LIBRERIAS

OPS:

Operators and special characters.

Arithmetic operators.

plus	- Plus	+
uplus	- Unary plus	+
minus	- Minus	-
uminus	- Unary minus	-
mtimes	- Matrix multiply	*
times	- Array multiply	.*
mpower	- Matrix power	^
power	- Array power	.^
mldivide	- Backslash or left matrix divide	\
mrdivide	- Slash or right matrix divide	/
ldivide	- Left array divide	.\
rdivide	- Right array divide	./
kron	- Kronecker tensor product	kron

Relational operators.

eq	- Equal	==
ne	- Not equal	~=
lt	- Less than	<
gt	- Greater than	>
le	- Less than or equal	<=
ge	- Greater than or equal	>=

Logical operators.

relop	- Short-circuit logical AND	&&
relop	- Short-circuit logical OR	||
and	- Element-wise logical AND	&
or	- Element-wise logical OR	|
not	- Logical NOT	~
xor	- Logical EXCLUSIVE OR	
any	- True if any element of vector is nonzero	
all	- True if all elements of vector are nonzero	

Special characters.

colon	- Colon	:
paren	- Parentheses and subscripting	( )
paren	- Brackets	[ ]

```
  paren      - Braces and subscripting          { }
  punct      - Function handle creation          @
  punct      - Decimal point                     .
  punct      - Structure field access            .
  punct      - Parent directory                  ..
  punct      - Continuation                      ...
  punct      - Separator                         ,
  punct      - Semicolon                         ;
  punct      - Comment                           %
  punct      - Invoke operating system command   !
  punct      - Assignment                        =
  punct      - Quote                             '
  transpose  - Transpose                         .'
  ctranspose - Complex conjugate transpose       '
  horzcat    - Horizontal concatenation          [,]
  vertcat    - Vertical concatenation            [;]
  subsasgn   - Subscripted assignment            ( ),{ },.
  subsref    - Subscripted reference             ( ),{ },.
  subsindex  - Subscript index
  metaclass  - Metaclass for MATLAB class        ?

Bitwise operators.
  bitand     - Bit-wise AND.
  bitcmp     - Complement bits.
  bitor      - Bit-wise OR.
  bitmax     - Maximum floating point integer.
  bitxor     - Bit-wise XOR.
  bitset     - Set bit.
  bitget     - Get bit.
  bitshift   - Bit-wise shift.

Set operators.
  union      - Set union.
  unique     - Set unique.
  intersect  - Set intersection.
  setdiff    - Set difference.
  setxor     - Set exclusive-or.
  ismember   - True for set member.

See also arith, relop, slash, function_handle.
```

LANG:

Programming language constructs.

Control flow.

```
  if          - Conditionally execute statements.
  else        - Execute statement if previous IF condition failed.
  elseif      - Execute if previous IF failed and condition is true.
  end         - Terminate scope of control statements.
  for         - Repeat statements a specific number of times.
  parfor      - Parallel FOR-loop.
  while       - Repeat statements an indefinite number of times.
  break       - Terminate execution of WHILE or FOR loop.
  continue    - Pass control to the next iteration of a loop.
  switch      - Switch among several cases based on expression.
  case        - SWITCH statement case.
  otherwise   - Default SWITCH statement case.
  try         - Begin TRY block.
  catch       - Begin CATCH block.
  return      - Return to invoking function.
  error       - Display message and abort function.
  MException  - Constructs MATLAB exception object.
  assert      - Generate an error when a condition is violated.
  rethrow     - Reissue error.
```

Evaluation and execution.

```
  eval        - Execute string with MATLAB expression.
  evalc       - Evaluate MATLAB expression with capture.
  feval       - Execute the specified function.
  evalin      - Evaluate expression in workspace.
  builtin     - Execute built-in function from overloaded method.
  assignin    - Assign variable in workspace.
  run         - Run script.
```

Scripts, functions, classes, and variables.

```
  script      - About MATLAB scripts and M-files.
  function    - Add new function.
  global      - Define global variable.
  persistent  - Define persistent variable.
  mfilename   - Name of currently executing M-file.
  lists       - Comma separated lists.
  exist       - Check if variables or functions are defined.
```

mlock - Prevent M-file from being cleared.
munlock - Allow M-file to be cleared.
mislocked - True if M-file cannot be cleared.
precedence - Operator Precedence in MATLAB.
isvarname - True for valid variable name.
iskeyword - Check if input is a keyword.
javachk - Validate level of Java support.
genvarname - Construct a valid MATLAB variable name from a string.
classdef - Define a new class or subclass.

Argument handling.
nargchk - Validate number of input arguments.
nargoutchk - Validate number of output arguments.
nargin - Number of function input arguments.
nargout - Number of function output arguments.
varargin - Variable length input argument list.
varargout - Variable length output argument list.
inputname - Input argument name.
inputParser - Construct input parser object.

Message display.
warning - Display warning message.
lasterr - Last error message.
lasterror - Last error message and related information.
lastwarn - Last warning message.
disp - Display array.
display - Display array.
intwarning - Controls the state of the 4 integer warnings.

Interactive input.
input - Prompt for user input.
keyboard - Invoke keyboard from M-file.

Abstract superclasses.
handle - Superclass of all handle classes.
hgsetget - HG-style set and get for MATLAB objects.
dynamicprops - Support for dynamic properties.

See also GENERAL, IOFUN, OPS, DATATYPES, MATFUN, FUNFUN, ELFUN, POLYFUN.

Parallel computing programming language constructs.

Single program multiple data support.
  spmd   - Begin single program multiple data block.
  Composite   - Composite object used with SPMD block.

See also parfor, matlabpool.

ELMAT:

Elementary matrices and matrix manipulation.

Elementary matrices.
  zeros   - Zeros array.
  ones   - Ones array.
  eye   - Identity matrix.
  repmat   - Replicate and tile array.
  linspace   - Linearly spaced vector.
  logspace   - Logarithmically spaced vector.
  freqspace   - Frequency spacing for frequency response.
  meshgrid   - X and Y arrays for 3-D plots.
  accumarray - Construct an array with accumulation.
  :   - Regularly spaced vector and index into matrix.

Basic array information.
  size   - Size of array.
  length   - Length of vector.
  ndims   - Number of dimensions.
  numel   - Number of elements.
  disp   - Display matrix or text.
  isempty   - True for empty array.
  isequal   - True if arrays are numerically equal.
  isequalwithequalnans - True if arrays are numerically equal.

Matrix manipulation.
  cat   - Concatenate arrays.
  reshape   - Change size.
  diag   - Diagonal matrices and diagonals of matrix.
  blkdiag   - Block diagonal concatenation.
  tril   - Extract lower triangular part.
  triu   - Extract upper triangular part.
  fliplr   - Flip matrix in left/right direction.
  flipud   - Flip matrix in up/down direction.

flipdim - Flip matrix along specified dimension.
rot90 - Rotate matrix 90 degrees.
: - Regularly spaced vector and index into matrix.
find - Find indices of nonzero elements.
end - Last index.
sub2ind - Linear index from multiple subscripts.
ind2sub - Multiple subscripts from linear index.
bsxfun - Binary singleton expansion function.

Multi-dimensional array functions.
ndgrid - Generate arrays for N-D functions and interpolation.
permute - Permute array dimensions.
ipermute - Inverse permute array dimensions.
shiftdim - Shift dimensions.
circshift - Shift array circularly.
squeeze - Remove singleton dimensions.

Array utility functions.
isscalar - True for scalar.
isvector - True for vector.

Special variables and constants.
ans - Most recent answer.
eps - Floating point relative accuracy.
realmax - Largest positive floating point number.
realmin - Smallest positive floating point number.
pi - 3.1415926535897....
i - Imaginary unit.
inf - Infinity.
nan - Not-a-Number.
isnan - True for Not-a-Number.
isinf - True for infinite elements.
isfinite - True for finite elements.
j - Imaginary unit.
why - Succinct answer.

Specialized matrices.
compan - Companion matrix.
gallery - Higham test matrices.
hadamard - Hadamard matrix.
hankel - Hankel matrix.
hilb - Hilbert matrix.

invhilb - Inverse Hilbert matrix.
magic - Magic square.
pascal - Pascal matrix.
rosser - Classic symmetric eigenvalue test problem.
toeplitz - Toeplitz matrix.
vander - Vandermonde matrix.
wilkinson - Wilkinson's eigenvalue test matrix.

ELFUN:

Elementary math functions.

Trigonometric.
sin - Sine.
sind - Sine of argument in degrees.
sinh - Hyperbolic sine.
asin - Inverse sine.
asind - Inverse sine, result in degrees.
asinh - Inverse hyperbolic sine.
cos - Cosine.
cosd - Cosine of argument in degrees.
cosh - Hyperbolic cosine.
acos - Inverse cosine.
acosd - Inverse cosine, result in degrees.
acosh - Inverse hyperbolic cosine.
tan - Tangent.
tand - Tangent of argument in degrees.
tanh - Hyperbolic tangent.
atan - Inverse tangent.
atand - Inverse tangent, result in degrees.
atan2 - Four quadrant inverse tangent.
atanh - Inverse hyperbolic tangent.
sec - Secant.
secd - Secant of argument in degrees.
sech - Hyperbolic secant.
asec - Inverse secant.
asecd - Inverse secant, result in degrees.
asech - Inverse hyperbolic secant.
csc - Cosecant.
cscd - Cosecant of argument in degrees.
csch - Hyperbolic cosecant.
acsc - Inverse cosecant.

acscd - Inverse cosecant, result in degrees.
acsch - Inverse hyperbolic cosecant.
cot - Cotangent.
cotd - Cotangent of argument in degrees.
coth - Hyperbolic cotangent.
acot - Inverse cotangent.
acotd - Inverse cotangent, result in degrees.
acoth - Inverse hyperbolic cotangent.
hypot - Square root of sum of squares.

Exponential.
exp - Exponential.
expm1 - Compute exp(x)-1 accurately.
log - Natural logarithm.
log1p - Compute log(1+x) accurately.
log10 - Common (base 10) logarithm.
log2 - Base 2 logarithm and dissect floating point number.
pow2 - Base 2 power and scale floating point number.
realpow - Power that will error out on complex result.
reallog - Natural logarithm of real number.
realsqrt - Square root of number greater than or equal to zero.
sqrt - Square root.
nthroot - Real n-th root of real numbers.
nextpow2 - Next higher power of 2.

Complex.
abs - Absolute value.
angle - Phase angle.
complex - Construct complex data from real and imaginary parts.
conj - Complex conjugate.
imag - Complex imaginary part.
real - Complex real part.
unwrap - Unwrap phase angle.
isreal - True for real array.
cplxpair - Sort numbers into complex conjugate pairs.

Rounding and remainder.
fix - Round towards zero.
floor - Round towards minus infinity.
ceil - Round towards plus infinity.
round - Round towards nearest integer.
mod - Modulus (signed remainder after division).

rem - Remainder after division.
sign - Signum.

GRAPH2D:

Two dimensional graphs.

Elementary X-Y graphs.
plot - Linear plot.
loglog - Log-log scale plot.
semilogx - Semi-log scale plot.
semilogy - Semi-log scale plot.
polar - Polar coordinate plot.
plotyy - Graphs with y tick labels on the left and right.

Axis control.
axis - Control axis scaling and appearance.
zoom - Zoom in and out on a 2-D plot.
grid - Grid lines.
box - Axis box.
rbbox - Rubberband box.
hold - Hold current graph.
axes - Create axes in arbitrary positions.
subplot - Create axes in tiled positions.

Graph annotation.
plotedit - Tools for editing and annotating plots.
title - Graph title.
xlabel - X-axis label.
ylabel - Y-axis label.
texlabel - Produces the TeX format from a character string.
text - Text annotation.
gtext - Place text with mouse.

Hardcopy and printing.
print - Print graph or Simulink system; or save graph to M-file.
printopt - Printer defaults.
orient - Set paper orientation.

See also graph3d, specgraph.

FUNCIONES DEL PROGRAMADOR

FUNCTION:

FUNCTION Add new function.

New functions may be added to MATLAB's vocabulary if they are expressed in terms of other existing functions. The commands and functions that comprise the new function must be put in a file whose name defines the name of the new function, with a filename extension of '.m'. At the top of the file must be a line that contains the syntax definition for the new function. For example, the existence of a file on disk called STAT.M with:

```
function [mean,stdev] = stat(x)
%STAT Interesting statistics.
n = length(x);
mean = sum(x) / n;
stdev = sqrt(sum((x - mean).^2)/n);
```

defines a new function called STAT that calculates the mean and standard deviation of a vector. The variables within the body of the function are all local variables. See SCRIPT for procedures that work globally on the work-space.

A subfunction that is visible to the other functions in the same file is created by defining a new function with the FUNCTION keyword after the body of the preceding function or subfunction. For example, avg is a subfunction within the file STAT.M:

```
function [mean,stdev] = stat(x)
%STAT Interesting statistics.
n = length(x);
mean = avg(x,n);
stdev = sqrt(sum((x-avg(x,n)).^2)/n);

%-------------------------
function mean = avg(x,n)
%AVG subfunction
mean = sum(x)/n;
```

Subfunctions are not visible outside the file where they are defined.

You can terminate any function with an END statement but, in most cases, this is optional. END statements are required only in M-files that employ one or more nested functions. Within such an M-file, every function (including primary, nested, private, and subfunctions) must be terminated with an END statement. You can terminate any function type with END, but doing so is not required unless the M-file contains a nested function.

Normally functions return when the end of the function is reached. A RETURN statement can be used to force an early return.

See also script, return, varargin, varargout, nargin, nargout, inputname, mfilename.

Reference page in Help browser
doc function

RETURN:

RETURN Return to invoking function.
RETURN causes a return to the invoking function or to the keyboard. It also terminates the KEYBOARD mode.

Normally functions return when the end of the function is reached. A RETURN statement can be used to force an early return.

Example

```
function d = det(A)
if isempty(A)
  d = 1;
  return
else
  ...
end
```

See also function, keyboard, break, continue.

Reference page in Help browser
doc return

CICLOS

WHILE:

WHILE Repeat statements an indefinite number of times.
  The general form of a WHILE statement is:

```
WHILE expression
 statements
END
```

  The statements are executed while the real part of the expression
  has all non-zero elements. The expression is usually the result of
  expr rop expr where rop is ==, <, >, <=, >=, or ~=.

  The BREAK statement can be used to terminate the loop prematurely.

  For example (assuming A already defined):

```
E = 0*A; F = E + eye(size(E)); N = 1;
while norm(E+F-E,1) > 0,
  E = E + F;
  F = A*F/N;
  N = N + 1;
end
```

  See also for, if, switch, break, continue, end.

  Reference page in Help browser
    doc while

FOR:

FOR   Repeat statements a specific number of times.
  The general form of a FOR statement is:

```
FOR variable = expr, statement, ..., statement END
```

  The columns of the expression are stored one at a time in
  the variable and then the following statements, up to the
  END, are executed. The expression is often of the form X:Y,
  in which case its columns are simply scalars. Some examples

(assume N has already been assigned a value).

```
for R = 1:N
  for C = 1:N
    A(R,C) = 1/(R+C-1);
  end
end
```

Step S with increments of -0.1

```
for S = 1.0: -0.1: 0.0, do_some_task(S), end
```

Set E to the unit N-vectors

```
for E = eye(N), do_some_task(E), end
```

Long loops are more memory efficient when the colon expression appears in the FOR statement since the index vector is never created.

The BREAK statement can be used to terminate the loop prematurely.

See also parfor, if, while, switch, break, continue, end, colon.

Reference page in Help browser

```
doc for
```

BREAK:

BREAK Terminate execution of WHILE or FOR loop.

BREAK terminates the execution of FOR and WHILE loops.
In nested loops, BREAK exits from the innermost loop only.

BREAK is not defined outside of a FOR or WHILE loop.
Use RETURN in this context instead.

See also for, while, return, continue.

Reference page in Help browser

```
doc break
```

CONTINUE

CONTINUE Pass control to the next iteration of FOR or WHILE loop.

CONTINUE passes control to the next iteration of FOR or WHILE loop

in which it appears, skipping any remaining statements in the body of the FOR or WHILE loop.

In nested loops, CONTINUE passes control to the next iteration of FOR or WHILE loop enclosing it.

See also for, while, break, return.

Reference page in Help browser
doc continue

CONDICIONALES

IF:

IF Conditionally execute statements.
The general form of the IF statement is

```
IF expression
  statements
ELSEIF expression
  statements
ELSE
  statements
END
```

The statements are executed if the real part of the expression has all non-zero elements. The ELSE and ELSEIF parts are optional. Zero or more ELSEIF parts can be used as well as nested IF's. The expression is usually of the form expr rop expr where rop is ==, <, >, <=, >=, or ~=.

Example

```
if I == J
  A(I,J) = 2;
elseif abs(I-J) == 1
  A(I,J) = -1;
else
  A(I,J) = 0;
end
```

See also relop, else, elseif, end, for, while, switch.

Reference page in Help browser
doc if

ELSE:

ELSE Used with IF.
ELSE is used with IF. The statements after the ELSE are executed
if all the preceding IF and ELSEIF expressions are false.

The general form of the IF statement is
IF expression
statements
ELSEIF expression
statements
ELSE
statements
END

See also if, elseif, end.

Reference page in Help browser
doc else

ELSEIF:

ELSEIF IF statement condition.
ELSEIF is used with IF. The statements after the ELSEIF are
executed if the expression is true and all the preceding IF and
ELSEIF expressions are false. An expression is considered true if
the real part has all non-zero elements.

ELSEIF does not need a matching END, while ELSE IF does.

The general form of the IF statement is
IF expression
statements
ELSEIF expression
statements
ELSE
statements

See also if, else, end.

Reference page in Help browser
doc elseif

GRAFICOS

PLOT:

PLOT Linear plot.
PLOT(X,Y) plots vector Y versus vector X. If X or Y is a matrix, then the vector is plotted versus the rows or columns of the matrix, whichever line up. If X is a scalar and Y is a vector, disconnected line objects are created and plotted as discrete points vertically at X.

PLOT(Y) plots the columns of Y versus their index.
If Y is complex, PLOT(Y) is equivalent to PLOT(real(Y),imag(Y)).
In all other uses of PLOT, the imaginary part is ignored.

Various line types, plot symbols and colors may be obtained with PLOT(X,Y,S) where S is a character string made from one element from any or all the following 3 columns:

b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star	(none)	no line
y	yellow	s	square		
k	black	d	diamond		
w	white	v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

For example, PLOT(X,Y,'c+:') plots a cyan dotted line with a plus at each data point; PLOT(X,Y,'bd') plots blue diamond at each data point but does not draw any line.

PLOT(X1,Y1,S1,X2,Y2,S2,X3,Y3,S3,...) combines the plots defined by the (X,Y,S) triples, where the X's and Y's are vectors or matrices and the S's are strings.

For example, PLOT(X,Y,'y-',X,Y,'go') plots the data twice, with a solid yellow line interpolating green circles at the data points.

The PLOT command, if no color is specified, makes automatic use of the colors specified by the axes ColorOrder property. The default ColorOrder is listed in the table above for color systems where the default is blue for one line, and for multiple lines, to cycle through the first six colors in the table. For monochrome systems, PLOT cycles over the axes LineStyleOrder property.

If you do not specify a marker type, PLOT uses no marker.
If you do not specify a line style, PLOT uses a solid line.

PLOT(AX,...) plots into the axes with handle AX.

PLOT returns a column vector of handles to lineseries objects, one handle per plotted line.

The X,Y pairs, or X,Y,S triples, can be followed by parameter/value pairs to specify additional properties of the lines. For example, PLOT(X,Y,'LineWidth',2,'Color',[.6 0 0]) will create a plot with a dark red line width of 2 points.

Example

```
  x = -pi:pi/10:pi;
  y = tan(sin(x)) - sin(tan(x));
  plot(x,y,'--rs','LineWidth',2,...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor','g',...
                'MarkerSize',10)
```

See also plottools, semilogx, semilogy, loglog, plotyy, plot3, grid, title, xlabel, ylabel, axis, axes, hold, legend, subplot, scatter.

Overloaded methods:

```
  timeseries/plot
  phytree/plot
  clustergram/plot
```

channel.plot
sfit/plot
cfit/plot
fints/plot
idmodel/plot
idfrd/plot
iddata/plot
idnlhw/plot
idnlarx/plot
cgrules/plot
xregtwostage/plot
xregtransient/plot
xregmodel/plot
xregarx/plot
localmulti/plot
localmod/plot
localavfit/plot
sweepset/plot
mdevtestplan/plot
cgdatasetnode/plot
mpc/plot
rfckt.plot
frd/plot
dspdata.plot
wdectree/plot
ntree/plot
dtree/plot
wvtree/plot
rwvtree/plot
edwttree/plot

Reference page in Help browser
doc plot

TITLE:

TITLE Graph title.
TITLE('text') adds text at the top of the current axis.

TITLE('text','Property1',PropertyValue1,'Property2',PropertyValue2,...)
sets the values of the specified properties of the title.

TITLE(AX,...) adds the title to the specified axes.

H = TITLE(...) returns the handle to the text object used as the title.

See also xlabel, ylabel, zlabel, text.

Reference page in Help browser
doc title

LEGEND

LEGEND Display legend.
LEGEND(string1,string2,string3, ...) puts a legend on the current plot using the specified strings as labels. LEGEND works on line graphs, bar graphs, pie graphs, ribbon plots, etc. You can label any solid-colored patch or surface object. The fontsize and fontname for the legend strings matches the axes fontsize and fontname.

LEGEND(H,string1,string2,string3, ...) puts a legend on the plot containing the handles in the vector H using the specified strings as labels for the corresponding handles.

LEGEND(M), where M is a string matrix or cell array of strings, and LEGEND(H,M) where H is a vector of handles to lines and patches also works.

LEGEND(AX,...) puts a legend on the axes with handle AX.

LEGEND OFF removes the legend from the current axes and deletes the legend handle.
LEGEND(AX,'off') removes the legend from the axis AX.

LEGEND TOGGLE toggles legend on or off. If no legend exists for the current axes one is created using default strings. The default string for an object is the value of the DisplayName property if it is non-empty and otherwise it is a string of the form 'data1','data2', etc.
LEGEND(AX,'toggle') toggles legend for axes AX

LEGEND HIDE makes legend invisible.
LEGEND(AX,'hide') makes legend on axes AX invisible.
LEGEND SHOW makes legend visible. If no legend exists for the

current axes one is created using default strings.
LEGEND(AX,'show') makes legend on axes AX visible.

LEGEND BOXOFF makes legend background box invisible when legend is visible.
LEGEND(AX,'boxoff') for axes AX makes legend background box invisible when legend is visible.
LEGEND BOXON makes legend background box visible when legend is visible.
LEGEND(AX,'boxon') for axes AX making legend background box visible when legend is visible.

LEGH = LEGEND returns the handle to legend on the current axes or empty if none exists.

LEGEND(...,'Location',LOC) adds a legend in the specified location, LOC, with respect to the axes. LOC may be either a 1x4 position vector or one of the following strings:

'North'	inside plot box near top
'South'	inside bottom
'East'	inside right
'West'	inside left
'NorthEast'	inside top right (default for 2-D plots)
'NorthWest'	inside top left
'SouthEast'	inside bottom right
'SouthWest'	inside bottom left
'NorthOutside'	outside plot box near top
'SouthOutside'	outside bottom
'EastOutside'	outside right
'WestOutside'	outside left
'NorthEastOutside'	outside top right (default for 3-D plots)
'NorthWestOutside'	outside top left
'SouthEastOutside'	outside bottom right
'SouthWestOutside'	outside bottom left
'Best'	least conflict with data in plot
'BestOutside'	least unused space outside plot

If the legend does not fit in the 1x4 position vector the position vector is resized around the midpoint to fit the preferred legend size. Moving the legend manually by dragging with the mouse or setting the Position property will set the legend Location property to 'none'.

LEGEND(...,'Orientation',ORIENTATION) creates a legend with the

legend items arranged in the specified ORIENTATION. Allowed
values for ORIENTATION are 'vertical' (the default) and 'horizontal'.

[LEGH,OBJH,OUTH,OUTM] = LEGEND(...) returns a handle LEGH to the
legend axes; a vector OBJH containing handles for the text, lines,
and patches in the legend; a vector OUTH of handles to the
lines and patches in the plot; and a cell array OUTM containing
the text in the legend.

Examples:

```
x = 0:.2:12;
plot(x,bessel(1,x),x,bessel(2,x),x,bessel(3,x));
legend('First','Second','Third');
legend('First','Second','Third','Location','NorthEastOutside')

b = bar(rand(10,5),'stacked'); colormap(summer); hold on
x = plot(1:10,5*rand(10,1),'marker','square','markersize',12,...
      'markeredgecolor','y','markerfacecolor',[.6 0 .6],...
      'linestyle','-','color','r','linewidth',2); hold off
legend([b,x],'Carrots','Peas','Peppers','Green Beans',...
       'Cucumbers','Eggplant')
```

Reference page in Help browser

```
doc legend
```

XLABEL:

XLABEL X-axis label.

XLABEL('text') adds text beside the X-axis on the current axis.

XLABEL('text','Property1',PropertyValue1,'Property2',PropertyValue2,...)
sets the values of the specified properties of the xlabel.

XLABEL(AX,...) adds the xlabel to the specified axes.

H = XLABEL(...) returns the handle to the text object used as the label.

See also ylabel, zlabel, title, text.

Reference page in Help browser

```
doc xlabel
```

YLABEL:

YLABEL Y-axis label.
YLABEL('text') adds text beside the Y-axis on the current axis.

YLABEL('text','Property1',PropertyValue1,'Property2',PropertyValue2,...)
sets the values of the specified properties of the ylabel.

YLABEL(AX,...) adds the ylabel to the specified axes.

H = YLABEL(...) returns the handle to the text object used as the label.

See also xlabel, zlabel, title, text.

Reference page in Help browser
doc ylabel

AXES:

AXES Create axes in arbitrary positions.
AXES('position', RECT) opens up an axis at the specified location
and returns a handle to it.
RECT = [left, bottom, width, height] specifies the location and
size of the side of the axis box, relative to the lower-left
corner of the Figure window, in normalized units where (0,0)
is the lower-left corner and (1.0,1.0) is the upper-right.

AXES, by itself, creates the default full-window axis and returns
a handle to it.

AXES(H) makes the axis with handle H current.

Execute GET(H) to see a list of axes object properties and
their current values. Execute SET(H) to see a list of axes
object properties and legal property values.

See also subplot, axis, figure, gca, cla.

Reference page in Help browser
doc axes

AXIS:

AXIS Control axis scaling and appearance.
  AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and y-axes
    on the current plot.
  AXIS([XMIN XMAX YMIN YMAX ZMIN ZMAX]) sets the scaling for the
    x-, y- and z-axes on the current 3-D plot.
  AXIS([XMIN XMAX YMIN YMAX ZMIN ZMAX CMIN CMAX]) sets the
    scaling for the x-, y-, z-axes and color scaling limits on
    the current axis (see CAXIS).
  V = AXIS returns a row vector containing the scaling for the
    current plot. If the current view is 2-D, V has four
    components; if it is 3-D, V has six components.

  AXIS AUTO returns the axis scaling to its default, automatic
    mode where, for each dimension, 'nice' limits are chosen based
    on the extents of all line, surface, patch, and image children.
  AXIS MANUAL freezes the scaling at the current limits, so that if
    HOLD is turned on, subsequent plots will use the same limits.
  AXIS TIGHT sets the axis limits to the range of the data.
  AXIS FILL sets the axis limits and PlotBoxAspectRatio so that
    the axis fills the position rectangle. This option only has
    an effect if PlotBoxAspectRatioMode or DataAspectRatioMode are
    manual.

  AXIS IJ puts MATLAB into its "matrix" axes mode. The coordinate
    system origin is at the upper left corner. The i axis is
    vertical and is numbered from top to bottom. The j axis is
    horizontal and is numbered from left to right.
  AXIS XY puts MATLAB into its default "Cartesian" axes mode. The
    coordinate system origin is at the lower left corner. The x
    axis is horizontal and is numbered from left to right. The y
    axis is vertical and is numbered from bottom to top.

  AXIS EQUAL sets the aspect ratio so that equal tick mark
    increments on the x-,y- and z-axis are equal in size. This
    makes SPHERE(25) look like a sphere, instead of an ellipsoid.
  AXIS IMAGE is the same as AXIS EQUAL except that the plot
    box fits tightly around the data.
  AXIS SQUARE makes the current axis box square in size.
  AXIS NORMAL restores the current axis box to full size and
     removes any restrictions on the scaling of the units.

This undoes the effects of AXIS SQUARE and AXIS EQUAL.
AXIS VIS3D freezes aspect ratio properties to enable rotation of
3-D objects and overrides stretch-to-fill.

AXIS OFF turns off all axis labeling, tick marks and background.
AXIS ON turns axis labeling, tick marks and background back on.

AXIS(H,...) changes the axes handles listed in vector H.

See also axes, grid, subplot, xlim, ylim, zlim.

Overloaded methods:
vrjoystick/axis

Reference page in Help browser
doc axis

GRID:

GRID Grid lines.
GRID ON adds major grid lines to the current axes.
GRID OFF removes major and minor grid lines from the current axes.
GRID MINOR toggles the minor grid lines of the current axes.
GRID, by itself, toggles the major grid lines of the current axes.
GRID(AX,...) uses axes AX instead of the current axes.

GRID sets the XGrid, YGrid, and ZGrid properties of
the current axes.

set(AX,'XMinorGrid','on') turns on the minor grid.

See also title, xlabel, ylabel, zlabel, axes, plot, box.

Reference page in Help browser
doc grid

HOLD:

HOLD Hold current graph
HOLD ON holds the current plot and all axis properties so that
subsequent graphing commands add to the existing graph.

HOLD OFF returns to the default mode whereby PLOT commands erase
the previous plots and reset all axis properties before drawing
new plots.
HOLD, by itself, toggles the hold state.
HOLD does not affect axis autoranging properties.

HOLD ALL holds the plot and the current color and linestyle so
that subsequent plotting commands will not reset the color and
linestyle.

HOLD(AX,...) applies the command to the Axes object AX.

Algorithm note:
HOLD ON sets the NextPlot property of the current figure and
axes to "add".
HOLD OFF sets the NextPlot property of the current axes to
"replace".

See also ishold, newplot, figure, axes.

Reference page in Help browser
  doc hold