AMO - Advanced Modeling and Optimization Volume 1, Number 2, 1999

The TOMLAB Graphical User Interface for Nonlinear Programming [1]

Erik Dotzauer[2] and Kenneth Holmström[3]

Center for Mathematical Modeling
Department of Mathematics and Physics
Mälardalen University, P.O. Box 883, SE-721 23 Västerås, Sweden

Abstract

The paper presents a Graphical User Interface (GUI) for nonlinear programming in Matlab. The GUI gives easy access to all features in the NLPLIB TB (NonLinear Programming LIBrary Toolbox); a set of Matlab solvers, test problems, graphical and computational utilities for unconstrained and constrained optimization, quadratic programming, unconstrained and constrained nonlinear least squares, box-bounded global optimization, global mixed-integer nonlinear programming, and exponential sum model fitting. The GUI also runs the linear programming problems in the linear and discrete optimization toolbox OPERA TB. Both NLPLIB TB and OPERA TB are part of TOMLAB; an environment in Matlab for research and teaching in optimization. Presently, NLPLIB TB implements more than 25 solver algorithms, and it is possible to call solvers in the Math Works Optimization Toolbox. MEX-file interfaces are developed for seven Fortran and C solvers, and others are easily added using the same type of interface routines. There are four ways to solve a problem: by a direct call to the solver routine or a call to a multi-solver driver routine, or interactively, using the Graphical User Interface or a menu system. The GUI may also be used as a preprocessor to generate Matlab code for stand-alone runs. A large set of standard test problems is implemented in TOMLAB. Furthermore, using MEX-file interfaces, problems in the CUTE test problem data base and problems defined in the AMPL modeling language can be solved.

Keywords: Nonlinear Programming, Matlab, CUTE, AMPL, Graphical User Interface, Software Engineering, Mathematical Software, Optimization, Algorithms, Exponential Sum Fitting, Nonlinear Least Squares.

AMS Subject Classification: 90C30, 90C20, 90C45

1 Introduction

Many scientists and engineers are using Matlab as a modeling and analysis tool, but for the solution of optimization problems, the support is weak. This was one motive for starting the development of TOMLAB [9, 8]; an open and general environment in Matlab for research and teaching in optimization. TOMLAB preferably solves problems implemented as Matlab m-file code. Using a MEX-file interface to communicate with solvers implemented in Fortran or C, there is no need to rewrite a problem into these languages.

In this paper we present a Graphical User Interface (GUI) for nonlinear programming, implemented in Matlab as part of the NLPLIB TB (NonLinear Programming LIBrary Toolbox) [10]; a set of

[1] Financed by the Mälardalen University Research Board, project *Applied Optimization and Modeling (TOM).*
[2] E-mail: **erik.dotzauer@mdh.se**
[3] E-mail: **hkh@mdh.se**; URL: **http://www.ima.mdh.se/tom**.

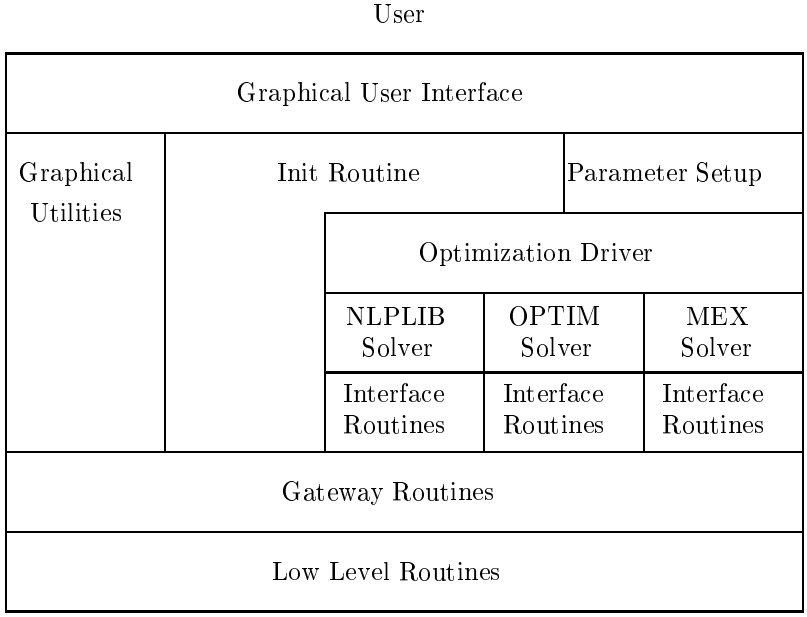

Figure 1: The NLPLIB Toolbox hierarchy.

Matlab solvers, test problems, graphical and computational utilities for nonlinear optimization and nonlinear parameter estimation. The GUI is also running linear programs defined in the linear and discrete optimization toolbox OPERA TB [11]. Both NLPLIB TB and OPERA TB are part of TOMLAB. The GUI gives easy access to the large set of optimization solvers, test problems and utilities in NLPLIB TB.

Interfaces to optimization problems formulated in the AMPL modeling language [5] and the CUTE SIF language [1] have been developed. Using the tools for analysis in TOMLAB, a researcher does not need to convert a problem written in any of these languages. This makes TOMLAB an excellent environment for the development of optimization algorithms and software.

TOMLAB currently solves small and medium size dense problems; the available memory is the principal restriction. Information about the current version of TOMLAB, including NLPLIB TB and OPERA TB, is available at the URL: **http://www.ima.mdh.se/tom**. A detailed description of TOMLAB and its subparts are given in the TOMLAB v1.0 User's Guide [12], which can be downloaded from this site.

In the next section a brief presentation of the NLPLIB TB is given. Section 3 and Section 4 presents the Graphical User Interface. In Section 5 the interaction with other program packages is discussed.

2 The NLPLIB Toolbox

NLPLIB TB is a set of Matlab m-files, which implements optimization solvers, test problems and utilities for nonlinear optimization and nonlinear parameter estimation. The focus is on dense problems. There are eight main fields; unconstrained and constrained optimization, quadratic programming, unconstrained and constrained nonlinear least squares, box-bounded global optimization, global mixed-integer nonlinear programming and fitting of sums of exponential functions to data. A ninth field in the GUI is linear programming using routines and problems in OPERA TB.

There are four ways to solve a problem: by a direct call to the solver routine (the NLPLIB TB QuickRun option) or by a call to a multi-solver driver routine, or interactively, using the GUI or a menu system.

A hierarchy describing the functionality of NLPLIB TB is shown in Figure 1. In the Low Level

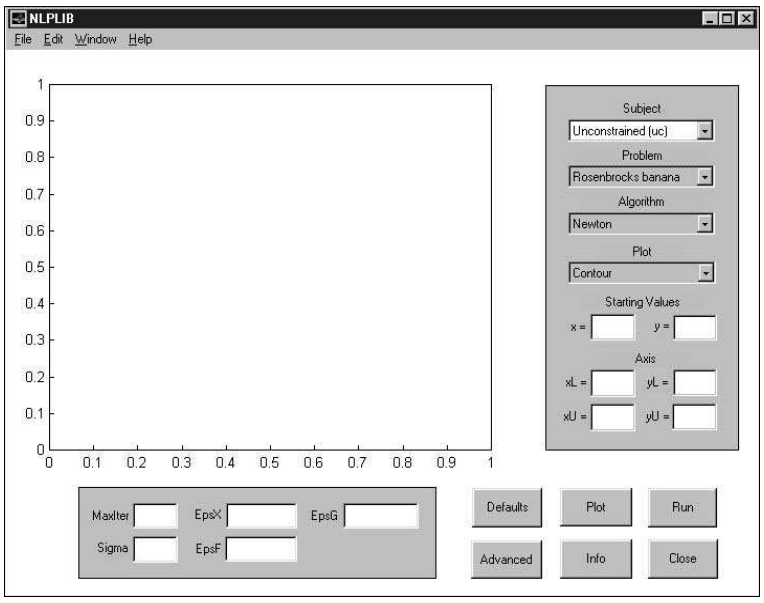

Figure 2: The GUI in Normal mode.

Routines the mathematical descriptions of the optimization problems are implemented. From the Graphical User Interface, the user can access the problems either as graphical information, using the Graphical Utilities, or as text information, using the Init Routine. Before solving a problem all parameters needed are defined using the Parameter Setup. The Optimization Driver calls the selected solver. These can either be NLPLIB TB solver algorithms, solvers in the Math Works Optimization Toolbox (OPTIM) [3] or general-purpose solvers implemented in Fortran or C. Presently, NLPLIB TB implements more than 25 solver algorithms. Fortran and C solvers are called from Matlab using a MEX-file interface. Depending on which solver is used, different Interface Routines are used to communicate with the Low Level Routines via the Gateway Routines. The Gateway Routines does the book-keeping, keep track of search directions, and determines type of differentiation; analytic, automatic, or any of the different types of numerical differentiation. A detailed description of NLPLIB TB (and OPERA TB) is given in the TOMLAB User's Guide [12].

3 The Graphical User Interface

The Graphical User Interface (GUI) is started by calling the Matlab m-file *nlplib.m*, i.e. by entering the command *nlplib* at the Matlab prompt. The GUI has two modes; Normal and Advanced. At start the GUI is in Normal mode, shown in Figure 2. There are one axes area, four menus; Subject, Problem, Algorithm and Plot, and six push buttons; Defaults, Advanced, Plot, Info, Run and Close.

There are also eleven edit controls where it is possible to enter parameter values used by the solver algorithms. To the right of the axes area, starting values for two dimensional problems can be given. How to define starting values for problems with more than two decision variables is discussed in Section 4. The edit controls labeled 'Axes' set the axes in the contour plot and the mesh plot. The edit controls below the axes area are used to set the optimization parameters sent to the solvers. These parameters are the maximum number of iterations (MaxIter), the line search accuracy σ (Sigma), the termination tolerance on the change in the decision variables (EpsX), the termination tolerance on the function value (EpsF) and the termination tolerance on the gradient (EpsG). If a solver for constrained optimization is selected, a twelfth edit control (EpsC) is shown. This edit control sets the termination tolerance on the constraint violation.

In the axes area plots and information given as text are displayed.

The Subject menu is used to select subject, i.e. which type of problem to be solved. There are cur-

rently nine main problem types; unconstrained (simply bounded) nonlinear optimization, quadratic programming, constrained nonlinear optimization, unconstrained (simply bounded) nonlinear least squares, exponential sum model fitting, constrained nonlinear least squares, linear programming, box-bounded global optimization and global mixed-integer nonlinear programming.

From the Problem menu, the user selects the problem to be solved. Presently, there are about 15 to 50 predefined test problems for each problem type. The user can easily define his own problems and try to solve them using any solver. On-line information about how to simply and fast define new problems is displayed when entering the command *usr* at the Matlab prompt. Instructions on how to define a large set of problems are given in the User's Guide [12].

The Algorithm menu is used to select a solver routine and any of the the solver's implemented algorithms. It can either be a NLPLIB TB solver, a solver in the Optimization Toolbox or a general-purpose solver implemented in Fortran or C.

Changing type of optimization problem in the Subject menu, will change the menu entries in the Problem menu and the Algorithm menu.

From the Plot menu, the type of plot to be drawn is selected. The different types are contour plot, mesh plot, plot of function values and plot of convergence rate. The contour plot and the mesh plot can be displayed either in the axes area or in a new figure. The plot of function values and convergence rate are always displayed in a new figure. For nonlinear least squares problems and exponential sum model fitting problems it is possible to plot the residuals, the starting model and the estimated model.

When pushing the Defaults button, the default values for every parameter are shown in the edit controls. If pushing the button again, the parameters will disappear. Before solving a problem, the user can change any of the values. If leaving an edit control empty, the default values are used.

The Advanced button and the Advanced mode are described in Section 4.

Pushing the Plot button gives a plot of the current problem. In the contour plot, known local minima, known local maxima and known saddle points are shown. It is possible to make a contour plot and a mesh plot without first solving the problem. After the problem is solved, a contour plot shows the search direction and trial step lengths for each iteration. A contour plot of the classical Rosenbrock banana function, together with the iteration search steps with marks for the line search trials, is shown in Figure 3. A contour plot for a constrained problem and a plot of the data and the obtained model for a nonlinear least squares problem are given in Figure 4. In the contour plot, (inequality) constraints are depicted as dots. Starting from the infeasible point $(x_1, x_2) = (-5.0, 2.5)$, the solution algorithm first finds a point inside the feasible region. The algorithm then iteratively finds new points. For several of the search directions, the full step is too long and violates one of the constraints. Marks show the line search trials. Finally, the algorithm converges to the optimal solution $(x_1^*, x_2^*) = (-9.5474, 1.0474)$.

The Info button gives some information about the current problem, e.g. the number of variables.

When the user has chosen a solver and a problem, he then pushes the Run button to solve it. When the algorithm has converged, information about the solution procedure is displayed. This information will include the solution found, the function value at the solution, the number of iterations performed, the number of function evaluations, the number of gradient evaluations, the number of floating point operations needed and the computation time.

To close the GUI, push the Close button.

4 The Advanced Mode

When pushing the Advanced button, the GUI will change to Advanced mode. The axes area is replaced by more edit controls and menus, see Figure 5. Furthermore, the Advanced button is renamed to Figure button. To change from Advanced mode to Normal mode, push the Figure button.

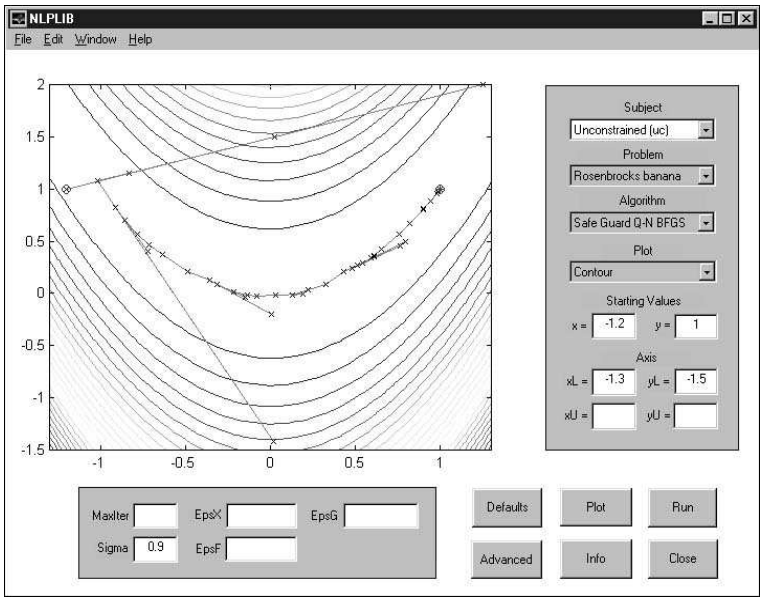

Figure 3: A contour plot with the search directions and marks for the line search trials for each iteration.

There are some new edit controls in the Advanced mode. FLow, the best guess on a lower bound for the optimal function value, is used by NLPLIB TB solver algorithms using the Fletcher line search algorithm [4]. The parameter EpsR is the rank test tolerance in the subspace minimization technique used when determining the search direction in some of the algorithms.

For problems with more than two decision variables, starting values for decision variable x_3 to x_n are given in the edit control named 'Starting Values x3 - xn'. Starting values for x_1 and x_2 are given in the edit controls labeled 'Starting Values'. To make a contour plot or a mesh plot for problems with more than two decision variables, the user selects the two-dimensional subspace to plot. The indices of the decision variables defining the subspace are given in the edit controls called 'Variables At Axis When $n > 2$'. The view for a mesh plot is changed using the edit controls 'Mesh View'.

There are six new menus in the Advanced mode. The first menu selects method to compute first and second derivatives. If not using analytical expressions, derivatives can be computed either by automatic differentiation using the ADMAT Toolbox, distributed by Arun Verma at the URL: **http://simon.cs.cornell.edu/home/verma/AD**, or by five different methods for numerical differentiation (three of them requiring the Spline Toolbox to be installed). The second menu determines if a quadratic or a cubic interpolation shall be used in the line search algorithm.

Two menus are used to select the level of output from the optimization driver and the optimization solver. All output printed during the optimization are displayed in the Matlab Command Window. If the 'Pause Each Iteration' check box is selected, the NLPLIB TB solvers are using the pause statement to halt after each iteration. The menu 'Init File' selects the file defining the current set of problems. Changing the set of problems will change the Problem menu. The menu named 'Method' differs between problem types. Using an unconstrained solver or a nonlinear least squares solver, the menu selects method to compute the search direction. In the constrained case, the Method menu gives the quadratic programming solver to be used in SQP algorithms.

If the check box 'Hold Previous Run' is selected, all information about the runs are stored. Making a contour plot, the step and trial step lengths for all solution attempts are drawn. This option is useful, e.g. when comparing the performance of different algorithms or checking how the choice of starting point affects the solution procedure.

For some predefined test problems it is possible to set parameter values when initializing the problem. These parameters can for example be the size of the problem, the number of residuals or the

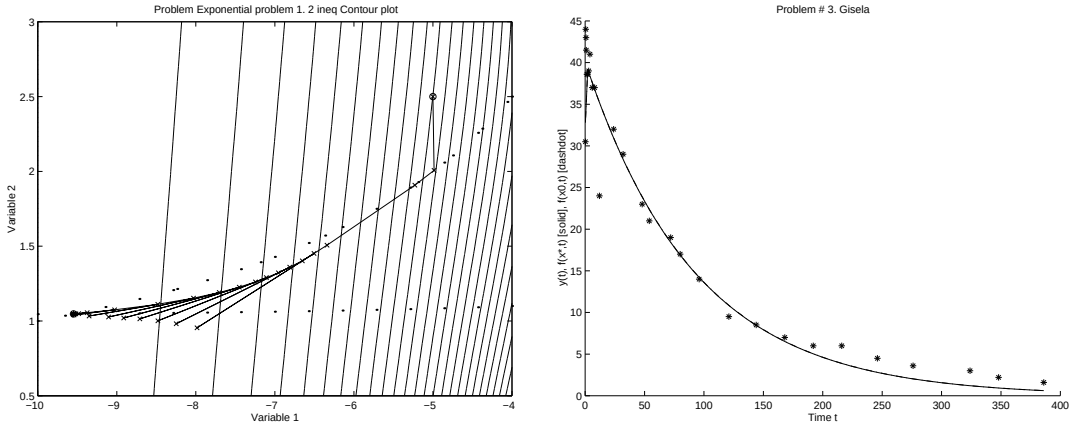

Figure 4: A contour plot for a constrained problem and a plot of data and model for a nonlinear least squares problem.

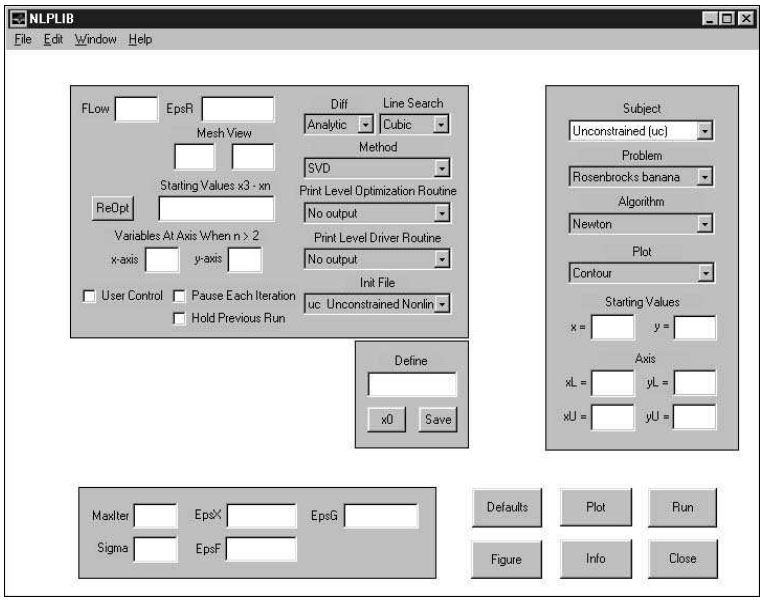

Figure 5: The GUI in Advanced mode.

number of constraints. Questions about the parameters will appear when selecting the check box named 'User Control'. If the 'User Control' check box is not selected, default values will be used.

When selecting an exponential sum model fitting problem, two new menus and a new edit control will appear. With these the number of exponential terms in the approximating model and one of four types of residual weighting are determined. Furthermore, there is a choice whether to solve the weighted least squares approximation problem using an ordinary or separable nonlinear least squares algorithm.

In the Advanced mode there are three new push buttons. If a contour plot is displayed in the axes area and the user pushes the button named 'x0', it is possible to select starting point for the current algorithm using the mouse. Pushing the ReOpt button, the current problem is re-optimized with the starting point defined as the solution found in the previous solution attempt.

Entering a name in the edit control labeled 'Define' and pushing the Save button will generate two files; one Matlab mat-file and one Matlab m-file. The name should not include any extension. For example, entering the name *test* in the edit control, the files *test.mat* and *test.m* will be generated.

Variable	Parameter
Result.Name	Problem name
Result.Solver	Solver name
Result.SolverAlgorithm	Algorithm name
Result.x_0	Starting point
Result.x_k	Solution found
Result.f_k	Function value at the solution
Result.Iter	Number of iterations performed
Result.FuncEv	Number of function evaluations
Result.GradEv	Number of gradient evaluations
Result.ConstrEv	Number of constraint evaluations
Result.Nflops	Number of floating point operations
Result.CPUtime	Computation time

Table 1: Some information stored in the global structure *Result*.

The files are saved in the current directory. Parameters are stored in the mat-file, and in the m-file all commands needed to make a stand-alone run without using the GUI are defined. The parameter values are those currently used by the GUI.

If entering a name in the 'Define' edit control and pushing the Defaults button, the default values for all parameters will be loaded from the current mat-file.

When a problem is solved, the user can access some of the results from in the Matlab Command Window, stored in the global structure *Result*. If the user has not run the NLPLIB TB initialization command *nlplibInit*, he must enter the command *global Result* at the Matlab prompt to declare *Result* as a global structure. Some information stored in the structure are given in Table 1. To display the full structure, enter *Result* at the prompt. To display a specific field in the structure, e.g. the solution found, enter *Result.x_k*. When the check box 'Hold Previous Run' is selected, *Result* becomes a structure array. As an example, to display the results from the third run, enter the command *Result(3)*. To display the solution found in the third run, enter the command *Result(3).x_k*.

5 Interaction with Other Program Packages

TOMLAB is an open system with possibilities to interact with other program packages. An optimization solver implemented in Fortran or C is called from TOMLAB using a MEX-file interface. Presently, MEX-file interfaces have been developed for six general-purpose solvers available from the Systems Optimization Laboratory, Department of Operations Research, Stanford University, California; NPSOL 5.02 [7], NPOPT 1.0-10 (updated version of NPSOL), NLSSOL 5.0-2, QPOPT 1.0-10, LSSOL 1.05 and LPOPT 1.0-10. Furthermore, an interface to MINOS 5.5 [13] has been developed. MEX-file interfaces are available for both UNIX and PC systems.

A TOMLAB interface to the well-known standard optimization testing environment CUTE [1, 2] has been developed. The interface is a further development of the Matlab interface routines supplied in the CUTE distribution. It is thus possible to run the huge set of CUTE test problems in NLPLIB TB and the GUI. Based on the Matlab interface described in [6], a TOMLAB interface to the AMPL modeling system [5] has been developed. The interface is using files in the AMPL internal binary *nl* format. Indeed, any solver callable from NLPLIB TB may now solve problems formulated in the AMPL language.

6 Conclusions

The Graphical User Interface is a powerful and unique utility, usable both for research and teaching in optimization. It further enhances the possible use of TOMLAB and simplifies the actual analysis. The GUI is a step forward in making the solution of applied nonlinear programming problems applicable to not only advanced researchers in the field of optimization.

References

[1] I. Bongartz, A. R. Conn, N. I. M. Gould, and P. L. Toint. CUTE: Constrained and Unconstrained Testing Environment. *ACM Transactions on Mathematical Software*, 21(1):123–160, 1995.

[2] I. Bongartz, A. R. Conn, Nick Gould, and Ph. L. Toint. CUTE: Constrained and Unconstrained Testing Environment. Technical report, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, September 2 1997.

[3] Mary Ann Branch and Andy Grace. *Optimization Toolbox User's Guide*. 24 Prime Park Way, Natick, MA 01760-1500, 1996.

[4] Roger Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, New York, 2nd edition, 1987.

[5] R. Fourer, D. M. Gay, and B. W.Kernighan. *AMPL - A Modeling Language for Mathematical Programming*. The Scientific Press, Redwood City, CA, 1993.

[6] David M. Gay. Hooking your solver to AMPL. Technical report, Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974, 1997.

[7] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. *User's Guide for NPSOL (Version 4.0): A Fortran package for nonlinear programming*. Department of Operations Research, Stanford University, Stanford, CA, 1986. SOL 86-2.

[8] Kenneth Holmström. TOMLAB - An Environment for Solving Optimization Problems in Matlab. In M. Olsson, editor, *Proceedings for the Nordic Matlab Conference '97, October 27-28*, Stockholm, Sweden, 1997. Computer Solutions Europe AB.

[9] Kenneth Holmström. The TOMLAB Optimization Environment in Matlab. *Advanced Modeling and Optimization*, 1(1):47–69, 1999.

[10] Kenneth Holmström and Mattias Björkman. The TOMLAB NLPLIB Toolbox for Nonlinear Programming. *Advanced Modeling and Optimization*, 1(1):70–86, 1999.

[11] Kenneth Holmström, Mattias Björkman, and Erik Dotzauer. The TOMLAB OPERA Toolbox for Linear and Discrete Optimization. *Advanced Modeling and Optimization*, 1(2):1–8, 1999.

[12] Kenneth Holmström, Mattias Björkman, and Erik Dotzauer. TOMLAB v1.0 User's Guide. Technical Report IMa-TOM-1999-01, Department of Mathematics and Physics, Mälardalen University, Sweden, 1999.

[13] Bruce A. Murtagh and Michael A. Saunders. MINOS 5.4 USER'S GUIDE. Technical Report SOL 83-20R, Revised Feb. 1995, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California 94305-4022, 1995.