Final Project

AMATH 586 Spring 2020

Due June 9 by 11pm on GitHub

1. This project is worth 20 points (same as the midterm). You may discuss it with other students. The point is for you to really understand the material and **write your own code**.
2. ***Please put your name clearly on the first page of your project.***
3. Provide plots to demonstrate your code. This is a good idea even if a plot is not explicitly asked for.
4. There are 6 extra credit points available if you want to make up some points or if you want to see how to extend your code and (hopefully) learn something new. If you complete the second extra credit problem you'll see something truly remarkable. That's in my opinion, of course.
5. This project is adapted from one given out by Randy LeVeque in 2008.

Problem 1

Consider the ODE:

$$v'(t) = \frac{1}{\epsilon} g(v(t)), \qquad g(v) = v(\alpha - v)(v - 1) - w.$$

Using the TR-BDF-2 method,

$$U^* = U^n + \frac{k}{4}(f(U^n) + f(U^*)),$$
$$U^{n+1} = \frac{1}{3}\left(4U^* - U^n + kf(U^{n+1})\right),$$

solve this ODE and verify that the method is indeed second-order accurate at $t = 0.1$. Use $v(0) = .3$, $\alpha = 0.5$, $w = 0.1$ and $\epsilon = 0.01$ as your parameters.

Note that you have to solve two nonlinear systems at each time step. The first one requires you to solve

$$G_1(u) = 0, \qquad G_1(u) = u - U^n - \frac{k}{4}(f(U^n) + f(u)), \qquad G_1'(u) = 1 - \frac{k}{4}f'(u).$$

The second is given by

$$G_2(u) = 0, \qquad G_2(u) = u - \frac{1}{3}\left(4U^* - U^n + kf(u)\right), \qquad G_2'(u) = 1 - \frac{k}{3}f'(u).$$

It is highly recommended that your write a function `TRBDF2(U,w,k)` that advances the solution one time step. This function may take in the `max_iter` and `tol` for Netwon's method as well.

Problem 2

The previous ODE becomes more interesting when it is coupled with another differential equation that evolves w:

$$\begin{aligned} v'(t) &= \frac{1}{\epsilon}(g(v(t)) - w(t) + I_a), \quad g(v) = v(\alpha - v)(v-1), \\ w'(t) &= \beta v(t) - \gamma w(t). \end{aligned}$$

This is a simple model that exhibits many features of excitable media, and is a simplification of the famous Hodgkin-Huxley equations that are a better model for propagation in neuronal systems. In this system $v(t)$ models the membrane potential while $w(t)$ models the concentration of an ion.

Since we have already gone through the effort to solve the above system, we would like to be able to reuse our code. Write the system as

$$\begin{bmatrix} v'(t) \\ w'(t) \end{bmatrix} = \mathcal{A}\left(\begin{bmatrix} v(t) \\ w(t) \end{bmatrix}\right) + \mathcal{B}\left(\begin{bmatrix} v(t) \\ w(t) \end{bmatrix}\right)$$

where

$$\begin{aligned} \mathcal{A}\left(\begin{bmatrix} v \\ w \end{bmatrix}\right) &= \begin{bmatrix} \frac{1}{\epsilon}(g(v) - w + I_a) \\ 0 \end{bmatrix}, \\ \mathcal{B}\left(\begin{bmatrix} v \\ w \end{bmatrix}\right) &= \begin{bmatrix} 0 \\ \beta v - \gamma w \end{bmatrix}. \end{aligned}$$

In the notation of (11.24) in the text, the function `TRBDF2(V,W,k)` described in the previous problem is the function $\mathcal{N}_{\mathcal{A}}(V, W, k)$. Construct the analogous function for $\mathcal{N}_{\mathcal{B}}(V, W, k)$, call it `RK2(V,W)` using the second-order Runge-Kutta method (5.30).

Then `Julia` code for Strang splitting (11.24) is given by

```
W[i] = RK2(V[i-1],W[i-1],k/2)
V[i] = TRBDF2(V[i-1],W[i],k)
W[i] = RK2(V[i],W[i],k/2)
```

This should be easily adapted to the language of your choosing.

Using

$$\alpha = 0.3, \quad \beta = 1, \quad \gamma = 1, \quad I_a = 0, \quad \epsilon = 0.001,$$

with initial data

$$v(0) = v_0, \quad w(0) = 0,$$

the solution of the ODE system exhibits two distinct behaviors:

1. If $v_0 = 0.31$ then $v(t)$ rapdily rises near $v = 1$ and then decays to the stable steady state $v = w = 0$.
2. If $v_0 = 0.29$ the the solution decays directly to the steady state $v = w = 0$.

Plot these two solutions. Numerically, verify that your method is second-order accurate at $t = 0.25$ when $v_0 = 0.29$.

Problem 3

Consider the heat equation

$$v_t = \kappa v_{xx}, \quad t > 0, \quad x \in (a, b).$$

We impose Neumann boundary conditions $v_x(0, t) = 0 = v_x(1, t)$. Because the boundary conditions do not directly specify $v(0, t)$ and $v(1, t)$, we need to include their approximations in our vector of unkowns. The MOL discretization can be written

$$V'(t) = \frac{\kappa}{h^2} A V(t).$$

where

$$V(t) = \begin{bmatrix} V_0(t) \\ V_1(t) \\ \vdots \\ V_{m+1}(t) \end{bmatrix}, \quad W(t) = \begin{bmatrix} W_0(t) \\ W_1(t) \\ \vdots \\ W_{m+1}(t) \end{bmatrix},$$

and

$$A = \begin{bmatrix} -2 & 2 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 2 & -2 & \end{bmatrix}.$$

Note: This is derived by introducing "ghost values" $V_{-1}(t)$ and $V_{m+2}(t)$ and then enforcing the Neumann conditions via

$$\frac{V_1(t) - V_{-1}(t)}{2h} = 0 = \frac{V_{m+2}(t) - V_m(t)}{2h}.$$

These relations are used to then eliminate the ghost values from the system. You need not derive this.

Apply the TR-BDF-2 method the MOL discretization and write a function `TRBDF2_heat(V)` that applies one time step of this method with time step `k`. It is best to keep `k` fixed here because then matrices do not have to be recomputed at each time step. With $a = 0, b = 4$, demonstrate that your code works by using the initial data

$$v(x, 0) = \begin{cases} 1 & a \le x < \frac{a+b}{2} \\ 0 & \text{otherwise.} \end{cases}$$

The integral of the solution should nearly be preserved, resulting in a long-time limit $v(x, t) \approx 1/2$ as $t \to \infty$. Demonstrate this with $h = 0.01$, $k = h$ and $\kappa = 1$ and discuss any discrepancy you see. It may help to choose other initial data to explain the situation.

Problem 4

The ODE system can be modified to allow for spatial variations: For $x \in [a, b]$

$$
\begin{aligned}
v_t(x,t) &= \kappa v_{xx}(x,t) + \frac{1}{\epsilon}(g(v(x,t)) - w(x,t) + I_a)\,, \qquad g(v) = v(\alpha - v)(v-1),\\
w_t(x,t) &= \beta v(x,t) - \gamma w(x,t),\\
v_x(a,t) &= 0,\\
v_x(b,t) &= 0.
\end{aligned}
$$

Now modify the function `TRBDF2_heat(V)` to apply one time step toward the solution of $v_t = \kappa v_{xx}$ with time step `k/2`.

Supposing that our previous code for `TRBDF2()` and `RK2()` is vectorized, Strang splitting becomes (with $I_a = 0$)

```
V = TRBDF2_heat(V)
W = RK2(V,W,k/2)
V = TRBDF2(V,W,k)
W = RK2(V,W,k/2)
V = TRBDF2_heat(V)
```

Use the parameters:

$$
\begin{gathered}
h = 0.02, \quad k = h/10, \quad a = 0, \quad b = 6, \quad \alpha = 0.3,\\
\beta = 1, \quad \gamma = 1, \quad \kappa = 0.2, \quad \epsilon = 0.001.
\end{gathered}
$$

and initial data

$$
v(x,0) = \begin{cases} 1 & 0 \le x < 1 \\ 0 & \text{otherwise,} \end{cases} \qquad w(x,0) = 0.
$$

Solve until $t = 4$. You should see a travelling wave develop and propagate.

Extra Credit Problem 1 (2 points)

Repeat the previous problem with

$$
v(x,0) = 0, \quad w(x,0) = 0, \quad I_a(x) = 0.8e^{-5x^2}.
$$

This should be a straightforward modification of the code if you've followed the outlined setup. This should produce a periodic firing of traveling waves.

Extra Credit Problem 2 (4 points)

Consider the extension of the previous problem to two spatial dimensions: For $x \in [a,b],\ y \in [a,b]$:

$$v_t(x,y,t) = \kappa v_{xx}(x,y,t) + \kappa v_{yy}(x,y,t) + \frac{1}{\epsilon}(g(v(x,y,t)) - w(x,y,t) + I_a), \quad g(v) = v(\alpha - v)(v-1),$$
$$w_t(x,y,t) = \beta v(x,y,t) - \gamma w(x,y,t).$$

with initial and boundary data given by

$$v(x,y,0) = \eta(x) = \begin{cases} 1 & x < 2, \\ 0 & \text{otherwise}, \end{cases} \qquad w(x,y,0) = 0,$$
$$v_x(a,y,t) = v_x(b,y,t) = v_y(x,a,t) = v_y(x,b,t) = 0.$$

The code we have created actually immediately generalizes to solve this problem. Suppose that matrices `W` and `V` contain the spatial array of the values obeying the ordering in Figure 3.2(a). For example, supposing that the grid spacing is the same in both directions the following code initializes `W` and `V`

```
x = a:h:b |> Array
y = x
X = repeat(reshape(x, 1, :), length(y), 1)
Y = repeat(reverse(y), 1, length(x));
η = (x,y) -> x < 2 ? 1.0 : 0.0
V = map(η,X,Y)
W = 0*V
```

Then the Strang splitting scheme is given by

```
V = TRBDF2_heat(V)
V = TRBDF2_heat(V')' |> Array
W = RK2(V,W,k/2)
V = TRBDF2(V,W,k)
W = RK2(V,W,k/2)
V = TRBDF2_heat(V)
V = TRBDF2_heat(V')' |> Array
```

The `|> Array` call just converts the transpose to a real array. `Julia` has a special data type for the transpose of an array that allows for efficient memory usage but we will avoid using that here. In `Matlab` or `Python` you wouldn't need an analogous call.

Set the parameters of the problem to be:

$$a = 0, \quad b = 12, \quad h = .05, \quad k = h/10, \quad \kappa = 1, \quad \epsilon = 0.01,$$
$$\alpha = 0.1, \quad \beta = 0.5, \quad \gamma = 1, \quad I_a = 0.$$

If you just solve with these parameters you should see a single pulse travel across the domain in the positive x-direction.

But if at time $t = 0.9$ zero out all elements of V with associated y coordinates being larger than 6, you break the symmetry in the y-direction. Solve until $t = 6$. You should see a spiral wave develop!