Computational Dynamics Homework5 Numerical Initial Value Problems

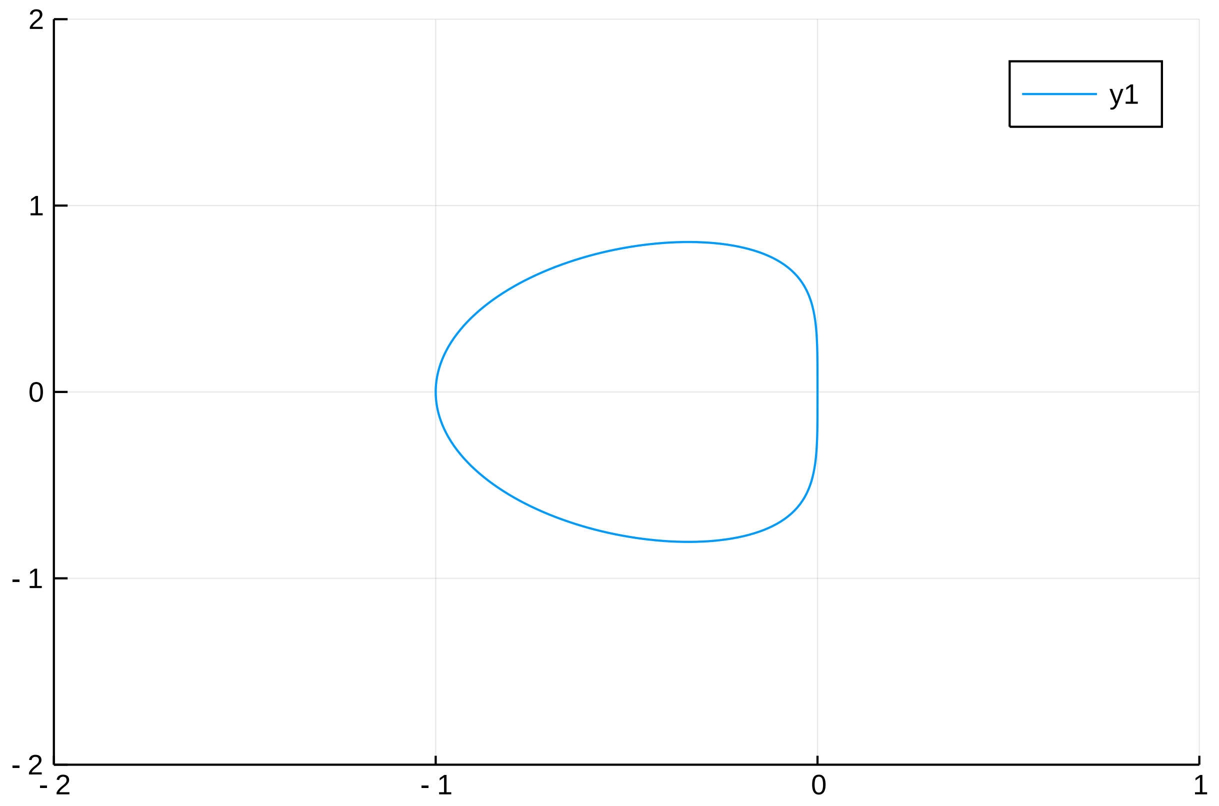
April 5, 2020

Number 2

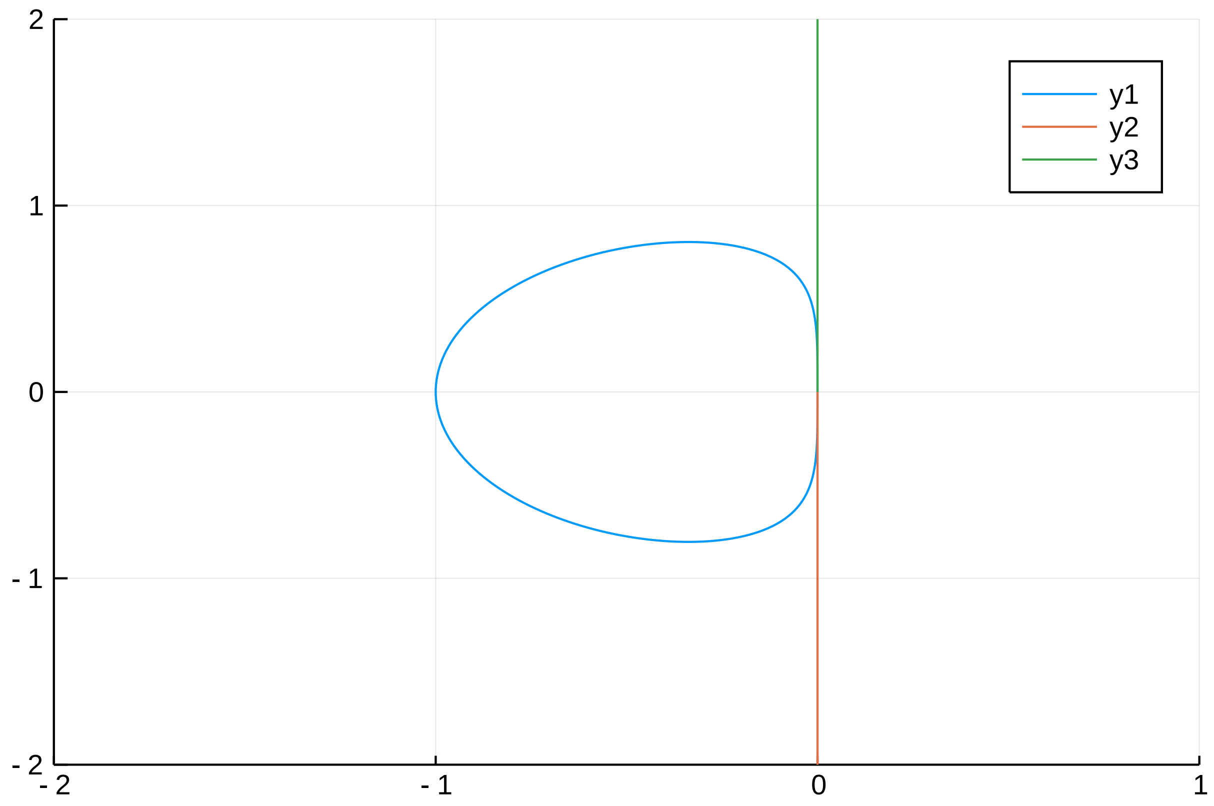
```
# First find the values that h ensure Stability
# Adams_Basthforth
using SymPy
using Plots
import LinearAlgebra
@vars lambda
(sigma,z,zx,zy,theta, h) = symbols("sigma z zx zy theta h",real=true)

eq1 = sigma^2 - sigma*(1+3/2*z)+1/2*z
eq1 = eq1 |> subs(sigma=>cos(theta) + sin(theta)*1im)
eq1 = eq1 |> subs(z=>zx + zy*1im)
eq2 = real(eq1)
eq3 = imag(eq1)



zsolution = solve([eq2, eq3],[zx,zy])
```

```
Dict{Any,Any} with 2 entries:
  zx => 2.0*(cos(theta)^2 - 2.0*cos(theta) + 1.0)/(3.0*cos(theta) - 5.0)
  zy => 2.0*(cos(theta) - 2.0)*sin(theta)/(3.0*cos(theta) - 5.0)
```

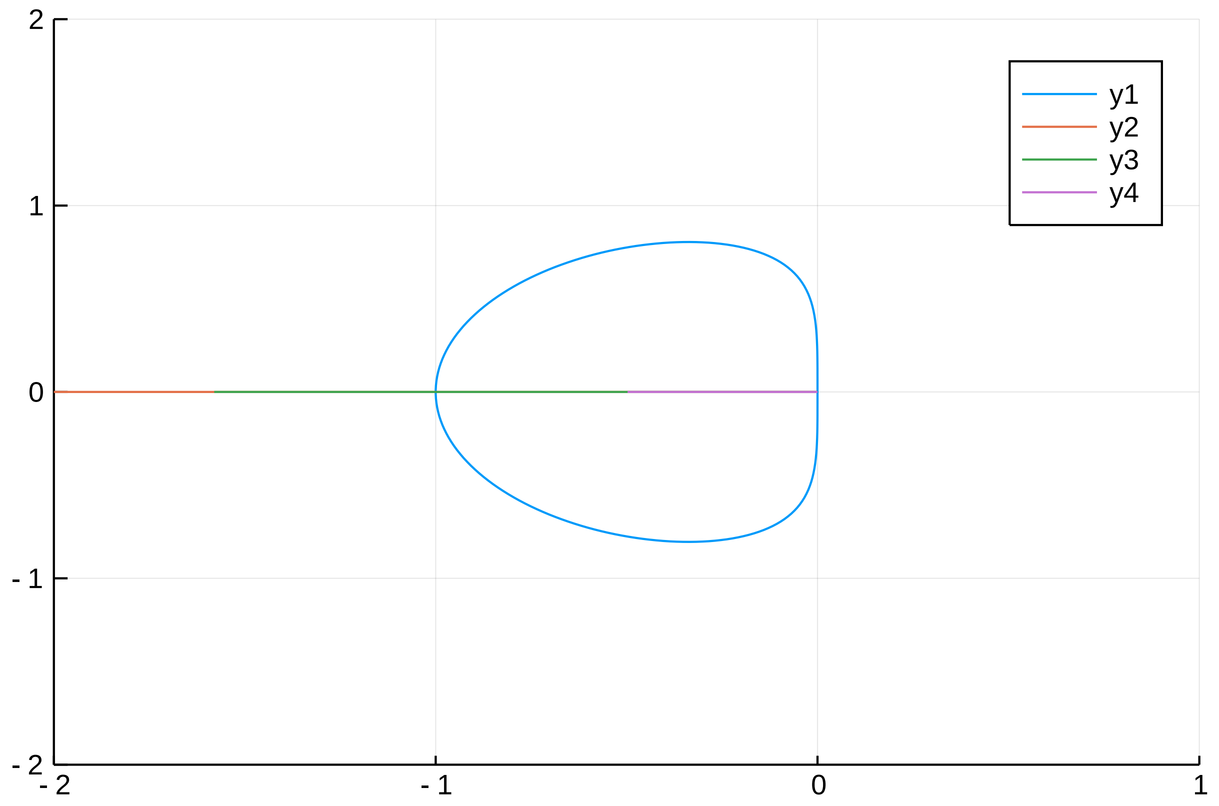
```
z_x = lambdify(zsolution[zx])
z_y = lambdify(zsolution[zy])
theta = collect(0:2*pi/100:2*pi)

plot(z_x, z_y, 0, 2*pi,xlims = (-2,1),ylims = (-2,2))
```



```
A = [0 1;-1 0]
lamda1 = LinearAlgebra.eigvals(A)[1]
lamda2 = LinearAlgebra.eigvals(A)[2]

hvals = collect(0:.01:5)
plot!(real(hvals*lamda1),imag(hvals*lamda1))
plot!(real(hvals*lamda2),imag(hvals*lamda2))
```



```
plot(z_x, z_y, 0, 2*pi,xlims = (-2,1),ylims = (-2,2))
A2 = [0 1 0;0 0 1;-23 -304 -732]
lamda3 = LinearAlgebra.eigvals(A2)[1]
lamda4 = LinearAlgebra.eigvals(A2)[2]
lamda5 = LinearAlgebra.eigvals(A2)[3]
plot!(real(hvals*lamda3),imag(hvals*lamda3))
plot!(real(hvals*lamda4),imag(hvals*lamda4))
plot!(real(hvals*lamda5),imag(hvals*lamda5))
```



Now I'm going to try to find the max h value analytically for Adams_Basthforth

```
a= 1
b = -(1+(3/2*z))
c = 1/2*z
root1 = -b+sqrt(b^2-4*a*c)/2/a |> subs(z=>h*lambda)
root2 = -b-sqrt(b^2-4*a*c)/2/a |> subs(z=>h*lambda)
h_Boundary1 = solve(root1,h)[1]
h_Boundary2 = solve(root2,h)[1]
# The time step h must be smaller than h_Boundary for Adams Basthforth
# CASE 1
hmax_Adams_Basthforth1 = h_Boundary2 |> subs(lambda=>lamda1) |> LinearAlgebra.norm
```

```
0.3463292355306104
```

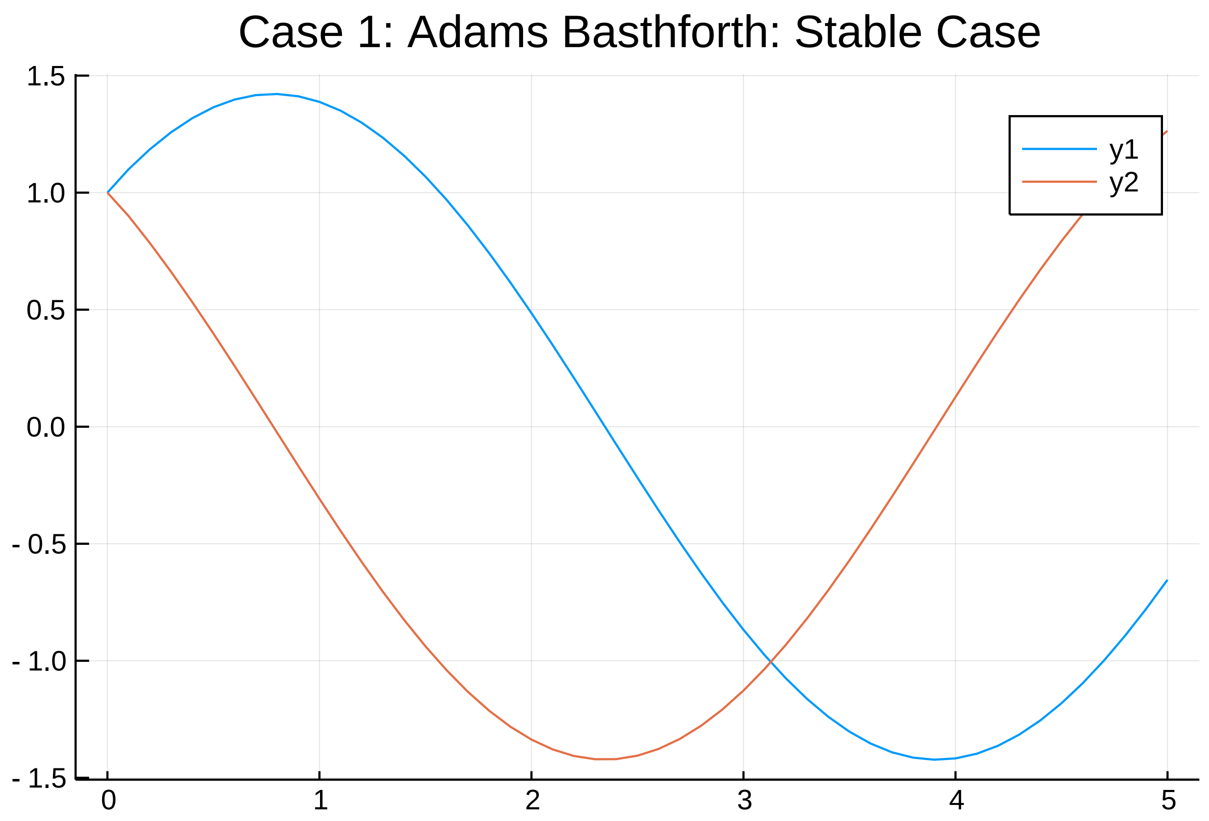
```
# CASE 2
hmax_Adams_Basthforth2 = h_Boundary2 |> subs(lambda=>lamda3) |> LinearAlgebra.norm
```

```
0.0004733960771775613
```

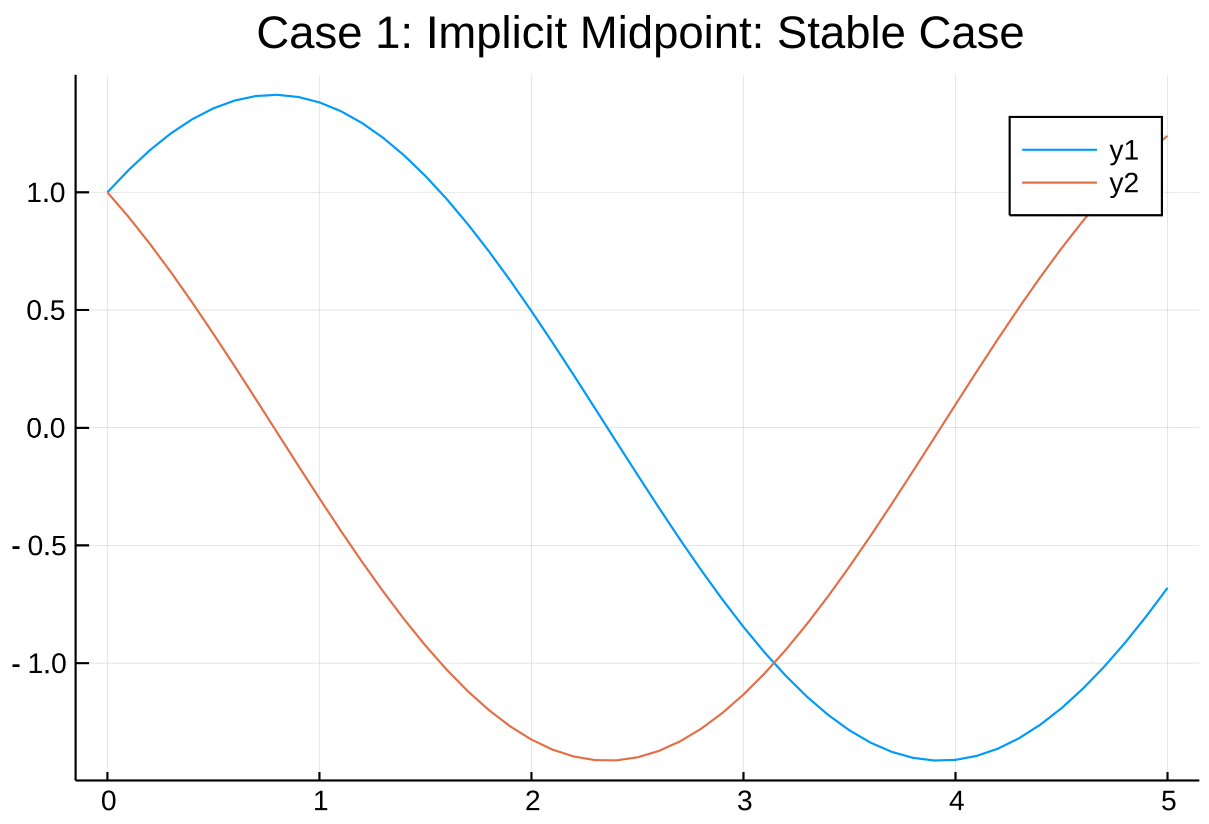
NOW WE NEED TO COMPUTE THESE METHODS NUMERICALLY I'll start with the Adams Basthforth

Start with IVP (x0 given) I'll use euler's to get the second point

```
function Adams_Basthforth(tf,A,h,x0)
    f(t,x) = A*x
    x_nPlus1(t_n,x_n,t_nMinus1,x_nMinus1) = x_n + 3/2*h*f(t_n,x_n) -
1/2*h*f(t_nMinus1,x_nMinus1)
    x = fill(NaN,oftype(1,tf/h+1),length(x0))
    x[1,:] = x0
    x[2,:] = x0 + h*f(0,x[1,:])
    for i = 3:oftype(1,tf/h+1)
        x[i,:] = x_nPlus1(0,x[i-1,:],0,x[i-2,:])
    end
    return x
end


# check the system of 2 equations with built in stuff
##
import DifferentialEquations

function ode2(dz,z,p,t)
        dz[1] = z[2]
        dz[2] = -z[1]
end

tspan = (0.0,5.)
z0 = [1.;1.]
prob = DifferentialEquations.ODEProblem(ode2,z0,tspan)
sol1 = DifferentialEquations.solve(prob)




# now do implicit midpoint method
##

function ImplicitMidPoint(A, tf, h, x0; tol = 1e-4, iterMax = 1000 )
        f(t,x) = A*(x)
        time = 0:h:tf
        n = length(time)
        p = length(x0)

        x = fill(NaN,n,p)

        # Initial Condition
        x[1,:] .= x0

        # 1 Euler Step (first guess)
        y = x[1,:] + h*f(time[1],x[1,:])

        # Implicit Midpoint for mulitiple steps
    for i = 1:n-1
        flag = 0
        iter = 0
        while flag == 0
            iter += 1

            ynew = x[i,:] + h*f(time[i]+h/2,1/2*(x[i,:]+y))


            residual = LinearAlgebra.norm( y - ynew)
            y = ynew


            if residual <= tol
                flag = 1
            elseif iter >= iterMax
                flag = -1
                error("Error: failed to converge")
            end
            x[i+1,:] = y
        end
    end
    return x,time
end




# check the system of 3 equations with built in stuff
##


function ode3(dz,z,p,t)
        A2 = [0 1 0 ;0 0 1;-23 -304 -372]
        dz[1] = z[2]
        dz[2] = z[3]
        dz[3] = -23*z[1] -304z[2] -372*z[3]

end

tspan = (1.,5.)
z0 = [-10.;10.;-7.5]
prob = DifferentialEquations.ODEProblem(ode3,z0,tspan)
sol3 = DifferentialEquations.solve(prob)


##
# Now we'll test them

A = [0 1;-1 0]
x0 = [1.;1.]
h = .1
t0 = 0
tf = 5

## Adams_Basthforth Test
xvals = Adams_Basthforth(tf,A,h,x0)

tvals = collect(0:h:tf)
plot(tvals,xvals[:,1])
    plot!(tvals,xvals[:,2])
    title!("Case 1: Adams Basthforth: Stable Case")
```

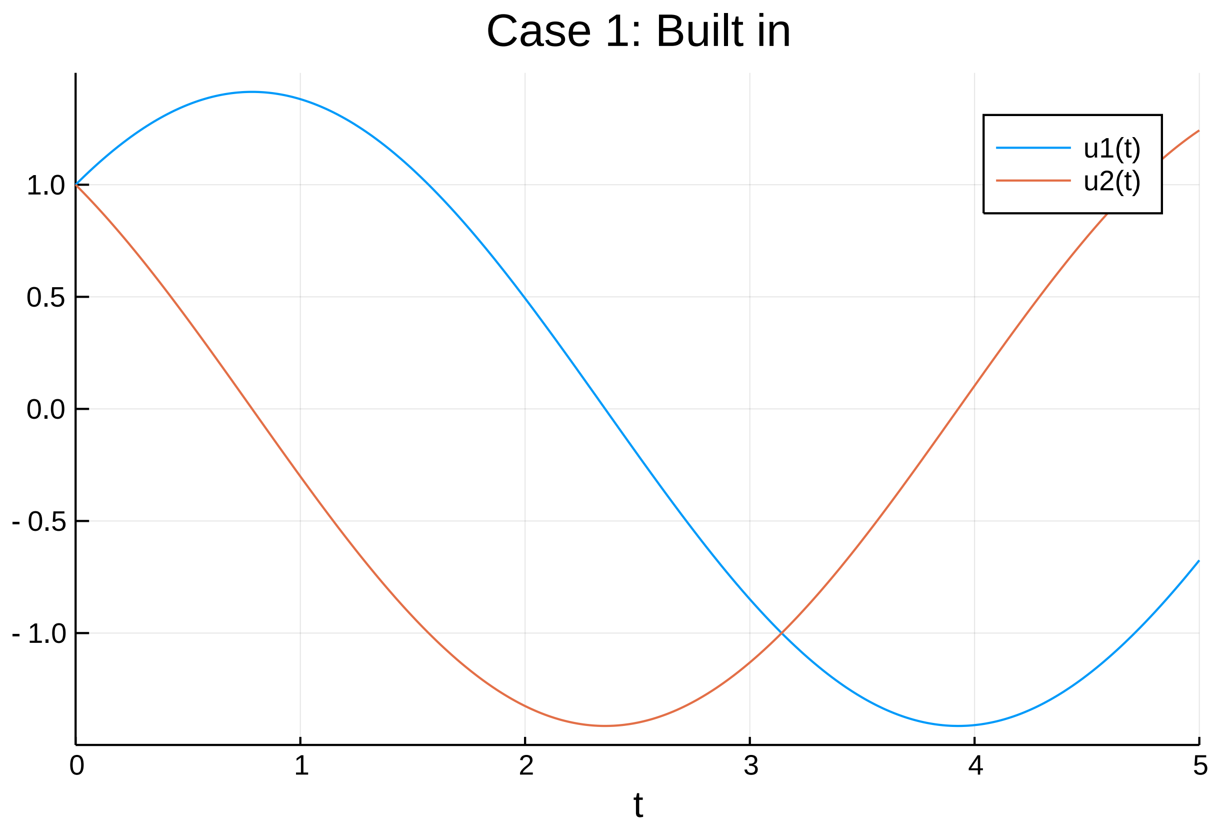


```julia
## ImplicitMidPoint
x,Time = ImplicitMidPoint(A, tf, h, x0)
plot( Time, x)
    title!("Case 1: Implicit Midpoint: Stable Case")
```



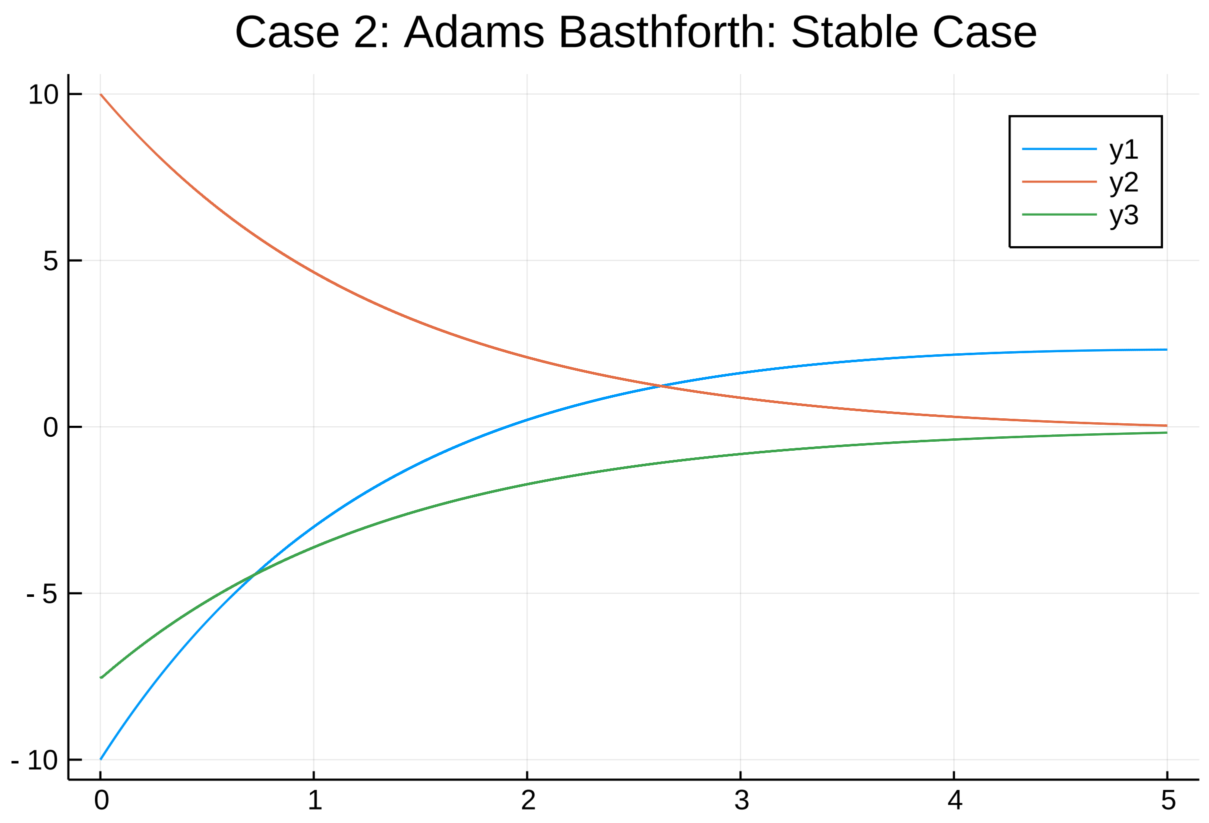
```julia
#DifferentialEquations.jl algorithm which which picks the best solver to use.
```

```
plot(sol1, vars = (0,1))
    plot!(sol1, vars = (0,2))
    title!("Case 1: Built in")
```

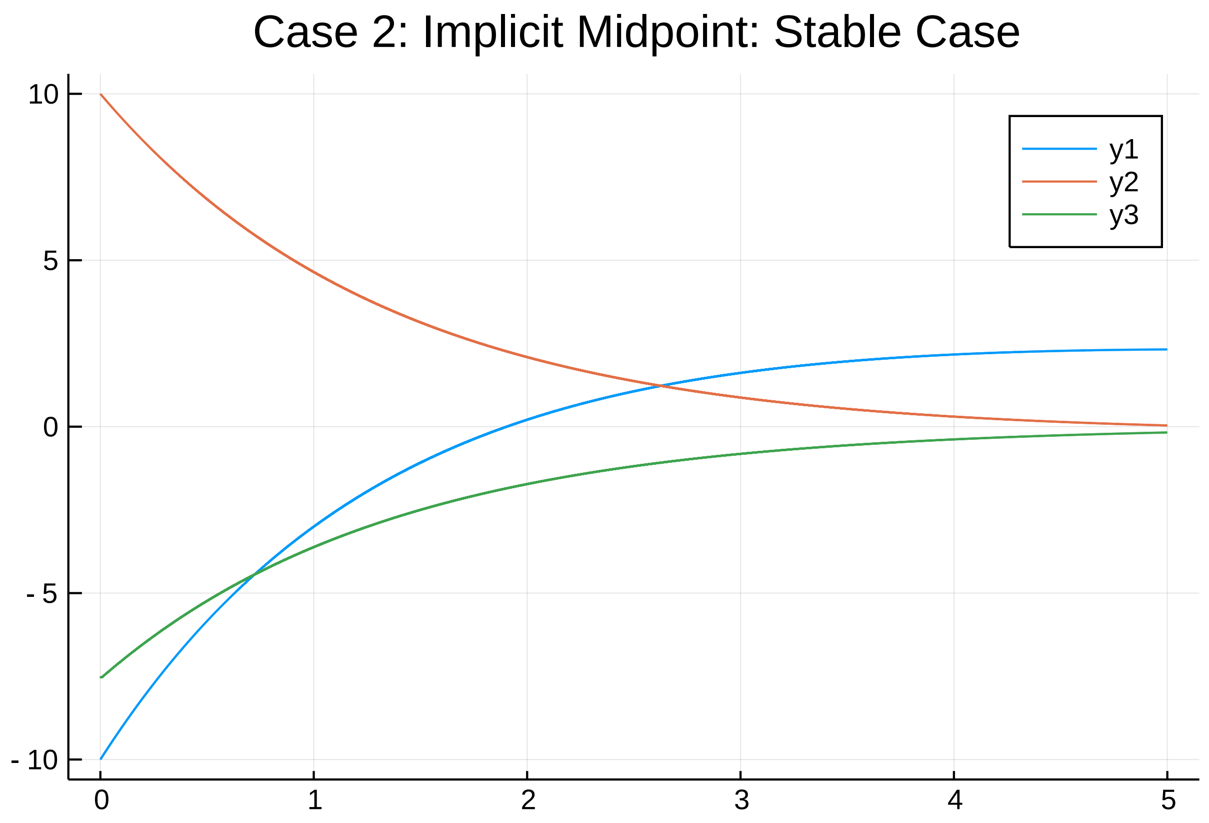


Now we'll test the system of 3 equations

```
A2 = [0 1 0 ;0 0 1;-23 -304 -372]
x02 = [-10.;10.;-7.5]
h2 = .0001

##
# Adams_Basthforth
xvals = Adams_Basthforth(tf,A2,h2,x02)

tvals = collect(0:h2:tf)
plot(tvals,xvals[:,1])
    plot!(tvals,xvals[:,2])
    plot!(tvals,xvals[:,3])
    title!("Case 2: Adams Basthforth: Stable Case")
```

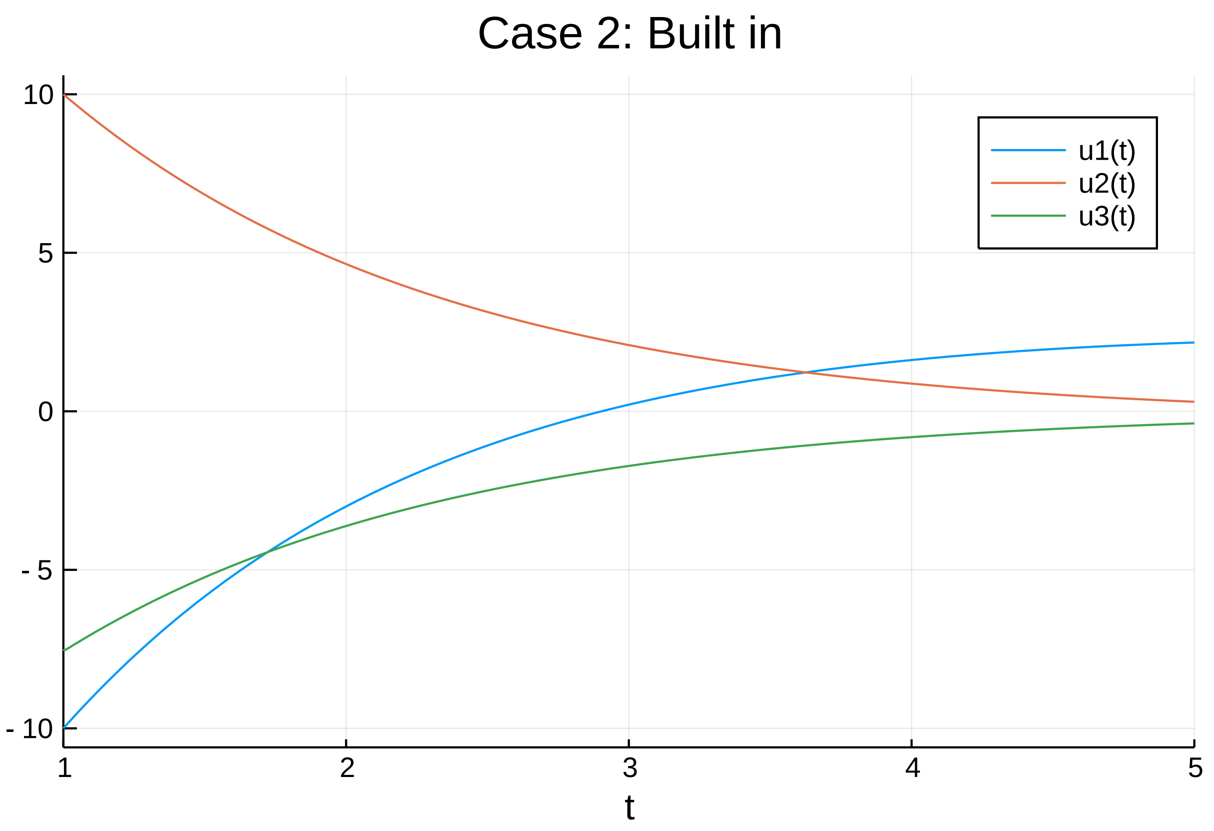


```
# ImplicitMidPoint method
x,Time = ImplicitMidPoint(A2, tf, h2, x02)
plot( Time, x)
    title!("Case 2: Implicit Midpoint: Stable Case")
```



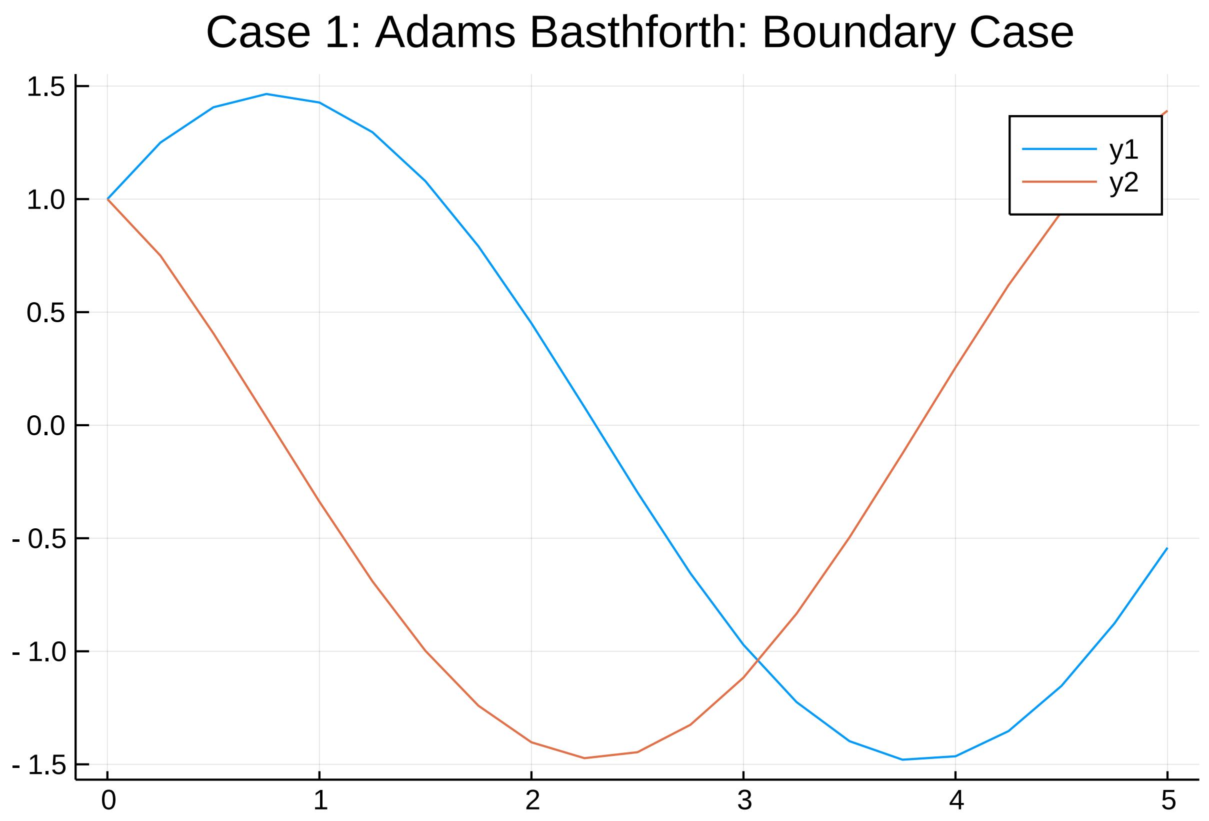
```
# Built in stuff
```

```
plot(sol3, vars = (0,1))
    plot!(sol3, vars = (0,2))
        plot!(sol3, vars = (0,3))
        title!("Case 2: Built in")
```

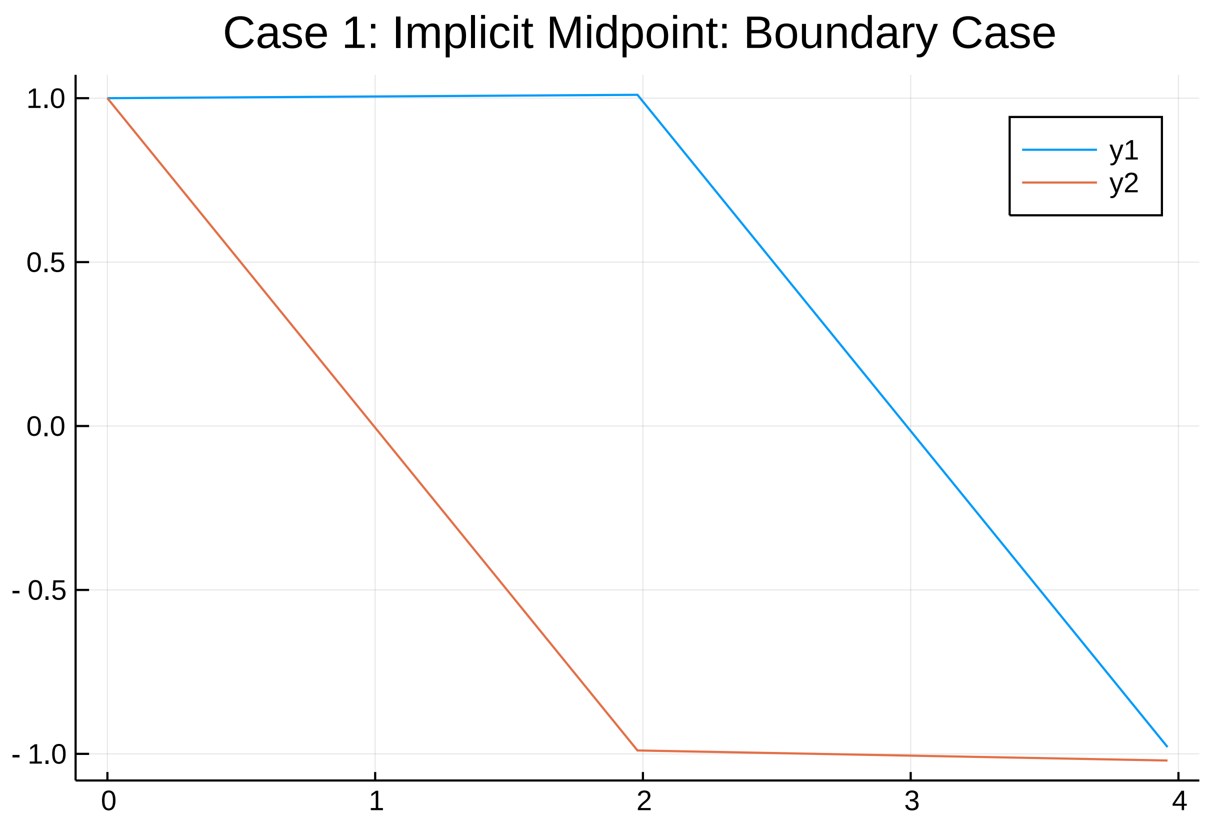


Ok! Now lets test the boundary case! I couldnt figure out how to solve it analytically so I did it with guess and check

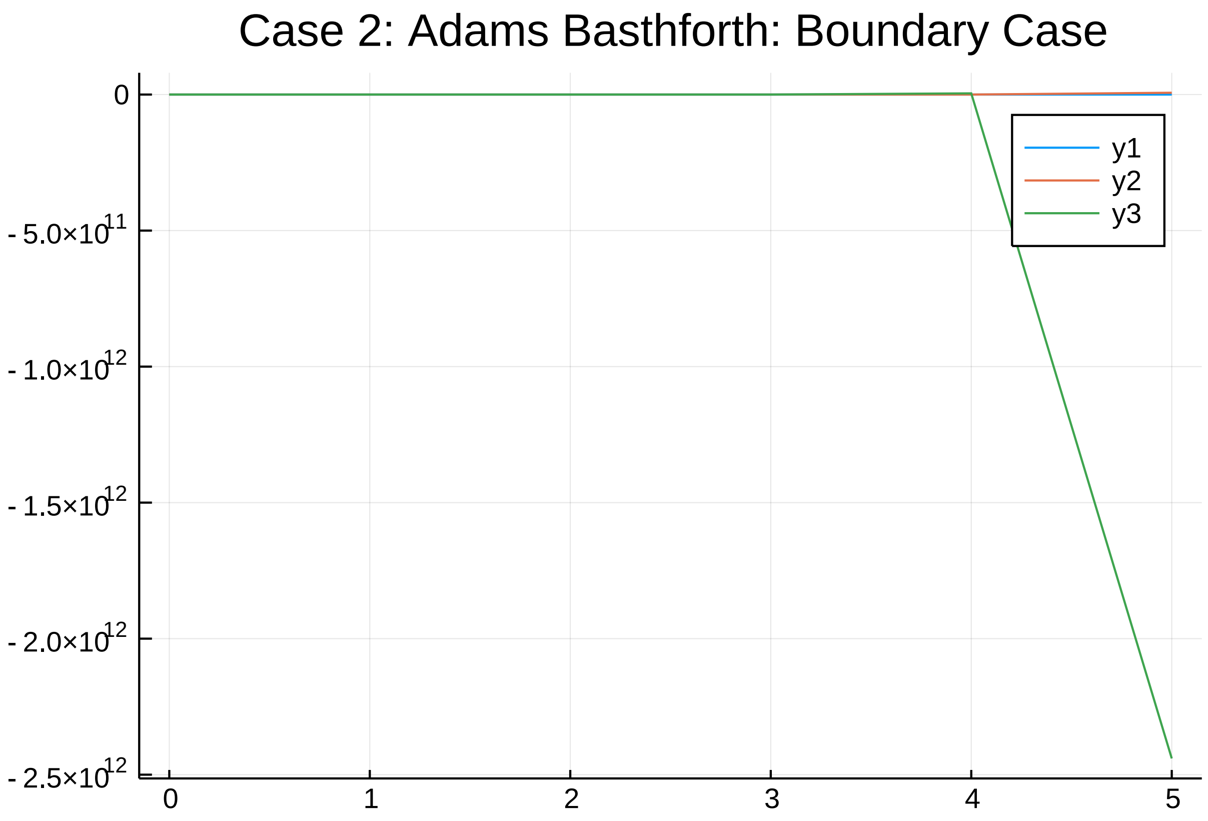
```
# by testing hmax for adams basthforth is .25 for case 1
## Adams_Basthforth Test
xvals = Adams_Basthforth(tf,A,.25,x0)

tvals = collect(0:.25:tf)
plot(tvals,xvals[:,1])
    plot!(tvals,xvals[:,2])
    title!("Case 1: Adams Basthforth: Boundary Case")
```



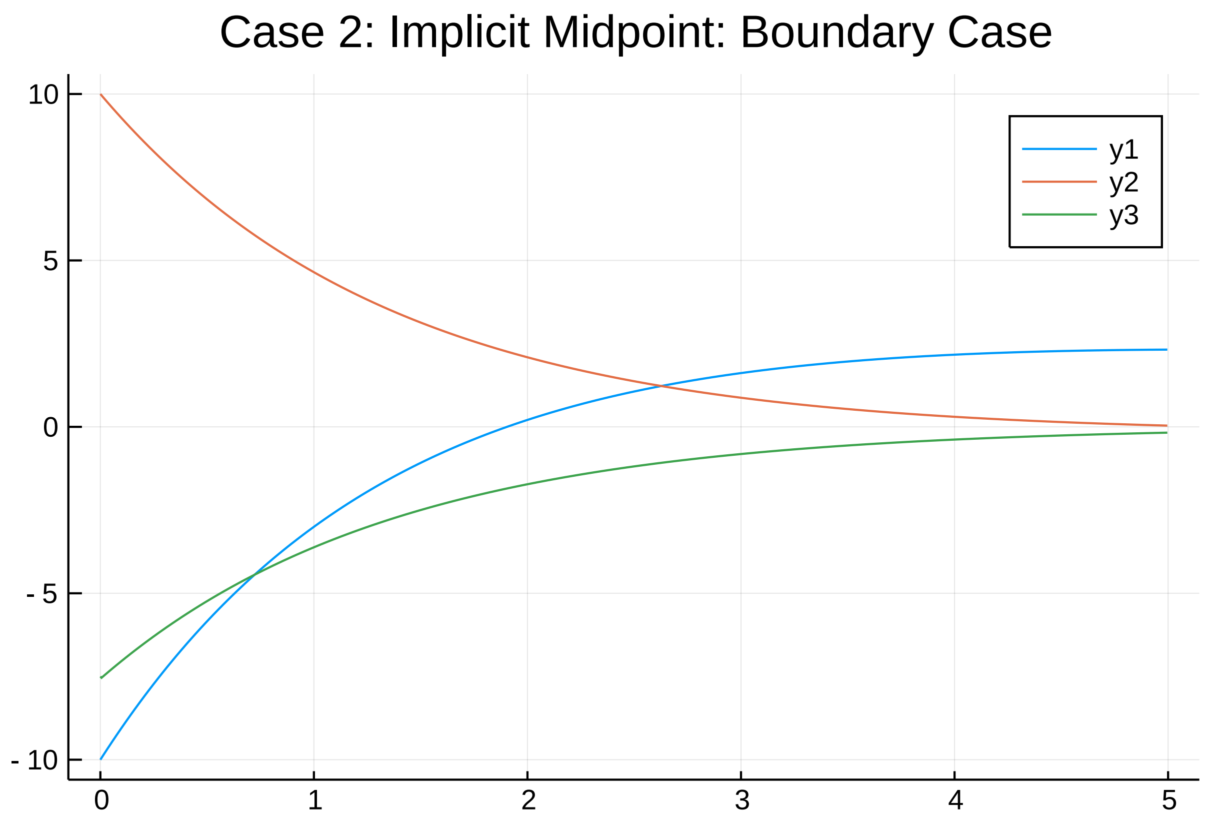
```
# by testing hmax for Implicit MidPoint is 1.9796 for case 1
x,Time = ImplicitMidPoint(A, tf, 1.9796, x0)
plot( Time, x)
    title!("Case 1: Implicit Midpoint: Boundary Case")
```



```
# by testing hmax for adams basthforth is 1 for case 2
# Adams_Basthforth
xvals = Adams_Basthforth(tf,A2,1.,x02)

tvals = collect(0:1:tf)
plot(tvals,xvals[:,1])
    plot!(tvals,xvals[:,2])
    plot!(tvals,xvals[:,3])
    title!("Case 2: Adams Basthforth: Boundary Case")
```

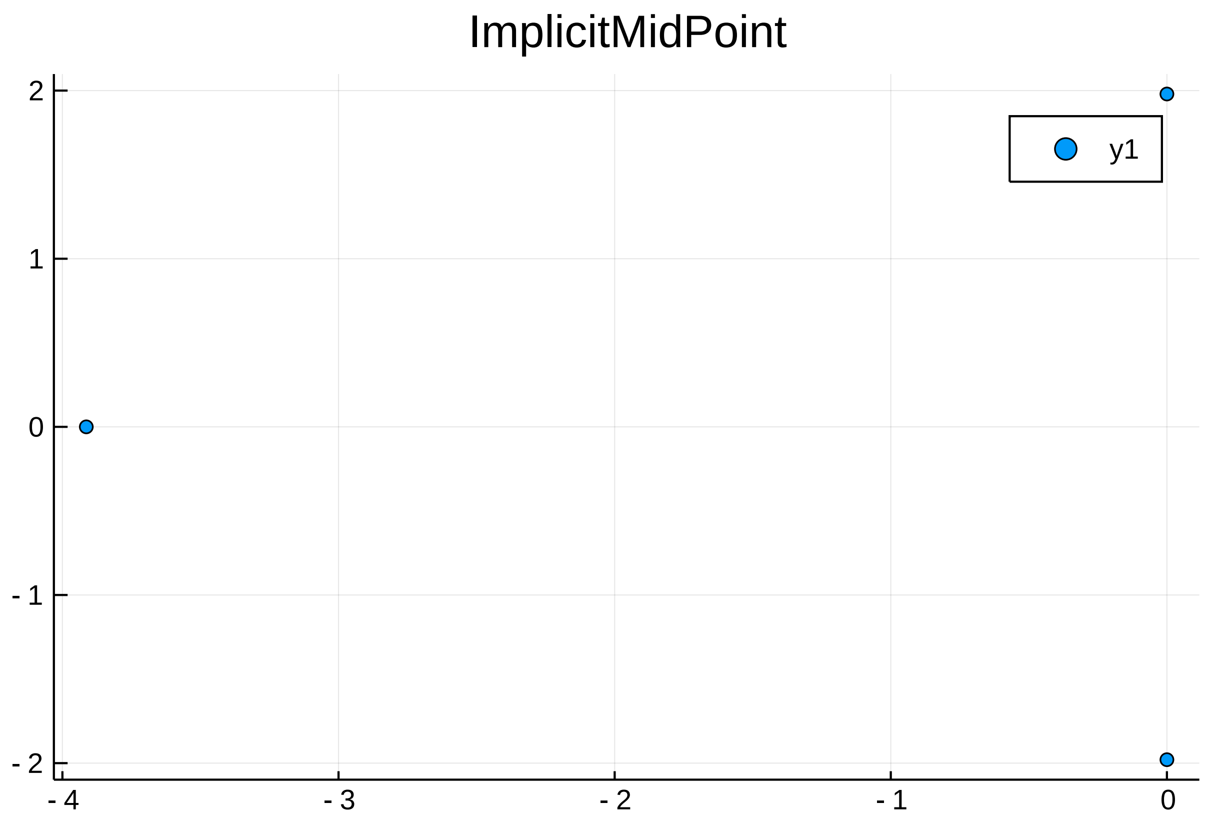


```
# by testing hmax for Implicit MidPoint is 0.0053494 for case 2
# ImplicitMidPoint method
x,Time = ImplicitMidPoint(A2, tf, .0053494, x02)
plot( Time, x)
    title!("Case 2: Implicit Midpoint: Boundary Case")
```

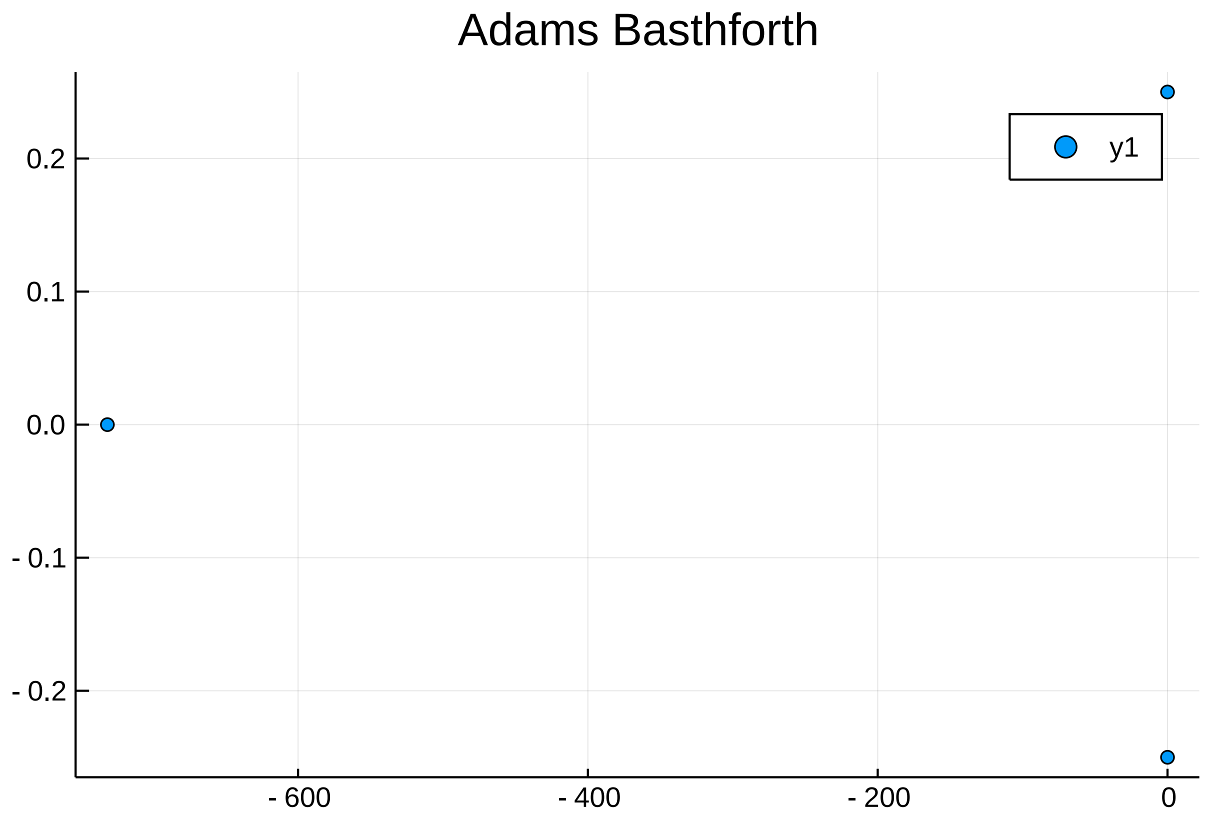


Now I am going to plot h*lambda for the largest values of h possilbe to try to get at least a few points of the stability curve.... I know Its not great but Its what I've got right now...

```
scatter(real([lamda3*.0053494,lamda1*1.9796,lamda2*1.9796]),imag([lamda3*.0053494,lamda1*1.9796,lamda2
    title!("ImplicitMidPoint")
```



```
scatter(real([lamda3*1,lamda1*.25,lamda2*.25]),imag([lamda3*1,lamda1*.25,lamda2*.25]))
        title!("Adams Basthforth")
```


Adams Basthforth
y1