Problem 1  ODEs for Matrix-Valued Functions

Let the matrix-valued function $Y : \mathbb{R} \rightarrow \mathbb{R}^{d \times d}$ be a solution of the (matrix) differential equation
\[ \dot{Y} = AY \quad \text{with} \quad A \in \mathbb{R}^{d \times d}. \] (1.1)

(1a) Assume $A^\top H = -HA$. Show that $Y(t)^\top HY(t) = H$ for all $t > 0$ provided $Y(0)^\top HY(0) = H$.

HINT: You might want to compute $\frac{d}{dt}$ of $Y^\top HY$.

(1b) Implement the following functions in MATLAB

(i) function $Y = \text{ExplEulStep}(A, Y0, h)$,
(ii) function $Y = \text{ImplEulStep}(A, Y0, h)$,
(iii) function $Y = \text{ImplMidpStep}(A, Y0, h)$,

which, for a given initial value $Y(t_0) = Y_0$ and for a given step size $h$, compute approximations to $Y(t_0 + h)$ for the solution of (1.1) using a (single) step of

(i) the explicit Euler method,
(ii) the implicit Euler method,
(iii) the implicit mid-point method.

For (ii) and (iii), write out the closed form for $Y_{k+1}$ instead of using Newton’s method. Explain how you get the formula on your answer sheet.

(1c) Take now $A = \begin{pmatrix} -3 & -6 \\ 6 & 3 \end{pmatrix}$, $Y(0) = \frac{1}{\sqrt{3}} \begin{pmatrix} 2 \\ -1 \end{pmatrix}$, and $H = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$. Complete the template CompareNorms.m where, using the functions from subproblem (1b), you should compute discrete approximations $Y_k$ of $Y(kh)$, for $k = 1, \ldots, 20$ with $h = 1/20$. Compare the norms $\|Y_k^\top HY_k - H\|_F$, for $k = 1, \ldots, 20$ and all three methods, and comment on your observations with regards to the invariant from subproblem (1a).

HINT: The Frobenius norm $\|\cdot\|_F$ of a matrix can be computed using the command `norm(A, 'fro')`. 

Exam Summer 2016
(1d) Show that the solution $Y_k$ computed via the implicit mid-point rule satisfies:

$$\text{if } Y_0^T H Y_0 = H \text{ then } Y_k^T H Y_k = H \text{ for all } k \geq 1.$$

**Hint:** You might find the identity $Y_1 - Y_0 = \frac{h}{2} A (Y_0 + Y_1)$ useful.
Problem 2  Stability Domain of a Rational Single Step Method  [24.5 Marks]

Consider the rational function

\[ R(z) = \frac{2 - z^2}{2(1 - z)}. \]

(2a)  ☒ Determine the maximal \( p \in \mathbb{N} \) such that

\[ |\exp(z) - R(z)| = \mathcal{O}(|z|^{p+1}) \quad \text{for} \quad z \to 0. \]

**Hint:** Compute the first three derivatives of \( R(z) \) and use them to compare the Taylor series of \( \exp(z) \) and \( R(z) \) around the point 0.

(2b)  ☒ Consider \( R(z) \) as a stability function of a Runge-Kutta single step method and plot its stability domain in MATLAB by completing the template StabilityRegion.m.

(2c)  ☒ Show that a Runge-Kutta method with stability function \( R(z) \) is of convergence order 2 when applied to linear ODEs, that is, to problems of the form \( \dot{y} = \lambda y, \; y(0) = y_0. \)

(2d)  ☒ Write down (in detail) the discrete evolution of the single step method (whose stability function is \( R(z) \)), when applied to the autonomous linear differential equation

\[ \dot{y} = Ay, \quad A \in \mathbb{R}^{d \times d}. \]  \hspace{1cm} (2.1)

(2e)  ☒ Implement the method (in MATLAB) for the approximate solution of (2.1) by completing the template RationalSSM.m to solve the initial value problem

\[ \dot{y} = \begin{pmatrix} \alpha & \beta \\ -\beta & \alpha \end{pmatrix} y, \quad y(0) = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \]

for \( t \in [0, 10] \) with the values

<table>
<thead>
<tr>
<th></th>
<th>(i)</th>
<th>(ii)</th>
<th>(iii)</th>
<th>(iv)</th>
<th>(v)</th>
<th>(vi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>1.5</td>
<td>1.5</td>
<td>1.5</td>
</tr>
<tr>
<td>( \beta )</td>
<td>-1</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( h )</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1.5</td>
<td></td>
</tr>
</tbody>
</table>

where \( h \) is the step size. Plot your results and compare them with the exact solution. **Explain the behaviour of the method with all the six different sets of parameters with the help of the stability domain of \( R(z) \).**
Problem 3 A Volume Preserving Splitting Scheme

Consider the following ODE

\[
\dot{y} = f(y) = \begin{pmatrix}
-y_2 - \frac{y_1}{a^2 + y_3^2} \\
y_1 - \frac{y_2}{a^2 + y_3^2} \\
\frac{2\arctan\left(\frac{y_3}{a}\right)}{a} \\
\end{pmatrix}, \quad y \in \mathbb{R}^3, \quad a > 0.
\]  

(3a) Show that the flow of the ODE (3.1) is volume preserving.

On one hand, [?, Lemma. 4.2.5] dictates that there does not exist a Runge-Kutta scheme that is volume preserving for all problems in \(\mathbb{R}^3\). On the other hand however, any SSM that preserves quadratic invariants is volume preserving in \(\mathbb{R}^2\).

(3b) Split the vector field \(f\), given as the right hand side of (3.1), as a sum of two-dimensional divergence-free vector fields, that is, as \(f = f_1 + f_2 = \begin{pmatrix} f_1^1 \\ 0 \\ f_3^1 \end{pmatrix} + \begin{pmatrix} 0 \\ f_2^2 \\ f_3^2 \end{pmatrix}\).

HINT: Consider the construction in [?, Lemma. 4.2.6]

Now that we have the splitting \(f(y) = f_1(y) + f_2(y)\) we can construct a volume preserving SSM. Take a Runge-Kutta SSM that preserves quadratic invariants, and denote by \(\Psi_i^h\) the flow of this SSM when applied to the ODE \(\dot{y} = f_i(y), i = 1, 2\). The functions \(f_1\) and \(f_2\) are divergence free vector fields, and are (essentially) two-dimensional, while the flows \(\Psi_i^h\) are volume preserving. Hence, by applying a suitable splitting scheme (here we use Strang splitting) we can construct a volume preserving scheme

\[
\Psi^h = \Psi_2^{h/2} \circ \Psi_1^h \circ \Psi_1^{h/2}
\]

since the composition of volume preserving schemes is again volume preserving.

(3c) Finish the \MATLAB\ code

```matlab
function y = GaussStep(y0, f, Df, h)
    % which computes one step of the Gauss collocation scheme of order 4.
    % The inputs are the initial value y0, the right hand side of the given ODE f, the derivative of the right-hand side Df, and the stepsize h.
    % In order to compute the coefficients of the given Runge-Kutta method use the codes collCoeffs.m and GaussNodes.m, and in order to solve the underlying implicit system for the stages, use Newton’s algorithm by completing the template newton.m.
```

(3d) Consider the initial value problem

\[
\dot{y} = \begin{pmatrix}
-y_2 - \frac{y_1}{a^2 + y_3^2} \\
y_1 - \frac{y_2}{a^2 + y_3^2} \\
\frac{2\arctan\left(\frac{y_3}{a}\right)}{a} \\
\end{pmatrix}, \quad y(0) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad a = 1.
\]
Find its solution by using the previously described volume-preserving splitting scheme. Here $\Psi_1^h$ and $\Psi_2^h$ are both to be computed using steps of the Gauss collocation method of order 4. Complete the template VolumePreservingSplitting.m in which you should use GaussStep.m from (3c) in each step, and Strang splitting to compute the approximate solution.
Problem 4  Extrapolating Implicit Mid-Point and Explicit Euler  [28 Marks]

In this problem we will apply the extrapolation method to implicit mid-point rule and to explicit Euler, and then compare their performance. Consider the logistic ODE

\[ \dot{y} = \lambda y (1 - y), \quad \lambda > 0, \quad (4.1) \]

with the initial value \( y(0) = y_0 > 0 \)

(4a) Find the fixed points of the ODE (4.1) and determine if any of them are attractive. Explain why given \( y(0) > 0 \) it follows that \( y(t) > 0 \) for all \( t > 0 \).

(4b) Give the closed form of the discrete evolution \( \Psi^h y \) of the implicit mid-point rule when applied to the logistic differential equation (4.1), and argue whether the solution is admissible assuming the initial value satisfies \( y(0) > 0 \).

Hint: The discrete evolution of the implicit mid-point rule leads to a quadratic equation which admits an explicit solution. Then use (4a) to conclude which of the two expressions makes sense for \( y(0) > 0 \).

(4c) Complete the templates

\[
\text{function } y = \text{ImplicitMidpoint}(y0, \lambda, h, n) \]

and

\[
\text{function } y = \text{ExplicitEuler}(y0, \lambda, h, n),
\]

where for a given initial value \( y0 \), positive parameter \( \lambda \), step size \( h \), and an integer \( n \) you should perform \( n \) iterations of implicit midpoint and explicit Euler method, respectively, for the solution of the IVP (4.1) and return only the value at the end point.

(4d) Implement a MATLAB function

\[
\text{function } y = \text{extrapolate}(Y, n, p),
\]

that interpolates the sequence of pairs \( \{n_i^{-1}, y_i\}_{i=1}^{k+1} \) with the polynomial

\[
q(t) = \alpha_1 t^{p+k-1} + \alpha_2 t^{p+k-2} + \cdots + \alpha_k t^p + \alpha_{k+1}
\]

and returns the extrapolated value \( q(t=0) \).

(4e) Use the MATLAB template

\[
\text{function } [yE, yI] = \text{ExtrapolatedMethods}(y0, T, N, n, \lambda) \]

\[
\text{function } [yE, yI] = \text{ExtrapolatedMethods}(y0, T, N, n, \lambda)
\]

to implement the extrapolated implicit midpoint and explicit Euler methods. The inputs are the initial value \( y0 \), the end point \( T \), the number of steps in the subdivision \( N \), the vector \( n \) of integers that define the extrapolation method, and the value \( \lambda \) from the right hand side of (4.1).

Hint: Evaluation of the extrapolation polynomial from the lecture notes can be done using (4d).
Consider again the ODE (4.1), where we take $y(0) = 0.03$, $\lambda = 5$, and complete the template ExtrapolateCompare.m which performs a convergence study of the extrapolated implicit midpoint and explicit Euler methods, and compares them. In order to conduct the convergence study we take $N = 2.^(3:8)$ subdivision of the time interval $[0 \ 1]$, and define the extrapolation method using $n = 1:3$ as your sequence of integers. Approximate the convergence order of the two methods using MATLAB’s function polyfit.