Numerical Optimization

Lecture Notes #25

Nonlinear Equations — Introduction & Local Algorithms

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://terminus.sdsu.edu/

Fall 2018

Outline

Nonlinear Equations: Introduction

Often we are asked to find values of the model parameters so that the model satisfies a number of fixed relationships — In the **special case** when we have n parameters and n relationships, we get a **system of nonlinear equations**.

We can formulate this problem mathematically as

$$\bar{\mathbf{r}}(\bar{\mathbf{x}}) = \bar{\mathbf{0}},$$

where $\bar{\mathbf{r}} : \mathbb{R}^n \to \mathbb{R}^n$ is a vector function, *i.e.* for $\bar{\mathbf{x}} \in \mathbb{R}^n$

$$\bar{\mathbf{r}}(\bar{\mathbf{x}}) = \begin{bmatrix} r_1(\bar{\mathbf{x}}) \\ r_2(\bar{\mathbf{x}}) \\ \vdots \\ r_n(\bar{\mathbf{x}}) \end{bmatrix}.$$

Nonlinear Equations: Introduction

A system of nonlinear equations

$$\bar{\mathbf{r}}(\bar{\mathbf{x}}) = \begin{bmatrix} r_1(\bar{\mathbf{x}}) \\ r_2(\bar{\mathbf{x}}) \\ \vdots \\ r_n(\bar{\mathbf{x}}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

may have

- No solutions.
- A unique solution.
- Many (possibly infinitely many) solutions.

Nonlinear Equations: Introduction

In the process of finding the solution(s), or **root(s)**, to systems of nonlinear equations we can reuse many of the ideas discussed in the context of unconstrained minimization.

One approach is to solve the least-squares-problem

$$\bar{\mathbf{x}}^* = \underset{\bar{\mathbf{x}}\in\mathbb{R}^n}{\arg\min}\left[\frac{1}{2}\sum_{i=1}^{n} r_i^2(\bar{\mathbf{x}})\right],$$

which clearly has a minimum at $\bar{\mathbf{x}}^*$ if $\bar{\mathbf{r}}(\bar{\mathbf{x}}^*) = 0$.

The connection between least squares problems and the solution of nonlinear equations is quite strong.

Nonlinear Equations — Least Squares

There are some key differences between solving nonlinear equations and solving general nonlinear least squares problems:

- In nonlinear equations, the number of equations (m in the least squares formulation) **equals** the number of variables ($\bar{\mathbf{x}} \in \mathbb{R}^n$), whereas in the typical least-squares situation $m \gg n$.
- For nonlinear equations, at the optimum $\bar{\mathbf{r}}(\bar{\mathbf{x}}^*) = 0$, whereas the minimum value of a general least squares problem is not required to reach zero.
- Often, the equations $r_i(\bar{\mathbf{x}}) = 0$ represent physical or economical constraints, such as conservation laws or consistency principles, which must hold exactly in order for the solution to be meaningful.

Nonlinear Equations — Example #1: **Gaussian Quadrature**

Suppose we want to find an optimal two-point formula:

$$\int_{-1}^{1} f(x)\,dx = c_1 f(x_1) + c_2 f(x_2).$$

Since we have 4 parameters to play with $\{x_1, x_2, c_1, c_2\}$, we can generate a formula that is exact up to polynomial of degree 3. We get the following 4 equations:

$$\begin{aligned}
\int_{-1}^{1} 1\,dx &= 2 = c_1 + c_2 \\
\int_{-1}^{1} x\,dx &= 0 = c_1x_1 + c_2x_2 \\
\int_{-1}^{1} x^2\,dx &= \tfrac{2}{3} = c_1x_1^2 + c_2x_2^2 \\
\int_{-1}^{1} x^3\,dx &= 0 = c_1x_1^3 + c_2x_2^3
\end{aligned}
\qquad\Bigg\|\qquad
\begin{aligned}
r_0(\circ) &= c_1 + c_2 - 2 \\
r_1(\circ) &= c_1x_1 + c_2x_2 \\
r_2(\circ) &= c_1x_1^2 + c_2x_2^2 - \tfrac{2}{3} \\
r_3(\circ) &= c_1x_1^3 + c_2x_2^3
\end{aligned}$$

Nonlinear Equations — Example #1: **Gaussian Quadrature**

Hence, we are looking for the vector

$$\bar{\mathbf{s}}^* = \begin{bmatrix} c_1^* \\ c_2^* \\ x_1^* \\ x_2^* \end{bmatrix}, \quad \text{for which} \quad \bar{\mathbf{r}}(\bar{\mathbf{s}}^*) = \begin{bmatrix} c_1^* + c_2^* - 2 \\ c_1^*x_1^* + c_2^*x_2^* \\ c_1^*[x_1^*]^2 + c_2^*[x_2^*]^2 - \frac{2}{3} \\ c_1^*[x_1^*]^3 + c_2^*[x_2^*]^3 \end{bmatrix} = 0$$

In this instance, the solution is given by

$$\begin{aligned} c_1^* &= 1 \\ c_2^* &= 1 \\ x_1^* &= -\frac{\sqrt{3}}{3} \\ x_2^* &= \frac{\sqrt{3}}{3} \end{aligned}$$

If we want a 3-point formula accurate to up to 5th degree polynomials, we get a system with 6 unknowns containing nonlinear terms of the type $c_i x_i^5 \ldots$

Nonlinear Equations — Example #$(1+\epsilon)$: **Gaussian Quadrature**

We can generate infinitely many examples of nonlinear equations by looking for the optimal (Gaussian Quadrature) placements and weight for k points:

$$\int_{-1}^{1} f(x)\,dx = \sum_{j=1}^{k} c_j f(x_j)$$

We can generate a numerical integration rule which is exact for polynomials up to degree $(2k-1)$ by solving the system of $(2k)$ equations and $(2k)$ unknowns ($\vec{c} \in \mathbb{R}^k, \vec{x} \in \mathbb{R}^k$):

$$r_n^{(k)}(\vec{c},\vec{x}) = \left(\left[\sum_{j=1}^{k} c_j x_j^n \right] - \left[\frac{1-(-1)^{n+1}}{n+1} \right] \right); \quad n = 0, \dots, (2k-1).$$

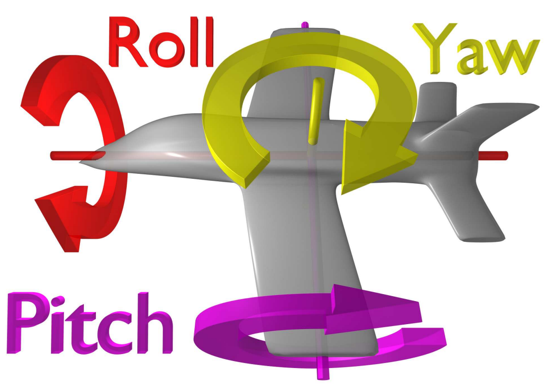
Nonlinear Equations — Example #2: **Aircraft Stability**

We can use the following 8 parameters to model the behavior of an aircraft:

State	x_1	The roll of the aircraft
	x_2	The pitch of the aircraft
	x_3	The yaw of the aircraft
	x_4	The incremental angle of attack
	x_5	The side-slip angle
Controls	x_6	Deflection of the elevator
	x_7	Deflection of the aileron
	x_8	Deflection of the rudder

x_1 through x_5 describe the state of the aircraft, and x_6 through x_8 are the controls.

Nonlinear Equations — Example #2: **Aircraft Stability**



Nonlinear Equations — Example #2: **Aircraft Stability**

Sideslip Angle [Wikipedia]

The sideslip angle relates the rotation of the aircraft centerline from the relative wind. In flight dynamics it is given the shorthand notation (β) and is usually assigned to be "positive" when the relative wind is coming from the right of the nose of the airplane. The sideslip angle is essentially the directional angle of attack of the airplane. It is the primary parameter in stability considerations.

Nonlinear Equations — Example #2: **Aircraft Stability**

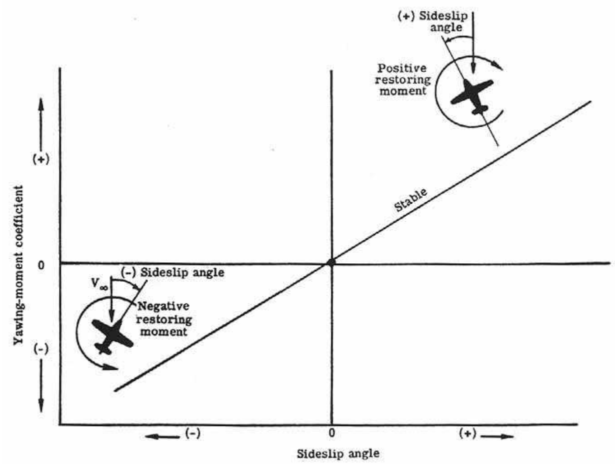


Figure: For more information on Aerodynamics (the theories of flight), visit http://www.centennialofflight.gov/essay_cat/9.htm at the "History of Flight" website presented by the U.S. Centennial of Flight Commission.

Nonlinear Equations — Example #2: **Aircraft Stability**

Using these 8 parameters, we can describe the force-balance equilibrium for an aircraft using the following model with 5 equations and 8 unknowns:

$$\bar{\mathbf{F}}(\bar{\mathbf{x}}) = A\bar{\mathbf{x}} + \bar{\boldsymbol{\Phi}}(\bar{\mathbf{x}}) = 0$$

Where A is a 5×8 matrix, and $\bar{\boldsymbol{\Phi}}(\bar{\mathbf{x}})$ the nonlinear term:

$$\bar{\boldsymbol{\Phi}}(\bar{\mathbf{x}}) = \begin{bmatrix} -0.727x_2x_3 + 8.39x_3x_4 - 684.4x_4x_5 + 63.5x_4x_2 \\ 0.949x_1x_3 + 0.173x_1x_5 \\ -0.716x_1x_2 - 1.578x_1x_4 + 1.132x_4x_2 \\ -x_1x_5 \\ x_1x_4 \end{bmatrix}$$

For each setting of the controls $[x_6, x_7, x_8]^T$ we can solve for the behavior $[x_1, x_2, x_3, x_4, x_5]^T$ of the aircraft.

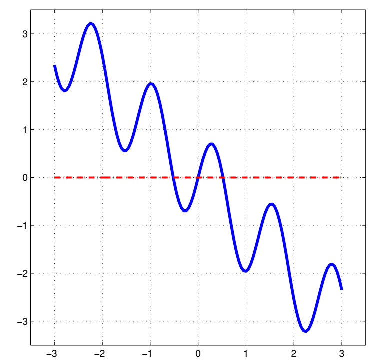
[link]

Nonlinear Equations — Unconstrained Optimization

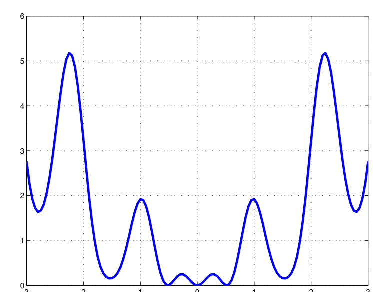
- To get **quadratic convergence** for solution of nonlinear equations (NLEs), we only need information about the first order derivatives (since the small-residual case applies at the solution), whereas for general unconstrained optimization (UCO) problems we need second order information.
- Therefore, quasi-Newton methods plays a smaller role in the solution of NLEs.
- In UCO, the objective function is the natural **merit function** (which indicates progress toward the optimum). In NLEs, there are various ways of selecting the merit function.
- For UCO, line-search and trust-region methods are equally important (successful) solution strategies. However, in the NLE case the trust-region approach tends to be more successful.

Nonlinear Equations: An Added Difficulty

Ponder the one-dimensional nonlinear equation problem



$$\mathbf{r}(\mathbf{x}) = \sin(\mathbf{5x}) - \mathbf{x} = \mathbf{0}$$



$$\mathbf{r}^2(\mathbf{x})$$

We notice that this nonlinear problem has three solutions **(roots)** — $\{0, \pm 0.519148\ldots\}$.

Nonlinear Equations: An Added Difficulty

This is not really news — in unconstrained optimization, we can have several **local minima** (stationary points).

In the optimization case we can distinguish the points by looking at the value of the objective — thus qualifying what stationary point is a "better" solution.

However, for nonlinear equations, we cannot distinguish the roots — they are all of the same "mathematical quality." This means that we must be careful when we construct our models, so that they do not allow for non-physical solutions.

Nonlinear Equations: Algorithms

We will look at the following solution strategies for nonlinear equations:

- Newton's method
- Broyden's quasi-Newton method
- Inexact Newton methods
- Tensor methods

We look at local convergence properties (convergence rate), and address global convergence (how robust is the method(s) with respect to starting "far away" from the solution).

Assumptions and Language

We make the **assumption** that the Jacobian $J(\bar{\mathbf{x}}) = \nabla \bar{\mathbf{r}}(\bar{\mathbf{x}})$ exists and is continuous.

In a couple of results we must assume (the stronger condition) that the Jacobian is **Lipschitz continuous**, *i.e.*

$$\|J(\bar{\mathbf{x}}) - J(\bar{\mathbf{y}})\| \leq \beta_L \|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|, \quad \text{for some } \beta_L > 0$$

A solution $\bar{\mathbf{x}}^* \in \mathbb{R}^n$ satisfying $\bar{\mathbf{r}}(\bar{\mathbf{x}}^*) = 0$ is said to be

a degenerate solution	if $J(\bar{\mathbf{x}}^*)$ is singular
a non-degenerate solution	if $J(\bar{\mathbf{x}}^*)$ is not singular

Newton's Method for Nonlinear Equations

As usual, our discussion is based on **Taylor's theorem**. The version of the theorem that is relevant to this discussion takes the form

Theorem

Suppose that $\bar{\mathbf{r}} : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable in some convex open set $\mathcal{D}$ and that $\bar{\mathbf{x}}$ and $\bar{\mathbf{x}} + \bar{\mathbf{p}}$ are vectors in $\mathcal{D}$. Then we have that

$$\bar{\mathbf{r}}(\bar{\mathbf{x}} + \bar{\mathbf{p}}) = \bar{\mathbf{r}}(\bar{\mathbf{x}}) + \int_0^1 J(\bar{\mathbf{x}} + t\bar{\mathbf{p}})\bar{\mathbf{p}}\, dt$$

We can define a **linear model** $M_k(\bar{\mathbf{p}})$ of $\bar{\mathbf{r}}(\bar{\mathbf{x}}_k + \bar{\mathbf{p}})$ by approximating the integral term in Taylor's theorem by $J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}$, *i.e.*

$$M_k(\bar{\mathbf{p}}) \stackrel{\text{def}}{=} \bar{\mathbf{r}}(\bar{\mathbf{x}}_k) + J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}.$$

Error in the Model

The difference between the model and $\bar{\mathbf{r}}(\bar{\mathbf{x}}_k + \bar{\mathbf{p}})$ is

$$\bar{\mathbf{r}}(\bar{\mathbf{x}}_k + \bar{\mathbf{p}}) - M_k(\bar{\mathbf{p}}) = \int_0^1 \Big[J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}}) \Big] \bar{\mathbf{p}}\, dt$$

since (by assumption) the Jacobian is continuous, we have

$$\lim_{\|\bar{\mathbf{p}}\| \to 0} \|J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}})\| = 0, \quad \forall t \in [0,1]$$

therefore

$$\left\| \int_0^1 \Big[J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}}) \Big] \bar{\mathbf{p}}\, dt \right\| \leq \int_0^1 \Big\| J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}}) \Big\| \cdot \|\bar{\mathbf{p}}\|\, dt = o(\|\bar{\mathbf{p}}\|)$$

Error in the Model

With continuity of the Jacobian we have

$$\left\| \int_0^1 \Big[J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}}) \Big] \bar{\mathbf{p}}\, dt \right\| \quad = \quad o(\|\bar{\mathbf{p}}\|).$$

with **Lipschitz continuity** we get the stronger result

$$\left\| \int_0^1 \Big[J(\bar{\mathbf{x}} + t\bar{\mathbf{p}}) - J(\bar{\mathbf{x}}) \Big] \bar{\mathbf{p}}\, dt \right\| \quad = \quad \mathcal{O}\left(\|\bar{\mathbf{p}}\|^2\right).$$

The "pure" form of Newton's method chooses the step $\bar{\mathbf{p}}_k$ to be the vector for which $M(\bar{\mathbf{p}}_k) = 0$, *i.e.*

$$\bar{\mathbf{p}}_k = -J(\bar{\mathbf{x}}_k)^{-1} \bar{\mathbf{r}}(\bar{\mathbf{x}}_k).$$

"Big Oh" — $\mathcal{O}(\cdot)$ and "Small Oh" — $o(\cdot)$

Definition

Let f and g be two functions from a subset X of the real numbers to the real numbers. Assume that g takes positive values. We say $f = \mathcal{O}(g)$ ("f is **Big-Oh** of g") if there is a real constant A such that $|f(x)| < A|g(x)|$ for all $x \in X$.

Definition

If X is a half-open interval $[r, \infty)$, or some subset of such an interval, and $f = \mathcal{O}(g)$ on this X, then we sometimes say $f = \mathcal{O}(g)$ as $x \to \infty$. Similarly, if X is an interval like $(0, r]$ then we might say $f = \mathcal{O}(g)$ as $x \to 0$.

Definition

We say that $f = o(g)$ as $x \to x_0$ ("f is **Little-Oh** of g as x goes to x_0") if the limit as $x \to x_0$ of f/g is zero.

Newton's Method for Nonlinear Equations

Algorithm: Newton's Method

Given a starting point $\bar{\mathbf{x}}_0$

k = 0
while($\|\bar{\mathbf{r}}(\bar{\mathbf{x}}_k)\| > \epsilon$)
  $J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}_k = -\bar{\mathbf{r}}(\bar{\mathbf{x}}_k)$ (solve for $\bar{\mathbf{p}}_k$) [1]
  $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k + \bar{\mathbf{p}}_k$ [add α-search for improved stability]
end(k = k + 1)

- Newton's method for unconstrained optimization can be derived from this algorithm by application to $\nabla f(\bar{\mathbf{x}}) = 0$.
- When $J(\bar{\mathbf{x}}_k)$ is non-singular, then [1] is equivalent to $J(\bar{\mathbf{x}}_k)^T J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}_k^{\text{GN}} = -J(\bar{\mathbf{x}}_k)^T \bar{\mathbf{r}}(\bar{\mathbf{x}}_k)$, which gives the Gauss-Newton direction for non-linear least squares.

Newton's Method for Nonlinear Equations

Properties & Problems

- Assuming that the iterate $\bar{\mathbf{x}}_k$ is close to a **non-degenerate** root $\bar{\mathbf{x}}^*$, Newton's method has local **super-linear convergence** when the Jacobian is a continuous function of $\bar{\mathbf{x}}$, and local **quadratic convergence** of the Jacobian is Lipschitz continuous.
- When $\|\bar{\mathbf{x}}_0 - \bar{\mathbf{x}}^*\|$ is large, the "pure" Newton algorithm can behave erratically. When $J(\bar{\mathbf{x}}_k)$ is singular, the Newton step is not even defined.
- When n is large it may be expensive to compute the Newton step $\bar{\mathbf{p}}_k$.
- The root $\bar{\mathbf{x}}^*$ may be degenerate, *i.e.* $J(\bar{\mathbf{x}}^*)$ may be singular. *E.g.* $r(x) = x^2$ has a single degenerate root $x^* = 0$. For any non-zero starting point x_0, the sequence of iterates is given by $x_k = x_0/2^k$, which converges to the solution but only at a linear rate.

Modifications & Improvements: Inexact Newton Methods

Instead of solving the linear system

$$J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}_k = -\bar{\mathbf{r}}(\bar{\mathbf{x}}_k)$$

exactly, **inexact Newton methods** use search directions $\bar{\mathbf{p}}_k$ which satisfy the condition

$$\|\bar{\mathbf{r}}(\bar{\mathbf{x}}_k) + J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}_k\| \leq \eta_k\|\bar{\mathbf{r}}(\bar{\mathbf{x}}_k)\| \quad \eta_k \in [0,\eta], \quad \eta \in [0,1),$$

where $\{\eta_k\}$ is the **forcing sequence**.

The convergence properties of inexact Newton methods depend only on the forcing sequence, not on the particular method used to get $\bar{\mathbf{p}}_k$.

Inexact Newton Methods: Comments

Usually inexact Newton methods are based on iterative techniques for solving the linear system

$$J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}}_k = -\bar{\mathbf{r}}(\bar{\mathbf{x}}_k).$$

Here, since $J(\bar{\mathbf{x}}_k)$ is not symmetric positive definite, we cannot directly apply the conjugate gradient method.

Inexact Newton Methods: Local Convergence

Theorem

Suppose that $\bar{\mathbf{r}}$ is continuously differentiable in a convex open set $\mathcal{D} \subset \mathbb{R}^n$. Let $\bar{\mathbf{x}}^* \in \mathcal{D}$ be a non-degenerate solution of $\bar{\mathbf{r}}(\bar{\mathbf{x}}) = 0$, and let $\{\bar{\mathbf{x}}_k\}$ be the sequence of iterates generated by the inexact Newton iteration. Then when $\bar{\mathbf{x}}_k \in \mathcal{D}$ is sufficiently close to $\bar{\mathbf{x}}^*$, the following are true:

- **(i)** *If η is sufficiently small, then the convergence of $\{\bar{\mathbf{x}}_k\}$ is linear.*
- **(ii)** *If $\eta_k \to 0$, then the convergence of $\{\bar{\mathbf{x}}_k\}$ is superlinear.*
- **(iii)** *If, in addition, $J(\cdot)$ is Lipschitz continuous on $\mathcal{D}$ and $\eta_k = \mathcal{O}(\|\bar{\mathbf{r}}_k\|)$, then the convergence of $\{\bar{\mathbf{x}}_k\}$ is quadratic.*

Broyden's Method

Secant Methods / quasi-Newton Methods

Secant methods (aka quasi-Newton methods), do not require calculation of the Jacobian. — Instead, they maintain an approximation of the Jacobian which gets updated in each iteration.

This sounds quite familiar — compare with the BFGS-method for unconstrained optimization

We present Broyden's (the "B" in BFGS) method for this approach.

Let $B_k \approx J(\bar{\mathbf{x}}_k)$ be the Jacobian approximation at iteration k, assuming it is non-singular we can find the next step

$$\bar{\mathbf{p}}_k = -B_k^{-1}\bar{\mathbf{r}}(\bar{\mathbf{x}}_k), \qquad \bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k + \alpha_k \bar{\mathbf{p}}_k.$$

Broyden's Method

The Secant Equation

We let $\bar{\mathbf{s}}_k$, and $\bar{\mathbf{y}}_k$ be the differences between successive iterates, and residuals, respectively:

$$\bar{\mathbf{s}}_k = \bar{\mathbf{x}}_{k+1} - \bar{\mathbf{x}}_k, \qquad \bar{\mathbf{y}}_k = \bar{\mathbf{r}}_{k+1} - \bar{\mathbf{r}}_k.$$

From Taylor's theorem we have the following relation

$$\bar{\mathbf{y}}_k = \int_0^1 J(\bar{\mathbf{x}}_k + t\bar{\mathbf{s}}_k)\bar{\mathbf{s}}_k \, dt \approx J(\bar{\mathbf{x}}_{k+1})\bar{\mathbf{s}}_k + o(\|\bar{\mathbf{s}}_k\|).$$

Hence, we require the updated Jacobian approximation B_{k+1} to satisfy the **secant equation**

$$\bar{\mathbf{y}}_\mathbf{k} = \mathbf{B}_{\mathbf{k+1}}\bar{\mathbf{s}}_\mathbf{k}.$$

Broyden's Method

The Update

The secant equation is a system of n equation, with n^2 unknowns, hence if $n > 1$ there are many ways to satisfy the equation.

Broyden's update makes the smallest possible change in the Jacobian approximation, measured in the Euclidean norm $\|B_k - B_{k+1}\|$, that is consistent with the secant equation. It takes the form

$$\mathbf{B}_{\mathbf{k+1}} = \mathbf{B}_\mathbf{k} + \frac{(\bar{\mathbf{y}}_\mathbf{k} - \mathbf{B}_\mathbf{k}\bar{\mathbf{s}}_\mathbf{k})\bar{\mathbf{s}}_\mathbf{k}^\mathbf{T}}{\bar{\mathbf{s}}_\mathbf{k}^\mathbf{T}\bar{\mathbf{s}}_\mathbf{k}}.$$

Broyden's algorithm: Given the direction

$$\bar{\mathbf{p}}_k = -B_k^{-1}\bar{\mathbf{r}}(\bar{\mathbf{x}}_k)$$

we perform a line-search in this direction, and then proceed as expected.

Broyden's Method

Local Convergence

Theorem

Suppose that $\bar{\mathbf{r}}$ is continuously differentiable in a convex open set $\mathcal{D} \subset \mathbb{R}^n$. Let $\bar{\mathbf{x}}^* \in \mathcal{D}$ be a non-degenerate solution of $\bar{\mathbf{r}}(\bar{\mathbf{x}}) = 0$. Then there are positive constants ϵ and δ such that if the starting point $\bar{\mathbf{x}}_0$ and the starting approximate Jacobian B_0 satisfy

$$\|\bar{\mathbf{x}}_0 - \bar{\mathbf{x}}^*\| \le \delta, \qquad \|B_0 - J(\bar{\mathbf{x}}^*)\| \le \epsilon$$

the sequence $\{\bar{\mathbf{x}}_k\}$ generated by the Broyden iteration is well-defined and converges super-linearly to $\bar{\mathbf{x}}^*$.

The second condition is particularly troublesome in practice. A good B_0 is **critical** to the performance of Broyden's method. $B_0 = J(\bar{\mathbf{x}}_0)$ may be called for (but may not be sufficiently good).

B_k is dense in general, even when $J(\bar{\mathbf{x}}_k)$ is sparse; when n is large, this may cause storage problems.

Tensor Methods

In tensor methods, the linear model $M_k(\bar{\mathbf{p}})$ used by Newton's method is augmented with an extra term. The goal of this term is to capture some of the non-linear behavior of $\bar{\mathbf{r}}(\bar{\mathbf{x}})$, and facilitate faster and more robust **convergence to degenerate roots**.

Tensor methods are most successful when

$$\text{rank}(J(\bar{\mathbf{x}}^*)) \in \{n-1, n-2\}.$$

The tensor model

$$\widehat{M}_k(\bar{\mathbf{p}}) = \bar{\mathbf{r}}(\bar{\mathbf{x}}_k) + J(\bar{\mathbf{x}}_k)\bar{\mathbf{p}} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{T_k}\bar{\mathbf{p}}\bar{\mathbf{p}},$$

where T_k is a tensor defined by n^3 elements $(T_k)_{ijl}$. The ith component of the action of the tensor on two vectors $\bar{\mathbf{u}}, \bar{\mathbf{v}} \in \mathbb{R}^n$ is defined by

$$(T_k\bar{\mathbf{u}}\bar{\mathbf{v}})_i = \sum_{j=1}^{n}\sum_{l=1}^{n}(T_k)_{ijl}u_j v_l.$$

Tensor Methods

Newton's method inspires us to build the tensor from Hessians, *i.e.*

$$(T_k)_{ijl} = [\nabla^2 r_i(\bar{\mathbf{x}}_k)]_{jl}.$$

This is, in most applications, prohibitively expensive.

Another approach is to select (T_k) such that $\widehat{M}_k(\bar{\mathbf{p}})$ interpolates the function $\bar{\mathbf{r}}(\bar{\mathbf{x}}_k + \bar{\mathbf{p}})$ at some previous iterates, *i.e.*

$$\widehat{M}_k(\bar{\mathbf{x}}_{k-j} - \bar{\mathbf{x}}_k) = r(\bar{\mathbf{x}}_{k-j}), \quad j = 1, 2, \ldots, q.$$

This gives

$$\frac{1}{2}T_k\bar{\mathbf{s}}_{jk}\bar{\mathbf{s}}_{jk} = \bar{\mathbf{r}}(\bar{\mathbf{x}}_{k-j}) - \bar{\mathbf{r}}(\bar{\mathbf{x}}_k) - J(\bar{\mathbf{x}}_k)\bar{\mathbf{s}}_{jk}, \quad \bar{\mathbf{s}}_{jk} = \bar{\mathbf{x}}_{k-j} - \bar{\mathbf{x}}_k,$$

which defines the tensor action of the form

$$T_k\bar{\mathbf{u}}\bar{\mathbf{v}} = \sum_{j=1}^{q} a_j(\bar{\mathbf{s}}_{jk}^T\bar{\mathbf{u}})(\bar{\mathbf{s}}_{jk}^T\bar{\mathbf{v}}), \qquad a_j \in \mathbb{R}^n.$$

Index