Theorem

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 $0 < \omega < 2$.

Theorem

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

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 $\rho(T_\omega) = \omega - 1$.

Example

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

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

Matrix A is tridiagonal. Also, the matrix is symmetric and,

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

Thus, this matrix is positive definite.

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

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

so

$$\det(T_j - \lambda I) = -\lambda(\lambda^2 - 0.625)$$

Thus

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

and

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

HOMEWORK 10:

Write a program to solve the system Ax=b with the SOR method. A is a strictly diagonally dominant matrix with dimension n. For n=100 and Matrix A with 10s on its main diagonal and 1s on other diagonals and vector b with all elements equal 1, via an iterative process and computing CPU time estimate the optimum relaxation parameter. The relative Euclidean norm must be less than 1E-6 for convergence.

Solving Nonlinear Systems of Equations

A system of nonlinear equations has the form

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ \vdots \qquad & \quad \vdots \\ f_n(x_1, x_2, \ldots, x_n) &= 0 \end{aligned} \tag{10.1}$$

This system can also be represented as,

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}.$$

where,

$$\mathbf{F}(x_1, x_2, \ldots, x_n) = (f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_n(x_1, x_2, \ldots, x_n))^t$$

The functions $f_1, f_2, \ldots, f_n$ are called the **coordinate functions** of $\mathbf{F}$.

Fixed Point Method

Definition

A function $\mathbf{G}$ from $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$ has a **fixed point** at $\mathbf{p} \in D$ if $\mathbf{G}(\mathbf{p}) = \mathbf{p}$.

Example

Place the nonlinear system

$$3x_1 - \cos(x_2x_3) - \frac{1}{2} = 0,$$

$$x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 = 0,$$

$$e^{-x_1x_2} + 20x_3 + \frac{10\pi - 3}{3} = 0$$

in a fixed-point form $\mathbf{x} = \mathbf{G}(\mathbf{x})$ by solving the ith equation for x_i, and iterate starting with $\mathbf{x}^{(0)} = (0.1, 0.1, -0.1)^t$ until accuracy within 10^{-5} in the l_∞ norm is obtained.

Solution Solving the ith equation for x_i gives the fixed-point problem

$$x_1 = \frac{1}{3}\cos(x_2x_3) + \frac{1}{6},$$

$$x_2 = \frac{1}{9}\sqrt{x_1^2 + \sin x_3 + 1.06} - 0.1,$$

$$x_3 = -\frac{1}{20}e^{-x_1x_2} - \frac{10\pi - 3}{60}.$$

Let $\mathbf{G} : \mathbb{R}^3 \to \mathbb{R}^3$ be defined by $\mathbf{G}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), g_3(\mathbf{x}))^t$, where

$$g_1(x_1, x_2, x_3) = \frac{1}{3}\cos(x_2 x_3) + \frac{1}{6},$$

$$g_2(x_1, x_2, x_3) = \frac{1}{9}\sqrt{x_1^2 + \sin x_3 + 1.06} - 0.1,$$

$$g_3(x_1, x_2, x_3) = -\frac{1}{20}e^{-x_1 x_2} - \frac{10\pi - 3}{60}.$$

To approximate the fixed point $\mathbf{p}$, we choose $\mathbf{x}^{(0)} = (0.1, 0.1, -0.1)^t$. The sequence of vectors generated by

$$x_1^{(k)} = \frac{1}{3}\cos x_2^{(k-1)} x_3^{(k-1)} + \frac{1}{6},$$

$$x_2^{(k)} = \frac{1}{9}\sqrt{\left(x_1^{(k-1)}\right)^2 + \sin x_3^{(k-1)} + 1.06} - 0.1$$

$$x_3^{(k)} = -\frac{1}{20}e^{-x_1^{(k-1)} x_2^{(k-1)}} - \frac{10\pi - 3}{60}$$

Converges to the solution of the system. The results in the following Table were generated until,

$$\left\| \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)} \right\|_\infty < 10^{-5}$$

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\left\| \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)} \right\|_\infty$
0	0.10000000	0.10000000	−0.10000000	
1	0.49998333	0.00944115	−0.52310127	0.423
2	0.49999593	0.00002557	−0.52336331	9.4×10^{-3}
3	0.50000000	0.00001234	−0.52359814	2.3×10^{-4}
4	0.50000000	0.00000003	−0.52359847	1.2×10^{-5}
5	0.50000000	0.00000002	−0.52359877	3.1×10^{-7}

Accelerating Convergence

One way to accelerate convergence is to use $x_1^{(k)}, \ldots, x_{i-1}^{(k)}$ instead of $x_1^{(k-1)}, \ldots, x_{i-1}^{(k-1)}$ to compute $x_i^{(k)}$, The component equations for the problem in the example then become

$$x_1^{(k)} = \frac{1}{3}\cos\left(x_2^{(k-1)}x_3^{(k-1)}\right) + \frac{1}{6},$$

$$x_2^{(k)} = \frac{1}{9}\sqrt{\left(x_1^{(k)}\right)^2 + \sin x_3^{(k-1)} + 1.06} - 0.1,$$

$$x_3^{(k)} = -\frac{1}{20}e^{-x_1^{(k)}x_2^{(k)}} - \frac{10\pi - 3}{60}.$$

With $\mathbf{x}^{(0)} = (0.1, 0.1, -0.1)^t$, the results of these calculations are listed in

the following table,

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|_\infty$
0	0.10000000	0.10000000	−0.10000000	
1	0.49998333	0.02222979	−0.52304613	0.423
2	0.49997747	0.00002815	−0.52359807	2.2×10^{-2}
3	0.50000000	0.00000004	−0.52359877	2.8×10^{-5}
4	0.50000000	0.00000000	−0.52359877	3.8×10^{-8}

Illustration

The equation $x^3 + 4x^2 - 10 = 0$ has a unique root in $[1, 2]$. There are many ways to change the equation to the fixed-point form $x = g(x)$,

(a) $x = g_1(x) = x - x^3 - 4x^2 + 10$

(b) $x = g_2(x) = \left(\frac{10}{x} - 4x\right)^{1/2}$

(c) $x = g_3(x) = \frac{1}{2}(10 - x^3)^{1/2}$

(d) $x = g_4(x) = \left(\frac{10}{4 + x}\right)^{1/2}$

(e) $x = g_5(x) = x - \frac{x^3 + 4x^2 - 10}{3x^2 + 8x}$

With initial guess p_0 =1.5, the following results were obtained for different choices of g,

n	(a)	(b)	(c)	(d)	(e)
0	1.5	1.5	1.5	1.5	1.5
1	−0.875	0.8165	1.286953768	1.348399725	1.373333333
2	6.732	2.9969	1.402540804	1.367376372	1.365262015
3	−469.7	$(-8.65)^{1/2}$	1.345458374	1.364957015	1.365230014
4	1.03×10^8		1.375170253	1.365264748	1.365230013
5			1.360094193	1.365225594	
6			1.367846968	1.365230576	
7			1.363887004	1.365229942	
8			1.365916734	1.365230022	
9			1.364878217	1.365230012	
10			1.365410062	1.365230014	
15			1.365223680	1.365230013	
20			1.365230236		
25			1.365230006		
30			1.365230013		

Only cases (c), (d) and (e) lead to the correct answer. But, the rate of the convergence is different. The case (e) has the fastest convergence.

- The convergence rate of the Fixed point method is dependent on the function G and also on the initial guess.

Newton's Method

One-dimensional case

Suppose that $f \in C^2[a, b]$. Let $p_0 \in [a, b]$ be an approximation to p such that $f'(p_0) \neq 0$ and $|p - p_0|$ is "small." Consider the first Taylor polynomial for $f(x)$ expanded about p_0 and evaluated at $x = p$. (p is the root of f on [a,b])

$$f(p) = f(p_0) + (p - p_0)f'(p_0) + \frac{(p - p_0)^2}{2} f''(\xi(p))$$

f(p)=0. So,

$$0 = f(p_0) + (p - p_0)f'(p_0) + \frac{(p - p_0)^2}{2} f''(\xi(p))$$

$|p - p_0|$ is "small." So,

$$0 \approx f(p_0) + (p - p_0) f'(p_0)$$

Solving for p gives

$$p \approx p_0 - \frac{f(p_0)}{f'(p_0)} \equiv p_1$$

This sets the stage for Newton's method,

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}, \quad \text{for } n \geq 1$$

Illustration

- Newton's method is a fixed point iteration technique $p_n = g(p_{n-1})$,

$$g(p_{n-1}) = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}, \quad \text{for } n \geq 1$$

- Provided that p_0 is a good initial approximation, this fixed point method gives rapid convergence.
- This method is most effective, when f' is bounded away from zero near p.

Newton's Method

n-dimensional case

For solving the nonlinear system,

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}.$$

where,

$$\mathbf{F}(x_1, x_2, \ldots, x_n) = (f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_n(x_1, x_2, \ldots, x_n))^t$$

Newton's method is extended to n-dimensional case. In this case the Function G is defined by,

$$\mathbf{G}(\mathbf{x}) = \mathbf{x} - J(\mathbf{x})^{-1}\mathbf{F}(\mathbf{x})$$

where the Jacobian matrix is,

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \frac{\partial f_2}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_2}{\partial x_n}(\mathbf{x}) \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \frac{\partial f_n}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}) \end{bmatrix}$$

Then, fixed point iteration procedure is applied as,

$$\mathbf{x}^{(k)} = \mathbf{G}\left(\mathbf{x}^{(k-1)}\right) = \mathbf{x}^{(k-1)} - J\left(\mathbf{x}^{(k-1)}\right)^{-1}\mathbf{F}\left(\mathbf{x}^{(k-1)}\right)$$

- Newton's method gives a rapid convergence, provided that a sufficiently accurate starting value is known and that $J(\mathbf{p})^{-1}$ exists.

- A weakness of Newton's method is the need to compute and invert the matrix $J(\mathrm{x})$ at each step. In practice the Newton's method is performed in the following two-step manner,

$$J(\mathbf{x}^{(k-1)})\mathbf{y} = -\mathbf{F}(\mathbf{x}^{(k-1)})$$

$$\mathrm{x}^{(\mathrm{k})} = \mathrm{x}^{(\mathrm{k}-1)} + \mathrm{y}$$