Numerical Methods for Linear Equation Systems

archibald.carrion

May 2025

Contents

1 Basic Notation

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} \tag{2}$$

2 Direct Methods

2.1 Gaussian Elimination

Algorithm 1 Gaussian Elimination with Back Substitution

```
Forward Elimination:
for k = 1 to n − 1 do
    for i = k + 1 to n do
        m_ik ← a_ik / a_kk
        a_ik ← 0
        for j = k + 1 to n do
            a_ij ← a_ij − m_ik a_kj
        end for
        b_i ← b_i − m_ik b_k
    end for
end for
Back Substitution:
x_n ← b_n / a_nn
for i = n − 1 down to 1 do
    s ← 0
    for j = i + 1 to n do
        s ← s + a_ij x_j
    end for
    x_i ← (b_i − s) / a_ii
end for
```

Matrix form:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \xrightarrow[\text{operations}]{\text{elementary row}} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a'_{22} & a'_{23} \\ 0 & 0 & a'_{33} \end{pmatrix} \tag{3}$$

2.2 LU Decomposition

$$A = LU = \begin{pmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix} \tag{4}$$

Algorithm:

Solving $A\mathbf{x} = \mathbf{b}$ using LU: $\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ LU\mathbf{x} &= \mathbf{b} \end{aligned}$

Algorithm 2 LU Decomposition (Doolittle's method)

```
1: for i = 1 to n do
2:     for j = i to n do
3:         u_ij ← a_ij − Σ_{k=1}^{i−1} l_ik u_kj
4:     end for
5:     for j = i + 1 to n do
6:         l_ji ← (1/u_ii) (a_ji − Σ_{k=1}^{i−1} l_jk u_ki)
7:     end for
8:     l_ii ← 1
9: end for
```

$$
\begin{aligned}
L\mathbf{y} &= \mathbf{b} \\
y_1 &= b_1 \\
y_i &= b_i - \sum_{j=1}^{i-1} l_{ij} y_j \quad \text{for } i = 2, \ldots, n \\
U\mathbf{x} &= \mathbf{y} \\
x_n &= \frac{y_n}{u_{nn}} \\
x_i &= \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{n} u_{ij} x_j \right) \quad \text{for } i = n-1, \ldots, 1
\end{aligned}
$$

2.3 Cholesky Decomposition

For symmetric positive definite matrices:

$$
A = LL^T = \begin{pmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ 0 & l_{22} & \cdots & l_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & l_{nn} \end{pmatrix} \tag{5}
$$

Algorithm:

Algorithm 3 Cholesky Decomposition

```
1: for i = 1 to n do
2:     l_ii ← sqrt(a_ii − Σ_{k=1}^{i−1} l_ik^2)
3:     for j = i + 1 to n do
4:         l_ji ← (1/l_ii) (a_ji − Σ_{k=1}^{i−1} l_jk l_ik)
5:     end for
6: end for
```

2.4 QR Decomposition

$$A = QR = \begin{pmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{pmatrix} \tag{6}$$

Gram-Schmidt Process:

Algorithm 4 QR Decomposition via Gram-Schmidt

```
for j = 1 to n do
    v_j ← a_j
    for i = 1 to j − 1 do
        r_ij ← q_i^T a_j
        v_j ← v_j − r_ij q_i
    end for
    r_jj ← ‖v_j‖_2
    q_j ← v_j / r_jj
end for
```

$$\text{Householder matrix: } H_k = I - 2\mathbf{u}_k\mathbf{u}_k^T$$

Householder reflections:

$$\mathbf{u}_k = \frac{\mathbf{v}_k}{\|\mathbf{v}_k\|_2}$$

$$\mathbf{v}_k = \mathbf{a}_k + \text{sign}(a_{kk})\|\mathbf{a}_k\|_2\mathbf{e}_1$$

3 Iterative Methods

3.1 Jacobi Method

$$A = D + L + U \quad \text{(Diagonal + Lower + Upper)}$$

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j^{(k)}\right)$$

$$\mathbf{x}^{(k+1)} = D^{-1}(\mathbf{b} - (L+U)\mathbf{x}^{(k)})$$

Algorithm:

3.2 Gauss-Seidel Method

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)$$

$$\mathbf{x}^{(k+1)} = (D+L)^{-1}(\mathbf{b} - U\mathbf{x}^{(k)})$$

Algorithm:

Algorithm 5 Jacobi Method

Choose initial guess $\mathbf{x}^{(0)}$
Set convergence tolerance ε
$k \leftarrow 0$
repeat
  for $i = 1$ to n **do**
    $x_i^{(k+1)} \leftarrow \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{(k)} \right)$
  end for
  $k \leftarrow k + 1$
until $\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_2 < \varepsilon$

Algorithm 6 Gauss-Seidel Method

Choose initial guess $\mathbf{x}^{(0)}$
Set convergence tolerance ε
$k \leftarrow 0$
repeat
  for $i = 1$ to n **do**
    $x_i^{(k+1)} \leftarrow \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$
  end for
  $k \leftarrow k + 1$
until $\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_2 < \varepsilon$

3.3 Successive Over-Relaxation

$$x_i^{(k+1)} = (1 - \omega) x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$$

$$\mathbf{x}^{(k+1)} = (D + \omega L)^{-1} ((1 - \omega) D - \omega U) \mathbf{x}^{(k)} + \omega (D + \omega L)^{-1} \mathbf{b}$$

Algorithm:

3.4 Conjugate Gradient Method

For symmetric positive definite matrices:

Algorithm 7 Successive Over-Relaxation Method

Choose initial guess $\mathbf{x}^{(0)}$
Set relaxation parameter $\omega \in (0, 2)$
Set convergence tolerance ε
$k \leftarrow 0$
repeat
  for $i = 1$ to n **do**
    $x_i^{(k+1)} \leftarrow (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$
  end for
  $k \leftarrow k + 1$
until $\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_2 < \varepsilon$

Algorithm 8 Conjugate Gradient Method

Choose initial guess $\mathbf{x}^{(0)}$
$\mathbf{r}^{(0)} \leftarrow \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{p}^{(0)} \leftarrow \mathbf{r}^{(0)}$
for $k = 0, 1, 2, \ldots$ until convergence **do**
  $\alpha_k \leftarrow \frac{\mathbf{r}^{(k)T}\mathbf{r}^{(k)}}{\mathbf{p}^{(k)T}A\mathbf{p}^{(k)}}$
  $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)} + \alpha_k \mathbf{p}^{(k)}$
  $\mathbf{r}^{(k+1)} \leftarrow \mathbf{r}^{(k)} - \alpha_k A\mathbf{p}^{(k)}$
  $\beta_k \leftarrow \frac{\mathbf{r}^{(k+1)T}\mathbf{r}^{(k+1)}}{\mathbf{r}^{(k)T}\mathbf{r}^{(k)}}$
  $\mathbf{p}^{(k+1)} \leftarrow \mathbf{r}^{(k+1)} + \beta_k \mathbf{p}^{(k)}$
end for

4 Special Cases

4.1 Tridiagonal Systems

$$A = \begin{pmatrix} b_1 & c_1 & 0 & \cdots & 0 \\ a_2 & b_2 & c_2 & \cdots & 0 \\ 0 & a_3 & b_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & a_n & b_n \end{pmatrix} \tag{7}$$

Thomas Algorithm:

Algorithm 9 Thomas Algorithm (Tridiagonal Matrix Algorithm)

Forward Sweep:
$c'_1 \leftarrow \frac{c_1}{b_1}$
$d'_1 \leftarrow \frac{d_1}{b_1}$
for $i = 2$ to n **do**
  $c'_i \leftarrow \frac{c_i}{b_i - a_i c'_{i-1}}$
  $d'_i \leftarrow \frac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}}$
end for
Back Substitution:
$x_n \leftarrow d'_n$
for $i = n - 1$ down to 1 **do**
  $x_i \leftarrow d'_i - c'_i x_{i+1}$
end for

4.2 Sparse Systems

$$\begin{aligned} \text{values} &= [a_{11}, a_{12}, \ldots, a_{mn}] \quad \text{(non-zero elements)} \\ \text{col\_indices} &= [j_1, j_2, \ldots, j_{nnz}] \quad \text{(column indices)} \\ \text{row\_ptr} &= [0, p_1, p_2, \ldots, p_m] \quad \text{(pointers to row starts)} \end{aligned}$$

5 Error Analysis

$$\begin{aligned} \text{True error: } & \mathbf{e} = \mathbf{x} - \tilde{\mathbf{x}} \\ \text{Residual: } & \mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}} \\ \text{Relationship: } & A\mathbf{e} = \mathbf{r} \end{aligned}$$

Condition Number:

$$\kappa(A) = \|A\| \cdot \|A^{-1}\| = \frac{\sigma_{\max}}{\sigma_{\min}} \tag{8}$$

$$\text{Error Bounds:} \quad \begin{aligned} \frac{\|\mathbf{e}\|}{\|\mathbf{x}\|} &\leq \kappa(A)\left(\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}\right) \\ \frac{\|\mathbf{e}\|}{\|\mathbf{x}\|} &\leq \frac{\kappa(A)}{1-\kappa(A)\varepsilon}\left(\frac{\|\Delta A\|}{\|A\|}+\frac{\|\Delta \mathbf{b}\|}{\|\mathbf{b}\|}\right) \end{aligned}$$

6 Convergence Criteria

$$\begin{aligned} \text{Absolute Error: } & \|\mathbf{x}^{(k+1)}-\mathbf{x}^{(k)}\| < \varepsilon_{\text{abs}} \\ \text{Relative Error: } & \frac{\|\mathbf{x}^{(k+1)}-\mathbf{x}^{(k)}\|}{\|\mathbf{x}^{(k+1)}\|} < \varepsilon_{\text{rel}} \\ \text{Residual: } & \|\mathbf{b}-A\mathbf{x}^{(k)}\| < \varepsilon_{\text{res}} \\ \text{Normalized Residual: } & \frac{\|\mathbf{b}-A\mathbf{x}^{(k)}\|}{\|\mathbf{b}\|} < \varepsilon_{\text{nres}} \end{aligned}$$

$$\text{Convergence Conditions:} \quad \begin{aligned} \text{Jacobi: } & \rho(D^{-1}(L+U)) < 1 \\ \text{Gauss-Seidel: } & \rho((D+L)^{-1}U) < 1 \\ \text{SOR: } & \rho((D+\omega L)^{-1}((1-\omega)D-\omega U)) < 1 \end{aligned}$$

For diagonally dominant matrices:

$$|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ij}| \quad \text{for all } i = 1, 2, \ldots, n \tag{9}$$