DFVLR Deutsche Forschungs- und
Versuchsanstalt
für Luft- und Raumfahrt

Forschungsbericht

A Software Package for Sequential Quadratic Programming

Dieter Kraft

DFVLR
Institut für Dynamik der Flugsysteme
Oberpfaffenhofen

33 pages
42 references

DFVLR-FB 88-28

Manuskript eingereicht am 28. Juli 1988

A Software Package for Sequential Quadratic Programming

Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt
Forschungsbereich Flugmechanik/Flugführung
Institut für Dynamik der Flugsysteme
Abteilung Regelung
Oberpfaffenhofen, D–8031 Weßling

Oberpfaffenhofen, im Juli 1988

Institutsleiter:
Prof. Dr.-Ing. J. Ackermann

Verfasser:
Dr.-Ing. D. Kraft

Abteilungsleiter:
Prof. Dr.-Ing. G. Grübel

Nonlinear Optimization, Quadratic Optimization, Software for Sequential Quadratic Programming

A Software Package for Sequential Quadratic Programming

Summary

The algorithmic principles of Sequential Quadratic Programming are summarized and motivated. The basic algorithms of Quadratic Programming (primal, primal/dual and dual approach) are given. The parameters of the Software Package are described.

Nichtlineare Optimierung, quadratische Optimierung, Software zur Sequentiellen Quadratischen Optimierung

Ein Software-Paket zur Sequentiellen Quadratischen Optimierung

Übersicht

Algorithmische Prinzipien zur Sequentiellen Quadratischen Optimierung werden zusammenfassend dargestellt und motiviert. Die Kernalgorithmen der Quadratischen Optimierung (primaler, primal/dualer und dualer Ansatz) werden erläutert. Die Parameter des Software-Pakets werden beschrieben.

Contents

Chapter 1

Introduction

Mathematical programming is a powerful aid in designing feedforward and feedback controllers for nonlinear dynamic systems. In the first case the inherent optimal control problem is transformed to a constrained nonlinear programming problem by discretising the control function and solving sequences of initial value problems [23]. In the second case various control specifications are defined by a performance vector which will be minimized with respect to the controller parameters within some given controller structure [40]. For the resulting vector programming problem PARETO-optimal parameters are found by a minmax strategy. The minmax problem is solved as an inequality-constrained nonlinear program [15]. These are only two examples within a wide variety of practical applications in complex industrial environments; other applications can be found in large-scale circuit design [30] or in the field of structural mechanics [20], for instance.

All of the above problem classes require an efficient and robust software package for the solution of the general constrained nonlinear optimization problem. The extensive tests of SCHITTKOWSKI [36] on a wide range of test examples [21] have shown that *sequential quadratic programming* is the most efficient method (in terms of function evaluations and computer time) to solve the above problem.

This report describes a software package developed and implemented at the DFVLR for the solution of optimization problems in control system analysis and design. This package uses several different subroutines for the calculation of the search direction (quadratic subproblem) according to either primal or dual methods.

The report is organized as follows. In chapter 2 the iterative algorithm is described and motivated. In chapter 3 three different approaches to solve the quadratic subproblem are summarized, a primal method based on the original problem formulation a primal/dual method based on a transformation into a constrained least squares problem, and a dual method based on an active set strategy from the outside of the feasible domain. In appendix A the software organization is explained. A computational comparison will be reported in the near future.

Chapter 2

Sequential Quadratic Programming

2.1 The Problem

2.1.1 Problem NLP

Sequential quadratic programming is known as to be the most efficient computational method to solve the general nonlinear programming problem

$$\textbf{(NLP):} \quad \min_{x \in R^n} f(x) \tag{2.1}$$

subject to

$$g_j(x) = 0, \quad j = 1, \ldots, m_e, \tag{2.2}$$

$$g_j(x) \geq 0, \quad j = m_e + 1, \ldots, m, \tag{2.3}$$

$$x_l \leq x \leq x_u, \tag{2.4}$$

for a local minimum, where the problem functions $f : R^n \to R^1$ and $g : R^n \to R^m$ are assumed to be continuously differentiable and to have no specific structure. The size of the problem should only be moderately large, $\hat{m} \leq n \leq 200$, for instance, where $\hat{m}$ is the number of active constraints at the solution.

2.1.2 Minimax Problem

The minimax problem

$$\textbf{(MINIMAX):} \quad \min_{x \in R^n} \max_{i \in L} \phi_i(x), \quad L := \{1, \ldots, l\}, \tag{2.5}$$

subject to

$$g_j(x) = 0, \quad j = 1, \ldots, m_e, \tag{2.6}$$

$$g_j(x) \geq 0, \quad j = m_e + 1, \ldots, m, \tag{2.7}$$

$$x_l \leq x \leq x_u, \tag{2.8}$$

with $\phi : R^n \to R^l$ sufficiently smooth, can be numerically reduced to problem (NLP) by introducing an extra variable x_{n+1} as an upper bound on the elements of the vector ϕ. This upper bound will then be minimized subject to $\phi_i \leq x_{n+1}$, $\forall\, i \in L$.

2.1.3 Notation

The following notation will be used: $\hat{g}(x)$ indicates those t components of the inequality constraints which are active, i.e. satisfied as equalities, at the solution. Similarly, let $\hat{a}_j(x) := \nabla \hat{g}_j^T(x)$, and $\hat{G}_j(x) := \nabla_x^2 \hat{g}_j(x)$. The matrix $A(x)$ is composed of the m columns $a_j(x), j = 1, \ldots, m$, and a similar notation holds for $\hat{A}(x)$.

Throughout the paper all gradient vectors are row vectors and all other vectors are column vectors; $\|x\|$ is used to denote their Euklidian vector norm $\sqrt{\sum x_i^2}$. $\|M\|$ is the corresponding induced matrix norm, $\max(\|Mx\| : \|x\| = 1)$. The machine precision is denoted by ϵ.

2.2 The Basic Algorithm

Problem (NLP) will be solved iteratively; starting with a given vector of parameters x^0, the $(k+1)^{st}$ iterate x^{k+1} will be obtained from x^k by the step

$$x^{k+1} := x^k + \alpha^k d^k, \tag{2.9}$$

where d^k is the search direction within the k^{th} step and α^k is the step length.

2.2.1 The Search Direction

The search direction is determined by a quadratic programming subproblem, which is formulated by a quadratic approximation of the LAGRANGE function of problem (NLP)

$$L(x, \lambda) = f(x) - \sum_{j=1}^{m} \lambda_j g_j(x), \tag{2.10}$$

and a linear approximation of the constraints g_j. This problem is of the following standard form of quadratic programming:

$$\textbf{(QP):} \quad \min_{d \in R^n} \frac{1}{2} d^T B^k d + \nabla f(x^k) d \tag{2.11}$$

subject to

$$\nabla g_j(x^k) d + g_j(x^k) = 0, \quad j = 1, \ldots, m_e, \tag{2.12}$$

$$\nabla g_j(x^k)d + g_j(x^k) \geq 0, \quad j = m_e + 1, \ldots, m. \tag{2.13}$$

Note again that in the above formulation the gradients $\nabla f(x)$ and $\nabla g_j(x)$ are row vectors.

The choice of the above search direction has been proposed by WILSON [42] already in 1963, with

$$B := \nabla^2_{xx} L(x, \lambda). \tag{2.14}$$

2.2.2 The Step Size

It will be demonstrated in section 2.2 that this direction has a strong analogy to the search direction in NEWTON's method for solving systems of nonlinear equations. Like in NEWTON's method this direction is optimal with stepsize $\alpha = 1$ for general nonlinear functions near a local optimum; but the stepsize has to be modified for vectors x^k far from the optimum, to guarantee global convergence of the method.

HAN [19] has proven that a one-dimensional minimization of the non-differentiable exact penalty function

$$\phi(x; \varrho) := f(x) + \sum_{j=1}^{m_e} \varrho_j |g_j(x)| + \sum_{j=m_e+1}^{m} \varrho_j |g_j(x)|_- \tag{2.15}$$

with $|g_j(x)|_- := |min(0, g_j(x)|$, as a *merit function* $\varphi : R^1 \rightarrow R^1$

$$\varphi(\alpha) := \phi(x^k + \alpha d^k), \tag{2.16}$$

with x^k and d^k fixed, leads to a stepsize α guaranteeing global convergence for values of the penalty parameters ϱ_j greater than some lower bound. POWELL [31] proposed to update the penalty parameters according to

$$\varrho_j := max(\frac{1}{2}(\varrho_j^- + |\mu_j|), |\mu_j|), \quad j = 1, \ldots, m, \tag{2.17}$$

where μ_j denotes the LAGRANGE multiplier of the j^{th} constraint in the quadratic subproblem and ϱ_j^- is the j^{th} penalty parameter of the previous iteration, starting with some $\varrho_j^0 = 0$, for instance.

To overcome possible difficulties in the line search of the non-differentiable merit function SCHITTKOWSKI [21] substituted equation (2.17) by the differentiable augmented LAGRANGE function

$$\begin{aligned} \Phi(x, \lambda; \varrho) \;&:=\; f(x) - \sum_{j=1}^{m_e} (\lambda_j g_j(x) - \frac{1}{2}\varrho_j g_j(x)^2) \\ &-\; \sum_{j=m_e+1}^{m} \begin{cases} \lambda_j g_j(x) - \frac{1}{2}\varrho_j g_j(x)^2, & \text{if } g_j(x) \leq \lambda_j/\varrho_j, \\ \frac{1}{2}\lambda_j^2/\varrho_j, & \text{otherwise,} \end{cases} \end{aligned} \tag{2.18}$$

a formula first introduced by ROCKAFELLAR [35] within the context of constrained optimization.

2.2.3 Inconsistent Constraints

It might be possible that the constraints in subproblem (QP) become inconsistent in the course of the iterations, although those of the original problem (NLP) are solvable. In order to overcome this difficulty, an additional variable δ is introduced into the quadratic subproblem as in POWELL [31], leading to an (n+1)–dimensional subproblem with consistent constraints:

$$\min_{d\in R^n} \frac{1}{2} d^T B^k d + \nabla f(x^k) d + \frac{1}{2}\rho^k (\delta^k)^2 \tag{2.19}$$

subject to

$$\nabla g_j(x^k) d + \delta^k g_j(x^k) = 0, \quad j = 1, \ldots, m_e, \tag{2.20}$$

$$\nabla g_j(x^k) d + \delta_j^k g_j(x^k) \geq 0, \quad j = m_e + 1, \ldots, m. \tag{2.21}$$

$$0 \leq \delta^k \leq 1, \tag{2.22}$$

where δ_j^k has the value

$$\delta_j^k := \begin{cases} 1, & \text{if } g_j(x^k) > 0 \\ \sigma^k, & \text{otherwise} \end{cases} \tag{2.23}$$

and it is made as large as possible, subject to the condition (2.22). Obviously the initial direction $(d^0, \delta^0)^T = (0, \ldots, 0, 1)^T$ satisfies the constraints (2.21–2.22), therefore it can be used as starting value for the augmented quadratic subproblem.

2.2.4 Update of the B–Matrix

It is a crucial requirement for the computational efficiency in practical applications of sequential quadratic programming not to evalute the matrix B^k as in equation (2.14) in every iteration, but to use only first-order information to approximate the HESSE–matrix of the LAGRANGE–function. This has been customary for unconstrained optimization since a long time, known as quasi-NEWTON methods [10]. A very popular update has become the BFGS–formula. Reference [8] gives a comprehensive overview on numerical methods for unconstrained optimization and the closely related field of nonlinear equations.

The analogue formula for the constrained case has been developed by POWELL [31]. The theoretical justifications of this so-called variable metric update can be found in reference [32]. The difficulty in constrained optimization is, that unlike in unconstrained optimization B^k need not remain positive definite for a positive definite initial estimate. Therefore POWELL proposed the following modification:

$$B^{k+1} := B^k + \frac{q^k (q^k)^T}{(q^k)^T s^k} - \frac{B^k s^k (s^k)^T B^k}{(s^k)^T B^k s^k}, \tag{2.24}$$

with

$$s^k := x^{k+1} - x^k = \alpha^k d^k, \tag{2.25}$$

and

$$q^k := \theta^k \eta^k + (1 - \theta^k) B^k s^k, \tag{2.26}$$

where η^k is the difference in gradients of the LAGRANGE function

$$\eta^k := \nabla_x L(x^{k+1}, \lambda^k) - \nabla_x L(x^k, \lambda^k), \tag{2.27}$$

and θ^k is chosen as

$$\theta^k := \begin{cases} 1, & \text{if } (s^k)^T \eta^k \geq 0.2 (s^k)^T B^k s^k, \\ \frac{0.8 (s^k)^T B^k s^k}{(s^k)^T B^k s^k - (s^k)^T \eta^k}, & \text{otherwise,} \end{cases} \tag{2.28}$$

which guarantees the condition

$$(s^k)^T q^k \geq 0.2 (s^k)^T B^k s^k, \tag{2.29}$$

which holds B^{k+1} positive definite within the linear manifold defined by the tangent planes to active constraints at x^{k+1}. The multipliers λ are taken to be those of the quadratic subproblem at it's solution. The choice of conditions (2.25 - 2.28) guarantees the updated B-matrix (2.24) to remain positive definite for an arbitrary initial estimate of this matrix.

It is important to use for both the above rank-one updates (2.24) recurrence formulae which need only $O(n^2)$ flops. One of these is the *composite t-method* of FLETCHER and POWELL described in [9], see also [13]. This method updates the factors

$$L^k D^k (L^k)^T =: B^k \tag{2.30}$$

instead of B^k itself, where $L = [l_1, \ldots, l_n]$ is a lower triangular matrix with unit diagonal entries and $D = diag(d_1, \ldots, d_n)$ is a diagonal matrix. Of course, the matrix D does not occur explicitely, their diagonal elements are stored in the diagonal elements of L. Suppose the factors of the following modified matrix $\tilde{B}$

$$\begin{aligned} \tilde{L} \tilde{D} \tilde{L}^T &= \tilde{B} = B + \sigma z z^T \\ &= \sum_i^n \tilde{l}_i \tilde{d}_i \tilde{l}_i^T \end{aligned} \tag{2.31}$$

are sought, then the following algorithm implements the

composite-t method: (2.32)

(1) **if** $\sigma = 0$ **then** terminate;
else set $t_1 := 1/\sigma$ and $z^{(1)} := z$,
(2) **if** $\sigma > 0$ **then** go to (6),
(3) **else** solve $Lv = z$ to get v,
(4) **for** $i = 1, \ldots, n$ **do** $t_{i+1} = t_i + v_i^2/d_i$,

(5) **if** any $t_{i+1} \geq 0$ **then** recompute t_{i+1}:
e.g. **if** $t_{j+1} \geq 0$ **then** set $t_{j+1} := \varepsilon/\sigma$,
for $i = j, j-1, \ldots, 1$ **do** $t_i = t_{i-1} - v_i^2/d_i$,
(6) **for** $i = 1, \ldots, n$ **do**
$v_i = z_i^{(i)}$,
if $\sigma > 0$ **then** $t_{i+1} = t_i + v_i^2/d_i$,
$\alpha_i = t_{i+1}/t_i$,
$\tilde{d}_i = \alpha_i d_i$,
if $i = n$ **then** stop;
$\beta_i = (v_i/d_i)/t_{i+1}$,
$z^{(i+1)} = z^{(i)} - v_i l_i$,
if $\alpha_i > 4$ **then** $\tilde{l}_i = (t_i/t_{i+1}) l_i + \beta_i z_i^{(i)}$,
else $\tilde{l}_i = l_i + \beta_i z^{(i+1)}$.

2.2.5 Least Squares Subproblems

SCHITTKOWSKI [38] has proposed to replace the quadratic programming subproblem (QP) by a linear least squares subproblem, using a stable LDL^T-factorization of the matrix B:

$$\textbf{(LSEI):} \quad \min_{d \in R^n} \|(D^k)^{1/2}(L^k)^T d + (D^k)^{-1/2}(L^k)^{-1} \nabla f(x^k)\| \tag{2.33}$$

subject to

$$\nabla g_j(x^k) d + g_j(x^k) = 0, \quad j = 1, \ldots, m_e, \tag{2.34}$$

$$\nabla g_j(x^k) d + g_j(x^k) \geq 0, \quad j = m_e + 1, \ldots, m. \tag{2.35}$$

This subproblem can be solved with the linear least squares software of LAWSON and HANSON [26].

2.3 Motivation of SQP Methods

2.3.1 Optimality Conditions

SQP-based methods are motivated as methods analog to NEWTON's method for solving systems of nonlinear equations. First we need the KUHN-TUCKER conditions for problem (QP), a set of necessary optimality conditions [28,2]:

$$\textbf{(KT):} \quad \nabla_x L(x, \lambda) = 0, \tag{2.36}$$

$$g_j(x) = 0, \quad j = 1, \ldots, m_e, \tag{2.37}$$

$$g_j(x) \geq 0, \quad j = m_e + 1, \ldots, m, \tag{2.38}$$

$$\lambda_j(x) \geq 0, \quad j = m_e + 1, \ldots, m, \tag{2.39}$$

$$g_j(x)\lambda_j(x) = 0, \quad j = m_e + 1, \ldots, m, \tag{2.40}$$

To start with the arguments we consider equality-constrained problems (2.1–2.2) first, inequality constrained problems are then treated by introducing the notion of active constraint sets.

From the the necessary conditions of the equality-constrained problem it follows that the condition

$$e(x, \lambda) := \begin{pmatrix} \nabla f(x) - A(x)\lambda \\ g(x) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{2.41}$$

be satisfied, where

$$A^T(x) := \begin{pmatrix} \nabla g_1(x) \\ \vdots \\ \nabla g_{m_e}(x) \end{pmatrix} \tag{2.42}$$

is the $n \times m_e$ JACOBI matrix of the equality constraints, which is assumed to be of full rank at the corresponding x.

Now NEWTON's method is applied to the solution of the system of $n + m_e$ (usually nonlinear) equations (2.41) in the $n + m_e$ unknowns $(x, \lambda)^T$. Let $(x^k, \lambda^k)^T$ be the k^{th} iterate of the NEWTON process, which is defined by

$$\nabla e(x^k, \lambda^k) \begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix} = -e(x^k, \lambda^k) \tag{2.43}$$

$$\begin{pmatrix} x^{k+1} \\ \lambda^{k+1} \end{pmatrix} := \begin{pmatrix} x^k \\ \lambda^k \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix}, \tag{2.44}$$

with

$$\nabla e(x^k, \lambda^k) = \begin{pmatrix} H(x^k, \lambda^k) & -A(x^k) \\ A(x^k)^T & 0 \end{pmatrix} \tag{2.45}$$

the JACOBian of $e(x, \lambda)$ with respect to $(x, \lambda)^T$ within the k^{th} iteration, and

$$H(x^k, \lambda^k) = \nabla_x^2 f(x^k) - \sum_{j=1}^{m_e} \lambda_j^k g_j(x^k) \tag{2.46}$$

the HESSE matrix with respect to x of the LAGRANGE function $L(x, \lambda)$. Explicitely written equations (2.43) and (2.44) become

$$\begin{pmatrix} H(x^k, \lambda^k) & -A(x^k) \\ A(x^k)^T & 0 \end{pmatrix} \begin{pmatrix} d \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} \nabla f(x^k) \\ g(x^k) \end{pmatrix}, \tag{2.47}$$

$$x^{k+1} := x^k + d. \tag{2.48}$$

Methods to solve equation (2.47) based on certain matrix decomposition techniques are given in reference [29]. Recent local convergence results can be obtained from FONTECILLA [12].

2.3.2 Nonlinear Equations and Quadratic Programming

It can be verified by direct evaluation of (KT) that the conditions (2.47) and (2.48) are equivalent to the KUHN-TUCKER conditions of the quadratic programming problem with pure equality constraints (2.11-2.12):

$$\min_{d\in R^n} \frac{1}{2} d^T B^k d + \nabla f(x^k) d \tag{2.49}$$

subject to

$$\nabla g_j(x^k) d + g_j(x^k) = 0, \quad j = 1, \ldots, m_e, \tag{2.50}$$

with $B^k = H(x^k, \lambda^k)$.

This equivalence turns out to be the motivation for the choice of the search direction (2.11–2.13). This is especially convenient in the inequality constrained case, where no practical implementation exists to solve the corresponding problem (2.47–2.48).

2.3.3 Active Set Strategy

A complete solution of the equality-constrained quadratic progamming problem (2.49–2.50) will be given in the next chapter. Here, a technique for solving the general QP-problem with inequalities included will be presented in the context of primal methods. The B-matrix will be assumed to be positive definite throughout; the slightly more complex case when the B-matrix is indefinite is treated by GILL and MURRAY [14] and by BUNCH and KAUFMAN [4].

First suppose a *feasible* iterate x^k is given, that means a point satisfying the constraints (2.12) and (2.13). If such a point is not available, it has to be provided by a suitable starting procedure, e.g. a phase I method of the simplex algorithm. Now the *active set* is an index set I_a^k consisting of the indices of *all* say t^k active constraints at the point x^k:

$$I_a^k := \{j = 1, \ldots, m_e\} \cup \{j = m_e + 1, \ldots, m |\ a_j^T(x^k) d - c_j^k = 0\}. \tag{2.51}$$

The *inactive* inequalities $a_j^T(x^k) d - c_j^k > 0$ will be temporarily disregarded.

Now the equality constrained quadratic programming problem with active constraints

$$\textbf{(EQP):} \quad \min_{d\in R^n} \frac{1}{2} d^T B^k d + \nabla f(x^k) d \tag{2.52}$$

subject to

$$\nabla g_j(x^k) d + g_j(x^k) = 0, \quad j = 1, \ldots, t^k, \tag{2.53}$$

must be solved; let the solution be $(d^k, \lambda^k)^T$. Next a step is taken into this direction according to (2.9)

$$x^{k+1} := x^k + \alpha^k d^k, \tag{2.54}$$

under the restrictions

$$f(x^k + \alpha^k d^k) < f(x^k), \tag{2.55}$$

and

$$\alpha^k \leq \hat{\alpha}^k = \begin{cases} \min \frac{c_j - a_j^T x^k}{a_j^T d^k}, & \text{if } a_j^T d^k < 0 \text{ for some } j \notin I_a^k, \\ +\infty, & \text{if } a_j^T d^k \geq 0 \text{ for all } j \notin I_a^k. \end{cases} \tag{2.56}$$

Note that $\hat{\alpha}^k$ will have a positive value under the conditions of (2.56), as index j is not within the active set.

If $a_j^T d^k \geq 0$, any positive step along d^k will not violate inactive constraint j. On the other hand, if $a_j^T d^k < 0$, there is a step α_j which *activates* constraint j: $c_j - a_j^T(x^k + \alpha_j d^k) = 0$. This is the reasoning behind condition (2.56).

As in unconstrained optimization this algorithm requires sufficient decrease of the cost function, stated in relation (2.55), as to prove it's global convergence. For further information see reference [15, pages 168-170].

Conditions 2.55 and 2.56 may leave the active set unaltered or may enlarge it, according to whether $\alpha^k < \hat{\alpha}^k$ or $\alpha^k = \hat{\alpha}^k$. As a third possibility the deletion of a constraint has to be considered. This will be the case when the optimality conditions, especially (2.39)

$$\lambda_j(x) \geq 0, \quad j = m_e + 1, \ldots, m, \tag{2.57}$$

are not satisfied for the active inequality constraints. This means that the active set is notyet correct. The minimum of the cost function is sought in a subspace of too small dimension. If the LAGRANGE multipliers of the current solution of (2.52) and (2.53) are assumed to be sufficient accurate estimates, then condition (2.57)can be used as a criterion to delete a binding constraint, for instance that one with the smallest (negative) value l:

$$l = \arg \min_j (\lambda_j(x) < 0, \quad j = m_e + 1, \ldots, m). \tag{2.58}$$

This completes the discussion on active set stratgies.

Chapter 3

Quadratic Programming Algorithms

In this chapter three main algorithms for the quadratic programming core problem of the sequential quadratic programming method are presented. First a primal method based on the direct formulation of the problem is given, followed by a primal/dual approch based on a least squares transformation, last a recently published dual method is shown, which finds the active set beginning from an unconstrained problem. No method based on linear complementarity problems is given in this comparison, instead we refer to [7].

A self-contained derivation of the basic structure of each algorithm is given. The respective updating techniques for changing active constraint sets can be found in the references.

3.1 A Primal Method

Primal methods can be characterized as methods which find the solution from the interior of the set of feasible points. Accordingly a step to find a feasible point (e.g. phase one of the simplex method) has to be done before starting the minimization process.

In this section we confine ourselves to inequality constrained problems not to overburden the notation. These problems are solved by active set strategies and it is obvious that equality constraints can easily be handled within this framework. The direct formulation of the inequality constrained quadratic programming problem (IQP) can be stated as

$$\textbf{(IQP):} \quad \min_{x \in R^n} F(x) = \frac{1}{2} x^T G x + h^T x \tag{3.1}$$

subject to

$$A^T x \geq b, \tag{3.2}$$

where G is a constant and symmetric $n \times n$ matrix, A^T is an $m \times n$ matrix, which is assumed to be of full rank, h an $n-$vector, and b an $m-$vector.

In the case of a positive-definite matrix G the following necessary and sufficient conditions hold for x^* to be a unique (global) solution of (QP):

1. the vector x^* is feasible,

2. there exists a unique *strictly positive* multiplier $t-$vector λ^* such that

$$A^*\lambda^* = g(x^*), \tag{3.3}$$

where $g(x) = Gx + h$ is the gradient of $F(x)$, and A^{*T} is the $t \times n$ sub-matrix of A^T corresponding to the active constraints at x^*.

With the introduction of an $n \times (n-t)$ matrix Z^*, the columns of which form a basis for the set of vectors orthogonal to the rows of A^{*T}, that means $A^{*T}Z^* = 0$, condition (3.3) is equivalent to $Z^{*T}g(x^*) = 0$. The vector $Z^Tg(x)$ will be called the *projected gradient* of F at x, and the matrix Z^TGZ will be called the *projected* HESSIAN *matrix*.

In the case of a general matrix G a positive-definite matrix $Z^{*T}GZ^*$ is a second-order sufficiency condition for x^* to be a solution of (QP).

3.1.1 Solution Method

If x^* is a solution of (QP) it is also a solution of the equality-constrained quadratic program (EQP):

$$\textbf{(EQP):} \quad \min_{x \in R^n} F(x) = \frac{1}{2}x^TGx + h^Tx \tag{3.4}$$

subject to

$$A^{*T}x = b^*. \tag{3.5}$$

Therefore, a solution of problem (EQP) combined with the *active set* strategy described above yields a solution of the general problem (QP). As the pair (A^{*T}, b^*) corresponding to the correct active set at the solution x^* is not known in advance, an estimate $(\hat{A}^{(k)T}, \hat{b}^{(k)})$ is used instead at the k^{th} iteration of the numerical algorithm which solves problem (QP).

To solve problem (EQP) iteratively, let x^k be a feasible vector with the corresponding matrix of active constraints A^k and vector of right-hand sides b^k. A minimum of $F(x)$ satisfying the constraints $(A^k)^T = b^k$ is sought starting from x^k into the direction p^k. Substituting $x^k + p$ into problem (EQP) gives the following quadratic progamming sub-problem in terms of the vector p:

$$\min_{p \in R^n} \frac{1}{2}p^TGp + (Gx^k + h)^Tp \tag{3.6}$$

subject to

$$(A^k)^T p = 0. \tag{3.7}$$

Introduce the relation

$$p^k = Z^k p_Z^k \tag{3.8}$$

for some $(n - t^k)$ vector p_Z^k. Therefore linear combinations of the columns of Z (a basis of the set of all vectors orthogonal to the rows of A^T) form the set of all feasible directions for the solution of problem (EQP).

Substituting equation (3.8) into equations (3.6) and (3.7) the following equivalent unconstrained problem in terms of the $(n - t^k)$ vector p_Z is obtained

$$\min_{p_Z \in R^{n-t^k}} \frac{1}{2} p_Z^T (Z^k)^T G Z^k p_Z + (Z^k)^T (Gx^k + h)^T p_Z, \tag{3.9}$$

the solution p_Z^k of which can be obtained from solving the following linear system of equations:

$$(Z^k)^T G Z^k p_Z^k + (Z^k)^T (Gx^k + h) = 0. \tag{3.10}$$

The optimal direction p^k for problem (EQP) is recovered by inserting the result of equation (3.10) into equation (3.8). The corresponding estimate of the LAGRANGE multiplier vector λ^k will be calculated from equation (3.3)

$$\lambda^k = (A^k)^+ (Gx^k + h), \tag{3.11}$$

where $(A^k)^+$ is the pseudo-inverse of A^k.

3.1.2 Numerical Implementation

Three main problems arise in the numerical solution of the direct quadratic programming problem:

1. the representation of the matrix Z in terms of the matrix A^T,
2. the efficient computation of the pseudo-inverse A^+,
3. and an effective solution of the linear system (3.10).

To deal with the first two items, the matrix A^k is factorized into the product

$$Q^k A^k = \begin{pmatrix} R^k \\ 0 \end{pmatrix}, \tag{3.12}$$

by HOUSEHOLDER-transformations, where Q^k is an orthogonal matrix and R^k is a non-singular upper triangular matrix. The matrix Q^k is partitioned into two submatrices, the $t^k \times n$ matrix Q_L^k and the $(n - t^k) \times n$ matrix Q_Z^k such that

$$Q^k = \begin{pmatrix} Q_L^k \\ Q_Z^k \end{pmatrix}, \tag{3.13}$$

It has been shown [14] that the last $(n - t^k)$ rows of the matrix Q^k are orthogonal to the rows of the matrix $(A^k)^T$ such that the rows of Q_Z^k can be used as a representation for the columns of Z^k. The order of the rows is not important; and from computational efficiency in the updating process within the active-set strategy the reverse row-order of the matrix Q_Z^k : $Z^k = (Q_Z^k)^T \bar{I}$, where $\bar{I}$ is the $t^k \times t^k$ unit matrix with reversely ordered columns.

A numerically stable and efficient realization of the pseudo-inverse $(A^k)^+$ of the matrix A^k can also be obtained from the factors of A^k, namely

$$(A^k)^+ = (R^k)^{-1} Q_L^k. \tag{3.14}$$

This result is due to BUSINGER and GOLUB [5], and it has the advantage that the condition of the problem is not changed as in other solution methods of the inherent "least-squares" problem.

The third computational problem identified above is the efficient solution of the positive-definite system of $(n - t^k)$ linear equations (3.10). This is accomplished by the following 'square-root-free' CHOLESKY factorization of the projected HESSE matrix

$$(Z^k)^T G Z^k p_Z^k = L^k D^k (L^k)^T, \tag{3.15}$$

with L^k a unit lower-triangular matrix and D^k a diagonal matrix. It is very important not to deteriorate the factorized problem compared to the initial problem, that means the condition number of the former must not be worse than that of the latter: $\kappa((Z^k)^T G Z^k) \leq \kappa(G)$. The choice of Z^k as described above guarantees this condition [14].

As constraints enter or leave the active set the factors L, D, Q and R need not be computed ab initio for every iteration but can be modified in a numerically stable way. This modification is performed in a number of steps which is usually one order less than that of the complete new build-up. The details can be found in reference [14].

3.2 A Primal/Dual method

3.2.1 Problem Equivalence

It can be shown that the general quadratic programming problem (QP)

$$\textbf{(QP):} \quad \min_{x \in R^n} \frac{1}{2} x^T G x + h^T x \tag{3.16}$$

subject to

$$A^T x \geq b, \tag{3.17}$$

and

$$C^T x = d, \tag{3.18}$$

is equivalent to the following linear least squares fomulation (LSEI)

$$\textbf{(LSEI):} \quad \min_{x \in R^n} \frac{1}{2} \|Ex - f\|^2 \qquad (3.19)$$

subject to

$$A^T x \geq b, \qquad (3.20)$$

and

$$C^T x = d. \qquad (3.21)$$

This problem has been extensively treated by LAWSON and HANSON [26]. In the context of Sequential Quadratic Programming (SQP) methods, square matrices E with dimension n and with special structure are considered. The matrices A^T and C^T are of dimension $(m - m_e) \times n$ and $m_e \times n$, respectively. All matrices are assumed to be of full rank. This assumption does not constrain the generality of the method; it has been introduced to relax the derivation from the necessary refinements in case of rank deficiencies.

3.2.2 Problem Transformation

According to LAWSON and HANSON [25] a transformation of the variables is introduced by an orthogonal basis K of the nullspace of C^T:

$$x = K \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \begin{matrix} m_e \\ n - m_e \end{matrix}, \qquad (3.22)$$

where the $n \times n$ orthogonal matrix K triangularizes the matrix C^T from the right:

$$\begin{matrix} \\ m_e \\ n \\ m - m_e \end{matrix} \begin{matrix} n \\ \begin{pmatrix} C^T \\ E \\ A^T \end{pmatrix} \end{matrix} K = \begin{matrix} m_e & n - m_e \\ \begin{pmatrix} \tilde{C}_1 & 0 \\ \tilde{E}_1 & \tilde{E}_2 \\ \tilde{A}_1 & \tilde{A}_2 \end{pmatrix} \end{matrix}. \qquad (3.23)$$

Then $\hat{x}_1$ is determined as the solution of the lower triangular system

$$\tilde{C}_1 x_1 = d, \qquad (3.24)$$

and $\hat{x}_2$ is the solution of the following inequality-constrained least squares problem (LSI)

$$\textbf{(LSI):} \quad \min_{x_2 \in R^{n-m_e}} \|\tilde{E}_2 x_2 - (f - \tilde{E}_1 \hat{x}_1)\| \qquad (3.25)$$

subject to

$$\tilde{A}_2 x_2 \geq b - \tilde{A}_1 \hat{x}_1. \qquad (3.26)$$

Problem (LSI) is now transformed into a least distance problem (LDP). First the QR factors of the $n \times (n - m_e)$ matrix $\tilde{E}_2$

$$\tilde{E}_2 = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{3.27}$$

are determined, with Q an $n \times n$ orthogonal matrix and R an upper triangular nonsingular matrix. Then a further change of variables is introduced

$$y = Rx_2 - \tilde{f}, \tag{3.28}$$

with $\tilde{f} = \tilde{Q}^T(f - \tilde{E}_1\hat{x}_1)$, where the $n \times (n - m_e)$ matrix $\tilde{Q}^T$ is composed of the first $n \times (n - m_e)$ columns of the matrix Q. Problem (LDP) can now be formulated as

$$\textbf{(LDP):} \quad \min_{y \in R^{n-m_e}} \|y\| \tag{3.29}$$

subject to

$$\tilde{A}_2 R^{-1} y \geq b - (\tilde{A}_1\hat{x}_1 + \tilde{A}_2 R^{-1}\tilde{f}). \tag{3.30}$$

The solution $\hat{y}$ of problem (LDP) is introduced into equation (3.28) from which the solution $\hat{x}_2$ of problem (LSI) is obtained, which in turn is put into equation (3.22) to give the solution x of the original problem (LSEI).

Problem (LDP) has a non-negative least squares problem (NNLS)

$$\textbf{(NNLS):} \quad \min_{z} \|Gz - h\| \tag{3.31}$$

subject to

$$z \geq 0 \tag{3.32}$$

as its dual problem, where the $(n+1) \times m$ matrix G is composed of the constraint matrix and the right hand side of equation (3.30) and the $(n+1)$ vector h consists of n leading zeros followed by a trailing one, $h = (0, \ldots, 0, 1)^T$. This observation has been stated by CLINE [6].

From the solution $\hat{z}$ of the dual problem (NNLS) and its residual $r := G\hat{z} - h$ the solution $\hat{y}$ of problem (LDP) is obtained by the relation

$$\hat{y}_i = r_i / r_{n+1}, \quad i = 1, \ldots, n - m_e, \tag{3.33}$$

and the corresponding LAGRANGE multipliers λ of problem (LDP) are

$$\lambda_j = -u_j / r_{n+1}, \quad j = 1, \ldots, m, \tag{3.34}$$

where the vector u is the multiplier vector of problem (NNLS).

3.3 Dual Quadratic Programming

3.3.1 The Problem

In this section a very efficient dual quadratic programming algorithm due to GOLDFARB and IDNANI [17,34,41] to solve the following problem

$$\min_{x \in R^n} F(x) := \frac{1}{2} x^T G x + h^T x \tag{3.35}$$

subject to

$$c(x) := A^T x \geq b, \tag{3.36}$$

will be described, which has several advantegeous features such as no initial feasible point necessity. The vector c may include genuine equality constraints. Their explicit treatment is ommitted here in order to simplify the presentation.

The algorithm is also of the active set type, but instead of finding a local minimizer from the interior of the feasible domain, this method approaches the minimum from the exterior. To be more specific, defining the index set $J \subseteq M := \{1, \ldots, m\}$, where m is the total number of constraints in equation (3.36), a sequence of subproblems P(J) is considered, starting with P($\emptyset$), where $\emptyset$ is the empty set. If the solution x of a subproblem P(J) lies on some linearly independent set of active constraints indexed by $K \subseteq J$, the pair (x, K) will be called an S-pair. The sequence of subproblems is constructed such that for two successive S-pairs (x, K) and (x^+, K^+), say,

$$F(x) < F(x^+)$$

is valid.

To describe the dual algorithm in detail, some notation will be necessary. The matrix of normal vectors of the active constraints will be denoted by $\hat{A}$ with its index set K and $\dim(K) =: k$. K^+ will be the index set of active constraints augmented by the index $j \in M \setminus K$, while a proper deletion of one element from K results in K^-. The matrices $\hat{A}^+$ and $\hat{A}^-$ correspond to K^+ and K^-, respectively. The vector a^+ will indicate the normal vector a_j added to $\hat{A}$ to give $\hat{A}^+$ and a^- will be the column vector deleted from $\hat{A}$ to give $\hat{A}^-$. For $\hat{A}$ being of full rank its MOORE–PENROSE inverse is

$$\hat{A}^* = (\hat{A}^T G^{-1} \hat{A})^{-1} \hat{A}^T G^{-1}. \tag{3.37}$$

The inverse reduced Hessian matrix of F in the linear manifold defined by $\hat{A}$ is

$$H = G^{-1}(I - \hat{A}\hat{A}^*) = G^{-1} - G^{-1}\hat{A}(\hat{A}^T G^{-1} \hat{A})^{-1} \hat{A}^T G^{-1}. \tag{3.38}$$

The multipliers $q = \hat{A}^* a^+$ will be denoted as infeasibility multipliers.

3.3.2 Optimality Conditions

With this notation necessary and sufficient conditions for P(J) to be optimal can be stated. For a vector $\hat{x}$ in the $(n-k)$ dimensional manifold $\mathcal{M} = \{x \in R^n \mid \hat{a}_i^T x = b_i, i \in K\}$ with the corresponding gradient $\nabla F(\hat{x}) =: g(\hat{x}) = G\hat{x} + h$ of $F(x)$ at $\hat{x}$ the minimum of F over $\mathcal{M}$ is obtained at

$$\bar{x} = \hat{x} - Hg(\hat{x}). \tag{3.39}$$

If for Lagrange multipliers $\lambda(\bar{x}) \geq 0$ the relation

$$g(\bar{x}) = \hat{A}\lambda(\bar{x}) \tag{3.40}$$

holds, then the vector $\bar{x}$ is an optimal solution of P(K) with

$$\lambda(\bar{x}) := \hat{A}^* g(\bar{x}) \geq 0 \tag{3.41}$$

and

$$Hg(\bar{x}) = 0. \tag{3.42}$$

3.3.3 The Algorithm

Now, the dual algorithm **(DQP)** can be implemented in the following steps:

1. Initialize:
   set $k = 0$ and $K = \emptyset$,
   $x := -G^{-1}h$ and $H := G^{-1}$,
   unconstrained minimum of $F(x) = \frac{1}{2}h^T x$.

2. Choose a violated constraint:
   compute $c_i(x), \quad \forall i \in M \setminus K$,
   if $V := \{i \in M \setminus K \mid c_i(x) < 0\} = \emptyset$
   then stop, solution is feasible and optimal;
   else choose smallest $j \in V$ such that

   $$j := \arg\min_{i \in V} \{a_i^T x - b_i < 0\}$$

   and set

   $$a^+ := a_j \quad \text{and} \quad \lambda^+ := \begin{pmatrix} \lambda \\ 0 \end{pmatrix}.$$

   If $k = 0$ **then** set $\lambda := 0$ and $K^+ = K \cup \{j\}$.

3. Check feasibility and determine step:

   (a) determine step direction:
   primal space direction: $p = Ha^+$
   and **if** $k > 0$ **then** negative dual space direction: $q = A^* a^+$,

   (b) compute step length:

      i. maximum step t_1 in dual space without violating dual feasibility:
      if $q \leq 0$ or $k = 0$ **then** set $t_1 = \infty$

$$\textbf{else set} \quad t_1 : \min_{\substack{q_i > 0 \\ i \in K}} \left\{ \frac{\lambda_i^+(x)}{q_i} \right\} =: \frac{\lambda_l^+(x)}{q_l}$$

      ii. minimum step t_2 in primal space such that j^{th} constraint becomes feasible:
      if $|p| = 0$ **then** set $t_2 := \infty$
      else set $t_2 := -\frac{c_j(x)}{p^T a^+}$

      iii. step length t:
      set $t := \min(t_1, t_2)$.

4. Determine new S-pair and take step:

   (a) no step in primal or dual space:
   if $t = \infty$ **then** stop, subproblem $\mathrm{P}(K^+)$ and hence (QP) are infeasible;

   (b) step in dual space:
   if $t_2 = \infty$ **then** set $\lambda^+ := \lambda^+ + t\binom{-q}{1}$, and drop constraint l;
   i.e. set $K := K \setminus \{l\}$, and $k := k - 1$, update H and A^*,
   go to step 3;

   (c) step in primal and dual space:
   set $x := x + tp$,
   $\lambda^+ := \lambda^+ + t\binom{-q}{1}$,
   $F := F + tp^T a^+ (\frac{1}{2}t + \lambda_{k+1}^+)$,
   if $t = t_2$ **then** set $\lambda := \lambda^+$,
   and add constraint j;
   i.e. set $K := K \cup \{j\}$, and $k := k + 1$, update H and A^*,
   go to step 2;
   if $t = t_1$ **then** drop constraint l;
   i.e. set $K := K \setminus \{l\}$, and $k := k - 1$, update H and A^*,
   go to step 3.

This finishes the algorithm (DQP).

Appendix A

Subroutine SQP

A.1 Implementation

The methods described in chapters 2 and 3 have been implemented in a subroutine SQP (in the RASP-library [18] this is subroutine RPSQP1). SQP is the header subroutine which initializes addresses for workspace, checks input parameters and calls SQPBDY, in which the Sequential Quadratic Programming algorithm of chapter 2 is programmed. SQPBDY determines search direction, step size, updates the B-matrix and calls Quadratic Programming solvers.

Three alternatives are possible. They depend on the problem to be solved, especially the number of constraints that are active at the solution. The primal method considered is QPSOL of *Stanford Optimization Laboratory* [16]. This program is a licensed software product and can be obtained from the authors for a nominal fee. The primal/dual method considered is LSEI a modification of the public domain least squares software of LAWSON and HANSON [26]. The modification concerns a software frame for the ideas in chapters 20 and 23 of [26, pages 135–139, 167–169] and the computation of LAGRANGE multipliers. The dual method considered is a modification with respect to the introduction of lower and upper bounds of POWELLs implementation ZQPCVX [33] of the GOLDFARB-IDNANI algorithm.

The line search is implemented in the subroutine LINMIN, a modification of BRENTs algorithm localmin [3]. The B-matrix update implementation LDLT is a modification of the FLETCHER-POWELL update MC11A contained in the *Harwell* library [22].

A.2 Description

In this Appendix the numerical implementation of Sequential Quadratic Programming in the subroutine **SQP** is described in RASP format [1].

RASPOPTCONSTR$RPSQP1$$

SUBROUTINE

Minimization of a scalar function subject to equality & inequality constraints.

Procedure purpose:

This subroutine solves the general nonlinear programming problem

$$\textbf{(NLP):} \quad \min_{x \in R^n} f(x)$$

subject to

$$g_j(x) = 0, \quad j = 1, \ldots, m_e,$$

$$g_j(x) \geq 0, \quad j = m_e + 1, \ldots, m,$$

$$x_l \leq x \leq x_u,$$

for a local minimum, where the problem functions $f : R^n \rightarrow R^1$ and $g : R^n \rightarrow R^m$ are assumed to be continuously differentiable and to have no specific structure.

The algorithm implements the method of WILSON, HAN and POWELL with a BFGS-update of the B-matrix and L1-test function within the steplength algorithm.

The search direction is determined by solving a quadratic subproblem. Three alternatives can be used to do this: primal, primal/dual and dual methods.

Usage:

CALL RPSQP1 (M, MEQ, LA, N, X, XL, XU, F ,C , G, A, ACC, ITR, MODE, W, LW, JW, KW)

M : IN, INTEGER
M is the total number of constraints, M $\geq$ 0.

MEQ : IN, INTEGER
MEQ is the number of equality constraints, MEQ $\geq$ 0.

LA : IN, INTEGER
LA is the row dimension of the matrix A storing the Jacobian of the problem, LA $\geq$ max(M, 1).

N : IN, INTEGER
N is the number of varibles, N $\geq$ 1.

X() : IN, OUT, DOUBLE PRECISION (N)
X() stores the current iterate of the vector X with dim(X) = N, on entry X() must be initialized, on exit X() stores the solution vector X if MODE = 0.

XL() : IN, DOUBLE PRECISION (N)
XL() stores an N vector of lower bounds XL to X.

XU() : IN, DOUBLE PRECISION (N)
XU() stores an N vector of upper bounds XU to X.

F : IN, DOUBLE PRECISION
F is the value of the objective function.

C() : IN, DOUBLE PRECISION (M)
C() stores the M vector C of constraints, equality constraints (if any) first. dim(C()) $\geq \max(1, M)$.

G() : IN, DOUBLE PRECISION (M)
G() stores the N vector G of partials of the objective function with respect to the parameters X, dim(G()) $\geq$ N.

A() : IN, DOUBLE PRECISION (LA, N+1)
the LA $\times$ (N + 1) array A() stores the M $\times$ N matrix A of constraint normals. Row dimension of A() is LA $\geq \max(1, M)$.

F,C,G,A must all be set by the user before each call.

ACC : IN, DOUBLE PRECISION
|ACC| controls the final accuracy. If ACC < 0 an exact linesearch is performed within the subroutine LINMIN, otherwise an Armijo-type linesearch is used.

ITR : IN, OUT, INTEGER
On entry ITR prescribes the maximum number of iterations. On exit ITR indicates the number of iterations.

MODE : IN, OUT, INTEGER
MODE controls calculation: reverse communication is used in the sense that the program is first initialized by MODE = 0; then it is to be called repeatedly by the user until a RETURN with MODE $\neq$ |1| occurs. If MODE = -1 then gradients have to be calculated, while with MODE = 1 functions have to be calculated. MODE must not be changed between subsequent calls of subroutine SQP.
Description of MODE:

-1: gradient evaluation, (G & A)
0: on entry: initialization, (F, G, C & A)
on exit : required accuracy for solution obtained
1: function evaluation, (F & C)
2: unbounded solution of QP-subproblem
3: positive directional derivative in linesearch
4: inequality constraints incompatible
5: cycling in LP or QP
6: more than 3333 iterations in LP or QP
7: more than ITR iterations in RPSQP
8: input error, wrong dimensions
$\geq$9: working array W or JW too small, should be enlarged to the integer that is returned by MODE.

W() : IN, OUT, DOUBLE PRECISION(LW)
W() is a one dimensional working space, the length LW of which should be at least 2*N1*N1 + N1*N/2 + 11*N1 + 5*M + LA + 3*MAX(1,M) + 11, where N1 = N + 1. W must not be changed between subsequent calls of SQP. On RETURN W(1),...,W(M) contain the multipliers associated with the general constraints, while W(M+1),...,W(M+N) store the bound multipliers.

LW : IN, INTEGER
LW is the least dimension of the working array W.

JW() : IN, OUT, INTEGER(KW)
JW() is a one dimensional integer working space, the length KW of which should be at least 2*N1 + M + MAX(N1,M) +11. JW must not be changed between subsequent calls of SQP.

KW : IN, INTEGER
KW is the least dimension of the working array JW.

Database Structure:

All I/O work is managed via the parameter list.

Dialog:

Subroutine RPSQP1 requires no dialog.

File Input/Output:

Subroutine RPSQP1 requires no file I/O.

Method:

The underlying method is described in detail in chapters 2 & 3.

Remarks:

The user has to provide the following subroutines:
LDL(N,A,Z,σ,W) : update of the LDL^T-factorization,
LINMIN(A,B,F,ϵ) : linesearch algorithm if exact = 1,
QP(M,MEQ,LA,N,NC,C,D,A,B,XL,XU,X,λ,...) :
solution of the quadratic subproblem,
together with a couple of subroutines from BLAS [27].
RPSQP1 is head subroutine for body subroutine RPSQP2 in which the algorithm has been implemented.

Life Cycle:

1981 10 Dieter Kraft, DFVLR: coded
1984 12 Dieter Kraft, DFVLR: modified
1987 09 Dieter Kraft, DFVLR: adapted to RASP format

Packages required:

None.

Libraries required:

RASP'87.

Example:

None.

*****

Bibliography

[1] Bals, J., D. Joos, M. Otter: *Software-Engineering-Richtlinien für* RASP *und* ANDECS: *Anforderung und Entwurf.* DFVLR-MITT. 88-07, 1988.

[2] Bertsekas, D.P.: *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, New York, 1982.

[3] Brent, R.P.: *Algorithms for Minimization without Derivatives.* Prentice-Hall, Englewood Cliffs, 1973.

[4] Bunch, J.R., L. Kaufman: A computational method for the indefinite quadratic programming problem. Lin. Algebra Applic. **34** (1980) 341–370.

[5] Businger, P., G.H. Golub: Linear least squares solutions by Householder transformations. Num. Mathematik **7** (1965) 269–276.

[6] Cline, A.: *The Transformation of a Quadratic Programming Problem into Solvable Form.* ICASE Rep. 75–14. NASA Langley Research Center, Hampton, VI, 1975.

[7] Cottle, R.W., J.S. Pang: On the convergence of a block successive overrelaxation method for a class of linear complementarity problems. Math. Progr. Study **17** (1982) 126–138.

[8] Dennis, J.E., R.B. Schnabel: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.*

[9] Fletcher R., M.J.D. Powell: On the modification of LDL^T factorizations. Math. Comp. **28** (1974) 1067–1087. Prentice-Hall, Englewood Cliffs, 1983.

[10] Fletcher, R.: *Practical Methods of Optimization.* Vol. I Unconstrained Optimization. John Wiley, Chichester, 1980.

[11] Fletcher, R.: *Practical Methods of Optimization.* Vol. II Constrained Optimization. John Wiley, Chichester, 1981.

[12] Fontecilla, R.: Local convergence of secant methods for nonlinear constrained optimization. SIAM J. Numer. Anal. **25** (1988) 692–712.

[13] Gill, P.E., G.H. Golub, W. Murray, M.A. Saunders: Methods for modifying matrix factorizations. Math. Comp. **28** (1974) 505–535.

[14] Gill, P.E., W. Murray: Numerically stable methods for quadratic programming. Math. Prog. **14** (1978) 349–372.

[15] Gill, P.E., W. Murray, M.H. Wright: *Practical Optimization*. Academic Press, London, 1981.

[16] Gill, P.E., W. Murray, M. Saunders, M.H. Wright: User's Guide for SOL/QPSOL: A Fortran Package for Quadratic Programming. Report SOL 82–7, Systems Optimization Laboratory, Stanford University, Stanford, 1982.

[17] Goldfarb, D., A. Idnani: A numerically stable dual method for solving strictly convex quadratic programs. Math. Prog. **27** (1983) 1–33.

[18] Grübel, G.: Die regelungstechnische Programmbibliothek RASP. Regelungstechnik **31** (1983) 248–256.

[19] Han, S.-P.: A globally convergent method for nonlinear programming. J.Opt.Theory Applic. **22** (1977) 297–309.

[20] Haug, E.J.: A review of distributed parameter structural optimization literature. In: E.J. Haug, J. Cea, (eds.): *Optimization of Distributed Parameter Structure*. Sijthoff and Noordhoff, Alphen aan den Rijn, 1981.

[21] Hock, W., K. Schittkowski: *Test Examples for Nonlinear Programming Codes*. Springer, Berlin, 1981.

[22] Hopper, M.J.: *Harwell Subroutine Library: A Catalog of Subroutines*. AERE Computer Science and Systems Report AERE–R 9185, 1978.

[23] Kraft, D.: Optimalsteuerungen — Ein systematisches Hilfsmittel zur rechnergestützten Erforschung der dynamischen Möglichkeiten eines Tieftemperaturwindkanals. DFVLR–FB 86–23, 1986.

[24] Kreisselmeier, G.,R. Steinhauser, R.: Systematische Auslegung von Reglern durch Optimierung eines vektoriellen Gütekriteriums. Regelungstechnik **27** (1979) 76–79.

[25] Lawson, C.L., R.J. Hanson: Extensions and applications of the Householder algorithm for solving linear least squares problems. Math. Comp. **23** (1969) 787–812.

[26] Lawson, C.L., R.J. Hanson: *Solving Least Squares Problems*. Prentice-Hall, Englewood Cliffs, 1974.

[27] Lawson, C.L., R.J. Hanson, D.R. Kincaid, F.T. Krogh: *Basic Linear Algebra Subprograms for FORTRAN Usage*. Sandia Laboratories Tech. Rept. SAND77-0898, and ACM Trans. Math. Softw. **5** (1979) 324–325.

[28] Luenberger, D.G.: *Introduction to Linear and Nonlinear Programming*. Addison-Wesley, Reading, 1973.

[29] Nocedal, J., M. Overton: Projected Hessian updating algorithms for nonlinearly constrained optimization. SIAM J. Numer. Anal. **22** (1985) 821–850.

[30] Polak, E., A. Sangiovanni-Vincentelli: Theoretical and computational aspects of design centering, tolerancing and tuning problems. IEEE Trans. Circuits Syst. **26** (1979) 295–318.

[31] Powell, M.J.D.: A fast algorithm for nonlinearly constrained optimization calculations. In: G.A. Watson (ed.): Numerical Analysis. Lecture Notes in Mathematics, Vol. 630. Springer, Berlin, 1978.

[32] Powell, M.J.D.: The convergence of variable metric methods for nonlinearly constrained optimization calculation. In: O.L. Mangasarian, R.R. Meyer, K.Ritter (eds.): Nonlinear Programming 3. Academic Press, New York, 1978.

[33] Powell, M.J.D.: ZQPCVX – A Fortran subroutine for convex quadratic programming. Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Report DAMTP/1983/NA17, 1983.

[34] Powell, M.J.D.: On the quadratic programming algorithm of Goldfarb and Idnani. Math. Prog. Study **25** (1985) 46–61.

[35] Rockafellar, R.T.: The multiplier method of Hestenes and Powell applied to convex programming. J.Opt.Theory Applic. **12** (1973) 555–562.

[36] Schittkowski, K.: *Nonlinear Programming Codes*. Springer, Berlin, 1980.

[37] Schittkowski, K.: The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function. Part 1: Convergence analysis. Numer. Math. **38** (1981) 83–114.

[38] Schittkowski, K.: The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function. Part 2: An efficient implementation with linear least squares subproblems. Numer. Math. **38** (1981) 115–127.

[39] Schittkowski, K.: On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search function. Math. Operat. Statist., Series Optimization **14** (1983) 197–216.

[40] Steinhauser, R.: Reglerentwurf für einen Tieftemperaturwindkanal mittels Gütevektoroptimierung. DFVLR–FB 85–37, 1985.

[41] Stoer, J.: *A Dual Algorithm for Solving Degenerate Linearly Constrained Linear Least Squares Problems.* Rep. 47/88. Institut für Angewandte Mathematik und Statistik, Universität Würzburg, 1988.

[42] Wilson, R.B.: A simplicial algorithm for concave programming. Ph.D. Dissertation, Graduate School of Business Administration, Harvard University, Boston, 1963.