Numerical Solutions of Nonlinear Systems of Equations

NTNU

Tsung-Min Hwang

November 30, 2003

1 – Fixed Point method

Definition 1 *A function G from $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$ has a fixed point at $p \in D$ if $G(p) = p$.*

1 – Fixed Point method

Definition 1 *A function G from $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$ has a fixed point at $p \in D$ if $G(p) = p$.*

Theorem 1 (Contraction Mapping Theorem) *Let $D = \{(x_1, \cdots, x_n)^T; a_i \le x_i \le b_i, \forall\, i = 1, \ldots, n\} \subset \mathbb{R}^n$. Suppose $G : D \to \mathbb{R}^n$ is a continuous function with $G(x) \in D$ whenever $x \in D$. Then G has a fixed point in D. Suppose, in addition, G has continuous partial derivatives and a constant $\alpha < 1$ exists with*

$$\left| \frac{\partial g_i(x)}{\partial x_j} \right| \le \frac{\alpha}{n}, \quad \text{whenever } x \in D,$$

for $j = 1, \ldots, n$ and $i = 1, \ldots, n$. Then, for any $x^{(0)} \in D$,

$$x^{(k)} = G(x^{(k-1)}), \quad \text{for each } k \ge 1$$

converges to the unique fixed point $p \in D$ and

$$\| x^{(k)} - p \|_\infty \le \frac{\alpha^k}{1-\alpha} \| x^{(1)} - x^{(0)} \|_\infty .$$

2 – Newton's Method

First consider solving the following system of nonlinear equations:

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0. \end{cases}$$

2 – Newton's Method

First consider solving the following system of nonlinear equations:

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0. \end{cases}$$

Suppose $(x_1^{(k)}, x_2^{(k)})$ is an approximation to the solution of the system above, and we try to compute $h_1^{(k)}$ and $h_2^{(k)}$ such that $(x_1^{(k)} + h_1^{(k)}, x_2^{(k)} + h_2^{(k)})$ satisfies the system.

2 – Newton's Method

First consider solving the following system of nonlinear equations:

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0. \end{cases}$$

Suppose $(x_1^{(k)}, x_2^{(k)})$ is an approximation to the solution of the system above, and we try to compute $h_1^{(k)}$ and $h_2^{(k)}$ such that $(x_1^{(k)} + h_1^{(k)}, x_2^{(k)} + h_2^{(k)})$ satisfies the system. By the Taylor's theorem for two variables,

$$\begin{aligned} 0 &= f_1(x_1^{(k)} + h_1^{(k)}, x_2^{(k)} + h_2^{(k)}) \\ &\approx f_1(x_1^{(k)}, x_2^{(k)}) + h_1^{(k)} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) + h_2^{(k)} \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ 0 &= f_2(x_1^{(k)} + h_1^{(k)}, x_2^{(k)} + h_2^{(k)}) \\ &\approx f_2(x_1^{(k)}, x_2^{(k)}) + h_1^{(k)} \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) + h_2^{(k)} \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{aligned}$$

Put this in matrix form

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} + \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Put this in matrix form

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} + \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The matrix

$$J(x_1^{(k)}, x_2^{(k)}) \equiv \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}$$

is called the Jacobian matrix.

Put this in matrix form

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} + \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The matrix

$$J(x_1^{(k)}, x_2^{(k)}) \equiv \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}$$

is called the Jacobian matrix. Set $h_1^{(k)}$ and $h_2^{(k)}$ be the solution of the linear system

$$J(x_1^{(k)}, x_2^{(k)}) \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} = - \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}$$

Put this in matrix form

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} + \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The matrix

$$J(x_1^{(k)}, x_2^{(k)}) \equiv \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}$$

is called the Jacobian matrix. Set $h_1^{(k)}$ and $h_2^{(k)}$ be the solution of the linear system

$$J(x_1^{(k)}, x_2^{(k)}) \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix} = - \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}$$

then

$$\begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \end{bmatrix} + \begin{bmatrix} h_1^{(k)} \\ h_2^{(k)} \end{bmatrix}$$

is expected to be a better approximation.

Sol. of Non-linear Systems

In general, we solve the system of n nonlinear equations $f_i(x_1, \cdots, x_n) = 0$, $i = 1, \ldots, n$.

Sol. of Non-linear Systems

In general, we solve the system of n nonlinear equations $f_i(x_1, \cdots, x_n) = 0$, $i = 1, \ldots, n$. Let

$$x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$$

and

$$F(x) = \begin{bmatrix} f_1(x) & f_2(x) & \cdots & f_n(x) \end{bmatrix}^T.$$

Sol. of Non-linear Systems

In general, we solve the system of n nonlinear equations $f_i(x_1, \cdots, x_n) = 0$, $i = 1, \ldots, n$. Let

$$x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$$

and

$$F(x) = \begin{bmatrix} f_1(x) & f_2(x) & \cdots & f_n(x) \end{bmatrix}^T.$$

The problem can be formulated as solving

$$F(x) = 0, \quad F : \mathbb{R}^n \to \mathbb{R}^n.$$

In general, we solve the system of n nonlinear equations $f_i(x_1, \cdots, x_n) = 0$, $i = 1, \ldots, n$. Let

$$x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$$

and

$$F(x) = \begin{bmatrix} f_1(x) & f_2(x) & \cdots & f_n(x) \end{bmatrix}^T.$$

The problem can be formulated as solving

$$F(x) = 0, \quad F : \mathbb{R}^n \to \mathbb{R}^n.$$

Let $J(x)$, where the (i,j) entry is $\frac{\partial f_i}{\partial x_j}(x)$, be the $n \times n$ Jacobian matrix.

Sol. of Non-linear Systems

In general, we solve the system of n nonlinear equations $f_i(x_1, \cdots, x_n) = 0$, $i = 1, \ldots, n$. Let

$$x = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$$

and

$$F(x) = \begin{bmatrix} f_1(x) & f_2(x) & \cdots & f_n(x) \end{bmatrix}^T.$$

The problem can be formulated as solving

$$F(x) = 0, \quad F : \mathbb{R}^n \to \mathbb{R}^n.$$

Let $J(x)$, where the (i, j) entry is $\frac{\partial f_i}{\partial x_j}(x)$, be the $n \times n$ Jacobian matrix. Then the Newton's iteration is defined as

$$x^{(k+1)} = x^{(k)} + h^{(k)},$$

where $h^{(k)} \in \mathbb{R}^n$ is the solution of the linear system

$$J(x^{(k)})h^{(k)} = -F(x^{(k)}).$$

Algorithm 1 (Newton's Method for Systems) *Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$, an initial guess $x^{(0)}$ to the zero of F, and stop criteria M, δ, and ε, this algorithm performs the Newton's iteration to approximate one root of F.*

Set $k = 0$ and $h^{(-1)} = e_1$.

while $(k < M)$ and $(\| h^{(k-1)} \| \geq \delta)$ and $(\| F(x^{(k)}) \| \geq \varepsilon$ **do**

  Calculate $J(x^{(k)}) = [\partial F_i(x^{(k)})/\partial x_j]$.

  Solve the $n \times n$ linear system $J(x^{(k)})h^{(k)} = -F(x^{(k)})$.

  Set $x^{(k+1)} = x^{(k)} + h^{(k)}$ and $k = k + 1$.

end while

Output ("Convergent $x^{(k)}$") or ("Maximum number of iterations exceeded")

Algorithm 1 (Newton's Method for Systems) *Given a function $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, an initial guess $x^{(0)}$ to the zero of F, and stop criteria M, δ, and ε, this algorithm performs the Newton's iteration to approximate one root of F.*

Set $k = 0$ and $h^{(-1)} = e_1$.

while $(k < M)$ and $(\| h^{(k-1)} \| \geq \delta)$ and $(\| F(x^{(k)}) \| \geq \varepsilon$ **do**

Calculate $J(x^{(k)}) = [\partial F_i(x^{(k)})/\partial x_j]$.

Solve the $n \times n$ linear system $J(x^{(k)})h^{(k)} = -F(x^{(k)})$.

Set $x^{(k+1)} = x^{(k)} + h^{(k)}$ and $k = k + 1$.

end while

Output ("Convergent $x^{(k)}$") or ("Maximum number of iterations exceeded")

Remark 1

(i) quadratic convergence if the starting point is near the exact solution point in terms of vector norm.

(ii) At each iteration, a Jacobian matrix has to be evaluated and an $n \times n$ linear system involving this matrix must be solved.

3 – Quasi-Newton's Method

Next we will extend secant method to solving system of nonlinear equations.

3 – Quasi-Newton's Method

Next we will extend secant method to solving system of nonlinear equations. Recall that in one dimensional case, one uses the linear model

$$\ell_k(x) = f(x_k) + a_k(x - x_k)$$

to approximate the function $f(x)$ at x_k.

3 – Quasi-Newton's Method

Next we will extend secant method to solving system of nonlinear equations. Recall that in one dimensional case, one uses the linear model

$$\ell_k(x) = f(x_k) + a_k(x - x_k)$$

to approximate the function $f(x)$ at x_k. That is, $\ell_k(x_k) = f(x_k)$ for any $a_k \in \mathbb{R}$.

3 – Quasi-Newton's Method

Next we will extend secant method to solving system of nonlinear equations. Recall that in one dimensional case, one uses the linear model

$$\ell_k(x) = f(x_k) + a_k(x - x_k)$$

to approximate the function $f(x)$ at x_k. That is, $\ell_k(x_k) = f(x_k)$ for any $a_k \in \mathbb{R}$. If we further require that $\ell'(x_k) = f'(x_k)$, then $a_k = f'(x_k)$.

3 – Quasi-Newton's Method

Next we will extend secant method to solving system of nonlinear equations. Recall that in one dimensional case, one uses the linear model

$$\ell_k(x) = f(x_k) + a_k(x - x_k)$$

to approximate the function $f(x)$ at x_k. That is, $\ell_k(x_k) = f(x_k)$ for any $a_k \in \mathbb{R}$. If we further require that $\ell'(x_k) = f'(x_k)$, then $a_k = f'(x_k)$. The zero of $\ell_k(x)$ is used to give a new approximate for the zero of $f(x)$, that is,

$$x_{k+1} = x_k - \frac{1}{f'(x_k)} f(x_k)$$

which yields Newton's method.

If $f'(x_k)$ is not available, one instead asks the linear model to satisfy

$$\ell_k(x_k) = f(x_k) \quad \text{and} \quad \ell_k(x_{k-1}) = f(x_{k-1}).$$

If $f'(x_k)$ is not available, one instead asks the linear model to satisfy

$$\ell_k(x_k) = f(x_k) \quad \text{and} \quad \ell_k(x_{k-1}) = f(x_{k-1}).$$

In doing this, the identity

$$f(x_{k-1}) = \ell_k(x_{k-1}) = f(x_k) + a_k(x_{k-1} - x_k)$$

gives

$$a_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Sol. of Non-linear Systems

If $f'(x_k)$ is not available, one instead asks the linear model to satisfy

$$\ell_k(x_k) = f(x_k) \quad \text{and} \quad \ell_k(x_{k-1}) = f(x_{k-1}).$$

In doing this, the identity

$$f(x_{k-1}) = \ell_k(x_{k-1}) = f(x_k) + a_k(x_{k-1} - x_k)$$

gives

$$a_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Solving $\ell_k(x) = 0$ yields the secant iteration

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} f(x_k).$$

In multiple dimension, the analogue affine model becomes

$$M_k(x) = F(x_k) + A_k(x - x_k),$$

where $x, x_k \in \mathbb{R}^n$ and $A_k \in \mathbb{R}^{n\times n}$, and satisfies

$$M_k(x_k) = F(x_k),$$

for any A_k.

In multiple dimension, the analogue affine model becomes

$$M_k(x) = F(x_k) + A_k(x - x_k),$$

where $x, x_k \in \mathbb{R}^n$ and $A_k \in \mathbb{R}^{n\times n}$, and satisfies

$$M_k(x_k) = F(x_k),$$

for any A_k. The zero of $M_k(x)$ is then used to give a new approximate for the zero of $F(x)$, that is,

$$x_{k+1} = x_k - A_k^{-1}F(x_k).$$

Sol. of Non-linear Systems

In multiple dimension, the analogue affine model becomes

$$M_k(x) = F(x_k) + A_k(x - x_k),$$

where $x, x_k \in \mathbb{R}^n$ and $A_k \in \mathbb{R}^{n \times n}$, and satisfies

$$M_k(x_k) = F(x_k),$$

for any A_k. The zero of $M_k(x)$ is then used to give a new approximate for the zero of $F(x)$, that is,

$$x_{k+1} = x_k - A_k^{-1} F(x_k).$$

The Newton's method chooses

$$A_k = F'(x_k) \equiv J(x_k) = \text{the Jacobian matrix.}$$

and yields the iteration

$$x_{k+1} = x_k - \left(F'(x_k)\right)^{-1} F(x_k).$$

Sol. of Non-linear Systems

When the Jacobian matrix $J(x_k) \equiv F'(x_k)$ is not available, one can require

$$M_k(x_{k-1}) = F(x_{k-1}).$$

When the Jacobian matrix $J(x_k) \equiv F'(x_k)$ is not available, one can require

$$M_k(x_{k-1}) = F(x_{k-1}).$$

Then

$$F(x_{k-1}) = M_k(x_{k-1}) = F(x_k) + A_k(x_{k-1} - x_k),$$

which gives

$$A_k(x_k - x_{k-1}) = F(x_k) - F(x_{k-1})$$

and this is the so-called secant equation.

When the Jacobian matrix $J(x_k) \equiv F'(x_k)$ is not available, one can require

$$M_k(x_{k-1}) = F(x_{k-1}).$$

Then

$$F(x_{k-1}) = M_k(x_{k-1}) = F(x_k) + A_k(x_{k-1} - x_k),$$

which gives

$$A_k(x_k - x_{k-1}) = F(x_k) - F(x_{k-1})$$

and this is the so-called secant equation. Let

$$h_k = x_k - x_{k-1} \quad \text{and} \quad y_k = F(x_k) - F(x_{k-1}).$$

When the Jacobian matrix $J(x_k) \equiv F'(x_k)$ is not available, one can require

$$M_k(x_{k-1}) = F(x_{k-1}).$$

Then

$$F(x_{k-1}) = M_k(x_{k-1}) = F(x_k) + A_k(x_{k-1} - x_k),$$

which gives

$$A_k(x_k - x_{k-1}) = F(x_k) - F(x_{k-1})$$

and this is the so-called secant equation. Let

$$h_k = x_k - x_{k-1} \quad \text{and} \quad y_k = F(x_k) - F(x_{k-1}).$$

The secant equation becomes

$$A_k h_k = y_k.$$

However, this secant equation can not uniquely determine A_k.

Sol. of Non-linear Systems

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation.

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$M_k(x) - M_{k-1}(x) \quad = \quad F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1})$$

Sol. of Non-linear Systems

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$\begin{aligned} M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\ &= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \end{aligned}$$

Sol. of Non-linear Systems

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$
\begin{aligned}
M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\
&= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\
&= A_k(x_k - x_{k-1}) + A_k(x - x_k) - A_{k-1}(x - x_{k-1})
\end{aligned}
$$

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$\begin{aligned}
M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\
&= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\
&= A_k(x_k - x_{k-1}) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\
&= A_k(x - x_{k-1}) - A_{k-1}(x - x_{k-1})
\end{aligned}$$

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$
\begin{aligned}
M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\
&= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\
&= A_k(x_k - x_{k-1}) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\
&= A_k(x - x_{k-1}) - A_{k-1}(x - x_{k-1}) \\
&= (A_k - A_{k-1})(x - x_{k-1}).
\end{aligned}
$$

Sol. of Non-linear Systems

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$\begin{aligned} M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\ &= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\ &= A_k(x_k - x_{k-1}) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\ &= A_k(x - x_{k-1}) - A_{k-1}(x - x_{k-1}) \\ &= (A_k - A_{k-1})(x - x_{k-1}). \end{aligned}$$

For any $x \in \mathbb{R}^n$, we express

$$x - x_{k-1} = \alpha h_k + t_k,$$

for some $\alpha \in \mathbb{R}$, $t_k \in \mathbb{R}^n$, and $h_k^T t_k = 0$.

Sol. of Non-linear Systems

However, this secant equation can not uniquely determine A_k. One way of choosing A_k is to minimize $M_k - M_{k-1}$ subject to the secant equation. Note

$$\begin{aligned} M_k(x) - M_{k-1}(x) &= F(x_k) + A_k(x - x_k) - F(x_{k-1}) - A_{k-1}(x - x_{k-1}) \\ &= (F(x_k) - F(x_{k-1})) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\ &= A_k(x_k - x_{k-1}) + A_k(x - x_k) - A_{k-1}(x - x_{k-1}) \\ &= A_k(x - x_{k-1}) - A_{k-1}(x - x_{k-1}) \\ &= (A_k - A_{k-1})(x - x_{k-1}). \end{aligned}$$

For any $x \in \mathbb{R}^n$, we express

$$x - x_{k-1} = \alpha h_k + t_k,$$

for some $\alpha \in \mathbb{R}$, $t_k \in \mathbb{R}^n$, and $h_k^T t_k = 0$. Then

$$M_k - M_{k-1} = (A_k - A_{k-1})(\alpha h_k + t_k) = \alpha(A_k - A_{k-1})h_k + (A_k - A_{k-1})t_k.$$

Since

$$(A_k - A_{k-1})h_k = A_k h_k - A_{k-1} h_k = y_k - A_{k-1} h_k,$$

both y_k and $A_{k-1} h_k$ are old values, we have no control over the first part $(A_k - A_{k-1})h_k$.

Since

$$(A_k - A_{k-1})h_k = A_k h_k - A_{k-1}h_k = y_k - A_{k-1}h_k,$$

both y_k and $A_{k-1}h_k$ are old values, we have no control over the first part $(A_k - A_{k-1})h_k$. In order to minimize $M_k(x) - M_{k-1}(x)$, we try to choose A_k so that

$$(A_k - A_{k-1})t_k = 0$$

for all $t_k \in \mathbb{R}^n$, $h_k^T t_k = 0$.

Since

$$(A_k - A_{k-1})h_k = A_k h_k - A_{k-1}h_k = y_k - A_{k-1}h_k,$$

both y_k and $A_{k-1}h_k$ are old values, we have no control over the first part $(A_k - A_{k-1})h_k$. In order to minimize $M_k(x) - M_{k-1}(x)$, we try to choose A_k so that

$$(A_k - A_{k-1})t_k = 0$$

for all $t_k \in \mathbb{R}^n$, $h_k^T t_k = 0$. This requires that $A_k - A_{k-1}$ to be a rank-one matrix of the form

$$A_k - A_{k-1} = u_k h_k^T$$

for some $u_k \in \mathbb{R}^n$.

Since

$$(A_k - A_{k-1})h_k = A_k h_k - A_{k-1}h_k = y_k - A_{k-1}h_k,$$

both y_k and $A_{k-1}h_k$ are old values, we have no control over the first part $(A_k - A_{k-1})h_k$. In order to minimize $M_k(x) - M_{k-1}(x)$, we try to choose A_k so that

$$(A_k - A_{k-1})t_k = 0$$

for all $t_k \in \mathbb{R}^n$, $h_k^T t_k = 0$. This requires that $A_k - A_{k-1}$ to be a rank-one matrix of the form

$$A_k - A_{k-1} = u_k h_k^T$$

for some $u_k \in \mathbb{R}^n$. Then

$$u_k h_k^T h_k = (A_k - A_{k-1})h_k = y_k - A_{k-1}h_k$$

which gives

$$u_k = \frac{y_k - A_{k-1}h_k}{h_k^T h_k}.$$

which gives

$$u_k = \frac{y_k - A_{k-1}h_k}{h_k^T h_k}.$$

Therefore,

$$A_k = A_{k-1} + \frac{(y_k - A_{k-1}h_k)h_k^T}{h_k^T h_k} \tag{1}$$

which gives

$$u_k = \frac{y_k - A_{k-1}h_k}{h_k^T h_k}.$$

Therefore,

$$A_k = A_{k-1} + \frac{(y_k - A_{k-1}h_k)h_k^T}{h_k^T h_k} \tag{1}$$

After A_k is determined, the new iterate x_{k+1} is derived from solving $M_k(x) = 0$.

which gives

$$u_k = \frac{y_k - A_{k-1} h_k}{h_k^T h_k}.$$

Therefore,

$$A_k = A_{k-1} + \frac{(y_k - A_{k-1} h_k) h_k^T}{h_k^T h_k} \tag{1}$$

After A_k is determined, the new iterate x_{k+1} is derived from solving $M_k(x) = 0$. It can be done by first noting that

$$h_{k+1} = x_{k+1} - x_k \quad \Longrightarrow \quad x_{k+1} = x_k + h_{k+1}$$

and

$$M_k(x_{k+1}) = 0 \quad \Rightarrow \quad F(x_k) + A_k(x_{k+1} - x_k) = 0 \quad \Rightarrow \quad A_k h_{k+1} = -F(x_k)$$

These formulations give the Broyden's method.

Algorithm 2 (Broyden's Method) *Given a n-variable nonlinear function $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, an initial iterate x_0 and initial Jacobian matrix $A_0 \in \mathbb{R}^{n\times n}$ (e.g., $A_0 = I$), this algorithm finds the solution for $F(x) = 0$.*

for $k = 0, 1, \cdots,$ **do**

Solve $A_k h_{k+1} = -F(x_k)$ for h_{k+1}

Update $x_{k+1} = x_k + h_{k+1}$

Compute $y_{k+1} = F(x_{k+1}) - F(x_k)$

Update

$$A_{k+1} = A_k + \frac{(y_{k+1} - A_k h_{k+1})h_{k+1}^T}{h_{k+1}^T h_{k+1}} = A_k + \frac{(y_{k+1} + F(x_k))h_{k+1}^T}{h_{k+1}^T h_{k+1}}$$

end for

Solve the linear system $A_k h_{k+1} = -F(x_k)$ for h_{k+1}:

Solve the linear system $A_k h_{k+1} = -F(x_k)$ for h_{k+1}:

☞ LU-factorization: cost $\frac{2}{3}n^3 + O(n^2)$ floating-point operations.

Solve the linear system $A_k h_{k+1} = -F(x_k)$ for h_{k+1}:

- ☞ LU-factorization: cost $\frac{2}{3}n^3 + O(n^2)$ floating-point operations.
- ☞ Applying the Shermann-Morrison-Woodbury formula

$$\left(B + UV^T\right)^{-1} = B^{-1} - B^{-1}U\left(I + V^T B^{-1} U\right)^{-1} V^T B^{-1}$$

to (1),

Sol. of Non-linear Systems

Solve the linear system $A_k h_{k+1} = -F(x_k)$ for h_{k+1}:

- ☞ LU-factorization: cost $\frac{2}{3}n^3 + O(n^2)$ floating-point operations.
- ☞ Applying the Shermann-Morrison-Woodbury formula

$$\left(B + UV^T\right)^{-1} = B^{-1} - B^{-1}U\left(I + V^T B^{-1} U\right)^{-1} V^T B^{-1}$$

to (1), we have

$$A_k^{-1} = A_{k-1}^{-1} + \frac{(h_k - A_{k-1}^{-1} y_k) h_k^T A_{k-1}^{-1}}{h_k^T A_{k-1}^{-1} y_k}.$$