Iterative Techniques in Matrix Algebra

Relaxation Techniques for Solving Linear Systems

Numerical Analysis (9th Edition)
R L Burden & J D Faires

Outline

1. Residual Vectors & the Gauss-Seidel Method

Outline

Outline

Outline

1. Residual Vectors & the Gauss-Seidel Method
2. Relaxation Methods (including SOR)
3. Choosing the Optimal Value of ω
4. The SOR Algorithm

Outline

1. Residual Vectors & the Gauss-Seidel Method
2. Relaxation Methods (including SOR)
3. Choosing the Optimal Value of ω
4. The SOR Algorithm

Residual Vectors & the Gauss-Seidel Method

Motivation

- We have seen that the rate of convergence of an iterative technique depends on the spectral radius of the matrix associated with the method.

Residual Vectors & the Gauss-Seidel Method

Motivation

- We have seen that the rate of convergence of an iterative technique depends on the spectral radius of the matrix associated with the method.
- One way to select a procedure to accelerate convergence is to choose a method whose associated matrix has minimal spectral radius.

Residual Vectors & the Gauss-Seidel Method

Motivation

- We have seen that the rate of convergence of an iterative technique depends on the spectral radius of the matrix associated with the method.
- One way to select a procedure to accelerate convergence is to choose a method whose associated matrix has minimal spectral radius.
- We start by introducing 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.

Residual Vectors & the Gauss-Seidel Method

Motivation

- We have seen that the rate of convergence of an iterative technique depends on the spectral radius of the matrix associated with the method.
- One way to select a procedure to accelerate convergence is to choose a method whose associated matrix has minimal spectral radius.
- We start by introducing 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.
- The method makes use of the vector described in the following definition.

Residual Vectors & the Gauss-Seidel Method

Definition

Suppose $\tilde{\mathbf{x}} \in \mathbb{R}^n$ is an approximation to the solution of the linear system defined by

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

The residual vector for $\tilde{\mathbf{x}}$ with respect to this system is

$$\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$$

Residual Vectors & the Gauss-Seidel Method

Definition

Suppose $\tilde{\mathbf{x}} \in \mathbb{R}^n$ is an approximation to the solution of the linear system defined by

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

The residual vector for $\tilde{\mathbf{x}}$ with respect to this system is

$$\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$$

Comments

- A residual vector is associated with each calculation of an approximate component to the solution vector.

Residual Vectors & the Gauss-Seidel Method

Definition

Suppose $\tilde{\mathbf{x}} \in \mathbb{R}^n$ is an approximation to the solution of the linear system defined by

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

The residual vector for $\tilde{\mathbf{x}}$ with respect to this system is

$$\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$$

Comments

- A residual vector is associated with each calculation of an approximate component to the solution vector.
- The true objective is to generate a sequence of approximations that will cause the residual vectors to converge rapidly to zero.

Residual Vectors & the Gauss-Seidel Method

Looking at the Gauss-Seidel Method

Suppose we let

$$\mathbf{r}_i^{(k)} = (r_{1i}^{(k)}, r_{2i}^{(k)}, \ldots, r_{ni}^{(k)})^t$$

denote the residual vector for the Gauss-Seidel method

Residual Vectors & the Gauss-Seidel Method

Looking at the Gauss-Seidel Method

Suppose we let

$$\mathbf{r}_i^{(k)} = (r_{1i}^{(k)}, r_{2i}^{(k)}, \ldots, r_{ni}^{(k)})^t$$

denote the residual vector for the Gauss-Seidel method corresponding to the approximate solution vector $\mathbf{x}_i^{(k)}$ defined by

$$\mathbf{x}_i^{(k)} = (x_1^{(k)}, x_2^{(k)}, \ldots, x_{i-1}^{(k)}, x_i^{(k-1)}, \ldots, x_n^{(k-1)})^t$$

Residual Vectors & the Gauss-Seidel Method

Looking at the Gauss-Seidel Method

Suppose we let

$$\mathbf{r}_i^{(k)} = (r_{1i}^{(k)}, r_{2i}^{(k)}, \ldots, r_{ni}^{(k)})^t$$

denote the residual vector for the Gauss-Seidel method corresponding to the approximate solution vector $\mathbf{x}_i^{(k)}$ defined by

$$\mathbf{x}_i^{(k)} = (x_1^{(k)}, x_2^{(k)}, \ldots, x_{i-1}^{(k)}, x_i^{(k-1)}, \ldots, x_n^{(k-1)})^t$$

The m-th component of $\mathbf{r}_i^{(k)}$ is

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

Residual Vectors & the Gauss-Seidel Method

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

Looking at the Gauss-Seidel Method (Cont'd)

Equivalently, we can write $r_{mi}^{(k)}$ in the form:

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

for each $m = 1, 2, \ldots, n$.

Residual Vectors & the Gauss-Seidel Method

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

Looking at the Gauss-Seidel Method (Cont'd)

In particular, the ith component of $\mathbf{r}_i^{(k)}$ is

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

Residual Vectors & the Gauss-Seidel Method

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

Looking at the Gauss-Seidel Method (Cont'd)

In particular, the ith component of $\mathbf{r}_i^{(k)}$ is

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

so

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

Residual Vectors & the Gauss-Seidel Method

$$\text{(E)} \qquad a_{ii}x_i^{(k-1)} + r_{ii}^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}$$

Looking at the Gauss-Seidel Method (Cont'd)

Recall, however, that in the Gauss-Seidel method, $x_i^{(k)}$ is chosen to be

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

Residual Vectors & the Gauss-Seidel Method

$$\text{(E)} \qquad a_{ii}x_i^{(k-1)} + r_{ii}^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}$$

Looking at the Gauss-Seidel Method (Cont'd)

Recall, however, that in the Gauss-Seidel method, $x_i^{(k)}$ is chosen to be

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

so (E) can be rewritten as

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

Residual Vectors & the Gauss-Seidel Method

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

Looking at the Gauss-Seidel Method (Cont'd)

Consequently, the Gauss-Seidel method can be characterized as choosing $x_i^{(k)}$ to satisfy

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

Residual Vectors & the Gauss-Seidel Method

A 2nd Connection with Residual Vectors

- We can derive another connection between the residual vectors and the Gauss-Seidel technique.

Residual Vectors & the Gauss-Seidel Method

A 2nd Connection with Residual Vectors

- We can derive another connection between the residual vectors and the Gauss-Seidel technique.
- Consider the residual vector $\mathbf{r}_{i+1}^{(k)}$, associated with the vector $\mathbf{x}_{i+1}^{(k)} = (x_1^{(k)}, \ldots, x_i^{(k)}, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)})^t$.

Residual Vectors & the Gauss-Seidel Method

A 2nd Connection with Residual Vectors

- We can derive another connection between the residual vectors and the Gauss-Seidel technique.
- Consider the residual vector $\mathbf{r}_{i+1}^{(k)}$, associated with the vector $\mathbf{x}_{i+1}^{(k)} = (x_1^{(k)}, \ldots, x_i^{(k)}, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)})^t$.
- We have seen that the m-th component of $\mathbf{r}_i^{(k)}$ is

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

Residual Vectors & the Gauss-Seidel Method

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

A 2nd Connection with Residual Vectors (Cont'd)

Therefore, the ith component of $\mathbf{r}_{i+1}^{(k)}$ is

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

Residual Vectors & the Gauss-Seidel Method

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

A 2nd Connection with Residual Vectors (Cont'd)

Therefore, the ith component of $\mathbf{r}_{i+1}^{(k)}$ is

$$\begin{aligned} r_{i,i+1}^{(k)} &= b_i - \sum_{j=1}^{i} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \\ &= b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} - a_{ii} x_i^{(k)} \end{aligned}$$

Residual Vectors & the Gauss-Seidel Method

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

A 2nd Connection with Residual Vectors (Cont'd)

By the manner in which $x_i^{(k)}$ is defined in

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

we see that $r_{i,i+1}^{(k)} = 0$.

Residual Vectors & the Gauss-Seidel Method

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

A 2nd Connection with Residual Vectors (Cont'd)

By the manner in which $x_i^{(k)}$ is defined in

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

we see that $r_{i,i+1}^{(k)} = 0$. In a sense, then, the Gauss-Seidel technique is characterized by choosing each $x_{i+1}^{(k)}$ in such a way that the ith component of $\mathbf{r}_{i+1}^{(k)}$ is zero.

Outline

From Gauss-Seidel to Relaxation Methods

Reducing the Norm of the Residual Vector

- Choosing $x_{i+1}^{(k)}$ so that one coordinate of the residual vector is zero, however, is not necessarily the most efficient way to reduce the norm of the vector $\mathbf{r}_{i+1}^{(k)}$.

From Gauss-Seidel to Relaxation Methods

Reducing the Norm of the Residual Vector

- Choosing $x_{i+1}^{(k)}$ so that one coordinate of the residual vector is zero, however, is not necessarily the most efficient way to reduce the norm of the vector $\mathbf{r}_{i+1}^{(k)}$.
- If we modify the Gauss-Seidel procedure, as given by

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

From Gauss-Seidel to Relaxation Methods

Reducing the Norm of the Residual Vector

- Choosing $x_{i+1}^{(k)}$ so that one coordinate of the residual vector is zero, however, is not necessarily the most efficient way to reduce the norm of the vector $\mathbf{r}_{i+1}^{(k)}$.
- If we modify the Gauss-Seidel procedure, as given by

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

to

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

From Gauss-Seidel to Relaxation Methods

Reducing the Norm of the Residual Vector

- Choosing $x_{i+1}^{(k)}$ so that one coordinate of the residual vector is zero, however, is not necessarily the most efficient way to reduce the norm of the vector $\mathbf{r}_{i+1}^{(k)}$.
- If we modify the Gauss-Seidel procedure, as given by

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

to

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

then for certain choices of positive ω we can reduce the norm of the residual vector and obtain significantly faster convergence.

From Gauss-Seidel to Relaxation Methods

Introducing the SOR Method

- Methods involving

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

are called relaxation methods.

From Gauss-Seidel to Relaxation Methods

Introducing the SOR Method

- Methods involving

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

are called relaxation methods. For choices of ω with $0 < \omega < 1$, the procedures are called under-relaxation methods.

From Gauss-Seidel to Relaxation Methods

Introducing the SOR Method

- Methods involving

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

  are called relaxation methods. For choices of ω with $0 < \omega < 1$, the procedures are called under-relaxation methods.

- We will be interested in choices of ω with $1 < \omega$, and these are called over-relaxation methods.

From Gauss-Seidel to Relaxation Methods

Introducing the SOR Method

- Methods involving

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

  are called relaxation methods. For choices of ω with $0 < \omega < 1$, the procedures are called under-relaxation methods.
- We will be interested in choices of ω with $1 < \omega$, and these are called over-relaxation methods.
- They are used to accelerate the convergence for systems that are convergent by the Gauss-Seidel technique.

From Gauss-Seidel to Relaxation Methods

Introducing the SOR Method

- Methods involving
$$x_i^{(k)} = x_i^{(k-1)} + \omega \frac{r_{ii}^{(k)}}{a_{ii}}$$
are called relaxation methods. For choices of ω with $0 < \omega < 1$, the procedures are called under-relaxation methods.
- We will be interested in choices of ω with $1 < \omega$, and these are called over-relaxation methods.
- They are used to accelerate the convergence for systems that are convergent by the Gauss-Seidel technique.
- The methods are abbreviated SOR, for Successive Over-Relaxation, and are particularly useful for solving the linear systems that occur in the numerical solution of certain partial-differential equations.

The SOR Method

A More Computationally-Efficient Formulation

Note that by using the i-th component of $\mathbf{r}_i^{(k)}$ in the form

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

The SOR Method

A More Computationally-Efficient Formulation

Note that by using the i-th component of $\mathbf{r}_i^{(k)}$ in the form

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

we can reformulate the SOR equation

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

for calculation purposes

The SOR Method

A More Computationally-Efficient Formulation

Note that by using the i-th component of $\mathbf{r}_i^{(k)}$ in the form

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

we can reformulate the SOR equation

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

for calculation purposes as

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

The SOR Method

A More Computationally-Efficient Formulation (Cont'd)

To determine the matrix form of the SOR method, we rewrite

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

The SOR Method

A More Computationally-Efficient Formulation (Cont'd)

To determine the matrix form of the SOR method, we rewrite

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

as

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

The SOR Method

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

A More Computationally-Efficient Formulation (Cont'd)

In vector form, we therefore have

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

The SOR Method

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

A More Computationally-Efficient Formulation (Cont'd)

In vector form, we therefore have

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

from which we obtain:

The SOR Method

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

The SOR Method

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

The SOR Method

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

Letting

$$T_\omega = (D - \omega L)^{-1}[(1-\omega)D + \omega U]$$

The SOR Method

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

Letting

$$\begin{aligned} T_\omega &= (D - \omega L)^{-1}[(1 - \omega)D + \omega U] \\ \text{and} \qquad \mathbf{c}_\omega &= \omega(D - \omega L)^{-1}\mathbf{b} \end{aligned}$$

The SOR Method

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

Letting

$$T_\omega = (D-\omega L)^{-1}[(1-\omega)D+\omega U]$$

and

$$\mathbf{c}_\omega = \omega(D-\omega L)^{-1}\mathbf{b}$$

gives the SOR technique the form

$$\mathbf{x}^{(k)} = T_\omega \mathbf{x}^{(k-1)} + \mathbf{c}_\omega$$

The SOR Method

Example

- The linear system $A\mathbf{x} = \mathbf{b}$ given by

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

has the solution $(3, 4, -5)^t$.

The SOR Method

Example

- The linear system $A\mathbf{x} = \mathbf{b}$ given by

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

  has the solution $(3, 4, -5)^t$.
- Compare the iterations from the Gauss-Seidel method and the SOR method with $\omega = 1.25$ using $\mathbf{x}^{(0)} = (1, 1, 1)^t$ for both methods.

The SOR Method

Solution (1/3)

The SOR Method

Solution (1/3)

For each $k = 1, 2, \ldots,$ the equations for the Gauss-Seidel method are

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

The SOR Method

Solution (1/3)

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 SOR Method: Solution (2/3)

Gauss-Seidel Iterations

k	0	1	2	3	$\cdots$	7
$x_1^{(k)}$	1	5.250000	3.1406250	3.0878906		3.0134110
$x_2^{(k)}$	1	3.812500	3.8828125	3.9267578		3.9888241
$x_3^{(k)}$	1	-5.046875	-5.0292969	-5.0183105		-5.0027940

The SOR Method: Solution (2/3)

Gauss-Seidel Iterations

k	0	1	2	3	$\cdots$	7
$x_1^{(k)}$	1	5.250000	3.1406250	3.0878906		3.0134110
$x_2^{(k)}$	1	3.812500	3.8828125	3.9267578		3.9888241
$x_3^{(k)}$	1	−5.046875	−5.0292969	−5.0183105		−5.0027940

SOR Iterations ($\omega = 1.25$)

k	0	1	2	3	$\cdots$	7
$x_1^{(k)}$	1	6.312500	2.6223145	3.1333027		3.0000498
$x_2^{(k)}$	1	3.5195313	3.9585266	4.0102646		4.0002586
$x_3^{(k)}$	1	−6.6501465	−4.6004238	−5.0966863		−5.0003486

The SOR Method

Solution (3/3)

The SOR Method

Solution (3/3)

For the iterates to be accurate to 7 decimal places,

- the Gauss-Seidel method requires 34 iterations,

The SOR Method

Solution (3/3)

For the iterates to be accurate to 7 decimal places,

- the Gauss-Seidel method requires 34 iterations,
- as opposed to 14 iterations for the SOR method with $\omega = 1.25$.

Outline

Choosing the Optimal Value of ω

- An obvious question to ask is how the appropriate value of ω is chosen when the SOR method is used?

Choosing the Optimal Value of ω

- An obvious question to ask is how the appropriate value of ω is chosen when the SOR method is used?
- Although no complete answer to this question is known for the general $n \times n$ linear system, the following results can be used in certain important situations.

Choosing the Optimal Value of ω

Theorem (Kahan)

If $a_{ii} \neq 0$, for each $i = 1, 2, \ldots, n$, then $\rho(T_\omega) \geq |\omega - 1|$. This implies that the SOR method can converge only if $0 < \omega < 2$.

Choosing the Optimal Value of ω

Theorem (Kahan)

If $a_{ii} \neq 0$, for each $i = 1, 2, \ldots, n$, then $\rho(T_\omega) \geq |\omega - 1|$. This implies that the SOR method can converge only if $0 < \omega < 2$.

The proof of this theorem is considered in Exercise 9, Chapter 7 of Burden R. L. & Faires J. D., Numerical Analysis, 9th Ed., Cengage, 2011.

Choosing the Optimal Value of ω

Theorem (Kahan)

If $a_{ii} \neq 0$, for each $i = 1, 2, \ldots, n$, then $\rho(T_\omega) \geq |\omega - 1|$. This implies that the SOR method can converge only if $0 < \omega < 2$.

The proof of this theorem is considered in Exercise 9, Chapter 7 of Burden R. L. & Faires J. D., Numerical Analysis, 9th Ed., Cengage, 2011.

Theorem (Ostrowski-Reich)

If A is a positive definite matrix and $0 < \omega < 2$, then the SOR method converges for any choice of initial approximate vector $\mathbf{x}^{(0)}$.

Choosing the Optimal Value of ω

Theorem (Kahan)

If $a_{ii} \neq 0$, for each $i = 1, 2, \ldots, n$, then $\rho(T_\omega) \geq |\omega - 1|$. This implies that the SOR method can converge only if $0 < \omega < 2$.

The proof of this theorem is considered in Exercise 9, Chapter 7 of Burden R. L. & Faires J. D., Numerical Analysis, 9th Ed., Cengage, 2011.

Theorem (Ostrowski-Reich)

If A is a positive definite matrix and $0 < \omega < 2$, then the SOR method converges for any choice of initial approximate vector $\mathbf{x}^{(0)}$.

The proof of this theorem can be found in Ortega, J. M., Numerical Analysis; A Second Course, Academic Press, New York, 1972, 201 pp.

Choosing the Optimal Value of ω

Theorem

If A is positive definite and tridiagonal, then $\rho(T_g) = [\rho(T_j)]^2 < 1$, and the optimal choice of ω for the SOR method is

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_j)]^2}}$$

Choosing the Optimal Value of ω

Theorem

If A is positive definite and tridiagonal, then $\rho(T_g) = [\rho(T_j)]^2 < 1$, and the optimal choice of ω for the SOR method is

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_j)]^2}}$$

With this choice of ω, we have $\rho(T_\omega) = \omega - 1$.

Choosing the Optimal Value of ω

Theorem

If A is positive definite and tridiagonal, then $\rho(T_g) = [\rho(T_j)]^2 < 1$, and the optimal choice of ω for the SOR method is

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_j)]^2}}$$

With this choice of ω, we have $\rho(T_\omega) = \omega - 1$.

The proof of this theorem can be found in Ortega, J. M., Numerical Analysis; A Second Course, Academic Press, New York, 1972, 201 pp.

The SOR Method

Example

Find the optimal choice of ω for the SOR method for the matrix

$$A = \begin{bmatrix} 4 & 3 & 0 \\ 3 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}$$

The SOR Method

Solution (1/3)

- This matrix is clearly tridiagonal, so we can apply the result in the SOR theorem if we can also show that it is positive definite.

The SOR Method

Solution (1/3)

- This matrix is clearly tridiagonal, so we can apply the result in the SOR theorem if we can also show that it is positive definite.
- Because the matrix is symmetric, the theory tells us that it is positive definite if and only if all its leading principle submatrices has a positive determinant.

The SOR Method

Solution (1/3)

- This matrix is clearly tridiagonal, so we can apply the result in the SOR theorem if we can also show that it is positive definite.
- Because the matrix is symmetric, the theory tells us that it is positive definite if and only if all its leading principle submatrices has a positive determinant.
- This is easily seen to be the case because

$$\det(A) = 24, \quad \det\left(\begin{bmatrix} 4 & 3 \\ 3 & 4 \end{bmatrix}\right) = 7 \quad \text{and} \quad \det\left([4]\right) = 4$$

The SOR Method

Solution (2/3)

We compute

$$T_j = D^{-1}(L + U)$$

The SOR Method

Solution (2/3)

We compute

$$\begin{aligned} T_j &= D^{-1}(L+U) \\ &= \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 0 & -3 & 0 \\ -3 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \end{aligned}$$

The SOR Method

Solution (2/3)

We compute

$$
\begin{aligned}
T_j &= D^{-1}(L+U) \\
&= \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 0 & -3 & 0 \\ -3 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & -0.75 & 0 \\ -0.75 & 0 & 0.25 \\ 0 & 0.25 & 0 \end{bmatrix}
\end{aligned}
$$

The SOR Method

Solution (2/3)

We compute

$$\begin{aligned} T_j &= D^{-1}(L+U) \\ &= \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 0 & -3 & 0 \\ -3 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -0.75 & 0 \\ -0.75 & 0 & 0.25 \\ 0 & 0.25 & 0 \end{bmatrix} \end{aligned}$$

so that

$$T_j - \lambda I = \begin{bmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{bmatrix}$$

The SOR Method

Solution (3/3)

Therefore

$$\det(T_j - \lambda I) = \begin{vmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{vmatrix}$$

The SOR Method

Solution (3/3)

Therefore

$$\det(T_j - \lambda I) = \begin{vmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{vmatrix} = -\lambda(\lambda^2 - 0.625)$$

The SOR Method

Solution (3/3)

Therefore

$$\det(T_j - \lambda I) = \begin{vmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{vmatrix} = -\lambda(\lambda^2 - 0.625)$$

Thus

$$\rho(T_j) = \sqrt{0.625}$$

The SOR Method

Solution (3/3)

Therefore

$$\det(T_j - \lambda I) = \begin{vmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{vmatrix} = -\lambda(\lambda^2 - 0.625)$$

Thus

$$\rho(T_j) = \sqrt{0.625}$$

and

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_j)]^2}} = \frac{2}{1 + \sqrt{1 - 0.625}} \approx 1.24.$$

The SOR Method

Solution (3/3)

Therefore

$$\det(T_j - \lambda I) = \begin{vmatrix} -\lambda & -0.75 & 0 \\ -0.75 & -\lambda & 0.25 \\ 0 & 0.25 & -\lambda \end{vmatrix} = -\lambda(\lambda^2 - 0.625)$$

Thus

$$\rho(T_j) = \sqrt{0.625}$$

and

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_j)]^2}} = \frac{2}{1 + \sqrt{1 - 0.625}} \approx 1.24.$$

This explains the rapid convergence obtained in the last example when using $\omega = 1.25$.

Outline

The SOR Algorithm (1/2)

To solve

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

given the parameter ω and an initial approximation $\mathbf{x}^{(0)}$:

The SOR Algorithm (1/2)

To solve

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

given the parameter ω and an initial approximation $\mathbf{x}^{(0)}$:

INPUT the number of equations and unknowns n;
the entries a_{ij}, $1 \leq i, j \leq n$, of the matrix A;
the entries b_i, $1 \leq i \leq n$, of $\mathbf{b}$;
the entries XO_i, $1 \leq i \leq n$, of $\mathbf{XO} = \mathbf{x}^{(0)}$;
the parameter ω; tolerance TOL;
maximum number of iterations N.

OUTPUT the approximate solution $x_1, \ldots, x_n$ or a message
that the number of iterations was exceeded.

The SOR Algorithm (2/2)

Step 1 Set $k = 1$

Step 2 While ($k \leq N$) do Steps 3–6:

The SOR Algorithm (2/2)

Step 1 Set $k = 1$
Step 2 While ($k \leq N$) do Steps 3–6:

Step 3 For $i = 1, \ldots, n$

$$\text{set } x_i = (1-\omega)XO_i + \frac{1}{a_{ii}}\left[\omega\left(-\sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}XO_j + b_i\right)\right]$$

The SOR Algorithm (2/2)

Step 1 Set $k = 1$

Step 2 While ($k \leq N$) do Steps 3–6:

Step 3 For $i = 1, \ldots, n$

$$\text{set } x_i = (1-\omega)XO_i + \frac{1}{a_{ii}}\left[\omega\left(-\sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}XO_j + b_i\right)\right]$$

Step 4 If $||\mathbf{x} - \mathbf{XO}|| < TOL$ then OUTPUT $(x_1, \ldots, x_n)$

STOP (*The procedure was successful*)

The SOR Algorithm (2/2)

Step 1 Set $k = 1$

Step 2 While ($k \leq N$) do Steps 3–6:

Step 3 For $i = 1, \ldots, n$

$$\text{set } x_i = (1-\omega)XO_i + \frac{1}{a_{ii}}\left[\omega\left(-\sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}XO_j + b_i\right)\right]$$

Step 4 If $||\mathbf{x} - \mathbf{XO}|| < TOL$ then OUTPUT $(x_1, \ldots, x_n)$
STOP (*The procedure was successful*)

Step 5 Set $k = k + 1$

The SOR Algorithm (2/2)

Step 1 Set $k = 1$

Step 2 While ($k \leq N$) do Steps 3–6:

Step 3 For $i = 1, \ldots, n$

$$\text{set } x_i = (1-\omega)XO_i + \frac{1}{a_{ii}}\left[\omega\left(-\sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}XO_j + b_i\right)\right]$$

Step 4 If $||\mathbf{x} - \mathbf{XO}|| < TOL$ then OUTPUT $(x_1, \ldots, x_n)$
STOP (*The procedure was successful*)

Step 5 Set $k = k + 1$

Step 6 For $i = 1, \ldots, n$ set $XO_i = x_i$

The SOR Algorithm (2/2)

Step 1 Set $k = 1$
Step 2 While ($k \leq N$) do Steps 3–6:

Step 3 For $i = 1, \ldots, n$

$$\text{set } x_i = (1-\omega)XO_i + \frac{1}{a_{ii}}\left[\omega\left(-\sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}XO_j + b_i\right)\right]$$

Step 4 If $||\mathbf{x} - \mathbf{XO}|| < TOL$ then OUTPUT $(x_1, \ldots, x_n)$
STOP (*The procedure was successful*)
Step 5 Set $k = k + 1$
Step 6 For $i = 1, \ldots, n$ set $XO_i = x_i$

Step 7 OUTPUT ('Maximum number of iterations exceeded')
STOP (*The procedure was successful*)

Questions?