Computing Powell's Dogleg Steps for Solving Adaptive Networks Nonlinear Least-Squares Problems.

Eiji Mizutani *
Dept. of Industrial Engineering and Operations Research
University of California at Berkeley, Berkeley, CA 94720
Email: eiji@biosys2.me.berkeley.edu

Abstract

This paper describes **Powell's single dogleg trust-region algorithms** *for solving adaptive (neuro-fuzzy) networks nonlinear least squares problems. The dogleg algorithms facilitate efficient implementation of trust-region* **Levenberg-Marquardt algorithms***, and may thus be more suitable than Marquardt's original algorithm that has become popular for adaptive (neuro-fuzzy) networks learning [1, 2, 3].*

Since such a dogleg implementation is still not discussed well in adaptive (neuro-fuzzy) networks literature, we show how the dogleg step within the trust region generates a good compromise between the **Cauchy step** *and the* **Gauss-Newton step***, in conjunction with a* **local quadratic model** *of a given nonlinear least squares objective function.*

1 Adaptive Nonlinear Least-squares Techniques

In adaptive networks (AN) nonlinear least-squares problems, our objective is to optimize an n-dimensional parameter vector $\boldsymbol{\theta}$ that represents an AN model, by minimizing a **squared error measure** between desired outputs and the AN's outputs, which can be expressed by residual vectors $\mathbf{r}(\boldsymbol{\theta})$ as:

$$E(\boldsymbol{\theta}) \quad = \quad \tfrac{1}{2}\mathbf{r}^T(\boldsymbol{\theta})\mathbf{r}(\boldsymbol{\theta}), \tag{1}$$

where $E(.)$ is our objective function. The gradient vector and Hessian matrix are given by

$$\begin{aligned} \mathrm{g} = \mathrm{g}(\boldsymbol{\theta}) &\equiv \mathbf{J}^T\mathbf{r}, \\ \mathbf{H} = \mathbf{H}(\boldsymbol{\theta}) &\equiv \mathbf{J}^T\mathbf{J} + \mathbf{S}, \end{aligned}$$

where $\mathbf{J}$ is the **Jacobian matrix** of $\mathbf{r}$, and $\mathbf{S}$ denotes the second order term.

1.1 Levenberg-Marquardt Methods

The well-known Newton's method is given by

$$\begin{aligned} \boldsymbol{\theta}_{\text{next}} &= \boldsymbol{\theta}_{\text{now}} - \mathbf{H}^{-1}\mathrm{g} \\ &= \boldsymbol{\theta}_{\text{now}} - (\mathbf{J}^T\mathbf{J} + \mathbf{S})^{-1}\mathbf{J}^T\mathbf{r}. \end{aligned} \tag{2}$$

*This research is supported by SONY US Research Laboratories, San Jose CA, and SONY Corporation, Tokyo JAPAN. This paper appeared in the Proceeding of the 8th Int'l Fuzzy Systems Association World Congress, Taiwan, 1999 (IFSA'99).

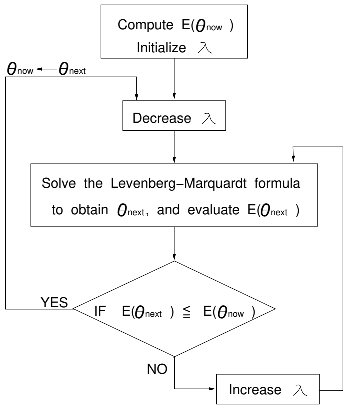


Figure 1: *The flow chart of Marquardt's original algorithm [4]. In the main loop, we need to solve the Levenberg-Marquardt formula defined in Equation (4) for each λ-update.*

The *Gauss-Newton* method is basically for so-called **small residual problems**, assuming that $\mathbf{S}$ is smaller than $\mathbf{J}^T\mathbf{J}$. This assumption amounts to neglecting $\mathbf{S}$:

$$\begin{aligned} \boldsymbol{\theta}_{\text{next}} &= \boldsymbol{\theta}_{\text{now}} - (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{r} \\ &= \boldsymbol{\theta}_{\text{now}} - (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{g}. \end{aligned} \tag{3}$$

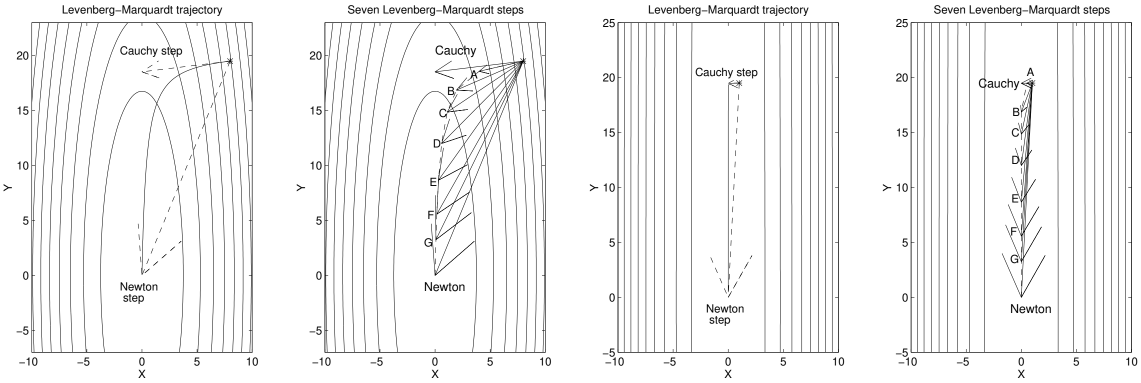
The Levenberg-Marquardt method is designed to handle well ill-conditioned Gauss-Newton model Hessian $\mathbf{J}^T\mathbf{J}$ by altering Equation (3) to

$$\begin{aligned} &\boldsymbol{\theta}_{\text{next}} = \boldsymbol{\theta}_{\text{now}} - (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{g}, \\ \text{or,}\quad &(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})\boldsymbol{\delta} = -\mathbf{g}, \end{aligned} \tag{4}$$

where λ is some nonnegative scalar value, and $\mathbf{I}$ is the identity matrix. We call Equation (4) the Levenberg-Marquardt formula. With a proper choice of λ, the Levenberg-Marquardt methods can be employed even when the Hessian matrix is rank-deficient; for instance, the Levenberg-Marquardt algorithm can be applied to training an "2-2-1" MLP (miltilayer perceptron) for solving the well-known XOR problem.

1.2 Marquardt's original algorithm

The Levenberg-Marquardt algorithms are not restricted to minimizing a sum-of-squares error; its con-



(a) (b) (c) (d)

Figure 2: *(a) The Levenberg-Marquardt trajectory, when the condition number equals 20 and angle α is 60.75 degree between the Newton step and the Cauchy step; (b) Seven Levenberg-Marquardt steps (steps A through G) associated with lambda* $= 1.0, 0.32, 0.16, 0.08, 0.04, 0.02, 0.01$, *respectively; (c) The Levenberg-Marquardt trajectory, when the condition number equals 2,000 and* $\alpha = 86.51$; *(d) Seven Levenberg-Marquardt steps (steps A through G) associated with lambda* $= 1.0, 0.0032, 0.0016, 0.0008, 0.0004, 0.0002, 0.0001$, *respectively.*

cept can be applied to improving the well-known Newton's method in a general setting [5, 6, 7]. Depending on the magnitude of λ, the Levenberg-Marquardt method transits smoothly between the two extremes: the Newton method ($\lambda \to 0$) and the steepest-descent method ($\lambda \to \infty$). Moreover, the λ controls the magnitude of the step $\|\boldsymbol{\theta}_{\text{next}} - \boldsymbol{\theta}_{\text{now}}\|$.

Figure 1 explains Marquardt's original algorithm [4, 8, 2], in which the Levenberg-Marquardt parameter λ is directly updated with an updating factor τ to implement a simple rule "increase λ when the error is increased whereas decrease λ when the error is decreased" in the following manner:

$$\begin{aligned} \lambda_{new} &= \tau \lambda_{old} \quad \text{for increase} \\ \lambda_{new} &= \tfrac{1}{\tau} \lambda_{old} \quad \text{for decrease}, \end{aligned} \tag{5}$$

where τ can be set equal to 2.0, for instance.

The Levenberg-Marquardt *steps* change along the so-called Levenberg-Marquardt *trajectory*, which is illustrated in Figures 2(a) and (c). Figures 2(b) and (d) illustrate seven representative Levenberg-Marquardt steps associated with seven different λ values. Table 1 indicates that numerical relation between various λ values and their corresponding step lengths may not be straightforward; hence, use of Equation (5) with a fixed τ value may require to solve the simultaneous equation (4) unexpectedly many times for each λ-update within one iteration.

This subsection concludes with Marquardt's finding [4] that the angle α between the Gauss-Newton direction and the steepest descent direction is often quite large, ranging from 80 degree to almost 90 degree [cf. Figure 2(c)] for a variety of problems. Accordingly, the length of the Gauss-Newton step tends to be much larger than that of the Cauchy step. In those situations, Powell's single dogleg is very useful because it well approximates the Levenberg-Marquardt trajectory even when the angle α is not too large (say, 60.75 degree) [compare Figures 2(a) and 3(b)].

Table 1: *Comparison of seven Levenberg-Marquardt steps A through G together with the Newton step and the Cauchy step in two cases where* $\alpha = 60.75$ *and* $\alpha = 86.51$. *Steps B through G were due to* $\tau = 2.0$.

	$\alpha = 60.75$	Fig. 2(b)	$\alpha = 86.51$	Fig. 2(d)
Step	λ	Step length	λ	Step length
A	1.0	4.10637	1.0	0.500094
B	0.32	6.60868	0.0032	2.81733
C	0.16	8.31382	0.0016	4.74903
D	0.08	10.5413	0.0008	7.56627
E	0.04	13.2865	0.0004	10.8793
F	0.02	15.985	0.0002	13.9644
G	0.01	18.0777	0.0001	16.2807
Newton	0	21.0772	0	19.5256
Cauchy	$--$	8.0592	$--$	1.00005

1.3 Trust-Region Approach

The **trust region** (or **restricted**) **step** moves the current point $\boldsymbol{\theta}_{\text{now}}$ to the intersection point ($\boldsymbol{\theta}_{\text{next}}$) between the Levenberg-Marquardt trajectory and the trust region boundary [see Figure 3(a)].

In the nonlinear least squares sense, the trust-region techniques are said to generate a **restricted Levenberg-Marquardt step** (or **trust region step**) by minimizing a **local quadratic model** $q(.)$ (see Figure 2) to the objective function over a trust-region circle centered at $\boldsymbol{\theta}_{\text{now}}$ with radius R_{now} in the two-dimensional subspace spanned by the Gauss-Newton direction and the steepest-descent direction. The value of R_{now}, **trust radius**, determines the size

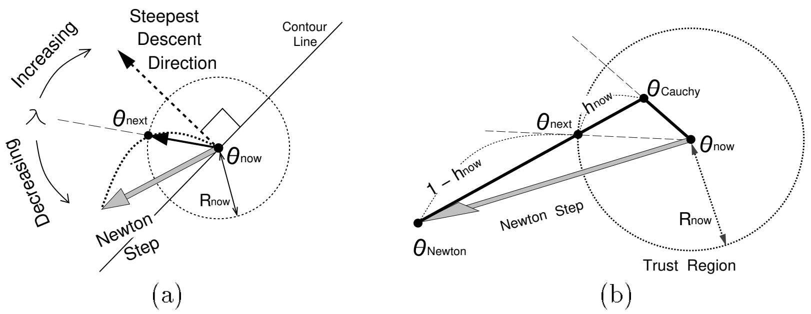


Figure 3: *(a) The trust-region step (in highlighted arrow) corresponds to a restricted Levenberg-Marquardt step determined by a trust region; (b) Powell's single dogleg trajectory, which approximates the Levenberg-Marquardt trajectory in a piecewise-linear fashion.*

of the trust region:

$$\{\boldsymbol{\delta} : ||\boldsymbol{\delta}|| \le R_{\text{now}}\} \quad (R_{\text{now}} > 0),$$

where $\boldsymbol{\delta} = \boldsymbol{\theta}_{\text{next}} - \boldsymbol{\theta}_{\text{now}}$. We now show that the trust region concept explains another interpretation of Levenberg-Marquardt parameter λ.

Consider the following **constrained quadratic minimization** problem: Find $\mathbf{s}$ such that

$$q(\mathbf{s}) = \min_{\boldsymbol{\delta}} q(\boldsymbol{\delta}) \text{ subject to } ||\mathbf{s}|| \le R_{\text{now}}, \quad (6)$$

where

$$q(\boldsymbol{\delta}) = c + \mathbf{g}_{\text{now}}^T \boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^T \mathbf{H}_{\text{now}} \boldsymbol{\delta}, \quad c = E(\boldsymbol{\theta}_{\text{now}}).$$

If $\mathbf{s}$ is a solution to this problem, then $\mathbf{s}$ satisfies

$$(\mathbf{H}_{\text{now}} + \lambda \mathbf{I})\mathbf{s} = -\mathbf{g}_{\text{now}}, \text{ with } \lambda(\boldsymbol{\delta}^T \boldsymbol{\delta} - R_{\text{now}}^2) = 0. \quad (7)$$

which is equivalent to the Levenberg-Marquardt formula with Gauss-Newton model Hessian. Hence the trust-region implementation of Levenberg-Marquardt algorithms is quite natural. Here, the constrained quadratic minimization is solved in conjunction with the **Lagrange multiplier** λ associated with the regional constraint $\boldsymbol{\delta}^T \boldsymbol{\delta} \le R_{now}^2$. Recall the second-order truncated Taylor series approximation to the objective function E:

$$\begin{aligned} E(\boldsymbol{\theta}_{\text{now}} + \boldsymbol{\delta}) &\approx E(\boldsymbol{\theta}_{\text{now}}) + \mathbf{g}_{\text{now}}^T \boldsymbol{\delta} + \tfrac{1}{2}\boldsymbol{\delta}^T \mathbf{H}_{\text{now}} \boldsymbol{\delta} \\ &\equiv q(\boldsymbol{\delta}), \end{aligned} \quad (8)$$

where $\boldsymbol{\delta} = \boldsymbol{\theta}_{\text{next}} - \boldsymbol{\theta}_{\text{now}}$. This $q(\boldsymbol{\delta})$ is the *local quadratic model* to the objective function E, and identical to $q(.)$ in Equation (6).

The effectiveness of the local quadratic model q can be measured by ν_{now}, which is defined as

$$\nu_{\text{now}} = \frac{\text{Actual reduction}}{\text{Predicted reduction}} = \frac{E(\theta_{\text{now}}) - E(\theta_{\text{now}} + \delta)}{E(\theta_{\text{now}}) - q(\delta)}. \quad (9)$$

The *trust radius* is expanded or contracted in a controlled fashion based upon how well the local quadratic model predicts behavior of the objective function, according to rules of thumb [9, 8]:

If ν_{now} is small, decrease R_{now}.
If ν_{now} is large, increase R_{now}.
Otherwise, keep the same.

This rule is quite problem-dependent. Notice that it might not be necessary to decrease R_{now} in some situations as long as the error decreases [10]. In the next section, we present how to approximate the trust-region Levenberg-Marquardt step.

1.4 Powell's Single Dogleg Steps

The **Powell's single dogleg step algorithm** ‡ offers an efficient implementation of the trust region approach because the *exact trust region step* is difficult or expensive to calculate [6].

Figure 3(b) shows how those dogleg steps approximate the trust region step over the two dimensional subspace spanned by the Cauchy step and the Newton step of the local quadratic model. The piecewise linear dogleg trajectory consists of the following:

- **Restricted Cauchy step**:

$$\mathbf{s}_{\text{RC}} \stackrel{\text{def}}{=} -R_{\text{now}} \frac{\mathbf{g}_{\text{now}}}{||\mathbf{g}_{\text{now}}||},$$

when the *trust radius* R_{now} is smaller than the Cauchy step $\mathbf{s}_{\text{Cauchy}}$:

$$\mathbf{s}_{\text{Cauchy}} \stackrel{\text{def}}{=} -\frac{\mathbf{g}_{\text{now}}^T \mathbf{g}_{\text{now}}}{\mathbf{g}_{\text{now}}^T \mathbf{H}_{\text{now}} \mathbf{g}_{\text{now}}} \mathbf{g}_{\text{now}} \quad (10)$$

- **Newton step**:

$$\mathbf{s}_{\text{Newton}} \stackrel{\text{def}}{=} -\mathbf{H}_{\text{now}}^{-1} \mathbf{g}_{\text{now}}, \quad (11)$$

‡Powell's original algorithm [9] is called the "single" dogleg in relation to Dennis-Mei's "double" dogleg modification [6]. Note that there is an error in Scales's description about Powell's algorithm on pages 120-121 in ref. [11].

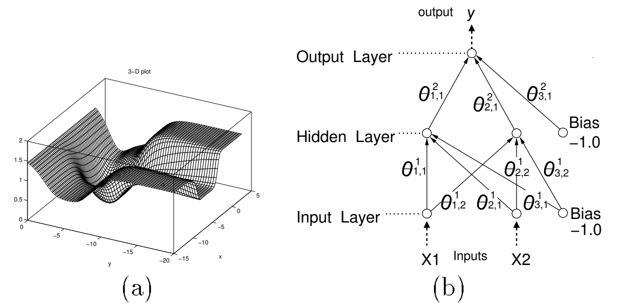


Figure 4: *(a) 3-D plot of the two weight [$\boldsymbol{\theta}^1_{1,1}$ (x-axis) and $\boldsymbol{\theta}^2_{1,1}$ (y-axis)] surface. (b) A typical "2-2-1" MLP architecture for solving the XOR problem.*

when the trust radius is larger than the Newton step. Here, $\mathbf{H}_{\text{now}}$ may be the Gauss-Newton model.

- **Dogleg step**:

$$\mathbf{s}_{\text{Dogleg}} \stackrel{\text{def}}{=} \mathbf{s}_{\text{Cauchy}} + h_{\text{now}}(\mathbf{s}_{\text{Newton}} - \mathbf{s}_{\text{Cauchy}}), \tag{12}$$

when the trust radius is in between the Cauchy step and the Newton step.

The parameter h_{now} [see Figure 3(b)] can be determined by the following straightforward calculation:

$$h_{\text{now}} = \frac{R^2_{\text{now}} - \mathbf{s}^T_{\text{Cauchy}}\mathbf{s}_{\text{Cauchy}}}{(\mathbf{s}_{\text{Newton}} - \mathbf{s}_{\text{Cauchy}})^T\mathbf{s}_{\text{Cauchy}} + \sqrt{p_{\text{now}}}}, \tag{13}$$

where

$$\begin{aligned} & p_{\text{now}} \\ & = (\mathbf{s}_{\text{Newton}} - \mathbf{s}_{\text{Cauchy}})^T(\mathbf{s}_{\text{Newton}} - \mathbf{s}_{\text{Cauchy}})R^2_{\text{now}} \\ & +(\mathbf{s}^T_{\text{Newton}}\mathbf{s}_{\text{Cauchy}})^2 \\ & -(\mathbf{s}^T_{\text{Newton}}\mathbf{s}_{\text{Newton}})(\mathbf{s}^T_{\text{Cauchy}}\mathbf{s}_{\text{Cauchy}}). \end{aligned}$$

In the primitive Marquardt's original algorithm shown in Figure 1 [4, 1, 2], when the next Levenberg-Marquardt step is not satisfactory (too large), we need to resolve the Levenberg-Marquardt formula (4) with a new increased λ. In other words, the Levenberg-Marquardt formula must be solved at each λ-update to determine $\boldsymbol{\theta}_{\text{next}}$. In contrast, once the Gauss-Newton step is computed for the current epoch, the dogleg algorithm, when the next dogleg step is not satisfactory, shrinks R_{now} and then resolves just Equation (12), which requires much less computational efforts than Equation (4). This efficiency cannot be found even in Fletcher's improved Marquardt algorithm [8].

2 Experiments

In this section, we demonstrate the aforementioned Powell's algorithm by solving the well-known XOR problem. Our objective function is a common *sum of squared error* measure, which forms an AN nonlinear least squares problem. Instead of ANFIS (or CANFIS)-type neuro-fuzzy model [7], here we employ a "2-2-1" MLP (multilayer perceptrons), as illustrated in Figure 4, where all neuron functions are sigmoidal-logistic functions, and there exist totally nine tunable parameters; only two of the weight parameters, $\boldsymbol{\theta}^1_{1,1}$ and $\boldsymbol{\theta}^2_{1,1}$, are adjusted while the rest of them are fixed *optimal*.

Figure 5 pictures the obtained algorithmic behaviors; (a) and (b) are obtained by Marquardt's original algorithm, which was quite sensitive to combinations of the initial λ and its updating factor τ.

Figure 5(c) shows a sequence of Powell's single dogleg steps: four restricted Cauchy steps, one dogleg step, and then two full Gauss-Newton steps to reach the minimum point. Notice that it is not necessary to solve simultaneous equation [Equation (11)] to shorten the dogleg step when it's too large [Figure 5(e)]. With a reduced trust radius, Equation (13) produces the next trial step [Figure 5(f)].

3 Discussion and Conclusion

We have addressed an implementation of Powell's single dogleg trust-region algorithm that approximates the restricted Levenberg-Marquardt steps in a piecewise-linear fashion.

Even through small-scale experiments, we have observed that choosing proper Levenberg-Marquardt parameters (λ) is quite state-of-the-art. More important, when the next step fails, Marquardt's original algorithm in Figure 1 requires to resolve simultaneous equations (Equation (4) with an increased λ), whereas the dogleg algorithm can avoid solving simultaneous equations. In addition, when taking the (restricted) Cauchy step, it is not necessary to solve the simultaneous equation, whereas the Marquardt algorithm always needs to do so even for taking any smaller step than the Cauchy step. In this context, the dogleg step algorithm is a much more promising implementaion of the Levenberg-Marquardt algorithm.

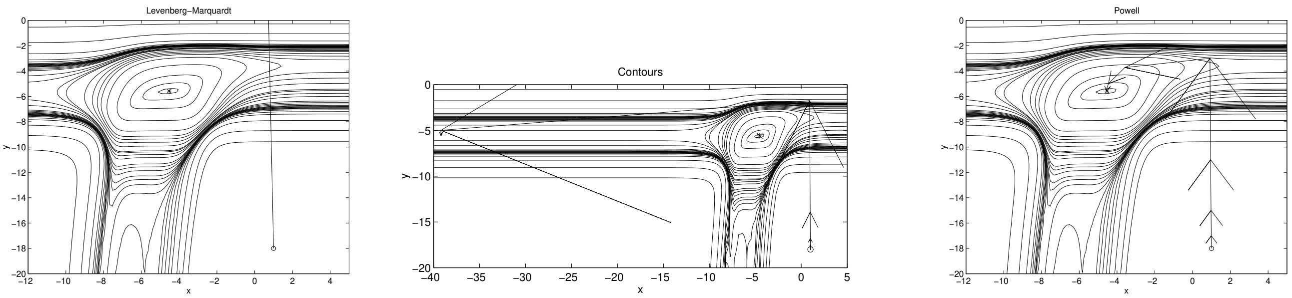
It should be also noticed that the dogleg-step method is quite robust in that it did solve the 14-dimensional parity problem (16,384 patterns) by an ordinary $14 \times 14 \times 1$ MLP (225 weights) at epoch 196. Other examples with algorithmic variations and more practical applications of dogleg algorithms are discussed in ref. [10, 12].

References

[1] H. Demuth and M. Beale. Neural Network Toolbox for Use with MATLAB. The MathWorks, Inc., Natick, Massachusetts, 1998. User's Guide (version 3.0).

[2] M.T. Hagan and M. Menhaj. Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994.

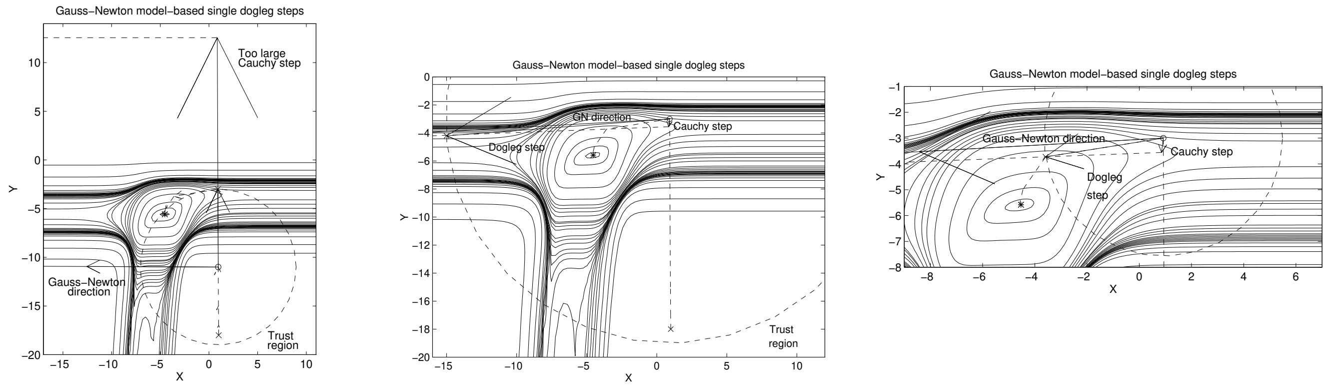
[3] J.-S. Roger Jang and E. Mizutani. Levenberg-Marquardt method for ANFIS learning. In *Proc. of the Int'l Conf. of the North Amer. Fuzzy Info. Proc. Soc. (NAFIPS'96)*, pages 87–91, California, 1996.



(a) Marquardt's original method $\lambda_{\text{init}} = \max\{\text{diag}(\mathbf{J}^T\mathbf{J})\}$ and $\tau = 5.0$.

(b) Marquardt's original method $\lambda_{\text{init}} = 0.01$ and $\tau = 2.0$. Eq. (4) was solved 26 times.

(c) Gauss-Newton-based Powell's single dogleg steps. Eq. (4) was solved 3 times.



(d) Restricted Cauchy step (Fourth step).

(e) Failed Powell's single dogleg step because it is too large.

(f) Accepted Powell's single dogleg step (Fifth step).

Figure 5: *Progresses of Marquardt's original algorithm (a) and (b), and Powell's dogleg algorithm (c). The rest of figures (d), (e), and (f) detail the fourth and fifth steps taken by Powell's dogleg implementation of Levenberg-Marquardt algorithm. The starting point is* $(1.0, -18.0)$ *and the minimum point is* $(-4.5299, -5.5821)$.

[4] Donald W. Marquardt. An algorithm for least squares estimation of nonlinear parameters. *Journal of the Society of Industrial and Applied Mathematics*, 2:431–441, 1963.

[5] S. M. Goldfeld, R. E. Quandt, and H. F. Trotter. Maximization by quadratic hill climbing. *Econometrica*, 34:541–551, 1966.

[6] Jr J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice Hall, New Jersey, 1983.

[7] J.-S. Roger Jang, C.-T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing: a computational approach to learning and machine intelligence.* Prentice Hall, Upper Saddle River, NJ, 1997.

[8] R. Fletcher. A modified Marquardt subroutine for nonlinear least squares. Technical Report AERE-R6799, Harwell Report, 1971.

[9] M. J. D. Powell. A new algorithm for unconstrained optimization. In Ritter Rosen, Mangasarian, editor, *Nonlinear Programming*, pages 31–65. Academic Press, New York, 1970.

[10] Eiji Mizutani. Powell's dogleg trust-region steps with the quasi-Newton augmented Hessian for neural nonlinear least-squares learning. In *Proceedings of the IEEE International Conference on Neural Networks*, Washington, D.C., July 1999.

[11] L. E. Scales. *Introduction to nonlinear optimization.* Macmillan, London, 1985.

[12] E. Mizutani, K. Nishio, N. Katoh, and M. Blasgen. Color device characterization of electronic cameras by solving adaptive networks nonlinear least squares problems. In *Proceedings of the 8th IEEE International Conference on Fuzzy Systems, vol. 2*, pages 858–862, Seoul, Korea, August 1999.