x = 1 % scalar
y = [1 2 3 4 5 6] % sequence vector (row)
z = [1;2;3;4;5;6] % list vector (column)
a = [1 2 3;4 5 6;7 8 9] % matrix
b = [y; y; y] % matrix (list of sequences)
c = [z z z] % matrix (sequences of lists)
y' % conjugate transpose
y = [j*1, j*2, j*3, j*4, j*5, j*6] % matrix (sequences of lists)
y' % conjugate transpose (converts column to row, row to column)
y(:) % makes column

f = 0:1/10:1 % sequence (beg: step: end)
e = sin(2*pi*f) % operation on sequence (operation on each element of sequence)
f = [0:1.1:1.16:5.0:1/32:25] % list of sequences
\[ g = \sin(2\pi f) \]  
 operation on list of sequences  (ex. could have a list of sines sequences with different frequencies)  
 \[ g = \begin{bmatrix} 0 & 0.70710678118655 & 1 & 0.70710678118655 \\ 0 & 0.38268343236509 & 0.70710678118655 & 0.92387953251129 \\ 0 & 0.19509032201613 & 0.38268343236509 & 0.55557023301960 \end{bmatrix} \]

\[ \text{Column 1 through 4} \]
\[ \text{Column 5 through 8} \]
\[ \text{Column 9} \]
\[ \text{whos} \]  
 variable list and sizes  (helpful when doing vector multiplication)  
 \begin{tabular}{llrr}
 Name & Size & Bytes & Class \\
 a & 3x3 & 72 & double array \\
 ans & 1x10 & 4192 & struct array \\
 b & 3x6 & 144 & double array \\
 c & 6x3 & 144 & double array \\
 d & 1x11 & 88 & double array \\
 e & 1x11 & 88 & double array \\
 f & 3x9 & 216 & double array \\
 g & 3x9 & 216 & double array \\
 x & 1x1 & 8 & double array \\
 y & 1x6 & 48 & double array \\
 z & 6x1 & 48 & double array \\
 \end{tabular}

Grand total is 276 elements using 5264 bytes

leaving 307693008 bytes of memory free.

\[ y-1 \]  
 addition  
 \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \end{bmatrix} 

\[ z \]
\[ z = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \]

\[ y^2 - 1 \]  
 outer product  \[ 6 \times 1 \times 6 = 6 \times 6 \] (makes a list of sequences, each sequence \( y \) multiplied by corresponding element of list in \( z \) )

\[ y^2 z \]  
 inner product  \[ 1 \times 6 \times 6 = 1 \times 1 \] (multiplies each corresponding element of \( y \) and \( z \), then adds the resulting elements together)

\[ (y-1)^2 \]  
 inner product  \[ 1 \times 6 \times 6 = 1 \times 1 \] (multiplies each corresponding element of \( y \) and \( z \), then adds the resulting elements together)

\[ (y-1)^2 z \]  
 inner product  \[ 1 \times 6 \times 6 = 1 \times 1 \] (multiplies each corresponding element of \( y \) and \( z \), then adds the resulting elements together)

\[ t = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \]
\[ g = \begin{bmatrix} \sin(1) & \sin(2) & \sin(3) \\ \sin(2) & \sin(3) & \sin(4) \\ \sin(3) & \sin(4) & \sin(5) \\ \sin(4) & \sin(5) & \sin(6) \\ \sin(5) & \sin(6) & \sin(7) \\ \sin(6) & \sin(7) & \sin(8) \end{bmatrix} \]

\[ \sqrt{g} \]  
 function

\begin{verbatim}
function xx = sumcos(f,X,fs,dur)
xx = ???
end
\end{verbatim}
save as sumcos.m

function \([xx, t] = \text{sumcos}(f, X, fs, \text{dur})\)

% make time sequence from fs & dur
\(t = \) ...

% use outer product to make a list of complex exponential sequences using \(f\) and \(t\)

% use the inner product to multiply list of phasors \(X\) with list of complex exponential sequences

\(xx = \) ...

% plot \(z_p\)

% stem \(z_p\)
```matlab
cd dspfirst  % change directory (you either must be in directory to use the functions, or add path of directory)
ls  % list directory

Writing_Fast_MATLAB_Code.pdf pkinterp.m
alphacon.m  pkpick.m
andemoc.m  pretty_w.m
bflip.m  pumpkin.m
baboon.mat  replacesz.m
beacon.m  mocks.mat
beaconb.m  show_img.m
clp.m  showspec.m
cosgen.m  snhid.m
dspf.mat  spectpr.m
dsp1.mat  stripplot.m
dsp2.mat  sumprod.m
dspfirst.m  tools.gif
dtmf.thick.m  tools.mat
dtmf.main.m  truisie.m
dehar12.mat  upplot.m
ehar.mat  vowel.d.m
eexpand.m  wavesnds.m
features.m  wavierac.m
ffrfr.m  wngui.m
fullsize.mat  wngui.mat
insample.m  woodwhei.m
inout.m  wrekl.mat
lab6.dat.mat  winotes.m
lab7.dat.mat  zcat.m
lenna.mat  zoom3d.m
lenna12.mat  zoom3d.m
lenna_bl.mat  zone.mat
manipsin.m  zone12.mat
mattsbr.m  zone_make.m
musiclab  zprint.m
mysound.m  zvplot.m
nveloper.m  zvect.m
nez_31

help zprint  % if you don't know what the command does, use help

ZPRINT  printout complex # in rect and polar form

```

```matlab
help zprint
```

```
usage:  zprint(z)
z = vector of complex numbers; each one will be printed in a format showing real, imag, mag and phase

```

```matlab
help zvect
```

```
ZVECT  Plot vectors in complex z-plane from zFrom to zTo

```

```matlab
help zvect
```

```
usage:  HV = zvect(zFrom, <zTo>, <LTYPE>, <SCALE>)

```

```matlab
plot(t,zp)
```

```

```matlab
» plot(t,zp)
```

```
```

```
```

```
```
** With only one input vector: `zvect(Z)` displays Z as arrows emanating from the origin.
** If either zFrom or zTo is a scalar all vectors will start or end at that point.
See also ZCAT, PLOT, COMPASS, ROSE, FEATHER, QUIVER.

```matlab
zvect([z1, z2])
```

```
% With only one input vector: zvect(Z) displays Z as arrows emanating from the origin.
% If either zFrom or zTo is a scalar all vectors will start or end at that point.
See also ZCAT, PLOT, COMPASS, ROSE, FEATHER, QUIVER.

```
```