Damping–undamping strategies for the Levenberg–Marquardt nonlinear least-squares method

Michael Lampton[a)]
Center for EUV Astrophysics, 2150 Kittredge Street, University of California, Berkeley, California 94720-5030

(Received 28 May 1996; accepted 25 October 1996)

The speed of the Levenberg–Marquardt (LM) nonlinear iterative least-squares method depends upon the choice of damping strategy when the fitted parameters are highly correlated. Additive damping with small damping increments and large damping decrements permits LM to efficiently solve difficult problems, including those that otherwise cause stagnation. © *1997 American Institute of Physics.* [S0894-1866(97)01801-4]

INTRODUCTION

Least-squares fitting is a common task in the sciences, and for nonlinear models the technique of choice is the iterative damped least-squares method of Levenberg[1] and Marquardt,[2] hereafter referred to as LM. The method has been widely presented, for example in Refs. 3–5, and is a component of several numerical mathematics packages, e.g., NL2SOL,[6] MINPACK,[7] and MathCad.[8]

In the LM method, an adjustable N-dimensional parameter vector controls an M vector of residuals, each component of which has the form (*computed–observed*)/σ, where σ normalizes each component to unit standard deviation. The discrepancy of the fit is measured by the sum of squares (SOS) of the components of the residuals' vector. The residuals depend nonlinearly on the parameters. From a given starting point, LM produces a sequence of parameter vectors, each step being an improvement in fit, i.e., a reduction in the SOS. The sequence terminates near a minimum of the SOS.

LM is applicable to a wide variety of nonlinear problems because it is adaptive. Each LM iteration determines its parameter step from the product of a damped inverse curvature matrix with a parameter space gradient. If the damping is set to a large value, the inverse curvature matrix is nearly diagonal and the LM step is near the steepest-descent direction. If the damping is small, the LM step approximates the exact quadratic step appropriate for a fully linear problem. LM is adaptive because it controls its own damping: it raises the damping if a step fails to reduce the SOS; otherwise it reduces the damping. In this way LM can navigate difficult model nonlinearities (although necessarily at low speed) yet it can also rapidly approach a best-fit minimum with nearly quadratic convergence speed.

The damping is implemented as follows. The LM parameter vector update depends on the damped curvature matrix α':

$$\Delta p = -[\alpha']^{-1} \times \beta, \tag{1}$$

where $\alpha'_{jj} = \alpha_{jj}(1+\lambda)$ for multiplicative damping or $\alpha'_{jj} = \alpha_{jj} + \lambda$ for additive damping, and $\alpha'_{jk} = \alpha_{jk}$ = for all $j \neq k$; $\alpha = J^T J$ = curvature matrix, Δp = change in parameter vector, $\beta = J^T R$ = gradient vector, $J = dR/dp$ = Jacobian matrix, R = (*computed–observed*)/σ = residual vector, I = identity matrix, and λ = damping term.

In LM, the damping term λ is adjusted at each step to assure a reduction in the SOS. For additive camping, a large value of λ makes the damped curvature matrix $(\alpha + \lambda I)$ diagonally dominant, and makes Δp lie in the direction of steepest descent. It also makes Δp shrink in magnitude. In this way LM can defensively navigate a region of parameter space in which the model is highly nonlinear. Additive damping improves the condition of the curvature matrix, i.e., it reduces the ratio of the largest eigenvalue to the smallest eigenvalue. In this way the additive term can stabilize the process through regions of parameter space where the Jacobian is rank deficient and the curvature matrix is, therefore, singular.

The alternative to additive damping is multiplicative damping. Multiplicative damping can be helpful in solving badly scaled problems, since it respects the order of magnitude of each parameter component. On the other hand, it offers no protection against a rank-deficient Jacobian.

The sequence of activities of a LM solver initially sets the damping constant λ at any reasonable positive value, and at each iteration computes the Jacobian, the gradient, the curvature matrix α, and (with λ) computes Δp and a new parameter vector p. If the SOS at this new p is reduced, LM reduces λ by some factor DROP, and begins a complete new iteration. Otherwise, LM abandons the new p in favor of the previous p, increases λ by some factor BOOST, and reevaluates Eq. (1) using the existing Jacobian. LM ceases iterating when reductions in the SOS become tiny.

There are four components of a damping strategy for a LM solver. First, there is the choice of the initial value of the damping variable. A large initial value, say, 1 or 10, will initially step LM in a more nearly steepest-descent direction, whereas a smaller value, say, 0.001 or 0.01, would begin in a more nearly quadratic-solution direction.

a)E-mail: mlampton@ssl.berkeley.edu

  0894-1866/97/11(1)/110/6/$10.00

A poor choice of initial damping is not serious, however, because LM will adjust the damping at each iteration. In the experiments reported here, the choices $\lambda_0=0.01$ and 1.0 are seen to give essentially the same performance. In this article I do not pursue this matter further, and the remaining examples have been run with $\lambda_0=0.01$ except where noted.

The remaining three components of a damping strategy for a LM solver are additive versus multiplicative damping, the choice of a value for DROP, and the choice of a value for BOOST. Popular damping strategies are additive, 0.1, 10 (Refs. 2 and 5) and multiplicative, 0.1, 10 (Refs. 3 and 4).

Most workers do not regard these alternatives as particularly critical, since on easy problems LM converges quickly for any choice of damping strategy. However, many least-squares problems exhibit a very high degree of correlation in the unknown parameters, combined with a smooth nonlinearity within the model. These can be regarded as navigational problems, whose solution demands a rather fine control over step direction without excessive damping having to be applied. For these problems, standard LM convergence can be very slow because the (defensive) steepest-descent direction lies nearly orthogonal to the direction in which progress is fastest. On these problems, the standard LM implementations incur a workload that increases rapidly and irregularly with problem difficulty. A better choice of damping strategy can enormously improve LM convergence speed on these problems, as will be shown.

I. ROSENBROCK'S PARABOLIC VALLEY

Rosenbrock's parabolic valley[9] is a minimization problem with adjustable difficulty. It has been widely employed in evaluating minimization algorithms, as, for example, in Ref. 10. In its usual implementation there are two adjustable parameters p_1 and p_2, combined in such a way as to produce a fourth-degree SOS function

$$(1-p_1)^2+D^2(p_2-p_1^2)^2$$

whose global minimum is located at $(p_1,p_2)=(+1,+1)$. For $D\gg 1$, the function increases rapidly at positions away from the parabola $p_2=p_1^2$, hence its name ''parabolic valley.'' This function can be identified as the χ^2 function comparing two data points to predictions from a particular nonlinear two parameter model. Generally, χ^2 functions are sums of squares of normalized residuals, each of which has form (*computed–observed*)/σ. Rosenbrock's function is the sum of squares of two residuals

$$r_1=D\times(p_2-p_1^2), \tag{2}$$

and

$$r_2=1-p_1, \tag{3}$$

where D is a coefficient that sets the difficulty of the problem. The starting point of the problem is traditionally taken at $(-1.2,1)$, and D is traditionally taken as 10. The global minimum lies at (1,1) but descent methods can reach that goal only via a number of moves along the curved valley. The total change in direction of the valley is 127°. The problem is well scaled since the required actions in the two parameters have the same order of magnitude. This problem has features that mimic many physics and astronomy fitting tasks, namely, a small number of parameters, a very high degree of coupling between these parameters, and an extended smooth nonlinearity that frustrates both steepest-descent and quadratic Newton methods.

II. THE TEST SUITE

To explore the consequences of various damping strategies, LM minimizations of the Rosenbrock valley were run with all combinations of BOOST=1.1, 1.2, 1.5, 2, 3, 5, 10, 20, 50, 100, and 200, with DROP=0.9, 0.8, 0.7, 0.5, 0.3, 0.2, 0.1, 0.05, 0.02, 0.01, and 0.005. Both multiplicative and additive damping were investigated. Difficulty coefficients of $D=1$, 10, 100, 1000, 10,000, and 100,000 were run.

The LM fitter was set up in a manner commonly used in fitting astronomical spectra, with single-sided numerical derivatives having a finite parameter step of 1×10^{-6} and double-precision math. On the Rosenbrock valley, this implementation costs three function calls per Jacobian, plus one call per trial SOS evaluation.

III. RESULTS

Each strategy grid point was evaluated by listing the number of function calls made during its run. Each fit terminated when the SOS was less than 1×10^{-12}, or 10,000 iterations had been taken, for which an ''xxxx'' is posted. The easy problems (i.e., those with difficulty coefficients of 1.0 or 10.0) showed only a mild variation in the number of function calls across the grid: all values of BOOST <50 and DROP <0.7 were equivalently good. Table I shows the number of function calls required by LM fitters using four strategies on the $D=10.0$ problem. Each block shows the computational cost of a LM fit for each combination of BOOST and DROP. The four blocks compare additive and multiplicative damping with initial damping set to 0.01 or 1.0. Table I reveals a mild preference for modest values of BOOST in the range of 1.2–3.0, and remarkably little dependence on DROP or for that matter anything else. Choice of initial damping is seen to be unimportant. This behavior is no doubt responsible for LM's reputation for being tolerant of alternative damping strategies.

With increasingly difficult problems the LM algorithm becomes slower, requiring more function calls to reach convergence. This behavior is understandable in view of the increased precision with which the narrow valley must be navigated. Remarkably, however, not all damping strategies are impacted equally by problem difficulty. In particular, additive damping strategies having BOOST in the range of 1.2–2.0 are impacted much less than other strategies. Table II shows four groups of Rosenbrock valley runs at difficulty $D=10{,}000$. Here, most LM strategies have been driven to very small step sizes (large damping coefficients) throughout most of each descent path, and need 10,000 or more function calls to converge. A cluster of strategies in the vicinity of add:0.1:1.5 remains nearly unaffected by the navigational difficulty of the problem. These strategies remain able to converge in fewer than 80 function calls. At difficulty 10,000 this amounts to a speed advantage of several hundred over traditional damping

Table I. Levenberg–Marquardt with Rosenbrock valley, $D=10$; number of function calls vs BOOST and DROP.

	BOOST=										
DROP	1.1	1.2	1.5	2.0	3.0	5.0	10.0	20.0	50.0	100.0	200.0
Additive damping, $\lambda_0=0.01$:											
0.900	174	156	163	173	165	209	228	328	272	337	402
0.800	140	116	113	113	136	129	138	183	157	187	222
0.700	121	102	99	99	101	120	134	133	112	132	152
0.500	110	84	75	75	71	102	105	104	103	118	174
0.300	105	79	68	67	63	89	92	85	89	141	156
0.200	108	78	65	57	84	104	100	131	123	175	179
0.100	125	86	63	60	62	103	158	147	98	229	172
0.050	133	94	69	58	81	82	102	146	95	184	187
0.020	150	99	72	73	62	110	113	108	326	172	215
0.010	168	104	81	77	104	90	175	150	100	1010	220
0.005	181	113	89	81	87	104	114	163	116	118	1592
Additive damping, $\lambda_0=1.00$:											
0.900	163	159	162	187	222	196	226	261	301	336	366
0.800	115	116	113	133	137	157	136	151	171	186	201
0.700	98	92	88	98	118	112	132	152	121	131	141
0.500	85	76	71	80	78	88	103	92	138	117	127
0.300	83	74	58	74	83	75	90	94	125	140	129
0.200	97	76	61	73	81	107	98	135	111	174	173
0.100	116	82	70	80	86	82	156	122	136	228	144
0.050	136	94	81	87	69	86	100	258	140	183	175
0.020	162	104	81	90	84	91	111	119	535	171	214
0.010	183	135	101	97	106	95	173	155	142	1009	195
0.005	203	133	96	104	96	99	112	134	120	117	1951
Multiplicative damping, $\lambda_0=0.01$:											
0.900	330	336	477	475	400	368	357	563	507	572	642
0.800	184	183	273	260	220	280	248	227	272	302	332
0.700	154	139	152	191	155	190	204	203	182	268	298
0.500	133	110	97	134	123	142	140	144	138	189	209
0.300	132	108	88	107	95	103	106	121	162	161	155
0.200	135	106	89	100	74	101	103	85	116	158	125
0.100	158	131	93	105	86	132	138	105	147	184	204
0.050	176	117	121	98	85	93	144	192	186	149	175
0.020	222	142	96	96	101	143	105	138	367	183	214
0.010	242	149	116	103	106	104	157	90	142	643	213
0.005	270	174	110	105	112	101	163	119	120	203	1255
Multiplicative damping, $\lambda_0=1.0$:											
0.900	453	450	455	458	462	588	572	642	541	792	606
0.800	341	325	278	296	351	308	353	332	372	301	437
0.700	299	260	252	234	220	208	202	222	247	201	287
0.500	174	146	139	163	164	161	154	153	173	224	203
0.300	199	152	116	114	121	112	121	114	145	118	175
0.200	209	136	96	95	96	116	107	111	104	125	135
0.100	170	121	111	96	103	110	130	138	185	172	203
0.050	233	176	122	100	120	110	117	209	110	131	252
0.020	231	151	127	102	105	134	96	130	264	199	219
0.010	253	174	125	122	111	109	148	147	221	696	235
0.005	237	169	117	136	92	123	161	138	106	159	1002

strategies. Table II shows that multiplicative damping does not have a corresponding zone of effectiveness for its damping constants.

Note that the Rosenbrock problem is well scaled so the scaling advantage of multiplicative damping is not manifested in these tests. To the extent that problems can be made well scaled by a change of variable, the findings presented above suggest that add:0.1:1.5 is superior to other LM damping strategies in the sense of resisting stagnation when presented with difficult navigational tasks.

It is important to test this claim on a wide variety of problems. Three generalizations of the Rosenbrock func-

Table II. Levenberg–Marquardt with Rosenbrock valley, D=10,000.0; number of function calls vs BOOST and DROP.

	BOOST=										
DROP	1.1	1.2	1.5	2.0	3.0	5.0	10.0	20.0	50.0	100.0	200.0
Additive damping, λ_0=0.01:											
0.900	174	162	163	173	3177	4834	6167	8475	7908	11518	12761
0.800	141	129	124	113	3992	6049	7646	10733	10165	14661	16009
0.700	129	109	99	99	4010	7154	8511	11687	11944	16187	20239
0.500	110	90	81	80	71	7776	9992	12691	12135	17769	21970
0.300	114	84	74	67	3764	5183	9697	14352	13603	18388	22947
0.200	116	83	72	62	4385	6519	8119	14722	13484	19102	23150
0.100	125	86	71	60	4372	9197	15495	15025	12656	29154	24003
0.050	139	94	74	63	5792	7617	11867	27495	9661	19498	24475
0.020	159	104	73	78	5142	10618	9540	16662	42369	15223	24661
0.010	168	113	82	77	6888	8724	18074	17377	14463	xxxx[a]	23767
0.005	185	113	89	81	5739	11639	13747	17626	16059	21203	51345
Additive damping, λ_0=1.0:											
0.900	169	165	199	218	3718	4436	6165	7834	9993	11517	13232
0.800	122	123	113	133	4581	6000	7644	9813	12662	14660	16835
0.700	106	99	100	2972	4682	6770	8509	10892	14127	16186	18799
0.500	96	85	84	1030	3486	7051	9990	11821	15204	17768	20707
0.300	104	83	72	86	4412	4427	9695	12503	16468	18387	21216
0.200	110	87	76	137	5066	5844	8117	12644	16313	19101	21367
0.100	131	96	101	147	5127	8340	15493	10103	16483	29153	22350
0.050	164	113	100	162	5798	6434	11865	18295	14803	19497	22927
0.020	195	133	124	5019	5830	9634	9538	14876	56018	15222	22204
0.010	225	164	119	5489	7895	7369	18072	14924	18131	xxxx[a]	17492
0.005	260	156	111	5959	5940	10923	13745	11851	19343	21202	31286
Multiplicative damping, λ_0=0.01:											
0.900	4242	4534	4646	5324	5830	6818	8389	10231	12166	14392	16181
0.800	4051	3563	5335	5813	6691	7954	9860	12107	15197	17479	19908
0.700	5181	5061	6023	6398	7438	8714	10905	13324	16674	19265	22009
0.500	6084	6456	6526	6265	8299	9779	12136	14683	18115	21118	23634
0.300	9088	7790	7714	8053	6179	10603	12776	15237	19028	21767	25050
0.200	11874	10056	8625	7565	9165	10587	13217	15707	19502	22420	25496
0.100	16098	11956	10120	8598	7115	11481	16465	16518	20051	35222	26179
0.050	16104	12881	10310	9626	11356	12443	14228	26276	20738	23726	26770
0.020	24638	16308	13729	12493	12289	13259	15534	17835	52738	24555	27647
0.010	25070	13881	13457	13667	11090	14256	19172	18541	21599	xxxx[a]	28359
0.005	30017	23221	11126	14888	14464	14148	16484	19285	22758	24680	xxxx[a]
Multiplicative damping, λ_0=1.0:											
0.900	4352	4670	4887	5472	6208	7033	8715	10164	12386	14612	16396
0.800	3856	3433	5529	6035	6822	8095	10118	12283	15302	17478	19887
0.700	4888	5321	6058	6452	7588	8826	10970	13261	16739	19264	22074
0.500	6134	6466	6481	6613	8351	9787	12028	14671	18119	21117	24177
0.300	9022	7798	7776	8092	6187	10732	12774	15068	19101	21766	24814
0.200	12351	10034	8961	7617	9219	12205	13215	15711	19490	22419	25458
0.100	16283	11878	10130	8608	7125	11505	16457	16453	20189	35353	26189
0.050	15837	12920	10337	9610	11290	12447	14226	30355	20783	23776	26758
0.020	24606	16235	13724	12497	12242	13219	15532	17909	56523	24554	27732
0.010	25075	14007	13462	13672	11196	14299	19177	18546	21604	xxxx[a]	28552
0.005	29908	22821	10952	14770	14450	14112	16431	19297	22992	24679	xxxx[a]

[a]xxxx: >10,000 iterations.

tion provide some additional tests. First, a statistical measure of convergence speed can be obtained by running a large number of trials from randomly chosen starting points. In this way a wide range of navigational difficulty is probed. Starting parameter pairs were randomly chosen from the square $-4<x<+4, -4<y<+4$. Results were tabulated separately for various damping strategies and are listed in Table III. On the easy $D=10$ problem, the recommended strategy add:0.1:1.5 is on the average more than twice as fast as any other strategy, and its worst case performance is comparably advantageous. On the difficult $D=10{,}000$ problem, add:0.1:1.5 is from 10 to 200 times

Table III. Convergence from random starting points, $-4<p_1<+4$ and $-4<p_2<+4$. Worst case and average number of function calls to converge.

Difficulty	NSTARTS	λ_0	Strategy	Worst	Average
10	1,000,000	0.01	add:0.1:10	206	87
10	1,000,000	0.01	add:0.1:1.5	115	29
10	1,000,000	1.0	add:0.1:10	204	58
10	1,000,000	1.0	add:0.1:1.5	114	41
10	1,000,000	0.01	mul:0.1:10	828	87
10	1,000,000	0.01	mul:0.1:1.5	176	68
10	1,000,000	1.0	mul:0.1:10	832	113
10	1,000,000	1.0	mul:0.1:1.5	197	86
10,000	1000	0.01	add:0.1:10	18141	1139
10,000	1000	0.01	add:0.1:1.5	9910	112
10,000	1000	1.0	add:0.1:10	18139	2826
10,000	1000	1.0	add:0.1:1.5	11960	579
10,000	1000	0.01	mul:0.1:10	60009	19953
10,000	1000	0.01	mul:0.1:1.5	16004	8273
10,000	1000	1.0	mul:0.1:10	60007	21195
10,000	1000	1.0	mul:0.1:1.5	15918	8245

faster on average than other strategies and its worst case performance is two to six times better.

Second, the Rosenbrock function can be generalized to nonlinearities that are higher than quadratic. As the exponent of p_1 in Eq. (2) is raised, the parabolic valley becomes a flat-bottomed trough bounded by increasingly abrupt walls at $x=-1$ and $x=+1$, and the starting point $(-1.2, +1.0)$ has an increasing SOS value. Consequently there is an increasingly wide range of feasible descent trajectories, and the navigational skills of a least-squares routine are not critically tested. With exponents as high as 100, both the traditional and the new LM damping strategies give rapid convergence, needing fewer than 100 function calls. However, if a more challenging (lower SOS) starting point is adopted, the new strategy shows marked superiority. For example, the starting point $(-1.0, +1.0)$ has no SOS dependence on exponent, and high exponents do not enlarge the descent trajectory space. At difficulty$=$10 and exponent $=$100, add:0.1:10 uses 2215 function calls, while add:0.1:1.5 needs only 447, a fivefold improvement.

Third, the Rosenbrock function can be generalized to higher-dimensional fitting problems. Following Ref. 5, with the typographical error corrected, let an N-dimensional parameter vector define an N-dimensional residual vector through the relations

$$r_{2i-1}=D\times(p_{2i}-p_{2i-1}^2), \tag{4}$$

and

$$r_{2i}=1-p_{2i-1}, \tag{5}$$

for $i=1,2,\ldots,N/2$ and N any even number. The starting point is the parameter vector $(-1.2, 1, -1.2, 1,...)$. On higher-dimensional problems, the new LM damping strategy is a great benefit. For example, in a 100-dimensional hypervalley with $D=100$, add:0.1:10 requires 7693 function calls to converge whereas add:0.1:1.5 needs only 960, an eightfold improvement in speed. And again the benefit increases with problem difficulty.

The success of the new damping strategy proposed here can be understood by imagining how a single LM step depends on its damping. If the damping were zero, the LM step would specify the parameter vector located at the minimum of the starting point's local quadratic surface. As the damping increases, LM sweeps out an arc in parameter space. On difficult problems, much of this arc lies in unfeasible (uphill) directions. As the damping approaches infinity, this arc approaches the starting point from the steepest-descent direction that is, of course, feasible (downhill) but has only a very small potential improvement. For the important class of difficult but smoothly differentiable problems considered here, zero damping is not a feasible descent step, and some small degree of damping has to be applied. Due to the smoothness of the problem, however, the local quadratic is a far better approximation to the direction of fastest progress than the steepest-descent direction. For this reason it is important to avoid applying excessive amounts of damping lest the local quadratic information be lost. Using a small factor for BOOST accomplishes this goal.

For all cases examined, the additive damping strategy 0.1:1.5 showed no disadvantages, and on all difficult problems showed large improvements in fitting speed. Exploration of this strategy's efficiency on other nonlinear least-squares problems will be useful.

ACKNOWLEDGMENTS

The author wishes to thank two anonymous referees for their useful comments, and also thanks Professor S. Bowyer for his support. The author acknowledges the support of NASA Contract Nos. NAS5-30180 and NAS5-29298 and Grant No. NAGW-3823. The Center for EUV Astrophysics is a division of UC Berkeley Space Sciences Laboratory.

REFERENCES

1. K. Levenberg, Q. Appl. Math. **2**, 164 (1944).
2. D. W. Marquardt, SIAM J. Appl. Math. **11**, 431 (1963).

3. P. R. Bevington, *Data Reduction and Error Analysis in the Physical Sciences* (McGraw-Hill, New York, 1st ed. 1969, 2nd ed. 1992).
4. W. H. Press *et al.*, *Numerical Recipes*: *The Art of Scientific Computing* (Cambridge University Press, New York, 1989).
5. J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations* (Prentice-Hall, Englewood Cliffs, NJ, 1983), App. B, p. 362.
6. J. E. Dennis, D. M. Gay, and R. E. Welsch, ACM Trans. Math. Software **7**, 348 (1981); **7**, 369 (1981).
7. J. J. Moré, B. S. Garbow, and K. E. Hillstrom, *Users Guide to MINPACK-I* (Argonne National Laboratory, Argonne, IL, 1980).
8. Mathcad Users Guide (Mathsoft Inc., Cambridge, MA, 1993), p. 429.
9. H. H. Rosenbrock, Comput. J. **3**, 175 (1960).
10. R. P. Brent, *Algorithms for Minimization without Derivatives* (Prentice-Hall, Englewood Cliffs, NJ, 1973).