Unit 1.2 Special Matrices

Numerical Analysis

EE/NTHU

Mar. 1, 2017

Cholesky Decomposition

- A special case of the matrix factorization
- Matrix $\mathbf{A}$ needs to be symmetric and positive definite

$$\mathbf{A} = \mathbf{A}^T \tag{1.2.1}$$

$$\mathbf{x}^T\mathbf{A}\mathbf{x} > 0, \text{ for any } \mathbf{x} \neq \mathbf{0}. \tag{1.2.2}$$

- Then we can factorize $\mathbf{A}$ to

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T. \tag{1.2.3}$$

  where $\mathbf{L}$ is a lower triangular matrix, i.e., $\ell_{ij} = 0$, if $j > i$.
- For example

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \begin{bmatrix} \ell_{11} & \ell_{21} & \ell_{31} \\ 0 & \ell_{22} & \ell_{32} \\ 0 & 0 & \ell_{33} \end{bmatrix}$$

Cholesky Decomposition – Example

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \begin{bmatrix} \ell_{11} & \ell_{21} & \ell_{31} \\ 0 & \ell_{22} & \ell_{32} \\ 0 & 0 & \ell_{33} \end{bmatrix}$$

$$\begin{aligned} a_{11} &= \ell_{11}\ell_{11} & \qquad \ell_{11} &= \sqrt{a_{11}} \\ a_{21} &= \ell_{11}\ell_{21} & \ell_{21} &= a_{21}/\ell_{11} \\ a_{31} &= \ell_{11}\ell_{31} & \ell_{31} &= a_{31}/\ell_{11} \\ a_{22} &= \ell_{22}\ell_{22} + \ell_{21}\ell_{21} & \ell_{22} &= \sqrt{a_{22} - \ell_{21}\ell_{21}} \\ a_{32} &= \ell_{31}\ell_{21} + \ell_{22}\ell_{32} & \ell_{32} &= (a_{32} - \ell_{31}\ell_{21})/\ell_{22} \\ a_{33} &= \ell_{31}\ell_{31} + \ell_{32}\ell_{32} + \ell_{33}\ell_{33} & \ell_{33} &= \sqrt{a_{33} - \ell_{31}\ell_{31} - \ell_{32}\ell_{32}} \end{aligned}$$

- Note that the number of variables in Cholesky decomposition is smaller than LU decomposition
    - Cholesky is 2X faster than LU decomposition

Cholesky Decomposition – Properties

Theorem 1.2.1.

For any positive definite (symmetric) matrix $\mathbf{A}$ the matrix $\mathbf{A}^{-1}$ exists and is also positive definite. All principal submatrices of a positive definite matrix is also positive definite, and all principal minors of a positive definite matrix are positive.

- If $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$, thus $\mathbf{A}\mathbf{x} \neq \mathbf{0}$, and $\mathbf{A}^{-1}$ exists.
- Let $\mathbf{y} = \mathbf{A}\mathbf{x}$ then

$$\mathbf{y}^T\mathbf{A}^{-1}\mathbf{y} = \mathbf{x}^T\mathbf{A}\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{x}^T\mathbf{A}\mathbf{x} > 0.$$

- Partition $\mathbf{x}$ to $\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}$, where $\mathbf{x}_2$ corresponds to the principal submatrix and $\mathbf{x}_1$ is not. Then since $\mathbf{x}^T\mathbf{A}\mathbf{x} > \mathbf{0}$, $\mathbf{y} = \begin{bmatrix}\mathbf{0} & \mathbf{x}_2\end{bmatrix}$ we also have $\mathbf{y}^{-1}\mathbf{A}\mathbf{y} > \mathbf{0}$.

Theorem 1.2.2.

For any positive definite symmetric matrix $\mathbf{A}$ there is a unique lower triangular matrix $\mathbf{L}$, $\ell_{ij} = 0$ if $j > i$, with $\ell_{ii} > 0$, such that $\mathbf{A} = \mathbf{L}\mathbf{L}^T$.

- Note that this theorem also holds if $\mathbf{A}$ is Hermitian, $\mathbf{A} = \mathbf{A}^H$ then $\mathbf{A} = \mathbf{L}\mathbf{L}^H$.

Cholesky Decomposition – Algorithm

- In-place Cholesky decomposition
  - Only lower triangle of $\mathbf{A}$ is affected
  - Forward and backward substitutions need to modified accordingly

Algorithm 1.2.3. Cholesky Decomposition

```
void Cholesky(double A[n][n])
{
   int i,j,k;

   for (i=0; i<n; i++) {
      A[i][i]=sqrt(A[i][i]);
      for (j=i+1; j<n; j++) {
         A[j][i] /= A[i][i];
      }
      for (j=i+1; j<n; j++) {
         for (k=i+1; k<=j; k++)
            A[j][k] -= A[j][i]*A[k][i];
      }
   }
}
```


$\mathbf{LDL}^T$ Decomposition

- For symmetric positive definite matrices it is also possible to decompose matrix $\mathbf{A}$ into the following form:

$$\mathbf{A} = \mathbf{LDL}^T. \tag{1.2.4}$$

Where $\mathbf{L}$ is a lower triangle matrix with 1's on the diagonal and $\mathbf{D}$ is a diagonal matrix.

- For example

$$\begin{aligned}
\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ \ell_{21} & 1 & 0 \\ \ell_{31} & \ell_{32} & 1 \end{bmatrix} \begin{bmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{bmatrix} \begin{bmatrix} 1 & \ell_{21} & \ell_{31} \\ 0 & 1 & \ell_{32} \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} d_{11} & 0 & 0 \\ d_{11}\ell_{21} & d_{22} & 0 \\ d_{11}\ell_{31} & d_{22}\ell_{32} & d_{33} \end{bmatrix} \begin{bmatrix} 1 & \ell_{21} & \ell_{31} \\ 0 & 1 & \ell_{32} \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} d_{11} & d_{11}\ell_{21} & d_{11}\ell_{31} \\ d_{11}\ell_{21} & d_{22} + d_{11}\ell_{21}^2 & \ell_{21}d_{11}\ell_{31} + d_{22}\ell_{32} \\ d_{11}\ell_{31} & \ell_{31}d_{11}\ell_{21} + d_{22}\ell_{32} & d_{11}\ell_{31}^2 + d_{22}\ell_{32}^2 + d_{33} \end{bmatrix}
\end{aligned}$$

LDL^T Decomposition, II

- Decomposition process is similar.
  - In-place decomposition feasible.

Requirements:

$$
\begin{aligned}
a_{11} &= d_{11} \\
a_{21} &= d_{11}\ell_{21} \\
a_{31} &= d_{11}\ell_{31} \\
a_{22} &= d_{22} + d_{11}\ell_{21}^2 \\
a_{32} &= \ell_{31}d_{11}\ell_{21} + d_{22}\ell_{32} \\
a_{33} &= d_{33} + d_{11}\ell_{31}^2 + d_{22}\ell_{32}^2
\end{aligned}
$$

Solving unknowns:

$$
\begin{aligned}
d_{11} &= a_{11} \\
\ell_{21} &= a_{21}/d_{11} \\
\ell_{31} &= a_{31}/d_{11} \\
d_{22} &= a_{22} - d_{11}\ell_{21}^2 \\
\ell_{32} &= (a_{32} - \ell_{31}d_{11}\ell_{21})/d_{22} \\
d_{33} &= a_{33} - d_{11}\ell_{31}^2 - d_{22}\ell_{32}^2
\end{aligned}
$$

Two-step process:

$$
\begin{aligned}
d_{11} &= a_{11} \\
\ell_{21} &= a_{21}/d_{11} \\
\ell_{31} &= a_{31}/d_{11} \\
a'_{22} &= a_{22} - d_{11}\ell_{21}^2 \\
a'_{32} &= a_{32} - d_{11}\ell_{31}\ell_{21} \\
a'_{33} &= a_{33} - d_{11}\ell_{31}^2 \\
\\
d_{22} &= a'_{22} \\
\ell_{32} &= a'_{32}/d_{22} \\
a''_{33} &= a'_{33} - d_{22}\ell_{32}^2 \\
\\
d_{33} &= a''_{33}
\end{aligned}
$$

Banded Matrices

- In some applications matrix $\mathbf{A}$ is banded

$$a_{ij} = 0, \qquad \text{if } |i-j| > m. \tag{1.2.5}$$

- For example, $m = 1$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix}. \tag{1.2.6}$$

- 5×5 tridiagonal matrix is shown.
  - For large $n \times n$ matrices, the elements are mostly zeros
  - Even for $m > 1$ cases
- We can exploit this property for more efficient solution for both CPU time and matrix storage.

Banded Matrices, II

- The matrix shown in Eq. (1.2.6) can be stored using a 5×3 array as

$$\mathbf{A}^T = \begin{bmatrix} 0 & a_{21} & a_{32} & a_{43} & a_{54} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} \\ a_{12} & a_{23} & a_{34} & a_{45} & 0 \end{bmatrix}. \tag{1.2.7}$$

  - The storage space reduced to 5×3 entries.
- In general, for an m-banded matrix, the storage requirement reduces from $n \times n$ to $n \times m$.
  - When $n \gg m$, the saving is very significant.
- The LU factors of a banded matrix is also banded. For example,

$$\begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \ell_{21} & 1 & 0 & 0 & 0 \\ 0 & \ell_{32} & 1 & 0 & 0 \\ 0 & 0 & \ell_{43} & 1 & 0 \\ 0 & 0 & 0 & \ell_{54} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ 0 & u_{22} & u_{23} & 0 & 0 \\ 0 & 0 & u_{33} & u_{34} & 0 \\ 0 & 0 & 0 & u_{44} & u_{45} \\ 0 & 0 & 0 & 0 & u_{55} \end{bmatrix}. \tag{1.2.8}$$

Banded Matrices, III

- The standard LU decomposition algorithm can be applied to the banded matrices as well.
- Applying in-place LU decomposition to the 5×5 tridiagonal matrix we get the following sequence.

$$\begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix} \quad \begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ \ell_{21} & u_{22} & u_{23} & 0 & 0 \\ 0 & \ell_{32} & u_{33} & u_{34} & 0 \\ 0 & 0 & \ell_{43} & a'_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix}$$

$$\begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ \ell_{21} & a'_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix} \quad \begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ \ell_{21} & u_{22} & u_{23} & 0 & 0 \\ 0 & \ell_{32} & u_{33} & u_{34} & 0 \\ 0 & 0 & \ell_{43} & u_{44} & u_{45} \\ 0 & 0 & 0 & \ell_{54} & a'_{55} \end{bmatrix}$$

$$\begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ \ell_{21} & u_{22} & u_{23} & 0 & 0 \\ 0 & \ell_{32} & a'_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix} \quad \begin{bmatrix} u_{11} & u_{12} & 0 & 0 & 0 \\ \ell_{21} & u_{22} & u_{23} & 0 & 0 \\ 0 & \ell_{32} & u_{33} & u_{34} & 0 \\ 0 & 0 & \ell_{43} & u_{44} & u_{45} \\ 0 & 0 & 0 & \ell_{54} & u_{55} \end{bmatrix}$$

Banded Matrices, IV

- Note that the LU decomposition does not change the band structure of the matrix
  - This is a property of the LU decomposition
  - Copying the current row to form U-row does not change any zero entry to nonzero
  - Making L-column does not add any nonzero entry, either
  - Updating submatrix - only update within the band
- Taking advantage of the band structure, LU factorization can be more efficient
  - Computation complexity $\mathcal{O}(m^2 n)$
  - Compared to $\mathcal{O}(n^3)$
  - If $m \ll n$, the saving can be very significant
  - A large number of real world applications can apply this method

Banded Matrices, V

Algorithm 1.2.4. LU Decomposition - Banded Matrix

```
void LUband(double A[n][n],int m)
{
   int i,j,k;
   for (i=0; i<n; i++) {
      // copy a[i][j] to u[i][j] needs no action due to in-place LU
      for (j=i+1; j<=i+m && j<n; j++) { // form l[j][i]
         a[j][i] /= a[i][i];
      }
      for (j=i+1; j<=i+m && j<n; j++) { // update lower submatrix
         for (k=i+1; k<=i+m && k<n; k++) {
            a[j][k] -= a[j][i]*a[i][k];
         }
      }
   }
}
```

- Forward and backward substitutions need to be modified accordingly
  - Lower computational complexity, $\mathcal{O}(n)$

Block Matrices

- The LU factorization of a matrix $\mathbf{A}$ is effective in solving the linear system using forward and backward substitutions

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1.2.9}$$
$$\mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{b} \tag{1.2.10}$$
$$\mathbf{L}\mathbf{y} = \mathbf{b} \tag{1.2.11}$$
$$\mathbf{U}\mathbf{x} = \mathbf{y} \tag{1.2.12}$$

where $\mathbf{y} = \mathbf{L}^{-1}\mathbf{b}$ is the forward substitution, and $\mathbf{x} = \mathbf{U}^{-1}\mathbf{y}$ is the backward substitution.

- In some applications, the matrix $\mathbf{A}$ is partitioned into blocks of submatrices. Example below shows $n \times n$ matrix $\mathbf{A}$ is partitioned into four submatrices

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \tag{1.2.13}$$

where $\mathbf{A}_{11}$ is an $n_1 \times n_1$ matrix; $\mathbf{A}_{22}$ is an $n_2 \times n_2$ matrix; $\mathbf{A}_{12}$ is $n_1 \times n_2$ and $\mathbf{A}_{21}$ is $n_2 \times n_1$, with $n_1 + n_2 = n$.

Block LU Factorization

- LU factorization can also be carried out on the block matrix $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{0} & \mathbf{U}_{22} \end{bmatrix} \tag{1.2.14}$$

where $\mathbf{L}_{11}$ and $\mathbf{L}_{22}$ are lower triangular matrices with dimension of $n_1 \times n_1$ and $n_2 \times n_2$, respectively; $\mathbf{U}_{11}$ and $\mathbf{U}_{22}$ are upper triangular matrices with similar dimensions. $\mathbf{L}_{21}$ and $\mathbf{U}_{12}$ are not necessarily triangular matrices. Then

$$\mathbf{A}_{11} = \mathbf{L}_{11}\mathbf{U}_{11} \tag{1.2.15}$$
$$\mathbf{A}_{12} = \mathbf{L}_{11}\mathbf{U}_{12} \tag{1.2.16}$$
$$\mathbf{A}_{21} = \mathbf{L}_{21}\mathbf{U}_{11} \tag{1.2.17}$$
$$\mathbf{A}_{22} = \mathbf{L}_{21}\mathbf{U}_{12} + \mathbf{L}_{22}\mathbf{U}_{22} \tag{1.2.18}$$

- Thus, $\mathbf{L}_{11}$ and $\mathbf{U}_{11}$ are found by LU factorizing $\mathbf{A}_{11}$.
- $\mathbf{U}_{12}$ and $\mathbf{L}_{21}$ are obtained by extending the row-copying and column-normalization operations while performing LU factorization on $\mathbf{A}_{11}$.
- $\mathbf{L}_{22}$ and $\mathbf{U}_{22}$ are the LU factors of the updated $\mathbf{A}_{22}$ submatrix

$$\mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{U}_{12} = \mathbf{L}_{22}\mathbf{U}_{22}. \tag{1.2.19}$$

Block Matrix Example

- Example of a block matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} = \left[\begin{array}{ccc|cc} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ \hline a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{array}\right]$$

$$\mathbf{A}_{11} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, \mathbf{A}_{12} = \begin{bmatrix} a_{14} & a_{15} \\ a_{24} & a_{25} \\ a_{34} & a_{35} \end{bmatrix},$$

$$\mathbf{A}_{21} = \begin{bmatrix} a_{41} & a_{42} & a_{43} \\ a_{51} & a_{52} & a_{53} \end{bmatrix}, \mathbf{A}_{22} = \begin{bmatrix} a_{44} & a_{45} \\ a_{54} & a_{55} \end{bmatrix}.$$

- Partitioning of matrices may help in operation or storage efficiency improvements.
- Many practical linear systems have special matrix forms and those matrices can be exploited using different partitions.

Block LU Factorization - Bordered Block Diagonal Form

- The same idea can be extended to bordered-block-diagonal form matrices

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_{1m} \\ \mathbf{0} & \mathbf{A}_{22} & \mathbf{0} & \cdots & \mathbf{A}_{2m} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \cdots & \mathbf{A}_{3m} \\ \cdots & \cdots & \cdots & \ddots & \vdots \\ \mathbf{A}_{m1} & \mathbf{A}_{m2} & \mathbf{A}_{m3} & \cdots & \mathbf{A}_{mm} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{22} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{L}_{33} & \cdots & \mathbf{0} \\ \cdots & \cdots & \cdots & \ddots & \vdots \\ \mathbf{L}_{m1} & \mathbf{L}_{m2} & \mathbf{L}_{m3} & \cdots & \mathbf{L}_{mm} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{11} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{U}_{1m} \\ \mathbf{0} & \mathbf{U}_{22} & \mathbf{0} & \cdots & \mathbf{U}_{2m} \\ \mathbf{0} & \mathbf{0} & \mathbf{U}_{33} & \cdots & \mathbf{U}_{3m} \\ \cdots & \cdots & \cdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{U}_{mm} \end{bmatrix}$$

- $\mathbf{L}_{ii}$ and $\mathbf{U}_{ii}$ are the LU factors of submatrices $\mathbf{A}_{ii}$, $1 \le i \le m-1$, and $\mathbf{L}_{mm}$ and $\mathbf{U}_{mm}$ are the LU factors of the updated submatrix:

$$\mathbf{A}_{mm} - \sum_{i=1}^{m-1} \mathbf{L}_{mi}\mathbf{U}_{im} = \mathbf{L}_{mm}\mathbf{U}_{mm}. \tag{1.2.20}$$

BBDF, II

- Thus, one can organize the matrix $\mathbf{A}$ as m matrices

$$\mathbf{A}^{(1)} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{1m} \\ \mathbf{A}_{m1} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{A}^{(2)} = \begin{bmatrix} \mathbf{A}_{22} & \mathbf{A}_{2m} \\ \mathbf{A}_{m2} & \mathbf{0} \end{bmatrix}$$

$$\cdots$$

$$\mathbf{A}^{(m-1)} = \begin{bmatrix} \mathbf{A}_{(m-1)(m-1)} & \mathbf{A}_{(m-1)m} \\ \mathbf{A}_{m(m-1)} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{A}^{(m)} = [\mathbf{A}_{mm}]$$

- The LU decomposition of the submatrices $\mathbf{A}^{(k)}$, $k = 1, \ldots, m-1$, can be done in parallel, and the LU decomposition is carried out on the diagonal blocks, $\mathbf{A}_{kk}$, only.
- The LU decomposition of the last block $\mathbf{A}_{mm}$, however, can only be performed after all other submatrices have been decomposed and it needs to be performed using Eq. (1.2.20).
- Note that each submatrix needs not to have the same dimension.

Sparse Matrices

- In many real applications, the matrix $\mathbf{A}$ is sparse, that is, it has many zero entries.
  - The bordered block diagonal matrix in preceding page is a good example of sparse matrix
- During LU decomposition (and forward and backward substitutions) any operations associated with the zero entries should not be performed at all for better efficiency
  - Even check for zero would take time
  - These entries should not be stored also
- Linked list that stores only nonzero entries should be used
- Need double linked list
  - Two next pointers: one for next in row and one for next in column
- Using double linked list for sparse matrix storage and operation has been shown to speed up linear system solution time
  - Full matrix solution time $\mathcal{O}(n^3)$
  - Sparse matrix solution time $\mathcal{O}(n^{1.1-1.5})$
  - Very significant saving when n is large

Summary

- Cholesky decomposition
  - Much faster for symmetric and positive definite matrices
- Banded matrices
  - Even faster due to large number of zero entries, $\mathcal{O}(m^2 n)$.
- Block matrices
  - Achieving parallel computations.
- Sparse matrices
  - Need linked list data structure.
  - Have been shown to have excellent efficiency, $\mathcal{O}(n^{1.1-1.5})$.