Nonlinear Programming Software

for Members and Students of Academic Institutions

Revised: February, 2010

Klaus Schittkowski

The subsequent pages contain brief summaries of mathematical programming codes, numerical analysis software and related interactive systems that were developed by the author. More detailed reports describing the mathematical algorithms, numerical tests or complete documentations can be provided on request. All numerical routines are implemented in Fortran without global variables (COMMON) or 'tricky' Fortran constructs like EQUIVALENCE or ENTRY. Transfer to C by f2c is possible. Nonlinear functions and, if required, partial derivatives must be provided by reverse communication, i.e., within the same code from where the optimization routine is called.

Address: Prof. Dr. K. Schittkowski
Siedlerstr. 3
D - 95488 Eckersdorf

Phone: +921 557750 (office)
+921 32887 (home)

Fax: +921 557752

E-mail: klaus.schittkowski@uni-bayreuth.de
Web: http://www.klaus-schittkowski.de/

Contents

1 NLPQLP - Nonlinear Constrained Optimization

Purpose:

NLPQLP solves general nonlinear programming problems with equality and inequality constraints. It is assumed that all problem functions are continuously differentiable.

Method:

NLPQLP is a new implementation of a sequential quadratic programming (SQP) method. Proceeding from a quadratic approximation of the Lagrangian function and a linearization of constraints, a quadratic programming subproblem is formulated and solved by QL (see below). The Hessian approximation is updated by the modified BFGS-formula. Depending on the number of nodes of the distributed system, objective and constraint functions can be evaluated simultaneously at predetermined test points along the search direction. Serial and parallel line search is performed with respect to an augmented Lagrangian merit function. In case of an error situation in the line search, a non-monotone line search is started. It can be shown that especially for very noisy objective or constraint functions, the robustness of the code is drastically improved.

Program Organization:

NLPQLP is written in double precision Fortran and is organized in form of a subroutine. Nonlinear problem functions and corresponding gradients must be provided by the user within the calling program by reverse communication. Depending on a flag, either new function values (IFAIL=-1) or new gradient values (IFAIL=-2) must be computed and NLPQLP must be started again. If NLPQLP is implemented under a distributed environment, each iteration of NLPQLP requires one simultaneous function evaluation for the line search and another one for approximation of gradients, if the number of parallel processors is sufficiently big. The quadratic programming solver can be exchanged by any other one, e.g., to exploit sparsity patterns.

Special Features:

- upper and lower bounds on the variables handled separately
- initial multiplier and Hessian estimates allowed
- bounds and linear constraints remain satisfied
- fast final convergence speed
- robust even in case of very noisy function values

- full documentation and example codes
- Fortran source code

Applications:

NLPQLP and its previous versions are part of commercial redistributed optimization systems, e.g.,

- IMSL Library (Visual Numerics Inc., Houston), Version 1.0 of 1981,
- ANSYS/POPT (CAD-FEM, Grafing) for structural optimization,
- DesignXplorer (ANSYS Inc., Canonsburg) for structural design optimization,
- STRUREL (RCP, Munich) for reliability analysis,
- Microwave Office Suit (Applied Wave Research, El Segundo) for electronic design,
- MOOROPT (Marintek, Trondheim) for the design of mooring systems,
- iSIGHT (Enginious Software, Cary, North Carolina) for multi-disciplinary CAE,
- POINTER (Synaps, Atlanta) for design automation,
- EXCITE (AVL, Graz) for non-linear dynamics of power units,
- FRONTIER (ESTECO, Trieste) for integrated multi-objective and multi-disciplinary design optimization,
- MathCad (MathSoft, Boston) for constrained least squares optimization,
- TOMLAB/MathLab (Tomlab Optimization, Västeras, Sweden) for general nonlinear programming, least squares optimization, data fitting in dynamical systems,
- OptiSLang (DYNARDO, Weimar), for structural design optimization,
- AMESim (IMAGINE, Roanne), for multidisciplinary system design,
- OPTIMUS (NOESIS, Leuven, Belgium) for multi-disciplinary CAE,
- RADIOSS/M-OPT (MECALOG, Antony, France) for multi-disciplinary CAE,
- CHEMASIM (BASF, Ludwigshafen) for the design of chemical reactors.

Customers include AMD, Applied Research Corp., Aware, Astrium, Axiva, BASF, Bastra, Bayer, Bell Labs, BMW, CEA, Chevron, DLR, Dornier Systems Dow Chemical, DuPont, EADS, EMCOSS, ENSIGC, EPCOS, ESOC, Eurocopter, Fantoft Prosess, Fernmeldetechnisches Zentralamt, General Electric, GLM Lasertechnik, Hidroelectrica Espanola, Hoechst, IABG, IBM, INRIA, INRS-Telecommunications, KFZ Karlsruhe, Kongsberg Maritime, Lockheed Martin, Markov Processes, Mathematical Systems Institute Honcho, Micronic Laser Systems, MTU, NASA Langley, Nevesbu, National Airspace Laboratory, Norsk Hydro Research, Norwegian Computing Center, Numerola, OECD Halden, Peaktime, Philips, Polysar, ProSim, Research Triangle Institute, Rolls-Royce, SAQ Kontroll, Shell, Siemens, Solar Turbines, Space Systems/Loral, Statoil, TNO, Transpower, USAF Research Lab, VTT Chemical Technology, Wright R & D Center, and in addition dozens of academic research institutions all over the world.

License:

Contact the author for getting the Fortran source code. Note that also a license for the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

2 NLPQLY - Easy-To-Use Nonlinear Constrained Optimization

Purpose:

NLPQLY is the easy-to-use version of NLPQLP for solving general nonlinear programming problems with equality and inequality constraints. It is assumed that all problem functions are continuously differentiable.

Method:

After setting some default tolerances, NLPQLP is executed. Derivatives are internally approximated by forward differences.

Program Organization:

NLPQLY is written in double precision Fortran and is organized in form of a subroutine. Nonlinear objective and constraint function values must be provided by the user within the calling program by reverse communication. Depending on a flag, either new function values (IFAIL<0) must be computed and NLPQLY has to be started again, an error occurred (IFLAG>0), or the optimality conditions are satisfied (IFAIL=0).

Special Features:

- upper and lower bounds on the variables handled separately
- bounds and linear constraints remain satisfied
- fast final convergence speed
- robust even in case of very noisy function values
- full documentation and example codes
- Fortran source code

Applications:

NLPQLY has the same applications as NLPQLP, as long as forward difference approximations for derivatives are used.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

3 NLPQLB - Nonlinear Programming with Very Many Constraints

Purpose:

NLPQLB is an extension of the general nonlinear programming code NLPQLP with the intention to solve also problems with very many constraints, even when the Jacobian matrix of the constraints does not possess any special sparsity structures.

Method:

The user defines the maximum number of lines in the matrix of the linearized constraints that can be stored in core. Constraint function values are investigated to update this working set, which must contain at least all violated constraints. The algorithm stops, if too many constraints are violated.

Program Organization:

NLPQLB is a double precision Fortran subroutine where all parameters are passed through subroutine arguments. The program organization is very similar to that of NLPQLP.

Special Features:

- solves problems with up to 200,000,000 nonlinear constraints
- NLPQLB executes NLPQLP in reverse communication
- full documentation by initial comments
- Fortran source code

Applications:

NLPQLB is implemented as part the structural mechanical optimization system LAGRANGE of EADS to solve large design optimization problems by means of finite element technique.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

4 NLPQLG - Nonlinear Global Optimization

Purpose:

NLPQLG is an extension of the general nonlinear programming code NLPQLP with the goal to improve local minima successively.

Method:

Successively, local minima are cut off by introducing additional restrictions. Moreover, artificial constraints prevent approximation of known local minima. These constraints are relaxed to prevent infeasible domains.

Program Organization:

NLPQLG is a double precision Fortran subroutine where all parameters are passed through subroutine arguments. The program organization is very similar to that of NLPQLP.

Special Features:

- NLPQLB executes NLPQLP in reverse communication
- full documentation by initial comments
- Fortran source code

Applications:

Used, e.g., at EPCOS AG for computing optimal designs of electronic components.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

5 NLPJOB - Multicriteria Optimization

Purpose:

NLPJOB solves multicriteria problems interactively, i.e. problems with more than one objective function.

Method:

By using a suitable transformation, a scalar nonlinear problem is created and solved by NLPQLP. There are 15 different options available for formulating the scalar subproblem.

Program Organization:

NLPJOB is a double precision Fortran subroutine where all parameters are passed through subroutine arguments. Problem functions and gradients must be provided by the user in form of subroutines.

Special Features:

- reverse communication
- Fortran source code

Applications:

NLPJOB is implemented within the structural mechanical optimization system LAGRANGE of EADS.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

6 NLPLSQ - Constrained Nonlinear Least Squares Optimization

Purpose:

NLPLSQ solves constrained nonlinear least squares problems, i.e., nonlinear optimization problems, where the objective function is the sum of squares of function. In addition, there may be any set of equality or inequality constraints. It is assumed that all individual problem functions are continuously differentiable. The most important application is data fitting, where the distance of experimental data from a model function evaluated at given experimental times is to be minimized by the L_2 or sum of squares norm.

Method:

By introducing additional variables and constraints, the problem is transformed into a general smooth nonlinear programming problem which is then solved by the sequential quadratic programming (SQP) code NLPQLP. It can be shown that typical features of special purpose algorithms are retained, i.e., a combination of a Gauss-Newton and a quasi-Newton search direction. The additionally introduced variables are eliminated in the quadratic programming subproblem, so that calculation time is not increased significantly.

Program Organization:

NLPLSQ is a double precision Fortran subroutine and parameters are passed through arguments.

Special Features:

- reverse communication
- bounds and linear constraints remain satisfied
- Fortran source code

Applications:

The code is in practical use to solve parameter estimation problems, e.g., in chemical and pharmaceutical applications, see EASY-FIT.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

7 NLPLSX - Constrained Nonlinear Least Squares Optimization with Very Many Experimental Data

Purpose:

NLPLSX solves constrained nonlinear least squares problems, i.e., nonlinear optimization problems, where the objective function is the sum of squares of function. In addition, there may be any set of equality or inequality constraints. It is assumed that all individual problem functions are continuously differentiable. The most important application is data fitting, where the distance of experimental data from a model function evaluated at given experimental times is to be minimized by the L_2 or sum of squares norm.

Method:

By assuming now that the transformation technique for NLPLSQ is not applicable because of a too large number of objective functions or measurements, respectively, the sum of squared functions is directly minimized by NLPQLP. Constraints, if available, are passed to NLPQLP.

Program Organization:

NLPLSX is a double precision Fortran subroutine and parameters are passed through arguments.

Special Features:

- reverse communication
- bounds and linear constraints remain satisfied
- Fortran source code

Applications:

The code is in practical use to solve parameter estimation problems, e.g., in chemical and pharmaceutical applications, as part of EASY-FIT.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

8 NLPINF - Constrained Nonlinear Maximum-Norm Optimization

Purpose:

NLPINF solves constrained nonlinear L_∞ problems, i.e., nonlinear optimization problems, where the objective function is the maximum of absolute function values. In addition there may be any set of equality or inequality constraints. It is assumed that all individual problem functions are continuously differentiable. The most important application is data fitting, where the distance of experimental data from a model function evaluated at given experimental times is to be minimized by the L_∞ or maximum norm, respectively.

Method:

By introducing one additional variable and additional inequality constraints, the problem is transformed into a general smooth nonlinear programming problem which is then solved by the sequential quadratic programming (SQP) code NLPQLP.

Program Organization:

NLPINF is a double precision Fortran subroutine and parameters are passed through arguments.

Special Features:

- reverse communication
- bounds and linear constraints remain satisfied
- Fortran source code

Applications:

The code is part of the data fitting software EASY-FIT.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

9 NLPMMX - Constrained Nonlinear Min-Max Optimization

Purpose:

NLPMMX solves constrained nonlinear min-max problems, i.e., nonlinear optimization problems, where the objective function is the maximum of absolute function values. In addition there may be any set of equality or inequality constraints. It is assumed that all individual problem functions are continuously differentiable.

Method:

By introducing one additional variable and additional constraints, the problem is transformed into a general smooth nonlinear programming problem which is then solved by the sequential quadratic programming code NLPQLP.

Program Organization:

NLPMMX is a double precision Fortran subroutine and parameters are passed through arguments.

Special Features:

- reverse communication
- bounds and linear constraints remain satisfied
- Fortran source code

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

10 NLPL1 - Constrained Optimization of Sums of Absolute Function Values

Purpose:

NLPL1 solves constrained nonlinear L_1 problems, i.e., nonlinear optimization problems, where the objective function is the sum of absolute function values. In addition there may be any set of equality or inequality constraints. It is assumed that all individual problem functions are continuously differentiable. The most important application is data fitting, where the distance of experimental data from a model function evaluated at given experimental times is to be minimized by the L_1 norm.

Method:

By introducing additional variables and constraints, the problem is transformed into a general smooth nonlinear programming problem which is then solved by the sequential quadratic programming code NLPQLP.

Program Organization:

NLPL1 is a double precision Fortran subroutine and parameters are passed through arguments.

Special Features:

- reverse communication
- bounds and linear constraints remain satisfied
- Fortran source code

Applications:

The code is part of the data fitting software EASY-FIT.

License:

Contact the author for getting the Fortran source code. Note that also a license for the SQP code NLPQLP and the quadratic programming solver QL must be obtained. The software is free for individuals of academic institutions, see below for details.

11 QL - Quadratic Programming

Purpose:

QL solves quadratic programming problems with a positive definite objective function matrix and linear equality and inequality constraints.

Method:

The algorithm is an implementation of the dual method of Goldfarb and Idnani and a modification of the original implementation of Powell. Initially, the algorithm computes a solution of the unconstrained problem by performing a Cholesky decomposition and by solving the triangular system. In an iterative manner violated constraints are added to a working set and a minimum with respect to the new subsystem with one additional constraint is calculated. Whenever necessary, a constraint is dropped from the working set. The internal matrix transformations are performed in numerically stable way.

Program Organization:

QL is a double precision Fortran subroutine where all data are passed by subroutine arguments.

Special Features:

- separate handling of upper and lower bounds
- initially given Cholesky decomposition can be exploited
- full documentation by initial comments
- Fortran source code

Applications:

As an essential part of the nonlinear programming routine NLPQLP, QL solves the internal quadratic programming subproblem of the SQP method and has therefore the same domain of application as NLPQLP.

License:

Contact the author for getting the Fortran source code. The software is free for individuals of academic institutions, see below for details.

12 MIQL - Mixed-Integer Quadratic Programming

Purpose:

The Fortran subroutine MIQL solves strictly convex mixed-integer quadratic programming problems subject to linear equality and inequality constraints by a branch-and-cut method. At the root node of the branch-and-bound search tree, disjunctive and complemented mixed-integer rounding cuts are generated. The continuous subproblem solutions are obtained by the primal-dual method of Goldfarb and Idnani. The code is designed for solving small to medium size mixed-integer programs. .

Method:

A branch-and-cut strategy is implemented where different options are available for selecting a branching variable and a free node:

1. *maximal fractional branching*: We select an integer variable from the solution of the relaxed subproblem with largest distance from next integer value.
2. *minimal fractional branching*: We select an integer variable from the solution of the relaxed subproblem with smallest distance from next integer value.

Then we search for a free node from where branching, i.e., the generation of two new subproblems, is started:

1. *best of two*: The optimal objective function values of the two child nodes are compared and the node with a lower value is chosen. If both are leafs, i.e., if the corresponding solution is integral, or if the corresponding problem is infeasible or if there is already a better integral solution, strategy *best of all* is started.
2. *best of all*: All free nodes are compared and a node with lowest objective function value is selected.
3. *depth first*: This search strategy selects a child node whenever possible. If the node is a leaf the *best of all* strategy is applied.

Moreover, there are complemented mixed-integer rounding cutting planes and disjunctive programming cuts to reduce the search space and to cut-off non-optimal integer points. The corresponding continuous relaxed problems are solved by the Fortran code QL.

Program Organization:

MIQL is a double precision Fortran subroutine where all data are passed by subroutine arguments.

Special Features:

- separate handling of upper and lower bounds
- full documentation by initial comments
- exploiting dual information for early branching
- warm starts
- Fortran source code

Applications:

As an essential part of the mixed-integer nonlinear programming routine MISQP, MIQL solves the internal mixed-integer quadratic programming subproblem of the SQP-trust-region method.

License:

Contact the author for getting the Fortran source code. Note that also a license for the quadratic programming solver QL must be obtained.

13 MISQP - Mixed-Integer Sequential Quadratic Programming

Purpose:

The Fortran subroutine MISQP solves mixed-integer nonlinear programming problems by a modified sequential quadratic programming (SQP) method. Under the assumption that integer variables have a *smooth* influence on the model functions, i.e., that function values do not change drastically when in- or decrementing an integer variable, successive quadratic approximations are applied. It is not assumed that integer variables are relaxable, i.e., problem functions are evaluated only at integer points. The code is applicable also to nonconvex optimization problems.

Method:

The algorithm is stabilized by a trust region method including Yuan's second order corrections. The Hessian of the Lagrangian function is approximated by BFGS updates subject to the continuous and integer variables. Successively, mixed-integer quadratic programs must be solved.

Program Organization:

MISQP is a Fortran subroutine where all data are passed by subroutine arguments. Function and gradient values must be submitted through reverse communication. Partial derivatives subject to integer variables are approximated internally at grid points. The generated mixed-integer quadratic programming subproblems must be solved by the code MIQL.

Special Features:

- boolean, integer, and continuous variables
- separate handling of upper and lower bounds
- full documentation by initial comments
- integer, real and logical option arrays
- separate handling of linear constraints
- exploiting known partial derivatives subject to integer variables
- Fortran source code

Applications:

The development of MISQP was supported by the Shell GameChanger program, and MISQP is in daily use as part of Shell's simulation packages. In addition, MISQP is in use at Dassault, General Electric, Epcos and at several smaller companies and research institutes.

License:

Contact the author for getting the Fortran source code. Note that also a license for the mixed-integer quadratic programming solver MIQL and the quadratic programming solver QL must be obtained.

14 MISQPOA - Mixed-Integer Outer Approximation

Purpose:

The Fortran subroutine MISQPOA solves mixed-integer nonlinear programming problems by the modified sequential quadratic programming code MISQP stabilized by an outer approximation master program. Convergence can be guaranteed for convex programs.

Method:

Outer approximation methods apply linear approximations at previous iterates to stabilize the algorithm and to guarantee global optimality for convex problems. A linear programming master program must be solved to determine a lower bound. Afterwards, a nonlinear optimization program is generated to improve the best known point by solving another mixed-integer nonlinear program by the code MISQP. Additional safeguards allow to apply MISQPOA also to non-convex and non-relaxable nonlinear mixed-integer programs, but without guaranteeing global optimality. In the non-relaxable case, derivatives are approximated at neighbored grid points.

Program Organization:

MISQPOA is a Fortran subroutine where all data are passed by subroutine arguments. Function and gradient values must be submitted through reverse communication. Partial derivatives subject to integer variables are approximated internally at grid points. The generated mixed-integer linear programming subproblems are solved by the code MISP.

Special Features:

- boolean, integer, and continuous variables
- separate handling of upper and lower bounds
- full documentation by initial comments
- integer, real and logical option arrays
- separate handling of linear constraints
- exploiting known partial derivatives subject to integer variables
- Fortran source code

Applications:

Same as for MISQP.

License:

Contact the author for getting the Fortran source code. Note that also a license for the mixed-integer sequential quadratic programming solver MISQP and the mixed-integer quadratic programming solver MIQL must be obtained.

15 MINLPB4 - Mixed-Integer Branch-and-Bound

Purpose:

The Fortran subroutine MINLPB4 solves mixed-integer nonlinear programming problems by a branch-and-bound method, where the nodes of the search tree consist either of mixed-integer or of continuous optimization problems differing only by modified bounds. These subproblems are solved by the sequential quadratic programming code MISQP in both situations.

Method:

A branch-and-bound strategy is implemented where different options are available for selecting a branching variable and a free node:

1. *maximal fractional branching*: We select an integer variable from the solution of the relaxed subproblem with largest distance from next integer value.

2. *minimal fractional branching*: We select an integer variable from the solution of the relaxed subproblem with smallest distance from next integer value.

Branching is possible subject to the integer and boolean variables or, alternatively, only to the boolean variables. Then we search for a free node from where branching, i.e., the generation of two new subproblems, is started:

1. *best of two*: The optimal objective function values of the two child nodes are compared and the node with a lower value is chosen. If both are leafs, i.e., if the corresponding solution is integral, or if the corresponding problem is infeasible or if there is already a better integral solution, strategy *best of all* is started.

2. *best of all*: All free nodes are compared and a node with lowest objective function value is selected.

3. *depth first*: This search strategy selects a child node whenever possible. If the node is a leaf the *best of all* strategy is applied.

Additional safeguards allow to apply MINLPB4 also to non-convex and non-relaxable nonlinear mixed-integer programs, but without guaranteeing global optimality. In the non-relaxable case, derivatives are approximated at neighbored grid points.

Program Organization:

MINLPB4 is a Fortran subroutine where all data are passed by subroutine arguments. Function and gradient values must be submitted through reverse communication. Partial derivatives subject to integer variables are approximated internally at grid points.

The generated mixed-integer nonlinear programming subproblems are solved by the code MISQP.

Special Features:

- boolean, integer, and continuous variables
- separate handling of upper and lower bounds
- full documentation by initial comments
- integer, real and logical option arrays
- separate handling of linear constraints
- exploiting known partial derivatives subject to integer variables
- Fortran source code

Applications:

Same as for MISQP.

License:

Contact the author for getting the Fortran source code. Note that also a license for the mixed-integer sequential quadratic programming solver MISQP and the mixed-integer quadratic programming solver MIQL must be obtained.

16 PCOMP - Automatic Differentiation

Purpose:

PCOMP reads symbolically defined nonlinear functions that are composed of standard elementary functions, and precompiles them. Subsequently function and, in particular, derivative values up to order two can be computed directly, i.e. without numerical approximation. Alternatively Fortran code for function and gradient evaluation can be generated automatically by PCOMP.

Method:

The underlying syntax of the proposed language is described by means of a formal grammar and is similar to the Fortran language with respect to the input format and arithmetic expressions. In case of successful syntax check, an intermediate code is stored in an integer and a real working array and is passed to the subroutines that evaluate function and gradient values. Gradient evaluation is performed either by forward or backward evaluation.

Program Organization:

PCOMP consists of three precision Fortran subroutines for parser, function and gradient evaluation, and code generation. Data must be passed on working arrays or files between these subroutines .

Special Features:

- arbitrary index sets
- constants
- names for variables
- SUM- and PROD-statements
- indexed functions, data and variables
- IF, ELSE, ENDIF statements, nested
- interface to user provided functions
- full documentation by separate report
- Fortran source code

Applications:

PCOMP is used to develop executable codes for nonlinear optimization which allow symbolic input of nonlinear problem functions, and is implemented, e.g., in EASY-FIT.

License:

Contact the author for getting the Fortran source code. The software is free for individuals of academic institutions, see below for details.

17 EASY-FIT ModelDesign - Data Fitting and Experimental Design of Dynamical Systems

Purpose:

EASY-FIT ModelDesign is an interactive user interface running under Windows, to facilitate the formulation and numerical solution of data fitting problems, where the underlying model is either an explicit function, a Laplace transform, a steady-state system of equations, an ordinary differential equation, a differential algebraic differential equation or a one-dimensional, time-dependent partial differential equation with or without algebraic equations. In addition to data fitting, also optimal experimental designs can be determined.

Method:

EASY-FIT ModelDesign is a user interface implemented in form of a database, to execute the parameter estimation codes MODFIT and PDEFIT in form of external programs. Nonlinear functions can be provided by the PCOMP modeling language or in Fortran. EASY-FIT ModelDesign creates input files to execute the numerical codes, and stores results in the database. EASY-FIT comes with statistical analysis utilities, i.e., confidence intervals, significance levels of parameters, experimental design, and optimal location of time values.

Program Organization:

EASY-FIT ModelDesign is a complex software system running under MS-Windows XP or Vista. A version for Windows 2K or earlier can be provided on request. The GUI is implemented in form of an MS Access 2–7 database, the numerical codes in Fortran. Interfaces for the INTEL Visual Fortran, WATCOM F77, Salford FTN77, Microsoft Fortran Powerstation, Digital Visual Fortran, Absoft Pro Fortran, and the Lahey F77L3 compiler are available.

Special Features:

- embedded modeling language with automatic differentiation (PCOMP),
- declaration of model type and solution parameters by pick-lists, combo boxes etc.
- import and export of data and functions (text, Excel)
- restarts with previously computed variable values

- interactive, context sensitive help
- distributed with run-time version of MS-Access 2007
- 1,300 test problems
- interactive plot facilities for model functions, measurements, residuals and 3D contours based on internal or external graphics systems
- report generator
- comprehensive documentation (about 380 pages)
- trial version (download from home page http://www.klaus-schittkowski.de/, activation by license key)

Applications:

EASY-FIT ModelDesign is applied in chemical engineering, pharmaceutics, mechanical engineering, and in natural sciences like biology, geology, ecology. Customers include ACA, BASF, Battery Design, Bayer, Boehringer Ingelheim, Envirogain, Eurocopter, Knoll, Oxeno, Novartis, Prodisc, Springborn Laboratories, and in addition more than 40 academic research institutions.

License:

Download trial version from home page and contact the author for license key. The software is free for individuals of academic institutions, see below for details.

18 EASY-FIT Express - Interactive Data Fitting

Purpose:

EASY-FIT Express is a subset of EASY-FIT ModelDesign to solve data fitting problems, where the underlying model is given by explicit analytical function. Nonlinear equality or inequality constraints are permitted.

Method:

Data must be defined through input masks, where nonlinear functions are provided by the modeling language PCOMP. EASY-FIT Express creates input files to call an executable nonlinear programming program EXPFIT, that can be used also independently from the interface. Functions are evaluated directly during run time and gradients are computed automatically by PCOMP.

Program Organization:
EASY-FIT Express is a user interface implemented in form of a database, to execute the parameter estimation code EXPFIT in form of an external program. Nonlinear functions are provided in the PCOMP language. EASY-FIT Express creates input files to execute the numerical codes, and stores results in the database. EASY-FIT Express comes with statistical analysis utilities, i.e., confidence intervals, significance levels of parameters, experimental design, and optimal location of time values.

Special Features:

- embedded modeling language with automatic differentiation (PCOMP),
- import and export of data and functions (text, Excel)
- restarts with previously computed variable values
- interactive, context sensitive help
- distributed with run-time version of MS-Access 2007
- 240 test problems
- interactive plots for model functions, measurements, residuals and 3D contours
- report generator
- comprehensive documentation (about 380 pages)

Applications:

To be used for academic purposes and data fitting applications based on analytical functions.

License:

Free, download from home page.

19 EASY-OPT Express - Interactive Optimization

Purpose:

EASY-OPT Express is an interactive user interface running under Windows, to facilitate the formulation of nonlinear programming models, their implementation and numerical solution. It is possible to solve general constrained nonlinear programming problems interactively, i.e., to minimize a nonlinear objective function subject to nonlinear equality or inequality constraints.

Method:

Nonlinear functions are defined in the PCOMP modeling language. EASY-OPT Express creates input files to call an executable program, that can be used also independently from the interface. Functions are evaluated during run time and gradients are computed automatically by PCOMP. The SQP code NLPQLP is internally implemented to solve optimization problems efficiently.

Program Organization:
The user interface is implemented in form of a database under MS-Access 2007, the numerical algorithms in Fortran. A distribution-free runtime version of Microsoft Access 2007 is included. EASY-OPT Express runs under Windows XP and Vista and can be downloaded from the home page of the author. A version for Windows 2K or earlier can be provided on request. Since nonlinear functions are interpreted by the modeling language PCOMP, it is not necessary to compile or link any subroutines.

Special Features:

- up to 200 variables and 1,000 nonlinear equality and inequality constraints
- interactive, context-sensitive help (F1)
- extensive documentation (pdf)
- internal editor for PCOMP-code
- large set of test problems
- optimization history plot
- report generator

Applications:

To be used for academic purposes and smaller optimization applications.

License:

Free, download from home page.

20 How to Get Software

EASY-FIT ModelDesign, EASY-FIT Express, and EASY-OPT Express can be downloaded from the home page of the author. To activate EASY-FIT ModelDesign, a license key is requested. To get the license key for EASY-FIT ModelDesign or to get any of the Fortran codes, contact the author under klaus@schittkowski.de or klaus.schittkowski@uni-bayreuth.de.
The software is free for members and students of academic institutions. An academic institution is defined by its ability to provide academic degrees, e.g., Bachelor, Master, Diploma, or Ph.D. degrees. If you are a member or a student of an academic institution, please outline your application in a few lines and prepare a letter confirming that

1. you have read and accept to be bound by the terms of the License Agreement listed below,
2. to use the code only for your own research,
3. to delete the code after finishing your research project or your thesis, respectively,
4. not to pass the code to any other person or organization, neither in form of the source code nor in form of object code or part of another piece of software,
5. to add a reference in related publications, and to send a copy to the author.

Note that especially nobody else in your department is allowed to use the software. Use your full institutional or, in case of a student, private address and sign it. Then scan the letter and attach it to an e-mail. If you are a student, add a letter of your supervisor confirming that you conduct a thesis (Bachelor, Master, Diploma, Ph.D.) under his supervision.
After receiving the agreement, you will get the license key for activating EASY-FIT ModelDesign or the requested Fortran codes by e-mail. Please let me know the operating system and your compiler.

License Agreement:

This license Agreement is a legal agreement between Klaus Schittkowski (Siedlerstr. 3, D-95448 Eckersdorf), subsequently called the Author, of the one part and You, the user, whose name and address are set out on the accompanying letter of the other part, for the software subsequently called the Software, which is specified on the accompanying letter and which includes computer software and electronic documentation. By installing, copying, or otherwise using the Software or any updates, you agree to be bound by the terms of this Agreement. If you do not agree to the terms of this Agreement, do not install, copy, or otherwise use the Software.

The software is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties.

LICENSE TO USE THE SOFTWARE:

1. **General License Grant (Single User License).** The Author grants to You a personal, non-exclusive, non-transferable license to make and use copies of the Software provided that You are the only individual using the Software, in perpetuity unless terminated under the specific provisions of this Agreement. If you are an entity, the Author grants you the right to designate one individual within your organization to have the sole right to use the Software in the manner provided above.

2. **Limitations on Commercial Use.** If the license for the Software is designated as something other than a commercial license, including academic or research licenses, student licenses, or certain free promotional licenses, then You may not use the license for commercial gain or purpose. Moreover, if the license is designated as a trial or evaluation license, then it may not be used for any commercial purposes.

3. **Transfer of Software.** You may permanently transfer your rights to use the Software, the Software itself including any updates to the purchased version of the Software, and the accompanying documentation, including this Agreement, provided that you retain no copies of the Software, updates, documentation, or this Agreement and the recipient accepts all of the terms and conditions of this Agreement without reservation or condition. The Author must be notified in writing of any such transfer within three months. Any attempts otherwise to rent, lease, sublicense, assign, or otherwise transfer any of the rights, duties or obligations of this Agreement is not permitted, shall be void and of no force or effect.

4. **Copyright.** Title and copyright to the Software, in whole and in part and all copies thereof, and all modifications, enhancements, derivatives and other alterations of the Software regardless of who made any modifications, if any, are, and will remain, the sole and exclusive property of the Author. You may not use, copy, modify, disassemble, decompile, reverse engineer or transfer the programs or any copy, modification, or merged portion, in whole or in part, except as expressly provided for this Agreement. If you transfer possession of any copy, modification or merged portion of the program to another party, your license is automatically terminated. You can not sublicense, assign or otherwise transfer the license to a third party granted by this Agreement. The software is licensed, not sold.

5. **Documentation.** This Agreement grants You a nonexclusive license to make and use an unlimited number of copies of any documentation, provided that such copies shall be used only for personal purposes and are not to be republished or distributed (either in hard copy or electronic form) beyond the user's premises.

6. **Storage/Network Use.** You may store or install a copy of the Software on a storage device, such as a network server, used only to install or run the Software on computers used by a licensed end user in accordance with Section 1. A license for the Software may not be shared or used concurrently by other end users.

7. **Limited Warranty and Liability.** The Author warrants that the Software will perform substantially as described in the documentation accompanying the Software for a period of ninety days from downloading or activating. Other than this limited warranty, the Author does not warrant that the functions contained in the program will meet your requirements or that the operation of the program is uninterrupted or error free. While the Author has attempted to assure that use of the Software will not result in any errors or miscalculations, the Author is not responsible in the event of any errors or miscalculations occurring. You assume the entire risk as to the quality, use and performance of the programs. Should the programs prove defective, you, and not the Author, assume the entire cost of necessary servicing, repair or correction. Your sole remedies and the entire liability of the Author are set forth above.

8. **Disclaimer of Damages.** IN NO EVENT SHALL THE AUTHOR BE LIABLE TO YOU UNDER ANY THEORY FOR ANY DAMAGES SUFFERED BY YOU OR ANY USER OF THE SOFTWARE, OR FOR ANY SPECIAL, INCIDENTAL, INDIRECT, CONSEQUENTIAL, OR SIMILAR DAMAGES (INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE, OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF THE AUTHOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, AND REGARDLESS OF THE LEGAL OR EQUITABLE THEORY (CONTRACT, TORT OR OTHERWISE) UPON WHICH THE CLAIM IS BASED. IN ANY CASE, THE AUTHOR'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

9. **Jurisdiction.** This agreement shall be governed by the laws of Germany. Any disputes arising in relation to this Agreement shall be subject to the exclusive jurisdiction of the German courts.

10. **Acknowledgement.** You acknowledge that you have read this Agreement, understand it, and agree to be bound by its terms and conditions. You further agree it is the complete and exclusive statement of the agreement between us that supersedes any proposal or prior agreement, oral or written and any other communications between us relating to this subject matter of this Agreement.

11. **Termination.** If You fail to comply with the terms and conditions of this Agreement, the Author may terminate this Agreement and Your right and license to use the Software. You may terminate this Agreement at any time by notifying the Author. Upon the termination of this Agreement for any reason, You must delete the Software from Your computer and archives.

12. **Validity.** The Agreement is part of the Software, accepted by You as it is, and valid without signing it.