L14. Arrays and Functions

Functions with array parameters.
Row and column vectors

Built-Ins: length, zeros, std
Revisit: rand, randn, max

Row and Column Vectors

```
>> v = [1 2 3]
v =
 1  2  3
```

```
>> v = [1 ; 2 ; 3]
v =
  1
  2
  3
```

Observe semicolons

```
zeros(   ,   )
```

```
>> x = zeros(3,1)
x =
  0
  0
  0
```

```
>> x = zeros(1,3)
x =
  0  0  0
```

```
rand(    ,    )
```

```
>> x = rand(3,1)
x =
  0.2618
  0.7085
  0.7839
```

```
>> x = rand(1,3)
x =
  0.9862  0.4733  0.9028
```

```
randn(    ,    )
```

```
• >> x = randn(1,3)
  x =
  0.2877  -1.1465  1.1909

• >> x = randn(3,1)
  x =
  1.1892
  -0.0376
  0.3273
```

Normal Distribution with Zero Mean and Unit STD
Affirmations

```matlab
>> n = 1000000;
>> x = randn(n,1);
>> ave = sum(x)/n
ave =
-0.0017
>> standDev = std(x)
standDev =
0.9989
```

```
length

>> v = randn(1,5);
>> n = length(v)
n =
5
>> u = rand(5,1);
>> n = length(u)
n =
5
```

Augmenting Row Vectors

```matlab
>> x = [10 20]
x =
10 20
>> x = [x 30]
x =
10 20 30
```

Augmenting Column Vectors

```matlab
>> x = [10; 20]
x =
10
20
>> x = [x ; 30]
x =
10
20
30
```

Observe semicolons!

“Concatenating” Row Vectors

```matlab
>> x = [10 20]
x =
10 20
>> y = [30 40 50]
y =
30 40 50
>> z = [x y]
z =
10 20 30 40 50
```

“Concatenating” Column Vectors

```matlab
>> x = [10 ; 20];
>> y = [30 ; 40 ; 50];
>> z = [ x ; y ]
z =
10
20
30
40
50
```

Observe semicolons!
Application

Plot sine across [0, 4*pi] and use the fact that it has period 2pi.

```matlab
x = linspace(0, 2*pi, 100);
y = sin(x);
x = [x x+2*pi];
y = [y y];
plot(x, y)
```

The Empty Vector

```matlab
x = [];
for k=1:50
    if floor(sqrt(k)) == sqrt(k)
        x = [x; k];
    end
end
x = x
```

Array Hints & Errors

Dimension Mismatch

```matlab
>> x = [1; 2]
x =
    1
    2
>> y = [3 4]
y =
    3
    4
>> z = x+y
??? Error using ==> plus
   Matrix dimensions must agree.
Can't add a row vector to a column vector
```

Not a Syntax Error

```matlab
>> x = rand(3)
x =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
```

You probably meant to say
```matlab
x = rand(1,3) or x = rand(3,1).
```
A Style Hint

Assume \( n \) is initialized.

```matlab
a = zeros(1,n);
for k=1:n
    a(k) = sqrt(k);
end
```

Better because it reminds you of the size and shape of the array you set up.

Error: Out-ofRange Subscript

```matlab
>> x = [10 20 30]
x =
    10    20    30
>> c = x(4)
??? Index exceeds matrix dimensions.
```

This is OK…

```matlab
>> x = [10 20 30]
x =
    10    20    30
>> x(4) = 100
x =
    10    20    30   100
```

Forgot the Semicolon?

```matlab
>> x = randn(1000000,1)
```

```matlab
Forgot the Semicolon?

```matlab
>> x = randn(1000000,1)
```

```matlab
Remember: ctrl-C
```

Question Time

```matlab
A = [1];
while(length(A) < 5)
    A = [length(A)+1 ; A];
end
A = A
```

Is this the same as

```matlab
A = linspace(1,5,5); ?
```

A. Yes  B. No
No!

Linspace:
1 2 3 4 5

Fragment:
5
4
3
2
1

Question Time

x = zeros(1,1);
for k=1:3
    x = [x x];
end
y = x(7)

Will this cause a subscript out of bounds error?

A. Yes              B. No

No!

How x changes:

After 1st pass: [0 0]
After 2nd pass: [0 0 0 0]
After 3rd pass: [0 0 0 0 0 0 0 0]

So y = x(7) makes sense.

Polygon Transformations

Functions & arrays

A Polygon

Store xy-coordinates in vectors x and y.

Operation 1: Centralize

Move a polygon so that the centroid of its vertices is at the origin.
Before Centralize

function [xNew, yNew] = Centralize(x,y)
    n = length(x);
    xBar = sum(x)/n;
    yBar = sum(y)/n;
    xNew = x - xBar;
    yNew = y - yBar;

Computes the vertices of the new polygon

Notice how length is used to figure out the size of the incoming vectors.

Operation 2: Normalize

Shrink (enlarge) the polygon so that the vertex furthest from the (0,0) is on the unit circle.

function [xNew, yNew] = Normalize(x,y)
    d = max(sqrt(x.^2 + y.^2));
    xNew = x/d;
    yNew = y/d;

Applied to a vector, max returns the largest value in the vector.

Operation 3: Smooth

Obtain a new polygon by connecting the midpoints of the edges.
Smooth

function [xNew, yNew] = Smooth(x, y)

n = length(x);
xNew = zeros(n, 1);
yNew = zeros(n, 1);

for i=1:n
    Compute the mdpt of ith edge.
    Store in xNew(i) and yNew(i)
end

xNew(1) = (x(1)+x(2))/2
yNew(1) = (y(1)+y(2))/2

xNew(2) = (x(2)+x(3))/2
yNew(2) = (y(2)+y(3))/2

xNew(3) = (x(3)+x(4))/2
yNew(3) = (y(3)+y(4))/2

xNew(4) = (x(4)+x(5))/2
yNew(4) = (y(4)+y(5))/2

xNew(5) = (x(5)+x(1))/2
yNew(5) = (y(5)+y(1))/2

Midpoints

(c,d)
(a,b)
\[ x_{\text{New}}(4) = \frac{x(4) + x(5)}{2} \]
\[ y_{\text{New}}(4) = \frac{y(4) + y(5)}{2} \]

\[ x_{\text{New}}(5) = \frac{x(5) + x(1)}{2} \]
\[ y_{\text{New}}(5) = \frac{y(5) + y(1)}{2} \]

Smooth

for \( i = 1:n \)
\[
x_{\text{New}}(i) = \frac{x(i) + x(i+1)}{2};
\]
\[
y_{\text{New}}(i) = \frac{y(i) + y(i+1)}{2};
\]
end

Will result in a subscript out of bounds error when \( i \) is \( n \).

Smooth

for \( i = 1:n-1 \)
\[
x_{\text{New}}(i) = \frac{x(i) + x(i+1)}{2};
\]
\[
y_{\text{New}}(i) = \frac{y(i) + y(i+1)}{2};
\]
end
\[
x_{\text{New}}(n) = \frac{x(n) + x(1)}{2};
\]
\[
y_{\text{New}}(n) = \frac{y(n) + y(1)}{2};
\]
Proposed Simulation

Create a polygon with randomly located vertices.

Repeat:
  Centralize
  Normalize
  Smooth