**Algorithm 2:** LU decomposition

|  |
| --- |
| Method introduction: |
| 简单叙述这个方法的计算思想，计算公式，适用对象，优缺点等内容。这部分属于偏数学的东西。  （英文书写不要超过1页）  To solve the equation: *Ax=b.*  The idea of algorithm is to decompose the coefficient matrix *A* into a lower triangular matrix *L* and an upper triangular matrix *U*. Then the original problem is decomposed into two simple problems.  The computational formulae are:  *Ly=b; Ux=y.*  This algorithm is suitable for any non-singular matrix.  The advantage of this method is easy to operate. This method may be occurred error when the diagonal entries are closed to the zeros. Besides, this method is very time-consuming for large-scale problem. |
| Algorithm Design |
| 算法设计和实施部分，这部分叙述计算机编程，怎么去实现。  可以是流程图，也可以伪代码，也可以用  step 1  step 2  step 3…  叙述，要求做到别人看了这部分也能编程实现。  需要说清楚输入变量，输出变量含义。  （英文书写最多不要超过2页）  Step 1. Decompose the matrix *A* into a lower triangular matrix *L* and an upper triangular matrix *U.*  Step 2. Solve the equation *Ly=b* and obtain the unknown vector *y*.  Step 3. Solve the equation *Ux=y* and obtain the solution vector *x*. |
| Matlab code |
| 此处是程序，直接从函数窗口拷贝过来  需要是function格式，关键部分最好有注释  此处长短不限，但代码最好优化一下，条例清楚  考虑到学生是初学代码，尽量不要用高级命令  function [L, U] = MyLU(A)  % This is the code for LU Decompsition without Pivoting. A = LU.  % Input:  % A n by n coefficient matrix  % Output:  % L n by n lower triangular matrix  % U n by n upper triangular matrix    % Get size of input.  [n, n] = size(A);    % Initialize matrixes.  L = eye(n);  U = A;    % LU Decomposition.  for j = 1 : n  for i = j+1 : n  L(i, j) = U(i, j) / U(j, j);  U(i, :) = U(i, :) - L(i, j)\*U(j, :);  end  end    % Display L and U  L  U    % Verify results  B = L\*U  A |
| Examples and Result |
| 此处需要展示你的程序对课本中的例题习题的计算结果  最好是书上的例题，这样结果正确能保证程序正确，也可以和其它的算法进行结果比较。  要求：迭代列出初值，前5次结果和最后3次结果，中间用省略号。一个方法可以展示一个例子，最多展示3个同类例子。  A=[6 4 2; 3 -2 -1;3 4 1];  >> [L,U]=MyLU(A)  L =  1.0000 0 0  0.5000 1.0000 0  0.5000 -0.5000 1.0000  U =  6 4 2  0 -4 -2  0 0 -1  Remarks |
| 此处写该方法程序设计的一些注意事项，也可以空白 |
|  |