CLOP: Confident Local Optimization for Noisy Black-Box Parameter Tuning

Rémi Coulom

Université de Lille, INRIA, CNRS, France

Abstract. Artificial intelligence in games often leads to the problem of parameter tuning. Some heuristics may have coefficients, and they should be tuned to maximize the win rate of the program. A possible approach is to build local quadratic models of the win rate as a function of program parameters. Many local regression algorithms have already been proposed for this task, but they are usually not sufficiently robust to deal automatically and efficiently with very noisy outputs and non-negative Hessians. The CLOP principle, which stands for Confident Local OPtimization, is a new approach to local regression that overcomes all these problems in a straightforward and efficient way. CLOP discards samples of which the estimated value is confidently inferior to the mean of all samples. Experiments demonstrate that, when the function to be optimized is smooth, this method outperforms all other tested algorithms.

1 Introduction

Authors of programs that play games, such as chess or Go, are faced with the problem of optimizing parameters. A game-playing program relies on heuristics for position evaluation, search-tree pruning, or time management. Most of these heuristics have parameters, and tuning them may improve the program's strength.

When optimizing parameters, a major difficulty is measuring the strength. The most usual approach is quite costly: it is based on the win rate against a reference opponent. In order to obtain an accurate measurement, it is necessary to play many games, which takes a large amount of computation time.

Since it is so costly, authors of game-playing programs sometimes try to avoid measuring win rates. Playing games may be replaced by testing over a database of one-move problems. It may also be replaced by machine-learning algorithms, such as temporal-difference methods [27,30].

Tuning parameters without measuring win rates may sometimes work, but it is dangerous; particularly, in the sense that it does not guarantee an optimal probability of winning. So far, it has not been proved that optimizing a criterion such as temporal difference also maximizes the strength.

Besides having no guarantee in terms of strength optimization, many learning algorithms also have a limited scope. For instance, temporal-difference methods can tune an evaluation function, but not selectivity or time management.

For a reliable and generic parameter-optimization method, it is necessary to measure the strength by the outcome of games played. The challenge is to come as close as possible to the optimal win rate with as few games as possible.

H.J. van den Herik and A. Plaat (Eds.): ACG 2011, LNCS 7168, pp. 146–157, 2012.

1.1 Problem Definition

More formally, the problem addressed in this paper is the estimation of an optimal value $\boldsymbol{x}^*$ of a vector of n continuous bounded parameters $\boldsymbol{x} \in [-1,1]^n$. Performance of a vector of parameters $\boldsymbol{x}$ is measured by its probability of success $f(\boldsymbol{x}) \in [0,1]$. The value of $f(\boldsymbol{x})$ is not known, but can be estimated by observing the outcome of independent Bernoulli trials that succeed with probability $f(\boldsymbol{x})$. These trials are observed in sequence. For each trial, parameters can be chosen in the $[-1,1]^n$ domain. After each trial, the optimization algorithm recommends a vector of parameters $\tilde{\boldsymbol{x}}$. The performance of the algorithm is measured by the simple regret $f(\boldsymbol{x}^*) - f(\tilde{\boldsymbol{x}})$. The objective is to find an algorithm that will make the simple regret go to zero as fast as possible.

The notion of "simple regret" is named in opposition to the more usual "online regret" of continuum-armed bandits algorithms [1,10]. When tuning a game-playing program, losing games does not matter. The objective is to optimize the final strength of the program, regardless of losses suffered while training.

The function f to be optimized will be assumed to have no local optima. The main difficulty that CLOP addresses is not getting out of tricky local optima, but dealing with noise.

1.2 Noisy Optimization

Because it has so many important applications in engineering, the problem of optimizing continuous parameters from noisy observations has been well studied. Many algorithms have been already proposed, even for the special case of binary response.

One of the oldest methods for optimization is stochastic gradient ascent. The principle of this approach is based on collecting samples of the function around the current parameters. These samples are used to estimate the gradient of the function. Parameters are then modified with a small step in the direction of the noisy gradient estimate. The most primitive form of this idea is the Kiefer-Wolfowitz algorithm [20]. The idea of Kiefer and Wolfowitz was improved in the multivariate case with the SPSA algorithm [28]. Several second-order refinements of SPSA were proposed [23,29].

A second kind of approach is population-based algorithms, such as evolution strategies or genetic algorithms [8,16,15]. These methods operate over a set of points in a parameter space, called the *population*. They iterate the following process: first, evaluate all elements of the population; then, discard those that perform badly; then, generate new elements similar to those that perform well.

Yet, a third approach is simulated annealing [21]. Simulated annealing is often used for combinatorial optimization with noiseless performance measurements. But it was generalized to the optimization of noisy functions [5], and to the optimization of continuous parameters [24].

1.3 Using Response-Surface Models

The algorithms presented so far often have to make a trade-off between fast inaccurate evaluation of many different parameter values, and slow accurate evaluation of few different parameter values. This trade-off is usually tuned by a meta-parameter that indicates the number of samples that are collected for each parameter value. In order to reduce noise, it may be necessary to replicate many independent measurements at the same point.

Many of these algorithms are also incremental, and discard old data. This may lead to a non-optimal use of all available information.

An approach that does not forget data and requires no trade-off between fast and accurate evaluation is the response-surface methodology [4]. The response-surface methodology fits a model to data, and performs optimization on the model. With response-surface models, it is not necessary to run more than one trial for each parameter value, which allows a dense coverage of the parameter space.

An important question when optimizing with response-surface models is the choice of a model. A first approach is to choose non-parametric models that are sufficiently general to fit any function [26,19,2,17,31,12,18]. A second approach is to use simpler models (linear or quadratic), over a shrinking domain [13,25,11,7]. The first approach is able to find the global optimum of a function with many local optima, whereas the second approach only performs local optimization.

Existing algorithms based on local regression all have some major limitations. The traditional response-surface methodology is not completely automated, and requires human judgment. Q2 [25] is rather similar to CLOP, except for its criterion for shrinking the region of interest (ROI): a sample is discarded only if there is a good confidence that the maximum of the quadratic regression lies within the ROI. This does not work if the Hessian is not definite negative, which happens frequently in practice (for instance, if one parameter turns out to have no influence on the strength). Noisy UOBYQA [11] uses a similar criterion, and it has the same defect. In addition, noisy UOBYQA takes the next sample at the estimated location of the optimum, which was found to be a rather inefficient sampling policy [25]. The trust-region method Elster and Neumaier [13] does not work in the very noisy case because it estimates the best parameters as those that obtained the best result so far. STRONG [7] has a proof of convergence, but (1) experiments show poor empirical performance compared to a straightforward stochastic gradient, and (2) the algorithm is extremely complicated.

The algorithm presented in this paper can deal in a simple, automatic and robust way with very noisy observations and a non-negative Hessian. The main difference with previous algorithms is in its criterion for deciding when to stop shrinking the regression area: the worst sample is discarded if its estimated value according to the regression is inferior with some level of confidence to the mean of all the remaining samples. Section 2 gives a detailed description of the algorithm and an intuitive analysis of its asymptotic rate of convergence. Section 3 presents empirical data that demonstrate its good performance compared to many alternative algorithms.

2 Algorithm

The general idea of the optimization algorithm is to perform regression over all the data, and then removing the worst samples (according to the regression) as long as sufficient samples are left. Samples are not removed one by one, because it would be too inefficient when the number of samples is high. Instead, a weight function, w, is computed with a formula that gives a weight close to zero to samples of which the estimated strength is confidently inferior to the average strength of samples. This process is iterated until convergence. Convergence is guaranteed by not allowing a weight to increase.

2.1 Detailed Algorithm Description

The details of the optimization algorithm for quadratic regression are given in Algorithm 1. The first parameter of the function is a positive number H that indicates how local the regression will be. $(\boldsymbol{x}_i, y_i)$ are pairs of past inputs and their outputs. y_i may be either 0 (loss) or 1 (win). QuadraticLogisticRegression is a function that performs weighted quadratic logistic regression, that is to say $f(\boldsymbol{x})$ is approximated at iteration k by $1/\left(1 + e^{-q_k(\boldsymbol{x})}\right)$, where q_k is a quadratic function. LogisticMean is logistic regression by a constant. Both QuadraticLogisticRegression and LogisticMean compute the maximum a posteriori with a Gaussian prior of variance 100. ConfidenceDeviation is the standard deviation of the posterior of LogisticMean.

The next sample is chosen at random (using Gibbs sampling) by using w as a probability density. The theory of optimal design offers many alternatives [6,14], but sampling according to w outperformed them in experiments. A problem with

Algorithm 1. Quadratic CLOP

procedure QuadraticCLOP($H, \boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_N, y_N$)
  $w_0 \leftarrow \lambda\boldsymbol{x}.1$ ▷ a function of $\boldsymbol{x}$ that returns 1
  $W_0 \leftarrow N$
  $k \leftarrow 0$

  repeat
    $w \leftarrow \lambda\boldsymbol{x}.\min_{i=0}^{k} w_i(\boldsymbol{x})$ ▷ weight function
    $k \leftarrow k + 1$
    $q_k \leftarrow$ QuadraticLogisticRegression($w, \boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_N, y_N$)
    $\mu_k \leftarrow$ LogisticMean($w, \boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_N, y_N$)
    $\sigma_k \leftarrow$ ConfidenceDeviation($w, \boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_N, y_N$)
    $w_k \leftarrow \lambda\boldsymbol{x}.e^{(q_k(\boldsymbol{x})-\mu_k)/(H\sigma_k)}$
    $W_k \leftarrow \Sigma_{i=1}^{N} \min\left(w(\boldsymbol{x}_i), w_k(\boldsymbol{x}_i)\right)$
  until $W_k > 0.99 \times W_{k-1}$

  $\boldsymbol{x}_{N+1} \leftarrow$ Random(w) ▷ next sample, distributed like w
  $\tilde{\boldsymbol{x}} \leftarrow \Sigma_{i=1}^{N+1} w(\boldsymbol{x}_i)\boldsymbol{x}_i / \Sigma_{i=1}^{N+1} w(\boldsymbol{x}_i)$ ▷ estimated optimal
end procedure

optimal design is that it assumes that the model perfectly fits the data, which is almost always wrong in practice. Even when lack of fit is taken into consideration [32], these algorithms take samples near the edge of the sampling domain. Samples near the edge are rapidly discarded when the domain is shrinking. It might be possible to do better, but sampling according to w is straightforward and works well.

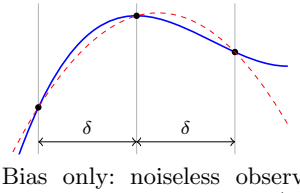
For quadratic regression, the maximum is estimated by the weighted average of samples. A different possibility would be to maximize the quadratic regression, but the estimation of the weighted average turned out to be more robust and perform better. In particular, it works even if the regression is not definite negative.

2.2 Choice of H and Asymptotic Rate of Convergence

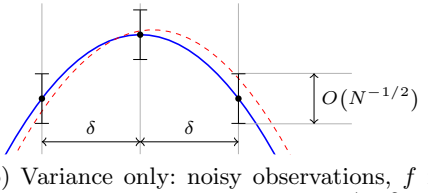
The most important meta-parameter of this algorithm is H. Besides H, other parameters are the priors of regressions, and the 0.99 constant that decides the end of the loop. But they have actually little influence on the performance. H tunes the bias-variance trade-off by making regression more or less local, and should be chosen carefully.

Figure 1 shows an analysis of the asymptotic bias-variance trade-off. In Algorithm 1, the "height" of the local regression is $H\sigma_k$. It should be proportional to the square of the "width" δ. If we assume that $\sigma_k = O\left(N^{-1/2}\right)$, this means that the optimal asymptotic rate of convergence is obtained with $H = O(N^{1/6})$.

An attempt at finding a more precise optimal value for H [3] shows that it depends on the magnitude of the cubic term: if the function to be optimized is perfectly quadratic, then $H = \infty$ is optimal. Otherwise, H should be smaller. Also, in the small-sample case, terms of degree four or more might not be negligible. So, it is difficult to find the optimal value of H. But, as will be shown in Section 3.1, choosing a constant value of $H = 3$ works quite well in practice, in a really wide range of situations.



(a) Bias only: noiseless observations, f is cubic. Regret: $O\left(\delta^4\right)$.



(b) Variance only: noisy observations, f is quadratic. Expected regret: $O\left(N^{-1}\delta^{-2}\right)$.

Fig. 1. The figures (a) and (b) illustrate an intuitive derivation of the optimal asymptotic bias-variance trade-off for local quadratic regression in dimension one [3]. Assuming N observations are made at x^*, $x^*-\delta$, and $x^*+\delta$, it is possible to calculate expected simple regret in both situations. The optimal trade-off is when they are the same, which gives $\delta = O\left(N^{-1/6}\right)$, and a simple regret of $O\left(N^{-2/3}\right)$. It was proved that $O\left(N^{-2/3}\right)$ is optimal when optimizing functions with bounded third-order derivatives [9], that is to say no algorithm can do better.

It is worth noting that the principle of CLOP is universal, and can be applied to any kind of regression, not only quadratic. So it is possible to use cubic regression or any other arbitrary polynomial regression instead. Such an approach may reach the optimal bound by Chen [9], that is to say $O\big(N^{-s/(s+1)}\big)$ simple regret for polynomial regression of degree s, provided f is sufficiently smooth. Figure 5 shows an experiment where cubic regression does outperform the best possible quadratic regression.

3 Experiments

The performance of CLOP was measured by artificial problems. Table 1, Fig. 2, and Fig. 3 show the artificial functions that were optimized. When an exponent is added to a problem name, it means that the dimension is multiplied by this exponent, and $r(\boldsymbol{x})$ is the average of $r(\boldsymbol{x})$ for each dimension (see Fig. 3 for the example of Log^2). The source code of the program that produced these results is available at `http://remi.coulom.free.fr/CLOP/`.

Table 1. Problem definitions. $f(\boldsymbol{x}) = 1/\big(1 + e^{-r(\boldsymbol{x})}\big)$. $\boldsymbol{x} \in [-1, 1]^n$.

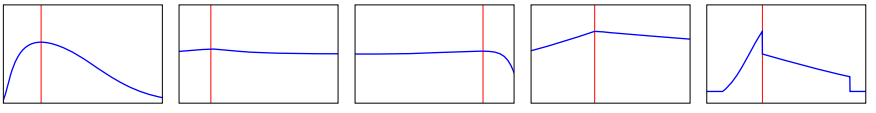
Log	$n = 1$	$r(x) = 2\log(4x + 4.1) - 4x - 3$
Flat	$n = 1$	$r(x) = 0.2/\big(1 + 6(x + 0.6)^2 + (x + 0.6)^3\big)$
Power	$n = 1$	$r(x) = 0.05(x + 1)^2 - ((x + 1)/2)^{20}$
Angle	$n = 1$	$r(x) = 1 + \begin{cases} \sqrt{2} - 2\sqrt{0.3 - x} & \text{if } x < -0.2, \\ \sqrt{2} - \sqrt{x + 2.2} & \text{otherwise.} \end{cases}$
Step	$n = 1$	$r(x) = \begin{cases} -2 & \text{if } x < -0.8, \\ -2 + 6(x + 0.8) & \text{if } -0.8 < x < -0.3, \\ -(x + 0.3)/1.1 & \text{if } -0.3 < x < 0.8, \\ -2 & \text{otherwise.} \end{cases}$
Rosenbrock	$n = 2$	$r(\boldsymbol{x}) = 1.0 - 0.1\big((1 - a)^2 + (b - a^2)^2\big)$, $a = 4x_1$, $b = 10x_2 + 4$
Correlated	$n = 2$	$r(\boldsymbol{x}) = 0.2(g(10(x_1 + x_2 + 0.1)) + g(x_1 - x_2 + 0.9)) + 0.2$, with $g(x) = -x^4 + x^3 - x^2$

3.1 Effect of Meta-parameter H

Figure 4 shows the effect of H when optimizing three different functions. As predicted in section 2.2, the optimal value of H depends on the quadraticity of the function to be optimized. For a moderately non-quadratic function such as Log, higher values of H perform better than for a strongly non-quadratic function such as Power. Results confirm the $O\big(N^{-2/3}\big)$ asymptotic simple regret with $H = O\big(N^{1/6}\big)$. Although it is not clear how to find the optimal value of H in practice, using a constant value of $H = 3$ seems to perform well in all cases.

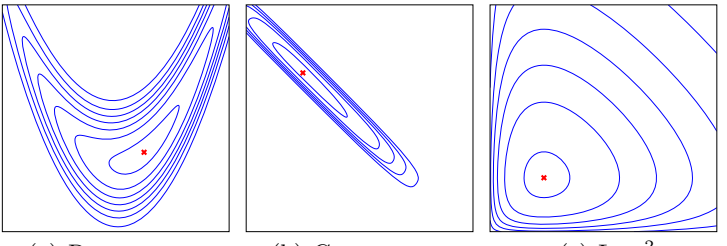
3.2 Comparison with Other Algorithms

Figures 5 and 6 compares CLOP to many algorithms. CLOP is best for smooth functions, works similarly for Angle, and does not work well for Step.



(a) Log (b) Flat (c) Power (d) Angle (e) Step

Fig. 2. Plots of one-dimensional problems. x^* is indicated by a vertical line.



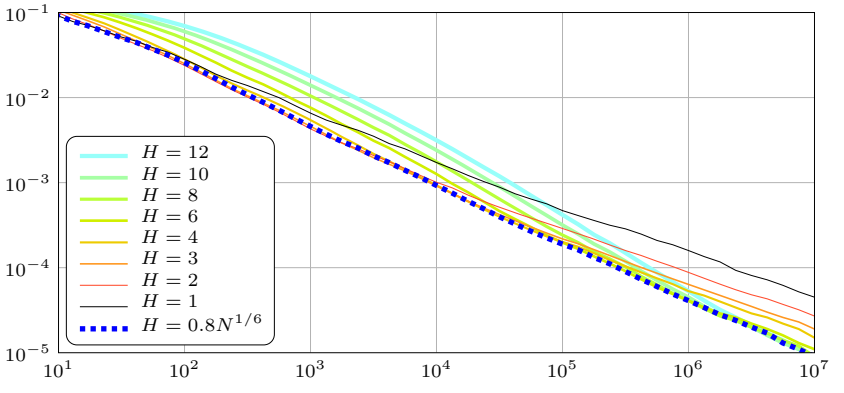
(a) Rosenbrock (b) Correlated (c) Log^2

Fig. 3. Plots of two-dimensional problems. Lines of constant probability are plotted every 0.1. $\boldsymbol{x}^*$ is indicated by a cross.

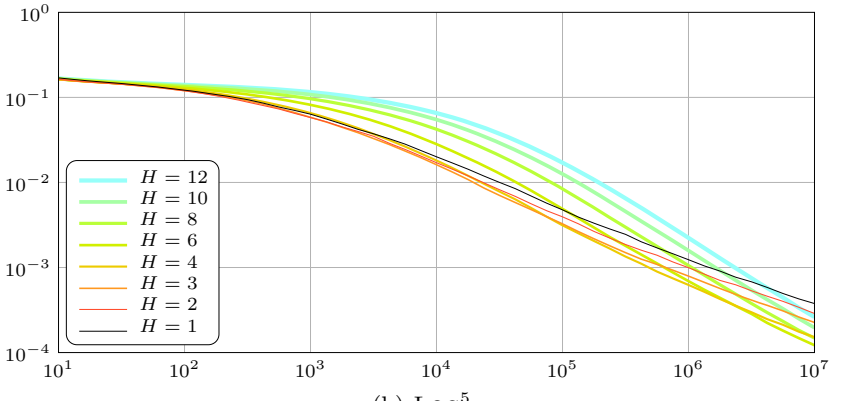
The population-based algorithms that were tested are variations of the Cross-Entropy method (CEM), using an independent Gaussian distribution. The basic version [8] was tested, as well as a version improved by dynamic parameter smoothing [16]. Initial population distribution was uniform random over $[-1, 1]^n$. Meta-parameters of the algorithms were: population = 100, elite = 10, initial batch size = 10, batch-size growth rate = 1.15, smoothing = 0.9, dynamic smoothing (improved version only) = 0.1. UH-CMA-ES [15] was tested too, but it was clearly not designed for that kind of problem, and it does not work well. Results were not plotted, because it sometimes fails to remain within the $[-1, 1]$ interval, even when started at $x = 0$ with a small variance.

A second algorithm that was tested is UCT [22], applied to a recursive binary partitioning of the parameter space. $\tilde{\boldsymbol{x}}$ is determined by choosing the child with the highest win rate.

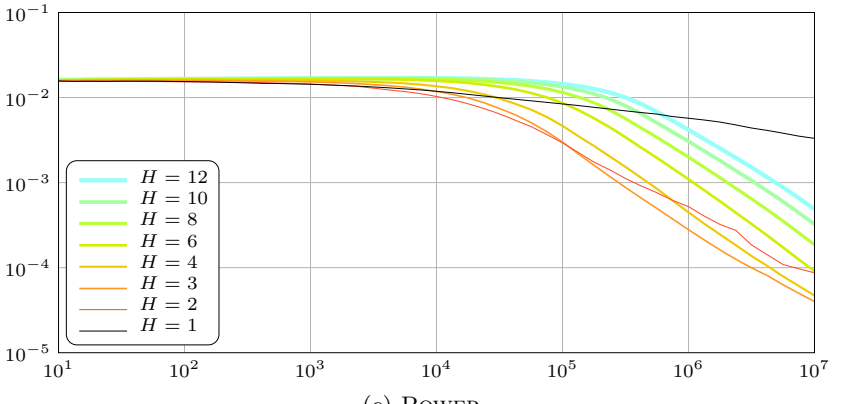
Finally, the SPSA algorithm [28] was tested. SPSA* is plain SPSA, with manually optimized meta-parameters ($a = 3$, $A = 0$, $\alpha = 1$, $c = 0.1$, $\gamma = 1/6$) starting at $\theta_0 = 0$. It performed quite well for Log, reaching the $O\left(N^{-2/3}\right)$ optimal asymptotic rate of convergence. But the performance of SPSA is rather sensitive to a good choice of meta-parameters, and these values do not work in practice for other problems. Many adaptive forms of SPSA have been proposed to automatically tune meta-parameters. RSPSA [23] is one such algorithm. Its meta-parameters were manually chosen to minimize simple regret at 10^5 samples (batch size = 1000, $\eta_+ = 1$, $\eta_- = 0.9$, $\delta_0 = 0.019$, $\delta_- = 0$, $\delta_+ = 0.02$, $\rho = 25$). These meta-parameters clearly overfit the problem, but they still do not outperform CLOP. Enhanced Adaptive SPSA (E2SPSA [29]) is another form of SPSA with better convergence guarantees. E2SPSA was not tested, but it can be expected that it would be at least twice slower than SPSA*, because half



(a) Log



(b) Log^5



(c) Power

Fig. 4. Effect of meta-parameter H. Simple regret (averaged over 1,000 replications) is plotted as a function of the number of samples.

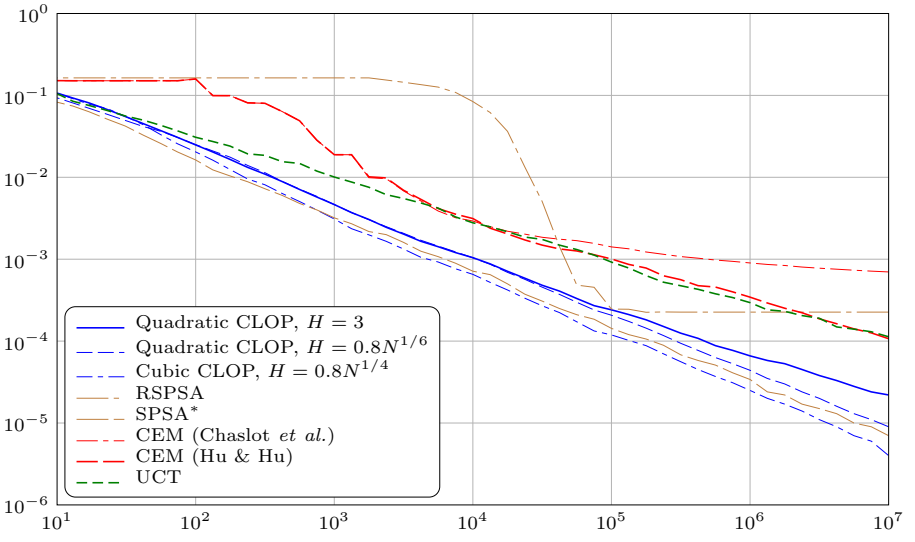


Fig. 5. Comparison of many algorithms applied to the Log problem

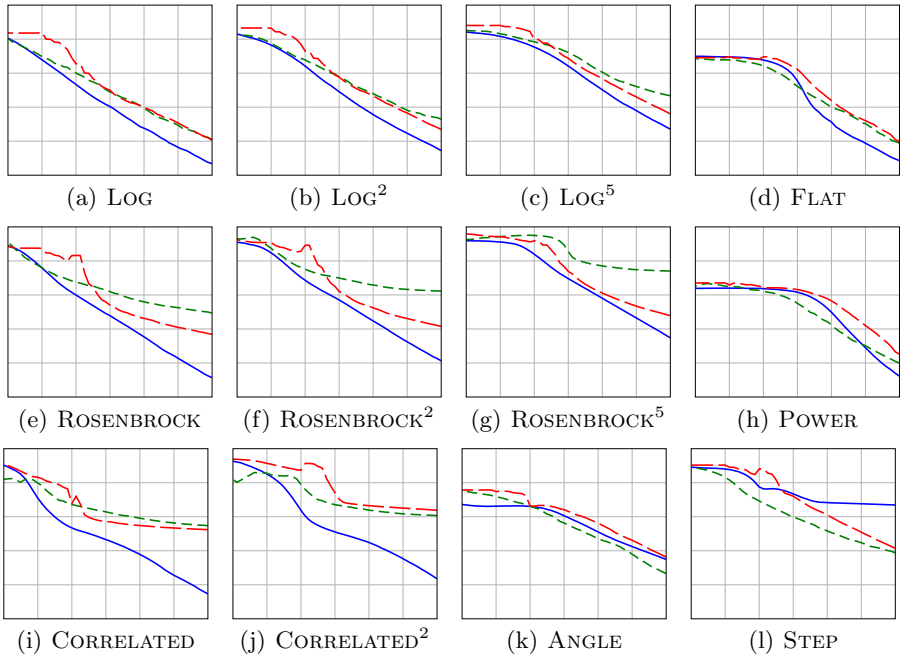


Fig. 6. Many problems. Scale: regret from 10^{-5} to 1, samples from 10 to 10^7. Legend: Quadratic CLOP ($H = 3$), UCT, CEM (Hu & Hu).

of the samples it collects are not used for gradient estimation. Also, E2SPSA only adapts a, but not c. So, it probably could not work well in problems such as CORRELATED. Still, the good performance of SPSA* is a clear sign that adaptive forms of SPSA could approach the performance of CLOP.

An aspect worth mentioning is the computational cost of these algorithms. Because logistic regression is a costly operation, CLOP tends to be slower. However, it can be made fast with a few tricks. First, w does not have to be re-computed at every sample: when the regression has been computed by N samples, $1 + N/10$ samples are collected without updating the regression. Second, an additional speed-up can be obtained by taking more than one sample at the same location. Experiments were run by averaging 1,000 runs of 10^7 samples, on a 24-core PC. For the LOG problem, CEM takes 1'20", CLOP 9'57", and UCT 21'19". UCT is slow because it was not particularly optimized, but it could certainly be sped-up considerably by using replications, too. Anyway, the cost of these algorithms is negligible compared to the cost of playing even super-fast games.

4 Conclusion

In summary, CLOP is a new approach to black-box optimization with local response-surface models. CLOP is completely automated, robust to very noisy outputs, and to non-negative Hessians. The algorithm is straightforward, and has only one meta parameter, H, that does not have a critical influence on performance. In practice, using a constant value of $H = 3$ works quite well in a wide range of function shapes and experiment sizes. Experiments demonstrate the excellent performance of CLOP for optimizing smooth functions.

In the future, CLOP could be applied to less noisy problems. It might even be possible to make it work efficiently for completely noiseless black-box optimization. This would probably require low-discrepancy algorithms (like in Q2 [25]), rather than random sampling. A second interesting question is the application of CLOP to other forms of regression. Quadratic regression is the most obvious and popular approach for local optimization, but, as was demonstrated in experiments, using more complex forms of regression, such as cubic regression, might produce better results. Finally, although the CLOP algorithm turned out to be extremely reliable in experiments, it would be good to have a mathematical proof of its convergence.

Acknowledgments. The work was supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. Moreover, it was supported in part by Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council and FEDER through the "CPER 2007–2013". This publication reflects only the author's views.

References

1. Agrawal, R.: The continuum-armed bandit problem. SIAM Journal on Control and Optimization 33(6), 1926–1951 (1995)
2. Anderson, B.S., Moore, A.W., Cohn, D.: A nonparametric approach to noisy and costly optimization. In: Langley, P. (ed.) Proceedings of the Seventeenth International Conference on Machine Learning, pp. 17–24. Morgan Kaufmann (2000)
3. Boesch, E.: Minimizing the mean of a random variable with one real parameter (2010)
4. Box, G.E.P., Wilson, K.B.: On the experimental attainment of optimum conditions (with discussion). Journal of the Royal Statistical Society 13(1), 1–45 (1951)
5. Branke, J., Meisel, S., Schmidt, C.: Simulated annealing in the presence of noise. Journal of Heuristics 14, 627–654 (2008)
6. Chaloner, K.: Bayesian design for estimating the turning point of a quadratic regression. Communications in Statistics—Theory and Methods 18(4), 1385–1400 (1989)
7. Chang, K.H., Hong, L.J., Wan, H.: Stochastic trust region gradient-free method (STRONG)—a new response-surface-based algorithm in simulation optimization. In: Henderson, S.G., Biller, B., Hsieh, M.H., Shortle, J., Tew, J.D., Barton, R.R. (eds.) Proceedings of the 2007 Winter Simulation Conference, pp. 346–354 (2007)
8. Chaslot, G.M.J.B., Winands, M.H.M., Szita, I., van den Herik, H.J.: Cross-entropy for Monte-Carlo tree search. ICGA Journal 31(3), 145–156 (2008)
9. Chen, H.: Lower rate of convergence for locating a maximum of a function. The Annals of Statistics 16(3), 1330–1334 (1988)
10. Coquelin, P.A., Munos, R.: Bandit algorithms for tree search. In: Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (2007)
11. Deng, G., Ferris, M.C.: Adaptation of the UOBYQA algorithm for noisy functions. In: Perrone, L.F., Wieland, F.P., Liu, J., Lawson, B.G., Nicol, D.M., Fujimoto, R.M. (eds.) Proceedings of the 2006 Winter Simulation Conference, pp. 312–319 (2006)
12. Deng, G., Ferris, M.C.: Extension of the DIRECT optimization algorithm for noisy functions. In: Henderson, S.G., Biller, B., Hsieh, M.H., Shortle, J., Tew, J.D., Barton, R.R. (eds.) Proceedings of the 2007 Winter Simulation Conference, pp. 497–504 (2007)
13. Elster, C., Neumaier, A.: A method of trust region type for minimizing noisy functions. Computing 58(1), 31–46 (1997)
14. Fackle Fornius, E.: Optimal Design of Experiments for the Quadratic Logistic Model. Ph.D. thesis, Department of Statistics, Stockholm University (2008)
15. Hansen, N., Niederberger, A.S.P., Guzzella, L., Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Transactions on Evolutionary Computation 13(1), 180–197 (2009)
16. Hu, J., Hu, P.: On the performance of the cross-entropy method. In: Rossetti, M.D., Hill, R.R., Johansson, B., Dunkin, A., Ingalls, R.G. (eds.) Proceedings of the 2009 Winter Simulation Conference, pp. 459–468 (2009)
17. Huang, D., Allen, T.T., Notz, W.I., Zeng, N.: Global optimization of stochastic black-box systems via sequential kriging meta-models. Journal of Global Optimization 34(3), 441–466 (2006)

18. Hutter, F., Bartz-Beielstein, T., Hoos, H., Leyton-Brown, K., Murphy, K.: Sequential model-based parameter optimisation: an experimental investigation of automated and interactive approaches. In: Bartz-Beielstein, T., Chiarandini, M., Paquete, L., Preuß, M. (eds.) Empirical Methods for the Analysis of Optimization Algorithms, ch.15, pp. 361–411. Springer (2010)
19. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13, 455–492 (1998)
20. Kiefer, J., Wolfowitz, J.: Stochastic estimation of the maximum of a regression function. Annals of Mathematical Statistics 23(3), 462–466 (1952)
21. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983)
22. Kocsis, L., Szepesvári, C.: Bandit Based Monte-Carlo Planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 282–293. Springer, Heidelberg (2006)
23. Kocsis, L., Szepesvári, C.: Universal parameter optimisation in games based on SPSA. Machine Learning 63(3), 249–286 (2006)
24. Locatelli, M.: Simulated annealing algorithms for continuous global optimization. In: Handbook of Global Optimization II, pp. 179–230. Kluwer Academic Publishers (2002)
25. Moore, A.W., Schneider, J.G., Boyan, J.A., Lee, M.S.: Q2: Memory-based active learning for optimizing noisy continuous functions. In: Shavlik, J. (ed.) Proceedings of the Fifteenth International Conference of Machine Learning, pp. 386–394. Morgan Kaufmann (1998)
26. Salganicoff, M., Ungar, L.H.: Active exploration and learning in real-valued spaces using multi-armed bandit allocation indices. In: Prieditis, A., Russell, S.J. (eds.) Proceedings of the Twelfth International Conference on Machine Learning, pp. 480–487. Morgan Kaufmann (1995)
27. Samuel, A.L.: Some studies in machine learning using the game of checkers. IBM Journal 3(3), 210–229 (1959)
28. Spall, J.C.: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Transactions on Automatic Control 37, 332–341 (1992)
29. Spall, J.C.: Feedback and weighting mechanisms for improving Jacobian estimates in the adaptive simultaneous perturbation algorithm. IEEE Transactions on Automatic Control 54(6), 1216–1229 (2009)
30. Tesauro, G.: Temporal difference learning and TD-Gammon. Communications of the ACM 38(3), 58–68 (1995)
31. Villemonteix, J., Vazquez, E., Walter, E.: An informational approach to the global optimization of expensive-to-evaluate functions. Journal of Global Optimization (September 2008)
32. Wiens, D.P.: Robustness of design for the testing of lack of fit and for estimation in binary response models. Computational Statistics & Data Analysis 54(12), 3371–3378 (2010)