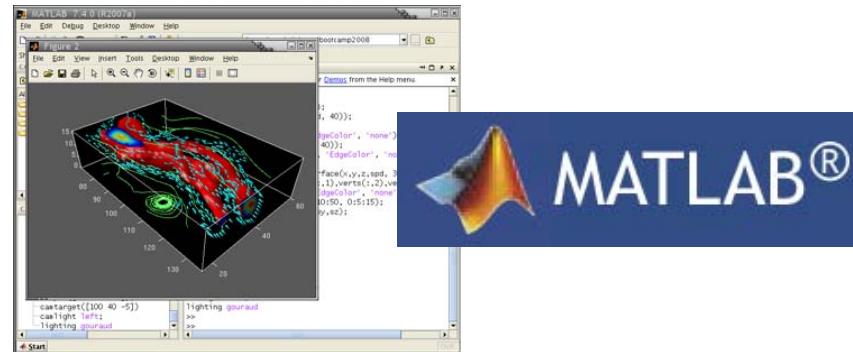




Introduction to Scientific Programming in MATLAB



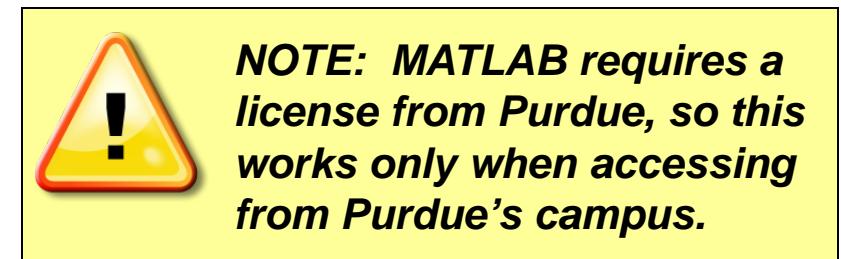
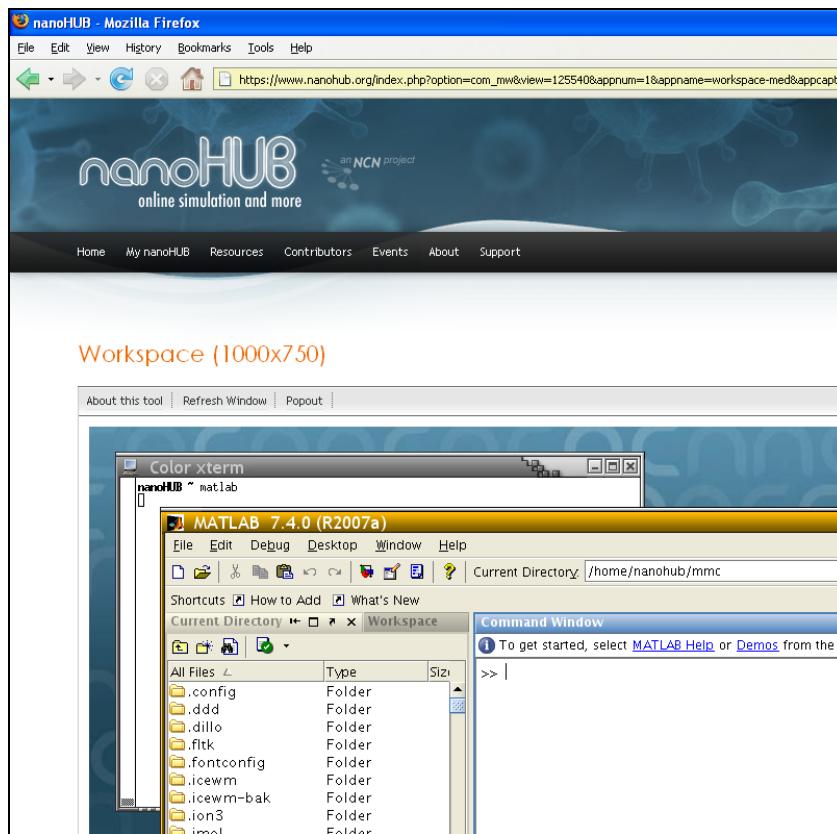
Michael McLennan

HUBzero® Platform for Scientific Collaboration

Purdue University



Start a workspace and type **matlab**



Try some simple commands:

```
>> x=1  
>> y=2  
>> z=2*x + y  
>> exit
```



Free Clone: GNU Octave

In your workspace, type `octave`

A screenshot of a Mozilla Firefox browser window. The title bar says "nanoHUB.org - Middleware: View: Workspace (1000x750) (10:39 am) - Mozilla Firefox". The address bar shows the URL "https://nandhub.org/mw/view/202820?tool=workspace-med". The main content area displays the nanoHUB.org homepage with a blue header "nanoHUB.org ONLINE SIMULATION AND MORE". Below the header are navigation links: Home, My HUB, Resources, Members, Explore, About, Support. A breadcrumb trail "You are here: Middleware > View > Workspace (1000x750) (10:39 am)" is shown. The main content is titled "Workspace (1000x750)". At the top of this section are buttons for "Tool", "Questions?", and "About". Below these is an "xterm" terminal window. The terminal window has a yellow title bar labeled "xterm". The text inside the terminal is:

```
xterm
nanoHUB ~ octave
GNU Octave, version 2.1.73 (i486-pc-linux-gnu).
Copyright (C) 2006 John W. Eaton.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. For details, type `warranty'.
Additional information about Octave is available at http://www.octave.org.

Please contribute if you find this software useful.
For more information, visit http://www.octave.org/help-wanted.html

Report bugs to <bug@octave.org> (but first, please read
http://www.octave.org/bugs.html to learn how to write a helpful
octave:1>
```

- Runs almost all MATLAB scripts
- Creates plots
- Missing fancy debugger
- Missing “simulink” toolbox



Introducing... OCTAViEw

<http://nanohub.org/tools/octaview>

The screenshot shows the OCTAViEw tool page on nanoHUB.org. The left side displays the tool's homepage with its title, author (Michael McLennan from Purdue University), and a scratchpad for Octave/MATLAB scripts. The right side shows a modal window for running scripts. The modal has tabs for "Upload...", "Open...", and "Save As...". A text area contains a simple MATLAB script to generate concentric ellipses:

```
# simple matlab script
angle=0:0.1:3.14*10
y=angle.*sin(angle)
x=angle.*cos(angle)
plot(y,x)
```

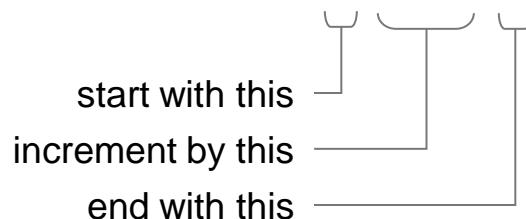
Below this is a "Run Script" button. To the right is a plot window titled "Run Script" showing a plot of the generated ellipses. The plot has axes labeled "x" and "y", with scales from -20 to 20. The plot area contains several nested, slightly elliptical curves centered at approximately (-10, 0). Below the plot are tabs for "Plots", "Debug", and "Console Output", with "Plots" currently selected.

Quick and dirty
 Simple plots

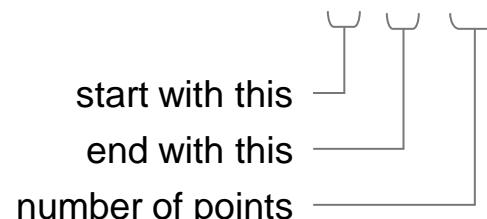
Create a series of number like this:



xaxis = 1:0.1:5



xaxis = linspace(1, 5, 41)



Try this:

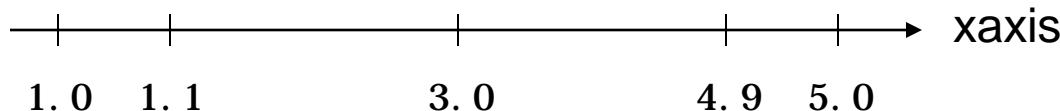
```
>> xaxis = 1:0.1:5;
```

```
>> length(xaxis)
```

```
>> xaxis
```

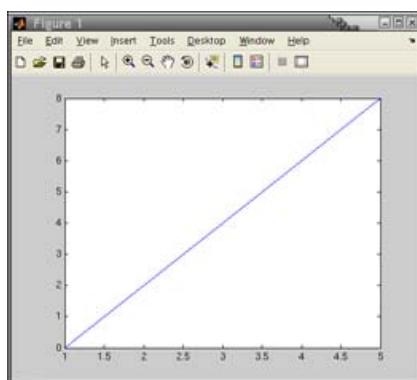
Try it with/without the semicolon.
What does the semicolon do?

Suppose you have an irregular spacing:



xaxis = [1. 0, 1. 1, 3. 0, 4. 9, 5. 0]

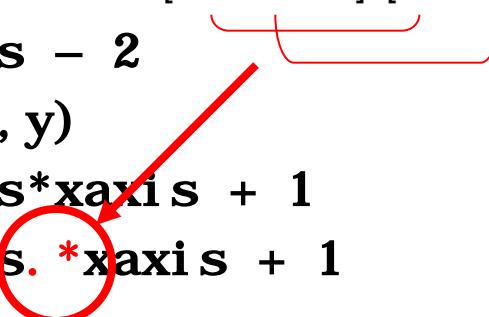
square brackets
comma-separated list of values



Try this:

```
>> y = 2*xaxis - 2
>> plot(xaxis, y)
>> y = 3*xaxis*xaxis + 1
>> y = 3*xaxis.*xaxis + 1
```

Multiply element by element:
[1.0 1.1 ...] [1.0 1.1 ...]





Suppose you want to define a matrix, like this:

$$a = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{in MATLAB, that's} \\ \gg \mathbf{a} = [-1 -1 -1 ; 0 0 0 ; 1 1 1]$$

$$a^T = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{in MATLAB, that's} \\ \gg \mathbf{a}'$$

Try this:

```
>> a * a'  
>> a' * a  
>> a' * a + 1
```

Try these built-in functions:

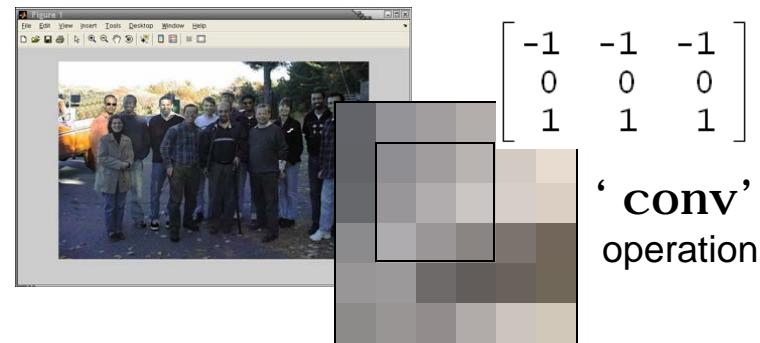
```
>> zeros(3)  
>> ones(4)  
>> eye(2)  
>> a * a' + eye(3)
```

Try this:

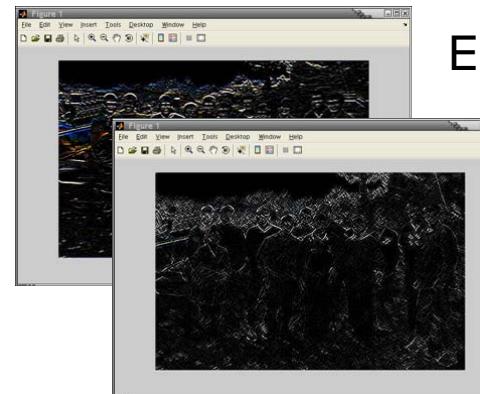
```
>> im = imread(' /apps/matlab/7.9/tool box/simulink/simulink/simulinkteam.jpg' );
>> figure;
>> imshow(im);

>> a = [ -1 -1 -1 ; 0 0 0 ; 1 1 1]
>> im2 = imfilter(im, a, 'conv');
>> imshow(im2);

>> im2 = imfilter(im, a*a', 'conv');
>> imshow(im2);
```



Edge detection!



charcoal effect



Octave Users:

Works in Octave 3.0

For earlier versions, download `imread` from

<http://www.cs.helsinki.fi/u/ahyvarin/kurssi/imread.m>



$$a = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

in MATLAB, that's

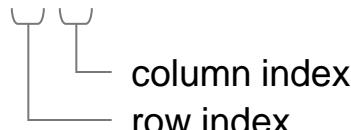
```
>> a = [-1 -1 -1 ; 0 0 0 ; 1 1 1]
```

Try this:

```
>> a(1, 1)
```

```
>> a(2, 1)
```

```
>> a(1, 2)
```



Try this:

```
>> a(1, 1: 3)
```

```
>> a(1, :)
```

A diagram showing a single horizontal bracket underneath a row of a matrix, indicating that the entire row is being selected.

Try this:

```
>> a(2, 2) = 3
```

```
>> a
```

Pick apart image pixel by pixel...

```
>> im(1, 5)
```

```
>> im(3, :)
```

```
>> im(1: 100, 1: 100)
```

Try this:

```
>> rv = im(3, :, 1)
```

first component: r g b
all column elements
row index

```
>> plot(1:length(rv), rv, 'r')
```

```
>> title('row 3 from image')
```

```
>> xlabel('pixel index')
```

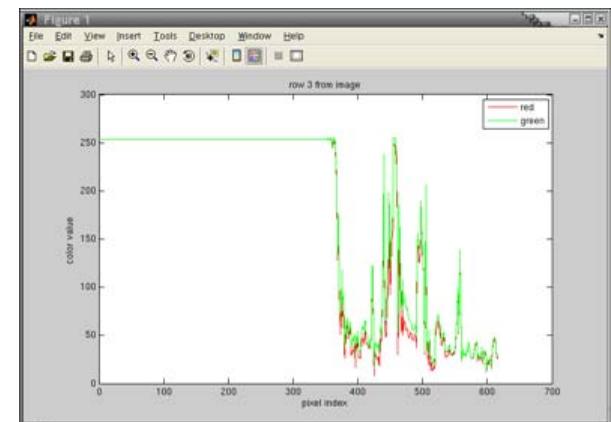
```
>> ylabel('color value')
```

```
>> hold
```

```
>> gv = im(3, :, 2)
```

```
>> plot(1:length(gv), gv, 'g')
```

```
>> legend('red', 'green')
```

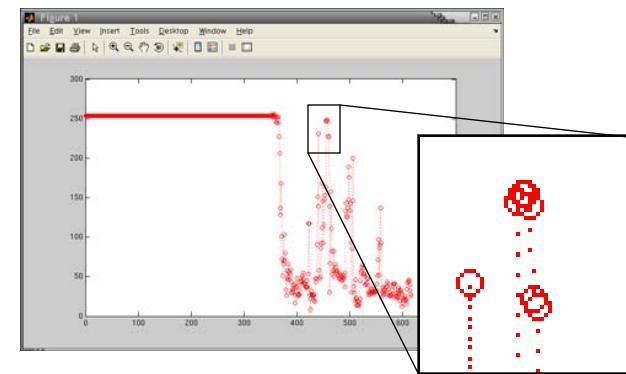


Try this:

```
>> clf
```

```
>> plot(1:length(rv), rv, 'r: o')
```

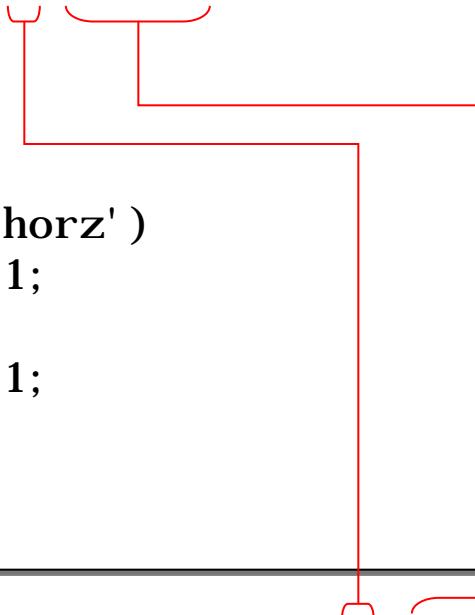
red, dotted line, circles



r	Red	-	Solid line (default)	+	Plus sign
g	Green	--	Dashed line	o	Circle
b	Blue	:	Dotted line	*	Asterisk
c	Cyan	-.	Dash-dot line	.	Point
m	Magenta			x	Cross
y	Yellow			s	Square
k	Black			d	Diamond
w	White			^	Upward-pointing triangle
				v	Downward-pointing triangle
				>	Right-pointing triangle
				<	Left-pointing triangle
				p	Five-pointed star (pentagram)
				h	Six-pointed star (hexagram)

file: edgematrix.m

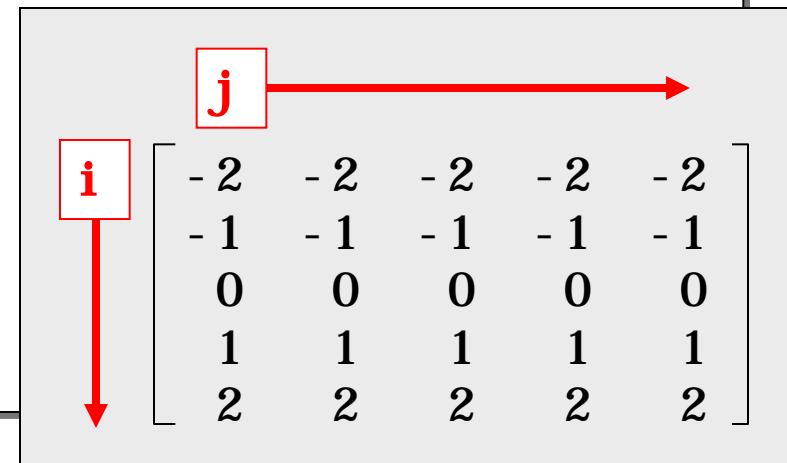
```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix(n, orient)
    hal f = floor(n/2);
    n = 2*hal f + 1;
    for i=1:n
        for j=1:n
            if strcmp(orient, 'horz')
                m(i,j) = j - hal f - 1;
            else
                m(i,j) = i - hal f - 1;
            end
        end
    end
```



```
im2 = imfilter(im, edgematrix(5, 'horz'), 'conv');
```

file: edgematrix.m

```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix(n, orient)
half = floor(n/2);
n = 2*half + 1;
for i=1:n
    for j=1:n
        if strcmp(orient, 'horz')
            m(i,j) = j - half - 1;
        else
            m(i,j) = i - half - 1;
        end
    end
end
```



file: edgematrix.m

```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix(n, orient)
half = floor(n/2);
n = 2*half + 1;
for i=1:n
    for j=1:n
        if strcmp(orient, 'horz')
            m(i,j) = j - half - 1; ——————
        else
            m(i,j) = i - half - 1; ——————
        end
    end
end
```

$$\mathbf{m} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{'horz'}$$

$$\mathbf{m} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{anything else}$$



file: edgematrix.m

```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix(n, orient)
half = floor(n/2);
n = 2*half + 1;
for i=1:n
    for j=1:n
```

```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix2(n, orient)
half = floor(n/2);
o = ones(2*half + 1);
m = [-half:half]' * o(1, :)
if strcmp(orient, 'horz')
    m = m'
end
```

$$\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$



file: hello.m

```
% prompt for a phrase and say "hello"
disp('Who are you?');
name = input('Enter your name: ', 's');
age = input('Enter your age: ');

mesg = sprintf('Hello, %s!', name);
disp(mesg);

file = input('Enter a file name: ', 's');
fid = fopen(file, 'w');
fprintf(fid, '%s is %d years old\n', name, age);
fclose(fid);
```

Use script name as a command to invoke the script

String input
Numbers and other stuff

```
>> hello
Who are you?
Enter your name: Michael
Enter your age: 43
Hello, Michael!
Enter a file name: info.txt
```



Other Resources

Tutorials at MathWorks

http://www.mathworks.com/academia/student_center/tutorials/launchpad.html

The screenshot shows the MathWorks MATLAB Tutorial page. On the left, there's a sidebar with links for Student Center Main Page, MATLAB & Simulink Student Version, Product Support, Tutorials, Homework Helper, From School to Industry, and Get Pricing and Purchase Student Version. The main content area is titled "MATLAB Tutorial" and includes a section "I want to..." with several options:

- Watch short tutorial videos and look at MATLAB examples.
- Take an interactive MATLAB tutorial designed for students.
- Read the User's Guide MATLAB tutorial.
- Read university-authored MATLAB tutorials from Union College, University of Dundee (PDF), Southern Illinois University, Massachusetts Tech, Stanford, MIT, University of Michigan & Carnegie Mellon (Controls), Georgia Tech (Signal Processing).
- Browse listings of other university resources on MATLAB.
- Buy a top-selling MATLAB book from Amazon.com.

Below these, there are sections for "Getting Started Tutorial Videos and MATLAB Examples" and "Interactive MATLAB Tutorial".

MATLAB DOs and DON'Ts

<http://nanohub.org/resources/1279>

The screenshot shows a nanohub.org resource page for "Matlab DOs and DON'Ts" by Dmitri Nikonorov. The page has a sidebar with "Nanotech" and "MATLAB DOs and DON'Ts" sections, and a main content area with a video player and a summary table.

Abstract: Matlab DOs and DON'Ts
"the unrepentant Matlab programmer"
Sponsored by: Intel
Contact: Dmitri.e.Nikonorov@intel.com

Matlab DOs and DON'Ts
Dmitri Nikonorov

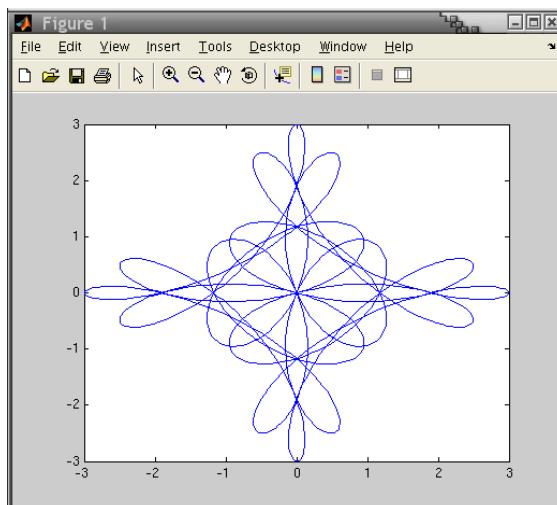
	Duration
Matlab DOs and DON'Ts	00:59
Outline	00:57
Matlab profiling	04:11
Tool: profiler	01:44
Tool: interrupt debug	01:21
Tool: windbg	00:49
Links	02:20



Assignment #3: Spirograph plot

Spirograph equation:

$$z(t) = e^{i2\pi n_1 t} + e^{i2\pi n_2 t} + e^{i2\pi n_3 t}$$



See theory at <http://linuxgazette.net/133/luana.html>

Where t has 1,000 points along $[0,1]$

Plot:

$\text{real}(z) \rightarrow x$
 $\text{imag}(z) \rightarrow y$

In MATLAB/Octave:

```
t = linspace(0, 1, 1000);  
z = exp(i * 2 * pi * n1 * t) + exp(i * 2 * pi * n2 * t) + exp(i * 2 * pi * n3 * t);  
plot(real(z), imag(z));
```

Hint: Try this in OCTAViEw - <http://nanohub.org/tools/octaview>