Root-finding in d dimensions – Newton method and Broyden method

Advanced Programming

November 19, 2024

1 Continuous differentiability in d dimensions

Recall that if $f\colon \mathbb{R}^d \to \mathbb{R}^d$ is at least once continuously differentiable, then for every $x \in \mathbb{R}^d$ there exists a *Jacobian* matrix $Df(x)$ depending continuously on x such that

$$f(x + \Delta x) = f(x) + Df(x)\,\Delta x + o(\|\Delta x\|)\,. \tag{1}$$

Here, $o(\|\Delta x\|)$ is shorthand for some function $\phi_x\colon \mathbb{R}^d \to \mathbb{R}^d$, defined for every $x \in \mathbb{R}^d$, such that

$$\lim_{\Delta x \to 0} \frac{\phi_x(\Delta x)}{\|\Delta x\|} = 0\,. \tag{2}$$

The Jacobian is the matrix of partial derivatives,

$$Df(x) = \begin{pmatrix} \partial_1 f_1(x) & \cdots & \partial_d f_1(x) \\ \vdots & & \vdots \\ \partial_1 f_d(x) & \cdots & \partial_d f_d(x) \end{pmatrix}\,. \tag{3}$$

2 Newton's method in d dimensions

We proceed as in the case of Newton's method for functions of a single variable. Suppose that $x_n \in \mathbb{R}^d$ is the current approximation of a root of f. Writing $\Delta x_{n+1} = x_{n+1} - x_n$, we seek to determine x_{n+1} as the root of the linear approximation to f at $x = x_n$. Dropping the higher-order terms in (1), this linear approximation reads

$$f(x_{n+1}) = f(x_n + \Delta x_{n+1}) \approx f(x_n) + Df(x_n)\,\Delta x_{n+1}\,. \tag{4}$$

Setting the right hand side to zero, we obtain the d-dimensional Newton update Δx_{n+1} by solving the linear system

$$Df(x_n)\,\Delta x_{n+1} = -f(x_n)\,. \tag{5}$$

This can be written in the form

$$x_{n+1} = x_n - Df(x_n)^{-1} f(x_n) \tag{6}$$

to emphasize that it is a proper generalization of the familiar formula for Newton's method in one variable. However, when d is large, it is better to solve (5) directly as there are iterative linear solvers that are much faster than the $O(d^3)$ complexity required to compute the inverse Jacobian.

Newton's method can be shown to converge quadratically in a sufficiently small neighborhood of a root. Even though, as in one dimension, there is no guarantee that it converges at all for arbitrary starting points x_0, in practice it is fairly robust and fast.

The drawback is that a Jacobian is required, which may not be readily available: for example, f may not be available in the form of a simple mathematical expression, but is defined implicitly as the output of a complex piece of computer code. A second drawback is that the Jacobian often requires $O(d^2)$ memory which may be too much for very large problems. An example of such a dense Jacobian is the discrete integral equation in Section 5.2 below; on the other hand, the Jacobian in a second example, a discrete boundary value problem described in Section 5.1, is sparse and can be stored more efficiently.

3 Secant-like methods in d dimensions

To generalize the derivation of the secant method from one dimension, let us write J_n to denote the approximation to the Jacobian at iteration n. Writing $f_n = f(x_n)$, setting $\Delta f_n = f_n - f_{n-1}$, and dropping, once again, higher order terms in (1), we obtain the so-called *secant condition*

$$\Delta f_n = J_n \, \Delta x_n \, . \tag{7}$$

This again is a linear system of equations, but the unknowns are the d^2 elements of J_n. On the other hand, we only have d equations, so the system is highly underdetermined when $d > 1$.

Broyden's idea (in a modern interpretation) is to start with a guess for the initial Jacobian J_0. Then, at each step, first update x_n, then use the resulting Δx_n and Δf_n solve an optimization problem for J_n, where the secant condition (7) appears as a constraint. In particular, the "good Broyden method" results from

$$\begin{aligned} &\text{minimize } \|J_n - J_{n-1}\|_F^2 \\ &\text{subject to } J_n \, \Delta x_n = \Delta f_n \end{aligned} \tag{8}$$

which, as shown in Appendix A.3, implies the update

$$J_n = J_{n-1} + \frac{\Delta f_n - J_{n-1} \Delta x_n}{\|\Delta x_n\|^2} \, \Delta x_n^T \, . \tag{9}$$

To avoid the need for matrix inversion (or for solving a linear system) at every step, the Sherman–Morrison formula, see equation (59) in Appendix A.4, can be used to convert this update rule into an update for the inverse matrix,

$$J_n^{-1} = J_{n-1}^{-1} + \frac{\Delta x_n - J_{n-1}^{-1}\Delta f_n}{\Delta x_n^T\, J_{n-1}^{-1}\Delta f_n}\, \Delta x_n^T\, J_{n-1}^{-1}\,. \tag{10}$$

Alternatively, the "bad Broyden method" results from

$$\begin{aligned} &\text{minimize } \|J_n^{-1} - J_{n-1}^{-1}\|_F^2 \\ &\text{subject to } J_n\,\Delta x_n = \Delta f_n \end{aligned} \tag{11}$$

which implies the update

$$J_n^{-1} = J_{n-1}^{-1} + \frac{\Delta x_n - J_{n-1}^{-1}\Delta f_n}{\|\Delta f_n\|^2}\, \Delta f_n^T\,. \tag{12}$$

While the "good" method appears to converge better in many simple test cases, this is not universally true; there are also problems where the "bad" method is better. Moreover, the "bad" method tends to be computationally cheaper. Thus, this historical distinction is not always justified [1].

This gives the following algorithm:

1. Choose an initial guess x_0 for the root and J_0^{-1} for the inverse Jacobian.
2. At step $n = 1, 2, \ldots$, first update the location of the root via
$$x_n = x_{n-1} - J_{n-1}^{-1}\, f(x_{n-1})\,. \tag{13}$$
3. If $f(x_n)$ is smaller than some tolerance (e.g., by testing the maximal absolute value of its components), terminate with success.
4. If not, update the inverse Jacobian via (10) ("good Broyden") or (12) ("bad Broyden") and go to Step 2.

Note: Without loss of generality, we may set $J_0 = I$, the identity matrix. For if another (nonsingular) initial Jacobian J_0 proves to be better, we can replace the original problem $f(x) = 0$ by the equivalent problem $g(x) = 0$ where

$$g(x) = J_0^{-1} f(x)\,. \tag{14}$$

Then the sequence of Broyden updates for f with initial matrix J_0 is identical to the sequence of Broyden updates of g with initial matrix I. Computationally, this inverse can often be computed efficiently by solving a linear system (e.g. for the discrete boundary value problem in Section 5.1 below).

4 Limited memory Broyden methods

In the Broyden method, each update adds a rank-1 matrix to the approximate Jacobian or to the inverse Jacobian. The update of the inverse formulation of the Broyden method can be written

$$J_n^{-1} = J_{n-1}^{-1} + u_n\, v_n^T \tag{15}$$

or, by recursive insertion with $J_0 = I$,

$$J_n^{-1} = I + \sum_{k=1}^{n} u_k\, v_k^T \tag{16}$$

where, e.g. for the "good" Broyden method, see (10),

$$u_k = \frac{\Delta x_k - J_{k-1}^{-1}\Delta f_k}{\Delta x_k^T\, J_{k-1}^{-1}\Delta f_k} \quad \text{and} \quad v_k^T = \Delta x_k^T\, J_{k-1}^{-1}\,. \tag{17}$$

Thus, if $d \gg n$, it is more efficient to store the u_k and v_k. In particular, when $J_0 = I$, matrix-vector products can be written

$$J_n^{-1}x = x + \sum_{k=1}^{n} u_k\, (v_k^T x) \tag{18}$$

where the term in parentheses is an inner product, thus can be evaluated in $O(d)$ elementary arithmetic operations, so that the overall complexity of evaluating (18) is $O(dn)$ which, when $d \gg n$, is much better than the $O(d^2)$ operations required for a full matrix-vector product.

Since both memory requirements and computation time increase with each Broyden iteration, it is often important to keep the rank of the update matrix bounded, say, with maximal rank $p + 1$. This can be done in two ways.

4.1 Simple pruning

When $n > p$, keep only the most recent p vectors in the sum (16). To be precise, for the limited memory Broyden (LMB) method, perform of Steps 1–3 as before, followed by

4'. Retain only $u_{n-p}, \ldots, u_{n-1}$ and $v_{n-p}, \ldots, v_{n-1}$. Then compute a new pair of update vectors u_n and v_n (using (17) for the "good" Broyden method or analogous expressions for the "bad" Broyden method), where the required matrix-vector products use (18), and matrix-vector products use an analogous expression, summing over update vectors with index $k = n - p, \ldots, n - 1$ only. Then go to Step 2, retaining update vectors $u_{n-p}, \ldots, u_n$ and $v_{n-p}, \ldots, v_n$.

4.2 SVD-based pruning

It is often better not to simply discard the oldest update vectors, but rather compute a best rank-p approximation of the update. In principle, a best rank-p approximation with respect to the Frobenius norm is known to be given by the singular value decomposition of the update matrix, retaining only the p largest singular values. To implement this efficiently when $d \gg p$, we proceed as follows (see [2]).

First, notice that (16) can be written as

$$J_n^{-1} = I + UV^T \tag{19}$$

where U is $d \times q$ and V^T is $q \times d$ (with $q = n$ if no pruning was applied in previous steps) with

$$U = \begin{pmatrix} | & & | \\ u_1 & \dots & u_q \\ | & & | \end{pmatrix} \quad \text{and} \quad V^T = \begin{pmatrix} — & v_1^T & — \\ & \vdots & \\ — & v_q^T & — \end{pmatrix} . \tag{20}$$

Then compute the RQ-decomposition of V^T,

$$RQ = V^T , \tag{21}$$

where Q is an orthogonal $q \times d$-matrix and R is $q \times q$. Next, compute the singular value decomposition

$$U\Sigma W^T = UR . \tag{22}$$

Assuming that the singular values in Σ are ordered in decreasing magnitude (as guaranteed in standard library implementations), a new rank-p pair of update matrices is given by the first p columns of $U\Sigma$ and the first p rows of W^TQ, respectively.

Using these reduced update matrices in the expressions for the Broyden update, a new pair of update vectors is generated just as before, and appended to the pruned update matrices.

5 Test problems

The following problems have been used historically as test problems for Broyden-type methods (see, e.g., [2, Appendix A]).

5.1 Discrete boundary value problem

On the interval $I = (0, 1)$, the boundary value problem

$$u''(t) = a\,(u(t) + t + 1)^3 , \tag{23}$$

$$u(0) = u(1) = 0 \tag{24}$$

can be discretized on d subintervals of equal length $h = 1/d$. Here, a is a parameter with default value $a = \frac{1}{2}$. Setting $t_i = ih$ and writing u_i as the discrete approximation to $u(t_i)$, we can write the standard second order difference approximation as

$$\frac{u_{i-1} - 2\,u_i + u_{i+1}}{h^2} - a\,(u_i + t_i + 1)^3 = 0 \tag{25}$$

for $i = 1, \ldots, d-1$, with $u_0 = u_d = 0$. Thus, the true dimension of this system of nonlinear equations is $d - 1$. Note that if the vector of unknowns is denoted $u = (u_1, \ldots, u_{d-1})^T$, the nonlinear system (25) takes the form

$$Bu - N(u) = 0\,, \tag{26}$$

where

$$B = \frac{1}{h^2}\begin{pmatrix} -2 & 1 & & 0 \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & -2 \end{pmatrix} \quad \text{and} \quad N(u) = a\begin{pmatrix} (u_1 + t_1 + 1)^3 \\ \vdots \\ (u_{d-1} + t_{d-1} + 1)^3 \end{pmatrix}. \tag{27}$$

Rearranging terms, (26) can be written in fixed point form,

$$u = B^{-1}N(u) \equiv \Phi(u)\,. \tag{28}$$

Computationally, the inverse of B should not be computed explicitly, as that would destroy the sparse, tri-diagonal structure of B, but rather be implemented by solving a system of linear equations.

5.2 Discrete integral equation

The nonlinear integral equation

$$u(t) + a\int_0^1 H(s,t)\,(u(s) + s + 1)^3\,\mathrm{d}s = 0 \tag{29}$$

kernel

$$H(s,t) = \begin{cases} s\,(1-t) & \text{for } s \le t\,, \\ t\,(1-s) & \text{for } s > t\,, \end{cases} \tag{30}$$

is approximated, on the same partition as before, by the discrete Riemann sum

$$u_i + a\,h\left(\sum_{j=1}^{i}(1-t_i)\,t_j\,(u_j + t_j + 1)^3 + \sum_{j=i+1}^{d} t_i\,(1-t_j)\,(u_j + t_j + 1)^3\right) = 0 \tag{31}$$

for $i = 1, \ldots, d$. As before, the parameter a takes the default value $a = \frac{1}{2}$. Writing $u = (u_1, \ldots, u_d)$ to denote the vector of unknowns, (31) can be written as

$$u - HN(u) = 0 \tag{32}$$

where $H = (h_{ij})$ is the $d \times d$-matrix with matrix elements

$$h_{ij} = \begin{cases} t_j\,(1 - t_i) & \text{for } i \geq j \\ (1 - t_j)\,t_i & \text{for } i < j \end{cases} \tag{33}$$

and

$$N(u) = -a\,h \begin{pmatrix} (u_1 + t_1 + 1)^3 \\ \vdots \\ (u_d + t_d + 1)^3 \end{pmatrix} . \tag{34}$$

Thus, this test problem also has a fixed point form similar to (28), namely

$$u = HN(u) \equiv \Phi(u)\,. \tag{35}$$

A Technical details

A.1 The trace of a matrix

Matrix optimization problems involving the Frobenius norm can often be written in a coordinate-free way using the trace. This leads to shorter and more concise computations, and will be the strategy pursued here.

Given a square matrix $A = (a_{ij})_{ij} \in \mathbb{R}^{d\times d}$, its *trace* is defined as the sum of its diagonal elements, i.e.

$$\operatorname{Tr} A = \sum_{i=1}^{d} a_{ii}\,. \tag{36}$$

The following properties, which are easy to check and will be stated without proof, are used:

(i) The trace is a linear map,

(ii) $\operatorname{Tr} A = \operatorname{Tr} A^T$ (invariance under transposition),

(iii) $\operatorname{Tr}(AB) = \operatorname{Tr}(BA)$ for all $A \in \mathbb{R}^{d\times k}$ and $B \in \mathbb{R}^{k\times d}$.

Property (iii) implies immediately that the trace is invariant under all cyclic permutations of matrix products and that the inner product of two vectors $u, v \in \mathbb{R}^d$ can be written

$$u^T v = \operatorname{Tr}(u^T v) = \operatorname{Tr}(v u^T)\,. \tag{37}$$

(iv) $\langle A, B\rangle_F = \operatorname{Tr}(AB^T)$ defines an inner product on the vector space of $d \times k$ matrices.

This inner product is called the *Frobenius inner product*, and the associated norm the *Frobenius norm*,

$$\|A\|_F^2 = \operatorname{Tr}(AA^T) = \sum_{i=1}^{d}\sum_{j=1}^{k} a_{ij}^2\,. \tag{38}$$

A.2 Lagrange multipliers

Recall the Lagrange multiplier theorem from Analysis II: Let $f \in C^1(\mathbb{R}^d, \mathbb{R})$ and $g \in C^1(\mathbb{R}^d, \mathbb{R}^k)$. Suppose $f(x)$ has a local extremum at $x = x^* \in \mathbb{R}^d$ subject to the constraint $g(x) = 0$, and the Jacobian matrix $Dg(x^*)$ has full rank. Then there exists a unique $\lambda \in \mathbb{R}^k$ such that the Lagrangian

$$L(x; \lambda) = f(x) + \lambda^T g(x) \tag{39}$$

has a critical point at $x = x^*$, i.e. $\nabla L(x^*) = 0$.

In practice, the theorem is always applied in reverse order: we compute solutions of $\nabla L(x^*) = 0$ as *candidate* solutions to the constrained extremal problem, then use other means (e.g. convexity, compactness) to argue for sufficiency.

In our setting, the objective function f is the Frobenius norm of a matrix expression. In other words, the domain of f is not $\mathbb{R}^d$, but some space of matrices. While it is certainly possible to identify a space of matrices with some $\mathbb{R}^d$ of matching dimension and use the formulation of the Lagrange multiplier theorem as stated above, this leads to terrible notation and is best avoided in favor of a slightly more abstract approach, which is explained now.

Let M be a normed vector space. Let $x \colon \mathbb{R} \to M$ be a smooth curve in M. Then $x'(t)$ is a tangent vector to the curve for any $t \in \mathbb{R}$. Since M is a vector space, its tangent space is isomorphic to M itself. In particular,

$$\delta x \equiv x'(0) \tag{40}$$

is a tangent vector and as such, is an element from M. Vice versa, any tangent vector is of the form (40). More generally, for a function $f \in C^1(M, \mathbb{R})$, we define the *variational derivative*

$$\delta f = \left.\frac{\mathrm{d} f(x(t))}{\mathrm{d}t}\right|_{t=0} = Df(x(0))\, \delta x\,. \tag{41}$$

Then f has a critical point at $x = x^*$ if and only if $\delta f(x^*) = 0$ for any $\delta x \in M$.

In this setting, the Lagrange multiplier theorem reads: Let $f \in C^1(M, \mathbb{R})$ and $g \in C^1(M, \mathbb{R}^k)$. Suppose $f(x)$ has a local extremum at $x = x^* \in M$ subject to the constraint $g(x) = 0$, and the Jacobian operator $Dg(x^*)$ has full rank. Then there exists a unique $\lambda \in \mathbb{R}^k$ such that the Lagrangian

$$L(x; \lambda) = f(x) + \lambda^T g(x) \tag{42}$$

has a critical point at $x = x^*$, i.e. $\delta L(x^*) = 0$ for any $\delta x \in M$.

Important note: In typical applications in the calculus of variations, this setting is further generalized by taking M to be "admissible sets" that are not vector spaces, but, for example, affine spaces or manifolds. In this case, tangent vectors δx are not elements from M. In this case, one has to think more carefully about the concept of tangent space. Everything above still stands, and the notation is useful, except that the membership of δx needs to be adapted to the correct notion of tangent space.

A.3 Derivation of the "good" Broyden method

Recall that the "good" Broyden method assumes that J_{n-1} is given and J_n is determined by solving the constrained optimization problem

$$\begin{aligned} &\text{minimize } \|J_n - J_{n-1}\|_F^2 = \operatorname{Tr}[(J_n - J_{n-1})(J_n - J_{n-1})^T] \\ &\text{subject to } J_n \,\Delta x_n = \Delta f_n \,. \end{aligned} \tag{43}$$

(Cf. (38) for expressing the Frobenius norm in terms of the trace.) Rewriting the inner product in (42) in terms of the trace, cf. (37), we can write out the Lagrangian for this problem as

$$L(J_n; \lambda) = \tfrac{1}{2} \operatorname{Tr}[(J_n - J_{n-1})(J_n - J_{n-1})^T] + \operatorname{Tr}[(J_n \,\Delta x_n - f_n)\lambda^T] \,. \tag{44}$$

(The factor $\frac{1}{2}$ is a convenience factor that does not change the problem because linear scaling of the objective function does not change the location of the extrema.)

We compute

$$\begin{aligned} \delta L &= \tfrac{1}{2} \operatorname{Tr}[\delta J_n \,(J_n - J_{n-1})^T] + \tfrac{1}{2} \operatorname{Tr}[(J_n - J_{n-1})\,\delta J_n^T] + \operatorname{Tr}[\delta J_n \,\Delta x_n \,\lambda^T] \\ &= \operatorname{Tr}[(J_n - J_{n-1})\,\delta J_n^T] + \operatorname{Tr}[\lambda \,\Delta x_n^T \,\delta J_n^T] \\ &= \operatorname{Tr}[(J_n - J_{n-1} + \lambda \,\Delta x_n^T)\,\delta J_n^T] \end{aligned} \tag{45}$$

where, in the second equality, we have used the invariance of the trace under transposition. Since we can interpret the final expression as an inner product between two matrices, cf. property (iv) above, we see that $\delta L = 0$ for all $\delta J_n \in \mathbb{R}^{d\times d}$ if and only if

$$J_n - J_{n-1} + \lambda \,\Delta x_n^T = 0 \,. \tag{46}$$

To eliminate the Lagrange multiplier λ from this expression, we multiply (46) with Δx_n from the right and use the constraint $J_n \,\Delta x_n = \Delta f_n$:

$$\Delta f_n - J_{n-1} \,\Delta x_n + \lambda \,\Delta x_n^T \,\Delta x_n = 0 \,, \tag{47}$$

so that

$$\lambda = \frac{J_{n-1} \,\Delta x_n - f_n}{\Delta x_n^T \,\Delta x_n} \,. \tag{48}$$

Inserting (48) into (46) and solving for J_n, we obtain the "good" Broyden update (9). The "bad" Broyden update (12) can be derived similarly.

A.4 Derivation of the Sherman–Morrison formula

Suppose we have an invertible matrix $A \in \mathbb{R}^{d\times d}$ and perform a rank-1 update, setting

$$B = A + uv^T \tag{49}$$

for some given vectors $u, v \in \mathbb{R}^d$. We ask the question whether the inverse of A is related to the inverse of B also by a rank-1 update. In other words, do there exist $x, y \in \mathbb{R}^d$ such that the ansatz

$$B^{-1} = A^{-1} + xy^T \tag{50}$$

holds true?

To answer this question, let us verify the properties of the inverse matrix. In order for (50) to hold, we must have

$$I = BB^{-1} = (A + uv^T)(A^{-1} + xy^T) = I + Axy^T + uv^T A^{-1} + uv^T xy^T \tag{51}$$

or, canceling I and multiplying by A^{-1} from the left,

$$xy^T + A^{-1}uv^T A^{-1} + A^{-1}uv^T xy^T = 0\,. \tag{52}$$

We can thus determine the *direction* of x by multiplying (52) with some column vector from the right. Then most factors in the resulting expression will be scalar, except for x in the first term and $A^{-1}u$ in the second and third term. Thus, x must be in the direction of $A^{-1}u$.

Similarly, we must have

$$I = B^{-1}B = (A^{-1} + xy^T)(A + uv^T) = I + A^{-1}uv^T + xy^T A + xy^T uv^T \tag{53}$$

or, canceling I and multiplying by A^{-1} from the right,

$$A^{-1}uv^T A^{-1} + xy^T + xy^T uv^T A^{-1} = 0\,. \tag{54}$$

We can thus determine the *direction* of y by multiplying (54) with some row vector from the left. Then most factors in the resulting expression will be scalar, except for y^T in the second term and $v^T A^{-1}$ in the first and third term. Thus, y^T must be in the direction of $v^T A^{-1}$.

Thus, we have established that (50) can only hold true if

$$xy^T = \alpha\, A^{-1}uv^T A^{-1} \tag{55}$$

for some value of α. To determine α, we multiply (52) or, equivalently, (54) by v^T from the left and by u from the right, to obtain

$$v^T xy^T u + (v^T A^{-1}u)^2 + v^T A^{-1}uv^T xy^T u = 0\,. \tag{56}$$

Likewise, we multiply (55) by v^T from the left and by u from the right, to obtain

$$v^T xy^T u = \alpha\,(v^T A^{-1}u)^2\,. \tag{57}$$

Inserting (57) into (56), we find that

$$\alpha = -\frac{1}{1 + v^T A^{-1}u}\,. \tag{58}$$

This derivation is only formal, as it assumes invertibility of B, but suggests the following theorem: Suppose $A \in \mathbb{R}^{d\times d}$ is invertible and $u, v \in \mathbb{R}^d$. Then $A + uv^T$ is invertible if and only if $1 + v^T A^{-1}u \neq 0$. In this case,

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1}u}\,. \tag{59}$$

A full proof is an exercise in Linear Algebra and left to the reader. Expression (59) is known as the *Sherman–Morrison formula*.

References

[1] A. Griewank, *Broyden updating, the Good and the Bad!* Doc. Math. (2012), 301–315.

[2] B. van de Rotten, *A limited memory Broyden method to solve high-dimensional systems of nonlinear equations*, PhD thesis, Universiteit Leiden, 2003.