

Contents

[M-0] Introduction to the Mata manual

Intro Introduction to the Mata manual

[M-1] Introduction and advice

Intro Introduction and advice
Ado Using Mata with ado-files
First Introduction and first session
help Obtaining help in Stata
How How Mata works
Interactive Using Mata interactively
LAPACK Linear algebra package (LAPACK) routines
Limits Limits and memory utilization
Naming Advice on naming functions and variables
Permutation An aside on permutation matrices and vectors
Returned args Function arguments used to return results
Source Viewing the source code
Tolerance Use and specification of tolerances

[M-2] Language definition

Intro Language definition
break Break out of for, while, or do loop
class Object-oriented programming (classes)
Comments Comments
continue Continue with next iteration of for, while, or do loop
Declarations Declarations and types
do do ... while (exp)
Errors Error codes
exp Expressions
for for (exp1; exp2; exp3) stmt
ftof Passing functions to functions
goto goto label
if if (exp) ... else ...
op_arith Arithmetic operators
op_assignment Assignment operator
op_colon Colon operators
op_conditional Conditional operator
op_increment Increment and decrement operators
op_join Row- and column-join operators
op_kronecker Kronecker direct-product operator
op_logical Logical operators

op_range	Range operators
op_transpose	Conjugate transpose operator
optargs	Optional arguments
pointers	Pointers
pragma	Suppressing warning messages
reswords	Reserved words
return	return and return(exp)
Semicolons	Use of semicolons
struct	Structures
Subscripts	Use of subscripts
Syntax	Mata language grammar and syntax
version	Version control
void	Void matrices
while	while (exp) stmt

[M-3] Commands for controlling Mata

Intro	Commands for controlling Mata
end	Exit Mata and return to Stata
lmbuild	Easily create function library
mata	Mata invocation command
mata clear	Clear Mata's memory
mata describe	Describe contents of Mata's memory
mata drop	Drop matrix or function
mata help	Obtain help in Stata
mata matsave	Save and restore matrices
mata memory	Report on Mata's memory usage
mata mlib	Create function library
mata mosave	Save function's compiled code in object file
mata rename	Rename matrix or function
mata set	Set and display Mata system parameters
mata stata	Execute Stata command
mata which	Identify function
namelists	Specifying matrix and function names

[M-4] Categorical guide to Mata functions

Intro	Categorical guide to Mata functions
Dates	Date and time functions
IO	I/O functions
Manipulation	Matrix manipulation
Mathematical	Important mathematical functions
Matrix	Matrix functions
Programming	Programming functions
Scalar	Scalar mathematical functions
Solvers	Functions to solve $AX=B$ and to obtain A inverse
Standard	Functions to create standard matrices

Stata	Stata interface functions
Statistical	Statistical functions
String	String manipulation functions
Utility	Matrix utility functions

[M-5] Alphabetical index to Mata functions

Intro	Alphabetical index to Mata functions
abbrev()	Abbreviate strings
abs()	Absolute value (length)
adosubdir()	Determine ado-subdirectory for file
all()	Element comparisons
args()	Number of arguments
asarray()	Associative arrays
AssociativeArray()	Associative arrays (class)
ascii()	Manipulate ASCII and byte codes
uchar()	Convert code point to Unicode character
assert()	Abort execution if false
base64encode()	Encode string into Base64 format
blockdiag()	Block-diagonal matrix
bufio()	Buffered (binary) I/O
byteorder()	Byte order used by computer
C()	Make complex
c()	Access c() value
callersversion()	Obtain version number of caller
cat()	Load file into string matrix
chdir()	Manipulate directories
cholesky()	Cholesky square-root decomposition
cholinv()	Symmetric, positive-definite matrix inversion
cholsolve()	Solve $AX=B$ for X using Cholesky decomposition
comb()	Combinatorial function
cond()	Condition number
conj()	Complex conjugate
corr()	Make correlation matrix from variance matrix
cross()	Cross products
crossdev()	Deviation cross products
cvpermute()	Obtain all permutations
date()	Date and time manipulation
deriv()	Numerical derivatives
designmatrix()	Design matrices
det()	Determinant of matrix
_diag()	Replace diagonal of a matrix
diag()	Create diagonal matrix
diag0cnt()	Count zeros on diagonal
diagonal()	Extract diagonal into column vector
dir()	File list

<code>direxists()</code>	Whether directory exists
<code>direxternal()</code>	Obtain list of existing external globals
<code>display()</code>	Display text interpreting SMCL
<code>displayas()</code>	Set display level
<code>displayflush()</code>	Flush terminal-output buffer
<code>Dmatrix()</code>	Duplication matrix
<code>_docx*()</code>	Generate Office Open XML (.docx) file
<code>dsign()</code>	FORTRAN-like <code>DSIGN()</code> function
<code>e()</code>	Unit vectors
<code>editmissing()</code>	Edit matrix for missing values
<code>edittoint()</code>	Edit matrix for roundoff error (integers)
<code>edittozero()</code>	Edit matrix for roundoff error (zeros)
<code>editvalue()</code>	Edit (change) values in matrix
<code>eigensystem()</code>	Eigenvectors and eigenvalues
<code>eigensystemselect()</code>	Compute selected eigenvectors and eigenvalues
<code>eltype()</code>	Element type, organizational type, and type name of object
<code>epsilon()</code>	Unit roundoff error (machine precision)
<code>_equilrc()</code>	Row and column equilibration
<code>error()</code>	Issue error message
<code>errprintf()</code>	Format output and display as error message
<code>exit()</code>	Terminate execution
<code>exp()</code>	Exponentiation and logarithms
<code>factorial()</code>	Factorial and gamma function
<code>favorspeed()</code>	Whether speed or space is to be favored
<code>ferrortext()</code>	Text and return code of file error code
<code>fft()</code>	Fourier transform
<code>fileexists()</code>	Whether file exists
<code>_fillmissing()</code>	Fill matrix with missing values
<code>findexternal()</code>	Find, create, and remove external globals
<code>findfile()</code>	Find file
<code>floatround()</code>	Round to float precision
<code>fmtwidth()</code>	Width of %fmt
<code>fopen()</code>	File I/O
<code>fullsvd()</code>	Full singular value decomposition
<code>geigensystem()</code>	Generalized eigenvectors and eigenvalues
<code>ghessenbergd()</code>	Generalized Hessenberg decomposition
<code>ghk()</code>	Geweke–Hajivassiliou–Keane (GHK) multivariate normal simulator
<code>ghkfast()</code>	GHK multivariate normal simulator using pregenerated points
<code>gschurd()</code>	Generalized Schur decomposition
<code>halton()</code>	Generate a Halton or Hammersley set
<code>hash1()</code>	Jenkins’s one-at-a-time hash function
<code>hessenbergd()</code>	Hessenberg decomposition
<code>Hilbert()</code>	Hilbert matrices
<code>I()</code>	Identity matrix
<code>inbase()</code>	Base conversion
<code>indexnot()</code>	Find byte not in list

<code>_invmat()</code>	Inverse and pseudoinverse of a square matrix
<code>invorder()</code>	Permutation vector manipulation
<code>invsym()</code>	Symmetric real matrix inversion
<code>invtokens()</code>	Concatenate string rowvector into string scalar
<code>isascii()</code>	Whether string scalar contains only ASCII codes
<code>isdiagonal()</code>	Whether matrix is diagonal
<code>isfleeting()</code>	Whether argument is temporary
<code>isreal()</code>	Storage type of matrix
<code>isrealvalues()</code>	Whether matrix contains only real values
<code>issamefile()</code>	Whether two file paths are pointing to the same file
<code>issymmetric()</code>	Whether matrix is symmetric (Hermitian)
<code>isview()</code>	Whether matrix is view
<code>J()</code>	Matrix of constants
<code>Kmatrix()</code>	Commutation matrix
<code>lapack()</code>	Linear algebra package (LAPACK) functions
<code>ldl()</code>	Bunch–Kaufman decomposition
<code>LinearProgram()</code>	Linear programming
<code>liststruct()</code>	List structure’s contents
<code>Lmatrix()</code>	Elimination matrix
<code>logit()</code>	Log odds and complementary log–log
<code>lowertriangle()</code>	Extract lower or upper triangle
<code>lsolve()</code>	Solve $Ax=c$ for x with equality constraints using least-squares method
<code>lsglsolve()</code>	Solves a general Gauss–Markov linear model problem
<code>lssolve()</code>	Solve $AX=B$ for X using least-squares method
<code>lud()</code>	LU decomposition
<code>luinv()</code>	Square matrix inversion
<code>lusolve()</code>	Solve $AX=B$ for X using LU decomposition
<code>makesymmetric()</code>	Make square matrix symmetric (Hermitian)
<code>matexpsym()</code>	Exponentiation and logarithms of symmetric matrices
<code>matpowersym()</code>	Powers of a symmetric matrix
<code>mean()</code>	Means, variances, and correlations
<code>mindouble()</code>	Minimum and maximum nonmissing value
<code>minindex()</code>	Indices of minimums and maximums
<code>minmax()</code>	Minimums and maximums
<code>missing()</code>	Count missing and nonmissing values
<code>missingof()</code>	Appropriate missing value
<code>mod()</code>	Modulus
<code>moptimize()</code>	Model optimization
<code>more()</code>	Create –more– condition
<code>mvnormal()</code>	Compute multivariate normal distributions and derivatives
<code>_negate()</code>	Negate real matrix
<code>norm()</code>	Matrix and vector norms
<code>normal()</code>	Cumulatives, reverse cumulatives, and densities
<code>optimize()</code>	Function optimization
<code>panelsetup()</code>	Panel-data processing

panelsum()	Panel sums
pathjoin()	File path manipulation
Pdf*()	Create a PDF file
pinv()	Moore–Penrose pseudoinverse
polyeval()	Manipulate and evaluate polynomials
printf()	Format output
qrd()	QR decomposition
qrinv()	Generalized inverse of matrix via QR decomposition
qrsolve()	Solve $AX=B$ for X using QR decomposition
quadcross()	Quad-precision cross products
Quadrature()	Numerical integration
range()	Vector over specified range
rank()	Rank of matrix
Re()	Extract real or imaginary part
reldif()	Relative/absolute difference
rows()	Number of rows and number of columns
rowshape()	Reshape matrix
runiform()	Uniform and nonuniform pseudorandom variates
runningsum()	Running sum of vector
schurd()	Schur decomposition
select()	Select rows, columns, or indices
setbreakintr()	Break-key processing
sign()	Sign and complex quadrant functions
sin()	Trigonometric and hyperbolic functions
sizeof()	Number of bytes consumed by object
solve_tol()	Tolerance used by solvers and inverters
solve_lower()	Solve $AX=B$ for X , A triangular
_solve_mat()	Solve $AX=B$ for X
solvenl()	Solve systems of nonlinear equations
sort()	Reorder rows of matrix
soundex()	Convert string to soundex code
spline3()	Cubic spline interpolation
sqrt()	Square root
st_addalias()	Add alias variable to current Stata dataset
st_addobs()	Add observations to current Stata dataset
st_addvar()	Add variable to current Stata dataset
st_data()	Load copy of current Stata dataset
st_dir()	Obtain list of Stata objects
st_dropvar()	Drop variables or observations
st_frame*()	Data frame manipulation
st_global()	Obtain strings from and put strings into global macros
st_isalias()	Properties of alias variable
st_isfmt()	Whether valid %fmt
st_isname()	Whether valid Stata name
st_local()	Obtain strings from and put strings into Stata macros
st_macroexpand()	Expand Stata macros in string
st_matrix()	Obtain and put Stata matrices

<code>st_numscalar()</code>	Obtain values from and put values into Stata scalars
<code>st_nvar()</code>	Numbers of variables and observations, and dataset label
<code>st_rclear()</code>	Clear <code>r()</code> , <code>e()</code> , or <code>s()</code>
<code>st_store()</code>	Modify values stored in current Stata dataset
<code>st_subview()</code>	Make view from view
<code>st_tempname()</code>	Temporary Stata names
<code>st_tsrevar()</code>	Create time-series <code>op.varname</code> variables
<code>st_update()</code>	Determine or set data-have-changed flag
<code>st_varformat()</code>	Obtain/set format, etc., of Stata variable
<code>st_varindex()</code>	Obtain variable indices from variable names
<code>st_varname()</code>	Obtain variable names from variable indices
<code>st_varrename()</code>	Rename Stata variable
<code>st_vartype()</code>	Storage type of Stata variable
<code>st_view()</code>	Make matrix that is a view onto current Stata dataset
<code>st_viewvars()</code>	Variables and observations of view
<code>st_vlexists()</code>	Use and manipulate value labels
<code>stata()</code>	Execute Stata command
<code>stataversion()</code>	Version of Stata being used
<code>strdup()</code>	String duplication
<code>strlen()</code>	Length of string in bytes
<code>ustrlen()</code>	Length of Unicode string in Unicode characters
<code>udstrlen()</code>	Length of Unicode string in display columns
<code>strmatch()</code>	Determine whether string matches pattern
<code>strofreal()</code>	Convert real to string
<code>strpos()</code>	Find substring in string
<code>ustrpos()</code>	Find substring in Unicode string
<code>strreverse()</code>	Reverse string
<code>ustrreverse()</code>	Reverse Unicode string
<code>strtoname()</code>	Convert a string to a Stata 13 compatible name
<code>ustrtoname()</code>	Convert a Unicode string to a Stata name
<code>strtoreal()</code>	Convert string to real
<code>strtrim()</code>	Remove blanks
<code>ustrtrim()</code>	Remove Unicode whitespace characters
<code>strupper()</code>	Convert ASCII letter to uppercase (lowercase)
<code>ustrupper()</code>	Convert Unicode string to uppercase, lowercase, or titlecase
<code>subinstr()</code>	Substitute text
<code>usubinstr()</code>	Replace Unicode substring
<code>sublowertriangle()</code>	Return a matrix with zeros above a diagonal
<code>_substr()</code>	Substitute into string
<code>_usubstr()</code>	Substitute into Unicode string
<code>substr()</code>	Extract substring
<code>usubstr()</code>	Extract Unicode substring
<code>udsubstr()</code>	Extract Unicode substring based on display columns
<code>sum()</code>	Sums
<code>svd()</code>	Singular value decomposition
<code>svsolve()</code>	Solve $AX=B$ for X using singular value decomposition
<code>swap()</code>	Interchange contents of variables
<code>Toeplitz()</code>	Toeplitz matrices

tokenget() Advanced parsing

tokens() Obtain tokens from string

trace() Trace of square matrix

_transpose() Transposition in place

transposeonly() Transposition without conjugation

trunc() Round to integer

uniqrows() Obtain sorted, unique values

unitcircle() Complex vector containing unit circle

unlink() Erase file

urlencode() Convert URL into percent-encoded ASCII format

ustrcompare() Compare or sort Unicode strings

ustrfix() Replace invalid UTF-8 sequences in Unicode string

ustrnormalize() Normalize Unicode string

ustrsplit() Split string into parts based on a Unicode regular expression

ustrto() Convert a Unicode string to or from a string in a specified encoding

ustrunescape() Convert escaped hex sequences to Unicode strings

ustrword() Obtain Unicode word from Unicode string

valofexternal() Obtain value of external global

Vandermonde() Vandermonde matrices

vec() Stack matrix columns

xl() Excel file I/O class

[M-6] Mata glossary of common terms

Glossary

Subject and author index

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow and NetCourseNow are trademarks of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.



For suggested citations, see the FAQ on [citing Stata documentation](#).