Ladislav Lukšan; Jan Vlček

Software system for universal functional optimization

In: Jan Chleboun and Petr Přikryl and Karel Segeth (eds.): Programs and Algorithms of Numerical Mathematics, Proceedings of Seminar. Dolní Maxov, June 6-11, 2004. Institute of Mathematics AS CR, Prague, 2004.
pp. 155–161.

Persistent URL: http://dml.cz/dmlcz/702789

SOFTWARE SYSTEM FOR UNIVERSAL FUNCTIONAL OPTIMIZATION*

Ladislav Lukšan, Jan Vlček

UFO is an interactive system for universal functional optimization that serves for solving both dense medium-size and sparse large-scale optimization problems. The UFO system can be used for formulation and solution of particular optimization problems, for preparation of specialized optimization routines and for designing and testing new optimization methods.

UFO is an extensive software system for solving a broad class of optimization problems. The solution of an optimization problem is processed in three phases. In the first phase the optimization problem is specified and an optimization method is selected. This can be made in three different ways:

- Full dialogue mode. The problem specification and the method selection are realized by using a conversation between the user and the UFO system.
- Batch mode. The problem specification and the method selection are realized by using the UFO control language. An input file written in the UFO control language has to be prepared.
- Combined mode. Only a part of the specification is written in the input file. The rest of the specification is obtained as in the dialogue mode. This possibility is usually the best one since the problem functions can be defined beforehand by using a convenient text editor.

The first phase is realized by using the UFO preprocessor. This preprocessor is written in the Fortran 77 language and its output is a Fortran 77 control program. In the second phase, the control program is translated by using a Fortran 77 compiler and a final program is linked by using library modules. In the third phase, the final program is executed and thus results, which can be viewed by using extensive output means, are obtained. A detailed description of the UFO system is contained in research report [1], which can be downloaded from the internet page `http://www.cs.cas.cz/~luksan/ufo.html`.

UFO is an open modular system. Individual optimization methods are realized by the connection of corresponding universal modules. In this way, we can generate a large number of modifications of a given method and find the most suitable

*This work was supported by the Grant Agency of the Czech Academy of Sciences, project code IAA1030405, and with the subvention from Ministry of Education of the Czech Republic, project code MSM 242200002.

implementation. In the case of unconstrained optimization or optimization with linear constraints, we can change the following modules ((O)–output, (C)–constraint handling, (M)–method realization, (F)–objective function evaluation):

- (O) Input and output subroutines (text or graphic).
- (C) Subroutines for the determination of a feasible point for various types of linear constraints and various representations of linear manifolds.
- (C) Subroutines for transformation of gradients, Hessian matrices and direction vectors.
- (C) Subroutines for adding or deleting active constraints.
- (M) Subroutines for direction determination (line-search, trust-region) that use various direct solvers for different matrix representations and various iterative methods with different preconditioners.
- (M) Subroutines for step-size selection (line-search, trust-region).
- (M) Subroutines for updating approximations of the Hessian matrix or its inverse (various variable metric updates for different matrix representations).
- (F) Subroutines that evaluate values and gradients of the objective function.

The UFO system can be used for solving a broad class of continuous optimization problems and also for some related problems. Essentially, it can be used for unconstrained optimization, optimization with linear and nonlinear constraints, nonsmooth optimization, global optimization and also for solving systems of nonlinear algebraic or ordinary differential equations. At the same time, the objective function can be quite general or it can have a special form that makes possible to use a special optimization method. Special objective function can be linear, quadratic, the sum of squares (or powers), the sum of absolute values or the maximum of values (minimax). It is also possible to use various integral criteria containing solution to ordinary differential equations.

To demonstrate the use of the UFO system, we consider the minimization of the objective function

$$F(x) = x_1 x_3$$

on the set given by box constraints $x_1 \geq 0$, $x_3 \geq 0$, $x_5 \geq 0$, $x_7 \geq 0$ and general nonlinear constraints

$$\begin{aligned}
(x_4 - x_6)^2 + (x_5 - x_7)^2 &\geq 4, \\
(x_3 x_4 - x_2 x_5)/\sqrt{x_2^2 + x_3^2} &\geq 1, \\
(x_3 x_6 - x_2 x_7)/\sqrt{x_2^2 + x_3^2} &\geq 1, \\
(x_1 x_3 + (x_2 - x_1) x_5 - x_3 x_4)/\sqrt{(x_2 - x_1)^2 + x_3^2} &\geq 1, \\
(x_1 x_3 + (x_2 - x_1) x_7 - x_3 x_6)/\sqrt{(x_2 - x_1)^2 + x_3^2} &\geq 1.
\end{aligned}$$

The starting point is $x_1 = 3.0$, $x_2 = 0.0$, $x_3 = 2.0$, $x_4 = -1.5$, $x_5 = 1.5$, $x_6 = 5.0$, $x_7 = 0.0$. The batch input file for the UFO system has the form:

```
$FLOAT W
$SET(INPUT)
  X(1)= 3.0D0 ; XL(1)= 0.0D0 ; IX(1)= 1 ; X(2)= 0.0D0
  X(3)= 2.0D0 ; XL(3)= 0.0D0 ; IX(3)= 1 ; X(4)=-1.5D0
  X(5)= 1.5D0 ; XL(5)= 1.0D0 ; IX(5)= 1 ; X(6)= 5.0D0
  X(7)= 0.0D0 ; XL(7)= 1.0D0 ; IX(7)= 1
  CL(1)=4.0D0 ; IC(1)= 1 ; CL(2)=1.0D0 ; IC(2)= 1
  CL(3)=1.0D0 ; IC(3)= 1 ; CL(4)=1.0D0 ; IC(4)= 1
  CL(5)=1.0D0 ; IC(5)= 1
$ENDSET
$SET(FMODELF)
  FF=X(1)*X(3)
$ENDSET
$SET(FMODELC)
  IF (KC.EQ.1) THEN
    FC=(X(4)-X(6))**2+(X(5)-X(7))**2
  ELSE IF (KC.EQ.2) THEN
    W=SQRT(X(2)**2+X(3)**2) ; FC=(X(3)*X(4)-X(2)*X(5))/W
  ELSE IF (KC.EQ.3) THEN
    W=SQRT(X(2)**2+X(3)**2) ; FC=(X(3)*X(6)-X(2)*X(7))/W
  ELSE IF (KC.EQ.4) THEN
    W=SQRT((X(2)-X(1))**2+X(3)**2)
    FC=(X(1)*X(3)+(X(2)-X(1))*X(5)-X(3)*X(4))/W
  ELSE IF (KC.EQ.5) THEN
    W=SQRT((X(2)-X(1))**2+X(3)**2)
    FC=(X(1)*X(3)+(X(2)-X(1))*X(7)-X(3)*X(6))/W
  ENDIF
$ENDSET
$NF=7 ; $NX=7 ; $NC=5
$BATCH
$STANDARD
```

The minimum function value is $F = 23.3137$. It has been reached at point $x_1 = 4.828$, $x_2 = 0.000$, $x_3 = 4.828$, $x_4 = 1.000$, $x_5 = 2.414$, $x_6 = 2.414$, $x_7 = 1.000$.

The UFO system contains optimization methods that can be divided into the following classes:

- Heuristic methods for small-size problems. This class contains the pattern search method of Hooke and Jeeves and the simplex method of Nelder and Mead.

- Conjugate direction methods that use no matrices. This class contains various modifications of the conjugate gradient methods together with the limited-memory variable metric method of Nocedal based on the Strang recursions.
- Variable metric methods that use rank-one and rank-two updates for obtaining a positive definite approximation of the inverse Hessian matrix.
- Limited-memory variable metric methods of Byrd, Nocedal and Schnabel based on a compact representation of variable metric updates that are applied to small-size matrices.
- Limited-memory variable metric methods based on product-form updates that are applied to a positive semidefinite approximation of the shifted inverse Hessian matrix.
- Limited-memory variable metric methods of Gill and Leonard based on reduced updates that are applied to an approximation of the reduced Hessian matrix.
- Various modifications of the Newton method that use Hessian matrices computed either analytically or numerically.
- Truncated version of the Newton method where the iterative conjugate gradient method is applied to the linear system and directional derivatives are approximated by using gradient differences. This method uses no matrices.
- Various modifications of the Gauss-Newton method for nonlinear least squares, where the Hessian matrix is approximated by the normal-equation matrix improved by the variable metric updates. Alternatively, these methods are realized in the form utilizing linear least squares subproblems using the Jacobian matrix or its approximation.
- Quasi-Newton methods for nonlinear least squares and nonlinear equations that use rank-one updates for obtaining an approximation of the inverse Jacobian matrix.
- Limited-memory quasi-Newton methods for nonlinear least squares and nonlinear equations based on a compact representation of variable metric updates that are applied to small-size matrices.
- Modifications of the Brent method for nonlinear equations.
- Proximal bundle methods for nonsmooth optimization.
- Bundle-Newton methods for nonsmooth optimization. These methods use second-order information and are intended for small-size problems.
- Variable metric methods for nonsmooth optimization that use bundles with three elements at most.
- Variable metric methods for large-scale partially separable nonsmooth functions.
- Simplex-type methods for linear and quadratic programming problems.
- Interior-point methods for linear and quadratic programming problems.
- Recursive quadratic programming methods for discrete nonlinear minimax problems.

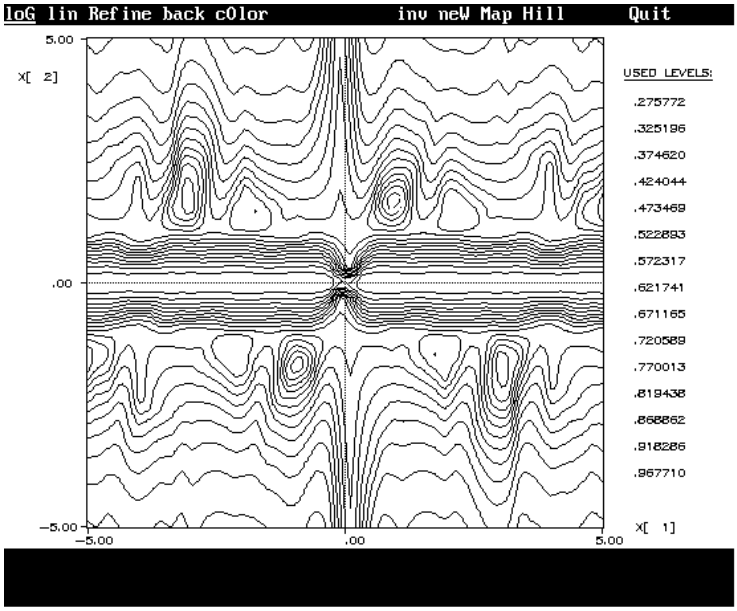
- Recursive quadratic programming methods for general nonlinear programming problems.
- Recursive quadratic programming methods for large-scale problems with sparse equality constraints.
- Interior-point methods for large-scale sparse nonlinear programming problems.
- Nonsmooth-equation methods for large-scale sparse nonlinear programming problems.
- Random search methods for global optimization.
- Continuation methods for global optimization. This class contains tunneling function methods and filled function methods.
- Clustering methods for global optimization. This class contains a density clustering method and a single linkage clustering method.
- Multi-level methods for global optimization. This class contains a multi-level single linkage clustering method and a multi-level mode analysis method.

These optimization methods can be realized with various strategies for step-size selection. There are line-search methods, general trust-region methods, special trust-region methods for nonlinear least squares, Marquardt-type methods for nonlinear least squares and filter-type methods for nonlinear programming including Fletcher-Leyffer filters, barrier filters and Markov filters. Moreover, various direct solvers for different matrix representations and various iterative methods with different preconditioners can be used for direction determination. The UFO system also contains many efficient methods for solving related subproblems, for example methods for solving systems of ordinary differential equations.

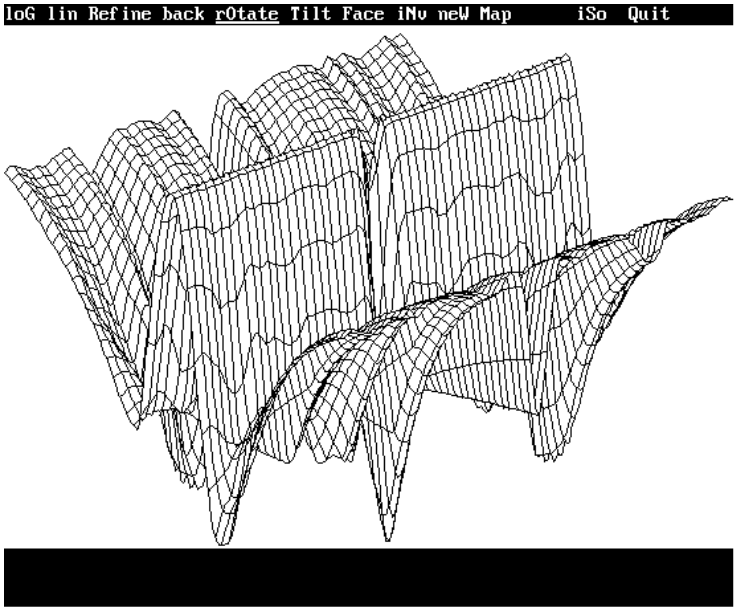
Besides numerical methods, the UFO system contains many input and output tools. There is a control language for batch processing, text dialogue for unix systems, graphic dialogue for Microsoft windows, text screen output for unix systems, graphic screen output for Microsoft windows and various types of output text files. Furhermore, many collections of problems for testing optimization methods and tools for testing external subroutines describing model functions are included. The UFO preprocessor contains also subroutines for automatic differentiation and the interface to the CUTE testing environment.

Finally, we introduce one practical application. We consider the optimization of a three-section cascaded transmission-line 10:1 transformer over a 100-percent bandwidth. Optimality criterion is to minimize the reflection coefficient over 11 frequency points in the band 0.5–1.5 GHz. The design parameters are the characteristic impedances and the lengths of individual transformer sections normalized with respect to quarter-wave length at the center of frequency. The nonsmooth objective function has the form

$$\begin{aligned} f(x) &= \max_{1\le i\le 11} f_i(x), \\ f_i(x) &= \left| 1 - 2\frac{v_1(x,\omega_i)}{w_1(x,\omega_i) + v_1(x,\omega_i)} \right|, \end{aligned}$$



Transformer network design: Isolines



Transformer network desogn : Surface

where $v_1(x,\omega_i)$ a $w_1(x,\omega_i)$ are complex numbers computed recursively in such a way that $v_4(x,\omega_i) = 1$, $w_4(x,\omega_i) = 10$ and

$$\begin{aligned} v_k(x,\omega_i) &= \cos(\vartheta_i x_{2k-1}) v_{k+1}(x,\omega_i) \\ &\quad + j \sin(\vartheta_i x_{2k-1}) \frac{1}{x_{2k}} w_{k+1}(x,\omega_i), \\ w_k(x,\omega_i) &= \cos(\vartheta_i x_{2k-1}) w_{k+1}(x,\omega_i) \\ &\quad + j \sin(\vartheta_i x_{2k-1}) x_{2k} v_{k+1}(x,\omega_i) \end{aligned}$$

for $k = 3, 2, 1$. Here $j = \sqrt{-1}$ is the imaginary unit, $\vartheta_i = (\pi/2)\omega_i$ and ω_i, $1 \leq i \leq 11$, are design frequencies.

References

[1] L. Lukšan, M. Tůma, M. Šiška, J. Vlček, N. Ramešová: *UFO 2002 – Interactive system for universal functional optimization.* Technical Report V-883. Institute of Computer Science, Academy of Sciences, Prague, ICS AS CR, 2002.