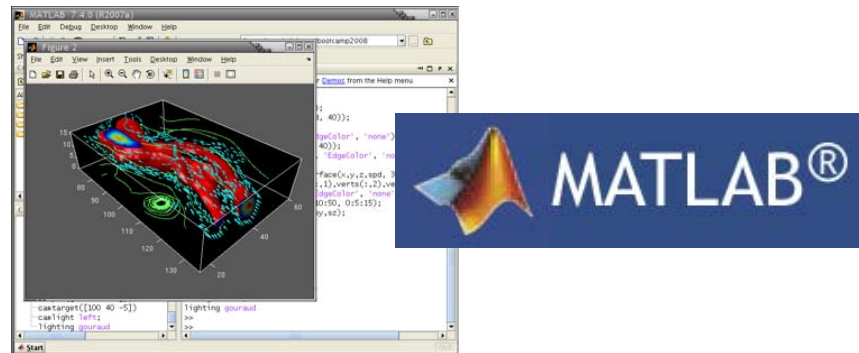


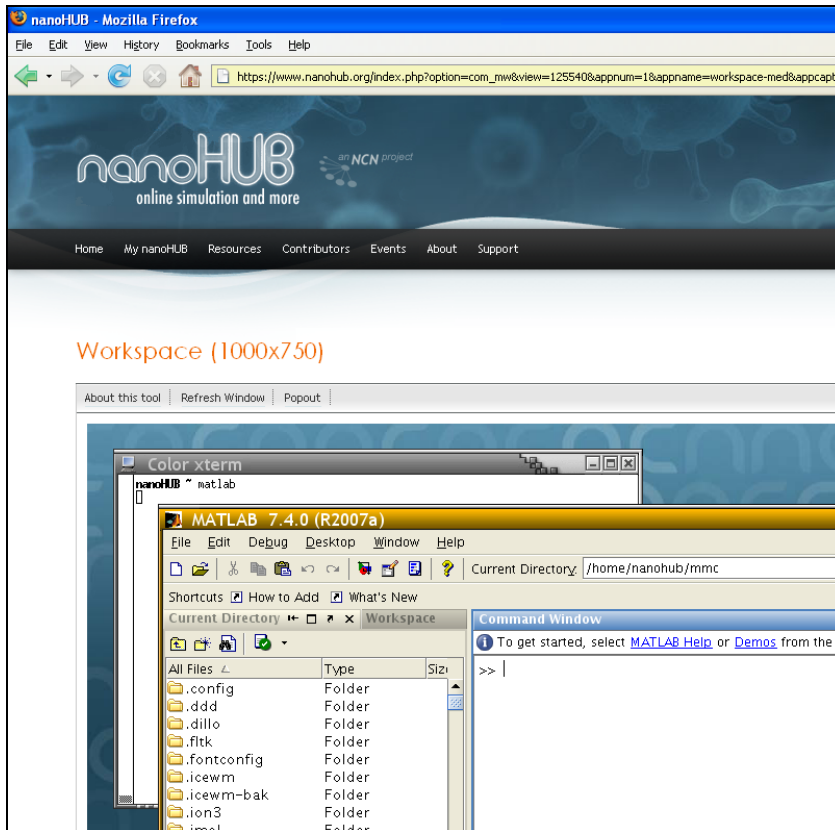
Introduction to Scientific Programming in MATLAB



Michael McLennan

*HUBzero® Platform for Scientific Collaboration
Purdue University*

Start a workspace and type `matlab`



NOTE: MATLAB requires a license from Purdue, so this works only when accessing from Purdue's campus.

Try some simple commands:

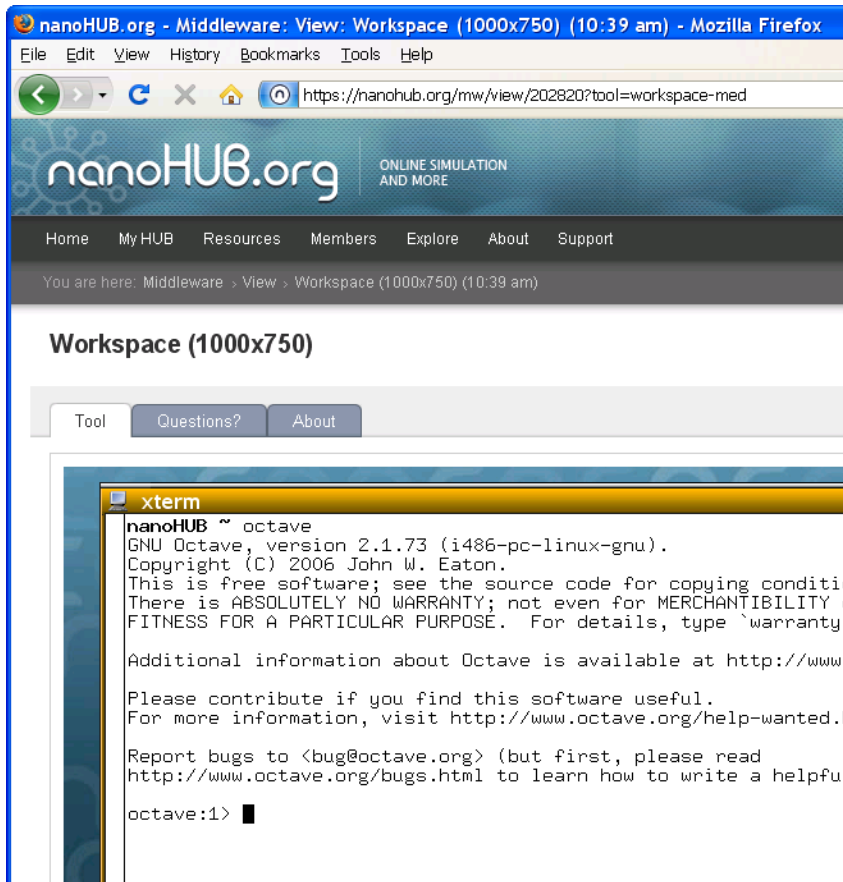
```
>> x=1
```

```
>> y=2
```

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

```
>> exit
```

In your workspace, type octave



The screenshot shows a web browser window titled "nanoHUB.org - Middleware: View: Workspace (1000x750) (10:39 am) - Mozilla Firefox". The address bar shows the URL "https://nanohub.org/rmw/view/202820?tool=workspace-med". The page header includes the nanoHUB.org logo and navigation links. Below the header, there is a section titled "Workspace (1000x750)" with tabs for "Tool", "Questions?", and "About". The "Tool" tab is active, displaying a terminal window titled "xterm". The terminal output shows the GNU Octave version 2.1.73 (i486-pc-linux-gnu) and its license information. The prompt "octave:1>" is visible at the bottom of the terminal.

```
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

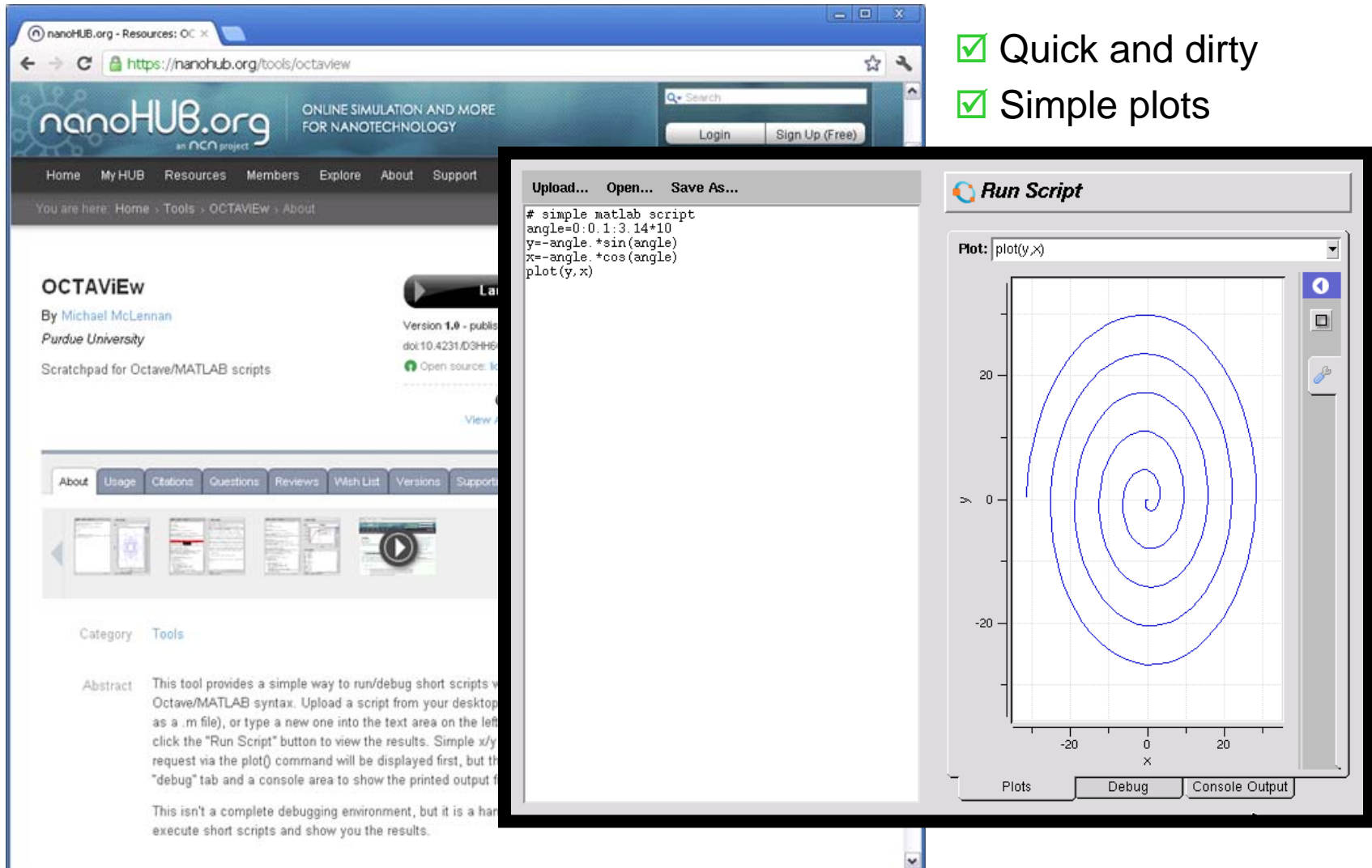
Report bugs to <bug@octave.org> (but first, please read
http://www.octave.org/bugs.html to learn how to write a helpful
bug report.)

octave:1> █
```

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

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

- ✓ Quick and dirty
- ✓ Simple plots



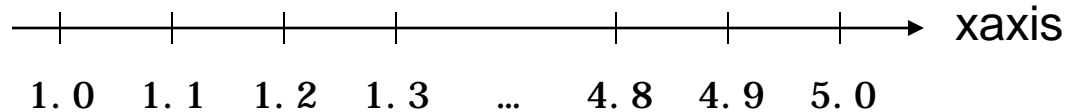
The screenshot shows the nanoHUB.org website for the OCTAViEw tool. The main content area displays the tool's name, author (Michael McLennan), and version (1.0). Below this is a code editor with the following MATLAB script:

```




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

To the right of the code editor is a 'Run Script' button. Below the code editor is a plot window titled 'Plot: plot(y,x)'. The plot shows a blue spiral curve on a grid. The x-axis is labeled 'x' and the y-axis is labeled 'y'. The plot window has a 'Plots' tab selected, along with 'Debug' and 'Console Output' tabs.




Create a series of number like this:



`xaxis = 1:0.1:5`

start with this 
 increment by this 
 end with this 

`xaxis = linspace(1, 5, 41)`

start with this 
 end with this 
 number of points 

Try this:

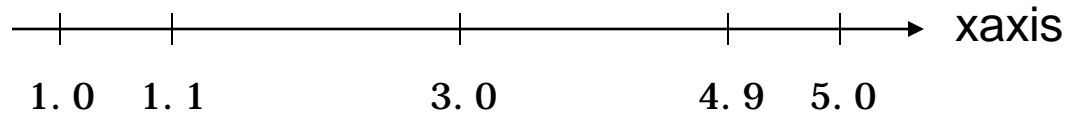
`>> xaxis = 1:0.1:5;`

`>> length(xaxis)`

`>> xaxis`

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

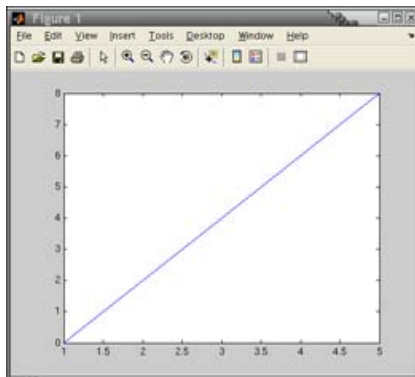
Suppose you have an irregular spacing:



```
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:

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

in MATLAB, that's

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

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

in MATLAB, that's

```
>> a'
```

Try this:

```
>> a * a'
```

```
>> a' * a
```

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

Try these built-in functions:

```
>> zeros(3)
```

```
>> ones(4)
```

```
>> eye(2)
```

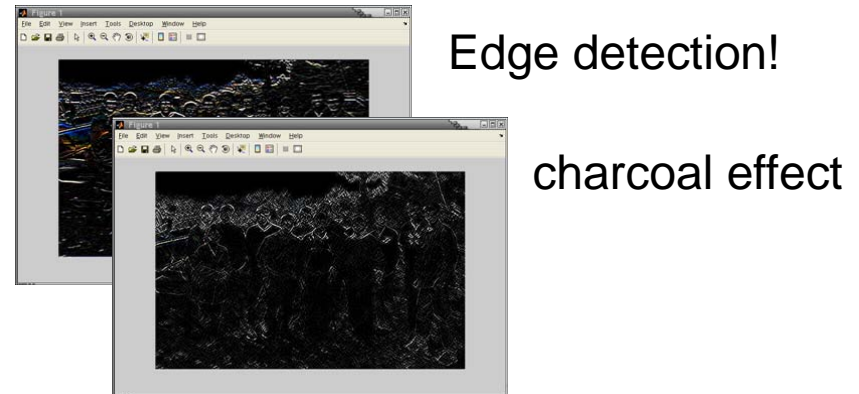
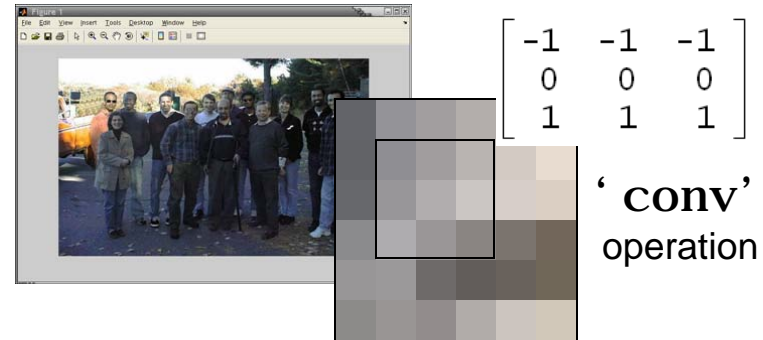
```
>> a * a' + eye(3)
```

Try this:

```
>> im = imread(' /apps/matlab/7.9/toolbox/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);
```



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

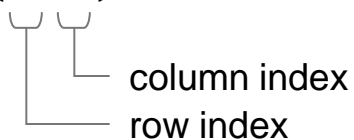
```
>> a = [-1 -1 -1 ; 0 0 0 ; 1 1 1]
```

Try this:

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

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

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

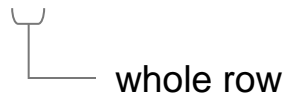


column index
row index

Try this:

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

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



whole row

Try this:

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

```
>> a
```

Pick apart image pixel by pixel...

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

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

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

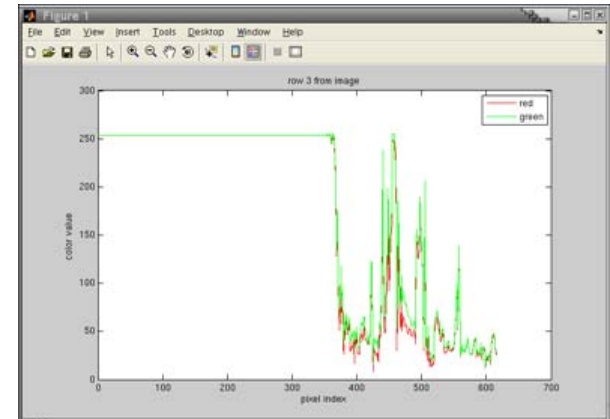
Try this:

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

Diagram illustrating the indexing in the MATLAB command `im(3, :, 1)`:

- `3`: row index
- `:`: all column elements
- `1`: first component: r g b

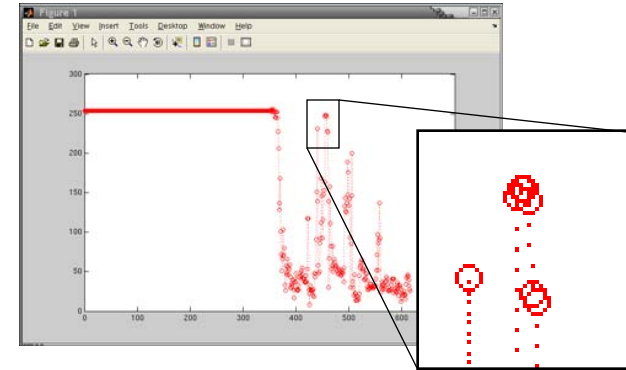
```
>> plot(1:length(rv), rv, 'r')
>> title('row 3 from image')
>> xlabel('pixel index')
>> ylabel('color value')
>> hold
>> gv = im(3, :, 2)
>> plot(1:length(gv), gv, 'g')
>> legend('red', 'green')
```



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)

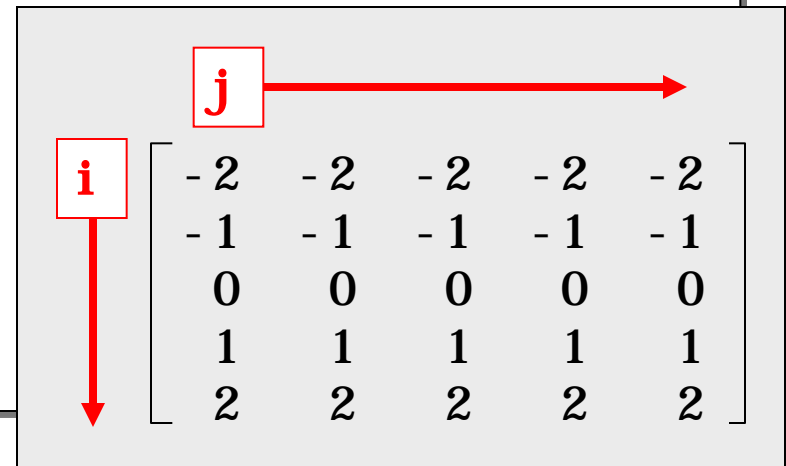
file: edgematrix.m

```
% returns edge detection matrix of size n
% orient is 'horz' for horizontal edges, and 'vert' for vertical
function m = edgematrix(n, orient)
    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
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
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
end
```

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

$m = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$ 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);
```

String input
Numbers and
other stuff

Use script name as a
command to invoke
the script

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


Spirograph equation:

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

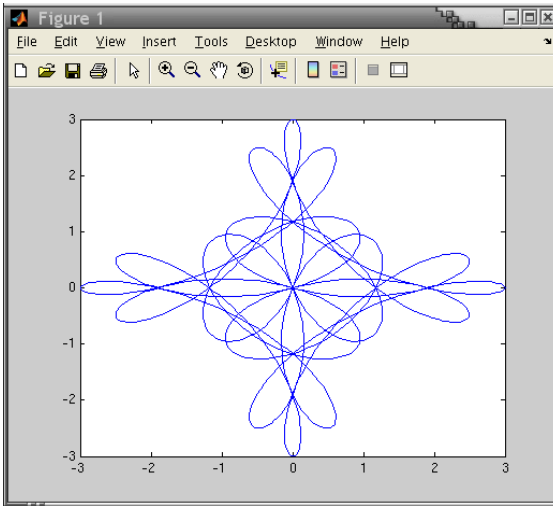
See theory at <http://linuxgazette.net/133/luana.html>

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>