

# iSIGHT/NLPQL

Prof. K. Schittkowski,  
Mathematical Institute,  
University of Bayreuth,  
D - 95440 Bayreuth, Germany

---

## 1. INTRODUCTION

In this document we describe some basic features of the sequential quadratic programming algorithm NLPQL of Schittkowski (1985/86), a Fortran code for solving nonlinear programming problems of the form

$$\begin{aligned} & \mathbf{min} \quad f(x) \\ x \in R^n: & \quad g_j(x) = 0 \quad , j = 1, \dots, m_g, \\ & \quad g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m, \\ & \quad x_l \leq x \leq x_u . \end{aligned}$$

It is assumed that objective function and constraints are continuously differentiable. The idea is to generate a sequence of quadratic programming subproblems obtained by a quadratic approximation of the Lagrangian function and a linearization of the constraints. Second order information is updated by a quasi-Newton formula and the method is stabilized by an additional line search.

A particular advantage of NLPQL is its ease of use based on a very robust implementation. Only the maximum number of iterations and the desired final accuracy must be defined by the user.

---

## 2. THE MATHEMATICAL ALGORITHM

Sequential quadratic programming or SQP methods are the standard general purpose tool for solving smooth nonlinear optimization problems under the following assumptions:

- The problem is not too large.
- The functions and gradients can be evaluated with sufficiently high precision.
- The problem is smooth and well-scaled.
- There is no further model structure that can be exploited.

The mathematical convergence and the numerical performance properties of SQP methods are very well understood now and are published in so many papers, that only a few can be mentioned here. Theoretical convergence is investigated in Han (1976,1977), Powell (1978a,1978b), Schittkowski (1983), e.g., and the numerical comparative studies of Schittkowski (1980) and Hock, Schittkowski (1981) show their superiority over other mathematical programming algorithms under the assumptions mentioned above.

The key idea is to approximate also second order information to get a fast final convergence speed. Thus we define a quadratic approximation of the Lagrange function  $L(x,u)$  and an approximation of the Hessian matrix of  $L(x_k, u_k)$  by a so-called quasi-Newton matrix  $B_k$ . Then we get the quadratic programming subproblem

$$\begin{aligned} \min \quad & \frac{1}{2} d^T B_k d + \nabla f(x_k)^T d \\ d \in R^n: \quad & \nabla g_j(x_k)^T d + g_j(x_k) = 0 \quad , j = 1, \dots, m_g, \\ & \nabla g_j(x_k)^T d + g_j(x_k) \geq 0 \quad , j = m_g + 1, \dots, m, \\ & x_l - x_k \leq d \leq x_u - x_k. \end{aligned}$$

To stabilize the algorithm particularly when starting from a bad initial guess  $x_0$ , and to ensure global convergence, an additional line search is performed, i.e. a steplength computation to accept a new iterate

$$x_{k+1} = x_k + \alpha_k d_k$$

for a suitable  $\alpha_k$  only if  $x_{k+1}$  satisfies a descent property with respect to a solution  $d_k$  of the quadratic programming subproblem. Following the approach of Schittkowski (1983), e.g., we need also a simultaneous line search with respect to the multiplier approximations and use an augmented Lagrangian merit function to determine the line search parameter. Moreover certain safeguards must be taken into

account to prevent that the linearized constraints are contradictory.

The update of the matrix  $B_k$  can be performed by standard techniques known from unconstrained optimization. In our case, the BFGS-method is applied, a numerically simple rank-2 correction starting from the identity matrix. Only the difference vectors  $x_{k+1} - x_k$  and  $\tilde{N} L(x_{k+1}, u_k) - \tilde{N} L(x_k, u_k)$  are required. Under some safeguards it is possible to guarantee that all matrices  $B_k$  are positive definite.

Among the most attractive features of SQP methods is the superlinear convergence speed in the neighborhood of a solution given by

$$\|x_{k+1} - x^*\| \leq \gamma_k \|x_k - x^*\|$$

where  $\gamma_k$  is a sequence of positive numbers converging to zero and  $x^*$  an optimal solution.

To understand this convergence behavior, replace  $B_k$  by the true Hessian of the Lagrangian function and consider only equality constraints. Then it is very easy to see that an SQP method is identical to Newton's method for solving the nonlinear system of  $n+m$  equations in  $n+m$  unknowns given by the Kuhn-Tucker conditions. This result can be extended to inequality constraints as well. Then we get immediately the quadratic convergence behavior and, if we replace  $B_k$  again by its approximation, the weaker superlinear convergence rate as proved in the references mentioned above.

---

### 3. EXTENSIONS

SQP methods allow the solution of a wide range of nonlinear programming problems in an efficient and reliable way. Either implicitly or proceeding from simple modifications of the underlying optimization problem, a much larger class of different nonlinear programming problems can be solved by NLPQL. A few are to be mentioned subsequently, for more details see Schittkowski (1988,1994).

#### 3.1 Linear Constraints

If some or all of the constraints are linear, we note that they are not changed at all during the constraint linearization process, i.e. they are passed directly to the algorithm solving the quadratic programming subproblem. A particular advantage is that once a linear constraint or bound is satisfied, then this

restriction remains satisfied during the whole iteration process.

### 3.2 System of Nonlinear Equations

In case of finding a solution of a system of nonlinear equations either with or without inequalities, i.e. when trying to find a solution of the system

$$x \in R^n: g_j(x) = 0, j = 1, \dots, m,$$

a nonlinear programming algorithm can be applied directly by defining the above equations as constraints and by adding an artificial objective function. A typical function to be minimized, could be of the form  $f(x) = x^T x$ , if a minimum-norm solution in case of an underdetermined system or inequalities is required.

### 3.3 Problems with Very Many Constraints

There exists an extension of NLPQL for solving nonlinear programming problems where the number of constraints is very large compared to the number of variables, see Schittkowski (1991). Typical application is semi-infinite optimization, when constraints possess an additional parameter varying in a given continuous set, and where we discretize these constraints at a finite number of break points.

In this situation an active set strategy is applied to prevent a too large number of constraints in the quadratic programming sub-problem. One reason is lack of memory, another one the dangerous generation of degenerate or nearly degenerate restrictions that prevent a stable solution of the sub-problem.

### 3.4 Least Squares Optimization

We consider a mathematical model in form of a least squares problem, i.e. the problem of minimization of a sum of squares of nonlinear functions in the following form:

$$\begin{aligned} \min \quad & \sum_{i=1}^l f_i(x)^2 \\ x \in R^n: \quad & g_j(x) = 0 \quad , j = 1, \dots, m_g \quad , \\ & g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m \quad , \\ & x_l \leq x \leq x_u \quad . \end{aligned}$$

Here we assume that the parameter vector  $x$  is  $n$ -dimensional and that all nonlinear functions are continuously differentiable with respect to  $x$ . Upper and lower bounds are treated independently from the remaining constraints.

The most important application is parameter estimation, where one model function is available with an additional variable called *time* . Proceeding now from  $l$  measurements of the form

$$(t_i, y_i) \quad , i = 1, \dots, l$$

We get the above least squares formulation by

$$f_i(x) = w_i (h(x, t_i) - y_i)$$

together with a real valued model function  $h(x, t)$  and suitable non-negative weighting factors.

Then the underlying idea is to minimize the distance between the model function at certain time points and the corresponding measurement values. This distance is denoted the residual of the problem. In the ideal case the residuals are zero indicating a perfect fit of the model function by the measurements.

Usually least squares problems are solved by the Gauss-Newton-algorithm or any of its numerous variants. If any of these special purpose implementations is not available, it is recommended to transform the original problem into a general nonlinear programming problem by introducing additional variables and equality constraints, i.e.

$$\begin{aligned} \min \quad & \sum_{i=1}^l z_i^2 \\ x \in R^n, z \in R^l: \quad & f_i(x) - z_i = 0 \quad , i = 1, \dots, l \quad , \\ & g_j(x) = 0 \quad , j = 1, \dots, m_g \quad , \\ & g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m \quad , \\ & x_l \leq x \leq x_u \quad . \end{aligned}$$

Typical features of a Gauss-Newton and quasi-Newton method are retained, see Schittkowski (1988).

The resulting optimization problem can be solved by NLPQL as in the general situation. The proposed method is implemented within an interactive system to solve parameter estimation problems in dynamical systems, see Schittkowski (1996).

EASY\_OPT.MMA001

### 3.5 $L_1$ - Optimization

In this case the model consists of minimizing the sum of absolute function values subject to general nonlinear equality or inequality constraints, i.e.

$$\begin{aligned} \mathbf{min} \quad & \sum_{i=1}^l |f_i(x)| \\ x \in R^n: \quad & g_j(x) = 0 \quad , j = 1, \dots, m_g \quad , \\ & g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m \quad , \\ & x_l \leq x \leq x_u \quad . \end{aligned}$$

Again we assume that the parameter vector  $x$  is  $n$ -dimensional and that all nonlinear functions are continuously differentiable with respect to  $x$ . Upper and lower bounds are handled separately. However the problem is not differentiable and it is proposed to introduce  $l$  additional variables

$$z = (z_1, \dots, z_l)^T$$

and  $2l$  additional inequality constraints of the form

$$\begin{aligned} f_i(x) + z_i &\geq 0 \quad , i = 1, \dots, l \\ -f_i(x) + z_i &\geq 0 \quad , i = 1, \dots, l \end{aligned}$$

to get an equivalent formulation

$$\begin{aligned}
& \min \sum_{i=1}^l z_i \\
& f_i(x) + z_i \geq 0 \quad , i = 1, \dots, l, \\
x \in R^n, z \in R^l: & -f_i(x) + z_i \geq 0 \quad , i = 1, \dots, l, \\
& g_j(x) = 0 \quad , j = 1, \dots, m_g, \\
& g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m, \\
& x_l \leq x \leq x_u.
\end{aligned}$$

This problem is a smooth one and can be solved by NLPQL.

### 3.6 $L\infty$ - Optimization

In this case we want to minimize the maximum of absolute function values, i.e.

$$\begin{aligned}
& \min \max \{|f_i(x)|, i = 1, \dots, l\} \\
x \in R^n: & g_j(x) = 0 \quad , j = 1, \dots, m_g, \\
& g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m, \\
& x_l \leq x \leq x_u.
\end{aligned}$$

It is assumed that all nonlinear functions are continuously differentiable with respect to  $x$ . Again this problem is not smooth and we transform it into a smooth one by introducing one additional variable  $z$  and  $2l$  additional inequality constraints of the form

$$\begin{aligned}
& f_i(x) + z \geq 0 \quad , i = 1, \dots, l, \\
& -f_i(x) + z \geq 0 \quad , i = 1, \dots, l,
\end{aligned}$$

to get the equivalent problem

$$\begin{aligned}
& \min z \\
& f_i(x) + z \geq 0 \quad , i = 1, \dots, l, \\
& -f_i(x) + z \geq 0 \quad , i = 1, \dots, l, \\
x \in R^n, z \in R: & \quad g_j(x) = 0 \quad , j = 1, \dots, m_e, \\
& \quad g_j(x) \geq 0 \quad , j = m_e + 1, \dots, m, \\
& \quad x_l \leq x \leq x_u.
\end{aligned}$$

This problem is smooth and can be solved by the SQP-algorithm NLPQL.

### 3.7 Min - Max - Optimization

The problem consists of minimizing the maximum of differentiable functions subject to equality and inequality constraints:

$$\begin{aligned}
& \min \max \{f_i(x), i = 1, \dots, l\} \\
x \in R^n: & \quad g_j(x) = 0 \quad , j = 1, \dots, m_e, \\
& \quad g_j(x) \geq 0 \quad , j = m_e + 1, \dots, m, \\
& \quad x_l \leq x \leq x_u.
\end{aligned}$$

Again we introduce one additional variable  $z$ , but only  $l$  additional inequality constraints of the form

$$-f_i(x) + z \geq 0 \quad , i = 1, \dots, l$$

to get the equivalent problem

$$\begin{aligned}
& \min z \\
& -f_i(x) + z \geq 0 \quad , i = 1, \dots, l, \\
x \in R^n, z \in R: & \quad g_j(x) = 0 \quad , j = 1, \dots, m_g, \\
& \quad g_j(x) \geq 0 \quad , j = m_g + 1, \dots, m, \\
& \quad x_l \leq x \leq x_u.
\end{aligned}$$

This problem is smooth and can be solved by NLPQL.

---

#### 4. INPUT PARAMETERS AND TOLERANCES

NLPQL represents a very robust implementation of a sequential quadratic programming algorithm and requires only very few user-provided parameters. All other parameters, solution tolerances or options usually required by a nonlinear programming code, are set internally. Only two parameters are essential:

*MAXIT* : Maximum number of iterations

Any reasonably large number of iterations can be used, since we suppose that NLPQL is executed to solve small-scale, but eventually highly nonlinear optimization problems. However, if function and gradient evaluations provided by iSIGHT, are expensive, it is recommended to perform only a few iterations and to perform a restart if the final iterate is not acceptable.

*ACC*: Final termination accuracy

Several termination criteria are implemented in NLPQL to prevent scaling effects as much as possible, and to stop the algorithm as soon as the desired tolerance is reached. For more details see Schittkowski

(1985/86). If gradients are evaluated within machine precision, a relatively small parameter is acceptable, e.g. 1.0E-8 to 1.0E-11. Otherwise  $ACC$  should not be smaller than  $h$ , where  $h$  is the stepsize applied for a numerical gradient approximation by forward differences, or the accuracy of objective function and constraint evaluation.

---

## 5. ERROR MESSAGES

The following termination or error messages, respectively, are displayed by NLPQL:

IFAIL = 0 :	Optimality conditions are satisfied.
IFAIL = 1 :	The algorithm terminated after MAXIT iterations.
IFAIL = 2 :	The algorithm computed an uphill search direction.
IFAIL = 3 :	Underflow occurred when determining a new approximation of the Hessian matrix of the Lagrangian function.
IFAIL = 4 :	More than MAXFUN function evaluations during line search.
IFAIL = 7 :	The search direction is close to zero, but the current iterate is still infeasible.
IFAIL > 10:	The solution of the quadratic sub-problem has been terminated with an error message IFQL > 0 and IFAIL is set to IFAIL = IFQL + 10 .

Some of the termination reasons depend on the accuracy supplied by iSIGHT for approximating gradients. If we assume that all functions and gradients are computed within machine precision and that the implementation is correct, there remain only the following possibilities that could cause the error messages mentioned:

1. The termination parameter  $ACC$  is too small, so that the numerical algorithm plays around with round-off errors without being able to improve the solution. Especially the Hessian approximation of the Lagrangian function will become unstable in this case. A straightforward remedy is to restart the optimization cycle again with a larger stopping tolerance.

2. The constraints are contradicting, i.e. the set of feasible solutions is empty. There is no way to find out, whether a general nonlinear and nonconvex set possesses a feasible point or not. Thus the nonlinear programming algorithms will proceed until running in any of the mentioned error situations. In this case there the correctness of the model is to be checked very carefully.
3. Constraints are feasible, but some of them there are degenerate, e.g. if some of the constraints are redundant. One should know that SQP algorithms require that the so-called constraint qualification is satisfied, i.e. that gradients of active constraints are linearly independent at each iterate and in a neighborhood of the optimal solution. Also in this situation it is recommended to check the formulation of the model very carefully.

However some of the error situations do also occur, if because of wrong or non-accurate gradients the quadratic programming sub-problem does not yield a descent direction for the underlying merit function. In this case one should try to improve the accuracy of function evaluations, scale the model functions in a proper way, or start the algorithm from other initial values.

---

## 6. REFERENCES

1. Han S.-P. (1976): *Superlinearly convergent variable metric algorithms for general nonlinear programming problems*, Mathematical Programming, Vol. 11, 263-282
2. Han S.-P. (1977): *A globally convergent method for nonlinear programming*, Journal of Optimization Theory and Applications, Vol. 22, 297-305
3. Hock W., Schittkowski K. (1981): *Test Examples for Nonlinear Programming Codes*, Lecture Notes in Economics and Mathematical Systems, Vol. 187, Springer
4. Powell M.J.D. (1978a): *A fast algorithm for nonlinearly constrained optimization calculations*, in:

Numerical Analysis, G.A. Watson ed., Lecture Notes in Mathematics, Vol. 630, Springer

5. Powell M.J.D. (1978b): *The convergence of variable metric methods for nonlinearly constrained optimization calculations*, in: Nonlinear Programming 3, O.L. Mangasarian, R.R. Meyer, S.M. Robinson eds., Academic Press
6. Schittkowski K. (1980): *Nonlinear Programming Codes*, Lecture Notes in Economics and Mathematical Systems, Vol. 183, Springer
7. Schittkowski K. (1983): *On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search function*, Optimization, Vol. 14, 197-216
8. Schittkowski K. (1985/86): *NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems*, Annals of Operations Research, Vol. 5, 485-500
9. Schittkowski K. (1987): *More Test Examples for Nonlinear Programming*, Lecture Notes in Economics and Mathematical Systems, Vol. 182, Springer
10. Schittkowski K. (1988): *Solving nonlinear least squares problems by a general purpose SQP-method*, in: *Trends in Mathematical Optimization*, K.-H. Hoffmann, J.-B. Hiriart-Urruty, C. Lemarechal, J. Zowe eds., International Series of Numerical Mathematics, Vol. 84, Birkhaeuser
11. Schittkowski K. (1991): *Solving nonlinear programming problems with very many constraints*, Report No. 294, DFG-Schwerpunktprogramm 'Anwendungsbezogene Optimierung und Steuerung', Mathematical Institute, University of Bayreuth
12. Schittkowski K. (1994): *Easy-to-use optimization programs with automatic differentiation*, Report, Mathematical Institute, University of Bayreuth
13. Schittkowski, K. (1996): *EASY-FIT: Parameter estimation in dynamic systems, User's Guide*, Mathematical Institute, University of Bayreuth