ECE 111 - Homework #3

Week #3: Trigonometry

Polar to Rectangular Conversions

1) Determine the final position of A: (x,y)

$$A = (16\angle 35^0) + (6\angle 81^0) + (1\angle -11^0)$$

In Matlab, use the identity

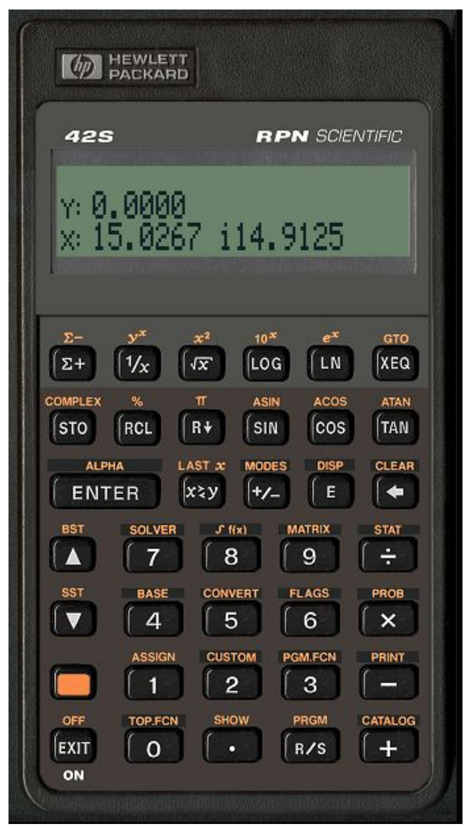
$$r\angle\theta \equiv (r \cdot \cos(\theta), r \cdot \sin(\theta))$$

Note that Matlab uses radians for angles.

```
>> x1 = 16*cos(35*pi/180);
>> y1 = 16*sin(35*pi/180);
>> x2 = 6*cos(81*pi/180);
>> y2 = 6*sin(81*pi/180);
>> x3 = 1*cos(-11*pi/180);
>> y3 = 1*sin(-11*pi/180);
>> Ax = x1 + x2 + x3

Ax =   15.0267

>> Ay = y1 + y2 + y3

Ay =   14.9125
```

This is easier on an HP42 calculator (Free42 app)

```
modes
polar
16
enter
35
complex
6
enter
81
complex
+
1
enter
-11
complex
+
modes
rect
```



2) Determine final position of B: (x,y)

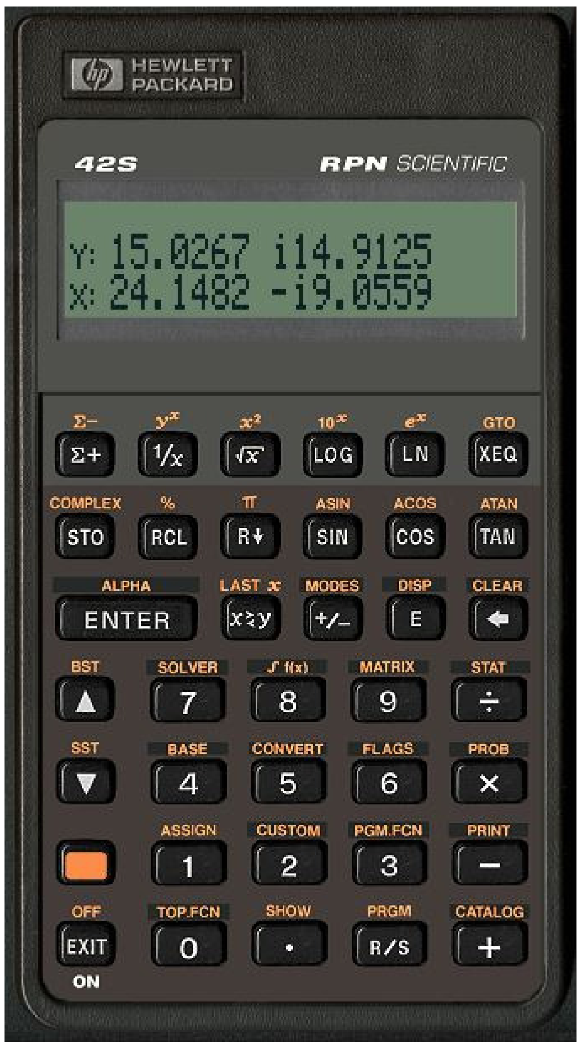
$$B = (8\angle 48^0) + (16\angle -56^0) + (10\angle -10^0)$$

In Matlab

```
>> x1 = 8*cos(48*pi/180);
>> y1 = 8*sin(48*pi/180);
>> x2 = 16*cos(-56*pi/180);
>> y2 = 16*sin(-56*pi/180);
>> x3 = 10*cos(-10*pi/180);
>> y3 = 10*sin(-10*pi/180);
>> Bx = x1 + x2 + x3

Bx =   24.1482

>> By = y1 + y2 + y3

By =   -9.0559
```

With an HP42

```
modes
polar
8
enter
48
complex
16
enter
-56
complex
+
10
enter
-10
complex
+
modes
rect
```



note:

- the result from the previous calculation is still on the stack
- press *0 enter enter enter* to clear the stack

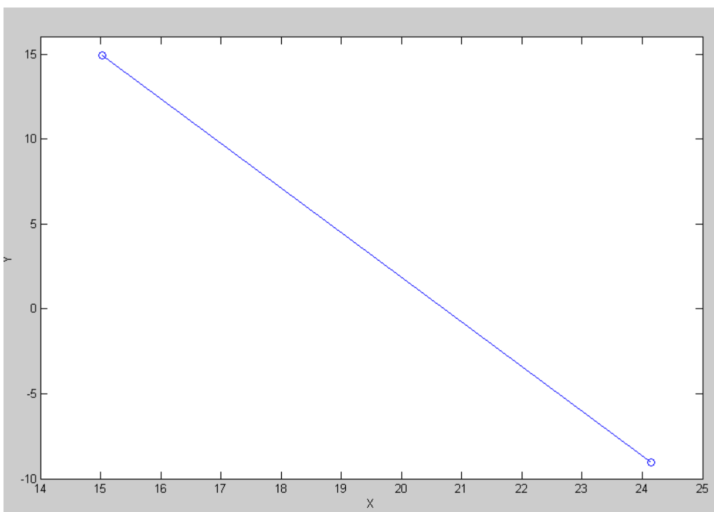
3) Where is B relative to A (i.e. what is C = B - A?)

- In (x,y) coordinates
- In polar coordinates

```
>> Cx = Bx - Ax

Cx =    9.1215

>> Cy = By - Ay

Cy =  -23.9685
```

Point B is

- **9.12 units to the right of A**
- **23.96 units down from A**



Point B (blue) and A (red)

Plotting Polar Functions

4) Plot the following functions in Matlab for $-2\pi < \theta < 2\pi$

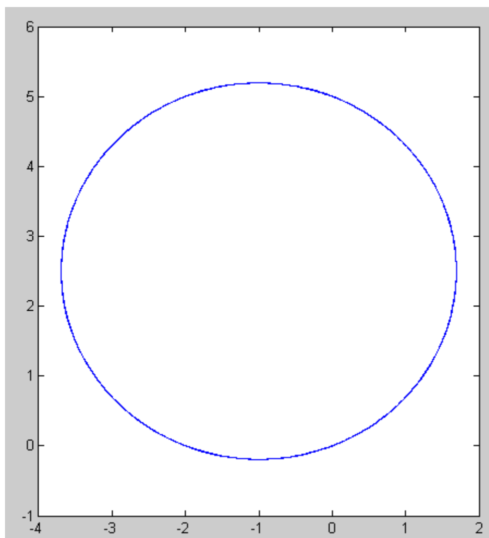
- Note: plot() plots in cartesian coordinates. Each function needs to be converted from polar to rectangular.

a) $r = 5\sin(\theta) - 2\cos(\theta)$

Matlab Code

```
q = [-2*pi : 0.01 : 2*pi]';
r = 5*sin(q) - 2*cos(q);
x = r .* cos(q);
y = r .* sin(q);
plot(x,y);
```

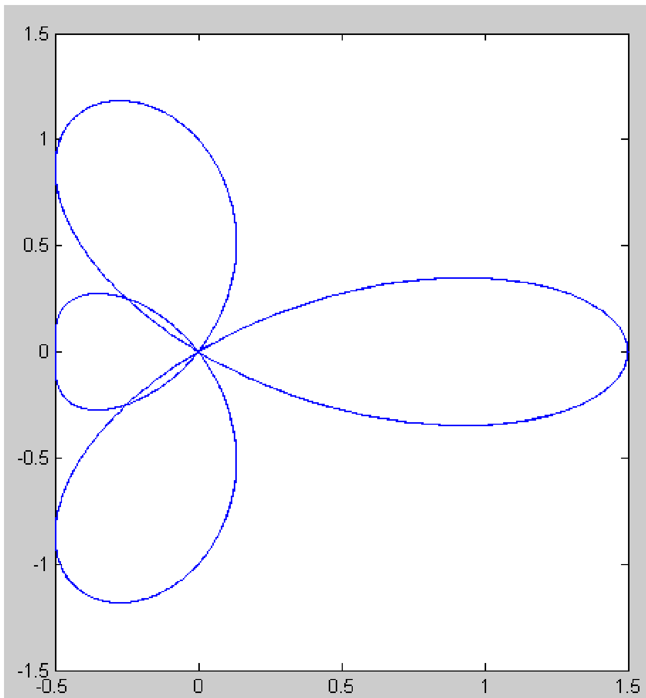
Not too surprisingly, you get a circle



b) $r = \cos(2\theta) + 0.5\cos(3\theta)$

Change one line of code:

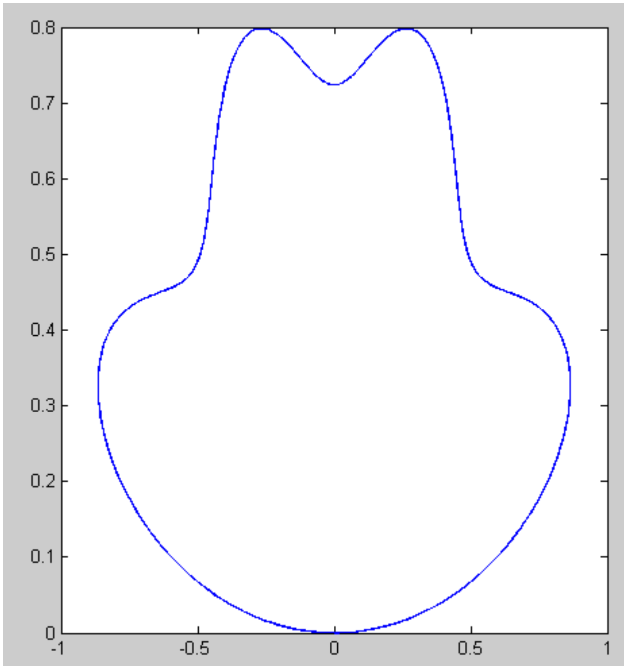
```
q = [-2*pi : 0.01 : 2*pi]';
r = cos(2*q) + 0.5*cos(3*q);
x = r .* cos(q);
y = r .* sin(q);
plot(x,y);
```



c) $r = \sin(\theta) + \frac{1}{3}\sin(3\theta) + \frac{1}{5}\sin(5\theta) + \frac{1}{7}\sin(7\theta)$

Matlab Code

```
q = [-2*pi : 0.01 : 2*pi]';
r = sin(q) + sin(3*q)/3 + sin(5*q)/5 + sin(7*q)/7;
x = r .* cos(q);
y = r .* sin(q);
plot(x,y);
```



Robot Tip Position (Forward Kinematics)

A 2D robot has three arms with lengths of {2.0, 1.0, 1.0} meters. The final tip position is

$$x_1 = 2\cos(\theta_1) \qquad y_1 = 2\sin(\theta_1)$$

$$x_2 = x_1 + \cos(\theta_1 + \theta_2) \qquad y_2 = y_1 + \sin(\theta_1 + \theta_2)$$

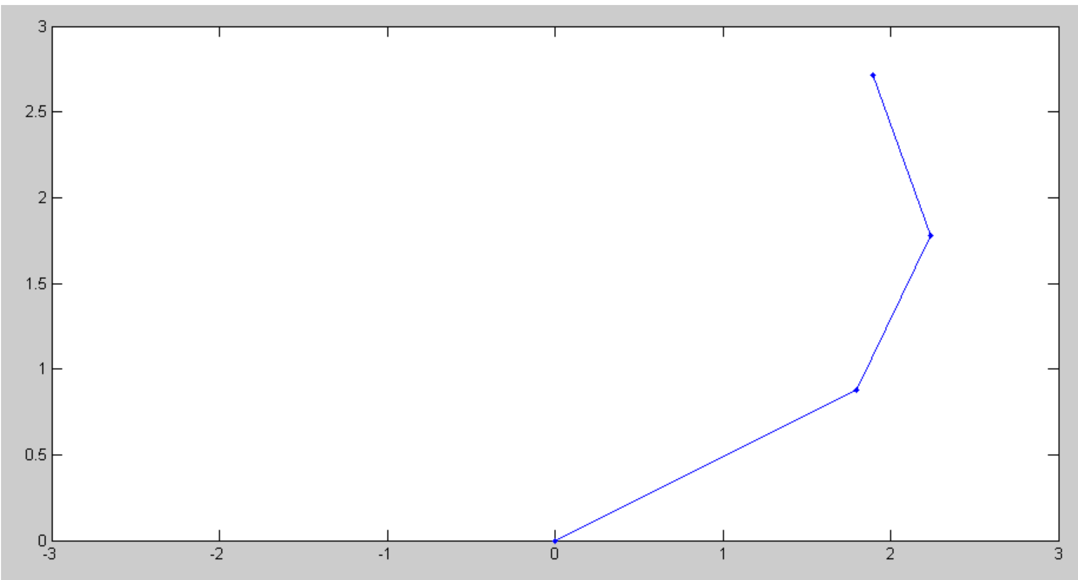
$$x_3 = x_2 + \cos(\theta_1 + \theta_2 + \theta_3) \qquad y_3 = y_2 + \sin(\theta_1 + \theta_2 + \theta_3)$$

5) Plot the tip position (x3, y3) for

$$\theta_1 = 26^0 \quad \theta_2 = 38^0 \quad \theta_3 = 46^0$$

Matlab Code

```
q1 = 26*pi/180;
q2 = 38*pi/180;
q3 = 46*pi/180;
x0 = 0;
y0 = 0;
x1 = 2*cos(q1);
y1 = 2*sin(q1);
x2 = x1 + cos(q1+q2);
y2 = y1 + sin(q1+q2);
x3 = x2 + cos(q1+q2+q3);
y3 = y2 + sin(q1+q2+q3);
disp('Tip Position')
disp([x3,y3])
plot([x0,x1,x2,x3],[y0,y1,y2,y3],'b.-')
xlim([-3,3])

Tip Position
    1.8939    2.7152
```

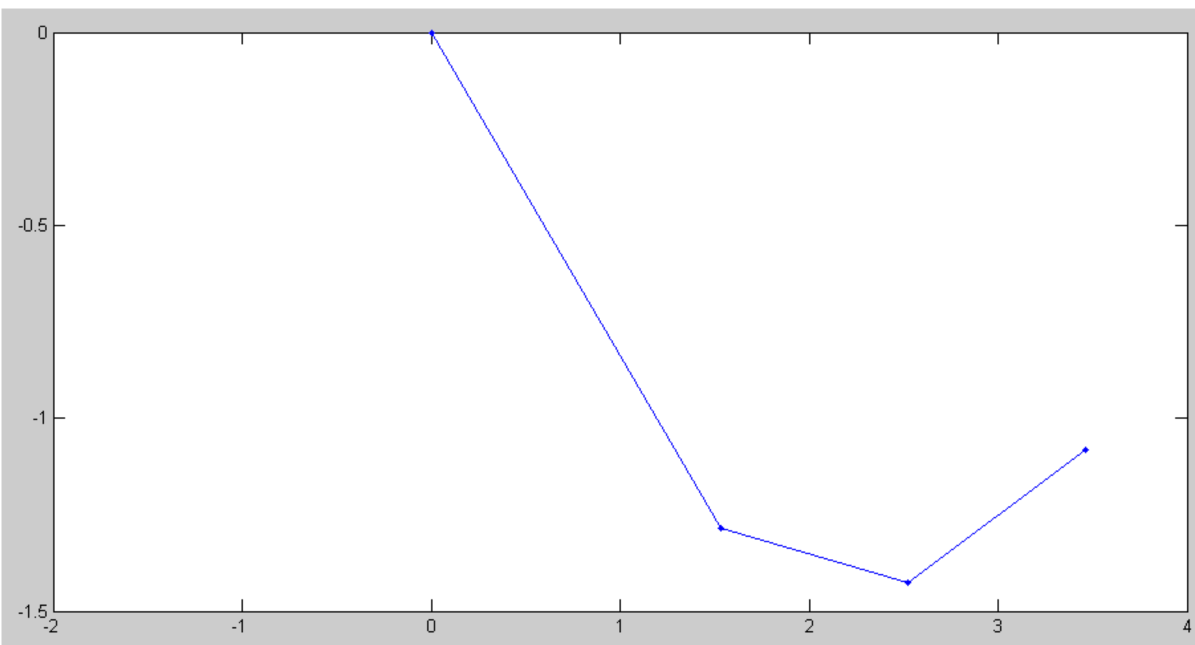


6) Plot the tip position (x3, y3) for

$$\theta_1 = -40^0 \qquad \theta_2 = 32^0 \qquad \theta_3 = 28^0$$

Matlab Code

```
q1 = -40*pi/180;
q2 = 32*pi/180;
q3 = 28*pi/180;
x0 = 0;
y0 = 0;
x1 = 2*cos(q1);
y1 = 2*sin(q1);
x2 = x1 + cos(q1+q2);
y2 = y1 + sin(q1+q2);
x3 = x2 + cos(q1+q2+q3);
y3 = y2 + sin(q1+q2+q3);
disp('Tip Position')
disp([x3,y3])
plot([x0,x1,x2,x3],[y0,y1,y2,y3],'b.-')
xlim([-2,4])

Tip Position
    3.4620   -1.0827
```



Robot Tip Position (Inverse Kinematics & fminsearch())

7) Write a Matlab function which

- Is passed the angles $(\theta_1, \theta_2, \theta_3)$,
- Computes the tip position, and
- Returns the distance from the tip position and point (x = 1.0, y = 2.0)

create a function which

- is passed the joint angles in degrees, and
- returns the tip position

```
function [x3, y3] = RRR(q1, q2, q3)

   % convert to radians
   q1 = q1 * pi / 180;
   q2 = q2 * pi / 180;
   q3 = q3 * pi / 180;

   % compute the joint positions
   x0 = 0;
   y0 = 0;

   x1 = x0 + 2*cos(q1);
   y1 = y0 + 2*sin(q1);

   x2 = x1 + cos(q1+q2);
   y2 = y1 + sin(q1+q2);

   x3 = x2 + cos(q1+q2+q3);
   y3 = y2 + sin(q1+q2+q3);

   plot([x0,x1,x2,x3],[y0,y1,y2,y3],'b.-');
   ylim([-1,4]);
   xlim([-4,4]);
   pause(0.01);

   end
```

Call this from a function which

- is passed the angles in an array (z), and
- returns the distance to the target (1,2)

```
function [J] = RRR_Cost(z)

   x0 = 1;
   y1 = 2;

   q1 = z(1);
   q2 = z(2);
   q3 = z(3);

   [x3, y3] = RRR(q1, q2, q3);

   J = sqrt( (x3-x0)^2 + (y3-y0)^2 );

   end
```

8) Use the fminsearch() to determine the joint angles which place the robot at (x3 = 1.0, y3 = 2.0)

Solve using fminsearch()

```
>> [Z,e] = fminsearch('RRR_Cost',[10,20,30])

         q1        q2        q3
Z =   17.7708   71.2916   68.0008

e =   9.1862e-007

>>
```

note:

- there are an infinte number of solutions
- different initial guesses will converge to a different solution