The University of Texas
at Austin

CS383C Numerical Analysis

HW06 LU Factorization

Edited by LaTeX

Department of Computer Science

STUDENT

Jimmy Lin

xl5224

COURSE COORDINATOR

Robert A. van de Geijn

UNIQUE NUMBER

53180

RELEASE DATE

Oct. 2014

DUE DATE

Oct. 2014

TIME SPENT

10 hours

October 13, 2014

Exercises

Exercise 6.

Show that

$$\left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)^{-1} = \left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) \tag{1}$$

Proof. To show the inverse relation between two matrices, we multiply two matrices

$$\left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} + l_{21} & I \end{array}\right) = I \tag{2}$$

Then it is proved that

$$\left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)^{-1} = \left(\begin{array}{c|c|c} I_k & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) \tag{3}$$

□

Exercise 7.

Assume that $\tilde{L}_{k-1} = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right)$, show that $\tilde{L}_k = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & I \end{array}\right)$.

Proof. It is known that

$$L_k = \hat{L}_k^{-1} = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)^{-1} = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) \tag{4}$$

Then we have

$$\tilde{L}_k = \tilde{L}_{k-1} L_k = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right) \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & I \end{array}\right) \tag{5}$$

which tells us that $\tilde{L}_k$ can be computed by simply placing computed vector l_{21} below the diagonal of a unit lower triangular matrix. □

Exercise 12.

12.1 Variant 1: Overwriting A and L

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Homework 06: LU with Partial Pivoting (overwrite L and A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Copyright 2014 The University of Texas at Austin
%
% For licensing information see
%                 http://www.cs.utexas.edu/users/flame/license.html
%
% Programmed by: Jimmy Lin
%                 jimmylin@utexas.edu
function [ A_out, L_out, p_out ] = LU_pp_unb_var1( A, L, p )
  [ ATL, ATR, ...
    ABL, ABR ] = FLA_Part_2x2( A, ...
                               0, 0, 'FLA_TL' );
  [ LTL, LTR, ...
    LBL, LBR ] = FLA_Part_2x2( L, ...
                               0, 0, 'FLA_TL' );
  [ pT, ...
    pB ] = FLA_Part_2x1( p, ...
                         0, 'FLA_TOP' );
  while ( size( ATL, 1 ) < size( A, 1 ) )
    [ A00,  a01,     A02,  ...
      a10t, alpha11, a12t, ...
      A20,  a21,     A22 ] = FLA_Repart_2x2_to_3x3( ATL, ATR, ...
                                                    ABL, ABR, ...
                                                    1, 1, 'FLA_BR' );
    [ L00,  l01,      L02,  ...
      l10t, lambda11, l12t, ...
      L20,  l21,      L22 ] = FLA_Repart_2x2_to_3x3( LTL, LTR, ...
                                                     LBL, LBR, ...
                                                     1, 1, 'FLA_BR' );
    [ p0, ...
      pi1, ...
      p2 ] = FLA_Repart_2x1_to_3x1( pT, ...
                                    pB, ...
                                    1, 'FLA_BOTTOM' );
    %------------------------------------------------------------%
    [ ~ , pi1 ] = max ([alpha11; a21]);
    if pi1 ~= 1,
        P = eye(size(A22)+[1 1]);
        P(1, 1) = 0; P(1, pi1) = 1;
        P(pi1, pi1) = 0; P(pi1, 1) = 1;
        new = P * [l10t, alpha11, a12t; L20, a21, A22];
        sep_col = size(ATL, 2);
        l10t = new(1, 1:sep_col);
        alpha11 = new(1, sep_col+1);
        a12t = new(1, (sep_col+2):end);
        L20 = new(2:end, 1:sep_col);
        a21 = new(2:end, sep_col+1);
        A22 = new(2:end, (sep_col+2):end);
    end
    l21 = a21 / alpha11;
    A22 = A22 - l21 * a12t;
    a21(:) = 0; lambda11 = 1;
    %------------------------------------------------------------%
    [ ATL, ATR, ...
      ABL, ABR ] = FLA_Cont_with_3x3_to_2x2( A00,  a01,     A02,  ...
                                             a10t, alpha11, a12t, ...
                                             A20,  a21,     A22,  ...
                                             'FLA_TL' );
    [ LTL, LTR, ...
```

```
      LBL, LBR ] = FLA_Cont_with_3x3_to_2x2( L00, zeros(size(l01)), zeros(size(L02)),  ...
                                             l10t, lambda11, zeros(size(l12t)), ...
                                             L20,  l21,      L22, ...
                                             'FLA_TL' );
    [ pT, ...
      pB ] = FLA_Cont_with_3x1_to_2x1( p0, ...
                                       pi1, ...
                                       p2, ...
                                       'FLA_TOP' );
  end
  A_out = [ ATL, ATR
            ABL, ABR ];
  L_out = [ LTL, LTR
            LBL, LBR ];
  p_out = [ pT
            pB ];
return
```

12.2 Variant 2: Overwriting A

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Homework 06: LU with Partial Pivoting (Overwrite A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Copyright 2014 The University of Texas at Austin
%
% For licensing information see
%                http://www.cs.utexas.edu/users/flame/license.html
%
% Programmed by: Jimmy Lin
%                jimmylin@utexas.edu
function [ A_out, p_out ] = LU_partial_pivot_unb( A, p )
  [ ATL, ATR, ...
    ABL, ABR ] = FLA_Part_2x2( A, ...
                               0, 0, 'FLA_TL' );
  [ pT, ...
    pB ] = FLA_Part_2x1( p, ...
                         0, 'FLA_TOP' );
  while ( size( ATL, 1 ) < size( A, 1 ) )
    [ A00,  a01,     A02,  ...
      a10t, alpha11, a12t, ...
      A20,  a21,     A22 ] = FLA_Repart_2x2_to_3x3( ATL, ATR, ...
                                                    ABL, ABR, ...
                                                    1, 1, 'FLA_BR' );
    [ p0, ...
      pi1, ...
      p2 ] = FLA_Repart_2x1_to_3x1( pT, ...
                                    pB, ...
                                    1, 'FLA_BOTTOM' );
    %------------------------------------------------------------%
    [ ~ , pi1 ] = max ([alpha11; a21]);
    if pi1 ~= 1,
        P = eye(size(A22)+[1 1]);
        P(1, 1) = 0; P(1, pi1) = 1;
        P(pi1, pi1) = 0; P(pi1, 1) = 1;
        new = P * [a10t, alpha11, a12t; A20, a21, A22];
        sep_col = size(ATL, 2);
        a10t = new(1, 1:sep_col);
        alpha11 = new(1, sep_col+1);
        a12t = new(1, (sep_col+2):end);
        A20 = new(2:end, 1:sep_col);
        a21 = new(2:end, sep_col+1);
        A22 = new(2:end, (sep_col+2):end);
    end
    a21 = a21 / alpha11;
    A22 = A22 - a21 * a12t;
    %------------------------------------------------------------%
```

```
    [ ATL, ATR, ...
      ABL, ABR ] = FLA_Cont_with_3x3_to_2x2( A00,  a01,     A02,  ...
                                             a10t, alpha11, a12t, ...
                                             A20,  a21,     A22, ...
                                             'FLA_TL' );
    [ pT, ...
      pB ] = FLA_Cont_with_3x1_to_2x1( p0, ...
                                       pi1, ...
                                       p2, ...
                                       'FLA_TOP' );
  end
  A_out = [ ATL, ATR
            ABL, ABR ];
  p_out = [ pT
            pB ];
return
```

Exercise 13.

Derivation Partition $U = \left(\begin{array}{c|c} v_{11} & u_{12}^T \\ 0 & U_{22} \end{array}\right)$, $x = \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right)$ and $y = \left(\begin{array}{c} \psi_1 \\ \hline y_2 \end{array}\right)$. Then

$$Ux = U = \left(\begin{array}{c|c} v_{11} & u_{12}^T \\ 0 & U_{22} \end{array}\right)\left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right) = \left(\begin{array}{c} v_{11}x_1 + u_{12}^T x_2 \\ \hline U_{22}x_2 \end{array}\right) = \left(\begin{array}{c} \psi_1 \\ \hline y_2 \end{array}\right) \tag{6}$$

Then the update is given by

$$\chi_1 = \frac{1}{v_{11}}(\psi_1 - u_{12}^T x_2) \tag{7}$$

Implementation

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Homework 06: Upper Triangular System Solver with DOT PRODUCT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Copyright 2014 The University of Texas at Austin
%
% For licensing information see
%                http://www.cs.utexas.edu/users/flame/license.html
%
% Programmed by: Jimmy Lin
%                jimmylin@utexas.edu
function [ U_out, y_out ] = UTS_Solver_unb_var1( U, y )
  [ UTL, UTR, ...
    UBL, UBR ] = FLA_Part_2x2( U, ...
                               0, 0, 'FLA_BR' );
  [ yT, ...
    yB ] = FLA_Part_2x1( y, ...
                         0, 'FLA_BOTTOM' );
  while ( size( UBR, 1 ) < size( U, 1 ) )
    [ U00,  u01,       U02,  ...
      u10t, upsilon11, u12t, ...
      U20,  u21,       U22 ] = FLA_Repart_2x2_to_3x3( UTL, UTR, ...
                                                      UBL, UBR, ...
                                                      1, 1, 'FLA_TL' );
    [ y0, ...
      psi1, ...
      y2 ] = FLA_Repart_2x1_to_3x1( yT, ...
                                    yB, ...
                                    1, 'FLA_TOP' );
    %------------------------------------------------------------%
    psi1 = (1.0 / upsilon11) * (psi1 - u12t * y2);
    %------------------------------------------------------------%
    [ UTL, UTR, ...
      UBL, UBR ] = FLA_Cont_with_3x3_to_2x2( U00,  u01,       U02,  ...
                                             u10t, upsilon11, u12t, ...
                                             U20,  u21,       U22, ...
                                             'FLA_BR' );
    [ yT, ...
      yB ] = FLA_Cont_with_3x1_to_2x1( y0, ...
                                       psi1, ...
                                       y2, ...
                                       'FLA_BOTTOM' );
  end
  U_out = [ UTL, UTR
            UBL, UBR ];
  y_out = [ yT
            yB ];
return
```

Exercise 14.

Derivation. Partition $U = \left(\begin{array}{c|c} U_{11} & u_{12} \\ \hline 0 & \upsilon_{22} \end{array}\right)$, $x = \left(\begin{array}{c} x_1 \\ \hline \chi_2 \end{array}\right)$ and $y = \left(\begin{array}{c} y_1 \\ \hline \psi_2 \end{array}\right)$. Then

$$Ux = \left(\begin{array}{c|c} U_{11} & u_{12} \\ \hline 0 & \upsilon_{22} \end{array}\right) \left(\begin{array}{c} x_1 \\ \hline \chi_2 \end{array}\right) = \left(\begin{array}{c} U_{11}x_1 + \chi_2 u_{12} \\ \hline \chi_2 \upsilon_{22} \end{array}\right) = \left(\begin{array}{c} y_1 \\ \hline \psi_2 \end{array}\right) \tag{8}$$

Then the update is given by

$$\chi_2 = \frac{\psi_2}{\upsilon_{22}} \tag{9}$$

$$U_{11}x_1 = y_1 - \chi_2 u_{12} = y_1 - \frac{\psi_2}{\upsilon_{22}} u_{12} \tag{10}$$

Implementation.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Homework 06: Upper Triangular System Solver with AXPY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Copyright 2014 The University of Texas at Austin
%
% For licensing information see
%                 http://www.cs.utexas.edu/users/flame/license.html
%
% Programmed by: Jimmy Lin
%                 jimmylin@utexas.edu
function [ U_out, y_out ] = UTS_Solver_unb_var2( U, y )
  [ UTL, UTR, ...
    UBL, UBR ] = FLA_Part_2x2( U, ...
                               0, 0, 'FLA_BR' );
  [ yT, ...
    yB ] = FLA_Part_2x1( y, ...
                         0, 'FLA_BOTTOM' );
  while ( size( UBR, 1 ) < size( U, 1 ) )
    [ U00,  u01,       U02,  ...
      u10t, upsilon11, u12t, ...
      U20,  u21,       U22 ] = FLA_Repart_2x2_to_3x3( UTL, UTR, ...
                                                      UBL, UBR, ...
                                                      1, 1, 'FLA_TL' );
    [ y0, ...
      psi1, ...
      y2 ] = FLA_Repart_2x1_to_3x1( yT, ...
                                    yB, ...
                                    1, 'FLA_TOP' );
    %------------------------------------------------------------%
    y0 = y0 - (psi1 / upsilon11) * u01;
    psi1 = (psi1 / upsilon11);
    %------------------------------------------------------------%
    [ UTL, UTR, ...
      UBL, UBR ] = FLA_Cont_with_3x3_to_2x2( U00,  u01,       U02,  ...
                                             u10t, upsilon11, u12t, ...
                                             U20,  u21,       U22,  ...
                                             'FLA_BR' );
    [ yT, ...
      yB ] = FLA_Cont_with_3x1_to_2x1( y0, ...
                                       psi1, ...
                                       y2, ...
                                       'FLA_BOTTOM' );
  end
  U_out = [ UTL, UTR
            UBL, UBR ];
  y_out = [ yT
            yB ];
return
```