L13. More on Arrays

Square Bracket Notation
Subscripts
Plotting and color
Built-In Functions:
  ginput, fill, sum, axis

The Square Bracket

These are equivalent:

\[
x = \text{linspace}(0,1,5)\\
x = [0 \ 0.25 \ 0.50 \ 0.75 \ 1.00]
\]

\[
x: \\
\begin{array}{cccc}
0.00 & 0.25 & 0.50 & 0.75 & 1.00 \\
\end{array}
\]

Handy for setting up "short" vectors.

Three "Short Vector" Examples

Line Segments
Little Polygons
Color

Plotting a Line Segment

This draws a line segment that connects (1,2) and (3,4):

\[
a = 1; \\
b = 2; \\
c = 3; \\
d = 4; \\
\text{plot}([a \ c],[b \ d])
\]

A natural mistake:

\[
\text{plot}([a \ b],[c \ d])
\]

Drawing Little Polygons

This draws an L-by-W rectangle with lower left corner at (a,b):

\[
x = [a \ a+L \ a+L \ a \ a]; \\
y = [b \ b \ b+W \ b+W \ b]; \\
\text{plot}(x,y)
\]

Connect (a,b) to (a+L,b) to (a+L,b+W) to (a,b+W) to (a,b)

Coloring Little Polygons

This draws an L-by-W rectangle with lower left corner at (a,b) and colors with the color named by c:

\[
x = [a \ a+L \ a+L \ a \ a]; \\
y = [b \ b \ b+W \ b+W \ b]; \\
\text{fill}(x,y,c)
\]

Connect (a,b) to (a+L,b) to (a+L,b+W) to (a,b+W) to (a,b) and then fill it in.
**Built-In Function Fill**

fill( ,        ,        )

Vectors that specify the vertices of a polygon.

Specify the fill-in color.

\[ x = [0.11\ -0.99\ -0.78\ 0.95] \]
\[ y = [0.99\ 0.08\ -0.62\ -0.30] \]
fill(x,y,'y')

**DrawRect**

function DrawRect(a,b,L,W,c)
x = [a a+L a+L a a];
y = [b b b+W b+W b];
fill(x,y,c)

**Color is a 3-vector**

Any color is a mix of red, green, and blue.

Represent a color with a length-3 vector and an "rgb convention".

\[ c = [0.25\ 0.63\ 0.00] \]

red value between 0 and 1
green value between 0 and 1
blue value between 0 and 1

**Some Favorites**

<table>
<thead>
<tr>
<th>Color</th>
<th>RGB</th>
<th>Color</th>
<th>RGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>[0 0 0]</td>
<td>Blue</td>
<td>[0 0 1]</td>
</tr>
<tr>
<td>Blue</td>
<td>[0 0 1]</td>
<td>Green</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>Green</td>
<td>[0 1 0]</td>
<td>Cyan</td>
<td>[0 1 1]</td>
</tr>
<tr>
<td>Cyan</td>
<td>[0 1 1]</td>
<td>Red</td>
<td>[1 0 0]</td>
</tr>
<tr>
<td>Red</td>
<td>[1 0 0]</td>
<td>Magenta</td>
<td>[1 0 1]</td>
</tr>
<tr>
<td>Magenta</td>
<td>[1 0 1]</td>
<td>Yellow</td>
<td>[1 1 0]</td>
</tr>
<tr>
<td>Yellow</td>
<td>[1 1 0]</td>
<td>Black</td>
<td>[1 1 1]</td>
</tr>
</tbody>
</table>

**A Problem**

Display all colors \([r\ g\ b]\) where

\[ r\ \text{ranges over the values} \]
\[ 0.00\ 0.25\ 0.50\ 0.75\ 1.00 \]

\[ g\ \text{ranges over the values} \]
\[ 0.00\ 0.25\ 0.50\ 0.75\ 1.00 \]

\[ b\ \text{ranges over the values} \]
\[ 0.00\ 0.25\ 0.50\ 0.75\ 1.00 \]
Preliminary Notes

There will be $5 \times 5 \times 5 = 125$ colors to display.

To display a color, we will draw a unit square with that color.

Script Derivation

```matlab
for r = 0:.25:1
    Display all colors with red value r.
end
```

Refine This!

```matlab
for r = 0:.25:1
    for g = 0:.25:1
        Display all colors with red value r and green value g.
    end
end
```

Refine This!

```matlab
for r = 0:.25:1
    for g = 0:.25:1
        for b = 0:.25:1
            Display the color with red value r, green value g, and blue value b.
        end
    end
end
```

Refine This!

```matlab
for r = 0:.25:1
    for g = 0:.25:1
        for b = 0:.25:1
            c = [r g b];
            fill(x,y,c)
        end
    end
end
```

Done!

Subscripts

It is possible to access and change specific entries in an array.

```latex
x: 0.00 0.25 0.50 0.75 1.00
```

The value of $x(1)$ is 0.00.
The value of $x(2)$ is 0.25.
The value of $x(3)$ is 0.50.
The value of $x(4)$ is 0.75.
The value of $x(5)$ is 1.00.
Subscripts

It is possible to access and change specific entries in an array.

\[ x = [0.00 \ 0.25 \ 0.50 \ 0.75 \ 1.00] \]

\[
\begin{align*}
    a &= x(1) \\
    a &= x(2) \\
    a &= x(3) \\
    a &= x(4) \\
    a &= x(5)
\end{align*}
\]

\[
\begin{align*}
    a &= 0.00 \\
    a &= 0.25 \\
    a &= 0.50 \\
    a &= 0.75 \\
    a &= 1.00
\end{align*}
\]

Subscripts

It is possible to access and change specific entries in an array.

\[ x = [0.00 \ 0.25 \ 0.50 \ 0.75 \ 1.00] \]

\[
\begin{align*}
    a &= x(1) + x(2) \\
    a &= x(2) + x(3) \\
    a &= x(3) + x(4) \\
    a &= x(4) + x(5)
\end{align*}
\]

\[
\begin{align*}
    a &= 0.25 \\
    a &= 0.75 \\
    a &= 1.25 \\
    a &= 1.75
\end{align*}
\]

Subscripts

This

\[ x = \text{linspace}(a, b, n) \]

is equivalent to this

\[
\begin{align*}
    h &= (b-a)/(n-1) \\
    \text{for } k=1:n \\
    x(k) &= a + (k-1)*h \\
    \text{end}
\end{align*}
\]

\[
\begin{align*}
    h &= (1-0)/(5-1) \\
    x(1) &= 0 + 0*h \\
    x(2) &= 0 + 1*h \\
    x(3) &= 0 + 2*h \\
    x(4) &= 0 + 3*h \\
    x(5) &= 0 + 4*h
\end{align*}
\]
Subscripts

\[ h = \frac{b-a}{n-1}; \]

for \( k=1:n \)

\[ x(k) = a + (k-1)*h. \]

end

Recipe for a value

Where to put it.*

* Only now we compute where to put it.

A Problem

Click in the three vertices of a triangle.

Display the triangle.

Compute the centroid \((x_c,y_c)\).

Highlight the centroid and the vertices.

Connect each vertex to the centroid.

Clicking in the Vertices

The command

\[ [x,y] = ginput(3) \]

will assign the xy coordinates of the clicks to the arrays \(x\) and \(y\).

Clicks: \((1,2),(3,4),(5,6)\)

\[
\begin{align*}
x &: 1 & 3 & 5 \\
y &: 2 & 4 & 6 \\
\end{align*}
\]

Display the Triangle

\[
\text{fill}(x,y,'
')
\]

Compute the Centroid

The \(x\) coordinate of the centroid is the average of the \(x\)-coordinates of the vertices. Ditto for \(y\)-coordinate.

\[
\begin{align*}
xc &= \frac{x(1) + x(2) + x(3)}{3}; \\
yc &= \frac{x(1) + y(2) + y(3)}{3};
\end{align*}
\]

Highlight the Vertices and Centroid

\[
\text{plot}(x,y,'ok',xc,yc,'*k')
\]
Connect Each Vertex to the Centroid

```matlab
plot([xc x(1)], [yc y(1)])
plot([xc x(2)], [yc y(2)])
plot([xc x(3)], [yc y(3)])
```

Overall

```matlab
[x, y] = ginput(3)
fill(x, y, 'y')
hold on
xc = (x(1) + x(2) + x(3)) / 3;
yc = (y(1) + y(2) + y(3)) / 3;
plot(x, y, 'ok', xc, yc, '*k')
plot([xc x(1)], [yc y(1)])
plot([xc x(2)], [yc y(2)])
plot([xc x(3)], [yc y(3)])
```

More General

```matlab
n = 3;
[x, y] = ginput(n)
fill(x, y, 'y')
xc = sum(x) / n;
yc = sum(y) / n;
plot(x, y, 'ok', xc, yc, '*k')
for k = 1:n
    plot([xc x(k)], [yc y(k)])
end
```

Question Time

What is the output?

```matlab
x = [10 20 30];
y = [3 1 2]
k = y(3) - 1;
z = x(k + 1)
```

A: 11   B: 20   C: 21   D: 30   E: 31

A Problem

Plot the function $y = \sin(x)$ across $[0, 2\pi]$ but add random noise if

$$\frac{2\pi}{3} \leq x \leq \frac{4\pi}{3}$$

In particular, use $\sin(x) + 0.1 \times \text{randn}(1)$ if $x$ is in this range.

Solution

```matlab
x = linspace(0, 2*pi, n);
y = sin(x);
for k = 1:n
    if 2*pi/3 <= x(k) && x(k) <= 4*pi/3
        y(k) = y(k) + 0.1*randn(1);
    end
end
plot(x, y)
axis([0 2*pi -1.2 1.2])
```

x range: [0 2*pi]  y range: [-1.2 1.2]
Problem
On a black background, randomly generate 10 stars. (A "constellation").
Repeat many times:
for k=1:10
    Redraw the k-th star using black or yellow with equal probability
end
This box redraws the constellation.

To Simulate Twinkling...
X = rand(1,10); y = rand(1,10);
for k = 1:100
    for j=1:10
        if rand < .5
            DrawStar(x(j),y(j),r,'k')
        else
            DrawStar(x(j),y(j),r,'y')
        end
        pause(.01)
    end
end

Subscripting Review
for j=1:3
    DrawStar(a(j),b(j),.1,'y')
end
j=1   DrawStar(3,2,.1,'y')
a: 3 1 7  
b: 2 8 4

j=2
    DrawStar(1,8,.1,'y')

j=3
    DrawStar(7,4,.1,'y')

Subscripting Review
for j=1:3
    DrawStar(a(j),b(j),.1,'y')
end
j=1   DrawStar(3,2,.1,'y')
a: 3 1 7  
b: 2 8 4

j=2
    DrawStar(1,8,.1,'y')

j=3
    DrawStar(7,4,.1,'y')
a: 3 1 7  
b: 2 8 4