MAP 2220 – FUNDAMENTOS DE ANÁLISE NUMÉRICA

2º Semestre - 2017

Prof. Dr. Luis Carlos de Castro Santos

lsantos@ime.usp.br/lccs13@yahoo.com

CHAPTER 2

Solutions of Equations in One Variable

2.2 Fixed-Point Iteration

Definition 2.2 The number p is a **fixed point** for a given function g if $g(p) = p$.

- Given a root-finding problem $f(p) = 0$, we can define functions g with a fixed point at p in a number of ways, for example, as

$$g(x) = x - f(x) \quad \text{or as} \quad g(x) = x + 3f(x).$$

- Conversely, if the function g has a fixed point at p, then the function defined by

$$f(x) = x - g(x)$$

has a zero at p.

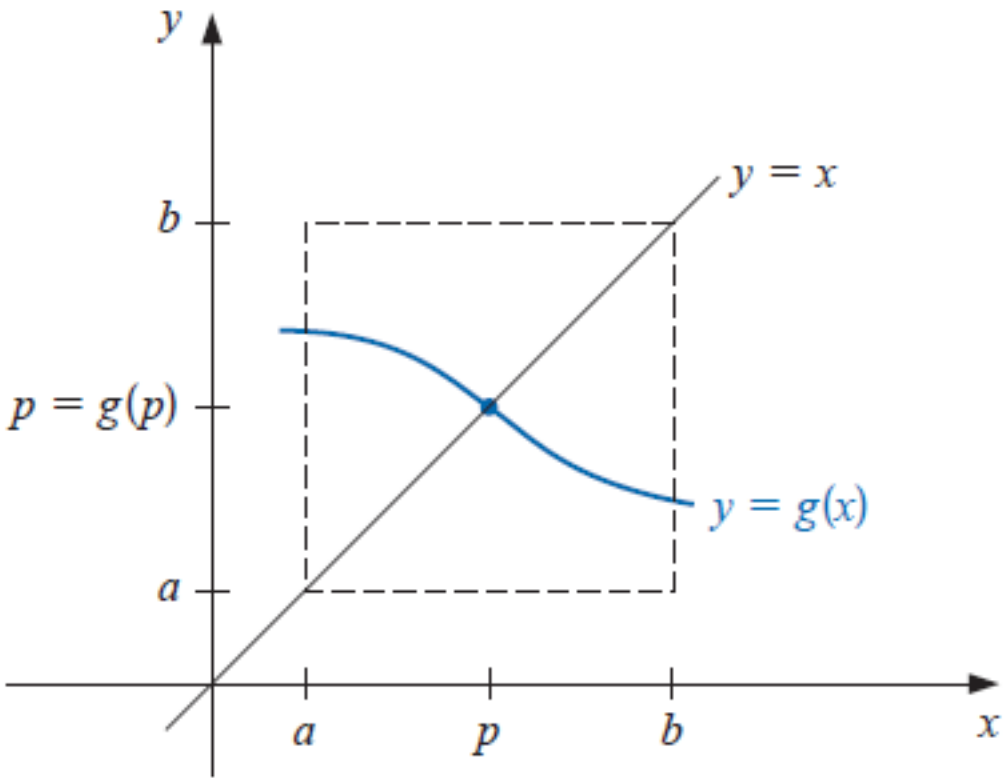
Theorem 2.3

(i) If $g \in C[a, b]$ and $g(x) \in [a, b]$ for all $x \in [a, b]$, then g has at least one fixed point in $[a, b]$.

(ii) If, in addition, $g'(x)$ exists on (a, b) and a positive constant $k < 1$ exists with

$$|g'(x)| \leq k, \quad \text{for all } x \in (a, b),$$

then there is exactly one fixed point in $[a, b]$. (See Figure 2.4.) ■



Proof

(i) If $g(a) = a$ or $g(b) = b$, then g has a fixed point at an endpoint. If not, then $g(a) > a$ and $g(b) < b$. The function $h(x) = g(x) - x$ is continuous on $[a, b]$, with

$$h(a) = g(a) - a > 0 \quad \text{and} \quad h(b) = g(b) - b < 0.$$

The Intermediate Value Theorem implies that there exists $p \in (a, b)$ for which $h(p) = 0$. This number p is a fixed point for g because

$$0 = h(p) = g(p) - p \quad \text{implies that} \quad g(p) = p.$$

(ii) Suppose, in addition, that $|g'(x)| \leq k < 1$ and that p and q are both fixed points in $[a, b]$. If $p \neq q$, then the Mean Value Theorem implies that a number ξ exists between p and q, and hence in $[a, b]$, with

$$\frac{g(p) - g(q)}{p - q} = g'(\xi).$$

Thus

$$|p - q| = |g(p) - g(q)| = |g'(\xi)||p - q| \leq k|p - q| < |p - q|,$$

which is a contradiction. This contradiction must come from the only supposition, $p \neq q$. Hence, $p = q$ and the fixed point in $[a, b]$ is unique. ■ ■ ■

Theorem 2.4 **(Fixed-Point Theorem)**

Let $g \in C[a,b]$ be such that $g(x) \in [a,b]$, for all x in $[a,b]$. Suppose, in addition, that g' exists on (a,b) and that a constant $0 < k < 1$ exists with

$$|g'(x)| \leq k, \quad \text{for all } x \in (a,b).$$

Then for any number p_0 in $[a,b]$, the sequence defined by

$$p_n = g(p_{n-1}), \quad n \geq 1,$$

converges to the unique fixed point p in $[a,b]$. ■

Proof Theorem 2.3 implies that a unique point p exists in $[a,b]$ with $g(p) = p$. Since g maps $[a,b]$ into itself, the sequence $\{p_n\}_{n=0}^{\infty}$ is defined for all $n \geq 0$, and $p_n \in [a,b]$ for all n. Using the fact that $|g'(x)| \leq k$ and the Mean Value Theorem 1.8, we have, for each n,

$$|p_n - p| = |g(p_{n-1}) - g(p)| = |g'(\xi_n)||p_{n-1} - p| \leq k|p_{n-1} - p|,$$

where $\xi_n \in (a,b)$. Applying this inequality inductively gives

$$|p_n - p| \leq k|p_{n-1} - p| \leq k^2|p_{n-2} - p| \leq \cdots \leq k^n|p_0 - p|. \tag{2.4}$$

Since $0 < k < 1$, we have $\lim_{n\to\infty} k^n = 0$ and

$$\lim_{n\to\infty} |p_n - p| \leq \lim_{n\to\infty} k^n|p_0 - p| = 0.$$

Hence $\{p_n\}_{n=0}^{\infty}$ converges to p. ■ ■ ■

Corollary 2.5 If g satisfies the hypotheses of Theorem 2.4, then bounds for the error involved in using p_n to approximate p are given by

$$|p_n - p| \leq k^n \max\{p_0 - a, b - p_0\} \tag{2.5}$$

and

$$|p_n - p| \leq \frac{k^n}{1-k}|p_1 - p_0|, \quad \text{for all} \quad n \geq 1. \tag{2.6}$$

■

Proof Because $p \in [a, b]$, the first bound follows from Inequality (2.4):

$$|p_n - p| \leq k^n |p_0 - p| \leq k^n \max\{p_0 - a, b - p_0\}.$$

For $n \geq 1$, the procedure used in the proof of Theorem 2.4 implies that

$$|p_{n+1} - p_n| = |g(p_n) - g(p_{n-1})| \leq k|p_n - p_{n-1}| \leq \cdots \leq k^n |p_1 - p_0|.$$

Thus for $m > n \geq 1$,

$$\begin{aligned}
|p_m - p_n| &= |p_m - p_{m-1} + p_{m-1} - \cdots + p_{n+1} - p_n| \\
&\leq |p_m - p_{m-1}| + |p_{m-1} - p_{m-2}| + \cdots + |p_{n+1} - p_n| \\
&\leq k^{m-1}|p_1 - p_0| + k^{m-2}|p_1 - p_0| + \cdots + k^n|p_1 - p_0| \\
&= k^n|p_1 - p_0|\left(1 + k + k^2 + \cdots + k^{m-n-1}\right).
\end{aligned}$$

By Theorem 2.3, $\lim_{m\to\infty} p_m = p$, so

$$|p - p_n| = \lim_{m\to\infty} |p_m - p_n| \leq \lim_{m\to\infty} k^n|p_1 - p_0| \sum_{i=0}^{m-n-1} k^i \leq k^n|p_1 - p_0| \sum_{i=0}^{\infty} k^i.$$

But $\sum_{i=0}^{\infty} k^i$ is a geometric series with ratio k and $0 < k < 1$. This sequence converges to $1/(1-k)$, which gives the second bound:

$$|p - p_n| \leq \frac{k^n}{1-k}|p_1 - p_0|.$$

■ ■ ■

2.3 Newton's Method and Its Extensions

Isaac Newton (1641–1727) was one of the most brilliant scientists of all time. The late 17th century was a vibrant period for science and mathematics and Newton's work touched nearly every aspect of mathematics. His method for solving was introduced to find a root of the equation $y^3 - 2y - 5 = 0$. Although he demonstrated the method only for polynomials, it is clear that he realized its broader applications.

Newton's Method

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 $x = p$.

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

where $\xi(p)$ lies between p and p_0. Since $f(p) = 0$, this equation gives

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

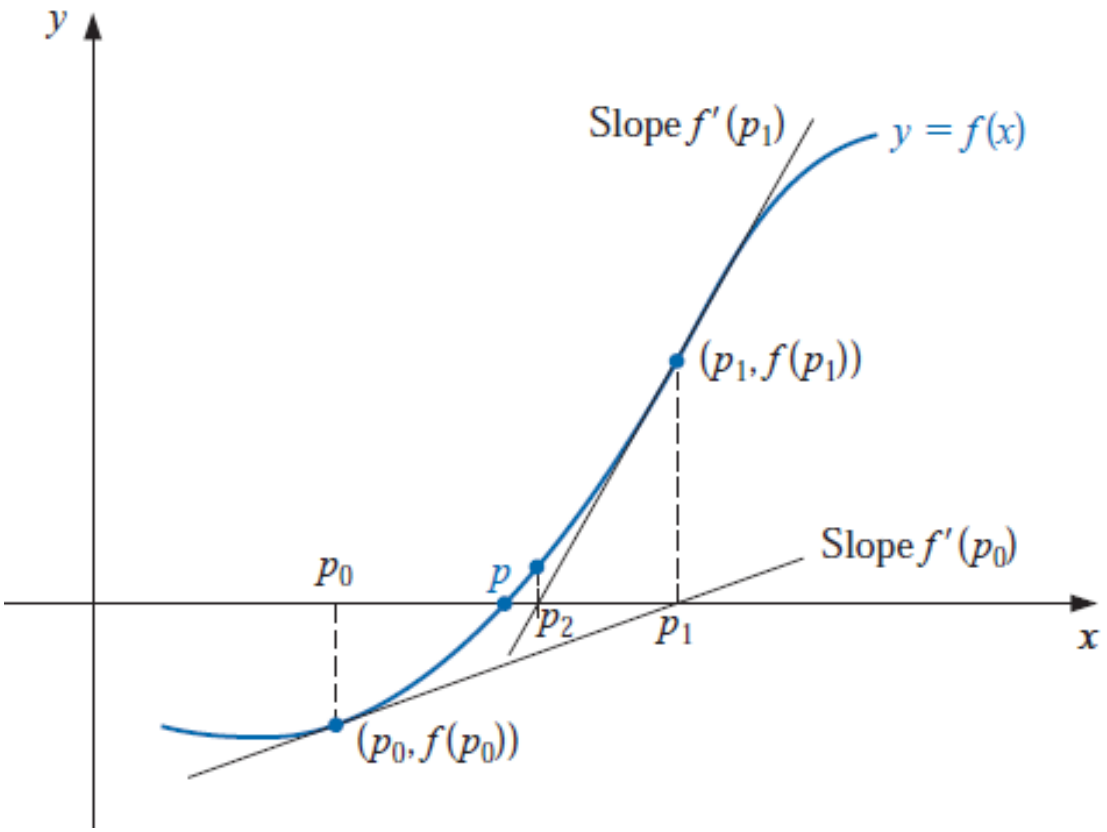
Newton's method is derived by assuming that since $|p - p_0|$ is small, the term involving $(p - p_0)^2$ is much smaller, so

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

Solving for p gives

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

Figure 2.8 on page 68 illustrates how the approximations are obtained using successive tangents. (Also see Exercise 15.) Starting with the initial approximation p_0, the approximation p_1 is the x-intercept of the tangent line to the graph of f at $(p_0, f(p_0))$. The approximation p_2 is the x-intercept of the tangent line to the graph of f at $(p_1, f(p_1))$ and so on. Algorithm 2.3 follows this procedure.



Newton's method is a functional iteration technique with $p_n = g(p_{n-1})$, for which

$$g(p_{n-1}) = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}, \quad \text{for } n \geq 1. \tag{2.11}$$

In fact, this is the functional iteration technique that was used to give the rapid convergence we saw in column (e) of Table 2.2 in Section 2.2.

It is clear from Equation (2.7) that Newton's method cannot be continued if $f'(p_{n-1}) = 0$ for some n. In fact, we will see that the method is most effective when f' is bounded away from zero near p.

Theorem 2.6 Let $f \in C^2[a,b]$. If $p \in (a,b)$ is such that $f(p) = 0$ and $f'(p) \neq 0$, then there exists a $\delta > 0$ such that Newton's method generates a sequence $\{p_n\}_{n=1}^{\infty}$ converging to p for any initial approximation $p_0 \in [p-\delta, p+\delta]$. ■

Proof The proof is based on analyzing Newton's method as the functional iteration scheme $p_n = g(p_{n-1})$, for $n \geq 1$, with

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

Let k be in $(0, 1)$. We first find an interval $[p-\delta, p+\delta]$ that g maps into itself and for which $|g'(x)| \leq k$, for all $x \in (p-\delta, p+\delta)$.

Since f' is continuous and $f'(p) \neq 0$, part (**a**) of Exercise 29 in Section 1.1 implies that there exists a $\delta_1 > 0$, such that $f'(x) \neq 0$ for $x \in [p-\delta_1, p+\delta_1] \subseteq [a, b]$. Thus g is defined and continuous on $[p-\delta_1, p+\delta_1]$. Also

$$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{[f'(x)]^2} = \frac{f(x)f''(x)}{[f'(x)]^2},$$

for $x \in [p-\delta_1, p+\delta_1]$, and, since $f \in C^2[a, b]$, we have $g \in C^1[p-\delta_1, p+\delta_1]$.

By assumption, $f(p) = 0$, so

$$g'(p) = \frac{f(p)f''(p)}{[f'(p)]^2} = 0.$$

Since g' is continuous and $0 < k < 1$, part (**b**) of Exercise 29 in Section 1.1 implies that there exists a δ, with $0 < \delta < \delta_1$, and

$$|g'(x)| \leq k, \quad \text{for all } x \in [p-\delta, p+\delta].$$

It remains to show that g maps $[p-\delta, p+\delta]$ into $[p-\delta, p+\delta]$. If $x \in [p-\delta, p+\delta]$, the Mean Value Theorem implies that for some number ξ between x and p, $|g(x)-g(p)| = |g'(\xi)||x-p|$. So

$$|g(x)-p| = |g(x)-g(p)| = |g'(\xi)||x-p| \le k|x-p| < |x-p|.$$

Since $x \in [p-\delta, p+\delta]$, it follows that $|x-p| < \delta$ and that $|g(x)-p| < \delta$. Hence, g maps $[p-\delta, p+\delta]$ into $[p-\delta, p+\delta]$.

All the hypotheses of the Fixed-Point Theorem 2.4 are now satisfied, so the sequence $\{p_n\}_{n=1}^{\infty}$, defined by

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

converges to p for any $p_0 \in [p-\delta, p+\delta]$.

■ ■ ■

10 Numerical Solutions of Nonlinear Systems of Equations 629

10.2 Newton's Method

ALGORITHM 10.1

Newton's Method for Systems

To approximate the solution of the nonlinear system $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ given an initial approximation $\mathbf{x}$:

INPUT number n of equations and unknowns; initial approximation $\mathbf{x} = (x_1, \ldots, x_n)^t$, tolerance TOL; maximum number of iterations N.
OUTPUT approximate solution $\mathbf{x} = (x_1, \ldots, x_n)^t$ or a message that the number of iterations was exceeded.

Step 1 Set $k = 1$.

Step 2 While $(k \leq N)$ do Steps 3–7.

Step 3 Calculate $\mathbf{F}(\mathbf{x})$ and $J(\mathbf{x})$, where $J(\mathbf{x})_{i,j} = (\partial f_i(\mathbf{x})/\partial x_j)$ for $1 \leq i, j \leq n$.

Step 4 Solve the $n \times n$ linear system $J(\mathbf{x})\mathbf{y} = -\mathbf{F}(\mathbf{x})$.

Step 5 Set $\mathbf{x} = \mathbf{x} + \mathbf{y}$.

Step 6 If $||\mathbf{y}|| < TOL$ then OUTPUT $(\mathbf{x})$;
(*The procedure was successful.*)
STOP.

Step 7 Set $k = k + 1$.

Step 8 OUTPUT ('Maximum number of iterations exceeded');
(*The procedure was unsuccessful.*)
STOP.

■

10.3 Quasi-Newton Methods

A significant weakness of Newton's method for solving systems of nonlinear equations is the need, at each iteration, to determine a Jacobian matrix and solve an $n \times n$ linear system that involves this matrix. Consider the amount of computation associated with one iteration of Newton's method. The Jacobian matrix associated with a system of n nonlinear equations written in the form $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ requires that the n^2 partial derivatives of the n component functions of $\mathbf{F}$ be determined and evaluated. In most situations, the exact evaluation of the partial derivatives is inconvenient, although the problem has been made more tractable with the widespread use of symbolic computation systems, such as Maple.

When the exact evaluation is not practical, we can use finite difference approximations to the partial derivatives. For example,

$$\frac{\partial f_j}{\partial x_k}(\mathbf{x}^{(i)}) \approx \frac{f_j(\mathbf{x}^{(i)} + \mathbf{e}_k h) - f_j(\mathbf{x}^{(i)})}{h}, \tag{10.10}$$

where h is small in absolute value and $\mathbf{e}_k$ is the vector whose only nonzero entry is a 1 in the kth coordinate. This approximation, however, still requires that at least n^2 scalar functional evaluations be performed to approximate the Jacobian and does not decrease the amount of calculation, in general $O(n^3)$, required for solving the linear system involving this approximate Jacobian.

The total computational effort for just one iteration of Newton's method is consequently at least $n^2 + n$ scalar functional evaluations (n^2 for the evaluation of the Jacobian matrix and n for the evaluation of $\mathbf{F}$) together with $O(n^3)$ arithmetic operations to solve the linear system. This amount of computational effort is extensive, except for relatively small values of n and easily evaluated scalar functions.

In this section we consider a generalization of the Secant method to systems of nonlinear equations, a technique known as **Broyden's method** (see [Broy]). The method requires only n scalar functional evaluations per iteration and also reduces the number of arithmetic calculations to $O(n^2)$. It belongs to a class of methods known as *least-change secant updates* that produce algorithms called **quasi-Newton**. These methods replace the Jacobian matrix in Newton's method with an approximation matrix that is easily updated at each iteration.

The disadvantage of the quasi-Newton methods is that the quadratic convergence of Newton's method is lost, being replaced, in general, by a convergence called **superlinear**. This implies that

$$\lim_{i\to\infty} \frac{\|\mathbf{x}^{(i+1)} - \mathbf{p}\|}{\|\mathbf{x}^{(i)} - \mathbf{p}\|} = 0,$$

where $\mathbf{p}$ denotes the solution to $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ and $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(i+1)}$ are consecutive approximations to $\mathbf{p}$.

In most applications, the reduction to superlinear convergence is a more than acceptable trade-off for the decrease in the amount of computation. An additional disadvantage of quasi-Newton methods is that, unlike Newton's method, they are not self-correcting. Newton's method will generally correct for roundoff error with successive iterations, but unless special safeguards are incorporated, Broyden's method will not.

To describe Broyden's method, suppose that an initial approximation $\mathbf{x}^{(0)}$ is given to the solution $\mathbf{p}$ of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. We calculate the next approximation $\mathbf{x}^{(1)}$ in the same manner as Newton's method. If it is inconvenient to determine $J(\mathbf{x}^{(0)})$ exactly, we use the difference equations given by (10.10) to approximate the partial derivatives. To compute $\mathbf{x}^{(2)}$, however, we depart from Newton's method and examine the Secant method for a single nonlinear equation. The Secant method uses the approximation

$$f'(x_1) \approx \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

as a replacement for $f'(x_1)$ in the single-variable Newton's method.

For nonlinear systems, $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$ is a vector, so the corresponding quotient is undefined. However, the method proceeds similarly in that we replace the matrix $J\left(\mathbf{x}^{(1)}\right)$ in Newton's method for systems by a matrix A_1 with the property that

$$A_1\left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right) = \mathbf{F}\left(\mathbf{x}^{(1)}\right) - \mathbf{F}\left(\mathbf{x}^{(0)}\right). \tag{10.11}$$

For nonlinear systems, $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$ is a vector, so the corresponding quotient is undefined. However, the method proceeds similarly in that we replace the matrix $J\left(\mathbf{x}^{(1)}\right)$ in Newton's method for systems by a matrix A_1 with the property that

$$A_1\left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right) = \mathbf{F}\left(\mathbf{x}^{(1)}\right) - \mathbf{F}\left(\mathbf{x}^{(0)}\right). \tag{10.11}$$

Any nonzero vector in $\mathbb{R}^n$ can be written as the sum of a multiple of $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$ and a multiple of a vector in the orthogonal complement of $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$. So, to uniquely define the matrix A_1, we also need to specify how it acts on the orthogonal complement of $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$. No information is available about the change in $\mathbf{F}$ in a direction orthogonal to $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$, so we specify that no change be made in this direction, that is,

$$A_1\mathbf{z} = J\left(\mathbf{x}^{(0)}\right)\mathbf{z}, \quad \text{whenever } \left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right)^t \mathbf{z} = 0. \tag{10.12}$$

Thus, any vector orthogonal to $\mathbf{x}^{(1)} - \mathbf{x}^{(0)}$ is unaffected by the update from $J\left(\mathbf{x}^{(0)}\right)$, which was used to compute $\mathbf{x}^{(1)}$, to A_1, which is used in the determination of $\mathbf{x}^{(2)}$.

Conditions (10.11) and (10.12) uniquely define A_1 (see [DM]) as

$$A_1 = J\left(\mathbf{x}^{(0)}\right) + \frac{\left[\mathbf{F}\left(\mathbf{x}^{(1)}\right) - \mathbf{F}\left(\mathbf{x}^{(0)}\right) - J\left(\mathbf{x}^{(0)}\right)\left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right)\right]\left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right)^t}{\left\|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right\|_2^2}.$$

It is this matrix that is used in place of $J\left(\mathbf{x}^{(1)}\right)$ to determine $\mathbf{x}^{(2)}$ as

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} - A_1^{-1}\mathbf{F}\left(\mathbf{x}^{(1)}\right).$$

Once $\mathbf{x}^{(2)}$ has been determined, the method is repeated to determine $\mathbf{x}^{(3)}$, using A_1 in place of $A_0 \equiv J\left(\mathbf{x}^{(0)}\right)$, and with $\mathbf{x}^{(2)}$ and $\mathbf{x}^{(1)}$ in place of $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(0)}$.

In general, once $\mathbf{x}^{(i)}$ has been determined, $\mathbf{x}^{(i+1)}$ is computed by

$$A_i = A_{i-1} + \frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\mathbf{s}_i^t \tag{10.13}$$

and

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - A_i^{-1}\mathbf{F}\left(\mathbf{x}^{(i)}\right), \tag{10.14}$$

where the notation $\mathbf{y}_i = \mathbf{F}\left(\mathbf{x}^{(i)}\right) - \mathbf{F}\left(\mathbf{x}^{(i-1)}\right)$ and $\mathbf{s}_i = \mathbf{x}^{(i)} - \mathbf{x}^{(i-1)}$ is introduced to simplify the equations.

If the method was performed as outlined in Eqs. (10.13) and (10.14), the number of scalar functional evaluations would be reduced from $n^2 + n$ to n (those required for evaluating $\mathbf{F}(\mathbf{x}^{(i)})$), but $O(n^3)$ calculations would still required to solve the associated $n \times n$ linear system (see Step 4 in Algorithm 10.1)

$$A_i \mathbf{s}_{i+1} = -\mathbf{F}(\mathbf{x}^{(i)}). \tag{10.15}$$

Employing the method in this form would not be justified because of the reduction to superlinear convergence from the quadratic convergence of Newton's method.

Sherman-Morrison Formula

A considerable improvement can be incorporated, however, by employing a matrix inversion formula of Sherman and Morrison (see, for example, [DM], p. 55).

Theorem 10.8

(Sherman-Morrison Formula)

Suppose that A is a nonsingular matrix and that $\mathbf{x}$ and $\mathbf{y}$ are vectors with $\mathbf{y}^t A^{-1}\mathbf{x} \neq -1$. Then $A + \mathbf{x}\mathbf{y}^t$ is nonsingular and

$$\left(A + \mathbf{x}\mathbf{y}^t\right)^{-1} = A^{-1} - \frac{A^{-1}\mathbf{x}\mathbf{y}^t A^{-1}}{1 + \mathbf{y}^t A^{-1}\mathbf{x}}.$$

■

The Sherman-Morrison formula permits A_i^{-1} to be computed directly from A_{i-1}^{-1}, eliminating the need for a matrix inversion with each iteration.

Letting $A = A_{i-1}$, $\mathbf{x} = (\mathbf{y}_i - A_{i-1}\mathbf{s}_i)/||\mathbf{s}_i||_2^2$, and $\mathbf{y} = \mathbf{s}_i$, in Eq. (10.13) gives

$$\begin{aligned} A_i^{-1} &= \left(A_{i-1} + \frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\mathbf{s}_i^t\right)^{-1} \\ &= A_{i-1}^{-1} - \frac{A_{i-1}^{-1}\left(\frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\mathbf{s}_i^t\right)A_{i-1}^{-1}}{1 + \mathbf{s}_i^t A_{i-1}^{-1}\left(\frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\right)} \\ &= A_{i-1}^{-1} - \frac{\left(A_{i-1}^{-1}\mathbf{y}_i - \mathbf{s}_i\right)\mathbf{s}_i^t A_{i-1}^{-1}}{||\mathbf{s}_i||_2^2 + \mathbf{s}_i^t A_{i-1}^{-1}\mathbf{y}_i - ||\mathbf{s}_i||_2^2}, \end{aligned}$$

so

$$A_i^{-1} = A_{i-1}^{-1} + \frac{\left(\mathbf{s}_i - A_{i-1}^{-1}\mathbf{y}_i\right)\mathbf{s}_i^t A_{i-1}^{-1}}{\mathbf{s}_i^t A_{i-1}^{-1}\mathbf{y}_i}. \tag{10.16}$$

This computation involves only matrix-vector multiplications at each step and therefore requires only $O(n^2)$ arithmetic calculations. The calculation of A_i is bypassed, as is the necessity of solving the linear system (10.15).

Algorithm 10.2 follows directly from this construction, incorporating (10.16) into the iterative technique (10.14).

Broyden

To approximate the solution of the nonlinear system $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ given an initial approximation $\mathbf{x}$:

INPUT number n of equations and unknowns; initial approximation $\mathbf{x} = (x_1, \ldots, x_n)^t$; tolerance TOL; maximum number of iterations N.

OUTPUT approximate solution $\mathbf{x} = (x_1, \ldots, x_n)^t$ or a message that the number of iterations was exceeded.

Step 1 Set $A_0 = J(\mathbf{x})$ where $J(\mathbf{x})_{i,j} = \frac{\partial f_i}{\partial x_j}(\mathbf{x})$ for $1 \leq i, j \leq n$;

$\mathbf{v} = \mathbf{F}(\mathbf{x})$. (*Note:* $\mathbf{v} = \mathbf{F}(\mathbf{x}^{(0)})$.)

Step 2 Set $A = A_0^{-1}$. (*Use Gaussian elimination.*)

Step 3 Set $\mathbf{s} = -A\mathbf{v}$; (*Note:* $\mathbf{s} = \mathbf{s}_1$.)

$\mathbf{x} = \mathbf{x} + \mathbf{s}$; (*Note:* $\mathbf{x} = \mathbf{x}^{(1)}$.)

$k = 2$.

Step 4 While ($k \leq N$) do Steps 5–13.

Step 5 Set $\mathbf{w} = \mathbf{v}$; (*Save* $\mathbf{v}$.)
$\mathbf{v} = \mathbf{F}(\mathbf{x})$; (*Note:* $\mathbf{v} = \mathbf{F}(\mathbf{x}^{(k)})$.)
$\mathbf{y} = \mathbf{v} - \mathbf{w}$. (*Note:* $\mathbf{y} = \mathbf{y}_k$.)

Step 6 Set $\mathbf{z} = -A\mathbf{y}$. (*Note:* $\mathbf{z} = -A_{k-1}^{-1}\mathbf{y}_k$.)

Step 7 Set $p = -\mathbf{s}^t\mathbf{z}$. (*Note:* $p = \mathbf{s}_k^t A_{k-1}^{-1}\mathbf{y}_k$.)

Step 8 Set $\mathbf{u}^t = \mathbf{s}^t A$.

Step 9 Set $A = A + \frac{1}{p}(\mathbf{s} + \mathbf{z})\mathbf{u}^t$. (*Note:* $A = A_k^{-1}$.)

Step 10 Set $\mathbf{s} = -A\mathbf{v}$. (*Note:* $\mathbf{s} = -A_k^{-1}\mathbf{F}(\mathbf{x}^{(k)})$.)

Step 11 Set $\mathbf{x} = \mathbf{x} + \mathbf{s}$. (*Note:* $\mathbf{x} = \mathbf{x}^{(k+1)}$.)

Step 12 If $||\mathbf{s}|| < TOL$ then OUTPUT ($\mathbf{x}$);
(*The procedure was successful.*)
STOP.

Step 13 Set $k = k + 1$.

Step 14 OUTPUT ('Maximum number of iterations exceeded');
(*The procedure was unsuccessful.*)
STOP.

■

Example 1 Use Broyden's method with $\mathbf{x}^{(0)} = (0.1, 0.1, -0.1)^t$ to approximate the solution to the nonlinear system

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

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

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

Solution This system was solved by Newton's method in Example 1 of Section 10.2. The Jacobian matrix for this system is

$$J(x_1, x_2, x_3) = \begin{bmatrix} 3 & x_3 \sin x_2 x_3 & x_2 \sin x_2 x_3 \\ 2x_1 & -162(x_2 + 0.1) & \cos x_3 \\ -x_2 e^{-x_1 x_2} & -x_1 e^{-x_1 x_2} & 20 \end{bmatrix}.$$

Let $\mathbf{x}^{(0)} = (0.1, 0.1, -0.1)^t$ and

$$\mathbf{F}(x_1, x_2, x_3) = (f_1(x_1, x_2, x_3), f_2(x_1, x_2, x_3), f_3(x_1, x_2, x_3))^t,$$

where

$$f_1(x_1, x_2, x_3) = 3x_1 - \cos(x_2 x_3) - \frac{1}{2},$$

$$f_2(x_1, x_2, x_3) = x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06,$$

and

$$f_3(x_1, x_2, x_3) = e^{-x_1 x_2} + 20x_3 + \frac{10\pi - 3}{3}.$$

Then

$$\mathbf{F}(\mathbf{x}^{(0)}) = \begin{bmatrix} -1.199950 \\ -2.269833 \\ 8.462025 \end{bmatrix}.$$

Because

$$A_0 = J\left(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}\right)$$

$$= \begin{bmatrix} 3 & 9.999833 \times 10^{-4} & -9.999833 \times 10^{-4} \\ 0.2 & -32.4 & 0.9950042 \\ -9.900498 \times 10^{-2} & -9.900498 \times 10^{-2} & 20 \end{bmatrix},$$

we have

$$A_0^{-1} = J\left(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}\right)^{-1}$$

$$= \begin{bmatrix} 0.3333332 & 1.023852 \times 10^{-5} & 1.615701 \times 10^{-5} \\ 2.108607 \times 10^{-3} & -3.086883 \times 10^{-2} & 1.535836 \times 10^{-3} \\ 1.660520 \times 10^{-3} & -1.527577 \times 10^{-4} & 5.000768 \times 10^{-2} \end{bmatrix}.$$

So

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} - A_0^{-1}\mathbf{F}\left(\mathbf{x}^{(0)}\right) = \begin{bmatrix} 0.4998697 \\ 1.946685 \times 10^{-2} \\ -0.5215205 \end{bmatrix},$$

$$\mathbf{F}\left(\mathbf{x}^{(1)}\right) = \begin{bmatrix} -3.394465 \times 10^{-4} \\ -0.3443879 \\ 3.188238 \times 10^{-2} \end{bmatrix},$$

$$\mathbf{y}_1 = \mathbf{F}(\mathbf{x}^{(1)}) - \mathbf{F}(\mathbf{x}^{(0)}) = \begin{bmatrix} 1.199611 \\ 1.925445 \\ -8.430143 \end{bmatrix},$$

$$\mathbf{s}_1 = \begin{bmatrix} 0.3998697 \\ -8.053315 \times 10^{-2} \\ -0.4215204 \end{bmatrix},$$

$$\mathbf{s}_1^t A_0^{-1} \mathbf{y}_1 = 0.3424604,$$

$$A_1^{-1} = A_0^{-1} + (1/0.3424604)\left[\left(\mathbf{s}_1 - A_0^{-1}\mathbf{y}_1\right)\mathbf{s}_1^t A_0^{-1}\right]$$

$$= \begin{bmatrix} 0.3333781 & 1.11050 \times 10^{-5} & 8.967344 \times 10^{-6} \\ -2.021270 \times 10^{-3} & -3.094849 \times 10^{-2} & 2.196906 \times 10^{-3} \\ 1.022214 \times 10^{-3} & -1.650709 \times 10^{-4} & 5.010986 \times 10^{-2} \end{bmatrix},$$

and

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} - A_1^{-1}\mathbf{F}(\mathbf{x}^{(1)}) = \begin{bmatrix} 0.4999863 \\ 8.737833 \times 10^{-3} \\ -0.5231746 \end{bmatrix}.$$

Additional iterations are listed in Table 10.4. The fifth iteration of Broyden's method is slightly less accurate than was the fourth iteration of Newton's method given in the example at the end of the preceding section. ■

Broyden's method

Table 10.4

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_2$
3	0.5000066	8.672157×10^{-4}	−0.5236918	7.88×10^{-3}
4	0.5000003	6.083352×10^{-5}	−0.5235954	8.12×10^{-4}
5	0.5000000	-1.448889×10^{-6}	−0.5235989	6.24×10^{-5}
6	0.5000000	6.059030×10^{-9}	−0.5235988	1.50×10^{-6}

Newton's Method

Table 10.3

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_\infty$
0	0.1000000000	0.1000000000	−0.1000000000	
1	0.4998696728	0.0194668485	−0.5215204718	0.4215204718
2	0.5000142403	0.0015885914	−0.5235569638	1.788×10^{-2}
3	0.5000000113	0.0000124448	−0.5235984500	1.576×10^{-3}
4	0.5000000000	8.516×10^{-10}	−0.5235987755	1.244×10^{-5}
5	0.5000000000	-1.375×10^{-11}	−0.5235987756	8.654×10^{-10}

EXERCISE SET 10.3

1. Use Broyden's method with $\mathbf{x}^{(0)} = \mathbf{0}$ to compute $\mathbf{x}^{(2)}$ for each of the following nonlinear systems.

a.
$$4x_1^2 - 20x_1 + \frac{1}{4}x_2^2 + 8 = 0,$$
$$\frac{1}{2}x_1x_2^2 + 2x_1 - 5x_2 + 8 = 0.$$


12
1
2
3
4
5
6
7
8
9
10
11

Broyden

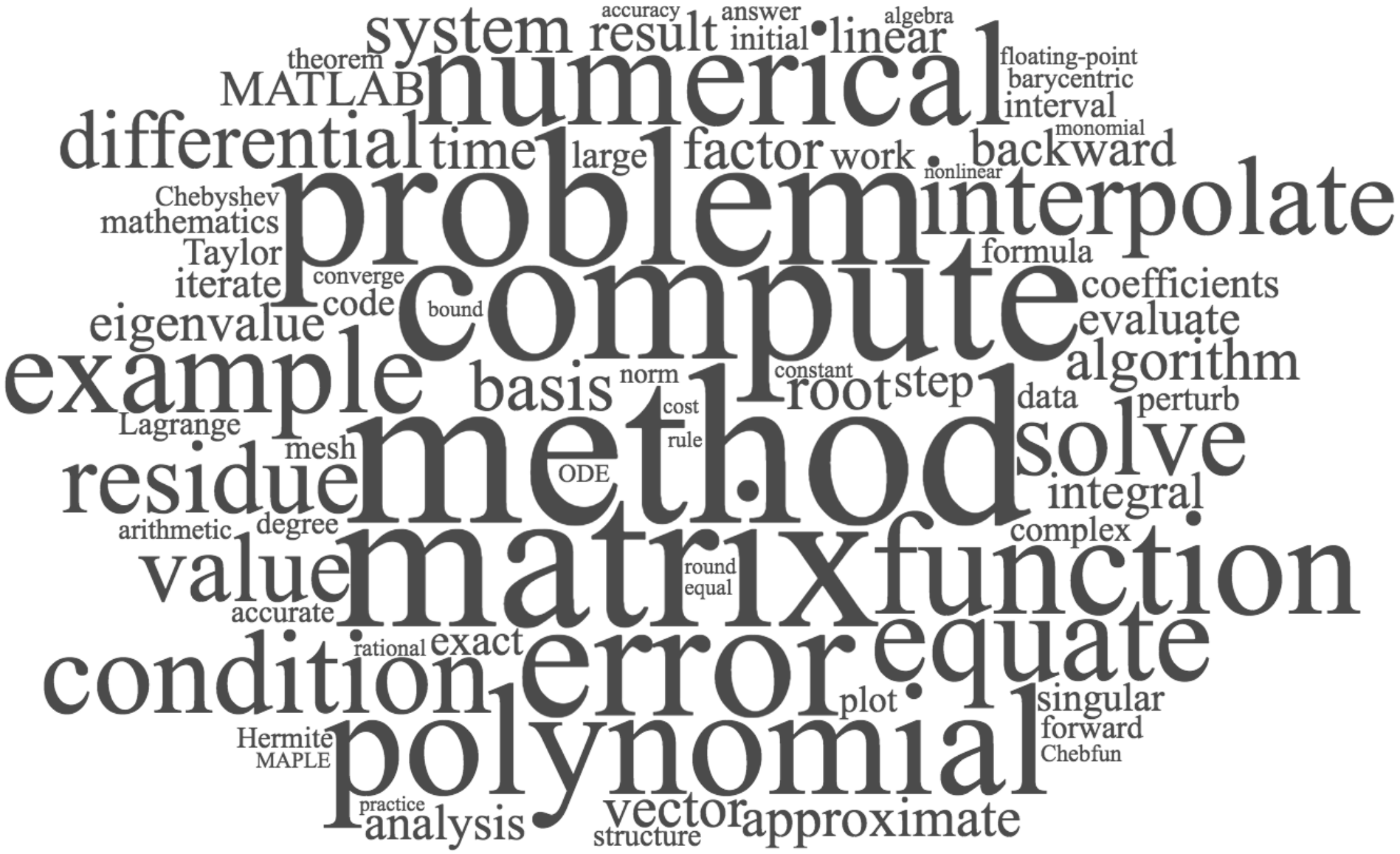
					Jacobiana					Inversa							
iteração	x1	x2	f1	f2	j11	j12	j21	j22	detJ	a11	a12	a21	a22				
0	0	0	8	8	-20	0	2	-5	100	-0,05	0	-0,02	-0,2				
	s1	s2	norm2s														
	0,4	1,76	3,257600											Inversa			
	x1	x2	v1	v2	y1	y2	z1	z2	p	u1t	u2t	(s1+z1)	(s2+z2)	a11	a12	a21	a22
1	0,4	1,76	1,4144	0,61952	-6,5856	-7,38048	-0,32928	-1,60781	2,961454	-0,0552	-0,352	0,07072	0,152192	-0,05132	-0,00841	-0,02284	-0,21809
	s1	s2	norm2s														
	0,077792	0,167411	0,034078											Inversa			
	x1	x2	v1	v2	y1	y2	z1	z2	p	u1t	u2t	(s1+z1)	(s2+z2)	a11	a12	a21	a22
2	0,477792	1,927411	0,28603	0,20601	-1,12837	-0,41351	-0,06138	-0,11595	0,024187	-0,00782	-0,03716	0,01641	0,05146	-0,05662	-0,03362	-0,03946	-0,29716
	s1	s2	norm2s														
	0,023121	0,072505	0,005792											Inversa			
	x1	x2	v1	v2	y1	y2	z1	z2	p	u1t	u2t	(s1+z1)	(s2+z2)	a11	a12	a21	a22
3	0,500913	1,999916	-0,01469	0,00399	-0,30072	-0,20202	-0,02382	-0,0719	0,005764	-0,00417	-0,02232	-0,0007	0,000605	-0,05612	-0,03092	-0,0399	-0,29951
	s1	s2	norm2s														
	-0,0007	0,000608	0,000001														
	x1	x2	v1	v2													
4	0,500212	2,000524	-0,00287	-0,00125													

Newton's Method

iteração	x1	x2	f1	f2	df1dx1	df1dx2	df2dx1	df2dx2	detJ	y1	y2	norm_2
0	0	0	8	8	-20	0	2	-5	100	0,4	1,76	3,2576
1	0,4	1,76	1,4144	0,61952	-16,8	0,88	3,5488	-4,296	69,04986	0,095894	0,223423	0,059114
2	0,495894	1,983423	0,04926185	0,050085	-16,0329	0,991712	3,966984	-4,01643	60,46077	0,004094	0,016514	0,000289
3	0,499988	1,999937	0,000135218	0,000202	-16,0001	0,999969	3,999874	-4,00006	60,00155	1,24E-05	6,3E-05	4,12E-09
4	0,5	2	1,60427E-09	2,55E-09	-16	1	4	-4	60	1,49E-10	7,87E-10	6,42E-19
5	0,5	2	0	0	-16	1	4	-4	60	0	0	0

10 Numerical Solutions of Nonlinear Systems of Equations 629

MAP2220


accuracy
answer
algebra
system
result
initial
linear
theorem
MATLAB
numerical
floating-point
barycentric
interval
differential
time
large
factor
work
backward
monomial
nonlinear
problem
interpolate
Chebyshev
mathematics
Taylor
iterate
converge
code
bound
compute
formula
coefficients
evaluate
eigenvalue
example
basis
norm
constant
root
step
algorithm
data
perturb
cost
Lagrange
mesh
rule
method
solve
residue
ODE
integral
arithmetic
degree
complex
value
matrix
function
round
equal
accurate
rational
exact
equate
condition
error
plot
singular
forward
Hermite
MAPLE
Chebfun
polynomial
practice
analysis
vector
approximate
structure