CS 1112 Introduction to Computing Using MATLAB

Instructor: Dominic Diaz

Website:
https://www.cs.cornell.edu/courses/cs1112/2022fa/

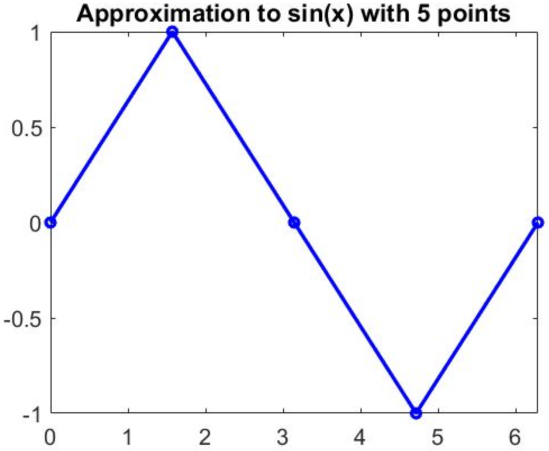
Today: Vectors (1D arrays)

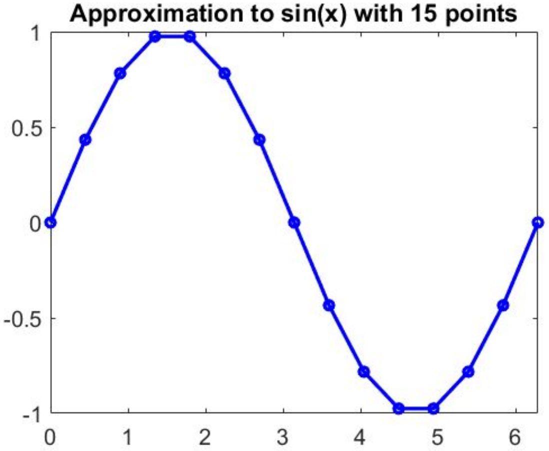
Agenda and announcements

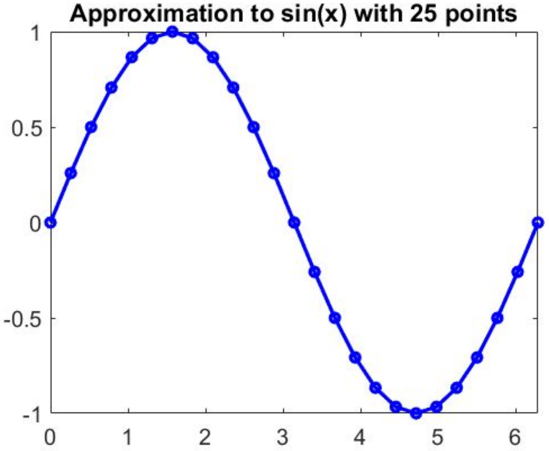
- Last time
  - More vectors (initializing vectors, plotting vectors, random numbers, and examples
- Today
  - Color computation with linear interpolation!
- Announcements
  - Project 3 due Wednesday 10/5
  - Yesterday’s discussion
    - Get first parts checked off by consultant or TA
    - Submit last parts on MATLAB grader

Discrete approximation of functions

Approximate the function sine by plotting N discrete points.


Approximation to sin(x) with 5 points

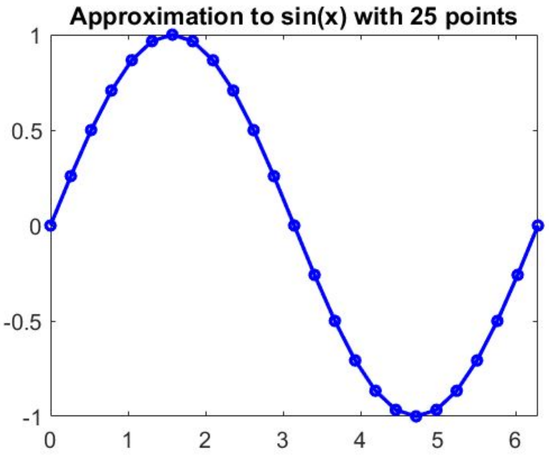

Approximation to sin(x) with 15 points

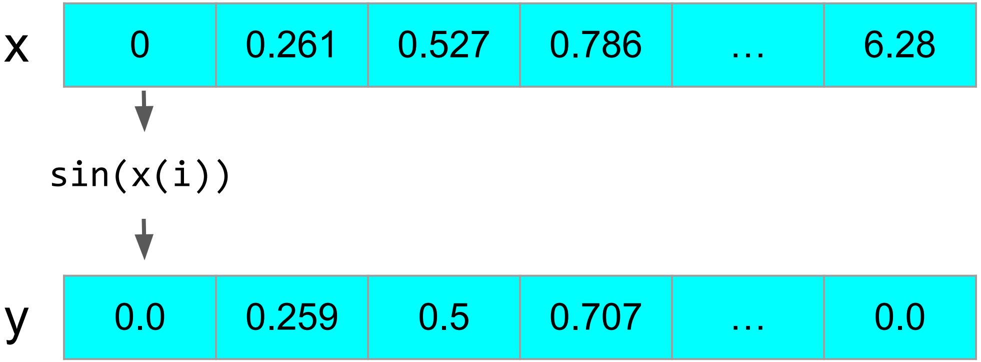

Approximation to sin(x) with 25 points

Plots are made from discrete points with lines between those points, but it can look continuous if there are many points!

Plotting the sine function

```
% plot the sine function with numPts number of points
numPts = 25;
x = linspace(0, 2*pi, numPts);

y = zeros(1, length(x));
for i = 1:length(x)
   y(i) = sin(x(i));
end

plot(x,y, "-ob")
```


Approximation to sin(x) with 25 points



x	0	0.261	0.527	0.786	…	6.28
y	0.0	0.259	0.5	0.707	…	0.0

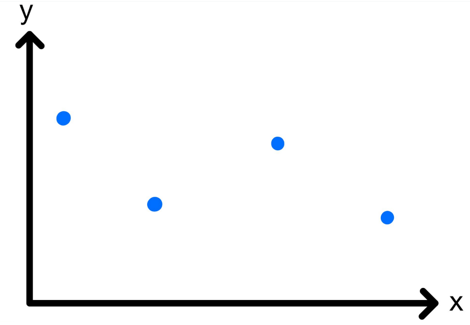
Interpolation

Interpolation is a method of constructing new data points based on information at known data points. Many different interpolation schemes:

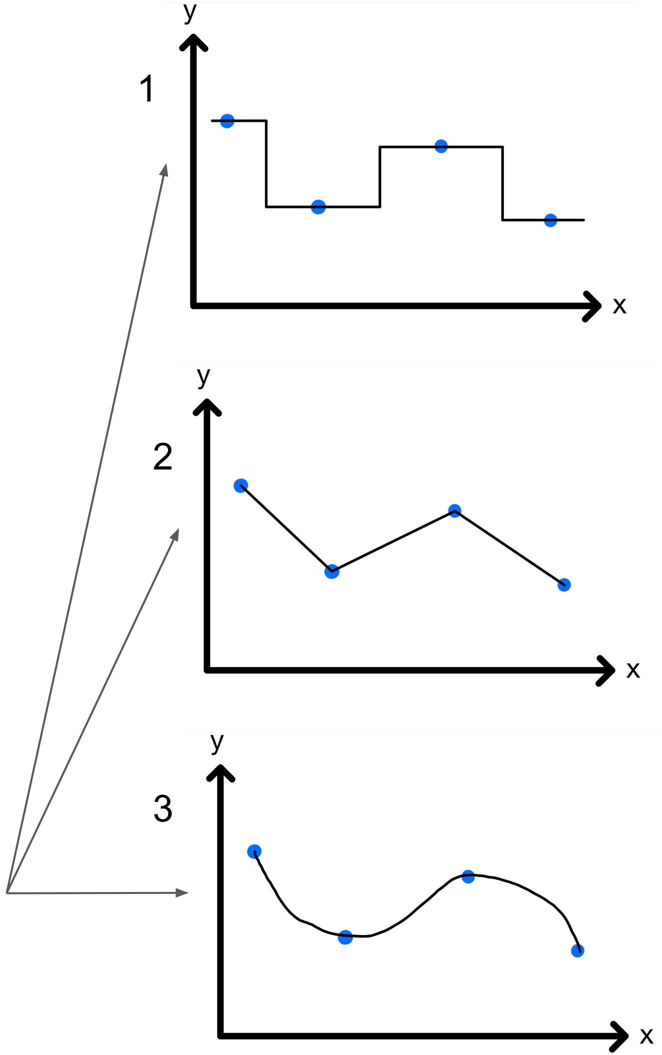
1. Copy value of closest known data point
2. Linearly interpolate between adjacent points
3. Interpolate across more points than two

“Best” choice depends on what you know about the data

Given these data points:



Interpolate to find values between the known values



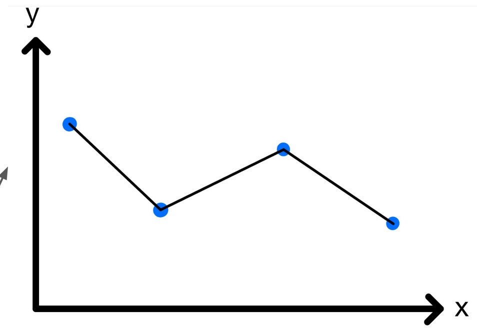
Interpolation

Should know how to do linear interpolation for prelim/final

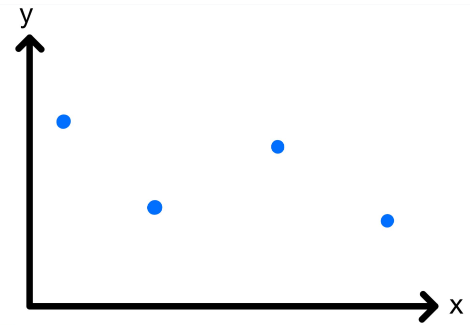
Interpolation is a method of constructing new data points based on information at known data points. Many different interpolation schemes:

1. Copy value of closest value
2. Linearly interpolate between adjacent points
3. Interpolate across more points than two

"Best" choice depends on what you know about the data



Given these data points:

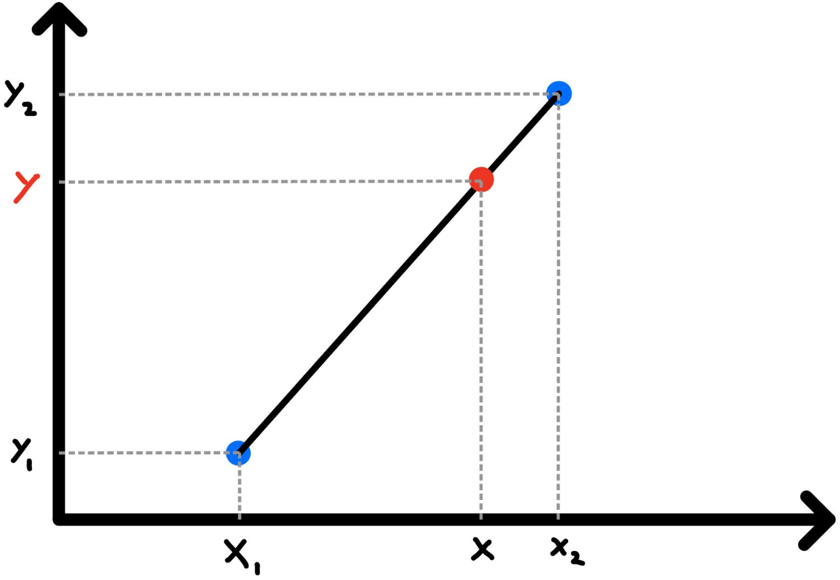


Interpolate to find values between the known values

Formula for linear interpolation

Given points (x1,y1) and (x2,y2), interpolate between these two points: given some new x in the interval (x1, x2), calculate the corresponding y.

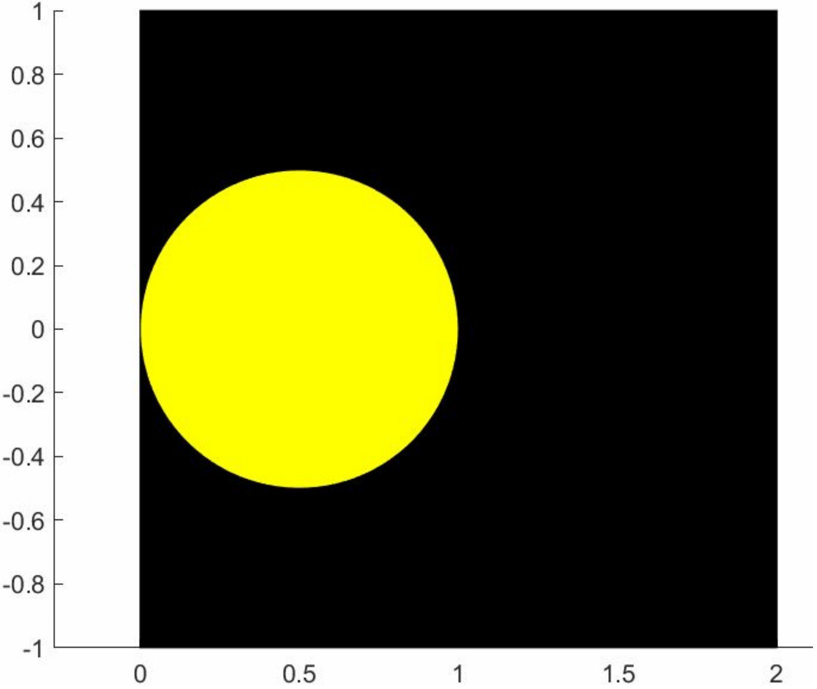
How? Solve for y in terms of x:
Note: the slope from (x1, y1) to (x,y) is the same as (x1,y1) to (x2, y2).



$$\frac{y - y1}{x - x1} = \frac{y2 - y1}{x2 - x1}$$

$$y = y1 + \frac{y2 - y1}{x2 - x1}(x\text{-}x1)$$

Example: Draw shrinking disks with interpolated colors between yellow and black



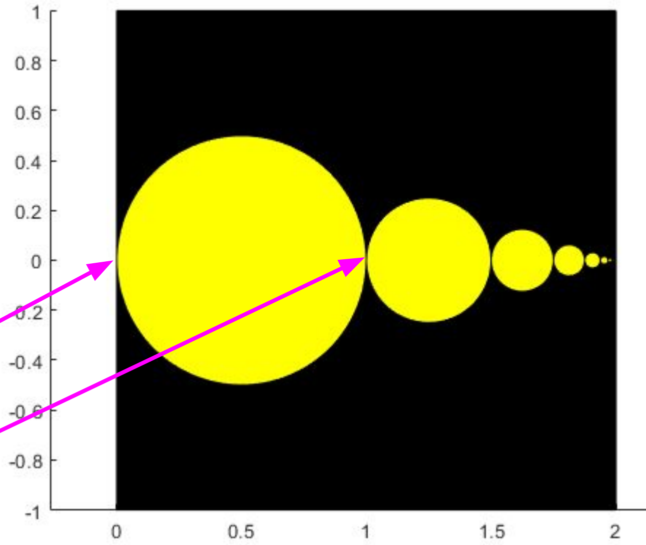
Draw n shrinking disks with color that fade into the background. Starting color is yellow and background color is black. The (k+1)th disk has half the diameter of the kth disk.

The first disk should have radius 0.5 and it’s left-most point should be at (0,0). Disks are tangent to each other and have centers on the x-axis.

Let’s start with an easier problem:
Forget color for now and let’s just draw the disks!

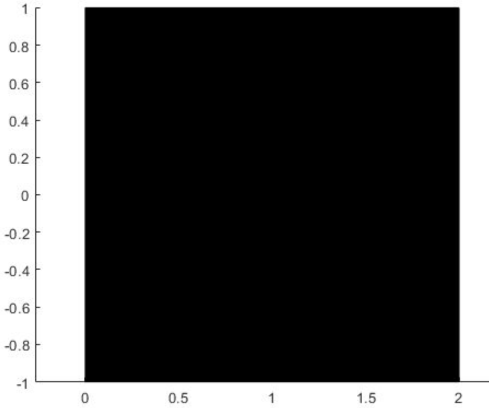
Draw yellow shrinking disks

Draw n shrinking disks whose centers are all along the x-axis. The first disk should have radius 0.5 and it’s leftmost point should be at (0,0). Consecutive disks should be tangent to each other and (k+1)th disk has half the radius of the kth disk.

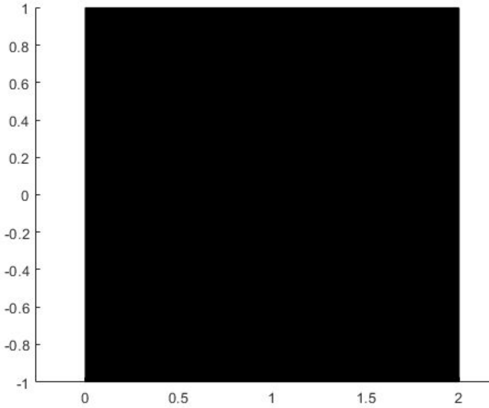


Disk	left tangent point	radius
1	0	0.5
2	0 + 1	0.25
3	0 + 1 + 1/2	0.125
4	0 + 1 + 1/2 + 1/4	0.0625

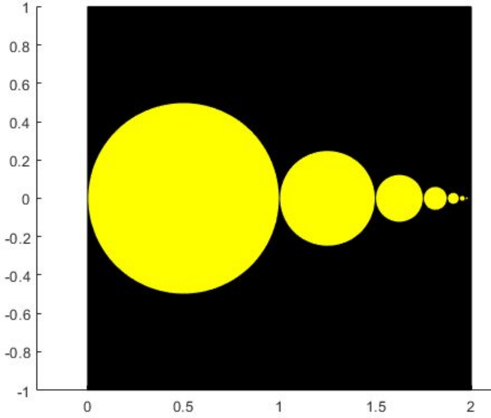
```
close all
figure
axis ([0 2 -1 1])
axis equal
hold on
% Draw black background
DrawRect(0,-1,2,2,'k')


% Create variables so we can draw the first disk




% Draw sequence of disks
for k = 1:n

     % Draw disk

     % update left tangent location and radius for next disk




end
hold off
```



```
close all
figure
axis ([0 2 -1 1])
axis equal
hold on
% Draw black background
DrawRect(0,-1,2,2,'k')
x = 0;                % Left tangent point of first disk
r = 0.5;              % radius of first disk

n = 10;               % Number of disks to be drawn
yellow = [1 1 0];

% Draw sequence of disks
for k = 1:n

     % Draw disk

     % update left tangent location and radius for next disk



end
hold off
```



```
close all
figure
axis ([0 2 -1 1])
axis equal
hold on
% Draw black background
DrawRect(0,-1,2,2,'k')
x = 0;                % Left tangent point of first disk
r = 0.5;              % radius of first disk

n = 10;               % Number of disks to be drawn
yellow = [1 1 0];

% Draw sequence of disks
for k = 1:n



     DrawDisk(x+r, 0, r, yellow)
     pause(.5)
     x = x+2*r;
     r = r/2;
end
hold off
```



Now let’s deal with the fading color!

Colors in MATLAB

Colors can be represented by a vector of length 3 storing RGB values

Examples: `colr = [1, 0, 0]` corresponds to red
`colr = [1, 1, 1]` corresponds to white
`colr = [0, 0, 0]` corresponds to black
`colr = [1, 1, 0]` corresponds to yellow

Each element of the vector must be a value between 0 and 1.

Color options

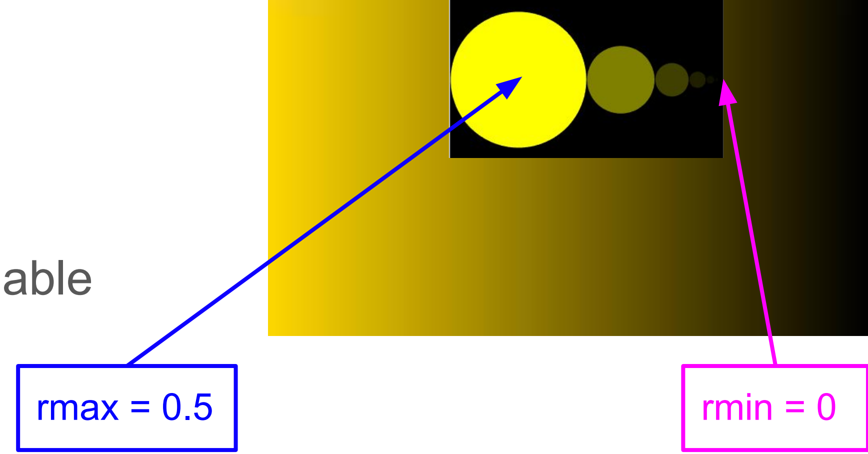
White	'w' or [1 1 1]	⬜
Black	'k' or [0 0 0]	⬛
Red	'r' or [1 0 0]	🟥
Blue	'b' or [0 0 1]	🟦
Green	'g' or [0 1 0]	🟩
Yellow	'y' or [1 1 0]	🟨
Magenta	'm' or [1 0 1]	🟪
Cyan	'c' or [0 1 1]	🟦
Random color	[0.6350 0.0780 0.1840]	🟥

Interpolated colors

Let's determine the color interpolation

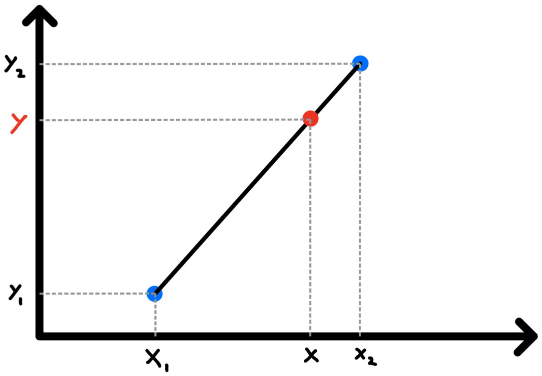
Steps in linear interpolation:

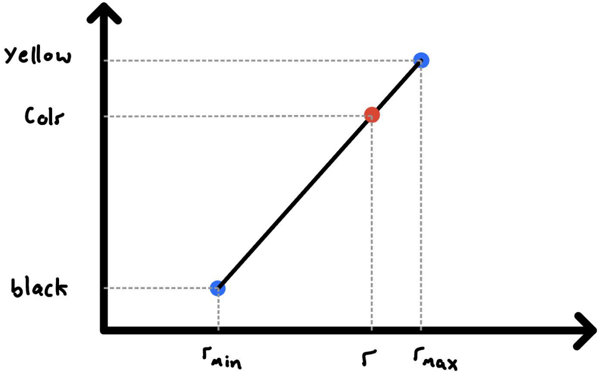
1. Determine independent and dependent variable
2. Draw plot
3. Use formula



rmax = 0.5

rmin = 0





(x1, y1) = (rmin, black)
(x2, y2) = (rmax, yellow)

(x, y) = (r, colr)

$$y = y1 + \frac{y2 - y1}{x2 - x1}(x-x1)$$

```
colr = black + (yellow-black)*(r-rmin)/(rmax-rmin);
```

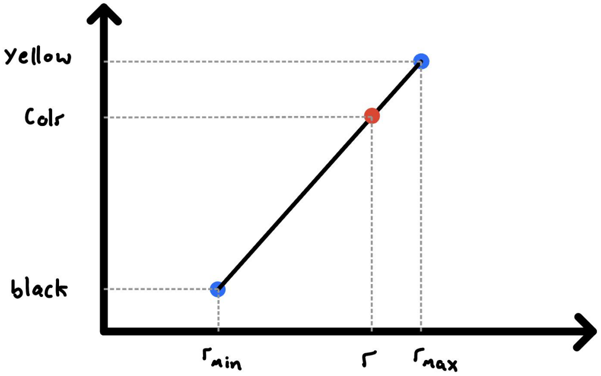
Sanity check for color interpolation

When r = rmax = 0.5, we expect colr = yellow

When r = rmin = 0, we expect colr = black

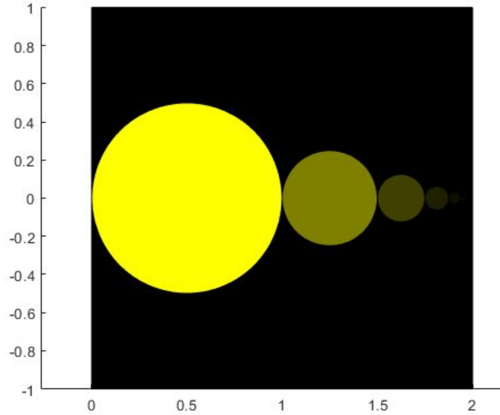
When r = 0.25, we expect colr to be something between black and yellow

```
r = rmax               ⇒ colr = yellow
r = rmin               ⇒ colr = black
r = (rmax+rmin)/2      ⇒ colr = yellow/2 + black/2
```



```
colr = black + (yellow-black)*(r-rmin)/(rmax-rmin);
```

```
close all
figure
axis ([0 2 -1 1])
axis equal
hold on
% Draw black background
DrawRect(0,-1,2,2,'k')
x = 0;                % Left tangent point of first disk
r = 0.5;              % radius of first disk
rmax = r; rmin = 0;   % max and min disk radius values (for interpolation)
n = 10;               % Number of disks to be drawn
yellow = [1 1 0];
black = [0 0 0];
% Draw sequence of disks
for k = 1:n
     % colr is the color vector of the kth disk
     % use equation y = y1 + (y2-y1)*(x-x1)/(x2-x1)
     colr = black + (yellow-black)*(r-rmin)/(rmax-rmin);
     DrawDisk(x+r, 0, r, colr)
     pause(.5)
     x = x+2*r;
     r = r/2;
end
hold off
```



1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2