Curve Tailoring with Interactive Computer Graphics

by

J.E. Dennis, Jr.[1]

and

Daniel J. Woods[2]

Technical Report TR84-4 , November 1984.
Revised April 1987.

[1]Mathematical Science Department, Rice University, Houston, Texas, 77251. Research sponsored by DOE DE-AS05-82ER13016, ARO DAAG-79-C-0124, NSF MCS81-16779.

[2]IMSL, Inc., 2500 ParkWest Tower One, 2500 CityWest Blvd., Houston, Texas, 77042. Research sponsored by NSF grant MCS-81-21884.

Abstract

We are exploring ways to use the increasingly common graphics workstations (we use a SUN) to make numerical computation more convenient. The particular experiment reported on here involves the use of the *mouse* to obtain input from the user that he usually supplies as a subroutine. Specifically, the user specifies a set of one-dimensional data and a model whose parameters are to be chosen to best fit the data. He does not specify what optimality criterion is to be used to pick the parameters. Instead, he is asked to point out his preference from displayed prospective fits. This information is used by the Nelder-Mead simplex algorithm to improve the fit during the exploratory data analysis phase. The information is also used to determine a fitting criterion that is consistent with the selections made during the process. It is then possible to determine a final set of parameters that are optimal for this criterion.

1 Introduction.

This paper reports on an experiment into the use of interactive computer graphics as an integral part of the formulation and solution of optimization problems which require multi-criterion objective functions. In this experiment, we are interested in curve fitting on a SUN3 workstation. We have a system to help a user with a parametric model and data of varying relevance to find an appropriate weighted fitting criterion and the corresponding optimal values of the parameters.

Our prototype is for the user who has 1-dimensional data and a model involving 8 or fewer parameters. We assume in this note that the user would be content with parameters that minimize a weighted sum-of-squares of the residuals at the data points but weighted l_1 and weighted l_∞ are also available. It is important that we do *not* require the user to specify the weights. We will explain in Section 2 how we deduce weights from information provided by the user. The system obtains information from the user by displaying a graphical representation of the problem and a pair of parameter vectors and then asking the user to indicate which parameter vector is preferable. This *quantitative* information is recorded for later use.

2 The Problem.

We assume the user is interested in finding an x^* which minimizes

$$\Phi(w^*, x) \equiv \sum_{i=1}^{n} w_i^* \left[y(x, t_i) - y_i\right]^2, \tag{1}$$

where $w^* = (w_1^*, w_2^*, \ldots, w_n^*)^T$ is a nonnegative weight vector, and the user provides the n pieces of data $(t_i, y_i), i = 1, 2, \ldots, n$ and a model $y(x, t)$, where $x \in \mathbb{R}^p$ is the vector of p-parameters. If w^* is known, then the user's problem is a nonlinear least squares problem for which efficient techniques and software are available. Our system is designed for the situation where w^* is not known and cannot be easily determined from the data. However, our system also provides a simple interface to efficient software for nonlinear least squares as well as providing graphical facilities for displaying the problem and solution.

The major requirement on w^* is that the corresponding values of $\Phi(w^*, x)$ be consistent with the order information indicated by the user for various values of x. Specifically, if the user has indicated the qualitative information that for some m parameter pairs (x_{j_l}, x_{k_l}), $l = 1, 2, \ldots, m$, he prefers the fit provided by x_{j_l} to the fit provided by x_{k_l}, then we say the weight vector w is Φ – *consistent* if

$$\Phi(w, x_{j_l}) \geq \Phi(w, x_{k_l}), \qquad l = 1, 2, \ldots, m. \tag{2}$$

We also add the n *nonnegativity* constraints

$$w_i \geq 0, \qquad i = 1, 2, \ldots, n, \tag{3}$$

as well as the *normalizing* constraint

$$\sum_{i=1}^{n} w_i = n. \tag{4}$$

By (1), it is clear that (4) is consistent with (2) and (3). It will be useful to let W denote the set of all $w \in \mathbb{R}^n$ that satisfy all of (2), (3), and (4). We will call any $w \in W$ *feasible* .

The constraints (3) and (4) ensure that any feasible w would be reasonable. We will resolve the remaining lack of specificity by asking that each w_i^* differ as little from some prescribed value $\bar{w}_i$ as possible. (Currently, the prescribed values are all set to 1 so that the w_i's will differ as little from *equality* as possible.) For example, we might choose w^* by the l_∞ criterion,

$$\min_{w \in W} \max_{1 \leq i \leq n} |\bar{w}_i - w_i|, \tag{5}$$

or the l_1 criterion

$$\min_{w \in W} \sum_{i=1}^{n} |\bar{w}_i - w_i|, \tag{6}$$

or the l_2 criterion

$$\min_{w \in W} \sum_{i=1}^{n} (\bar{w}_i - w_i)^2. \tag{7}$$

The l_∞ and l_1 criteria require solving linear programming problems and the l_2 criterion requires solving a quadratic program. We are currently using the l_2 criterion given by (7).

3 The Solution.

It would be conceivable to generate some random parameter values, ask the user to rank the corresponding fits, solve for weights w^* using one of the criteria of the last section, and then apply some library optimizer to find an x^* that minimizes $\Phi(w^*, x)$. We favor another scheme which asks for user rankings that are used to improve the current parameter estimates. We believe that our scheme will be more efficient and will find better weights because it bases Φ-consistency on comparisons more interesting to the user.

We generate successive parameter values using the Nelder-Mead [NelM] simplex algorithm. This algorithm is known to be efficient for $p \leq 5$ and it is extremely tolerant of inaccuracy in the objective function values. We will not give details of the algorithm here (see [NelM] or [Wood]), but it is useful to point out that the algorithm is iterative and that each iterate is not a point but is a p-simplex of parameters characterized by its $p+1$ vertices. Also, the amount of work per iteration is usually $O(n)$.

At each iteration, objective function values at the vertices are used only to label the best, the worst and the next-worst vertices. Since w^* is not known, we can not evaluate $\Phi(w^*, x)$ at the vertices to obtain this information, so we get it directly by asking the user to rank the fits corresponding to the vertices of the simplex applied to the user's model.

Of course, we could identify the optimal parameters independent of any assumed form for Φ by this scheme alone. We allow the user this option, but we also provide a library subroutine to minimize Φ. If the user wishes to minimize Φ, we check to make sure w^* is up-to-date and minimize $\Phi(w^*, x)$. If the weights are not up-to-date we solve the subproblem to find w^* and then invoke the library optimizer. It is possible for the user to make inconsistent choices. In the current version of the software, we handle inconsistencies in the rankings only by telling the user that his choices have been inconsistent. We could add the capability of adjusting or removing inconsistent constraints. This is discussed in greater detail by Woods [Wood].

4 Current Work.

There are several reasons why we want to proceed to the stage of assuming a form like (1) and then to finding weights. We think it would be an interesting part of the data analysis for the user to have weights arrived at adaptively as a part of the analysis rather than by *a priori* assignment or by ranking plots that were chosen artificially. Once he has weights w^*, then similar data can be analyzed directly using our system in *automatic* mode to interface to a library optimization routine without redoing the interactive Nelder-Mead portion which we call *user* mode, or else *user* mode could be entered at an advanced stage. In fact, the user will be able to enter *user* mode with his *a priori* estimates of the weights. Our favorite reason to find weights adaptively is that we believe that comparisons at the Nelder-Mead vertex points are likely to point up inconsistencies in improperly assigned weights and lead to a redefining of the weights so that they are feasible. We expect one of three results if the Nelder-Mead process goes on long enough:

1. The user is satisfied with the fit and the session ends with parameters corresponding to the fit and a set of feasible weights.
2. The user selects *automatic* mode and is satisfied with the fit. The user obtains a set of optimal parameters and the associated weights.
3. The user makes inconsistent choices. In this case, we inform the user of his inconsistent choices and have no facility for resolving this problem. However, the user may continue via the Nelder-Mead process.

There are many other features that we intend to add. These options include such items as:

- Allowing the user to adjust the graphical display of the data and plots. We intend to add numerous graphical capabilities to the system.
- Letting the user decide which norm to use in defining Φ in (1) and in determining the 'optimal' weights.
- We would like to slowly take over for the user. We will put the mouse where we expect he will rank the next plot. This would enable us to build up the user's confidence in the weights and in our procedure.

- Add efficient solution methods for solving the subproblem to determine the weights. These techniques are discussed in Woods [Wood]. Also, this provides immediate recognition of an inconsistent choice by a user (since no feasible weights will exist).
- Inconsistent choices by a user should be handled gracefully by the system as described in [Wood].
- Letting a user specify the prescribed weights $\bar{w}$ that are used to determine w^* in (5), (6), and (7).

REFERENCES

[NelM] Nelder, J.A. and R. Mead, "A simplex method for function minimization", *Computer Journal 7* , pp.308-318.

[Wood] Woods, Daniel, J., *An Interactive Approach for Solving Multi-Objective Optimization Problems*, Mathematical Science Technical Report 85-5, May 1985, Rice University, Houston, Texas, 77251.