that color will be used to draw all polylines.
- If -1 is specified as the first argument, MarketBrowser checks the overplot index, the fourth argument.
- If an overplot index is specified, MarketBrowser draws all polylines using the color that corresponds to the index.
- If the color argument is -1 and the overplot index is -1 (or omitted), MarketBrowser draws the polylines using the color of the primary series.

SEE ALSO: LINEANN
LINECOPY
LINEDEL
LINEMOVE

LINEDEL

PURPOSE: Deletes a line created with LINECUR.

RETURNS: Nothing.

You may delete any number of lines with LINEDEL. Pressing the right mouse button (or ESC) indicates that you are done.

LINEDEL puts a “handle” at each polyline point. You can delete a polyline annotation by crossing the polyline with the mouse cursor while holding down the left mouse button. Or you can place the cursor anywhere on the polyline and click the left mouse button. Your polyline then disappears.

SEE ALSO: TEXTDEL

LISTEDIT(list, title_bar, header, add_function, edit_function, del_function, help_function)

PURPOSE: Creates a graphical list editor for the creation and maintenance of user-defined lists of information.

list (*Optional*). List to prepopulate list editor with. A "list" is a collection of keys and associated entries. It is physically represented as string containing NEWLINE-delimited string entries.

title_bar (*Optional*). Title bar of list editor. If not specified, defaults to "Edit List".

header (*Optional*). Not currently used. Use "" as placeholder.

add_function (*Optional*). To supply your own XPL function to handle the adding of entries to the list, specify the name of the function here. For example, you may want to parse the user's entry or check it for validity before adding it to the list. If you supply your own function, you must use a custom menu, GETSTR, or INPUT to receive input from the user. If no function is specified, the default behavior is to pop up a box for the user to enter a list entry key and list entry body and then add it to the list.

edit_function (*Optional*). To supply your own XPL function to handle the editing of entries in the list, specify the name of the function here. For example, you may want to parse the user's entry or check it for validity before adding it to the list. If you supply your own function, you must use a custom menu, GETSTR, or INPUT to receive input from the user. If no function is specified, the default behavior is to pop up a box for the user to edit an entry key and body. The edited entry replaces the original in the list.

del_function (*Optional*). To supply your own XPL function to handle the deletion of entries from the list, specify the name of the function here. For example, you may want to guard against certain entries being deleted. If no function is specified, the default behavior is to confirm that the user wants to delete the entry and if so, to delete it.

help_function (*Optional*). To supply your own help file for the list editor, write an XPL function to do this and then specify the name of the function here. You may wish to use MENUFILE, VIEWFILE or MESSAGE to display your own help. If no function is specified, MarketBrowser's on-line help is launched.

RETURNS: A list.

EXAMPLE:

```
my_original_list = STRCAT( 'HWP 428236103', STRESCAPE('\n'), 'IBM
459200101', STRESCAPE ('\n'), 'CPQ 204493100' );
my_new_list = LISTEDIT( my_original_list, "Equity CUSIPS" )
```

SEE ALSO: STRESCAPE STRCAT
TOKENIZE MESSAGE
MENUFILE GETSTR
INPUT

LLU(matrix)

PURPOSE: (A Macro). Computes a lower triangular matrix in permuted LU decomposition.

matrix A real square matrix.

EXAMPLE: x =

1	2	3
4	5	6
7	8	10

LLU(x) =

.14	1.0	0.0
.57	0.5	1.0
1.0	0.0	0.0

REMARKS: Equivalent to: LU(matrix, 0, 1)

SEE ALSO: LU
ULU

LN(expr)

PURPOSE: Calculates the natural logarithm of the specified expression.

expr Any expression evaluating to a series, table, integer, real or complex number.

RETURNS: A series, table or number.

EXAMPLES: LN(W2)
creates a new series from the contents of window 2 and places the result in the current window. The value of each point in the new series will be the natural logarithm (base e) of the corresponding point in window 2.

LN(1)
displays the natural log of 1 which is 0.

REMARKS: LN and LOG are identical.

SEE ALSO: LOG
LOG10
DEG
E
PHI
EXP
GEXP
GAMMA

LOAD(filename)

- PURPOSE:** Loads and executes a command file directly from a worksheet.
- filename** Name of command file to loaded, in quotes.
- EXAMPLE:** LOAD("MYCFIL")
loads and executes the command file "MYCFIL".
- REMARKS:** The specified command file will be loaded into the current worksheet. Be sure your command file is meant to run in an open worksheet.
- SEE ALSO:** CALLRUN

LOADWORKSHEET(wname)

- PURPOSE:** Loads a worksheet.
- wname** The full path and name of the worksheet, in quotes.
- RETURNS:** 1 if the file is successfully opened, 0 if it could not be opened.
- EXAMPLE:** LOADWORKSHEET("c:\expo\My worksheets\example.xpw")

LOG(expr)

- PURPOSE:** Calculates the natural logarithm of the specified expression.
- expr** Any expression evaluating to a series, table, integer, real or complex number.
- RETURNS:** A series, table or number.
- EXAMPLES:** LOG(W1)
creates a new series from the contents of window 1 and places the result in the current window. The value of each point in the new series will be the natural logarithm (base e) of the corresponding point in window 1.
- LOG(1)
displays the natural log of 1 which is 0.
- REMARKS:** LOG and LN are identical.
- SEE ALSO:** LN
LOG10

LOG10(expr)

- PURPOSE:** Calculates the common (base 10) logarithm of the specified expression.
- expr** Any expression evaluating to a series, table, integer, real or complex number.
- RETURNS:** A series, table or number.
- EXAMPLES:** LOG10(W3)
This creates a new series from the contents of window 3 and places the result in the current window. The value of each point in the new series will be the base 10 logarithm of the corresponding point in window 3.
- LOG10(10)
displays the common log of 10 which is 1.
- SEE ALSO:** LN
LOG

LOOKUP(series1, series2, factor, offset)

- PURPOSE:** Selects data points from one series according to a "table" of point numbers contained in a second series. The point values isolated by this method are then plotted in the current window.
- series1** A series or table containing the point numbers to be selected from series2. Points in this series must be integers because they refer to point numbers and not y values.
- series2** A series or table.
- factor** (Optional). A multiplicative factor for series1, the "table" of point numbers. Defaults to 1.
- offset** (Optional). An offset added to table after multiplying by factor argument. Defaults to 0.
- RETURNS:** A series or table.
- EXAMPLES:** If window 1 contains the target series (series2) generated by GLINE(10, 0.1, 2, 1), and window 2 holds the series table (series1) with the points GSER(1,2,4,5), then:
LOOKUP(W2, W1)
yields a new series, in the current window, comprised of four points: 1.0, 1.2, 1.6, and 1.8.
- LOOKUP(W2, W1, 2, 1)
yields a four point series with the following coordinates: 1.4, 1.8, 2.6, and 0.0.

LOWPASS(order, rate, fc, ripple, attn, fs)

PURPOSE: Designs a FIR linear phase lowpass filter.

order (Optional). The filter length. If specified, the order must be an integer value. If not specified, MarketBrowser will automatically estimate the required filter order.

rate A real number that specifies the sampling rate of the filter in Hertz.

fc A real number that specifies the cutoff frequency of the filter in Hertz.

ripple (Optional). A real number for the passband ripple in dB. The default value is 3 dB.

attn (Optional). A real number for the stopband attenuation in dB. The default value is 40 dB.

fs (Optional). A real number that specifies the stopband edge frequency of the filter in Hertz. Default value is $fc + rate * 0.05$.

RETURNS: The time domain impulse response of the filter.

EXAMPLES: `LOWPASS(1000.0, 100.0)`
creates a lowpass filter with a sampling rate of 1000 Hz, and a cutoff frequency of 100 Hz. The stopband edge frequency defaults to 150 Hz. The resulting filter is 24 points long, with a passband ripple of 2.6 dB and a stopband attenuation of 49 dB.

`LOWPASS(1000.0, 100.0, 3.0, 50.0, 130.0)`
creates a similar filter to above except the stopband attenuation is set to 50 dB and the stopband edge is lowered to 130 Hz. The resulting filter is 45 points long with a passband ripple of 2.75 dB and the stopband attenuation increases to 58 dB.

REMARKS: The band edges must lie between 0.0 and $rate/2$ Hz. The cutoff frequency must be less than the stopband frequency.

The resulting characteristics of the filter are written to an ASCII file named `LOPASSn.FIR`, where `n` is the `n`th filter designed. This file can be displayed by using the `LOPASS` macro. For example, to display the filter characteristic file named `LOPASS4.FIR`, try:

`LOPASS(4)`

Use the `FIRMAG` function to display the frequency response of the filter.

LU(matrix, type, permute)

PURPOSE: Computes an LU decomposition matrix.

matrix A real square matrix.

type (Optional). Integer. Defaults to 2.

- 0 - Lower LU decomposition matrix
- 1 - Upper LU decomposition matrix
- 2 - Complete LU decomposition matrix

permute (Optional) Integer. Defaults to 1.

- 0 - No permutation
- 1 - Permute

RETURNS: A square matrix.

REMARKS: If only two arguments are supplied, the second argument is considered to represent type. The default type is the complete LU decomposition matrix; the complete LU decomposition matrix is the upper decomposition matrix added to the unpermuted lower decomposition matrix without its diagonal.

SEE ALSO: LLU
ULU
SVD

MACREAD(filename, verbosity, invisible, transient)

PURPOSE:	Reads an external flat-file of macros into MarketBrowser.
filename	Quoted string. The pathname to a valid MarketBrowser macro file.
verbosity	(Optional). Determines the verbosity with which MarketBrowser returns the error messages. Options are: <ul style="list-style-type: none">• 0 - Beep and print message upon error. Return error if error occurs (default).• 1 - Return error if error occurs. Do not print message or beep.• 2 - If error occurs, print message, but return OK.• 3 - Return silently without error regardless of error status.
invisible	(Optional). Determines if the macros defined in the file that do not start with underscores (_) will be displayed in the list displayed with the MACROS command (Custom / Macros / List/Edit). Options are: <ul style="list-style-type: none">• 0 - Display the macros (default).• 1 - Hide the macros.
transient	(Optional). Determines if the macros in the file are saved with a worksheet. Options are: <ul style="list-style-type: none">• 0 - Save all macros that do not start with an underscore (_) with worksheets (default).• 1 - Do not save any macros with worksheets.
RETURNS:	Nothing.
EXAMPLE:	MACREAD('c:\lmt\expo\mymac.mac', 2) reads the macro file 'mymac.mac', prints error messages, but does not return an actual error.
REMARKS:	Option 2 for the verbosity flag allows XPL files to continue processing even if there has been a failure in opening the file.
SEE ALSO:	MACWRITE XPLREAD XPLWRITE

MACROS

PURPOSE:	Displays the list of macros defined within the current worksheet, along with their definitions and arguments.
RETURNS:	Nothing.
SEE ALSO:	MACWRITE ALLMACROS

MACWRITE(filename, prefix, regexp1, regexp2, ..., regexprn, start, end, flag, exit_policy, case_sense, append)

PURPOSE: Writes macros defined within an MarketBrowser worksheet to an external ASCII file.

filename Quoted string. The path and filename to which MarketBrowser will write macros.

prefix (Optional). Quoted string. A string prefix prepended to every macro written to filename. Defaults to no prefix. Use "" (an empty pair of quotes) to maintain the default.

regexp1, ..., regexprn (Optional). Quoted string(s). Valid regular expressions that filter which macros get written to disk. See a definition of regular expressions under the REMARKS section.

start (Optional). Integer. Starts writing macros to the file from the specified line number. Defaults to 0, or the first line.

end (Optional). Integer. Writes macros previous to this line to the file. Defaults to the last line in the macro file.

flag (Optional). Integer flag. Options are:

- 0 - Write visible macros only (default)
- 1 - Write all macros
- 2 - Write hidden (system) macros only
- 3 - Write non-transient macros only
- 4 - Write transient macros only. This argument is processed before Market-Browser has filtered for any regular expressions

exit_policy (Optional). Determines the verbosity with which MarketBrowser returns error messages. Options are:

- 0 - Beep and print message upon error. Return error if error occurs. Default.
- 1 - Return error if error occurs. Do not print message or beep.
- 2 - If error occurs, print message, but return OK.
- 3 - Return silently without error regardless of error status.

case_sense (Optional). Integer flag. Converts all macros to upper case. Options are:

- 0 - Causes all macros and regular expressions to be converted to upper case before being compared.
- 1 - Case sensitive. No conversion (default).

append (Optional). Integer. Overwrite or append existing macro file. Options are:

- 0 - Overwrite any existing macro file.
- 1 - Append to any existing macro file.

RETURNS: Nothing.

EXAMPLE: MACWRITE('mymac.mac', "", "*FX?", -1, -1, 1, 2, 0,1)
appends any macros that fit the regular expression *FX? to the file 'mymac' in MarketBrowser's main installation directory. Macro names are converted to uppercase. In the case of an error, MarketBrowser prints the error message, but

returns OK.

REMARKS: An example of a regular expression is the string:

"DEFAULT[1-9]*"

MarketBrowser would write to file all macros that begin with the string DEFAULT, followed by the number 1 through 9, followed by an _ (underscore) character, and ending with any sequence of printable characters.

Any number of regular expressions may be entered on the command line, and a macro must match AT LEAST one of these to get written. If you do not provide any regular expression filters, all macros get written to file.

SEE ALSO: MACREAD
XPLREAD
XPLWRITE

MAGNIFY

PURPOSE: To zoom in on a selected part of a series.

RETURNS: A magnified view of the area you selected that fills the entire window.

REMARKS: To use this function type magnify and press return. Click and hold down the left mouse button and drag the box over the area you want to magnify. When you release the mouse button, the area you selected will fill the entire window.

This function is also available from a pull-down menu and from the toolbar for an activated window.

MAGNITUDE(expr)

PURPOSE: Returns the magnitude component (always positive) of a series that is in Polar (magnitude/angle), Cartesian (real/imaginary) or other form.

expr Any expression evaluating to a series, table, integer, real or complex number.

RETURNS: A table, series, or number.

EXAMPLES: MAGNITUDE(-3)
returns three.
MAGNITUDE(3.0 + 4.0i)
returns 5, the hypotenuse length of a 3-4-5 triangle.

MAGNITUDE(W1)
returns a new series corresponding to the magnitude component of the original series, whether Polar or Cartesian form.

SEE ALSO: REAL IMAGINARY
ANGLE (*Trig Functions*) PHASE

MAX(series)

PURPOSE: Calculates the maximum value of a series.

series (Optional). A series or table. Defaults to the current window.

RETURNS: A number.

SEE ALSO: MIN COLMAX
CLIP FMAX
FPEAK FMIN

MEAN(series, first, points)

PURPOSE: Calculates the mean value of a series.

series (Optional). A series or table. Defaults to the current window.

first (Optional). The first point to include in the calculation of the mean. The default is the first point.

points (Optional). The number of points to include in the calculation starting from the provided *first* point. The default is to the end of the series.

RETURNS: A number.

SEE ALSO: AVGS STATS
STDEV MEDIAN
RMS

MEDIAN(series)

PURPOSE: Returns the median of a series.

series (Optional) A series or table. Defaults to the current window.

RETURNS: A number.

EXAMPLES: MEDIAN(GSER(2,8,4,32,16))
returns 8, because there are two observations greater than eight and two lower.
MEDIAN(GSER(2,8,4,32))
returns 6.0, the average of the two middle values.

REMARKS: MEDIAN must sort its input, which can be slow for large amounts of data.

SEE ALSO: COLMEDIAN
MEAN
STDEV

MENUCLEAR(num)

PURPOSE: Clears the menus from the screen.

num (Optional). The number of menus to clear. If num is positive, the same number of menus are cleared from the screen. If num is negative, all but that number of menus are cleared. The default is to clear all menus.

RETURNS: Nothing.

EXAMPLES: MENUCLEAR()
clears all the menus from the screen.

MENUCLEAR(-2)

This clears all but 2 menus from the screen.

REMARKS: The parenthesis must be included, regardless of whether or not num is supplied.
The statement MENUCLEAR (); should precede macro specifications within menus. MarketBrowser does not automatically clear menus off the screen.

For example, if you wished to enable a user to execute the MAX function from the root MarketBrowser menu, you would add the following line to "automenu.mnu":
MAXIMUM VALUE ~ MENUCLEAR();MAX

SEE ALSO: MENUFILE MENULIST
 MENUPRINT INPUT
 VIEWFILE

MENUFILE(x, y, text_color, bg_color, filename, useviewfont)

PURPOSE:	Reads a text menu file and displays the menu on the screen in accordance with the specified parameters.	
x	(Optional). X-coordinate in text columns. The default is centered.	
y	(Optional). Y-coordinate in text rows. The default is centered.	
text_color	(Optional). The color of menu text. The default is white.	
bg_color	(Optional). The background color of menu. The default is red.	
filename	The file name of the menu file, in quotes.	
useviewfont	(Optional). When 1, forces the menu to use the VIEWFILE font, which is typically of fixed width. This is useful for getting character formatted displays to line up.	
RETURNS:	Nothing.	
EXAMPLE:	MENUFILE(0,0,WHITE,LBLUE, "menu1.men") reads the menu file "menu1.men" and displays it in white on light-blue in the upper-left corner of the worksheet.	
SEE ALSO:	MENUCLEAR MENUPRINT VIEWFILE	MENULIST INPUT

MENUINCALLBACK()

PURPOSE:	Determines if a function has been called as a result of a user interaction.
RETURNS:	1 if the function where MENUINCALLBACK() is being used is being run as the result of user interaction, 0 if it is not.
EXAMPLE:	Examples of the use of this function can be found in numerous functions in the MarketBrowser program file pickd.xpl.
REMARKS:	This function can be useful when a pull-down list within a menu should display something different (such as the full list of choices) when a user causes it to be displayed by clicking on it vs. when the pull-down list is simply displayed by the program (where you may wish to display a default or a choice that is correct for the current window). Typically, MENUINCALLBACK() would be used with an XPL function that controls what is displayed in the pull-down list and not within the menu text file itself.

MENULIST(x, y, text_color, bg_color, options)

PURPOSE: Generates a pop-up menu at the worksheet level of MarketBrowser in accordance with the specified parameters.

x (Optional). X-coordinate in text columns. The default is centered.

y (Optional). Y-coordinate in text rows. The default is centered.

text_color (Optional). The color of menu text. The default is white.

bg_color (Optional). The background color of menu. The default is red.

options Menu selection options in quotes.

RETURNS: Nothing.

EXAMPLES: `MENULIST(0,0,WHITE,LBLUE," MIN ~MIN(W1)"," MAX ~MAX(W1)")`

pops up a white on light-blue menu in the upper-left corner of the screen.

`MENULIST(0, 2, " MIN ~ECHO(STRCAT('MIN. of W1 =',STRNUM(MIN(W1))))")`

pops up a menu centered on the x-axis and 2 rows down in the default colors (white text, red background) with one selection ('MIN').

SEE ALSO: `MENUCLEAR` `MENUFILE`
`MENUPRINT` `INPUT`
`VIEWFILE`

MENUPRINT(x, y, filename)

PURPOSE: Reads a text menu file and prints the menu to a file, rather than the screen.

x (Optional). X-coordinate in text columns. The default is centered.

y (Optional). Y-coordinate in text rows. The default is centered.

filename The file name of the menu file, in quotes.

RETURNS: Nothing.

EXAMPLE: `MENUPRINT("MENU1.MEN")`

reads the menu file 'MENU1.MEN,' evaluates it, and writes the result to a file. By default, the output file will be named "menu1.prn". The user is given an opportunity to override this default name before the file is written.

SEE ALSO: `MENUCLEAR`
`MENUFILE`
`MENULIST`
`INPUT`
`VIEWFILE`

MENUREPOP(action)

- PURPOSE:** Signals a request for a repaint of the current menu.
- action** Integer. 1 to do the repaint; 0 to do nothing.
- REMARKS:** This function is typically used from an XPL function that provides services for the "@u" user configurable panel button
- SEE ALSO:** MENUFILE
MENUCLEAR

MENURETURN(action)

- PURPOSE:** Used to stop processing steps inside user-defined panels
- action** (Optional). Integer. The default is 1.
- 0 = No effect.
 - 1 = Stops processing and causes panel to clear.
 - 2 = Stops processing but does not cause panel to clear.
- RETURNS:** Nothing.
- REMARKS:** Used to test conditions in panels and allow users to make changes to input values and try again.
- SEE ALSO:** MENUFILE
MENUCLEAR

MERGE(series1, ..., seriesn, n)

- PURPOSE:** Splices two series.
- series1, ..., seriesn** Any number of series or tables.
- n** An optional "no-pad" argument. The integer 0 is the only legal value.
- RETURNS:** A new series where the first point is point 1 of series 1, the second point is point 1 of series 2, the third point is point 1 of series 3, etc.
- EXAMPLES:** MERGE(W1, W2, W6, (GCOS(100, 0.1), 0)
- creates a new series by splicing the series in window 1, window 2, window 6, and a generated cosine wave in the method described above but will not pad unequal sized series with zeros.
- To merge series in six consecutive windows, (W3's series is 100 points long while the rest have 75 points each) type:
- MERGE(W3..W8)

This merges the series in windows 3 through 8 and pads the series in windows 4 through 8 with zeros to the length of 100 points.

REMARKS: MERGE operates on any number of input series. Input series can be real or complex; MERGE returns a complex series if any of the input series are complex. If the merged series have different lengths, MarketBrowser pads the series with zeros to the length of the longest series. The optional argument 0 will suspend the padding function. If anything other than two dots separates the defined set of windows, e.g. MERGE(W3....W8), MarketBrowser will not perform the command.

SEE ALSO: CONCAT
REPLICATE
DECIMATE
REMOVE

MESSAGE(titlebar, message, iconstyle)

PURPOSE: Displays a message box, and allows the user to select OK or Cancel, using the native GUI.

titlebar String enclosed in quotes.

message String enclosed in quotes.

iconstyle An integer. Options are:

- 1 - Question (OK/Cancel)
- 2 - Warning (OK/Cancel)
- 3 - Advice
- 4 - Error description
- 5 - Question (Yes/No)
- 6 - Warning (Yes/No)
- 7 - Question (Yes/No/Cancel)

RETURNS: An integer: 1 if 'OK/Yes' was selected, 0 if 'Cancel/No' was selected. If iconstyle is 7, a -1 is returned if cancel is selected.

EXAMPLE: MESSAGE("MONITOR ERROR", "The server went down. Try to reconnect?", 1)

REMARKS: The type of message box that appears is system dependent.

SEE ALSO: GETSTR
PICKLIST
PICKFILE

MIN(series)

PURPOSE: Calculates the minimum value of a series.

series (Optional). A series or table. Defaults to the current window.

RETURNS: A number.

SEE ALSO: MAX MIN
COLMIN FMIN

MKDIR(directory, behavior)

PURPOSE: Creates a directory.

directory String. The name of the directory to create.

behavior (Optional). Integer. 0 (default) - don't inform the end user if the directory already exists; 1 - inform the end user if the directory already exists.

RETURNS: 1 if the creation is successful, 0 if it is not.

EXAMPLE: MKDIR("c:\expo\mydir")

REMARKS: If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).

SEE ALSO: GETPATH
DIREXISTS
RMDIR

MMULT(matrix1, matrix2)

PURPOSE: (A macro). Multiplies two matrices.

matrix1 A matrix.

matrix2 A matrix.

RETURNS: A matrix.

EXAMPLE: MMULT(W1, W2)

SEE ALSO: INNERPROD
INTERPOSE
INVERSE
OUTERPROD
SVD

MONITOR(symbol, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, gap_2_end, interval, paint_tick, update, add_nas, inside, na_interp)

PURPOSE:	Monitors an instrument as a line chart. The values that MONITOR plots are the close values of a user-defined real-time interval. It also allows for the fine-tuning of the update frequency of the bars, and for automatic extraction and cleaning of historical data via MarketBrowser's API.
symbol	String. Valid market symbol and its field.
start_date	(Optional). Quoted string of the form "mm/dd/yy", which represents the date from which to start extracting data from symbol . Use "" (a pair of empty quotes) to leave the default, which is the start date of symbol .
start_time	(Optional). Quoted string of the form "hh:mm:ss", representing the time to start extracting data from symbol . To leave the default (the start time of symbol), use "" as a placeholder.
end_date	(Optional). Quoted string of the form "mm/dd/yy" representing the date on which to stop extracting data from symbol . Use "" to leave the default, which is the end date of symbol .
end_time	(Optional). Quoted string of the form "hh:mm:ss", representing the time to stop extracting data from symbol . To leave the default (the end time of symbol), use "" as a placeholder.
gap_1_start	(Optional). Quoted string of the form "hh:mm:ss", representing the beginning of the first gap in the extraction of data from symbol . To leave the default (00:00:00), use "" as a placeholder.
gap_1_end	(Optional). Quoted string of the form "hh:mm:ss", representing the end of the first gap in the extraction of data from symbol . To leave the default (23:59:59), use "" as a placeholder.
gap_2_start	(Optional). Quoted string of the form "hh:mm:ss", representing the beginning of the second gap in the extraction of data from symbol . To leave the default (00:00:00), use "" as a placeholder.
gap_2_end	(Optional). Quoted string of the form "hh:mm:ss", representing the end of the second gap in the extraction of data from symbol . To leave the default (23:59:59), use "" as a placeholder.
interval	(Optional). Integer. Multiple of the underlying real-time interval. Defaults to 1.
paint_tick	(Optional). Integer: 0 = OFF, 1 = ON. Paints a continuously updating bar at the end of the chart which updates on every tick of the instrument but does NOT cause any dependent studies to update. Defaults to 0, or OFF.
update	Integer. Determines how frequently the series, and any series dependent on it, updates. Options are: <ul style="list-style-type: none">• 0 - update every interval.

- 1 - update every (**interval** * real-time interval)
- 2 - every tick and real-time interval

add_nas (Optional). Integer. Type of NA processing. Options are:

- 0 - Do not fill gaps with NAs (default).
- 1 - Fill all gaps with NAs
- 2 - Fill only valid gaps on business days inside of trading hours (represented by **gap_1_start/end** and **gap_2_start/end**).

inside (Optional). Integer. How to process gaps:

- 1 - Keep data INSIDE (within) of **gap_1_start/gap_1_end** and **gap_2_start/gap_2_end**. (default).
- 0 - Keep data OUTSIDE of **gap_1_start/gap_1_end** and **gap_2_start/gap_2_end**.

na_interp (Optional). Integer. Type of interpolation. Options are:

- 1 - Perform linear interpolation through valid gaps
- 0 - Leave gaps. (default)

RETURNS: A series updating in real-time.

EXAMPLE: `MONITOR('IBM.LAST', "", "", "", "", "09:00:00", "17:00:00", "12:00:00", "13:00:00", "", "", 5, 0, 1, 2, 0, 0)`

collects data from 9 AM to 5 PM, with a gap between 12 and 1 PM. Data is collected every 5 minutes, no updating bar is painted at the end of the series, the gap is filled with NAs, and NA values are not interpolated.

REMARKS: MONITOR extracts historical data (and real-time data) using the same mechanism as DTEXTTRACT.

Data extraction is enabled only if the configuration variable EXTRACT_RT_HISTORY is set to 1, or TRUE. NA filling and interpolation is only performed upon historical data.

Use **update** option 2 with care when monitoring fast-ticking items.

SEE ALSO: BARMON
DTEXTTRACT

MOUSEROTATE

- PURPOSE:** Activates mouse-driven axes rotation of a PLOT3D graph.
- RETURNS:** Nothing.
- REMARKS:** To select an axis of rotation, keep the left mouse button pressed while moving the mouse across the desired axis. Once selected, the XYZ axes will rotate as the mouse is dragged and the left button remains pressed. To redraw the graph, release the left button. To quit mouse rotation, hit [ESC] or press the right mouse button.
- SEE ALSO:** PLOT3D
ROTATE3D

MOVAVG(series, points, rampflag, sum_only, type, factor, perform_on, lag, lag_amt)

- PURPOSE:** "Smooths" a series by averaging around each point.
- series** A window or variable reference.
- points** Integer. The number (*n*) of points to average as the series is processed.
- rampflag** (Optional). Integer flag. Method of calculating first *n* points in the series (see the Example section for more information). Options are:
- 0 - show "ramp-in" edge of moving average
 - 1 - Average by *n*, even for first *n* points (default).
- sum_only** (Optional). Integer flag, which if set to 1, or TRUE, produces a moving sum (rather than a moving average). Defaults to 0.
- type** (Optional). Type of moving average to perform. The options are:
- 0 - Normal (default)
 - 1 - Modified
 - 2 - Weighted
 - 3 - Exponential
 - 4 - Hamming
 - 5 - Hanning
 - 6 - Kaiser
 - 7 - Modified Exponential
 - 8 - Modified Exponential as described by J. Welles Wilder in his book *New Concepts in Technical Trading Systems*
- factor** (Optional). The decay factor if type is 3, 7, or 8. Otherwise factor is ignored.
- perform_on** (Optional). Integer. If input series consists of multiple columns (e.g., CHLO bars), this number indicates on which series to perform the moving average. Defaults to 0, i.e., the first valid series.
- lag** (Optional). Integer flag.

- 1 - lag the moving average
- 0 - do not lag (default)

lag_amt (Optional). Real number. If lag is 1, or TRUE, the amount by which to lag the moving average. If lag_amt is set to 0, then the moving average is lagged by 1/2 of the length of the input series.

RETURNS: A series or table.

EXAMPLES: Given the formula:

W1: GLINE(10,1,1,1)

which has as values [1,2,3,4,5,6,7,8,9,10]

W2: MOVAVG(W1,3,1)

produces a 3-point moving average of the series in window 1, with the values [1,1.5,2,3,4,5,6,7,8,9].

W3: MOVAVG(w1,3,0)

produces a 3-point moving average with the values [0.33,1,2,3,4,5,6,7,8,9].

For a simple moving average, the rampflag argument changes the formula used to calculate the moving average in this way:

MOVAVG(s, 3, 0) returns the series:

$((s_1)/3, (s_1*s_2)/3, (s_1+s_2+s_3)/3, (s_2+s_3+s_4)/3, \dots (s_{n-2}, s_{n-1}, s_n)/3)$

while MOVAVG(s, 3, 1) returns the series:

$((s_1)/1, (s_1+s_2)/2, (s_1+s_2+s_3)/3, (s_2+s_3+s_4)/3, \dots (s_{n-2}, s_{n-1}, s_n)/3)$

REMARKS: A moving sum is calculated with the same moving boxcar logic as applied to a moving average.

SEE ALSO: MOVMAX
MOVMIN
CONV
CONV2D
IIRFIR

MOVE(offset)

PURPOSE:	Moves the cursor by an offset x-axis units from the current cursor position.
offset	How far to move in x-axis coordinates (must be positive or negative integer or a real number)
RETURNS:	Nothing.
EXAMPLE:	MOVE(5.0) moves the cursor 5.0 x-axis units.
REMARKS:	This function moves cursor by x-axis units whereas NMOVE moves cursor by a number of data points.
SEE ALSO:	NMOVE NPUT PUT

MOVEFILE(file1, file2, behavior)

PURPOSE:	Moves a file to a new location or name without copying it.
file1	String. The existing source file.
file2	String. The destination file.
behavior	(Optional). Integer. <ul style="list-style-type: none">• 0 (default) - don't overwrite destination file if it already exists• 1 - confirm before overwriting destination file if it already exists• 2 - overwrite destination file without confirmation if it already exists
RETURNS:	An error if file1 does not exist, 1 if the move is successful, 0 if it is not.
EXAMPLE:	MOVEFILE("c:\expo\data.dat","c:\mydata\data.dat",2)
REMARKS:	If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).
SEE ALSO:	GETPATH DELFILE COPYFILE FILEEXISTS

NAFILL(series, style)

PURPOSE:	Replaces NA values based on other known data points.
series	(Optional). A series or table. Defaults to the current window.
style	(Optional). The method used to "fill" NA values. <ul style="list-style-type: none">• 0 = No fill; retain all NA values (default)• 1 = Fill forward, using the last known value• 2 = Fill forward, then backward• 3 = Fill backward, using the next known value• 4 = Fill backward then forward
RETURNS:	A series or table.
EXAMPLES:	<p>NAFILL(curr, 1)</p> <p>replaces each NA value with the preceding known value; leading NAs are not replaced.</p> <p>NAFILL(curr, 2)</p> <p>replaces each NA value with the preceding known value, then fills backward to replace any leading NAs.</p>
REMARKS:	Note that the default is for NAFILL not to replace NA values. To replace NA values, supply the "style" argument and give it a value other than 0.
SEE ALSO:	ISNAVALUE NAVALUE SETNAVALUE

NAVALUE

PURPOSE:	The value actually used to represent NAs in numeric data.
RETURNS:	A real number.
EXAMPLE:	<p>GSER(1, 2, NAVALUE, 4, 5)</p> <p>generates a series with the third element set to NA.</p>
REMARKS:	When editing data in a worksheet from within a tabular view, NAVALUE may be used to overwrite erroneous data points.
SEE ALSO:	ISNAVALUE SETNAVALUE NAFILL

NBEIGVAL(matrix)

PURPOSE: Computes the Eigenvalues of a square matrix without a preliminary balancing step.

matrix A real or complex square matrix.

RETURNS: A series with as many rows as the input matrix. Each entry in the series is an Eigenvalue. The Eigenvalue in row n of NBEIGVAL corresponds to the Eigenvector in column n of NBEIGVEC.

EXAMPLE: x =

0 + 8i	0	1 + i
0	1001	0 + 3i
90	0 + i	200

NBEIGVAL(x) =

-0.43153 + 7.5348i
200.44 + 0.46525i
1001 - 4.2384e - 07i

REMARKS: EIGVAL and EIGVEC first perform a balancing step in which the rows and columns are transformed; this ensures that root mean squares are as close as possible while Eigenvalues and Eigenvectors are left unchanged.

In most cases, this improves the accuracy of EIGVAL and EIGVEC, but in some cases it does not. BALANCE can be used to check that relatively small matrix elements have not become unduly magnified by the balancing step. If they have, then NBEIGVAL and NBEIGVEC are likely to yield better results.

SEE ALSO: BALANCE
EIGVAL
EIGVEC
NBEIGVEC

NBEIGVEC(matrix)

PURPOSE: Computes the Eigenvectors of a square matrix without a preliminary balancing step.

matrix A real or complex square matrix.

RETURNS: A square matrix of the same dimensions as the input matrix. Each column of the output matrix is an Eigenvector. The Eigenvector in column n of NBEIGVEC corresponds to the Eigenvalue in row n of NBEIGVAL.

EXAMPLE: x =

0 + 8i	0	1 + i
0	1001	0 + 3i
90	0 + i	200

NBEIGVEC(x) =

1.0e-06*908850.0-80005.0i
6010.3-4832.4i
1.2636-1.2308i52.355+1225.2i
593.09+4055.7i
-1.0e06-5024.7i
-408870.0+20554.0i
- 1082200+158900i
6.4151-1248.6i

REMARKS: EIGVAL and EIGVEC first perform a balancing step in which the rows and columns are transformed; this ensures that root mean squares are as close as possible while Eigenvalues and Eigenvectors are left unchanged.

In most cases, this improves the accuracy of EIGVAL and EIGVEC, but in some cases it does not. BALANCE can be used to check that relatively small matrix elements have not become unduly magnified by the balancing step. If they have, the NBEIGVAL and NBEIGVEC are likely to yield better results.

SEE ALSO: BALANCE
EIGVAL
EIGVEC
NBEIGVAL

NEATEN

- PURPOSE:** Fills the gaps between windows that have been manually resized.
- RETURNS:** A custom layout with windows that have been adjusted to fit tightly together.
- REMARKS:** This function plays the same role as the Arrange pull-down.
- SEE ALSO:** ROWLAYOUT
COLLAYOUT
TILE

NEGATE(series)

- PURPOSE:** Creates a series which is the arithmetic negative of an input series.
- series** A series, table or number.
- RETURNS:** The arithmetic negative of the input series.
- REMARKS:** Equivalent to $-1(\text{series})$, the arithmetic negative of series. Avoids the grouping problems sometimes encountered with the unary minus.

NFORMAT(control, values)

- PURPOSE:** Formats a list of numbers.
- control** Control string conforming to C language printf specifications, containing only real number field specifiers.
- values** A list of real numbers.
- RETURNS:** A string.
- EXAMPLE:** NFORMAT("Max: %4.2f Min: %4.2f", max, min)
produces a string like "Max: 47.20 Min: 22.03"
- REMARKS:** See any standard C language reference for further information.
- SEE ALSO:** ANYFORMAT
SFORMAT

NUMITEMS(window, series index)

PURPOSE:	Counts the number of items in a window.
window	(Optional). Window reference. Defaults to current window.
series index	(Optional). Series number. Defaults to the first (or only) series.
RETURNS:	An integer number representing the number of items in a particular window or series.
EXAMPLE:	NUMITEMS(W11) returns the number of items in window 11.
REMARKS:	Compare with NUMCOLS, which counts actual data series. If the current window contains two Close/High/Low/Open traces, the number of items is 2, while the number of columns is 8.
SEE ALSO:	NUMCOLS NUMOVERLAYS

NUMOBSV (series)

PURPOSE:	Returns the count of the number of observations in a series or table that are not NA values.
series	A series or table. Defaults to the current window.
RETURNS:	An integer. The length of the series, minus any NA values.
EXAMPLE:	NUMOBSV(MONITOR('IBM')) returns the total number of points generated by the real-time monitor that are not NA values.
SEE ALSO:	LENGTH COLNUMOBSV

NUMOVERLAYS(window, stacked)

PURPOSE:	Counts the number of overlays in a window.
window	(Optional) Defaults to the current window.
stacked	(Optional) When set to 1 counts only the number of "stacked" (partitioned) overlays.
RETURNS:	The number of overlays in the window.

NUMROWS(table)

PURPOSE: (A macro). Calculates the number of rows in a table.

table (Optional). A table. Defaults to the current window.

RETURNS: A number.

**EXPAN-
SION:** INT(MAX(COLLENGTH(M)))

SEE ALSO: SERCOUNT

NUMSTR(str)

PURPOSE: Converts a string into a number.

str A string.

RETURNS: A number.

EXAMPLE: NUMSTR(STRFIND("XINC", "YOR:12.3 XINC:1.0 YREF:120.0"))
returns: 1.0.

SEE ALSO: STRNUM

NUMWINDOWS

PURPOSE: Returns the number of windows currently contained in the worksheet, whether they have formulas in them or not.

RETURNS: An integer.

SEE ALSO: GETWNUM

OFF

- PURPOSE:** (A Macro) Returns the integer 0.
- RETURNS:** The integer 0.
- REMARKS:** The OFF macro is simply a more readable way of specifying the value 0.
- SEE ALSO:** ON

ON

- PURPOSE:** (A Macro) Returns the integer 1.
- RETURNS:** The integer 1.
- REMARKS:** The ON Macro is simply a more readable way of specifying the value 1.
- SEE ALSO:** OFF

ONEXIT(commands)

- PURPOSE:** Runs specified function(s) just before an XPL function is exited (normally or abnormally).
- commands** A list of strings separated by commas. The command(s) to be run when the function is exited.
- RETURNS:** Nothing.
- EXAMPLE:** The following excerpt turns on the hourglass cursor at the start of the function and uses ONEXIT to turn it off when the function is finished.
- ```
myxpl()
(
ONEXIT("WAITCURSOR(0)");
WAITCURSOR(1);
...
)
```
- REMARKS:** The commands argument can reference local variables if you wish. For example:
- ```
myxpl()
(
_my_localvar = GETCONF("EVAL_NEVER_FAIL");
ONEXIT("SETCONF('EVAL_NEVER_FAIL', _my_localvar)");
...
)
```
- SEE ALSO:** GETCONF
SETCONF
WAITCURSOR

ONPLOT(formula)

PURPOSE:	Evaluates formula at the time window is being redrawn; typically used to add text and drawings to a window to enhance data display.
formula	Formula to evaluate.
RETURNS:	Typically nothing. Depends on the contents of the formula.
REMARKS:	Similar to EVAL, the argument is a formula in quotes. ONPLOT allows arbitrarily complex text and drawings to be added to a window. Typically the formula argument would be a call to some user-defined routine that would perform some complicated custom renderings as the window is redrawn. ONPLOT would be called within ADDFORM.
SEE ALSO:	TEXTANN LINEANN EVAL

OUTERPROD(series1, series2, op)

PURPOSE:	Computes the outer product of two vectors.
series1	A series or table.
series2	A series or table.
op	Quoted string containing a binary operator.
RETURNS:	Matrix with as many rows as series1 and as many columns as series2.
EXAMPLE:	OUTERPROD(QUANTITY, PRICE, "*") If QUANTITY and PRICE are series, this expression results in a table of COSTS.
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR".
SEE ALSO:	INNERPROD REDUCE INTERPOSE COLREDUCE ROWREDUCE MMULT

OVERLAY(series, target, color, sync, staggery, scales, ticks, partition, span_y_b, span_y_t, ymin, ymax)

PURPOSE:	Overlays a series into a window.
series	The series to overlay.
target	(Optional). The window in which series is overlaid. Defaults to the current window.
color	(Optional). Color of the overplotted series
sync	(Optional). Sync mode. How the overlaid series scrolls along the horizontal and vertical axes in relation to the window's primary series. See a complete table of synchronization options under SYNC. Defaults to 0, or no sync.
staggery	(Optional). Integer flag. Options are: <ul style="list-style-type: none">• 1 - Stagger y-axis scales vertically (default).• 0 - Keep y-axis scales flush with plotting area.
scales	(Optional). Scale setting. Defaults to 5 (x top y right). See the SCALES function for a complete description of available options.
ticks	(Optional). Number of ticks marks to place along the y-axis.
partition	When PARTITION is 1, the four arguments SPANYBOTTOM, SPANYTOP, YMIN and YMAX are interpreted as defining a hard partition of the drawing area of the MarketBrowser window. SPANYBOTTOM and SPANYTOP are used to set the Y-AXIS span. If they are set to 0.0, and 0.0 this indicates that the AXIS should autocalculate itself based on the data in that subwindow. Setting these to 0.0 and 100.0 would force the y-axis scales to have that range. YMIN and YMAX are used to set the portion of the MarketBrowser window that the subwindow should occupy. YMIN and YMAX are reals ranging from 0.0 to 1.0. Setting YMIN = 0.0 and YMAX = 0.5 would make the subwindow occupy the bottom half of the MarketBrowser window.
span_y_b	(Optional). Lower end of the y-axis scale's range.
span_y_t	(Optional). Top of the y-axis scale's range.
ymin	(Optional). Real number spanning from 0.0 to 1.0, representing the lower placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
ymax	(Optional). Real number spanning from 0.0 to 1.0, representing the upper placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
RETURNS:	Nothing.
EXAMPLE:	Given the following window formulas:

```
W1: READAHIST('IBM.CLS','D',2,1)
W2: MOVAVG(W1,15)
W3: W1;OVERLAY(W2,black,1,0,5,10,-1, -1,0.1,0.3)
```

copies the series in W1 into the window, and transparently overlays the series from W2. The two series scroll together along the horizontal and vertical axes, the scales of the overlay are kept flush with the plotting area, the default scale style is used, 5 tick marks are plotted along the y-axis, and the overlay is contained between 0.1 and 0.3 of the y-axis.

REMARKS: You can overlay an unlimited number of series into a single window. Each overlay has an independent set of scales associated with it. The concept of focus applies to overlaid series. Specify which series in a window is the current focus either with the FOCUS function, or by clicking on the series' scale.

SEE ALSO: FOCUS
OVERPLOT
SYNC
OVERLAYCMD

OVERLAYCMD(series, target, color, sync, staggery, scales, ticks, span_y_b, span_y_t, ymin, ymax, command1, ..., commandn)

PURPOSE:	Overlays a series into a window as per the OVERLAY function, and applies an MarketBrowser formula to the overlaid series.
series	The series to overlay.
target	(Optional). The window in which series is overlaid. Defaults to the current window.
color	(Optional). Color of the overplotted series
sync	(Optional). Sync mode. How the overlaid series scrolls along the horizontal and vertical axes in relation to the window's primary series. See a complete table of synchronization options under SYNC. Defaults to 0, or no sync.
staggery	(Optional). Integer flag. Options are: <ul style="list-style-type: none">• 1 - Stagger y-axis scales vertically (default).• 0 - Keep y-axis scales flush with plotting area.
scales	(Optional). Scale setting. Defaults to 5 (x top y right). See the SCALES function for a complete description of available options.
ticks	(Optional). Number of ticks marks to place along the y-axis.
span_y_b	(Optional). Lower end of the y-axis scale's range.
span_y_t	(Optional). Top of the y-axis scale's range.
ymin	(Optional). Real number spanning from 0.0 to 1.0, representing the lower placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
ymax	(Optional). Real number spanning from 0.0 to 1.0, representing the upper placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
command1, ..., commandn	(Optional). String in quotes. An MarketBrowser expression which is applied to the series that is being overlaid.

RETURNS: Nothing.

EXAMPLE: Given the following window formulas:

```
W1: READAHIST('IBM.CLS','D',2,1)
```

```
W2: MOVAVG(W1,15)
```

```
W3: W1;OVERLAYCMD(W2,"setcomment('movavg of IBM CLOSE')")
```

copies the series in W1 into the window, and transparently overlays the series from W2. It then sets the comment for the overlaid series.

REMARKS: You can overlay an unlimited number of series into a single window. Each overlay has an independent set of scales associated with it. The concept of focus applies to overlaid series. Specify which series in a window is the current focus either with the FOCUS function, or by clicking on the series' scale.

SEE ALSO: OVERLAY
OVERPLOT CMD
FOCUS
SYNC

OVERPLOT(series1, ..., seriesn)

PURPOSE: Overplots any number of series in the current, active window.

series1 The first series to overplot.

seriesn The nth series to overplot.

RETURNS: Nothing.

EXAMPLES: OVERPLOT(W7)
plots the series from W7 on top of any series in the current window.
OVERPLOT(0)
clears all the overplotted series in the current window.

REMARKS: Any overplotted series remains in the window during scrolling, zooming, or exiting the window. When the cursor is on, you may switch the cursor between the overplotted series and the original by using the up and down arrow keys.

SEE ALSO: OP (macro shorthand)
UNOVERPLOT
OVERLAY
OVERPLOT CMD

OVERPLOTCMD(series, command1, ..., commandn)

- PURPOSE:** Overplots a series in the current, active window, and applies the MarketBrowser expressions specified in the command arguments to the overplotted series.
- series** The series to overplot. A window or variable reference.
- command1,..., commandn** (Optional). String in quotes. An MarketBrowser expression applied to the overplotted series.
- RETURNS:** Nothing.
- EXAMPLE:** Given the following formulas:
W1:MONITOR('IBM.LAST')
A := MOVAVG(w1,10)W2:
W1;OVERPLOTCMD(A,"setcomment('movavg of w1')")
copies the series in W1 into W2, then overplots the moving average (a) into the window. It also sets the comment for the moving average series.
- REMARKS:** This function is similar to the OVERPLOT function. However, unlike OVERPLOT, it does not take multiple series arguments, i.e., it can only overplot one series at a time. Any overplotted series remains in the window during scrolling, zooming, or exiting the window. When the cursor is on, you may switch the cursor between the overplotted series and the original by using the up and down arrow keys.
- SEE ALSO:** OVERPLOT
OVERLAYCMD

PARABOLIC(series)

- PURPOSE:** Computes the Parabolic SAR of the input series
- series** A valid window or variable reference (variable must contain a series).
- RETURNS:** A series.
- REMARKS:** See standard technical analysis literature for a discussion of how the SAR is calculated.
- SEE ALSO:** GENSTUDY

PARTPROD(series)

- PURPOSE:** Calculates the partial (cumulative) product of a series, as of performance returns for example.
- series** A series or table.
- RETURNS:** A series or table.
- EXAMPLE:** PARTPROD(W2)
creates a new series containing the partial products of the series points in the windows.
- REMARKS:** The partial product Y, of a series X is equal to the following:
 $Y_1 = X_1$
for all $i > 1$,
 $Y_i = (X_i + 1) * (Y_{i-1} + 1) - 1$
Note the +1 and -1 terms. This form is convenient for producing “cumulative returns” from a series of period-over-period returns.
To produce a partial product without the effect of the +1 and -1 terms, you can use
PARTPROD(W2 - 1) + 1
or
INTERPOSE(W2, “*”)
- SEE ALSO:** INTERPOSE
PARTSUM

PARTSUM(series)

- PURPOSE:** Produces a new series that is the partial sum of any series or table.
- series** A series or table.
- RETURNS:** A series or table.
- EXAMPLES:** PARTSUM(W2)
creates a new series containing the partial sums of the series points in window 2.
If window 2 contains real-time volume data,
PARTSUM(W2)
charts the cumulative volume over the day.
- REMARKS:** The values of each point i in the new series are defined as the sum of values of all points beginning with the first point up to the i th point.
- SEE ALSO:** PARTPROD

PASS(form)

PURPOSE: Evaluates a formula.

form Formula to evaluate

RETURNS: Depends on the contents of the formula.

REMARKS: If the formula is in quotes, it is not evaluated but is simply passed along as a string. If the fomula is not in quotes then it is evaluated similar to EVAL.

SEE ALSO: CAST
EVAL
WHILE

PATHCHAR

PURPOSE: Returns the path character of the current operating system.

RETURNS: A character.

PCTCHANGE(hotvar)

PURPOSE: Calculates the percentage change between the current value of a scalar numeric hot variable and it's previous value.

hotvar A hot variable reference.

RETURNS: The percentage change between the hot variable's current value and it previous value.

EXAMPLE: Given the following hot variable:
mydata:= RTQUOTE("IBM.LAST")
PCTCHANGE(mydata)
returns the percentage change between the two latest points.

SEE ALSO: CHANGE
PRIOR

PCTSTACK

- PURPOSE:** Creates a stack chart, annotated with the percent contribution of each element.
- RETURNS:** Nothing.
- REMARKS:** This display mode should be used with positive real data, and for legibility it works best with a small number of points.
- SEE ALSO:** BARS
LINES
STACK
STICKS
TABLEVIEW
TICKFORM

PERCONVERT(timeseries, period)

- PURPOSE:** Converts historical data to trading bars/candlesticks of a selected periodicity.
- timeseries** A series or trading bars/candlesticks.
- period** Periodicity, in quotes:
- “D” Daily
 - “Wk” Weekly
 - “Mo” Monthly
 - “Qtr” Quarterly
 - “Yr” Yearly
- RETURNS:** Four columns of data, treated as trading bars/candlesticks.
- EXAMPLE:** PERCONVERT(W1, “Wk”)
converts a daily time series into weekly trading bars/candlesticks.
- REMARKS:** Conversion to smaller units (e.g., weeks to days) is not possible.
- SEE ALSO:** PERCONV (shorthand)
BARCONVERT

PFMON(symbol, boxsize, reversal, allticks, hilopf)

- PURPOSE:** Registers a data item for updating in an accumulated Point and Figure chart.
- symbol** A valid price identifier for your data service, in quotes.
- boxsize** Real. The price range represented by box.
- reversal** Real. The price movement required to indicate a reversal.
- allticks** (Optional). 0 = Do not fill; 1 = Fill all points between reversals. The default is 1.
- hilopf** (Optional). 0 = Use only close data when calculating reversals; 1 = Use high and low columns of bar data, if available, to calculate reversals. The default is 0.
- RETURNS:** A series.
- EXAMPLE:** PFMON("IBM", 2.0, 1,0, 1)
causes a Point and Figure chart to accumulate for IBM.
- REMARKS:** The symbol naming convention depends on the data service being used. If PFMON is used while real-time updating is in effect, MarketBrowser automatically attempts to return a daily history, using the RTHISTORY functions. If the allticks option is set to 1, PFMON will return an empty Point and Figure chart if the initial data value returned is an NA value.
- SEE ALSO:** POINTFIG

PFORMAT(data, method, reserved, denominator, reduce, trim)

PURPOSE:	Returns a string formatted as a decimal or fraction.
data	Series reference or number. If data is a series, PFORMAT formats the value of the last observation in the series.
method	Integer. The desired format. Choices are: <ul style="list-style-type: none">• 1 - DECIMAL• 2 - FRACTION• -1 - DEFAULT, as defined in the FORMAT_AS_FRACTIONS configuration variable.
reserved	Argument reserved for future use by LMT. Use -1 as a placeholder.
denominator	(Optional). Integer. Applies to fractional formats. Choices are: 8, 16, 32, 64, 128. Defaults to the value defined in the configuration variable FRAC_DENOMINATOR, which defaults to 8.
reduce	(Optional). Integer flag. Reduce fractions such as 16/32 to 1/2. Defaults to the value specified in the configuration variable FRAC_REDUCE, which defaults to 0, or FALSE.
trim	(Optional). Integer flag. Set to 1, produces a shortened form of fractional representations, for example 101 13, instead of 101 13/32. Defaults to 0, or FALSE.
RETURNS:	A formatted string.
EXAMPLE:	Given the formula: A:= MONITOR("IBM.LAST"), PFORMAT(A, 2, -1, 32, 1, 0) formats the last point in the series as a string (in fractional form), with a denominator of 32, and the reduce flag set to TRUE.
REMARKS:	If you set method to be -1, it will inherit the value specified in the configuration variable FORMAT_AS_FRACTIONS. If it is set to 0 or 2, it will format values as decimals or fractions, respectively. If it is set to 1, it uses the following logic to determine which formatting to use: <ul style="list-style-type: none">• If the data is a scalar, MarketBrowser will format it as a fraction.• If the data is a series, MarketBrowser will evaluate whether the minimum, maximum, and last value in the series can be formatted as fractions.• If they cannot, MarketBrowser will instead format them as decimals.
SEE ALSO:	FORMAT_AS_FRACTIONS (configuration variable)

PHASE(expression)

PURPOSE: Calculates the phase angle of a complex expression.

expression Any expression resolving to a series or scalar.

RETURNS: A series or a scalar.

EXAMPLE: PHASE (1 + I)
returns the value 0.78539816, which is PI/4

REMARKS: PHASE returns a value from -PI to PI. ANGLE returns a value from 0 to 2*PI.

SEE ALSO: ANGLE & ATAN (Trig function) MAGNITUDE
REAL IMAGINARY
POLAR CARTESIAN

PHI

PURPOSE: (A macro). The “golden mean” $(1 + \text{SQRT}(5))/2$.

RETURNS: 1.61803398874989484820

REMARKS: You can derive this constant by solving the following set of equations:
if $A/B = B/(A+B)$
and $A = 1$
then $B = \text{PHI}$

SEE ALSO: DEG E
LN GAMMA
PI SETDEGREE

PI

PURPOSE: (A Macro). Approximates the value of Π

**EXPAN-
SION:** 3.1415926535897932384626

RETURNS: 3.1415926535897932384626

EXAMPLES: COS(PI)
displays -1.0.
EXP(PI * I)
yields -1.0 + 0.0I.

SEE ALSO: DEG E GAMMA

appearance of the Picklist is system dependent.

SEE ALSO: PICKFILE GETSTR
MESSAGE

PIE(series)

PURPOSE: Displays the data points of a series in a pie chart form.

series (Optional). A series or table. Defaults to the current window.

RETURNS: A pie chart.

REMARKS: The sum of the points in the series is represented by the entire circumference of the circle, and each piece of the circle corresponds to a data point. This display mode should be used only with positive real data, and for legibility it works best with a small number of points. A pie chart will be created for each column of data in the window.

SEE ALSO: BARS LINES
STICKS STEPS
PCTSTACK TABLEVIEW

PLOT(window, title, hpfile, colormode)

PURPOSE: Creates an HPGL file of the current window which can be directly copied to an HPGL compatible plotter or printer.

window (Optional). A window reference. Defaults to the current window.

title (Optional). A window title, in quotes. Defaults to the window formula.

hpfile (Optional). An HPGL output file. Defaults to "hpgl.out".

colormode (Optional). An integer. The color output flag. 1 = Color; 0 = black and white. The default is 1.

RETURNS: Nothing.

REMARKS: PLOT creates an output file, which must then be sent to a hardcopy device (or other software) by site-specific means.

SEE ALSO: PLOTALL PLOTWS
PRINT PRINTALL
PRINTWS PS
PSALL PSWS

PLOTALL(hpfile, colormode)

- PURPOSE:** Creates an HPGL file of all windows in the current worksheet (one per page), that can be directly copied to an HPGL compatible printer.
- hpfile** (Optional). An HPGL output file, defaults to "hpgl.out".
- colormode** (Optional). An integer. The color output flag. 1 = Color; 0 = black and white. The default is 1.
- RETURNS:** Nothing.
- REMARKS:** PLOTALL creates an output file, which must then be sent to a hardcopy device (or other software) by site-specific means.
- SEE ALSO:**
- | | |
|---------|----------|
| PLOT | PLOTWS |
| PRINT | PRINTALL |
| PRINTWS | PS |
| PSALL | PSWS |

PLOT3D(series)

- PURPOSE:** Creates a true perspective plot of multi-column data with XYZ axes.
- series** (Optional). A series or table. Defaults to the current window.
- RETURNS:** A 3D plot.
- REMARKS:** PLOT3D is similar to WATERFALL. The resulting graph can be rotated with the ROTATE3D and MOUSEROTATE functions.
- SEE ALSO:**
- MOUSEROTATE
 - ROTATE3D
 - WATERFALL

PLOTMODE(OnOff)

- PURPOSE:** Turns plotting ON or OFF to facilitate the updating of graphically complex windows.
- OnOff** On = 1; Off = 0.
- RETURNS:** Nothing.
- EXAMPLE:** PLOTMODE(0)
Suppresses plotting until PLOTMODE(1) is issued.
- REMARKS:** This is useful with the semicolon (";") for drawing multi-step windows all at once.
- SEE ALSO:**
- | | |
|--------|-----|
| FREEZE | PON |
| POFF | |

PLOTTYPE(style)

PURPOSE: Controls the display mode of an XY series.

style An integer. The style parameter.

RETURNS: Nothing.

REMARKS: If style = 1, a normal XY plot is displayed. If style = 0, the series is displayed as an overplot of the X and Y components of the series. This function is deprecated.

SEE ALSO: XY

PLOTWS(hpfile, colormode)

PURPOSE: Creates an HPGL file of all windows in the current worksheet (on one page), that can be directly copied to an HPGL compatible printer.

hpfile (Optional). An HPGL output file, defaults to "hpgl.out."

colormode (Optional). The color output flag. 1 = color; 0 = black and white. The default is 1.

RETURNS: Nothing.

SEE ALSO: PLOT PLOTALL
PRINT PRINTALL
PS PSALLPSWS

POFF

PURPOSE: (A Macro). Turns off plotting in the current window.

**EXPAN-
SION:** PLOTMODE(0)

RETURNS: Nothing.

REMARKS: Can save time in redrawing or recalculating worksheets with complicated graphs.

SEE ALSO: PLOTMODE
PON

POINTFIG(series, boxsize, reversal, filled, hilopf)

PURPOSE:	Creates a Point and Figure chart for a price series.
series	A series.
boxsize	Real. The price range represented by box
reversal	Real. The price movement required to indicate a reversal.
filled	(Optional). 0 = Do not fill; 1 = Fill all points between reversals. The default is 1.
hilopf	(Optional). 0 = Use only close data when calculating reversals; 1 = Use high and low columns of bar data, if available, to calculate reversals. The default is 0.
EXAMPLE:	If W1 contains a price chart of IBM then: POINTFIG(W1, 2.0, 1.0, 1) creates a Point and Figure chart for IBM.
SEE ALSO:	PFMON

POINTS

PURPOSE:	Displays the data points of a series as unconnected points.
RETURNS:	Nothing.
SEE ALSO:	BARS LINES STICKS TABLEVIEW TICKFORM

POLAR(expr)

PURPOSE:	Converts an input value to magnitude/phase form.
expr	Any expression evaluating to a series, table, integer, real or complex number.
RETURNS:	Complex series or scalar.
EXAMPLES:	POLAR(GSIN(20, .05, 1)) creates a 1 Hz sine wave consisting of 20 points spaced every 0.05 radians apart. The value of each point in the sine wave is a complex number in magnitude/phase form. POLAR(-1) produces a complex number where the magnitude = 1.0 and the phase = pi radians.
REMARKS:	Returns a complex value regardless of the input value.
SEE ALSO:	CARTESIAN CONJUGATE PHASE

POLYFIT(series, order, overwrite, filename)

PURPOSE:	Performs a least squares fit of a series, returning the fit coefficients.
series	A series or table.
order	An integer. The order of polynomial fit.
overwrite	(Optional). 0 = verify before overwriting file of fit statistics. 1 = overwrite without verifying. The default is 0.
filename	(Optional). A text string, in quotes. The default filename is polyn.fit where n is the nth file of fit statistics.
RETURNS:	coefficients of the power series: $y = a_0 + a_1*x + a_2*x^2 + \dots + a_N*x^N$ where N is the supplied order.
EXAMPLE:	W1: gline(100, .01, 1.0, 1.0)^3 generates a curve to model. W2: POLYFIT(W1, 3) returns a 4 point series with values 1,3,3,1 as the resulting 3rd order coefficients. W3: POLYGRAPH(W2,xvals(W1)) graphs the fitted curve.
REMARKS:	POLYFIT creates a file of "goodness of fit" statistics, including Chi and Chi Square.
SEE ALSO:	POLYGRAPH XVALS

POLYGRAPH(coeff, xdata)

PURPOSE:	Graphs the polynomial with the given coefficients for the data points in the xdata series.
coeff	A series or table of polynomial expression coefficients.
xdata	A series or table of X axis values over which to evaluate the polynomial expression
RETURNS:	A series or table.
EXAMPLE:	POLYGRAPH(gser(1,2),gser(1,2,3,4)) graphs a four point exponential line.
SEE ALSO:	POLYFIT XVALS

PON

PURPOSE:	(A Macro). Turns on plotting in the current widow.
RETURNS:	Nothing.
EXPAN- SION:	PLOTMODE(1)
REMARKS:	Causes the window to redraw.
SEE ALSO:	PLOTMODE POFF

POPTOOL

PURPOSE:	Returns to the previously displayed toolbar.
RETURNS:	Nothing
EXAMPLE:	After accessing a custom toolbar using PUSHTOOL(11), return to the previously displayed toolbar by using POPTOOL.
REMARKS:	The Back button that is automatically added to any custom toolbar performs the POPTOOL function when you click it.
SEE ALSO:	TOOLBAR PUSHTOOL

POPWINDOW(window)

PURPOSE:	Zooms a specified window.
window	Window reference
RETURNS:	Nothing.
EXAMPLE:	POPWINDOW(W3) zooms window 3.
REMARKS:	Unlike ZOOM, POPWINDOW can zoom hidden windows. POPWINDOW also works whether the specified window is activated or not.
SEE ALSO:	UNPOPWIN ZOOM UNZOOM

PREVIEWWIN, PREVIEWWSWIN, PREVIEWALLWIN, PREVIEWINFO

PURPOSE: Displays previews of the images that will be printed when one selects from various print options

RETURNS: Nothing.

SEE ALSO: PRINT
PRINTWS
PRINTALL
INFOPRINT

PRINT(window, title, colormode)

PURPOSE: Expands a window to full screen size and send the image to the default printer.

window (Optional). A window reference to a series. Defaults to the current window.

title (Optional). A string to be printed at the top of the series, in quotes. The default is the window's formula.

colormode (Optional). An integer. The color output flag. 1 = color; 0 = black and white. The default is 0.

RETURNS: Nothing

EXAMPLE: PRINT(W7,"CASH FLOWS")
prints the series contained in window 7 with the title "CASH FLOWS".

REMARKS: MarketBrowser prints a series using the entire screen area. After sending a series to the printer, MarketBrowser returns to the original screen configuration.

SEE ALSO: PRINTALL
PRINTWS
PLOTPTS
INFOPRINT

PRINTALL(title, colormode)

- PURPOSE:** Creates a print of every window in the current worksheet. Expands each window to full screen size one at a time and sends the image (one per page) to the default printer.
- title** (Optional). A string to be printed at the top of the series, in quotes. The default is the window's formula.
- colormode** (Optional). An integer. The color output flag. 1 = color; 0 = black and white. The default is 0.
- RETURNS:** Nothing.
- EXAMPLE:** PRINTALL("MKT WORKSHEET")
creates a full sized print of each series in the current worksheet. If the worksheet contained nine windows, all nine printouts are titled "MKT WORKSHEET".
- REMARKS:** MarketBrowser returns to the original screen configuration after all windows have been printed.
- SEE ALSO:** PRINT
PRINTWS
PLOTPS
INFOPRINTALL

PRINTOPT(legends, titles, wbar, wborder, wmargin)

PURPOSE:	Selects worksheet elements to be visible or hidden from a printout.
legends	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Legends are text annotations in a window.
titles	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Titles are text annotations on the worksheet.
wbar	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wbar specifies the text for the window number, window formula and/or window label.
wborder	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wborder specifies the outer border outline of each window.
wmargin	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wmargin specifies the border outline on the inner window (separating the inner window from the window plotting margin).
RETURNS:	Nothing
EXAMPLES:	<p>PRINTOPT(1,1,0,0,0)</p> <p>leaves legends and titles in the printouts of the worksheet, and disables printing of window bars, borders, and margins.</p> <p>PRINTOPT(-1,-1,1)</p> <p>leaves all settings as they currently are, but enables the printing of the window bars.</p>
REMARKS:	PRINTOPT is useful when formatting a worksheet for presentations, demonstrations, printouts, and custom applications. All parameters are optional integer arguments, defaulting to current values. Use -1 to leave a parameter unchanged.
SEE ALSO:	SCREENOPT LAYOUT

PRINTWS(colormode)

PURPOSE:	Prints the entire worksheet (on a single page) on the system printer.	
colormode	(Optional). An integer. The color output flag. 1 = Color; 0 = black and white. The default is 1.	
RETURNS:	Nothing.	
REMARKS:	Only the windows are printed; no borders, worksheet title, or labeling is printed.	
SEE ALSO:	PRINT PRINTOPT	PRINTALL PLOTPS

PRIOR(hotvar)

PURPOSE: Returns the previous value of a scalar numeric hot variable.

hotvar A hot variable reference.

RETURNS: The hot variable's previous value.

EXAMPLE: Given the following hot variables:
mydata := RTQUOTE("IBM.LAST")
PRIOR(mydata)
returns the previous point stored in the hot variable.

SEE ALSO: CHANGE PCTCHANGE

PRNSCREEN

PURPOSE: Create a print snapshot of a screen.

RETURNS: Nothing

EXAMPLE: GRAND(100, .01); PRNSCREEN
creates a 100 point random series and dumps the screen to the printer.

REMARKS: The default output medium for PRNSCREEN (Postscript, HPGL, or Native) can be set in the file, "expo.cnf".

PROTECT(window, name)

PURPOSE: Isolates a window from any dependencies on other windows. Protects a window from the effects of series propagation.

window (Optional). A window reference. Defaults to the current window.

name A new title, in quotes, replacing the window formula.

RETURNS: Nothing.

EXAMPLE: If W3 equals W1 + W2, then
PROTECT(W3,"NEW INDEX")
will replace the formula in window 3 with the name "NEW INDEX", and if W1 or W2 are altered, W3 will not be affected.

REMARKS: Avoid confusion by protecting a window with a genuinely new name, rather than re-using the window formula, e.g. PROTECT(W3, "W1+W2"). This would make W3 look like a normal window. Instead, use PROTECT(W3, "PROTECTED W1+W2").

SEE ALSO: EDIT

PS(window, title, psfile, colormode)

PURPOSE:	Creates a PostScript file of the current window.	
window	(Optional). The window to print; defaults to the current window.	
title	(Optional). A window title in quotes. Defaults to the current window formula.	
psfile	(Optional). A PostScript output file. Defaults to "post.eps"	
colormode	(Optional). Integer color output flag, 1 = color, 0 = black and white. The default is 1.	
RETURNS:	Nothing.	
REMARKS:	A PostScript file contains a list of commands that reproduce the window on the PostScript printer. PostScript files can also be read by a number of third party packages for complete desktop publishing capabilities.	
SEE ALSO:	PSALL PRINT PRINTWS PLOTALL	PSWS PRINTALL PLOT PLOTWS

PSALL(title, psfile, colormode)

PURPOSE:	Creates a PostScript file of the all the windows (one per page) in the current worksheet.	
title	(Optional). String enclosed in quotes that denotes the window's title.	
psfile	(Optional). PostScript output file. Defaults to "post.eps".	
colormode	(Optional) Integer color output flag, 1 = color, 0 = black and white. The default is 1.	
RETURNS:	Nothing.	
REMARKS:	A PostScript file contains a list of commands that reproduce the windows on the PostScript printer. PostScript files can also be read by a number of third party packages for complete desktop publishing capabilities.	
SEE ALSO:	PLOT PLOTWS PSWS PRINTALL	PLOTALL PS PRINT PRINTWS

PSWS(title, psfile)

PURPOSE: Creates a PostScript file (on one page) of all the windows in the current worksheet.

title (Optional) A text string in quotes that is printed out at the top of the worksheet.

psfile (Optional). PostScript output filename in quotes. Defaults to "post.eps".

RETURNS: Nothing

EXAMPLES: PSWS("My Title")

creates a PostScript file with the title My Title and with the default filename post.eps.

PSWS("", "myfile.eps")

creates a PostScript file without a title and with the filename myfile.eps.

REMARKS: A PostScript file contains a list of commands that reproduce the windows on the PostScript printer. Post Script files can also be read by a number of third party packages for complete desktop publishing capabilities.

SEE ALSO:

PLOT	PLOTALL
PLOTWS	PRINT
PRINTALL	PRINTWS
PS	PSALL

PUSHTOOL(toolbar)

PURPOSE: Displays a specified toolbar.

toolbar The number of the toolbar to display:

- 5 - drawing toolbar
- 6 - data toolbar
- 7 through 10 - reserved for future LMT use
- 11 through 15 - use for user-defined toolbars

EXAMPLE: Add a toolbar button to the main toolbar that access a custom toolbar:

```
TOOLBAR(1, -1, 4, RED, "Custom", "PUSHTOOL(11)")
```

Then add a button to that new toolbar #11:

```
TOOLBAR(11, -1, 4, RED, "Stats", "display(w1..w4)")
```

REMARKS: Any custom toolbar automatically has a Back button as its first button. Clicking this button takes you back to the previously displayed toolbar (the same as the POPTOOL function).

SEE ALSO: TOOLBAR
POPTOOL

PUT(x-units)

PURPOSE: Places the cursor on a specified amount of x units from the beginning of the current series.

x-units Number of x-axis to move cursor from beginning of series.

RETURNS: Nothing

EXAMPLE: PUT(5.0)
puts the cursor on the point at 5.0 x-units (not delta-x units).

SEE ALSO: NPUT MOVE
NMOVE

PUTENV(string)

PURPOSE: Sets an environment string.

string Legal environment string, in quotes.

RETURNS: Integer value of the operating system's "PUTENV" routine.

SEE ALSO: GETENV
QMODE

QMODE(=symbol_name)

PURPOSE: Gets the latest value for a requested symbol.

= The character that indicates to MarketBrowser that a quote request is being made.

symbol_name A valid symbol name for your data service.

RETURNS: A string.

EXAMPLE: =IBM
returns the latest value of IBM.

REMARKS: The QMODE function (=symbol_name) calls the macro _qquote. By default, this macro is defined as QUOTE("symbol_name"). Note, however, that the _QQUOTE macro is fully customizable.

SEE ALSO: QUOTE QPAGE
QSTRING

QPAGE(symbol)

PURPOSE: Returns a “page” for a given quote

symbol A valid symbol name for your data service, in quotes.

RETURNS: A string with embedded newline characters

EXAMPLE: QPAGE(“IBM”) returns a quote “page” for IBM

SEE ALSO: QUOTE QMODE
QSTRING

QUOTE(symbol)

PURPOSE: Returns a formatted "one-shot" quotation string.

symbol A valid data identifier for your data service, in quotes.

EXAMPLE: QUOTE("IBM") might return a string such as "IBM N +122/1 at 11:22"

REMARKS: QUOTE can also used to communicate with a data service for the purpose of parameter setting, etc. rather than to return quotations. See the MarketBrowser Real-Time Data Interface (API) documentation for further discussion.

SEE ALSO: QMODE QPAGE
QSTRING RTSEND

QSTRING(symbol)

PURPOSE: Returns a formatted “one-shot” arbitrary string for a given symbol.

symbol A valid symbol name for your data service, in quotes.

RETURNS: A string

EXAMPLE: QSTRING(“IBM”) returns the latest value of IBM

SEE ALSO: QPAGE QUOTE
QMODE

RATE(window)

PURPOSE: Displays the sampling rate of a series.

window (Optional). A window reference to a series or table. The default is the current window.

RETURNS: A number.

SEE ALSO: DELTAX SETDELTAX

RAVEL(series, length, start, overlap, keep_item_type)

PURPOSE:	Creates a list of series from one or more sources.								
series	Series to ravel; may be a list of series. If it is a list of series, the length, start, and overlap arguments are rendered invalid, and are replaced by the optional argument keep_item_type.								
length	(Optional). Integer length of ravel segments. The default is the length of the input series.								
start	(Optional). Integer start point in series. The default is 1.								
overlap	(Optional). Integer segment overlap. The default is 0.								
keep_item_type	(Optional). Integer. Valid only if you're providing two or more series to ravel. Options are: <ul style="list-style-type: none">• 0 - Ravel multiple series into a matrix (single data item).• 1 - Preserve item types of input series. Similar behavior to OVERPLOT.								
RETURNS:	A table.								
EXAMPLES:	<p>RAVEL(W1, 100, 1, 10) ravels the series in W1 into multiple 100 point long segments where each segment overlaps the previous segment by 10 points. The overlap parameter must be less than the segment length.</p> <p>RAVEL(W1, W2, W3, W5) This RAVEL appears just like a similarly constructed overplot, but in subsequent operations would be treated as a four column table.</p>								
REMARKS:	<p>RAVEL with more than one series argument makes a matrix out of the series by putting the series into columns in the matrix. RAVEL with matrix arguments makes one large matrix out of the input matrices by connecting them right-to-left, i.e. connecting the rows end-to-end. The number of rows stays the same, and the number of columns of the output is equal to the sum of the number of columns of the input.</p> <p>In certain cases, it may be faster to use COLADD or COLDEL. These functions are used to add and remove columns of data in an existing matrix. They are a much faster alternative to RAVEL(), which copies data and returns the result.</p>								
SEE ALSO:	<table><tr><td>UNRAVEL</td><td>CONCAT</td></tr><tr><td>REMOVE</td><td>REPLICATE</td></tr><tr><td>WATERFALL</td><td>OVERPLOT</td></tr><tr><td>COLADD</td><td>COLDEL</td></tr></table>	UNRAVEL	CONCAT	REMOVE	REPLICATE	WATERFALL	OVERPLOT	COLADD	COLDEL
UNRAVEL	CONCAT								
REMOVE	REPLICATE								
WATERFALL	OVERPLOT								
COLADD	COLDEL								

RAWROW(table, rownum)

PURPOSE:	Extracts a row from a table.
table	Input table or compound data structure (i.e., any multiple column data series).
rownum	Integer. The row number.
RETURNS:	A single row containing the values of the points in the requested row, including NA values.
EXAMPLE:	Given the formula: W1: RAVEL(GRANDOM(100,1),25) W2: TRANSPOSE(RAWROW(W1, 3)) produces a series containing the elements in the third row of the table in window 1, including any NA values.
REMARKS:	To perform series operations on a single row of data, first use the TRANSPOSE function to convert the row into a column, or series. To select more than one row of data, use the REGION function.
SEE ALSO:	ROW COL GETSERIES

RDERIV(series)

PURPOSE:	Returns the derivative of a series or series expression using a right-to-left algorithm.
series	A series or table.
RETURNS:	A series or table.
EXAMPLE:	RDERIV(W3) creates a new series from the contents of window 3 and places the result in the current window. The value of each point in the new series will be the slope of the series in window 3 at that point.
REMARKS:	The formula used to compute derivatives with the RDERIV function for each point i is as follows: $\text{RDERIV}(i) = (\text{series}\langle i+1 \rangle - \text{series}\langle i \rangle) / (\text{x-axis interval})$ The derivative of the last point is computed using the method of LDERIV.
SEE ALSO:	INTEG LDERIV DERIV

READA(filename, column)

- PURPOSE:** Reads an ASCII data file from disk and load it directly into the current window.
- filename** Name of input file, in quotes. If no path is given, READA looks for the file in the current, working directory.
- column** (Optional) Column number to read (origin 0). Defaults to 0.
- RETURNS:** A signal.
- EXAMPLE:** READA("TARGET87")
reads the ASCII file "TARGET87", without a header, from disk into the current window.
- REMARKS:** READA is useful for making a quick record of data in a few windows.
You can always write any series back to disk with WRITEA or WRITEB.
This function is deprecated. READANYHIST should be used instead.
- SEE ALSO:** READAHIST
READB
WRITEA

READAHIST(window, filename, periodicity, datetime_cols, data_cols, interval, swap, skip)

PURPOSE:	Reads tables of historical or intraday data.	
window	(Optional). A window to receive the data. Defaults to the current window.	
filename	Name of ASCII file to read, in quotes.	
periodicity	Periodicity of the data; a string denoting any of the date or time units internally defined in MarketBrowser.	
datetime_cols	Number of date/time columns (typically 1 or 2).	
data_cols	Number of columns of data to read in, (typically 1 to 4).	
interval	(Optional). Integer. Sets the interval, in seconds, of intraday interval data (periodicity "RT"). Defaults to the interval between the first two observations in the file. Use -1 as a placeholder.	
swap	(Optional). Integer. If set to 1, or true, swaps the first and last columns of a four-column sequence of data (i.e., what would normally be read as a C-H-L-O sequence would be read as O-H-L-C). Defaults to 0, or false.	
skip	(Optional). Integer. The number of rows from the top to skip over.	
RETURNS:	A table of data, displayed as line chart or trading bars/candlesticks by default. If no data is found, READAHIST returns a series of length 0.	
EXAMPLE:	READAHIST("mytable.dat", "D", 1, 4, -1, 1) Reads in the CHLO file of the form: 01/01/09 122.125 122.5 122.0 121.75 and graphs it as an OHLC file: 01/01/09 121.75 122.5 122.0 122.125	
REMARKS:	READAHIST is a string denoting any of the date or time units internally defined in MarketBrowser (such as "D" for daily, "RT" for real-time, etc.) READAHIST performs date and time filtering to interject NAs where dates/times are missing and to drop observations which are not valid (according to the holiday and weekend schedule in effect).	
SEE ALSO:	WRITEAHIST READTABLE SETMATRIX RTHISTP	READA READBHIST NAVALUE READANYHIST

READANYHIST(window, filename, periodicity, datetime_cols, data_cols, interval, swap, skip)

PURPOSE:	Reads tables of historical or intraday data.	
window	(Optional). A window to receive the data. Defaults to the current window.	
filename	Name of ASCII file to read, in quotes.	
periodicity	(Optional). Periodicity of the data; a string denoting any of the date or time units internally defined in MarketBrowser.	
datetime_cols	(Optional). Number of date/time columns (typically 1 or 2).	
data_cols	(Optional). Number of columns of data to read in, (typically 1 to 4).	
interval	(Optional). Integer. Sets the interval, in seconds, of intraday interval data (periodicity "RT"). Defaults to the interval between the first two observations in the file. Use -1 as a placeholder.	
swap	(Optional). Integer. If set to 1, or true, swaps the first and last columns of a four-column sequence of data (i.e., what would normally be read as a C-H-L-O sequence would be read as O-H-L-C). Defaults to 0, or false.	
skip	(Optional.) Integer. The number of rows from the top to skip over.	
RETURNS:	A table of data, displayed as line chart or trading bars/candlesticks by default. If no data is found, READANYHIST returns a series of length 0.	
EXAMPLE:	READANYHIST("mytable.dat","D") Attempts to determine the number of date and data columns in the file. Once it has determined this information, it reads the file into MarketBrowser.	
REMARKS:	READANYHIST first looks through the data file and attempts to determine the periodicity, and number of date/time and data columns in the file. It then passes any information it has gathered to the READAHIST function, which reads in the data. See the READAHIST function definition for further discussion of the arguments.	
SEE ALSO:	READAHIST READA READBHIST NAVALUE	WRITEAHIST READTABLE SETMATRIX RTHISTP

READB(filename, type, num_pts, offset)

PURPOSE: Reads a BINARY data file from disk and loads it directly into the current window.

filename Name of input file, in quotes. If no path is given, READB looks for the file in the current working directory (which can be gotten using GETPATH()).

type Format of the disk file described by either its name or code from the list below.

Name	Code	Data Type	Range
SBYTE	1	Signed Byte	-128 to +127
UBYTE	2	Unsigned Byte	0 to 255
BYTE	2	(same as UBYTE)	0 to 255
SINT	3	Signed Integer	-32768 to +32768
UINT	4	Unsigned Integer	0 to 65536
LONG	5	4-byte Signed Integer	-2,147,483,648 to +2,147,483,647
FLOAT	6	4-byte Floating Point	-10^{37} to $+10^{38}$ -10^{-37} to $+10^{-38}$
DOUBLE	7	8-byte Floating Point	-10^{307} to $+10^{308}$ -10^{-307} to $+10^{-308}$

num_pts (Optional). The number of points to read. The default is -1 (all points).

offset (Optional). An integer. The starting point in the file.

RETURNS: A signal.

EXAMPLES: READB("myfile", FLOAT, 1024, 18)
reads 1024 floating point numbers starting at the 19th byte in the file.

READB("myfile", FLOAT, -1, 18)
reads all floating point numbers in the file starting at the 19th byte.

REMARKS: READB ignores any header information in the data file. You can always write any series back to disk with WRITEA or WRITEB.

SEE ALSO: READBHIST
WRITEB
READAREADDT

READBHIST(window, filename, periodicity, datetime_cols, data_cols, interval, swap)

- PURPOSE:** Reads tables of binary historical or intraday data.
- window** (Optional). A window to receive the data. Defaults to the current window.
- filename** Name of binary file to read.
- periodicity** Periodicity of the data; a string denoting any of the date or time units internally defined in MarketBrowser (see PERCONV).
- datetime_cols** Integer flag. Whether or not the binary file contains any UNIX-format dates. The option '2' is provided as a convenience for READAHIST users. Options are:
- 0 - no UNIX-format date/time byte
 - 1 - date/time bytes exist
- data_cols** Number of columns in which to return the data (typically 1 to 4).
- interval** (Optional). Integer. Sets the interval, in seconds, of intraday interval data (periodicity "RT"). Defaults -1, or the interval between the first two observations in the file.
- swap** (Optional). Integer. If set to 1, or true, swaps the first and last columns of a four-column sequence of data (i.e., what would normally be read as a C-H-L-O sequence would be read as O-H-L-C). Defaults to 0, or false.
- RETURNS:** A table of data, displayed as line chart or trading bars/candlesticks by default.
- EXAMPLE:** READBHIST("mytable.dat", "D", 1, 4)
Reads in a file of the form:
01/01/09 122.125 122.5 122.0 122.25
- REMARKS:** READBHIST performs date and time filtering to interject NAs where dates/times are missing and to drop observations which are not valid (according to the holiday and weekend schedule in effect).
If date info, then takes first sizeof(long) bytes, and interprets it as standard unix-format date/time format.
- SEE ALSO:** READB
READTABLE
READAHIST
WRITEBHIST

READDT(filename, column, type)

- PURPOSE:** Read an ASCII data file of dates or times from disk and loads it directly into the current window.
- filename** Name of input file, in quotes. If no path is given, READDT looks for the file in the current, working directory.
- column** Column number to read (origin 0).
- type** Data type - 0 = time, 1 = date.
- EXAMPLE:** READDT("TARGET87", 0, 1)
reads the first column of ASCII file "TARGET87" as dates.
- REMARKS:** Supported formats for dates are mm/dd/yy, mm dd yy, and mm-dd-yy, where yy can also be a four digit year. For time stamps, the format is hh:mm:ss in 24 hour day form.
- SEE ALSO:** READB READTABLE
WRITEA WRITETABLE

READTABLE(filename, startrow, startcol, collist)

- PURPOSE:** Reads tables of historical or intraday data.
- filename** Name of ASCII file to read as a multi-column table.
- startrow** The number of the first row to start reading from which can contain a header or data (Origin 1).
- startcol** Number of the first data column (Origin 1).
- collist** List of numbers indicating which columns of data to accept.
- RETURNS:** A table of data.
- EXAMPLE:** READTABLE("mytable.dat", 1,1,12,17))
produces a matrix with two columns of data, as found in columns 12 and 17 of "mytable.dat."
- REMARKS:** READTABLE translates the literal string NA (or NULL) as NA.
- SEE ALSO:** READAHIST READBHIST
READDT WRITETABLE
SETMATRIX NAVALUE

REAL(expr)

- PURPOSE:** Finds the real component of a complex series, table or numbers.
- expr** Any expression evaluating to a series, table, integer or real or complex number.
- RETURNS:** A series, table or number.
- EXAMPLES:** REAL(42.1)
displays 42.1.
REAL(3.2 + 4.7i)
yields 3.2.
REAL(W8)
returns a new series, in the current window, composed of real numbers..
- SEE ALSO:** IMAGINARY MAGNITUDE
ANGLE (Trig Function) INT
CARTESIAN CONJUGATE
PHASE

REDRAW

- PURPOSE:** Redraws all the windows in a worksheet.
- RETURNS:** Nothing.
- SEE ALSO:** REDRAWALL

REDRAWALL

- PURPOSE:** Redraws the entire MarketBrowser screen.
- RETURNS:** Nothing.
- EXAMPLE:** TOOLBAR(1,4,1, "mybutton", "menufile('mymenu.mnu')"); REDRAWALL
creates a toolbar button, then redraws the entire MarketBrowser screen.
- REMARKS:** Use this function to force MarketBrowser to redraw itself after calling a plot-time function.
- SEE ALSO:** REDRAW
PON

REDUCE(series, op)

PURPOSE:	Inserts an operator between every observation of a series, then evaluates the expression.	
series	A series or table.	
op	Quoted string containing the binary operator.	
RETURNS:	A number.	
EXAMPLE:	REDUCE(GSER(1,2,3), "*") expands to the expression "1*2*3", that evaluates to 6.	
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR".	
SEE ALSO:	COLREDUCE INTERPOSE OUTERPROD	ROWREDUCE INNERPROD

REFRESH

PURPOSE:	Reevaluates a worksheet, including a reacquisition of the source data.
RETURNS:	Nothing.
SEE ALSO:	CLEARDATA UPDATE

REGION(table, row, rowlen, col, collen)

PURPOSE:	Copies a rectangular region out of a table, padding with zeros as necessary.	
table	Input matrix.	
row	Row to start copying from.	
rowlen	Number of rows to copy; will pad with zero, if necessary.	
col	Column to start copying from.	
collen	Number of columns to copy; will pad with zero, if necessary.	
RETURNS:	A matrix whose column and row numbers start with 1.	
EXAMPLE:	REGION(W3, 1, 10, 1, 10) produces a 10 by 10 tabular subset of window 3, padded with zeros where window 3 had no data in the requested dimensions.	
SEE ALSO:	COL	

REMOVE(series, interval, start, blocksize)

PURPOSE:	Removes points from a series on a periodic basis.	
series	A series from which to remove points.	
interval	An integer specifying the remove interval.	
start	(Optional). An integer starting point. The default is 1.	
blocksize	(Optional). The number of points to remove at every interval (default 1).	
RETURNS:	A series or table.	
EXAMPLES:	REMOVE(W1, 64, 4) removes every 64th point, starting at the 4th point. REMOVE(W1, 64, 4, 8) removes a block of eight points, every 64 points, starting at the 4th point.	
SEE ALSO:	EXTRACT	DECIMATE
	MERGE	RAVEL

REMOVEWINDOW(n)

PURPOSE:	Removes the indicated number of windows from the worksheet.
n	An integer. Represents the number of windows to be removed.
RETURNS:	Nothing.
REMARKS:	Removes windows from the point of the cursor.
SEE ALSO:	ADDWINDOW

REORDER(series, indices)

PURPOSE:	Rearranges a series based on a list of indices.
series	Series or table to reorder.
table	Series of indices on which to reorder.
RETURNS:	A series or table.
EXAMPLES:	REORDER(W1, GRADE(W1)) produces the same result as SORT(W1). Now, try this: REORDER(SORT(W1), GRADE(GRADE(W1),1)) To sort the rows of a matrix by one column, preserving the rows, type: REORDER(W1, GRADE(COL(W1,1)))

REMARKS: REORDER is similar to LOOKUP, but much faster on larger series.

SEE ALSO: GRADE SORT
LOOKUP

REPLICATE(series, n)

PURPOSE: Concatenates a series with itself.

series A series or table.

n An integer indicating the number of times the specified series should be concatenated to itself.

RETURNS: A series or table equal in length to n times the original series.

EXAMPLE: REPLICATE(W2, 5)

concatenates five copies of the series from window 2. If the original series is 100 points, the resulting series will be 500 points.

REMARKS: REPLICATE is a special version of CONCAT.

SEE ALSO: CONCAT MERGE
RAVEL EXTRACT

REVERSE(series)

PURPOSE: Plots the data points of a given series in reverse order.

series A series or table.

RETURNS: A series or table.

SEE ALSO: EXTRACT

RMDIR(directory, behavior)

PURPOSE: Deletes a directory if it is empty.

directory String. The name of the directory to delete.

behavior (Optional). Integer. 0 (default) - don't confirm before deleting; 1 - confirm before deleting.

RETURNS: An error if the directory does not exist, 1 if the deletion is successful, and 0 if it is not.

EXAMPLE: RMDIR("c:\mydir")

REMARKS: If you use relative paths, they will be interpreted as relative to the current working directory (which can be obtained by using the GETPATH() function).

SEE ALSO: GETPATH DELFILE
 DIREXISTS FILEEXISTS
 MKDIR

RMFORM(window, expression)

PURPOSE: Removes an expression (function) and any embedded arguments from the compiled part of a window formula. It does not edit the formula string.

window (Optional). The window containing the formula that you want to remove. Defaults to the current window.

expression String. The expression that you want to remove, in quotes.

RETURNS: Nothing.

EXAMPLE: Given the window with the formula:
 W1:ADDFORM("overplot(movavg(curr, 20))")
 then
 W2: RMFORM(W1, "overplot")
 removes the overplot function (and any arguments, if appropriate).

REMARKS: RMFORM works by reading the "compiled" formula of the window from right to left and searching for the first complete statement which begins with the specified expression. RMFORM does not edit the formula string so if the expression being removed exists in the formula string in addition to the compiled formula, re-evaluating the window will cause the expression that was removed to be reapplied.

SEE ALSO: ADDWFORM ADDFORM

RMS(expr)

PURPOSE: (A macro). Calculates the root mean square of a series, table or number.

expr Any expression resolving to a series, table or number.

RETURNS: A number.

**EXPAN-
SION:** SQRT(MEAN(ARG*ARG))

EXAMPLE: RMS(W1*W2)
 calculates the root mean square of W1*W2.

SEE ALSO: MEAN
 SQRT

ROOTS(n)

PURPOSE: Generates a series or table containing the n-complex roots of unity.

n The number of roots to generate.

RETURNS: A series or table with n points.

SEE ALSO: RTHROOT

ROTATE3D(xa, ya, za)

PURPOSE: Rotates a PLOT3D graph.

xa Angle in degrees of x-axis.

ya Angle in degrees of y-axis.

za Angle in degrees of z-axis.

RETURNS: Nothing.

SEE ALSO: PLOT3D
MOUSEROTATE

ROUNDUP(expr)

PURPOSE: Finds the smallest integer greater than or equal to the input value.

expr An expression evaluating to a scalar, series, table, integer, or real or complex number.

RETURNS: A scalar, series, table or number, depending on the input.

EXAMPLES: ROUNDUP(7.2) displays 8.0.
ROUNDUP(W2) creates a new series by applying ROUNDUP to each series element of window 2. The integer value returned by ROUNDUP is converted to a floating point value.

SEE ALSO: TRUNC

ROW(table, rownum)

PURPOSE: Extracts a row from a table.

table Input table.

rownum Integer. The row number.

RETURNS: A series.

EXAMPLE: ROW(W1, 3)
produces a row with the elements in the third row of the table in W1.

REMARKS: To perform series operations on a single row of data, first use the TRANSPOSE function to convert the row into a column, or series. To select more than one row of data, use the REGION function.

If ROW encounters an NA value in the row, it returns the value of the element in the following row, but the same column.

SEE ALSO: COL
RAWROW

ROWLAYOUT(int1,...,intn)

PURPOSE: Sets how many rows the MarketBrowser screen is divided into, and in turn, how many windows the individual rows are divided into.

int1, ..., intn An integer of how many windows in a row.

RETURNS: A screen with a specified number of rows and windows per row.

EXAMPLE: ROWLAYOUT(2,3,4)
In a 9 window worksheet, this would return three even rows of 2, 3, and 4 windows, respectively.

REMARKS: ROWLAYOUT will return an error message if the total number of windows specified as parameters exceeds the number of displayed windows in the worksheet.

On the other hand, if you specify the layout for fewer windows than the displayed total, MarketBrowser will group the remaining windows into a single row. For example, in a 9 window worksheet, ROWLAYOUT(2) will return a screen with 2 rows of 2 and 7 windows, respectively.

SEE ALSO: COLLAYOUT NEATEN
TILE

ROWREDUCE(**table**, **op**)

PURPOSE:	Applies the REDUCE function to each row of a table.	
table	A table.	
op	Quoted string containing the binary operator.	
RETURNS:	A table with one column and as many rows as the input matrix.	
EXAMPLE:	ROWREDUCE(RAVEL(GSER(1, 2, 3), GSER(2, 3, 4)), "**") Produces a table with the single column 2, 6, 12.	
REMARKS:	Binary operators include the arithmetic and logical operators. The "Exclusive OR" operator is represented by the string "XOR".	
SEE ALSO:	COLREDUCE	INNERPROD
	INTERPOSE	OUTERPROD
	REDUCE	

RTAMEND(**trigger**, **hist_series**, **string_args**, **int_args**)

PURPOSE:	Amends or extends a historical series with a currently ticking real-time value. Analogous to what the BARMON function does with the following difference: BARMON automatically retrieves its own history and appends updates. RTAMEND expects the history to be supplied and uses BARMON-type logic to amend/extend that history with real-time events.	
trigger	Integer, typically a hot variable used to cause reevaluation when the initial values of hist_series change.	
hist_series	Series, a price series to use as the initial values for the output series. This can be either of type "RT" or "daily".	
string_args	Optionally use the various string arguments of BARMON (symbol, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, and gap_2_end. See that entry for details.) to supply a symbol name to update the current value of hist_series.	
int_args	Optionally use the various integer arguments of BARMON (interval, paint_tick, update, add_nas, inside, and na_interp. See that entry for details.) to supply a symbol name to update the current value of hist_series. Note that if the "frequency" argument is not compatible with the frequency implied by the hist_series, the hist_series prevails.	
RETURNS:	A series. The result from RTAMEND is always of type RT.	
SEE ALSO:	BARMON RTPMATRIX	

RTDEBUG(level)

PURPOSE:	Sets a debugging trace level for the data service transactions.
level	Integer. The tracing level. The most common option is 3.
RETURNS:	Nothing.
REMARKS:	The debugging trace appears in the window from which MarketBrowser was launched. Results vary widely depending on data service.
SEE ALSO:	RTSEND

RTDEPEND(target, symbol, type)

PURPOSE:	Causes a window or a hot variable to reevaluate based on a real-time event or evaluation cycle.
target	(Optional). The name of the window or the hot variable to make dependent, in quotes. Defaults to the current window.
symbol	The symbol to add dependency for, in quotes. Argument is not required if type is 0, or CLEAR.
type	Integer. Type of dependency to add. Options are: <ul style="list-style-type: none">• 0 - CLEAR all dependencies for target.• 1 - RT_POLL; reevaluate target on master real-time clock cycle.• 2 - RT_INTERRUPT; reevaluate target whenever there is a new event for the symbol.• 4 - reserved.• 8 - RT_ANNOTATIONS; reevaluate target's plot-time evaluations (e.g., update a calculated value displayed as a text annotation) whenever a new event for the symbol arrives.
RETURNS:	Nothing.
EXAMPLE:	RTDEPEND("W2","IBM.LAST",2) causes window 2 to reevaluate every time a new value for IBM.LAST is registered.
REMARKS:	If type is set to 8 (RT_ANNOTATIONS), it will be logically ORed into the target; the others are simply assigned.
SEE ALSO:	CALC DEPEND

RTHISTORY(symbol, data_type, start_date, start_time, end_date, end_time, gap_1_start_, gap_1_end, gap_2_start, gap_2_end, add_nas, inside, na_interp)

PURPOSE:	Retrieves historical data from a live data service.
symbol	Quoted string. Instrument symbol and field.
data_type	(Optional). Integer. The data type. Options are: <ul style="list-style-type: none">• 1 - Historical (daily). Default.• 2 - Intraday
start_date	(Optional). Quoted string of the form “mm/dd/yy”, which represents the date from which to start extracting data. Use the "" characters (an empty pair of quotes) to leave the default, which is the start date of the series.
start_time	(Optional). Quoted string of the form “hh:mm:ss”, representing the time from which to start extracting data . To leave the default (the start time), use "" as a placeholder.
end_date	(Optional). Quoted string of the form “mm/dd/yy” representing the date on which to stop extracting data from symbol . Use "" to leave the default, which is the last available day of data in symbol .
end_time	(Optional). Quoted string of the form “hh:mm:ss”, representing the time to stop extracting data from symbol . To leave the default (the last available time for symbol), use "" as a placeholder.
gap_1_start, gap_2_start	(Optional). Quoted string of the form “hh:mm:ss”, representing the beginning of a gap in the extraction of data from symbol . To leave the default (00:00:00), use "" as a placeholder.
gap_1_end, gap_2_end	(Optional). Quoted string of the form “hh:mm:ss”, representing the end of a gap in the extraction of data from symbol . To leave the default (23:59:59), use "" as a placeholder.
add_nas	(Optional). Integer. Type of NA processing. Options are: <ul style="list-style-type: none">• 0 - Do not fill gaps with NAs (default).• 1 - Fill all gaps with NAs• 2 - Fill valid gaps on business days inside of trading hours with NAs (represented by gap_1_start/end and gap_2_start/end).
inside	(Optional). Integer. How to process gaps: <ul style="list-style-type: none">• 1 - Keep data INSIDE (within) of gap_1_start or gap_1_end and gap_2_start or gap_2_end (default).• 0 - Keep data OUTSIDE of gap_1_start/gap_1_end and gap_2_start/gap_2_end.
na_interp	(Optional). Integer. Type of interpolation. Options are: <ul style="list-style-type: none">• 0 - Leave gaps. (default)• 1 - Perform linear interpolation through valid gaps

RETURNS: A series.

EXAMPLE: `RTHISTORY("IBM.LAST",1, "10/25/95","", "11/25/95","", "12:00:00", "13:00:00","", "", 2,0,0)`

displays historical closing prices for IBM from the market open on 10.25.95, to the market close on 11/25/95, with a gap between 12 and 1 PM every day. Values during trading gaps are represented as NAs.

REMARKS: The data and time range of the retrieval is set by the RTRANGE command currently in effect. Data extraction is performed with the same mechanism as the DTEXTRACT function uses.

For details of data reading and conditioning, see the READAHIST function.

Availability of symbols and time ranges will vary according to your data service.

This function is deprecated. RTHISTP should be used instead.

SEE ALSO: RTHISTP
READAHIST

RTHISTP(symbol, periodicity, start_date, start_time, end_date, end_time, gap_1_start, gap_1_end, gap_2_start, gap_2_end, datetime_cols, data_cols, interval, add_nas, na_interp, inside, single_col, vol_bars)

PURPOSE:	Reads in data of various periodicities through a real-time data interface.
symbol	Instrument symbol.
periodicity	Quoted string denoting the periodicity of the data. Can be any of the date or time units internally defined within MarketBrowser: <ul style="list-style-type: none">• "RT" - Real-time or Intraday• "D" - Daily• "W" - Weekly• "M" - Monthly• "Q" - Quarterly• "Y" - Yearly• "T" - Ticks
start_date	(Optional). Quoted string of the form "mm/dd/yy", which represents the date from which to start extracting data. Use the "" characters (an empty pair of quotes) to leave the default, which is the start date of the series.
start_time	(Optional). Quoted string of the form "hh:mm:ss", representing the time from which to start extracting data . To leave the default (the start time), use "" as a placeholder.
end_date	(Optional). Quoted string of the form "mm/dd/yy" representing the date on which to stop extracting data from symbol . Use "" to leave the default, which is the last available day of data in symbol .
end_time	(Optional). Quoted string of the form "hh:mm:ss", representing the time to stop extracting data from symbol . To leave the default (the last available time for symbol), use "" as a placeholder.
gap_1_start, gap_2_start	(Optional). Quoted string of the form "hh:mm:ss", representing the beginning of a gap in the extraction of data from symbol . To leave the default (00:00:00), use "" as a placeholder.
gap_1_end, gap_2_end	(Optional). Quoted string of the form "hh:mm:ss", representing the end of a gap in the extraction of data from symbol . To leave the default (23:59:59), use "" as a placeholder.
datetime_cols	Integer. Number of date/time columns. Typically 1 or 2. Defaults to 2.
data_cols	Integer. Number of columns of time-series data to read in. Typically between 1 and 4. Defaults to 4.
interval	(Optional). For data with periodicity "RT". Sets the interval, in seconds, of the data. Defaults to the interval between the first two observations in the file.
add_nas	(Optional). Integer. Type of NA processing. Options are:

- 0 - Do not fill gaps with NAs (default).
- 1 - Fill all gaps with NAs
- 2 - Fill valid gaps on business days inside of trading hours with NAs (represented by gap_1_start/end and gap_2_start/end).

na_interp (Optional). Integer. Type of interpolation. Options are:

- 0 - Leave gaps. (default)
- 1 - Perform linear interpolation through valid gaps

inside (Optional). Integer. How to process gaps:

- 1 - Keep data INSIDE (within) of gap_1_start or gap_1_end and gap_2_start or gap_2_end (default).
- 0 - Keep data OUTSIDE of gap_1_start/gap_1_end and gap_2_start/gap_2_end.

single_col (Optional). Integer. How to treat single column data. Options are:

- 0 - Make the input into CHLO bars. (default)
- 1 - Leave the input as is.

vol_bars (Optional). Integer. What kind of bars to make out of single column data. Options are:

- 0 - Make CHLO bars. (default)
- 1 - Makes volume bars (sum the ticks in each bar).

RETURNS: A series.

EXAMPLE: RTHISTP('IBM.LAST', "D", 2, 1)

REMARKS: This function is similar to the RTHISTORY function, except that it allows you to read in data that is not daily.

By default, data is displayed as a line chart.

SEE ALSO: RTHISTORY
READAHIST
DTEXTTRACT

RTHROOT(r)

- PURPOSE:** Returns the principal complex rth root of unity.
- r** A real number.
- RETURNS:** A complex scalar, $e^{(2\pi i/r)}$.
- EXAMPLES:** RTHROOT(2.0)
returns mag = 1; angle = 3.14159, 180.
RTHROOT(6.9)
displays mag = 1; angle = 0.91061, 52.17391
- REMARKS:** Returns a complex number in polar form.
Use CARTESIAN(RTHROOT(r)) to see the root in Cartesian form.
Use RTHROOT(r)^n to see the other rth roots of unity.
- SEE ALSO:** ROOTS

RTINTERVAL(interval)

- PURPOSE:** Sets the timing interval for timer-bound functions.
- interval** Integer. The time between updates, in seconds.
- RETURNS:** A number.
- REMARKS:** This parameter is NOT saved between MarketBrowser sessions.
The default interval is 60 seconds. The smallest interval is 30 seconds. Different windows cannot have different real-time sampling intervals.
In order for an RTINTERVAL command to work properly, real-time monitoring must be off when the command is issued. New real-time sampling intervals will take effect the next time real-time monitoring is turned on (with the RTON function).
If you set the real-time sampling interval in a worksheet with real-time monitor windows, be sure to issue the REFRESH command after setting the sampling interval. This way, all incoming data will be collected at the same time intervals.
- SEE ALSO:** RTON
RTOFF
BARMON
MONITOR

RTLINK

- PURPOSE:** Returns the current “Hot Link.”
- RETURNS:** A string.
- REMARKS:** The implementation of this function is platform specific:
Microsoft Windows: When data has been registered as a “Link,” this function returns a string that represents the address of Microsoft Windows clipboard data. In order for this function work properly, the application from which the Link has been chosen must be open.
UNIX, VMS: Currently unimplemented.

RTNOBS(observations)

- PURPOSE:** Sets the default number of observations shown in a real-time window.
- observations** Integer. The number of observations.
- RETURNS:** Nothing.
- REMARKS:** Data points not shown by default in the window may be seen by scrolling right or compressing horizontally. The scrolling effect can be turned off by appropriate settings in the file, "expo.cnf". This function has no effect when the value of the configuration variable FORCE_AUTOSCALE is a positive number.

RTOFF

- PURPOSE:** Deactivates real-time processing.
- RETURNS:** Nothing.
- REMARKS:** It is almost never appropriate to use RTOFF. When RTOFF is used, no updates will arrive for any data series.
- SEE ALSO:** RTON

RTON

- PURPOSE:** Activates real-time processing.
- RETURNS:** Nothing.
- REMARKS:** Real-time processing is normally always on, so it is not normally necessary to use RTON. When real-time processing is on, a series generated by a function such as BARMON and MONITOR changes as new data arrives.
- SEE ALSO:** RTOFF
RTINTERVAL

RTQUOTE(symbol)

PURPOSE:	Registers a data item for event-driven updating, without accumulating a data series.
symbol	A valid data identifier for the data service, in quotes.
RETURNS:	A real Number.
EXAMPLE:	RTQUOTE("IBM.N.BID") causes the window to monitor bids for IBM on the NYSE. The window will be reevaluated whenever new bid information comes in.
REMARKS:	RTQUOTE can also be used to receive communications from a data service which reflect events other than quotations. See the MarketBrowser Real-Time Data Interface documentation for further discussion. RTQUOTE is primarily used to create watchlists and other real-time annotations.
SEE ALSO:	CAPTURE BARMON MONITOR RTON

RTPMATRIX(trigger, frequency, psources, psyms, csources, csyms, quantities, inversions, buy-ins, longshorts, cashfutures, mult_factors, div_factors)

PURPOSE:	Used to tabulate current values for a portfolio from the real-time values of its components. The current value tabulated is returned as a part of the output matrix described below and is also published internally in MarketBrowser as a synthetic symbol. The function takes one of two formats for its arguments; either all the details explicitly (as above), or an argument specifying an existing portfolio with an appropriate family of argument variables already created (RTPMATRIX(trigger, frequency, family_name)).
trigger	Integer, typically a hot variable used to cause the portfolio to reinitialize when its structure changes.
frequency	Integer, sets the frequency of updating the current value of the portfolio. Typically this is 0, meaning whenever any contributing price or currency changes, or 1, meaning "resume on the minute."
family_name	String, naming an existing portfolio. For example, if the portfolio "rtdemo" is already historically priced by BasketTrader, the family of variables needed to specify the arguments to RTPM already exist, so just use them. In this case, the variable names all begin with "rtdemo" and are suffixed "_rtpsrc", "_rtpsym", "_rtcsrc", "_rtcsym", "_rtqty", "_rtinv", "_rtbuyin", "_rtls", "_rtcf", "_rtmult", "_rtdiv", and are used in place of the following formal arguments:
psources	A new-line delimited string of "sources" for pricing symbols. Use "" to mean "use the current source."

psyms	A new-line delimited string of symbols for the holdings.
csources	A new-line delimited string of "sources" for currency rates. Use "" to mean "use the current source".
csyms	A new-line delimited string of symbols for currency rates. Use "" to mean "no currency conversion".
quantities	A series with the quantity of each holding. Use "1.0" (one share) as a placeholder.
inversions	A series of 0/1s to indicate whether the currency quote, if any is inverted. Use "0.0" to mean "do not invert currency quotation".
buy-ins	A series with the buy-in price for each holding. Use "0.0" as a placeholder.
longshorts	A series of 0/1s to indicate whether a holding is long or short. Use "1.0" (meaning "long") as a placeholder.
cashfutures	A series of 0/1s to indicate whether a holding is a cash (1) or future (0) position. Use "1.0" (meaning "cash") as a placeholder.
mult_factors	A series of factors to apply multiplicatively to each holding price. Use "1.0" as a placeholder.
div_factors	A series of factors to apply by dividing each holding price. Use "1.0" as a placeholder.

RETURNS: The result from RTPMATRIX is a matrix. The first column has 5 values; total value, number of holdings, reserved1, reserved2, and reserved3. There is an additional column for each holding in the portfolio, containing current price, currency rate, quantity, inversion, buyin, longshorts, cashfutures, mult_factor, div_factor, and current holding value. In addition to the output matrix, a new synthetic symbol is published within MarketBrowser, showing the current value of the portfolio in real-time. This symbol is named "family_name_SYN" (e.g. "rtdemo_SYN") and is available to all MarketBrowser RT type functions, but is typically used by RTAMEND to update a historical portfolio.

REMARKS: Note that each of the variables describing an attribute of a holding should have the same number of entries. That is, for example, 10 pricing sources, 10 pricing symbols, etc. "Placeholder" values can be used to round out the series and newline embedded strings as needed.

SEE ALSO: RTAMEND

RTPOST(string)

- PURPOSE:** Sends an arbitrary string to the real-time data interface.
- string** Any string, in quotes.
- RETURNS:** Nothing.
- EXAMPLE:** RTPOST(STRCAT("Message From:",STRNUM(hostid)))
- REMARKS:** RTPOST can be used to send information to the MarketBrowser Real Time Data Interface ("RTDI", also known as "EXPOSRV") module. An example might be to send a value to the RTDI to publish to a data network. Unlike RTSEND, RTPOST *will* send a message during the processing of an incoming tick event
- SEE ALSO:** QUOTE
RTDEBUG
RTSEND

RTRANGE(start_date, start_time, end_date, end_time)

- PURPOSE:** Sets the date and time ranges for intraday and historical data queries. These parameters are applied (as appropriate) to queries accessed through MONITOR, BARMON, RTHISTORY, etc.
- start_date** Start date of data to be retrieved.
- start_time** Start time of data to be retrieved.
- end_date** End date of data to be retrieved.
- end_time** End time of data to be retrieved.
- RETURNS:** If no arguments are specified, a string showing the current date and time range settings, otherwise nothing.
- EXAMPLE:** RTRANGE("01/01/91",GETDATE())
sets the date range used in historical data retrievals from January 1, 1991 to the present day.
- REMARKS:** RTRANGE displays the range currently in effect if no arguments are provided.
- SEE ALSO:** BARMON
MONITOR
RTHISTORY

RTREADA(filename, column, append)

PURPOSE:	Reads a file periodically, replacing or updating worksheet data.
filename	A valid filename, in quotes.
column	(Optional). Integer. The file column to read. Columns are numbered beginning with 0. The default is 0.
append	(Optional). Integer. 0 = replace all data, 1 = append to data already present. The default is 0.
RETURNS:	A series or table.
EXAMPLE:	RTREADA("IBM.N",1,0) periodically reads the second column of file "IBM.N", replacing the data in the window with the values found in the file.
REMARKS:	Only ASCII data is supported. The evaluation interval is set via RTINTERVAL
SEE ALSO:	RTHISTORY RTON

RTSEND(string)

PURPOSE:	Sends an arbitrary string to the real-time Data Interface.
string	Any string, in quotes.
RETURNS:	Nothing.
EXAMPLE:	RTSEND(STRCAT("Message From:",STRNUM(hostid)))
REMARKS:	RTSEND is used to communicate with a data service or other external services. See the MarketBrowser Real-Time Data Interface (API) documentation for further discussion.
SEE ALSO:	QUOTE RTDEBUG

RTSTATUS(what_to_do, which_to_do, string)

PURPOSE:	To get, set, or clear a string describing the status of the Real Time Data Interface (RTDI).
what_to_do	Integer value indicating: <ul style="list-style-type: none">• 0 - clear the status string• 1 - report the status string• 2 - set the status string
which_to_do	Integer. What to clear, report or set:

- 1 - Query Table
- 2 - Query String
- 3 - Query Number
- 4 - Register Number
- 5 - Register Page
- 6 - Data Diagnostic
- 7 - Data Public

string The string to set. Applies only when setting the status.

RETURNS: The string associated with the latest transaction of the type specified in "which_to_do." The string may be empty. These status strings can be set by this function or by receipt of DD or DP messages from the RTDI.

REMARKS: Typical usage is for an exposrv module to send a data command with an RTSTATUS statement up to MarketBrowser to set a relevant message. The MarketBrowser applications layer then reports the status and clears it.

RTSUPPRESS(window, setting)

PURPOSE: Sets whether or not to suppress reevaluation of a window during real-time updates.

window (Optional.) The window to set. Defaults to the current window.

setting An integer. 1 - Turns on real-time suppression (meaning the window will not reevaluate when a real-time update occurs); 0 - Turns off real-time suppression; -1 - Reports the value of the RTSUPPRESS setting.

RETURNS: An integer.

EXAMPLE In a window that contains a custom window formula that performs a very computationally-intensive calculation:

```
RTSUPPRESS(1)
```

causes the window to not be reevaluated and to perform the calculation again when a tick occurs for the data on which the window depends.

REMARKS: Use care when applying this function because it sets an attribute of the window itself and not simply data or a study within the window. For example, while it is appropriate to apply this function to a window that contains a window formula like "=LINREG(W1)", it is not necessarily appropriate to use it in a window where the Studies/Trend menu function has been applied to data in a window. If the Trend study is dropped, the RTSUPPRESS setting will not be dropped and the data in the window will not update. If the window is cleared, the RTSUPPRESS attribute will be set back to the default (of 0).

SEE ALSO: CALC RTDEPEND
DEPEND

RUN(filename, pause, noclear)

PURPOSE: Runs an external program from an open worksheet.

filename Name of program to run, in quotes.

pause (Optional). 1 runs the specified program but will pause before returning to the MarketBrowser screen. Any keystroke will bring you back to the worksheet. The default is 0 (no pause).

noclear (Optional). -1 runs the specified program without disturbing the current worksheet screen, i.e. in the background. MarketBrowser regains control when the external program has finished. The Default is -1 (do not disturb).

EXAMPLES: RUN("MYPROG", -1)
runs the program "MYPROG" without disturbing the current screen. MarketBrowser will continue execution when "MYPROG" has finished.

REMARKS: RUN can call executable programs written in any language.

SEE ALSO: GETENV
LOAD

SAVEWORKSHEET(wsname)

PURPOSE: Saves a worksheet under a specified name.

wsname The full path and name of the worksheet to be saved, in quotes.

RETURNS: 1 if the file is successfully saved, 0 if it could not be saved.

EXAMPLES: SAVEWORKSHEET("c:\expo\My worksheets\example.xpw")
SAVEWORKSHEET(GETWORKSHEET)

SEE ALSO: GETWORKSHEET
LOADWORKSHEET

SCALES(window, style)

PURPOSE: Sets the types of scales in a window.

window (Optional). A window reference. Defaults to the current window.

style Integer. Specifies the type of scales.

- 0 - No scales
- 1 - X bottom Y left
- 2 - X bottom Y left with labels
- 3 - X bottom Y right with labels
- 4 - X bottom Y right
- 5 - X top Y right
- 6 - X top Y left
- 7 - X top Y left with labels
- 8 - X top Y right with labels
- 9 - Y left
- 10 - Y right
- 11 - X bottom
- 12 - X top
- 13 - Y left with labels
- 14 - Y right with labels
- 15 - X bottom with labels
- 16 - X top with labels

RETURNS: Nothing

EXAMPLE: SCALES(8)

places the x-axis scale along the top of the plotting area, and the y-axis scale with its labels along the right side of the plotting area.

SEE ALSO: GETSCALES SCALESON
 SCALESOFF FOCUS
 OVERLAY SYNC

SCALESOFF, SCALESON

PURPOSE: Adds/removes x-y scales from the current window.

RETURNS: Nothing. Toggles the scales in a window and expands the series display slightly.

SEE ALSO: SCALES
 PLOTMODE
 FREEZE

SCHUR(matrix)

PURPOSE: Finds the Schur form of a matrix.

matrix An expression resolving to a real or complex square matrix.

EXAMPLES: $x =$

1	3	4
5	6	7
8	9	12

SCHUR(x)=

19.964	4.353	-2.2431
0.0	-1.4739	0.1399
0.0	0.0	0.50976

SCHUR(x)=

0.25387	0.96612	-0.046551
0.50456	-0.17334	-0.84579
0.82521	-0.19124	0.53147

$x = \text{SCHUR}(x) * \text{SCHUR}(x) * \text{TRNASPOSE}(\text{USCHUR}(x)) * \text{SCHUR}(x)$

gets a Schur matrix

USCHUR(x)

gets a unitary matrix.

$x = \text{MMULT}(\text{MMULT}(\text{USCHUR}(x), \text{SCHUR}(x)), \text{TRANSPOSE}(\text{USCHUR}(x)))$
 $\text{MMULT}(\text{TRANSPOSE}(\text{USCHUR}(x)), \text{USCHUR}(x))$

is an identity matrix which is the same size as x.

REMARKS: If matrix x is real, SCHUR returns the real Schur form which has the real Eigenvalues on the diagonal and the complex Eigenvalues in 2-by-2 blocks on the diagonal. If matrix x is complex, Schur returns the upper triangular with the Eigenvalues of the matrix on the diagonal.

SEE ALSO: USCHUR

SCREENOPT(legends, titles, wbar, wborder, wmargin)

PURPOSE:	Selects worksheet elements to be visible or hidden from the screen display.
legends	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Legends are text annotations in the windows.
titles	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Titles are text annotations on the worksheet.
wbar	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wbar specifies the text for the window number, window formula and/or window label.
wborder	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wborder specifies the outer border outline of each window.
wmargin	(Optional). An integer value; 1 = ON, 0 = OFF, -1 = Keep current setting. Wmargin specifies the border outline on the inner window (separating the inner window from the window plotting margin).
RETURNS:	Nothing.
EXAMPLES:	<p>SCREENOPT(1,1,0,0,0)</p> <p>leaves legends and titles in the worksheet display, and disables the display of window bars, borders, and margins.</p> <p>SCREENOPT(-1,-1,1)</p> <p>leaves all the settings as they currently are, and enables the display of the window bars.</p>
REMARKS:	SCREENOPT is useful in formatting a worksheet for presentations, demonstrations, and custom applications. All parameters are optional integer arguments, defaulting to current values. Use -1 to leave a parameter unchanged.
SEE ALSO:	PRINTOPT LAYOUT

SCROLLD, SCROLLU, SCROLLL, SCROLLR(amount)

PURPOSE:	Scrolls the current window.
amount	(Optional). The number of units to scroll the window. The default is 1/3 of window height.
RETURNS:	Nothing.
REMARKS:	The SCROLL commands act like their arrow key or button equivalents when no arguments are provided.

SEEDRAND(val)

PURPOSE: Sets the seed value for the random number generator.

val The real seed value.

RETURNS: Nothing.

REMARKS: MarketBrowser sets the same seed value every time it is started up.

Every random number in MarketBrowser is calculated in a deterministic way from the previous random number, except for the first random number, which is calculated from the seed. (In calculating the next random number, MarketBrowser does NOT refer to the clock or anything other than the previous random number or the seed). The random numbers generated cannot be distinguished from actual random numbers by statistical methods. However, because the formula is deterministic, such random numbers are sometimes called pseudo-random numbers.

Since MarketBrowser uses the same seed every time it starts up, all the random numbers are duplicated in every MarketBrowser session. You can vary this pattern by setting a seed of your choosing. When the seed is reset with this function, all random numbers are thenceforth calculated from the new seed.

SEE ALSO: GRANDOM

SERCOLOR(color)

PURPOSE: Modifies the series color in a window without changing the window color.

color Any pre-defined macro name for a color supported by MarketBrowser.

RETURNS: Nothing.

EXAMPLE: SERCOLOR(GREEN)
sets the current window's series color to green.

SEE ALSO: WINCOLOR

SERCOUNT(series)

- PURPOSE:** Counts the number of series in a window or table.
- series** (Optional). A series or table. Defaults to the current window.
- RETURNS:** An integer.
- EXAMPLE:** SERCOUNT(RAVEL(GRAND(100,.01),10))
Returns 10, the number of series (columns) created by the RAVEL command.
- SEE ALSO:** GETWCOUNT
COL
NUMCOLS
NUMROWS

SETAORIX, SETAORIY(window, orient)

- PURPOSE:** Sets the orientation of the axis labels.
- window** (Optional). Window reference. Defaults to the current window.
- orient** An integer value; 1 = Horizontal, 2 = Vertical.
- RETURNS:** Nothing
- EXAMPLE:** W1: gsin(100,.01);setvunits("Volts")
W2: gcos(100,.01);setvunits("Amps")
W3: w1;staggery(0);staggerx(0); Scales(2); sety(-4,1.5);spany(-1,1);
overlay(w2,red); focus(2); staggery(0); staggerx(0); scales(13); sety(-1.5,4); spany(-1,1); focus(1); setaoriy(1); focus(2); setaoriy(2)
Window 3 contains the 2 overlaid curves with flush scales and different axis label orientations. The axis labels for the sine wave are horizontal, while those for the cosine wave are vertical.
- REMARKS:** SETAORIX and SETAORIY apply to the scales associated with the current focus of the specified window. By setting the orientation, the label will be drawn horizontally or vertically, i.e. succeeding characters in a label string will be drawn below or above (vertical) or to the right or left (horizontal) of preceding characters. If the axis label is vertically oriented, then its rotation is defined to be the vertical label default rotation for its label type and axis.
- SEE ALSO:** SETTORIX, SETTORIY SETTVDEFX, SETTVDEFY
SETAVDEFX, SETAVDEFY SETTICK
SETAROTX, SETAROTY SETTROTX, SETTROT
OVERLAY FOCUS
SCALES

SETAVDEFX, SETAVDEFY(window, rotation)

PURPOSE: Sets the default rotation for labels on x- and y-axis.

window (Optional). Window reference. Defaults to the current window.

rotation An integer value:

- 0 = 90 degree rotation of entire string, all sides
- 1 = 270 degree rotation of entire string (i.e. 90 degrees CW), all sides
- 2 = Horizontal characters laid out in vertical sequence, all sides. Horizontal left-most letter becomes vertical topmost letter.
- 3 = 90 degree rotation of entire string for left and bottom sides; 270 degree rotation of entire string for right and top sides
- 4 = degree rotation of entire string for left and bottom sides; 90 degree rotation of entire string for right and top sides

RETURNS: Nothing.

EXAMPLE: W1: GRAND(100,.5,1); SCALES(2); SETAORIY(2); SETAVDEFY(4)

Now, click on the scales button in the toolbar (or F5), and see how the y-axis labels change their rotation as the scales move from the left side to the right side of the window.

REMARKS: SETAVDEFX and SETAVDEFY set the vertical default rotation for axis labels. The default settings are 90 degree rotation for x-axis labels, 90 degree rotation for y-axis labels on left and bottom sides, 270 degree rotation for y-axis labels on right and top sides.

SEE ALSO: SETTVDEFX, SETTVDEFY SETAORIX, SETAORIY
SETTORIX, SETTORIY SETTROT X, SETTROT Y
SETAROTX, SETAROTY OVERLAY
FOCUS SCALES

SETBUFSIZE(size)

PURPOSE: Sets the buffer size (the number of points that MarketBrowser keeps in main memory before using the disk for further storage).

size Number of points to buffer (between 101 and 32767).

RETURNS: Nothing.

REMARKS: The default values are 4500 points under Windows and 8192 points under UNIX. There is a trade-off between buffer size and the number of series a worksheet can process. If you are working with long data series, raising this parameter will improve throughput, but may cause an out-of-memory condition for worksheets with a large numbers of series.

SETBUFSIZE can also be used without the size argument to view the current buffer size.

Using the function SETBUFSIZE sets the configuration variable BUFSIZE, which means the new value is preserved even when the product is shut down and restarted. It is normally not necessary to use SETBUFSIZE.

SETCANVAS(x, y, w, h)

PURPOSE: Resizes/locates the application frame.

x When positive, distance of the left edge of the application frame from the left edge of the screen. When negative, distance from the right edge of the screen.

y When positive, distance of the top edge of the application frame from the top edge of the screen. When negative, distance from the bottom edge of the screen.

w The width of the application frame.

h The height of the application frame.

RETURNS: Nothing.

REMARKS: All parameters are expressed as fractions of the overall screen between 0.0 and 1.0.

SETCOLOR(color, series, index,)

- PURPOSE:** Sets the color of a series in a window.
- series** (Optional). A series or table. Defaults to the current window.
- color** An integer or macro (text) designating the color.
- index** (Optional). The series to set. Defaults to the first series.
- RETURNS:** Nothing.
- EXAMPLE:** SETCOLOR(GREEN, 2)
Sets second series in current window to GREEN.
- SEE ALSO:** GETWCOLOR WINCOLOR

SETCOMMENT(window, string, item)

- PURPOSE:** Sets the comment for the first series in a window.
- window** (Optional). Defaults to the current window.
- string** Any text, in quotes.
- item** (Optional). Index to the series in the window that you wish to comment. Defaults to the first item.
- member** (Optional). Integer. Which member (column) of the item. Defaults to the first member.
- RETURNS:** Nothing
- EXAMPLE:** SETCOMMENT(W4, strcat("IBM as of ", getdate))
places "IBM as of 04/14/89" into the current comment field.
- SEE ALSO:** COMMENT (shorthand) GETCOMMENT
GETSCOMMENT LABEL

SETCONF(item, value)

- PURPOSE:** (A Macro). Sets the item named in the string "item" to the value in the string "value".
- item** A string enclosed in quotes that represents the configuration variable that you wish to modify.
- value** The value to which you would like to set the configuration variable, in quotes.
- RETURNS:** A string representing the value of the specified configuration variable.
- EXAMPLE:** SETCONF("PLOT_STYLE", "1")
sets the plot style configuration variable to 1, which represents points.

REMARKS: SETCONF saves changes to the configuration variables into your session file.

SEE ALSO: GETCONF

SETDATE, SETTIME(date/time)

PURPOSE: Sets the beginning date or time for a series in the current window and propagates any changes to dependent windows.

date/time Date or time string, as appropriate.

RETURNS: Nothing.

EXAMPLE: SETDATE("12/01/82")

Sets the origin of the series to December 1, 1982, and causes the scales to be rewritten with appropriate labeling.

SEE ALSO: DEFDATE DEFTIME
GETDATE GETTIME

SETDEGREE

PURPOSE: Changes the mode of MarketBrowser trigonometric functions to degrees.

RETURNS: Nothing.

EXAMPLE: After invoking SETDEGREE, SIN(90) yields 1.

After invoking SETRADIAN, SIN(90) yields 0.89399666.

SEE ALSO: SETRADIAN DEG
E GAMMA
PI

SETDELTA(newdelta-x)

PURPOSE: Modifies the spacing between points on the horizontal axis of the current window.

newdelta-x Numeric value of new delta-x.

RETURNS: Nothing.

EXAMPLE: SETDELTA(0.2)

Sets the current DELTAX to 0.2 (a sampling rate of 5).

REMARKS: SETDELTA causes MarketBrowser to recalculate the scales of windows depending on the DELTAX change. Series requiring recalculation based on a new DELTAX will no longer be correct. These include integrals, derivatives, and expressions using DELTAX explicitly, such as:

W3: W2 * DELTAX(W2)

SEE ALSO: DELTAX RATE
SETXOFFSET

SETDTFORMAT(series, style, format)

PURPOSE: Sets the date/time formatting in a window.

series (Optional). String. Window or variable reference, in quotes. Defaults to the current window.

style Integer, which specifies which date format you wish to change, where:

- 0 = short format
- 1 = long format

format Integer. The integer number corresponding to the date time format you wish to set it to.

RETURNS: An integer representing the date format

EXAMPLE: W1: SETDTFORMAT("W2", 0, 8)
defines the short format date and time display in W2 to be hh:mm:ss (e.g. 11:59:59).

REMARKS: For the integer values corresponding to valid date and time formats, please refer to the INDEXTODT function definition. The short format refers to the dates and times displayed on time (x) axis, while the long format refers to the manner in which MarketBrowser displays dates in a cursor view, or in status messages at the bottom of the screen.

SEE ALSO: INDEXTODT
GETDTFORMAT

SETFORMAT(mode)

PURPOSE: Sets the display type for numerical values, including table listings.

mode	Mode	Meaning
	-1	Auto format (default).
	0	Regular floating notation, without exponent.
	1	Exponential notation with uppercase 'E'.
	2	Picks mode 0 or 1, based on value, using uppercase 'G'.
	3	Exponential notation with lowercase 'e'.
	4	Picks mode 0 or 1, based on value, using lowercase 'g'.

RETURNS: Nothing.

REMARKS: Use in conjunction with SETPRECISION to control the appearance of numerical values. This function affects the display of numerical values everywhere in the worksheet, not just the current window.

SETGCOLOR(color_param, color)

PURPOSE:	Dynamically sets a global MarketBrowser color parameter.
color_param	Integer. Corresponds to the color-related parameters.
color	Integer. Represents any color supported by MarketBrowser.
RETURNS:	Nothing.
EXAMPLE:	SETGCOLOR(2, 9) sets the 2nd parameter (background color) to the color that is represented by the integer 9.
REMARKS:	The color parameters are defined and documented in the file, dspcolor.
SEE ALSO:	GETGCOLOR

SETHIGHWATER(series, num)

PURPOSE:	Sets the high-water mark for a real-time dependent series.
series	(Optional). Quoted string. Valid window reference or variable. Variables must contain a series. Defaults to current window.
num	Positive integer. The number of points in the dependent series with which MarketBrowser is to recalculate a new point at the end of a real-time interval.
RETURNS:	1 if the high-water mark is successfully set, otherwise 0.
REMARKS:	Set num to 0 to turn off high-water processing for the given series.
SEE ALSO:	GETHIGHWATER

SETHOTVARIABLE(name, value)

PURPOSE: Sets a hot variable.

name A string. The name of the variable.

value Any scalar, series, table, or expression resulting in a scalar, series, or table.

RETURNS: Nothing.

EXAMPLE: SETHOTVARIABLE(Rate_Of_Change, DERIV(W1))

Sets the hot variable Rate_Of_Change to the derivative of window 1. Whenever window 1 changes, as when a real-time update comes in to W1, Rate_Of_Change will be updated.

REMARKS: Hot variables are linked such that if the dependent value changes, then the value for the hot variable changes.

Hot variables can also be assigned with the syntax:

Rate_Of_Change := DERIV(W1)

SEE ALSO: SETHOTVAR (shortcut name) DELALLVARIABLES
DELVARIABLE GETVARIABLE
SETVARIABLE SETLOCALVARIABLE
XPLREAD VARS

SETHUNITS(name)

PURPOSE: Sets the horizontal units for a series in the current window and, to a limited extent, propagates any change to any dependent windows.

name The unit name, in quotes.

RETURNS: Nothing.

EXAMPLE: SETHUNITS("Days")

sets the horizontal units of the series in the current window to days.

SEE ALSO: SETVUNITS SETVHUNIT
GETHUNITS DEFHUNITS
SETXLABEL CLEARXLABEL

SETLINE(linestyle, series index)

- PURPOSE:** Sets the line style of the data.
- linestyle** 1: SOLID
2: DASHED
3: DOTTED
- series index** (Optional). The particular series in a window whose line style you want to set. Defaults to the first series.
- RETURNS:** Nothing.
- EXAMPLE:** SETLINE(3,2)
sets the second series of the current window to a dotted line.

SETLINEWIDTH(width, series index)

- PURPOSE:** Sets the width of a series line, where width is measured in pixels.
- width** Measured in pixels.
- series index** (Optional). The series whose line width you want to set. Defaults to the first series.
- RETURNS:** A series line with a specified thickness.
- EXAMPLE:** SETLINEWIDTH(1,2)
sets the second line series in the window to a thickness of 1 pixel.

SETLOCALVARIABLE(name, value)

- PURPOSE:** Creates a local variable and assigns a value to it.
- name** The name of the local variable, optionally in quotes.
- value** The assigned value to the local variable.
- RETURNS:** Nothing.
- EXAMPLE:** SETLOCALVARIABLE("localvar", 34)
creates a local variable "localvar", with a value of 34.
- SEE ALSO:** SETLOCALVAR (shortcut name) GETLOCALVARIABLE
SETVARIABLE GETVARIABLE

SETMACDEPTH(level)

- PURPOSE:** Sets the maximum macro depth, which is the number of macros that can be nested inside each other.
- level** (Optional). Integer. The number of levels.
- RETURNS:** The maximum macro depth.
- REMARKS:** The default max macro depth is 50. If the level argument is given the new max macro depth will be returned. If no argument is specified the current max macro depth will be returned. A setting of 0 disables all macros.

SETMATRIX(OnOff)

- PURPOSE:** Treats the data in a window as a two dimensional matrix of data or simply as a list of unrelated series.
- OnOff** Integer. 1= ON; 0 = OFF.
- RETURNS:** Nothing.
- EXAMPLE:** READTABLE("mytable.dat");SETMATRIX(0)
reads in a table of data, which is interpreted as a matrix by default, and then changes the characterization of the data from a matrix to individual series.
- REMARKS:** Because trading bars/candlesticks are subject to matrix style propagation, SETMATRIX can also be used to turn a trading bar into a set of unrelated individual line graphs.

SETNAVALUE(window, value)

- PURPOSE:** Replaces the NA values in a window with user-defined input.
- window** (Optional). A window reference. Defaults to the current window.
- value** Value used to replace all NAs in the specified window.
- RETURNS:** Series, table or number.
- EXAMPLE:** SETNAVALUE(CURR, 100)
sets all NA values in the current window to 100.
- SEE ALSO:** ISNAVALUE NAFILL
NAVALUE CONFORM

SETPALETTE(color1, ..., colorn)

PURPOSE: Sets an ordered list of colors to use in shading.

color1, ..., colorn Integer or macro name representing a color.

RETURNS: Nothing.

EXAMPLE: SETPALETTE(YELLOW, GREEN, BLUE, PURPLE, RED, BLACK)
sets a palette of six colors for use by plots that employ shading. In a DENSITY plot, for example, points falling within the first sixth of the range of the data are colored YELLOW, those in the next sixth are GREEN, and so on.

REMARKS: The range of colors available is machine dependent. The color names shown in the example are actually pre-defined macros representing some of the colors available on any color machine. Each window can have its own palette of colors.

SEE ALSO: SETSHADING SHADEWITH
 CONTOUR DENSITY

SETPLOTMETHOD(method)

PURPOSE: Sets the method of plotting a waterfall.

method Integer. 0 = Horizon method; 1 = Painter's method. The default is (-1) cross hatches at each observation.

RETURNS: A plot.

REMARKS: In general, MarketBrowser uses the Painter's method except when sending output to a pen plotter. By default, surfaces are drawn with cross hatches at each observation. The lines are drawn using the grid color and then filled using the data series color.

SETPLOTSTYLE(series, style, index)

PURPOSE:	Sets the plotting style of a series in a window.
series	(Optional). A series or table. Defaults to the current window.
style	Integer. Valid styles are: <ul style="list-style-type: none">• 0 = Lines• 1 = Points• 2 = Sticks• 3 = Filled Bars• 4 = Trading Bars• 5 = Table of Numbers
index	(Optional). The series to set. Defaults to the first series.
RETURNS:	Nothing.
EXAMPLE:	SETPLOTSTYLE(2,2) sets the second series in the current window to sticks.
REMARKS:	SETPLOTSYTLLE offers more control than the simple plot style functions like POINTS, BARS, LINES, TICKFORM, etc.
SEE ALSO:	SETPLOTTYPE INHSERSTYLE INHWINSTYLE

SETPLOTTYPE(type)

PURPOSE:	Sets the plotting type for a table of data.
type	Integer. Valid types are: <ul style="list-style-type: none">• 0 = Lines• 1 = Waterfall• 2 = Contour• 3 = Density
EXAMPLE:	SETPLOTTYPE(3) shows the table in the current window as a density plot.
REMARKS:	SETPLOTTYPE is only meaningful for windows that have table/matrix data. SETPLOTTYPE sets the "major" plotting type. You might use it to set the window to be a waterfall plot, and then use BARS to change from a surface to 3D bars as the "minor" plotting style.
SEE ALSO:	SETPLOTSTYLE

SETPRECISION(prec)

- PURPOSE:** Sets the number of significant digits after the decimal point.
- prec** An integer specifying how many digits after the decimal point you wish to see.(-1) returns the default precision level, that is, 8.
- RETURNS:** All the displayed values in a worksheet set to a specified precision.
- EXAMPLE:** SETPRECISION(2)
displays values with two decimal places.
- REMARKS:** All numerical values in a worksheet are affected by SETPRECISION, not just the values in the selected window. However, only the display of a value is affected by SETPRECISION, not the internal precision used in calculations.

SETPT(series, val, index)

- PURPOSE:** Modifies the value of a single point in a series: the indicated scalar value is assigned to the point with the given index.
- series** (Optional). A series or table. Defaults to the current window.
- val** A number.
- index** Integer. A point index.
- RETURNS:** A series or table.
- REMARKS:** An index is a "point number"; index n refers to the nth point in the series.
- SEE ALSO:** GETPT

SETRADIAN

- PURPOSE:** Changes the mode of MarketBrowser trigonometric functions to radians.
- RETURNS:** Nothing.
- EXAMPLE:** After invoking SETRADIAN:
SIN(PI/2) yields 1.
But if SETDEGREE is invoked, then:
SIN(PI/2) yields .027412134.
- REMARKS:** This function has no effect unless SETDEGREE has been invoked.
- SEE ALSO:** SETDEGREE

SETSCSTYLE(series, color, sync, staggery, scales, ticks, partition, span_y_b, span_y_t, ymin, ymax)

PURPOSE:	Sets the series style for given window's focus.
series	(Optional). The focused series. Optional window argument. Defaults to current window.
color	(Optional). Integer or macro color name of the overplotted series
sync	(Optional). Sync mode. How the overlaid series scrolls along the horizontal and vertical axes in relation to the window's primary series. See a complete table of synchronization options under SYNC. Defaults to 0, or no sync.
staggery	(Optional). Integer flag. Options are: <ul style="list-style-type: none">• 1 - Stagger y-axis scales vertically (default).• 0 - Keep y-axis scales flush with plotting area.
scales	(Optional). Scale setting. Defaults to 5 (x top y right). See the SCALES function for a complete description of available options.
ticks	(Optional). Number of ticks marks to place along the y-axis.
partition	When PARTITION is 1, the four arguments SPANYBOTTOM, SPANYTOP, YMIN and YMAX are interpreted as defining a hard partition of the drawing area of the MarketBrowser window. SPANYBOTTOM and SPANYTOP are used to set the Y-AXIS span. If they are set to 0.0, and 0.0 this indicates that the AXIS should autocalculate itself based on the data in that subwindow. Setting these to 0.0 and 100.0 would force the y-axis scales to have that range. YMIN and YMAX are used to set the portion of the MarketBrowser window that the subwindow should occupy. YMIN and YMAX are reals ranging from 0.0 to 1.0. Setting YMIN = 0.0 and YMAX = 0.5 would make the subwindow occupy the bottom half of the MarketBrowser window.
span_y_b	(Optional). Lower end of the y-axis scale's range.
span_y_t	(Optional). Top of the y-axis scale's range.
ymin	(Optional). Real number spanning from 0.0 to 1.0, representing the lower placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
ymax	(Optional). Real number spanning from 0.0 to 1.0, representing the upper placement of an overlay along the y-axis, where the y-axis spans from 0.0 (bottom of plotting area) to 1.0 (top of plotting area). Defaults to the normal MarketBrowser plotting area.
RETURNS:	Nothing.
EXAMPLE:	Given the following window formulas: W1: READAHIST('IBM.CLS','D',2,1)

W2: MOVAVG(W1,15)

W3: W1;OVERLAY(W2,black,1,0,5,ticks,-1, -1,0.1,0.3)

copies the series in W1 into the window, and transparently overlays the series from W2. The two series scroll together along the horizontal and vertical axes, the scales of the overlay are kept flush with the plotting area, the default scale style is used, 5 tick marks are plotted along the y-axis, and the overlay is contained between 0.1 and 0.3 of the y-axis.

REMARKS: You can overlay an unlimited number of series into a single window. Each overlay has an independent set of scales associated with it. The concept of focus applies to overlaid series. Specify which series in a window is the current focus either with the FOCUS function, or by clicking on the series' scale.

SEE ALSO: FOCUS
OVERPLOT
SYNC

SETSHADING(startcolor, endcolor)

PURPOSE: Sets the range of colors used in shading on systems which support dynamic color assignment.

startcolor Integer or macro name (text) representing a color name, in quotes.

endcolor Integer or macro name (text) representing a color name, in quotes.

RETURNS: Nothing

EXAMPLE: SETSHADING("Spring Green", "Slate Blue")

sets a smoothly graded palette of colors for use by plots which employ shading. The number of shades created depends on machine resources available. See the discussion on color in the User Manual.

REMARKS: SETSHADING is not currently supported under Windows. SETSHADING sets the shading scheme for the entire worksheet, but each window can use either the smooth shades or its private palette of discrete colors (see SETPALETTE). For any given window, the discrete colors are in effect by default until SETSHADING is used in that window.

SEE ALSO: SETPALETTE
SHADEWITH
CONTOUR
DENSITY

SETSYMBOL(window, symbol, series index, interval, offset)

PURPOSE: Defines symbols for overlaid series.

window (Optional) The window reference; defaults to the current window.

symbol Integer representing symbols

- 0 = No symbols
- 1 = Box
- 2 = Triangle
- 3 = Upside down triangle
- 4 = +
- 5 = X
- 6 = Empty up arrow
- 7 = Empty down arrow
- 8 = Filled up arrow
- 9 = Filled down arrow
- 10 = Empty diamond
- 11 = Empty square
- 12 = Empty triangle
- 13 = Empty upside down triangle

series index (Optional) Number of the series to be changed; defaults to the first series.

interval (Optional) Data point interval for symbol. Defaults to 1.

offset (Optional) Start point for symbol. Defaults to 1.

RETURNS: Nothing.

EXAMPLE: SETSYMBOL(1,2)
sets a box as the symbol that surrounds every data point of the second series in the current window.

SETSYMBOL(W2,1,2)
does the same for the second series in W2.

SETSYMBOL(W2,3,4,5,6)
sets an upside down triangle every fifth point starting at the sixth point for the fourth series in W2.

SETTICK(xtic, ytic)

PURPOSE: Sets the tic spacing on the axes.

xtic A number. The xtic interval.

ytic A number. The ytic interval.

RETURNS: Nothing.

EXAMPLES: SETTICK(.01)
sets the x tic interval on the current window to 0.01

REMARKS: If you specify a negative interval, the tic intervals are automatically calculated. Thus, to set the Y axis interval while leaving X unchanged, you might use SETTICK(-1,0.125).

SETTORIX, SETTORIY(window, orient)

PURPOSE: Sets the orientation of the labels associated with the tick marks on axes.

window (Optional). Window reference. Defaults to the current window.

orient An integer value; 1 = Horizontal, 2 = Vertical. Defaults to 1 (Horizontal).

RETURNS: Nothing.

EXAMPLE: W1: GSIN(100,.01);SETVUNITS("Volts")
W2: GCOS(100,.01);SETVUNITS("Amps")
W3: W1; STAGGERY(0); STAGGERX(0); SCALES(2); SETY(-4,1.5); SPANY(-1,1); OVERLAY(w2,red); FOCUS(2); STAGGERY(0); STAGGERX(0); SCALES(13); SETY(-1.5,4); SPANY(-1,1); FOCUS(1); SETTORIY(1); FOCUS(2); SETTORIY(2)

Window 3 contains the 2 overlaid curves with flush scales and different tick label orientations. The tick labels for the sine wave are horizontal, while those for the cosine wave are vertical.

REMARKS: SETTORIX and SETTORIY apply to the scales associated with the current focus of the specified window. By setting the orientation, the label will be drawn horizontally or vertically, i.e., succeeding characters in a label string will be drawn below or above (vertical) or to the right or left (horizontal) of preceding characters. If the tick label is vertically oriented, then its rotation is defined to be the vertical label default rotation for its label type and axis.

SEE ALSO: SETAORIX, SETAORIY
SETTVDEFX, SETTVDEFY
SETAVDEFX, SETAVDEFY
SETTROT X, SETTROT Y
SETAROT X, SETAROT Y
SETTICK OVERLAY
FOCUS SCALES

SETTROTX, SETTROTY(window, rotation)

- PURPOSE:** Sets the rotation for tick labels on x- and y-axis.
- window** (Optional). Window reference. Defaults to the current window.
- rotation** An integer value:
- 0 = No Rotation
 - 1 = 90 degree rotation of entire string
 - 2 = 180 degree rotation (i.e. upside down)
 - 3 = 270 degree rotation (i.e. 90 degrees clockwise)
 - 4 = Horizontal characters laid out in vertical sequence (horizontal leftmost letter becomes vertical topmost letter.)
- RETURNS:** Nothing.
- EXAMPLE:**
- ```
W1: GSIN(100,.01,4);scales(3);settroty(0); Label("No Rotation")
W2: GSIN(100,.01,4);scales(3);settroty(1); Label("90 deg Rotation")
W3: GSIN(100,.01,4);scales(3);settroty(3); Label("270 deg Rotation")
W4: GSIN(100,.01,4);scales(3);settroty(4); Label("Character Rotation")
```
- REMARKS:** Setting rotation explicitly via SETTROTX, SETTROTY functions is intended for users who want to fine-tune a plot presentation for printing. It is recommended that the user use the SETTORI-, SETAORI-, SETTVDEF-, and SETAVDEF- functions. SETTROTX and SETTROTY apply to the scales associated with the current focus of the specified window. If you are executing the SCALES command, pressing the F5 key, or the toolbar button changes the rotation of the tick labels, they have been replaced by default orientation and vertical default rotations.
- SEE ALSO:** SETAROTX, SETAROTY  
SETTORIX, SETTORIY  
SETAORIX, SETAORIY  
SETTVDEFX, SETTVDEFY  
SETAVDEFX, SETAVDEFY  
SETTICK  
OVERLAY  
FOCUS  
SCALES

## SETTVDEFX, SETTVDEFY(window, rotation)

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                                               |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the vertical default rotation for tic labels on x- and y-axis.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |                                                               |
| <b>window</b>    | (Optional). Window reference. Defaults to the current window.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                                               |
| <b>rotation</b>  | An integer value: <ul style="list-style-type: none"><li>• 0 = 90 degree rotation of entire string, all sides</li><li>• 1 = 270 degree rotation of entire string (i.e. 90 degrees CW), all sides</li><li>• 2 = Horizontal characters laid out in vertical sequence, all sides. Horizontal left-most letter becomes vertical topmost letter.</li><li>• 3 = 90 degree rotation of entire string for left and bottom sides 270 degree rotation of entire string for right and top sides</li><li>• 4 = 270 degree rotation of entire string for left and bottom sides 90 degree rotation of entire string for right and top sides</li></ul> |                                                               |
| <b>RETURNS:</b>  | Nothing.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                               |
| <b>EXAMPLE:</b>  | W1: GRAND(100,.5,1); SCALES(2);SETTORIY(2); SETTVDEFY(3)<br>Now, click the scales button in the toolbar and see how the y-axis tic labels change their rotation as the scales move from the left side to the right side of the window.                                                                                                                                                                                                                                                                                                                                                                                                 |                                                               |
| <b>REMARKS:</b>  | The default settings are 90 degree rotation for x-axis tic labels, 90 degree rotation for y-axis tics on left and bottom sides, 270 degree rotation for y-axis tics on right and top sides.                                                                                                                                                                                                                                                                                                                                                                                                                                            |                                                               |
| <b>SEE ALSO:</b> | SETAVDEFX, SETAVDEFY<br>SETAROTX, SETAROTY<br>SETTORIX, SETTORIY<br>FOCUS                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | SETTROTX, SETTROTY<br>SETAORIX, SETAORIY<br>OVERLAY<br>SCALES |

## SETVARIABLE (name, value)

|                  |                                                                                  |                                    |
|------------------|----------------------------------------------------------------------------------|------------------------------------|
| <b>PURPOSE:</b>  | Creates an XPL variable and assigns it a value.                                  |                                    |
| <b>name</b>      | The variable name, optionally in quotes.                                         |                                    |
| <b>value</b>     | The value of the variable.                                                       |                                    |
| <b>EXAMPLE:</b>  | SETVARIABLE("myvar", 10)<br>creates a global variable "myvar" with the value 10. |                                    |
| <b>SEE ALSO:</b> | SETVAR (shortcut name)<br>GETVARIABLE                                            | SETLOCALVARIABLE<br>SETHOTVARIABLE |

## **SETVCOMMENT(vname, comment, item, member )**

|                  |                                                                                                                                                                |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the comment field of a variable.                                                                                                                          |
| <b>vname</b>     | Window or variable reference in quotes. Variables must contain a series.                                                                                       |
| <b>comment</b>   | Quoted string. Comment to attach to variable.                                                                                                                  |
| <b>item</b>      | (Optional). Integer. Window item number, i.e., which overplot or overlay. Defaults to the first item.                                                          |
| <b>member</b>    | (Optional). Integer. Which member (column) of the item (default 1).                                                                                            |
| <b>RETURNS:</b>  | 1 if variable is successfully set, otherwise 0.                                                                                                                |
| <b>EXAMPLE:</b>  | Given the following formulas:<br>myvar := MONITOR("IBM.LAST")<br>SETVCOMMENT("myvar", "IBM.LAST")<br>sets the comment for the variable myvar to be "IBM.LAST". |
| <b>SEE ALSO:</b> | SETCOMMENT                                                                                                                                                     |

## **SETVDATE, SETVTIME(var, date/time)**

|                  |                                                                                                                     |
|------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the beginning date or time for a series in a variable and propagates any changes to dependent windows.         |
| <b>var</b>       | Quoted string. The name of a variable, in quotes, that contains series data.                                        |
| <b>date/time</b> | Date or time string, as appropriate.                                                                                |
| <b>RETURNS:</b>  | Nothing.                                                                                                            |
| <b>EXAMPLE:</b>  | SETVDATE("myvar", "12/01/82")<br>Sets the origin of the series contained in the variable myvar to December 1, 1982. |
| <b>SEE ALSO:</b> | SETDATE<br>SETTIME                                                                                                  |

## SETVHUNIT(variable, unit, item, member)

- PURPOSE:** Sets the horizontal units for a series contained in an variable, to a limited extent, propagates any change to dependent windows.
- variable** String in quotes. The variable name of the series that you want to set.
- unit** String in quotes. The unit name.
- item** (Optional). Integer. The index to the series item.
- member** (Optional). Integer. The index to the series member.
- RETURNS:** Nothing.
- EXAMPLE:** Given the formulas:  
a := monitor("DEM=.BID")  
W1: a; setvhunit("a", "NU")  
sets the horizontal units of the one series to "NU", or "no units".
- REMARKS:** MarketBrowser attempts to reduce subsequent vertical unit calculations in terms of the above units. If the unit string is not one internally defined by MarketBrowser, the string is still assigned to the vertical unit, but further unit reduction is not performed.
- SEE ALSO:** SETHUNITS            SETVVUNIT  
GETHUNITS

## SETVITEMTYPE(varname,type)

- PURPOSE:** Used to set an item explicitly.
- varname** The name of a window or variable. This must be a string rather than a reference to a variable.
- type** 0 = Series. 1 = XY pairing. 2 = Matrix. 3 = Trading bars. 5 = Equivolume structure. The default is 0.
- RETURNS:** Nothing.
- EXAMPLE:** If you have a single series of Close, High, Low, and Open values in windows 1 through 4,  
AAA = ravel(w1..w4); setvitemtype("AAA",3)  
creates a variable with a trading bar item.
- REMARKS:** SETVITEMTYPE is a generalization of the SETMATRIX function. SETVITEMTYPE(0) is equivalent to SETMATRIX(0); SETVITEMTYPE(2) is equivalent to SETMATRIX(1).
- SEE ALSO:** SETMATRIX            RAVEL  
SETPLOTSTYLE            SETPLOTTYPE

## SETPLOTSTYLE(*vari\_name*, *plotstyle*, *item*, *member*)

|                  |                                                                                                                                                                                                                             |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets plotting style of a variable.                                                                                                                                                                                          |
| <b>var_name</b>  | Quoted string. Valid window or variable reference that contains a series.                                                                                                                                                   |
| <b>plotstyle</b> | Integer. The style of the plotted series. Options are: <ul style="list-style-type: none"><li>• 0 - lines</li><li>• 1 - points</li><li>• 2 - sticks</li><li>• 3 - bars</li><li>• 4 - ticks</li><li>• 5 - tableview</li></ul> |
| <b>item</b>      | (Optional). Integer. Window item number, i.e., which overplot or overlay. Defaults to first item in window.                                                                                                                 |
| <b>member</b>    | (Optional). Integer. Member (column number) of item. Defaults to first member.                                                                                                                                              |
| <b>RETURNS:</b>  | Nothing.                                                                                                                                                                                                                    |
| <b>EXAMPLE:</b>  | Given the variable:<br>myvar := MONITOR("IBM.LAST")<br>SETPLOTSTYLE("myvar", 3)<br>sets the variable's plotting style to be bars.                                                                                           |
| <b>SEE ALSO:</b> | SETPLOTSTYLE                                                                                                                                                                                                                |

## SETVUNITS(*name*)

|                  |                                                                                                                                                                                                                                                                                                                                               |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the vertical units for the series in the current window and, to a limited extent, propagates any change to dependent windows.                                                                                                                                                                                                            |
| <b>name</b>      | The vertical unit, in quotes .                                                                                                                                                                                                                                                                                                                |
| <b>RETURNS:</b>  | Nothing.                                                                                                                                                                                                                                                                                                                                      |
| <b>EXAMPLE:</b>  | SETVUNITS("\$")<br>Sets the vertical units of the series in the current window to Price.<br><br>MarketBrowser attempts to reduce subsequent vertical unit calculations in terms of the above units. If the unit string is not internally defined by MarketBrowser, the string is still assigned, but further unit reduction is not performed. |
| <b>SEE ALSO:</b> | SETHUNITS<br>GETVUNITS<br>SETVVUNIT                                                                                                                                                                                                                                                                                                           |

## SETVVUNIT(variable, unit, item, member)

**PURPOSE:** Sets the vertical units for a series contained in an variable, to a limited extent, propagates any change to dependent windows.

**variable** String in quotes. The variable name of the series that you want to set.

**unit** String in quotes. The unit name.

**item** (Optional). Integer. The index to the series item.

**member** (Optional). Integer. The index to the series member.

**RETURNS:** Nothing.

**EXAMPLE:** Given the formulas:  
a := monitor("DEM=BID")  
W1: a; setvvunit("a", "US \$")  
sets the vertical units of the one series to "US \$".

**REMARKS:** MarketBrowser attempts to reduce subsequent vertical unit calculations in terms of the above units. If the unit string is not one internally defined by MarketBrowser, the string is still assigned to the vertical unit, but further unit reduction is not performed.

**SEE ALSO:** SETVHUNITS                      SETVUNITS  
                  GETVUNITS

## SETWFORM(window, formula)

**PURPOSE:** Sets the formula for a window.

**window** (Optional) A window reference. Defaults to the current window.

**formula** Any valid MarketBrowser statement, in quotes.

**RETURNS:** Nothing.

**SEE ALSO:** SETWF (shorthand)              GETWFORMULA  
                  GETVFORM                      ADDWFORM

## SETWKSATTRIBUTE(attribute, setting)

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets various worksheet locking attributes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>attribute</b> | <p>A string enclosed in quotes that represents the attribute of the worksheet you wish to set. Choose from:</p> <ul style="list-style-type: none"><li>• "WKSLOCK" - Locks (1) the worksheet so changes cannot be made to it with SETWKSATTRIBUTE or unlocks it (0).</li><li>• "PASSWORD" - Sets the password used to lock and unlock a worksheet. For the setting argument, type the password within quotes.</li><li>• "WKSPASSLOCK" - Same as "WKSLOCK" above except you give an additional argument of a password within quotes to be used for locking and unlocking.</li><li>• "READONLY" - Sets whether the worksheet is read-only. The setting argument for this can be 0 = off, 1 = on (cannot be saved, deleted or copied over), or 2 = read-only (cannot be copied or saved under a new name).</li><li>• "WKSHELP", - Specifies the worksheet-specific help file that appears when the [?] button is selected. Enter the name for a Windows help file in quotes for the setting argument.</li><li>• "WKSMENU", - Specifies the worksheet-specific menu that appears when the / button is pressed. Enter the name of a text file in quotes for the setting argument.</li><li>• "WINMENU" - Specifies the worksheet-specific menu for an activated window that appears when the / button is pressed. Enter the name of a text file in quotes for the setting argument.</li><li>• "ADDREMOVE" - Turns on (1) or off (0) the user's ability to add and remove windows.</li><li>• "HIDEDISPLAY" - Turns on (1) or off (0) the user's ability to display and hide windows.</li><li>• "TILEARRANGE" - Turns on (1) or off (0) the user's ability to modify the window layout with the Tile, Neaten and Arrange commands.</li><li>• "MOUSERESIZE" - Turns on (1) or off (0) the user's ability to change the size or position of windows with the mouse.</li><li>• "CURSORINFO" - Sets the level of the window information listed in the command (formula) line when it is activated. 0 = formula, 1 = label, or 2 = nothing</li><li>• "OVERRIDE_REFRESH_POLICY" - Sets whether a worksheet will never automatically refresh when opened (1) or whether it will follow the global configuration variable RERESH_POLICY.</li></ul> |
| <b>setting</b>   | What to set the attribute to. See the attribute argument above for what integer or string to use for this argument.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>RETURNS:</b>  | Nothing.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>EXAMPLES:</b> | <pre>SETWKSATTRIBUTE("WKSLOCK",1)</pre> <p>locks the worksheet.</p> <pre>SETWKSATTRIBUTE("WKSPASSLOCK",1,"MYPASS")</pre> <p>lock the worksheets such that the password MYPASS is required to unlock it.</p> <pre>SETWKSATTRIBUTE("WKSMENU","myfile.txt")</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

causes the contents of myfile.txt to be displayed when the / button is pressed.

**REMARKS:** Note that the READONLY attribute is not related to the read-only property at the Windows system level and should be used only when this system-level read-only manipulation is unable to fulfill the application's requirements.

**SEE ALSO:** GETWKSATTRIBUTE

## **SETWLIKE(window, level)**

**PURPOSE:** Copies the attributes of one window to another.

**window** A window reference.

**level** Integer indicating level of similarity; 0 means copy only static attributes (like color), 1 means copy computed attributes (like tic interval) as well.

## **SETWMARGIN(window, margin, size)**

**PURPOSE:** Sets the plotting margin for a given window of a worksheet.

**window** (Optional). A window reference. Defaults to the current window.

**margin** (Optional). An integer. Defaults to 1. Margin can be one of the following:

- 1 - Left
- 2 - Right
- 3 - Top
- 4 - Bottom.

**size** A scalar. Size is specified in terms of number of characters (actual size per character will vary depending on the system). If size is negative, then the margin is sized automatically.

## **SETWMARGINALL(margin, size)**

**PURPOSE:** Sets the plotting margin for all windows of a worksheet.

**margin** (Optional). An integer. Defaults to 1. Margin can be one of the following:

- 1 - Left
- 2 - Right
- 3 - Top
- 4 - Bottom

**size** A scalar. Size is specified in terms of number of characters (actual size per character will vary depending on the system). If size is negative, then the margin is sized automatically.

## **SETWSIZE(window,x,y,width,height,drawmode)**

|                  |                                                                                                                                                                                                                                         |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the size of a window.                                                                                                                                                                                                              |
| <b>window</b>    | (Optional). Window reference. Defaults to the current window.                                                                                                                                                                           |
| <b>x</b>         | The window's left location.                                                                                                                                                                                                             |
| <b>y</b>         | The window's top location.                                                                                                                                                                                                              |
| <b>width</b>     | The window's width.                                                                                                                                                                                                                     |
| <b>height</b>    | The window's height.                                                                                                                                                                                                                    |
| <b>drawmode</b>  | (Optional). 0 = Redraw worksheet; 1 = Do not redraw; 2 = Clear display of all windows. The default is 0.                                                                                                                                |
| <b>EXAMPLE:</b>  | <p>SETWSIZE(W1,0,0,.5,.5)<br/>resizes W1 to be 1/4 the size of the worksheet.</p> <p>SETWSIZE(W1...,-1,-1,-1,-1)<br/>resets all the windows to autosize mode. DISPLAYALL and adding windows also reset the windows to be autosized.</p> |
| <b>REMARKS:</b>  | The size is specified in terms of a coordinate system where 0.0, 0.0 is the upper left corner of the worksheet and 1.0,1.0 is the lower right corner.                                                                                   |
| <b>SEE ALSO:</b> | COLLAYOUT<br>LAYOUT<br>ROWLAYOUT                                                                                                                                                                                                        |

## **SETX(low, high)**

|                  |                                                                                                                         |
|------------------|-------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the range of the x axis to be shown.                                                                               |
| <b>low</b>       | A real number.                                                                                                          |
| <b>high</b>      | A real number.                                                                                                          |
| <b>RETURNS:</b>  | Nothing.                                                                                                                |
| <b>EXAMPLE:</b>  | <p>SETX( (LENGTH-100) * DELTAX, LENGTH * DELTAX )<br/>causes current window x-axis to show latest 100 observations.</p> |
| <b>SEE ALSO:</b> | SETY<br>SETXY<br>SETTICK                                                                                                |

## **SETXLABEL, SETYLABEL(window, label)**

**PURPOSE:** Sets the x-axis or y-axis label independently of units.

**window** (Optional). Window reference. Defaults to the current window.

**label** A string, in quotes, of the axis label.

**EXAMPLE:** SETXLABEL("US Treasury Bonds")  
sets the x-axis label to display: US Treasury Bonds.

**REMARKS:** There are two types of labels that can be applied to an axis: axis labels and unit labels. Unit labels are assigned automatically and have a maximum length of 15 characters. Axis labels, as set with SETXLABEL and SETYLABEL do not have this restriction. In addition, because they are independent of units, using SETXLABEL and SETYLABEL to label an axis allows for the automatic translation of units through numeric calculations. Do not force MarketBrowser to use the same string for a units label and an axis label.

Axis labels, if set, will "overwrite" unit labels in both screen and printed output. Axis labels apply to the current focus, so to apply axis labels to an overlay, use the FOCUS command before calling any axis label functions.

**SEE ALSO:** SETHUNITS  
SETVUNITS  
GETXLABEL, GETYLABEL  
CLEARXLABEL, CLEARYLABEL

## **SETXLOG(OnOff)**

**PURPOSE:** Toggles the log scales for the x axes of the current window.

**OnOff** Integer. 0 = Off: 1 = On. The default is 0.

**RETURNS:** Nothing.

**REMARKS:** All data is plotted on a log basis until log scales are turned off via SETXLOG(0).

**SEE ALSO:** SETYLOG

## SETXOFFSET(window, xoffset)

**PURPOSE:** Specifies the X offset.

**window** (Optional). A window reference. Defaults to the current window.

**xoffset** Starting x value of the series.

**RETURNS:** A series or table.

**EXAMPLES:** SETXOFFSET(W1, 10.0)

sets the offset of the series in W1 to 10.0. If you do not specify a window, MarketBrowser uses the current window. You **MUST** specify the offset as a floating point number with a decimal point (i.e. 5.0, not 5).

SETXOFFSET(W1, XOFFSET(W2))

The arithmetic functions (+ - / \*) set the offset of the resulting series to the offset of the first series in your expression. For example, offset of W1 + W2 is set to the offset of W1.

**SEE ALSO:** SETDELTAX

## SETXY(xleft, xright, ybottom, ytop)

**PURPOSE:** Specifies the overall coordinate range of a window.

**xleft** Left-hand window boundary.

**xright** Right-hand window boundary.

**ybottom** Bottom window boundary.

**ytop** Top window boundary.

**RETURNS:** Nothing.

**EXAMPLES:** SETXY(0.0, 100.0, -1.0, 1.0)

sets the x-axis from 0.0 to 100.0 and the y-axis boundaries from -1.0 to 1.0.

If window 1 contains a 1024-point series, then

SETXY(0.0, 100\*DELTAX, 0.0, MAX(W1))

would display the first 100 points of the current series falling between the values of 0.0 and the maximum y value of the entire series.

**REMARKS:** SETXY will expand or compress the current units scale. To refresh the window and redraw appropriate scales, toggle the scales.

**SEE ALSO:** SCALES SETX  
SETY SPANXY

## **SETY(low, high)**

**PURPOSE:** Sets the range of the y axis to be shown.

**low** A real number.

**high** A real number.

**RETURNS:** Nothing.

**EXAMPLE:** SETY(0.0, MAX(W1))

causes the current window's y-axis to run from zero to the highest value in window 1.

**SEE ALSO:** SETXSETXY

## **SETYLOG(OnOff)**

**PURPOSE:** Toggles the log scales for the y axes of the current window.

**OnOff** Integer. 0 = Off. 1 = On. The defaults is 0.

**RETURNS:** Nothing.

**REMARKS:** All data is plotted on a log basis until the log scales are turned off via SETYLOG(0).

**SEE ALSO:** SETXLOG

## **SFORMAT(control, value1, ..., valuen)**

**PURPOSE:** Formats a list of strings.

**control** Control string conforming to C language printf specifications, containing only string field specifiers.

**value1, ..., valuen** List of strings.

**RETURNS:** Nothing.

**EXAMPLE:** SFORMAT("Comment: %s", GETCOMMENT)

produces a string like "Comment IBM"

**REMARKS:** See any standard C language reference for further information.

**SEE ALSO:** ANYFORMAT  
NFORMAT

## SHADEWITH(matrix)

**PURPOSE:** Adds a fourth "shading" dimension to various three dimensional plots of matrix data.

**matrix** A conforming matrix of data.

**RETURNS:** Nothing.

**EXAMPLE:** SHADEWITH(W2)

If the current window contains a rectangular matrix of data shown with a meshed waterfall plot, and W2 contains the column-wise derivative of that data, the surface colors of the waterfall will be mapped to represent the rate of change of the data, using the shading scheme currently in effect.

**REMARKS:** This function has no effect on monochrome hardware.

**SEE ALSO:** PLOT3D  
SETPALETTE  
SETSHADING  
WATERFALL  
CONTOUR  
DENSITY

## SINC(expr)

**PURPOSE:** Calculates the sinc function ( $\text{SIN}(X)/X$ ) of the specified expression.

**expr** Any expression evaluating to a series, table, integer, real or complex number.

**RETURNS:** A series or table.

**EXAMPLE:** SINC(W1)

creates a new series from the contents of window 1 and places the result in the current window. The value of each point in the new series will be the SIN of the corresponding point in window 1 divided by itself.

**REMARKS:** SINC(X) is equivalent to  $\text{SIN}(X)/X$  in SETRADIAN mode and to  $\text{SIN}(x/(2*\text{PI}))/X$  in SETDEGREE MODE.

## **SORT(series, order)**

|                  |                                                                             |
|------------------|-----------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Rearranges the y values of a series in numerical order.                     |
| <b>series</b>    | A series or table to sort.                                                  |
| <b>order</b>     | (Optional.) 0 = descending; 1 = ascending. The default is 0.                |
| <b>RETURNS:</b>  | A series or table.                                                          |
| <b>EXAMPLE:</b>  | <pre>SORT(W1)</pre> <p>sorts W1 from the largest to the smallest value.</p> |
| <b>SEE ALSO:</b> | GRADE<br>REORDER<br>LOOKUP                                                  |

## **SPANX, SPANY(window, min, max)**

|                  |                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Restricts the scale display to a sub-range of the plotting window.                                                                                                                                                                                                                                                                                                                   |
| <b>window</b>    | (Optional). Window reference. Defaults to the current window.                                                                                                                                                                                                                                                                                                                        |
| <b>min</b>       | Real number. Minimum x or y value for displayed range.                                                                                                                                                                                                                                                                                                                               |
| <b>max</b>       | Real number. Maximum x or y value for displayed range.                                                                                                                                                                                                                                                                                                                               |
| <b>EXAMPLE:</b>  | <pre>W1: gsin(100,.01);setvunits("Volts") W2: gcos(100,.01);setvunits("Amps") W3: w1;staggery(0);staggerx(0);Scales(2);sety(-4,1.5);spany(-1,1);overlay(w2,red); focus(2);staggery(0);staggerx(0);scales(13);sety(-1.5,4);spany(-1,1)</pre> <p>Window 3 contains the 2 curves overlaid with flush scales with a defined y-axis span which is a portion of the entire plot range.</p> |
| <b>REMARKS:</b>  | SPANX and SPANY require scales to be flush with the plotting area, that is, they must be used in conjunction with STAGGERX(0) and STAGGERY(0), respectively. SPANX and SPANY apply to the scales associated with the current focus of the specified window. Use SETXY to set the range (full span) covered by the plotting area                                                      |
| <b>SEE ALSO:</b> | STAGGERX, STAGGERY<br>SETXY<br>OVERLAY<br>SCALES<br>FOCUS                                                                                                                                                                                                                                                                                                                            |

## SPECTRUM(series)

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>         | (A macro). Returns the magnitude of the first half of the FFT of a series normalized by the length of the series.                                                                                                                                                                                                                                                                                                                                                            |
| <b>series</b>           | A series or table.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>RETURNS:</b>         | A real series or table.                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>EXPAN-<br/>SION:</b> | $\text{MAG}(\text{EXTRACT}(\text{FFT}(\text{S}),1,\text{INT}(\text{SERSIZE}(\text{S})/2))) * 2/\text{SERSIZE}(\text{S})$                                                                                                                                                                                                                                                                                                                                                     |
| <b>EXAMPLE:</b>         | W1: GSIN(128, 1/128, 1.0)<br>W2: GSIN(128, 1/128, 4.0)<br>W3: SPECTRUM(W1 + W2)<br><br>returns a real series in W3 with a peak at 1.0 Hz and a peak at 4.0 Hz. Compare this result to FFT(W1 + W2).                                                                                                                                                                                                                                                                          |
| <b>REMARKS:</b>         | The FFT of a real series (i.e. a series with no imaginary components) results in a complex series where the second half of the result is the mirror image of the first half. Also, the relative amplitudes of the FFT depend on the length of the original series.<br><br>The SPECTRUM macro returns the magnitude of the first half of the FFT and factors out the length of the series. This macro is very useful for comparing the frequency spectra of different series. |
| <b>SEE ALSO:</b>        | DFT<br>FFTP<br>GHAMMING<br>GKAISER<br><br>FFT<br>PSD<br>GHANNING                                                                                                                                                                                                                                                                                                                                                                                                             |

## **SPLINE(s, n)**

**PURPOSE:** Performs cubic spline interpolation.

**s** Series to interpolate.

**n** Integer specifying interpolation factor.

**RETURNS:** Interpolated time domain series.

**EXAMPLES:** W1: GSER(6, 5, 8, 11, 6)

W2: SPLINE(W1, 10)

W3: W2; OVERPLOT(W1, RED); SETSYMBOL(SQUARE, 2)

Compare the spline fit with the original data (red curve with square symbols). The spline interpolation is smoother.

SPLINE is also useful for XY plots. For example:

W1: GSER(1, 5, 8, 10, 15)

W2: GSER(2, 6, 4, 8, 10)

W3: XY(W1, W2)

W4: XY(XYINTERP(W1, 10), SPLINE(W2, 10))

**REMARKS:** SPLINE effectively fits a third order polynomial between adjacent samples and uses the polynomial equation to calculate the interpolated values.

**SEE ALSO:** INTERPOLATE

## SPRINTF(control, arg1, ..., argn)

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Produces an output string in the format of the C language printf function.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>control</b>   | <p>Format control string. Conforms to C language printf specifications. Control strings may contain ordinary characters, escape sequences, and format specifications. Ordinary characters are copied to the output string in order of their appearance. Escape sequences are introduced by a \ (backslash) character. Format specifications in the control statement are introduced by a % (percent sign) character, and are matched to the specified arguments in order. If there are more arguments than there are format specifications, the extra arguments are ignored.</p> <p>Format Specification Fields:</p> <p><i>% [flags], [width], [precision], type</i></p>                                                                                                                                                                                                                                                           |
| <b>flags</b>     | <p>(Optional.) Options are:</p> <ul style="list-style-type: none"><li>• -: Left Justify.</li><li>• +: Explicit sign (+ or -) before number.</li><li>• 0: If width is prefixed with 0, zeroes are added until the minimum width is reached. If both 0 and - appear, the 0 is ignored. If 0 is specified with an integer format (i, u, x, X, o, d), the 0 is ignored.</li><li>• blank: Insert blank before positive number.</li><li>• #: When used with the o, x, or X type format, the # prefixes any non-zero output value with 0, 0x, or 0X. When used with the e, E, or f type format, the # forces the output value to contain a decimal point in all cases. Used with the g or G format, the # forces the output value to contain a decimal point in all cases and prevents the truncation of trailing zeroes.</li></ul>                                                                                                       |
| <b>width</b>     | (Optional). Integer that specifies the minimum number of character output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>precision</b> | <p>(Optional). Integer which specifies the maximum number of characters printed for all or part of the output field, or the minimum number of digits printed for integer values.</p> <p>For types: <b>d, I, u, o, x, X</b> - specifies the minimum number of digits to print. If the number of digits in the argument is less than the precision, the output value is padded on the left with zeroes. The value is not truncated when the number of digits exceeds the precision.</p> <p>For types: <b>e, E</b> - specifies the number of digits to be printed after the decimal point. The last digit is rounded.</p> <p>For type: <b>f</b> - specifies the number of digits to be printed after the decimal point. If a decimal point appears, at least one digit appears before it. The value is rounded to the appropriate number of digits.</p> <p>For type: <b>s</b> - specifies the number of characters to be printed.</p> |
| <b>type</b>      | <p>Required character. Determines whether the associated argument is interpreted as a character, string, or number. Options are:</p> <p><b>d, i</b> - Signed decimal integer</p> <p><b>u, o</b> - Unsigned decimal integer</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**x, X** - Unsigned hex integer, using 'abcdef' or 'ABCDEF'.

**f** - Signed value of the form -dddd.dddd, where dddd is one or more decimal digits

**e** - Signed value of the form [-]d.ddd e [sign]ddd, where d is a single digit, dddd is one or more decimal digits, ddd is exactly three decimal digits, and sign is + or -.

**E** - Identical to 'e' format, except that E introduces the exponent.

**g** - Signed value printed in f or e format, whichever is more compact for the given value and precision. The e format is used when the exponent of the value is less than 4, or greater than or equal to the precision argument. Trailing zeroes are truncated, and the decimal point appears only if one or more digits follow it.

**G** - Identical to g format, except that G introduces the exponent.

**c** - Single character

**s** - String. Characters printed up to the first null character or until the precision value is reached.

**arg1, ..., argn**      Scalar or string value that matches the control string.

**RETURNS:**      A string.

**EXAMPLES:**      `PRINTF("Today is %s, at %s. The temperature is %3.2f degrees.", getdate, gettime, 75.636)`

returns a string like:

Today is 04-40-1995, at 14:23:45.32. The temperature is 75.64 degrees.

`PRINTF("Mean:%8.2f Stdev:%8.2f Max%8.2f",mean, stdev, max)`

returns a string like:

Mean: 0.52 Stdev: 0.28 Max: 0.98

`PRINTF("Hex Value: %x ", 10)`

returns the string:

Hex Value: a

**REMARKS:**      For more detailed information about C language formatting specifications, see a C language reference manual.

**SEE ALSO:**      ECHO                              FPUTS  
                     STRCAT                              TEXTANN

## **SQRT(expr)**

**PURPOSE:** Calculates the square root of a specified expression.

**expr** Any expression evaluating to a series, table, integer, real or complex number.

**RETURNS:** A series, table or number.

**EXAMPLE:** SQRT(W1)

This creates a new series from the contents of window 1 and places the result in the current window. The value of each point in the new series will be the square root of the corresponding point in window 1.

**SEE ALSO:** RMS

## **STACK**

**PURPOSE:** Sets the window's display style to stacked bars.

**RETURNS:** Nothing.

**REMARKS:** Stacked bars are meaningful only for small collections of data. Series of more than a few data points will become visually indistinct.

**SEE ALSO:** PCTSTACK

## STAGGERX, STAGGERY(window, flag)

- PURPOSE:** Staggers the x- or y-axis scale display, and locates the scale in a region which is farther away from the plotting area than the regions taken by any preceding scales.
- window** (Optional). Window reference. Defaults to the current window.
- flag** An integer value; 1 = ON, 0 = OFF. Defaults to 1
- RETURNS:** Nothing.
- EXAMPLE:**  
W1: gsin(100,.01);setvunits("Volts")  
W2: gcos(100,.01);setvunits("Amps")  
W3: W1;SCALES(2); OVERLAY(w2,red); FOCUS(2); SCALES(2);  
STAGGERY(1); LABEL("Staggered Scales")  
Window 3 contains the 2 curves overlaid with staggered scales.
- REMARKS:** Staggered scales are only applicable in OVERLAY plots, and apply to the scales associated with the window's current focus. Overlays created earlier have precedence. If flag is OFF, then MarketBrowser will turn off staggering, and the current focus scale will stay flush with the plotting area.
- SEE ALSO:** SPANX, SPANY OVERLAY  
SCALES

## STATS(series, first, points)

- PURPOSE:** Finds the arithmetic mean and the standard deviation of any series or table.
- series** (Optional). A series or table. Defaults to the current window.
- first** (Optional). The first series point to include in the calculation of statistics. The default is the first point.
- points** (Optional). The number of points to include in the calculation starting from the provided *first* point. The default is to the end of the series.
- RETURNS:** Two numbers representing the mean and the standard deviation.
- REMARKS:** Cannot be used in MarketBrowser expressions. Strictly a calculator function. In expressions, use MEAN and STDEV.
- SEE ALSO:** STDEV  
MEAN

## STDEV(series, first, points)

|                  |                                                                                                                                                        |        |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| <b>PURPOSE:</b>  | Calculates the standard deviation of any series or table.                                                                                              |        |
| <b>series</b>    | (Optional). A series or table. The default is the current window.                                                                                      |        |
| <b>first</b>     | (Optional). The first point to include in the calculation of standard deviation. The default is the first point.                                       |        |
| <b>points</b>    | (Optional). The number of points to include in the calculation starting from the provided <i>first</i> point. The default is to the end of the series. |        |
| <b>RETURNS:</b>  | A number.                                                                                                                                              |        |
| <b>REMARKS:</b>  | STATS provides both STDEV and MEAN.                                                                                                                    |        |
| <b>SEE ALSO:</b> | STATS                                                                                                                                                  | MEAN   |
|                  | COLSTDEV                                                                                                                                               | MEDIAN |

## STEPS

|                  |                                                   |        |
|------------------|---------------------------------------------------|--------|
| <b>PURPOSE:</b>  | Displays a line graph as a step plot.             |        |
| <b>RETURNS:</b>  | A step plot for the series in the current window. |        |
| <b>SEE ALSO:</b> | BAR                                               | POINTS |
|                  | STICKS                                            |        |

## STICKS

|                  |                                                          |  |
|------------------|----------------------------------------------------------|--|
| <b>PURPOSE:</b>  | Displays the data points of a series as vertical sticks. |  |
| <b>RETURNS:</b>  | Nothing.                                                 |  |
| <b>REMARKS:</b>  | STICKS has the same function as toggling the F7 key.     |  |
| <b>SEE ALSO:</b> | BAR                                                      |  |
|                  | LINES                                                    |  |
|                  | POINTS                                                   |  |
|                  | TABLEVIEW                                                |  |
|                  | TICKFORM                                                 |  |
|                  | PCTSTACK                                                 |  |
|                  | STEPS                                                    |  |

## **STOPREFRESH(action)**

- PURPOSE:** Interrupts a refresh or inquires whether the refresh is interrupted.
- action** Integer. -1 - returns 1 if the refresh is being interrupted, 0 otherwise; 0 - does nothing; 1 - interrupts a refresh in progress (no effect if not in progress).
- RETURNS:** An integer if -1 is used for the action argument.
- SEE ALSO:** REFRESH

## **STRCAT(string1, ..., stringn)**

- PURPOSE:** Concatenates two or more strings.
- string1, ..., stringn** List of strings to concatenate.
- RETURNS:** A string.
- EXAMPLE:** STRCAT("The mean value is ", STRNUM(MEAN(W1)))  
displays:  
The mean value is 2.0.
- SEE ALSO:** STRNUM

## **STRCHAR(integer)**

- PURPOSE:** Returns the character represented by a given ASCII integer.
- integer** An integer.
- RETURNS:** A string.
- SEE ALSO:** CHARSTR  
STRESCAPE  
STRCAT  
STRNUM  
NUMSTR

## **STRCMP(string1, string2, caseflag)**

- PURPOSE:** Compares two strings in lexicographical order.
- string1** First input string, in quotes.
- string2** Second input string, in quotes.
- caseflag** (Optional). Case sensitivity flag. Enter 1 to specify case sensitivity or 0 to ignore case. Defaults to 0.
- RETURNS:** A negative number if string1 is less than string2, 0 if they are the same, and a positive number if string 1 is greater than string 2.
- REMARKS:** Non-alphabetic characters are assigned lexicographical positions based on their ASCII codes.
- When a non-zero number is returned, the exact number that is returned is the difference in ASCII codes between the characters in the first position which is different.
- Leading and trailing blanks are significant. By default, case is not significant. If the optional case sensitivity flag is non-zero, STRCMP performs a case-sensitive comparison.

## **STRCOLOR(color\_num)**

- PURPOSE:** Returns the name of the color corresponding to the input argument.
- color\_num** Integer. The color number.
- RETURNS:** A string.
- REMARKS:** Used to retrieve the macro name of a color number as known to the system.

## **STRDATE(series, pointnum)**

- PURPOSE:** Returns the string form of an index into a window.
- series** A series or table.
- pointnum** An index number.
- RETURNS:** A date string in mm/dd/yy format.
- EXAMPLE:** STRDATE( W4, 72 )  
returns "4/14/87" if window 4 contained daily data starting on 1/1/87.
- SEE ALSO:** DATESTR

## **STRESCAPE(string)**

**PURPOSE:** Converts special "escape" characters in a string.

**string** A text string in quotes.

**RETURNS:** A string.

**EXAMPLE:** Type:  
=FOPEN("TEST.TXT")  
=FPUTS(STRCAT("This displays",STRESCAPE("\n"),"two lines.))  
=FCLOSE("TEST.TXT")  
=VIEWFILE("TEST.TXT")  
The "\n" is evaluated as a carriage return when the string is written to a file.

**SEE ALSO:** STRCAT

## **STREXTRACT(string, start, length)**

**PURPOSE:** Extracts parts of a string.

**string** String to extract from, in quotes.

**start** Integer. The starting point.

**length** Integer. The number of characters to extract.

**RETURNS:** A string.

**EXAMPLE:** STREXTRACT("ONE and a two and a three", 11, 3)  
returns two

**REMARKS:** Blanks are considered a character.

**SEE ALSO:** STRFIND  
STRGET

## **STRFILE(filename, reverse, no\_interpret, no\_blanks)**

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>     | Reads and converts a plain text file into a string with embedded newlines.                                                                                                                                                                                                                                                                                                                                                      |
| <b>filename</b>     | Name of the file with path that you wish to convert into a string, enclosed in quotes.                                                                                                                                                                                                                                                                                                                                          |
| <b>reverse</b>      | (Optional). 0 or 1, where 0 is Off and 1 is On. Use this parameter to reverse a file so that it reads from end to beginning rather than from beginning to end. Defaults to 0, or off.                                                                                                                                                                                                                                           |
| <b>no_interpret</b> | (Optional). 0 or 1, where 0 is Off and 1 is On. Specifies whether or not expressions in curly brackets should be interpreted by MarketBrowser. Defaults to 0, or off.                                                                                                                                                                                                                                                           |
| <b>no_blanks</b>    | (Optional). 0 or 1, where 1 ignores any blank lines in the file being read. Defaults to 0.                                                                                                                                                                                                                                                                                                                                      |
| <b>RETURNS:</b>     | A string.                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>EXAMPLE:</b>     | <pre>STRFILE("symbols.txt",1,1)</pre> returns the contents of the file "symbol.txt" as a single string with embedded newlines, with the lines of the file in reverse order. Expressions in curly brackets are not interpreted.                                                                                                                                                                                                  |
| <b>REMARKS:</b>     | <p>There is a buffer limit of 4095 characters per line. If any one line within the file being read exceeds the 4095 character limit then the function will not run properly and will cause an evaluation error.</p> <p>This function is useful with primitive annotation functions such as TEXTANN for placing the contents of text files in windows. For example:</p> <pre>GSER(0,1,2);TEXTANN(1,1,STRFILE("ascii.txt"))</pre> |
| <b>SEE ALSO:</b>    | TLABEL                                                                                                                                                                                                                                                                                                                                                                                                                          |

## **STRFIND(string1, string2)**

|                  |                                                                                                       |
|------------------|-------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Returns the string that starts at the first occurrence of string1 in string2.                         |
| <b>string1</b>   | String to search for, in quotes.                                                                      |
| <b>string2</b>   | String to search in, in quotes.                                                                       |
| <b>RETURNS:</b>  | A string.                                                                                             |
| <b>EXAMPLE:</b>  | <pre>STRFIND("XINC", "YOR:12.3 XINC:1.0 YREF:120.0")</pre> returns the string:<br>XINC:1.0 YREF:120.0 |
| <b>SEE ALSO:</b> | STREXTRACT<br>STRGET<br>STRREVERSE                                                                    |

## **STRFLDSORT(string, which\_field, direction, how\_sort)**

|                    |                                                                                                                                                                                                              |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>    | Sorts a list of strings.                                                                                                                                                                                     |
| <b>string</b>      | A newline-delimited list, which may have several fields, separated by tabs.                                                                                                                                  |
| <b>which_field</b> | (Optional). The field to use as the key for sorting (origin = 1). Use -1 to indicate the default.                                                                                                            |
| <b>direction</b>   | (Optional). Direction in which to sort. 1 - forwards, 0 - backwards. The default is 1. Use -1 to indicate the default.                                                                                       |
| <b>how_sort</b>    | (Optional). Indicates what sorting method to use. 0 or less means sort in alphabetical order, 1 means sort by date order, 2 means sort by numerical order. The default is 0. Use -1 to indicate the default. |
| <b>RETURNS:</b>    | A newline-delimited string, which is a sorted copy of the input string.                                                                                                                                      |
| <b>SEE ALSO:</b>   | STREXTRACT<br>STRGET<br>STRREVERSE                                                                                                                                                                           |

## **STRGET(num, string, delimiter)**

|                  |                                                                                                                                                                                                                                                                                                          |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Returns the nth sub-string of a string.                                                                                                                                                                                                                                                                  |
| <b>num</b>       | Integer. Specifies the number of the sub-string to return.                                                                                                                                                                                                                                               |
| <b>string</b>    | String to search in.                                                                                                                                                                                                                                                                                     |
| <b>delimiter</b> | (Optional). String specifying the delimiter characters. The default delimiter character is a space. Defaults to the end of the string.                                                                                                                                                                   |
| <b>RETURNS:</b>  | A string. If the delimiter is not found, the entire original string is returned.                                                                                                                                                                                                                         |
| <b>EXAMPLES:</b> | STRGET(2, "YOR:12.3 XINC:1.0 YREF:120.0")<br>returns XINC:1.0<br><br>By default sub-string is delimited by spaces. You may optionally specify your own delimiter characters.<br><br>STRGET(2, "YOR:12.3 XINC:1.0 YREF:120.0",":")<br>returns 12.3 XINC because the : is the now the separator character. |
| <b>SEE ALSO:</b> | STREXTRACT<br>STRFIND<br>STRREVERSE                                                                                                                                                                                                                                                                      |

## **STRJUL(julian)**

**PURPOSE:** Converts an integer Julian date to a string.

**julian** Integer. A Julian date.

**RETURNS:** A string.

**EXAMPLE:** STRJUL(12345)  
creates the date string "10/18/73"

**SEE ALSO:** JULSTR  
JULDAY  
ADDBDAY

## **STRLEN(string)**

**PURPOSE:** Returns the length of a string.

**string** A string, in quotes.

**RETURNS:** An integer.

**SEE ALSO:** STREVERSE

## **STRLIST(str1, ..., strn)**

**PURPOSE:** Converts a list of several strings into one long string with embedded newline characters.

**str1, ..., strn** String enclosed in quotes.

**RETURNS:** One string.

**EXAMPLE:** STRLIST("One", "Two")  
returns the string "One Two" (with "One" and "Two" on different lines separated by a newline character).

## **STRNUM(num)**

**PURPOSE:** Converts a number into a string.

**num** Number to convert into a string.

**RETURNS:** A string.

**EXAMPLES:** STRCAT("The number ", STRNUM(11), "is prime")

produces:

The number 11 is prime.

STRCAT("The mean value is ", STRNUM(MEAN(W1)))

produces:

The mean value is 2.0 (if the mean of W1 is 2.0).

**SEE ALSO:** NUMSTR  
STRCAT

## **STRREVERSE(string)**

**PURPOSE:** Reverses the order of characters in a string.

**string** A string of characters, in quotes.

**RETURNS:** A string.

**EXAMPLE:** STRREVERSE("abc")

returns "cba".

**SEE ALSO:** STRFIND  
STRGET  
STRLEN

## **STRTOD(series, pointnumber)**

**PURPOSE:** Returns the time from an index into a window.

**series** A series or table.

**pointnumber** An index number.

**RETURNS:** A time string in hh:mm:ss format.

**SEE ALSO:** TODSTR

## **STRWIN(window)**

**PURPOSE:** Converts a window reference into a string.

**window** Window reference.

**RETURNS:** A string.

## **SUMS(series1, ..., seriesn)**

**PURPOSE:** Creates a series that is the arithmetic sum of the input series.

**series1, ..., seriesn** A series or table.

**RETURNS:** A series or table.

**EXAMPLES:** SUMS(W1, W2, W6, W9)

creates a new series by adding the series in window 1, window 2, window 6, and window 9. The result is placed in the current window.

SUMS(W3..W8)

sums windows 3 through 8 and places the result in the current window.

**REMARKS:** Shorter series are padded with 0.0 to the length of the longest series.

**SEE ALSO:** AVGS

## SVD(matrix, otype)

**PURPOSE:** Calculates the singular value decomposition of a matrix.

**matrix** A square matrix.

**otype** Type of matrix to output, where options are:

- 00 - W: Singular values (default).
- 01 - V: Right singular value matrix.
- 10 - U: Left singular value matrix.
- 11 - UVW: Combined UVW matrix.

**RETURNS:** A matrix.

**EXAMPLE:** W1:

|   |   |   |
|---|---|---|
| 1 | 4 | 7 |
| 2 | 5 | 8 |
| 3 | 6 | 9 |

W2: SVD(W1,01)

|       |       |       |
|-------|-------|-------|
| -0.21 | -0.89 | 0.41  |
| -0.52 | -0.25 | -0.82 |
| -0.83 | 0.39  | 0.41  |

W3: SVD(W1,10)

|       |       |       |
|-------|-------|-------|
| -0.48 | 0.78  | 0.41  |
| -0.57 | 0.08  | -0.82 |
| -0.67 | -0.63 | 0.41  |

W4: SVD(W1, 00)

|       |
|-------|
| 16.85 |
| 1.07  |
| 0.00  |

W5: MMULT(MMULT(W3,DIAGONAL(W4)), TRANSPOSE(W2))

|   |   |   |
|---|---|---|
| 1 | 4 | 7 |
| 2 | 5 | 8 |
| 3 | 6 | 9 |

**REMARKS:** The input matrix is decomposed into a left singular value matrix U, a diagonal matrix W, and a right singular matrix V such that:

$$M = U * W * \text{TRANSPOSE}(V)$$

For additional functionality, please refer to matrix.mac in the macros sub-directory.

**SEE ALSO:** DIAGONAL                      HESS  
MMULT                                  TRANSPOSE  
LU

## SYNC(series, SyncMode)

**PURPOSE:** Sets the synchronization of an overlaid window.

**series** (Optional). A series or table. Defaults to the current window.

**SyncMode** Integer from the following table:

| Integer | Expand | Expand & Scroll |
|---------|--------|-----------------|
| 0       |        |                 |
| 1       |        | X               |
| 2       |        | Y               |
| 3       |        | X & Y           |
| 4       | X      |                 |
| 5       | Y      |                 |
| 6       | X & Y  |                 |

**RETURNS:** Nothing.

**EXAMPLE:** SYNC(4)

For a window with two or more OVERLAYS, keeps all horizontal scrolling synchronized, while allowing independent vertical scrolling. In this case, all stretching/shrinking remains independent.

**SEE ALSO:** FOCUS  
OVERLAY  
SCALES

## TABLE(series)

**PURPOSE:** Lists the x and y values of one series. This allows you to scroll through point values.

**series** (Optional). A series or table. Defaults to the current window.

**RETURNS:** A table.

**EXAMPLE:** TABLE(W4)  
displays a table of point values for the series in window 4.

**SEE ALSO:** TABLES  
EDIT



|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>fg_clr</b>      | (Optional). An integer specifying the color of the series in the window. Defaults to the color of the primary series.                                                                                                                                                                                                                                                                                                                       |
| <b>bg_clr</b>      | (Optional). An integer specifying the background color of the annotated text. Defaults to the window's background color.                                                                                                                                                                                                                                                                                                                    |
| <b>font</b>        | (Optional). An integer specifying the font size. Defaults to 0. <ul style="list-style-type: none"> <li>• 0 - NORM_FONT</li> <li>• 1 - SMALL_FONT</li> <li>• 2 - STATLINE_FONT</li> <li>• 3 - POPBOX_FONT</li> <li>• 4 - WINLABEL_FONT</li> <li>• 5 - TOOLBAR_FONT</li> <li>• 6 - LISTBOX_FONT</li> <li>• 7 - MENU_FONT</li> <li>• 8 - USER1_FONT</li> <li>• 9 - USER2_FONT</li> <li>• 10 - USER3_FONT</li> <li>• 11 - PANEL_FONT</li> </ul> |
| <b>box_flg</b>     | (Optional). An integer specifying presence or absence of solid line box surrounding the text (with margin if legend_flg is ON, otherwise, no margin). 1 = ON; 0 = OFF. Defaults to 1.                                                                                                                                                                                                                                                       |
| <b>legend_flg</b>  | (Optional). An integer specifying whether legend symbols are present or absent. 1 = ON; 0 = OFF. Defaults to 0.                                                                                                                                                                                                                                                                                                                             |
| <b>stretch_flg</b> | (Optional). An integer specifying whether to stretch the annotation to fit the rectangular region where the text block resides. 1 = ON; 0 = OFF. Defaults to 0.                                                                                                                                                                                                                                                                             |
| <b>margin_flg</b>  | (Optional). An integer specifying margin to be adjusted. Defaults to -1. <ul style="list-style-type: none"> <li>• -1 = No Adjustment</li> <li>• 0 = Top Margin</li> <li>• 1 = Right Margin</li> <li>• 2 = Bottom Margin</li> <li>• 3 = Left Margin</li> </ul>                                                                                                                                                                               |
| <b>focus</b>       | (Optional). An integer specifying focus for PAPER annotations. Defaults to 1.                                                                                                                                                                                                                                                                                                                                                               |
| <b>s1, ..., sn</b> | The text that will be printed at coordinates x and y, in quotes. At least one string is required; additional strings are optional.                                                                                                                                                                                                                                                                                                          |

**RETURNS:** Nothing.

**EXAMPLES:** GRAND(100,1)\*10;TEXTANN(6.0, 3.0, 0, -1, 5, 1, 1, 0, "Temp over Time")

In this example, TEXTANN prints the text, "Temp over Time" at axis coordinates (6.0, 3.0) of the window. The 0 as the target indicates that the text will scroll with the data (PAPER). The -1, 5 instruct MarketBrowser to use the color of the primary series and color 5 for the background. The text will be drawn using the small font (1), and it will be surrounded by a box.

```
W1:GSIN(100,.01); W2:W1;OVERLAY(GCOS(100,.01),RED)
ADDWFORM("TEXTANN(.1,.8,2,-1,-1,1,1,0, 3,'Sine','Cosine')");PON
```

puts a legend in the left window margin.

```
ADDWFORM("TEXTANN(.1,.9,1,-1,-1,1,1,0,
3,strcat('max:',strnum(MAX)))");PON
```

places the concatenated string "max:" followed by the window's current maximum value in the main plotting area.

**REMARKS:**

X and Y coordinate systems differ depending on whether your target is PAPER or GLASS. All GLASS coordinates are normalized to the specified rectangular regions in the worksheet, where the upper left corner is (0.0, 0.0) and the lower right corner is (1.0, 1.0). GLASS annotations "stick" to the window like the viewfinder in a camera. Paper coordinates, on the other hand, are taken from the x and y values of the series in the window. PAPER annotations scroll with the data.

To use TEXTANN from the command line, you must enclose a call to TEXTANN() in a string passed to ADDWFORM() or ADDFORM() manually, or append it to the current window formula. This adds the command to the window formula. You must then call PON to see the effect. Because it is a plot-time function, TEXTANN() is re-evaluated on every PON redraw.

To use the default value for any integer parameter (from target to focus), use -1 as the argument to TEXTANN.

If the box\_flg is ON, then its background will be filled with the background color, and its edges will be drawn as solid lines in the foreground color.

If legend\_flg is ON, then the x, y parameters refer to the lower left corner of the first symbol in the legend block, not to the lower left corner of the first line in the text block. Each next line in the legend refers to the next overplot for color, line style, and symbols. Also, the interline spacing in legends is greater than in the no-legend-symbol case.

**SEE ALSO:**

|          |          |
|----------|----------|
| TEXTCUR  | TEXTDEL  |
| TEXTEDIT | TEXTMOVE |
| LEGEND   | ADDWFORM |
| TLABEL   | ADDFORM  |

## **TEXTCUR(target, fg\_clr, bg\_clr, font, box\_flg, legend\_flg, stretch\_flg, margin\_flg, focus)**

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>    | Brings up a free-roaming crosshair cursor in the middle of the window.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>target</b>      | (Optional). An integer specifying the relationship of the text to the window. Defaults to 0. <ul style="list-style-type: none"><li>• 0 = PAPER. Text on the “graph paper” in the window; within the coordinate system of the data.</li><li>• 1 = GLASS. Text within the plotting area of window.</li><li>• 2 = GLASS_WMARGIN. Text within the area of the entire window.</li><li>• 3 = GLASS_WPMARGIN. Text within the vertical dimensions of a window, and within the horizontal dimensions of the plotting area.</li><li>• 4 = GLASS_WSMARGIN. Text within the entire worksheet area. Bounded above by the toolbar.</li></ul> |
| <b>fg_clr</b>      | (Optional). An integer specifying the color of the series in the window. Defaults to the color of the primary series.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>bg_clr</b>      | (Optional). An integer specifying the background color of the annotated text. Defaults to window background color.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>font</b>        | (Optional). An integer specifying size of font. Defaults to 0. <ul style="list-style-type: none"><li>• 0 - NORM_FONT</li><li>• 1 - SMALL_FONT</li><li>• 2 - STATLINE_FONT</li><li>• 3 - POPBOX_FONT</li><li>• 4 - WINLABEL_FONT</li><li>• 5 - TOOLBAR_FONT</li><li>• 6 - LISTBOX_FONT</li><li>• 7 - MENU_FONT</li><li>• 8 - USER1_FONT</li><li>• 9 - USER2_FONT</li><li>• 10 - USER3_FONT</li><li>• 11 - PANEL_FONT</li></ul>                                                                                                                                                                                                   |
| <b>box_flg</b>     | (Optional). An integer specifying presence or absence of solid line box surrounding the text (with margin if legend_flg is ON, otherwise, no margin). 1 = ON; 0 = OFF. Defaults to 1.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>legend_flg</b>  | (Optional). An integer specifying whether legend symbols are present or absent. 1 = ON; 0 = OFF. Defaults to 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>stretch_flg</b> | (Optional). An integer specifying whether to stretch the annotation to fit the rectangular region where the text block resides. 1 = ON; 0 = OFF. Defaults to 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>margin_flg</b>  | (Optional). An integer specifying margin to be adjusted. Defaults to -1. <ul style="list-style-type: none"><li>• -1 = No Adjustment</li><li>• 0 = Top Margin</li><li>• 1 = Right Margin</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                               |

- 2 = Bottom Margin
- 3 = Left Margin

**focus** (Optional). An integer specifying focus for PAPER annotations. Defaults to 1.

**s1...sn** At least one string is required. Additional strings are optional. This is the text that will be printed at coordinates x and y above. Annotation lines are in top to bottom order.

**RETURNS:** Nothing.

**EXAMPLE:** TEXTCUR brings up a free-roaming crosshair cursor in the middle of the window.

**REMARKS:** To evaluate functions or macros and have their scalar or string return value(s) displayed as a text annotation, surround the function name by curly braces. For example, {max } evaluates "max" and displays the maximum value of the current series in the text annotation. Use SETPRECISION to control display of numeric values returned from MarketBrowser functions that you have embedded in text.

To erase single lines of text (while in the input mode) use [CTRL]-[X]; use TEXTDEL to erase blocks of text. TEXTCUR does not work in an empty window.

**SEE ALSO:** TEXTANN TEXTDEL  
 TEXTEDIT TEXTMOVE  
 LEGCUR TLEGEND  
 TORIGIN

## TEXTDEL

**PURPOSE:** Deletes a block of text created with TEXTCUR.

**RETURNS:** Nothing

**REMARKS:** TEXTDEL surrounds each text block in a window with four "handles", one at each corner. Position the mouse cursor over your text block and press the left mouse button. The text block then disappears. You may delete multiple blocks of text with TEXTDEL. Press the right mouse button (or ESC) when you are finished deleting your text blocks.

**SEE ALSO:** LINEDEL  
 TEXTEDIT

## TEXTEDIT

**PURPOSE:** Edits text annotation, by use of a mouse, the keyboard, and the command line buffer.

**RETURNS:** Nothing

**REMARKS:** You can edit any text block by moving the mouse cursor over any line in the block and pressing the left mouse button. After a line has been selected, it is surrounded by line handles, one at each corner and the chosen line is placed in its unevaluated form, (with curly braces around MarketBrowser expressions to be evaluated), in the command line buffer. You may edit the line in the line buffer and indicate that you are done by pressing the left mouse button, RETURN or the up or down arrow keys.

You can leave the line you are editing and move to the line above or below by pressing the up (or down) arrow keys. When you do, the line handles also move up and down and the current line appears in the command buffer.

Leaving a line with an evaluation in the text, updates the screen with the evaluated result. For example, you have recently changed the values in W2 and your text in W1 reads, "Max of W2: {max(W2)}. When you leave this line, the string gets evaluated and the screen in window 1 is updated with the new maximum value of window 2.

Arrow actions wrap around the text block, i.e., if the up arrow is applied to the first line, then the last line of the text block appears, or if the down arrow is applied to the last line, then the first line of the text block appears.

Pressing the right mouse button (or the ESC key) aborts the line that you are editing. At this point you may place the mouse cursor on a new line of text and resume inputting text. When you've completed your text block editing, pressing the right mouse button (or ESC) a second time indicates that you're done.

TEXTEDIT allows you to edit single lines of text. It does not, however, allow you to erase single lines of text. Use TEXTDEL for deleting your text.

Cursoring through lines of text to edit causes the surrounding box to be partially erased. The box can be redrawn by with the PON command.

**SEE ALSO:** TEXTCUR  
TEXTDEL

## TEXTMOVE

- PURPOSE:** Moves a block of text created with TEXTCUR.
- RETURNS:** Nothing
- REMARKS:** TEXTMOVE surrounds each text block in a window with four "handles", one at each corner. You may choose a text block by moving the mouse cursor from within a text block while pressing the left mouse button . When you begin moving your text, all handles disappear and a rubberband box replaces the handles around the chosen text block. Releasing the mouse button completes the move.
- You are free to move multiple blocks of texts with TEXTMOVE.
- To leave TEXTMOVE, press the right mouse key (or ESC).
- SEE ALSO:** LINEMOVE

## TICKFORM

- PURPOSE:** Displays data points in tick chart form.
- RETURNS:** Nothing.
- REMARKS:** TICKFORM depends on the number of series in the window. With one series, the window displays as sticks, dropping to zero. With two series in the window, the (positive) difference between the two is shown as floating bars. A third series registers as a tick mark to the right, and a fourth adds a tick mark to the left. For sensible displays in this form, it is important that the series be overplotted in the correct order (i.e. close, high, low, open).
- SEE ALSO:** BARMON                      BARS  
LINES                                  POINTS  
STICKS                                TABLEVIEW  
PCTSTACK

## TILE

- PURPOSE:** Arranges the screen into equal-sized windows.
- RETURNS:** Nothing.
- SEE ALSO:** COLLAYOUT  
ROWLAYOUT  
NEATEN

## **TLABEL(string1, ..., stringn)**

|                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>              | Labels the points in a series.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>string1, ..., stringn</b> | A quoted string.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>RETURNS:</b>              | Nothing.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>EXAMPLE:</b>              | <pre>GSER(1,2,3);TLABEL("Larry", "Moe", "Curly")</pre> <p>labels the three points in the generated series "Larry", "Moe", and "Curly", respectively.</p>                                                                                                                                                                                                                                                                                                                                                                    |
| <b>REMARKS:</b>              | <p>This function can be useful in conjunction with the STRFILE function. For example, given the file 'stooges.txt' with the contents:</p> <pre>Larry<br/>Moe<br/>Curly</pre> <pre>TLABEL(STRFILE("stooges.txt"))</pre> <p>would label the first three points in the series "Larry", "Moe", and "Curly", respectively. Like other text annotation functions, if you are adding text to a graph after the graph has been plotted, you need to append the TLABEL command to the window formula using the ADDWFORM command.</p> |
| <b>SEE ALSO:</b>             | TEXTANN<br>ADDWFORM<br>STRFILE                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

## **TLEGEND(box, string1, ..., stringn)**

|                              |                                                                                                                                               |
|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>              | Writes a vertical list of strings, with the first string at the top and at the origin established by TORIGIN.                                 |
| <b>box</b>                   | (Optional). Integer. 1 = draw box; 0 = do not draw box. The default is 0.                                                                     |
| <b>string1, ..., stringn</b> | A list of strings, in quotes.                                                                                                                 |
| <b>RETURNS:</b>              | If box is set to 1, a box is drawn around the strings.                                                                                        |
| <b>REMARKS:</b>              | Unlike TEXTCUR, this formula has no effect if executed when the window is activated. It must be part of the window formula to have an effect. |
| <b>SEE ALSO:</b>             | TEXTCUR<br>TORIGIN                                                                                                                            |

## **TOCONTINUOUS(insert)**

**PURPOSE:** Takes an input series or matrix and copies it. If the input has horizontal units of discrete time of daily or lower frequency ("Daily", "Weekly", etc.), the output is converted to a continuous time series, with horizontal units and an appropriate DELTAX (e.g. 86400 for Daily).

**insert** The input series or matrix.

**RETURNS:** A series.

**REMARKS:** NAs are preserved, so this function is not symmetrical with TODISCRETE() regarding NAs.

**SEE ALSO:** TODISCRETE

## **TODISCRETE(insert)**

**PURPOSE:** Takes an input series or matrix and copies it. If the input has horizontal units "Real Time" and it is daily or lower frequency (its DELTAX is  $\geq 86400$ ), it is converted to a discrete series, with units such as Daily, Weekly, etc. as appropriate.

**insert** The input series or matrix.

**RETURNS:** A series.

**REMARKS:** Non business day gaps show as NAs.

**SEE ALSO:** TOCONTINUOUS

## **TODSTR(window, time)**

**PURPOSE:** Returns the index number of the data point corresponding to a given time.

**window** (Optional). A window reference. Defaults to the current window.

**time** A time string, in quotes.

**RETURNS:** An integer nearest the specified time.

**SEE ALSO:** STRTOD

## **TOKENIZE(str, which, trim\_quotes, returnEOL)**

|                    |                                                                                                                                                                                                                                                                                                                                                             |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>    | Takes a string and returns a string which is the nth token in the string, where n is specified by "which" (origin 1).                                                                                                                                                                                                                                       |
| <b>str</b>         | The string to tokenize.                                                                                                                                                                                                                                                                                                                                     |
| <b>which</b>       | An integer greater than zero. Default is 1. Which token to return from the string.                                                                                                                                                                                                                                                                          |
| <b>trim_quotes</b> | An integer, 0 or 1, where 1 indicates to strip the enclosing quotes from a token. Default is 1                                                                                                                                                                                                                                                              |
| <b>returnEOL</b>   | An integer, 0 or 1. How to signify no token found. When returnEOL is 1, a request for a non-existent token returns the special string "*EOL*". When returnEOL is 0, a request for a non-existent token returns a string of length 0. Default is 1.                                                                                                          |
| <b>RETURNS:</b>    | A string.                                                                                                                                                                                                                                                                                                                                                   |
| <b>REMARKS:</b>    | Tokens are delimited by whitespace or enclosing quotation marks. Placeholders for tokens can be indicated by pairs of single or double quotes. Special consideration must be given to distinguish empty tokens from non-existent tokens. When trim_quotes is on, and returnEOL is off, it is impossible to distinguish a missing token from an empty token. |
| <b>SEE ALSO:</b>   | TOKENWRAP                                                                                                                                                                                                                                                                                                                                                   |

## **TOKENWRAP(str, wrapper)**

|                  |                                                                                                                                                                                                                                                                                                                                                                  |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Takes a string and returns the same string wrapped in quotation marks if the input string is of zero length or contains whitespace or contains exactly one kind of quotation mark. Its purpose is to convert a string into a form suitable for parsing by the function TOKENIZE. The string will be returned as-is if it contains both types of quotation marks. |
| <b>str</b>       | The string to wrap.                                                                                                                                                                                                                                                                                                                                              |
| <b>wrapper</b>   | An integer, 1 for single quote or 2 for double quote (default: 2). This argument suggests which type of quotation mark to use if TOKENWRAP is not forced by the existence of one type of quotation mark to use the other type of quotation to wrap with.                                                                                                         |
| <b>RETURNS:</b>  | A string.                                                                                                                                                                                                                                                                                                                                                        |
| <b>SEE ALSO:</b> | TOKENIZE                                                                                                                                                                                                                                                                                                                                                         |

## TOLOWER, TOUPPER(string)

**PURPOSE:** Converts the case of a string.

**string** A string, in quotes.

**RETURNS:** A string.

**EXAMPLES:** TOLOWER("Myfile.dat")

Returns "myfile.dat".

TOUPPER("myfile.dat")

Returns "MYFILE.DAT"

**SEE ALSO:** INPUT  
STRCAT

## TOOLBAR( which\_toolbar, which\_button, method, fg, bg, action\_key, label, command )

**PURPOSE:** Edits the properties of an MarketBrowser toolbar.

**which\_toolbar** The toolbar to which you want to add or remove a button. No default. Valid choices are:

- 1 - main worksheet toolbar
- 2 - activated window toolbar
- 3 - data cursor toolbar

**which\_button** The location on the toolbar, counted from left, starting from 1. No default. This count refers to the locations of the pre-installed buttons.

If you use TOOLBAR to hide the first button on a toolbar, the second button is still referred to as button 2, although it will appear first on the screen.

**method** The method for rendering the buttons on the screen. No default. Valid options are:

- 1 - BIT MAPPED. Button is rendered using either its pre-installed bitmap or the "label" (See below).
- 2 - DRAWN. Button is rendered using its pre-installed drawn figure.
- 4 - WRITTEN. Button is a string, supplied by "label"
- 8 - REMOVE. The button is removed from the screen.

**fg, bg** (Optional). Integer or macro color name. Foreground and background colors. Color mapping varies, depending on "method."

**action\_key** (Optional). Integer. Returns a single character code to the application. Keys are integer key codes, based on ASCII. Non-ASCII keys are private to the application. Action\_keys are used internally in the application and are mentioned here for completeness, but "command" strings (below) are the preferred method of customization.

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>label</b>    | The label for the button under the “WRITTEN” rendering, in quotes.                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>command</b>  | String in quotes. The action taken when the button is pressed (can include any commands that can be executed in the application’s current state).                                                                                                                                                                                                                                                                                                     |
| <b>RETURNS:</b> | Nothing.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>EXAMPLE:</b> | <p>Add a button called “Stats”, to the main worksheet toolbar, which pops up the “Vital Statistics” menu:</p> <pre>TOOLBAR(1, -1, 4, RED, “ Stats”, “_MF('statvit.men')”)</pre> <p>Once the MarketBrowser screen has redrawn (e.g., by zooming a window or by resizing the screen), a new toolbar button will appear.</p> <p>Likewise, you can convert the “style” button to a menu of choices:</p> <pre>TOOLBAR(1,5,2, “ ”, “_MF('view.men')”)</pre> |
| <b>REMARKS:</b> | <p>“Command,” if present, overrides “action_key”. An empty command string (“”) will allow the “action_key”, if any, to take precedence.</p> <p>BIT_MAPPED rendering is not currently available on UNIX platforms.</p> <p>It is possible to install buttons that are inappropriate to the state of the worksheet (e.g., a button to fetch new data, which would be fine on the main toolbar, would be inappropriate to the data cursor toolbar).</p>   |

## **TORIGIN(x-coord, y-coord, target, fg\_color, bg\_color, font)**

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Establishes the origin for placing text.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>x-coord</b>   | X coordinate of the origin.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>y-coord</b>   | Y coordinate of the origin.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>target</b>    | Integer. Coordinate type: 0 = paper, 1 = glass                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>fg_color</b>  | Integer. The foreground color or -1 (default).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>bg_color</b>  | Integer. The background color or -1 (default).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>font</b>      | Integer font number: 0 = large; 1 = small. The default is 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>REMARKS:</b>  | <p>The parameter x-coord and y-coord use the same units as the data displayed in the window. However, the origin may vary: if the target is 0 (paper), the origin of the coordinate system is the same as the origin of the series; if the target is 1 (glass), the origin of the coordinate system is the center of the window.</p> <p>If fg_color is -1, then the foreground color is the color of the primary series in the window. If bg_color is -1, then the background color is the background color of the window.</p> <p>Unlike TEXTCUR, this formula has no effect if executed when the window is activated. It must be part of the window formula.</p> |
| <b>SEE ALSO:</b> | <p>TEXTCUR</p> <p>TLEGEND</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

## TOTAL(series)

|                         |                                             |
|-------------------------|---------------------------------------------|
| <b>PURPOSE:</b>         | (A Macro). Sums a series                    |
| <b>series</b>           | A series or table.                          |
| <b>RETURNS:</b>         | A number.                                   |
| <b>EXPAN-<br/>SION:</b> | REDUCE(series, '+')                         |
| <b>EXAMPLE:</b>         | TOTAL(GSER(1,2,3))<br>returns the value 6.0 |

## TPRINT(row, column, string1, ..., stringn)

|                                  |                                                                                                                                                                                                                                                                         |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>                  | Places text in a window, using row and column coordinates.                                                                                                                                                                                                              |
| <b>row</b>                       | Integer. The row number                                                                                                                                                                                                                                                 |
| <b>column</b>                    | Integer. The column number                                                                                                                                                                                                                                              |
| <b>string1, ...,<br/>stringn</b> | List of strings                                                                                                                                                                                                                                                         |
| <b>RETURNS:</b>                  | Nothing.                                                                                                                                                                                                                                                                |
| <b>REMARKS:</b>                  | The origin established in TORIGIN is not used. This function is usually used in windows not containing data plots.<br><br>Unlike TEXTCUR, this formula has no effect if executed when the window is activated. It must be part of the window formula to have an effect. |

## TRACE(matrix)

|                         |                                                                  |
|-------------------------|------------------------------------------------------------------|
| <b>PURPOSE:</b>         | (A Macro). Calculates the sum of the major diagonal of a matrix. |
| <b>matrix</b>           | A square matrix.                                                 |
| <b>RETURNS:</b>         | A number.                                                        |
| <b>EXPAN-<br/>SION:</b> | REDUCE(DIAG(M), "+")                                             |

## TRANSPOSE(matrix)

|                 |                                         |
|-----------------|-----------------------------------------|
| <b>PURPOSE:</b> | Swaps the rows and columns of a matrix. |
| <b>matrix</b>   | Input matrix; must be rectangular.      |
| <b>RETURNS:</b> | A matrix.                               |

**SEE ALSO:**     SVD

## TRIG FUNCTIONS(expr)

**PURPOSE:** Calculates the trig functions of a window.

**expr** Any expression evaluating to a series, table, integer, real or complex number. MarketBrowser assumes operation is in radians unless you have invoked the SETDEGREE function. Real input values must be in the range -1 to +1 inclusive.

**RETURNS:** A series, table or number.

| <b>Function Name</b> | <b>Description:</b>                                                |
|----------------------|--------------------------------------------------------------------|
| ACOS                 | Calculates the arc-cosine of any expression.                       |
| ACOSH                | Calculates the hyperbolic arc-cosine of any expression in radians. |
| ACOT                 | Calculates the arc-cotangent of any expression.                    |
| ACOTH                | Calculates the hyperbolic arc-cotangent of any expression.         |
| ACSC                 | Calculates the arc-cosecant of any expression.                     |
| ACSCH                | Calculates the hyperbolic arc-cosecant of any expression.          |
| ANGLE                | Calculates the phase component of a complex expression.            |
| ASEC                 | Returns the arc-secant of any expression.                          |
| ASECH                | Calculates the hyperbolic arc-secant of any expression.            |
| ASIN                 | Calculates the arc-sine of any expression.                         |
| ASINH                | Calculates the hyperbolic arc-sine of any expression.              |
| ATAN                 | Calculates the arc-tangent of any expression.                      |
| ATANH                | Calculates the hyperbolic arc-tangent of any expression.           |

| <b>Function Name</b> | <b>Description:</b>                                  |
|----------------------|------------------------------------------------------|
| COS                  | Calculates the cosine of any expression.             |
| COSH                 | Calculates the hyperbolic cosine of any expression.  |
| COT                  | Calculates the cotangent of any expression.          |
| COTH                 | Returns the hyperbolic cotangent of any expression.  |
| CSC                  | Returns the cosecant of any expression.              |
| CSCH                 | Returns the hyperbolic cosecant of any expression.   |
| SEC                  | Calculates the secant of any expression.             |
| SECH                 | Calculates the hyperbolic secant of any expression.  |
| SIN                  | Calculates the sine of any expression.               |
| SINH                 | Calculates the hyperbolic sine of any expression.    |
| TAN                  | Calculates the tangent of any expression.            |
| TANH                 | Calculates the hyperbolic tangent of any expression. |

## TRIG GENERATORS (points, spacing, factor, offset)

**PURPOSE:** Generates trig functions in a specified window.

**points** Number of points in the curve.

**spacing** Spacing between each point on the x-axis measured in seconds.

**factor** (Optional). A multiplicative factor to expand or contract the waveform along the x-axis. The default is 1.

**offset** (Optional). Operand used to adjust the x position of the waveform, specified in radians. The default shift is 0.

**RETURNS:** A series or table.

| Function Names | Description                                                                             |
|----------------|-----------------------------------------------------------------------------------------|
| GACOS          | Generates an arc-cosine curve in accordance with the specified parameters.              |
| GACOSH         | Generates a hyperbolic arc-cosine curve in accordance with the specified parameters.    |
| GACOT          | Generates an arc-cotangent curve in accordance with the specified parameters.           |
| GACOTH         | Generates a hyperbolic arc-cotangent curve in accordance with the specified parameters. |
| GACSC          | Generates an arc-cosecant curve in accordance with the specified parameters.            |
| GACSCH         | Generates a hyperbolic arc-cosecant curve in accordance with the specified parameters.  |
| GASEC          | Generates an arc-secant curve in accordance with the specified parameters.              |
| GASECH         | Generates a hyperbolic arc-secant curve in accordance with the specified parameters.    |
| GASIN          | Generates an arc-sine curve in accordance with the specified parameters.                |
| GASINH         | Generates a hyperbolic arc-sine curve in accordance with the specified parameters.      |
| GATAN          | Generates an arc-tangent curve in accordance with the specified parameters.             |

| <b>Function Names</b> | <b>Description</b>                                                                    |
|-----------------------|---------------------------------------------------------------------------------------|
| GATANH                | Generates a hyperbolic arc-tangent curve in accordance with the specified parameters. |
| GCOS                  | Generates a cosine curve in accordance with the specified parameters.                 |
| GCOSH                 | Generates a hyperbolic cosine curve in accordance with the specified parameters.      |
| GCOT                  | Generates a cotangent curve in accordance with the specified parameters.              |
| GCOTH                 | Generates a hyperbolic cotangent curve in accordance with the specified parameters.   |
| GCSC                  | Generates a cosecant curve in accordance with the specified parameters.               |
| GCSCH                 | Generates a hyperbolic cosecant curve in accordance with the specified parameters.    |
| GSEC                  | Generates a secant curve in accordance with the specified parameters.                 |
| GSECH                 | Generates a hyperbolic secant curve in accordance with the specified parameters.      |
| GSIN                  | Generates a sine curve in accordance with the specified parameters.                   |
| GSINC                 | Generates a SINC function ( $\sin(x)/x$ ) in accordance with the specified parameter. |
| GSINH                 | Generates a hyperbolic sine curve in accordance with the specified parameters.        |
| GTAN                  | Generates a tangent curve in accordance with the specified parameters.                |
| GTANH                 | Generates a hyperbolic tangent curve in accordance with the specified parameters.     |

## TRUNC(expr)

- PURPOSE:** Finds the greatest integer less than or equal to the input value.
- expr** Any expression evaluating to a scalar, series, table, integer, or real or complex number.
- RETURNS:** A scalar, series, table or number.
- EXAMPLES:** TRUNC(3.4)  
displays 3.  
TRUNC(W2)  
creates a new series in the current window by applying TRUNC to each element of W2. The integer value returned by TRUNC is converted to a floating point value.
- SEE ALSO:** ROUNDUP

## TRYGETSERIES(table, n)

- PURPOSE:** Analogous to the GETSERIES function, but does not produce an error if the series is not located.
- table** Any compound data item, i.e., table, matrix, trading bars, etc.
- n** Integer. The number of the column to return.
- RETURNS:** A series.
- EXAMPLE:** Given the variable:  
A = RAVEL(GRANDOM(100,1),25)  
TRYGETSERIES(A, 2)  
returns the second column from the matrix contained in A.  
TRYGETSERIES(A, 5)  
would return nothing.
- REMARKS:** This function is synonymous with the COL macro, which returns a single column of data.  
  
Exercise caution when using this function, as it doesn't return anything if the series is non-existent. This function is mainly reserved for internal use.  
  
If you are further interested in circumventing errors resulting from non-existent series variables, consider using the IGNORE\_HOTVAR\_ERROR configuration variable.
- SEE ALSO:** GETSERIES  
COL

## ULU(matrix)

**PURPOSE:** Computes an upper triangular matrix in LU decomposition.

**matrix** An expression resolving to a real or complex square matrix.

**RETURNS:** A matrix.

**EXAMPLES:** x =

|   |   |    |
|---|---|----|
| 1 | 2 | 3  |
| 4 | 5 | 6  |
| 7 | 8 | 10 |

ULU(x) =

|     |        |        |
|-----|--------|--------|
| 7.0 | 8.0    | 10.0   |
| 0.0 | 0.8571 | 1.5714 |
| 0.0 | 0.0    | -0.5   |

**SEE ALSO:** LU  
LLU

## UNOVERPLOT(window, op\_num)

**PURPOSE:** Removes an overplotted series in the specified window.

**window** (Optional). A window reference. Defaults to the current window.

**op\_num** An integer. The overplot to remove.

**RETURNS:** Nothing.

**EXAMPLE:** UNOVERPLOT(W7, 3)  
removes the third overplotted series in window 7.

**REMARKS:** OVERPLOT(0) will also clear all overplotted series from the current window.

**SEE ALSO:** OVERPLOT

## UNPOPWINDOW(window)

**PURPOSE:** Unzooms a specified window.

**window** Window reference

**RETURNS:** Nothing.

**EXAMPLE:** UNPOPWINDOW(W3)  
unzooms window 3.

**SEE ALSO:** POPWINDOW  
ZOOM  
UNZOOM

## UNRAVEL(table)

**PURPOSE:** Creates a series from the columns of a table.

**table** A table of any shape.

**RETURNS:** A series.

**EXAMPLE:** UNRAVEL(W1)  
If W1 contains a 3 by 3 matrix, this expression produces a series with 9 observations; the elements of column 1 of W1, followed by the elements of column 2 of W1, followed by the elements of the last column of W1.

**SEE ALSO:** RAVEL

## UNWIND

**PURPOSE:** Reverts MarketBrowser to a known state, that is, idle in a window. For example, it could deactivate a window, take you out of line or text annotation modes, or clear a menu.

**RETURNS:** Nothing.

**REMARKS:** This function is useful when controlling MarketBrowser from an outside program, as it guarantees that MarketBrowser is in a known state.

## **UNZOOM(window)**

- PURPOSE:** Returns a zoomed window to its normal size.
- window** (Optional). A window reference. Defaults to the current window.
- RETURNS:** Nothing.
- EXAMPLE:** See ZOOM
- SEE ALSO:** ZOOM  
UNPOPWINDOW

## **UPDATE**

- PURPOSE:** Updates each formula in a worksheet window.
- RETURNS:** The entire worksheet is re-evaluated - just as if each formula were to be re-typed.
- EXAMPLE:** UPDATE
- MarketBrowser sequences through each window in the worksheet and re-evaluates all formulas. Unlike REFRESH, each formula is re-evaluated just as if it were manually re-entered into the window. UPDATE is useful for updating worksheets from prior versions of MarketBrowser to the latest version. This is particularly true if the old worksheet contains functions that have been revised in the latest MarketBrowser release.
- SEE ALSO:** CALC  
REFRESH

## USCHUR(matrix)

**PURPOSE:** Computes the Unit Schur form of a matrix.

**matrix** An expression resolving to a real or complex square matrix.

**RETURNS:** A matrix.

**EXAMPLES:**  $x =$

|   |   |    |
|---|---|----|
| 1 | 3 | 4  |
| 5 | 6 | 7  |
| 8 | 9 | 12 |

SCHUR(x) =

|        |         |         |
|--------|---------|---------|
| 19.964 | 4.353   | -2.2431 |
| 0.0    | -1.4739 | 0.1399  |
| 0.0    | 0.0     | 0.50976 |

USCHUR(x)=

|         |          |           |
|---------|----------|-----------|
| 0.25387 | 0.96612  | -0.046551 |
| 0.50456 | -0.17334 | 0.84579   |
| 0.82521 | -0.19124 | 0.53147   |

$x = \text{USCHUR}(x) * \text{SCHUR}(x) * \text{TRANSPPOSE}(\text{USCHUR}(x)) * \text{SCHUR}(x)$

gets a Schur matrix,

$\text{USCHUR}(x)$

gets a unitary matrix.

$x = \text{USCHUR}(cx) * \text{SCHUR}(x) * \text{TRANSPPOSE}(\text{USCHUR}(x)) * \text{TRANSPPOSE}(\text{USCHUR}(x)) * \text{USCHUR}(x)$

is an identity matrix which is the same size as  $x$ .

**REMARKS:** If matrix  $x$  is complex, USCHUR returns the upper triangular matrix with the Eigenvalues of the matrix on the diagonal. If matrix  $x$  is real, USCHUR returns the Schur form that has the real Eigenvalues on the diagonal and the complex Eigenvalues in 2-by-2 blocks on the diagonal.

**SEE ALSO:** SCHUR

## VALUETYPE(variable)

|                  |                                                                                                                                                                                                                                                                                                                                                                  |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Returns the type of a given XPL variable.                                                                                                                                                                                                                                                                                                                        |
| <b>variable</b>  | XPL variable, optionally in quotes.                                                                                                                                                                                                                                                                                                                              |
| <b>type</b>      | (Optional). Integer. By default, VALUETYPE will find only global variables. To find other types, supply this second argument. Options are: <ul style="list-style-type: none"><li>• 1 - Global variable (default)</li><li>• 2 - Local variable</li><li>• 3 - User-defined function</li><li>• 4 - Hot variable (real-time)</li><li>• 5 - Formal variable</li></ul> |
| <b>RETURNS:</b>  | A integer which represents the given variable's type. Possible XPL variable types are: <ul style="list-style-type: none"><li>• 1 - Integer</li><li>• 2 - Real</li><li>• 3 - Complex</li><li>• 4 - String</li><li>• 5 - Series</li></ul>                                                                                                                          |
| <b>EXAMPLE:</b>  | a = GSER(1,2,3); VALUETYPE("a")<br>returns the integer 5, which indicates that "a" is a series variable.                                                                                                                                                                                                                                                         |
| <b>SEE ALSO:</b> | ISVAR<br>EVAL<br>EVALTOSTR<br>CAST                                                                                                                                                                                                                                                                                                                               |

## VARS

|                  |                                                                                           |
|------------------|-------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Lists the current values of all the variables defined in an open MarketBrowser worksheet. |
| <b>RETURNS:</b>  | Nothing.                                                                                  |
| <b>SEE ALSO:</b> | FUNCTIONS                                                                                 |

## **VARWRITE(filename, prefix, regexp1, ..., regexpn, flag, exit\_policy, case\_sense, append)**

|                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>              | Writes variables defined within an MarketBrowser worksheet to an external ASCII file. Only writes variables that have scalar values (integers, reals, strings).                                                                                                                                                                                                                                                                                    |
| <b>filename</b>              | Quoted string. The path and filename to which MarketBrowser will write the variables.                                                                                                                                                                                                                                                                                                                                                              |
| <b>prefix</b>                | (Optional). Quoted string. A string prefix prepended to every variable written to <b>filename</b> . Defaults to no prefix. Use "" (an empty pair of quotes) to maintain the default.                                                                                                                                                                                                                                                               |
| <b>regexp1, ..., regexpn</b> | (Optional). Quoted string(s). Valid regular expressions that filter which variables get written to disk. See a definition of regular expressions under the REMARKS section.                                                                                                                                                                                                                                                                        |
| <b>flag</b>                  | (Optional). Integer flag. This argument is processed before MarketBrowser has filtered for any regular expressions. <ul style="list-style-type: none"><li>• 0 - Write non-transient variables only</li><li>• 1 - Write all variables</li><li>• 2 - Write transient variables only</li><li>• 3 - Write visible variables only</li><li>• 4 - Write hidden (system) variables only.</li></ul>                                                         |
| <b>exit_policy</b>           | (Optional). Determines the verbosity with which MarketBrowser returns error messages. Options are: <ul style="list-style-type: none"><li>• 0 - Beep and print message upon error. Return error if error occurs (default).</li><li>• 1 - Return error if error occurs. Do not print message or beep.</li><li>• 2 - If error occurs, print message, but return OK.</li><li>• 3 - Return silently without error regardless of error status.</li></ul> |
| <b>case_sense</b>            | (Optional). Integer flag. Converts all variables to upper case. Options are: <ul style="list-style-type: none"><li>• 0 - Causes all variables and regular expressions to be converted to upper case before being compared.</li><li>• 1 - Case sensitive. No conversion (default).</li></ul>                                                                                                                                                        |
| <b>append</b>                | (Optional). Integer. Overwrite or append existing variable file. Options are: <ul style="list-style-type: none"><li>• 0 - Overwrite any existing variable file.</li><li>• 1 - Append to any existing variable file.</li></ul>                                                                                                                                                                                                                      |
| <b>quote_mode</b>            | (Optional). When writing a string to file, determines whether or not to surround the string in quotes, and if so, which type of quotes. Options are: <ul style="list-style-type: none"><li>• 0 - No quotes</li><li>• 1 - Surround the string in double quotes (default).</li><li>• 2 - Surround the string in single quotes.</li></ul>                                                                                                             |
| <b>RETURNS:</b>              | Nothing.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>EXAMPLE:</b>              | VARWRITE('myvars', '', '*FX?', 0, 0, 1, 2, 1,1)                                                                                                                                                                                                                                                                                                                                                                                                    |

appends any variables that fit the regular expression \*FX?. to the file 'myvars' in MarketBrowser's main installation directory. Variable names are converted to uppercase. In the case of an error, MarketBrowser prints the error message, but returns OK.

**REMARKS:** An example of a regular expression is the string:

"DEFAULT[1-9]\_\*"

MarketBrowser would write to file all macros that begin with the string DEFAULT, followed by the number 1 through 9, followed by an \_ (underscore) character, and ending with any sequence of printable characters. Any number of regular expressions may be entered on the command line, and a macro must match AT LEAST one of these to get written. If you do not provide any regular expression filters, all macros get written to file.

**SEE ALSO:**       MACREAD                               XPLREAD  
                  XPLWRITE

## **VIEWFILE(x, y, filename)**

**PURPOSE:** Displays the contents of an ASCII file on the screen.

**x**                       (Optional). The x coordinate in text columns. The default is centered.

**y**                       (Optional). The y coordinate in text rows. The default is centered.

**filename**               The name of the file to display, in quotes.

**RETURNS:** Nothing.

**REMARKS:** X and Y are calculated in characters from the upper left of the MarketBrowser session window. The default is centered. A value of 0 will cascade the display in the context of any currently popped menus.

VIEWFILE is really a special case of MENUFILE, with menu preprocessing suppressed.

**SEE ALSO:**       MENUCLEAR                            MENUFILE  
                  MENULIST                            MENUPRINT  
                  INPUT

## WAITCURSOR(**onoff**)

**PURPOSE:** Turns the hourglass cursor on or off explicitly during lengthy sections of XPL code.

**onoff** Use 1 to turn the hourglass cursor on, use 0 to turn it off.

**RETURNS:** Nothing

**EXAMPLE:** This should typically be used in the idiom:

```
myxpl()
(
("WAITCURSOR(0)");
WAITCURSOR(1);
...do something lengthy...
)
```

**REMARKS:** Note that this function should be used carefully to avoid confusing the overall state of the hourglass cursor.

## WAITFILE(**filename, timeout, settle**)

**PURPOSE:** Waits for file to exist and stop changing. Useful for forcing asynchronous RUN or DDE commands which create files to appear synchronous.

**filename** The name of the file in quotes.

**timeout** The maximum number of seconds to wait for success.

**settle** (Optional) The number of seconds the file size should be stable before returning.

**RETURNS:** 1 if successful, 0 for failure.

**EXAMPLE:** In an XPL routine:

```
RUN('DEL tmpfile',-1); RUN('REQ_TMP',-1); /* ask for tmpfile to be created */
r = WAITFILE('tmpfile', 15, 2); /* wait up to 15 seconds for its file size to be stable
for 2 seconds */if (r)
dataret = readany('tmpfile');
```

**SEE ALSO:** RUN  
DDE Commands

## **WATERFALL(hide, hatch\_style, hatch\_interval, h\_exp, v\_exp, series1, ..., seriesn)**

|                              |                                                                                                                                                                                                                                                                                                                                                                    |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>              | Displays several series in waterfall format.                                                                                                                                                                                                                                                                                                                       |
| <b>hide</b>                  | Integer. Toggles line hiding. 1 = ON; 0 = OFF.                                                                                                                                                                                                                                                                                                                     |
| <b>hatch_interval</b>        | Interval (number) used to draw cross hatch grids on the waterfall surface. 0.0 means no cross hatching, -1.0 means to hatch at the scale tick interval.                                                                                                                                                                                                            |
| <b>h_exp</b>                 | Fraction (>0) by which to shift traces horizontally.                                                                                                                                                                                                                                                                                                               |
| <b>v_exp</b>                 | Fraction (>0) by which to shift traces vertically.                                                                                                                                                                                                                                                                                                                 |
| <b>series1, ..., seriesn</b> | Series or matrix of data to plot. A list of individual series are raveled into a matrix                                                                                                                                                                                                                                                                            |
| <b>RETURNS:</b>              | A matrix, displayed by default as a waterfall plot.                                                                                                                                                                                                                                                                                                                |
| <b>EXAMPLE:</b>              | <p>WATERFALL(RAVEL(W1, 100))</p> <p>If W1 contains 1000 data points, ravel will provide a list of 100 point series for waterfall plotting. By default, the waterfall plot will include hidden line removal and cross hatching of the surface at each observation.</p> <p>WATERFALL(W1, W2, W3, W5)</p> <p>displays the four source series in waterfall format.</p> |
| <b>REMARKS:</b>              | The expansion factors are expressed as a fraction which is multiplied by the range of the data and added in to the starting point of the trace. The defaults are 0.03 and 0.03, which shift each trace up by three percent and over by three percent.                                                                                                              |
| <b>SEE ALSO:</b>             | RAVEL<br>WFSET<br>SHADEWITH                                                                                                                                                                                                                                                                                                                                        |

## **WFSET(hide, hatch\_style, hatch\_interval, h\_exp, v\_exp)**

|                       |                                                                                                                                |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>       | Sets the attributes of a Waterfall plot.                                                                                       |
| <b>hide</b>           | Integer - 0 = No line hiding, 1 = Line hiding. The default is 1.                                                               |
| <b>hatch_style</b>    | Integer - 0 = No cross hatch, 1 = On X-axis tics, 2 = On every point, 3 = User defined interval. The default is 2.             |
| <b>hatch_interval</b> | Real - Meaningful only when hatch_style is User defined.                                                                       |
| <b>h_exp</b>          | Real - Fraction used to shift traces horizontally. Higher shift provides more of an end view of the plot. The default is 0.03. |
| <b>v_exp</b>          | Real - Fraction used to shift traces vertically. Higher shift provides more of a top view of the plot. The default is 0.03.    |
| <b>RETURNS:</b>       | Nothing. Effects are seen when window is next plotted.                                                                         |
| <b>EXAMPLES:</b>      | WFSET(-1, -1, -1, 0.01)<br>reduces the horizontal shift to 1 percent.                                                          |
| <b>REMARKS:</b>       | These arguments can also be applied directly in the WATERFALL function itself.                                                 |
| <b>SEE ALSO:</b>      | WATERFALL                                                                                                                      |

## **WINBOX( OnOff )**

|                 |                                                                                                                             |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b> | Turns on or off the display of the auto legend in the current window.                                                       |
| <b>OnOff</b>    | 0 - Turn off the display of the auto legend. 1 - Turn on the display of the auto legend.                                    |
| <b>RETURNS:</b> | A 0 if display of auto legend is turned off, 1 if it is turned on.                                                          |
| <b>REMARKS:</b> | When you turn on the auto legend, it follows the display preferences set under the Custom/Auto Legend Settings menu choice. |

## **WINCOLOR(window, color1, color2)**

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>  | Sets the window background color and (optionally) primary series color.                |
| <b>window</b>    | (Optional) Window reference. Defaults to the current window.                           |
| <b>color1</b>    | Integer or macro. The window's background color.                                       |
| <b>color2</b>    | (Optional). Integer or macro designating the color for the first series in the window. |
| <b>RETURNS:</b>  | Nothing                                                                                |
| <b>SEE ALSO:</b> | GETWCOLOR                      SETCOLOR<br>SERCOLOR                                    |

## **WINFORMAT (window, format\_method, reserved, denominator, reduce, trim, reserved, precision)**

|                      |                                                                                                                                                                                                                                                                                                              |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PURPOSE:</b>      | Sets a window's numeric formatting attributes.                                                                                                                                                                                                                                                               |
| <b>window</b>        | (Optional). The window whose numeric formatting you want to set. Defaults to the current window.                                                                                                                                                                                                             |
| <b>format_method</b> | (Optional). Integer, specifying which formatting option you wish to use: <ul style="list-style-type: none"><li>• 1 = Decimal</li><li>• 2 = Fractional</li></ul>                                                                                                                                              |
| <b>reserved</b>      | (Optional). Reserved for future use. Use a -1 as a placeholder.                                                                                                                                                                                                                                              |
| <b>denominator</b>   | (Optional). Integer. The denominator of the fraction if format_method is set to 2. Can be one of {8,16,32,64,128}.                                                                                                                                                                                           |
| <b>reduce</b>        | (Optional). Integer flag, where: <ul style="list-style-type: none"><li>• 0 = do not reduce fractions</li><li>• 1 = reduce fractions before display</li></ul>                                                                                                                                                 |
| <b>trim</b>          | (Optional). Integer flag, where <ul style="list-style-type: none"><li>• 0 = Do not leave off denominator in a fractional display</li><li>• 1 = shorten display by leaving off denominator</li></ul>                                                                                                          |
| <b>reserved</b>      | (Optional). Reserved for future use. Use a -1 as a placeholder.                                                                                                                                                                                                                                              |
| <b>precision</b>     | (Optional). Integer. If format_method is set to 1 (decimal), sets the precision of the decimal display to {0..8}.                                                                                                                                                                                            |
| <b>RETURNS:</b>      | Nothing                                                                                                                                                                                                                                                                                                      |
| <b>EXAMPLE:</b>      | The command:<br><pre>WINFORMAT(2,-1,16,0,1)</pre> sets the current window to display fractions as shortened, unreduced sixteenths. WINFORMAT(1) uses decimals with program defaults. WINFORMAT(1,-1,-1,-1,-1,-1,4) uses 4 decimals.                                                                          |
| <b>REMARKS:</b>      | Note that any of the integer arguments can take a value of -2, which means that the attribute should be "UNDEFINED" and therefore get inherited from MarketBrowser's general configuration. The integer arguments will also accept the value -1 as a "placeholder" to avoid having to specify each argument. |
| <b>SEE ALSO:</b>     | GETWINFORMAT                                                                                                                                                                                                                                                                                                 |

## **WINNAME(window, name)**

**PURPOSE:** Creates an alternative window reference.

**win** (Optional). A window reference. Defaults to the current window.

**name** The window name, in quotes.

**RETURNS:** Nothing.

**EXAMPLE:** WINNAME(W5, "Inflation")  
FFT(Inflation)

is equivalent to FFT(W5).

## **WINSTATUS(attrib)**

**PURPOSE:** Returns the status of the current window.

**attrib** (Optional) Integer value indicating the window attribute to query. Currently, attrib can be one of the following:

- 0 - window number (default)
- 1 - active status
- 2 - zoomed status
- 3 - hidden status

**RETURNS:** For attribute 0, returns the window number. For attributes 1-3, WINSTATUS returns 0 or 1.

## **WRITEA(filename, series)**

- PURPOSE:** Writes a series as an ASCII directly to disk from the worksheet, without a file header (as created by the EXPORT function).
- filename** Name for output file, in quotes. If no path is given, WRITEA puts the file in your current working directory.
- series** (Optional) A series or table. Defaults to the current window.
- RETURNS:** Nothing
- EXAMPLE:** WRITEA("A:AUTOS",W7)  
writes the data in window 7 as an ASCII file and stores it on the A drive under the filename, "AUTOS".
- SEE ALSO:** WRITEB READA  
READB READDT

## **WRITEAHIST(filename, series, overwrite, title)**

- PURPOSE:** Writes tables of historical and intraday data to an ASCII file.
- series** (Optional). The series or window reference containing the series to write to ASCII. Defaults to the current window.
- filename** Name of ASCII file to write as multi-column table, in quotes.
- overwrite** (Optional). Integer. 1 = overwrite any existing file; 0 = do not overwrite existing file. The default is 0.
- title** (Optional). Integer. 1 = write out comment field as column title; 0 = do not write comment field as column title. The default is 0.
- RETURNS:** Nothing.
- EXAMPLE:** WRITEAHIST("trades.dat", 1, 0)  
writes contents of current window as ASCII file "trades.dat," overwriting any file of the same name, without column titles.
- REMARKS:** WRITEAHIST formats dates and times as appropriate to the units of the data written. Even if several columns of data values are written, only one set of dates and/or times are written. NAs are written as appropriate.
- SEE ALSO:** READAHIST  
WRITETABLE  
WRITEBHIST

## WRITEB(filename, filetype, series)

**PURPOSE:** Writes one or more series to disk in binary format.

**filename** Name of the output file, in quotes. If no path is given, WRITEB puts the file in the current working directory.

**filetype** Determines the type of binary format in which to store the data file. See the list below.

| Name   | Code | Data Type             | Range                                                      |
|--------|------|-----------------------|------------------------------------------------------------|
| SBYTE  | 1    | Signed Byte           | -128 to +127                                               |
| UBYTE  | 2    | Unsigned Byte         | 0 to 255                                                   |
| BYTE   | 2    | (same as UBYTE)       | 0 to 255                                                   |
| SINT   | 3    | Signed Integer        | -32768 to +32768                                           |
| UINT   | 4    | Unsigned Integer      | 0 to 65536                                                 |
| LONG   | 5    | 4-byte Signed Integer | -2,147,483,648 to +2,147,483,647                           |
| FLOAT  | 6    | 4-byte Floating Point | $-10^{37}$ to $+10^{38}$<br>$-10^{-37}$ to $+10^{-38}$     |
| DOUBLE | 7    | 8-byte Floating Point | $-10^{307}$ to $+10^{308}$<br>$-10^{-307}$ to $+10^{-308}$ |

**series** (Optional). A series or table reference. The default is the series in the current window.

**RETURNS:** Nothing.

**EXAMPLE:** WRITEB("TESTDAT",6,W4)

will write the contents of window 4 out to a binary file named TESTDAT in a floating point format. If using READB to bring the file back into a worksheet, be sure to use the same binary format, i.e. in this case, FLOAT.

**REMARKS:** 1. WRITEB does not write any header information.

2. You can read back in data that you have saved this way with READB

**SEE ALSO:** WRITEA  
READA  
READB



## XOFFSET(series)

- PURPOSE:** Returns the x-offset of a given series.
- series** Series argument. Defaults to current window.
- RETURNS:** The specified window's x-offset
- EXAMPLE:** W1: GRANDOM(100,1);setxoffset(20)  
W2: GLINE(length(W1),1,1,xoffset(W1))  
sets the y-intercept of the line to the x-offset of the source window, that is, 20.
- SEE ALSO:** SETXOFFSET

## XPLLOAD(filename)

- PURPOSE:** Reads in and compiles an ASCII file of XPL functions.
- filename** The name and path to the function file you want to read in, in quotes.
- RETURNS:** Nothing.
- EXAMPLE:** XPLLOAD("xpl/expo.xpl") loads the file "expo.xpl".
- REMARKS:** When a file \*.xpl is successfully loaded, MarketBrowser creates a file \*.opl, which is a compiled version of the \*.xpl file.
- SEE ALSO:** MACREAD XPLREAD  
XPLWRITE FUNCTIONS  
ALLFUNCTIONS

## XPLREAD(filename, invisible, transient)

- PURPOSE:** Reads in, but does not compile, an ASCII file of XPL functions.
- filename** The name and path to the function file you want to read in, in quotes.
- invisible** (Optional). Determines if the functions defined in the file that do not start with underscores (\_) will be displayed in the list displayed with the FUNCTIONS command (XPL / List/Edit XPL Functions). Options are: 0 - Display the functions; 1 - Hide the functions.
- transient** (Optional). Determines if the functions in the file are saved with a worksheet. Options are: 0 - Save all functions that do not start with an underscore (\_) with worksheets; 1 - Do not save any functions with worksheets.
- RETURNS:** Nothing.
- EXAMPLE:** XPLREAD("xpl/expo.xpl") reads in the file "expo.xpl".
- REMARKS:** Use XPLREAD (instead of XPLLOAD) when you do not want to compile a \*.opl file

for the \*.xpl file, for example, when macros are defined in the \*.xpl file.

**SEE ALSO:**      MACREAD                      XPLLOAD  
                  XPLWRITE                      FUNCTIONS  
                  ALLFUNCTIONS

## **XPLWRITE(filename, start, end, allfuncs)**

**PURPOSE:**      Writes the current function list to an external file.

**filename**      Name of external function text file, in quotes.

**start**          (Optional). An integer. The number of the first function in the current macro table that you want to write out to a file. The default is -1 (the first function).

**end**            (Optional). An integer. The last function in the current macro table that you want to write. The default is -1 (the last function in the table).

**allfuncs**      (Optional). Options are:

- 0 - Write visible functions only (default)
- 1 - Write all functions
- 2 - Write hidden (system) functions only
- 3 - Write non-transient functions only
- 4 - Write transient macros only

**RETURNS:**      Nothing.

**EXAMPLE:**      XPLWRITE writes the current function list (one macro per line as displayed by the FUNCTIONS command) to the specified file as ASCII text. For example: XPLWRITE("myfunc.xpl", 4, 15) writes functions 4 through 15 to the file "myfunc.xpl".

**REMARKS:**      Unless otherwise specified, XPLWRITE writes the function file to MarketBrowser's current working directory.

**SEE ALSO:**      XPLREAD                      MACWRITE  
                  FUNCTIONS                    ALLFUNCTIONS

## **XVALS(series)**

**PURPOSE:** Returns the x values from a series.

**series** A series or table to get x values from.

**RETURNS:** A new series containing the x values of series.

**EXAMPLE:** XVALS(W1)  
returns a series that consists of the x values of W1.

**SEE ALSO:** XY  
YVALS  
XYINTERP

## **XY(xseries, yseries)**

**PURPOSE:** Creates an XY plot in a window.

**xseries** Series to be used as X values.

**yseries** Series to be used as Y values.

**RETURNS:** XY series.

**EXAMPLES:** XY(W1,GSIN(100,.01,10))  
creates an XY graph in the current window where the X values are identical to the Y values of W1 and the y values of the XY plot are obtained from the specified GSIN function.

XY(LOG10(XVALS(W1)+.0001),LOG10(W1))  
produces a log-log plot of W1.

**REMARKS:** The arguments of the XY function should have only one column of data.

**SEE ALSO:** XYINTERP  
XVALS  
YVALS  
PLOTTYPE

## **XYINTERP(window, interval)**

- PURPOSE:** Linearly interpolates an XY series to a standard interval series that can be numerically processed by MarketBrowser.
- window** Window containing an XY series.
- interval** (Optional). The number specifying the interpolation interval. Defaults to the smallest x increment.
- RETURNS:** An interval series.
- EXAMPLES:** XYINTERP(W3)  
linearly interpolates the XY series in W3 using the smallest X interval of W3 as the interpolation increment. The X values of the XY series must be monotonically rising (i.e. XYINTERP cannot interpolate things like circles). XYINTERP accepts an interpolation interval, so  
XYINTERP(W3, .01)  
interpolates W3 with an increment of .01. You may also explicitly specify the X and Y values as in:  
XYINTERP(W3, GSIN(100, .01, 4.0))  
In this case, the X values are the Y values of W1 and the Y values of the resulting series are interpolated from the Y values of the sine function. Once an XY series has been interpolated, you can operate with other functions, for example:  
FFT(XYINTERP(W3))
- SEE ALSO:** XY Functions  
XVALS  
YVALS

## **YN(series, order)**

- PURPOSE:** Performs the Bessel function on an entire input series or a single scalar input.
- series** A real series, table or number.
- order** An integer order.
- RETURNS:** Real and integer Bessel functions.

## **YVALS(series)**

- PURPOSE:** Returns the y value from a window.
- series** A series to get y values from.
- RETURNS:** A series that contains the y values of series.
- EXAMPLE:** YVALS(W1)  
returns a series that consists of the y values of W1.
- SEE ALSO:** XVALS  
XY Functions  
XYINTERP

## **ZOOM**

- PURPOSE:** Enlarges the current, active window to fill screen.
- RETURNS:** An enlarged, highlighted window display.
- SEE ALSO:** UNZOOM