数值计算方法实验报告

实验二：追赶法解线性方程组

数据科学与计算机学院　17大数据与人工智能

17341015　陈鸿峥

一、 实验题目

写出用追赶法求解下述线性方程组的程序，其中$n = 101$

$$\begin{bmatrix} 12 & 1 & 0 & \cdots & 0 \\ 1 & 12 & 1 & \cdots & 0 \\ 0 & 1 & 12 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 1 \\ 0 & 0 & \cdots & 1 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 11 \\ 10 \\ 10 \\ \vdots \\ 11 \end{bmatrix}$$

二、 实验目的

理解追赶法解三对角方程的原理，并实现相关程序。

三、 实验原理与内容

由课本2.2.3节，给出一般的三对角方程解法，设n阶方程组$A\mathbf{x} = \mathbf{d}$，其中

$$A = \begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix}, \mathbf{d} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

对矩阵A做克洛脱分解，有

$$L = \begin{bmatrix} l_1 & & & & \\ v_2 & l_2 & & & \\ & \ddots & \ddots & & \\ & & v_{n-1} & l_{n-1} & \\ & & & v_n & l_n \end{bmatrix}, U = \begin{bmatrix} 1 & u_1 & & & \\ & 1 & u_2 & & \\ & & \ddots & \ddots & \\ & & & 1 & u_{n=1} \\ & & & & 1 \end{bmatrix}$$

进而$A = LU$，做两次回代即可

$$\begin{cases} L\mathbf{y} = \mathbf{d} \\ U\mathbf{x} = \mathbf{y} \end{cases}$$

可得下面程序中的公式，其中Forward为追的过程，Backward为赶的过程。

下面为本次实验的Mathematica源代码，完整文件已在附件中TriDiagSolver.nb。

```
AllMat[j_, n_] := Table[j, {i, 1, n}];
ResMat[n_] := Table[If[i == 1 || i == n, 11, 10], {i, 1, n}];
ZeroMat[n_] := Table[0, {i, 1, n}];
Num = 101;
(*Initialization*)
a = AllMat[1, Num]; b = AllMat[12, Num]; c = AllMat[1, Num];
d = ResMat[Num];
l = ZeroMat[Num]; y = ZeroMat[Num]; u = ZeroMat[Num]; x = ZeroMat[Num];
(*TriDiagSolver[n_]*)
TriDiagSolverForward[n_] := For[i = 1, i <= n, i++,
   If[i == 1, l[[i]] = b[[i]],
    If[i == n, l[[i]] = b[[n]] - a[[n - 1]]*u[[n - 1]],
     l[[i]] = b[[i]] - a[[i - 1]]*u[[i - 1]]]];
   If[i == 1, y[[1]] = d[[1]]/l[[1]],
    If[i == n, y[[n]] = (d[[n]] - y[[n - 1]]*a[[n - 1]])/l[[n]],
     y[[i]] = (d[[i]] - y[[i - 1]]*a[[i - 1]])/l[[i]]]];
   If[i == 1, u[[1]] = c[[1]]/l[[1]],
    If[i < n, u[[i]] = c[[i]]/l[[i]],
     0]]];
TriDiagSolverBackward[n_] := For[i = n, i > 0, i--,
   If[i == n, x[[n]] = y[[n]],
    x[[i]] = y[[i]] - u[[i]]*x[[i + 1]]]];
TriDiagSolverForward[Num];
TriDiagSolverBackward[Num];
N[x]
```

四、 实验结果与分析

下面是$\mathbf{x}$的结果，一共101个元素，为了编排分成11行。

0.858149	0.702213	0.715299	0.714201	0.714293	0.714285	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286	0.714286
0.714286	0.714286	0.714286	0.714286	0.714286	0.714285	0.714293	0.714201	0.715299	0.702213
0.858149									

通过将结果回代验证，结果是正确的，且精度非常高。

本实验说明用追赶法解三对角方程对于计算机来说非常的简单，而且可以简便快捷地得到很高精度的结果。

五、 实验总结和心得

本次实验较为简单，只需将三对角方程的算法在计算机上实现一遍即可，而且由于是简单的循环赋值，用其他高级语言如C++/Python等也可轻松实现。而轻松实现的前提是有深厚的数学基础，由此看来数值计算的作用非常强大，既能告诉你如何解出高精度的正确结果，又能通过优化算法尽可能快地解出答案。