RMIT University, School of Science
Math2136 – Computational Mathematics
Assignment 1

Submit by 11pm, August 21, 2016.

Tridiagonal Algorithm: Multiple Right Hand Sides

Sometimes we need to solve tridiagonal linear systems with the same coefficient matrix but different right hand sides. We then only need to do one *LU*-factorization of the coefficient matrix. The forward and backward substitutions can be performed separately for each right hand side.

Initially arrays `a[j]`, `b[j]` and `c[j]` store the three diagonals of the tridiagonal coefficient matrix.

- The main diagonal is `b[j]`, for `j=1,2,...,n`.
- The upper diagonal is `c[j]`, for `j=1,2,...n-1`.
- The lower diagonal is `a[j]`, for `j=1,2,...,n-1`.
- The right hand side vector is stored in the array `f[j]`, for `j=1,2,...,n`.

The LU factorization of the tridiagonal coefficient matrix is:

```
For j from 1 to ndim-1 do
    a[j] = a[j]/b[j]
    b[j+1] = b[j+1] - a[j]*c[j]
    end do
```

Arrays `a[j]` and `b[j]` are overwritten and now store the required elements of the *LU*-factorization.

The forward substitution stage is:

```
For j from 1 to ndim-1 do
    f[j+1] = f[j+1] - a[j]*f[j]
    end do
```

Note array `f[j]` is overwritten.

The backward substitution stage is:

```
f[ndim] = f[ndim]/b[ndim]
For k from ndim-1 to 1 by -1 do
    f[k] = (f[k] - c[k]*f[k+1])/b[k]
    end do
```

On completion the solution vector is stored in the array `f[j]`, for `j=1,2,...,n`.

Assignment

1. (a) Write a Maple procedure, a Matlab function or a *Mathematica* module to implement the *LU*-factorization of the Tridiagonal algorithm for multiple right hand sides.
    - The inputs to the procedure are the three one dimensional arrays a, b, and c.
    - A Maple procedure should check that a, b and c are of type `Array`.

- Your Maple procedure, Matlab function or *Mathematica* module will have to determine the dimension of the system (from array b) or this could be a fourth input which should have type positive integer (`posint`) if coded in Maple.
- On completion the procedure should output the modified arrays a and b.
- An example Maple procedure is:

```
procedurename := proc(Aarray::Array, Barray::Array,.....)
   local i, j, ....;
   description "some comments";
   Commands;
   Aarray,Barray;
end:
```

  Make sure to choose an appropriate name for your procedure.
- An example Matlab function is:

```
function [Aout,Bout]=SomeName(Aarray,Barray,Carray)
Commands;
end
```

  Make sure to choose an appropriate name for your function.
- An example *Mathematica* module is:

```
SomeName[a_,b_,c_]:=
    Module[{i, j, .....,Aout,Bout},  (* local variables *)
    Commands;
    {Aout,Bout}
]
```

  Make sure to choose an appropriate name for your module. a, b and c are one dimensional lists.

(b) Write a Maple procedure, a Matlab function or *Mathematica* module to implement the forward and backward substitution stages of the Tridiagonal algorithm for multiple right hand sides (given that the LU-factorization has already been determined).

- The inputs to the procedure are the modified one-dimensional arrays a and b from the LU-factorization, the array c (upper diagonal) and the right hand side vector f.
- A Maple procedure should check that a, b, c and f are of type `Array`.
- Your Maple procedure, Matlab function or *Mathematica* module will have to determine the dimension of the system (from arrays b or f) or this could be a fifth input which should have type positive integer (`posint`) if coded in Maple.
- On completion the procedure should output the solution of the tridiagonal linear system.

(c) Create a 6 by 6 tridiagonal linear system with known solution and use it to test your Maple procedures, Matlab functions or *Mathematica* module.

That is, solve the 6 by 6 tridiagonal linear system using your Maple procedures, Matlab functions or *Mathematica* module.

Heat Conduction in Clothing

An arctic construction worker is wearing a protective suit made of M400 Thinsulate. Inside a warm building, the suit and heat generated by the worker, keep her skin temperature at about 30°C.

When the worker leaves the protection of the building (at $t = 0$) into sub-zero air temperature, the temperature on the surface of the suit ($x = 0$) reduces instantaneously to the ambient temperature U_A.

The suit has thickness Lcm. Initially (at $t = 0$) the temperature profile in the suit satisfies a linear relationship on $0 \le \frac{x}{L} \le 1$.

At the workers skin ($\frac{x}{L} = 1$), the heat flux generated by the worker ($\frac{\partial u}{\partial t} = Fl$) remains constant for all time t.

Heat flow in the suit is governed by the heat equation: $\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2}$, where κ is the thermal diffusivity. For M400 Thinsulate the thermal diffusivity is $0.0055\text{cm}^2\text{sec}^{-1}$.

If the suit is divided into N equally spaced grid points, then the grid spacing is $h = \frac{L}{N}$ and the grid points are $x_i = i.h$, for $i = 0, 1, 2, \ldots, N$.

Let U_i^j represent the temperature at $x = x_i$ when $t = j.\Delta t$, for $j = 0, 1, 2, \ldots$

The initial temperature at each grid point is given by $U_i^0 = A(i\frac{h}{L}) + B$ for $i = 1, 2, 3, \ldots, N$ and $U_0^j = U_A$, for all j.

Using a finite difference discretization the heat equation reduces to solving the following tridiagonal linear system:

$$\begin{aligned}
&-sU_{i-1}^{j+1} + (1+2s)U_i^{j+1} - sU_{i+1}^{j+1} = U_i^j, && \text{for } i = 2, 3, \ldots, N-1,\\
&(1+2s)U_1^{j+1} - sU_2^{j+1} = U_1^j + s.U_A, && \text{for } i = 1,\\
&-2sU_{N-1}^{j+1} + (1+2s)U_N^{j+1} = U_N^j - 2s.h.Fl, && \text{for } i = N, \text{ where } s = \kappa\frac{\Delta t}{h^2}.
\end{aligned}$$

In matrix form the system looks like:

$$\begin{bmatrix}
1+2s & -s & 0 & 0 & 0 & \cdots & \cdots & 0\\
-s & 1+2s & -s & 0 & 0 & \cdots & \cdots & 0\\
0 & -s & 1+2s & -s & 0 & \cdots & \cdots & 0\\
- & - & \ddots & \ddots & \ddots & - & - & -\\
- & - & - & \ddots & \ddots & \ddots & - & -\\
0 & \cdots & \cdots & 0 & -s & 1+2s & -s & 0\\
0 & \cdots & \cdots & 0 & 0 & -s & 1+2s & -s\\
0 & \cdots & \cdots & 0 & 0 & 0 & -2s & 1+2s
\end{bmatrix}
\begin{bmatrix} U_1^{j+1}\\ U_2^{j+1}\\ U_3^{j+1}\\ \vdots\\ \vdots\\ \vdots\\ U_{N-1}^{j+1}\\ U_N^{j+1} \end{bmatrix}
=
\begin{bmatrix} U_1^{j}\\ U_2^{j}\\ U_3^{j}\\ \vdots\\ \vdots\\ \vdots\\ U_{N-1}^{j}\\ U_N^{j} \end{bmatrix}
+
\begin{bmatrix} s.U_A\\ 0\\ 0\\ \vdots\\ \vdots\\ \vdots\\ 0\\ 2s.h.Fl \end{bmatrix}$$

Given the temperature profile at time $t = j.\Delta t$, we can then solve the tridiagonal linear system and find the temperature profile at time $t = (j+1).\Delta t$.

The tridiagonal coefficient matrix does not change so we need only do one LU-factorization.

Assignment

Based on the last digit of your student number, the remaining parameters are given in the following Table.

Last digit	0	1	2	3	4
L(cm)	4	6	5	4	7
U_A(°C)	-5	-10	-12	0	-25
N	50	100	75	60	70
Fl(°Csec^{-1})	2.5	2.0	1.8	2.2	2.7
A	10	12	5	9	12
B	20	20	25	22	18

Last digit	5	6	7	8	9
L(cm)	5	5.5	6	4.5	5.5
U_A(°C)	-16	-15	-13	-3	-6
N	80	100	90	85	110
Fl(°Csec^{-1})	2.2	1.6	1.6	2.25	1.75
A	5	12	10	8	4
B	26	22	22	18	22

2. Set $\Delta t = 0.5$sec. Calculate h and s.
   Create an array of N elements storing the initial temperature profile.
   Plot the initial temperature profile.

3. Create three one dimensional arrays storing the diagonals of the tridiagonal coefficient matrix. Use your MatLab .m function, Maple procedure or *Mathematica* module from Question 1(a) to find the LU-factorization of the coefficient matrix.
   You will not need to find the LU-factorization again.

4. Find the temperature profile after ten minutes. That is: Do 1200 time steps. (Solve the linear system 1200 times.)

   Use your MatLab .m function, Maple procedure or *Mathematica* module from Question 1(b) to solve the linear system given the LU-factorization.

   Plot the temperature profile against thickness (cm) after ten minutes. Remember to add the surface temperature point to the set of data.

   Label the axes and include a plot title.

5. Start the process again, (re-set the initial conditions), increase the number of time steps if necessary and determine in minutes when the skin temperature of the construction worker drops below 20°C. (The last element of the temperature array is the skin temperature of the worker.)