Chapter 2

Numerical methods for solving linear systems

Let $A \in \mathbb{C}^{n \times n}$ be a nonsingular matrix. We want to solve the linear system $Ax = b$ by (a) Direct methods (finite steps); Iterative methods (convergence). (See Chapter 4)

2.1 Elementary matrices

Let $X = \mathbb{K}^n$ and $x, y \in X$. Then $y^* x \in \mathbb{K}$, $xy^* = \begin{pmatrix} x_1 y_1 \cdots x_1 y_n \\ \vdots \\ x_n y_1 \cdots x_n y_n \end{pmatrix}$. The eigenvalues of $xy^*$ are $\{0, \cdots, 0, y^*x\}$, since rank($xy^*$) = 1 by $(xy^*)z = (y^*z)x$ and $(xy^*)x = (y^*x)x$.

**Definition 2.1.1** A matrix of the form

$$I - \alpha xy^* \quad (\alpha \in \mathbb{K}, x, y \in \mathbb{K}^n)$$

is called an elementary matrix.

The eigenvalues of $(I - \alpha xy^*)$ are $\{1, 1, \cdots, 1, 1 - \alpha y^*x\}$. Compute

$$(I - \alpha xy^*)(I - \beta xy^*) = I - (\alpha + \beta - \alpha \beta y^*)x y^*.$$  

(2.1.2)

If $\alpha y^*x - 1 \neq 0$ and let $\beta = \frac{\alpha}{\alpha y^*x - 1}$, then $\alpha + \beta - \alpha \beta y^*x = 0$. We have

$$(I - \alpha xy^*)^{-1} = (I - \beta xy^*), \quad \frac{1}{\alpha} + \frac{1}{\beta} = y^*x.$$  

(2.1.3)

**Example 2.1.1** Let $x \in \mathbb{K}^n$, and $x^* x = 1$. Let $H = \{z : z^* x = 0\}$ and

$$Q = I - 2xx^* \quad (Q = Q^*, Q^{-1} = Q).$$

Then $Q$ reflects each vector with respect to the hyperplane $H$. Let $y = \alpha x + w, w \in H$. Then, we have

$$Qy = \alpha Qx + Qw = -\alpha x + w - 2(x^*w)x = -\alpha x + w.$$
Example 2.1.2 Let \( y = e_i \) be the \( i \)-th column of unit matrix and \( x = l_i = [0, \ldots, 0, l_{i+1,i}, \ldots, l_{n,i}]^T \). Then,

\[
I + l_i e_i^T = \begin{bmatrix}
1 \\
\vdots \\
l_{i+1,i} \\
\vdots \\
l_{n,i} & 1
\end{bmatrix}
\]  

(2.1.4)

Since \( e_i^T l_i = 0 \), we have

\[
(I + l_i e_i^T)^{-1} = (I - l_i e_i^T).
\]

(2.1.5)

From the equality

\[
(I + l_1 e_1^T)(I + l_2 e_2^T) = I + l_1 e_1^T + l_2 e_2^T + l_1 (e_1^T l_2) e_2^T = I + l_1 e_1^T + l_2 e_2^T
\]

follows that

\[
(I + l_1 e_1^T) \cdots (I + l_i e_i^T) \cdots (I + l_{n-1} e_{n-1}^T) = I + l_1 e_1^T + l_2 e_2^T + \cdots + l_{n-1} e_{n-1}^T
\]

(2.1.6)

Theorem 2.1.1 A lower triangular with “1” on the diagonal can be written as the product of \( n - 1 \) elementary matrices of the form (2.1.4).

Remark 2.1.1 \((I + l_1 e_1^T + \ldots + l_{n-1} e_{n-1}^T)^{-1} = (l - l_{n-1} e_{n-1}^T) \cdots (I - l_1 e_1^T)\) which can not be simplified as in (2.1.6).

### 2.2 LR-factorization

Definition 2.2.1 Given \( A \in \mathbb{C}^{n \times n} \), a lower triangular matrix \( L \) and an upper triangular matrix \( R \). If \( A = LR \), then the product \( LR \) is called a LR-factorization (or LR-decomposition) of \( A \).

Basic problem:

Given \( b \neq 0, b \in \mathbb{K}^n \). Find a vector \( l_1 = [0, l_{21}, \ldots, l_{n1}]^T \) and \( c \in \mathbb{K} \) such that

\[
(I - l_1 e_1^T)b = ce_1.
\]

Solution:

\[
\begin{cases}
  b_1 = c, \\
  b_i - l_{i1}b_1 = 0, & i = 2, \ldots, n. \\
  b_1 = 0, & \text{it has no solution (since } b \neq 0), \\
  b_1 \neq 0, & \text{then } c = b_1, \ l_{i1} = b_i/b_1, \ i = 2, \ldots, n.
\end{cases}
\]
Construction of LR-factorization:
Let \( A = A^{(0)} = [a_1^{(0)} | \ldots | a_n^{(0)}] \). Apply basic problem to \( a_1^{(0)} \): If \( a_{11}^{(0)} \neq 0 \), then there exists \( L_1 = I - l_1 e_1^T \) such that \((I - l_1 e_1^T) a_1^{(0)} = a_{11}^{(0)} e_1\). Thus
\[
A^{(1)} = L_1 A^{(0)} = [L_1 a_1^{(0)} | \ldots | L_1 a_n^{(0)}] = \begin{bmatrix}
  a_{11}^{(0)} & a_{12}^{(0)} & \cdots & a_{1n}^{(0)} \\
  0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & a_{n2}^{(n)} & \cdots & a_{nn}^{(1)}
\end{bmatrix}.
\tag{2.2.1}
\]

The \( i \)-th step:
\[
A^{(i)} = L_i A^{(i-1)} = L_i L_{i-1} \ldots L_1 A^{(0)} = \begin{bmatrix}
  a_{11}^{(0)} & \cdots & \cdots & a_{1n}^{(0)} \\
  0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \cdots & a_{nn}^{(i)}
\end{bmatrix}.
\tag{2.2.2}
\]

If \( a_{ii}^{(i-1)} \neq 0 \), for \( i = 1, \ldots, n - 1 \), then the method is executable and we have that
\[
A^{(n-1)} = L_{n-1} \ldots L_1 A^{(0)} = R
\tag{2.2.3}
\]
is an upper triangular matrix. Thus, \( A = LR \). Explicit representation of \( L \):
\[
L_i = I - l_i e_i^T, \quad L_i^{-1} = I + l_i e_i^T \\
L = L_1^{-1} \ldots L_{n-1}^{-1} = (I + l_1 e_1^T) \ldots (I + l_{n-1} e_{n-1}^T) \\
= I + l_1 e_1^T + \ldots + l_{n-1} e_{n-1}^T \quad \text{(by (2.1.6)).}
\]

**Theorem 2.2.1** Let \( A \) be nonsingular. Then \( A \) has an LR-factorization \((A=LR)\) if and only if \( k_i := \det(A_i) \neq 0 \), where \( A_i \) is the leading principal matrix of \( A \), i.e.,
\[
A_i = \begin{bmatrix}
  a_{11} & \cdots & a_{1i} \\
  \vdots & \ddots & \vdots \\
  a_{i1} & \cdots & a_{ii}
\end{bmatrix},
\]
for \( i = 1, \ldots, n - 1 \).

**Proof:** (Necessity “\( \Rightarrow \)”) Since \( A = LR \), we have
\[
\begin{bmatrix}
  a_{11} & \cdots & a_{1i} \\
  \vdots & \ddots & \vdots \\
  a_{i1} & \cdots & a_{ii}
\end{bmatrix} = \begin{bmatrix}
  l_{11} \\
  \vdots \\
  l_{i1} \quad l_{ii}
\end{bmatrix} \begin{bmatrix}
  r_{11} & r_{1i} \\
  \vdots & \ddots \\
  O & \cdots & r_{ii}
\end{bmatrix}.
\]
Consider the \((j = 1)\) where \(l_{ii} \neq 0\). Hence \(k_i = l_{i1} \ldots l_{ir1} \ldots r_{ii} \neq 0\).

(Sufficiency \(" \Rightarrow\)\): From (2.2.2) we have

\[
A^{(0)} = (L_1^{-1} \ldots L_i^{-1})A^{(i)}.
\]

Consider the \((i + 1)\)-th leading principle determinant. From (2.2.3) we have

\[
\begin{bmatrix}
  a_{11} & \ldots & a_{i,i+1} \\
  \vdots & & \vdots \\
  a_{i+1} & \ldots & a_{i+1,i+1}
\end{bmatrix}
= \begin{bmatrix}
  1 & & 0 \\
  l_{i2} & \ddots & \vdots \\
  \vdots & \ddots & \ddots \\
  l_{i,i+1} & \ldots & l_{i,i+1} 1
\end{bmatrix}
\begin{bmatrix}
  a_{11}^{(0)} & a_{12}^{(0)} & \ldots & \ldots & * \\
  a_{22}^{(1)} & \ddots & \vdots & & \\
  \vdots & \ddots & \ddots & \vdots & \\
  a_{ii}^{(i-1)} & a_{i,i+1}^{(i-1)} & \ldots & 1 & \\
  0 & & \ddots & & \ddots
\end{bmatrix}.
\]

Thus, \(k_i = 1 \cdot a_{11}^{(0)} a_{22}^{(1)} \ldots a_{i+1,i+1} \neq 0\) which implies \(a_{i+1,i+1}^{(i)} \neq 0\). Therefore, the LR-factorization of \(A\) exists.

**Theorem 2.2.2** If a nonsingular matrix \(A\) has an LR-factorization with \(A = LR\) and \(l_{11} = \cdots = l_{nn} = 1\), then the factorization is unique.

**Proof**: Let \(A = L_1R_1 = L_2R_2\). Then \(L_2^{-1}L_1 = R_2R_1^{-1} = I\).

**Corollary 2.2.1** If a nonsingular matrix \(A\) has an LR-factorization with \(A = LDR\), where \(D\) is diagonal, \(L\) and \(R^T\) are unit lower triangular (with one on the diagonal) if and only if \(k_i \neq 0\).

**Theorem 2.2.3** Let \(A\) be a nonsingular matrix. Then there exists a permutation \(P\), such that \(PA\) has an LR-factorization.

(Proof): By construction! Consider (2.2.2): There is a permutation \(P_i\), which interchanges the \(i\)-th row with a row of index large than \(i\), such that \(0 \neq a_{ii}^{(i)}(\in P_iA^{(i-1)})\). This procedure is executable, for \(i = 1, \ldots, n - 1\). So we have

\[
L_{n-1}P_{n-1} \ldots L_iP_i \ldots L_1P_1A^{(0)} = R.
\]  

Let \(P\) be a permutation which affects only elements \(i+1, \ldots, n\). It holds

\[
P(I - l_i e_i^T)P^{-1} = I - (Pl_i)e_i^T = I - \tilde{l}_i e_i^T = \tilde{L}_i, \quad (e_i^TP^{-1} = e_i^T)
\]

where \(\tilde{L}_i\) is lower triangular. Hence we have

\[
P \tilde{L}_i = \tilde{L}_i P.
\]

Now write all \(P_i\) in (2.2.4) to the right as

\[
L_{n-1}\tilde{L}_{n-2} \ldots \tilde{L}_1P_{n-1} \ldots P_1A^{(0)} = R.
\]

Then we have \(PA = LR\) with \(L^{-1} = L_{n-1}\tilde{L}_{n-2} \cdots \tilde{L}_1\) and \(P = P_{n-1} \cdots P_1\).
2.3 Gaussian elimination

2.3.1 Practical implementation

Given a linear system

\[ Ax = b \]  \hspace{1cm} (2.3.1)

with \( A \) nonsingular. We first assume that \( A \) has an LR-factorization, i.e., \( A = LR \). Thus

\[ LRx = b. \]

We then (i) solve \( Ly = b \); (ii) solve \( Rx = y \). These imply that \( LRx = Ly = b \). From (2.2.4), we have

\[ L_{n-1}\ldots L_2L_1(A \mid b) = (R \mid L^{-1}b). \]

Algorithm 2.3.1 (without permutation)

For \( k = 1, \ldots, n - 1, \)

if \( a_{kk} = 0 \) then stop (*);

else \( \omega_j := a_{kj} \ (j = k+1, \ldots, n); \)

for \( i = k+1, \ldots, n, \)

\[ \eta := a_{ik}/a_{kk}, \ a_{ik} := \eta; \]

for \( j = k+1, \ldots, n, \)

\[ a_{ij} := a_{ij} - \eta \omega_j, \ b_j := b_j - \eta b_k. \]

For \( x \): (back substitution!)

\[ x_n = b_n/a_{nn}; \]

for \( i = n-1, n-2, \ldots, 1, \)

\[ x_i = (b_i - \sum_{j=i+1}^{n} a_{ij}x_j)/a_{ii}. \]

Cost of computation (one multiplication + one addition \( \equiv \) one flop):

(i) LR-factorization: \( n^3/3 - n/3 \) flops;

(ii) Computation of \( y \): \( n(n-1)/2 \) flops;

(iii) Computation of \( x \): \( n(n+1)/2 \) flops.

For \( A^{-1} \): \( 4/3n^3 \approx n^3/3 + kn^2 \ (k = n \text{ linear systems}). \)

Pivoting: (a) Partial pivoting; (b) Complete pivoting.

From (2.2.2), we have

\[
A^{(k-1)} = \begin{bmatrix}
    a_{11}^{(0)} & \cdots & \cdots & \cdots & \cdots & a_{1n}^{(0)} \\
    0 & \ddots & & & & \\
    \vdots & & a_{k-1,k-1}^{(k-2)} & \cdots & \cdots & a_{k-1,n}^{(k-2)} \\
    \vdots & & 0 & a_{kk}^{(k-1)} & \cdots & a_{kn}^{(k-1)} \\
    \vdots & & & \ddots & & \\
    0 & \cdots & 0 & a_{nk}^{(k-1)} & \cdots & a_{nn}^{(k-1)}
\end{bmatrix}.
\]
For (a):
\[
\begin{cases}
\text{Find } p \in \{k, \ldots, n\} \text{ such that } & |a_{pk}| = \max_{k \leq i \leq n} |a_{ik}| \quad (r_k = p) \\
\text{swap } a_{kj}, b_k \text{ and } a_{pj}, b_p \text{ respectively, } (j = 1, \ldots, n). 
\end{cases}
\]

Replacing (*) in Algorithm 2.3.1 by (2.3.2), we have a new factorization of \( A \) with partial pivoting, i.e., \( PA = LR \) (by Theorem 2.2.1) and \( |l_{ij}| \leq 1 \) for \( i, j = 1, \ldots, n \). For solving linear system \( Ax = b \), we use
\[
P A x = P b \Rightarrow L(Rx) = P^T b \equiv \bar{b}.
\]
It needs extra \( n(n - 1)/2 \) comparisons.

For (b):
\[
\begin{cases}
\text{Find } p, q \in \{k, \ldots, n\} \text{ such that } & |a_{pq}| \leq \max_{k \leq i, j \leq n} |a_{ij}|, \quad (r_k := p, c_k := q) \\
\text{swap } a_{kj}, b_k \text{ and } a_{pj}, b_p \text{ respectively, } (j = k, \ldots, n), \\
\text{swap } a_{ik} \text{ and } a_{iq}(i = 1, \ldots, n). 
\end{cases}
\]
Replacing (*) in Algorithm 2.3.1 by (2.3.3), we also have a new factorization of \( A \) with complete pivoting, i.e., \( P A P^T x = P b \Rightarrow LR \tilde{x} = \bar{b} \Rightarrow x = \Pi \tilde{x} \). For solving linear system \( Ax = b \), we use
\[
P A P(\Pi^T x) = P b \Rightarrow LR \tilde{x} = \bar{b} \Rightarrow x = \Pi \tilde{x}.
\]
It needs \( n^3/3 \) comparisons.

**Example 2.3.1**  Let \( A = \begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix} \) be in three decimal-digit floating point arithmetic. Then \( \kappa(A) = \|A\|\|A^{-1}\| \approx 4 \). \( A \) is well-conditioned.

- Without pivoting:
  \[
  L = \begin{bmatrix} 1 & 0 \\ fl(1/10^{-4}) & 1 \end{bmatrix}, \quad fl(1/10^{-4}) = 10^4,
  \]
  \[
  R = \begin{bmatrix} 10^{-4} & 1 \\ 0 & fl(1 - 10^4 \cdot 1) \end{bmatrix}, \quad fl(1 - 10^4 \cdot 1) = -10^4.
  \]
  \[
  LR = \begin{bmatrix} 1 & 0 \\ 10^4 & 1 \end{bmatrix} \begin{bmatrix} 10^{-4} & 1 \\ 0 & -10^4 \end{bmatrix} = \begin{bmatrix} 10^{-4} & 1 \\ 1 & 0 \end{bmatrix} \neq \begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix} = A.
  \]
  Here \( a_{22} \) entirely “lost” from computation. It is numerically unstable. Let \( Ax = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \).

  Then \( x \approx \begin{bmatrix} 1 \\ 2 \end{bmatrix} \). But \( Ly = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \) solves \( y_1 = 1 \) and \( y_2 = fl(2 - 10^4 \cdot 1) = -10^4 \), \( R \tilde{x} = y \) solves \( \tilde{x}_2 = fl((-10^4)/(-10^3)) = 1 \), \( \tilde{x}_1 = fl((1 - 1)/10^{-4}) = 0 \). We have an erroneous solution with \( \text{cond}(L), \text{cond}(R) \approx 10^8 \).

- Partial pivoting:
  \[
  L = \begin{bmatrix} 1 & 0 \\ fl(10^{-4}/1) & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10^{-4} & 1 \end{bmatrix},
  \]
  \[
  R = \begin{bmatrix} 1 & 1 \\ 0 & fl(1 - 10^{-4}) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.
  \]

\( L \) and \( R \) are both well-conditioned.
2.3 Gaussian elimination

2.3.2 \textit{LDR- and }$LL^T$\textit{-factorizations}

Let $A = LDR$ as in Corollary 2.2.1.

**Algorithm 2.3.2 (Crout’s factorization or compact method)**

For $k = 1, \ldots , n,$

for $p = 1, 2, \ldots , k - 1,$

\begin{align*}
r_p & := d_p a_{pk}, \\
\omega_p & := a_{kp} d_p, \\
d_k & := a_{kk} - \sum_{p=1}^{k-1} a_{kp} r_p,
\end{align*}

if $d_k = 0,$ then stop; else

for $i = k + 1, \ldots , n,$

\begin{align*}
a_{ik} & := (a_{ik} - \sum_{p=1}^{k-1} a_{ip} r_p)/d_k, \\
a_{ki} & := (a_{ki} - \sum_{p=1}^{k-1} \omega_p a_{pi})/d_k.
\end{align*}

Cost: $n^3/3$ flops.

- With partial pivoting: see Wilkinson EVP pp. 225-.
- Advantage: One can use double precision for inner product.

**Theorem 2.3.1** If $A$ is nonsingular, real and symmetric, then $A$ has a unique $LDL^T$-factorization, where $D$ is diagonal and $L$ is a unit lower triangular matrix (with one on the diagonal).


**Theorem 2.3.2** If $A$ is symmetric and positive definite, then there exists a lower triangular $G \in \mathbb{R}^{n \times n}$ with positive diagonal elements such that $A = GG^T$.

Proof: $A$ is symmetric positive definite $\Leftrightarrow x^T A x \geq 0,$ for all nonzero vector $x \in \mathbb{R}^{n \times n}$ $\Leftrightarrow k_i \geq 0$, for $i = 1, \ldots , n$ $\Leftrightarrow$ all eigenvalues of $A$ are positive.

From Corollary 2.2.1 and Theorem 2.3.1 we have $A = LDL^T$. From $L^{-1} A L^{-T} = D$ follows that $d_k = (e_k^T L^{-1}) A (L^{-T} e_k) > 0$. Thus, $G = L \text{diag}\{d_1^{1/2}, \ldots , d_n^{1/2}\}$ is real, and then $A = GG^T$. 

**Algorithm 2.3.3 (Cholesky factorization)** Let $A$ be symmetric positive definite. To find a lower triangular matrix $G$ such that $A = GG^T$.

For $k = 1, 2, \ldots , n,$

\begin{align*}
a_{kk} & := (a_{kk} - \sum_{p=1}^{k-1} a_{kp}^2)^{1/2}, \\
\text{for } i = k + 1, \ldots , n,
\end{align*}

\begin{align*}
a_{ik} & := (a_{ik} - \sum_{p=1}^{k-1} a_{ip} a_{kp})/a_{kk}.
\end{align*}

Cost: $n^3/6$ flops.

**Remark 2.3.1** For solving symmetric, indefinite systems: See Golub/ Van Loan Matrix Computation pp. 159-168. PAPT = LDL$^T$, $D$ is $1 \times 1$ or $2 \times 2$ block-diagonal matrix, $P$ is a permutation and $L$ is lower triangular with one on the diagonal.
Consider the linear system
\[ Ax = b, \]  
(2.3.4)
and the perturbed linear system
\[ (A + \delta A)(x + \delta x) = b + \delta b, \]  
(2.3.5)
where \( \delta A \) and \( \delta b \) are errors of measure or round-off in factorization.

**Definition 2.3.1** Let \( \| \cdot \| \) be an operator norm and \( A \) be nonsingular. Then \( \kappa \equiv \kappa(A) = \|A\|\|A^{-1}\| \) is a condition number of \( A \) corresponding to \( \| \cdot \| \).

**Theorem 2.3.3 (Forward error bound)** Let \( x \) be the solution of the (2.3.4) and \( x + \delta x \) be the solution of the perturbed linear system (2.3.5). If \( \|\delta A\|\|A^{-1}\| < 1 \), then
\[ \frac{\|\delta x\|}{\|x\|} \leq \frac{\kappa}{1 - \kappa \|\delta A\|} \left( \|\delta A\| + \|\delta b\| \right). \]  
(2.3.6)

**Proof:** From (2.3.5) we have
\[ (A + \delta A)\delta x + Ax + \delta Ax = b + \delta b. \]
Thus,
\[ \delta x = -(A + \delta A)^{-1}[(\delta A)x - \delta b]. \]  
(2.3.7)
Here, Corollary 2.7 implies that \((A + \delta A)^{-1}\) exists. Now,
\[ \|(A + \delta A)^{-1}\| = \|(I + A^{-1}\delta A)^{-1}\| \leq \|A^{-1}\| \frac{1}{1 - \|A^{-1}\| \|\delta A\|}, \]
On the other hand, \( b = Ax \) implies \( \|b\| \leq \|A\|\|x\| \). So,
\[ \frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}. \]  
(2.3.8)
From (2.3.7) follows that \( \|\delta x\| \leq \frac{\|A^{-1}\|\|\delta A\|}{1 - \|A^{-1}\| \|\delta A\|}(\|\delta A\|\|x\| + \|\delta b\|) \). By using (2.3.8), the inequality (2.3.6) is proved.

**Remark 2.3.2** If \( \kappa(A) \) is large, then \( A \) (for the linear system \( Ax = b \)) is called ill-conditioned, else well-conditioned.

**2.3.4 Error analysis for Gaussian algorithm**

A computer is characterized by four integers: (a) the machine base \( \beta \); (b) the precision \( t \); (c) the underflow limit \( L \); (d) the overflow limit \( U \). Define the set of floating point numbers.
\[ F = \{ f = \pm 0.d_1d_2 \cdots d_t \times \beta^e \mid 0 \leq d_i < \beta, d_1 \neq 0, L \leq e \leq U \} \cup \{0\}. \]  
(2.3.9)
2.3 Gaussian elimination

Let \( G = \{ x \in \mathbb{R} | m \leq |x| \leq M \} \cup \{ 0 \} \), where \( m = \beta^{L-1} \) and \( M = \beta^U (1 - \beta^{-t}) \) are the minimal and maximal numbers of \( F \setminus \{ 0 \} \) in absolute value, respectively. We define an operator \( fl : G \to F \) by

\[
fl(x) = \text{the nearest } c \in F \text{ to } x \text{ by rounding arithmetic.}
\]

One can show that \( fl \) satisfies

\[
fl(x) = x(1 + \varepsilon), \quad |\varepsilon| \leq \epsilon_s,
\]

where \( \epsilon_s = \frac{1}{2} \beta^{1-t} \). (If \( \beta = 2 \), then \( \epsilon_s = 2^{-t} \).) It follows that

\[
fl(a \circ b) = (a \circ b)(1 + \varepsilon)
\]

or

\[
fl(a \circ b) = (a \circ b)/(1 + \varepsilon),
\]

where \( |\varepsilon| \leq \epsilon_s \) and \( \circ = +, -, \times, / \).

Algorithm 2.3.4 Given \( x, y \in \mathbb{R}^n \). The following algorithm computes \( x^T y \) and stores the result in \( s \).

\[
s = 0,
\]

for \( k = 1, \ldots, n \),

\[
s = s + x_k y_k.
\]

Theorem 2.3.4 If \( n2^{-t} \leq 0.01 \), then

\[
fl(\sum_{k=1}^{n} x_k y_k) = \sum_{k=1}^{n} x_k y_k [1 + 1.01(n + 2 - k)\theta_k 2^{-t}], \quad |\theta_k| \leq 1
\]

Proof: Let \( s_p = fl(\sum_{k=1}^{p} x_k y_k) \) be the partial sum in Algorithm 2.3.4. Then

\[
s_1 = x_1 y_1 (1 + \delta_1)
\]

with \( |\delta_1| \leq \epsilon_s \) and for \( p = 2, \ldots, n \),

\[
s_p = fl(s_{p-1} + fl(x_p y_p)) = [s_{p-1} + x_p y_p(1 + \delta_p)](1 + \varepsilon_p)
\]

with \( |\delta_p|, |\varepsilon_p| \leq \epsilon_s \). Therefore

\[
fl(x^T y) = s_n = \sum_{k=1}^{n} x_k y_k (1 + \gamma_k),
\]

where \( 1 + \gamma_k = (1 + \delta_k) \prod_{j=k}^{n} (1 + \varepsilon_j) \), and \( \varepsilon_1 \equiv 0 \). Thus,

\[
fl(\sum_{k=1}^{n} x_k y_k) = \sum_{k=1}^{n} x_k y_k [1 + 1.01(n + 2 - k)\theta_k 2^{-t}].
\]

The result follows immediately from the following useful Lemma.

\[ \square \]
Lemma 2.3.5 If \((1 + \alpha) = \prod_{k=1}^{n}(1 + \alpha_k)\), where \(|\alpha_k| \leq 2^{-t}\) and \(n 2^{-t} \leq 0.01\), then
\[
\prod_{k=1}^{n}(1 + \alpha_k) = 1 + 1.01n\theta 2^{-t} \text{ with } |\theta| \leq 1.
\]

Proof: From assumption it is easily seen that
\[
(1 - 2^{-t})^n \leq \prod_{k=1}^{n}(1 + \alpha_k) \leq (1 + 2^{-t})^n. \tag{2.3.12}
\]

Expanding the Taylor expression of \((1 - x)^n\) as \(-1 < x < 1\), we get
\[
(1 - x)^n = 1 - nx + \frac{n(n - 1)}{2}(1 - \theta x)^{n-2} x^2 \geq 1 - nx.
\]

Hence
\[
(1 - 2^{-t})^n \geq 1 - n 2^{-t}. \tag{2.3.13}
\]

Now, we estimate the upper bound of \((1 + 2^{-t})^n:\)
\[
e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = 1 + x + \frac{x^2}{2} (1 + \frac{x}{3} + \frac{2x^2}{4!} + \cdots).
\]

If \(0 \leq x \leq 0.01\), then
\[
1 + x \leq e^x \leq 1 + x + 0.01x \frac{1}{2} e^x \leq 1 + 1.01x \tag{2.3.14}
\]
(Here, we use the fact \(e^{0.01} < 2\) to the last inequality.) Let \(x = 2^{-t}\). Then the left inequality of (2.3.14) implies
\[
(1 + 2^{-t})^n \leq e^{2^{-t} n} \tag{2.3.15}
\]
Let \(x = 2^{-t} n\). Then the second inequality of (2.3.14) implies
\[
e^{2^{-t} n} \leq 1 + 1.01n 2^{-t} \tag{2.3.16}
\]
From (2.3.15) and (2.3.16) we have
\[
(1 + 2^{-t})^n \leq 1 + 1.01n 2^{-t}.
\]

Let the exact \(LR\)-factorization of \(A\) be \(L\) and \(R\) \((A = LR)\) and let \(\tilde{L}, \tilde{R}\) be the \(LR\)-factorization of \(A\) by using Gaussian Algorithm (without pivoting). There are two possibilities:

(i) Forward error analysis: Estimate \(|L - \tilde{L}|\) and \(|R - \tilde{R}|\).

(ii) Backward error analysis: Let \(\tilde{L}\tilde{R}\) be the exact \(LR\)-factorization of a perturbed matrix \(\tilde{A} = A + F\). Then \(F\) will be estimated, i.e., \(|F| \leq ?\).
2.3 Gaussian elimination

2.3.5 Apriori error estimate for backward error bound of LR-factorization

From (2.2.2) we have

$$A^{(k+1)} = L_k A^{(k)},$$

for \( k = 1, 2, \ldots, n - 1 \) (\( A^{(1)} = A \)). Denote the entries of \( A^{(k)} \) by \( a_{ij}^{(k)} \) and let \( l_{ik} = fl(a_{ik}^{(k)}/a_{kk}^{(k)}) \), \( i \geq k + 1 \). From (2.2.2) we know that

$$a_{ij}^{(k+1)} = \begin{cases} 0; & \text{for } i \geq k + 1, j = k \\ fl(a_{ij}^{(k)} - fl(l_{ik} a_{kj}^{(k)})); & \text{for } i \geq k + 1, j \geq k + 1 \\ a_{ij}^{(k)}; & \text{otherwise.} \end{cases} \quad (2.3.17)$$

From (2.3.10) we have \( l_{ik} = (a_{ik}^{(k)}/a_{kk}^{(k)})(1 + \delta_{ik}) \) with \( |\delta_{ik}| \leq 2^{-t} \). Then

$$a_{ik}^{(k)} - l_{ik} a_{kk}^{(k)} + a_{ij}^{(k)} \delta_{ik} = 0, \quad \text{for } i \geq k + 1. \quad (2.3.18)$$

Let \( a_{ik}^{(k)} \delta_{ik} \equiv \varepsilon_{ij}^{(k)} \). From (2.3.10) we also have

$$a_{ij}^{(k+1)} = fl(a_{ij}^{(k)} - fl(l_{ik} a_{kj}^{(k)})) = (a_{ij}^{(k)} - (l_{ik} a_{kj}^{(k)} (1 + \delta_{ij}))/ (1 + \delta_{ij})^{(k)} \quad (2.3.19)$$

with \( |\delta_{ij}|, |\delta_{ij}'| \leq 2^{-t} \). Then

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)} - l_{ik} a_{kj}^{(k)} \delta_{ij} + a_{ij}^{(k+1)} \delta_{ij}', \quad \text{for } i, j \geq k + 1. \quad (2.3.20)$$

Let \( \varepsilon_{ij}^{(k)} \equiv -l_{ik} a_{kj}^{(k)} \delta_{ij} + a_{ij}^{(k+1)} \delta_{ij}' \), which is the computational error of \( a_{ij}^{(k)} \) in \( A^{(k+1)} \). From (2.3.17), (2.3.18) and (2.3.20) we obtain

$$a_{ij}^{(k+1)} = \begin{cases} a_{ij}^{(k)} - l_{ik} a_{kk}^{(k)} + \varepsilon_{ij}^{(k)}; & \text{for } i \geq k + 1, j = k \\ a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)} + \varepsilon_{ij}^{(k)}; & \text{for } i \geq k + 1, j \geq k + 1 \\ a_{ij}^{(k)} + \varepsilon_{ij}^{(k)}; & \text{otherwise,} \end{cases} \quad (2.3.21)$$

where

$$\varepsilon_{ij}^{(k)} = \begin{cases} a_{ij}^{(k)} \delta_{ij}; & \text{for } i \geq k + 1, j = k, \\ -l_{ik} a_{kj}^{(k)} \delta_{ij} + a_{ij}^{(k+1)} \delta_{ij}' & \text{for } i \geq k + 1, j \geq k + 1 \\ 0; & \text{otherwise.} \end{cases} \quad (2.3.22)$$

Let \( E^{(k)} \) be the error matrix with entries \( \varepsilon_{ij}^{(k)} \). Then (2.3.21) can be written as

$$A^{(k+1)} = A^{(k)} - M_k A^{(k)} + E^{(k)}, \quad (2.3.23)$$

where

$$M_k = \begin{bmatrix} 0 & \ddots & \ddots & \ddots \\ & 0 & l_{k+1,k} & \ddots & \ddots \\ & & \ddots & \ddots & \ddots \\ & & & 0 & l_{n,k} \end{bmatrix}. \quad (2.3.24)$$
For $k = 1, 2, \ldots, n - 1$, we add the $n - 1$ equations in (2.3.23) together and get
\[
M_1 A^{(1)} + M_2 A^{(2)} + \cdots + M_{n-1} A^{(n-1)} + A^{(n)} = A^{(1)} + E^{(1)} + \cdots + E^{(n-1)}.
\]

From (2.3.17) we know that the $k$-th row of $A^{(k)}$ is equal to the $k$-th row of $A^{(k+1)}, \ldots, A^{(n)}$, respectively and from (2.3.24) we also have
\[
M_k A^{(k)} = M_k A^{(n)} = M_k \tilde{R}.
\]

Thus,
\[
(M_1 + M_2 + \cdots + M_{n-1} + I) \tilde{R} = A^{(1)} + E^{(1)} + \cdots + E^{(n-1)}.
\]

Then
\[
\tilde{L} \tilde{R} = A + E, \quad (2.3.25)
\]

where
\[
\tilde{L} = \begin{bmatrix}
1 & & & \\
0 & 1 & & \\
& \ddots & \ddots & \\
0 & \cdots & 0 & 1
\end{bmatrix}, \quad \text{and } E = E^{(1)} + \cdots + E^{(n-1)}. \quad (2.3.26)
\]

Now we assume that the partial pivoting in Gaussian Elimination are already arranged such that pivot element $a_{kk}^{(k)}$ has the maximal absolute value. So, we have $|l_{ik}| \leq 1$. Let
\[
\rho = \max_{i, j, k} |a_{ij}^{(k)}|/\|A\|_\infty. \quad (2.3.27)
\]

Then
\[
|a_{ij}^{(k)}| \leq \rho \|A\|_\infty. \quad (2.3.28)
\]

From (2.3.22) and (2.3.28) follows that
\[
|e_{ij}^{(k)}| \leq \rho \|A\|_\infty \begin{cases} 2^{-t}; & \text{for } i \geq k + 1, j = k, \\ 2^{1-t}; & \text{for } i \geq k + 1, j \geq k + 1, \\ 0; & \text{otherwise.} \end{cases} \quad (2.3.29)
\]

Therefore,
\[
|E^{(k)}| \leq \rho \|A\|_\infty 2^{-t}. \quad (2.3.30)
\]

From (2.3.26) we get
\[
|E| \leq \rho \|A\|_\infty \cdot 2^{-t} \quad (2.3.31)
\]

Hence we have the following theorem.
2.3 Gaussian elimination

Theorem 2.3.6 The LR-factorization $L$ and $R$ of $A$ using Gaussian Elimination with partial pivoting satisfies

$$\tilde{L} \tilde{R} = A + E,$$

where

$$\|E\|_\infty \leq n^2 \rho \|A\|_\infty 2^{-t}$$ (2.3.32)

Proof:

$$\|E\|_\infty \leq \rho \|A\|_\infty 2^{-t} \left( \sum_{j=1}^{n} (2j - 1) - 1 \right) < n^2 \rho \|A\|_\infty 2^{-t}.$$

Now we shall solve the linear system $Ax = b$ by using the factorization $\tilde{L}$ and $\tilde{R}$, i.e., $\tilde{L}y = b$ and $\tilde{R}x = y$.

- For $Ly = b$: From Algorithm 2.3.1 we have

$$y_1 = fl(b_1/l_{11}),$$

$$y_i = fl\left(\frac{-l_{i1}y_1 - l_{i2}y_2 - \cdots - l_{i,i-1}y_{i-1} + b_i}{l_{ii}}\right),$$ (2.3.33)

for $i = 2, 3, \ldots, n$. From (2.3.10) we have

$$\begin{cases}
  y_1 = b_1/l_{11}(1 + \delta_{11}), \text{ with } |\delta_{11}| \leq 2^{-t} \\
  y_i = fl\left(\frac{fl(-l_{i1}y_1 - l_{i2}y_2 - \cdots - l_{i,i-1}y_{i-1}) + b_i}{l_{ii}(1 + \delta_{ii})}\right) \\
  = \frac{fl(-l_{i1}y_1 - l_{i2}y_2 - \cdots - l_{i,i-1}y_{i-1}) + b_i}{l_{ii}(1 + \delta_{ii})}, \text{ with } |\delta_{ii}|, |\delta'_{ii}| \leq 2^{-t}.
\end{cases}$$ (2.3.34)

Applying Theorem 2.3.4 we get

$$fl(-l_{i1}y_1 - l_{i2}y_2 - \cdots - l_{i,i-1}y_{i-1}) = -l_{i1}(1 + \delta_{i1})y_1 - \cdots - l_{i,i-1}(1 + \delta_{i,i-1})y_{i-1},$$

where

$$|\delta_{ii}| \leq (i - 1)1.01 \cdot 2^{-t}; \text{ for } i = 2, 3, \ldots, n,$$

$$|\delta'_{ij}| \leq (i + 1 - j)1.01 \cdot 2^{-t}; \text{ for } \begin{cases} i = 2, 3, \ldots, n, \\
  j = 2, 3, \ldots, i - 1.
\end{cases}$$ (2.3.35)

So, (2.3.34) can be written as

$$\begin{cases}
  l_{11}(1 + \delta_{11})y_1 = b_1, \\
  l_{i1}(1 + \delta_{i1})y_1 + \cdots + l_{i,i-1}(1 + \delta_{i,i-1})y_{i-1} + l_{ii}(1 + \delta_{ii})(1 + \delta'_{ii})y_i = b_i, \\
  \text{for } i = 2, 3, \cdots, n.
\end{cases}$$ (2.3.36)

or

$$(L + \delta L)y = b.$$ (2.3.37)
From (2.3.35) (2.3.36) and (2.3.37) follow that
\[
|\delta L| \leq 1.01 \cdot 2^{-t}
\]
\[
\begin{bmatrix}
|l_{11}| & 0 \\
|l_{21}| & 2|l_{22}| \\
2|l_{31}| & 2|l_{32}| & 2|l_{33}| \\
3|l_{41}| & 3|l_{42}| & 2|l_{43}| & \ddots \\
\vdots & \vdots & \ddots & \ddots \\
(n-1)|l_{n1}| & (n-1)|l_{n2}| & (n-2)|l_{n3}| & \cdots & 2|l_{n,n-1}| & 2|l_{nn}|
\end{bmatrix}
\]

This implies,
\[
\|\delta L\|_\infty \leq \frac{n(n+1)}{2} \cdot 1.01 \cdot 2^{-t} \max_{i,j} |l_{ij}| \leq \frac{n(n+1)}{2} \cdot 1.01 \cdot 2^{-t}.
\]
\[
(2.3.38)
\]

**Theorem 2.3.7** For lower triangular linear system \(Ly = b\), if \(y\) is the exact solution of \((L + \delta L)y = b\), then \(\delta L\) satisfies (2.3.38) and (2.3.39).

Applying Theorem 2.3.7 to the linear system \(\tilde{L}y = b\) and \(\tilde{R}x = y\), respectively, the solution \(x\) satisfies
\[
(\tilde{L} + \delta \tilde{L})(\tilde{R} + \delta \tilde{R})x = b
\]
or
\[
(\tilde{L}\tilde{R} + (\delta \tilde{L})\tilde{R} + \tilde{L}(\delta \tilde{R}) + (\delta \tilde{L})(\delta \tilde{R}))x = b.
\]
\[
(2.3.40)
\]
Since \(\tilde{L}\tilde{R} = A + E\), substituting this equation into (2.3.40) we get
\[
[A + E + (\delta \tilde{L})\tilde{R} + \tilde{L}(\delta \tilde{R}) + (\delta \tilde{L})(\delta \tilde{R})]x = b.
\]
\[
(2.3.41)
\]
The entries of \(\tilde{L}\) and \(\tilde{R}\) satisfy
\[
|\tilde{l}_{ij}| \leq 1, \text{ and } |\tilde{r}_{ij}| \leq \rho \|A\|_\infty.
\]

Therefore, we get
\[
\begin{align*}
\|\tilde{L}\|_\infty & \leq n, \\
\|\tilde{R}\|_\infty & \leq n\rho \|A\|_\infty, \\
\|\delta \tilde{L}\|_\infty & \leq \frac{n(n+1)}{2}1.01 \cdot 2^{-t}, \\
\|\delta \tilde{R}\|_\infty & \leq \frac{n(n+1)}{2}1.01\rho 2^{-t}.
\end{align*}
\]
\[
(2.3.42)
\]

In practical implementation we usually have \(n^22^{-t} << 1\). So it holds
\[
\|\delta \tilde{L}\|_\infty \|\delta \tilde{R}\|_\infty \leq n^2\rho \|A\|_\infty 2^{-t}.
\]

Let
\[
\delta A = E + (\delta \tilde{L})\tilde{R} + \tilde{L}(\delta \tilde{R}) + (\delta \tilde{L})(\delta \tilde{R}).
\]
\[
(2.3.43)
\]

Then, (2.3.32) and (2.3.42) we get
\[
\begin{align*}
\|\delta A\|_\infty & \leq \|E\|_\infty + \|\delta \tilde{L}\|_\infty \|\tilde{R}\|_\infty + \|\tilde{L}\|_\infty \|\delta \tilde{R}\|_\infty + \|\delta \tilde{L}\|_\infty \|\delta \tilde{R}\|_\infty \\
& \leq 1.01(n^2 + 3n^2)\rho \|A\|_\infty 2^{-t}
\end{align*}
\]
\[
(2.3.44)
\]
Theorem 2.3.8 For a linear system \( Ax = b \) the solution \( x \) computed by Gaussian Elimination with partial pivoting is the exact solution of the equation \((A + \delta A)x = b\) and \( \delta A \) satisfies (2.3.43) and (2.3.44).

Remark 2.3.3 The quantity \( \rho \) defined by (2.3.27) is called a growth factor. The growth factor measures how large the numbers become during the process of elimination. In practice, \( \rho \) is usually of order 10 for partial pivot selection. But it can be as large as \( \rho = 2^{n-1} \), when

\[
A = \begin{bmatrix}
1 & 0 & \cdots & \cdots & 0 & 1 \\
-1 & 1 & 0 & \cdots & 0 & 1 \\
\vdots & -1 & \ddots & \ddots & \vdots & 1 \\
\vdots & & \ddots & \ddots & 0 & 1 \\
-1 & -1 & \cdots & -1 & 1 & 1 \\
-1 & -1 & \cdots & -1 & 1 & 1
\end{bmatrix}.
\]

Better estimates hold for special types of matrices. For example in the case of upper Hessenberg matrices, that is, matrices of the form

\[
A = \begin{bmatrix}
\times & \cdots & \cdots & \times \\
\times & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
0 & \times & \times
\end{bmatrix}
\]

the bound \( \rho \leq (n - 1) \) can be shown. (Hessenberg matrices arise in eigenvalue problems.) For tridiagonal matrices

\[
A = \begin{bmatrix}
\alpha_1 & \beta_2 & 0 \\
\gamma_2 & \ddots & \ddots \\
& \ddots & \ddots & \ddots \\
0 & & \gamma_n & \beta_n
\end{bmatrix}
\]

it can even be shown that \( \rho \leq 2 \) holds for partial pivot selection. Hence, Gaussian elimination is quite numerically stable in this case.

For complete pivot selection, Wilkinson (1965) has shown that

\[
|a_{ij}^k| \leq f(k) \max_{i,j} |a_{ij}|
\]

with the function

\[
f(k) := k^k[2^{k^2} 3^{k^3} 4^{k^4} \cdots k^{(k-1)}]^{\frac{1}{2}}.
\]

This function grows relatively slowly with \( k \):

\[
\begin{array}{c|cccc}
k & 10 & 20 & 50 & 100 \\
\hline
f(k) & 19 & 67 & 530 & 3300
\end{array}
\]
Even this estimate is too pessimistic in practice. Up until now, no matrix has been found which fails to satisfy

$$|a_{ij}^{(k)}| \leq (k + 1) \max_{ij} |a_{ij}| \quad k = 1, 2, \ldots, n - 1,$$

when complete pivot selection is used. This indicates that Gaussian elimination with complete pivot selection is usually a stable process. Despite this, partial pivot selection is preferred in practice, for the most part, because:

(i) Complete pivot selection is more costly than partial pivot selection. (To compute $A^{(i)}$, the maximum from among $(n - i + 1)^2$ elements must be determined instead of $n - i + 1$ elements as in partial pivot selection.)

(ii) Special structures in a matrix, i.e. the band structure of a tridiagonal matrix, are destroyed in complete pivot selection.

### 2.3.6 Improving and Estimating Accuracy

**Iterative Improvement:**

Suppose that the linear system $Ax = b$ has been solved via the $LR$-factorization $PA = LR$. Now we want to improve the accuracy of the computed solution $x$. We compute

$$\begin{cases}
  r &= b - Ax, \\
  Ly &= Pr, \quad Rz = y, \\
  x_{new} &= x + z.
\end{cases} \quad (2.3.45)$$

Then in exact arithmetic we have

$$Ax_{new} = A(x + z) = (b - r) + Az = b.$$  

Unfortunately, $r = fl(b - Ax)$ renders an $x_{new}$ that is no more accurate than $x$. It is necessary to compute the residual $b - Ax$ with extended precision floating arithmetic.

**Algorithm 2.3.5**

1. Compute $PA = LR$ \hspace{1em} (t-digit)
2. Repeat: $r := b - Ax$ \hspace{1em} (2t-digit)
3. Solve $Ly = Pr$ for $y$ \hspace{1em} (t-digit)
4. Solve $Rz = y$ for $z$ \hspace{1em} (t-digit)
5. Update $x = x + z$ \hspace{1em} (t-digit)

This is referred to as an iterative improvement. From (2.3.45) we have

$$r_i = b_i - a_{i1}x_1 - a_{i2}x_2 - \cdots - a_{in}x_n. \quad (2.3.46)$$

Now, $r_i$ can be roughly estimated by $2^{-t} \max_j |a_{ij}| |x_j|$. That is

$$\|r\| \approx 2^{-t} \|A\| \|x\|. \quad (2.3.47)$$
Let \( e = x - A^{-1}b = A^{-1}(Ax - b) = -A^{-1}r. \) Then we have
\[
\|e\| \leq \|A^{-1}\| \|r\|. \tag{2.3.48}
\]
From (2.3.47) follows that
\[
\|e\| \approx \|A^{-1}\| \cdot 2^{-t} \|A\| \|x\| = 2^{-t} \text{cond}(A) \|x\|.
\]
Let
\[
\text{cond}(A) = 2^p, \; 0 < p < t, \; (p \text{ is integer}). \tag{2.3.49}
\]
Then we have
\[
\|e\|/\|x\| \approx 2^{-(t-p)}. \tag{2.3.50}
\]
From (2.3.50) we know that \( x \) has \( q = t - p \) correct significant digits. Since \( r \) is computed by double precision, so we can assume that it has at least \( t \) correct significant digits. Therefore for solving \( Az = r \) according to (2.3.50) the solution \( z \) (comparing with \( -e = A^{-1}r \)) has \( q \)-digits accuracy so that \( x_{new} = x + z \) has usually \( 2q \)-digits accuracy. From above discussion, the accuracy of \( x_{new} \) is improved about \( q \)-digits after one iteration. Hence we stop the iteration, when the number of the iterates \( k \) (say!) satisfies \( kq \geq t \).

From above we have
\[
\|z\|/\|x\| \approx \|e\|/\|x\| \approx 2^{-q} = 2^{-(t-p)}. \tag{2.3.51}
\]
From (2.3.49) and (2.3.51) we have
\[
\text{cond}(A) = 2^t \cdot (\|z\|/\|x\|).
\]
By (2.3.51) we get
\[
q = \log_2(\|z\|/\|x\|) \quad \text{and} \quad k = \frac{t}{\log_2(\|z\|/\|x\|)}.
\]
In the following we shall give a further discussion of convergence of the iterative improvement. From Theorem 2.3.8 we know that \( z \) in Algorithm 5.5 is computed by \( (A + \delta A)z = r \).

That is
\[
A(I + F)z = r, \tag{2.3.52}
\]
where \( F = A^{-1}\delta A. \)

\textbf{Theorem 2.3.9} Let the sequence of vectors \( \{x_v\} \) be the sequence of improved solutions in Algorithm 5.5 for solving \( Ax = b \) and \( x^* = A^{-1}b \) be the exact solution. Assume that \( F_k \) in (2.3.52) satisfies \( \|F_k\| \leq \sigma < 1/2 \) for all \( k \). Then \( \{x_k\} \) converges to \( x^* \), i.e., \( \lim_{v \to \infty} \|x_k - x^*\| = 0. \)

\textit{Proof:} From (2.3.52) and \( r_k = b - Ax_k \) we have
\[
A(I + F_k)z_k = b - Ax_k. \tag{2.3.53}
\]
Since \( A \) is nonsingular, multiplying both sides of (2.3.53) by \( A^{-1} \) we get
\[
(I + F_k)z_k = x^* - x_k.
\]
From $x_{k+1} = x_k + z_k$ we have $(I + F_k)(x_{k+1} - x_k) = x^* - x_k$, i.e.,

$$(I + F_k)x_{k+1} = F_kx_k + x^*. \tag{2.3.54}$$

Subtracting both sides of (2.3.54) from $(I + F_k)x^*$ we get

$$(I + F_k)(x_{k+1} - x^*) = F_k(x_k - x^*).$$

Applying Corollary 1.2.1 we have

$$x_{k+1} - x^* = (I + F_k)^{-1}F_k(x_k - x^*).$$

Hence,

$$
\|x_{k+1} - x^*\| \leq \|F_k\| \frac{\|x_k - x^*\|}{1 - \|F_k\|} \leq \frac{\sigma}{1 - \sigma} \|x_k - x^*\|. 
$$

Let $\tau = \sigma/(1 - \sigma)$. Then

$$
\|x_k - x^*\| \leq \tau^{k-1} \|x_1 - x^*\|. 
$$

But $\sigma < 1/2$ follows $\tau < 1$. This implies convergence of Algorithm 2.3.5. \hfill \blacksquare

**Corollary 2.3.1** If

$$1.01(n^3 + 3n^2)\rho 2^{-t}\|A\| \|A^{-1}\| < 1/2,$$

then Algorithm 2.3.5 converges.

**Proof:** From (2.3.52) and (2.3.44) follows that

$$
\|F_k\| \leq 1.01(n^3 + 3n^2)\rho 2^{-t}\text{cond}(A) < 1/2. 
$$

\hfill \blacksquare

### 2.3.7 Special Linear Systems

**Toeplitz Systems**

**Definition 2.3.2** (i) $T \in \mathbb{R}^{n \times n}$ is called a Toeplitz matrix if there exists $r_{-n+1}, \ldots, r_0, \ldots, r_{n-1}$ such that $a_{ij} = r_{j-i}$ for all $i, j$. e.g.,

$$T = \begin{bmatrix}
    r_0 & r_1 & r_2 & r_3 \\
    r_{-1} & r_0 & r_1 & r_2 \\
    r_{-2} & r_{-1} & r_0 & r_1 \\
    r_{-3} & r_{-2} & r_{-1} & r_0
\end{bmatrix}, \quad (n = 4).
$$

(ii) $B \in \mathbb{R}^{n \times n}$ is called a Persymmetric matrix if it is symmetric about northeast-southwest diagonal, i.e., $b_{ij} = b_{n-j+1,n-i+1}$ for all $i, j$. That is,

$$B = EBT E, \quad \text{where} \quad E = [e_n, \ldots, e_1].$$
Given scalars \( r_1, \ldots, r_{n-1} \) such that the matrices

\[
T_k = \begin{bmatrix}
1 & r_1 & r_2 & \cdots & r_{k-1} \\
r_1 & 1 & r_1 & \vdots & \\
\vdots & \vdots & \ddots & \ddots & \\
r_{k-1} & \cdots & \cdots & 1
\end{bmatrix}
\]

are all positive definite, for \( k = 1, \ldots, n \). Three algorithms will be described:

(a) Durbin’s Algorithm for the Yule-Walker problem

\[
T_n y = -(r_1, \ldots, r_n)^T.
\]

(b) Levinson’s Algorithm for the general right hand side \( T_n x = b \).

(c) Trench’s Algorithm for computing \( B = T_n^{-1} \).

- To (a): Let \( E_k = [e_k^{(k)}, \ldots, e_1^{(k)}] \). Suppose the \( k \)-th order Yule-Walker system

\[
T_k y = -(r_1, \ldots, r_k)^T = -r^T
\]

has been solved. Consider the \((k+1)\)-st order system

\[
\begin{bmatrix}
T_k & E_k r \\
r^T E_k & 1
\end{bmatrix}
\begin{bmatrix}
z \\
\alpha
\end{bmatrix}
= \begin{bmatrix}
-r \\
-r_{k+1}
\end{bmatrix}
\]

can be solved in \( O(k) \) flops. Observe that

\[
z = T_k^{-1}(-r - \alpha E_k r) = y - \alpha T_k^{-1} E_k r \tag{2.3.55}
\]

and

\[
\alpha = -r_{k+1} = -r^T E_k z. \tag{2.3.56}
\]

Since \( T_k^{-1} \) is persymmetric, \( T_k^{-1} E_k = E_k T_k^{-1} \) and \( z = y + \alpha E_k y \). Substituting into (2.3.56) we get

\[
\alpha = -r_{k+1} - r^T E_k (y + \alpha E_k y) = -(r_{k+1} + r^T E_k y)/(1 + r^T y).
\]

Here \( 1 + r^T y \) is positive, because \( T_{k+1} \) is positive definite and

\[
\begin{bmatrix}
I & E_k y \\
0 & 1
\end{bmatrix}^T
\begin{bmatrix}
T_k & E_k r \\
r^T E_k & 1
\end{bmatrix}
\begin{bmatrix}
I & E_k y \\
0 & 1
\end{bmatrix}
= \begin{bmatrix}
T_k & 0 \\
0 & 1 + r^T y
\end{bmatrix}.
\]

**Algorithm 2.3.6 (Durbin Algorithm, 1960)** Let \( T_k y^{(k)} = -r^{(k)} = -(r_1, \ldots, r_k)^T \).

For \( k = 1, \ldots, n \),

\[
y^{(1)} = -r_1,
\]

for \( k = 1, \ldots, n - 1 \),

\[
\begin{align*}
\beta_k &= 1 + r^{(k)} y^{(k)}, \\
\alpha_k &= -(r_{k+1} + r^{(k)} E_k y^{(k)}) / \beta_k, \\
z^{(k)} &= y^{(k)} + \alpha_k E_k y^{(k)}, \\
y^{(k+1)} &= \begin{bmatrix} z^{(k)} \\ \alpha_k \end{bmatrix}.
\end{align*}
\]
This algorithm requires $\frac{3}{2}n^2$ flops to generate $y = y^{(n)}$.

Further reduction:

$$
\beta_k = 1 + r^{(k)}y^{(k)}
= 1 + [r^{(k-1)}T, r^{(k)}] \begin{bmatrix}
y^{(k-1)} + \alpha_{k-1}E_{k-1}y^{(k-1)} \\
\alpha_{k-1}
\end{bmatrix}
= 1 + r^{(k-1)}y^{(k-1)} + \alpha_{k-1}(r^{(k-1)}TE_{k-1}y^{(k-1)} + r_k)
= \beta_{k-1} + \alpha_{k-1}(-\beta_{k-1}\alpha_{k-1}) = (1 - \alpha_{k-1}^2)\beta_{k-1}.
$$

\begin{itemize}
\item To (b):
\end{itemize}

$$
T_k x = b = (b_1, \ldots, b_k)^T, \text{ for } 1 \leq k \leq n. \quad (2.3.57)
$$

Want to solve

$$
\begin{bmatrix}
T_k & E_k r \\
r^T E_k & 1
\end{bmatrix}
\begin{bmatrix}
\nu \\
b
\end{bmatrix}
= \begin{bmatrix}
b \\
b_{k+1}
\end{bmatrix},
$$

where $r = (r_1, \ldots, r_k)^T$. Since $\nu = T_k^{-1}(b - \mu E_k r) = x + \mu E_k y$, it follows that

$$
\mu = b_{k+1} - r^T E_k \nu = b_{k+1} - r^T E_k x - \mu r^T y
= (b_{k+1} - r^T E_k x)/(1 + r^T y).
$$

We can effect the transition from (2.3.57) to (2.3.58) in $O(k)$ flops. We can solve the system $T_n x = b$ by solving

$$
T_k x^{(k)} = b^{(k)} = (b_1, \ldots, b_k)^T
$$

and

$$
T_k y^{(k)} = -r^{(k)} = -(r_1, \ldots, r_k)^T.
$$

It needs $2n^2$ flops. See Algorithm Levinson (1947) in Matrix Computations, pp.128-129 for details.

\begin{itemize}
\item To (c):
\end{itemize}

$$
T_n^{-1} = \begin{bmatrix}
A & Er \\
r^T E & 1
\end{bmatrix}^{-1} = \begin{bmatrix}
B & \nu \\
\nu^T & \gamma
\end{bmatrix},
$$

where $A = T_{n-1}, E = E_{n-1}$ and $r = (r_1, \ldots, r_{n-1})^T$. From the equation

$$
\begin{bmatrix}
A & Er \\
r^T E & 1
\end{bmatrix} = \begin{bmatrix}
\nu \\
\gamma
\end{bmatrix} = \begin{bmatrix}
0 \\
1
\end{bmatrix}
$$

follows that

$$
A\nu = -\gamma Er = -\gamma E(r_1, \ldots, r_{n-1})^T \text{ and } \gamma = 1 - r^T E \nu.
$$

If $y$ is the solution of $(n - 1)$-st Yule-Walker system $Ay = -r$, then

$$
\gamma = 1/(1 + r^T y) \text{ and } \nu = \gamma E y.
$$

Thus the last row and column of $T_n^{-1}$ are readily obtained. Since $AB + Er\nu^T = I_{n-1}$, we have

$$
B = A^{-1} - (A^{-1}Er)\nu^T = A^{-1} + \frac{\nu\nu^T}{\gamma}.
$$
Since $A = T_{n-1}$ is nonsingular and Toeplitz, its inverse is persymmetric. Thus
\[
b_{ij} = (A^{-1})_{ij} + \frac{\nu_i\nu_j}{\gamma} = (A^{-1})_{n-j,n-i} + \frac{\nu_i\nu_j}{\gamma}
\]
\[
= b_{n-j,n-i} - \frac{\nu_{n-i}\nu_{n-j}}{\gamma} + \frac{\nu_i\nu_j}{\gamma}
\]
\[
= b_{n-j,n-i} - \frac{1}{\gamma}(\nu_i\nu_j - \nu_{n-i}\nu_{n-j}).
\]

It needs $\frac{7}{4}n^2$ flops. See Algorithm Trench (1964) in Matrix Computations, pp.132 for details.

**Banded Systems**

**Definition 2.3.3** Let $A$ be a $n \times n$ matrix. $A$ is called a $(p, q)$-banded matrix, if $a_{ij} = 0$ whenever $i - j > p$ or $j - i > q$. $A$ has the form

\[
A = \begin{bmatrix}
\times & \ldots & \times & O \\
\vdots & \ddots & \ddots & \vdots \\
\times & \ldots & \times & O \\
O & \ldots & \vdots & \times \\
\end{bmatrix}\quad \begin{bmatrix}
T \\
q \\
\downarrow
\end{bmatrix},
\]

where $p$ and $q$ are the lower and upper band widths, respectively.

**Example 2.3.2** $(1, 1)$: tridiagonal matrix; $(1, n-1)$: upper Hessenberg matrix; $(n-1, 1)$: lower Hessenberg matrix.

**Theorem 2.3.10** Let $A$ be a $(p, q)$-banded matrix. Suppose $A$ has a $LR$-factorization $(A = LR)$. Then $L = (p, 0)$ and $R = (0, q)$-banded matrix, respectively.

**Algorithm 2.3.7** See Algorithm 4.3.1 in Matrix Computations, pp.150.

**Theorem 2.3.11** Let $A$ be a $(p, q)$-banded nonsingular matrix. If Gaussian Elimination with partial pivoting is used to compute Gaussian transformations $L_j = I - l_j e_j^T$, for $j = 1, \ldots, n-1$, and permutations $P_1, \ldots, P_{n-1}$ such that

\[
L_{n-1}P_{n-1}\cdots L_1 P_1 A = R
\]

is upper triangular, then $R$ is a $(0, p + q)$-banded matrix and $l_{ij} = 0$ whenever $i \leq j$ or $i > j + p$. (Since the $j$-th column of $L$ is a permutation of the Gaussian vector $l_j$, it follows that $L$ has at most $p + 1$ nonzero elements per column.)
Symmetric Indefinite Systems

Consider the linear system $Ax = b$, where $A \in \mathbb{R}^{n \times n}$ is symmetric but indefinite. There are a method using $n^3/6$ flops due to Aasen (1971) that computes the factorization $PAP^T = LTL^T$, where $L = [l_{ij}]$ is unit lower triangular, $P$ is a permutation chosen such that $|l_{ij}| \leq 1$, and $T$ is tridiagonal.

Rather than the above factorization $PAP^T = LTL^T$ we have the calculation of

$$PAP^T = LDL^T,$$

where $D$ is block diagonal with 1 by 1 and 2 by 2 blocks on diagonal, $L = [l_{ij}]$ is unit lower triangular, and $P$ is a permutation chosen such that $|l_{ij}| \leq 1$.

Bunch and Parlett (1971) has proposed a pivot strategy to do this, $n^3/6$ flops are required. Unfortunately the overall process requires $n^3/12 \sim n^3/6$ comparisons. A better method described by Bunch and Kaufmann (1977) requires $n^3/6$ flops and $O(n^2)$ comparisons.

A detailed discussion of this subsection see p.159-168 in Matrix Computations.