MA2915 Linear and Numerical Methods
FIRST MATLAB ASSIGNMENT, AUTUMN 2003

Deadline: 1700 on Monday 24 November 2003

Electronic submission required, see below.

This is the first of two Matlab assignments which forms part of your assessment for MA2915 Linear and Numerical Methods.

Description

This assignment is concerned with writing Matlab code to display the solution to a system of linear, second order, constant coefficient ordinary differential equations of the form

$$\underline{u}'' = A\underline{u}' + B\underline{u}, \quad \underline{u}(t_0) = \underline{u}_0, \quad \underline{u}'(t_0) = \underline{u}_{d0} \quad \text{where } \underline{u}' = \frac{d\underline{u}}{dt}, \quad \underline{u}'' = \frac{d^2\underline{u}}{dt^2}, \tag{1}$$

where $\underline{u}$ is a vector of length $m \geq 1$ and where A, B are real constant matrices of size $m \times m$. This system of second order equations can be re-expressed as an equivalent system of first order equations of length $2m$ which we can write as

$$\begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \end{bmatrix}' = \begin{bmatrix} \mathbf{0} & I \\ B & A \end{bmatrix} \begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \end{bmatrix}, \quad \text{where } \begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \end{bmatrix}(t_0) = \begin{bmatrix} \underline{u}_0 \\ \underline{u}_{d0} \end{bmatrix} \tag{2}$$

where I is the $m \times m$ identity matrix and where $\mathbf{0}$ is $m \times m$ matrix with all entries equal to 0. Provided the matrix

$$M = \begin{bmatrix} \mathbf{0} & I \\ B & A \end{bmatrix}$$

has distinct eigenvalues, the solution of (2) is given by

$$\begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \end{bmatrix}(t) = c_1 e^{\lambda_1 t}\underline{w}_1 + c_2 e^{\lambda_2 t}\underline{w}_2 + \cdots + c_{2m} e^{\lambda_{2m} t}\underline{w}_{2m}, \tag{3}$$

where $\{\lambda_i, \underline{w}_i\}$, $i = 1, 2, \ldots, 2m$ are eigenvalue-eigenvector pairs for the matrix M defined above and $c_1, c_2, \ldots, c_{2m}$ are determined from the initial conditions. In matrix form the coefficients $\underline{c} = (c_i) \in \mathbb{R}^{2m}$ satisfy the linear system

$$W\underline{c} = \begin{bmatrix} \underline{u}_0 \\ \underline{u}_{d0} \end{bmatrix} \quad \text{where} \quad W = \left[\underline{w}_1, \cdots, \underline{w}_{2m}\right]. \tag{4}$$

The solution to (1) is then given by $\underline{u}(t) = \underline{y}_1(t)$.

Let $t_f > t_0$ denote the final time at which the solution is required, let n be the number of equally spaced times we need to evaluate the solution given by (3) and let

$$h = \frac{t_f - t_0}{n} \quad \text{and} \quad t_i = t_0 + ih, \; i = 1, 2, \cdots, n.$$

Also let $\underline{Z}(t)$ denote the vector of length $2m$ given by

$$\underline{Z}(t) = \begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \end{bmatrix}(t) \quad \text{and note that } \underline{Z}(t_0) = \begin{bmatrix} \underline{u}_0 \\ \underline{u}_{d0} \end{bmatrix}.$$

Given matrices A and B and given vectors $\underline{u}_0$ and $\underline{u}_{d0}$, the aim of this assignment is to write Matlab code which will evaluate $\underline{Z}_i = \underline{Z}(t_i)$ for $i = 1, 2, \cdots, n$ and to create various plots as described below.

The Tasks

You need to do the following:

- Create a Matlab function file called `lde.m` with a first line of the form

  ```
  function [t,U] = lde(A,B,t0,tf,u0,ud0,n)
  ```

  This has 7 input arguments and 2 output arguments. The first 4 inputs are the constant matrices `A` and `B` in the initial value problem (1) to be solved, the initial time `t0` and the final time `tf`. The 5th and 6th inputs `u0` and `ud0` are both column vectors of length m corresponding respectively to the column vectors $\underline{u}_0$ and $\underline{u}_{d0}$ giving the initial values, and the 7th argument is the number of time steps `n`. The output argument `t` is a vector of length `n+1` containing the times $t_0, t_1, \cdots, t_n$ and the the output argument `U` to be generated is a matrix of size $m\times$`(n+1)` with the first column of `U` being equal to the vector `u0` and with the $(i+1)$th column of `U` being $\underline{u}(t_i)$ for $i = 1, \cdots, n$. To create `U` you need to use Matlab built-in functions to determine the eigenvalues and eigenvectors of M and to solve the linear system (4) and you need to evaluate (3) at the values t_i, $i = 1, \cdots, n$.

- You need to create a Matlab script file called `astest1.m` which will use `lde` to generate the approximate solution to (1) in the following case

  $$A = \begin{bmatrix} -1 & -1 \\ 2 & -4 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 1 & -1 \\ 1 & 2 \end{bmatrix},$$

  $t_0 = 0$, $t_f = 10$, $n = 100$ and the initial conditions of

  $$\underline{u}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \underline{u}_{d0} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

  Once you have obtained the solution at times $t_0, t_1, \cdots, t_n$ your script should have instructions which creates 2 plots. On one of the plots you should show the graphs of both $u_1(t)$ against t and $u_2(t)$ against t for $0 \le t \le t_f$. On the other plot you should show the graph of u_1 against u_2 corresponding to $0 \le t \le t_f$.

A summary of the tasks

You need to create the function file `lde.m` and the script file `astest1.m` . We expect each of the files to contain suitable comments describing what they do and in each of the files **you should put your name in a comment line at the beginning of the file.** For the 2 plots generated we expect these to have suitable labelling and a title.

What you have to submit

For the submission **we only require an electronic version of your work** and this can be done using either of the following ways.

By email

(i) Once you have the versions of `lde.m` and `astest1.m` that you wish to submit you need to create a `zip` file which contains these 2 files. You can do this by typing the following at the command line in a command window.

```
zip ma2915as1 lde.m astest1.m
```

(ii) You should send the `zip` file `ma2915as1.zip` as an attachment to `Mike.Warby@brunel.ac.uk` using your preferred mail program.

The reason for asking you to create the `zip` file here is that some mail programs corrupt attachments which only involve plain text.

You will receive a reply by email indicating whether or not the submission was successful. You should not consider the submission as complete until you receive a satisfactory reply.

By allowing us to copy the files across the network

The following steps will enable us to retrieve the final versions of `lde.m` and `astest1.m` that you wish to submit and attempts to ensure that nobody else on the network can obtain useable versions of your work.

(i) Check that you have a `u:` drive. This should usually be mounted as part of your Matlab session. If it is not mounted then in a command window you will need to type

```
net use u: \\pinn.brunel.ac.uk\ma02xyz /user:ma02xyz
```

with `ma02xyz` replaced by your login id.

(ii) Login to `molnir` using putty. You get this by typing in a command window

```
putty molnir
```

You will need a `putty` window and a command window at the same time.

(iii) In the command window on windows you should type at the prompt

```
copy lde.m u:\
copy astest1.m u:\
```

If you get any error here then you have not done this successfully.

(iv) At the command prompt on `molnir` (i.e. at the prompt `%`) in the `putty` window you should type

`~icstmkw/bin/ma2915_as1` *`your_8_character_encode_key`*

This will adjust the permissions on `u:\lde.m` and `u:\astest1.m` to ensure that nobody else can copy these files and it will create encrypted versions of these files that we can copy and decode. Your 8 character key will be sent to you by email well before the submission deadline.

(v) To get an acknowledgement to confirm that everything has been done successfully you should send an email to `Mike.Warby@brunel.ac.uk` indicating that your files are ready to be collected. You will receive a reply indicating whether or not the 2 files could be collected. You should not consider the submission as complete until you receive a satisfactory reply.

Some remarks, comments and tips

1. You need to create `lde.m` before you can test it using `astest1.m`.

2. Your function file `lde.m` should cope with systems of any size and thus the function will need to determine the the length m of the system. You can obtain this with the Matlab statement

   ```
   m=length(u0);
   ```

   which should be near the start of the function file.

3. To compute the vector `t`, to reserve space for the matrix `U` and to reserve space for a matrix `Z` to store the intermediate vectors corresponding to $\underline{Z}_0, \underline{Z}_1, \cdots, \underline{Z}_n$ you can use statements of the form

   ```
   t=linspace(t0,tf,n+1);
   U=zeros(m,n+1);
   Z=zeros(2*m,n+1);
   ```

4. In Matlab you have to use `t(1)` for the initial time `t0` with `t(i+1)` corresponding to t_i. When you refer to the entries of a vector or matrix in Matlab the indices start at 1.

5. If `U` and `Z` have been created as in 3 then in Matlab you can refer to the ith column of `Z` by using `Z(:,i)` and you can set the ith column of `Z` with a statement of the form

   ```
   Z(:,i)= some expression which gives a column vector of length 2*m
   ```

6. You can create the matrix `M` by a statement of the form

   ```
   M=[zeros(m,m), eye(m);
      B          , A];
   ```

   `zeros(m,m)` generates a matrix of the specified size with all entries equal to 0 and `eye(m)` generates the identity matrix of the specified size.

7. Matlab has many built-in functions to perform matrix operations. For example the function `eig` computes eigenvalues and eigenvectors and `\` is for solving linear equations. With the statements

```
[W D]=eig(M);
c=W\[u0;ud0];
d=diag(D);
```

you get the matrix W of (4), the vector `c` corresponds to the vector $\underline{c}$ in (4) and `d` contains the eigenvalues $\lambda_1, \cdots, \lambda_{2m}$.

8. With column vectors `u0` and `ud0` each of length `m` you can set the first column of `Z` which is of length `2*m` by the statement

```
Z(:,1)=[u0;ud0];
```

9. You will need to use a `for` loop to successively generate $\underline{Z}_1, \cdots, \underline{Z}_n$ after you have initially set $\underline{Z}_0$. You can however avoid using a `for` loop to evaluate

$$\sum_{k=1}^{2m} c_k e^{\lambda_k s} \underline{w}_k \tag{5}$$

at any $s = t_i$ in (3) if you want by using the vectorised facilities in Matlab. If you have created a matrix of the eigenvectors `W`, a column vector `d` of the eigenvalues and a vector `c` of the coefficients as in 7 then (5) can be computed as a matrix vector product by the statements

```
ce=c.*exp(s*d);
Z(:,i+1)=W*ce;
```

In this `s*d` is the column vector containing $[sd_1, \cdots, sd_{2m}]^T$. The column vector `exp(s*d)` contains $[e^{sd_1}, \cdots, e^{sd_{2m}}]^T$. The column vector `ce` contains $[c_1 e^{sd_1}, \cdots, c_{2m} e^{sd_{2m}}]^T$. The product `W*ce` gives the product of the matrix `W` and the column vector `ce`. This is what is needed. Note that a matrix times a vector gives a vector which is a linear combination of the columns of the matrix which is what we have in (5).

10. Once you have computed the matrix `Z` of size `(2*m,n+1)` the matrix `U` that you need corresponds to the first `m` rows. You can get this with the Matlab statement

```
U=Z(1:m,:);
```

11. You should know that a real matrix M can have complex eigenvalues and associated complex eigenvectors and thus depending on A and B some of the calculations in `lde.m` may be complex. The complex eigenvalues will however occur in complex conjugate pairs, i.e. if $u + iv$ is an eigenvalue then $u - iv$ is an eigenvalue, $u, v \in \mathbb{R}$, and the answer `U` you will get will be real if all computations are done exactly. With floating point computations as performed in Matlab computations are not performed exactly but are computed to an accuracy of about 16 significant figures. To remove warnings connected with any very small imaginary parts in your final answer you can ensure that the result is real with a statement of the form

```
U=real(U);
```

12. To do the plots you should note that `U(1,:)` and `U(2,:)` give respectively row vectors of length `n+1` containing the values of u_1 and u_2.

13. You can label your plots using the `xlabel`, `ylabel` and `legend` commands as appropriate and you can get a title using the `title` command. To have both figures available at the same time you should use the `figure` command before each set of plot instructions; see the Matlab notes for some examples.

14. For the final submission of your work we require that `A`, `B` and the initial conditions are as given. You can however check that everything is working properly by using other matrices for which you can more easily compute the eigenvalues of M, solve the system for $\underline{c}$ and determine what the plots should look like.