Report on Numerical Optimization Coding Assignment
Non-linear Least Squares Problem

Hongyi Zhang
Yuanpei College, Peking University
hongyizhang@pku.edu.cn

June 2, 2018

1 Overview

Given a smooth residual function $r : \mathbb{R}^n \to \mathbb{R}^m$, we use a series of methods to solve the non-linear least squares problem

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{2}\|r(x)\|^2,$$

including Gauss-Newton (GN) method, Levenberg-Marquardt (LM) method, the dogleg method and Dennis-Gay-Welsch (DGW) method. For comparison, the trust-region reflective (TRF) algorithm implemented in Python Scipy package is tried as well.

Denote by $J \in \mathbb{R}^{m\times n}$ the Jacobian of the residual r, then the objective has gradient $g = J^T r$ and Hessian $G = J^T J + S$ where $S = \sum_{i=1}^m r_i \nabla^2 r_i$. The matrix S measures the linearity of the problem — the larger its norm, the more significant the non-linearity. For small residual problems, the term S can be omitted without loss in performance. For large residual problems, however, this term is not negligible and some approximation is necessary to capture the curvature of the objective.

1.1 Non-linear least squares algorithms

Line search methods iterate by first determining a suitable direction d, and then finding a step length α along the direction with proper line search routines. Trust region methods iterate by solving a subproblem for a direction d, in which a quadratic approximation q to the objective f is optimized within a region of radius Δ, and then updating the radius Δ

by investigating the magnitude of $\gamma = \Delta f/\Delta q$ to determine whether a step of d should be taken. The following is a brief review on all the methods implemented.

- Gauss-Newton (GN) method: a line search method, solves the equation $J^T Jd = -J^T r$ (usually by minimizing $\frac{1}{2}\|Jd+r\|^2$, an equivalent linear least squares problem) for the direction d, and searches for an appropriate step length along the direction.

- Levenberg-Marquardt (LM) method: a trust region method, solves the equation $(J^T J + \lambda I)d = -J^T r$ for a direction d, where $\lambda > 0$ is dynamically adjusted. The adjustment routines correspond to the updates of the trust region radius, and two of the routines, proposed by Fletcher (LMF) and Nielsen (LMN), are implemented.

- Dogleg method: a trust region method, solves the quadratic subproblem by considering a dogleg path, which first runs a line segment from origin to the optimal steepest descent step (the Cauchy point), then switches to another line to reach the Gauss-Newton step. The intersection of the path and the trust region boundary is chosen as the update step, unless when the GN step d^{GN} lies within the region, in which case simply the GN step is chosen.

- Double dogleg method: a variant of dogleg method, slightly drags the dogleg step towards the GN step by replacing the path destination d^{GN} with a "shrinked" step $\eta \cdot d^{GN}$, where $0 < \eta < 1$ is carefully designed to ensure that the quadratic function q is monotonically decreasing along the whole path.

- Dennis-Gay-Welsch (DGW) method: a quasi-Newton-like update of approximation B_k of the matrix S by

  $$B_{k+1} = B_k + \frac{(\hat{y}_k - B_k s_k)y_k^T + y_k(\hat{y}_k - B_k s_k)^T}{y_k^T s_k} - \frac{(\hat{y}_k - B_k s_k)^T s_k}{(y_k^T s_k)^2} y_k y_k^T,$$

  where $\hat{y}_k = (J_{k+1} - J_k)^T r_{k+1}$. The method is effective for large residual problems, and can be combined with either line search or trust region algorithms.

1.2 Implementation details

For line search methods, we employ an inexact line search routine, and impose strong Wolfe conditions on the step length α. Parameter ρ for sufficient decrease is set to 10^{-4}, and the parameter σ for curvature condition is set to 0.5 for a moderate search. For trust region methods, we double the trust region radius if $\gamma = \Delta f/\Delta q$ is close to 1 ($\gamma > 0.75$), and cut it down to a quarter if γ is close to 0 ($\gamma < 0.25$). An update is accepted only if $\gamma > 0$.

Here are some remarks on the implementation of DGW method. The initial approximation B_0 of S can have an effect on the performance, and thus we choose $B_0 = 0.5I$ as a compromise between 0 and I. An important modification to the method is based on the fact that B_k does not necessarily vanish when x_k approaches a zero residual solution, and this problem can be solved by performing a scaling $B_k \leftarrow \tau_k B_k$ prior to its update, where

$$\tau_k = \min\left\{1, \frac{|\hat{y}_k^T s_k|}{|s_k^T B_k s_k|}\right\},$$

which guarantees fast convergence for small residual problems in our numerical experiments. DGW method, if combined with line search (DGW-LS), employs a line search routine with strong Wolfe conditions, and DGW combined with trust region (DGW-TR), otherwise, employs a dogleg routine to proceed in our implementation.

Since the problems we are going to deal with vary greatly in scale, we adopt a stopping criterion based on the relative error. To be specific, the iteration procedure is terminated when the condition $|f_k - f_{k-1}|/|f_{k-1}| < \epsilon$ is met, and we pick $\epsilon = 10^{-8}$.

2 Small residual problems

2.1 Exponential data fitting: Osborne functions

We now start with two data fitting problems of small residuals formulated by Osborne. The first one (given in the form of residual) is defined as

$$r_i(x) = y_i - [x_1 + x_2 \exp(-t_i x_4) + x_3 \exp(-t_i x_5)], \quad 1 \le i \le m, \quad x \in \mathbb{R}^5,$$

where $t_i = 10(i-1)$ and $\{(t_i, y_i)\}_{i=1}^m$ are a total of $m = 33$ data points supplied. The initial point $x_0 = (0.5, 1.5, -1, 0.01, 0.02)^T$, and the global optimum is approximately $f(x^*) \approx 2.7324 \times 10^{-5}$. Another function, in a slightly more complicated form, is defined as

$$\begin{aligned} r_i(x) = y_i - [&x_1 \exp(-t_i x_5) + x_2 \exp(-(t_i - x_9)^2 x_6) \\ &+ x_3 \exp(-(t_i - x_{10})^2 x_7) + x_4 \exp(-(t_i - x_{11})^2 x_8)], \quad x \in \mathbb{R}^{11}, \end{aligned}$$

where $t_i = (i-1)/10$ and $\{(t_i, y_i)\}_{i=1}^m$ are another $m = 65$ data points supplied. The initial point $x_0 = (1.3, 0.65, 0.65, 0.7, 0.6, 3, 5, 7, 2, 4.5, 5.5)^T$, and the global optimum is given by $f(x^*) \approx 2.0069 \times 10^{-2}$.

Table 1 summarizes the results of how different algorithms perform on these objectives. DL and DDL are short for dogleg and double dogleg respectively. Notations in the table are: the objective value $f(x^*)$, the magnitude of gradient $-\lg \|g^*\|$, the number of iterations

Table 1: Result comparison for the exponential data fitting problems

Method	Osborne 1				Osborne 2			
	$f(x^*)$	$-\lg\|g^*\|$	N_{iter}	N_{eval}	$f(x^*)$	$-\lg\|g^*\|$	N_{iter}	N_{eval}
GN	2.732e-05	8.7	8	21	2.007e-02	6.6	11	28
LMF	2.732e-05	6.7	29	30	2.007e-02	6.0	16	17
LMN	2.732e-05	6.2	25	26	2.007e-02	6.8	16	16
DL	2.732e-05	9.6	8	8	2.007e-02	6.7	14	12
DDL	2.732e-05	9.3	13	12	2.007e-02	6.4	13	12
TRF	2.732e-05	7.9	19	17	2.007e-02	6.6	13	11
DGW-LS	2.732e-05	7.2	26	27	2.007e-02	5.9	15	16
DGW-TR	2.732e-05	7.4	23	22	2.007e-02	6.9	18	18

N_{iter} and the number of function evaluations N_{eval}. We set the maximum number of iterations to be 1000, and $N_{iter} = \max$ if the method does not meet the stopping criterion after 1000 iterations.

As can be seen from Table 1, Osborne functions are fairly easy to solve from the given starting point. For Osborne function 1, GN method takes the fewest iterations to converge, but it also takes the most function evaluations per iteration. In contrast, trust region methods typically take only one function evaluation at each iteration. LM methods seem to be less effective than dogleg methods in this setting, but later experiments suggest that this is not always the case. For Osborne function 2, all methods achieve a similar performance, and there is no significant difference in their accuracy or number of iterations. GN and DGW-LS both fall into the category of line search methods, but the latter is apparently more economical if it takes relatively long to evaluate an objective value.

2.2 Chebyshev quadrature problem

Chebyshev polynomials are one of the most important tools in numerical analysis and approximation theory, defined by the recurrence relation

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n \geq 1,$$

with $T_0(x) = 1$ and $T_1(x) = x$. It can be expressed in a closed and more compact form: $T_n(x) = \cos(n \arccos x)$ for $-1 \leq x \leq 1$. In the appropriate space, the set of Chebyshev polynomials form an orthogonal basis, and thus a function $f(x)$ (in the same space) can be expanded as a series $f(x) = \sum_{n=0}^{\infty} c_n T_n(x)$.

Table 2: Result comparison for the Chebyshev quadrature problem

Method	$n = 8$				$n = 10$			
	$f(x^*)$	$-\lg \lVert g^* \rVert$	N_{iter}	N_{eval}	$f(x^*)$	$-\lg \lVert g^* \rVert$	N_{iter}	N_{eval}
GN	4.134e-03	1.2	14	139	1.366e-02	0.2	5	60
LMF	1.758e-03	6.0	24	22	3.252e-03	5.2	11	11
LMN	1.758e-03	5.5	20	20	3.252e-03	6.2	16	14
DL	1.758e-03	5.2	65	49	3.252e-03	4.3	85	69
DDL	1.758e-03	5.3	63	44	3.252e-03	4.5	87	72
TRF	1.758e-03	5.5	34	20	3.252e-03	6.0	21	11
DGW-LS	1.758e-03	7.8	12	13	3.252e-03	5.0	12	13
DGW-TR	1.758e-03	6.5	21	15	3.252e-03	5.3	15	13

Consider a numerical quadrature formula with equal weights

$$\int_0^1 f(x)\mathrm{d}x \approx \frac{1}{n}\sum_{j=1}^{n} f(x_j), \quad 0 \le x_j \le 1.$$

Inspired by the Chebyshev expansion, it is natural to define the residual

$$r_i(x) = \frac{1}{n}\sum_{j=1}^{n} T_i(x_j) - \int_0^1 T_i(x)\mathrm{d}x, \quad 1 \le i \le m,$$

where $T_i(x)$ are shifted to the interval $[0, 1]$, and minimize the ℓ^2 approximation error of the formula for the first m Chebyshev polynomials. The solution to this minimization problem leads to proper quadrature nodes of the formula. The initial point $x_0 = \frac{1}{n+1}(1, 2, \cdots, n)^T$. Assuming $m = n$, then the problem has a zero residual solution for $1 \le n \le 7$ and $n = 9$, while we have $f(x^*) \approx 1.7584 \times 10^{-3}$ for $n = 8$, and $f(x^*) \approx 3.2520 \times 10^{-3}$ for $n = 10$.

Table 2 compares the numerical results obtained by different algorithms for $n = 8$ and $n = 10$. GN method is the only one that fails to converge to the minimum, and its iteration procedure is forced to stop by an extremely small step length. Contrary to the observations from the last problem, LM methods obviously outperform dogleg methods a great deal in this setting, with fewer number of iterations and also higher precision of gradient. DGW method ranks top two in solving this problem, and its remarkable results, compared with GN method, demonstrate that a better approximation of Hessian is likely to be less susceptible to numerical issues and produce more accurate solutions. TRF method implemented in Scipy, however, does not seem as much competitive.

3 Large residual problems

3.1 Jennrich-Sampson function

We now turn to large residual problems. The first objective function we are going to tackle is proposed by Jennrich and Sampson, defined as

$$r_i(x) = 2 + 2i - \exp(ix_1) - \exp(ix_2), \quad 1 \le i \le m, \quad x \in \mathbb{R}^2,$$

where $m \ge n$. The initial point $x_0 = (0.3, 0.4)^T$, and the global optimum $f(x^*) \approx 62.181$ is obtained at $x^* \approx (0.2578, 0.2578)^T$ for $m = 10$.

Table 3: Result comparison for Jennrich-Sampson function

Method	$m=10$				$m=30$			
	$f(x^*)$	$-\lg\|g^*\|$	N_{iter}	N_{eval}	$f(x^*)$	$-\lg\|g^*\|$	N_{iter}	N_{eval}
GN	1.720e+03	-4.6	3	28	2.190e+04	-6.1	30	238
LMF	6.218e+01	1.7	20	15	4.289e+03	-0.1	12	13
LMN	6.218e+01	2.1	21	17	4.289e+03	-0.1	12	13
DL	6.218e+01	2.1	22	17	2.943e+03	-1.1	31	19
DDL	6.218e+01	1.7	24	16	2.943e+03	-0.7	27	15
TRF	6.218e+01	2.8	20	14	2.943e+03	0.8	30	17
DGW-LS	6.218e+01	2.8	9	10	2.943e+03	1.6	23	24
DGW-TR	6.218e+01	2.4	14	13	2.943e+03	2.1	30	27

Table 3 displays the numerical results corresponding to different methods, and we have tried two different settings $m = 10$ and $m = 30$. GN method once again fails to reach the global minimizer, which is caused by inappropriate step lengths and the consequential overflow problem. For the standard $m = 10$ setting, LM methods successfully give a proper solution, but they perform badly for the $m = 30$ setting where the methods stop at an objective value about a half more than the optimum. It is the only case we have found during our experiments where LM methods can fail.

In comparison, dogleg methods are able to achieve reasonable results in both $m = 10$ and $m = 30$ settings with a small number of iterations. Dogleg and double dogleg do not show significant differences. TRF and DGW methods have offered more satisfactory results for this large residual problem — they reach the optimum with equal or less iterations than dogleg methods, and their gradient precision at the optimal point is markedly higher than that of dogleg methods as well.

3.2 Meyer function

The next objective we are trying to optimize arises in the analysis of thermistor resistance as a function of the temperature. The problem, formulated by Meyer, is of the form

$$r_i(x) = y_i - x_1 \exp\left(\frac{x_2}{t_i + x_3}\right), \quad x \in \mathbb{R}^3,$$

where $t_i = 45 + 5i$ and $\{(t_i, y_i)\}_{i=1}^m$ are a total of $m = 16$ data points supplied. The initial point $x_0 = (0.02, 4000, 250)^T$, and the global optimum is approximately $f(x^*) \approx 43.973$.

Table 4: Result comparison for Meyer function

Method	$x_0 = (0.02, 4000, 250)$				$x_0 = (0.02, 2000, 250)$			
	$f(x^*)$	$-\lg \|g^*\|$	N_{iter}	N_{eval}	$f(x^*)$	$-\lg \|g^*\|$	N_{iter}	N_{eval}
GN	4.397e+01	1.2	9	23	8.437e+08	-7.1	124	1491
LMF	4.397e+01	-0.1	274	223	4.397e+01	-0.1	415	336
LMN	4.397e+01	-1.1	168	163	4.397e+01	1.2	261	253
DL	4.397e+01	3.3	48	44	4.397e+01	3.1	125	102
DDL	4.397e+01	0.1	59	44	4.397e+01	3.4	111	106
TRF	4.397e+01	3.0	186	148	4.397e+01	-0.5	396	313
DGW-LS	4.397e+01	1.7	186	187	4.397e+01	-0.8	267	268
DGW-TR	4.397e+01	0.0	208	183	4.397e+01	3.6	387	336

Table 4 provides a review on how different methods perform on this problem. Since the problem is fixed, we vary the starting point around. From the standard starting point $x_0 = (0.02, 4000, 250)^T$, GN method converges with the fewest iterations, followed by dogleg methods, and all other methods come the last. From another starting point $x_0 = (0.02, 2000, 250)^T$, however, GN method fails one more time, and the dogleg methods rank the top. It is in this problem that LMF and LMN methods start to show significant quantitative differences — LMN saves about half of the iterations to converge.

It is worth mentioning that, DGW method cannot achieve a proper solution shown in Table 4 without the following modification. As a matter of fact, this is the first case where DGW methods do not converge. The matrix S is indefinite in general, and thus we cannot not expect B_k, or $J^T J + B_k$, to be positive definite, which means the direction d_k produced by DGW method is not guaranteed to be a descent direction. Based on this observation, we add a modification to DGW method, that is, replace $J^T J + B_k$ with $J^T J$ (Gauss-Newton approximation) if the former does not generate a descent direction, measured by $g_k^T d_k$.

Table 5: Effectiveness of the modification to DGW method

	Method	$x_0 = (0.02, 6000, 250)$			$x_0 = (0.02, 2000, 250)$		
		$f(x^*)$	$-\lg \|g^*\|$	N_{iter}	$f(x^*)$	$-\lg \|g^*\|$	N_{iter}
Before	TRF	4.397e+01	0.7	331	4.397e+01	-0.5	396
	DGW-LS	9.492e+08	-12.3	6	2.883e+05	-6.5	5
	DGW-TR	4.790e+04	-6.0	41	9.352e+05	-6.3	max
After	TRF	4.397e+01	0.7	331	4.397e+01	-0.5	396
	DGW-LS	4.397e+01	1.6	231	4.397e+01	-0.8	267
	DGW-TR	4.397e+01	-1.5	318	4.397e+01	3.6	387

To demonstrate its effectiveness, we present in Table 5 the performance of DGW methods from two different starting points before and after the modification, and TRF method is also provided as a baseline condition. As one can see, DGW method without the modification runs into a point far from optimum in both two settings, either due to early stop or due to slow convergence. In comparison, when the modification is added, the performance of DGW method is dramatically improved and becomes comparable to that of TRF method. This encourages the use of Gauss-Newton steps during the iterations.

3.3 Brown-Dennis function

The final objective we are coping with is formulated by Brown and Dennis, defined as

$$r_i(x) = (x_1 + t_i x_2 - \exp(t_i))^2 + (x_3 + x_4 \sin t_i - \cos t_i)^2, \quad x \in \mathbb{R}^4,$$

where $t_i = i/5$ and $m \geq n$. The initial point is chosen as $x_0 = (25, 5, -5, -1)^T$, and the optimum is given by $f(x^*) \approx 42911.1$ for $m = 20$, which is unexpectedly large.

Table 6 presents the numerical solutions produced by different algorithms, and we run these algorithms under two settings $m = 20$ and $m = 40$. Two settings both have extremely large residuals, and the optimum $f(x^*) \approx 2.928 \times 10^{12}$ for $m = 40$. GN method converges with success after hundreds of iterations when $m = 20$, but it does not converge fast enough so that the number of iterations reaches the maximum when $m = 40$.

All methods other than GN are able to obtain a decent result, and dogleg methods perform much poorly than LM method, TRF method and DGW method. DGW method is the most powerful algorithm for this incredibly large residual problem, the smallest gradient obtained with the fewest iterations. It is noteworthy that, when m goes higher, the convergence behavior of LM, TRF and DGW almost remains unchanged, while the number

Table 6: Result comparison for Brown-Dennis function

Method	$m = 20$				$m = 40$			
	$f(x^*)$	$-\lg \|g^*\|$	N_{iter}	N_{eval}	$f(x^*)$	$-\lg \|g^*\|$	N_{iter}	N_{eval}
GN	4.291e+04	-1.5	387	3280	3.108e+12	-10.3	max	11943
LMF	4.291e+04	0.6	26	21	2.928e+12	-6.5	27	15
LMN	4.291e+04	0.1	25	21	2.928e+12	-6.0	26	19
DL	4.291e+04	-0.7	162	163	2.928e+12	-7.0	638	637
DDL	4.291e+04	-0.7	129	129	2.928e+12	-7.1	659	659
TRF	4.291e+04	0.2	26	17	2.928e+12	-6.6	20	15
DGW-LS	4.291e+04	0.5	13	14	2.928e+12	-5.1	13	14
DGW-TR	4.291e+04	0.2	15	15	2.928e+12	-4.7	19	19

of iterations needed for dogleg methods to converge grows rapidly. We may speculate that this numerical behavior of LM, TRF and DGW is independent of the dimension m of the residual, while dogleg methods can be greatly affected by this parameter.

References

[1] Wright, S. and Nocedal, J. (1999). Numerical optimization. Springer, 35(67-68), 7.
[2] Fletcher, R. (2013). Practical methods of optimization. John Wiley & Sons.
[3] More, J. J., Garbow, B. S. & Hillstrom, K. E. (1981). Testing unconstrained optimization software. ACM Transactions on Mathematical Software (TOMS), 7(1), 17-41.
[4] Nazareth, L. (1980). Some recent approaches to solving large residual nonlinear least squares problems. Siam Review, 22(1), 1-11.