

# **JMathLib Documentation**

## **Handbook for using JMathLib**

**Published February 21, 2009**

**Stefan Müller**

---

# JMathLib Documentation: Handbook for using JMathLib

by Stefan Müller

*This release is connected to JMathLib in Version 0.9.4 and following versions*

Published February 21, 2009

Copyright © 2005, 2006, 2007, 2008, 2009 Stefan Müller

## Abstract

JMathLib [<http://www.jmathlib.de>] is a program to evaluate of mathematical functions designed to be used in evaluating complex expressions and display the results graphically. It can be used either interactively or to interpret script files. It is designed to work almost like Matlab [<http://www.mathworks.com>], Octave [<http://www.gnu.org/software/octave/>], FreeMat [<http://freemat.sourceforge.net>] or Scilab [<http://www.scilab.org>]. The link to the sourcecode repository can be found at Sourceforge [<http://sourceforge.net/projects/mathlib/>].

Permission to use, copy, modify and distribute JMathLib accompanying documentation for any purpose and without fee is hereby granted in perpetuity, provided that the above copyright notice and this paragraph appear in all copies.

The copyright holder makes no representation about the fitness of this documentation for any purpose. It is provided “as is”.

---

---

# Table of Contents

|                                               |      |
|-----------------------------------------------|------|
| Foreword .....                                | xiii |
| I. Installing And Running .....               | 1    |
| 1. Installing And Running .....               | 2    |
| Obtaining JMathLib .....                      | 2    |
| Installing JMathLib .....                     | 2    |
| Running JMathLib .....                        | 2    |
| II. Usage .....                               | 4    |
| 2. Basic Operations .....                     | 5    |
| 3. Variables .....                            | 6    |
| 4. Complex Numbers .....                      | 7    |
| 5. Matrices .....                             | 8    |
| 6. Control Structures .....                   | 9    |
| If .....                                      | 9    |
| Switch .....                                  | 9    |
| For .....                                     | 10   |
| While .....                                   | 10   |
| 7. Symbolic computations .....                | 11   |
| III. Datatypes .....                          | 12   |
| 8. Numbers .....                              | 13   |
| Usage .....                                   | 13   |
| Internal - NumericToken.class .....           | 13   |
| 9. Matrices .....                             | 14   |
| Usage .....                                   | 14   |
| Internal - MatrixToken .....                  | 14   |
| 10. Strings .....                             | 15   |
| Usage .....                                   | 15   |
| Internal - StringToken .....                  | 15   |
| 11. Variables .....                           | 16   |
| Usage .....                                   | 16   |
| Internal - VariableToken .....                | 16   |
| 12. Symbolic Variables .....                  | 17   |
| .....                                         | 17   |
| 13. Structures .....                          | 18   |
| Usage .....                                   | 18   |
| Internal - MathLibObject .....                | 18   |
| 14. Casting .....                             | 19   |
| IV. Graphical Subsystem .....                 | 20   |
| 15. Basics about JMathLib graphics .....      | 21   |
| Basic Command .....                           | 21   |
| V. Embedding and Extending JMathLib .....     | 22   |
| 16. Embedding JMathLib .....                  | 23   |
| 17. Embedding JMathLib in a java applet ..... | 24   |
| 18. Access JMathLib From Perl .....           | 25   |
| 19. Extending JMathLib .....                  | 26   |
| m Files .....                                 | 26   |
| External Functions .....                      | 26   |
| 20. Testing JMathLib .....                    | 29   |
| Testing Introduction .....                    | 29   |
| VI. Functions Reference .....                 | 30   |
| 21. char .....                                | 39   |
| Introduction to the toolbox char .....        | 39   |
| _char .....                                   | 39   |
| _double .....                                 | 39   |
| blanks .....                                  | 40   |
| char .....                                    | 40   |

|                                           |    |
|-------------------------------------------|----|
| deblank .....                             | 41 |
| double .....                              | 41 |
| findstr .....                             | 41 |
| left .....                                | 42 |
| lower .....                               | 42 |
| mid .....                                 | 43 |
| num2str .....                             | 44 |
| right .....                               | 44 |
| sprintf .....                             | 44 |
| str2num .....                             | 45 |
| strcat .....                              | 46 |
| strcmp .....                              | 46 |
| strcmpi .....                             | 47 |
| strfind .....                             | 47 |
| strlength .....                           | 47 |
| strncmp .....                             | 48 |
| strncmpi .....                            | 48 |
| strvcat .....                             | 49 |
| substring .....                           | 49 |
| upper .....                               | 50 |
| 22. control .....                         | 51 |
| Introduction to the control toolbox ..... | 51 |
| __abccdims__ .....                        | 51 |
| abccdim .....                             | 51 |
| axis2dlim .....                           | 52 |
| is_abcd .....                             | 52 |
| is_sample .....                           | 53 |
| ss2sys .....                              | 53 |
| swap .....                                | 54 |
| sysgettsam .....                          | 54 |
| sysgettype .....                          | 55 |
| zgfmul .....                              | 55 |
| 23. crypto .....                          | 56 |
| aes .....                                 | 56 |
| 24. demos .....                           | 57 |
| vdp1 .....                                | 57 |
| vdp2 .....                                | 57 |
| 25. deprecated .....                      | 58 |
| finite .....                              | 58 |
| isstr .....                               | 58 |
| 26. engine .....                          | 60 |
| Introduction to the toolbox audio .....   | 60 |
| engine .....                              | 60 |
| 27. finance .....                         | 61 |
| Introduction to the toolbox audio .....   | 61 |
| fv .....                                  | 61 |
| fv1 .....                                 | 61 |
| nper .....                                | 62 |
| pmt .....                                 | 62 |
| pv .....                                  | 62 |
| pvl .....                                 | 63 |
| vol .....                                 | 63 |
| 28. ftp .....                             | 65 |
| Introduction to the toolbox audio .....   | 65 |
| ftp .....                                 | 65 |
| 29. funfun .....                          | 66 |
| Introduction to the toolbox audio .....   | 66 |
| euler .....                               | 66 |

|                                         |    |
|-----------------------------------------|----|
| feval .....                             | 66 |
| 30. general .....                       | 68 |
| Introduction to the toolbox audio ..... | 68 |
| _class .....                            | 68 |
| angle .....                             | 68 |
| beep .....                              | 69 |
| bench .....                             | 69 |
| bitand .....                            | 70 |
| bitor .....                             | 70 |
| bitshift .....                          | 71 |
| bitxor .....                            | 71 |
| ceil .....                              | 72 |
| cell .....                              | 72 |
| clear .....                             | 73 |
| clock .....                             | 73 |
| combinations .....                      | 74 |
| complex .....                           | 74 |
| conj .....                              | 75 |
| diff .....                              | 75 |
| e .....                                 | 76 |
| eps .....                               | 76 |
| exp .....                               | 77 |
| factor .....                            | 77 |
| false .....                             | 78 |
| fibonacci .....                         | 78 |
| fix .....                               | 79 |
| func2str .....                          | 79 |
| getpfilecaching .....                   | 80 |
| global .....                            | 80 |
| harmonic .....                          | 81 |
| i .....                                 | 81 |
| imag .....                              | 82 |
| int16 .....                             | 83 |
| int32 .....                             | 83 |
| int64 .....                             | 84 |
| int8 .....                              | 84 |
| isa .....                               | 85 |
| iscell .....                            | 85 |
| ischar .....                            | 86 |
| isdouble .....                          | 86 |
| isfunctionhandle .....                  | 87 |
| isglobal .....                          | 87 |
| isint16 .....                           | 88 |
| isint32 .....                           | 88 |
| isint64 .....                           | 89 |
| isint8 .....                            | 90 |
| islogical .....                         | 90 |
| ismatrix .....                          | 91 |
| isnumeric .....                         | 91 |
| isprime .....                           | 92 |
| isscalar .....                          | 92 |
| issingle .....                          | 93 |
| issquare .....                          | 94 |
| isstruct .....                          | 94 |
| isstudent .....                         | 95 |
| isuint16 .....                          | 95 |
| isuint32 .....                          | 96 |
| isuint64 .....                          | 96 |

|                                         |     |
|-----------------------------------------|-----|
| isuint8 .....                           | 97  |
| j .....                                 | 97  |
| kbhit .....                             | 98  |
| length .....                            | 98  |
| linspace .....                          | 99  |
| logical .....                           | 99  |
| ndims .....                             | 100 |
| performfunction .....                   | 101 |
| permutations .....                      | 101 |
| pi .....                                | 101 |
| primes .....                            | 102 |
| rand .....                              | 103 |
| real .....                              | 103 |
| return .....                            | 103 |
| round .....                             | 104 |
| setpfilecaching .....                   | 104 |
| sign .....                              | 105 |
| size .....                              | 106 |
| sqrt .....                              | 106 |
| str2func .....                          | 107 |
| struct .....                            | 107 |
| template .....                          | 108 |
| true .....                              | 108 |
| uint16 .....                            | 109 |
| uint32 .....                            | 109 |
| uint8 .....                             | 110 |
| who .....                               | 110 |
| whos .....                              | 111 |
| 31. graph3d .....                       | 112 |
| surf .....                              | 112 |
| 32. graphics .....                      | 113 |
| cla .....                               | 113 |
| clf .....                               | 113 |
| close .....                             | 114 |
| figure .....                            | 114 |
| gca .....                               | 115 |
| gcf .....                               | 115 |
| get .....                               | 115 |
| ginput .....                            | 116 |
| grid .....                              | 116 |
| hold .....                              | 117 |
| meshgrid .....                          | 117 |
| plot .....                              | 118 |
| plot3 .....                             | 119 |
| plotfunction .....                      | 120 |
| polar .....                             | 121 |
| print .....                             | 122 |
| propertyeditor .....                    | 122 |
| rotate .....                            | 123 |
| set .....                               | 123 |
| subplot .....                           | 124 |
| title .....                             | 124 |
| xlabel .....                            | 125 |
| ylabel .....                            | 125 |
| zlabel .....                            | 126 |
| 33. image .....                         | 127 |
| Introduction to the toolbox audio ..... | 127 |
| autumn .....                            | 127 |

|                                         |     |
|-----------------------------------------|-----|
| bone .....                              | 127 |
| brighten .....                          | 128 |
| colormap .....                          | 128 |
| cool .....                              | 129 |
| copper .....                            | 129 |
| gray .....                              | 130 |
| gray2ind .....                          | 130 |
| hot .....                               | 131 |
| jet .....                               | 131 |
| ntsc2rgb .....                          | 132 |
| pink .....                              | 132 |
| rainbow .....                           | 133 |
| spring .....                            | 133 |
| summer .....                            | 134 |
| white .....                             | 134 |
| winter .....                            | 135 |
| 34. integration .....                   | 136 |
| Introduction to the toolbox audio ..... | 136 |
| crule .....                             | 136 |
| ncrule .....                            | 136 |
| 35. internal .....                      | 138 |
| minusminus .....                        | 138 |
| plusplus .....                          | 138 |
| 36. io .....                            | 140 |
| Introduction to the toolbox audio ..... | 140 |
| cd .....                                | 140 |
| createnewfile .....                     | 140 |
| csvread .....                           | 141 |
| csvwrite .....                          | 141 |
| delete .....                            | 142 |
| dir .....                               | 142 |
| exist .....                             | 143 |
| isdirectory .....                       | 143 |
| isfile .....                            | 144 |
| ishidden .....                          | 144 |
| lastmodified .....                      | 145 |
| load .....                              | 145 |
| loadvariables .....                     | 146 |
| mkdir .....                             | 146 |
| pwd .....                               | 147 |
| rmdir .....                             | 147 |
| runfile .....                           | 148 |
| savevariables .....                     | 149 |
| systemcommand .....                     | 149 |
| 37. jmathlib .....                      | 151 |
| Introduction to the toolbox audio ..... | 151 |
| freemat .....                           | 151 |
| jmathlib .....                          | 151 |
| matlab .....                            | 152 |
| octave .....                            | 152 |
| scilab .....                            | 153 |
| 38. linearalgebra .....                 | 154 |
| Introduction to the toolbox audio ..... | 154 |
| commutation_matrix .....                | 154 |
| dmult .....                             | 154 |
| dot .....                               | 154 |
| duplication_matrix .....                | 155 |
| housh .....                             | 155 |

|                     |     |
|---------------------|-----|
| logm .....          | 156 |
| norm .....          | 156 |
| null .....          | 156 |
| orth .....          | 157 |
| rank .....          | 157 |
| trace .....         | 158 |
| vec .....           | 158 |
| vech .....          | 159 |
| 39. matrix .....    | 160 |
| abs .....           | 160 |
| adjoint .....       | 160 |
| all .....           | 161 |
| and .....           | 161 |
| any .....           | 162 |
| chol .....          | 162 |
| col .....           | 163 |
| columns .....       | 163 |
| ctranspose .....    | 164 |
| cumprod .....       | 164 |
| cumsum .....        | 165 |
| det .....           | 165 |
| determinant .....   | 166 |
| diag .....          | 166 |
| eig .....           | 167 |
| elementat .....     | 167 |
| eq .....            | 168 |
| eye .....           | 168 |
| find .....          | 169 |
| fliplr .....        | 169 |
| flipud .....        | 170 |
| floor .....         | 170 |
| ge .....            | 171 |
| gt .....            | 171 |
| inf .....           | 171 |
| inv .....           | 172 |
| inversematrix ..... | 172 |
| isempty .....       | 173 |
| isfinite .....      | 173 |
| isimaginary .....   | 174 |
| isinf .....         | 174 |
| isnan .....         | 175 |
| isreal .....        | 175 |
| ldivide .....       | 176 |
| le .....            | 176 |
| ln .....            | 177 |
| log .....           | 177 |
| lowertriangle ..... | 178 |
| lt .....            | 178 |
| lu .....            | 179 |
| magic .....         | 179 |
| max .....           | 180 |
| min .....           | 180 |
| minus .....         | 181 |
| mldivide .....      | 181 |
| mpower .....        | 182 |
| mrdivide .....      | 182 |
| mtimes .....        | 183 |
| nan .....           | 183 |



|                                         |     |
|-----------------------------------------|-----|
| ne .....                                | 184 |
| nnz .....                               | 184 |
| not .....                               | 185 |
| numel .....                             | 185 |
| ones .....                              | 185 |
| or .....                                | 186 |
| plus .....                              | 186 |
| pow2 .....                              | 187 |
| power .....                             | 187 |
| prod .....                              | 188 |
| qr .....                                | 188 |
| rdivide .....                           | 189 |
| repmat .....                            | 189 |
| reshape .....                           | 190 |
| row .....                               | 190 |
| rows .....                              | 191 |
| simultaneouseq .....                    | 191 |
| sort .....                              | 192 |
| subassign .....                         | 192 |
| submatrix .....                         | 193 |
| sum .....                               | 193 |
| sumsq .....                             | 194 |
| svd .....                               | 194 |
| times .....                             | 195 |
| transpose .....                         | 196 |
| uminus .....                            | 196 |
| uplus .....                             | 197 |
| uppertriangle .....                     | 197 |
| xor .....                               | 197 |
| zeros .....                             | 198 |
| 40. miscellaneous .....                 | 199 |
| Introduction to the toolbox audio ..... | 199 |
| ans .....                               | 199 |
| comma .....                             | 199 |
| flops .....                             | 199 |
| mexext .....                            | 200 |
| semicolon .....                         | 200 |
| single .....                            | 201 |
| texas_lotto .....                       | 201 |
| 41. net .....                           | 203 |
| Introduction to the toolbox audio ..... | 203 |
| urlread .....                           | 203 |
| 42. polynomial .....                    | 204 |
| Introduction to the toolbox audio ..... | 204 |
| binomial .....                          | 204 |
| compan .....                            | 204 |
| mkpp .....                              | 205 |
| poly .....                              | 205 |
| polyinteg .....                         | 205 |
| polyreduce .....                        | 206 |
| polyval .....                           | 206 |
| roots .....                             | 207 |
| unmkpp .....                            | 207 |
| 43. quaternion .....                    | 208 |
| qtransvmat .....                        | 208 |
| 44. set .....                           | 209 |
| Introduction to the toolbox audio ..... | 209 |
| complement .....                        | 209 |

|                                         |     |
|-----------------------------------------|-----|
| create_set .....                        | 209 |
| 45. signal .....                        | 210 |
| Introduction to the toolbox audio ..... | 210 |
| bartlett .....                          | 210 |
| blackman .....                          | 210 |
| durbinlevinson .....                    | 211 |
| hamming .....                           | 211 |
| hanning .....                           | 212 |
| hurst .....                             | 212 |
| rectangle_lw .....                      | 213 |
| rectangle_sw .....                      | 213 |
| sinc .....                              | 214 |
| sinewave .....                          | 215 |
| triangle_lw .....                       | 215 |
| triangle_sw .....                       | 216 |
| 46. specfun .....                       | 217 |
| Introduction to the toolbox audio ..... | 217 |
| beta .....                              | 217 |
| betaln .....                            | 217 |
| gammaln .....                           | 218 |
| perms .....                             | 218 |
| 47. specialmatrix .....                 | 220 |
| Introduction to the toolbox audio ..... | 220 |
| hankel .....                            | 220 |
| hilb .....                              | 220 |
| lauchli .....                           | 220 |
| pascal .....                            | 221 |
| rosser .....                            | 221 |
| sylvestermatrix .....                   | 222 |
| toeplitz .....                          | 222 |
| wilkinson .....                         | 222 |
| 48. statistics .....                    | 224 |
| Introduction to the toolbox audio ..... | 224 |
| average .....                           | 224 |
| center .....                            | 224 |
| cloglog .....                           | 225 |
| cov .....                               | 225 |
| mean .....                              | 226 |
| meansq .....                            | 226 |
| std .....                               | 227 |
| var .....                               | 227 |
| variation .....                         | 228 |
| 49. system .....                        | 229 |
| addpath .....                           | 229 |
| checkforupdates .....                   | 229 |
| createfunctionslist .....               | 230 |
| dbquit .....                            | 230 |
| debug .....                             | 231 |
| disp .....                              | 231 |
| error .....                             | 231 |
| exit .....                              | 232 |
| foreach .....                           | 232 |
| format .....                            | 233 |
| getdebug .....                          | 233 |
| getenv .....                            | 234 |
| getjmathlibproperty .....               | 234 |
| getproperty .....                       | 235 |
| jmathlibcreateuniqueid .....            | 235 |

|                                                 |     |
|-------------------------------------------------|-----|
| nargchk .....                                   | 236 |
| nargoutchk .....                                | 236 |
| newline .....                                   | 237 |
| path .....                                      | 237 |
| print_usage .....                               | 238 |
| printstacktrace .....                           | 238 |
| quit .....                                      | 238 |
| rehash .....                                    | 239 |
| rmpath .....                                    | 239 |
| setdebug .....                                  | 240 |
| setjmathlibproperty .....                       | 241 |
| update .....                                    | 241 |
| usage .....                                     | 242 |
| ver .....                                       | 242 |
| version .....                                   | 243 |
| warning .....                                   | 243 |
| 50. test .....                                  | 245 |
| test_complex .....                              | 245 |
| test_for .....                                  | 245 |
| test_graph .....                                | 245 |
| test_matlabfor .....                            | 246 |
| test_standard .....                             | 246 |
| test_switch .....                               | 247 |
| test_trig .....                                 | 247 |
| testfunctionfor001 .....                        | 247 |
| testfunctionfor002 .....                        | 248 |
| testfunctionswitch002 .....                     | 248 |
| testfunctionwhile001 .....                      | 249 |
| 51. time .....                                  | 250 |
| Introduction to the toolbox audio .....         | 250 |
| date .....                                      | 250 |
| is_leap_year .....                              | 250 |
| pause .....                                     | 251 |
| tic .....                                       | 251 |
| time .....                                      | 252 |
| toc .....                                       | 252 |
| 52. trigonometric .....                         | 254 |
| Introduction to the toolbox trigonometric ..... | 254 |
| acos .....                                      | 254 |
| acosd .....                                     | 254 |
| acosh .....                                     | 255 |
| acot .....                                      | 255 |
| acotd .....                                     | 256 |
| acoth .....                                     | 256 |
| acsc .....                                      | 257 |
| acscd .....                                     | 257 |
| acsch .....                                     | 258 |
| asec .....                                      | 258 |
| asecd .....                                     | 259 |
| asech .....                                     | 259 |
| asin .....                                      | 260 |
| asind .....                                     | 260 |
| asinh .....                                     | 261 |
| atan .....                                      | 261 |
| atan2 .....                                     | 262 |
| atand .....                                     | 262 |
| atanh .....                                     | 263 |
| cos .....                                       | 263 |

|                                                                                           |     |
|-------------------------------------------------------------------------------------------|-----|
| cosd .....                                                                                | 264 |
| cosh .....                                                                                | 264 |
| cot .....                                                                                 | 265 |
| cotd .....                                                                                | 265 |
| coth .....                                                                                | 266 |
| csc .....                                                                                 | 266 |
| cscd .....                                                                                | 267 |
| csch .....                                                                                | 267 |
| degtograd .....                                                                           | 268 |
| degtorad .....                                                                            | 268 |
| gradtodeg .....                                                                           | 268 |
| gradtorad .....                                                                           | 269 |
| radtodeg .....                                                                            | 269 |
| radtograd .....                                                                           | 270 |
| sec .....                                                                                 | 270 |
| secd .....                                                                                | 271 |
| sech .....                                                                                | 271 |
| sin .....                                                                                 | 272 |
| sind .....                                                                                | 272 |
| sinh .....                                                                                | 273 |
| tan .....                                                                                 | 273 |
| tand .....                                                                                | 274 |
| tanh .....                                                                                | 274 |
| 53. ui .....                                                                              | 276 |
| questdlg .....                                                                            | 276 |
| A. GNU Lesser General Public License - GNU Project - Free Software Foundation (FSF) ..... | 277 |
| GNU Lesser General Public License .....                                                   | 277 |
| B. Frequently Asked Questions .....                                                       | 285 |
| C. History .....                                                                          | 287 |
| History .....                                                                             | 287 |
| Old versions .....                                                                        | 287 |
| D. To Do .....                                                                            | 289 |
| Open .....                                                                                | 289 |
| Closed .....                                                                              | 289 |
| Index .....                                                                               | 290 |
| Glossary .....                                                                            | 295 |

---

# Foreword

This is a the main documentation for the JMathLib Program.

## Work in progress

Like the JMathLib program this is a work in progress. All efforts will be made to keep the documentation in sync with the main program but it might not be completely up to date.

In mid of 2001 I was looking for a computational engine which has an easy scripting language and which should be very flexible. Also I did have no other projects to spend my leisure time on, the last project was finished and at work was no access to a flexible simulation tool like Matlab [<http://www.mathworks.com/>]. A search in the open source world on the internet came up with a couple of hits like octave, scilab, rlab and some more. Some were already very advanced, but they are all written in C or C++ and I was looking for something in my favourite programming language, which is java. So finally I found JMathLib [<http://mathlib.sourceforge.net>], which is hosted at sourceforge.net. The project had just started some months ago and only had one developer so far. This is when I met Mark Sparshatt and very soon I became an active member of the developing team. Which had the magical size of two individuals at that time.

Some of my other projects can be found here [<http://www.held-mueller.de/>].

Hallo

you

Stefan Müller (2006/10/23)

---

# Part I. Installing And Running

## Table of Contents

|                                 |   |
|---------------------------------|---|
| 1. Installing And Running ..... | 2 |
| Obtaining JMathLib .....        | 2 |
| Installing JMathLib .....       | 2 |
| Running JMathLib .....          | 2 |

---

# Chapter 1. Installing And Running

## Obtaining JMathLib

The latest release of JMathLib can be obtained from the Sourceforge site.

Checkout the website at JMathLib [<http://mathlib.sourceforge.net>]

and click on Download [<http://mathlib.sourceforge.net/download.php>]

Also the latest development version can be downloaded using anonymous cvs

use the following commands

```
cvs -d:pserver:anonymous@cvs.mathlib.sourceforge.net:/cvsroot/mathlib login
```

```
cvs -z8 -d:pserver:anonymous@cvs.mathlib.sourceforge.net:/cvsroot/mathlib co mathlib
```

## Installing JMathLib

JMathLib has a couple of ways to install: there is a windows installer, there is the option to download the source, there is the option to download the class files, there is the option to run a java applet.

## Installing JMathLib on Windows systems

## Installation on other systems

To install JMathLib you should just need to uncompress the downloaded file.

## Running JMathLib

## Running JMathLib on Windows

For the Windows version just doubleclick on the JMathLib icon

run JMathLib:

from the JMathLib main directory enter

```
java -cp bin jmathlib.ui.awt.GUI
```

for the simple AWT user interface, or enter

```
java -cp bin jmathlib.ui.swing.SwingGUI
```

for the more sophisticated swing version.

Just be sure that you are in the main directory of JMathLib. That is the directory with the subdirectories (e.g. bin, doc) in it.

There are script files to run MathLib from Windows or Linux

just enter **execute.bat**

options:

-t run in console mode

-width specify the width of the gui window

-height specify the height of the gui window

-width and -height cannot be used with -t

## Running JMathLib on Linux

HERE I NEED SOME SUPPORT FROM THE USERS

There are script files to run MathLib from Linux

just enter `execute.sh`



---

# Part II. Usage

## Table of Contents

|                                |    |
|--------------------------------|----|
| 2. Basic Operations .....      | 5  |
| 3. Variables .....             | 6  |
| 4. Complex Numbers .....       | 7  |
| 5. Matrices .....              | 8  |
| 6. Control Structures .....    | 9  |
| If .....                       | 9  |
| Switch .....                   | 9  |
| For .....                      | 10 |
| While .....                    | 10 |
| 7. Symbolic computations ..... | 11 |

---

# Chapter 2. Basic Operations

JMathLib understands the basic operators  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $^$  and  $!$ .

```
> 2+3
5.0
> 5!
120.0
> 2*3
6.0
> 2^3
8.0
> 2/3
0.6666666666666666
```

The operators work in the following order

```
^
* /
+ -
!
```

This can be modified by using brackets

```
> 2+3*4
14.0

> (2+3)*4
20.0

> 2*3^2
18.0

> (2*3)^2
36.0

> 2^3!
40320.0

> 2^(3!)
64.0

> 2+3!
120.0

> 2+(3!)
8.0
```

---

# Chapter 3. Variables

Variables are assigned values by using the assignment operator =

```
> x=5
```

```
5.0
```

```
> x*2
```

```
10.0
```

```
> x^2
```

```
25.0
```

---

# Chapter 4. Complex Numbers

Complex numbers are used with the variables  $j$  or  $i$ . This is defined as being the square root of  $-1$ .

```
> 1+i
(1.0 + i)
> i^2
-1.0
> i^3
-1.0i
> i^4
1.0
> (1+i)*(1-j)
2.0i
```

---

# Chapter 5. Matrices

Matrices are entered by enclosing the values in [ ]. Each item in a row is separated by a , and each row is separated by a ; .

entering [1,2] will produce the 1 by 2 matrix

[1 2]

entering [1,2;3,4] will produce the 2\*2 matrix

[1 2]

[3 4]

entering [1;2] will produce the 2\*1 matrix

[1]

[2]

The identity matrix can be entered using the function `eye` with the size of the matrix as the parameter. A zero filled matrix can be entered using the `zeros` function. This accepts either one parameter to produce a square matrix or two to produce a rectangular matrix. A one filled matrix can be entered using the `ONES` function. This accepts either one parameter to produce a square matrix or two to produce a rectangular matrix.

A one dimensional matrix containing a series of numbers can be created using the colon operator either with the form

start : end

start : increment : end

e.g

2:9 = [2,3,4,5,6,7,8,9]

2:2:8 = [2,4,6,8]

---

# Chapter 6. Control Structures

## If

There are two ways of writing if statements

```
if(test1)
{
    code1;
}
elseif(test2)
{
    code2;
}
else
{
    code3;
};
```

```
if test1
    code1;
elseif test2
    code2;
else
    code3;
end;
```

## Switch

The two ways of writing switch statements are:

```
switch(variable)
{
    case(val1):
    {
        code1;
    }
    case(val2):
    {
        code2;
    }
    default/otherwise:
    {
        code3;
    }
};
```

```
switch(variable)
case val1:
    code1;
case val2:
    code2;
```

```
        default/otherwise:  
            code3;  
end;
```

## For

The two ways of writing the for statement are:

```
for(initialisation, test, increment)  
{  
    code;  
};
```

```
for variable = vector  
    code;  
end;
```

## While

The two ways of writing a while statement are:

```
while(test)  
{  
    code;  
};
```

```
while(test)  
    code;  
end;
```

---

# Chapter 7. Symbolic computations

## **Warning**

Symbolic functions are not supported right now!



---

# Part III. Datatypes

## Table of Contents

|                                     |    |
|-------------------------------------|----|
| 8. Numbers .....                    | 13 |
| Usage .....                         | 13 |
| Internal - NumericToken.class ..... | 13 |
| 9. Matrices .....                   | 14 |
| Usage .....                         | 14 |
| Internal - MatrixToken .....        | 14 |
| 10. Strings .....                   | 15 |
| Usage .....                         | 15 |
| Internal - StringToken .....        | 15 |
| 11. Variables .....                 | 16 |
| Usage .....                         | 16 |
| Internal - VariableToken .....      | 16 |
| 12. Symbolic Variables .....        | 17 |
| .....                               | 17 |
| 13. Structures .....                | 18 |
| Usage .....                         | 18 |
| Internal - MathLibObject .....      | 18 |
| 14. Casting .....                   | 19 |

---

---

# Chapter 8. Numbers

## Usage

These are used to store all numeric values including real and complex scalar values and real and complex matrices.

`1`

`1 + i`

`[1, 2; 3, 4]`

are all numeric values

## Internal - NumericToken.class

The data within the number token is stored as a 3 dimensional array of double values

`double values[y][x][a]`

y stores the rows of the matrix

x stored the coloumns of the matrix

a stored the real and imaginary part

---

# Chapter 9. Matrices

## Usage

Matrix objects are used to store matrices of non numeric values. i.e. variables, symbolic variables, structures and strings.

## Internal - MatrixToken

The data is stored as a 2d array of OperandTokens

OperandToken values[y][x]

---

# Chapter 10. Strings

## Usage

These are used to represent text data. Either "" or " can be used to mark a string

```
"Hello World"
```

```
'Hello World'
```

## Internal - StringTokenizer

String data is stored using a string object.

---

# Chapter 11. Variables

## Usage

Variables are used to store references to other JMathLib datatypes.

```
>x = 5
```

```
5
```

```
>x
```

```
5
```

## Internal - VariableToken

The name of the variable is stored as a string.

---

# Chapter 12. Symbolic Variables

Symbolic variables are not supported in the current versions of JMathLib.

---

# Chapter 13. Structures

## Usage

Structures are used to store structural data.

e.g. `struct(a,1,b,2)` will create a structure with two fields called `a` and `b` which are assigned the values 2 and 4

If the first element is a variable storing a structure then the new structure will contain it's fields.

```
x = struct(a, 1, b, 2)
```

```
a = 1 : b = 2 :
```

```
y = struct(x, c, 3)
```

```
a = 1 : b = 2 : c = 3 :
```

## Internal - MathLibObject

The structured data is stored as an array of `Variable` objects

---

# Chapter 14. Casting

When an expression contains two different data types JMathLib looks for a special class to cast from one type to the other.

It first looks for a class called

`Cast_[from type]_[to type].class`

If it doesn't find that it looks for the class

`Cast_[to type].class`



---

# Part IV. Graphical Subsystem

## Table of Contents

|                                          |    |
|------------------------------------------|----|
| 15. Basics about JMathLib graphics ..... | 21 |
| Basic Command .....                      | 21 |

---

# Chapter 15. Basics about JMathLib graphics

## Basic Command

Some very simple commands are `plot(...)` `plot3(...)` `bar(...)`

---

# Part V. Embedding and Extending JMathLib

## Table of Contents

|                                               |    |
|-----------------------------------------------|----|
| 16. Embedding JMathLib .....                  | 23 |
| 17. Embedding JMathLib in a java applet ..... | 24 |
| 18. Access JMathLib From Perl .....           | 25 |
| 19. Extending JMathLib .....                  | 26 |
| m Files .....                                 | 26 |
| External Functions .....                      | 26 |

---

---

# Chapter 16. Embedding JMathLib

The JMathLib interpreter can be called from other programs which can use it to perform it mathematical calculations

The interpreter classes main function is the `evaluateExpression` function. This takes a string containing the expression and returns a string containing the answer.

Currently JMathLib can only be called from other Java programs. (though see accessing JMathLib from Perl.)

---

# Chapter 17. Embedding JMathLib in a java applet

The following is an example on how to use JMathLib as a java applet. The basic java applet is called using the class `jmathlib.ui.applet.JMathLibGUI.class`.

The following code is an example for usage in mozilla.

```
<applet          CODE      = "jmathlib.ui.applet.JMathLibGUI.class"
  CODEBASE = "bin"      ARCHIVE = "JMathLibSmallApplet.jar"
  WIDTH    = "700"      HEIGHT  = "400"
  ALIGN    = "middle"
  VSPACE   = "0"        HSPACE  = "0"      >
<PARAM NAME="startup"  VALUE  = "plot(rand(2,30))">
<PARAM NAME="fgcolor"  VALUE  = "ff00ff">
<PARAM NAME="bgcolor"  VALUE  = "f64033">
</applet>
```

There are some parameters in order to configure the applet.

Use the parameter `startup` to specify the first command which should be executed after the JMathLib applet has been started. (e.g. `<PARAM NAME="startup" VALUE="plot(rand(2,30))">` will plot a figure with two rows of random numbers.)

Use the parameter `fgcolor` to specify the foreground color (basically the color of the letters) in JMathLib's workspace. The color is coded as `rgb`-values, please use hex-notation (e.g. `2200ff`).

Use the parameter `bgcolor` to specify the background color in JMathLib's workspace. The color is coded as `rgb`-values, please use hex-notation (e.g. `2200ff`).

---

# Chapter 18. Access JMathLib From Perl

The tools/perl/MathLib.pm file allows Perl programs to create objects of the JMathLib program and access its functions. Though with the following caveats.

- 1) The JMathLib module requires the JVM.pm module (available from CPAN).
- 2) This module has so far only been successfully installed under Linux
- 3) Not all of the functionality of JMathLib is available from the module.

---

# Chapter 19. Extending JMathLib

There are two ways to add extra functions to the JMathLib. These are:

- 1) m files
- 2) External Functions

## m Files

M files are script files written in the JMathLib language.

```
function A = perms (v)
    v = v(:);
    n = length (v);
    if (n == 1)
        A = v;
    else
        B = perms (v(1:n-1));
        Bidx = 1:size (B, 1);
        A = v(n) * ones (prod (2:n), n);
        A(Bidx,1:n-1) = B;
        k = size (B, 1);
        for x = n-1:-1:2
            A(k+Bidx,1:x-1) = B(Bidx,1:x-1);
            A(k+Bidx,x+1:n) = B(Bidx,x:n-1);
            k = k + size (B, 1);
        endfor
        A(k+Bidx,2:n) = B;
    endif
endfunction
```

## External Functions

These are java class files which descend from the Functions/ExternalFunction class file. They must override the evaluate function.

```
package MathLib.Functions.General;

import MathLib.Tokens.*;
import MathLib.Functions.ExternalFunction;

/**An external function for computing a mesh of a matrix */
public class template extends ExternalFunction
{
    /**returns two matrices
     * @param operands[0] = x values (e.g. [-2:0.2:2])
     * @param operands[1] = y values (e.g. [-2:0.2:2])
     * @return [X,Y] as matrices */
    public OperandToken evaluate(Token[] operands)
    {
        // one operand (e.g. [x,y]=template([-2:0.2:2],[-2:0.2:2]) )
        if (getNArgIn(operands)!=2)
```

```

        throwMathLibException("template: number of input arguments != 2");

// Check number of return arguments
if (getNoOfLeftHandArguments()!=2)
    throwMathLibException("template: number of output arguments != 2");

if ( !(operands[0] instanceof NumberToken) ||
    !(operands[0] instanceof NumberToken) )
    throwMathLibException("template: works only on numbers");

// get data from arguments
double[][] x = ((NumberToken)operands[0]).getValues();
double[][] y = ((NumberToken)operands[1]).getValues();

if ((x.length != 1) ||
    (y.length != 1) )
    throwMathLibException("template: works only row vectors");

int sizeX = x[0].length;
int sizeY = y[0].length;

double[][] X = new double[sizeY][sizeX];
double[][] Y = new double[sizeY][sizeX];

for (int i=0; i<sizeY; i++)
{
    for (int j=0; j<sizeX; j++)
    {
        X[i][j] = x[0][j];
        Y[i][j] = y[0][i];
    }
}

OperandToken values[][] = new OperandToken[1][2];
values[0][0] = new NumberToken(X);
values[0][1] = new NumberToken(Y);
return new MatrixToken( values );

    } // end eval
}

/*
@GROUP
General
@SYNTAX
answer = template (value)
@DOC
Returns the sign of value.
@EXAMPLES
SIGN(-10)=-1
SIGN(10)=1
@NOTES
@SEE
template
*/

public int getNoOfLeftHandArguments()

```



```
protected int getNArgOut()
protected int getNArgIn(Token[] operands)
public void setNoOfLeftHandArguments(int _number)

public void throwMathLibException(String errorMessage)
public void debugLine(String s)
```

```
RootObject.java
protected final VariableList getVariables()
protected final VariableList getGlobalVariables()
protected final Variable getVariable(String name)
protected final ContextList getContextList()
protected final Interpreter getInterpreter()
public final FunctionManager getFunctionManager()
protected jmathlib.plugins.PluginsManager getPluginsManager()
protected File getWorkingDirectory()
protected void setWorkingDirectory(File _workingDir)
protected jmathlib.core.graphics.GraphicsManager getGraphicsManager()
```

---

# Chapter 20. Testing JMathLib

## Testing Introduction

JMathLib also has a testing engine which is used to support quality issues. A central goal of JMathLib is to test all functions automatically with a regression test framework. Currently we use JUnit [<http://www.junit.org>] to improve code quality.

There is also a mechanism to attach testing code to the standard user functions so that code and test-code can be kept in the same file. These tests are extracted from the general source, individual test case are going to be generated automatically.

---

# Part VI. Functions Reference

## Table of Contents

|                                           |    |
|-------------------------------------------|----|
| 21. char .....                            | 39 |
| Introduction to the toolbox char .....    | 39 |
| _char .....                               | 39 |
| _double .....                             | 39 |
| blanks .....                              | 40 |
| char .....                                | 40 |
| deblank .....                             | 41 |
| double .....                              | 41 |
| findstr .....                             | 41 |
| left .....                                | 42 |
| lower .....                               | 42 |
| mid .....                                 | 43 |
| num2str .....                             | 44 |
| right .....                               | 44 |
| sprintf .....                             | 44 |
| str2num .....                             | 45 |
| strcat .....                              | 46 |
| strcmp .....                              | 46 |
| strcmpi .....                             | 47 |
| strfind .....                             | 47 |
| strlength .....                           | 47 |
| strncmp .....                             | 48 |
| strncmpi .....                            | 48 |
| strvcat .....                             | 49 |
| substring .....                           | 49 |
| upper .....                               | 50 |
| 22. control .....                         | 51 |
| Introduction to the control toolbox ..... | 51 |
| __abctd dims__ .....                      | 51 |
| abctd dim .....                           | 51 |
| axis2dlim .....                           | 52 |
| is_abcd .....                             | 52 |
| is_sample .....                           | 53 |
| ss2sys .....                              | 53 |
| swap .....                                | 54 |
| sysgettsam .....                          | 54 |
| sysgettype .....                          | 55 |
| zgfmul .....                              | 55 |
| 23. crypto .....                          | 56 |
| aes .....                                 | 56 |
| 24. demos .....                           | 57 |
| vdp1 .....                                | 57 |
| vdp2 .....                                | 57 |
| 25. deprecated .....                      | 58 |
| finite .....                              | 58 |
| isstr .....                               | 58 |
| 26. engine .....                          | 60 |
| Introduction to the toolbox audio .....   | 60 |
| engine .....                              | 60 |
| 27. finance .....                         | 61 |
| Introduction to the toolbox audio .....   | 61 |

---

---

|                                         |    |
|-----------------------------------------|----|
| fv .....                                | 61 |
| fv1 .....                               | 61 |
| nper .....                              | 62 |
| pmt .....                               | 62 |
| pvt .....                               | 62 |
| pvl .....                               | 63 |
| vol .....                               | 63 |
| 28. ftp .....                           | 65 |
| Introduction to the toolbox audio ..... | 65 |
| ftp .....                               | 65 |
| 29. funfun .....                        | 66 |
| Introduction to the toolbox audio ..... | 66 |
| euler .....                             | 66 |
| feval .....                             | 66 |
| 30. general .....                       | 68 |
| Introduction to the toolbox audio ..... | 68 |
| _class .....                            | 68 |
| angle .....                             | 68 |
| beep .....                              | 69 |
| bench .....                             | 69 |
| bitand .....                            | 70 |
| bitor .....                             | 70 |
| bitshift .....                          | 71 |
| bitxor .....                            | 71 |
| ceil .....                              | 72 |
| cell .....                              | 72 |
| clear .....                             | 73 |
| clock .....                             | 73 |
| combinations .....                      | 74 |
| complex .....                           | 74 |
| conj .....                              | 75 |
| diff .....                              | 75 |
| e .....                                 | 76 |
| eps .....                               | 76 |
| exp .....                               | 77 |
| factor .....                            | 77 |
| false .....                             | 78 |
| fibonacci .....                         | 78 |
| fix .....                               | 79 |
| func2str .....                          | 79 |
| getpfilecaching .....                   | 80 |
| global .....                            | 80 |
| harmonic .....                          | 81 |
| i .....                                 | 81 |
| imag .....                              | 82 |
| int16 .....                             | 83 |
| int32 .....                             | 83 |
| int64 .....                             | 84 |
| int8 .....                              | 84 |
| isa .....                               | 85 |
| iscell .....                            | 85 |
| ischar .....                            | 86 |
| isdouble .....                          | 86 |
| isfunctionhandle .....                  | 87 |
| isglobal .....                          | 87 |
| isint16 .....                           | 88 |
| isint32 .....                           | 88 |
| isint64 .....                           | 89 |

---

---

|                       |     |
|-----------------------|-----|
| isint8 .....          | 90  |
| islogical .....       | 90  |
| ismatrix .....        | 91  |
| isnumeric .....       | 91  |
| isprime .....         | 92  |
| isscalar .....        | 92  |
| issingle .....        | 93  |
| issquare .....        | 94  |
| isstruct .....        | 94  |
| isstudent .....       | 95  |
| isuint16 .....        | 95  |
| isuint32 .....        | 96  |
| isuint64 .....        | 96  |
| isuint8 .....         | 97  |
| j .....               | 97  |
| kbhit .....           | 98  |
| length .....          | 98  |
| linspace .....        | 99  |
| logical .....         | 99  |
| ndims .....           | 100 |
| performfunction ..... | 101 |
| permutations .....    | 101 |
| pi .....              | 101 |
| primes .....          | 102 |
| rand .....            | 103 |
| real .....            | 103 |
| return .....          | 103 |
| round .....           | 104 |
| setpfilecaching ..... | 104 |
| sign .....            | 105 |
| size .....            | 106 |
| sqrt .....            | 106 |
| str2func .....        | 107 |
| struct .....          | 107 |
| template .....        | 108 |
| true .....            | 108 |
| uint16 .....          | 109 |
| uint32 .....          | 109 |
| uint8 .....           | 110 |
| who .....             | 110 |
| whos .....            | 111 |
| 31. graph3d .....     | 112 |
| surf .....            | 112 |
| 32. graphics .....    | 113 |
| cla .....             | 113 |
| clf .....             | 113 |
| close .....           | 114 |
| figure .....          | 114 |
| gca .....             | 115 |
| gcf .....             | 115 |
| get .....             | 115 |
| ginput .....          | 116 |
| grid .....            | 116 |
| hold .....            | 117 |
| meshgrid .....        | 117 |
| plot .....            | 118 |
| plot3 .....           | 119 |
| plotfunction .....    | 120 |

---

---

|                                         |     |
|-----------------------------------------|-----|
| polar .....                             | 121 |
| print .....                             | 122 |
| propertyeditor .....                    | 122 |
| rotate .....                            | 123 |
| set .....                               | 123 |
| subplot .....                           | 124 |
| title .....                             | 124 |
| xlabel .....                            | 125 |
| ylabel .....                            | 125 |
| zlabel .....                            | 126 |
| 33. image .....                         | 127 |
| Introduction to the toolbox audio ..... | 127 |
| autumn .....                            | 127 |
| bone .....                              | 127 |
| brighten .....                          | 128 |
| colormap .....                          | 128 |
| cool .....                              | 129 |
| copper .....                            | 129 |
| gray .....                              | 130 |
| gray2ind .....                          | 130 |
| hot .....                               | 131 |
| jet .....                               | 131 |
| ntsc2rgb .....                          | 132 |
| pink .....                              | 132 |
| rainbow .....                           | 133 |
| spring .....                            | 133 |
| summer .....                            | 134 |
| white .....                             | 134 |
| winter .....                            | 135 |
| 34. integration .....                   | 136 |
| Introduction to the toolbox audio ..... | 136 |
| crule .....                             | 136 |
| ncrule .....                            | 136 |
| 35. internal .....                      | 138 |
| minusminus .....                        | 138 |
| plusplus .....                          | 138 |
| 36. io .....                            | 140 |
| Introduction to the toolbox audio ..... | 140 |
| cd .....                                | 140 |
| createnewfile .....                     | 140 |
| csvread .....                           | 141 |
| csvwrite .....                          | 141 |
| delete .....                            | 142 |
| dir .....                               | 142 |
| exist .....                             | 143 |
| isdirectory .....                       | 143 |
| isfile .....                            | 144 |
| ishidden .....                          | 144 |
| lastmodified .....                      | 145 |
| load .....                              | 145 |
| loadvariables .....                     | 146 |
| mkdir .....                             | 146 |
| pwd .....                               | 147 |
| rmdir .....                             | 147 |
| runfile .....                           | 148 |
| savevariables .....                     | 149 |
| systemcommand .....                     | 149 |
| 37. jmathlib .....                      | 151 |

---

---

|                                         |     |
|-----------------------------------------|-----|
| Introduction to the toolbox audio ..... | 151 |
| freemat .....                           | 151 |
| jmathlib .....                          | 151 |
| matlab .....                            | 152 |
| octave .....                            | 152 |
| scilab .....                            | 153 |
| 38. linearalgebra .....                 | 154 |
| Introduction to the toolbox audio ..... | 154 |
| commutation_matrix .....                | 154 |
| dmult .....                             | 154 |
| dot .....                               | 154 |
| duplication_matrix .....                | 155 |
| housh .....                             | 155 |
| logm .....                              | 156 |
| norm .....                              | 156 |
| null .....                              | 156 |
| orth .....                              | 157 |
| rank .....                              | 157 |
| trace .....                             | 158 |
| vec .....                               | 158 |
| vech .....                              | 159 |
| 39. matrix .....                        | 160 |
| abs .....                               | 160 |
| adjoint .....                           | 160 |
| all .....                               | 161 |
| and .....                               | 161 |
| any .....                               | 162 |
| chol .....                              | 162 |
| col .....                               | 163 |
| columns .....                           | 163 |
| ctranspose .....                        | 164 |
| cumprod .....                           | 164 |
| cumsum .....                            | 165 |
| det .....                               | 165 |
| determinant .....                       | 166 |
| diag .....                              | 166 |
| eig .....                               | 167 |
| elementat .....                         | 167 |
| eq .....                                | 168 |
| eye .....                               | 168 |
| find .....                              | 169 |
| fliplr .....                            | 169 |
| flipud .....                            | 170 |
| floor .....                             | 170 |
| ge .....                                | 171 |
| gt .....                                | 171 |
| inf .....                               | 171 |
| inv .....                               | 172 |
| inversematrix .....                     | 172 |
| isempty .....                           | 173 |
| isfinite .....                          | 173 |
| isimaginary .....                       | 174 |
| isinf .....                             | 174 |
| isnan .....                             | 175 |
| isreal .....                            | 175 |
| ldivide .....                           | 176 |
| le .....                                | 176 |
| ln .....                                | 177 |

---

---

|                                         |     |
|-----------------------------------------|-----|
| log .....                               | 177 |
| lowertriangle .....                     | 178 |
| lt .....                                | 178 |
| lu .....                                | 179 |
| magic .....                             | 179 |
| max .....                               | 180 |
| min .....                               | 180 |
| minus .....                             | 181 |
| mldivide .....                          | 181 |
| mpower .....                            | 182 |
| mrdivide .....                          | 182 |
| mtimes .....                            | 183 |
| nan .....                               | 183 |
| ne .....                                | 184 |
| nnz .....                               | 184 |
| not .....                               | 185 |
| numel .....                             | 185 |
| ones .....                              | 185 |
| or .....                                | 186 |
| plus .....                              | 186 |
| pow2 .....                              | 187 |
| power .....                             | 187 |
| prod .....                              | 188 |
| qr .....                                | 188 |
| rdivide .....                           | 189 |
| repmat .....                            | 189 |
| reshape .....                           | 190 |
| row .....                               | 190 |
| rows .....                              | 191 |
| simultaneouseq .....                    | 191 |
| sort .....                              | 192 |
| subassign .....                         | 192 |
| submatrix .....                         | 193 |
| sum .....                               | 193 |
| sumsq .....                             | 194 |
| svd .....                               | 194 |
| times .....                             | 195 |
| transpose .....                         | 196 |
| uminus .....                            | 196 |
| uplus .....                             | 197 |
| uppertriangle .....                     | 197 |
| xor .....                               | 197 |
| zeros .....                             | 198 |
| 40. miscellaneous .....                 | 199 |
| Introduction to the toolbox audio ..... | 199 |
| ans .....                               | 199 |
| comma .....                             | 199 |
| flops .....                             | 199 |
| mexext .....                            | 200 |
| semicolon .....                         | 200 |
| single .....                            | 201 |
| texas_lotto .....                       | 201 |
| 41. net .....                           | 203 |
| Introduction to the toolbox audio ..... | 203 |
| urlread .....                           | 203 |
| 42. polynomial .....                    | 204 |
| Introduction to the toolbox audio ..... | 204 |
| binomial .....                          | 204 |

---



---

|                                         |     |
|-----------------------------------------|-----|
| companion .....                         | 204 |
| mkpp .....                              | 205 |
| poly .....                              | 205 |
| polyinteg .....                         | 205 |
| polyreduce .....                        | 206 |
| polyval .....                           | 206 |
| roots .....                             | 207 |
| unmkpp .....                            | 207 |
| 43. quaternion .....                    | 208 |
| qtransvmat .....                        | 208 |
| 44. set .....                           | 209 |
| Introduction to the toolbox audio ..... | 209 |
| complement .....                        | 209 |
| create_set .....                        | 209 |
| 45. signal .....                        | 210 |
| Introduction to the toolbox audio ..... | 210 |
| bartlett .....                          | 210 |
| blackman .....                          | 210 |
| durbinlevinson .....                    | 211 |
| hamming .....                           | 211 |
| hanning .....                           | 212 |
| hurst .....                             | 212 |
| rectangle_lw .....                      | 213 |
| rectangle_sw .....                      | 213 |
| sinc .....                              | 214 |
| sinewave .....                          | 215 |
| triangle_lw .....                       | 215 |
| triangle_sw .....                       | 216 |
| 46. specfun .....                       | 217 |
| Introduction to the toolbox audio ..... | 217 |
| beta .....                              | 217 |
| betaln .....                            | 217 |
| gammaln .....                           | 218 |
| perms .....                             | 218 |
| 47. specialmatrix .....                 | 220 |
| Introduction to the toolbox audio ..... | 220 |
| hankel .....                            | 220 |
| hilb .....                              | 220 |
| lauchli .....                           | 220 |
| pascal .....                            | 221 |
| rosser .....                            | 221 |
| sylvestermatrix .....                   | 222 |
| toeplitz .....                          | 222 |
| wilkinson .....                         | 222 |
| 48. statistics .....                    | 224 |
| Introduction to the toolbox audio ..... | 224 |
| average .....                           | 224 |
| center .....                            | 224 |
| cloglog .....                           | 225 |
| cov .....                               | 225 |
| mean .....                              | 226 |
| meansq .....                            | 226 |
| std .....                               | 227 |
| var .....                               | 227 |
| variation .....                         | 228 |
| 49. system .....                        | 229 |
| addpath .....                           | 229 |
| checkforupdates .....                   | 229 |

---

---

|                                                 |     |
|-------------------------------------------------|-----|
| createfunctionslist .....                       | 230 |
| dbquit .....                                    | 230 |
| debug .....                                     | 231 |
| disp .....                                      | 231 |
| error .....                                     | 231 |
| exit .....                                      | 232 |
| foreach .....                                   | 232 |
| format .....                                    | 233 |
| getdebug .....                                  | 233 |
| getenv .....                                    | 234 |
| getjmathlibproperty .....                       | 234 |
| getproperty .....                               | 235 |
| jmathlibcreateuniqueid .....                    | 235 |
| nargchk .....                                   | 236 |
| nargoutchk .....                                | 236 |
| newline .....                                   | 237 |
| path .....                                      | 237 |
| print_usage .....                               | 238 |
| printstacktrace .....                           | 238 |
| quit .....                                      | 238 |
| rehash .....                                    | 239 |
| rmpath .....                                    | 239 |
| setdebug .....                                  | 240 |
| setjmathlibproperty .....                       | 241 |
| update .....                                    | 241 |
| usage .....                                     | 242 |
| ver .....                                       | 242 |
| version .....                                   | 243 |
| warning .....                                   | 243 |
| 50. test .....                                  | 245 |
| test_complex .....                              | 245 |
| test_for .....                                  | 245 |
| test_graph .....                                | 245 |
| test_matlabfor .....                            | 246 |
| test_standard .....                             | 246 |
| test_switch .....                               | 247 |
| test_trig .....                                 | 247 |
| testfunctionfor001 .....                        | 247 |
| testfunctionfor002 .....                        | 248 |
| testfunctionswitch002 .....                     | 248 |
| testfunctionwhile001 .....                      | 249 |
| 51. time .....                                  | 250 |
| Introduction to the toolbox audio .....         | 250 |
| date .....                                      | 250 |
| is_leap_year .....                              | 250 |
| pause .....                                     | 251 |
| tic .....                                       | 251 |
| time .....                                      | 252 |
| toc .....                                       | 252 |
| 52. trigonometric .....                         | 254 |
| Introduction to the toolbox trigonometric ..... | 254 |
| acos .....                                      | 254 |
| acosd .....                                     | 254 |
| acosh .....                                     | 255 |
| acot .....                                      | 255 |
| acotd .....                                     | 256 |
| acoth .....                                     | 256 |
| acsc .....                                      | 257 |

---

---

|                 |     |
|-----------------|-----|
| acscd .....     | 257 |
| acsch .....     | 258 |
| asec .....      | 258 |
| asecd .....     | 259 |
| asech .....     | 259 |
| asin .....      | 260 |
| asind .....     | 260 |
| asinh .....     | 261 |
| atan .....      | 261 |
| atan2 .....     | 262 |
| atand .....     | 262 |
| atanh .....     | 263 |
| cos .....       | 263 |
| cosd .....      | 264 |
| cosh .....      | 264 |
| cot .....       | 265 |
| cotd .....      | 265 |
| coth .....      | 266 |
| csc .....       | 266 |
| cscd .....      | 267 |
| csch .....      | 267 |
| degtograd ..... | 268 |
| degtorad .....  | 268 |
| gradtodeg ..... | 268 |
| gradtorad ..... | 269 |
| radtodeg .....  | 269 |
| radtograd ..... | 270 |
| sec .....       | 270 |
| secd .....      | 271 |
| sech .....      | 271 |
| sin .....       | 272 |
| sind .....      | 272 |
| sinh .....      | 273 |
| tan .....       | 273 |
| tand .....      | 274 |
| tanh .....      | 274 |
| 53. ui .....    | 276 |
| questdlg .....  | 276 |

---

---

# Chapter 21. char

## Introduction to the toolbox char

This toolbox is intended to provide functions in order to manipulate char arrays and strings.

### **`_char`**

**Type: External**

**Group: char**

### **Syntax**

```
string = _char(number)
```

### **Description**

Converts a number to a string.

### **Examples**

```
_char([104, 101, 108, 108, 111, 32, 49, 50]) = "hello 12"
```

### **See Also**

`double`

### **`_double`**

**Type: External**

**Group: char**

### **Syntax**

```
number = str2num( string )
```

### **Description**

Convert strings into numbers

## Notes

## Examples

`str2num("hello 12")` returns `[104, 101, 108, 108, 111, 32, 49, 50]`

## See Also

`num2str`, `char`

## blanks

### Type: External

### Group: char

## Syntax

`BLANKS(number)`

## Description

Outputs a number of spaces equal to number.

## Examples

`"-" + BLANKS(5) + "-" = "- -"`

## See Also

`deblank`

## char

### Type: M-File

### Group: char

## Syntax

`char(number)`

## Description

external function for changing numbers into strings

## See Also

blanks, deblank

# deblank

**Type: M-File**

**Group: char**

## Syntax

deblank

## Description

Outputs a number of spaces equal to number.

## See Also

blanks

# double

**Type: M-File**

**Group: char**

## Syntax

double(char)

## Description

external function for changing strings into numbers

## See Also

blanks, deblank, char

# findstr

**Type: External**

## Group: char

### Syntax

```
findstr(string1, string2)
```

### Description

Finds all occurrences of the shorter string within the longer

### Notes

### Examples

```
findstr("this is a test string", "is") = [3, 6]  
findstr("is", "this is a test string") = [3, 6]
```

### See Also

strfind

## left

### Type: Internal

## Group: char

### Syntax

```
left(string, no. of characters)
```

### Description

Returns the left most characters from string.

### Examples

```
left("Hello world", 5) = world
```

### See Also

mid, right

## lower

## Type: External

## Group: char

## Syntax

```
answer=FindSTR(string1, string2)
```

## Description

Finds all occurrences of the shorter string within the longer

## Notes

## Examples

## See Also

strfind, upper

## mid

## Type: Internal

## Group: char

## Syntax

```
mid(string, leftm right)
```

## Description

returns the part of string between left and right.

## Examples

```
mid("Hello world", 3, 6) = "llo"
```

## See Also

left, right



# num2str

**Type: External**

**Group: char**

## Syntax

```
string = num2str(number)
```

## Description

Converts a number to a string.

## Examples

```
num2str([104, 101]) = ["104" "101"]
```

## See Also

str2num

# right

**Type: Internal**

**Group: char**

## Syntax

```
right(string, no. of characters)
```

## Description

Returns the right most characters from string.

## Examples

```
right("Hello world", 5) = world
```

## See Also

left, mid

# sprintf

## Type: External

## Group: char

## Syntax

```
number = sprintf( formatString, arg0, arg1, arg2, ... )
```

## Description

Convert strings into numbers

## Notes

.

## Examples

```
str2num("hello 12") returns [104, 101, 108, 108, 111, 32, 49, 50]
```

## See Also

num2str

## str2num

## Type: External

## Group: char

## Syntax

```
number = str2num( string )
```

## Description

Convert strings into numbers

## Notes

.

## Examples

```
str2num("hello 12") returns [104, 101, 108, 108, 111, 32, 49, 50]
```

## See Also

num2str

## strcat

**Type: External**

**Group: char**

## Syntax

```
STRCAT(string1, string2, ...)
```

## Description

Concatenates a set of strings together.

## Examples

```
STRCAT("Hello", "World") HelloWorld
```

## See Also

strvcat

## strcmp

**Type: External**

**Group: char**

## Syntax

```
STRCMP(string1, string2)
```

## Description

Compares string1 to string2.

## Examples

```
STRCMP("ABcd", "abce") 0 STRCMP("ABc", "abc") 0 STRCMP("abc", "abc") 1
```

## See Also

strcmpi, strncmpi, strncmp

# strcmpi

**Type:** External

**Group:** char

## Syntax

```
STRCMPi(string1, string2)
```

## Description

Compares string1 to string2, ignoring case.

## Examples

```
STRCMPi("ABcd", "abce") 0 STRCMPi("ABc", "abc") 1
```

## See Also

strcmp, strncmpi, strncmp

# strfind

**Type:** External

**Group:** char

## Syntax

```
STRFIND(string1, string2)
```

## Description

finds all occurrences of string2 within string1.

## Examples

```
STRFIND("This is a test string", "is") [3,6]
```

## See Also

findstr

# strlen

**Type: External**

**Group: char**

**Syntax**

```
length=STRLENGTH(string)
```

**Description**

Returns the length of a string.

**Examples**

```
STRLENGTH("HELLO WORLD") = 11
```

**strncmp**

**Type: External**

**Group: char**

**Syntax**

```
STRNCMP(string1, string2, no. of characters)
```

**Description**

Compares a number of characters in string1 to string2.

**Examples**

```
STRNCMP("ABcd", "abce", 3) 0 STRNCMP("abcd", "abce", 3) 1 STRNCMPI("abcd", "abce", 4) 0
```

**See Also**

strcmp, strcmpi, strncmpi

**strncmppi**

**Type: External**

**Group: char**

## Syntax

```
STRNCMP(string1, string2, no. of characters)
```

## Description

Compares a number of characters in string1 to string2, ignoring case.

## Examples

```
STRNCMP("ABcd", "abce", 3) 1 STRNCMP("abcd", "abce", 3) 1 STRNCMPI("abcd", "abce", 4) 0
```

## See Also

strcmp, strcmpi, strncmp

## strvcat

### Type: External

### Group: char

## Syntax

```
STRVCAT(string1, string2, ...)
```

## Description

A function to join a group of strings into a vertical matrix

## Examples

```
STRVCAT("a", "list", "of", "strings")
```

## See Also

strcat

## substring

### Type: External

### Group: char

## Syntax

```
answer = SUBSTRING(string, start, [length])
```

## Description

Returns a portion of a string.

## Examples

```
SUBSTRING("HELLO WORLD", 0, 5) = "HELLO" SUBSTRING("HELLO WORLD", 3, 5) =  
"LLO W"
```

## upper

## Type: External

## Group: char

## Syntax

```
answer = upper(string)
```

## Description

Converts a string to upper case.

## Notes

.

## Examples

.

## See Also

[lower](#)

---

# Chapter 22. control

## Introduction to the control toolbox

There will be some great features for control systems analysis in the future.

### `__abccddims__`

**Type: M-File**

**Group: control**

### Syntax

`__abccddims(A, B, C, D)__`

### Description

Used internally in `abccddim`.

### Examples

`__abccddims(A, B, C, D)__`

### See Also

`abccddim`

## `abccddim`

**Type: M-File**

**Group: control**

### Syntax

`abccddim(A, B, C, D);`

### Description

Check for compatibility of the dimensions of the matrices defining the linear system



## Examples

```
abddim(A,B,C,D);
```

## See Also

is\_abcd, is\_sample, ss2sys

## axis2dlim

### Type: M-File

### Group: control

### Syntax

```
axis2dlim()
```

### Description

## Examples

```
axis2dlim()
```

## See Also

## is\_abcd

### Type: M-File

### Group: control

### Syntax

```
is_abcd (a, b, c, d)
```

### Description

## Examples

```
is_abcd (a, b, c, d)
```

## See Also

```
abddim
```

## is\_sample

**Type:** M-File

**Group:** control

## Syntax

```
is_sample(Ts)
```

## Description

## Examples

```
is_sample(Ts)
```

## See Also

```
is_abcd
```

## ss2sys

**Type:** M-File

**Group:** control

## Syntax

```
ss2sys()
```

## Description

## Examples

```
ss2sys()
```

## See Also

## swap

Type: M-File

Group: control

## Syntax

```
swap(a,b)
```

## Description

Swap inputs

## Examples

```
swap(a,b) -> [b,a]
```

## See Also

## sysgettsam

Type: M-File

Group: control

## Syntax

```
sysgettsam()
```

## Description

## Examples

```
sysgettsam()
```

## See Also

# sysgettype

**Type: M-File**

**Group: control**

## Syntax

```
sysgettype()
```

## Description

## Examples

```
sysgettype()
```

## See Also

# zgfmul

**Type: M-File**

**Group: control**

## Syntax

```
zgfmul()
```

## Description

## Examples

```
zgfmul()
```

## See Also

---

# Chapter 23. crypto

## aes

**Type: External**

**Group: Crypto**

### Syntax

```
answer = AES ("encrypt", plain, key)
answer = AES ("e",      plain, key)
answer = AES ("decrypt", cipher, key)
answer = AES ("d",      cipher, key)
```

### Description

does encryption and decryption using the AES (advanced encryption standard) algorithm

### Examples

```
> k=str2num("0987098709870987")
k=[48,57,56,55,48,57,56,55,48,57,56,55,48,57,56,55]

> a=str2num("asdfasdfasdf")
a=[97,115,100,102,97,115,100,102,97,115,100,102,97,115,100,102]

> b=aes("e",a,k)
b=[-103,34,84,18,-46,-58,35,89,80,17,81,118,-4,-15,-93,-31]

> c=aes("d",b,k)
c=[97,115,100,102,97,115,100,102,97,115,100,102,97,115,100,102]

> num2str(c)
ans = asdfasdfasdf
```

---

# Chapter 24. demos

## vdp1

**Type: M-File**

**Group: Demos**

### Syntax

`vdp1(t,y)`

### Description

van der Pol differential equation with  $m=1$

### See Also

`vdp2`

## vdp2

**Type: M-File**

**Group: Demos**

### Syntax

`vdp2(t,y)`

### Description

van der Pol differential equation with  $m=2$

### See Also

`vdp1`

---

# Chapter 25. deprecated

## finite

Type: M-File

Group: deprecated

### Syntax

```
finite(x);
```

### Description

this function is deprecated.

### Examples

```
isfinite(x)
```

### See Also

isfinite

## isstr

Type: M-File

Group: deprecated

### Syntax

```
isstr(s)
```

### Description

check if argument is a string.

### Notes

This function is deprecated.

### Examples

## See Also

[ischar](#)



---

# Chapter 26. engine

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### engine

**Type: M-File**

**Group: engine**

### Syntax

`engine`

### Description

this is the basic command for using the "engine" of JMathLib

### Examples

---

# Chapter 27. finance

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### **fv**

**Type: M-File**

**Group: Finance**

### **Syntax**

$v = fv(r, n, p, l, m)$

### **Description**

Future value of an investment

### **Examples**

### **See Also**

[fv1](#)

### **fv1**

**Type: M-File**

**Group: Finance**

### **Syntax**

$v = fv(r, n, p, l, m)$

### **Description**

Future value of an initial lump sum investment

### **Examples**

## See Also

`fvl`

## `nper`

**Type: M-File**

**Group: Finance**

## Syntax

`n = nper (r, p, a, l, m)`

## Description

.

## Examples

## `pmt`

**Type: M-File**

**Group: Finance**

## Syntax

`p = pmt (r, n, a, l, m)`

## Description

.

## Examples

## `pv`

**Type: M-File**

## Group: Finance

### Syntax

$$v = pv(r, n, p, l, m)$$

### Description

.

### Examples

## pvl

### Type: M-File

## Group: Finance

### Syntax

$$v = pvl(r, n, p)$$

### Description

.

### Examples

## vol

### Type: M-File

## Group: Finance

### Syntax

$$vol(X, m, n)$$

### Description

.

## Examples

---

# Chapter 28. ftp

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### ftp

**Type: M-File**

**Group: ftp**

### Syntax

```
ftp('server')
```

### Description

Opens an ftp connection. THIS FUNCTION IS NOT YET IMPLEMENTED

### Examples

### See Also

[urlread](#)

---

# Chapter 29. funfun

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### euler

**Type: External**

**Group: FunFun**

### Syntax

```
[t,y] = euler (function, [t0 tf], y0, dt)
```

### Description

Integration of a given function using Euler's method.

### Examples

```
[t,y]=euler('vdp1',[0,10],[2,0],0.1)
```

### See Also

### feval

**Type: External**

**Group: FunFun**

### Syntax

```
feval(function, parameters)
```

### Description

evaluates the function on the supplies parameters.

### Examples

```
feval('cos', 1) = 0  
feval("max", 1, 3) = 3
```



---

# Chapter 30. general

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### **`_class`**

**Type: External**

**Group: general**

### **Syntax**

```
class(value)
```

### **Description**

returns the class name of some array.

### **Notes**

.

### **Examples**

```
a=55  
class(a) -> double
```

```
b='asdf'  
class(b) -> char
```

### **See Also**

isa, ismatrix, isnumeric, isscalar, issquare, islogical

## **angle**

**Type: External**

**Group: general**

## Syntax

```
angle(complex)
```

## Description

Returns the angle of a complex number

## Examples

```
angle(2i) = pi/2  
real(1 + i) = pi/4
```

## See Also

real, conj, imag, abs

# beep

## Type: External

## Group: general

## Syntax

```
beep();
```

## Description

this sounds an audible beep

## Examples

```
beep();
```

# bench

## Type: M-File

## Group: general

## Syntax

```
bench
```

## Description

this runs a benchmark test on the current machine

## Examples

```
bench
```

## bitand

**Type: External**

**Group: general**

## Syntax

```
bitand(number1, number2);
```

## Description

Calculates the bitwise AND of number1 and number2

## Examples

```
bitand(5, 9) = 1;  
bitand(5, 7) = 5;
```

## See Also

bitor, bitshift, bitxor

## bitor

**Type: External**

**Group: general**

## Syntax

```
bitor(number1, number2);
```

## Description

Calculates the bitwise OR of number1 and number2 @EXAMPLE

```
bitor(5, 9) = 13  
bitor(5, 7) = 7
```

## See Also

bitand, bitshift, bitxor

# bitshift

**Type: External**

**Group: general**

## Syntax

```
bitshift(number1, number2);
```

## Description

Shifts the bit pattern of number1 by number2 bits.

## Examples

```
bitshift(5, 2) = 1  
bitshift(5, -2) = 20
```

## See Also

bitand, bitor, bitxor

# bitxor

**Type: External**

**Group: general**

## Syntax

```
bitxor(number1, number2);
```

## Description

Calculates the bitwise Exclusive OR of number1 and number2

## Examples

```
bitxor(5, 9) = 12  
bitxor(5, 7) = 2
```

## See Also

bitand, bitor, bitshift

## ceil

### Type: External

### Group: general

### Syntax

```
ceil(value)
```

### Description

Rounds the value of the first operand up to the nearest integer

## Examples

```
ceil(-5.5) = -5 ceil(2.3) = 3
```

## See Also

floor, round

## cell

### Type: External

### Group: general

### Syntax

```
matrix = cell("a",7,"b",9)
```

### Description

creates a cell array

## Examples

```
matrix = cell("a",7,"b",9)
d.value = 33; d.name="hello"
```

## See Also

struct, iscell, isstruct

## clear

### Type: External

### Group: general

## Syntax

```
clear(variable)
```

## Description

Clears the specified variable or, if blank, clears all variables. `clear()` clears local variables `clear("variables")` clears local variables `clear("globals")` clears global variables `clear("functions")` clear function cache `clear("all")` clear local variables, global variables and function cache  
`@EXAMPLE`

```
clear('x');
clear();
```

## Notes

The variable should be given as a string containing the variable name.

## See Also

who, whos

## clock

### Type: External

### Group: general

## Syntax

```
clock()
```

## Description

Returns the current date and time as a six element vector.

## Notes

The variable should be given as a string containing the variable name.

## Examples

```
clock() = [2,002, 6, 6, 5, 23, 34]
```

## See Also

date, tic, toc, time

## combinations

### Type: External

### Group: general

## Syntax

```
answer = combinations(items, total)
```

## Description

Returns the number of combinations when taking count items from a set of total items.

## Notes

## Examples

```
combinations(3,5) = 10 combinations(3,6) = 20
```

## See Also

permutations

## complex

### Type: External

### Group: general

## Syntax

```
complex(real, imaginary)
```

## Description

Creates a complex number from real and imaginary

## Examples

```
complex(1,1) = 1+i
```

## conj

### Type: External

### Group: general

## Syntax

```
conj(complex)
```

## Description

Returns the conjugate of a complex number.

## Examples

```
conj(5) = 5  
conj(2 + i) = 2 - i
```

## See Also

imag, real, angle, abs

## diff

### Type: M-File

### Group: general

## Syntax

```
diff(x)
```



## Description

## Examples

## See Also

**e**

**Type: Internal**

**Group: general**

## Syntax

e

## Description

euler number

## Examples

```
e
ans = 2.718281828459046
```

## See Also

pi, eps, i, j

## eps

**Type: Internal**

**Group: general**

## Syntax

eps

## Description

smallest number epsilon

## Notes

.

## Examples

eps

## See Also

e, pi, i, j

## exp

### Type: External

### Group: general

## Syntax

`exp(value)`

## Description

.

## Examples

$\exp(-5.5) = -5 \exp(2.3) = 3$

## See Also

log, ln

## factor

### Type: External

### Group: general

## Syntax

```
answer=factor(values)
```

## Description

Calculates all the prime factors of value.

## Notes

This function will take a long time when factorising large values

## Examples

```
factor(10)=[2,5]  
factor(30)=[2,3,5]
```

## false

## Type: M-File

## Group: general

## Syntax

```
false(x)
```

## Description

.

## Notes

.

## Examples

```
false(2) -> [false, false; false, false]
```

## See Also

true, islogical, logical

## fibonacci

## Type: External

## Group: general

### Syntax

```
answer = fibonacci(value)
```

### Description

Returns the fibonacci number with an index of value.

### Examples

```
fibonacci(5) = 5  
fibonacci(10) = 55
```

## fix

### Type: External

## Group: general

### Syntax

```
answer=fix(matrix)
```

### Description

Rounds the element to the nearest element towards zero.

### Examples

```
fix(2.3)=3  
fix(-2.3)=-2  
fix(1.9+2.4i)=1+2i  
fix(1.9-2.4i)=1-2i
```

### See Also

ceil, round, floor

## func2str

### Type: External

## Group: general

### Syntax

```
answer=isfunctionhandle(value)
```

### Description

Checks if value is a function handle. Returning 1 if it is.

### Examples

```
isfunctionhandle( @sin ) -> 1
```

### See Also

[isa](#)

## getpfilecaching

### Type: External

## Group: General

### Syntax

```
x = getPFileCaching()
```

### Description

Returns whether or not caching of p-files is enabled or disabled. Returns 1 if caching is enabled. Returns 0 if caching is disabled.

### Examples

```
x = getPFileCaching()
```

### See Also

[setpfilecaching](#)

## global

### Type: External

## Group: general

### Syntax

```
global aaa
```

### Description

Returns 1 if the first operand is a cell array, else it returns 0.

### Examples

```
a=33;  
global a
```

### See Also

`isglobal`

## harmonic

### Type: External

## Group: general

### Syntax

```
answer = harmonic(value)
```

### Description

Returns the harmonic number with an index of value.

### Notes

This calculates  $\text{sum}(1..\text{value})\{1/n\}$

### Examples

```
harmonic(5) = 2.283333333333333  
harmonic(10) = 2.9289682539682538
```

## i

**Type: Internal**

**Group: general**

**Syntax**

`i`

**Description**

imaginary unit

**Examples**

```
1+i
3i
5j
```

**See Also**

`j`

**imag**

**Type: External**

**Group: general**

**Syntax**

`imag(complex)`

**Description**

Returns the imaginary part of a complex number

**Examples**

```
imag(2i) = 2
imag(5 + 3i) = 3
```

**See Also**

`real`, `conj`, `angle`, `abs`

# int16

**Type: External**

**Group: general**

## Syntax

`int16(x)`

## Description

converts a double array into an array of int16 (range  $-32768=-2^{15}$  up to  $+32767=+2^{15}-1$ )

## Examples

## See Also

`double`, `int8`, `uint8`, `uint16`, `uint32`

# int32

**Type: External**

**Group: general**

## Syntax

`int16(x)`

## Description

converts a double array into an array of int16 (range  $-32768=-2^{15}$  up to  $+32767=+2^{15}-1$ )

## Examples

## See Also

`double`, `int8`, `uint8`, `uint16`, `int16`, `int32`, `int64`



# int64

**Type: External**

**Group: general**

## Syntax

`int16(x)`

## Description

converts a double array into an array of int16 (range  $-32768=-2^{15}$  up to  $+32767=+2^{15}-1$ )

## Examples

## See Also

`double`, `int8`, `uint8`, `uint16`

# int8

**Type: External**

**Group: general**

## Syntax

`int8(x)`

## Description

converts a double array into an array of int8 (range  $-128$  up to  $+127$ )

## Examples

## See Also

`double`, `int16`, `uint8`, `uint16`, `uint32`

# isa

**Type: External**

**Group: general**

## Syntax

```
isa(value, 'class')
```

## Description

.

## Notes

.

## Examples

```
a=55
isa(a, 'double')  -> 1
isa(a, 'char')    -> 0
```

## See Also

ismatrix, isnumeric, isscalar, issquare, islogical

# iscell

**Type: External**

**Group: general**

## Syntax

```
answer = iscell(value)
```

## Description

Returns 1 if the first operand is a cell array, else it returns 0.

## Examples

```
a={4,5,'barfoo'}  
iscell(a)=1  
iscell('hello')=0  
iscell([5,6,7])=0
```

## See Also

ismatrix, isnumeric

# ischar

## Type: External

## Group: general

## Syntax

```
answer = ischar(value)
```

## Description

Returns 1 if the first operand is a struct, else it returns 0.

## Notes

.

## Examples

```
ischar("hello") returns 1  
ischar(55) returns 0
```

## See Also

ismatrix, isnumeric, isscalar, issquare

# isdouble

## Type: M-File

## Group: general

## Syntax

```
isdouble(x)
```

## Description

## Notes

## Examples

## See Also

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

# isfunctionhandle

**Type:** External

**Group:** general

## Syntax

```
answer=isfunctionhandle(value)
```

## Description

Checks if value is a function handle. Returning 1 if it is.

## Examples

```
isfunctionhandle( @sin ) -> 1
```

## See Also

`isa`

# isglobal

**Type:** External

**Group:** general

## Syntax

```
answer = iscell(value)
```

## Description

Returns 1 if the first operand is a cell array, else it returns 0.

## Examples

```
a={4,5,'barfoo'}  
iscell(a)=1  
iscell('hello')=0  
iscell([5,6,7])=0
```

## See Also

ismatrix, isnumeric

## isint16

### Type: M-File

### Group: general

## Syntax

```
isint16(x)
```

## Description

.

## Notes

.

## Examples

## See Also

ischar, iscell, isnumeric, ismatrix, isprime

## isint32

**Type: M-File**

**Group: general**

**Syntax**

`isint32(x)`

**Description**

.

**Notes**

.

**Examples**

**See Also**

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

**isint64**

**Type: M-File**

**Group: general**

**Syntax**

`isint64(x)`

**Description**

.

**Notes**

.

**Examples**

## See Also

ischar, iscell, isnumeric, ismatrix, isprime

# isint8

**Type:** M-File

**Group:** general

## Syntax

```
isint8(x)
```

## Description

.

## Notes

.

## Examples

## See Also

ischar, iscell, isnumeric, ismatrix, isprime

# islogical

**Type:** External

**Group:** general

## Syntax

```
answer = islogical(value)
```

## Description

Returns 1 if the first operand is a logical array, else it returns 0.

## Notes

.

## Examples

```
a=logical([1,2,0])  
islogical(a)  -> 1
```

## See Also

ismatrix, isnumeric, isscalar, issquare

## ismatrix

### Type: M-File

### Group: general

### Syntax

```
ismatrix(values)
```

### Description

.

### Notes

.

## Examples

```
ismatrix([88])  
ismatrix("hello")
```

## See Also

ischar, iscell, isnumeric, isprime

## isnumeric

### Type: External

### Group: general

### Syntax

```
answer = isnumeric(value)
```



## Description

Returns 1 if the first operand is a number, else it returns 0.

## Notes

## Examples

```
isnumeric(2) = 1  
isnumeric("test") = 0
```

## See Also

ismatrix, isscalar, isprime, issquare

## isprime

### Type: External

### Group: general

## Syntax

```
answer=isprime(value)
```

## Description

Checks if value is a prime. returning 1 if it is.

## Examples

```
isprime(3)=1  
isprime(4)=0
```

## See Also

primes

## isscalar

### Type: M-File

## Group: general

### Syntax

```
isscalar(values)
```

### Description

.

### Notes

.

### Examples

```
ismatrix([88])  
isnumeric([55, 66])
```

### See Also

ischar, iscell, isnumeric, ismatrix, isprime

## issingle

### Type: M-File

## Group: general

### Syntax

```
issingle(x)
```

### Description

.

### Notes

.

### Examples

### See Also

ischar, iscell, isnumeric, ismatrix, isprime, double, int8, isint8, isuint16, isint16, isint32, isint64

# issquare

**Type: M-File**

**Group: general**

## Syntax

```
issquare(values)
```

## Description

.

## Notes

.

## Examples

```
issquare([88, 99; 33, 44])  
issquare([22, 33])
```

## See Also

ischar, iscell, isnumeric, isprime

# isstruct

**Type: External**

**Group: general**

## Syntax

```
answer = isstruct(value)
```

## Description

Returns 1 if the first operand is a struct, else it returns 0.

## Examples

```
bar.foo=7  
isstruct(bar) returns 1  
isstruct(55) returns 0
```

## See Also

ismatrix, isnumeric, isscalar, issquare

# isstudent

**Type: External**

**Group: general**

## Syntax

```
isstudent()
```

## Description

checks if a student version or a commercial version is used. Since JMathLib is open source this function always returns 0

## Examples

```
isstudent() -> 0
```

## See Also

# isuint16

**Type: M-File**

**Group: general**

## Syntax

```
isuint16(x)
```

## Description

.

## Notes

.

## Examples

## See Also

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

## isuint32

**Type:** M-File

**Group:** general

## Syntax

`isuint32(x)`

## Description

.

## Notes

.

## Examples

## See Also

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

## isuint64

**Type:** M-File

**Group:** general

## Syntax

`isuint64(x)`

## Description

## Notes

## Examples

## See Also

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

# isuint8

## Type: M-File

## Group: general

## Syntax

`isuint8(x)`

## Description

## Notes

## Examples

## See Also

`ischar`, `iscell`, `isnumeric`, `ismatrix`, `isprime`

# j

## Type: Internal

## Group: general

### Syntax

$j$

### Description

imaginary unit

### Examples

$1+j$   
 $3i$   
 $5j$

### See Also

$i$

## kbhit

### Type: External

## Group: general

### Syntax

`beep();`

### Description

this sounds an audible beep

### Examples

`beep();`

## length

### Type: External

## Group: general

## Syntax

```
answer = length(string)
answer = length(matrix)
answer = number(matrix)
```

## Description

Returns the length of a matrix or a string.

## Examples

```
length("test") = 4
length([1, 2, 3]) = 3
```

## See Also

size

# linspace

## Type: External

## Group: general

## Syntax

```
matrix = linspace(start, end, no of values)
```

## Description

Returns a matrix going from start to end with count elements.

## Notes

A similar result can be obtained by using the colon operator.

## Examples

```
linspace(0, 1, 11) = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
linspace(0, 10 , 3) = [0, 5, 10]
```

# logical

## Type: External



## Group: general

### Syntax

```
logical(x)
```

### Description

converts a double array into an array of boolean

### Examples

```
logical([1,3,0,4]) -> [true, true, false, true]
```

### See Also

true, false, islogical

## ndims

### Type: External

### Group: General

### Syntax

```
answer = ndims(value)
```

### Description

Returns the sign of value.

### Notes

.

### Examples

```
a=rand(5,5);  
ndims(a) -> 2
```

```
a=rand(5,5,5)  
ndims(a) -> 3
```

### See Also

template

# performfunction

**Type: External**

**Group: general**

## Syntax

`PERFORMFUNCTION(function, parameters)`

## Description

Performs the function on the supplies parameters.

## Examples

`PERFORMFUNCTION(ACOS, 1) = 0` `PERFORMFUNCTION(MAX,1,3) = 3`

# permutations

**Type: External**

**Group: general**

## Syntax

`answer = PERMUTATIONS(items, total)`

## Description

Returns the number of permutations when taking count items from a set of total items.

## Notes

## Examples

`PERMUTATIONS(3,5) = 60` `PERMUTATIONS(3,6) = 120`

## See Also

[combinations](#)

# pi

**Type: Internal**

**Group: general**

**Syntax**

pi

**Description**

mathematical constant pi

**Notes**

Pi is a mathematical constant that represents the ratio of any circle's circumference to its diameter in Euclidean geometry, which is the same as the ratio of a circle's area to the square of its radius.

**Examples**

```
pi
ans=3.1415926
```

**See Also**

e, eps, i, j

**primes**

**Type: External**

**Group: general**

**Syntax**

PRIMES(value)

**Description**

Calculates all the primes up to value.

**Examples**

```
PRIMES(10) = [2, 3, 5, 7]
```

**See Also**

isprime

# rand

**Type: External**

**Group: general**

## Syntax

```
matrix = rand(size)
matrix = rand(x,y,...)
number = rand()
```

## Description

Returns a matrix filled with random numbers.

## Examples

```
rand(3,2) = [0.34, 0.45, 0.55 0.22, 0.33, 0.72]
```

# real

**Type: External**

**Group: general**

## Syntax

```
real(complex)
```

## Description

Returns the real part of a complex number

## Examples

```
real(2) = 2
real(5 + 3i) = 5
```

## See Also

imag, conj, angle, abs

# return

**Type: Internal**

**Group: general**

**Syntax**

`return`

**Description**

returns from a function. (Internal Note: this function is done in the function token)

**Examples**

`return`

**round**

**Type: External**

**Group: general**

**Syntax**

`answer = round(value)`

**Description**

Rounds a value to the nearest integer.

**Examples**

`round(2.2) = 2 round(5.5) = 6`

**See Also**

`ceil, floor`

**setpfilecaching**

**Type: External**

**Group: General**

## Syntax

`setPFileCaching(value)`

## Description

enables or disables caching of p-files

## Examples

```
setPFileCaching(1)      enable caching
setPFileCaching('on')   enable caching
setPFileCaching(0)      disable caching
setPFileCaching('off')  disable caching
```

## See Also

`getpfilecaching`

## sign

## Type: External

## Group: general

## Syntax

`sign(value)`

## Description

Returns the sign of value.

## Notes

The sign of a complex number is calculated as  $\text{sign}(x) = x./\text{abs}(x)$

## Examples

```
sign(-10) = -1
sign(10)  = 1
sign(3+2i) = 0.832 + 0.555i
```

## See Also

`abs`

# size

**Type: External**

**Group: general**

## Syntax

```
[y, x] = size(matrix)
z      = size(matrix, n)
```

## Description

returns the size of a matrix.

## Examples

```
size([1,2;3,4]) = [2,2]
size([1,2,3;4,5,6]) = [2,3]
```

```
a=rand(4,4,2)
size(a) -> [4,4,2]
```

```
size(a,3) -> 2
```

## See Also

rows, columns, row, col, ndims

# sqrt

**Type: External**

**Group: general**

## Syntax

```
answer = sqrt(value)
```

## Description

Returns the sqrt of a value.

## Examples

```
sqrt(4) = 2 sqrt(9) = 3
```

## See Also

angle, abs

## str2func

**Type: External**

**Group: general**

## Syntax

```
answer=isfunctionhandle(value)
```

## Description

Checks if value is a function handle. Returning 1 if it is.

## Examples

```
isfunctionhandle( @sin ) -> 1
```

## See Also

isa

## struct

**Type: External**

**Group: general**

## Syntax

```
structure = STRUCT(variable1, value1, variable2, value2,..., variableN, valueN)  
structure = STRUCT(structure, variable1, value1, variable2, value2,..., variableN, valueN)
```

## Description

Creates a structured variable. If the first paramater is a structure then the structure inherits it's values.

## Examples

```
x=STRUCT("a", 1, "b", 2) = a = 1 : b = 2 :
```



```
y=STRUCT(x,"c",3) = a = 1 : b = 2 : c = 3 :
```

## template

**Type: External**

**Group: General**

### Syntax

```
answer = template (value)
```

### Description

Returns the sign of value.

### Notes

This functions is used as a template for developing toolbox functions.

### Examples

```
sign(-10)=-1  
sign(10)=1
```

### See Also

template

## true

**Type: M-File**

**Group: general**

### Syntax

```
true(x)
```

### Description

.

### Notes

.

## Examples

```
true(2) -> [true, true; true true]
```

## See Also

false, islogical, logical

## uint16

### Type: External

### Group: general

### Syntax

```
uint8(x)
```

### Description

converts a double array into an array of uint16

## Examples

## See Also

double, int16, int8, uint16

## uint32

### Type: External

### Group: general

### Syntax

```
uint32(x)
```

### Description

converts a double array into an array of uint8 (range 0 up to +255)

## Examples

## See Also

`double`, `int16`, `int8`, `uint8`, `uint16`

## uint8

**Type: External**

**Group: general**

## Syntax

`uint8(x)`

## Description

converts a double array into an array of uint8 (range 0 up to +255)

## Examples

## See Also

`double`, `int16`, `int8`, `uint16`, `uint32`

## who

**Type: External**

**Group: general**

## Syntax

`who`

## Description

Returns a list of all the variables in the system.

## Examples

```
who  
ans  
x  
y
```

## See Also

whos

## whos

**Type: External**

**Group: general**

## Syntax

```
whos  
whos("global")
```

## Description

Returns a list of all the variables in the system.

## See Also

who, clear, global

---

# Chapter 31. graph3d

## surf

**Type:** External

**Group:** Graph3d

### Syntax

`SURF(matrix)`

### Description

Plots a 3 dimensional surface

### Examples

```
[x,y]=MESHGRID([-2:0.2:2],[-2:0.2:2]) SURF(sin(x).*cos(y))
```

### See Also

`plot`, `plot3`

---

# Chapter 32. graphics

## cla

**Type:** External

**Group:** graphics

### Syntax

`cla`

### Description

clear current figure

### Examples

.

### See Also

`gca`, `clf`, `gcf`, `get`, `set`

## clf

**Type:** External

**Group:** graphics

### Syntax

`clf`

### Description

clear current figure

### Examples

.

### See Also

`gca`, `cla`, `gcf`, `get`, `set`

# close

**Type: External**

**Group: graphics**

## Syntax

```
close(figure number)  
close()
```

## Description

Creates a graphics window (*figure*)

## Examples

```
close(1)
```

## See Also

*figure*, *plot*

# figure

**Type: External**

**Group: graphics**

## Syntax

```
figure(figure number)  
figure()
```

## Description

Creates a graphics window (*figure*)

## Examples

```
figure(1)
```

## See Also

*plot*, *close*

# gca

**Type: External**

**Group: graphics**

**Syntax**

`gca`

**Description**

get handle to current axis

**Examples**

.

**See Also**

`gcf`, `get`, `set`

# gcf

**Type: External**

**Group: graphics**

**Syntax**

`gcf`

**Description**

get handle to current figure

**Examples**

.

**See Also**

`gca`, `get`, `set`

# get



**Type: External**

**Group: graphics**

**Syntax**

`get`

**Description**

.

**Examples**

.

**See Also**

`set`, `plot`

**ginput**

**Type: External**

**Group: graphics**

**Syntax**

`ginput`

**Description**

.

**Examples**

`[x,y]=ginput`

`[x,y]=ginput(4)`

**See Also**

`plot`, `get`, `set`

**grid**

**Type: External**

**Group: graphics**

## Syntax

```
grid(value)
```

## Description

Turns the grid on or off on a graph.

## Examples

```
grid("on")      turns the grid on
grid("off")     turns the grid off
```

# hold

**Type: External**

**Group: graphics**

## Syntax

```
HOLD(figure)
```

## Description

Holds a graph so that more graphs can be drawn over the top of it.

## Examples

```
hold(1)
```

# meshgrid

**Type: External**

**Group: graphics**

## Syntax

```
[x,y]=MESHGRID(x-values,y-values)
```

## Description

computes the mesh of a matrix.

## Examples

```
[x,y]=MESHGRID([-2:0.2:2],[-2:0.2:2])
```

## See Also

plot, plot3

# plot

## Type: External

## Group: graphics

## Syntax

```
PLOT(x-values, y-values, graph settings)
```

## Description

Plots a graph of the supplied points.

## Notes

graph settings

color

y yellow

m magenta

c cyan

r red

g green

b blue

w white

k black

linestyle

- solid

: dotted

-. dashdot

-- dashed

marker

|   |                  |
|---|------------------|
| . | point            |
| o | circle           |
| x | x-mark           |
| + | plus             |
| * | star             |
| s | square           |
| d | diamond          |
| v | triangle (down)  |
| ^ | triangle (up)    |
| < | triangle (left)  |
| > | triangle (right) |
| p | pentagram        |
| h | hexagram         |

## Examples

```
plot([1,2,3], rand(3), 'b')
```

## See Also

plotfunction, plot3

## plot3

### Type: External

### Group: graphics

## Syntax

```
plot3(x-values, y-values, z-values, [graph settings])
```

## Description

Plots a 3 dimensional graph of the supplied points.

## Notes

graph settings  
color  
y yellow  
m magenta  
c cyan  
r red  
g green  
b blue  
w white  
k black  
  
linestyle

```
-      solid
:      dotted
- .    dashdot
--     dashed

marker
.      point
o      circle
x      x-mark
+      plus
*      star
s      square
d      diamond
v      triangle (down)
^      triangle (up)
<      triangle (left)
>      triangle (right)
p      pentagram
h      hexagram
```

## Examples

```
t = 0:0.5:100; plot3(SIN(t), COS(t), t/100);
```

## See Also

plot, plotfunction

# plotfunction

## Type: External

## Group: graphics

## Syntax

```
PLOTFUNCTION(function, start, end, [no of points], [graph options])
```

## Description

plots a graph of the function between the start and end points. The number of points defaults to 100.

## Notes

graph settings color y yellow m magenta c cyan r red g green b blue w white k black linestyle - solid : dotted -. dashdot -- dashed marker . point o circle x x-mark + plus \* star s square d diamond v triangle (down) ^ triangle (up) < triangle (left) > triangle (right) p pentagram h hexagram

## Examples

```
PLOTFUNCTION("SIN", 0, 4, 20)
```

## See Also

plot, plot3

# polar

## Type: External

## Group: graphics

## Syntax

polar(x-values, y-values, graph settings)

## Description

Plots a graph of the supplied points in polar coordinates

## Notes

```
graph settings
color
y    yellow
m    magenta
c    cyan
r    red
g    green
b    blue
w    white
k    black

linestyle
-    solid
:    dotted
- .  dashdot
--   dashed

marker
.    point
o    circle
x    x-mark
+    plus
*    star
s    square
d    diamond
v    triangle (down)
^    triangle (up)
<    triangle (left)
>    triangle (right)
p    pentagram
h    hexagram
```

## Examples

```
polar([1,2,3], RAND(3), 'b')
```

## See Also

`plot`

## print

## Type: External

## Group: graphics

## Syntax

```
print
```

## Description

Prints the current figure to a printer

## Examples

```
print
```

## See Also

`plot`, `plot3`, `surf`, `xlabel`, `ylabel`, `zlabel`, `rotate`, `meshgrid`, `subplot`, `title`, `figure`

## propertyeditor

## Type: External

## Group: graphics

## Syntax

```
propertyeditor( handle )
```

## Description

this functions is used to display a table of graphical properties of the element described by handle.

## Examples

```
propertyeditor( gca() )
```

## See Also

get, set

## rotate

### Type: External

### Group: graphics

## Syntax

```
rotate(phiX, phiY, phiZ)
```

## Description

Turns the grid on or off on a graph.

## See Also

plot3, surf

## set

### Type: External

### Group: graphics

## Syntax

```
get
```

## Description

.

## Examples

.

## See Also

set, plot



# subplot

**Type: External**

**Group: graphics**

## Syntax

```
SUBPLOT(y, x, axes)
```

## Description

divides a graph into cells.

## Examples

```
subplot(2,4, 1)
```

## See Also

plot, plot3, surf

# title

**Type: External**

**Group: graphics**

## Syntax

```
title(title-string)
```

## Description

Displays a title on a graph

## Examples

```
title("graph title")
```

## See Also

xlabel, ylabel, zlabel

# xlabel

**Type: External**

**Group: graphics**

## Syntax

```
xlabel(label)
```

## Description

Sets the label on the x axis of a graph

## Notes

## Examples

```
xlabel("x axis")
```

## See Also

ylabel, zlabel, title

# ylabel

**Type: External**

**Group: graphics**

## Syntax

```
ylabel(label)
```

## Description

Sets the label on the y axis of a graph

## Examples

```
ylabel("y axis")
```

## See Also

xlabel, ylabel, title

# ylabel

**Type: External**

**Group: graphics**

## Syntax

```
ylabel(label)
```

## Description

Sets the label on the z axis of a graph

## Examples

```
ylabel("z axis")
```

## See Also

xlabel, ylabel, title

---

# Chapter 33. image

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### autumn

**Type: M-File**

**Group: image**

#### Syntax

```
autumn
```

#### Description

Create color colormap. This colormap is red through orange to yellow.

#### Examples

.

#### See Also

bone, cool, copper, gray, hot, jet, pink, rainbow, spring, summer, white, winter

### bone

**Type: M-File**

**Group: image**

#### Syntax

```
bone
```

#### Description

Create color colormap. This colormap is a gray colormap with a light blue tone.

## Examples

.

## See Also

autumn, cool, copper, gray, hot, jet, pink, rainbow, spring, summer, white, winter

## brighten

### Type: M-File

### Group: image

### Syntax

`brighten`

### Description

Darkens or brightens the given colormap.

## Examples

.

## colormap

### Type: M-File

### Group: image

### Syntax

`colormap`

### Description

.

## Examples

.

# cool

**Type: M-File**

**Group: image**

## Syntax

`cool`

## Description

Create color colormap. The colormap is cyan to magenta.

## Examples

.

## See Also

autumn, bone, copper, hot, jet, pink, rainbow, spring, summer, white, winter

# copper

**Type: M-File**

**Group: image**

## Syntax

`copper`

## Description

Create color colormap. This colormap is black to a light copper tone.

## Examples

.

## See Also

autumn, bone, cool, hot, jet, pink, rainbow, spring, summer, white, winter

# gray

**Type: M-File**

**Group: image**

## Syntax

```
gray( URL )
```

## Description

Return a gray colormap

## Notes

.

## Examples

## See Also

gray2ind, ntsc2rgb

# gray2ind

**Type: M-File**

**Group: image**

## Syntax

```
gray2ind( )
```

## Description

Convert a gray scale intensity image to an Octave indexed image.

## Notes

.

## Examples

## See Also

gray, ntsc2rgb

## hot

**Type:** M-File

**Group:** image

## Syntax

hot

## Description

Create color colormap. This colormap is black through dark red, red, orange, yellow to white.

## Examples

.

## See Also

autumn, bone, cool, copper, gray, jet, pink, rainbow, spring, summer, white, winter

## jet

**Type:** M-File

**Group:** image

## Syntax

jet

## Description

Create color colormap. This colormap is dark blue through blue, cyan, green, yellow, red to dark red.

## Examples

.



## See Also

bone, cool, copper, gray, hot, pink, rainbow, spring, summer, white, winter

## ntsc2rgb

### Type: M-File

### Group: image

### Syntax

```
ntsc2rgb( )
```

### Description

Image format conversion

### Notes

.

### Examples

## See Also

gray, gray2ind

## pink

### Type: M-File

### Group: image

### Syntax

```
pink
```

### Description

Create color colormap. This colormap gives a sephia tone on black and white images.

### Examples

.

## See Also

bone, cool, copper, gray, hot, jet, rainbow, spring, summer, white, winter

# rainbow

**Type: M-File**

**Group: image**

## Syntax

`rainbow`

## Description

Create color colormap. This colormap is red through orange, yellow, green, blue to violet.

## Examples

.

## See Also

autumn, bone, cool, copper, gray, hot, jet, pink, spring, summer, white, winter

# spring

**Type: M-File**

**Group: image**

## Syntax

`spring`

## Description

Create color colormap. This colormap is magenta to yellow.

## Examples

.

## See Also

bone, cool, copper, gray, hot, jet, pink, rainbow, summer, white, winter

## summer

**Type: M-File**

**Group: image**

## Syntax

summer

## Description

Create color colormap. This colormap is green to yellow.

## Examples

.

## See Also

bone, cool, copper, gray, hot, jet, pink, rainbow, spring, white, winter

## white

**Type: M-File**

**Group: image**

## Syntax

white

## Description

Create color colormap. This colormap is completely white.

## Examples

.

## See Also

bone, cool, copper, gray, hot, jet, pink, rainbow, spring, summer, winter

## winter

### Type: M-File

### Group: image

### Syntax

winter

### Description

Create color colormap. This colormap is blue to green.

### Examples

.

## See Also

bone, cool, copper, gray, hot, jet, pink, rainbow, spring, summer, white

---

# Chapter 34. integration

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### crule

**Type: M-File**

**Group: integration**

### Syntax

`crule`

### Description

This function computes Gauss-Chebyshev base points and weight factors using the algorithm given by somebody in 'SomeBook', page 365, Academic Press, 1975, but modified by a change in index variables:  $j=i+1$  and  $m=n+1$ . The weights are all  $wf_j=\pi/m$  and the base points are  $bp_j=\cos((2j-1)*\pi/2/m)$ .

### Examples

.

### See Also

`ncrule`

### ncrule

**Type: M-File**

**Group: integration**

### Syntax

`ncrule`

## Description

This function returns the Newton-Coates base points and weight factors up to an 8 point Newton-Coates formula.

## Examples

.

## See Also

[crule](#)

---

# Chapter 35. internal minusminus

**Type:** Internal

**Group:** internal

**Syntax**

--

**Description**

subtract 1.0 to a number or variable

**Examples**

```
4 - -  
ans=3
```

```
[1, 2, 3] - -  
ans=[0, 1, 2]
```

**See Also**

plusplus, plus, minus

## plusplus

**Type:** Internal

**Group:** internal

**Syntax**

++

**Description**

add 1.0 to a number or variable

**Examples**

```
2++  
ans=3
```

```
[1, 2, 3]++  
ans=[2, 3, 4]
```

## See Also

minusminus, plus, minus



---

# Chapter 36. io

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### cd

**Type: External**

**Group: IO**

### Syntax

```
cd(directory)
```

### Description

Sets the working directory to directory. Also switches between directories.

### Examples

```
cd("C:\barfoo");
```

### See Also

dir, cd, isdirectory

## createnewfile

**Type: External**

**Group: IO**

### Syntax

```
createnewfile(filename)
```

### Description

use this script to create a new file.

## Examples

```
createnewfile("bar.txt");
```

## See Also

cd, dir, exist, mkdir, rmdir, delete, isfile, isdirectory, ishidden, lastmodified

## csvread

### Type: External

### Group: IO

### Syntax

```
matrix=csvread(filename, startrow, endrow)
```

### Description

Reads in a matrix from a comma seperated value file.

## Examples

```
cvsreac("testfile.csv", 0, 0)=[1,2;3,4]
```

## See Also

csvwrite, urlread, dir, cd

## csvwrite

### Type: External

### Group: IO

### Syntax

```
csvwrite(filename, matrix, startrow, endrow)
```

### Description

Writes a matrix out to a file in comma seperated value format.

## Examples

```
csvwrite("testfile.csv", [1,2;3,4], 0 ,0)
```

## See Also

csvread, urlread, dir, delete

## delete

### Type: External

### Group: IO

### Syntax

```
delete(filename)
```

### Description

delete a file

## Examples

```
delete("bar.txt");
```

## See Also

cd, createnewfile, dir, exist, mkdir, rmdir, isfile, isdirectory, ishidden, lastmodified

## dir

### Type: External

### Group: IO

### Syntax

```
dir()  
dir(directory)
```

### Description

Displays the file list for the current directory.

## Examples

```
dir()
MathLib.MathLib.html
MathLib.log
MathLib.bat
object.txt
Demos.class
MathLib/
CVS/
Demos$1.class
1.0
```

## See Also

[cd](#), [creatnewfile](#), [exist](#), [mkdir](#), [rmdir](#), [delete](#), [isfile](#), [isdirectory](#), [ishidden](#), [lastmodified](#)

## exist

### Type: External

### Group: IO

### Syntax

```
exist(filename)
```

### Description

Checks if a file exists

## Examples

```
exist("bar.txt")
```

## See Also

[cd](#), [creatnewfile](#), [dir](#), [mkdir](#), [rmdir](#), [delete](#), [isfile](#), [isdirectory](#), [ishidden](#), [lastmodified](#)

## isdirectory

### Type: External

### Group: IO

## Syntax

```
isdirectory(name)
```

## Description

checks if the given name is a directory

## Examples

```
isdirectory("bar");
```

## See Also

[cd](#), [createnewfile](#), [dir](#), [exist](#), [mkdir](#), [rmdir](#), [delete](#), [isfile](#), [ishidden](#), [lastmodified](#)

## isfile

### Type: External

### Group: IO

## Syntax

```
isfile(filename)
```

## Description

Checks if filename is a valid file in the current working directory.

## Examples

```
isfile("bar.txt");
```

## See Also

[cd](#), [createnewfile](#), [dir](#), [exist](#), [mkdir](#), [rmdir](#), [delete](#), [isdirectory](#), [ishidden](#), [lastmodified](#)

## ishidden

### Type: External

### Group: IO

## Syntax

```
ishidden(filename)
```

## Description

Check the file attribute of filename and returns if the file is a hidden file.

## Examples

```
ishidden("bar.txt");
```

## See Also

cd, createnewfile, dir, exist, mkdir, rmdir, delete, isfile, isdirectory, lastmodified

# lastmodified

## Type: External

## Group: IO

## Syntax

```
lastmodified(filename)
```

## Description

Returns the date of the last modification of given file

## Examples

```
lastmodified("barfoo.txt");
```

## See Also

cd, createnewfile, dir, exist, mkdir, rmdir, delete, isfile, isdirectory, ishidden

# load

## Type: External

## Group: IO

## Syntax

```
load(file)
```

## Description

Reads in a matrix from a comma seperated value file.

## Examples

```
load
```

## See Also

csvread, csvwrite

# loadvariables

## Type: External

## Group: IO

## Syntax

```
loadvariables()  
loadvariables("name.mlf")
```

## Description

loads the variables from a serialized .mlf-file

## Notes

The variables are written as a serialized stream of java objects. Therefor the format of the .mlf-file is java-specific. The file is also specific to the versions of MathLib token and classes.

## Examples

```
loadvariables()
```

## See Also

savevariables, load, dir, csvread, csvwrite

# mkdir

**Type: External**

**Group: IO**

**Syntax**

```
mkdir(directory)
```

**Description**

Creates a new directory.

**Examples**

```
mkdir("bar");
```

**See Also**

cd, createnewfile, dir, exist, rmdir, delete, isfile, isdirectory, ishidden, lastmodified

**pwd**

**Type: External**

**Group: IO**

**Syntax**

```
pwd
```

**Description**

displays the current working directory

**Examples**

```
pwd
```

**See Also**

cd, createnewfile, dir, exist, mkdir, rmdir, delete, isfile, ishidden, lastmodified

**rmdir**



## Type: External

## Group: IO

## Syntax

```
rmmdir(directory)
```

## Description

Deletes a given directory.

## Notes

Directory must be empty in order to be removed.

## Examples

```
rmmdir("bar");
```

## See Also

cd, createnewfile, dir, exist, mkdir, delete, isfile, isdirectory, ishidden, lastmodified

# runfile

## Type: External

## Group: IO

## Syntax

```
runfile("filename")
```

## Description

Runs the script file specified by filename. @NOTE This is used to run script files, not function files.

## Examples

```
runfile("script.m");
```

## See Also

dir, cd, systemcommand

# savevariables

**Type: External**

**Group: IO**

## Syntax

```
savevariables()  
savevariables("name.mlf")
```

## Description

saves the variables from the current workspace into a serialized .mlf-file

## Notes

The variables are written as a serialized stream of java objects. Therefor the format of the .mlf-file is java-specific. The file is also specific to the versions of MathLib token and classes.

## Examples

```
savevariables()
```

## See Also

loadvariables, csvread, csvwrite

# systemcommand

**Type: External**

**Group: IO**

## Syntax

```
systemcommand(command name)
```

## Description

A function for running external commands

## Examples

`systemcommand("EMACS")`

## See Also

`runfile`

---

# Chapter 37. jmathlib

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### freemat

**Type: M-File**

**Group: jmathlib**

### Syntax

`freemat`

### Description

Freemat is a great tool. JMathLib is a clone of Freemat, but written 100% in java.

### Examples

.

### See Also

`jmathlib`, `matlab`, `scilab`, `octave`

## jmathlib

**Type: M-File**

**Group: jmathlib**

### Syntax

`jmathlib`

### Description

JMathLib is a great tool. It works like Matlab, Freemat, Octave, Scilab, but it is written completely in java

## Examples

.

## See Also

matlab, scilab, freemat, octave

## matlab

### Type: M-File

### Group: jmathlib

### Syntax

matlab

### Description

Matlab is a great tool. JMathLib is a clone of Matlab, but written 100% in java.

## Examples

.

## See Also

jmathlib, scilab, freemat, octave

## octave

### Type: M-File

### Group: jmathlib

### Syntax

octave

### Description

Octave is a great tool. JMathLib is a clone of Octave, but written 100% in java.

## Examples

.

## See Also

jmathlib, matlab, freemat, scilab

## scilab

### Type: M-File

### Group: jmathlib

## Syntax

scilab

## Description

Scilab is a great tool. JMathLib is a clone of Scilab, but written 100% in java.

## Examples

.

## See Also

jmathlib, matlab, freemat, octave

---

# Chapter 38. linearalgebra

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### **commutation\_matrix**

**Type:** M-File

**Group:** LinearAlgebra

#### **Syntax**

`commutation_matrix`

#### **Description**

.

#### **Examples**

### **dmult**

**Type:** M-File

**Group:** LinearAlgebra

#### **Syntax**

`dmult`

#### **Description**

.

#### **Examples**

### **dot**

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`dot()`

**Description**

.

**Examples**

**duplication\_matrix**

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`duplication_matrix`

**Description**

.

**Examples**

**housh**

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`housh()`



## Description

.

## Examples

## logm

Type: M-File

Group: LinearAlgebra

## Syntax

`logm(x)`

## Description

.

## Examples

## norm

Type: M-File

Group: LinearAlgebra

## Syntax

`norm()`

## Description

.

## Examples

## null

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`null()`

**Description**

.

**Examples**

**orth**

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`orth (A, tol)`

**Description**

.

**Examples**

**rank**

**Type: M-File**

**Group: LinearAlgebra**

**Syntax**

`rank()`

## Description

.

## Examples

## trace

## Type: M-File

## Group: linearalgebra

## Syntax

`trace(array)`

## Description

.

## Notes

.

## Examples

.

## See Also

`sum`

## vec

## Type: M-File

## Group: linearalgebra

## Syntax

`vec(value)`

## Description

.

## Notes

.

## Examples

.

## See Also

`sum`

## vech

## Type: M-File

## Group: LinearAlgebra

## Syntax

`vech()`

## Description

.

## Examples

---

# Chapter 39. matrix

## abs

**Type:** External

**Group:** matrix

### Syntax

```
abs(value)
```

### Description

Returns the absolute positive value of value.

### Examples

```
abs(-5)      = 5
abs(2)       = 2
abs(3 + 4I)  = 5
```

### See Also

sign, angle

## adjoint

**Type:** External

**Group:** matrix

### Syntax

```
answer = adjoint(square matrix)
```

### Description

Returns the adjoint matrix for the supplied matrix.

### Examples

## See Also

`inversematrix`, `inv`

# all

**Type:** External

**Group:** matrix

## Syntax

```
answer = all(matrix)
```

## Description

Returns 1 if all of the elements of the supplied matrix are not zero.

## Examples

```
all([0,0;0,0]) = 0 all([1,2;2i,-6]) = 1
```

## See Also

`any`, `and`, `not`, `abs`

# and

**Type:** External

**Group:** matrix

## Syntax

```
answer = AND(matrix1, matrix2)
```

## Description

Returns the boolean and of all the elements of the two matrices.

## Examples

```
and([1,1;1,1], [0,0;0,0]) = [0,0;0,0]  
and([1,2], [2,1]) = [1,1]
```

## See Also

not, or, xor

## any

**Type: External**

**Group: matrix**

## Syntax

```
answer = any(matrix)
```

## Description

Returns 1 if any of the elements of the supplied matrix are not zero.

## Examples

```
any([0,0;0,0]) = 0  
any([1,0;0,0]) = 1
```

## See Also

and, eye, all

## chol

**Type: External**

**Group: matrix**

## Syntax

```
answer=CHOL(matrix)
```

## Description

Calculates the Cholewsky decomposition of matrix

## Examples

```
CHOL([2,3;3,5])=[1.414,0;2.121,0.707]
```

# col

**Type:** M-File

**Group:** matrix

## Syntax

```
col(matrix, n)
```

## Description

returns the specified column from a matrix.

## Notes

.

## Examples

```
col([1,2;3,4], 1) = [1;3]  
col([1,2;3,4], 2) = [2;4]
```

## See Also

columns, rows, row

# columns

**Type:** M-File

**Group:** matrix

## Syntax

```
answer = columns (value)
```

## Description

.

## Notes

.



## Examples

.

## See Also

[col](#), [row](#), [rows](#)

## ctranspose

## Type: M-File

## Group: matrix

## Syntax

```
answer = template (value)
```

## Description

.

## Notes

.

## Examples

.

## See Also

[transpose](#)

## cumprod

## Type: External

## Group: Matrix

## Syntax

```
ans = cumprod(array)
```

## Description

return the cummulated product of an array

## Examples

```
cumsum([1,2,3;4,5,6])
ans = [1 , 2 , 3]
      [4 , 10 , 18]
```

## See Also

cumsum, sum, diag, trace, zeros, ones

## cumsum

### Type: External

### Group: Matrix

### Syntax

```
ans = cumsum(array)
```

### Description

return the cummulated sum of an array

## Examples

```
cumsum([1,2,3;4,5,6;7,8,9])
ans = [1 , 2 , 3]
      [5 , 7 , 9]
      [12 , 15 , 18]
```

## See Also

sum, diag, trace, zeros, ones

## det

### Type: M-File

### Group: Matrix

### Syntax

```
det(A)
```

## Description

## Examples

## See Also

## determinant

**Type:** External

**Group:** matrix

## Syntax

```
answer = DETERMINANT(square matrix)
```

## Description

Returns the determinant for the first operand which must be a square matrix.

## Examples

```
DETERMINANT([1,0;0,1]) = 1 DETERMINANT([1,2;3,4]) = -2
```

## diag

**Type:** External

**Group:** Matrix

## Syntax

```
answer = diag (value)
```

## Description

Returns the diagonal of a matrix

## Notes

## Examples

```
diag([1,2,3;4,5,6;7,8,9]
[1,
5,
9]

diag([2,3,4;5,6,7],1)
[3,
7]
```

## See Also

sum, trace, zeros, ones

## eig

### Type: External

### Group: matrix

### Syntax

```
eig(matrix)
```

### Description

Calculates the eigenvalues of the matrix.

## Examples

```
eig([1,2;3,4])=[-0.372;5.372]
```

## elementat

### Type: External

### Group: matrix

### Syntax

```
element = ELEMENTAT(matrix, rowno, colno)
```

### Description

Returns the values in the specified position of the matrix.

## Examples

`ELEMENTAT([1,2;3,4],0,0) = 1` `ELEMENTAT([1,2;3,4],1,1) = 4`

## See Also

### eq

Type: M-File

Group: Matrix

## Syntax

`eq(A,B)`

## Description

.

## Examples

## See Also

`ge`, `gt`, `lt`, `le`, `ne`

### eye

Type: External

Group: matrix

## Syntax

`matrix = eye(size)`

## Description

returns a matrix with the specified size.

## Examples

`eye(2) = [1,0;0,1]`

```
eye(3) = [1,0,0;0,1,0;0,0,1]
```

## See Also

ones, zeros

## find

### Type: External

### Group: matrix

### Syntax

```
find(matrix)  
find(matrix, n)
```

### Description

Finds all the non zero elements of a matrix

### Examples

```
find([0,1,0,1]) = [2;4]  
find([0,1,0,1],1) = [2]
```

## See Also

## fliplr

### Type: External

### Group: matrix

### Syntax

```
fliplr(A)
```

### Description

Flips a matrix from left to right.

### Examples

```
fliplr([1,2,3;4,5,6]) = [3,2,1;6,5,4]
```

## See Also

`flipud`

# flipud

**Type: External**

**Group: matrix**

## Syntax

`flipud(A)`

## Description

Flips a matrix from up to down.

## Examples

`flipud([1,2,3;4,5,6]) = [4,5,6;1,2,3]`

## See Also

`fliplr`

# floor

**Type: External**

**Group: matrix**

## Syntax

`floor(value)`

## Description

Rounds the value of the first operand down to the nearest integer.

## Examples

`floor(-5.5) = -6 floor(2.3) = 2`

## See Also

`ceil`, `round`

## ge

**Type: M-File**

**Group: Matrix**

**Syntax**

`ge(A,B)`

**Description**

.

**Examples**

**See Also**

`eq, gt, lt, le, ne`

## gt

**Type: M-File**

**Group: Matrix**

**Syntax**

`gt(A,B)`

**Description**

.

**Examples**

**See Also**

`eq, ge, lt, le, ne`

## inf



**Type: External**

**Group: matrix**

**Syntax**

```
answer = nan(sizey, [sizey])
```

**Description**

Returns a matrix filled with infinity.

**Examples**

```
nan(1,2) -> [Inf, Inf]
```

**See Also**

ones, zeros, nan

**inv**

**Type: M-File**

**Group: Matrix**

**Syntax**

```
inv(A)
```

**Description**

.

**Examples**

**See Also**

**inversematrix**

**Type: External**

## Group: matrix

### Syntax

```
answer=INVERSEMATRIX(square matrix)
```

### Description

Returns the inverse of the first operand, which must be a square matrix.

### Examples

```
INVERSEMATRIX([1,0;0,1]) = [1,0;0,1]
```

### See Also

adjoint, simultaneousseq

## isempty

### Type: External

## Group: matrix

### Syntax

```
answer=ISEMPTY(value)
```

### Description

Checks if a matrix is empty (no number or string)

### Examples

```
isempty([]) -> 1
```

```
a=[]  
isempty(a) -> 1
```

### See Also

isfinite, isimaginary, isreal, isnan, isinf

## isfinite

### Type: External

## Group: matrix

### Syntax

```
answer = isfinite(matrix)
```

### Description

.

### Examples

```
isfinite([5,Inf,-Inf,NaN]) -> ans = [1 , 0 , 0 , 0]
```

### See Also

isreal, isnan, isinf

## isimaginary

### Type: External

## Group: matrix

### Syntax

```
answer = isimaginary(matrix)
```

### Description

Returns 1 if any of the elements of the supplied matrix are not zero.

### Examples

```
any([0,0;0,0]) = 0 any([1,0;0,0]) = 1
```

### See Also

any

## isinf

### Type: External

## Group: matrix

### Syntax

```
answer = isinf(matrix)
```

### Description

.

### Examples

```
isinf([3,4;Inf,9]) -> [0,0;1,0]
```

### See Also

isreal, isnan, isfinite

## isnan

### Type: External

## Group: matrix

### Syntax

```
answer = isnan(matrix)
```

### Description

.

### Examples

```
isnan([3,4;NaN,9]) -> [0,0;1,0]
```

### See Also

isreal, isfinite, isnan

## isreal

### Type: External

## Group: matrix

## Syntax

```
answer = isreal(matrix)
```

## Description

Returns 1 if any of the elements of the supplied matrix are not zero.

## Examples

```
isreal([0,2i]) = 0 isreal([11,0]) = 1 isreal([]) = 1
```

## See Also

isnan, isimaginary

## ldivide

### Type: M-File

### Group: Matrix

## Syntax

```
ldivide(A,B)
```

## Description

.

## Examples

## See Also

## le

### Type: M-File

### Group: Matrix

## Syntax

```
le(A,B)
```

## Description

## Examples

## See Also

eq, ge, gt, lt, ne

## ln

## Type: External

## Group: matrix

## Syntax

```
answer = ln(value)
```

## Description

Returns the natural logarithm of value.

## Examples

```
ln(2) = 0.6931471805599  
ln(10) = 2.30258509299
```

## See Also

log

## log

## Type: External

## Group: matrix

## Syntax

```
answer = log(value, base)
```

## Description

Returns the logarithm of value to the base base.

## Examples

## See Also

[ln](#)

## lowertriangle

**Type:** External

**Group:** matrix

## Syntax

```
answer = LOWERTRIANGLE(matrix)
```

## Description

Converts a square matrix into it's lower triangular form.

## Examples

```
LOWERTRIANGLE([1,2;3,4]) = [-0.5, 0; 3, 4]
```

## See Also

[uppertriangle](#)

## lt

**Type:** M-File

**Group:** Matrix

## Syntax

```
lt(A,B)
```

## Description

.

## Examples

## See Also

eq, ge, gt, le, ne

## lu

## Type: External

## Group: matrix

## Syntax

```
[L,U]      = lu (A)
[L,U,P]    = lu (A)
[L]        = lu (A)
x          = lu (A,B) as a solution to A*X = B;
```

## Description

Returns the LU decomposition of a matrix

## Notes

For an m-by-n matrix A with  $m \geq n$ , the LU decomposition is an m-by-n unit lower triangular matrix L, an n-by-n upper triangular matrix U, and a permutation vector piv of length m so that  $A(\text{piv},:) = L*U$ . If  $m < n$ , then L is m-by-m and U is m-by-n. The LU decomposition with pivoting always exists, even if the matrix is singular, so the constructor will never fail. The primary use of the LU decomposition is in the solution of square systems of simultaneous linear equations. This will fail if `isNonsingular()` returns false.

## See Also

qr

## magic

## Type: External

## Group: matrix

## Syntax

magic(n)



## Description

creates a magic matrix with equal sums of all rows and column

## Examples

```
> magic(3) ans = [8,1,6] [3,5,7] [4,9,2]
```

## See Also

## max

## Type: External

## Group: matrix

## Syntax

```
answer = max(matrix)
answer = max(matrix,matrix)
answer = max(scalar,matrix)
answer = max(matrix,scalar)
```

## Description

returns the maximum of a matrix

## Examples

```
max(3,4) return 3 max([2,3,4],3) return [2,3,4]
```

## See Also

[min](#)

## min

## Type: External

## Group: matrix

## Syntax

```
answer = min(matrix)
```

```
answer = min(matrix, matrix)
answer = min(scalar, matrix)
answer = min(matrix, scalar)
```

## Description

returns the minimum of a matrix

## Examples

```
min(3,4) return 3 min([2,3,4],3) return [2,3,4]
```

## See Also

[max](#)

## minus

### Type: M-File

### Group: Matrix

## Syntax

```
minus(A,B)
```

## Description

.

## Examples

## See Also

## mldivide

### Type: M-File

### Group: Matrix

## Syntax

```
mldivide(A,B)
```

## Description

.

## Examples

## See Also

[mrdivide](#)

## mpower

Type: M-File

Group: Matrix

## Syntax

`mpower(A,B)`

## Description

.

## Examples

## See Also

## mrdivide

Type: M-File

Group: Matrix

## Syntax

`mrdivide(A,B)`

## Description

.

## Examples

## See Also

`mldivide`

## mtimes

**Type:** M-File

**Group:** Matrix

## Syntax

`mtimes(A,B)`

## Description

.

## Examples

## See Also

`plus`, `minus`

## nan

**Type:** External

**Group:** matrix

## Syntax

`answer = nan(sizey, [sizex])`

## Description

Returns a matrix filled with NaN.

## Examples

```
nan(1,2) -> [NaN, NaN]
```

## See Also

ones, zeros, inf

## ne

### Type: M-File

### Group: Matrix

### Syntax

```
ne(A,B)
```

### Description

.

### Examples

## See Also

eq, ge, gt, lt, le

## nnz

### Type: External

### Group: matrix

### Syntax

```
answer=NNZ(matrix)
```

### Description

Calculates the number of non zero elements within a matrix.

### Examples

```
NNZ([1,0,2;0,0,5]) = 3
```

# not

**Type: External**

**Group: matrix**

## Syntax

```
answer = not(matrix)
```

## Description

Returns the boolean NOT of all the elements of a matrix.

## Examples

```
not([false, true]) = [true, false]
not([1,1;1,1])     = [0,0;0,0]
not([0,1,2])       = [1,0,0]
```

## See Also

and, or, xor

# numel

**Type: External**

**Group: matrix**

## Syntax

```
numel(array)
```

## Description

returns the number of elements in array

## Examples

```
numel([2,5,3]) -> 3
```

# ones

**Type: External**

**Group: matrix**

## Syntax

```
ones(sizeX, sizeY)  
ones(n,m,k,...)
```

## Description

Returns a matrix of ones.

## Examples

```
ISNUMERIC(2,3) = [1,1,1;1,1,1] ISNUMERIC(3,4) = [1,1,1,1;1,1,1,1;1,1,1,1]
```

## See Also

[zeros](#)

**or**

**Type: External**

**Group: matrix**

## Syntax

```
answer = OR(matrix1, matrix2)
```

## Description

Returns the boolean or of two matrices.

## Examples

```
OR([1,1,1,1],[0,0;0,0]) = [1,1;1,1]
```

## See Also

[and](#), [not](#)

**plus**

**Type: M-File**

**Group: Matrix**

**Syntax**

`plus(A,B)`

**Description**

.

**Examples**

**See Also**

**pow2**

**Type: External**

**Group: matrix**

**Syntax**

`answer = POW2(matrix)`

**Description**

Returns 2 to the power of all the elements in a matrix.

**Examples**

`POW2([1,2;3,4]) = [1,4;8,16]`

**See Also**

`prod, power, mpower`

**power**

**Type: M-File**



## Group: Matrix

### Syntax

```
power(A,B)
```

### Description

.

### Examples

### See Also

## prod

### Type: External

### Group: matrix

### Syntax

```
answer = PROD(matrix)
```

### Description

Returns the products of the rows of a matrix.

### Examples

```
PROD([1,2;3,4]) = [3,8] PROD([1,2,3;4,5,6;7,8,9]) = [28, 80, 162]
```

### See Also

```
pow2, cumprod
```

## qr

### Type: External

### Group: matrix

## Syntax

```
QR(matrix)
```

## Description

Calculates the QR decomposition of matrix.

## Notes

For an m-by-n matrix A with  $m \geq n$ , the QR decomposition is an m-by-n orthogonal matrix Q and an n-by-n upper triangular matrix R so that  $A = Q \cdot R$ . The QR decomposition always exists, even if the matrix does not have full rank, so the constructor will never fail. The primary use of the QR decomposition is in the least squares solution of nonsquare systems of simultaneous linear equations. This will fail if `isFullRank()` returns false.

## See Also

`chol`, `lu`

## rdivide

### Type: M-File

### Group: Matrix

## Syntax

```
rdivide(A,B)
```

## Description

.

## Examples

## See Also

`mldivide`, `mrdivide`

## repmat

### Type: External

### Group: matrix

## Syntax

```
matrix = repmat(matrix to copy, no of rows, no of columns)
```

## Description

Copies part of a matrix.

## Examples

```
repmat([1,2,3;4,5,6;7,8,9], 2, 2) = [1,2;4,5]
```

## See Also

[reshape](#)

# reshape

## Type: External

## Group: matrix

## Syntax

```
reshape(matrix to reshape, no of rows, no of columns)
```

## Description

Reshapes a matrix.

## Examples

```
reshape([1,2,3;4,5,6;7,8,9], 2, 2,) = [1,2;4,5]
```

## See Also

[repmat](#)

# row

## Type: M-File

## Group: matrix

## Syntax

```
row(matrix, n)
```

## Description

Returns the specified row of a matrix.

## Examples

```
row([1,2;3,4],1) = [1,2]  
row([1,2;3,4],2) = [3,4]
```

## See Also

col, rows, columns

## ROWS

## Type: M-File

## Group: Matrix

## Syntax

```
rows(A)
```

## Description

.

## Examples

## See Also

row, col, columns

## simultaneouseq

## Type: External

## Group: matrix

## Syntax

```
solution = SIMULTANEOUSEQ(square matrix, results)
```

## Description

Calculates the solution to a series of simultaneous equations.

## Examples

```
equations 2x+3y=7 5x+y=11 SIMULTANEOUSEQ([2,3;5,1],[7,11]) = [2;1]
```

## See Also

## sort

## Type: External

## Group: matrix

## Syntax

```
sort(a)  
[x,y]=sort(A)
```

## Description

sorting of vectors and arrays.

## Examples

```
sort([6,2,9])    -> [2,6,9]  
sort([5,1;2,3]) -> [2,1; 5,3]  
[x,y]=sort([3,7,4;2,9,1]) -> x=[2,7,1;3,9,4]  y=[2,1,2;1,2,1]
```

## See Also

find

## subassign

## Type: External

## Group: matrix

## Syntax

```
subassign(matrix, replacement matrix, vertical limit, [horizontal limit])
```

## Description

Assigns a section of replacement matrix to matrix.

## Examples

```
subassign(a,b,1,1:2)
a=[2,3,4,5]; a(1,3)=77
```

## See Also

[submatrix](#)

## submatrix

### Type: External

### Group: matrix

## Syntax

```
answer = submatrix(matrix, sizey, sizex)
answer = submatrix(matrix, index)
```

## Description

Returns a portion of a matrix.

## Examples

```
a=[1,2,3,4,5;6,7,8,9,10;1,2,3,4,5] submatrix(a,1,2) = 2 submatrix(a,3,:) = [3,8,3]' submatrix(a,2:5,2) =
[7,8,9,10] submatrix(a,1:end,:) submatrix(a,[2,6,3,2])
```

## See Also

[subassign](#)

## sum

### Type: External

### Group: matrix

## Syntax

```
answer = sum(matrix)
answer = sum(structure)
```

## Description

Returns the sum of all the elements of a matrix or a structure.

## Examples

```
sum([1,2;3,4]) = 10
sum([1,2,3;4,5,6]) = 21
x=struct(a, 1, b, 2, c, 3) = a = 1 : b = 2 : c = 3 :
sum x = 6
```

## See Also

trace, diag

## sumsq

### Type: M-File

### Group: Matrix

## Syntax

```
sumsq(x,dim)
```

## Description

.

## Examples

## See Also

## svd

### Type: External

### Group: matrix

## Syntax

```
s = svd(A)
[U, S, V]=svd(A)
```

## Description

Calculates the single value decomposition of a matrix

## Notes

For an m-by-n matrix A with  $m \geq n$ , the singular value decomposition is an m-by-n orthogonal matrix U, an n-by-n diagonal matrix S, and an n-by-n orthogonal matrix V so that  $A = U*S*V'$ . The singular values,  $\sigma[k] = S[k][k]$ , are ordered so that  $\sigma[0] \geq \sigma[1] \geq \dots \geq \sigma[n-1]$ . The singular value decomposition always exists, so the constructor will never fail. The matrix condition number and the effective numerical rank can be computed from this decomposition.

## Examples

```
SVD([1,2,3;4,5,6;7,8,9]) = [16.848; 1.068; 0]
```

## See Also

lu, qr

## times

## Type: M-File

## Group: matrix

## Syntax

```
C=times(A,B)
```

## Description

.

## Notes

.

## Examples

.

## See Also



# transpose

**Type:** M-File

**Group:** matrix

## Syntax

`transpose(A)`

## Description

.

## Notes

`C=A.'`

## Examples

.

## See Also

`ctranspose`

# uminus

**Type:** M-File

**Group:** Matrix

## Syntax

`uminus(A)`

## Description

.

## Examples

## See Also

`plus`, `minus`

# uplus

**Type:** M-File

**Group:** Matrix

## Syntax

`uplus(A)`

## Description

.

## Examples

## See Also

`plus`, `minus`, `uminus`

# uppertriangle

**Type:** External

**Group:** matrix

## Syntax

`answer = UPPERTRIANGLE(matrix)`

## Description

Converts a matrix to upper triangular form.

## Examples

`UPPERTRIANGLE([1,2;3,4]) = [1,2;0,-2]`

## See Also

`lowertriangle`

# xor

**Type: External**

**Group: matrix**

## Syntax

```
answer = xor(matrix1, matrix2)
```

## Description

Returns the boolean xor of the elements of two matrices.

## Examples

```
xor([1,0;0,1], [1,1;0,0]) = [0,1;0,1]
```

## See Also

and, or

## zeros

**Type: External**

**Group: matrix**

## Syntax

```
zeros(sizey, [sizex])  
zeros(n,m,k,l...)
```

## Description

Returns a matrix filled with zeros.

## Examples

```
zeros(2,2) = [0,0;0,0]  
zeros(2,3) = [0,0,0;0,0,0]
```

## See Also

ones

---

# Chapter 40. miscellaneous

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### **ans**

**Type: M-File**

**Group: miscellaneous**

**Syntax**

ans

**Description**

.

**Examples**

### **comma**

**Type: M-File**

**Group: miscellaneous**

**Syntax**

comma

**Description**

.

**Examples**

### **flops**

**Type: M-File**

**Group: miscellaneous**

**Syntax**

`flops()`

**Description**

.

**Notes**

.

**Examples**

.

**See Also**

**mexext**

**Type: M-File**

**Group: miscellaneous**

**Syntax**

`mexext`

**Description**

.

**Examples**

**semicolon**

**Type: M-File**

## Group: miscellaneous

### Syntax

`semicolon`

### Description

.

### Examples

## single

### Type: M-File

## Group: miscellaneous

### Syntax

`single(x)`

### Description

.

### Examples

## texas\_lotto

### Type: M-File

## Group: miscellaneous

### Syntax

`texas_lotto`

### Description

Pick 6 unique numbers between 1 and 50 that are guaranteed to win.

## Examples

```
texas_lotto
ans = [5 , 16 , 30 , 33 , 34 , 42]
```

## See Also

---

# Chapter 41. net

## Introduction to the toolbox audio

There will be some great features for audio manipulations in the future.

### urlread

**Type: External**

**Group: net**

### Syntax

```
urlread( URL )
```

### Description

Read a text file from the network. This function supports many transfer protocols e.g. http, gopher, ftp, file. @EXAMPLE

```
urlread("http://www.heise.de/newsticker/");  
urlread("file:///c:/home/text.txt");
```

### Notes

.

### See Also

csvread, csvwrite



---

# Chapter 42. polynomial

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### binomial

**Type: External**

**Group: polynomial**

#### Syntax

```
answer = binomial(value)
```

#### Description

Calculates the binomial coefficients of  $(x+y)^{\text{value}}$ .

#### Examples

```
binomial(3) = [1, 3, 3, 1]
binomial(4) = [1, 4, 6, 4, 1]
```

#### See Also

poly, roots

### compan

**Type: M-File**

**Group: polynomial**

#### Syntax

```
A = compan(c)
```

#### Description

.

# mkpp

**Type: M-File**

**Group: polynomial**

## Syntax

```
pp = mkpp (x, P, d)
```

## Description

.

## See Also

poly, polyreduce, polyval, roots, unmkpp

# poly

**Type: M-File**

**Group: polynomial**

## Syntax

```
y = poly (x)
```

## Description

.

## See Also

mkpp, polyreduce, polyval, roots, unmkpp

# polyinteg

**Type: M-File**

**Group: polynomial**

## Syntax

```
y = polyinteg (x)
```

## Description

.

## See Also

poly, polyreduce, polyval, roots, unmkpp

# polyreduce

## Type: M-File

## Group: polynomial

## Syntax

```
y = polyreduce (x)
```

## Description

.

## See Also

poly, polyinteg, polyval, roots, unmkpp

# polyval

## Type: M-File

## Group: polynomial

## Syntax

```
y = polyval (c, x)
```

## Description

.

## See Also

poly, polyreduce, polyinteg, roots, unmkpp

## roots

**Type:** M-File

**Group:** polynomial

### Syntax

```
y = roots (x)
```

### Description

.

### See Also

poly, polyreduce, polyval, polyinteg, unmkpp

## unmkpp

**Type:** M-File

**Group:** polynomial

### Syntax

```
[x, P, n, k, d] = unmkpp (pp)
```

### Description

.

### See Also

poly, polyreduce, polyval, roots, polyinteg

---

# Chapter 43. quaternion

## qtransvmat

Type: M-File

Group: quaternion

### Syntax

```
qtransvmat([a,b,c,d])
```

### Description

.

### Examples

### See Also

---

# Chapter 44. set

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### complement

Type: M-File

Group: set

#### Syntax

```
x=complement(y)
```

#### Description

.

### create\_set

Type: M-File

Group: set

#### Syntax

```
x=create_set(y)
```

#### Description

.

---

# Chapter 45. signal

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### bartlett

**Type:** M-File

**Group:** signal

#### Syntax

```
answer = bartlett (value)
```

#### Description

.

#### Notes

.

#### Examples

.

#### See Also

blackman, hamming, hanning

### blackman

**Type:** M-File

**Group:** signal

#### Syntax

```
answer = blackman (value)
```

#### Description

.

## Notes

.

## Examples

.

## See Also

bartlett, hamming, hanning

## durbinlevinson

### Type: M-File

### Group: signal

### Syntax

`durbinlevinson()`

### Description

.

## Notes

.

## Examples

.

## See Also

bartlett, hamming, hanning

## hamming

### Type: M-File

### Group: signal

### Syntax

`answer = template (value)`



## Description

.

## Notes

.

## Examples

.

## See Also

[bartlett](#), [blackman](#), [hanning](#)

# hanning

## Type: M-File

## Group: signal

## Syntax

`hanning()`

## Description

.

## Notes

.

## Examples

.

## See Also

[bartlett](#), [blackman](#), [hamming](#)

# hurst

## Type: M-File

## Group: signal

### Syntax

`hurst()`

### Description

.

### Notes

.

### Examples

.

## rectangle\_lw

### Type: M-File

## Group: signal

### Syntax

`rectangle_lw`

### Description

.

### Notes

Rectangular lag window. Subfunction used for spectral density estimation.

### Examples

.

### See Also

`rectangle_sw`

## rectangle\_sw

**Type: M-File**

**Group: signal**

**Syntax**

`rectangle_sw`

**Description**

.

**Notes**

Rectangular spectral window. Subfunction used for spectral density estimation.

**Examples**

.

**See Also**

`rectangle_lw`

**sinc**

**Type: M-File**

**Group: signal**

**Syntax**

`sinc()`

**Description**

.

**Notes**

.

**Examples**

.

**See Also**

## **sinewave**

**Type:** M-File

**Group:** signal

### **Syntax**

`sinewave()`

### **Description**

.

### **Notes**

.

### **Examples**

.

### **See Also**

## **triangle\_lw**

**Type:** M-File

**Group:** signal

### **Syntax**

`triangle_lw()`

### **Description**

.

### **Notes**

.

### **Examples**

.

# triangle\_sw

**Type:** M-File

**Group:** signal

## Syntax

```
triangle_sw()
```

## Description

.

## Notes

.

## Examples

.

---

# Chapter 46. specfun

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### beta

**Type:** M-File

**Group:** specfun

### Syntax

`beta(a, b)`

### Description

.

### Examples

`beta(2, 3)`

### See Also

`gammaln`, `betaln`

### betaln

**Type:** M-File

**Group:** specfun

### Syntax

`betaln(a, b)`

### Description

.

## Examples

```
betaln(2, 3)
```

## See Also

`gammaln`, `beta`

## `gammaln`

### Type: External

### Group: specfun

## Syntax

```
gammaln(x)
```

## Description

return the log of the gamma function

## Examples

```
gammaln(3.5)  
gammaln([2,3,4;5,6,7])
```

## See Also

`beta`

## `perms`

### Type: M-File

### Group: specfun

## Syntax

```
perms([2,7,4])
```

## Description

.

## Notes

.

## Examples

.

## See Also



---

# Chapter 47. specialmatrix

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### hankel

**Type: M-File**

**Group: specialmatrix**

#### Syntax

```
hankel()
```

#### Description

#### See Also

`rosser`

### hilb

**Type: M-File**

**Group: specialmatrix**

#### Syntax

```
answer = template (value)
```

#### Description

#### See Also

`rosser`

### lauchli

**Type: M-File**

**Group: specialmatrix**

**Syntax**

```
lauchli(x)
lauchli(x,mu)
```

**Description**

Creates the matrix [ ones(1,n); mu\*eye(n) ]

**See Also**

**pascal**

**Type: M-File**

**Group: specialmatrix**

**Syntax**

```
pascal()
```

**Description**

**See Also**

rosser

**rosser**

**Type: M-File**

**Group: specialmatrix**

**Syntax**

```
answer = template (value)
```

## Description

.

## Notes

.

## Examples

.

## See Also

[hilb](#)

## **sylvester\_matrix**

**Type: M-File**

**Group: specialmatrix**

## Syntax

```
retval = sylvester_matrix (k)
```

## Description

.

## **toeplitz**

**Type: M-File**

**Group: specialmatrix**

## Syntax

```
retval = toeplitz (c, r)
```

## Description

.

## **wilkinson**

**Type: M-File**

**Group: specialmatrix**

**Syntax**

```
retval = wilkinson(n)
```

**Description**

.

---

# Chapter 48. statistics

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### average

**Type: External**

**Group: statistics**

### Syntax

```
average(1, 2, 3, ..., n)
```

### Description

Calculates the average of the passed parameters.

### Examples

```
Average(a, b, 3, 4)
```

### See Also

std, mean, var

### center

**Type: M-File**

**Group: statistics**

### Syntax

```
center(x, dim)
```

### Description

Subtract center value along the chosen dimension

## Examples

```
center([2,3,5,3],2)
```

## See Also

std, var, cov

## cloglog

### Type: M-File

### Group: statistics

## Syntax

```
cloglog(x)
```

## Description

Return the complementary log-log function of

## Examples

```
cloglog(8)
```

## See Also

std, var, cov, log, ln

## COV

### Type: M-File

### Group: statistics

## Syntax

```
cov(x,y)  
cov(x)
```

## Description

Calculates the covariance

## Examples

```
cov(x, y)
```

## See Also

std, var, mean

## mean

### Type: M-File

### Group: statistics

## Syntax

```
mean([2, 3, 8, ..., n])
```

## Description

Calculates the mean value of the passed operand.

## Examples

```
mean([a, b, 3, 4])  
mean([3, 4, 5]) -> 4
```

## See Also

std, var, meansq

## meansq

### Type: M-File

### Group: statistics

## Syntax

```
meansq([2, 3, 8], dim)
```

## Description

Calculates the square mean value of the passed operand.

## Examples

```
meansq([4,2])
```

## See Also

std, var, mean, cov

## std

### Type: M-File

### Group: statistics

## Syntax

```
std([2,3,8,...])
```

## Description

Calculates the standard deviation.

## Examples

```
std([a,b,3,4])  
std([3,4,5]) -> 1
```

## See Also

mean, var

## var

### Type: M-File

### Group: statistics

## Syntax

```
var([2,3,8,...])
```

## Description

Calculates the variance.



## Examples

```
var([3,4,5]) -> 1
```

## See Also

mean, std

## variation

### Type: External

### Group: statistics

## Syntax

```
Variation(1,2,3,...,n)
```

## Description

Calculates the variation of the parameters.

## Examples

```
Variation(0,x,y,z,3,t)
```

## See Also

var, std

---

# Chapter 49. system

## addpath

**Type:** External

**Group:** system

### Syntax

```
addpath(path)
```

### Description

Adds path to the current search path.

### Examples

```
addpath("../newpath")
```

### See Also

path, rmpath

## checkforupdates

**Type:** External

**Group:** system

### Syntax

```
checkForUpdates()  
checkForUpdates(site)  
checkForUpdates("-silent")
```

### Description

This functions checks via network if the current installation of JMathLib is up to date. This functions is also called during startup of JMathLib's GUI. @EXAMPLE checkforupdates()

### Notes

This functions checks via network if the current installation of JMathLib is up to date. This functions is also called during startup of JMathLib's GUI.

## See Also

update

# createfunctionslist

**Type: External**

**Group: system**

## Syntax

```
createfunctionslist()
```

## Description

Creates a file called webFunctionList.dat containing the name of all external functions. This functions is used when running JMathLib as an applet.

## Examples

```
createfunctionslist()
```

## See Also

path, rmpath, update, checkforupdates

# dbquit

**Type: M-File**

**Group: system**

## Syntax

```
dequit
```

## Description

stops debugging messages immediately

## Examples

```
setdebug(1)  
...
```

```
some commands
...
dbquit
```

## See Also

setdebug

## debug

**Type: External**

**Group: system**

## Syntax

```
DEBUG(expression)
```

## Description

This evaluates expression, displaying the op tree which is created.

## Examples

```
DEBUG("3*5")=15
```

## disp

**Type: External**

**Group: system**

## Syntax

```
DISP(string1, string2, string3, ...)
```

## Description

This displays a set of lines of text.

## Examples

```
DISP("Hello", "world") Hello world
```

## error

**Type: External**

**Group: system**

## Syntax

```
error(message)
```

## Description

Displays message and aborts the current operation.

## Examples

```
Error("There has been an error") There has been an error
```

## See Also

warning, exit

## exit

**Type: M-File**

**Group: system**

## Syntax

```
exit
```

## Description

```
exits JMathLib @EXAMPLE
```

```
exit
```

## Notes

.

## See Also

quit

## foreach

**Type: External**

**Group: system**

## Syntax

```
FOREACH(Matrix, Variable, Expression)
```

## Description

executes an expression on each element of a matrix.

## Examples

```
FOREACH([1,2;3,4],"x","DISP(x)") 1 2 3 4
```

## format

**Type: External**

**Group: system**

## Syntax

```
format('type')
```

## Description

changes the numerical format for numbers.

## Notes

It only affects the display of numbers, the internal format of numbers is not affected at all.

## Examples

```
format() format('long') format('short') format('long e') format('short e') format('long g') format('short g') format('long eng') format('short eng')
```

## See Also

disp

## getdebug

**Type: External**

**Group: system**

## Syntax

```
getdebug( )
```

## Description

returns state of debug logging

## Examples

```
getdebug()
```

## See Also

setdebug

# getenv

**Type: External**

**Group: system**

## Syntax

```
GETENV(variablename)
```

## Description

Returns the value of the enviromental variable variablename.

## Examples

```
GETENV("HOME")= "/home/user"
```

# getjmathlibproperty

**Type: External**

**Group: system**

## Syntax

```
getjmathlibproperty(variablename)
```

## Description

Returns the value of the enviromental variable variablename.

## Examples

```
getlocal("HOME")= "/home/user"
```

## See Also

getenv, setjmathlibproperty

# getproperty

## Type: M-File

## Group: system

## Syntax

```
getproperty(name)
```

## Description

returns value of system property

## Examples

```
getproperty("os.name")
```

```
ans="Windows XP"
```

## See Also

getenv

# jmathlibcreateuniqueid

## Type: External

## Group: system



## Syntax

```
jmathlibcreateuniqueid
```

## Description

create a unique id for jmathlib

## See Also

update, checkforupdates

# nargchk

## Type: External

## Group: system

## Syntax

```
nargchk(min, max, no of input args)
```

## Description

Checks that the number of input args is between min and max

## Examples

```
NARGOUTCHK(1, 3, NARGIN)
```

## See Also

nargoutchk

# nargoutchk

## Type: External

## Group: system

## Syntax

```
nargoutchk(min, max, no of output args)
```

## Description

Checks that the number of output args is between min and max

## Examples

```
NARGOUTCHK(1, 3, NARGOUT)
```

## See Also

nargchk

## newline

**Type:** External

**Group:** system

## Syntax

```
newline(lines)
```

## Description

Displays number of blank lines equal to the first parameter.

## Examples

```
NEWLINE(1) NEWLINE(2)
```

## path

**Type:** External

**Group:** system

## Syntax

```
path()
```

## Description

Lists the current search path.

## Examples

```
path()
```

## See Also

`addpath`, `rmpath`, `createfunctionslist`

## `print_usage`

**Type:** External

**Group:** system

## Syntax

`print_usage`

## Description

print out the valid usage of the calling function

## See Also

`warning`, `error`

## `printstacktrace`

**Type:** External

**Group:** system

## Syntax

`printstacktrace(message)`

## Description

Displays message and the current execution stack trace.

## Examples

```
Error("There has been an error") There has been an error
```

## See Also

`warning`, `error`

## `quit`

**Type: External**

**Group: system**

## Syntax

```
quit
```

## Description

```
exits JMathLib @EXAMPLE
```

```
quit
```

## Notes

`quit("force")` will terminate JMathLib without saving any variables or saving any changed settings.

## See Also

`exit`

# rehash

**Type: External**

**Group: system**

## Syntax

```
rehash
```

## Description

rehashes all m-files in the cache for user functions

## Examples

```
rehash
```

## See Also

`path`, `rmpath`, `addpath`

# rmpath

**Type: External**

**Group: system**

## Syntax

```
rmpath(path)
```

## Description

Removes path from the current search path

## Notes

Using `rmpath` will stop the system from finding any external functions and M-files stored within the path.

## Examples

```
rmpath("./Functions/General")
```

## See Also

`addpath`, `path`

# setdebug

**Type: External**

**Group: system**

## Syntax

```
setdebug(value)
```

## Description

Switches debug output on or off

## Examples

`setdebug(1)` turns debug output on `setdebug(0)` turns debug output off

## See Also

`getdebug`

# setjmathlibproperty

**Type: External**

**Group: system**

## Syntax

```
setjmathlibproperty(property name, value)
```

## Description

Returns the value of the enviromental variable variablename.

## Examples

```
GETENV("HOME")= "/home/user"
```

## See Also

getjmathlibproperty, getenv

# update

**Type: External**

**Group: system**

## Syntax

```
update
```

## Description

updates JMathLib over the web

## Notes

The script first checks the webserver in order to determine, if it is possible to update the current version and if yes which version is next. It also updates local properties of JMathlib.

## Examples

```
update
```

## See Also

`checkforupdates`

## usage

**Type:** External

**Group:** system

## Syntax

`USAGE(message)`

## Description

Used within script files to display the paramaters for the script.

## Examples

`Usage("Function(paramaters)")`

## See Also

`error`, `warning`

## ver

**Type:** External

**Group:** system

## Syntax

`ver`

## Description

detailed version information about toolboxes @EXAMPLE

`ver`

## Notes

.

## See Also

`version`

## version

**Type: External**

**Group: system**

## Syntax

`version`

## Description

displays current version of JMathLib @EXAMPLE

`version`

## Notes

.

## See Also

`ver`

## warning

**Type: External**

**Group: system**

## Syntax

`warning(message)`

## Description

Displays a warning message

## Notes

This function sets the lastwarning variable



## Examples

```
warning("Danger, danger") Danger, danger
```

## See Also

error, exit

---

# Chapter 50. test

## test\_complex

Type: M-File

Group: test

### Syntax

```
TEST_COMPLEX(real1, im1, re2, im2)
```

### Description

A set of routines for testing complex numebers

### Examples

```
TEST_COMPLEX(1, 1, 2, 2)
```

## test\_for

Type: M-File

Group: test

### Syntax

```
TEST_FOR(limit)
```

### Description

A function to test test the for command. It adds all integers up to limit.

### Examples

```
TEST_COMPLEX(4) x=0 y=0 x=1 y=1 x=2 y=3 x=3 y=6 x=4 y=10
```

## test\_graph

Type: M-File

## Group: test

### Syntax

```
TEST_Graph()
```

### Description

A function to test basic graph drawing.

### Examples

```
TEST_Graph()
```

## test\_matlabfor

### Type: M-File

## Group: test

### Syntax

```
TEST_MATRIXFOR()
```

### Description

A function to test the for command using the matrix syntax.

### Examples

```
TEST_MATRIXFOR()
```

## test\_standard

### Type: M-File

## Group: test

### Syntax

```
TEST_STANDARD(val1, va2, val3)
```

### Description

A set of routines for testing standard operations

## Examples

```
TEST_STANDARD(1, 2, 3)
```

## test\_switch

**Type: M-File**

**Group: test**

## Syntax

```
TEST_SWITCH(value)
```

## Description

A function to test the switch command.

## Examples

```
TEST_SWITCH(2) your choice is x = 2 TEST_SWITCH(5) invalid x 5
```

## test\_trig

**Type: M-File**

**Group: test**

## Syntax

```
TEST_TRIG(value)
```

## Description

A function to test the trigonometric functions.

## Examples

```
TEST_TRIG(2)
```

## testfunctionfor001

**Type: M-File**

## Group: test

### Syntax

```
testFunctionFor001(limit)
```

### Description

A function to test the for command. It adds all integers up to limit.

### Examples

```
TEST_COMPLEX(4) x=0 y=0 x=1 y=1 x=2 y=3 x=3 y=6 x=4 y=10
```

## testfunctionfor002

### Type: M-File

## Group: test

### Syntax

```
TEST_MATRIXFOR( )
```

### Description

A function to test the for command using the matrix syntax.

### Examples

```
TEST_MATRIXFOR()
```

## testfunctionswitch002

### Type: M-File

## Group: test

### Syntax

```
TEST_SWITCH(value)
```

### Description

A function to test the switch command.

## Examples

TEST\_SWITCH(2) your choice is x = 2 TEST\_SWITCH(5) invalid x 5

# testfunctionwhile001

**Type: M-File**

**Group: test**

## Syntax

TEST\_FOR(limit)

## Description

A function to test the while command. It adds all integers up to limit.

## Examples

TEST\_COMPLEX(4) x=0 y=0 x=1 y=1 x=2 y=3 x=3 y=6 x=4 y=10

---

# Chapter 51. time

## Introduction to the toolbox audio

There will be some great features for this toolbox in the future.

### date

**Type:** External

**Group:** time

### Syntax

```
data = date()
```

### Description

Returns the current date.

### Examples

```
d=date()  
returns  
d= 7-7-2002
```

### See Also

tic, toc, pause, time

### is\_leap\_year

**Type:** M-File

**Group:** time

### Syntax

```
is_leap_year()
```

### Description

Return 1 if the given year is a leap year and 0 otherwise

## Examples

```
is_leap_year()
```

## See Also

tic, toc, pause, time

## pause

### Type: External

### Group: time

### Syntax

```
pause(value)
```

### Description

Wait for a specified period of time. The command `pause(n)` waits for `n` seconds

### Notes

Keep in mind that the parsing of the `pause(x)` command will also take some time. Therefore the waiting time will be longer than expected.

## Examples

```
pause(3) waits for 3 seconds  
pause(3.5) waits for 3.5 seconds
```

## See Also

tic, toc, date, time

## tic

### Type: External

### Group: time

### Syntax

```
tic()
```



## Description

Starts the stop watch.

## Examples

```
tic()  
toc()  
ans = 1.34
```

## See Also

toc, pause, date, time

## time

### Type: External

### Group: time

## Syntax

```
t = time()
```

## Description

Returns the elapsed miliseconds from 1.1.1970.

## Examples

```
t=time()
```

## See Also

tic, pause, date, time

## toc

### Type: External

### Group: time

## Syntax

```
toc()
```

## Description

Returns the time difference in seconds between the call to `tic()` and `toc()`.

## Examples

```
tic()
toc()
returns
ans = 1.34
```

## See Also

`tic`, `pause`, `date`, `time`

---

# Chapter 52. trigonometric

## Introduction to the toolbox trigonometric

This toolbox is intended to provide functions in order to compute trigonometric functions.

### **acos**

**Type: External**

**Group: trigonometric**

### **Syntax**

```
angle = acos(value)
```

### **Description**

Returns the arc cosine of value.

### **Examples**

```
acos(1) = 0  
acos(0) = 1.5707963267948966
```

### **See Also**

cos, acosh, sin, asin, asinh

### **acosd**

**Type: M-File**

**Group: trigonometric**

### **Syntax**

```
angle = acosd(value)
```

### **Description**

Compute the inverse cosine of an angle in degrees.

## Examples

.

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## acosh

### Type: External

### Group: trigonometric

### Syntax

`angle = ACOS(value)`

### Description

Returns the arc cosine of value.

## Examples

`ACOS(1) = 0` `ACOS(0) = 1.5707963267948966`

## See Also

`cos`, `acosh`

## acot

### Type: External

### Group: trigonometric

### Syntax

`acot(value)`

### Description

calculates the inverse cotangent

## Notes

## Examples

$$y = \operatorname{acot}(x)$$

## See Also

`sec`, `cot`, `csch`, `sech`, `coth`

## acotd

## Type: M-File

## Group: trigonometric

## Syntax

$$\text{angle} = \operatorname{acotd}(\text{value})$$

## Description

Compute the inverse cotangent of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## acoth

## Type: External

## Group: trigonometric

## Syntax

$$\operatorname{acoth}(\text{value})$$

## Description

calculates the inverse hperbolic cotangent

## Examples

```
y = acoth(x)
```

## See Also

sec, cot, csch, sech, coth

## acsc

## Type: External

## Group: trigonometric

## Syntax

```
acsc(value)
```

## Description

calculates the inverse cosecans

## Examples

```
.
```

## See Also

sec, cot, csch, sech, coth

## acscd

## Type: M-File

## Group: trigonometric

## Syntax

```
angle = acsd(value)
```

## Description

Compute the inverse cosecant of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## acsch

## Type: External

## Group: trigonometric

## Syntax

`acsch(value)`

## Description

calculates the inverse hperbolic cosecant

## Examples

$y = \operatorname{acsch}(x)$

## See Also

`sec`, `cot`, `csch`, `sech`, `coth`

## asec

## Type: External

## Group: trigonometric

## Syntax

`asec(value)`

## Description

calculates the inverse secant

## Examples

```
y = asec(x)
```

## See Also

sec, cot, csch, sech, coth

## asecd

## Type: M-File

## Group: trigonometric

## Syntax

```
angle = asecd(value)
```

## Description

Compute inverse secant in degrees.

## Examples

## See Also

cos, acosh, sin, asin, asinh

## asech

## Type: External

## Group: trigonometric

## Syntax

```
asech(value)
```



## Description

calculates the inverse hperbolic secant

## Examples

```
y = asech(x)
```

## See Also

sec, cot, csch, sech, coth

## asin

### Type: External

### Group: trigonometric

## Syntax

```
angle=asin(value)
```

## Description

Returns the arc sine of the first operand.

## Examples

```
asin(1) = 1.5707963267948966  
asin(0) = 0
```

## See Also

sin, asinh, cos, acos, acosh

## asind

### Type: M-File

### Group: trigonometric

## Syntax

```
angle = asind(value)
```

## Description

Compute the inverse sine of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## asinh

## Type: External

## Group: trigonometric

## Syntax

```
angle = asinh(value);
```

## Description

Returns the arc hyperbolic sine

## Examples

```
asinh(1) = 0.8813735870195429  
asinh(0) = 0
```

## See Also

`sinh`, `asin`, `cosh`, `acos`

## atan

## Type: External

## Group: trigonometric

## Syntax

```
atan(angle)
```

## Description

Calculates the inverse tangent of angle.

## Examples

```
atan(1) = 0.7853981633974483  
atan(0) = 0
```

## See Also

atan2, tan

## atan2

### Type: Internal

### Group: trigonometric

## Syntax

```
answer = ATAN2(value1, value2)
```

## Description

Returns the arc tangent of the gradient represented by the value1 / value2.

## Examples

```
ATAN(2, 1) = 0.4636476090008061 ATAN(-2, 1) = 2.677945044588981
```

## See Also

atan, tan

## atand

### Type: M-File

### Group: trigonometric

## Syntax

```
angle = atand(value)
```

## Description

Compute the inverse tangent of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## `atanh`

## Type: External

## Group: trigonometric

## Syntax

`atanh(value)`

## Description

Returns the arc hyperbolic tangent of the first value.

## Examples

```
atanh(1) = NaN
atanh(0) = 0
```

## See Also

`tan`, `tanh`

## `COS`

## Type: External

## Group: trigonometric

## Syntax

`cos(angle)`

## Description

Returns the cosine of angle.

## Examples

$\cos(0) = 1$   $\cos(1.5707963267948966) = 0$

## See Also

[acos](#), [cosh](#)

## cosd

### Type: M-File

### Group: trigonometric

## Syntax

`angle = cosd(value)`

## Description

Compute the cosine of an angle in degrees.

## Examples

## See Also

[cos](#), [acosh](#), [sin](#), [asin](#), [asinh](#)

## cosh

### Type: External

### Group: trigonometric

## Syntax

`cosh(angle)`

## Description

Returns the hyperbolic cosine of angle.

## Examples

```
cosh(0) = 1
cosh(1.5707963267948966) = 2.5091784786580567
```

## See Also

cos, acosh, acos, sin, asin, asinh

## cot

### Type: External

### Group: trigonometric

### Syntax

```
cot(value)
```

### Description

.

### Examples

.

## See Also

sec, csc, csch, sech, coth

## cotd

### Type: M-File

### Group: trigonometric

### Syntax

```
angle = cotd(value)
```

### Description

Compute the cotangent of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## `coth`

**Type:** External

**Group:** trigonometric

## Syntax

`coth(value)`

## Description

.

## Examples

.

## See Also

`sec`, `csc`, `csch`, `sech`, `cot`

## `CSC`

**Type:** External

**Group:** trigonometric

## Syntax

`csc(value)`

## Description

.

## Examples

.

## See Also

sec, cot, csch, sech, coth

## cscd

### Type: M-File

### Group: trigonometric

### Syntax

`angle = cscd(value)`

### Description

Compute the cosecant of an angle in degrees.

### Examples

## See Also

cos, acosh, sin, asin, asinh

## csch

### Type: External

### Group: trigonometric

### Syntax

`cscd(value)`

### Description

.

### Examples

.

## See Also

sec, cot, csch, sech, coth



# degtograd

**Type: External**

**Group: trigonometric**

## Syntax

`gradients = degtograd(degrees)`

## Description

converts the angle from degrees to gradients.

## Examples

`degtograd(180) = 200 degtograd(90) = 100`

## See Also

`gradtodeg, degtograd, radtograd, degtorad, radtodeg`

# degtorad

**Type: External**

**Group: trigonometric**

## Syntax

`radians = DEGTORAD(degrees)`

## Description

converts the angle from degrees to radians.

## Examples

`DEGTORAD(180) = 3.141592653589793 DEGTORAD(90) = 1.5707963267948966`

## See Also

`gradtodeg, degtograd, radtograd, degtorad, radtodeg`

# gradtodeg

**Type: External**

**Group: trigonometric**

## Syntax

`degrees = gradtodeg(gradients)`

## Description

converts the angle from gradients to degrees.

## Examples

`gradtodeg(200) = 180` `gradtodeg(100) = 100`

## See Also

`gradtodeg`, `degtograd`, `radtograd`, `degtorad`, `radtodeg`

# gradtorad

**Type: External**

**Group: trigonometric**

## Syntax

`radians = gradtorad(gradients)`

## Description

converts the angle from gradients to radians.

## Examples

`gradtorad(200) = 3.141592653589793` `gradtorad(100) = 1.5707963267948966`

## See Also

`gradtodeg`, `degtograd`, `radtograd`, `degtorad`, `radtodeg`

# radtodeg

**Type: External**

## Group: trigonometric

### Syntax

`degrees = radtodeg(radians)`

### Description

converts the angle from radians to degrees.

### Examples

`radtodeg(3.141592653589793) = 180` `radtodeg(1.5707963267948966) = 90`

### See Also

`gradtodeg`, `degtograd`, `radtograd`, `degtorad`, `radtodeg`

## radtograd

### Type: External

## Group: trigonometric

### Syntax

`gradients = radtograd(radians)`

### Description

converts the angle from radians to gradients.

### Examples

`radtograd(3.141592653589793) = 200` `radtograd(1.570796326794896) = 100`

### See Also

`degtograd`, `radtograd`, `degtorad`, `radtodeg`

## sec

### Type: External

## Group: trigonometric

## Syntax

`sec(value)`

## Description

.

## Examples

.

## See Also

`csc`, `cot`, `csch`, `sech`, `coth`

## secd

## Type: M-File

## Group: trigonometric

## Syntax

`angle = secd(value)`

## Description

Compute the secant of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## sech

## Type: External

## Group: trigonometric

## Syntax

`sech(value)`

## Description

## Examples

## See Also

csc, cot, csch, sec, coth

## sin

## Type: External

## Group: trigonometric

## Syntax

```
answer = sin(angle)
```

## Description

Returns the sine of angle.

## Examples

```
sin(0) = 0 sin(1.5708) = 1
```

## See Also

asin, sinh

## sind

## Type: M-File

## Group: trigonometric

## Syntax

```
angle = sind(x)
```

## Description

Compute the sine of each element of x.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## `sinh`

**Type:** External

**Group:** trigonometric

## Syntax

`sinh(angle)`

## Description

Returns the hyperbolic sine of angle.

## Examples

```
sinh(0) = 0
sinh(1) = 1.175201
```

## See Also

`sin`, `asinh`, `cos`, `cosh`, `acosh`

## `tan`

**Type:** External

**Group:** trigonometric

## Syntax

`answer = tan(angle)`

## Description

Returns the tangent of angle.

## Examples

$\tan(0) = 0$   $\tan(1) = 1.55740772$

## See Also

`atan`, `tanh`

## tand

### Type: M-File

### Group: trigonometric

## Syntax

`angle = tand(value)`

## Description

Compute the tangent of an angle in degrees.

## Examples

## See Also

`cos`, `acosh`, `sin`, `asin`, `asinh`

## tanh

### Type: External

### Group: trigonometric

## Syntax

`tanh(angle)`

## Description

Returns the hyperbolic tangent of angle.

## Examples

$\tanh(0) = 0$   
 $\tanh(1) = 0.76159$

## See Also

[tan](#), [atanh](#), [cos](#), [sin](#)



---

# Chapter 53. ui

## questdlg

**Type:** External

**Group:** ui

### Syntax

```
button = questdlg(question, title, button1, button2, options);
```

### Description

Rounds the element to the nearest element towards zero.

### Examples

```
button = questdlg("do you like JMathLib","What about JMathLib","yes","even mor"
```

### See Also

---

# Appendix A. GNU Lesser General Public License - GNU Project - Free Software Foundation (FSF)

## GNU Lesser General Public License

This GNU Lesser General Public License counts as the successor of the GNU Library General Public License. For an explanation of why this change was necessary, read the Why you shouldn't use the Lesser GPL for your next library article.

### Table of Contents

- GNU LESSER GENERAL PUBLIC LICENSE
  - Preamble
  - TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
  - How to Apply These Terms to Your New Libraries

## GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.  
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA  
Everyone is permitted to copy and distribute verbatim copies  
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts  
as the successor of the GNU Library Public License, version 2, hence  
the version number 2.1.]

### Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software

(and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former

contains code derived from the library, whereas the latter must be combined with the library in order to run.

## TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

**0.** This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

**1.** You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

**2.** You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- **a)** The modified work must itself be a software library.
- **b)** You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- **c)** You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- **d)** If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

**3.** You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

**4.** You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

**5.** A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

**6.** As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- **a)** Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- **b)** Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- **c)** Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- **d)** If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- **e)** Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

**7.** You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- **a)** Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- **b)** Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

**8.** You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties

who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

**9.** You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

**10.** Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

**11.** If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

**12.** If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

**13.** The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

**14.** If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

## **NO WARRANTY**

**15.** BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

**16.** IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

## How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
    one line to give the library's name and an idea of what it does.
Copyright (C)
    year
    name of author
```

```
This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.
```

```
This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.
```

```
You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:



Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker. signature of Ty Coon, 1 April 1990 Ty Coon, President of Vice

That's all there is to it! Return to GNU's home page [<http://www.gnu.org>].

FSF & GNU inquiries & questions to [gnu@gnu.org](mailto:gnu@gnu.org) [<mailto:gnu@gnu.org>]. Other ways to contact [<http://www.gnu.org>] the FSF.

Comments on these web pages to [webmasters@www.gnu.org](mailto:webmasters@www.gnu.org) [<mailto:webmasters@www.gnu.org>], send other questions to [gnu@gnu.org](mailto:gnu@gnu.org) [<mailto:gnu@gnu.org>].

Copyright notice above.

Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111, USA

---

# Appendix B. Frequently Asked Questions

These are the most frequently asked questions from the users of JMathLib.

## B.1. General Information

These are some general solutions.

### B.1.1. General Items

**Q:** What is the purpose of JMathLib?

**A:** JMathLib is designed to be a clone of Matlab [<http://www.mathworks.com/>] or Octave [<http://www.gnu.org/software/octave/>] or Scilab [<http://www.scilab.org>] or Freemath [<http://freemat.sourceforge.net>].

**Q:** Where can I obtain JMathLib?

**A:** Look at the website [<http://www.jmathlib.de>] or look at the download page [<http://www.jmathlib.de/download.php>].

**Q:** How do I exit JMathLib?

**A:** Just type `exit` in the command window.

### B.1.2. Installation of JMathLib

**Q:** Installation on Windows?

**A:** Just download the installer and double-click on the EXE file.

**Q:** Installation on Linux?

**A:** ...

### B.1.3. Starting of JMathLib

**Q:** How to start JMathLib?

**A:** On windows: just double click the program icon

## B.2. Graphics

These are some solutions to questions related to the graphical system of JMathLib.

### B.2.1. Toolboxes

These are some answers regarding toolboxes

**Q:** What is a toolbox?

**A:** A toolbox is a collection of `.m` and `.class` files that work on some specialized area. For example: plot functions are in the graphics toolbox. Functions for reading and writing of data from an the you harddisc are located the io-toolbox.

**Q:** How do I plot a vector?

A: `plot(name_of_vector)`

---

# Appendix C. History

## History

A lot has been done already

- 0.7.0: added printing of graphs

## Old versions

- 0.6.0: Windows Installer, support for cell arrays and many many bugfixes.
- 0.x.x: lots of changes in between
- 0.3.4: Seperated GUI class
- 0.3.3.5: Control structures and Matrix functions added
- 0.3.3.2: Improved support for Numeric matrices

Some complex number functions

- 0.3.3.1: Support for matrices added
  - 0.3.3: Added support for complex numbers
  - 0.3.2.2: Improved factorisation so  $(x+1)*(x+1)$  returns  $x^2 + 2x + 1$
  - 0.3.2.1: Modified ErrorLogger class so that log lines get written to a file as well as the console
  - 0.3.2: Improved factorisation of expressions so  $2*x + x$  returns  $3*x$
  - 0.3.1.5: Split binary operators into seperate classes
  - 0.3.1: Added support for strings
  - 0.3.0.2: Changed way standard and trigonometric functions
  - 0.3.0.1: Corrected bug when displaying expressions, caused  $y - 1$  to appear as  $1 - y$
  - 0.3.0: Modified GUI
  - 0.2.4.1: corrected a bug which prevented the letter z being used in variable names.
  - 0.2.4: Seperated the lexical analysis code into a seperate object
  - 0.2.3: Added support for user functions
  - 0.2.2: Standard Mathematical functions
- Program closes when you click on the x
- use up & down arrows to access previous commands
- 0.2.1: Added trigonometric functions
  - 0.2.0: Rudimentary error checking + new GUI;
  - 0.1.4: Add variables support

- 0.1.3: Factorial operator, !, added
- 0.1.2: New operator ^ for calculating x to the yth power
- 0.1.1: Replaced the Binary tree with a tree where each node has a variable number of children. It doesn't have much effect on the program at the moment but it should make implementing functions easier.
- 0.1: It handles simple operations involving the four main operators and any real numbers. It understands precedence and brackets.

---

# Appendix D. To Do

## Open

JMathLib is available for a couple of years already, but still ..... a lot needs to be done.

## Closed

Some simple graphics stuff is already available.

---

# Index

## Symbols

`__abctddims__`, 51  
`_char`, 39  
`_class`, 68  
`_double`, 39

## A

`abctddim`, 51  
`abs`, 160  
`acos`, 254  
`acosd`, 254  
`acosh`, 255  
`acot`, 255  
`acotd`, 256  
`acoth`, 256  
`acsc`, 257  
`acscd`, 257  
`acsch`, 258  
`addpath`, 229  
`adjoint`, 160  
`aes`, 56  
`all`, 161  
`and`, 161  
`angle`, 68  
`ans`, 199  
`any`, 162  
`asec`, 258  
`asecd`, 259  
`asech`, 259  
`asin`, 260  
`asind`, 260  
`asinh`, 261  
`atan`, 261  
`atan2`, 262  
`atand`, 262  
`atanh`, 263  
`autumn`, 127  
`average`, 224  
`axis2dlim`, 52

## B

`bartlett`, 210  
`beep`, 69  
`bench`, 69  
`beta`, 217  
`betaln`, 217  
`binomial`, 204  
`bitand`, 70  
`bitor`, 70  
`bitshift`, 71  
`bitxor`, 71  
`blackman`, 210  
`blanks`, 40  
`bone`, 127

`brighten`, 128

## C

`cd`, 140  
`ceil`, 72  
`cell`, 72  
`center`, 224  
`char`, 40  
`checkforupdates`, 229  
`chol`, 162  
`cla`, 113  
`clear`, 73  
`clf`, 113  
`clock`, 73  
`cloglog`, 225  
`close`, 114  
`col`, 163  
`colormap`, 128  
`columns`, 163  
`combinations`, 74  
`comma`, 199  
`commutation_matrix`, 154  
`compan`, 204  
`complement`, 209  
`complex`, 74  
`conj`, 75  
`control toolbox`, 51  
`cool`, 129  
`copper`, 129  
`cos`, 263  
`cosd`, 264  
`cosh`, 264  
`cot`, 265  
`cotd`, 265  
`coth`, 266  
`cov`, 225  
`create_set`, 209  
`createfunctionslist`, 230  
`createnewfile`, 140  
`crule`, 136  
`csc`, 266  
`cscd`, 267  
`csch`, 267  
`csvread`, 141  
`csvwrite`, 141  
`ctranspose`, 164  
`cumprod`, 164  
`cumsum`, 165

## D

`date`, 250  
`dbquit`, 230  
`deblank`, 41  
`debug`, 231  
`degtograd`, 268  
`degtorad`, 268  
`delete`, 142

det, 165  
determinant, 166  
diag, 166  
diff, 75  
dir, 142  
disp, 231  
dmult, 154  
dot, 154  
double, 41  
duplication\_matrix, 155  
durbinlevinson, 211

## E

e, 76  
eig, 167  
elementat, 167  
engine, 60  
eps, 76  
eq, 168  
error, 231  
euler, 66  
exist, 143  
exit, 232  
exp, 77  
eye, 168

## F

factor, 77  
false, 78  
feval, 66  
fibonacci, 78  
figure, 114  
find, 169  
findstr, 41  
finite, 58  
fix, 79  
fliplr, 169  
flipud, 170  
floor, 170  
flops, 199  
foreach, 232  
format, 233  
freemat, 151  
ftp, 65  
func2str, 79  
fv, 61  
fvl, 61

## G

gammaln, 218  
gca, 115  
gcf, 115  
ge, 171  
get, 115  
getdebug, 233  
getenv, 234  
getjmathlibproperty, 234

getpfilecaching, 80  
getproperty, 235  
ginput, 116  
global, 80  
gradtodeg, 268  
gradtorad, 269  
gray, 130  
gray2ind, 130  
grid, 116  
gt, 171

## H

hamming, 211  
hankel, 220  
hanning, 212  
harmonic, 81  
hilb, 220  
hold, 117  
hot, 131  
housh, 155  
hurst, 212

## I

i, 81  
imag, 82  
inf, 171  
int16, 83  
int32, 83  
int64, 84  
int8, 84  
inv, 172  
inversematrix, 172  
is\_abcd, 52  
is\_leap\_year, 250  
is\_sample, 53  
isa, 85  
iscell, 85  
ischar, 86  
isdirectory, 143  
isdoube, 86  
isempty, 173  
isfile, 144  
isfinite, 173  
isfunctionhandle, 87  
isglobal, 87  
ishidden, 144  
isimaginary, 174  
isinf, 174  
isint16, 88  
isint32, 88  
isint64, 89  
isint8, 90  
islogical, 90  
ismatrix, 91  
isnan, 175  
isnumeric, 91  
isprime, 92



isreal, 175  
isscalar, 92  
issingle, 93  
issquare, 94  
isstr, 58  
isstruct, 94  
isstudent, 95  
isuint16, 95  
isuint32, 96  
isuint64, 96  
isuint8, 97

## J

j, 97  
jet, 131  
jmathlib, 151  
jmathlibcreateuniqueid, 235

## K

kbhit, 98

## L

lastmodified, 145  
lauchli, 220  
ldivide, 176  
le, 176  
left, 42  
length, 98  
linspace, 99  
ln, 177  
load, 145  
loadvariables, 146  
log, 177  
logical, 99  
logm, 156  
lower, 42  
lowertriangle, 178  
lt, 178  
lu, 179

## M

magic, 179  
matlab, 152  
max, 180  
mean, 226  
meansq, 226  
meshgrid, 117  
mexext, 200  
mid, 43  
min, 180  
minus, 181  
minusminus, 138  
mkdir, 146  
mkpp, 205  
mldivide, 181  
mpower, 182  
mrdivide, 182

mtimes, 183

## N

nan, 183  
nargchk, 236  
nargoutchk, 236  
ncrle, 136  
ndims, 100  
ne, 184  
newline, 237  
nnz, 184  
norm, 156  
not, 185  
nper, 62  
ntsc2rgb, 132  
null, 156  
num2str, 44  
numel, 185

## O

octave, 152  
ones, 185  
or, 186  
orth, 157

## P

pascal, 221  
path, 237  
pause, 251  
performfunction, 101  
perms, 218  
permutations, 101  
pi, 101  
pink, 132  
plot, 118  
plot3, 119  
plotfunction, 120  
plus, 186  
plusplus, 138  
pmt, 62  
polar, 121  
poly, 205  
polyinteg, 205  
polyreduce, 206  
polyval, 206  
pow2, 187  
power, 187  
primes, 102  
print, 122  
print\_usage, 238  
printstacktrace, 238  
prod, 188  
propertyeditor, 122  
pv, 62  
pvl, 63  
pwd, 147

## Q

qr, 188  
qtransvmat, 208  
questdlg, 276  
quit, 238

## R

radtodeg, 269  
radtograd, 270  
rainbow, 133  
rand, 103  
rank, 157  
rdivide, 189  
real, 103  
rectangle\_lw, 213  
rectangle\_sw, 213  
rehash, 239  
repmat, 189  
reshape, 190  
return, 103  
right, 44  
rmdir, 147  
rmpath, 239  
roots, 207  
rosser, 221  
rotate, 123  
round, 104  
row, 190  
rows, 191  
runfile, 148

## S

savevariables, 149  
scilab, 153  
sec, 270  
secd, 271  
sech, 271  
semicolon, 200  
set, 123  
setdebug, 240  
setjmathlibproperty, 241  
setpfilecaching, 104  
sign, 105  
simultaneouseq, 191  
sin, 272  
sinc, 214  
sind, 272  
sinewave, 215  
single, 201  
sinh, 273  
size, 106  
sort, 192  
spring, 133  
sprintf, 44  
sqrt, 106  
ss2sys, 53  
std, 227

str2func, 107  
str2num, 45  
strcat, 46  
strcmp, 46  
strcmpi, 47  
strfind, 47  
strlength, 47  
strncmp, 48  
strncmpi, 48  
struct, 107  
strvcat, 49  
subassign, 192  
submatrix, 193  
subplot, 124  
substring, 49  
sum, 193  
summer, 134  
sumsq, 194  
surf, 112  
svd, 194  
swap, 54  
sylvester\_matrix, 222  
sysgettsam, 54  
sysgettype, 55  
systemcommand, 149

## T

tan, 273  
tand, 274  
tanh, 274  
template, 108  
test\_complex, 245  
test\_for, 245  
test\_graph, 245  
test\_matlabfor, 246  
test\_standard, 246  
test\_switch, 247  
test\_trig, 247  
testfunctionfor001, 247  
testfunctionfor002, 248  
testfunctionswitch002, 248  
testfunctionwhile001, 249  
texas\_lotto, 201  
tic, 251  
time, 252  
times, 195  
title, 124  
toc, 252  
toeplitz, 222  
toolbox audio, 60, 61, 65, 66, 68, 127, 136, 140, 151, 154, 199, 203, 204, 209, 210, 217, 220, 224, 250  
toolbox char, 39  
toolbox trigonometric, 254  
trace, 158  
transpose, 196  
triangle\_lw, 215  
triangle\_sw, 216  
true, 108

**U**

uint16, 109  
uint32, 109  
uint8, 110  
uminus, 196  
unmkpp, 207  
update, 241  
uplus, 197  
upper, 50  
uppertriangle, 197  
urlread, 203  
usage, 242

**V**

var, 227  
variation, 228  
vdp1, 57  
vdp2, 57  
vec, 158  
vech, 159  
ver, 242  
version, 243  
vol, 63

**W**

warning, 243  
white, 134  
who, 110  
whos, 111  
wilkinson, 222  
winter, 135

**X**

xlabel, 125  
xor, 197

**Y**

ylabel, 125

**Z**

zeros, 198  
zgfmul, 55  
zlabel, 126

# Glossary

This is the glossary of JMathLib.

**A**

AAA

Some reasonable definition here.

# B

BBB

Some reasonable definition here.

## C

CCC

Some reasonable definition here.

# D

DDD

Some reasonable definition here.

# E

# Extensible Markup Language

Some reasonable definition here.  
See Also Standard Generalized Markup Language.

**F**

FFF

Some reasonable definition here.

FreeMat

FreeMat ist another clone of Matlab.

## G

GGG

Some reasonable definition here.

## H

HHH

Some reasonable definition here.

## 1

### III

Some reasonable definition here.

## J

JJJ

Some reasonable definition here.

# K

KKK

Some reasonable definition here.

**L**

LLL Some reasonable definition here.

**M**

Matlab Matlab is the mother of everything.

m-File This a file e.g. foo.m which can contain a sript file or a JMathLib function written in script code.  
See Also Script-File.

**N**

NNN Some reasonable definition here.

**O**

Octave Octave [<http://www.gnu.org/software/octave/>] is another clone of Matlab [<http://www.mathworks.com>].  
See Also Scilab.

**P**

PPP Some reasonable definition here.

**Q**

QQQ Some reasonable definition here.

**R**

RRR Some reasonable definition here.

**S**

Scilab Scilab [<http://www.scilab.org>] is another clone of Matlab [<http://www.mathworks.com>].  
See Also Octave.

Script-File This a file e.g. foo.m which contains a sript file.  
See Also m-File.

SGML See Standard Generalized Markup Language.

Standard Generalized Markup Language Some reasonable definition here.  
See Also Extensible Markup Language.

**T**

TTT Some reasonable definition here.

