Assignment 4

Kevin Corcoran

May 1, 2021

1 Problem 1

1.1 Part 1

Show method has second order accuracy

First order consistency condition

$$\sum_{i=1}^{p=1} b_i = 1$$
$$\implies 1 = 1$$

So method has first order accuracy

Second order consistency condition

$$\sum_{i=1}^{p=2} b_i c_i = \frac{1}{2}$$
$$\implies 1(\frac{1}{2}) = \frac{1}{2}$$

Therefore, this method is second order.

1.2 Part 2

From homework three, I derived the stability function (z) for any RK method,

$$\Phi(z) = 1 + z\vec{b}^T(I - zA)^{-1}\vec{e}.$$

So in this case,

$$
\begin{aligned}
\Phi(z) &= 1 + z(1 - z\frac{1}{2})^{-1} \\
&= 1 + \frac{z}{1 - z\frac{1}{2}}
\end{aligned}
$$

$$
\implies \boxed{\Phi(z) = \frac{1 + \frac{z}{2}}{1 - \frac{z}{2}}}
$$

1.3 Part 3

For A-stability, we want $|\Phi(z)| < 1$. So,

$$
\begin{aligned}
\left|\frac{1 + \frac{z}{2}}{1 - \frac{z}{2}}\right| &\leq 1 \\
\left|1 + \frac{z}{2}\right|^2 &\leq \left|1 - \frac{z}{2}\right|^2
\end{aligned}
$$

Let $z = a + ib$ then,

$$
\left(a + \frac{a}{2}\right)^2 + \left(\frac{b}{2}\right)^2 \leq \left(a - \frac{a}{2}\right)^2 + \left(\frac{b}{2}\right)^2
$$

Which is satisfied when $a \leq 0$. This implies $Re(z) \leq 0$, and therefore the method is A-stable.

For L-stability, we want $\lim_{z\to\infty} \Phi(z) = 0$.

$$
\begin{aligned}
\lim_{z\to\infty} \Phi(z) &= \lim_{z\to\infty} \frac{1 + \frac{z}{2}}{1 - \frac{z}{2}} \\
&= \lim_{z\to\infty} \frac{\frac{2}{z} + 1}{\frac{2}{z} - 1} \\
&= -1
\end{aligned}
$$

Therefore, this method is not L-stable

Including required packages

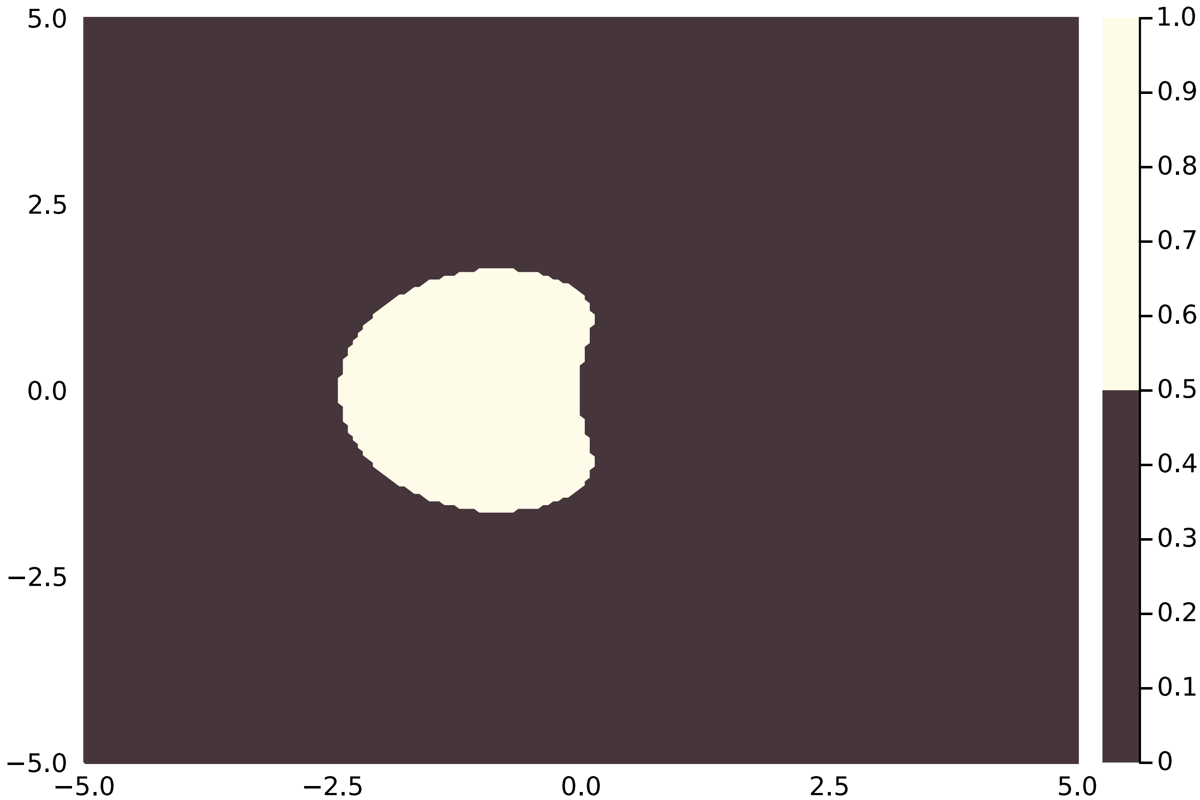
```julia
using Plots
theme(:mute)
using SparseArrays
using LinearAlgebra

using Pkg
```

```
Pkg.activate("DiffyQ")
include("code/DiffyQ.jl")
using .DiffyQ: rk4, Newtons_n

using Pkg
Pkg.activate("StabilityRegion")
include("code/StabilityRegion.jl")
using .StabilityRegion: RAS, RASlmm, RASrk
```

2 Problem 2: Plot Stability Region

2.1 3-step Adams-Moulton

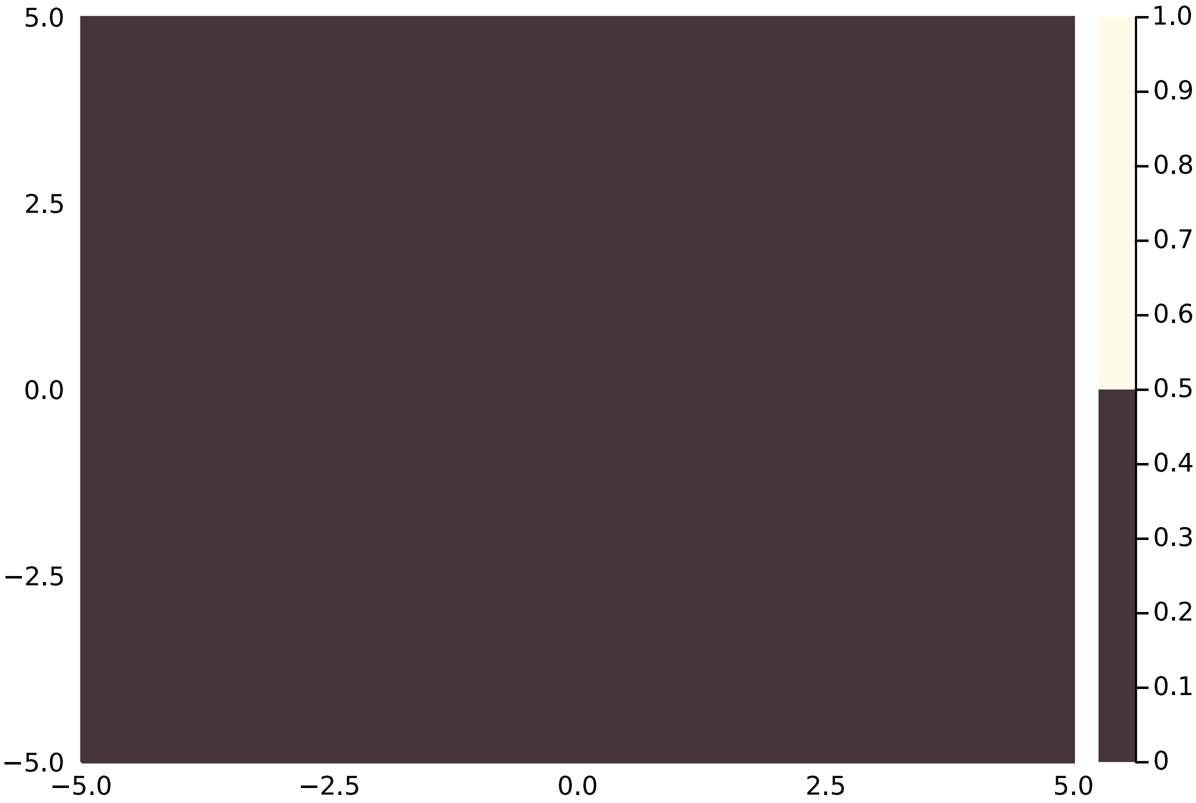
```
# "Light" is the stability region

# 3-step Adams-Moulton
α = [0.0, 0.0, -1.0, 1.0]; β = [1/24, -5/14, 19/24, 9/24]
xs,Z = RASlmm(α, β)

contourf(xs, xs, Z, levels = 1)
```



2.2 2-step 4th order LMM

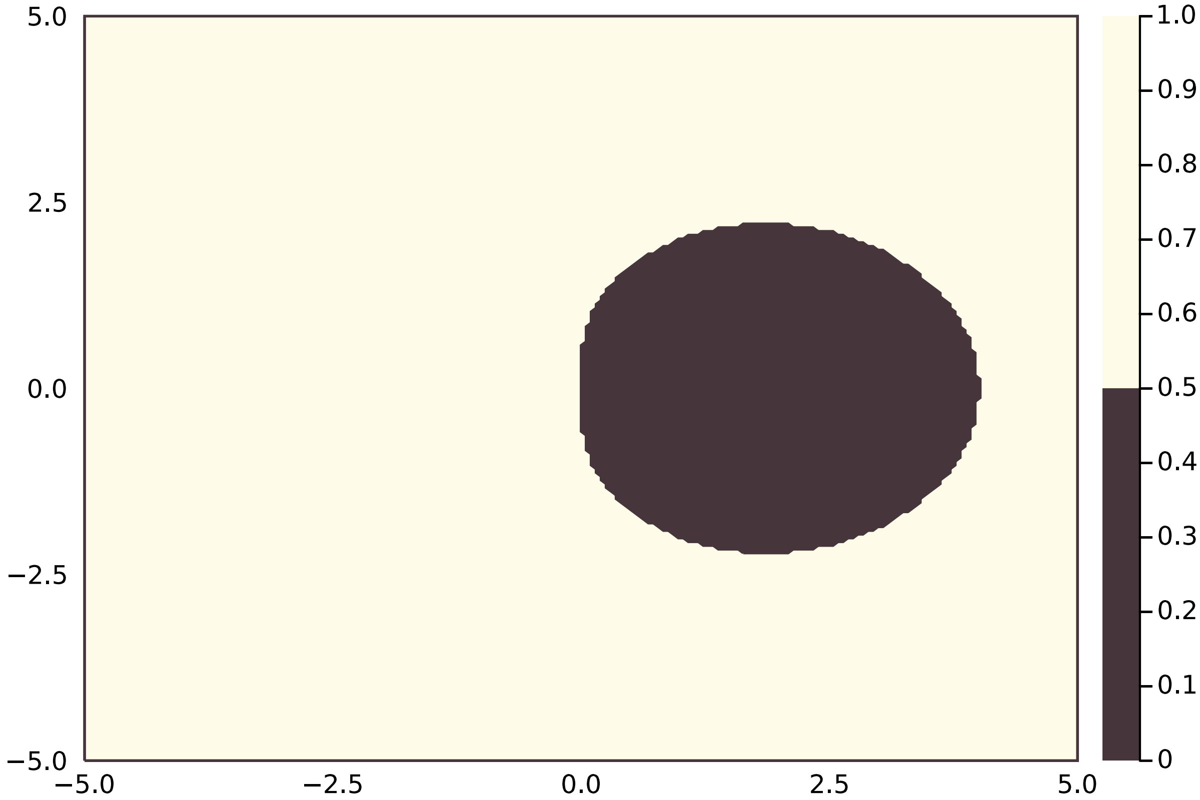
```
# 2-step 4th order LMM
α = [-1.0, 0.0, 1.0]; β = [1/3, 4/3, 1/3];
xs,Z = RASlmm(α, β)
```

```
contourf(xs, xs, Z, levels = 1)
```



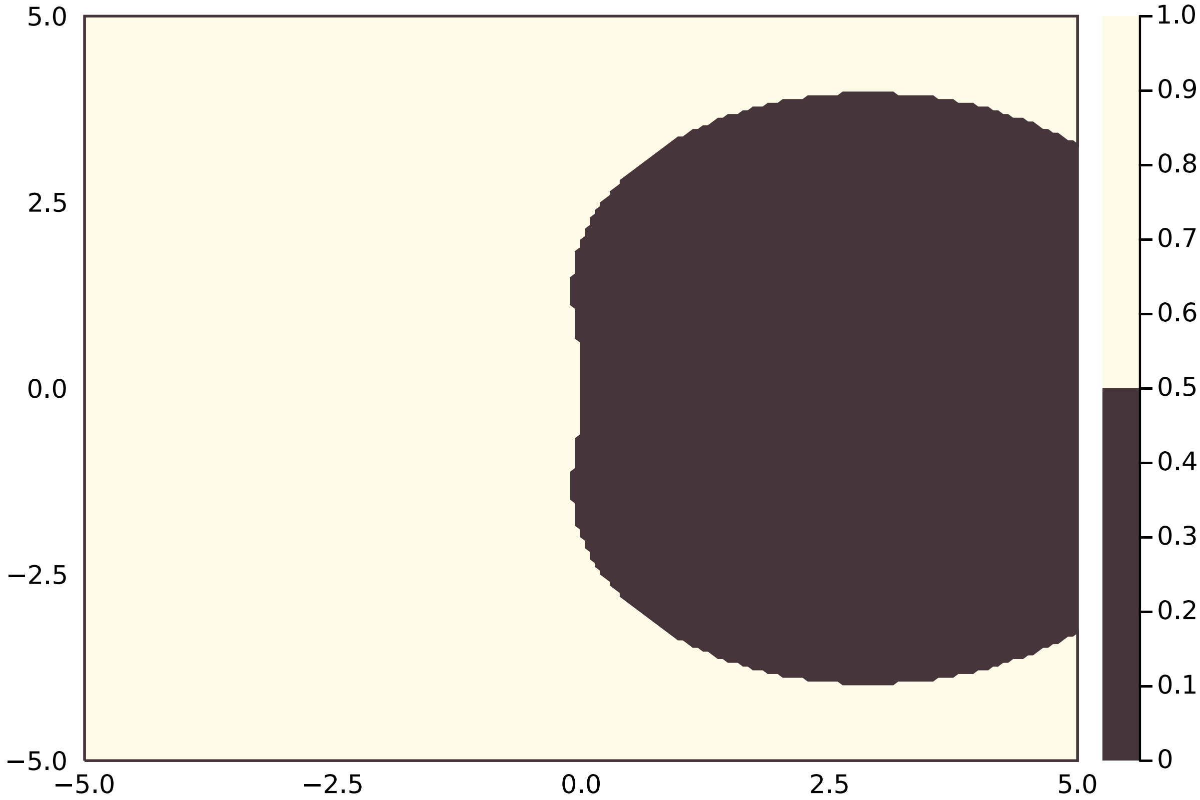
2.3 2-step BDF (BDF2)

```
# 2-step BDF (BDF2)
α = [1/2, -2.0, 3/2]; β = [0.0, 0.0, 1.0];
xs,Z = RASlmm(α, β)

contourf(xs, xs, Z, levels = 1)
```



2.4 3-step BDF (BDF3)

```
# 3-step BDF (BDF3)
α = [-1/3, 3/2, -3.0, 11/6]; β = [0.0, 0.0, 0.0, 1.0];
xs,Z = RASlmm(α, β)

contourf(xs, xs, Z, levels = 1)
```



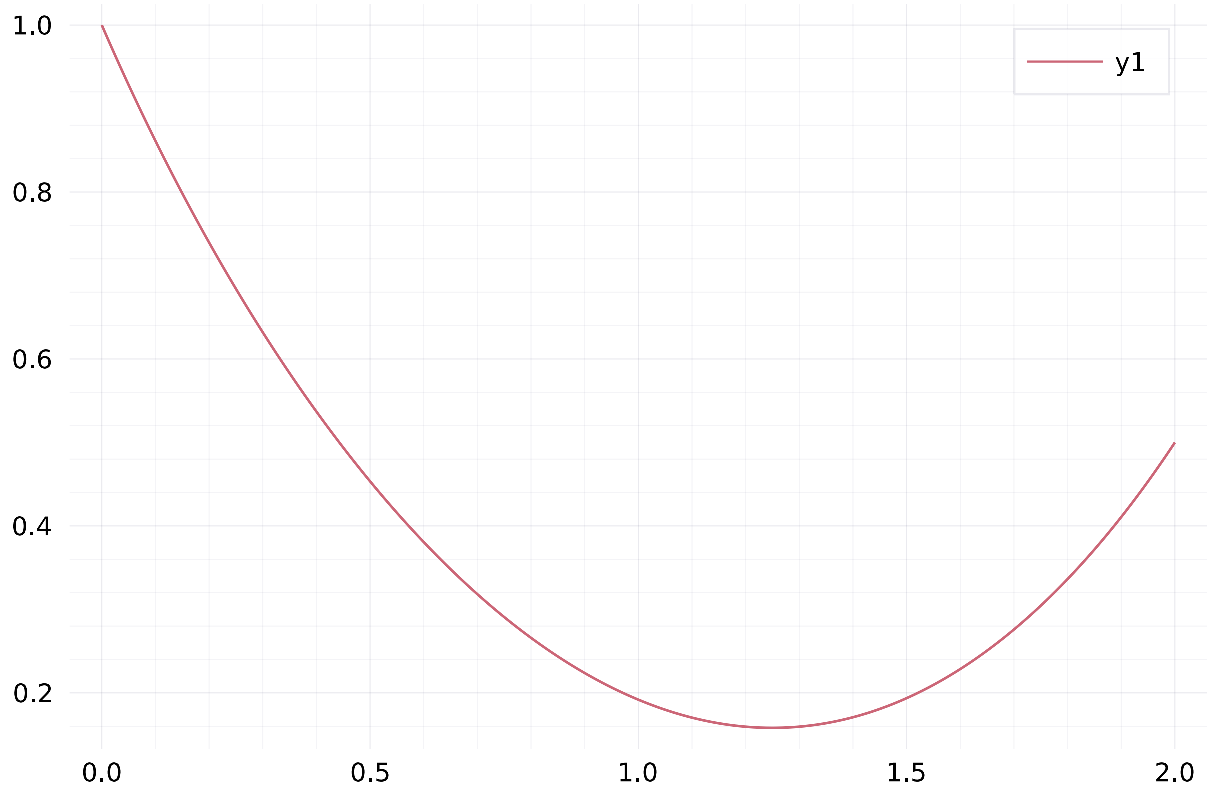
3 Problem 3: Shooting Method

```
# Differential equation
f(u, x, μ) = [u[2]; sin(x) + (1+0.5*u[2]^2)*u[1]]

# BVP conditions
a = 0.0; b = 2.0; α = 1; β = 0.5;

# Variables
h = 0.002; T = b-a; N = Int(T/h);

# Find ν for IVP
# ν = (β - α)/(b-a) # guess slope avg rate of change over interval
ν = -1.0

function G(ν)
    u0 = [α; ν]
    u = rk4(f, N, T, u0)
    return u[1,end] - β
end

ν = Newtons_n(G, ν, 0.0)
```

```
-1.4850899639208666
```

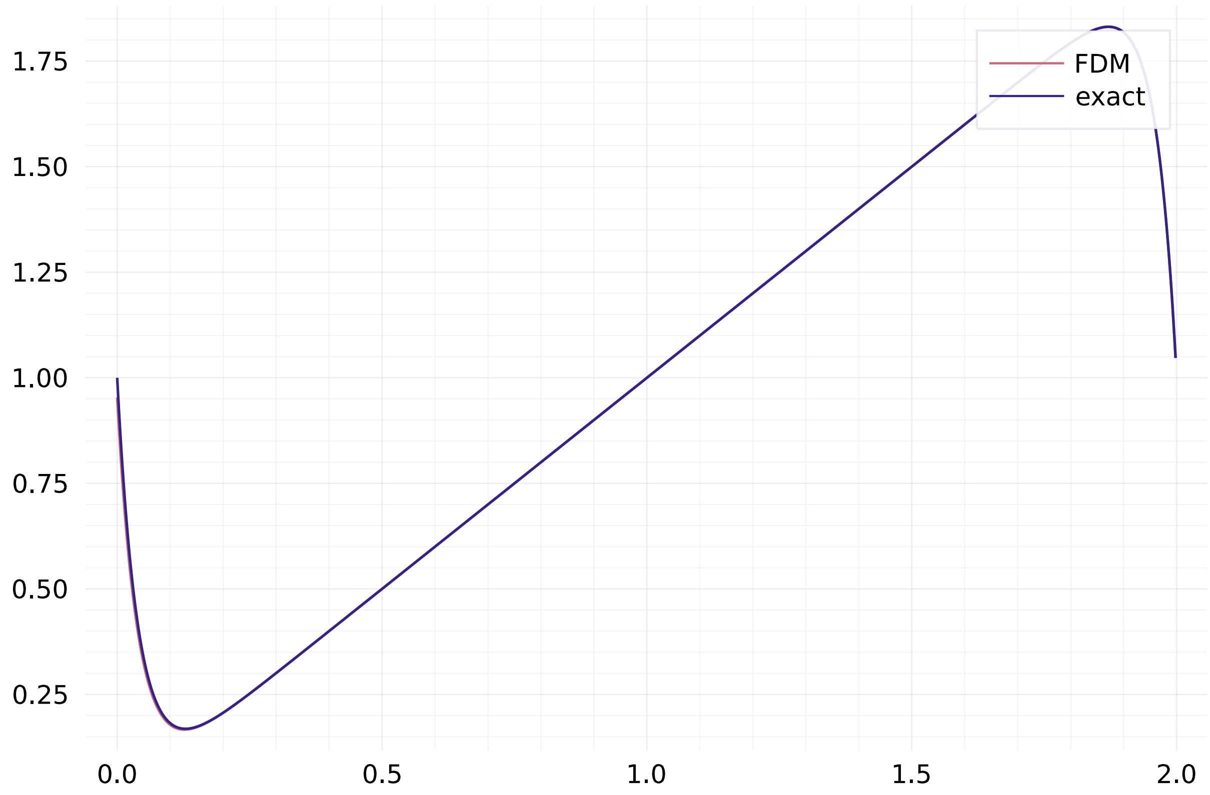
```
# Solve ode using ν found above
u0 = [α; ν]
u = rk4(f, N, T, u0)

t = collect(0:N)*h
```

```
plot(t, u[1,:])
```



4 Problem 4: FDM

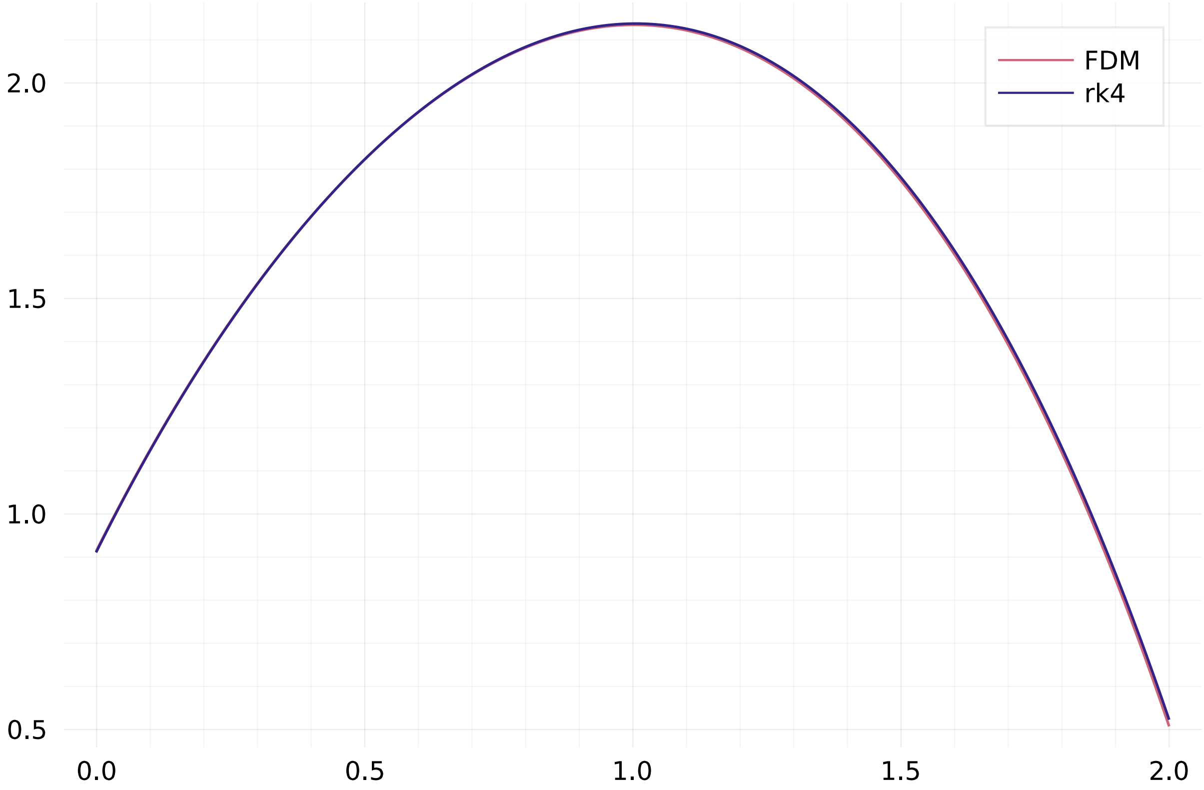
```
T = 2.0; N = 1000; h = T / N
x = collect(0:N-1)*h

# Laplacian
L = 1/h^2 * spdiagm(-1=>ones(N-1),0=>-2.0*ones(N),1=>ones(N-1))

# Differential equation
α = 1; β = 1;
g = -625*x;

# Boundary conditions
g[1] = g[1] - (1/h^2)*α
g[N] = g[N] - (1/h^2)*β

u = (L - 625*I)\g

u_exact(x) = x + (1+exp(-50))/(1-exp(-100))*(exp.(-25*x) - exp.(25*(x.-2)))
plot(x, u, label="FDM")
plot!(x,u_exact(x), label ="exact")
```



5 Problem 5

```
a = 0.0; b = 2.0
T = b-a; N = 1000;
h = T / (N-0.5);
xs = zeros(N+1)
for i = 1:N+1
    xs[i] = a + ((i-1)-0.5)*h
end

# Discretization
q = -(1 .+ exp.(-sin.(xs)))
L =  1/h^2 * spdiagm(-1=>ones(N),0=>-2.0*ones(N+1),1=>ones(N))
A = L + spdiagm(0=>q)
A[1,1] = -1/h^2

# Boundary conditions and forcing term
α = 2.5; β = 0.5;
g = -5 .- sin.(xs).^2;

# Incorporate boundary conditions in FDM
g[1] = g[1] + (1/h)*α
g[end] = g[end] - (1/h^2)*β

u = A\g

plot(xs, u, label="FDM")

# verify solution
f(u,x,μ) = [u[2]; -5 - sin(x)^2 + (1 + exp(-sin(x)))*u[1]]
```

```
u0 = [u[1] - α*h/2; α]
u = rk4(f, N, T, u0)
plot!(xs, u[1,1:N+1], label="rk4")
```



6 Problem 6

```
a = 0.0; b = 2.0
T = b-a; N = 1000; h = T / N
xs = collect(0:N-1)*h

# Discretization
q = -(1 .+ exp.(-sin.(xs)))
# L =  1/h^2 * spdiagm(-1=>ones(N),0=>-2.0*ones(N+1),1=>ones(N))
L =  1/h^2 * spdiagm(-1=>ones(N-1),0=>-2.0*ones(N),1=>ones(N-1))
A = L + spdiagm(0=>q)

# Boundary conditions and forcing term
α = 2.5; β = 0.5;
g = -5 .- sin.(xs).^2;
A[1,1] += (1/h^2)*(2-h)/(2+h)

# Incorporate boundary conditions in FDM
g[1] = g[1] - (1/h^2)*2*α*h/(2+h)
g[end] = g[end] - (1/h^2)*β

# Solve system
u = A\g
plot(xs, u, label="FDM")

# verify solution
```

```
f(u,x,μ) = [u[2]; -5 - sin(x)^2 + (1 + exp(-sin(x)))*u[1]]
u0 = [(2*α*h + (2-h)*u[1])/(2+h); 2*(u[1] - α)/(2+h)]
u = rk4(f, N, T, u0)
plot!(xs[1:N], u[1,1:N], label="rk4")
```