Mixed-Effects Modeling of Optimisation Algorithm Performance

Matteo Gagliolo[1,2], Catherine Legrand[3], and Mauro Birattari[1]

[1] IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
[2] Faculty of Informatics, University of Lugano, Lugano, Switzerland
[3] Institut de Statistique, Université Catholique de Louvain, Louvain-la-Neuve, Belgium
{mgagliolo,mbiro}@iridia.ulb.ac.be, catherine.legrand@uclouvain.be

Abstract. The learning curves of optimisation algorithms, plotting the evolution of the objective vs. runtime spent. can be viewed as a sample of *longitudinal* data. In this paper we describe *mixed-effects* modeling, a standard technique in longitudinal data analysis, and give an example of its application to algorithm performance modeling.

1 Introduction

Models of algorithm performance can be useful for analysis purposes, but also for automating algorithm selection, or parameter tuning. The correct analysis technique depends on the kind of problem to be solved. For algorithms solving *optimisation* problems, in which each solution is characterized by a measure of its quality, the most general performance model is a bivariate distribution, relating runtime to solution quality. In order to gather performance data for a given algorithm, one can solve a benchmark of instances, storing a time, quality pair each time the best solution is improved. The resulting sample will be a set of observations of solution quality vs. time, grouped based on the individual runs of the algorithm. In statistical terminology, this is an example of *longitudinal data* [1], i.e. measurements of the same quantity repeated over time on each of a set of *subjects*. In the following, we describe parametric *mixed-effects* models, a standard technique in longitudinal data analysis (Sec. 2), and present preliminary experiments on performance data from a TSP solver (Sec. 3). Section 4 gives references for further reading, while Section 5 concludes the paper with a perspective on ongoing research.

2 Longitudinal Data Analysis

Consider a sample of measurements of a scalar y: for the i-th of a set of M *subjects*, or *individuals*, n_i values y_{ij} are collected at distinct times t_{ij}, with $j = 1, \ldots, n_i$. A pair (y, t) is termed an *observation*. The object of modeling is the distribution of y given t.

T. Stützle, M. Birattari, and H.H. Hoos (Eds.): SLS 2009, LNCS 5752, pp. 150–154, 2009.

In our case, y is the objective being optimized, and each y_{ij} records the value of the *best* solution found by a randomized algorithm within a time t_{ij}; the M subjects correspond to M distinct runs of the algorithm, which may be solving different problem instances, or differ only in the random seed used.

The main issue posed by longitudinal data is *within-subject* correlation: measurements taken on a same subject cannot be considered independent. To address this, in parametric *mixed-effects* models [1] the evolution of y for each subject is modeled by a separate curve, whose parameters are a random perturbation of those of a baseline curve, describing the overall behavior of the set of subjects. In a *nonlinear* mixed-effects model (NLME) [2],

$$\begin{aligned} y_{ij} &= f(\boldsymbol{\alpha}_i, t_{ij}) + \epsilon_{ij} \\ \boldsymbol{\alpha}_i &= \boldsymbol{\beta} + \mathbf{b}_i \\ \mathbf{b}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}),\ \boldsymbol{\epsilon}_i &\sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{\Lambda}_i), \end{aligned} \tag{1}$$

the evolution of y is modeled by a parametric function $f(\boldsymbol{\alpha}, t)$, whose parameter vector $\boldsymbol{\alpha}$ is the sum of a subject-specific vector of *random effects* $\mathbf{b}_i$ and a common vector of *fixed effects* $\boldsymbol{\beta}$. The M vectors $\mathbf{b}_i$ are assumed to be generated independently, once for each subject i, from the same zero mean Gaussian distribution, whose covariance $\boldsymbol{\Sigma}$ is estimated along with the M $\mathbf{b}_i$ and $\boldsymbol{\beta}$. The arbitrary subject-specific covariance structure $\boldsymbol{\Lambda}_i$ can capture within-subject correlations that are not modeled by the random effects: in the simplest case, the ϵ_{ij} are homoscedastic, (i.e., their variance does not change with j), independent, and identically distributed, so $\boldsymbol{\Lambda}_i = \sigma^2 \mathbf{I}_{n_i}$ for all subjects i. Both stationary and time-varying covariates, as well as nested factors, can be easily included [3].

3 Experiments

One important issue in optimisation is that, while bounds may be easy to compute, the actual value of the global optimum is usually not available beforehand. Consider an algorithm solving a problem instance: in general, there is no way of telling whether the current best solution will be further improved, or the algorithm has already found the global optimum. Our interest in nonlinear models is motivated by the hope that they may allow to extrapolate the performance on a new instance, based on previous runs on similar instances. In order to test this idea, we collected performance data on a benchmark of small instances of the traveling salesman problem (TSP) [4], for which the global optima were available. The benchmark was composed of four groups of 100 symmetric Euclidean instances each, randomly generated, with 200, 300, 400, and 500 cities respectively. The solver used was ILS-FDD [5], an iterated version of the local search algorithm `3-opt`, with fitness-distance diversification. Each instance was solved 25 times, with different random seeds. In the TSP, the objective is represented by the cost l of a path visiting all cities once, which has to be minimized. A lower bound l_b on l, and the global optimum l_o, were evaluated using Concorde. The value of the objective was collected, along with the runtime, each time an

improvement was found, resulting in a sequence of (l, t) values for each instance and each random seed. In order to perform the modeling across different instances, the objective was scaled relative to the lower bound as $y = (l - l_b)/l_b$, and the corresponding sample of (y, t) values was used as longitudinal data. Modeling was performed using `nlme` [3], a software package for linear and nonlinear mixed effects, in its version for the R language [6]. As the data presented an exponential decay towards the optimum, which is typical of optimisation algorithms in general, we fit a model of the form $y = a + be^{-ct}$, implemented in `nlme` by the function `SSasymp`. A fundamental issue of the use of such model is that the NLME algorithm is based on a zero mean Gaussian noise model (1). In our case, this assumption is violated as the algorithm converges towards the optimum, as it cannot further decrease. As a solution, we used an heteroscedastic variance structure (`varPower`), expressing the variance as a function of time, $\mathrm{Var}(\epsilon_{ij}) = \sigma^2 t_{ij}^{2\rho}$, with one parameter ρ being estimated, in order to be able to model a decreasing variance. For the correlation structure, we used the function `corCAR1` (continuous time $AR(1)$), $\mathrm{Cor}(\epsilon_{ij}, \epsilon_{ik}) = \phi^{|t_{ij} - t_{ik}|}$, $0 < \phi < 1$, as we expected the correlation among y values to decrease with distance in time.

The aim of the following experiments was to find out if the estimated value of a could be used to approximate the global optimum l_o. Figure 1 reports results in terms of the relative deviation from the optimum, $d = (l_a - l_o)/l_o$, where $l_a = l_b(1 + a)$ is the asymptote of `SSasymp` scaled back. We first fit a separate model for each of the 400 instances, using data from all the 25 runs available. In this case the factor which identifies the subjects is the random seed. Random effects were negligible, and the model seemed to account for the variations among runs only with the variance-covariance structure. Here and in all other experiments, the value of ρ in `varPower` was estimated to be negative, giving a decreasing variance structure, and the correlation parameter ϕ was also significant. On some of the instances, `nlme` failed for numerical issues, namely the singularity of a matrix: this was observed mostly on the group of smaller instances, on which the algorithm was often too fast in converging, producing only a few (y, t) points for each run. The parameters estimated were found to be significant (p-value < 0.05), with the exception of c on some of the instances. Figure 1(a) plots statistics of the deviation from the optimum d, evaluated using the fixed effect of the asymptote a.

The next experiment was performed grouping the data based on the instance: 25 models were fit, one for each random seed, on the four separate groups of 100 instances each. This time the random effects were more remarkable, as each individual corresponded to a different instance, with a different optimum. In this case d was evaluated based on the mixed effect $(a + a_i)$, a_i being the random effect on the asymptote for instance i. Aggregated statistics of d are reported in Figure 1(b): the performance decreased visibly, but the estimates are still close to the real global optima.

The third experiment was a feasibility study for a model based stopping criterion, aimed at investigating the predictive power of our model. In a first phase, the model was trained based on data for a single random seed, grouped based

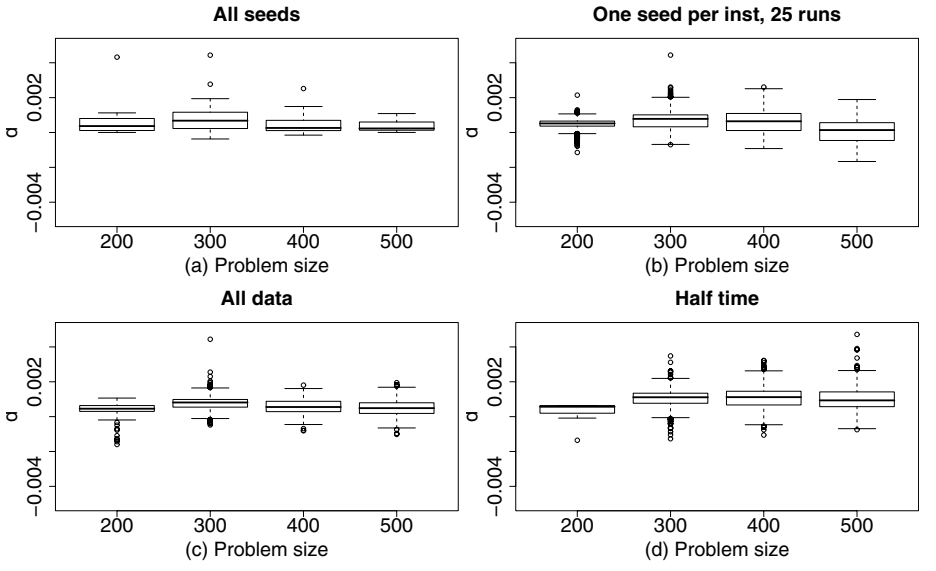

Fig. 1. All plots display statistics of $d = (l_a - l_o)/l_o$, the deviation of the estimate l_a from the actual optimum l_o. See text for details.

on the instance. In a second phase, a fresh model was trained on a subset of the same sample, obtained dropping the data from the second half of the time axis, for a half of the instances, randomly picked. The idea was to simulate a situation in which 50 "training" instances have been already solved to convergence, and the relative data recorded; while another 50 "test" instances are being solved, the algorithm is paused some time before convergence, and the model is used to predict the value of the optima, based on the previously solved instances, and on the learning curves observed so far. This scheme was repeated for each random seed, each time with a different random pick of the 50 test instances. Also in this case $(a + a_i)$ was used to evaluate l_a. Figures 1(c,d) report statistics of d for the two models, measured only on the test instances. The performance of the model from the first phase, which serves as an "oracle" for comparison, is clearly superior, but the one for the second phase is still reasonable.

4 Related Work

An up-to-date review of longitudinal data analysis, including nonparametric methods, can be found in [1]. We mainly followed [3] which is also rich in usage examples of `nlme`, from the same authors. The literature on algorithm performance modeling is mostly focused on decision problems, and the related concept of runtime distribution. Extreme value statistics was proposed in [7] to estimate the value of the global optimum for large problem instances, based on a sequence of suboptimal solutions; this method is integrated into local search in [8]. The

learning curve of local search solvers is fit in [9] using a separate model for each run. Solution quality distributions are used in [10] to rank the performance of heuristic solvers. We refer the reader to [4] for further references.

5 Conclusions

We reviewed a class of models for longitudinal data, and showed how they can be used to model the performance of optimisation algorithms, investigating their predictive power with a preliminary experiment on data from a TSP solver. The results were quite promising, and we are currently analyzing other algorithm/problem combinations, as well as the impact of covariates. The models are quite general in this sense, as they allow time-varying covariates to be easily included: besides runtime, the distribution of solution quality could also be related to memory usage, bandwidth, or other resources, as well as dynamic variables of the algorithms. Adding algorithm parameters as stationary covariates could allow to tune them based on derivatives of the objective. Longitudinal data analysis could be useful to implement stopping criteria, or dynamic restart strategies, which take into account past experience in detecting the convergence of an algorithm; and to perform algorithm selection, or some more general form of resource allocation.

Acknowledgement. The first author was supported by the Swiss National Science Foundation with a grant for prospective researchers (n. PBTI2–118573).

References

1. Fitzmaurice, G., Davidian, M., Verbeke, G., Molenberghs, G.: Longitudinal Data Analysis. Chapman & Hall/CRC Press (2008)
2. Lindstrom, M.J., Bates, D.M.: Nonlinear mixed effects models for repeated measures data. Biometrics 46(3), 673–687 (1990)
3. Pinheiro, J.C., Bates, D.M.: Mixed Effects Models in S and S-Plus. Springer, Heidelberg (2002)
4. Hoos, H.H., Stützle, T.: Stochastic Local Search: Foundations & Applications. Morgan Kaufmann, San Francisco (2004)
5. Stützle, T., Hoos, H.H.: Analysing the run-time behaviour of iterated local search for the travelling salesman problem. In: Hansen, P., et al. (eds.) Essays and Surveys on Metaheuristics, pp. 589–611. Kluwer Academic Publishers, Dordrecht (2001)
6. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2009)
7. Dannenbring, D.G.: Procedures for estimating optimal solution values for large combinatorial problems. Management Science 23(12), 1273–1283 (1977)
8. Ovacik, I.M., Rajagopalan, S., Uzsoy, R.: Integrating interval estimates of global optima and local search methods for combinatorial optimization problems. Journal of Heuristics 6(4), 481–500 (2000)
9. Oppen, J., Woodruff, D.: Parametric models of local search progression. Technical Report 06-08, UC Davis Graduate School of Management Research (2008)
10. Schreiber, G.R., Martin, O.C.: Cut size statistics of graph bisection heuristics. SIAM J. on Optimization 10(1), 231–251 (1999)