Exercise 1: Problem 9.6

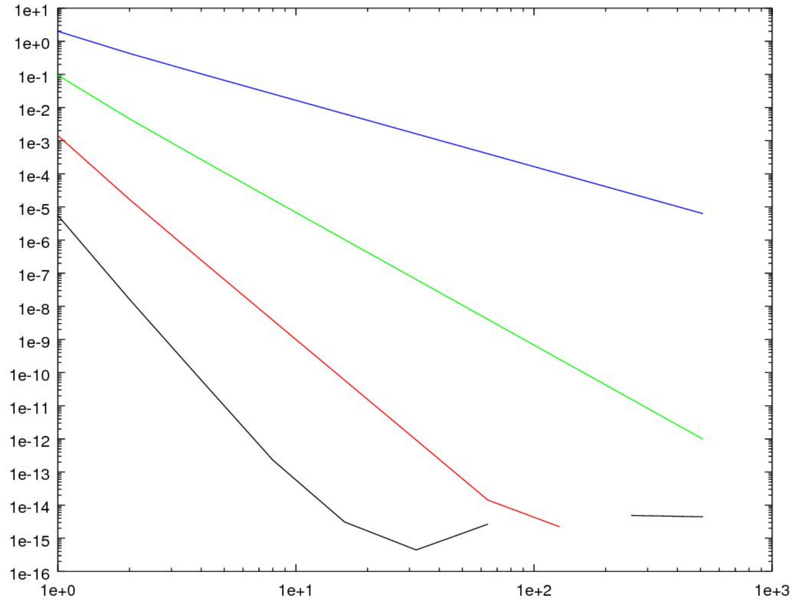
Note that the taylor estimation is $f(x+c) = f(x)+f'(x)c+f''(x)c^2/2+f'''(x)c^3/6+O(c^4)$. We are asked to find the values of the constants in the equation $f''(x) \approx Af(x)+Bf(x+h)+Cf(x+2h)$ that give the maximal acuracy in terms of the magnitude of h. Note that using taylor's approximation we get $f''(x) \approx Af(x)+B[f(x)+f'(x)h+f''(x)h^2/2+f'''(x)h^3/6]+C[f(x)+2f'(x)h+2f''(x)h^2+4f'''(x)h^3/3]+O(h^4)$. We then get the equations $A+B+C = 0$ to link values, $Bh+2Ch = 0$, $Bh^2/2+2Ch^2 = 1$. Solving we get $-B = 2C$, $Bh^2/2-Bh^2 = 1$, $Bh^2(1/2-1) = -Bh^2/2 = 1$, $B = -2/h^2$, $C = 1/h^2$, $A = 1/h^2$. To determine order accuracy let's ask if we also happen to get $0f'''(x)$, $Bh^3/6 + C4h^3/3 = 0$, $B + 8C = 0$, clearly false, thus we get error of order $O(h^3/h^2) = O(h)$.

Exercise 2: Turn in the last problem from the worksheet on Richardson extrapolation.

```
function mintegral=traprule(f,a,b,nm)
    mintegral=[];
    for(n=nm)
        w=(b-a)./n;
        cent=linspace(a+w,b-w,n-1);
        integral=w.*(1/2*f(a)+sum(f(cent))+1/2*f(b));
        mintegral=[mintegral integral];
    end
endfunction
```

```
I0=@(n) arrayfun( @(k) traprule(@(x) sin(x),0,pi,k), n);
I1=@(n) (I0(n*2)- 2^(-2).*I0(n)  )./(1-2^(-2));
I2=@(n) (I1(n*2)- 2^(-4).*I1(n)  )./(1-2^(-4));
I3=@(n) (I2(n*2)- 2^(-6).*I2(n)  )./(1-2^(-6));
N=2.^[0:9];
loglog(N,abs(I0(N)-2),'b');
hold on;
loglog(N,abs(I1(N)-2),'g');
loglog(N,abs(I2(N)-2),'r');
loglog(N,abs(I3(N)-2),'k');
hold off;
```

This is a plot of the $Ik(N)$ lines for $k = [0 : 3]$ notice that there slopes are $-2, -4, -6, -8$ indicating that the error goes as $O(n^{-2}), O(n^{-4}), O(n^{-6}), O(n^{-8})$.



Exercise 3: Problem 10.8. Note that Romberg integration is just Richardson extrapolation applied to the trapezoidal rule, as you did in the previous problem.

```
>> [q,cnt]=romberg(0,1,1e-12)
    3.0000e+00   9.0266e-01   1.0000e-10
    5.0000e+00   9.0462e-01   1.0000e-10
    9.0000e+00   9.0452e-01   9.9983e-05
    1.7000e+01   9.0452e-01   1.1338e-07
    3.3000e+01   9.0452e-01   1.3115e-09
    6.5000e+01   9.0452e-01   4.8783e-13
warning: Matlab-style short-circuit operation performed for operator &
warning: called from
    romberg at line 52 column 1
q =  0.90452
cnt =  65
```

```
>> n=2.^[2:7;]
n =

       4     8    16    32    64   128

>> I0=@(n) arrayfun (@(k) traprule (@(x) cos (x.^2), 0, 1, k), n)
I0 =

@(n) arrayfun (@(k) traprule (@(x) cos (x .^ 2), 0, 1, k), n)

>> abs(I0(n)-q).*n.^2
ans =

   0.14025   0.14025   0.14025   0.14025   0.14025   0.14025

>> sqrt(0.14025/1e-12)
ans =    3.7450e+05
>> diary off
```

The trapazodal rule will take arround $300,000$ steps to accheve the desired accuracy. This is because the error in the trapazoid rule goes as $O(n^2)$ where as the Romberg integration is just Richardson extrapolation and increases the order of convergence as the number of steps increases.

Exercise 4: Problem 11.1 (a,b)

Solve analitically

(a)

$$\frac{dy}{dt} = t^3$$

$$y = \int dy = \int t^3 dt = t^4/4 + C$$

$$y = t^4/4$$

(b)

$$\frac{dy}{dt} = 2y$$

$$\int \frac{1}{y} dy = \int 2dt$$

$$\ln(y) = 2t + c$$

$$\ln(3) = 2 + c$$

$$\ln(y) = 2t + \ln(3) - 2$$

(c)

$$\frac{dy}{dt} = ay + b$$

$$\text{Ansatz: } y = ce^{\alpha t} + \beta$$
$$\alpha ce^{\alpha t} = ace^{\alpha t} + a\beta + b$$
$$\alpha = a \text{ and } \beta = -b/a$$
$$y = ce^{at} - b/a$$
$$y_0 = ce^{a0} - b/a$$
$$y = (y_0 + b/a)e^{at} - b/a$$

Exercise 5: Problem 11.3

If $y = t^{3/2}$ then $y' = 3/2t^{1/2}$. Noting that $t^{1/2} = y^{1/3}$ we see $y' = 3/2y^{1/3}$ and note that $(0, 0)$ is a solution to $y = t^{3/2}$. Clearly $y = t^{3/2}$ is a solution to the given ODE. As a demonstration I will use one step of euler method on this problem. Note that $y_1 = 0 + 0 * h$ and $t_1 = h$. By inspection eular method will give us $y = 0$ always. Note that $y = 0$ is actually a solution to the given ODE and initial conditions, this is the problem, there are multiple solutions to the ODE passing thrugh $(0, 0)$.

Exercise 6: Problem 11.4

```
>> dy=@(y,t) (t+1)*e^(-y)
dy =

@(y, t) (t + 1) * e ^ (-y)

>> #euler
>> h=.1
h =  0.10000
>> y1=dy(0,0)*h
y1 =  0.10000
>> #midpoint
>> temp=dy(0,0)*h/2
temp =  0.050000
>> y1=dy(temp,h/2)*h
y1 =  0.099879
>> #now for Heun's method
>> temp=dy(0,0)*h
temp =  0.10000
>> y1=(dy(0,0)+dy(temp,h))/2*h
y1 =  0.099766
>> diary off
```

Exercise 7: Problem 11.6

For $y'(y) = \lambda y$ we know that the slpoes needed for RK4 will be

$$s_1 = y'(y_k) = \lambda y_k$$

$$s_2 = y'(y_k + s_1 * h/2) = \lambda(y_k + \lambda y_k * h/2) = \lambda y_k + \lambda^2 y_k * h/2$$
$$s_3 = y'(y_k + s_2 * h/2) = \lambda(y_k + (\lambda y_k + \lambda^2 y_k * h/2) * h/2) = \lambda y_k + \lambda^2 y_k * h/2 + \lambda^3 y_k * (h/2)^2$$
$$s_4 = y'(y_k + s_3 * h) = \lambda(y_k + s_3 * h) = \lambda y_k + \lambda^2 y_k h + \lambda^3 y_k * h^2/2 + \lambda^4 y_k * h^3/4$$
$$y_{k+1} = y_k + h/6(s_1 + 2s_2 + 2s_3 + s_4) = y_k + h/6(6\lambda y_k + 3\lambda^2 y_k * h + \lambda^3 y_k * h^2 + \lambda^4 y_k * h^3/4) =$$
$$[1 + \lambda h + \lambda^2 h^2/2 + \lambda^3 h^3/6 + \lambda^4 h^4/24)]y_k$$

Exercise 8: Problem 11.14 (a,b)

(a) The new system of equations would be

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} z \\ w \\ \frac{-x}{(x^2+y^2)^{2/3}} \\ \frac{-y}{(x^2+y^2)^{2/3}} \end{bmatrix}$$

(b) This is my general orbit evaluator

```
function out=orbit(pos,dt,t,m);
#pos=[x,y,dx,dy]
distance=@(pos) sqrt(pos(1)^2+pos(2)^2);
dpos=@(pos,t) [pos(3),pos(4),-pos(1)/distance(pos)^3,-pos(2)/distance(pos)^3];
n=ceil(t/dt);
px=zeros(1,n);
py=zeros(1,n);
for i=1:n
    px(i)=pos(1);
    py(i)=pos(2);
    pos=m(pos,dpos,dt,t);
end
plot(px,py);
axis('equal');
```

and here are euler and RK4

```
function out=euler(pos,dpos,dt,t)
    out=pos+dpos(pos,t)*dt;
endfunction
```

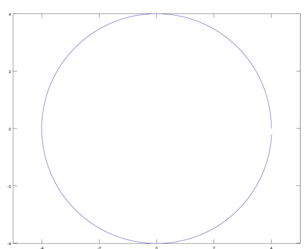
```
function out=RK4(pos,dpos,dt,t)
    s1=dpos(pos,t);
    s2=dpos(pos+s1*dt/2,t+dt/2);
    s3=dpos(pos+s2*dt/2,t+dt/2);
    s4=dpos(pos+s3*dt,t);
    out=pos+(s1+2*s2+2*s3+s4)*dt/6;
endfunction
```

here is the evaluation

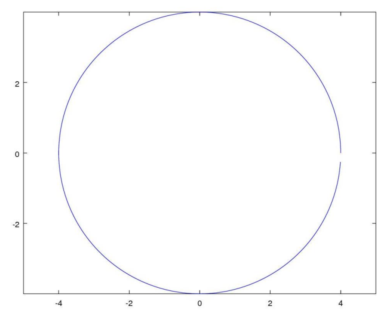
```
>> orbit([4,0,0,.5],.0025,50,@(pos,dpos,dt,t) euler(pos,dpos,dt,t))
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> diary off
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.6],.5,150,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.8],.5,200,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.4],.25,35,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.2],.25,30,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.2],.05,30,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> diary off
```

and the figures

euler

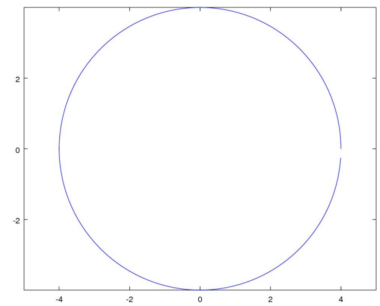


RK4

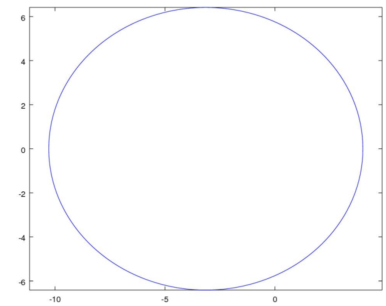


And here ar the requested runs notation is [w(0),h,tmax]

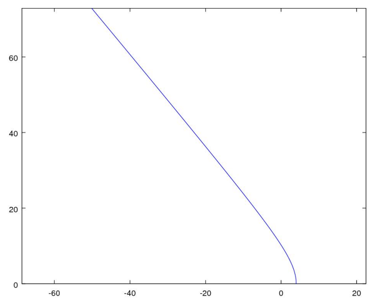
[0.5, 0.25, 50]



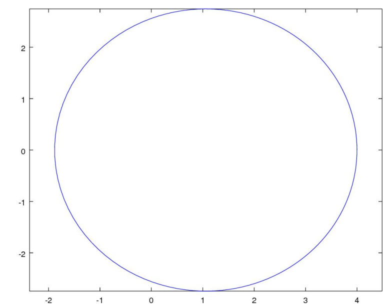
[0.6, 0.5, 150]



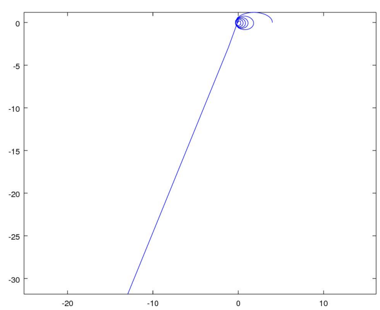
[0.8, 0.5, 200]



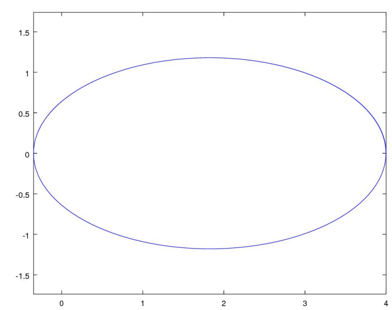
[0.4, 0.25, 35]



[0.2, 0.25, 30]



[0.2, 0.05, 30]



this is my command line

```
>> orbit([4,0,0,.5],.0025,50,@(pos,dpos,dt,t) euler(pos,dpos,dt,t))
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> diary off
>> orbit([4,0,0,.5],.25,50,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.6],.5,150,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.8],.5,200,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.4],.25,35,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.2],.25,30,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> orbit([4,0,0,.2],.05,30,@(pos,dpos,dt,t) RK4(pos,dpos,dt,t))
>> diary off
```