Chp.7 - Iterative Techniques in Matrix Algebra

Arindam Saikia

May 20, 2023

Context **

! The slides where I have added clarifications or additional examples etc are marked by ** on the top.

Introduction

Basic Idea for the Chapter

We look into methods of approximating solutions to **'Systems of Linear Equations'** while taking the help of **matrices**, and also use multiple **iterations** (repetitions) rather than direct methods in order to achieve a more accurate result (or more 'convergent' result towards the solution as we will see)

Methods we will encounter

1. **Jacobi Iterative**
2. **Gauss-Seidel Iterative**
3. **Successive Over-Relaxation**
4. **Iterative Refinement**
5. **Preconditioned Conjugate Gradient Method**

*Each of these methods improve upon or introduce certain mathematical methods for finding the solution better than the previous one.

Motivation behind using Matrix algebra

1. **Compact representation:** Matrices provide a concise and organized way to represent systems of linear equations. Instead of writing out each equation individually, the entire system can be represented as a matrix equation, which simplifies the notation and makes it easier to work with.
2. **Exploitation of matrix operations and properties :** Iterative techniques for solving linear equations, such as the Jacobi method or the Gauss-Seidel method, are well-suited to matrix algebra. These methods involve updating approximations of the solution iteratively based on matrix operations, making use of matrix properties and operations to improve convergence and computational efficiency.
3. **Numerical stability:** Matrix algebra provides a framework for analyzing the stability and convergence of iterative methods. By studying the properties of matrices and their eigenvalues, it is possible to determine conditions under which iterative techniques will converge to the correct solution and to assess their numerical stability.

A few prerequisites

- Matrix Norms and its different types such as :
  l_1 norm a.k.a Manhattan Norm $|| * ||_2$
  l_2 norm a.k.a Euclidean Norm $|| * ||_3$
  ...
  l_∞ norm a.k.a Maximum/Infinity Norm $|| * ||_\infty$
- Real Spaces of different dimensions such as :
  R^2 a.k.a Cartesian/Euclidean Plane
  R^3 a.k.a 3-D Real Space
  etc...
- Cauchy-Bunyakovsky-Schwarz Inequality for Sums
  $x^t y = \sum_{i=1}^{n} x_i y_i \leq \sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2} = \|x\|_2 \cdot \|y\|_2$
- Distance between two vectors : defined as the norm of the difference of the vectors just as distance between two real numbers is the absolute value of their difference.
- Characteristic Polynomial given by:$p(\lambda) = det(A - \lambda I)$;where it satisfies the equation - $(A - \lambda I)x = 0$ with x as the eigenvector and all λ satisfying the equation as eigenvalues.
- Spectral Radius $\rho(A)$ of a matrix A defined by : $\rho(A) = max|\lambda|$

Jacobi's Method

* An iterative technique to solve the $n \times n$ linear system $Ax = b$ starts with an initial approximation $x^{(0)}$ to the solution x and generates a sequence of vectors $\{x^{(k)}\}_{k=0}^{\infty}$ that converges to x.
** Here k on the superscript represents the number of iterations.

* The Jacobi iterative method is obtained by solving the ith equation in $Ax = b$ for x_i to obtain (provided $a_{ii} \neq 0$):

$$x_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \left(-\frac{a_{ij}x_j}{a_{ii}}\right) + \frac{b_i}{a_{ii}}, \quad \text{for } i = 1, 2, \ldots, n.$$

We get this expression by converting the equation $Ax = b$ into an equivalent system of the form $x = Tx + c$.
For each $k \geq 1$, generate the components $x_i^{(k)}$ of $x^{(k)}$ from the components of $x^{(k-1)}$ by

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[\sum_{\substack{j=1 \\ j \neq i}}^{n} \left(-a_{ij}x_j^{(k-1)}\right) + b_i \right], \quad \text{for } i = 1, 2, \ldots, n.$$

* When to stop the iterations?
When $||x^k - x^{k-1}|| < Tolerance$

Example :

Express the Jacobi iteration method for the linear system $Ax = b$ given by

$$
\begin{aligned}
E1 &: 10x_1 - x_2 + 2x_3 = 6,\\
E2 &: -x_1 + 11x_2 - x_3 + 3x_4 = 25,\\
E3 &: 2x_1 - x_2 + 10x_3 - x_4 = -11,\\
E4 &: 3x_2 - x_3 + 8x_4 = 15,
\end{aligned}
$$

in the form $x^{(k)} = Tx^{(k-1)} + c$.
Solution: The Jacobi method for this system has the form

$$
\begin{aligned}
x_1 &= \frac{1}{10}x_2 - \frac{1}{5}x_3 + \frac{3}{5},\\
x_2 &= \frac{1}{11}x_1 + \frac{1}{11}x_3 - \frac{3}{11}x_4 + \frac{25}{11},\\
x_3 &= -\frac{1}{5}x_1 + \frac{1}{10}x_2 + \frac{1}{10}x_4 - \frac{11}{10},\\
x_4 &= -\frac{3}{8}x_2 + \frac{1}{8}x_3 + \frac{15}{8}.
\end{aligned}
$$

Hence, we have

$$T = \begin{bmatrix} 0 & \frac{1}{10} & -\frac{1}{5} & 0 \\ \frac{1}{11} & 0 & \frac{1}{11} & -\frac{3}{11} \\ -\frac{1}{5} & \frac{1}{10} & 0 & \frac{1}{10} \\ 0 & -\frac{3}{8} & \frac{1}{8} & 0 \end{bmatrix}, \quad c = \begin{bmatrix} \frac{3}{5} \\ \frac{25}{11} \\ -\frac{11}{10} \\ \frac{15}{8} \end{bmatrix}.$$

Thus, the equation we need to perform the iterations on becomes :

$$\begin{bmatrix} x_1^k \\ x_2^k \\ x_3^k \\ x_4^k \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{10} & -\frac{1}{5} & 0 \\ \frac{1}{11} & 0 & \frac{1}{11} & -\frac{3}{11} \\ -\frac{1}{5} & \frac{1}{10} & 0 & \frac{1}{10} \\ 0 & -\frac{3}{8} & \frac{1}{8} & 0 \end{bmatrix} \begin{bmatrix} x_1^{k-1} \\ x_2^{k-1} \\ x_3^{k-1} \\ x_4^{k-1} \end{bmatrix} + \begin{bmatrix} \frac{3}{5} \\ \frac{25}{11} \\ -\frac{11}{10} \\ \frac{15}{8} \end{bmatrix}$$

From the initial approximation $x^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, we have x^1 given by

$$x_1^1 = \frac{1}{10}x_2^0 - \frac{1}{5}x_3^0 + \frac{3}{5} = 0.6000$$
$$x_2^1 = \frac{1}{11}x_1^0 + \frac{1}{11}x_3^0 - \frac{3}{11}x_4^0 + \frac{25}{11} = 2.2727$$
$$x_3^1 = -\frac{1}{5}x_1^0 + \frac{1}{10}x_2^0 + \frac{1}{10}x_4^0 - \frac{11}{10} = -1.1000$$
$$x_4^1 = -\frac{3}{8}x_2^0 + \frac{1}{8}x_3^0 + \frac{15}{8} = 1.8750$$

Additional iterates, $x^k = (x_1^k, x_2^k, x_3^k, x_4^k)^T$, are generated in a similar manner and are presented in Table 7.1.

Table 7.1

k	0	1	2	3	4	5	6	7	8	9	10
$x_1^{(k)}$	0.0000	0.6000	1.0473	0.9326	1.0152	0.9890	1.0032	0.9981	1.0006	0.9997	1.0001
$x_2^{(k)}$	0.0000	2.2727	1.7159	2.053	1.9537	2.0114	1.9922	2.0023	1.9987	2.0004	1.9998
$x_3^{(k)}$	0.0000	−1.1000	−0.8052	−1.0493	−0.9681	−1.0103	−0.9945	−1.0020	−0.9990	−1.0004	−0.9998
$x_4^{(k)}$	0.0000	1.8750	0.8852	1.1309	0.9739	1.0214	0.9944	1.0036	0.9989	1.0006	0.9998

Gauss-Seidel Method

* There is room for improvement in the Jacobi's Method as the components of x^{k-1} are used to compute all the components of $x_i^{(k)}$ of $x^{(k)}$ matrix.

*But, for $i > 1$, the components $x_1^{(k)}, ..., x_{i-1}^{(k)}$ of $x^{(k)}$ matrix have already been computed and are expected to be better approximations to the actual solutions $x_1, ..., x_{i-1}$.

*Thus, we update the iteration formula with the following equation to include the use of these recently calculated values:

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

Instead of using,

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[\sum_{\substack{j=1 \\ j \neq i}}^{n} (-a_{ij} x_j^{(k-1)}) + b_i \right], \quad \text{for } i = 1, 2, \ldots, n.$$

*This modification is called the **Gauss-Seidel iterative technique**

*How do we check whether the iterations are giving us convergent results or not?
*We make use of two theorems :

- For any $x^{(0)} \in \mathbb{R}^n$, the sequence $\{x^{(k)}\}_{k=0}^{\infty}$ defined by

$$x^{(k)} = Tx^{(k-1)} + c, \quad \text{for each } k \geq 1,$$

converges to the unique solution of $x = Tx + c$ if and only if

$$\rho(T) < 1$$

; where $\rho(T)$ is the **Spectral Radius**.

- **Spectral Radius**, $\rho(T)$ is defined as :

$$\rho(T) = max|\lambda|$$

; where λ is an eigenvalue of T

Example 3: Use the Gauss-Seidel iterative technique to find approximate solutions to

$$\begin{aligned} 10x_1 - x_2 + 2x_3 &= 6, \\ -x_1 + 11x_2 - x_3 + 3x_4 &= 25, \\ 2x_1 - x_2 + 10x_3 - x_4 &= -11, \\ 3x_2 - x_3 + 8x_4 &= 15 \end{aligned}$$

starting with $x = \left[0, 0, 0, 0\right]^t$ and iterating until $\|x(k) - x(k-1)\|_\infty \|x(k)\|_\infty < 10^{-3}$.
Solution: The solution $x = \left[1, 2, -1, 1\right]^t$ was approximated by Jacobi's method in Example 1. For the Gauss-Seidel method, we write the system, for each $k = 1, 2, \ldots$ as

$$\begin{aligned} x_1^1 &= \frac{1}{10}x_2^0 - \frac{1}{5}x_3^0 + \frac{3}{5}, \\ x_2^1 &= \frac{1}{11}x_1^1 + \frac{1}{11}x_3^0 - \frac{3}{11}x_4^0 + \frac{25}{11}, \\ x_3^1 &= -\frac{1}{5}x_1^1 + \frac{1}{10}x_2^1 + \frac{1}{10}x_4^0 - \frac{11}{10}, \\ x_4^1 &= -\frac{3}{8}x_2^1 + \frac{1}{8}x_3^1 + \frac{15}{8}. \end{aligned}$$

When $x^0 = [0, 0, 0, 0]$, we have $x^1 = [0.6000, 2.3272, -0.9873, 0.8789]$. Subsequent iterations give the values in Table 7.2.

Table 7.2

k	0	1	2	3	4	5
$x_1^{(k)}$	0.0000	0.6000	1.030	1.0065	1.0009	1.0001
$x_2^{(k)}$	0.0000	2.3272	2.037	2.0036	2.0003	2.0000
$x_3^{(k)}$	0.0000	−0.9873	−1.014	−1.0025	−1.0003	−1.0000
$x_4^{(k)}$	0.0000	0.8789	0.9844	0.9983	0.9999	1.0000

Successive Over-Relaxation

* Now,one way to select a procedure to accelerate convergence is to choose a method whose associated matrix has minimal spectral radius.
* We need to introduce a new means of measuring the amount by which an approximation to the solution to a linear system differs from the true solution to the system.
*We make use of the **Residual Vector** for $\tilde{x}$ with respect to the linear system $Ax = b$, which is defined by :

$$r = b - a\tilde{x}$$

;where $\tilde{x}$ represents the approximated value.
*The objective now is to generate a sequence of approximations that will cause the residual vectors to converge rapidly to zero.
*The Gauss-Seidel method can incorporate residual vector while choosing $x_i^{(k)}$ as follows:

$$x_i^{(k)} = x_i^{(k-1)} + \frac{r_{ii}^{(k)}}{a_{ii}}$$

The mth component of r_i^k is given by

$$r_{mi}^k = b_m - \sum_{j=1}^{i-1} a_{mj} x_j^k - \sum_{j=i+1}^{n} a_{mj} x_j^{k-1},$$

or, equivalently,

$$r_{mi}^k = b_m - \sum_{j=1}^{i-1} a_{mj} x_j^k - \sum_{j=i+1}^{n} a_{mj} x_j^{k-1} - a_{mi} x_i^{k-1},$$

for each $m = 1, 2, \ldots, n$.
In particular, the ith component of r_i^k is given by

$$r_{ii}^k = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^k - \sum_{j=i+1}^{n} a_{ij} x_j^{k-1} - a_{ii} x_i^{k-1},$$

so

$$a_{ii} x_i^{k-1} + r_{ii}^k = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^k - \sum_{j=i+1}^{n} a_{ij} x_j^{k-1}. \quad (7.14)$$

Recall, however, that in the Gauss-Seidel method, $x(k)_i$ is chosen to be

$$x_i^k = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^k - \sum_{j=i+1}^{n} a_{ij} x_j^{k-1} \right), \quad (7.15)$$

so Equation (7.14) can be rewritten as

$$a_{ii} x_i^{k-1} + r_{ii}^k = a_{ii} x_i^k.$$

Consequently, the Gauss-Seidel method can be characterized as choosing x_i^k to satisfy

$$x_i^k = x_i^{k-1} + \frac{r_{ii}^k}{a_{ii}}.$$

Implementing Relaxation method

* Now we introduce the Relaxation Factor (ω) into the equation:

$$x_i^{(k)} = x_i^{(k-1)} + \omega \frac{r_{ii}^{(k)}}{a_{ii}}$$

* This Residual Factor (ω) can reduce the norm of the residual vector and obtain significantly faster convergence.
* For choices of ω with $0 < \omega < 1$, the procedures are called **under-relaxation methods.**
* And, for choices of ω with $1 < \omega$, the procedures are called **over-relaxation methods.**
* We will now use Successive **over-relaxation** to accelerate the convergence for Gauss-Seidel Method.

Example:

The linear system $Ax = b$ given by

$$\begin{aligned} 4x_1 + 3x_2 &= 24, \\ 3x_1 + 4x_2 - x_3 &= 30, \\ -x_2 + 4x_3 &= -24, \end{aligned}$$

has the solution $(3, 4, -5)^t$.Compare the iterations from the Gauss-Seidel method and the SOR method with $\omega = 1.25$ using $x^{(0)} = (1, 1, 1)^t$ for both methods.
Solution: For each $k = 1, 2, \ldots$, the equations for the Gauss-Seidel method are:

$$\begin{aligned} x_1^{(k)} &= -0.75x_2^{(k-1)} + 6, \\ x_2^{(k)} &= -0.75x_1^{(k)} + 0.25x_3^{(k-1)} + 7.5, \\ x_3^{(k)} &= 0.25x_2^{(k)} - 6, \end{aligned}$$

And the equations for the SOR method with $\omega = 1.25$ are:

$$\begin{aligned} x_1^{(k)} &= -0.25x_1^{(k-1)} - 0.9375x_2^{(k-1)} + 7.5, \\ x_2^{(k)} &= -0.9375x_1^{(k)} - 0.25x_2^{(k-1)} + 0.3125x_3^{(k-1)} + 9.375, \\ x_3^{(k)} &= 0.3125x_2^{(k)} - 0.25x_3^{(k-1)} - 7.5. \end{aligned}$$

The first seven iterates for each method are listed in Tables 7.3 and 7.4. For the iterates to be accurate to seven decimal places, the Gauss-Seidel method requires 34 iterations, as opposed to 14 iterations for the SOR method with $\omega = 1.25$.

Table 7.3

k	0	1	2	3	4	5	6	7
$x_1^{(k)}$	1	5.250000	3.1406250	3.0878906	3.0549316	3.0343323	3.0214577	3.0134110
$x_2^{(k)}$	1	3.812500	3.8828125	3.9267578	3.9542236	3.9713898	3.9821186	3.9888241
$x_3^{(k)}$	1	−5.046875	−5.0292969	−5.0183105	−5.0114441	−5.0071526	−5.0044703	−5.0027940

Table 7.4

k	0	1	2	3	4	5	6	7
$x_1^{(k)}$	1	6.312500	2.6223145	3.1333027	2.9570512	3.0037211	2.9963276	3.0000498
$x_2^{(k)}$	1	3.5195313	3.9585266	4.0102646	4.0074838	4.0029250	4.0009262	4.0002586
$x_3^{(k)}$	1	−6.6501465	−4.6004238	−5.0966863	−4.9734897	−5.0057135	−4.9982822	−5.0003486

Iterative Refinement

* It seems intuitively reasonable that if $\tilde{x}$ is an approximation to the solution x of $Ax = b$ and the residual vector $r = b - A\tilde{x}$ has the property that $\|r\|$ is small, then $\|x - \tilde{x}\|$ would be small as well. This is often the case, but certain systems, which occur frequently in practice, fail to have this property.

Example:

The linear system $Ax = b$ given by

$$\begin{bmatrix} 1 & 2 \\ 1.0001 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 3.0001 \end{bmatrix}$$

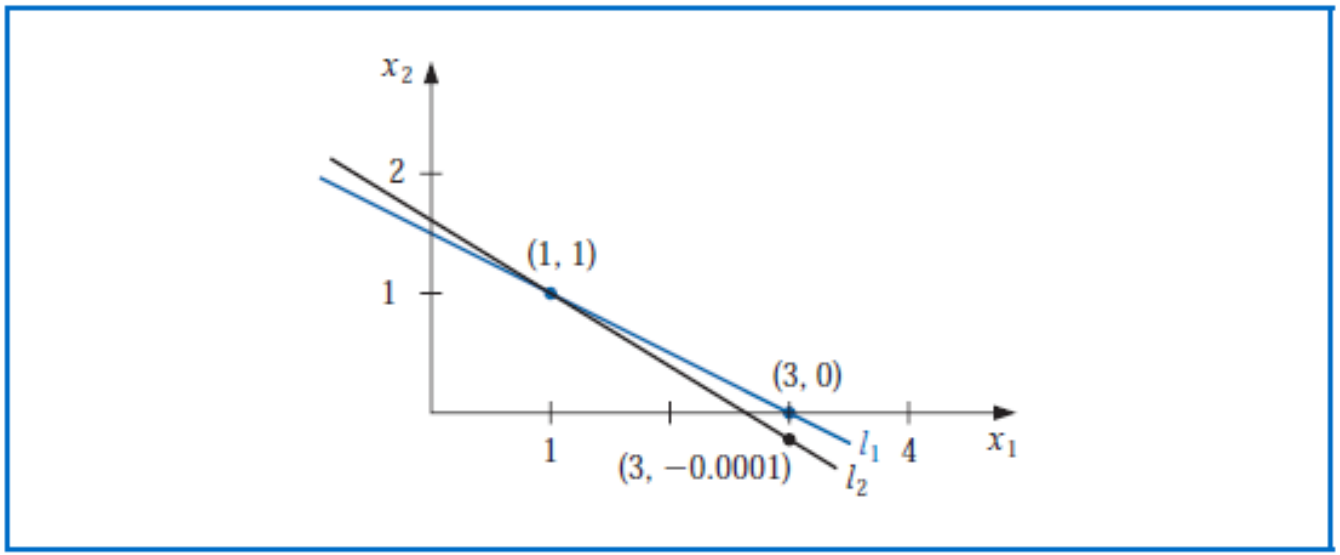
has the unique solution $x = (1, 1)^t$.Determine the residual vector for the poor approximation $\tilde{x} = (3, -0.0001)^t$.

Solution :

We have $r = b - A\tilde{x} = \begin{bmatrix} 3 \\ 3.0001 \end{bmatrix} - \begin{bmatrix} 1 & 2 \\ 1.0001 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ -0.0001 \end{bmatrix} = \begin{bmatrix} 0.0002 \\ 0 \end{bmatrix}$, so $\|r\|_\infty = 0.0002$. Although the norm of the residual vector is small, the approximation $\tilde{x} = (3, -0.0001)^t$ is obviously quite poor; in fact, $\|x - \tilde{x}\|_\infty = 2$.

The difficulty in Example 1 is explained quite simply by noting that the solution to the system represents the intersection of the lines $l_1 : x_1 + 2x_2 = 3$ and $l_2 : 1.0001x_1 + 2x_2 = 3.0001$. The point $(3, -0.0001)$ lies on l_2, and the lines are nearly parallel. This implies that $(3, -0.0001)$ also lies close to l_1, even though it differs significantly from the solution of the system, given by the intersection point $(1, 1)$. (See Figure 7.7.)

Figure 7.7



Had the lines not been nearly coincident, we would expect a small residual vector to imply an accurate approximation. In the general situation, we cannot rely on the geometry of the system to give an indication of when problems might occur. We can, however, obtain this information by considering the norms of the matrix A and its inverse.

Theorem 2.27:

Suppose that $\tilde{x}$ is an approximation to the solution of $Ax = b$, A is a nonsingular matrix, and r is the residual vector for $\tilde{x}$. Then for any natural norm,

$$\|x - \tilde{x}\| \leq \|r\| \cdot \|A^{-1}\|$$

and if $\|x\| = 0$ and $\|b\| = 0$,

$$\frac{\|x - \tilde{x}\|}{\|x\|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{\|r\|}{\|b\|}$$

Introducing Condition Numbers

* The inequalities in Theorem 7.27 imply that $\|A^{-1}\|$ and $\|A\| \cdot \|A^{-1}\|$ provide an indication of the connection between the residual vector and the accuracy of the approximation.
* The condition number of the nonsingular matrix A relative to a norm $\|\cdot\|$ is defined as $K(A) = \|A\| \cdot \|A^{-1}\|$.
* With this notation, the inequalities in Theorem 7.27 become

$$\|x - \tilde{x}\| \leq K(A)\frac{\|r\|}{\|A\|}$$

and

$$\frac{\|x - \tilde{x}\|}{\|x\|} \leq K(A)\frac{\|r\|}{\|b\|}$$

For any nonsingular matrix A and natural norm $\|\cdot\|$,

$$1 = \|I\| = \|A \cdot A^{-1}\| \leq \|A\| \cdot \|A^{-1}\| = K(A).$$

* A matrix A is well-conditioned if $K(A)$ is close to 1, and is ill-conditioned when $K(A)$ is significantly greater than 1. Conditioning in this context refers to the relative security that a small residual vector implies a correspondingly accurate approximate solution.

Example :

Determine the condition number for the matrix

$$A = \begin{bmatrix} 1 & 2 \\ 1.0001 & 2 \end{bmatrix}$$

Solution :
We saw in Example 1 that the very poor approximation $(3, -0.0001)^t$ to the exact solution $(1, 1)^t$ had a residual vector with small norm, so we should expect the condition number of A to be large. We have
$\|A\|_\infty = \max\{|1| + |2|, |1.0001| + |2|\} = 3.0001$, which would not be considered large. However,

$$A^{-1} = \begin{bmatrix} -10000 & 10000 \\ 5000.5 & -5000 \end{bmatrix},$$

so $\|A^{-1}\|_\infty = 20000$, and for the infinity norm, $K(A) = (20000)(3.0001) = 60002$. The size of the condition number for this example should certainly keep us from making hasty accuracy decisions based on the residual of an approximation.

Iterative Refinement Method:

Let us define : $\tilde{y} \approx x - \tilde{x}$, where $\tilde{y}$ is the approximate solution to the system $Ay = r$. In general, $\tilde{x} + \tilde{y}$ is a more accurate approximation to the solution of the linear system $Ax = b$ than the original approximation $\tilde{x}$. The method using this assumption is called iterative refinement or iterative improvement and consists of performing iterations on the system whose right-hand side is the residual vector for successive approximations until satisfactory accuracy results.

**

Let's solve the linear system (whose solutions are (1,1,1)):

$$\begin{bmatrix} 3.3330 & 15920 & -10.333 \\ 2.2220 & 16.710 & 9.6120 \\ 1.5611 & 5.1791 & 1.6852 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 15913 \\ 28.544 \\ 8.4254 \end{bmatrix}$$

using Gaussian elimination, we find:

$$\tilde{x}^{(1)} = (1.2001, 0.99991, 0.92538)^t$$

and the solution to $Ay = r^{(1)}$ to be:

$$\tilde{y}^{(1)} = (-0.20008, 8.9987 \times 10^{-5}, 0.074607)^t$$

According to Iterative Refinement Method,

$$\tilde{x}^{(2)} = \tilde{x}^{(1)} + \tilde{y}^{(1)} = (1.0000, 1.0000, 0.99999)^t$$

and the actual error in this approximation is $\|x - \tilde{x}^{(2)}\|_\infty = 1 \times 10^{-5}$.

Again we can compute,

$$r^{(2)} = b - A\tilde{x}^{(2)}$$

and solve the system

$$Ay^{(2)} = r^{(2)}$$

which gives:

$$\tilde{y}^{(2)} = (1.5002 \times 10^{-9}, 2.0951 \times 10^{-10}, 1.0000 \times 10^{-5})^t$$

Since $\|\tilde{y}(2)\|_\infty \leq 10^{-5}$
we conclude that $\tilde{x}(3) = \tilde{x}(2) + \tilde{y}(2) = (1.0000, 1.0000, 1.0000)^t$ is sufficiently accurate, which is certainly correct.

Preconditioned Conjugate Gradient Method

* The Conjugate Gradient Method is a direct method designed to solve an *nxn* positive definite linear system.
* It is generally inferior to Gaussian elimination with pivoting. Both methods require n steps to determine a solution, and the steps of the conjugate gradient method are more computationally expensive than those of Gaussian elimination.
* However, the conjugate gradient method is useful when employed as an iterative approximation method for solving large sparse systems with nonzero entries occurring in predictable patterns.
* These problems frequently arise in the solution of boundary-value problems. When the matrix has been preconditioned to make the calculations more effective, good results are obtained in only about $\sqrt{n}$ iterations. Employed in this way, the method is preferred over Gaussian elimination and the previously-discussed iterative methods.

Preconditioning :

* If the matrix A is ill-conditioned, the conjugate gradient method is highly susceptible to rounding errors. So, although the exact answer should be obtained in n steps, this is not usually the case.
* When preconditioning is used, the conjugate gradient method is not applied directly to the matrix A but to another positive definite matrix that a smaller condition number.
* The expectation is that this will reduce the rounding error when the method is applied. To maintain the positive definiteness of the resulting matrix, we need to multiply on each side by a nonsingular matrix. We will denote this matrix by C^{-1}, and consider

$$\tilde{A} = C^{-1}A(C^{-1})^t$$

with the hope that $\tilde{A}$ has a lower condition number than A.
* C can be chosen in a number of ways depending on the problem we are solving. One example for choosing C can be a Lower triangular Matrix defined by an incomplete Cholesky factorisation.

**Preconditioned Conjugate Gradient Method:

* We will use the inner product notation

$$< x, y >= x^t y$$

* The following result is a basic tool in the development of the conjugate gradient method.

$$< x, Ay >= (Ax)^t y = x^t A^t y = x^t Ay =< Ax, y > .$$

* The vector x^* is a solution to the positive definite linear system $Ax = b$ if and only if x^* produces the minimal value of

$$g(x) = x^t Ax - 2x^t b.$$

* To begin the conjugate gradient method, we choose x, an approximate solution to $Ax^* = b$, and $v = 0$, which gives a search direction in which to move away from x to improve the approximation. Let $r = b - Ax$ be the residual vector associated with x and

$$t = \frac{< v, b - Ax >}{< v, Av >} = \frac{< v, r >}{< v, Av >}.$$

** Here, t is the step length.

**

$$g(x+tv) =< x+tv, Ax+tAv > -2 < x+tv, b >$$

** Here, v is the search direction.

$$=< x, Ax > +t < v, Ax > +t < x, Av > +t^2 < v, Av > -2 < x, b > -2t < v, b >$$

$$=< x, Ax > -2 < x, b > +2t < v, Ax > -2t < v, b > +t^2 < v, Av >$$

so

$$g(x+tv) = g(x) - 2t < v, b-Ax > +t^2 < v, Av >$$

With x and v fixed, we can define the quadratic function h in t by

$$h(t) = g(x+tv).$$

Then h assumes a minimal value when $h^{'}(t) = 0$, because its t^2 coefficient, $< v, Av >$, is positive. Because

$$h^{'}(t) = -2 < v, b-Ax > +2t < v, Av >$$

the minimum occurs when

$$\hat{t} = \frac{< v, b-Ax >}{< v, Av >},$$

**

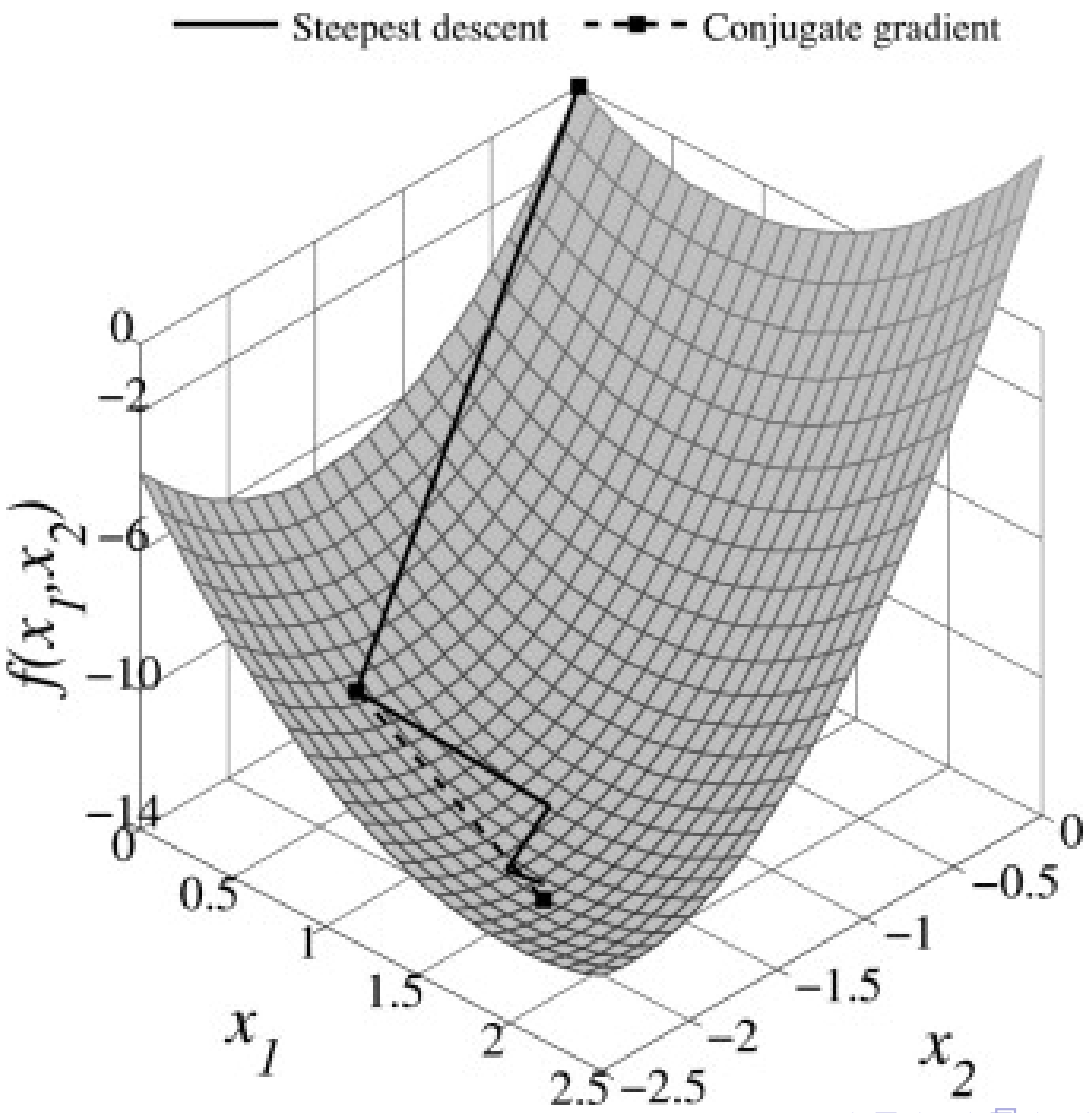
$$h(\hat{t}) = g(x + \hat{t}v)$$

$$= g(x) - 2\hat{t} < v, b - Ax > + \hat{t}^2 < v, Av >$$

$$= g(x) - 2\frac{< v, b - Ax >}{< v, Av >} < v, b - Ax > + (\frac{< v, b - Ax >}{< v, Av >})^2 < v, Av >$$

$$= g(x) - \frac{< v, b - Ax >^2}{< v, Av >}.$$

**


Steepest descent
Conjugate gradient
0
−2
−6
−10
−14
f(x1,x2)
0
0.5
1
1.5
2
2.5
x1
0
−0.5
−1
−1.5
−2
−2.5
x2

**

* From multivariable calculus, we know that the direction of greatest decrease in the value of $g(x)$ is the direction given by $-\nabla g(x)$; that is, in the direction of the residual r. The method that chooses

$$v(k+1) = r(k) = b - Ax(k)$$

is called the method of steepest descent. Although we will see in Section 10.4 that this method has merit for nonlinear systems and optimization problems, it is not used for linear systems because of slow convergence.

An alternative approach uses a set of nonzero direction vectors $\{v(1), \ldots, v(n)\}$ that satisfy

$$v(i), Av(j) = 0, \text{ if } i \neq j.$$

This is called an A-orthogonality condition, and the set of vectors $\{v(1), \ldots, v(n)\}$ is said to be A-orthogonal. It is not difficult to show that a set of A-orthogonal vectors associated with the positive definite matrix A is linearly independent. (See Exercise 13(a).) This set of search directions gives

$$t_k = \frac{v(k), b - Ax(k-1)}{v(k), Av(k)} = \frac{v(k), r(k-1)}{v(k), Av(k)}$$

and $x(k) = x(k-1) + t_k v(k)$.

The following theorem shows that this choice of search directions gives convergence in at most n steps, so as a direct method it produces the exact solution, assuming that the arithmetic is exact.

Conclusion:

The linear system $Ax = b$ with

$$A = \begin{bmatrix} 0.2 & 0.1 & 1 & 1 & 0 \\ 0.1 & 4 & -1 & 1 & -1 \\ 1 & -1 & 60 & 0 & -2 \\ 1 & 1 & 0 & 8 & 4 \\ 0 & -1 & -2 & 4 & 700 \end{bmatrix}$$

and

$$b = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$$

has the solution

$$x^* = (7.859713071, 0.4229264082, -0.07359223906, -0.5406430164, 0.01062616286)^t$$

Table 7.5 lists the results obtained by using the Jacobi, Gauss-Seidel, and SOR (with $\omega = 1.25$) iterative methods applied to the system with A with a tolerance of 0.01, as well as those when the Conjugate Gradient method is applied both in its unpreconditioned form and using the preconditioning matrix described in Example 3. The preconditioned conjugate gradient method not only gives the most accurate approximations, it also uses the smallest number of iterations.

**

- For Jacobi's Method (Tx+c form)-

$$x_1^k = -0.5x_2^{k-1} - 5x_2^{k-1} + 1$$

$$x_2^k = -0.025x_1^{k-1} + 0.25x_3^{k-1} - 0.25x_4^{k-1} + 0.25x_4^{k-1} + 2$$

$$x_3^k = -0.016x_1^{k-1} + 0.016x_2^{k-1} + 0.02x_5^{k-1} + 3$$

$$x_4^k = -0.125x_1^{k-1} - 0.125x_2^{k-1} - 0.5x_5^{k-1} + 4$$

$$x_5^k = 0.0014x_2^{k-1} + 0.0028x_3^{k-1} - 0.0056x_4^{k-1} + 5$$

**

- For Gauss-Seidel Method (Tx+c form)-

$$x_1^k = -0.5x_2^{k-1} - 5x_2^{k-1} + 1$$

$$x_2^k = -0.025x_1^k + 0.25x_3^{k-1} - 0.25x_4^{k-1} + 0.25x_4^{k-1} + 2$$

$$x_3^k = -0.016x_1^k + 0.016x_2^k + 0.02x_5^{k-1} + 3$$

$$x_4^k = -0.125x_1^k - 0.125x_2^k - 0.5x_5^{k-1} + 4$$

$$x_5^k = 0.0014x_2^k + 0.0028x_3^k - 0.0056x_4^k + 5$$

- For SOR Method -
  Value of ω choosen : 1.25

- Calculating the condition number :
  with,

$$A^{-1} = \begin{bmatrix} 17.241 & 0.041 & -0.286 & -2.166 & 0.011 \\ 0.041 & 0.259 & 0.003 & -0.037 & 0.0005 \\ -2.86 & 0.003 & 0.021 & 0.035 & 0.0005 \\ -2.166 & -0.037 & 0.035 & 0.401 & -0.002 \\ 0.011 & 0.0005 & -0.0001 & -0.0022 & 0.001 \end{bmatrix}$$

$$K(A) = \|A\| \cdot \|A^{-1}\| = (707) \cdot (19.745) = 13,959.715$$

*Thus, some appropriate preconditioning needs to be implemented before iterating.

Table 7.5

Method	Number of Iterations	$\mathbf{x}^{(k)}$	$\|\mathbf{x}^* - \mathbf{x}^{(k)}\|_\infty$
Jacobi	49	$(7.86277141, 0.42320802, -0.07348669, -0.53975964, 0.01062847)^t$	0.00305834
Gauss-Seidel	15	$(7.83525748, 0.42257868, -0.07319124, -0.53753055, 0.01060903)^t$	0.02445559
SOR ($\omega = 1.25$)	7	$(7.85152706, 0.42277371, -0.07348303, -0.53978369, 0.01062286)^t$	0.00818607
Conjugate Gradient	5	$(7.85341523, 0.42298677, -0.07347963, -0.53987920, 0.008628916)^t$	0.00629785
Conjugate Gradient (Preconditioned)	4	$(7.85968827, 0.42288329, -0.07359878, -0.54063200, 0.01064344)^t$	0.00009312

Thank You!