

# Nonlinear least-squares and nonlinear equations

Mark S. Gockenbach

## 1 Introduction

Many optimization problems encountered in applications are *nonlinear least-squares* problems. Such problems have the form

$$\min_x \frac{1}{2} \|F(x)\|^2,$$

where  $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$  and (usually)  $m > n$ . Typically  $F$  has the form  $F(x) = m(x) - d$ , where  $m$  describes a mathematical model depending on the parameters  $x$  and  $d$  is observed data. The model is known to have the form  $m(x)$  for theoretical reasons, but the actual values of the parameters must be determined by performing experiments, measuring the data, and choosing the value of  $x$  that gives the best fit to the data. The nonlinear least-squares problem can be regarded as a general unconstrained minimization problem by defining

$$f(x) = \frac{1}{2} \|F(x)\|^2 \tag{1}$$

and minimizing  $f$ . However, as I explain below, problem can (at least under certain circumstances) be solved more efficiently by taking into account the special form of the objective function.

The problem of solving a system of nonlinear equations, represented by  $F(x) = 0$ , where  $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$  (that is,  $m = n$ ), is closely related to the nonlinear least-squares problem. I have already explained Newton's method and Broyden's method for solving  $F(x) = 0$ , given a good estimate of the solution. To globalize these methods, one can use the objective function (1) and a line search to ensure progress when far from the solution.

## 2 Nonlinear least-squares

When  $f$  is defined by (1), then

$$\nabla f(x) = J(x)^T F(x)$$

and

$$\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^m F_i(x) \nabla^2 F_i(x),$$

where  $J(x)$  is the Jacobian of  $F$  at  $x$ . The key observation is the following: By computing  $J(x)$ , one knows not only the gradient of  $f$  but also part of the Hessian. One can use  $J(x)^T J(x)$  as a "free" approximation to  $\nabla^2 f(x)$ , and this approximation is useful for two reasons:

1. The matrix  $J(x)^T J(x)$  is always at least positive semidefinite, and it is positive definite if  $J(x)$  has full rank. Therefore, choosing  $H_k = J(x^{(k)})^T J(x^{(k)})$  leads to descent directions in many cases. (If  $J(x^{(k)})$  is rank-deficient, so that  $H_k$  is singular, then  $H_k$  can be modified directly to make it positive definite, say by adding a multiple of the identity matrix.)

2. If the problem at hand is a *zero-residual* problem ( $F(x^*) = 0$ ) or a *small-residual* problem ( $\|F(x^*)\|$  small), then  $H_k$  is an increasingly good approximation to  $\nabla^2 f(x^{(k)})$  as  $x^{(k)} \rightarrow x^*$ . This implies that a quasi-Newton algorithm using  $H_k = J(x^{(k)})^T J(x^{(k)})$  can achieve fast local convergence.

The approximation  $J(x)^T J(x)$  to  $\nabla^2 f(x)$  is often referred to as the *Gauss-Newton* Hessian and the quasi-Newton algorithm based on it as the Gauss-Newton method. The Gauss-Newton step is

$$x^{(k+1)} = x^{(k)} - \left( J(x^{(k)})^T J(x^{(k)}) \right)^{-1} J(x^{(k)})^T F(x^{(k)}).$$

As the above discussion suggests, the Gauss-Newton method may converge slowly in a large-residual problem, since in such a problem the Gauss-Newton Hessian is a poor approximation to the true Hessian at the solution. A large residual problem can be solved using general unconstrained minimization algorithms.

Since one often does not know in advance if a least-squares problem is large- or small-residual, various algorithms have been proposed that use the Gauss-Newton Hessian, modifying it only if the step it produces fails to yield good progress. The reader can consult Chapter 10 of Nocedal and Wright [1] for details.

### 3 Nonlinear equations

A system of nonlinear equations represented by  $F(x) = 0$ ,  $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ , can be posed as the nonlinear least-squares problem

$$\min_x \frac{1}{2} \|F(x)\|^2.$$

Assuming that the system of equations has a solution, the problem is known to be a zero-residual problem, and it makes sense to apply the Gauss-Newton algorithm. However, in this case  $J(x^{(k)})$  is square and (hopefully) nonsingular, so the Gauss-Newton step is

$$\begin{aligned} x^{(k+1)} &= x^{(k)} - \left( J(x^{(k)})^T J(x^{(k)}) \right)^{-1} J(x^{(k)})^T F(x^{(k)}) \\ &= x^{(k)} - J(x^{(k)})^{-1} J(x^{(k)})^{-T} J(x^{(k)})^T F(x^{(k)}) \\ &= x^{(k)} - J(x^{(k)})^{-1} F(x^{(k)}). \end{aligned}$$

This calculation shows that the Gauss-Newton step for minimizing  $f(x) = (1/2)\|F(x)\|^2$  is the same as the Newton step for solving  $F(x) = 0$ . (On the other hand, the Newton step for minimizing  $f$  is generally different.) Therefore, the analysis presented earlier guarantees fast local convergence, and a line search can be used to ensure global convergence.

There is, however, one problem with the global convergence guarantee: numerical optimization techniques only guarantee global convergence to a *local* minimizer of the objective function. The function  $f(x) = (1/2)\|F(x)\|^2$  can have local, non-global minimizers. Such a “solution”  $x^*$  is useless since  $F(x^*) \neq 0$ . There is really no way around this difficulty in general when using the Gauss-Newton approach. If the algorithm converges to a local, non-global minimizer of  $f$ , it must be restarted with a different  $x^{(0)}$ .

An alternative approach to solving  $F(x) = 0$  that is generally more robust is the *homotopy* method. To apply this method, one must define a smooth function  $H : \mathbb{R}^n \times [0, 1] \rightarrow \mathbb{R}^n$  with the properties that a solution  $x_0$  of  $H(x, 0) = 0$  is known and

$$H(x, 1) = F(x) \text{ for all } x \in \mathbb{R}^n.$$

The goal is then to follow a path  $x_t$  with the property that

$$H(x_t, t) = 0 \text{ for all } t \in [0, 1].$$

Following the path or trajectory  $x_t$  is similar to solving a system of ordinary differential equations. For details, the interested reader can consult Nocedal and Wright [1], Section 11.3, and the references contained therein.

## References

- [1] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999.