INFORMATICA, 1990, Vol.1, No.1, 071–088

DESIGN OF SOFTWARE FOR GLOBAL OPTIMIZATION

Audris MOCKUS and Linas MOCKUS

Institute of Mathematics and Cybernetics,
Lithuanian Academy of Sciences,
232600 Vilnius, Akademijos St.4, Lithuania

Abstract. In the paper the global optimization is described from the point of an interactive software design. The interactive software that implements numeric methods and other techniques to solve global optimization problems is presented. Some problems of such a software design are formulated and discussed.

Key words: global optimization, software environment.

1. Introduction. The optimization of a nonconvex continuous function involves different techniques. In the paper an approach to this problem from the point of the designer of interactive software is presented.

The "Global Minimum" software implements numeric methods and other techniques to solve the global optimization problems. "Global Minimum" will be used to illustrate the discussed problems in global optimization and software design.

At the beginning,the Bayesian approach to the optimization, the underlying idea of numeric methods used in "Global Minimum," is presented. Then,the implemented optimization methods are briefly described. Finally, the design of interactive software environment for optimization is considered. The stated design problems may be similar for other software environments.

2. Global optimization problems. Almost every engineering and design problem has a subproblem, requiring optimization. The functions to be examined usually have multiply extrema and the best of them must be chosen. Local optimization techniques are usually not sufficient to solve the problem.

Numeric optimization techniques never give exact results, the achieved answer is some approximation of the real minimum. Some techniques guarantee that deviation from the real minimum for all problems is no worse than the predefined. In "Global Minimum" the Bayesian approach is used.

The optimization methods implemented in "Global Minimum" are the result of a long research based on real problems and are available in portable FORTRAN (Mockus J.,1989; MINIMUM, 1984). Their portable implementation lack interactive features, so in "Global Minimum" they are redesigned and implemented in C on IBM PC.

2.1. Bayesian approach to optimization. The idea of the Bayesian approach is as follows: optimization techniques are optimal in the sense of mean deviation (usually optimality is understood as a minimization of the maximal deviation). Implementing this approach at each iteration one must minimize the expected deviation from the global maximum (or minimum) for a given class of functions with given a priori distribution of probabilities and with regard to the calculated function values (Mockus J.,1989).

It is desirable to make a procedure for calculation of the expected deviation (risk function) as simple as possible, but there are natural restrictions. For example, in the global optimization of continuous functions in order to minimize the expected deviation one must minimize an auxiliary function, which in the simplest case can't be reduced to the unimodal

function if we want to provide the convergence to global minimum for every continuous function.

2.2. About implemented methods. The implemented methods can be divided into three categories: global, local and multicriterial. An objective function is the n-dimensional continuous function, but for some methods it also can be non-differentiable or with noise. Usually, the function is defined on the n-dimensional rectangle, but in some local methods the region is defined by arbitrary continuous functions. Here is some information about each method; its purpose, restrictions and accuracy.

The global Bayesian method by J.Mockus (1984b). It finds the global minimum of a continuous noise or deterministic function of n variables defined on the rectangle. In terms of efficiency this program becomes increasingly less successful as the dimension of the rectangle increases. The method is using a considerable amount of auxiliary calculations. As a result, the method can be efficiently used only when the objective function is "difficult" - takes more of CPU time than the auxiliary calculations to find the coordinates of the next point. It performs search in the whole area and can find the point in the neighborhood of global minimum. The method not always can fix the point of minimum with sufficient accuracy, so some local methods should be used to carry out the local minimization. The method provides the minimum average deviation in accordance with a given statistical model (Mockus J.,1984b) and the convergence to global minimum for any continuous function. It means that if we solve many problems, the average error will be as small as possible. For some fixed samples it can happen to be great, if the iteration number is limited.

The global Bayesian method based on the extrapolation theory by Žilinskas (1986). It finds the global minimum of a

continuous deterministic function of n variables defined on the rectangle. Restrictions are like in the global Bayesian method by J.Mockus, except that if the function is differentiable, then the local search of a variable metrics type can be directly incorporated into the method. The method provides the minimal average deviation in accordance with the set of assumptions (Žilinskas,1986). All other accuracy considerations are similar to those in the global Bayesian method by J.Mockus. The method terminates after the fixed number of local minima is reached.

The global deterministic uniform search method (Sobolj, 1969). It finds the global minimum of a continuous deterministic function of n variables defined on the rectangle. The amount of arbitrary calculations is less than that of the preceding methods, so we can have more iterations. The method is more efficient for simpler functions. It asymptotically converges to the global minimum. The method also makes the statistical analysis of a function structure (Dzemyda,1983).

The global method of clustering by Tőrn (1978). It finds the global minimum of a continuous deterministic function of n variables defined on the rectangle. Restrictions are the same as in all preceding methods with the exception that this method uses smaller amount of auxiliary calculations, so it can be recommended to minimize "simpler" functions in comparison with all previous methods if the convergence is not necessary. The convergence to minimum is not provided, but usually the accuracy satisfies practical needs.

The global method of Monte-Carlo (uniform random search). It finds the global minimum of a continuous deterministic function of n variables defined on the rectangle. The restrictions are similar to those in all other global methods except for the amount of auxiliary calculations being minimal, so the method can be recommended to minimize very "simple" functions, when the convergence in probability is sufficient.

The method converges in probability to the global minimum of continuous functions. The average deviation is considerably greater, comparing to other global methods with the same number of function evaluations.

The global Bayesian coordinate line search method by Žilinskas (1986). It finds the global minimum of a continuous deterministic function of n variables defined on the rectangle. Restrictions are similar to those of global methods. The local search is included into the method. The method provides the minimal average deviation from the optimum under the assumption that the objective function can be regarded as a sample of Wiener process and converges to the minimum of continuous functions.

The local simplex method of Nelder and Mead (Himmelblau,1972). It finds the local minimum of a continuous deterministic nondifferentiable function of n variables with nonlinear constraints. Only the local minimum of a function with constraints can be found. The convergence to the minimum is not provided,but usually the accuracy satisfies practical needs if number of the iterations is large enough.

The local method of nonlinear programming by Shittkowski (1981). It finds the local minimum of a deterministic differentiable function of n variables with nonlinear constraints or defined on the rectangle. Only the local minimum of a function can be found. An arbitrary close approach to the minimum can be made depending on the parameter. In our system this parameter is set to satisfy practical needs.

The local method of stochastic approximation with Bayesian step size control by J.Mockus (1984a). It finds the local minimum of a unimodal function with noise of n variables defined on the rectangle. Only the optimum of an unimodal function is found. An arbitrary close approach to the minimum can be made with a probability one,when the number of iterations is large enough. The Bayesian step length provides

the minimal average deviation in accordance with a given statistical model (Mockus J.,1984b). The number of iterations should be increased sharply if one wishes to make the average error considerably less than the level of noise.

The method for search of pareto optimal set by L.Mockus. It finds the approximation of this set on the rectangle and the approximation improves as the number of iterations increases. Asymptotically, the method founds the set of pareto optimal points. It has two variations: uniform search with the use of LP-sequences (Sobolj,1969) or nonuniform search,using Bayesian techniques (Mockus J. and Mockus L.,1989). It was shown that the second variation is better in the case,when the answer is approximately known

3. Design and implementation of software environment. Solving of practical problems usually involves a construction of their mathematical model (problem definition) and its exploration. Sometimes exploration of the problem model requires to change the model itself (for example, if the model is found to be not adequate).

Most software tools (optimization methods, statistical analysis techniques, etc.) are suitable for the exploration of the already implemented model, and are not integrated into one environment with techniques for the construction of the model.

It is suggested that software tools for the problem definition integrated with tools for exploration must be the main property of the software environment. Implementation in terms of a specific problem might restrict the software usability for the problems in adjacent fields. This can be solved by implementing common mathematical concepts that underlie techniques of application domain. For optimization such common base might be a set of linear algebra objects. The definition of a model in terms of vectors, matrices, gradients will be more convenient than in terms of integers, reals, ar-

rays and loops as in a common programming language. Such common base can be used to define and solve different optimization problems (differential equations, statistic, etc.).

Implementing application domain requires its mapping to the programming language, operating system and computer system. This is a hard to formalize task because of the difference of concepts and structure of domain and range. Finding or constructing adequate structures in programming environment can greatly help the implementation.

3.1. Design problems. A software system must be an adequate model of application domain. A large number of implementation specific concepts can be time consuming for a user to learn and the properties of such concepts are hard to prove. A better way is to implement the basic mathematical concepts that underlie the definition and solving of a problem.

A software representation of mathematical concepts must comply the corresponding axioms. This task is inherently difficult because of a nonconstructive approach used in mathematics.

Software augments the functionality of application domain concepts. The representation on display devices and interaction with a user must be provided. Time additional enhancements are needed also because of the specific software tools used as the basis of the system and to enable the new possibilities present only in the software implementation of the application domain. For example, software implementation of a vector must define the creation of such an object, provide its representation (if any) on display and specify a connection of the implemented vector to the space and basis it belongs.

The problem equally important for a design and implementation is a choice of the basic software environment (operating system, windows system, programming language) on which the system is to be implemented. The basic environment includes the basic interaction possibilities (windows,

menus, dialog boxes etc.) and the programming language specifics (object-oriented, functional, logic). The implementation is a mapping from application domain to the basic environment. For the mapping to be possible, the following tasks must be completed: decomposition of the application domain into the structure available in the basic environment and construction of the specific software concepts that are present in the application domain structure. The tasks are mutually dependent, and show the importance of the basic software environment specifics.

3.2. Structure of "Global Minimum".

3.2.1. Overview. The software is to solve multivariable local, global and multiobjective optimization problems. The implemented optimization techniques are designed to solve the problems of different classes and complexities. They vary from pure deterministic to pure random ones, from the Monte-Carlo method for global search to the variable metrics method for local optimization. The software has a graphical user interface designed specially for optimization problems, so that a user can influence the process of solving at every moment (to change a method, parameters, etc.). The information on the current state of the problem is presented both in graphical and text form. A dialogue with a user is multilevel and the help is context sensitive.

"Global Minimum" is implemented on IBM PC\XT\AT compatibles. The software is implemented in C (except for some methods in FORTRAN and some graphical procedures in ASSEMBLER). "Global Minimum" is compatible with CGA, EGA, HERCULES graphical cards.

The subset (excluding graphical and interactive part) of the package is implemented on various machines, such as PDP-11, VAX and IBM mainframe.

The system interface is implemented as a number of win-

dows and menus. After each iteration of the method, a check is made for user's input, so an optimization process and user's actions seem to be asynchronous.

The optimal point given by one method doesn't mean much when optimizing a complex function. Experimentation is needed to analyze the behavior of a function and to evaluate what methods and method combinations do the work. "Global Minimum" graphic capabilities and a user interface addresses this problem.

3.2.2. Software techniques and the main objects. The framework of "Global Minimum" is an object structure. It is assumed that everything is represented as an object having state and well defined functionality. The objects sharing structure and functionality are grouped to classes. The object framework is chosen for its generality and simplicity. It is implemented as some programming rules in the C language (Mockus A.,1989).

The objective function is a part of the system that must be changed frequently. In PC-DOS that means the time consuming linking of a function module to a large module of the system. To avoid it,the objective function is linked to the small body of parent process,which, when invoked, calls the main system module – the child process. The objective function is then accessed from the child process as software interrupt. The savings in linking time can be significant, if frequently changing function and using slower computer.

The user interface is realized by menus, function keys and dialog boxes and is sensitive to the context. The latter is provided to make an interaction adequate for the specific of global optimization problems. After the global methods, local ones can be invoked with the chosen parameters and initial points to refine the result.

In "Global Minimum" a dynamic graphical presentation of the function approximation is provided in two forms. The

first one is the best value achieved so far versus the iteration number and the second one is the projection of points onto any chosen plain. The projection of the objective function on a chosen plain with the control menu is shown in Fig. 1. The control menu is at bottom left.

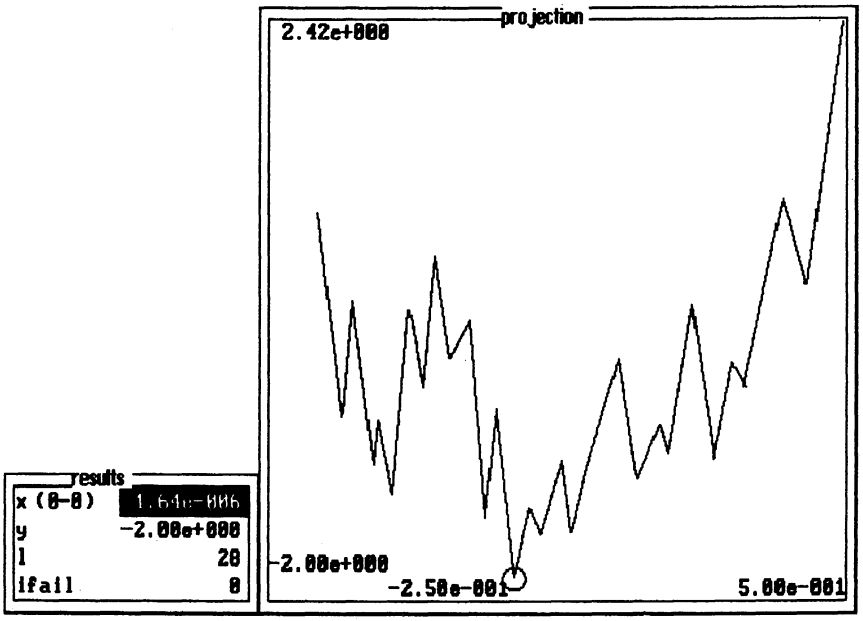

Fig. 1. A projection of the objective function with control menu

The "Global Minimum" consists of functions, constraints, optimization methods and other objects. An objective function is the function to be minimized. It is some computational algorithm, which takes the point in n-dimensional space as an input and returns a function value at this point. The objective function is implemented as a procedure in C or FORTRAN and its changing involves compilation and linking.

The important extension of a function concept used in "Global Minimum" is a function approximation. The function approximation is a collection of all function evaluations. The Bayesian global methods are effective when optimizing a computationally difficult function. For depicting and optimizing such functions, the evaluation several times at the same point may be too time consuming. The function approximation may be used for these tasks. The function approximation can be used by all optimization methods and depicting techniques. It also may be used by other methods for a function exploration.

Constraints are restrictions put onto the domain of objective function. The constraints may have different semantics (they can mean some penalty, can be a region where an objective function is defined, can mean some region inside which the extrema probably exist) (Mockus L.,1990). The constraints are implemented in "Global Minimum" in three ways. For global methods the constraints usually must be rectangle and supplied as the parameters of the method. For local methods the constraints can be expressed as a penalty function, so they are automatically added to an objective function. Most general constraints can be given as a separate instance of a function with the special interpretation of its values. The listed options are useful in solving optimization problems of a different type.

Optimization methods are iterative numeric algorithms that a given function and constraints generate a set of points where to evaluate the function. In "Global Minimum" the methods were decomposed to one iteration level. After each iteration,the methods return control to check for user input and to update the image of function approximation. Some theoretic algorithms and almost all software implementations assume that all iterations are done in one step, without interruption. This can be changed by introducing a state and

iterator for an object optimization method. The iterator yields the next point where to evaluate the objective function and changes the state of optimization method.

3.2.3. Design principles and directions for development. The main idea in a "Global Minimum" design was to implement into software the knowledge from corresponding domain in mathematics and optimization. The implementation of a theory underlying optimization is useful because it is relatively well formalized if compared with software for optimization. In practice,to define a problem is no less difficult than to solve it. The implementation of basic mathematical concepts can facilitate a construction of adequate mathematical model. Besides, mathematical techniques are more "portable" and more productive than software implementation. Not every theoretic concept or technique can be implemented. The objective of the design is to recognize the most important concepts that can be implemented with available computer resources.

The idea of programming environment that mirrors its application domain comes from the development of programming techniques. One direction into which they are evolving is to close a gap between "the way of thinking" and the way of programming. FORTRAN implemented loop and conditional, array and function abstractions. Without them it is quite inconvenient to implement a computational algorithm. The programming abstractions, such as stacks, queue, virtual memory and dispatchers, are implemented in hardware or operating systems. The programming environment Smalltalk-80 implements such important abstractions as: object, classification and classification hierarchy (inheritance). It also implements much of the programming knowledge into its environment using this framework. All abstractions are quite different and come from the corresponding application domains (algorithms, systems programming, real life simulation).

The optimization problem solving using FORTRAN or C can be criticized as a building of an automobile from atoms, not from macro constituent parts, such as wheels, corpus, windows. The existing software systems for optimization usually implement optimization methods and fixed interface to them. This can be criticized at least from three points. First is that every method cannot be appropriate for all problems, so a method is too "big" an abstraction to match all cases from application domain. Second point is that before using a method you must know it is appropriate for the problem. This process frequently cannot be formalized so some software tools must be developed to aid it. We assume that the implementation of mathematical background of optimization can greatly facilitate this process. The third point is that the fixed interface restricts the software system using. The problem can be solved by integration of all (most of) the things needed for an optimization process into one environment.

In the future the "Global Minimum" will be developed towards environment for exploring optimization techniques. The concept of optimization method will be an algorithm,which for the given function approximation and constraints generates a point or a number of points where to evaluate an objective function next. Such an approach allows the easy construction of optimization methods sequence or other exploration techniques within the environment.

The implementation of mathematics underlying the optimization raises two problems. First is the adequacy of software implementation. The main reasoning techniques and abstractions used in application domain must be implemented and they must conform to the corresponding axioms. The second problem is that additional interaction concepts must be invented to use or to communicate with the implemented objects. The problem is addressed in (Mazurik,1989), where the integrated system is constructed from three types of objects.

A display objects or fields are some logic places of a display screen. Functions, scalars, vectors and matrices as application domain objects and mapping objects that map application domain objects into display objects. In the future the "Global Minimum" will be based on the environment (now in development), implementing the main linear algebra objects and continuous optimization techniques. This environment is an effort to solve the described problems.

Depicting of a multivariable function is a part of optimization process. Presenting the 3-dimensional graphs, stars, Andrew's curves, Chernoff's faces and other methods (see du Toit,1986) to represent the functions with up to 20 variables are to be implemented in "Global Minimum". The object-oriented structure of "Global Minimum" allows an easy addition of depicting techniques. (The stars method is to put point components and function value as vectors, pointing in all directions from one point. Fig. 2).

3.3. Evaluating the design. Given two designs, it is desirable to compare them. Usually, the comparison is based on the experience in using a complete system, but it is no less important to compare designs, not only working systems or prototypes. In the case of software for optimization, the formal properties of the design can be valuable. One of the ways to evaluate the design is to find out if it has some useful properties. For example, in (Mazurik,1989) the property of an interface completeness is defined. The idea is that all functionality of implemented application domain concepts must be accessible by an available user interface. Other properties may be stability (small changes in interactive control lead to small changes of results), adequacy of implemented objects to their axiomatic. An interesting property is orthogonality. It is the possibility to make a direct product of objects and functions belonging to different classes. For example, if we have objects a_j of the class A and functions f_i of the class F, and for some

i and j $f_i(a_j)$ make sense, then $f_i(a_j)$ is meaningful for any i, j; While it is a trivial property, it frequently not holds in software and user interface. In practice this property can be expressed in many different ways depending on what is evaluated and chosen as objects and composition. For example, in an interaction technique this property may mean that some function key (say for changing a size of an object) is applicable for all sorts of display objects (windows, menus, lists, dialog boxes, pictures, etc.).

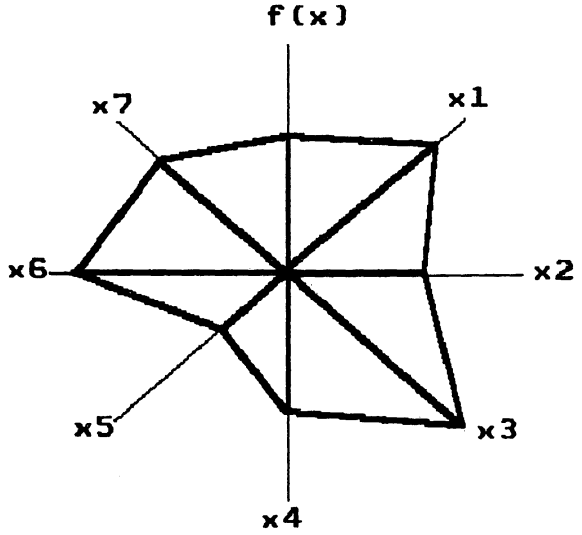

Fig. 2. The stars method

4. Conclusions. The formulation of design problems was helpful when designing the system. Describing the application domain as objects (functions, optimization methods, etc.) was especially useful for implementation of the system. The most challenging problem was to invent techniques for the interaction and presentation of the main objects.

Implementing an interactive system raised some problems connected with the previous software algorithms that did not allow a greater interactivity. Some theoretic algorithms also bore an assumption on restricted interactivity.

For a highly interactive system, the optimization algorithm should have only one iteration, yielding a set of points where to evaluate an objective function, because an iterative method can be constructed as some macro in software environment.

The concept of function approximation is proposed as the means of keeping a record of evaluations for relatively difficult to compute functions.

Programming techniques enable a fast and simple system extension. There are plans to make the system as a test site for optimization methods.

REFERENCES

Dzemyda, G. (1983). LP-search, taking into account the structure of an extremal problem. In A.Žilinskas and G.Dzemyda (Eds.), *Theorija Optimaljnych Reshenij*. Vol.9. Inst. Math. Cybern. Lithuanian Acad. Sci., Vilnius. pp. 39–44 (in Russian).

Himmelblau, D. (1972). *Applied Nonlinear Programming*. McGrawHill Book Company.

Mazurik, V.P. (1989). *Doctor of Physical and Mathematical Sciences Thesis*. Computing Center, Academy of Sciences of the USSR, Moscow (in Russian).

Mockus, A. (1989). Programming in C in an object-oriented style. In *Programinė ESM Įranga, Proceedings of the Lithuanian Software Conference*. Palanga. pp. 62–64 (Lithuanian).

Mockus, J. (1984a). *The Bayesian Approach to Local Optimization*, Preprint No 175. Free University Berlin, Berlin.

Mockus, J. (1984b). The Bayesian Approach to Global Optimiza-

tion. In *Proc. of the ICI Golden Jubilee Int. Conf. on Statistic,* Calcutta.

Mockus, J. (1989). *The Bayesian Approach to Global Optimization,* Kluwer Academic Publishers, the Netherlands. 254pp.

Mockus, J., and L.Mockus (1989). The Bayesian approach to Global Optimization and Application. *Journal of Optimization Theory and Applications,* (in print).

Mockus, L. (1990). *Candidate of Physical and Mathematical Sciences Thesis.* Moscow Physical-Technical Institute, Moscow (in Russian).

Shittkowski, K. (1981). The nonlinear programming method of Wilson, Han and Powel with an augmented Lagrangian type line search function, Part 1: Convergence analysis. *Numerische Mathematik.*

Sobolj, I. (1969). *Multidimensional Numerical Quadrature Formulas and Haar Functions.* Nauka, Moscow (in Russian).

du Toit, S.H.C., A.G.W.Steyn and R.H.Stumpf (1986). *Graphical Exploratory Data Analysis.* Springer-Verlag.

Törn, A. (1978). *A Program for Global Optimization, Multistart with Clustering,* Rapporter Fron DataCentralen vid abo Academi, No.7.

Žilinskas, A. (1986). *Global Optimization-Axiomatic of Statistical Models, Algorithms and Their Application.* Mokslas, Vilnius (in Russian).

MINIMUM (1984). *Software Package "MINIMUM" for Global Optimization.* The fund of Algorithms and Programs, registration No.50860000112, Vilnius.

Received January 1990

A. Mockus graduated the Moscow Physical-Technical Institute, USSR, in 1988. He is a software engineer at the Department of Optimal Decisions Theory, Institute of Mathematics and Cybernetics, Vilnius, Lithuania and a postgraduate student at the Department of Optimization of the Computing Center, the USSR Academy of Sciences, Moscow, USSR. His field of interest is software development techniques and interactive software systems for engineering.

L. Mockus graduated the Moscow Physical-Technical Institute, USSR, in 1984. He is a junior researcher at the Department of Optimal Decisions Theory, Instit. Math. and Cybern., Vilnius, Lithuania. His field of interest is development of software systems for global optimization.