Computational Dynamics Homework5 Numerical Initial Value Problems

April 5, 2020

1 Stablity

1.1 The Trapezoidal Rule

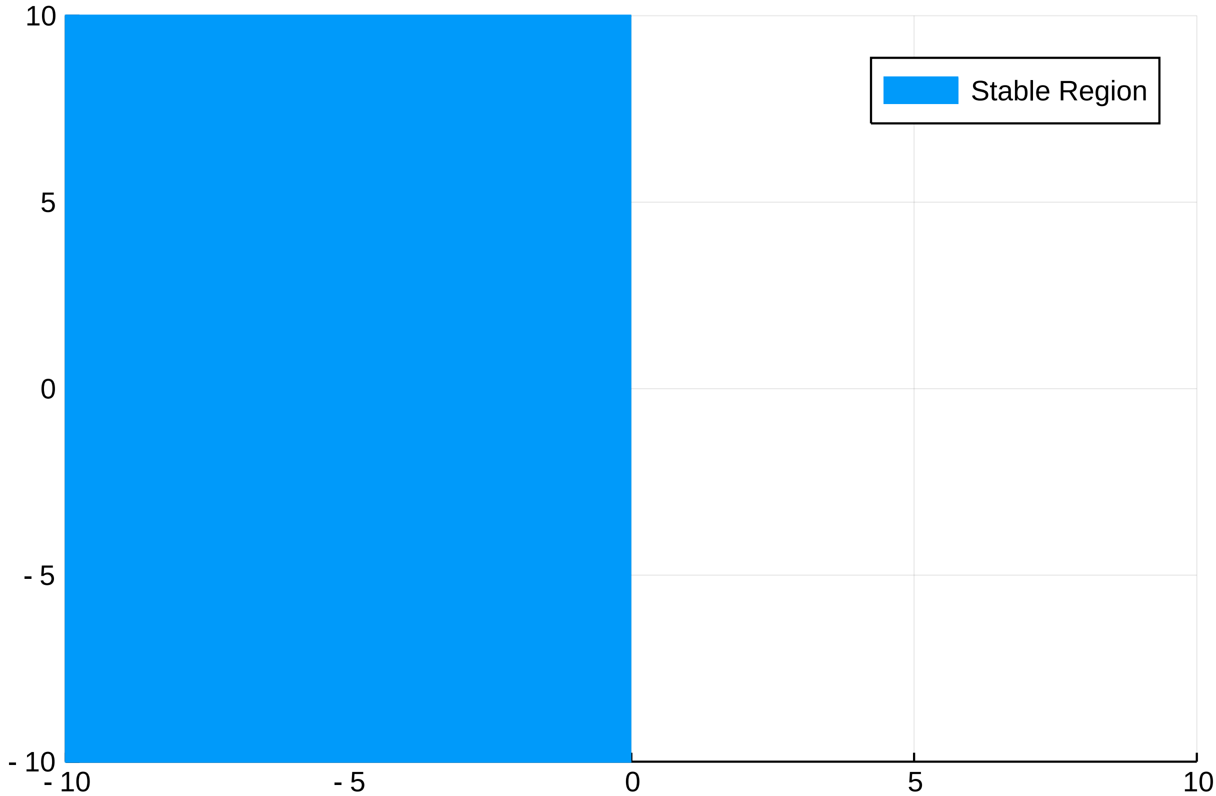
```
using SymPy
using Plots
import LinearAlgebra
# f = xdot

@syms x t lambda h
x = SymFunction("x")(t)
f(x,t) = lambda*x;
```

$$x_(n+1) = x_(n) + h/2 * (f(x_(n), t_(n)) + f(x_(n+1), t_(n+1))$$
$$x_(n+1) = x_(n) + h/2 * (lambda * x_(n) + lambda * x_(n+1))$$
$$x_(n+1)(1 - lambda * h/2) = x_(n)(1 + lambda * h/2)$$
$$x_(n+1) = (1 + lambda * h/2)/(1 - lambda * h/2)x_(n)$$
$$|\frac{1 + lambda * h/2}{1 - lambda * h/2}| < 1$$
$$so....$$
$$1 + lambda * h/2 < 1 - lambda * h/2$$
$$and$$
$$1 + lambda * h/2 > -1 + lambda * h/2$$

$$0 < -lambda * h$$
$$and$$
$$2 > 0$$

```
xmax = 10
plot([0,-xmax,-xmax,0],[-xmax,-xmax,xmax,xmax],xlims = (-xmax,xmax),ylims =
(-xmax,xmax),fill = true,label = "Stable Region")
```



1.2 Runge Kutta 2

$$x_(n+1) = x_n + h * \frac{1}{2*\alpha} f(t_n, x_n) + \frac{1}{2*\alpha} f(t_n + \alpha * h, x_n + \alpha * h * f(t_n, x_n))$$
$$but$$

$ f(t*n*,*xn*) = lambda*x_n$

$$x_(n+1) = x_n + \frac{h * lambda * x_n}{2*\alpha} + \frac{1}{2*\alpha} f(t_n + \alpha * h, x_n + \alpha * h * lambda * x_n)$$
$$x_(n+1) = x_n + \frac{h * lambda * x_n}{2*\alpha} + \frac{1}{2*\alpha} f(t_n + \alpha * h, x_n(1 + \alpha * h * lambda)$$
$$x_(n+1) = x_n + \frac{h * lambda * x_n}{2*\alpha} + \frac{lambda * x_n(1+\alpha * h)}{2*\alpha}$$
$$x_(n+1) = x_n(1 + \frac{h * lambda}{2*\alpha} + \frac{lambda + lambda * \alpha * h)}{2*\alpha})$$
$$x_(n+1) = x_n(1 + h * lambda(\frac{h * lambda}{2} + 1))$$
$$so...$$
$$|1 + h * lambda(\frac{h * lambda}{2} + 1)| < 1$$

```
@vars x_n t_n alpha hlambda
k1 = f(x_n,t_n)
k2 = f(x_n +alpha*h*k1, t_n + alpha*h)
x_nPlus1 = x_n + h*((1-1/2/alpha)*k1+1/2/alpha*k2) |> simplify
```

$$x_n \left(h\lambda \left(0.5h\lambda + 1\right) + 1\right)$$

```
inequality(hlambda) = x_nPlus1 / x_n |> subs(h*lambda => hlambda)
```

```
inequality (generic function with 1 method)
```

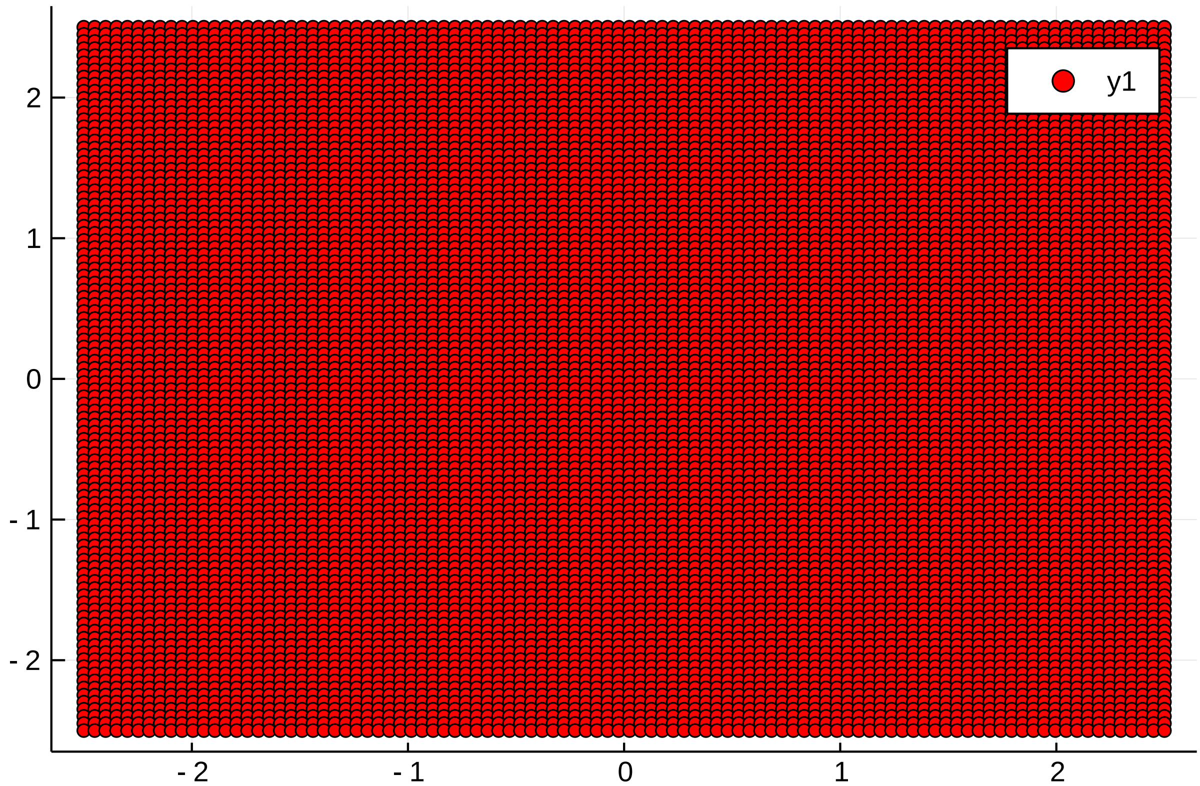
populate the domain hlambda

```
function ComplexLinespace(resolution,x_ymax)
  domainOf_h_lambda= fill(NaN+NaN*im,resolution*resolution,1)
  SizeOfPlot = collect(-x_ymax:2*x_ymax/(resolution-1):x_ymax)
  for i = 1:resolution
    newVals = (SizeOfPlot .+ (SizeOfPlot[i])*im)'
    indexlocale = collect((i-1)*resolution+1:(i-1)*resolution+resolution)'

    domainOf_h_lambda = setindex!(domainOf_h_lambda,newVals,indexlocale)

  end
  domainOf_h_lambda = domainOf_h_lambda
end
resolution = 100
x_ymax = 2.5
h_lambda_Vals = ComplexLinespace(resolution,x_ymax)

scatter([real(h_lambda_Vals)],[imag(h_lambda_Vals)],markercolor=[:red])
```



now we need to see if each point satifies stability

```
function StabilityTester(domain,inequality)
  n = size(domain,1)
  StablePoints = fill(NaN+NaN*im,n,1)
  UnstablePoints = fill(NaN+NaN*im,n,1)
```

```
  for i = 1:n
    if LinearAlgebra.norm(inequality(domain[i]))>=1
      #unstable
      UnstablePoints[i] = domain[i]
    else
      StablePoints[i] = domain[i]
    end
  end
  StablePoints = StablePoints
end


StablePoints = StabilityTester(h_lambda_Vals,inequality)

scatter([real(StablePoints)],[imag(StablePoints)],markercolor=[:blue],label = "Stable
Region",ylims = (-2.5,2.5),xlims=(-3.5,1.5))
```