Nonlinear Least Squares

Ying Xiong
School of Engineering and Applied Sciences
Harvard University
yxiong@seas.harvard.edu

Created: January 19th, 2014.
Last Updated: January 30th, 2014 (v0.2).

1 Problem Statement

The least squares problem is to find a (local) minimizer for cost function

$$F(\mathbf{x}) = \sum_{i=1}^{m} \left(f_i(\mathbf{x})\right)^2 = \left\|\mathbf{f}\left(\mathbf{x}\right)\right\|^2 = \mathbf{f}\left(\mathbf{x}\right)^\top \mathbf{f}\left(\mathbf{x}\right), \tag{1}$$

where $f_i : \mathbb{R}^n \mapsto \mathbb{R},\ i = 1, \ldots, m$ are given nonlinear functions.

A least squares problem is a special variant of the more general nonlinear programming problem, and the special form provides useful structure that we can exploit. Define

$$\left(\boldsymbol{J}\left(\mathbf{x}\right)\right)_{i,j} = \frac{\partial f_i}{\partial x_j}\left(\mathbf{x}\right) \tag{2}$$

the Jacobian matrix of $\mathbf{f}\left(\mathbf{x}\right)$, then we have

$$\mathbf{F}'\left(\mathbf{x}\right) = 2\boldsymbol{J}\left(\mathbf{x}\right)^\top \mathbf{f}\left(\mathbf{x}\right), \tag{3}$$

$$\boldsymbol{F}''\left(\mathbf{x}\right) = 2\boldsymbol{J}\left(\mathbf{x}\right)^\top \boldsymbol{J}\left(\mathbf{x}\right) + 2\sum_{i=1}^{m} f_i\left(\mathbf{x}\right) \boldsymbol{f}_i''\left(\mathbf{x}\right), \tag{4}$$

which means even when we do not have second-order information of $\mathbf{f}\left(\mathbf{x}\right)$, we still know *something* about $\boldsymbol{F}''\left(\mathbf{x}\right)$ from $\boldsymbol{J}\left(\mathbf{x}\right)$ alone.

2 Algorithms

We make a linear approximation on $\mathbf{f}\left(\mathbf{x}\right)$ near a given $\mathbf{x}$ as

$$\mathbf{f}\left(\mathbf{x}+\mathbf{h}\right) \approx \boldsymbol{\ell}\left(\mathbf{h}\right) = \mathbf{f}\left(\mathbf{x}\right) + \boldsymbol{J}\left(\mathbf{x}\right)\mathbf{h}, \tag{5}$$

which yields

$$\begin{aligned} F\left(\mathbf{x}+\mathbf{h}\right) \approx L\left(\mathbf{h}\right) =& \boldsymbol{\ell}\left(\mathbf{h}\right)^\top \boldsymbol{\ell}\left(\mathbf{h}\right) & (6)\\ =& \mathbf{f}^\top\mathbf{f} + 2\mathbf{h}^\top\boldsymbol{J}^\top\mathbf{f} + \mathbf{h}^\top\boldsymbol{J}^\top\boldsymbol{J}\mathbf{h} & (7)\\ =& F(\mathbf{x}) + 2\mathbf{h}^\top\boldsymbol{J}^\top\mathbf{f} + \mathbf{h}^\top\boldsymbol{J}^\top\boldsymbol{J}\mathbf{h}. & (8) \end{aligned}$$

Note that this is equivalent to perform a second order Taylor expansion on $F\left(\mathbf{x}\right)$ and approximate $\boldsymbol{F}''$ as $2\boldsymbol{J}^\top\boldsymbol{J}$.

2.1 Gauss-Newton algorithm

The Gauss-Newton algorithm minimize (8) directly, with

$$\mathbf{h}_{\text{gn}} = -\left(\boldsymbol{J}^\top \boldsymbol{J}\right)^{-1} \boldsymbol{J}^\top \mathbf{f}. \tag{9}$$

The algorithm has at least two short-comings: (1) $\left(\boldsymbol{J}^\top \boldsymbol{J}\right)$ might be singular and (2) $\mathbf{h}_{\text{gn}}$ might not be a descending direction.

2.2 Levenberg-Marquardt algorithm [1, 2, 3]

Levenberg-Marquardt algorithm is a damped Gaussian-Newton method

$$\mathbf{h}_{\text{lm,1}} = -\left(\boldsymbol{J}^\top \boldsymbol{J} + \mu \boldsymbol{I}\right)^{-1} \boldsymbol{J}^\top \mathbf{f}, \tag{10}$$

or, as suggested by Marquardt

$$\mathbf{h}_{\text{lm,2}} = -\left(\boldsymbol{J}^\top \boldsymbol{J} + \mu \operatorname{diag}\left(\boldsymbol{J}^\top \boldsymbol{J}\right)\right)^{-1} \boldsymbol{J}^\top \mathbf{f}. \tag{11}$$

We write the two forms together as

$$\mathbf{h}_{\text{lm}}^\top = -\left(\boldsymbol{J}^\top \boldsymbol{J} + \mu \boldsymbol{D}\right)^{-1} \boldsymbol{J}^\top \mathbf{f}, \tag{12}$$

where the “damping matrix” $\boldsymbol{D}$ can either be $\boldsymbol{I}$ or diag $\left(\boldsymbol{J}^\top \boldsymbol{J}\right)$.

2.2.1 Choice of damping factor [4]

Define a *gain ratio*

$$\varrho = \frac{F(\mathbf{x}) - F(\mathbf{x} + \mathbf{h}_{\text{lm}})}{L(\mathbf{0}) - L(\mathbf{h}_{\text{lm}})}, \tag{13}$$

where $L(\mathbf{h})$ is defined in (6), and the denominator can be calculated as

$$L(\mathbf{0}) - L(\mathbf{h}_{\text{lm}}) = \mathbf{h}_{\text{lm}}^\top \left(\mu \boldsymbol{D} \mathbf{h}_{\text{lm}} - \boldsymbol{J}^\top \mathbf{f}\right) \tag{14}$$

The update rule for μ will be

$$\mu_{k+1} = \begin{cases} \mu \cdot \max\left\{\frac{1}{3}, 1 - (2\varrho - 1)^3\right\}; \quad \nu = 2 & \text{if } \varrho > 0, \\ \mu \cdot \nu; \quad \nu = 2 \cdot \nu & \text{otherwise.} \end{cases} \tag{15}$$

The initial μ is usually set as $\tau \cdot \max_i \left\{\left(\boldsymbol{J}^\top \boldsymbol{J}\right)_{i,i}\right\}$, where τ is a user specified parameter, which should be a small value, *e.g.* $\tau = 10^{-6}$ if $\mathbf{x}_0$ is a good approximation to the final local minimum, and 10^{-3} or even 1 otherwise.

2.2.2 Algorithm description

Algorithm 1: Levenberg-Marquardt method

```
Input  : $\mathbf{f}(\mathbf{x}), \boldsymbol{J}(\mathbf{x})$: Input function and its Jacobian matrix.
Input  : $\mathbf{x}_0$: Initial guess.
Input  : $\tau$: A parameter specifying initial damping factor, default $10^{-3}$.
Input  : A stopping criterion.
Output: $\mathbf{x}$: A local minimum.
1  $\mathbf{x} = \mathbf{x}_0,\ \mu = \tau \cdot \max_i \left\{ \left( \boldsymbol{J}^\top \boldsymbol{J} \right)_{i,i} \right\},\ \nu = 2.$
2  while the stopping criterion is not met do
3      Calculate $\mathbf{h}_{\text{lm}}$ according to (12).
4      Calculate $\varrho$ according to (13).
5      if $\varrho > 0$ then
6          $\mathbf{x} = \mathbf{x} + \mathbf{h}_{\text{lm}},\ \mu = \mu \cdot \max\left\{ \frac{1}{3}, 1 - (2\varrho - 1)^3 \right\},\ \nu = 2.$
7      else
8          $\mu = \mu \cdot \nu,\ \nu = 2\nu.$
9      end
10 end
```

2.2.3 Implementation notes

1. When the step size $\mathbf{h}_{\text{lm}}$ is very small, the calculation of ϱ in (Step 4) can suffer from numerical underflow. One needs to check whether $L(\mathbf{0}) - L(\mathbf{h}_{\text{lm}}) < \varepsilon$, where ε is the machine's numerical percision, and if so, terminate the algorithm. When the algorithm terminates this way, we are usually very close to a local minimum.

2. Due to possible ill-conditioning, the matrix $\left(\boldsymbol{J}^\top \boldsymbol{J}\right)$ can be singular, and when μ is very small — which happens after a number of consecutive success descent — the matrix $\left(\boldsymbol{J}^\top \boldsymbol{J} + \mu \boldsymbol{I}\right)$ can also be close to singular, which causes numerical issues. To circumvent this, we put a minimum on μ in (Step 6), changing it to $\mu = \max\left\{\mu_{\text{min}}, \mu \cdot \max\left\{\frac{1}{3}, 1 - (2\varrho - 1)^3\right\}\right\}$, in order to make sure $\left(\boldsymbol{J}^\top \boldsymbol{J} + \mu \boldsymbol{I}\right)$ is always well-conditioned. We choose $\mu_{\text{min}} = 10^{-12}$ in our implementation.

3 Bounded Constraints

One of the common variants of the unconstrained nonlinear least squares problem is to add bounded constraints

$$l_i \leq x_i \leq u_i, \tag{16}$$

with $-\infty \leq l_i < u_i \leq +\infty$ (infinity bound means not constraint). To incorporate such constraint, we define a mapping from the unconstrained space to constrained space

$$\mathbf{x}(\mathbf{y}) : \mathbb{R}^n \mapsto [\mathbf{l}, \mathbf{u}] = \prod_i [l_i, u_i], \tag{17}$$

and perform an unconstrained optimization on the function $f(\mathbf{x}(\mathbf{y}))$ with respect to $\mathbf{y}$.

3.1 Mapping function

We list the specific mapping function from the unconstrained space to constrained space. The general rule is that (1) the mapping is smooth and (2) the absolute value of derivative is smaller than 1 (but also close to 1 in most of the place). We also provide one possible inverse of the mapping $y_i^0(x_i)$, which is used for initialization.

- $l_i = -\infty$, $u_i = +\infty$, x_i unconstrained

$$x_i = y_i, \quad \frac{dx_i}{dy_i} = 1; \quad y_i^0 = x_i. \tag{18}$$

- $l_i = -\infty$, $x_i \leq u_i < +\infty$

$$x_i = u_i + 1 - \sqrt{y_i^2 + 1}, \quad \frac{dx_i}{dy_i} = -\frac{y_i}{\sqrt{y_i^2 + 1}}; \quad y_i^0 = \sqrt{(u_i + 1 - x_i)^2 - 1}. \tag{19}$$

- $-\infty < l_i \leq x_i$, $u_i = +\infty$

$$x_i = l_i - 1 + \sqrt{y_i^2 + 1}, \quad \frac{dx_i}{dy_i} = \frac{y_i}{\sqrt{y_i^2 + 1}}; \quad y_i^0 = \sqrt{(l_i - 1 - x_i)^2 - 1}. \tag{20}$$

- $-\infty < l_i \leq x_i \leq u_i < +\infty$

$$x_i = \frac{l_i + u_i}{2} + \frac{u_i - l_i}{2} \sin \frac{2y_i}{u_i - l_i}, \quad \frac{dx_i}{dy_i} = \cos \frac{2y_i}{u_i - l_i}; \quad y_i^0 = \frac{u_i - l_i}{2} \arcsin \frac{2x_i - (u_i + l_i)}{u_i - l_i}. \tag{21}$$

References

[1] K. Levenberg, “A method for the solution of certain problems in least squares,” *Quarterly of applied mathematics*, vol. 2, pp. 164–168, 1944.

[2] D. W. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *Journal of the Society for Industrial & Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[3] Wikipedia, “Plagiarism — Wikipedia, the free encyclopedia,” 2014, [Online; accessed 20-January-2014]. [Online]. Available: http://en.wikipedia.org/wiki/Levenberg-Marquardt_algorithm

[4] K. Madsen, H. B. Nielsen, and O. Tingleff, *Methods for non-linear least squares problems*, 2nd ed., 2004.