Matlab cheat sheet

Numbers and variables

Multiple instructions can go in a .m file (*script*).

```
a = 5e7 %means 5 · 10^7
b = 2; %semicolons suppress output
c = 3.5+2i %complex numbers
a = b * c
format('long') %displays more digits
a = 2*sqrt(pi) + exp(1)
help('length') %shows docs
```

Flow control

```
if a == 5 && b <= 6
  y = true
else
  y = false
end
```

`||` is 'or'; `~` is 'not'; `a ~= b` is 'not equal'

```
while x < 10
  x = x + 1
end
```

```
for x = 1:10
  disp(x)
end
for x = 10:-1:1 %loop backwards
  disp(x)
end
v = [2, 3, 5, 7, 11];
for x = v %RHS can be any vector
  fprintf('%d is prime\n', x)
end
```

Functions

One function per file (usually), named as the function + .m

```
function y = sum_two_numbers(a, b)
% the first comment is a docstring
y = a + b; %value of y at end of evaluation returned
```

```
function [x, y] = returns_multiple_values(a, b)
x = a + b;
y = a - b;
```

```
f = @(a, b) a + b; %lambda expression
f(1, 2)
some_function(@f) %to pass a function as an argument
```

Vectors and matrices

```
M = [1 2 3; 4 5 6]
M = [M N; P Q] %concatenate matrices with same syntax
M = zeros(2, 3) %2x3 zero matrix
M = zeros(4) %square 4x4
M = ones(2, 3)
rand(2, 3) %uniform random in [0, 1]
randn(2, 3) %normal distribution N(0, 1)
size(M), length(v), sum(v), max(v)
2:5 %range [2, 3, 4, 5]
1:2:10 %range [1, 3, 5, 7, 9]
5:-1:1 %range [5, 4, 3, 2, 1]
```

Indexes start with 1 (boo).

```
v(1), M(1:5, 1:2), v(2:end), M(end:-1:1, end:-1:1)
```

Writing out of range resizes the matrix (wtf?!):

```
M(7, 9) = 2.5 %now M is at least 7x9, padded with zeros
v + w, v .* w, v ./ w, v .^ w %elementwise
sin(v), abs(M), sqrt(v) %elementwise
```

Linear algebra

```
A * B %linear algebra row-by-column product
det(A), inv(A) %usually avoided in numerics
A \ b %same as inv(A)*b, but more efficient
rowvector / A %same as rowvector * inv(A)
eig(A)
norm(v, 1), norm(v, inf), norm(M, 'fro')
norm(v) %same as norm(v, 2)
M' %transpose conjugate
```

Matrix decompositions

```
[P, L, U] = lu(A)
[Q, R] = qr(A)
[Q, T] = schur(A)
[V, D] = eig(A) %eigvecs=columns of V, eigvals=diag(D)
[U, S, V] = svd(A)
[Q, R] = qr(A, 0), [U, S, V] = svd(A, 0) %economy-sized
```

Sparse matrices

```
M = sparse(7, 9) %7x9 all zeros
M(2,3) = 5 %work like dense matrices in most cases
nnz(M) %number of nonzeros
[V, D] = eigs(A, 10) %top 10 eigenvalues
[U, S, V] = svds(A, 10) %singular values
spy(A) %plot sparsity pattern
```

Plots

```
plot(x, y) %line joining (x(1),y(1)) -- (x(2),y(2)) -- ...
plot(x1, y1, x2, y2) %multiple plots
hold('on'), hold('off') %keeps previous plot
semilogx(x, y), semilogy(x, y), loglog(x, y) %log scales
plot(x, y, 'rx') %set plot options
```

Colors: `ymcrgbwk`, markers: `o+*.xsd`, styles: `- -- : -.`

Debugging

To toggle breakpoints: click on the left of a line in the IDE. The debugger has prompt `K>>`.

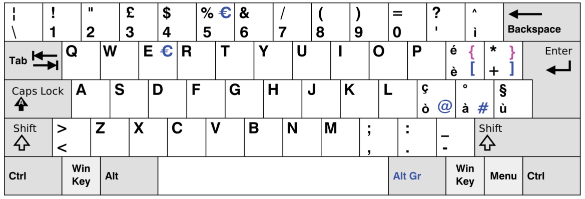
```
dbstop if error %starts debugger on error
dbquit %quits debugger
```

Optimization toolbox

```
linprog(c, A, b, Aeq, beq, lb, ub)
%solve the linear optimization problem
%min c'*x
%A*x <= b
%Aeq*x = beq
%lb <= x <= ub
quadprog(Q, c, A, b, Aeq, beq, lb, ub)
%solve the quadratic optimization problem
%min 0.5*x'*Q*x + c'*x
%A*x <= b
%Aeq*x = beq
%lb <= x <= ub
fminunc('f',x0,options)
%solve the unconstrained optimization problem
%min f(x)
%x in domain(f)
%with starting point x0 and possible options
```

Keyboard issues

To set the Italian keyboard on Linux: `setxkbmap it` (undo with `setxkbmap us`).



To set MATLAB keyboard shortcuts to the Windows default (e.g. copy with CTRL+C): Preferences (top-right in the ribbon)→ Matlab→ Keyboard→ Shortcuts→ Active set→ Windows default set.