MODEL DEVELOPMENT AND OPTIMIZATION WITH *Mathematica*™

János D. Pintér[1] and Frank J. Kampas[2]
[1] *Pintér Consulting Services, Inc., Halifax, Nova Scotia, Canada*
jdpinter@hfx.eastlink.ca http://www.pinterconsulting.com http://www.dal.ca/~jdpinter

[2] *WAM Systems, Inc., Plymouth Meeting, PA, USA*
fkampas@wamsystems.com http://www.wamsystems.com

Appeared in:

Golden, B. L., Raghavan, S. and Wasil, E. A., Editors
The Next Wave in Computing, Optimization, and Decision Technologies
pp. 285-302.

Springer Science + Business Media, New York, 2005.

MODEL DEVELOPMENT AND OPTIMIZATION WITH *Mathematica*™

János D. Pintér[1] and Frank J. Kampas[2]
[1] *Pintér Consulting Services, Inc., Halifax, Nova Scotia, Canada*
jdpinter@hfx.eastlink.ca http://www.pinterconsulting.com http://www.dal.ca/~jdpinter

[2] *WAM Systems, Inc., Plymouth Meeting, PA, USA*
fkampas@wamsystems.com http://www.wamsystems.com

Abstract: *Mathematica* is an integrated scientific and technical computing system, with impressive numerical calculation, programming, symbolic manipulation, visualization and documentation capabilities. In recent years *Mathematica*'s optimization related features have been significantly expanded, both by in-house development and by application packages. Such developments make it an increasingly useful tool also in Operations Research studies. We review and illustrate these features, placing added emphasis on nonlinear (global and convex) optimization, and – within this context – discussing the application packages *MathOptimizer* and *MathOptimizer Professional*.

Key words: *Mathematica*; built-in optimization functions; modeling and optimization packages; *MathOptimizer*; *MathOptimizer Professional*; illustrative examples and applications.

1. INTRODUCTION

Mathematica – an integrated scientific and technical computing environment by Wolfram Research (2004) – is arguably one of the most sophisticated software products available today. Its capabilities and range of applications are documented in the massive *Mathematica* tome (Wolfram, 2003) and in the supplementary documentation. Further information is found in nearly 400 topical books, and in thousands of articles and presentations. According to Wolfram Research, the software is used by well over a million people worldwide.

Mathematica can also increasingly meet the needs of Operations Research professionals, including business analysts, model, algorithm and software developers, researchers, professors, and students. O.R. related

features include data analysis and management, model prototyping, concise programming (in several paradigms), advanced computing, visualization, and documentation – all in the same 'live' notebook document, if preferred. Such notebooks can also be directly converted to tex, html, xml, ps, and pdf file formats. *Mathematica* also supports direct links to external application packages, to other software products, and to the Internet. A significant further advantage is portability across a broad range of hardware platforms and operating systems, due to the standardized notebook document format.

For further general information, visit the websites of Wolfram Research, specifically including the *Mathematica* Information Center (2004) that provides extensive details and links. We also refer to a recent review of *Mathematica* in *ORMS Today* (Sodhi, 2003), as well as to an illustrative list of *Mathematica* books with a modeling and/or optimization related content (Bahder, 1995; Schwalbe and Wagon, 1996; Gass, 1998; Bhatti, 2000; Maeder, 2000; Jacob, 2001; Hollis, 2003; Pemmaraju and Skiena, 2003; Kampas and Pintér, 2004). Let us note here that *MathReader*, a freely available viewer, can be used to display and print *Mathematica* notebooks, animate graphics, play sounds, and copy information from notebooks to other documents; *MathReader* can also be used in most web browsers.

In this work we review and illustrate *Mathematica*'s O.R. modeling and optimization related features. Within the broad category of optimization models, we see particularly strong application potentials for *Mathematica* in the analysis of (possibly complex) nonlinear systems when the corresponding decision model can not be brought to simple standard forms. In such cases, problem-specific modeling and code development are essential, and using *Mathematica* as the development platform can be a good choice. For this reason, here we shall place added emphasis on nonlinear (global and convex) optimization, where – in addition to built-in functionality – our packages *MathOptimizer* and *MathOptimizer Professional* can be put to good use.

2. NUMERICAL OPTIMIZATION IN *Mathematica*

We start with a concise summary of built-in optimization functionality. Most of the related *Mathematica* functions can be invoked in several variations, and have a number of optional settings. Here we shall use their basic forms with default settings; for further details, consult (Wolfram, 2003) and the *Mathematica* help system. We shall also refer to several closely related articles and presentations. For simplicity, only minimization problems are considered: several functions also have a maximization equivalent, with identical solver functionality.

In the illustrative statements we shall use **`bold Courier`** fonts for displaying *Mathematica* input and `regular Courier` fonts for *Mathematica* output; however, in the explanatory text we retain the standard

(Times New Roman) fonts used in this article. All input/output statements and calculations presented in this work are directly imported from a corresponding *Mathematica* notebook.

2.1 LinearProgramming

The function LinearProgramming[c, A, b] finds a vector x that solves the LP problem stated as min $c^T x$ subject to $Ax \geq b$ and $x \geq 0$. Here c and x are (real) n-vectors; b is an m-vector and A is an m-row, n-column matrix. We will not discuss the ConstrainedMin (and ConstrainedMax) functions since these are also LP solvers, and both became obsolete since the release of *Mathematica* version 5.

A simple example of using LinearProgramming is shown below. Let us remark that in *Mathematica* vectors are denoted by lists: each component of a list is followed by a comma, and the entire list is enclosed by curly braces {}. The next three lines describe the model data (semicolon is used to suppress *Mathematica* output that in this case would simply echo the input lines shown):

```
c={1,2,1,1,3};
A={{2,-3,3,5,4},{-1,2,1,-4,-2},{2,2,2,1,1}};
b={3,8,12};
```

The solution is then simply obtained by entering the statement

```
xopt=LinearProgramming[c,A,b]
{0,2,4,0,0}
```

The result (i.e., the listed components of *xopt*) is shown in the row immediately following the *Mathematica* input statement. The solution is verified and the optimum value obtained by the following statements (the symbol **.** denotes the matrix-vector and vector-vector (dot) products):

```
A.xopt
{6,8,12}

c.xopt
8
```

The solution time for this ‘mini-problem’ is less than 0.001 seconds. *Mathematica* timings are usually displayed in one-thousandth of a second precision. All illustrative timing information in this article is measured using a Pentium 4 1.6 GHz processor based desktop machine that runs under Windows XP Professional; we are using *Mathematica* version 5.0.

Let us note here that recent LP related development includes the *Mathematica* implementation of the LAPACK package that has been used

worldwide to solve the most common tasks in numerical linear algebra (Leyk, 2003). Another notable development is discussed by Hu (2003): a new interior point algorithm option has been added to LinearProgramming that is now capable of solving large-scale linear optimization problems with hundreds of thousands of variables and equations.

2.2 FindMinimum

The function FindMinimum locally solves unconstrained nonlinear optimization problems, optionally using various methods that include conjugate gradient and BFGS quasi-Newton search strategies. As a simple illustration, we shall demonstrate its application in the form FindMinimum[f, {{x, x0},{y, y0}}] that uses the initial solution estimate {x0, y0} in solving the two-variable problem min $f(x,y)$. The multiplication symbol * is used below for clarity: it could be replaced by a space between the multiplier constant and the variable.

```
FindMinimum[Sin[x^2-x]+3*y^2, {{x,3},{y,1}}]
{-1., {x → 2.72764, y → -2.91001 × 10^-11}}
```

In the result received, -1 is the objective function value, and → denotes a symbol-to-value assignment. FindMinimum is a local search method: hence, this could be – in fact, is – only one of the local or global solutions (most likely, the one closest to the starting point). This point is illustrated by

```
FindMinimum[Sin[x^2-x]+3*y^2, {{x,13},{y,11}}]
{-1., {x → 11.6509, y → -1.49268 × 10^-9}}
```

2.3 NMinimize

The *Mathematica* function NMinimize[{f, cons},{x, y,…}] attempts to find the global minimum of *f*, subject to the listed constraints *cons*. The following simple example illustrates its application; notice the double equality sign == that denotes a strict equality constraint:

```
NMinimize[{(x1^2-x2)^2,
x1-x1*x2==0,  x1≥-10,  x1≤20,  x2≥-15,  x2≤10},
{x1,x2}]
{2.46519 × 10^-30, {x1 → 1., x2 → 1.}}
```

We will use NMinimize later on in some illustrative comparisons.

3. MODELING AND OPTIMIZATION PACKAGES

There is a range of application packages offered by Wolfram Research and by independent developers with apparent O.R. relevance. A brief review of these is provided below, for simplicity in alphabetical order. We will not mention or display the (quite possibly changing) version numbers, when discussing the packages: for further details see the related references and visit the website of Wolfram Research.

All packages discussed can be seamlessly integrated into *Mathematica*, when properly installed: in particular, their documentation can be directly invoked from *Mathematica*'s help system. Since all packages present detailed application examples, these can be directly used and customized to create new model development and optimization projects.

Needless to say, we do not intend to specifically endorse any of these applications, and – in lack of access to all listed packages – we rely partly on the product descriptions provided by Wolfram Research and the developers. Packages will be referred to using *italics* fonts.

Advanced Numerical Methods expands the functionality of the *Control System Professional* package with an extensive collection of numerical algorithms. These algorithms solve a wide class of control and linear algebra problems.

Combinatorica extends *Mathematica*‘s capabilities by over 450 new functions: these serve to construct graphs and other combinatorial objects, and to display them. The detailed guide to *Combinatorica* is Pemmaraju and Skiena (2003) that can also be used as a course textbook.

Control System Professional Suite is an extensible framework of integrated *Mathematica* application packages for handling common, interdisciplinary control problems that arise in engineering, as well as in chemistry, biology, economics and financial studies.

Database Access Kit brings *Mathematica*'s data analysis and management tools to large data sets. These capabilities can be interfaced with relational databases (including Oracle, Microsoft Access, SQLServer, and DB2) and to a number of flat-file databases (like Excel or dBase files).

DiffEqs is a collection of individual packages that accompanies the textbook by Hollis (2003): the book presents an introduction to *Mathematica*, and to differential equations.

Experimental Data Analyst integrates a set of programs that help to analyze experimental data, from error analysis and data fitting capabilities to data visualization and transformation. A collection of examples based on real experimental data is included.

Fuzzy Logic provides a set of tools for creating, modifying, and visualizing fuzzy sets and fuzzy logic-based systems. It also includes practical examples that introduce the basic concepts and demonstrate the numerical solution of various system design problems.

Global Optimization offers a collection of functions for constrained and unconstrained nonlinear optimization, as well as several tools of interest for statistical studies.

Industrial Optimization is designed to solve a range of O.R. models, by providing algorithms for linear, pure and mixed integer linear, and convex optimization, as well as some heuristic techniques such as genetic programming.

Mathematica Link for Excel provides Excel users with a seamless connection to *Mathematica*: one can directly activate a range of advanced *Mathematica* calculations and functions from the calling spreadsheet.

MathOptimizer and *MathOptimizer Professional* are our own application packages (Pintér, 2002b; Pintér and Kampas, 2003): these will be discussed in more details later on.

ModelMaker serves to build and analyze finite element (FE) models. The package permits building parametric models, where the FE database contains both numeric data and symbolic *Mathematica* expressions which can be used to morph the model geometry.

Neural Networks provides tools to define, train, visualize, and validate neural network models. It supports a set of network structures; it also implements training (unconstrained local optimization) algorithms.

Operations Research offers tools for solving linear optimization, quadratic programming, shortest path, and combinatorial optimization problems, including both exact and heuristic approaches.

Optimization Toolbox contains programs that accompany Bhatti's well-written textbook (2000), targeted primarily to an undergraduate and graduate student (and instructor) readership. Optimization theory is presented in an informal style; pedagogical *Mathematica* algorithms are presented and illustrated by examples.

Parallel Computing Toolkit brings parallel computation tools to a computer network, or to multiprocessor machines. It implements parallel programming primitives and includes high-level commands for the parallel execution of operations such as animation, plotting, and matrix manipulation.

VisualDSolve has been developed along with the textbook by Schwalbe and Wagon (1996) that serves as its reference manual. The book covers many of the topics in a first course in ordinary differential equations, and provides a wide variety of tools for visualizing solutions.

4. *MathOptimizer*

4.1 Introduction and Usage

MathOptimizer (Pintér, 2002b) is a native *Mathematica* software package that serves to solve general – global or local – nonlinear optimization models stated in the form

$$
\begin{array}{lll}
(1)\ \min f(x) & & f\colon D_0 \to \boldsymbol{R}^1 \\
\quad g(x)=0 & & g\colon D_0 \to \boldsymbol{R}^{m1} \\
\quad h(x)\le 0 & & h\colon D_0 \to \boldsymbol{R}^{m2} \\
\quad D_0 := \{x\colon xl \le x \le xu\} & & x, xl, xu \in \boldsymbol{R}^n
\end{array}
$$

It is assumed that all functions f, g, h are at least continuous, and that xl, xu are finite (known) real n-vectors. All bound, equality and inequality constraints are interpreted component-wise. Notice that the equality and inequality constraints are treated separately: their number is denoted by $m1$ and $m2$, respectively.

In addition to *MathOptimizer*'s built-in local solver methodology, a special emphasis is placed on finding the global solution of models that may have a number of local solutions. Fairly comprehensive reviews of global optimization are presented e.g., in the *Handbooks* edited by Horst and Pardalos (1995), Pardalos and Romeijn (2002); see also the topical website of Neumaier (2004).

MathOptimizer consists of two core solver packages and a solver integrator package. One of these solver components is called MS, abbreviating MultiStart (global search). MS serves for the – as a rule, approximate – global optimization of an exact penalty function that aggregates f, g, and h in the given n-dimensional interval range. MS uses an adaptive stochastic search method, combined with a statistical bounding procedure. The second component package – called CNLP, abbreviating Constrained NonLinear Programming (for local search) – implements a Lagrangian approach that is aimed at finding a (global or local) solution that satisfies the Karush-Kuhn-Tucker optimality conditions. (Note that, theoretically, this component requires smooth problem structure.) CNLP is used for 'precise' local optimization, based on a given initial solution: the latter is either produced by the global search phase, or it can be directly provided by the user. The solver integrator package, called Optimize, supports the individual or combined use of the two solver packages. It is planned to add further solver components to *MathOptimizer*: the presence of the integrator package directly supports this objective.

The *MathOptimizer* User Guide is a *Mathematica* notebook (currently consisting of over 70 printed pages) that can be directly invoked through *Mathematica*'s online help system. The manual presents installation and technical notes, provides concise mathematical background information and

modeling tips, and discusses a number of test problems as well as several more advanced applications.

MathOptimizer is invoked by the following *Mathematica* statement. Observe the notation used to identify the entire package and the integrator package component: the latter then indirectly activates both MS and CNLP.

```
Needs["MathOptimizer`Optimize`"];
```

The following *Mathematica* code illustrates the definition of a small non-convex optimization model that is made up by decision variables (denoted below as vars); lower/upper bounds and nominal (initial) values of the variables (varlb, varub, and varnom); objective function to minimize (objf); and the separate lists of equality constraints (eqs) and inequality constraints (ineqs, by assumption, are stated in ≤0 form).

```
vars={x1,x2};
varlb={-10,-15};
varub={20,10};
varnom={8,-14};
objf=10*(x1^2-x2)^2+(x1+3*x2-4)^2;
eqs={x1^4-x1*x2^3};
ineqs={3*x1+4*x2^2-8};
```

The next statement calls *MathOptimizer* to solve the model:

```
Optimize[objf, eqs, ineqs, vars, varnom, varlb,
varub]
{{1., 1.}, 6.23774×10^-21, {-3.94744×10^-11},
 {-1.}, {3.94744×10^-11, 2.73735×10^-9, 0.}}
```

The result shows the composite list of the following elements: the list of global solution components (x1=x2=1), the optimum value (a close numerical approximation to the theoretical value 0), as well as the lists of constraint function values at the solution, and finally the list of violation levels with respect to feasibility, the Kuhn-Tucker equation (defined by the gradient of the Lagrangian), and the complementary slackness condition at the solution found. The *MathOptimizer* runtime is less than 0.5 seconds.

Note that it is very easy to make changes to the model, and then to immediately repeat the solution procedure. For example, we can replace the constraints by defining (over-writing)

```
eqs={x1^4-Sin[1-x1*x2^3]-1};
ineqs={3*x1-4*x2^2+1};
```

After evaluating these statements – on MS Windows machines, by using the Shift-Enter key combination while pointing anywhere in the

Mathematica cell that includes the above input (so that they can be evaluated in a single move) – we can run *MathOptimizer* again. Observe passing by that the optimum value should be the same, except numerical rounding errors, since the previously found global solution {1, 1} meets also the new constraints.

```
Optimize[objf, eqs, ineqs, vars, varnom, varlb,
varub]
{{1., 1.}, 5.68314×10^-19, {8.07199×10^-11},
 {-1.07488×10^-9}, {8.07199×10^-11, 9.53077×10^-9, 0.}}
```

The numerical solution received is essentially the same as the one found above. For comparison, now we attempt to solve this model by using the built-in function NMinimize (in default mode, similarly to *MathOptimizer*). The NMinimize formulation for the model is slightly different:

```
NMinimize[objf,
x1^4-Sin[1-x1*x2^3}-1==0, 3*x1-4*x2^2+1≤0,
x1≥-10, x1≤20, x2≥-15, x2≤10}, {x1,x2}]
{2532.29,{x1→1.1892, x2→-10.354}}
```

The solution found by NMinimize is obviously sub-optimal. Of course, this finding is not sufficient per se to draw far-reaching conclusions. However, it certainly shows that the solution of nonlinear models can be tricky, even in (very) low dimensions.

4.2 Applications

In addition to a number of relatively simple numerical test examples, the *MathOptimizer* User Guide discusses illustrative applications from the following areas: chemical equilibrium modeling, industrial design, acoustic engineering design, and two numerical mathematics challenges (Problems 4 and 9 from Trefethen (2002)). In solving some of these – specifically, the sonar transducer model formulated by Purcell and the numerical integration problem of Trefethen – it is essential that *MathOptimizer* can handle arbitrary computable (preferably also continuous) *Mathematica* functions. This feature makes it suitable to handle 'black box' models defined by functions that are evalulated by complex, numerically intensive procedures. Pintér and Purcell (2003) discuss the sonar transducer design problem: its solution requires a combination of the *ModelMaker* (Purcell, Dai, and Xue, 2001) and *MathOptimizer* packages.

To mention other areas of application, Kampas and Pintér (2002) solve configuration analysis and design models using *MathOptimizer*: such problems arise e.g. in applied mathematics, statistics, physics, chemistry, and robotics. Pintér (2003c) discusses nonlinear model calibration: the

illustrative numerical results demonstrate that *MathOptimizer* produces superior results to local search based model fitting. The article then reviews several case studies in which global optimization has been applied to model calibration problems related to water quality, environmental engineering, time series analysis and photoelectron spectroscopy applications.

5. *MathOptimizer Professional*

5.1 Introduction and Usage

MathOptimizer Professional (Pintér and Kampas, 2003) is another *Mathematica* model development and nonlinear optimization package: however, it is based on an entirely different approach from the native *Mathematica* solver systems reviewed and discussed above. *MathOptimizer Professional* solves globally or locally nonlinear optimization models stated in the following general form (notice that the *m*-vector function *g* below now includes both equality and equality constraints):

$$
\begin{array}{lll}
(2) & \min f(x) & f\colon D_0 \to \boldsymbol{R}^1 \\
 & g(x) \le 0 & g\colon D_0 \to \boldsymbol{R}^m \\
 & D_0 := \{x\colon xl \le x \le xu\} & x, xl, xu \in \boldsymbol{R}^n
\end{array}
$$

The core of the package is the LGO external solver system that is activated and then used via *MathLink*, a general-purpose interface that supports communication between *Mathematica* and external programs. LGO – originally abbreviating the Lipschitz (continuous) Global Optimizer – can handle general (continuous) nonlinear optimization models, using a suite of global and local search algorithms. The currently implemented LGO algorithm options include branch-and-bound (BB), global adaptive random search (single-start, GARS) and multi-start (MS) based search strategies, as well as a (local) generalized reduced gradient (GRG) method. Note that in the global search phase the model functions are aggregated applying an exact penalty function; in the local search phase – that either automatically follows one of the global search modes or is used as a 'local search only' option – all constraint functions are treated individually.

The global search methods are, in theory, globally convergent (deterministically, or with probability 1, at least for box-constrained global optimization models). The actual code implementations are numerical approximations of the underlying theory. Due to the usage of an aggregated merit function, the automatic 'switching point' from global to local search, and other parameter settings, there are heuristic elements in LGO (similarly to most – if not all – numerical optimization methods). The optional choice of global methods often helps in solving difficult models, since BB, GARS,

and MS apply different search strategies. The parameterization of these component algorithms (e.g., intensified global search) can also help to solve difficult models, although the internally set default search effort typically produces a close numerical approximation of the global solution. The latter statement has been verified by solving some difficult global optimization problems in which the solution is reproducible and publicly available: some examples will be mentioned later on.

Note also that all LGO search algorithms are derivative-free: specifically, in the local search phase central differences are used to approximate gradients. This choice reflects our objective to handle models with merely computable, continuous functions, including 'black box' systems.

LGO has been developed and maintained for well over a decade (as of 2004), and the software is discussed in details elsewhere: consult, e.g., Pintér (1996, 2001, 2002a, 2004a), or the peer review by Benson and Sun (2000). LGO is currently available for essentially arbitrary C and Fortran compiler platforms, with seamless links to Excel, GAMS, Maple, *Mathematica*, and TOMLAB (the latter provides a solver interface and a collection of solvers for optimization using MATLAB). The details of these implementation versions are described in the corresponding documentation: see Frontline Systems and Pintér Consulting Services (2001); Pintér (2003a); Pintér (2004b); Pintér, Holmström, Göran and Edvall (2004).

The computational study (Pintér, 2003b) reviews the performance of LGO in comparison to several state-of-art local nonlinear solvers linked to the GAMS platform. This evaluation has been done in a fully automated and reproducible manner using publicly available GAMS model libraries: hence, it can be considered as reasonably objective, even if the collection of models and other circumstances (solver options and parameters) always carry elements of arbitrariness and subjectivity. The numerical experiments described in this study show that global optimization tools are needed to solve nearly half of the GAMS models from the chosen library, even when – possibly quite useful – initial solution points are provided to the local solvers. (We conjecture that providing random starting points from a search box that contains the feasible region would demonstrate even more pronounced need for global scope search.)

MathOptimizer Professional combines the model development power of *Mathematica* with the LGO solver suite: this leads to enhanced nonlinear solver capabilities, and a performance (solution speed) that – especially on larger models – is comparable to compiler-based solver implementations.

The functionality of *MathOptimizer Professional* is summarized by the following steps (all steps are fully automatic, except the first one):

- model formulation in *Mathematica*
- translation of the *Mathematica* optimization model into C or Fortran code (LGO model function file)
- generation of LGO input parameter file

- compilation of the C or Fortran model code into object code or dynamic link library (dll): this step needs a suitable compiler
- call to the LGO solver engine: the latter is typically provided as object code or an executable program that is linked together with the model object or dll file
- model solution and report generation by LGO
- report of LGO results back to the calling *Mathematica* notebook.

Obviously, the approach outlined supports 'only' the solution of models defined by *Mathematica* functions that can be directly converted into C or Fortran program code. This, however, still allows the handling of a broad range of continuous nonlinear optimization models. A 'side-benefit' of using *MathOptimizer Professional* is that *Mathematica* models are automatically translated into C or Fortran format: this can be useful e.g., in generating new test models.

Following installation, the *MathOptimizer Professional* User Guide (Pintér and Kampas, 2003) can be directly invoked as part of *Mathematica*'s help system. The package is activated by the following statement

```
Needs["MathOptimizerPro`callLGO`"];
```

Upon executing this statement, on MS Windows machines a command window opens that serves to monitor the *MathLink* connection that support external system calls to/from LGO. In our case, this window will display the background compiler and linker operations.

The numerical solution of an optimization model now can be launched by a single *Mathematica* statement of the form callLGO[f, g, {x, xl, xn, xu}]. Here we use the notation corresponding to (2); in addition, *xn* is the nominal setting of *x* (used in the first model function evaluation and/or as a starting point of the 'local search only' LGO solver mode). The following call illustrates the basic *MathOptimizer Professional* functionality:

```
callLGO[x^2 + 3y^2, {x + Sin[y] ≥ 1}, {{x, -2, 0, 2}, {y, -2, 0, 2}}]
{0.753796,{x→0.757485,y→0.244957},0}
```

The result shows (again, in *Mathematica* list format) the optimum value found, the list of corresponding variable settings, and the maximal model function infeasibility at the solution: all values are numerical approximations, of course. Note that the function callLGO currently has 15 optionally set parameters: these are all documented and illustrated in the User Guide, but their discussion is outside of the scope of this paper. For further details, consult the manual or Pintér and Kampas (2004).

5.2 Applications

For over a decade, LGO has been applied in a variety of professional, as well as academic research and educational contexts. In recent years, LGO has been used to solve models in up to a few thousand variables and constraints. Some recent applications and case studies – including e.g., model fitting in econometrics and laboratory analysis, potential energy models in computational chemistry, laser design, cancer therapy planning, and non-uniform sphere packings – are discussed by Pintér (2001a, b, 2002a), Isenor, Pintér, and Cada (2003), Tervo et al. (2003), Kampas and Pintér (2004a), Pintér and Kampas (2004). Note additionally that some of the LGO software users in the financial industry, process industries, biotechnology, etc. develop other advanced (but confidential) applications. We expect essentially similar performance from the recently released *MathOptimizer Professional* that enables the solution of sizeable, sophisticated nonlinear models formulated in *Mathematica*. The role of communication overhead between *Mathematica* and the external solver suite becomes relatively less significant in solving larger models, in which the external LGO solver time dominates.

The *MathOptimizer Professional* User Guide (an approximately 150-page document when printed) describes several tens of test problems starting with simple LP problems, through convex and non-convex nonlinear models, to a number of fairly challenging optimization models originating from mathematics, physics, chemistry, engineering and economics. For illustration, we shall consider here a pair of transcendental equations:

```
eq1 = (x - Sin[2x + 3y] - Cos[3x - 5y])^2;
eq2 = (y - Sin[x - 2y] + Cos[x + 3y])^2;
```

We wish to find a solution in the region $-2 \le x \le 3$, $-2.5 \le y \le 1.5$, or to numerically verify that there is no solution in the region specified. The surface and contour plots of eq1+eq2 (this corresponds to the squared l_2-norm based error function) reveal the rather complex multi-extremality of the induced optimization model: see Figures 1 and 2.

Let us apply *MathOptimizer Professional* to solve this problem. First, we define the equations (eqs), the constraints (cons: note here that the relations eq1=eq2=0 can be expressed by using the *Mathematica* function Thread), and the variables with bounds and nominal values (varswithbounds). Then we call LGO.

```
eqs={eq1,eq2};
cons=Thread[eqs==0];
varswithbounds={{x,-2,1,3},{y,-2.5,1,1.5}};
callLGO[0,cons,varswithbounds]
{0, {x → -0.173363, y → -0.256098}, 1.44819 × 10^-9}
```

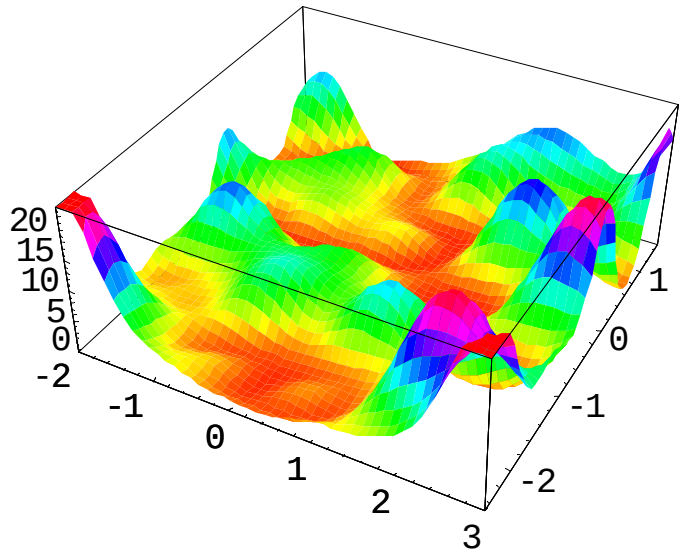

Figure 1. Surface plot of error function in solving a system of equations.

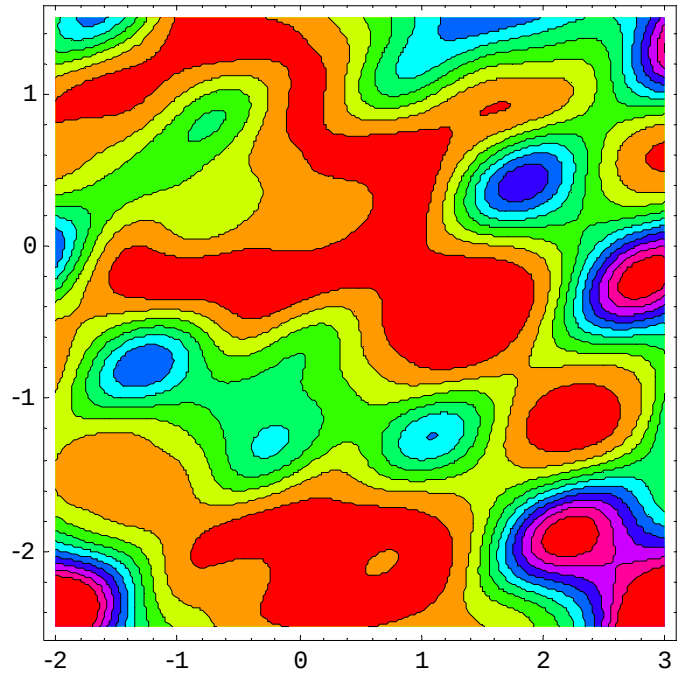

Figure 2. Contour plot of error function in solving a system of equations.

As the result shows, *MathOptimizer Professional* finds a numerical solution that is precise to about $1.45\ddot{y}10^{-9}$, when substituted into the equations. The external LGO runtime is 0.03 seconds. (In total, 7843 search steps – model function evaluations, including gradient estimates in the local search phase – are done in using the default MS+LS search mode with default parameterization; all results are exactly reproducible.) Note also that the User Guide addresses the issue of finding (possible) multiple solutions to systems of equations and inequalities.

As for another illustrative application, in (Kampas and Pintér, 2004a) we state and solve a challenging new model type: our objective is to find the 'best' non-overlapping arrangement of a set of given non-uniform size circles in an embedding circle. The best packing is defined here by a combination of two criteria: the size (radius) of the circumscribed circle, and the average pair-wise distance between the centers of the embedded circles. The relative weight of the two objective function components can be selected as a model-instance parameter.

Detailed numerical results are reported in (Kampas and Pintér, 2004a) for circles defined by the radii $r_i=i^{-0.5}$, $i=3,\ldots,N$, up to 40-circle configurations. For illustration, the configuration found for the case $N=20$ circles using *MathOptimizer Professional* is displayed below. In this example, equal consideration (weight) is given to minimizing the radius of the circumscribed circle and the average distance between the circle centers.

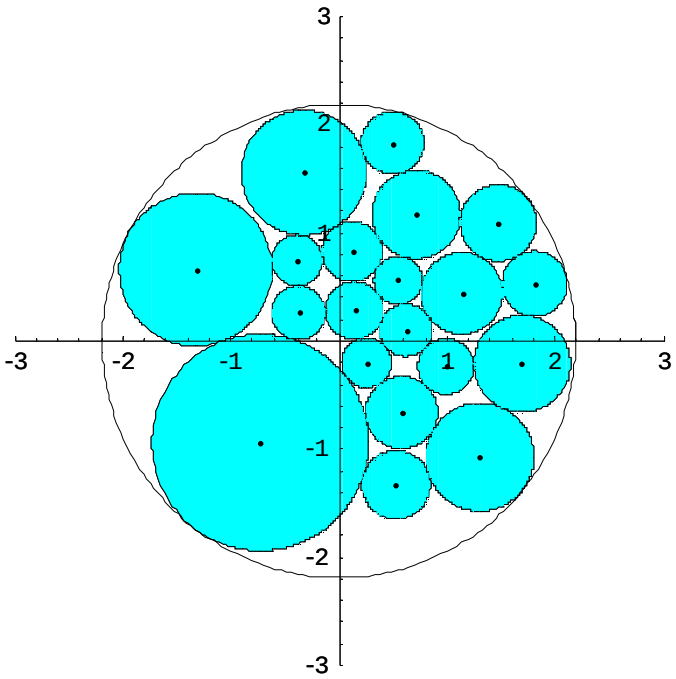

Figure 3. A non-uniform circle packing result for $N=20$ circles.

Let us remark in this context that in Kampas and Pintér (2002) we have attempted to solve instances of the circle packing problem applying the built-in *Mathematica* function NMinimize, but (in default mode) it could not find a solution of acceptable quality even for the case N=5. *MathOptimizer* worked better and found good quality solutions for small configurations (up to N=10), but – due to its native *Mathematica* solver functions – solution times are increasing far more rapidly than for *MathOptimizer Professional*. Again, this is just a numerical observation, as opposed to a conclusion: we plan to make a more systematic comparison of global solvers (available for use with *Mathematica*) in the near future, based on detailed numerical tests.

Let us also mention finally that both *MathOptimizer* and *MathOptimizer Professional* are included in a recent peer review of optimization capabilities using *Mathematica* (Cogan, 2003).

6. CONCLUDING REMARKS

This article discusses the potentials of *Mathematica* in Operations Research related modeling and optimization studies. Within this context, we review built-in *Mathematica* optimization functionality and provide an annotated list of relevant application packages. Next, we introduce the packages *MathOptimizer* and *MathOptimizer Professional*, and discuss their usage by solving a few illustrative (yet non-trivial) optimization problems. We think that integrated modeling and solver environments will have a significant role in O.R. modeling and optimization studies, in an increasing range of business, research, and educational contexts.

ACKNOWLEDGEMENTS

JDP wishes to acknowledge the support received from Wolfram Research over the years, in forms of a visiting scholarship, books, software, and professional advice. The *MathOptimizer* software development project has profited from advice and comments by Dr. Christopher J. Purcell (DRDC Atlantic Region), and it has been partially funded by DRDC (Contract No. W7707-01-0746/001/HAL), and by NRC IRAP (Project No. 362093). We also wish to thank Dr. Mark Sofroniou (Wolfram Research) for his kind permission to use – and to modify – the Format.m *Mathematica* package in our *MathOptimizer Professional* development work.

The comments received from an anonymous referee helped us to improve the content and style of the paper.

REFERENCES

Bahder, T.B. (1995) *Mathematica for Scientists and Engineers*. Addison-Wesley, Reading, MA.

Benson, H.P. and Sun, E. (2000) LGO – Versatile Tool for Global Optimization. *ORMS Today* 27 (5) 52-55. See also http://www.lionhrtpub.com/orms/orms-10-00/swr.html.

Bhatti, M.A. (2000) *Practical Optimization Methods With Mathematica Applications*. Springer, New York.

Frontline Systems and Pintér Consulting Services (2001) *Premium Solver Platform – LGO Global Solver Engine for Excel*. Published by Frontline Systems, Inc., Incline Village, NV. See also http://www.solver.com/xlslgoeng.htm.

Cogan, B. (2003) How to get the best out of optimization software. *Scientific Computing World* 71 (2003) 67-68. See http://www.scientific-computing.com/scwjulaug03review_optimisation.html.

Gass, R. (1998) *Mathematica for Scientists and Engineers: Using Mathematica to do Science*. Prentice Hall, Englewood Cliffs, NJ.

Hollis, S. (2003) *A Mathematica Companion for Differential Equations*. Prentice Hall, NJ.

Horst, R. and Pardalos, P.M., eds. (1995) *Handbook of Global Optimization*, *Vol. 1.* Kluwer Academic Publishers, Dordrecht.

Hu, Y. (2003) Solving large linear optimization problems. Lecture presented at the *2003 Mathematica Developer Conference*, Champaign, IL.

Isenor, G., Pintér, J.D., and Cada, M. (2003) A global optimization approach to laser design. *Optimization and Engineering* 4, 177-196.

Jacob, C. (2001) *Illustrating Evolutionary Computation with Mathematica*. Morgan Kaufmann Publishers, San Francisco, CA.

Kampas, F.J. and Pintér, J.D. (2002) Configuration analysis and design by using optimization tools in *Mathematica*. *The Mathematica Journal* (to appear).

Kampas, F.J. and Pintér, J.D. (2004a) Generalized circle packings: model formulations and numerical results. *Proceedings of the 2004 International Mathematica Symposium*, Banff, AB.

Kampas, F.J. and Pintér, J.D. (2004b) *Advanced Optimization: Scientific, Engineering, and Economic Applications with Mathematica Examples*. Elsevier, Amsterdam (to appear).

Leyk, Z. (2003) Fast linear algebra in *Mathematica*. Lecture presented at the *2003 Mathematica Developer Conference*, Champaign, IL.

Maeder, R.E. (2000) *Computer Science with Mathematica*. Cambridge University Press, Cambridge, UK.

Mathematica Information Center (2004) http://library.wolfram.com/infocenter/.

Neumaier, A. (2004) Global Optimization. http://www.mat.univie.ac.at/~neum/ glopt.html.

Pardalos, P.M. and Romeijn, H.E., eds. (2002) *Handbook of Global Optimization, Vol. 2.* Kluwer Academic Publishers, Dordrecht.

Pemmaraju, S. and Skiena, S. (2003) *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press, Cambridge, UK.

Pintér, J.D. (1996) *Global Optimization in Action.* Kluwer Academic Publishers, Dordrecht.

Pintér, J.D. (2001a) *Computational Global Optimization in Nonlinear Systems: An Interactive Tutorial.* Lionheart Publishing, Atlanta, GA.

Pintér, J.D. (2001b) Globally optimized spherical point arrangements: Model variants and illustrative results. *Annals of Operations Research* 104, 213-230.

Pintér, J.D. (2002a) Global optimization: software, test problems, and applications; Chapter 15 (pp. 515-569) in: Pardalos and Romeijn, eds. *Handbook of Global Optimization, Vol. 2.*

Pintér, J.D. (2002b) *MathOptimizer – An Advanced Modeling and Optimization System for Mathematica Users. User Guide.* Published and distributed by Pintér Consulting Services, Inc., Halifax, NS, Canada.

Pintér, J.D. (2003a) *GAMS/LGO User Guide.* Published and distributed by the GAMS Development Corporation, Washington, DC. See http://www.gams.com/solvers/lgo.pdf.

Pintér, J.D. (2003b) GAMS/LGO nonlinear solver suite: key features, usage, and numerical performance. (Submitted for publication.) Available for download at http://www.gams.com/solvers/solvers.htm#LGO.

Pintér, J.D. (2003c) Globally optimized calibration of nonlinear models: techniques, software, and applications. *Optimization Methods and Software* 18, 335-355.

Pintér, J.D. (2004a) *LGO – An Integrated Model Development and Solver Environment for Continuous Global Optimization. User Guide.* (Current edition.) Published and distributed by Pintér Consulting Services, Inc., Halifax, NS, Canada.

Pintér (2004b) *The Maple Global Optimization Toolbox*. Published and distributed by Maplesoft, Inc., Waterloo, ON. See http://www.maplesoft.com/products/toolboxes/globaloptimization/index.shtml.

Pintér, J.D. and Kampas, F.J. (2003) *MathOptimizer Professional – An Advanced Modeling and Optimization System for Mathematica Users with an External Solver Link*. *User Guide.* Published and distributed by Pintér Consulting Services, Inc., Halifax, NS, Canada.

Pintér, J.D. and Kampas, F.J. (2004) Global optimization in *Mathematica* with MathOptimizer Professional. (Submitted for publication.)

Pintér, J.D. and Purcell, C.J. (2003) Optimization of finite element models with *MathOptimizer* and *ModelMaker*. Lecture presented at the *2003 Mathematica Developer Conference*, Champaign, IL.

Pintér, J.D., Holmström, K., Göran, A.O. and Edvall, M.M. (2004) *TOMLAB /LGO User Guide.* Published and distributed by TOMLAB Optimization AB, Västerås, Sweden and Arcata, CA. See http://tomlab.biz/docs/TOMLAB_LGO.pdf.

Purcell, C.J., Dai, N.M. and Xue, L. (2001) Modelling, analysis & prototyping for rapid manufacturing. Lecture presented at the *2001 Mathematica Developer Conference*, Champaign, IL.

Schwalbe, D. and Wagon, S. (1996) *VisualDSolve: Visualizing Differential Equations with Mathematica.* Springer, New York.

Sodhi, M.S. (2003) *Mathematica 5*. *ORMS Today* 30 (6), 44-47.

Tervo, J., Kolmonen, P., Lyyra-Laitinen, T., Pintér, J.D., and Lahtinen, T. (2003) An optimization-based approach to the multiple static delivery technique in radiation therapy. *Annals of Operations Research* 119, 205-227.

Trefethen, N.L. (2002) A Hundred-dollar, Hundred-digit Challenge. SIAM News 35 (1), p. 3. See also http://www.siam.org/siamnews/01-02/challenge.pdf.

Wolfram, S. (2003) *The Mathematica Book.* (5th Edition.) Wolfram Media, Inc., Champaign, IL.

Wolfram Research (2004) http://www.wolfram.com/.