REDUCTION METHODS IN NONLINEAR PROGRAMMING

G. VAN DER HOEK

Stellingen behorende bij het proefschrift
Reduction methods in Nonlinear Programming, G. van der Hoek,
Erasmus Universiteit Rotterdam, 12 juni 1980.

I

De in Oren [1], Oren en Spedicato [2] en Shanno en Phua [3] vermelde experimenten zijn niet illustratief met betrekking tot de door hen voorgestelde variabele metriek algoritmen. Bovendien kan toepassing van de door hen gehanteerde stopcriteria leiden tot minder bevredigende oplossingen.

[1] Oren, S.S., On the selection of parameters in self scaling variable metric algorithms, Math.Prog. <u>7</u>, 351-367, 1974.
[2] Oren, S.S. and E. Spedicato, Optimal conditioning of self scaling variable metric algorithms, Math.Prog. <u>10</u>, 70-90, 1976.
[3] Shanno, D.F. and K.H. Phua, Matrix conditioning and nonlinear optimization, Math.Prog. <u>14</u>, 149-160, 1978.
[4] Hoofdstuk II van dit proefschrift.

II

Bräuniger geeft in [5] een onjuist bewijs van de stelling dat, bij toepassing van een gemodificeerde Robinson-algoritme, de verzameling $I(z^*)$ bepaald is zodra in het bereikte iteratiepunt z_k aan enkele vrij algemene voorwaarden is voldaan.

[5] Bräuniger, J., A modification of Robinson's algorithm for general nonlinear programming problems requiring only approximate solutions of subproblems with linear equality constraints, in: Optimization Techniques, Proc. 8th IFIP Conf. on Opt.Tech. Würzburg, J. Stoer (ed.). Lect. Notes in Control and Inf. Sciences nr. 7, part 2, Springer, 1977.
[6] Hoofdstuk IV van dit proefschrift.

III

Het in Fletcher en Powell [7] gegeven bewijs van de convergentie van de door hen voorgestelde variabele metriek algoritme is niet volledig.

[7] Fletcher, R. and M.J.D. Powell, A rapidly convergent descent method for minimization, Computer J., <u>6</u>, 1963.

IV

Het verdient aanbeveling bij de vrije minimalisatie van boetefunkties gebruik te maken van de formules van zelf schalende variabele metriek algoritmen voor het bijwerken van de benaderingen van de inverse Hessiaan van de te minimaliseren funktie.

[8] Stelling VI bij F.A. Lootsma, Boundary properties of penalty functions for constrained minimization, Proefschrift, Eindhovèn, 1970.

V

Laat alle probleemfunkties in probleem (4.1) van dit proefschrift continu zijn met $F(x)$ en $-c_i(x)$ convex en zij p_k een in het k^e iteratiepunt x_k gedefinieerde zoekrichting. Zij $\lambda = \lambda_k$ de oplossing van minimalisatie langs $x_k + \lambda p_k$ van de met een vaste actieve verzameling $I(x_k)$ gedefinieerde boetefunktie

$$P_1(x, r_k) = F(x) + r_k^{-1} \sum_{i \in I(x_k)} c_i^2(x)$$

($I(x_k)$ bevat tenminste alle in x_k geschonden restricties). Noem nu $x_{k+1} = x_k + \lambda_k p_k$ en veronderstel ten slotte dat alle funkties $c_i(x_k + \lambda p_k)$ monotoon (niet dalend dan wel niet stijgend) in λ zijn op $(0, \lambda_k + \varepsilon)$ voor een $\varepsilon > 0$.

In verband met de mogelijk slechts lokale geldigheid van $I(x_k)$ wordt in werkelijkheid de boetefunktie

$$P_2(x, r_k) = F(x) + r_k^{-1} \sum_{i \in I_v(x)} c_i^2(x)$$

geminimaliseerd langs $x_k + \lambda p_k$, waarbij $I_v(x)$ de verzameling van in x geschonden restricties is. Stel dat dit lijnminimum wordt aangenomen voor $\lambda = \lambda_k'$. Dan gelden de volgende beweringen:

i Als $I_v(x_{k+1}) = I(x_k) \cup \{j\}$ ("restrictie $c_j(x)$ is vergeten") dan is $\lambda_k' \le \lambda_k$.

ii Als $I(x_k) = I_v(x_{k+1}) \cup \{j\}$, $j \notin I_v(x_{k+1})$ ("restrictie $c_j(x)$ is ten onrechte opgenomen")
dan is

a. $\lambda_k' \le \lambda_k$ als $c_j(x_k) > c_j(x_{k+1})$

b. $\lambda_k' \ge \lambda_k$ als $c_j(x_k) \le c_j(x_{k+1})$

[9] Van der Hoek, G. and R. Th. Wymenga, Aspects of recursive quadratic programming, Report Econometric Institute, Erasmus University Rotterdam (forthcoming).

VI

Het probleem van een kleine belegger die zijn vermogen in K gelijke delen splitst en deze delen vervolgens in niet meer dan N van n gegeven verschillende aandelen belegt ($n \geq N$ en $K \geq N$) is te herleiden tot een parametrisch geheeltallig programmeringsprobleem.
De in Faaland [10] voorgestelde impliciete aftellingsmethode om dit probleem op te lossen kan worden verbeterd door gebruik te maken van scherpere onder- en bovengrenzen op respectievelijk de variantie en de verwachte opbrengst van toegelaten completeringen van een gegeven partiële oplossing.

[10] Faaland, B., An integer programming algorithm for portfolio selection, Man.Sci., 20, no.10, 1974.

[11] Blog, B., G. van der Hoek, A.H.G. Rinnooy Kan and G.T. Timmer, The optimal selection of small portfolios, Report 7923/O, Econometric Institute, Erasmus University Rotterdam.

VII

Het bewijs van de convergentie van de in Wolfe [12] en Holtgrefe [13] voorgestelde gemodificeerde cutting plane reductie methoden zou gebruik moeten maken van een geschikt gekozen aktieve verzamelingsstrategie.

[12] Wolfe, Ph., Convergence theory in NLP, in: Integer and nonlinear programming, J. Abadie (ed.), North Holland, 1970.

[13] Holtgrefe, A.A.I., An optimizing medium-term planning model for the Netherlands railways, Proefschrift, Erasmus Universiteit Rotterdam, 1975.

VIII

De efficiency en de betrouwbaarheid van geconjugeerde gradient algoritmen kunnen verbeterd worden door iedere n stappen (n is het aantal beslissingsvariabelen) een gedeeltelijke herstart procedure toe te passen.

[14] Stern, V. and G. van der Hoek, Restart procedures for the method of conjugate gradients, Report Econometric Institute, Erasmus University Rotterdam (forthcoming).

IX

Het verdient aanbeveling de eigenschappen te onderzoeken van een recursief kwadratisch programmerings algoritme die gebaseerd is op het benaderen van een gemengde, inwendig punt- uitwendig punt boetefunktie.

[15] Van der Hoek, G., A reduction method for nonlinear programming, Report 7619/O, Econometric Institute, Erasmus University Rotterdam.

X

De asymptotische convergentie eigenschappen die in hoofdstuk IV van dit proefschrift bewezen worden voor de 2-fase reductie methoden, zijn vermoedelijk op analoge wijze af te leiden voor de in hoofdstuk III besproken recursief kwadratische programmerings reductie methoden.

XI

Een optimaal nuttig effekt van de aanschaf van meer geavanceerde rekenapparatuur wordt slechts dan verkregen indien tegelijkertijd de beschikbaar gestelde programmatuur wordt aangepast en de gebruikers wèl doordacht de nieuwe mogelijkheden benutten.

XII

Het woord van Thomas à Kempis: "Hij tot wie het eeuwige Woord spreekt is vrijgemaakt van de vele meningen", behoort bij het overleg in kerkelijke aangelegenheden voor ogen te staan en de gezindheid te bepalen.

[16] Thomas à Kempis, De navolging van Christus (vertaling door B. Wielenga, Delft, 1948).

XIII

De uitspraak van Luther: "mijn geweten is gebonden in het Woord van God" verklaart zijn stellingname tegen Erasmus en diens opvatting over de vrije wil.

[17] Kooiman, W.J., Luther, zijn weg en werk, Amsterdam, 1962.

[18] Erasmus, D., De vrije wil.

[19] Luther, M., De knechtelijke wil.
Voor referenties [18] en [19] werd gebruik gemaakt van de door C.C. van der Geer-de Ruiter verzorgde uitgave, Bleiswijk, 1973.

[20] Augustijn, C., Erasmus, vernieuwer van kerk en theologie, Baarn, 1967.

[21] Melles, G., Albertus Pighius en zijn strijd met Calvijn over het liberum arbitrium, Proefschrift, Kampen, 1973.

XIV

De spreuk: "Laat een vreemde u prijzen en niet uw mond; een onbekende, en niet uw lippen", is ook van toepassing bij het beoordelen van wetenschappelijk onderzoek.

[22] Spreuken 27:2.

XV

Het zou rendabel kunnen zijn wanneer over de aanschaf van meer geavanceerde rekenapparatuur langer werd vergaderd dan tot nu toe gebruikelijk.

REDUCTION METHODS IN NONLINEAR PROGRAMMING

REDUCTION METHODS IN NONLINEAR PROGRAMMING

PROEFSCHRIFT

TER VERKRIJGING VAN DE GRAAD VAN DOCTOR IN DE
ECONOMISCHE WETENSCHAPPEN AAN DE ERASMUS UNIVERSITEIT
ROTTERDAM, OP GEZAG VAN DE RECTOR MAGNIFICUS
PROF. DR. J. SPERNA WEILAND EN VOLGENS BESLUIT VAN HET
COLLEGE VAN DEKANEN.
DE OPENBARE VERDEDIGING ZAL PLAATSVINDEN
OP DONDERDAG 12 JUNI 1980
DES NAMIDDAGS TE 4 UUR

DOOR

GERARD VAN DER HOEK

GEBOREN TE ROTTERDAM

1980
MATHEMATISCH CENTRUM AMSTERDAM

PROMOTOR: PROF.DR.IR. H.W. VAN DEN MEERENDONK

CO-REFERENT: PROF.DR. M. HAZEWINKEL

In dankbare herinnering aan mijn vader.

Aan mijn moeder.

Aan mijn vrouw Janneke

en onze kinderen Hanneke

Esther

Niels-Jan

Paul.

Contents

Voorwoord

Bij het verschijnen van dit proefschrift wil ik allereerst de promotor, prof.dr.ir. H.W. van den Meerendonk hartelijk dank zeggen voor de wijze waarop hij, na mij eerst geattendeerd te hebben op de niet lineaire programmering, het onderzoek vervolgens heeft bevorderd en begeleid.
De co-referent, prof.dr. M. Hazewinkel ben ik erkentelijk voor de grote nauwgezetheid waarmee hij het manuscript gelezen heeft en voor het daaruit voortvloeiende commentaar.
De adviezen van prof.dr. A.H.G. Rinnooy Kan, in het bijzonder betreffende de verslaggeving, heb ik zeer op prijs gesteld.
Zeer goede herinneringen bewaar ik aan de intensieve samenwerking met de heren A.S. Louter en R. Th. Wymenga, ter verkrijging van de in het laatste hoofdstuk vermelde numerieke vergelijking van algoritmen.
De discussies en de samenwerking met de heer J.F. Ballintijn, drs. C.L. Hooykaas en drs. M.W. Dijkshoorn heb ik als zeer zinvol ervaren.
Voor de verdere uitvoering ben ik veel dank verschuldigd aan de heer K. Lammerse voor het kritisch bezien van het door mij gebruikte Engels, mevr. E.F.F. Jonker, mevr. N. Kauffman-Dumoulin en mevr. H.M. Sagum voor het typen van het manuscript en de heer J. van Dijk voor het ontwerpen van de omslag en het tekenen van de figuren.
Het Mathematisch Centrum ben ik erkentelijk voor haar bereidheid dit proefschrift uit te geven, de heer D. Zwarst en zijn medewerkers voor de technische realisering daarvan.
Mijn zeer bijzondere dank gaat uit naar mijn vrouw. Zonder haar begrip, steun en medewerking zou ik dit niet hebben kunnen voltooien.

I. INTRODUCTION AND PRELIMINARY REMARKS

I.1. Introduction and scope of the monograph

This monograph deals with techniques for the solution of *nonlinear programming problems*, i.e., problems in which an *objective function* is to be optimized subject to some *constraints*, where at least one of the decision variables appears in a nonlinear way in the objective function or in one or more of the constraints.
Hence we shall consider the following problem:

$$(1.1) \qquad \begin{cases} \text{minimize} \quad F(x) \\ \text{subject to} \\ \qquad\qquad c_i(x) \begin{Bmatrix} \geq \\ = \end{Bmatrix} 0 \qquad i = 1, \ldots, m \end{cases}$$

where $x \in E^n$, n-dimensional Euclidian space. The *problem functions* $F(x)$ and $c_i(x)$, $i = 1,2,\ldots,m$ are mappings E^n to E^1.
Special cases of this general problem formulation are *unconstrained nonlinear programming problems* (when the constraint functions $c_i(x)$ in (1.1)are absent) and *linearly constrained nonlinear programming problems* (all $c_i(x)$ are linear functions of $x_1, \ldots, x_n$).
Both these special cases of the general problem formulation (1.1) will play an important role in the solution techniques to be discussed.
Our basic approach will be to look for solution procedures which reduce the solution of problem (1.1) to the solution of a sequence of *simpler* nonlinear programming problems, such as the unconstrained or the linearly constrained problems mentioned above.
The development of techniques which apply unconstrained auxiliarly problems to solve the constrained problem (1.1) goes back to Courant (1943). He suggested to study the relations between the solutions of a purely equality constrained version of problem (1.1) and the solutions of the unconstrained problems

$$(1.2) \qquad \text{minimize } F(x) + r_k \sum_{i=1}^{m} c_i^2(x)$$

for a sequence of positive parameters r_k such that $\{r_k\} \to \infty$.
Later Frisch (1954, 1955) and Carroll (1961) applied similar techniques.
Fiacco and Mc Cormick performed a thorough analysis resulting in the class

of *S*equential *U*nconstrained *M*inimization *T*echniques (SUMT). See e.g., Fiacco and Mc Cormick (1963, 1964 a, b, 1966, 1968).
They use the auxiliary problem (1.2) to develop an *exterior point penalty function technique* in which a solution x* of the original problem (1.1) is approached from the outside of the feasible region. The penalty term $r_k\Sigma_{i=1}^{m} c_i^2(x)$ is intended to penalize constraint violations. These exterior point techniques should be distinguished from the *interior point penalty function techniques* (also known as *barrier function techniques*) in which a solution is approached from the inside of the feasible region. A hybrid approach evolves from treating some of the constraints by means of an exterior point *loss term* while a barrier function is applied to the other constraints: *mixed interior point-exterior point penalty function techniques*.
Lootsma investigated the *boundary properties* of the resulting solution techniques in Lootsma (1970), starting from a classification of penalty and barrier functions. The resulting continuity properties - in terms of the penalty parameter - of the *trajectory of penalty function minima* provides a sound theoretical basis for the application of extrapolation techniques to accelerate the convergence to the optimum.
Some further developments were reported in Ryan (1971, 1974), who gave the following definition for *transformation methods:*

Definition. A *transformation method* is a method which solves problem (1.1) by transforming the constrained optimization problem into one or more unconstrained optimization problems.

Obviously penalty function techniques belong to the class of transformation methods thus defined. The development of transformation methods stimulated research in the field of unconstrained optimization, which led to efficient algorithms such as those presented in Fletcher and Powell (1963), and Broyden, Fletcher, Goldfarb and Shanno (see e.g., Broyden (1970)).
As the auxiliary unconstrained penalty functions turn out to be increasingly ill conditioned if the penalty parameter increases, the solution of the unconstrained optimization problems becomes more and more difficult. The numerical difficulties encountered, as reported in Murray (1967) and Lootsma (1969), stimulated further research. A way out of this difficulty seems to be to use some kind of *scaling* technique to the

unconstrained problems. For instance the algorithms presented in Fletcher (1970a) Oren and Luenberger (1974) or Shanno and Phua (1978a) could be applied. These proposals will be discussed in more detail in chapter II. An alternative way is to prevent the occurrence of ill conditioned problems by the introduction of auxiliary problems which do not suffer from this complication. This approach is followed by means of the so-called *augmented Lagrangian* functions, see e.g., Rockafellar (1974), Fletcher (1969) and Powell (1969b).
Another proposal in the same category is to use quadratic linearly constrained auxiliary problems. It can be found in Murray (1969), Biggs (1972, 1978), Han (1977, 1979) and Powell (1977a, 1978). Chapter III deals with properties of such a *Recursive Quadratic Programming* algorithm, which directly applies the results of chapter II.

However, these last mentioned algorithms are no longer transformation methods, as they involve constrained auxiliary problems. That is why we introduce a natural extension of the class of transformation methods in the following definition.

Definition. A *reduction method* is a method which solves problem (1.1) by reducing this optimization problem into one or more 'simpler' optimization problems, where 'simpler' means a decrease in the degree of nonlinearity of the problem functions and/or a decrease in the number of constraints.

Examples of reduction methods are: all transformation methods and all other approximation methods that apply successive linearizations and/or quadratic approximations to solve problem (1.1), e.g., Kelley's cutting plane method (Kelley (1960)), the method of approximation programming (Griffith and Stewart (1967)), the method of gradient projection (Rosen (1961)), the generalized reduced gradient methods (Abadie and Guigou (1969) and Abadie and Haggag (1979)), the methods of feasible directions (Zoutendijk (1960)), the SOLVER algorithm (Wilson (1963)), Robinson's algorithm (1972) and its refinements as reported in Bräuniger (1977), Best, Bräuniger, Ritter and Robinson (1979) and Van der Hoek (1979), and, finally, the Recursive Quadratic Programming Algorithms mentioned above.

The reduction methods to be considered in chapter IV use reduced problems

which evolve from the original problem by the linearization of the currently most relevant constraints whereas a linear penalty-like term is added to the original objective function. The numerical aspects of the resulting algorithms will be treated in chapter V.
The last chapter, chapter VI, concerns the design of computational experiments to compare the reduction methods developed. A discussion of the computational results concludes this chapter.
The appendices A-E give additional information on the test functions used and the implementations of the algorithms.

I.2. Preliminary remarks

This section concerns several basic notations and definitions. It is devided into two parts: I.2.1. on the set of constraints and I.2.2. on optimality conditions.

I.2.1. The set of constraints

Let us assume that the *inequality* constraints of problem (1.1)are labelled by the indices i = 1, ..., p and the *equality* constraints by i = p+1, ..., m, respectively. Then, at a current iteration point x_k, k = 1, 2, ... the *status* of a constraint $c_i(x)$ will be one of the three following types:

passive : $c_i(x_k) > 0$ $\quad i \in \{1,\dots,p\}$

violated : $c_i(x_k) < 0$ $\quad$ if $i \in \{1,\dots,p\}$

or

$c_i(x) \neq 0$ $\quad$ if $i \in \{p+1,\dots,m\}$

active (binding): $c_i(x) = 0$ $\quad i \in \{1,\dots,m\}$

The *feasible region* of problem (1.1) consists of all $x \in E^n$ which satisfy $c_i(x) \leq 0$, $i = 1,\dots,p$ *and* $c_i(x) = 0$, $i = p+1,\dots,m$. A feasible point $x \in E^n$ is said to be a *regular point* of the constraints if at x the gradient vectors of the active constraints are linearly independent.
In the reduction methods discussed we shall distinguish at each current iteration point x_k, k = 1,2,... between more and less relevant constraints. The aim is to recognize as soon as possible the constraints which will be active at the optimum x^*. The currently more relevant constraints are considered to be liable to become active at x^*. Hence they are considered separately. Their indices constitute the so-called *active set of constraints* at x_k, denoted by $I(x_k)$ or I_k. These constraints will be treated as equality constraints.
Usually such an active set contains at least the indices of the currently binding constraints. The decision whether some passive or some violated constraint will join this actice set as well, depends on the design of the particular reduction method. E.g., different *active set strategies* will be applied in the reduction methods of chapters III and IV respectively, with as a common goal that $I(x^*)$ will be obtained in an early stage of the

iteration process. Thus, we expect that $I(x_k) = I(x^*)$ for $k \geq K$, where K is some acceptably small natural number. The design of an active set strategy should prevent the *same* constraints from entering and leaving the active set repeatedly. This phenomenon, known as *'zigzagging'*, can lead to nonconvergence or even to convergence to the wrong point.

I.2.2. Optimality conditions

The well known necessary and/or sufficient optimality conditions for problem (1.1) use a *Lagrangian* function $L(x,u,v)$, associated with problem (1.1):

(1.3) $$L(x,u,v) = F(x) - \sum_{i=1}^{p} u_i c_i(x) - \sum_{i=p+1}^{m} v_i c_i(x)$$

where u_i, $i = 1, \ldots, p$ and v_i, $i = p+1, \ldots, m$ are the *Lagrangian multipliers* of the inequality and the equality constraints respectively.
The first- and second order optimality conditions below are of course well known cf., for instance Fiacco and Mc Cormick (1968) or Luenberger (1973).

The *first order necessary conditions* or *Kuhn-Tucker conditions* can be defined as follows:

Let x^* be a relative minimum point for the problem (1.1) and suppose x is a regular point for the constraints. Then there exist vectors $u \in E^p$ and $v \in E^{m-p}$ such that

(1.4) $$\nabla_x L(x^*,u,v) = 0$$

(1.5) $$u_i c_i(x^*) = 0 \qquad i = 1, \ldots, p$$

(1.6) $$c_i(x^*) = 0 \qquad i = p+1, \ldots, m$$

(1.7) $$c_i(x^*) \geq 0 \qquad i = 1, \ldots, p$$

(1.8) $$u_i \geq 0 \qquad i = 1, \ldots, p$$

□

The points $z = (x\ ,u,v) \in E^{n+m}$ which satisfy (1.4)-(1.8) are referred to as *first order Kuhn-Tucker points* of (1.1). The active set at $z = (x,u,v)$ will sometimes be denoted by $I(z)$.

If a constraint $c_i(x)$ is active at the optimum x^* with $u_i = 0$ it is called *weakly active* or *degenerate* as opposed to *strongly active* constraints which possess a positive Lagrangian multiplier.
The complementarity formulated in (1.5) will be referred to as *strict complementary slackness* if (1.5) holds and at least one of its factors is positive. Strict complementarity for all i means that there are no weakly active inequality constraints at x^*. The Lagrangian multipliers are uniquely determined if at a regular point x^* the equations (1.4)-(1.6) are satisfied with strict complementary slackness in (1.5).
The *second order conditions* are an extension of the first order necessary conditions in which the Hessian matrix of $L(x,u,v)$ is required to be positive definite in the subspace orthogonal to the normals in x^* of the equality constraints and the strongly active inequality constraints. The resulting second order sufficiency conditions for problem (1.1) are:

Let all problem functions be twice continuously differentiable. Sufficient conditions that a regular point x^* be a strict relative minimum point of (1.1) is that there exist vectors $u \in E^p$ and $v \in E^{m-p}$ such that

(1.4) - (1.8) are satisfied, together with:

(1.9) $\nabla^2_{xx} L(x^*,u,v)$ is positive definite in the subspace M defined by

$$M = \{y \in E^n \mid \nabla^T c_i(x^*) y = 0 \qquad \forall i \in I(x^*)\}$$

where $I(x^*) = \{i : c_i(x^*) = 0 \text{ and } u_i > 0,\ i = 1,\dots,p\} \cup \{p+1,\dots,m\}$.

□

The condition (1.9) which requires $\nabla^2_{xx} L(x^*,u,v)$ to be positive definite in a subspace of E^n will be applied in chapter III to generate a sequence of positive definite matrices that approximate to $\nabla^2_{xx} L(x^*,u,v)$.
Functions, such as $L(x^*,u,v)$, with positive definite Hessian matrix, will be said to possess *positive curvature*.

II. A COMPUTATIONAL COMPARISON OF SELF SCALING VARIABLE METRIC ALGORITHMS

II.1. Introduction

Most algorithms for constrained or unconstrained nonlinear programing have in common that along currently defined search directions a sequence of iteration points is generated by performing a line search. Hence both the search direction and the unidimensional search procedure characterise and distinguish these algorithms. The importance of a suitable definition of search direction is even greater as reduction methods, especially transformation methods, solve constrained nonlinear programming problems by solving a sequence of unconstrained nonlinear programming problems. The latter problems are solved efficiently by performing a linesearch along a currently defined direction of search.

Concerning the generation of search directions, well known methods as steepest descent and Newton-Raphson have been improved in the last two decades by conjugate direction methods (e.g., Fletcher-Reeves, Polak-Ribière) and quasi-Newton or variable metric methods (e.g., Davidon, Fletcher and Powell). In the last mentioned class of methods, the subclass of Self Scaling Variable Metric methods (SSVM) was introduced in Oren and Luenberger (1974) and Oren (1974a). These methods were further developed in Oren and Spedicato (1976) and in Shanno and Phua (1978a). These recent algorithms for unconstrained optimization focus on the solution of badly scaled problems. This chapter describes a uniform computational comparison of these algorithms. It is performed to get better insight in their relative behaviour and to verify empirically their ability to solve badly scaled problems. Hence special attention is paid to the effect of increasingly bad scaling, the influence of the accuracy of the line search and of the dimension of the problem.

A further reason to design these experiments is the fact that reported numerical results in literature are based on rather different test batteries. Surprisingly, up to now experiments have not focused on the main goal of these algorithms: their ability to handle badly scaled problems where the spectrum of eigenvalues of the matrix R_1, which is a measure of the discrepancy between the current inverse Hessian approximation and the true inverse Hessian, does not contain the unit element.
This is why a suitable battery of testproblems will be suggested. The

description of the design of the experiments and their results are preceded by a brief presentation of the theoretical backgrounds of these algorithms, which can be found in more detail in the above mentioned references. The classical Davidon-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS)-algorithms will be considered as well.

II.2.1. Self Scaling Variable Metric algorithms

The problem considered in this chapter is the nonlinear, unconstrained minimization problem

$$(2.1) \qquad \min_{x} F(x)$$

where $x \in E^n$, the n-dimensional Euclidian space. The objective function $F(x)$ is supposed to be a sufficiently differentiable convex function of $x \in E^n$.
As twice continuously differentiable convex functions $F(x)$ can be approximated in a neighbourhood of their optimum x^* by the first terms of their Taylor series expansion, we shall only consider *quadratic* convex functions $F(x)$ in the analysis and development of algorithms for unconstrained optimization.
Variable metric or quasi-Newton algorithms generate sequences of iteration points x_k, search directions p_k and approximations H_k of the inverse Hessian matrix of $F(x)$ at x^*, on the basis of such information as the previous step $s_{k-1} = x_k - x_{k-1}$ and the gradient difference vector $y_{k-1} = g_k - g_{k-1} = \nabla F(x_k) - \nabla F(x_{k-1})$.

A general, stepwise description of quasi-Newton methods is:

Step 1. Initialization: given an arbitrary starting point x_0 with $g_0 = \nabla F(x_0)$, a positive definite symmetric matrix H_0 is chosen as first approximation of the inverse Hessian.
Go to step 2.

Step 2. At point x_k, $k = 0, 1, 2, \ldots$ define x_{k+1} as

$$(2.2) \qquad x_{k+1} = x_k - \alpha_k H_k g_k,$$

where $\alpha_k > 0$ is determined by a line search along the direction

(2.3) $p_k = - H_k g_k$

Go to step 3

Step 3. Stop in case certain (to be specified) termination criteria are met. Otherwise continue with step 4

Step 4. Update H_k, put $k := k + 1$ and go to step 2.

The method of steepest descent and Newton's method are special cases which arise from taking $H_k = I_n$ for all k and $H_k = [\nabla^2 F(x_k)]^{-1}$ with $\alpha_k = 1$ respectively.

An important theorem on the *global convergence* (i.e., convergence from any starting point x_0) of quasi-Newton algorithms applied to a quadratic objective function originates with Luenberger:

THEOREM 2.1 (Luenberger, 1973)

For a positive definite quadratic objective function F(x) the quasi-Newton algorithms converge to the unique optimum x^* of F(x) for any initial point x_0.

At every step the following inequality holds:

$$F(x_{k+1}) - F(x^*) \leq \left\{\frac{\kappa(R_k) - 1}{\kappa(R_k) + 1}\right\}^2 (F(x_k) - F(x^*)) \tag{2.4}$$

where $\kappa(R_k)$ is the condition number (the ratio of its largest and its smallest eigenvalue) of the matrix $R_k = G^{\frac{1}{2}} H_k G^{\frac{1}{2}}$. □

The matrix R_k is used as a measure of the difference between H_k and $G^{-1} = (\nabla^2 F(x^*))^{-1}$. Clearly $R_k = I$ means $H_k = G^{-1}$.

It is obvious from theorem 2.1 that convergence is accelerated if the quotients $\left\{\frac{\kappa(R_k)-1}{\kappa(R_k)+1}\right\}^2$, which can be viewed as 'local convergence ratios', form a decreasing null sequence. Thus preferably $\lim_{k\to\infty} \kappa(R_k) = 1$ should hold. The fact that this property does not generally hold for variable metric algorithms was a motivation to search for a subclass of algorithms for which $\lim_{k\to\infty} \kappa(R_k) = 1$ is satisfied. Variable metric algorithms are known to have a number of less favourable properties and this stimulated additional research, such as in the direction of the influence of the accuracy of the applied line search on the efficiency of the algorithms, the possible singularity of the matrices H_k (Mc Cormick and Pearson (1969), Lenard (1976), and Powell (1977b)) and the sensitivity to scaling of the objective function (Bard (1968)). However, the following favourable properties of variable metric algorithms should be preserved:

(i) The matrices H_k, $k = 1, 2, \dots$ are positive definite, provided that H_0 is chosen to be positive definite.

(ii) If $F(x)$ is a positive definite quadratic function and $H_0 = I_n$, the algorithm is a conjugate gradient method and thus converges in at most n steps.

(iii) If $F(x)$ is a positive definite quadratic function and the algorithm requires all n steps, then $H_n = G^{-1}$.

The Self Scaling Variable Metric (SSVM) algorithms, presented in Oren and Luenberger (1974), satisfy all the above mentioned requirements. The main characteristic of these algorithms is the way in which they update the inverse Hessian approximation in step 4 of the stepwise description given above. The individual elements of the subclass arise form the choice of two parameters ϕ_k and θ_k in the formulae (2.5) - (2.7). All these updates are elements of Huang's family of update formulae (Huang (1970), Osborne (1972)). Essentially, the results are an extension of work of Fletcher (1970a) who developed update formulae with guaranteed monotone convergence of the eigenvalues of the matrices H_kG.
The update formulae for the SSVM algorithms are:

$$(2.5)\qquad H_{k+1} = \left\{H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \theta_k v_k v_k^T\right\} \gamma_k + \frac{s_k s_k^T}{s_k^T y_k}$$

with

$$(2.6)\qquad v_k = (y_k^T H_k y_k)^{\frac{1}{2}} \left\{\frac{s_k}{s_k^T y_k} - \frac{H_k y_k}{y_k^T H_k y_k}\right\}$$

and

$$(2.7)\qquad \gamma_k = \frac{s_k^T y_k}{y_k^T H_k y_k} \cdot (1 - \phi_k) + \frac{s_k^T g_k}{g_k^T H_k y_k} \cdot \phi_k$$

Particular choices of the parameters ϕ_k, θ_k (note that γ_k is determined by ϕ_k and vice versa) yield:

(i) The DFP-update where $\gamma_k = 1$ and $\theta_k = 0$ for all k.

(ii) The BFGS-update where $\gamma_k = 1$ and $\theta_k = 1$ for all k.

(iii) SSVM-updates for all other combinations satisfying some restrictions on the values of the parameters.

Note that the factor γ_k is determined by the value of ϕ_k and the currently available information. It will turn out to be a scaling factor of the objective function. The terminology self scaling will be used for algorithms using formulae (2.5) - (2.7) if for any fixed positive definite quadratic function the parameters θ_k and γ_k are automatically selected such that $\kappa(R_{k+1}) \leq \kappa(R_k)$ for all k.

Before proceeding with the presentation of the theoretical background, we illustrate the effect of scaling the objective function by an example. We apply three algorithms to minimize

(2.8) $$F(x) = 30x_1^2 + 20x_2^2$$

starting from $x_0^T = (1, 1)$.

The values of $\kappa(R_k)$ are calculated for the following algorithms.

Algorithm 1. DFP: $\gamma_k = 1$ and $\theta_k = 0$ for all k.

Algorithm 2. DFP after scaling the objective function. In this example a scaling factor of 40 is used which transforms the eigenvalues of R_0 into 1 and $1\frac{1}{2}$.

Algorithm 3. SSVM with $\theta_k = \phi_k = 0$ for all k.

The next tables contain for these algorithms the iteration matrices H_k, G and R_k for k = 0, 1, while λ_1 and λ_2 are the eigenvalues of R_k for k = 0,1. It will be clear from the last line of table 2.2 that the condition number $\kappa(H_1) = \kappa(R_1)$ of the inverse Hessian approximation is improved by applying a scaling procedure.

Table 2.1 Iteration matrices at the starting point

	Algorithm 1	Algorithm 2	Algorithm 3
H_0	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
G	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$	$\begin{pmatrix} 1\frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$
R_0	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$	$\begin{pmatrix} 1\frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$
λ_1	40	1	40
λ_2	60	$1\frac{1}{2}$	60
$\kappa(R_0)$	$1\frac{1}{2}$	$1\frac{1}{2}$	$1\frac{1}{2}$
$\kappa(G)$	$1\frac{1}{2}$	$1\frac{1}{2}$	$1\frac{1}{2}$

Exact line minimization in the direction $- \alpha H_0 g_0$ and application of (2.5) - (2.7) yields: table 2.2

Table 2.2 Iteration matrices after one iteration

	Algorithm 1	Algorithm 2	Algorithm 3
H_1	$\begin{pmatrix} .17781 & -.36256 \\ -.36256 & .84077 \end{pmatrix}$	$\begin{pmatrix} .67923 & -.02828 \\ -.02828 & 1.06362 \end{pmatrix}$	$\begin{pmatrix} .01584 & .00188 \\ .00188 & .02773 \end{pmatrix}$
R_1	$\begin{pmatrix} 10.6683 & 21.7537 \\ -14.5025 & 33.6306 \end{pmatrix}$	$\begin{pmatrix} 1.01885 & -.04242 \\ -.02828 & 1.06362 \end{pmatrix}$	$\begin{pmatrix} .94997 & .11253 \\ .07502 & .83118 \end{pmatrix}$
λ_1	1	1	.781165
λ_2	43.298964	1.082475	1
$\kappa(R_1)$	43.298964	1.082475	1.280139

We proceed now with a brief description of the theoretical background as found in Oren and Luenberger (1974), without presenting proofs of the theorems. First we introduce a shortened notation for formulae (2.5) and (2.6):

$$(2.9)\qquad H^{\theta}(H,\gamma,s,y) = \left\{H - \frac{Hyy^TH}{y^THy} + \theta vv^T\right\}\gamma + \frac{ss^T}{s^Ty}$$

with

$$(2.10)\qquad v = (y^THy)^{\frac{1}{2}}\left\{\frac{s}{s^Ty} - \frac{Hy}{y^THy}\right\}$$

In (2.9) and (2.10) the subscripts are suppressed as we are mainly interested in the change of eigenvalues after one particular iteration.
The following fundamental lemma concerns a scaled problem (with γH as Hessian): part (i). The update formula is given as a combination of two elementary formulae (result (ii) of the lemma, the restriction $\theta \in [0,1]$ will turn out to be necessary) and a duality relation is derived (part (iii)).

LEMMA 2.1

Let $H^{\theta}(H,\gamma,s,y)$ be defined by (2.9) and (2.10). Then for any symmetric non-singular matrix H, non-zero vectors $s, y \in E^n$ and scalars θ, $\gamma (\neq 0)$,

(i) $H^{\theta}(H,\gamma,s,y) = H^{\theta}(\gamma H,1,s,y)$

(ii) $H^{\theta}(H,\gamma,s,y) = (1-\theta)\, H^{0}(H,\gamma,s,y) + \theta H^{1}(H,\gamma,s,y)$

(iii) $[H^{1}(H,\gamma,s,y)]^{-1} = H^{0}(H^{-1},\frac{1}{\gamma},y,s)$ □

As $R = G^{\frac{1}{2}}HG^{\frac{1}{2}}$, we expect similar relations to hold for the updating of R. This is expressed in the next lemma. For simplicity in notation again the indices k are suppressed. The indices (k+1) are replaced by an upper bar, hence $D = D_k$ and $\bar{D} = D_{k+1}$ etc.

LEMMA 2.2

Let $H^{\theta}(H,\gamma,s,y)$ be defined by (2.9), (2.10) and let G be a positive definite symmetric matrix. Assume $s^Ty > 0$ and $y = Gs$. Then for $R = G^{\frac{1}{2}}HG^{\frac{1}{2}}$ and $z = G^{\frac{1}{2}}s$ the following relation holds:

$$(2.11)\qquad \bar{R} = H^{\theta}(R,\gamma,z,z) \qquad □$$

As R is nonsingular if $\gamma \neq 0$ and H is nonsingular, lemma 2.1 applies to $\bar{R}$ with $z = s = y$ and $R = H$, thus yielding relations for the updating of R.

We intend to analyse the eigenvaluestructure of $\bar{R} = H^{\theta}(R,\gamma,z,z)$. This will be carried out in two steps.
First, in theorem 2.2, relations are stated between the eigenvalues of two general symmetric (nxn) matrices, say S and T, that satisfy

(2.12) $$T = S - \frac{Srr^T S}{r^T Sr} + \frac{rr^T}{r^T r}$$

with $r \in E^n$, $r \neq 0$. This means that T is obtained by adding two rank 1 matrices to the matrix S.

Second, the results obtained are extended to $H^\theta(R,\gamma,z,z)$. The results of theorem 2.2 are extensions of the following lemma:

LEMMA 2.3 (interlocking eigenvalue lemma, Loewner, 1957)

Let A be a symmetric (nxn)-matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ and let $a \in E^n$ be an arbitrary vector. Let the matrix B be defined by $B = A + aa^t$, with eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$
Then: $\lambda_1 \leq \mu_1 \leq \lambda_2 \leq \mu_2 \leq \ldots \leq \lambda_n \leq \mu_n$. □

A direct application of this lemma to

(2.13) $$P = S - \frac{Srr^T S}{r^T Sr},$$

and

(2.14) $$T = P + \frac{rr^T}{r^T r}$$

yields theorem 2.2:

THEOREM 2.2

Let S be a positive definite symmetric matrix with eigenvalues $0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ and let $r \in E^n$ be a non- zero vector. Let the matrix T be defined by (2.12) with eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$. Then there are three possibilities:

(i) if $\lambda_1 \geq 1$, then $\mu_1 = 1$ and $1 \leq \lambda_{i-1} \leq \mu_i \leq \lambda_i$ for $i = 2,3,\ldots,n$

(ii) if $\lambda_n \leq 1$, then $\mu_n = 1$ and $\lambda_i \leq \mu_i \leq \lambda_{i+1} \leq 1$ for $i = 1,2,\ldots,n-1$

(iii) if $\lambda_1 \leq 1 \leq \lambda_n$ and the index J is such that $\lambda_J \leq 1 \leq \lambda_{J+1}$, then
$\lambda_1 \leq \mu_1 \leq \lambda_2 \leq \mu_2 \leq \ldots \leq \lambda_J \leq \mu_J \leq 1 \leq \mu_{J+1} \leq \ldots \leq \mu_n \leq \lambda_n$
and at least one of the two eigenvalues μ_J, μ_{J+1} equals unity.

□

In figure 2.1 we illustrate the way in which the eigenvalues (and consequently the condition numbers) change in the construction of the matrix T from P and S. The eigenvalues of the matrices T, P and S are denoted by μ_i, ζ_i and λ_i respectively. Then the 3 possible cases of theorem 2.2 are:

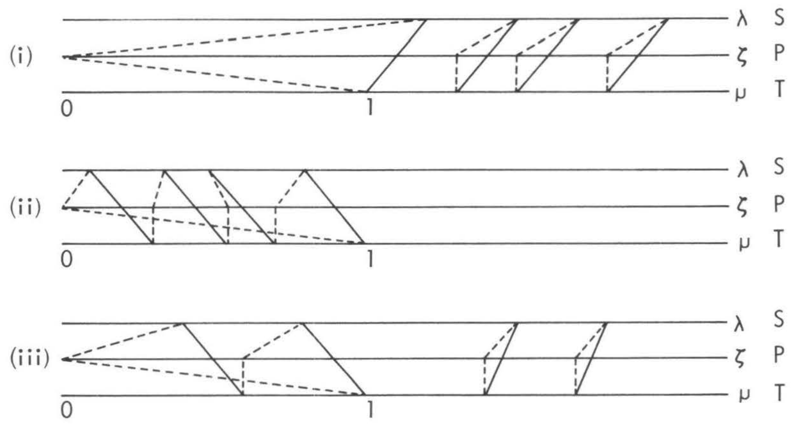


Figure 2.1

Note that in all cases the smallest eigenvalue λ, of S is transformed into the eigenvalue $\zeta = 0$ of P, which in turn becomes the eigenvalue $\mu = 1$ of the final matrix T.

It will be clear from theorem 2.2, especially part (iii), that in order to guarantee that T will have a lower condition number than S, the interval spanned by the eigenvalues of S must contain the unit element. This observation forms the basis of the development of the SSVM algorithms.

An intermediate result relates the eigenvalues

$$\mu_1^\theta(\gamma) \le \mu_2^\theta(\gamma) \le \dots \le \mu_n^\theta(\gamma) \text{ of } \bar{R}^\theta(\gamma) = H^\theta(R,\gamma,z,z)$$

for $\theta \in [0,1]$ to the corresponding eigenvalues for $\theta = 0$ and $\theta = 1$.

THEOREM 2.3

Let $\bar{R}^\theta = H^\theta(R,\gamma,z,z)$ be given by (2.11) for some fixed positive definite matrix R and $z \in E^n$, $z \neq 0$. Then, for $\theta \in [0,1]$ and $\gamma > 0$,

(2.15) $$\mu_i^0(\gamma) \le \mu_i^\theta(\gamma) \le \mu_i^1(\gamma) \qquad \text{for } i = 1,2,\dots,n$$ □

The next theorem, which relates the eigenvalues of the matrices R and $\bar{R}^{\theta}(\gamma)$, now follows readily,

THEOREM 2.4

Let $\bar{R}^{\theta}(\gamma) = H^{\theta}(R,\gamma,z,z)$ be given by (2.11) for a fixed positive definite matrix R and $z \in E^n$, $z \neq 0$. Let the eigenvalues of R and $\bar{R}^{\theta}(\gamma)$ be respectively $\lambda_1 \le \lambda_2 \le \dots \le \lambda_n$ and $\mu_1^{\theta}(\gamma) \le \mu_2^{\theta}(\gamma) \le \dots \le \mu_n^{\theta}(\gamma)$. Then, provided that $\theta \in [0,1]$ and $\gamma > 0$ there are three possible cases:

(i) if $\gamma\lambda_1 \ge 1$, then $\mu_1^{\theta}(\gamma) = 1$ and $\gamma\lambda_{i-1} \le \mu_i^{\theta}(\gamma) \le \gamma\lambda_i$ for $i = 2,3, \dots, n$.

(ii) if $\gamma\lambda_n \le 1$, then $\mu_n^{\theta}(\gamma) = 1$ and $\gamma\lambda_i \le \mu_i^{\theta}(\gamma) \le \gamma\lambda_{i+1} \le 1$ for $i = 1,2, \dots, n-1$

(iii) if $\gamma\lambda_1 \le 1 \le \gamma\lambda_n$ and the index J is such that $\gamma\lambda_J \le 1 \le \gamma\lambda_{J+1}$, then $\gamma\lambda_1 \le \mu_1^{\theta}(\lambda) \le \gamma\lambda_2 \le \dots \le \gamma\lambda_J \le \mu_J^{\theta}(\gamma) \le 1 \le \mu_{J+1}^{\theta}(\gamma) \le$ $\le \gamma\lambda_{J+1} \le \dots \le \mu_n^{\theta}(\gamma) \le \gamma\lambda_n$, and at least one of the eigenvalues $\mu_J^{\theta}(\gamma)$, $\mu_{J+1}^{\theta}(\gamma)$ equals unity. □

COROLLARY 1

With $\bar{R}^{\theta}(\gamma)$, R, λ_i and $\mu_i^{\theta}(\gamma)$ as in theorem 2.4 there holds

$$|\mu_i^{\theta}(1) - 1| \le |\lambda_i - 1| \text{ for } i = 1,2, \dots, n \tag{2.16}$$

Proof

The result is obvious from the observation that $\lambda_i \le \mu_i^{\theta}(1) \le 1$ or $1 < \mu_i^{\theta}(1) \le \lambda_i$ for all i. □

As a result the eigenvalues of the matrices generated successively tend monotonically to the unit element. This is exactly the result in Fletcher (1970a).

COROLLARY 2

With the same notation as corollary 1 and $\kappa(.)$ denoting to be the conditionnumber of a matrix, then for $\theta \in [0,1]$ and $\gamma > 0$ there are three cases:

(i) if $\gamma\lambda_1 \geq 1$, then $\gamma\lambda_n \geq \kappa(\bar{R}^{\theta}(\gamma)) \geq \gamma\lambda_{n-1}$,

(ii) if $\gamma\lambda_n \leq 1$, then $1/\gamma\lambda_1 \geq \kappa(\bar{R}^{\theta}(\gamma)) \geq 1/\gamma\lambda_2$,

(iii) if $\gamma\lambda_n \geq 1 \geq \gamma\lambda_1$, then $\kappa(\bar{R}^{\theta}(\gamma)) \leq \kappa(R)$. □

As we are looking for matrices R_k with decreasing condition number, case (iii) of corollary 2 is the most interesting one. The necessity of the condition that $\theta \in [0,1]$ for every value of the suppressed index k follows from a counterexample due to Fletcher (1970a) in which both $\theta \leq -\varepsilon$ and $\theta \geq 1+\varepsilon$ for $\varepsilon \in (0,1)$ lead to a contradiction. Hence it remains to define factors γ which satisfy $\gamma\lambda_n \geq 1 \geq \gamma\lambda_1$. As it is rather time consuming to evaluate the eigenvalues λ_1 and λ_n, we are interested in scaling factors γ based on currently available information and which still satisfy : $\gamma\lambda_n \geq 1 \geq \gamma\lambda_1$. Oren (1974a) introduced a convex class of scaling factors which meet these requirements. Let H be a nonsingular symmetric nxn matrix and $s,y \in E^n$ with $s \neq 0$, $y \neq 0$. Then the scalar $\gamma^{\phi}(H,s,y)$ is defined by

$$\gamma^{\phi}(H,s,y) = (1-\phi)\frac{s^Ty}{y^THy} + \phi\frac{s^Tg}{g^THy} \tag{2.17}$$

If H is positive definite, $s^Ty > 0$ and $\phi \in [0,1]$ then $\gamma^{\phi}(H,s,y)$ is strictly positive. The next theorem states that $\gamma^{\theta}(H,s,y)$ as defined in (2.17) satisfies $\gamma\lambda_n \geq 1 \geq \gamma\lambda_1$ for all $\phi \in [0,1]$ automatically.

THEOREM 2.5

Let $s,y \in E^n$, $s \neq 0, y \neq 0$ with $s^Ty > 0$. The matrices H and G are positive definite symmetric such that $y = Gs$ while the positive definite matrix R is defined by $R = G^{\frac{1}{2}}HG^{\frac{1}{2}}$. Then for all $\phi \in [0,1]$ there holds

$$\frac{1}{\lambda_n} \leq \gamma^{\phi}(H,s,y) \leq \frac{1}{\lambda_1} \quad ,$$

where λ_1 and λ_n are the smallest and the largest eigenvalue of R respectively.

Proof

First we rewrite (2.17) as

$$\gamma^{\phi}(H,s,y) = (1-\phi)\frac{H^Ty}{y^THy} + \phi\frac{s^TH^{-1}s}{s^Ty} \quad \text{using} \tag{2.18}$$

(2.19) $$\frac{s^T H^{-1} s}{s^T y} = \frac{s^T g}{g^T H y} \qquad \text{for } s = -\alpha H g.$$

As $\gamma^{\phi}(H,s,y)$ is defined as a convex combination of $\gamma^{1}(H,s,y)$ and $\gamma^{0}(H,s,y)$ it suffices to prove the theorem for $\gamma = 0$ and $\gamma = 1$. These proofs can be found in Oren (1974a) and, slightly modified, in Van der Hoek and Dijkshoorn (1979). □

Conclusion

We found in lemma 2.1 (i) that

$$H^{\theta}(\gamma H,1,s,y) = H^{\theta}(H,\gamma,s,y)$$

I.e., scaling of the objective function by multiplying the inverse Hessian approximation before updating by a constant ξ can be implemented in SSVM algorithms by simply choosing $\gamma = \xi$. So γ can be interpreted as a scaling factor and varying γ from iteration to iteration has the effect of re-scaling the objective function.

The scaling factors defined by (2.17) can be calculated from the information gathered in the preceding step as expressed by the vectors s,g and y and by the matrix H. The resulting algorithm is invariant under scaling of the objective function and/or the variables. The last remark is proved in the next theorem.

THEOREM 2.6

Let H_k, x_k, θ_k and ϕ_k be defined as above. Suppose that the sequences $\{H_k\}$, $\{x_k\}$ and $\{\hat{H}_k\}$, $\{\hat{x}_k\}$ are generated by application of the algorithm to the functions $F(x)$ and $\alpha F(\beta x)$ respectively ($\alpha > 0$, $\beta > 0$). For the initialisation we assume $\hat{H}_0 = \delta H_0$ ($\delta > 0$) and $\beta\hat{x}_0 = x_0$. Both applications use the same sequences $\{\theta_k\}$ and $\{\phi_k\}$. Then, for a twice continuously differentiable function $F(x)$, we have that

$$\hat{H}_k = \frac{H_k}{\alpha\beta^2} \text{ and } \hat{x}_k = \frac{x_k}{\beta} \text{ for all k.}$$

Proof

The proof follows immediately from substitution in the update formulae. □

Conclusion

The variable metric algorithms presented above generate positive definite matrices H_k, the condition numbers of the corresponding R_k matrices form a monotone decreasing sequence and the algorithms are self scaling, and invariant up to scaling of the objective function and the variables. In the course of the iterations the matrices H_k increasingly resemble the true inverse Hessian. This will provide a good local convergence rate even without performing a line search (by simply taking the Newton steplength 1). Thus we expect to find in our experiments good results with inexact line searches, as well as a decrease in the influence of roundoff errors.

II.2.2. Optimally Conditioned Self Scaling Algorithms

It will be clear from chapter II.2.1. that there is still a wide variety of possible choices of the SSVM-parameters $\gamma(\phi)$ and θ. A first trial to find preferable parameter combinations was performed by Oren. He reported in Oren (1974b) the results of experiments in which the 9 possible pairs (ϕ_k,θ_k) in the set $\{(\phi,\theta)\,|\,\phi,\theta \in \{0,0.5,1\}\}$ were substituted in formulae (2.5) - (2.7). Besides that this study contained two devices to generate parameters γ_k and θ_k from currently available information on the objective function. According to these rules γ_k is selected as close as possible to unity and θ_k is chosen such as to offset an estimated bias in det (H_kG) relative to unity. The main result of Oren was that he showed that a further improvement of the SSVM algorithms could be expected by a proper selection of the parameters.
In a subsequent paper, Oren and Spedicato (1976), a theory was developed to obtain *a sharper bound on the condition number of the positive definite updates H_k*. A low condition number of H_k is desirable from a numerical point of view since it will reduce the round-off error in the determination of the succeeding points (formula (2.2)) and thus it will improve the numerical stability of the resulting algorithm.
As a result of their analysis Oren and Spedicato present the following theorem which characterizes *optimally conditioned* updates.

THEOREM 2.7 (Oren and Spedicato, 1976)

The matrix H_{k+1} is optimally conditioned if and only if either $\pi\tau = \sigma^2$ or

(2.20) $$\theta = \frac{\sigma(\pi - \gamma\sigma)}{\gamma(\pi\tau - \sigma^2)}$$ □

Here σ, τ and π are defined by

(2.21) $$\sigma = s^T y$$

(2.22) $$\tau = y^T H y$$

(2.23) $$\pi = s^T H^{-1} s = \frac{s^T y g^T s}{g^T H y}$$

Imposing this relation on the SSVM updates yields the one parameter class of Optimally Conditioned Self Scaling Updates. Following previous publications the resulting strategies will be called switches I-IV.

Switch I

(2.24) If $\frac{\pi}{\sigma} \leq 1$, choose $\gamma = \frac{\pi}{\sigma}$ and $\theta = 0$;

(2.25) If $\frac{\sigma}{\tau} \geq 1$, choose $\gamma = \frac{\sigma}{\tau}$ and $\theta = 1$;

(2.26) If $\frac{\sigma}{\tau} \leq 1 \leq \frac{\pi}{\sigma}$, choose $\gamma = 1$ and $\theta = \frac{\sigma(\pi - \sigma)}{\pi\tau - \sigma^2}$.

Switch II

(2.27) $$\gamma = \left(\frac{\pi}{\tau}\right)^{\frac{1}{2}} \text{ and } \theta = \frac{1}{1 + \left(\frac{\tau\pi}{\sigma^2}\right)^{\frac{1}{2}}}.$$

Switch III

(2.28) If $\frac{\pi}{\sigma} \leq 1$, choose $\gamma = \frac{\pi}{\sigma}$ and $\theta = 0$;

(2.29) If $\frac{\sigma}{\tau} \geq 1$, choose $\gamma = \frac{\sigma}{\tau}$ and $\theta = 1$;

(2.30) If $\frac{\sigma}{\tau} \leq 1 \leq \frac{\pi}{\sigma}$, choose $\gamma = 1$ and $\theta = \frac{\sigma(\tau - \sigma)}{\pi\tau - \sigma^2}$.

Switch IV

(2.31) $$\gamma = \frac{\pi}{\tau} \text{ and } \theta = \tfrac{1}{2}.$$

These strategies with automatically determined parameters were succeeded by a paper of Shanno and Phua which will be discussed in the next section.

II.2.3. Initial Scaling of BFGS

The principal drawback of the SSVM updates is that for a quadratic object function the sequence of matrices thus generated fails to converge to the inverse Hessian matrix, unless either γ_k is appropriately chosen or H_0 is chosen to be of a special form. Starting from this observation Shanno and Phua consider two possible initial scalings of H_0 which satisfy the invariance of the algorithms under scaling of the objective function. Moreover these scalings appear to improve the numerical stability of the resulting algorithms.
With respect to this proposed initial scaling the following lemma can be proved. It states a relation between initial scaling and the application of an appropriate SSVM update. In this lemma initial scaling means that $H_0 = I$ is used to determine x_1 while using a steplength α_0. After the determination of x_1 but before updating H_0, we now scale H_0 by

$$(2.32) \qquad \tilde{H}_0 = \alpha_0 H_0$$

and then compute H_1 using the BFGS update formulae and $\tilde{H}_0$.

LEMMA 2.4

Initial scaling of the inverse Hessian approximation by a factor α followed by the application of the BFGS update formulae is equivalent to the application of the SSVM update formulae with $\gamma = \alpha$ and $\theta = 1$.

Proof

Substitution of $\tilde{H}_0 = \alpha H_0$ in formulae (2.5) - (2.7) yields

$$H_1 = \left\{ H_0 - \frac{H_0 y_0 y_0^T H_0}{y_0^T H_0 y_0} + v_0 v_0^T \right\} \alpha + \frac{s_0 s_0^T}{s_0^T y_0}$$

which proves the lemma. $\square$

The use of the steplength α_0 as a scaling factor is motivated by the fact that if H is a good approximation to the true inverse Hessian, then α will be equal to 1. The interpretation of the lemma is that instead of scaling H by γ at each step, like in SSVM algorithms, one can scale only the matrix H_0. After this first scaling the approximate Hessian is never rescaled. This idea of performing a simple initial scaling was first suggested by Shanno and Phua (1978a). Another advantage of this approach is that the resulting algorithm still uses the BFGS update formulae which gives a robust and efficient algorithm for unconstrained optimization. Both computational and theoretical studies confirm this (see e.g., Van der Hoek and Dijkshoorn (1979) and Nazareth (1979)). Besides the initial scaling as given in (2.32) Shanno and Phua combine the relation expressed in (2.20) with $\theta = 1$ for BFGS, and thus obtain a second alternative for initial scaling of the BFGS-algorithm:

$$(2.33) \qquad \tilde{H}_0 = \frac{\sigma}{\tau} H_0$$

Both initial scalings will be considered in the comparison of section II.3, where special attention will be paid to the question how the efficiency of the algorithms depends on the conditioning of the problem, on the number of variables and on the accuracy of the applied line search. Finally, for the class of homogeneous functions, as introduced by Jacobson and Oksman (1970), BFGS algorithms, initially scaled or not, can be proved to be inferior to e.g. DFP. This proof relies on a comparison of resulting step-size predictions and the definition of what they call a homogeneous function. That is a function F(x) such that

$$(2.34) \qquad F(x) = \beta^{-1}(x - x^*)^T g(x) + F(x^*),$$

with x^* the minimizer, β the degree of homogeneity and $g(x) = \nabla F(x)$.

II.3 Computational experiments

As mentioned in the introduction the computational experiments were designed to verify empirically the ability of the algorithms discussed above to solve badly scaled problems. A detailed description of the design of the performed experiments, the choice of suitable testproblems, the considered algorithms etc. will be given in the remaining part of this chapter. A discussion of the results will lead to a choice of update formulae to be applied in the context of the Recursive Quadratic Programming algorithms of chapter III.

II.3.1. Algorithms implemented

The flowchart given in figure 2.2 gives a general representation for the implementation of the considered algorithms. The different algorithms are defined by particular choices for the line search and the formulae for updating the inverse Hessian approximation.

We investigated implementations of the following 9 algorithms:

1. Davidon-Fletcher and Powell. Fletcher and Powell (1963);
2. Broyden-Fletcher-Goldfarb and Shanno. e.g., Broyden (1970);
3. Self Scaling Variable Metric (25 parameter choices). Oren and Luenberger (1974);

4-7 Four Optimally Conditioned Self Scaling Switches. Oren and Spedicato (1976);

8,9 Two devices for initial scaling of BFGS. Shanno and Phua (1978a).

For these algorithms we varied the accuracy of the line search. Also the effect of the test of Goldstein and Price (1967), to avoid line searches was investigated for a range of accuracies of this test.

The experiments were performed on an IBM 370/158 computer using the FORTRAN-G compiler under OS/VS2 (MVS-Multiprogramming Virtual Storage), in double precision. The implementation consisted of a main program SSVM which calls the subroutines CUBIC (line search) and UPDAT (updating inverse Hessian approximation).

Special remarks on the implementation:

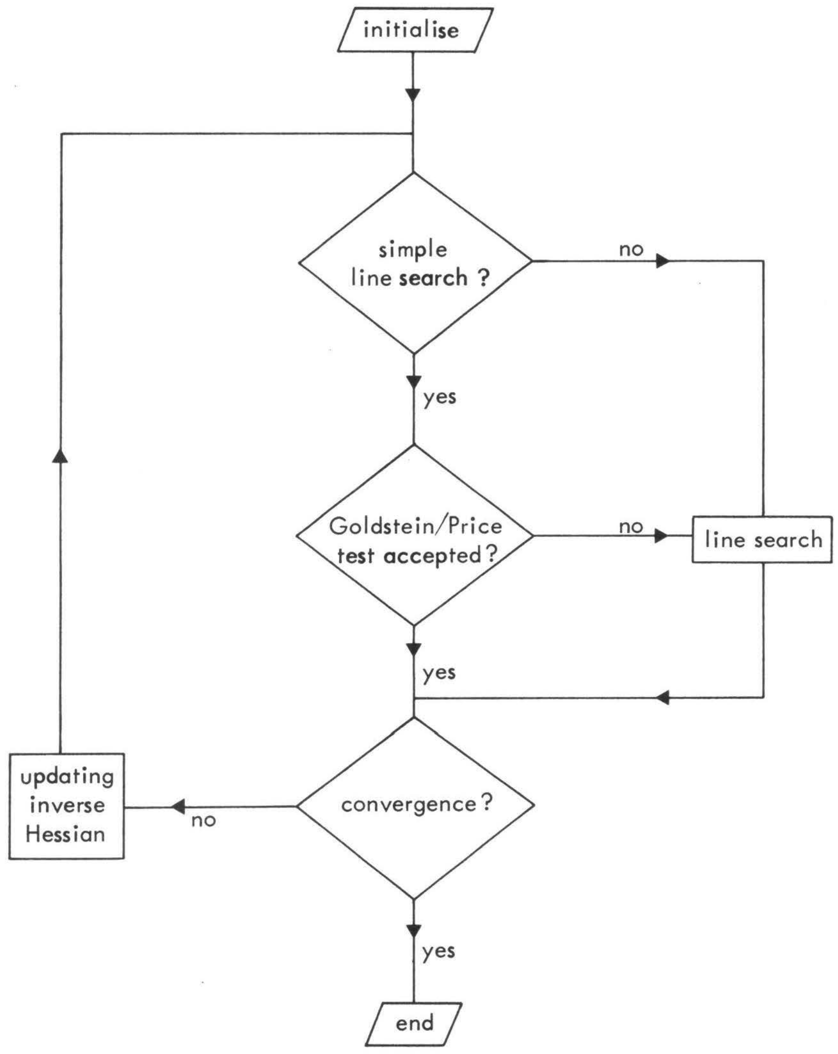

initialise
simple line search ?
no
yes
Goldstein/Price test accepted ?
no
line search
yes
convergence ?
no
updating inverse Hessian
yes
end

Figure 2.2

SSVM: 1. The Goldstein and Price condition to test whether the Newton steplength '1' is acceptable or not is applied in the main program SSVM. This means that the following condition is checked:

$$\sigma < \frac{F(x_k + p_k) - F(x_k)}{p_k^T \nabla F(x_k)} < 1 - \sigma \qquad \text{for some } 0 < \sigma < \tfrac{1}{2}.$$

2. As theoretically H_{k+1} need no longer be positive definite if $s^T y < 0$ we test this relation before updating. If $s^T y < 0$, no updating takes place: $H_{k+1} = H_k$. If this happens IFAIL times during the execution of one test problem the run is terminated with a message. This only occurred in the execution of test problem 6, with prechosen IFAIL = 10.

3. If the number of used function evaluations exceeds a predesigned number NFMAX, the execution is terminated with a message. We used the extremely high value NFMAX = 1000, to distinguish problems that are hard to solve from unsolvable ones.

CUBIC This line search is a bracketing process followed by cubic interpolations. Because of possible nonconvexity in the problems, $s^T y < 0$ can occur, the line search has a built in safety in the sense that it reverses a generated search direction which is not initially downhill.

UPDAT In this subroutine the updating of the inverse Hessian approximation takes place. The Oren-Spedicato switches require the calculating of

$$\pi = s^T H^{-1} s \text{ which is equivalent to } \pi' = \frac{s^T y \cdot g^T s}{g^T H y} \qquad \text{(Oren (1974a))}.$$

The latter expression is used in the computations as it is cheaper than the first. (In the case of an exact line search we can use $\pi'' = \alpha^2 g^T H g$ or $\pi''' = \alpha s^T y$).

II.3.2. The choice of test problems, termination criteria and performance indicators

The subjects to be treated in this section are motivated by the necessity of a proper design of the experiments, in order to be able to draw correct conclusions from the numbers that will be generated.

Test problems To meet our goal in the design of the numerical experiments, we composed a collection of 12 test problems, mentioned in appendix B. The test problems, whose gradients are given analytically, are taken from the literature. The new problems required are generated by varying parameters which influence the condition number of the test problem and/or the dimension. Though the convergence properties of the developed algorithms are proved for <u>convex</u> minimization problems, usually test batteries, including ours, also contain nonconvex problems. For the moment we only remark that recent research on <u>global</u> minimization algorithms to minimize nonconvex problems, Rinnooy Kan (1979), provides an entirely different approach. The set of 12 test functions consists of the following problems:

1, 2, 3, 4:	Increasingly badly scaled variants of Rosenbrock's function, Rosenbrock (1961), Colville (1968).
5, 6 :	10- and 30-dimensional generalizations of Rosenbrock's function.
7, 8, 9 :	2-, 10- and 30-dimensional Quartic functions, Oren (1973), to test the behaviour on homogeneous functions of different dimension.
10, 11, 12:	2-, 4- and 6-dimensional Hilbert problems, Oren (1973), to test the influence of increasingly extreme ill-conditioning on purely quadratic functions.

<u>Termination criteria</u>. As a wide variety of these criteria is known and has been applied we had to make a choice and decided to stop iterating as soon as <u>both</u> the following conditions were met:

$$||g_k|| \le 10^{-6}$$
$$||x_{k+1} - x_k|| \le 10^{-4}$$

We preferred this criterion consisting of two components as it guarantees a certain accuracy in determining both the optimal function value F* and the coordinates of the optimum x*.
The linear Taylor approximation of F(x) around x_k yields

$$||F(x_{k+1}) - F(x_k)|| \le ||g_k|| \; ||x_{k+1} - x_k||$$

so

$$||F(x_{k+1}) - F(x_k)|| \le 10^{-10} \text{ in our case.}$$

Table 2.3 illustrates the inaccuracy in x* which is still possible under our stopping rules.
A single component criterium as $||F(x_{k+1}) - F(x_k)|| \le 10^{-10}$, as applied in

Oren (1974b), Oren and Spedicato (1976) and Shanno and Phua (1978a) locates x* even less accurately.

Table 2.3
Last iteration point (x_1,x_2) for the two dimensional Quartic function for different algorithms with the applied termination criterium.

Algorithm	x_1	x_2	$F(x_1,x_2)$
SSVM $\phi = 1, \theta = .25$	$.2882\ 10^{-3}$	$.3880\ 10^{-4}$	$.740\ 10^{-16}$
$\phi = .50, \theta = .25$	$.2885\ 10^{-3}$	$.3841\ 10^{-4}$	$.742\ 10^{-16}$
$\phi = .75, \theta = .25$	$.2883\ 10^{-3}$	$.3862\ 10^{-4}$	$.741\ 10^{-16}$
Switch I	$.2883\ 10^{-3}$	$.3862\ 10^{-4}$	$.741\ 10^{-16}$
II	$.2883\ 10^{-3}$	$.3862\ 10^{-4}$	$.741\ 10^{-16}$
III	$.2883\ 10^{-3}$	$.3862\ 10^{-4}$	$.741\ 10^{-16}$
IV	$.9527\ 10^{-4}$	$-.1401\ 10^{-3}$	$.233\ 10^{-16}$
SH/PH I	$.9561\ 10^{-4}$	$.2622\ 10^{-3}$	$.215\ 10^{-15}$
II	$.1284\ 10^{-3}$	$-.2552\ 10^{-3}$	$.215\ 10^{-15}$
DFP	$.3747\ 10^{-3}$	$-.1238\ 10^{-3}$	$.292\ 10^{-15}$
BFGS	$.1102\ 10^{-2}$	$-.2708\ 10^{-3}$	$.185\ 10^{-13}$

The cubic linesearch terminates if the Euclidian distance of succeedingly generated points along the search direction is smaller than or equal to a preset parameter called EPSCU.

Performance indicators. Candidates for performance indicators are: number of function evaluations, number of iterations and required CPU-secs to solve a testproblem (an iteration consists of the generation and exploration of a search direction). These three indicators are mentioned in the tables in Van der Hoek and Dijkshoorn (1979). The number of required function evaluations was used as the main indicator. That is why only the results for this indicator will be given here (these results correspond directly to the number of iterations as the number of function evaluations per iteration does not vary much. The main disadvantage of counting function evaluations to solve the whole set of testproblems is that different objective functions are equally weighed though they may differ substantially in complexity: in Van der Hoek and Dijkshoorn (1979) we mentioned that one evaluation of the 30-dimensional Rosenbrock-function

is approximately as expensive as five evaluations of the 2-dimensional Quartic function. This disturbing infuence is compensated for by separate consideration of classes of test functions, such as the higher dimensional ones and separate conclusions for those classes.
The required CPU-time gives additional information on the overhead of computations such as matrix manipulations which the program performs. However, the CPU-time cannot be measured very accurately because of the inaccuracy of the internal clock of the machine and, more importantly, because of the multiprogramming facility.
We found that times varied up to 10% for jobs run in daytime and requiring less than 10 measured secs CPU-time. Because of this lack of accuracy, we do not present these tables here. Table 2.8 should be regarded as an illustration of the accuracy reached in determining F*.

II 3.3. Design of the experiments and results

The experiments were designed in the following way:

Experiment I Find the three best (ϕ,θ)-combinations of the Oren-Luenberger SSVM-algorithms, without application of the Goldstein and Price test. The accuracy of the line search EPSCU varies from 10^{-1} to 10^{-6}. The resulting algorithms are called A, B and C.

Experiment II The algorithms A, B and C which arose from experiment I and implementations of the four Oren-Spedicato switches are compared. The parameter σ of the Goldstein and Price test varies from 0.01 to 0.49 and EPSCU has the same range as in experiment I.

Experiment III DFP and BFGS are implemented together with the two devices for initial scaling of BFGS of Shanno and Phua (1978a).

Under the termination criteria stated above the generalized Rosenbrock function with $c = 10^6$ appeared to be too hard for all algorithms. That is why it is not incorporated in the following tables.
The 25 algorithms resulting from 5 particular choices for each of the parameters ϕ and θ were generated by the loops:

```
          DO 10   I = 1,5
          PHI = .25 * (I-1)
          DO 10   J = 1,5
          TETTA = .25 * (J-1)
       10 CONTINUE
```

The most relevant results are summarized in the tabless 2.4 - 2.7 using the following notation:

# F	: number of required function evaluations
F*	: function value reached
F	: failure
Σ	: column sum

In the calculation of Σ, a failure will be counted as 1000 function evaluations. The F of failure is repeated below the corresponding value of Σ.

Table 2.4: # F for 25 (ϕ,θ)-combinations. Accuracy line search 10^{-1}. No Goldstein/Price test.

algorithm	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
test function																									
Ros(c=1)	32	29	29	28	28	29	28	28	26	25	29	28	25	25	26	28	26	25	23	22	28	25	26	22	23
(c=10^2)	160	102	94	91	98	102	100	101	98	111	93	101	108	119	119	91	98	119	126	103	98	111	119	103	127
(c=10^4)	863	357	329	328	315	350	330	313	322	306	339	300	307	374	356	311	331	362	381	380	321	307	348	345	540
(n=10)	305	211	207	210	241	193	174	175	183	209	171	159	162	179	195	173	157	160	159	168	166	160	170	161	168
(n=30)	F	726	F	F	F	588	471	527	584	655	552	391	429	486	523	525	372	391	434	466	511	362	382	420	463
Quartic(n=2)	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47
(n=10)	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	68	70	74	72	72	69	70	71	74	69
(n=30)	91	89	91	91	87	91	89	89	90	90	89	91	91	91	91	89	90	91	91	91	90	89	91	91	91
Hilbert(n=2)	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
(n=4)	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
(n=6)	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22
Σ	2626	1689	1925	1923	1944	1528	1367	1408	1478	1571	1451	$\boxed{1245}$	1297	1449	1485	1388	$\boxed{1247}$	1325	1389	1405	1386	$\boxed{1227}$	1310	1319	1584
	F		F	F	F							B					C					A			

Table 2.5: # F for 25 (ϕ,θ)-combinations. Accuracy line search 10^{-3}. No Goldstein/Price test

algorithm	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
test function																									
Ros(c=1)	34	31	31	30	30	31	30	30	28	27	31	30	27	27	28	30	28	27	25	24	30	27	28	24	25
$(c=10^2)$	169	134	121	130	132	134	136	136	129	123	120	136	122	127	127	130	129	127	132	140	132	122	128	140	149
$(c=10^4)$	882	470	436	443	429	471	443	450	447	544	436	450	530	577	433	442	447	577	441	436	429	543	433	436	434
(n=10)	350	262	290	293	288	235	215	230	258	266	233	205	205	235	244	234	218	207	213	233	237	204	206	220	219
(n=30)	F	F	F	F	F	715	608	674	F	F	710	488	567	650	729	725	486	550	582	660	772	489	501	544	605
Quartic(n=2)	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59	59
(n=10)	94	95	95	95	87	90	94	94	95	91	90	94	94	94	94	91	91	94	94	96	90	89	101	97	97
(n=30)	134	131	130	130	131	131	132	133	131	131	131	132	132	132	130	129	133	132	132	131	129	131	133	133	132
Hilbert(n=2)	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
(n=4)	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
(n=6)	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22
Σ	2778	2238	2218	2236	2212	1922	1773	1862	2203	2297	1866	1650	1792	1957	1900	1896	1646	1829	1734	1835	1934	1720	1645	1709	1776
	F	F	F	F	F				F	F															

Table 2.6: # F for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-6}. No Goldstein/Price test.

algorithm	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
test function																									
Ros(c=1)	36	33	33	32	32	33	32	32	30	29	33	32	29	29	30	32	30	29	27	26	32	29	30	26	27
$(c=10^2)$	177	145	125	140	142	145	145	145	151	128	126	145	129	133	135	140	151	133	138	145	142	129	135	146	156
$(c=10^4)$	F	534	520	498	508	534	511	529	528	534	520	528	539	500	492	498	528	501	492	487	508	533	492	486	484
(n=10)	384	291	325	342	305	266	262	267	304	284	274	244	242	268	295	270	252	243	244	273	281	240	247	266	250
(n=30)	F	F	F	F	F	F	679	803	F	F	F	595	684	753	F	F	760	622	683	781	F	599	597	647	689
Quartic(n=2)	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62	62
(n=10)	101	106	106	106	102	101	101	106	106	106	101	100	106	106	106	194	105	105	106	105	103	102	105	105	104
(n=30)	146	143	145	145	148	144	147	147	147	147	145	146	149	149	146	145	147	147	149	149	149	147	147	148	148
Hilbert(n=2)	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
(n=4)	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
(n=6)	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22	22
Σ	2962	2370	2372	2381	2355	2341	2015	2147	2384	2346	2317	1908	1996	2056	2322	2307	2091	1898	1957	2084	2333	1897	1871	1942	1976
	F	F	F	F	F	F			F	F	F				F	F					F				

Table 2.7: # F for algorithms A, B, C, the 4 Oren-Spedicato switches, DFP, BFGS and the 2 Shanno-Phua variants. Accuracy line search 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm	A	B	C	SWI	SWII	SWIII	SWIV	DFP	BFGS	SH/PH I	SH/PH II
test function											
Ros(c=1)	28	22	29	21	21	21	17	22	17	15	15
($c=10^2$)	110	106	114	111	111	98	493	133	67	72	72
($c=10^4$)	346	371	347	358	350	364	F	300	227	225	228
(n=10)	128	133	124	130	117	166	F	262	112	116	113
(n=30)	237	264	256	225	268	312	F	F	381	244	231
Quartic(n=2)	38	38	38	38	38	38	52	60	41	58	58
(n=10)	47	47	47	48	47	48	F	479	128	172	172
(n=30)	52	53	53	53	53	53	F	716	253	415	414
Hilbert(n=2)	13	13	13	13	13	13	14	10	10	12	12
(n=4)	28	28	28	28	28	28	18	18	30	32	32
(n=6)	27	27	27	27	27	27	16	16	26	35	35
Σ	1054	1102	1076	1052	1073	1168		3016	1292	1396	1382

Table 2.8: F*, obtained by algorithms A, B, C and the 4 Oren-Spedicato switches.
Accuracy line search 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$

algorithm	A	B	C	SWI	SWII	SWIII	SWIV
test function							
Ros(c=1)	$.179\ 10^{-17}$	$.528\ 10^{-17}$	$.376\ 10^{-18}$	$.513\ 10^{-19}$	$.513\ 10^{-19}$	$.513\ 10^{-19}$	$.687\ 10^{-20}$
$(c=10^2)$	$.453\ 10^{-16}$	$.270\ 10^{-22}$	$.473\ 10^{-17}$	$.120\ 10^{-22}$	$.120\ 10^{-22}$	$.439\ 10^{-26}$	$.215\ 10^{-21}$
$(c=10^4)$	$.132\ 10^{-26}$	$.586\ 10^{-21}$	$.596\ 10^{-20}$	$.210\ 10^{-22}$	$.123\ 10^{-18}$	$.724\ 10^{-21}$	F
(n=10)	$.116\ 10^{-16}$	$.661\ 10^{-18}$	$.720\ 10^{-18}$	$.114\ 10^{-16}$	$.152\ 10^{-18}$	$.191\ 10^{-17}$	F
(n=30)	$.142\ 10^{-15}$	$.601\ 10^{-17}$	$.432\ 10^{-16}$	$.221\ 10^{-16}$	$.645\ 10^{-17}$	$.156\ 10^{-17}$	F
Quartic(n=2)	$.741\ 10^{-14}$	$.742\ 10^{-14}$	$.741\ 10^{-14}$	$.741\ 10^{-14}$	$.741\ 10^{-14}$	$.741\ 10^{-14}$	$.233\ 10^{-14}$
(n=10)	$.792\ 10^{-14}$	$.857\ 10^{-14}$	$.823\ 10^{-14}$	$.295\ 10^{-14}$	$.849\ 10^{-14}$	$.295\ 10^{-14}$	F
(n=30)	$.316\ 10^{-13}$	$.153\ 10^{-13}$	$.144\ 10^{-13}$	$.218\ 10^{-13}$	$.148\ 10^{-13}$	$.218\ 10^{-13}$	F
Hilbert(n=2)	$.933\ 10^{-32}$	$.975\ 10^{-32}$	$.105\ 10^{-31}$	$.887\ 10^{-32}$	$.916\ 10^{-32}$	$.887\ 10^{-32}$	$.739\ 10^{-31}$
(n=4)	$.611\ 10^{-16}$	$.616\ 10^{-16}$	$.611\ 10^{-16}$	$.619\ 10^{-16}$	$.616\ 10^{-16}$	$.619\ 10^{-16}$	$.206\ 10^{-13}$
(n=6)	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.151\ 10^{-10}$

II.3.4. Discussion of the results

Experiment I

The numbers of function evaluations required by all 25 algorithms for EPSCU = 10^{-1}, 10^{-3} and 10^{-6} are given in tables 2.4, 2.5 and 2.6. We selected the seven 'best' algorithms for three cases: EPSCU = 10^{-1}, EPSCU = 10^{-1} *and* EPSCU = 10^{-3} and, finally for *all three* accuracies: EPSCU = 10^{-1}, 10^{-3} and 10^{-6}. The results are given in table 2.9. We mention that obviously nontrivial values are to be preferred and that all three columns of table 2.9 contain the *same* seven parameter combinations. From tables 2.4, 2.5, 2.6 and figure 2.3 it can be deduced that increasing the accuracy makes all algorithms more expensive from which we conclude that EPSCU = 10^{-1} should be used. This confirms our remarks in Ch. II.2.1. on inexact line searches. These arguments led to the following choice of three 'best' parameter combinations evolving from experiment I on our set of testproblems:

ϕ = 1. ,	θ = .25	: algorithm 22
ϕ = .50,	θ = .25	: algorithm 12
ϕ = .75,	θ = .25	: algorithm 17.

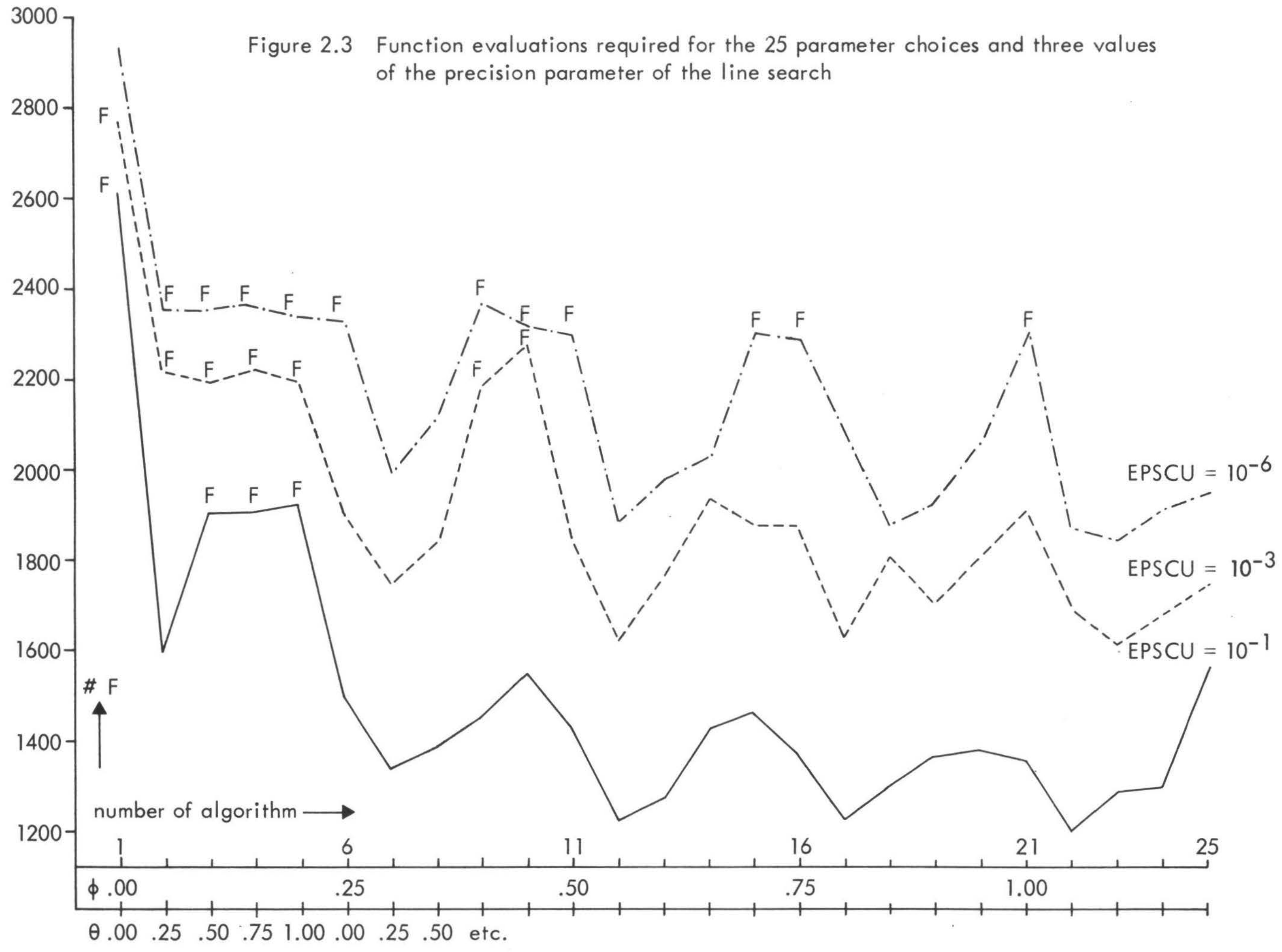
From now on we shall call these algorithms A, B and C respectively.

Table 2.9: # F for: I EPSCU = 10^{-1}
II EPSCU = 10^{-1} and 10^{-3} (cumulative)
III EPSCU = 10^{-1}, 10^{-3} and 10^{-6} (cumulative)

I		II		III	
algorithm	#F	algorithm	#F	algorithm	#F
22	- 1227	17	- 2893	12	- 4803
12	- 1245	12	- 2895	23	- 4826
17	- 1247	22	- 2947	22	- 4844
13	- 1297	23	- 2955	24	- 4970
23	- 1310	24	- 3028	17	- 4980
24	- 1319	13	- 3089	18	- 5052
18	- 1325	7	- 3140	13	- 5058

The results of experiment I are illustrated in figure 2.3. In this figure

Figure 2.3 Function evaluations required for the 25 parameter choices and three values of the precision parameter of the line search



the points found experimentally are connected to simplify 'reading' of the picture. There is no intention to suggest any analytically proved continuity of number of function evaluations in terms of parameter combinations!
From this figure we see that $\phi = 0$ is unsatisfactory, while for any given nontrivial value of ϕ the results get worse for values of θ higher than $\theta = .25$. Obviously the parameter θ, which is the weighing factor of the correction term vv^T in (2.5) is of more importance than the parameter ϕ which defines the scaling factor γ of the objective function! Testing of the sensitivity of the algorithms with respect to the accuracy of the line search is further continued in experiment II.

Experiment II

We considered implementations of the algorithms A, B and C and the four Oren-Spedicato switches.
First the sensitivity with respect to the parameter σ of the Goldstein and Price test is investigated. We tested $\sigma = 0.01$, 0.10, 0.25 and 0.49. For $\sigma = 0.01$ the Newton steplength '1' will often be accepted and no line search is performed. Increasing σ causes more line searches, for $\sigma = 0.49$ almost all iterations use the cubic line search with EPSCU = 10^{-1}. In our experiment $\sigma = 0.10$ generally yielded the best results. The final results are given in tables 2.7 and 2.8. Clearly switch IV is dominated by the other algorithms.

Experiment III

Implementations of DFP, BFGS and the two Shanno-Phua algorithms were run for $\sigma = 0.10$ and EPSCU = 10^{-1}. Obviously DFP prefers (requires) an exact line search, which confirms known results. Table 2.7 presents the relevant figures.

Our *general* conclusion from table 2.7 is that switches I, II and III are competitive with the algorithms A, B and C, which apply optimally chosen parameters. BFGS is slightly worse than the Shanno/Phua variants. The results of the last two variants are clearly influenced by their problems in solving the 3 homogeneous testfunctions. Further it should be realised that the algorithms A, B and C evolve from an optimization of algorithms with respect to the parameters ϕ and θ. Thus the performance of the *general* scaling devices of the switches I, II and III and Shanno and Phua's variant is really excellent! Finally the results with the algorithms A,B,C suggest the replacing of $\theta_k = 1$ for all k in BFGS by $\theta_k = .25$ for all k.

The influence of the dimension of the test problem and remarks on homogeneous test problems.

Now we are only interested in those figures from table 2.7 which concern the 10- and 30-dimensional Rosenbrock and Quartic test functions. Clearly initial scaling of BFGS should not be recommended for homogeneous test problems such as the Quartics. Tnis confirms Shanno and Phua (1978b). Furthermore these figures suggest to apply Shanno/Phua I or II or one of the switches I or II for higher dimensional problems. If it is known beforehand that F(x) is homogeneous, which rarely happens in real-life problems, switch II is to be preferred.

Influence of the conditioning of the test problem.

Two effects were investigated:

a) The ability of the algorithms to solve problems with a shifted spectrum of eigenvalues of R_1. We varied the parameter c of a family of Rosenbrock-problems $c = 1, 10^2, 10^4, 10^6$. Increasing c only slightly influences the conditioning at the starting point (-1.2,1) but creates increasingly extremely ill-conditioned optimal points (1,1). All algorithms failed to solve the problem with $c = 10^6$.

b) Increasingly ill-conditioned pure quadratic problems are the Hilbert problems for increasing dimension. We investigated n = 2, 4, 6.

The results on these test functions are summarized in table 2.10.

Conclusion

From the experiments with the Rosenbrock-family we conclude that the BFGS algorithms (BFGS with or without initial scaling) behave better for ill-conditioned optimal points.

The differences on purely quadratic functions are negligible.

Table 2.10: # F for ill-conditioned test problems

algorithm / test function	A	B	C	SW I	SW II	SW III	SW IV	SH/PH I	SH/PH II	DFP	BFS
Ros(c=1)	28	22	29	21	21	21	17	15	15	22	17
($c=10^2$)	110	106	114	111	111	98	493	72	72	133	67
($c=10^4$)	346	371	347	358	350	364	F	225	228	300	227
Hilbert(n=2)	13	13	13	13	13	13	13	12	12	10	10
(n=4)	28	28	28	28	28	28	28	32	32	18	30
(n=6)	27	27	27	27	27	27	27	35	35	16	26
Σ	552	567	558	558	550	551	1578 F	391	394	499	377

Final conclusion

Recently developed self scaling algorithms for unconstrained minimization were described and compared in numerical experiments. All algorithms, except DFP and the fourth Oren-Spedicato switch, showed a good performance with an inexact line search. Generally an iteration requires about 2 objective function evaluations. For reasons of robustness (initially scaled) BFGS algorithms and the second Oren-Spedicato switch seem to be preferable in most practical situations. This establishes once more the superiority of the classical BFGS algorithm. This conclusion is valid even more generally: numerical comparisons by Grandinetti (1978) and Shanno and Phua (1978b) show that this variable metric algorithm is competitive even with sophisticated quasi-Newton algorithms such as those based on factorizations or projections of search directions. Concerning the choice for an update procedure in the context of the algorithms of chapter III this means that preferably one of the Shanno and Phua algorithms or switch II should be applied.

III. RECURSIVE QUADRATIC PROGRAMMING WITH SELF SCALING UPDATES OF THE SECOND-ORDER INFORMATION

III.1. Introduction

The first class of reduction methods which will be considered was developed from a proposal in Murray (1969). Biggs developed it further in Biggs (1972, 1974, 1978). Similar approaches were followed in Han (1977, 1979) and Powell (1977a, 1978).
The aim of these so-called *recursive quadratic programming* algorithms is to avoid the increasingly ill conditioned reduced problems of penalty function methods, reported in Murray (1967) and Lootsma (1969). Furthermore too high an accuracy in the unconstrained minimization of the penalty functions in an early stage of the iteration process is avoided as well. It will appear to be sufficient to generate iteration points that only approximate the *minimizing trajectory* of the applied exterior penalty functions. The sufficiency of this approximation is based on the boundary properties of penalty functions as they were developed in Lootsma (1970). The convergence of the generated iteration points to the constrained optimum x^* was proved in Biggs (1978). Our presentation will be along the lines of Biggs's approach: in every reduced problem a quadratic approximation of the objective function is minimized subject to a local linearization of the first order Kuhn-Tucker conditions of the currently defined exterior penalty function. The solutions of the reduced problems thus defined can be proved to converge to a Kuhn-Tucker point of the original constrained nonlinear programming problem. As these reduced problems are equality constrained quadratic programming problems, their solution can be written down algebraically. The theorems on the convergence and the rate of convergence of the algorithms thus defined will be stated in section III.2.
A further point of particular interest, to be treated in ch. III.3, is the incorporation of self scaling variable metric update formulae, discussed in ch. II, in the framework of recursive quadratic programming. A motivation to do this is that the approximated penalty functions will still be more and more ill conditioned. We shall compare computationally the effect of the use of these self scaling update formulae for the second order information, with the algorithms of chapter IV.
The advantage of these scaling strategies is even greater, as we found in

ch. II in that they allow for inexact line searches. Hence the approximative character of the recursive quadratic programming approach is combined with an efficient strategy to solve ill conditioned reduced problems. In addition ch. III.3 will contain a discussion on the influence of a not yet correct active set $I(x_k)$ on the determination of the stepsize in the line search. The selection of the constraints which will constitute the active set and subsequently define the quadratic loss term of the exterior penalty function will be treated in ch. III.3 as well. The results of the computational experiments with this algorithm will be presented and discussed in ch. VI of this monograph.

III.2. Convergence properties of recursive quadratic programming

We shall consider the general nonlinear programming problem

$$(3.1) \qquad \begin{cases} \text{minimize} \quad F(x) \\ \text{subject to} \\ c_i(x) \geq 0 \qquad i = 1, \ldots, p \\ c_i(x) = 0 \qquad i = p+1, \ldots, m \end{cases}$$

The Lagrangian function associated with problem (3.1) is

$$(3.2) \qquad L(x,u,v) = F(x) - \sum_{i=1}^{p} u_i c_i(x) - \sum_{i=p+1}^{m} v_i c_i(x)$$

where u_i, $i = 1, \ldots, p$ and v_i, $i = p+1, \ldots, m$ denote the Lagrangian multipliers of the inequality and the equality constraints respectively. All problem functions are assumed to be twice continuously differentiable and a regular solution x^* of (3.1) is assumed to exist.

In the application of exterior point penalty function methods, see Fiacco and Mc Cormick (1968), the penalty functions

$$(3.3) \qquad \begin{aligned} P(x,r_k) &= F(x) + \frac{1}{r_k} \sum_{i \in I_k} c_i^2(x) \\ &= F(x) + \frac{1}{r_k} w_k^T(x) w_k(x) \end{aligned}$$

are minimized for a sequence $\{r_k\} \downarrow 0$.

Here the vector $w = w(x)$ is the vector of currently active constraints,

hence $w_k^T = w^T(x_k) = (c_{i_1}(x), \ldots, c_{i_k}(x))$ where $I_k = \{i_1, \ldots, i_k\}$.
In this formulation of the penalty function an active set I_k is used. As a consequence, the reduced problems are *equality constrained* problems. Usually I_k will consist of the currently active or violated constraints, augmented by those constraints whose Lagrange multiplier makes them liable to become active or violated in the next iteration. In Pietrzkowski (1962) it is shown that if $P(x,r_k)$ is strictly convex for each $r_k > 0$ and has a minimum x_k^*, the sequence $\{x_k^*\}$ converges to x^* if and only if $\{r_k\}$ is a monotone null sequence with $r_k > 0$ for all k. Then the points $\{x_k^*\}$ are located on the so-called *minimizing trajectory* whose properties were extensively treated in Lootsma (1970). It will appear that the solutions of the reduced problems of this chapter approximate the minimizing trajectory for penalty functions which use for $I(x_k)$: all currently violated constraints. The algorithm proposed in Biggs (1972) suggested to replace the direct unconstrained optimization of (3.3) by the solution of an equality constrained quadratic programming problem. This reduced problem comes from the requirement that the locally defined quadratic approximation to the objective function F(x) should be minimized, subject to the linear constraint that the truncated Taylor series expansion of $\nabla P(x,r_k)$ in a neighbourhood of the current iteration point x_k vanishes.
From (3.3) we see, denoting the Jacobian matrix of w(x) at x_k by A_k, that

(3.4) $$\nabla P(x_k,r_k) = \nabla F(x_k) + \frac{2}{r_k} A_k^T w(x_k)$$

$$= g_k + \frac{2}{r_k} A_k^T w_k$$

has as truncated Taylor series expansion at $x_k + p$:

(3.5) $$\nabla P(x_k + p,r_k) = g_k + B_k p + \frac{2}{r_k} A_k^T w_k + \frac{2}{r_k} A_k^T A_k p$$

Note, however, that this approximation is made under the assumptions that $I(x_k) = I(x_k + p)$, and that the applied linear approximation is still acceptable at x_k+p. In Van der Hoek and Wymenga (1980) it will be proved that, if $I(x_k) \neq I(x_k+p)$ and if the stepsize is limited above by '1', then the theoretically required stepsize will usually meet this limitation as well. Another benefit of this stepsize limitation is that the linear approximation to $\nabla P(x_k + p,r_k)$ will be better. As in a neighbourhood of x^*

the curvature of the penalty term will dominate the curvature of $F(x)$, we can neglect the term $B_k p$ in (3.5) in the treatment below.
If x_k+p is the minimum of $P(x, r_k)$, then equation (3.5) yields

$$g_k + \frac{2}{r_k} A_k^T w_k + \frac{2}{r_k} A_k^T A_k p = 0 \tag{3.6}$$

Given a positive definite symmetric matrix B_k (for instance, but not necessarily, the current approximation of $\nabla^2 F(x^*)$), we can premultiply (3.6) by $A_k B_k^{-1}$ which yields:

$$A_k p = - \frac{r_k}{2} (A_k B_k^{-1} A_k^T)^{-1} A_k B_k^{-1} g_k - w_k \tag{3.7}$$

Now the step p can be determined using (3.7). For instance by the minimization of a quadratic approximation to $F(x)$ in the null space of A_k (which does not contain information on p). Thus we obtain as reduced problem

$$\begin{cases} \text{minimize} & \tfrac{1}{2} p^T B_k p + g_k^T p \\ \text{subject to} & \\ & A_k p = - \frac{r_k}{2} \hat{\lambda}_k - w_k \end{cases} \tag{3.8}$$

where the vector

$$\hat{\lambda}_k = (A_k B_k^{-1} A_k^T)^{-1} A_k B_k^{-1} g_k \tag{3.9}$$

can be considered to be an estimate of the vector of Lagrange multipliers corresponding to the constraints of the active set (Fiacco and Mc Cormick, 1968). The solution of this equality constrained quadratic programming problem can be written down immediately, following Fletcher (1971), as

$$p_k = B_k^{-1} (A_k^T (A_k B_k^{-1} A_k^T)^{-1} (A_k B_k^{-1} g_k - \frac{r_k}{2} \hat{\lambda}_k - w_k) - g_k) \tag{3.10}$$

An alternative reduced problem arises if B_k, the current estimate of the Hessian matrix of $F(x)$ is incorporated in (3.5). Skipping the precise formulation of the resulting reduced problem we proceed immediately with the most successful situation in which the curvature of the constraints is represented as well.
Let W_k be some approximation to the matrix

$$(3.11) \qquad \nabla^2 F(x_k) + \frac{2}{r_k} \sum_{i \in I_k} c_i(x_k) \nabla^2 c_i(x_k)$$

Then we obtain, instead of (3.6):

$$(3.12) \qquad \nabla P(x_k + p, r_k) = g_k + W_k p + \frac{2}{r_k} A_k^T w_k + \frac{2}{r_k} A_k^T A_k p$$

which ultimately leads to the reduced problem

$$(3.13) \qquad \begin{cases} \text{minimize} & \tfrac{1}{2} p^T W_k p + g_k^T p \\ \text{subject to} & \\ & A_k p = -\frac{r_k}{2} \hat{\tau}_k - w_k \end{cases}$$

where

$$(3.14) \qquad \hat{\tau}_k = \left(\frac{r_k}{2} I + A_k W_k^{-1} A_k^T\right)^{-1} (A_k W_k^{-1} g_k - w_k)$$

The third alternative reduced problem arises from the application of the approximating matrix W_k in the context of reduced problem (3.8), which yields

$$(3.15) \qquad \begin{cases} \text{minimize} & \tfrac{1}{2} p^T W_k p + g_k^T p \\ \text{subject to} & \\ & A_k p = -\frac{r_k}{2} \hat{\pi}_k - w_k \end{cases}$$

where

$$(3.16) \qquad \hat{\pi}_k = (A_k W_k^{-1} A_k^T)^{-1} A_k W_k^{-1} g_k$$

A closer examination of (3.11) yields that the currently defined matrix W_k can be considered as an approximation to the Hessian matrix of the Lagrangian function (3.2). This follows from

$$\lim_{k \to \infty} -\frac{2}{r_k} w(x_k) = \lambda^*$$

if $\lim_{k \to \infty} x_k = x^*$ along the minimizing trajectory (see Fiacco and Mc Cormick, 1968). As all the above vectors $\hat{\lambda}_k$, $\hat{\tau}_k$ and $\hat{\pi}_k$ can be regarded as approximate Lagrange multipliers, matrices, such as

$$(3.17) \qquad \nabla^2 F(x_k) - \hat{\tau}_k \sum_{i \in I_k} \nabla^2 c_i(x_k)$$

can be used as an approximation of the Hessian matrix of the Lagrangian function. In ch. III.3.4 updating strategies for these matrices will be discussed.

Now that the reduced problems (3.8), (3.13) and (3.15) are known, we proceed with a concise presentation of the convergence theorems of the corresponding algorithms. Basically the approach followed amounts to proving that the search directions p defined by these reduced problems can be used to locate the unconstrained minima of an augmented Lagrangian function of the class introduced in Fletcher (1969). In turn the unconstrained minima of this function can be proved to coincide with the constrained minima of problem (3.1). The rate of convergence will appear to be superlinear. The analysis will concern *'well behaved'* functions, which means functions that are bounded below and that have bounded derivatives.
A basic theorem, due to L.C.W. Dixon, on the unconstrained minimization of such a function mainly states conditions on the applied search directions p and the stepsizes α along those search directions. The theorem as stated below is a slightly adjusted modification of the original theorem.

Theorem 3.1. (Dixon, 1974)

Suppose that $\Phi(x)$ is a well behaved function. An iterative minimization algorithm is applied to $\Phi(x)$ which calculates a direction of search p from the point x and obtains a new point $\bar{x} = x + \alpha p$. The scalar α is chosen so that $\Phi(\bar{x}) \leq \Phi(x)$. The algorithm will find a point x^* such that $||\nabla\Phi(x^*)|| \leq \varepsilon_0$ for some specified $\varepsilon_0 > 0$, if for a regular subsequence of iterations the following conditions are met for some $\varepsilon_1, \varepsilon_2, \varepsilon_3$ which can be specified in terms of ε_0:

Condition I $\quad p^T\nabla\Phi(x) \leq -\varepsilon_1||p|| \; ||\nabla\Phi(x)||$ for some $\varepsilon_1 > 0$

Condition II $\quad |\Phi(\bar{x}) - \Phi(x) - \alpha p^T\nabla\Phi(x)| \geq \varepsilon_2|\alpha p^T\nabla\Phi(x)|$ for some $\varepsilon_2 > 0$

Condition III $\quad \Phi(\bar{x}) - \Phi(x) \leq \varepsilon_3\alpha p^T\nabla\Phi(x)$ for some $\varepsilon_3 > 0$

□

This theorem assures that, in the absence of rounding error, a stationary point x^* will be located within precision ε_0. Condition I means that the search directions should be significantly 'downhill', condition II prohibits a stepsize α which is too small while condition III assures that

the reduction in function value has a non-zero upper bound.
The application of this theorem is simplified remarkably by the following theorem:

THEOREM 3.2 (Biggs, 1978)

If a search direction p satisfies condition I of theorem 3.1 for a well behaved function $\Phi(x)$, then a suitable value of α can be found to satisfy conditions II and III for any ε_2, ε_3 such that $(1-\varepsilon_3) > \varepsilon_2$.

□

This theorem means that once p satisfies condition I, the conditions II and III can be met such that $(1-\varepsilon_3) > \varepsilon_2$ holds for the given parameters ε_2, ε_3. That is why the following theorems concern search directions p that satisfy condition I. The directions p emanating from the reduced problems (3.8), (3.13) and (3.15) will be considered in that light. Once it has been proved that these directions satisfy condition I, it is justified to use them for the unconstrained minimization of $P(x,r_k)$.
In the next theorems no explicit use will be made of the interpretation of the iteration matrices B_k or W_k that occurred in (3.8), (3.13) and (3.15). It is only required that they both *satisfy the following relation:*

for all unit vectors $x \in \mathbb{R}^n$ there exist constants m, $M \in \mathbb{R}$ such that $0 < m \leq x^T B_k x \leq M$. Furthermore the Jacobian matrix A should be such that $0 \leq x^T A^T Ax \leq T$ for some $T \in \mathbb{R}$.

THEOREM 3.3 (Biggs, 1978)

Let $\nabla P(x,r) = g + \frac{2}{r} A^T w$
where $r > 0$ and the rows of A are linearly independent. Let p be the solution of any of the reduced problems (3.8), (3.13) or (3.15). Then there is a value $\bar{r}$ such that for all $r < \bar{r}$, $p^T \nabla P(x,r)$ satisfies condition I of theorem 3.1.

□

Remark

For reduced problem (3.13) even a stronger result can be proved: the rows of A need not be linearly independent, while no upper limit on r is necessary.

Instead of using the vectors p thus defined to minimize the penalty functions $P(x,r_k)$ directly, they can be proved to be suitable for the minimization of an augmented Lagrangian function related to problem (3.1). The augmented Lagrangian used here is an element of the class introduced in Fletcher (1969), and it can be proved that under suitable conditions the unconstrained minimization of the function

$$(3.18) \qquad P(x,q) = F(x) - w(x)^T\pi(x) + qw(x)^Tw(x)$$

yields a solution of the original problem (3.1). Here $\pi(x) = (AA^T)^{-1}A\nabla F(x)$ and q is a scalar which should be greater than some lower bound.
Hence the *sequence* of penalty function minimizations is replaced by *a single* minimization of (3.18). The applicability of the search directions evolving from (3.8), (3.13) and (3.15) is stated in the following theorem.

THEOREM 3.4 (Biggs, 1978)

Consider the objective function and constraints of problem (3.1) and let $P(x,q)$ be defined by (3.18). Suppose that at a point x the rows of A are linearly independent and that the approximating matrix B is positive definite. Let p be given as the solution of one of the reduced problems (3.8), (3.13), (3.15). Then there exists a value of q such that $p^T\nabla P(x,q)$ satisfies condition I of theorem 3.1.

□

Remark

Again if p comes from the reduced problem (3.13) the result can be proved without using the linear independence of the rows of A.
Now that the convergence theorems of the recursive quadratic programming algorithms have been stated, the next point of interest is their *rate of convergence* to x*.
The point x* is called a *point of attraction* of an applied algorithmic scheme if there exists an open neighbourhood $O(x^*)$ of x* such that for any starting point $x_0 \in O(x^*)$ the sequence $\{x_k\}$, generated by the algorithm, converges to x*. If

$$\lim_{k\to\infty} \frac{||x^* - x_{k+1}||}{||x^* - x_k||} = 0$$

the *rate of convergence* to x* is said to be *superlinear*.

The next theorem states the superlinear rate of convergence of the recursive quadratic programming algorithms, if the search is initiated in a region where the active set of constraints $I(x_k)$ equals $I(x^*)$.

THEOREM 3.5 (Biggs, 1978)

If the approximating matrix B applied at x^* only differs from the Hessian of the Lagrangian function $\nabla^2 L(x^*, u^*, v^*)$ in the subspace spanned by the normals of the active constraints at x^*, there exists a value r^* such that the solution x^* of (3.1) is a point of attraction of the recursive quadratic programming algorithm and the convergence to x^* is superlinear.

□

Remark

The value of r^* depends on the particular choice of reduced problem. The proof of the theorem consists in showing that x^* is a *fixed point* of the algorithm used, i.e., a point that is its own image under the application of the algorithm, and that the conditions for the application of theorem 10.1.6 of Ortega and Rheinboldt (1970) are satisfied.
As a consequence of theorem 3.5 one can use positive definite matrices B_k to approximate $\nabla^2 L(x^*, u^*, v^*)$, even if the latter matrix is indefinite. Note, however, that B_k and $\nabla^2 L(x^*, u^*, v^*)$ have to agree in the intersection of linearized active constraints at x^*; in this subspace $L(x,u,v)$ can be guaranteed to have positive curvature at the solution.
Now that the convergence properties of the recursive quadratic programming algorithms have been dealt with, we shall proceed with a discussion of several algorithmic aspects of these reduction methods.

III.3. Algorithmic aspects of recursive quadratic programming

This chapter treats some algorithmic aspects of the algorithms developed above, such as the applied active set strategy, the unidimensional search and the updating of the second order information. All these three different aspects strongly influence the robustness and the efficiency of the implementation of the algorithm discussed. They will be treated below in the order stated.

III.3.1. Stepwise description of the algorithms

The following stepwise description of recursive quadratic programming algorithms gives a general framework for their implementation. Particular members of this class of reduction methods will result from the exact specification of their characteristics such as: the active set strategy, the updating of inverse Hessian information, the calculation of the search directions, the determination of the penalty parameter and, finally, the line search incorporated. These points will be discussed elsewhere in this section. We shall start now with the steps which constitute the algorithms.

Step 1. Initialization. Put $k := 0$. Choose a penalty parameter r_0, a starting point x_0 and some positive definite matrix H_0 as first inverse Hessian approximation. Go to step 2.

Step 2. Determine $I(x_k)$, the current set of active constraints. $I(x_k)$ will contain at least all equality constraints and all currently binding or violated constraints.
If $k = 0$, go to step 5, otherwise go to step 3.

Step 3. Apply the stopping criterion. This means: STOP if

(3.19) $$||x_k - x_{k-1}|| \leq \varepsilon(||x_k|| + 1) \quad \text{for some pregiven } \varepsilon > 0$$

and

(3.20) $$|c_i(x_k)| \leq \varepsilon(||x_k|| + 1) \quad \text{for some pregiven } \varepsilon > 0$$

for all currently violated constraints $c_i(x)$.

Otherwise go to step 4.

Step 4. Update H_k by applying the updating strategy chosen, for instance a self scaling updating can be applied.
Go to step 5.

Step 5. Calculate the approximate Lagrange multipliers from (3.9)

$$\hat{\lambda}_k = (A_k B_k^{-1} A_k^T)^{-1} A_k B_k^{-1} g_k$$

or, alternatively, from (3.14) or (3.16).
Go to step 6.

Step 6. Calculate the search direction p_k from (3.10):

$$p_k = B_k^{-1}(A_k^T(A_k B_k^{-1} A_k^T)^{-1}(A_k B_k^{-1} g_k - \frac{r_k}{2}\hat{\lambda}_k - w_k) - g_k)$$

or, alternatively, from the reduced problems (3.13) or (3.15).
Go to step 7.

Step 7. Determine the new penalty parameter r_k such that

$$p_k^T \nabla P(x_k, r_k) < 0 \quad \text{(see condition I of theorem 3.1)}$$

Go to step 8.

Step 8. Find the stepsize α_k along p_k by minimizing $P(x,r_k)$ along p_k.
Define $x_{k+1} := x_k + \alpha_k p_k$, put $k := k+1$ and go to step 2.

Given the starting point x_0, both r_0 and H_0 are to be defined in the initializing step 1. Mostly the choice $H_0 = I_n$ is made. This choice could be improved by the use of analytically calculated or numerically approximated second order information. However, $H_0 = I_n$ is a simple initialization which enables the comparison with competing algorithms.
Concerning r_0, Himmelblau (1972) discusses some strategies to choose this first penalty parameter. It is unlikely that there will be a strategy which yields an optimal value for all problems. The main reason for this is that usually at the starting point x_0 the active set I^* at the optimum will not be available. Besides the naive strategy of choosing some $r_0 \in \mathbb{R}$ such as $r_0 = 1$, another approach could be to choose r_0 such that the norm of the gradient of the initial exterior penalty function is minimized with respect to r_0.
For a penalty function this means, given

$$(3.21) \qquad P(x_0, r_0) = F(x_0) + \frac{1}{r_0} w(x_0)^T w(x_0)$$

with as gradient vector

(3.22) $$\nabla P(x_0, r_0) = g_0 + \frac{2}{r_0} A_0^T w_0$$

that

(3.23) $$r_0 = - \frac{2(A_0^T w_0)^T (A_0^T w_o)}{g_0^T A_0^T w_0} ,$$

provided that $g_0^T A_0^T w_0 > 0$.

A simplified flowchart of the algorithms is now presented in fig. 3.1 (a detailed flowchart can be found in appendix E, together with a description of the subroutines used).

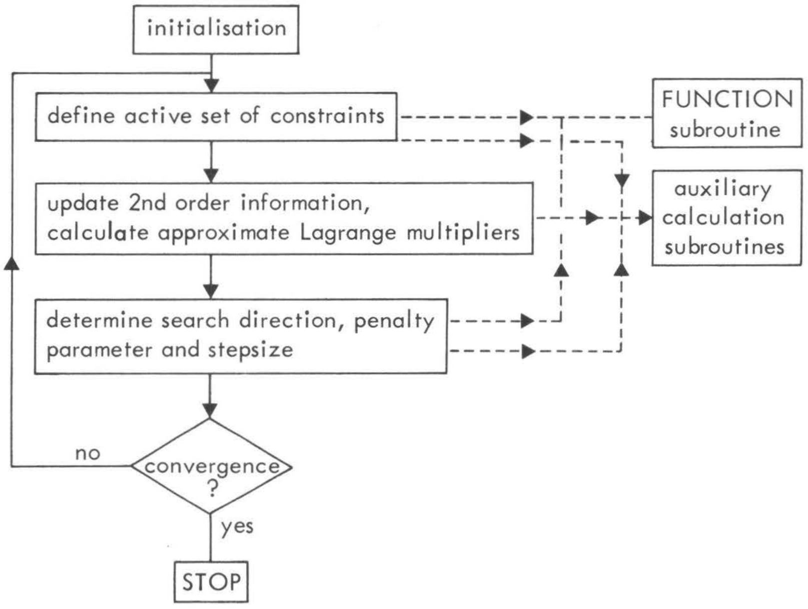

initialisation
define active set of constraints
update 2nd order information, calculate approximate Lagrange multipliers
determine search direction, penalty parameter and stepsize
convergence ?
no
yes
STOP
FUNCTION subroutine
auxiliary calculation subroutines

Figure 3.1 Simplified flowchart of recursive quadratic programming algorithms

III.3.2. Active set strategies

For algorithms which use an active set strategy, the question arises whether this active set should be kept as small as possible or, alternatively, whether the active set should be defined more generously. The first situation could be realized with an active set consisting only of currently violated constraints and the equality constraints. It will be clear that this strategy will cause more changes in the active set than for instance a strategy which forbids the constraints to leave the active set within a certain number of iterations after their entrance. Both kinds of strategies can be supported by arguments. A strategy which maintains equality for a large number of constraints may waste time minimizing the penalty function with 'wrong' constraints. On the other hand a strategy which keeps the number of constraints in the active set as small as possible will inevitably waste time by repeatedly adding and dropping (perhaps the same) constraints. Zigzagging may occur which can lead to nonconvergence or even convergence to the wrong point. In Lenard (1979) and Gill and Murray (1974c) advantages and drawbacks of alternative active set strategies are discussed, especially in the case of linearly constrained nonlinear programming problems. For the nonlinearly constrained nonlinear programming problem proposals for active set strategies can be found in e.g., Murray (1969), Fletcher (1970b, 1971), Fletcher and Lill (1970), Lill (1972) and Biggs (1972).
The active set strategy used in the recursive quadratic programming algorithms defines at the k-th iteration, $k = 1,2, \ldots$, an active set $I(x_k)$ consisting of: all currently violated constraints, all equality constraints and all constraints $c_i(x)$ with $i \in I(x_{k-1})$ with positive approximate Lagrange multiplier.
The last mentioned constraints are added to prevent satisfied constraints to leave the active set too early. In this way zigzagging may be prevented. The removal from the active set of satisfied constraints $c_i(x)$ with $i \in I(x_{k-1})$ and a negative approximate Lagrange multiplier is motivated by the interpretation of this negative multiplier as an indication that these constraints will not be binding at x^*. The efficiency of the resulting algorithm in the determination of the final active set $I(x^*)$ is illustrated in table 3.1. The quotients mentioned have the following meaning:

$$\frac{\text{number of iterations required to detect } I(x^*)}{\text{number of iterations required for convergence}} = \frac{\#\ IT(I(x^*))}{\#\ IT(x^*)}$$

The figures of table 3.1 concern the application of the implemented algorithm to the collection of test problems given in appendix A. Further details on the computational experiments will be presented in chapter VI of this monograph.

Table 3.1: Efficiency in the determination of I(x*)

Problem	$\frac{\#\ IT(I(x^*))}{\#\ IT(x^*)}$	Problem	$\frac{\#\ IT(I(x^*))}{\#\ IT(x^*)}$
1	2 : 12	13	2 : 16
2	0 : 4	14	2 : 30
3	3 : 10	15	18 : 23
4	2 : 88	16	10 : 18
5	25 : 53	17	18 : 24
6	5 : 7	18	36 : 38
7	20 : 35	19	5 : 32
8	11 : 13	20	47 :116
9	59 : 95	21	8 : 12
10	0 : 24	22	0 : 25
11	3 : 5	23	2 : 10
12	6 : 10	24	0 : 7

CONCLUSION. For most problems the algorithm detects I(x*) in one of the first iterations, especially if the number of active constraints at x* is small in comparison with the dimension. More iterations are required to detect I(x*) if the number of active constraints at x* is (almost) equal to the dimension. In that situation it incidentally occured that one of the constraints active at x* did not join $I(x_k)$ for some k, though no longer passive constraints at x* belonged to $I(x_k)$.

Possible consequences of a necessary change of the active set when moving from x_k to $x_k + p$ are discussed in Biggs (1972). It will be evident from table 3.1 that the active set strategy applied in conjunction with a stepsize limitation (the step α has '1' as upper bound) yields an algorithm which is no longer sensitive to this point and determines $I(x^*)$ efficiently. A theoretical background of this observation will be discussed in Van der Hoek and Wymenga (1980), especially the effect of an incorrect active set on the resulting stepsize.

III.3.3. Line search

Though the solution p_k of the reduced problem applied can be written down immediately algebraically, it appeared to be better to perform a line search along p_k. Hence the predicted *step* p_k to the optimum is explored as a *search direction*. Then the ideal case, in which all assumptions are satisfied, will correspond to a stepsize $\alpha = 1$. Violation of the assumptions and local validity of the approximations may give rise to a stepsize $\alpha \neq 1$. We may expect that in a neighbourhood of the solution x^* the line minimizations will produce stepsizes close to 1. For this reason α will be required to satisfy $\alpha \in [0,1]$. Furthermore we know from chapter II that the application of the updating strategies which we discussed there allows for an *inexact* line search. Hence an estimate $\bar{\alpha}$ for α, produced by the line search will be accepted if it corresponds to a 'sufficient' decrease in the value of either the objective function $F(x)$ or the penalty function $P(x, r_k)$. Hence $\bar{\alpha}$ will be accepted if it satisfies either

$$(3.24) \qquad \sigma < \frac{F(x_k + \bar{\alpha} p_k) - F(x_k)}{\bar{\alpha} p_k^T \nabla F(x_k)} < 1 - \sigma$$

or

$$(3.25) \qquad \sigma < \frac{P(x_k + \bar{\alpha} p_k) - P(x_k)}{\bar{\alpha} p_k^T \nabla P(x_k)} < 1 - \sigma$$

for some prechosen $0 < \sigma < \frac{1}{2}$. This is exactly the Goldstein and Price test of chapter II. The either/or character of this test reflects the opinion that both a decrease in $F(x)$ and a decrease in the constraint violation are desirable. Note however, that the objective function in the line search is the penalty function $P(x, r_k)$. The succeeding estimates $\bar{\alpha}$ of α result from *quadratic interpolation* on the interval $[0,1]$. For reasons of robustness several *safeguards* are incorporated such as those suggested in Gill and Murray (1974b), the search direction may even be reversed and if all sophisticated predictions fail, a simple *golden section* line search is applied. As a result an efficient and robust line search is obtained which constitutes the basis of the robustness and the efficiency of the whole algorithm, together with the definition of p_k and the updating of the inverse Hessian approximation.

III.3.4. Updating of the inverse Hessian approximation

This section deals with the question which inverse Hessian matrix should be updated and how this updating should be accomplished. Two possible choices for the Hessian matrix were mentioned in the presentation of the reduced problems (3.8), (3.13) and (3.15): either $[\nabla^2 F(x)]^{-1}$ or $[\nabla^2 L(x,u,v)]^{-1}$. The main reasons for choosing the last alternative are that it contains information on the curvature of both the objective function and the constraints, it is not directly dependent on r_k while it can use the Lagrange multiplier estimates given in (3.9), (3.14) or (3.16). Hence reduced problems (3.13) and (3.15) are to be preferred. Furthermore theorem 3.4 and the remarks at the end of chapter III.2 suggest to use again *positive definite* approximating matrices, as they are obtained in the application of variable metric update formulae for unconstrained optimization. A possible strategy to prevent loss of this positive definiteness in the case of constrained optimization was suggested in Powell (1977a). It amounts to replacing the gradient difference vector y_k by a suitable convex combination of y_k and the last step s_k before applying the BFGS formula. In the experiments, the so called switch II of Oren and Spedicato (discussed in chapter II) was implemented by us. We did not implement the initially scaled BFGS formulae as suggested by Shanno and Phua, as the scaling factor is calculated in a neighbourhood of the starting point where we cannot expect any adequate information to be available on $I(x^*)$, hence on $[\nabla^2 L(x^*,u^*,v^*)]^{-1}$.
The final discussion on the efficiency of the resulting reduction methods is postponed to chapter VI which discusses the design and the results of the computational experiments performed.

IV. ASYMPTOTIC PROPERTIES OF REDUCTION METHODS USING LINEARLY EQUALITY CONSTRAINED REDUCED PROBLEMS

IV.1. Introduction

In this chapter the following general nonlinear programming problem will be considered:

$$(4.1)\qquad \begin{cases} \min F(x) \\ \text{subject to} \\ c_i(x) \begin{Bmatrix} \geq \\ = \end{Bmatrix} 0 \qquad i = 1, 2, \ldots, m \end{cases}$$

The problemfunctions $F(x)$ and $-c_i(x)$, $i = 1, \ldots, m$, are supposed to be sufficiently differentiable convex real functions on E^n.

In this chapter our attention focuses on reduction methods that are based on linearization of the restrictions.

The idea to replace the minimization of a restricted nonlinear programming problem by sequentially minimizing local linearizations of the given problem is not new. One of the first successful implementations of such a reduction method is the Method of Approximation Programming, the MAP-code of Griffith and Stewart (1961).

It replaces the solution procedure of (4.1) by solving the following *sequence* of problems:

$$(4.2)\qquad \begin{cases} \min LF(x_k,x) \\ \text{subject to} \\ Lc_i(x_k,x) \begin{Bmatrix} \geq \\ = \end{Bmatrix} 0 \qquad i = 1, \ldots, m \end{cases}$$

The functions $LF(x_k,x)$ and $Lc_i(x_k,x)$, $i = 1, \ldots, m$ are the linearizations of $F(x)$ and $c_i(x)$ respectively around x_k:

$$LF(x_k,x) := F(x_k) + (x-x_k)^T\nabla F(x_k)$$

and

$$Lc_i(x_k,x) := c_i(x_k) + (x-x_k)^T\nabla c_i(x_k)$$

A natural extension of Griffith and Stewarts algorithm is Wilson's method, Wilson (1963), in which a quadratic approximation of the Lagrangian function of problem (4.1) is minimized subject to the local linearizations of the constraints.

In essence this means that Wilson defines a local linearization of the first order Kuhn-Tucker conditions of (4.1) which is optimized using an algorithm for quadratic programming.

Just as in Beale's algorithm for quadratic programming, Beale (1959), we need the equation of the hypersurfaces at which the partial derivative of F(x) with respect to any nonbasic variable vanishes and the equation of the hypersurfaces at which any new constraint becomes active. Beale (1967) showed that the error in both equations is $o(x_{k+1} - x_k)$ as long as the active set is constant.

The main advantages and drawbacks of linearizations in reduction methods are summarized below:

(i) Linearization methods are relatively simple to present and to implement.

(ii) If the next iteration point x_{k+1} happens to be infeasible, an intermediate step is required to move back to the feasible region if the algorithm is a feasible point method. This may give rise to slow convergence.

(iii) In almost all proposed methods *all* constraints are linearized at every step and no use is made of information on the status of constraints (active, passive) gathered in the course of the iteration process. Exceptions in this respect are e.g., Wolfe (1961) and Holtgrefe's implementation of Kelley's cutting plane method, Holtgrefe (1975).

(iv) The linearizations of nonlinear problem functions are only acceptable approximations in a neighbourhood of x_k. This makes stepsize limitations such as

(4.3) $$|(x_k - x_{k+1})_j| \leq \delta_{kj}, \ \delta_{kj} > 0, \ j = 1,\dots,n, \text{ inevitable.}$$

(v) A consequence of linearizations is also that poor search directions may be generated.

(vi) The local validity of the linearizations prohibits the application of extrapolation techniques to accelerate convergence.

(vii) Wilson's method, which uses a second order approximation, requires the expensive calculation of second order derivatives.

During the last two decades alternative, more sophisticated reduction methods, still using linearizations, have been proposed which are designed to avoid the above mentioned drawbacks. We mention: Rosen (1960, 1961); Robinson (1972); Rosen and Kreuser (1972); Gruver and Engersbach (1974,1976);

Rosen (1977); Ballintijn, Van der Hoek and Hooykaas (1978); Van der Hoek (1979) and Van der Hoek and Hooykaas (1979). In those cases the reduced problem is defined by:

$$(4.4)\qquad \begin{cases} \min F(x) + \phi(x_k,x) \\ \text{subject to} \\ Lc_i(x_k,x) \begin{Bmatrix} \geq \\ = \end{Bmatrix} 0 \qquad i = 1, 2, \ldots, m \end{cases}$$

In (4.4) the objective function is corrected with a term $\phi(x_k,x)$ which is supposed to offset by means of a corrected objective function in the reduced problem possible poor behaviour of the algorithm caused by the applied linearizations. So $\phi(x_k,x)$ will generally depend on $c_i(x)$ and/or on $Lc_i(x_k,x)$, $(i = 1, 2, \ldots, m)$, and different reduction methods arise from different choices of $\phi(x_k,x)$. For instance, Rosen and Kreuser (1972) use

$$(4.5)\qquad \phi(x_k,x) = \sum_{\lambda_i<0} \lambda_i(x_k)c_i(x)$$

where $\lambda_i(x_k)$, $i = 1, \ldots, m$, are the current Lagrange multiplier estimates. Here $\phi(x_k,x)$ can be viewed as a linear penalty term or as a restricted Lagrangian function. In Van der Hoek (1978), this reduction method is further simplified by merely linearizing the constraints of the active set $I(x_k)$ at x_k. Robinson (1972) proposes to use

$$(4.6)\qquad \phi(x_k,x) = \sum_{i=1}^{m} \lambda_i(x_k)[Lc_i(x_k,x) - c_i(x)]$$

Rosen (1977), Bräuniger (1977), Ballintijn, Van der Hoek and Hooykaas (1978) apply modifications of (4.6) in their definition of the reduced problem. We shall want to take advantage of the possible presence of already linear constraints and we shall want to distinguish equality constraints from inequality constraints. Thus we formulate the problem (4.1) in another way: renumber the constraints in such a way that the indices $i = 1, \ldots, m_1$ correspond with linear equality constraints and $i = m_1+1, \ldots, m_2$ with linear inequality constraints. Let $L \subset E^n$ be the collection of all $x \in E^n$ satisfying the linear constraints:
$a_i^T x \begin{Bmatrix} \geq \\ = \end{Bmatrix} b_i$, $i = 1, \ldots, m_2$, with $a_i \in E^n$, $b_i \in \mathbb{R}$.

$$(4.7)\qquad L := \left\{ x \in E^n \;\middle|\; \begin{array}{l} a_i^T x - b_i = 0,\ i = 1, \ldots, m_1, \text{ and} \\ a_i^T x - b_i \geq 0,\ i = m_1+1, \ldots, m_2 \end{array} \right\}$$

Further, using $i = m_2+1, \ldots, m_3$ and $i = m_3+1, \ldots, m_4$ for the nonlinear equality and inequality constraints respectively, we denote by NL the collection of all $x \in E^n$ that meet the nonlinear constraints:

$$(4.8) \qquad NL := \left\{ x \in E^n \;\middle|\; \begin{array}{l} c_i(x) = 0,\ i = m_2+1, \ldots, m_3 \text{ and} \\ c_i(x) \geq 0,\ i = m_3+1, \ldots, m_4 \end{array} \right\}$$

Then (4.1) can be stated as

$$(4.9) \qquad \begin{array}{l} \min \ F(x) \\ x \in L \cap NL \end{array}$$

Finally, the collection of all linearized nonlinear constraints, linearized around x_k, is given by $LNL(x_k)$:

$$(4.10) \qquad LNL(x_k) := \left\{ x \in E^n \;\middle|\; \begin{array}{l} Lc_i(x_k, x) = 0,\ i = m_2+1, \ldots, m_3, \text{ and} \\ Lc_i(x_k, x) \geq 0,\ i = m_3+1, \ldots, m_4 \end{array} \right\}$$

Then Robinson's reduced problem is

$$(4.11) \qquad \min_{x \in L \cap LNL(x_k)} \quad F(x) + \sum_{i=m_2+1}^{m_4} \lambda_i(x_k)[Lc_i(x_k,x) - c_i(x)]$$

Clearly the linear constraints (the indices $i = 1, \ldots, m_2$) do not contribute to the objective function of the reduced problem. One of the proposed algorithms in this chapter is to linearize merely the restrictions of the current active set $I(x_k)$. Then the reduced problem becomes:

$$(4.12) \qquad \min_{x \in L \cap LNL(x_k)} \quad F(x) + \sum_{i \in I(x_k)} \lambda_i(x_k)[Lc_i(x_k,x) - c_i(x)]$$

It is obvious that the active set strategy must define $I(x_k)$ in such a way that the indices of all equality constraints belong to the active set:

$$\{1, \ldots, m_1\} \cup \{m_2+1, \ldots, m_3\} \subset I(x_k)$$

Note that both (4.11) and (4.12) have the property that a linearly constrained original problem equals its reduced problem, which means that the solution of the original problem amounts to the solution of only one

reduced problem.
In comparison with the reduction methods mentioned above, we also investigated and implemented the following new aspects:

(i) The reduced problem is defined solely in terms of the objective function and the constraints of the current active set;

(ii) The (non-)linearity of constraints is used explicitly;

(iii) The algorithm for linearly constrained reduced problems only tests whether a constraint has to be dropped from the active set if:
  a. the optimum with respect to the current active set is obtained;
  b. accumulation of calculation errors forces a reinitialization of the current inverse Hessian approximation.
  c. changes in the active set occur.

(iv) The coupling of the applied so called phase I, designed to provide us with a good starting point, and phase II (the algorithm to be developed in this chapter) is discussed.
Suggestions to obtain a good starting point together with a good initial set of active constraints are discussed as well.

(v) The code applied for linearly constrained nonlinear programming uses Cholesky decompositions for the matrices B_k and $N_k^T H_k N_k$ (see Ballintijn, Van der Hoek and Hooykaas (1978)).

(vi) The active set strategy required new updating formulae for updating the Cholesky factors of $N_k^T H_k N_k$ (see Ballintijn, Van der Hoek and Hooykaas (1978)).

(v) Theoretical results on the convergence of the algorithm are presented.

IV.2. Definition and solution of linearly constrained reduced problems

A general form for a sequence of linearly constrained reduced problems was given in (4.4) . Once the idea of linearization of the constraints is accepted, the reduction method is characterized by the particular choice of $\phi(x_k,x)$.

Rosen and Kreuser (1972) considered a linear penalty term for $\phi(x_k,x)$:

$$(4.13) \qquad \phi(x_k,x) = \sum_{i\in I(x_k)} \lambda_i(x_k)c_i(x)$$

Their reduction method uses an analogous function $\phi(x_k,x)$ as Kelley and Speyer (1970) used to improve Rosen (1961). Lill (1972) also applies a similar function $\phi(x_k,x)$. In (4.13) the index set $I(x_k)$ consists of the indices of all violated nonlinear constraints; but *all* nonlinear constraints, linearized around x_k, are kept (see (4.4)). In a computational study, Van der Hoek (1978), we investigated an implementation of this reduction method in which only the constraints of the current active set $I(x_k)$ contribute to the objective function of the reduced problem, and only those constraints are linearized. A further, extensive treatment of the backgrounds of reduction methods based on the application of (4.13), their convergence properties and computational results can be found in Kreuser (1972), Rosen and Kreuser (1972) and Van der Hoek (1978).

If we compare the reduction methods based on the application of (4.13) with Griffith and Stewarts MAP-method we see that $\phi(x_k,x)$ should give a compensation in the objective function for the effect of linearizing the constraints. Beale (1967) summarizes the geometrical backgrounds as: 'the constraints are straightened out at the expense of the contours of constant values of the objective function. If the latter contours are drawn as broken lines, we must transform a problem looking as in figure 4.1.

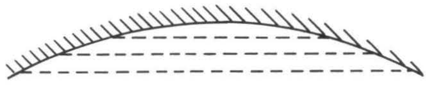


Figure 4.1

into one looking as in figure 4.2.

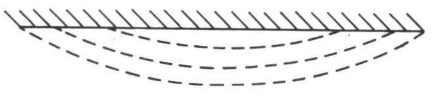


Figure 4.2

A motivation to move nonlinearities of constraint functions to the objective function is that essentially most algorithms for linearly constrained nonlinear programming apply adjusted (e.g., projected) unconstrained search directions. In this way the experience with solving unconstrained nonlinear programming problems is applied. Problems that can arise from the application of linearizations are illustrated by the following simple example:
minimize $x^2 - 2x$ subject to $0 \leq x \leq 2$. Solving this problem via successive linearization will always yield trial values of x at either the upper or the lower bound imposed by stepsize limitations, which means that in essence those stepsize limitations control the convergence. Applying linear approximations to the same problem formulated as
minimize z subject to $z \geq x^2 - 2x$, in which nonlinearities only occur in the constraints, again requires stepsize limitations to converge (obviously a quadratic approximation of the objective function solves the problem in one step).
Rosen (1963) showed that the geometrical transformation illustrated in figures 4.1 and 4.2 can be obtained algebraically using the shadowprices on the constraints. In that way the nonlinearities in the constraints are thrown into the objective function. By means of a counterexample he showed that the trivial case $\phi(x_k,x) = 0$ will not, in general, solve the standard convex nonlinear programming problem: Rosen (1977).

A comparison of the first order Kuhn-Tucker conditions of problems (4.1) and (4.4) suggests to look for functions $\phi(x_k,x)$ with the properties

(4.14) $\qquad \phi(x_k,x_k) = 0$ and $\nabla_x \phi(x_k,x)(x_k) = 0$

The reduced problems (4.11) and (4.12) possess ϕ-functions which meet these requirements while (4.13) does not!

The discussion so far can be summarized in the following stepwise description of reduction methods based on the application of (4.4), where $\phi(x_k,x)$ is still to be chosen, for instance from (4.11) or (4.12).

Step 1. Set k := 0. Initialize variables.

Step 2. Arrived at x_k, find a first order Kuhn-Tucker point x_{k+1} of

$$\min_{x \in L \cap LNL(x_k)} F(x) + \phi(x_k,x) \tag{4.15}$$

If x_{k+1} is not unique, choose the Kuhn-Tucker point which is closest to the preceding Kuhn-Tucker point x_k.

Step 3. Apply convergence tests.
In case of non-convergence set k := k+1 and go to step 2.

Remarks

1. We shall prove in theorem 4.7 that, if the algorithm is started close enough to a Kuhn-Tucker point of problem (4.1), convergence is guaranteed and will be R-quadratic, if $\phi(x_k,x)$ satisfies (4.11) or (4.12). In this respect the algorithm possesses similar properties as the algorithms of Robinson (1972) and Bräuniger (1977).
2. If the original problem is linearly constrained, the algorithm requires one major iteration as then the reduced problem equals the original problem.
3. If the original problem is a convex programming problem (both F(x), $\phi(x_k,x)$ and all $-c_i(x)$ are convex functions, while the equality constraints are affine), the reduced problem (4.15) is a convex programming problem as well, as the Lagrange multipliers $\lambda(x_k)$ of the inequality constraints are nonnegative for all k.
4. If L is compact, L∩NL is a closed subset of the compact set L, so L∩NL is compact as well. Then a continuous function will attain a global minimum value F_1 at some point x_1 of L and a global minimum value F_2 at some point $x_2 \in L \cap NL$.
   If $x_1 \in L \cap NL$, the nonlinear constraints are redundant. These minima are unique if F(x) is convex on a convex feasible region.
5. Linearization of concave, differentiable nonlinear constraint functions $c_i(x)$ enlarges the feasible region. This follows directly from the following equivalent definition of concavity (see e.g. Zangwill (1969)): the function $c_i(x)$, which is differentiable on E^n, is concave if and only if

(4.16) $$c_i(x) \le c_i(x_k) + (x-x_k)^T\nabla c_i(x_k) \qquad \text{for all } x,\ x_k \in E^n$$

This means that $c_i(x)$ is concave if and only if

(4.17) $$c_i(x) \le Lc_i(x_k,x) \qquad \text{for all } x,\ x_k \in E^n$$

which is especially valid for all $x \in NL$. This proves the remark. Note that there are no restrictions on the choice of x_k. So we conclude that linearization of the constraints of a feasible problem yields a feasible reduced problem. What may happen in the case of linearization of an infeasible problem is illustrated in remark 7.

6. If we drop the requirement that all $c_i(x)$ should be concave functions of x, linearization may yield an infeasible reduced problem from a feasible nonlinearly constrained problem. This is illustrated by the following feasible problem with a nonconcave function $c_i(x)$. It shows that $L \cap LNL(x_k) = \phi$ may occur in this situation.
   Consider the following constraint set:

$$L := \{x \mid -2 \le x \le \tfrac{1}{2}\}$$
$$NL := \{x \mid x - x^3 \ge 0\} = \{x \mid x \le -1\} \cup \{x \mid 0 \le x \le 1\}$$

   Then $c(x)$, which defines NL is not concave on E^1 and NL is not connected.
   The feasible region is: $L \cap NL = \{x \mid -2 \le x \le -1\} \cup \{x \mid 0 \le x \le \tfrac{1}{2}\}$.
   From $Lc(x_k,x) = 2x_k^3 - 3xx_k^2 + x$, we obtain as linearized constraint $Lc(-\tfrac{1}{2},\ x) \ge 0 : x \ge 1$.
   Then $L \cap LNL(-\tfrac{1}{2}) = \{x \mid -2 \le x \le \tfrac{1}{2}\} \cap \{x \mid x \ge 1\}$
   $= \phi$
   So linearization around this infeasible point x_k $(x_k \in L,\ x_k \notin NL)$ yields an empty linearized constraint set. From a further analysis of this example we can see that infeasible linearized problems can arise both from points $x_k \in L$ and $x_k \notin L$.

7. An infeasible problem with a concave differentiable function $c(x)$ may lead to both feasible and infeasible linearized constraint sets.
   This is illustrated by the following example:

$$L := \{x \mid \tfrac{3}{2} \le x \le 10\}$$

$$NL := \{x \mid 1 - x^2 \ge 0\}.\ \text{Then } L \cap NL = \phi.$$

Linearization of the constraint function $c(x) = 1 - x^2$ yields

$$Lc(x_k,x) = -2x_k x + x_k^2 + 1$$

The choice $x_k = \frac{1}{2}$ leads to

$$\begin{aligned} LNL(\tfrac{1}{2}) &= \{x \mid Lc(\tfrac{1}{2},x) \geq 0\} \\ &= \{x \mid x \leq \frac{5}{4}\}. \end{aligned}$$

We see that $L \cap LNL(\frac{1}{2}) = \phi$: linearization around an infeasible point that satisfies the nonlinear constraint yields an infeasible reduced problem.

The choice $x_k = 4$ leads to

$$\begin{aligned} LNL(4) &= \{x \mid Lc(4, x) \geq 0\} \\ &= \{x \mid x \leq \frac{17}{8}\} \end{aligned}$$

and $L \cap LNL(4) = \{x \mid \frac{3}{2} \leq x \leq \frac{17}{8}\}$: a nonempty feasible region for an $x_k \in L$ whereas $L \cap NL = \phi$.

8. Remarks 6 and 7 illustrate the necessity of the requirements of remark 5 to guarantee feasible reduced problems. The question now arises what will be the best strategy for choosing the points x_k. The importance of this question is even greater when we reflect that in practical, real-life problems the situations sketched in remarks 6 and 7 really occur. That is why we decided to require $x_k \in L \cap NL$, i.e. x_k is feasible for feasible problems.
Then

$$Lc_i(x_k,x_k) = c_i(x_k) \geq 0 \quad \text{for all } i,$$

independent of the concavity of $c_i(x)$, so x_k is feasible with respect to the linearized constraints as well and consequently $L \cap LNL(x_k) \neq \phi$. Generally during the iteration process infeasible points might be generated, so some restoration-procedure should be available to move back to the feasible region. Examples of such restoration procedures can be found in Gruver and Engersbach (1974, 1976), de Jong (1977) and Van der Hoek (1978). The implemented procedure is described in ch. V.

9. As mentioned in remark 1, the starting point must be 'close enough' to the solution. That is why an initializing, so called 'phase I', procedure is incorporated.
   Phase I generates an acceptable starting point. It amounts to solving the problem

$$\min_{x \in L} F(x) + P(x) \tag{4.18}$$

   where $P(x)$ is an exterior penalty term which is defined by

$$P(x) = t_k \left[\sum_{i=m_2+1}^{m_3} [c_i(x)]^2 + \sum_{i=m_3+1}^{m_4} [c_i^-(x)]^2 \right] \tag{4.19}$$

   In this definition $t_k > 0$ is a penaltyparameter whose choice will be discussed in section 7, while $c_i^-(x)$ is defined by

$$c_i^-(x) := \min[c_i(x), 0], \text{ for all } x, \ i = m_3+1, \ldots, m_4 \tag{4.20}$$

   The solution of the phase I step defined by (4.18) requires the use of an algorithm for linearly constrained nonlinear programming which is required in phase II of the algorithm as well.

With these remarks this chapter on the definition and solution of linearly constrained subproblems is completed. Our next task is to investigate the relations between the Kuhn-Tucker points of the original- and the reduced problems.

IV.3 Relations between the first order Kuhn-Tucker conditions of the original- and the reduced problems

As the properties to be discussed in this chapter only depend on the fact whether a constraint is an equality or an inequality constraint, and not on its being linear or not, we prefer to return to the original problem formulation (4.1) in which we renumber the constraints in such a way that the indices $i = 1, \ldots, p$ correspond with the inequality constraints while with $i = p+1, \ldots, m$ the equality constraints are meant.
Thus we get as problem formulation

$$(4.21) \quad \begin{cases} \min F(x) & \\ \text{subject to} & \\ c_i(x) \geq 0 & i = 1, \ldots, p \\ c_i(x) = 0 & i = p+1, \ldots, m \end{cases}$$

The Langrangian function associated with problem (4.21) is:

$$(4.22) \qquad L(x,u,v) = F(x) - \sum_{i=1}^{p} u_i c_i(x) - \sum_{i=p+1}^{m} v_i c_i(x)$$

where u_i, $i = 1, \ldots, p$, and v_i, $i = p+1, \ldots, m$ denote the Lagrangian multipliers of the inequality and equality constraints respectively.

The first order Kuhn-Tucker conditions for problem (4.21) are:

$$(4.23) \qquad \nabla_x L(x,u,v) = 0$$

$$(4.24) \qquad u_i c_i(x) = 0 \qquad i = 1, \ldots, p$$

$$(4.25) \qquad c_i(x) = 0 \qquad i = p+1, \ldots, m$$

$$(4.26) \qquad c_i(x) \geq 0 \qquad i = 1, \ldots, p$$

$$(4.27) \qquad u_i \geq 0 \qquad i = 1, \ldots, p$$

We shall denote the first order Kuhn-Tucker points of (4.21) by $z = (x,u,v) \in E^{n+m}$ or, at iteration k, by $z_k = (x_k, u_k, v_k) \in E^{n+m}$.

In this notation the conditions (4.23) - (4.25) can be studied in terms of a mapping $f : E^{n+m} \to E^{n+m}$ given by the following

DEFINITION

(4.28) $$f(z) := \begin{bmatrix} \nabla_x F(x) - \sum_{i=1}^{p} u_i \nabla_x c_i(x) - \sum_{i=p+1}^{m} v_i \nabla_x c_i(x) \\ \\ u_1 c_1(x) \\ \vdots \\ u_p c_p(x) \\ c_{p+1}(x) \\ \vdots \\ c_m(x) \end{bmatrix}$$

The following lemma is clear from the definition of $f(z)$

LEMMA 4.1

$z \in E^{n+m}$ is a first order Kuhn-Tucker point of (4.21) if and only if $f(z) = 0$ and (4.26) and (4.27) are satisfied. $\square$

This approach to the first order Kuhn-Tucker conditions by means of $f(z)$ was first followed by Mc Cormick (1971) who pointed out that $\nabla_z f(z_k)$ is nonsingular and $\| \nabla_z f(z_k)^{-1} \|$ exists if z_k satisfies the second order sufficiency conditions of problem (4.21) with strict complementary slackness of u_i and $c_i(x)$ for $i = 1, \ldots, p$.

If we state the kth reduced problem in a formulation analogous to the original problem we get

(4.29) $$\left\{ \begin{array}{ll} \min F(x) + \phi(x_k, x) & \\ \text{subject to} & \\ Lc_i(x_k, x) \geq 0 & i = 1, \ldots, p \\ Lc_i(x_k, x) = 0 & i = p+1, \ldots, m \end{array} \right.$$

with the additional requirement (4.14):

$$\phi(x_k, x) = 0 \text{ and } \nabla_x \phi(x_k, x) = 0 \text{ at } x = x_k.$$

The Lagrangian function associated with (4.29) is

$$(4.30) \qquad L'(x,\mu,\nu) = F(x) + \phi(x_k,x) - \sum_{i=1}^{p} \mu_{k,i} Lc_i(x_k,x) + \\ - \sum_{i=p+1}^{m} \nu_{k,i} Lc_i(x_k,x)$$

with as Lagrangian multipliers $\mu_{k,i}$, $i = 1, \ldots, p$ and $\nu_{k,i}$ $i = p+1, \ldots, m$ for the inequality and the equality constraints respectively.
The first order Kuhn-Tucker conditions of (4.29) are

$$(4.31) \qquad \nabla_x L'(x,\mu,\nu) = 0$$

$$(4.32) \qquad \mu_{k,i} Lc_i(x_k,x) = 0 : \mu_{k,i}\{c_i(x_k) + (x-x_k)^T \nabla_x c_i(x_k)\} = 0 \qquad i = 1, \ldots, p$$

$$(4.33) \qquad Lc_i(x_k,x) = 0 \qquad : c_i(x_k) + (x-x_k)^T \nabla_x c_i(x_k) = 0 \qquad i = p+1, \ldots, m$$

$$(4.34) \qquad Lc_i(x_k,x) \ge 0 \qquad : c_i(x_k) + (x-x_k)^T \nabla_x c_i(x_k) \ge 0 \qquad i = 1, \ldots, p$$

$$(4.35) \qquad \mu_{k,i} \ge 0 \qquad i = 1, \ldots, p$$

Note that in the reduced problem (4.29) all nonlinear constraints are linearized. As it is our intention to linearize only the nonlinear constraints of the current active set, we shall only pay limited attention to the relations between the Kuhn-Tucker points of problems (4.21) and (4.29). Anticipating on the discussion below we merely mention here the following proposition.

PROPOSITION 4.1

If $z_k = (x_k,u_k,v_k)$ is a regular Kuhn-Tucker point of (4.21) and if strict complementary slackness holds in both (4.24) and (4.32), then z_k is also a regular Kuhn-Tucker point of (4.29) as well.

Proof:

$$\nabla_x L'(x,\mu,\nu) = \nabla_x F(x) + \nabla_x \phi(x_k,x) - \sum_{i=1}^{p} \mu_{k,i} \nabla_x Lc_i(x_k,x) +$$

$- \sum_{i=p+1}^{m} \nu_{k,i} \nabla_x Lc_i(x_k,x)$. (4.14) gives $\nabla_x \phi(x_k,x_k) = 0$ and

$\nabla_x Lc_i(x_k,x) = \nabla_x c_i(x_k)$ hence it follows that $\nabla_x L'(x_k,\mu,\nu)$

$= \nabla_x F(x_k) - \sum_{i=1}^{p} \mu_{k,i} \nabla_x c_i(x) - \sum_{i=p+1}^{m} \nu_{k,i} \nabla_x c_i(x_k) = \nabla_x L(x_k,\mu,\nu)$.

As z_k is a regular Kuhn-Tucker point of (4.21) under the strict complementarity assumption we know that the Lagrangian multipliers u_k, v_k are uniquely determined by $\nabla_x L(x_k,u,v) = 0$ and (4.24) which means that $\nabla_x L'(x_k,u_k,v_k) = 0$; the validity of the remaining conditions (4.32) - (4.35) follows directly from (4.24) - (4.27) , while the regularity of z_k with respect to (4.29) is a consequence from its regularity with respect to (4.21). □

For the special case that $\phi(x_k,x)$ is defined by (4.6) this proposition extends to the following theorem.

THEOREM 4.1 (Robinson, 1972)

Let all problem functions be differentiable. Then the following statements concerning a given point (x^*,u^*,v^*) are equivalent:

(i) There exist $u \in \mathbb{R}^p$, $v \in \mathbb{R}^{m-p}$ such that (x^*, u^*, v^*) satisfies the Kuhn-Tucker conditions for (4.29) with $x_k = x^*$.

(ii) (x^*, u^*, v^*) satisfies the Kuhn-Tucker conditions for (4.21)

(iii) For every $u \in \mathbb{R}^p$ and every $v \in \mathbb{R}^{m-p}$, (x^*, u^*, v^*) satisfies the Kuhn-Tucker conditions for (4.29) with $x_k = x^*$.

The proof of this theorem is clear from the definition of the Kuhn-Tucker conditions above.

Proposition 4.1 and theorem 4.1 mean that as soon as the primal variables x_k of a Kuhn-Tucker point of (4.21) are identified, there exist dual variables u_k, v_k such that z_k solves the next reduced problem, independent of the correctness of the dual variables applied in the definition of the reduced problem. □

Just as in problem (4.21) , the first order Kuhn-Tucker conditions of (4.29) can be described in terms of a mapping $d(z_k,z)$ from E^{n+m} into E^{n+m}. For an arbitrary $z_k = (x_k,u_k,v_k)$ this mapping is defined as follows for $z = (x,u,v)$:

DEFINITION

(4.36)
$$d(z_k,z) = \begin{bmatrix} \nabla_x L'(x,\mu,\nu,) \\ \mu_{k,1} Lc_1(x_k,x) \\ \vdots \\ \mu_{k,p} Lc_p(x_k,x) \\ Lc_{p+1}(x_k,x) \\ \vdots \\ Lc_m(x_k,x) \end{bmatrix}$$

$$= \begin{bmatrix} \nabla_x F(x) + \nabla_x \phi(x_k,x) - \sum_{i=1}^{p} \mu_{k,i} \nabla_x Lc_i(x_k,x) - \sum_{i=p+1}^{m} \nu_{k,i} \nabla_x Lc_i(x_k,x) \\ \mu_{k,1} Lc_1(x_k,x) \\ \vdots \\ \mu_{k,p} Lc_p(x_k,x) \\ Lc_{p+1}(x_k,x) \\ \vdots \\ Lc_m(x_k,x) \end{bmatrix}$$

$$= \begin{bmatrix} \nabla_x F(x) - \sum_{i=1}^{p} u_i \nabla_x c_i(x) - \sum_{i=p+1}^{m} v_i \nabla_x c_i(x) \\ u_1 c_1(x) \\ \vdots \\ u_p c_p(x) \\ c_{p+1}(x) \\ \vdots \\ c_m(x) \end{bmatrix} +$$

$$- \begin{bmatrix} -\nabla_x \phi(x_k,x) - \sum_{i=1}^{p} u_i \nabla_x c_i(x) + \sum_{i=1}^{p} \mu_{k,i} \nabla_x c_i(x_k) + \\ \qquad - \sum_{i=p+1}^{m} v_i \nabla_x c_i(x) + \sum_{i=p+1}^{m} \nu_{k,i} \nabla c_i(x_k) \\ u_1 c_1(x) - \mu_{k,1} Lc_1(x_k,x) \\ \vdots \\ u_p c_p(x) - \mu_{k,p} Lc_p(x_k,x) \\ c_{p+1}(x) - Lc_{p+1}(x_k,x) \\ \vdots \\ c_m(x) - Lc_m(x_k,x) \end{bmatrix}$$

This means that

$$(4.37) \qquad d(z_k,z) = f(z) - \psi(z_k,z)$$

where $\psi(z_k,z)$ follows from the equation above. In the case of Robinson's reduction method (see e.g. (4.6)) this gives rise to

$$(4.38)\quad d(z_k,z) = f(z) - \begin{bmatrix} \sum_{i=1}^{p} (\mu_{k,i} - u_i) \nabla_x c_i(x_k) + \sum_{i=p+1}^{m} (\nu_{k,i} - v_i) \nabla_x c_i(x_k) \\ u_1 c_1(x) - \mu_{k,1} Lc_1(x_k,x) \\ \vdots \\ u_p c_p(x) - \mu_{k,p} Lc_p(x_k,x) \\ c_{p+1}(x) - Lc_{p+1}(x_k,x) \\ \vdots \\ c_m(x) - Lc_m(x_k,x) \end{bmatrix}$$

which can be interpreted as a relation expressing the difference between the Kuhn-Tucker condition of problems (4.21) and (4.29).
In an analogous way as for $f(z)$, we can formulate from the definition of $d(z_k,z)$ a lemma on the first order Kuhn-Tucker points of (4.29):

LEMMA 4.2

$z \in E^{n+m}$ is a first order Kuhn-Tucker point of (4.29) if and only if $d(z_k,z) = 0$ and (4.34), (4.35) are satisfied.

We shall denote by $S(z_k)$ the collection of all first order Kuhn-Tucker points of (4.29). Hence $S(z_k)$ is defined by:

(4.39) $\quad S(z_k) := \{z \in R^{n+m} \mid d(z_k,z) = 0$ and (4.34),(4.35) are satisfied$\}$

The relation between the first order Kuhn-Tucker points of problems (4.21) and (4.29), as given in proposition 4.1 now extends as follows:

PROPOSITION 4.2

The Taylor expansions around $z_k = (x_k,u_k,v_k)$ of $f(z)$ and $d(z_k,z)$ are equal up to second order terms if we use Robinson's $\phi(x_k,x)$ proposal

$$\phi(x_k,x) = \sum_{i=1}^{p} u_{k,i}(Lc_i(x_k,x) - c_i(x)) + \sum_{i=p+1}^{m} v_{k,i}(Lc_i(x_k,x) - c_i(x))$$

Proof:

Clearly the proposition relates the first order Kuhn-Tucker conditions of problems (4.21) and (4.29). The reduced problem (4.29) is derived from (4.21) after the point $z_k = (x_k,u_k,v_k)$ is reached. Where necessary we shall denote the p elements of the Lagrangian vector u_k by $u_{k,i}$ and the (m-p) elements of v_k by $v_{k,i}$. The Lagrangian coefficients of the reduced problem are again denoted by $\mu_{k,i}$ and $\nu_{k,i}$ respectively.

From the Taylor expansions

$$f(z) = f(z_k) + (z-z_k)^T\nabla_z f(z_k) + \ldots$$

and

$$d(z_k,z) = d(z_k,z_k) + (z - z_k)^T\nabla_z d(z_k,z)(z_k) + \ldots$$

we see that we have to prove:

(i) $\quad f(z_k) = d(z_k,z_k)$

(ii) $\quad \nabla_z f(z_k) = \nabla_z d(z_k,z_k)$ which means

(iia) $\quad \nabla_x f(z_k) = \nabla_x d(z_k,z)(z_k)$

(iib) $\quad \nabla_u f(z_k) = \nabla_\mu d(z_k,z)(z_k)$

(iic) $\quad \nabla_v f(z_k) = \nabla_\nu d(z_k,z)(z_k)$

Relation (i) follows immediately from (4.38).

For (iia) we observe:

$$\nabla_x f(z_k) = \begin{bmatrix} \nabla^2_{xx} F(x_k) - \sum_{i=1}^{p} u_{k,i} \nabla^2_{xx} c_i(x_k) - \sum_{i=p+1}^{m} v_{k,i} \nabla^2_{xx} c_i(x_k) \\ u_{k,i} \nabla_x c_1(x_k) \\ \vdots \\ u_{k,p} \nabla_x c_p(x_k) \\ \nabla_x c_{p+1}(x_k) \\ \vdots \\ \nabla_x c_m(x_k) \end{bmatrix} \quad \text{and}$$

$$\nabla_x d(z_k,z)(z_k) = \begin{bmatrix} \nabla^2_{xx} F(x_k) + \nabla^2_{xx} \phi(x_k,x_k) \\ u_{k,i} \nabla_x Lc_1(x_k,x_k) \\ \vdots \\ u_{k,p} \nabla_x Lc_p(x_k,x_k) \\ \nabla_x Lc_{p+1}(x_k,x_k) \\ \vdots \\ \nabla_x Lc_m(x_k,x_k) \end{bmatrix}$$

From the definition of $\phi(x_k,x)$ we see

$$\nabla^2_{xx} \phi(x_k,x_k) = - \sum_{i=1}^{p} u_{k,i} \nabla^2_{xx} c_i(x_k) - \sum_{i=p+1}^{m} v_{k,i} \nabla^2_{xx} c_i(x_k)$$

which, together with $\nabla_x Lc_i(x_k,x_k) = \nabla_x c_i(x_k)$ for $i = 1, \ldots, m$ yields $\nabla_x f(z_k) = \nabla_x d(z_k,z)(z_k)$.

The proof of (iib) follows directly from (4.28) and (4.36). Indeed

$$\frac{\partial f}{\partial u_i}(z_k) = \begin{bmatrix} -\nabla_x c_i(x_k) \\ 0 \\ \vdots \\ 0 \\ c_i(x_k) \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad \frac{\partial d(z_k,z)}{\partial \mu_i}(z_k) = \begin{bmatrix} -\nabla_x c_i(x_k) \\ 0 \\ \vdots \\ 0 \\ Lc_i(x_k,x_k) \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

combined with $Lc_i(x_k,x_k) = c_i(x_k)$ for all i.

Finally (iic) follows from (4.28) and (4.36) again:

$$\frac{\partial f}{\partial v_i}(z_k) = \begin{bmatrix} -\nabla_x c_i(x_k) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \frac{\partial d(z_k,z)}{\partial \nu_i}(z_k) = \begin{bmatrix} -\nabla_x c_i(x_k) \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

□

COROLLARY

$\nabla_z f(z) = \nabla_z d(z,z)$ for all $z \in E^{n+m}$.

Proof:

This corollary restates part (ii) of the preceding proof. Note that the proof is purely formal and hence independent of the fact whether z is a Kuhn-Tucker point or not.

□

Remark

In terms of the original problem formulation this result can be interpreted as follows: For this $\phi(x_k,x)$ the quadratic approximations of the Lagrangian functions $L(x,u,v)$ and $L'(x,\mu,\nu)$ associated with the original problem (4.21) and the reduced problem (4.29) respectively, are identical in a neighbourhood of $z_k \in E^{n+m}$.

Linearizing the first order Kuhn-Tucker conditions is also the key of Wilson's reduction method, Wilson (1963). It then solves those linearized conditions using Dantzig's version of Wolfe's quadratic programming algorithm. The main disadvantage of Wilson's method is that it requires repeated calculation of second derivatives. This makes it less attractive.

Robinson (1972) stated a number of properties relating $f(z)$, $d(z_k,z)$ and their respective gradients in a neighbourhood of a Kuhn-Tucker point z^* of the original problem (4.21). These we state without proof in the theorem below. Note that in z^* we have $f(z^*) = 0$, $\nabla_z f(z^*)$ is nonsingular. We set $\beta = ||\nabla_z f(z^*)^{-1}||$. There exists an open neighbourhood of z^* in which z^* is the unique solution of $f(z) = 0$, hence z^* is the locally unique Kuhn-Tucker point of (4.21). The following shortened notation will be used from now on: $\nabla_z d(z_1,z_2) := \nabla_z d(z_1,z)(z_2)$.

THEOREM 4.2 (Robinson (1972))

If all problem functions are twice continuously differentiable in an open neighbourhood $O(x^*)$ of x^*, there exist constants $r > 0$ and $M > 0$ such that z^* is the unique solution of $f(z) = 0$ in the closed ball $\bar{B}(z^*, \frac{1}{2}r)$ with radius $\frac{1}{2}r$ about z^*. Moreover for any $z_1, z_2 \in B(z^*,r)$ with μ_i as Lagrange multipliers of the reduced problem, we have

(i) $\quad ||\nabla_z d(z_1,z_2) - \nabla_z d(z^*,z^*)|| < (2\beta)^{-1}$

(ii) $\quad ||f(z_2) - d(z_1,z_2)|| \le M||z_1 - z_2||^2$

(iii) $\quad c_i(x^*) > 0$ implies $Lc_i(x_1,x_2) > 0$

(iv) $\quad u_i^* > 0$ implies $\mu_i > 0$

□

This theorem of Robinson will be applied in the comparison below of Kuhn-Tucker points of the original problem and of equality constrained reduced problems. It turns out to be a fundamental theorem. First we mention that a simplified problem is obtained from (4.21) if the constraint set is reduced to a set of equality constraints $c_i(x)$ whose index i belongs to a currently defined active set $I(z_k)$. Usually this active set $I(z_k)$ consists of all equality constraints, the currently binding inequality constraints and the inequality constraints that are expected to be binding at the next iteration point.

For example the algorithm to be described now features an active set $I(z_k)$ which consists of all equality constraints, all linear constraints and a selection of nonlinear inequality constraints, containing at least the binding constraints. This means that $i \notin I(z_k)$ corresponds with $c_i(x_k) > 0$. Thus we consider the reduced problem

$$(4.40) \quad \begin{cases} \min F(x) \\ \text{subject to} \\ c_i(x) = 0 \quad \text{for all } i \in I(z_k) \end{cases}$$

The first order Kuhn-Tucker conditions of (4.40) are:

$$(4.41) \qquad \nabla_x (F(x) + \sum_{i \in I(z_k)} v_i c_i(x)) = 0$$

(4.42) $\qquad c_i(x) = 0 \qquad i \in I(z_k)$

where v_i, $i \in I(z_k)$ is the Lagrangian multiplier corresponding to $c_i(x) = 0$. The equality constrained problem (4.40) can be solved using the reduced problem

(4.43) $$\begin{cases} \min F(x) + \phi(x_k,x) \\ \text{subject to} \\ Lc_i(x_k,x) = 0 \quad \text{for all } i \in I(z_k) \end{cases}$$

with the following first order Kuhn-Tucker conditions:

(4.44) $$\nabla_x(F(x) + \phi(x_k,x) - \sum_{i \in I(z_k)} \nu_{k,i} Lc_i(x_k,x)) = 0$$

(4.45) $$Lc_i(x_k,x) = 0 \qquad i \in I(z_k)$$

Analogous with the definition of $S(z_k)$ as the collection of Kuhn-Tucker points of (4.29), we define $S(z_k, I(z_k))$ to be the collection of all solutions of the Kuhn-Tucker conditions of problem (4.43).
If $I(z_k)$ contains all equality constraints and all inequality constraints with positive estimated Lagrange multiplier (if x_k is close enough to x^* this estimate has the correct sign), then conditions (4.41), (4.42) arise from (4.23) - (4.27). The estimated multiplier can have the wrong sign if x_k is remote from x^*. For the linearized, reduced problems similar relations apply for the conditions (4.44), (4.45) as compared to (4.31) - (4.35).
Our next point of interest is to find relations between the solution sets $S(z_k)$ and $S(z_k, I(z_k))$. The next two lemma's contain mutual inclusion relations.

LEMMA 4.3

If $I(z_k) := \{i \mid c_i(x^*) = 0,\ i = 1, \ldots, m\}$ with $\mu_{k,i} = 0$ for all $i \notin I(z_k)$ at a point $z_k \in B(z^*,r)$ with $S(z_k, I(z_k)) \subset B(z^*,r)$ and strict complementary slackness in (4.24), then $S(z_k,I(z_k)) \subset S(z_k)$.

Proof:

By definition $z_{k+1} \in S(z_k, I(z_k))$ satisfies (4.44), (4.45). These equations can be extended to (4.31), (4.32), (4.33) using $\mu_{k,i} = 0$

for all $i \notin I(z_k)$. To prove (4.34) we remark that, using (4.45), we only need to consider indices $i \in \{1, \ldots, p\} - I(z_k)$ which correspond to inactive inequality constraints. Then from $z_k, z_{k+1} \in B(z^*,r)$, $c_i(x^*) > 0$ and theorem 4.2 (iii) we know that $Lc_i(x_k,x) > 0$, hence (4.34) is satisfied.

Finally (4.35) is true for all $i \notin I(z_k)$ by the definition $\mu_{k,i} = 0$ for these i. For $i \in I(z_k)$ equation (4.45) gives $Lc_i(x_k,x) = 0$, which implies $c_i(x^*) = 0$ ($c_i(x^*) < 0$ violates the K.T. conditions at x^* and $c_i(x^*) > 0$ contradicts $Lc_i(x_k,x) = 0$ by theorem 4.2 (iii)).

But this means $u_i^* > 0$ (strict complementary slackness in (4.24)) which again implies $\mu_{k,i} > 0$ (theorem 4.2 (iv)). Thus (4.35) is proved and hence the lemma.

□

LEMMA 4.4

If $I(z_k) := \{i \mid c_i(x^*) = 0,\ i = 1, \ldots, m\}$ for a point $z_k \in B(z^*,r)$ with $S(z_k, I(z_k)) \subset B(z^*,r)$ and strict complementary slackness in (4.24) and (4.32) , then $S(z_k) \subset S(z_k, I(z_k))$.

Proof:

We have to prove (4.44), (4.45) for z_{k+1} satisfying (4.31) - (4.35).
For $1 \le i \le p$ and $i \notin I(z_k)$ we have

$c_i(x^*) > 0$	(definition of $I(z_k)$)
so that $Lc_i(x_k, x_{k+1}) > 0$	(theorem 4.2 (iii))
and $\mu_{k,i} = 0$	(strict complementary slackness in (4.32)).

Then (4.44) follows from (4.30), (4.31) and the substitution $\mu_{k,i} = 0$ for nonbinding inequality constraints.

As (4.45) obviously applies for equality constraints, we only need to consider indices $1 \le i \le p$ with $i \in I(z_k)$. Then

$c_i(x^*) = 0$	(definition of $I(z_k)$)
which yields $u_i^* > 0$	(strict complementary slackness in (4.24)).
but then $\mu_{k,i} > 0$	(theorem 4.2 (iv))
and $Lc_i(x_k,x_{k+1}) = 0$	(strict complementary slackness in (4.32)

which completes the proof of (4.45) and the proof of the lemma.

□

When comparing lemmas 4.3 and 4.4 we see that beside the strict complementary slackness the definition of the correct set $I(z_k)$ is of importance. The lemmas can be summarized in a corollary:

COROLLARY

$S(z_k) = S(z_k, I(z_k))$ if $z_k \in B(z^*,r)$, $S(z_k, I(z^*)) \subset B(z^*,r)$ under strict complementarity in (4.24) and (4.32) and
$I(z_k) := \{i \mid c_i(x^*) = 0,\ i = 1, \ldots, m\}$.

Proof:

The proof is obvious from the lemmas 4.3 and 4.4.

□

How stringent or unrealistic are these conditions?
The required strict complementary slackness means that there should be no weakly active constraints, a condition imposed on the problem considered, that can be met (in case of violation) by suitably perturbing weakly active constraints, though this will generally give very small values of r. Further $z_k \in B(z^*,r)$ can be realised by a preceding so-called phase I procedure which yields a starting point close enough to z^*. In practice the correctness of $I(z_k)$ is usually obtained after a few iterations, unless zigzagging occurs. This means, given a phase I procedure, that the conditions of lemmas 4.3 and 4.4 are not unrealistic.

IV.4. Convergence of sequences of Kuhn-Tucker points

In section IV.3 the mappings $f(z)$ and $d(z_k,z)$ from E^{n+m} to E^{n+m} were introduced. A further investigation of the algorithms considered requires properties of the operators $f(z)$ and $d(z_k,z)$ as presented in Ortega and Rheinboldt (1970) and Kantorovic and Akilov (1964). In these statements $||z||$ will denote the Euclidian norm. It is easy to see, however, that the results remain valid for any norm on $\mathbb{R}^{n+m}$. Note that though we use point-to-set maps below, the resulting implementations will define uniquely the next iteration point. Let X_0 and X be subsets of E^{n+m} with $X_0 \subset X$, where X is assumed to be bounded.

DEFINITION

A mapping $A : X \subset E^{n+m} \to E^{n+m}$ is called *nonexpansive* on a set $X_0 \subset X$ if $A(z)$ is unique for all $z \in X_0$ and

(4.46) $$||A(z_1) - A(z_2)|| \le ||z_1 - z_2|| \quad \text{for all } z_1, z_2 \in X_0$$

and *strictly nonexpansive* on X_0 if strict inequality holds in (4.46) whenever $z_1 \neq z_2$.

This definition means that a nonexpansive mapping on X is Lipschitz-continuous on X_0.
In general A may be a nonlinear operator on $X \subset E^{n+m}$. Examples are f(z) and $d(z_k, z)$, with the domain X being the feasible regions $L \cap NL$ and $L \cap LNL(x_k)$ respectively.
Another example of such an operator on E^n is provided by the algorithm given by (4.15) which, starting from $z_k = (z_k, u_k, v_k)$ defines the next iteration point z_{k+1} as a certain Kuhn-Tucker point of the reduced problem. In this example the uniqueness of z_{k+1} is established by an additional requirement, which makes the mapping *deterministic*. Removing this additional selection rule yields a *non-deterministic* mapping, which therefore does not necessarily determine z_{k+1} uniquely.
Special points of interest are *fixed points* $z^* \in X$ of A which are defined by

(4.47) $$z^* \in A(z^*)$$

which means

(4.48) $$z^* = A(z^*)$$

for a deterministic mapping.

LEMMA 4.5 (Banach)

If $A: X \subset E^{n+m} \to E^{n+m}$ is deterministic and strictly nonexpansive on $X_0 \subset X$ then A has at most one fixed point.

Proof:

Let us assume that there exist two distinct fixed points z_1^* and $z_2^* \in X_0$.
Then:

$$||z_1^* - z_2^*|| = ||A(z_1^*) - A(z_2^*)|| < ||z_1^* - z_2^*||$$

which is a contradiction.

□

However, Ortega and Rheinboldt (1970) show by a counterexample that strict nonexpansivity is not sufficient to guarantee the existence of a fixed point.
That is why (4.46) is strengthened.

DEFINITION

A mapping $A : X \subset E^{n+m} \to E^{n+m}$ is *contractive* on a set $X_0 \subset X$ if there exist an α, $0 < \alpha < 1$ such that

(4.49) $$||A(z_1) - A(z_2)|| \leq \alpha \, ||z_1 - z_2|| \text{ for all } z_1, z_2 \in X_0$$

From (4.46) and (4.48) we see that a contractive mapping is strictly nonexpansive and, as a consequence, Lipschitz continuous with at most one fixed point. The existence of a fixed point of a contractive mapping is given by the perhaps best known fixed point theorem:

THEOREM 4.3 (Banach, contraction mapping theorem)

If $A : X \subset E^{n+m} \to E^{n+m}$ is contractive on a closed set $X_0 \subset X$ and $A(X_0) \subset X_0$ then A has a unique fixed point in X_0.

Proof:

Choose $z_0 \in X_0$ arbitrarily and define the sequence $\{z_k\}$ by
$z_k = A(z_{k-1})$, $k = 1, 2, \ldots.$.
From $A(X_0) \subset X_0$ we know that $z_k \in X_0$ for all $k = 1, 2, \ldots.$
Further there exists an $0 < \alpha < 1$ such that

$$||z_{k+1} - z_k|| = ||A(z_k) - A(z_{k-1})|| \leq \alpha||z_k - z_{k-1}||,$$ which

yields

(4.50) $$||z_{k+p} - z_k|| \leq \sum_{i=1}^{p} ||z_{k+i} - z_{k+i-1}|| \leq (\alpha^{p-1}+\ldots+1)||z_{k+1} - z_k|| \leq \frac{\alpha^k}{1-\alpha} ||z_1 - z_0||$$

From (4.50) we see that $\{z_k\}$ is a Cauchy sequence in the closed set X_0, so it has a limit $z^* \in X_0$ with image $A(z^*)$. Then the continuity of A implies $\lim_{k\to\infty} A(z_k) = Az^*$

thus $z^* = Az^*$: z^* is a fixed point of A.

□

Further generalizations of the contraction mapping theorem and conditions to ensure $A(X_0) \subset X_0$ can be found in e.g., Ortega and Rheinbold (1970), Istratescu (Fixed Point Theory, an Introduction,1980,to appear).

The properties derived above are necessary to prove a proposition which originates from Robinson (1972) and which guarantees under weak conditions that if the iteration process given by (4.15) is initiated in a point z_k close enough to a Kuhn-Tucker point z^* of (4.21) then the reduced problem (4.29) defined in z_k yields a unique Kuhn-Tucker point z_{k+1} close to z_k, hence close to z^*. An extension of this proposition to a reduction method which applies purely equality constrained reduced problems will be used to prove a theorem on the convergence and the rate of convergence of this reduction method. The proof is a slight modification of Robinson's proof.

PROPOSITION 4.3

If both (4.21) and (4.29) satisfy strict complementary slackness, the problemfunctions are twice continuously differentiable, $z_k \in B(z^*, \tfrac{1}{2}r)$ such that $4\beta||f(z_k)|| \leq r$ and z^* is a Kuhn-Tucker point satisfying the second order sufficiency conditions for (4.21) then there exists a point $z_{k+1} \in B(z_k, \tfrac{1}{2}r)$ such that z_{k+1} is the unique Kuhn-Tucker triple of (4.29) defined at z_k such that $||z_{k+1} - z_k|| \leq 2\beta||f(z_k)||$

Proof:

First we note that $\bar{B}(z_k,\tfrac{1}{2}r) \subset B(z^*,r)$ as $z_k \in B(z^*,\tfrac{1}{2}r)$.

Then theorem 4.2(i) gives

$$||\nabla_z d(z_k,z) - \nabla_z d(z_k,z_k)|| < (2\beta)^{-1}$$

for all $z \in \bar{B}(z_k,\tfrac{1}{2}r)$.

Now we define a mapping T on $\bar{B}(z_k, \frac{1}{2}r)$ by

$$T(z) := z - \nabla_z f(z^*)^{-1} d(z_k, z).$$

Then T is differentiable and since $\nabla_z f(z) = \nabla_z d(z,z)$ (corollary 1 of proposition 4.2) for any z we have

$$\nabla_z T(z) = I - \nabla_z d(z^*, z^*)^{-1} \nabla_z d(z_k, z)$$

$$= \nabla_z d(z^*, z^*)^{-1} [\nabla_z d(z^*, z^*) - \nabla_z d(z_k, z)].$$

Hence

$$||\nabla_z T(z)|| \le ||\nabla_z d(z^*, z^*)^{-1}|| . ||\nabla_z d(z_k, z_k) - \nabla_z d(z_k, z)||$$

$< \beta . (2\beta)^{-1} = \frac{1}{2}$. (recall that by definition $\beta = ||\nabla_z d(z^*, z^*)^{-1}||$)

Using the mean value theorem 3.2.3 of Ortega and Rheinboldt (1970) this implies

(4.51) $$||T(z_1) - T(z_2)|| \le \tfrac{1}{2}||z_1 - z_2|| \text{ for all } z_1, z_2 \in \bar{B}(z_k, \tfrac{1}{2}r).$$

hence T is a contraction on $\bar{B}(z_k, \frac{1}{2}r)$.

From $||T(z_k) - z_k|| = ||-\nabla_z f(x^*)^{-1} d(z_k, z_k)||$

$\le \beta \; ||f(z_k)||$

$\le \beta \cdot \frac{r}{4\beta} = \frac{1}{4}r$

and (4.51) we conclude that

$$||T(z) - z_k|| \le ||T(z) - T(z_k)|| + ||T(z_k) - z_k||$$

$\le \frac{1}{2} \; ||z - z_k|| + \frac{1}{4}r$

$\le \frac{1}{4}r + \frac{1}{4}r = \frac{1}{2}r$ for all $z \in \bar{B}(z_k, \frac{1}{2}r)$,

which means that $T(z) \in \bar{B}(z_k, \frac{1}{2}r)$ for all $z \in \bar{B}(z_k, \frac{1}{2}r)$.

Now all requirements of the contraction mapping theorem are met, so T has a unique fixed point $z_{k+1} \in \bar{B}(z_k, \frac{1}{2}r)$.
For z_{k+1} we have:

$$||z_{k+1} - z_k|| = ||T(z_k) - z_k|| \le ||-\nabla_z f(z^*)^{-1} d(z_k, z_k)|| \le$$

$$\le \beta||f(z_k)|| \le 2\beta||f(z_k)||$$

It remains to be proved that z_{k+1} is the unique Kuhn-Tucker point of (4.29) in $\bar{B}(z_k, \frac{1}{2}r)$.
Following lemma 4.2 we have to show that (4.34) and (4.35) are met for all i, and $d(z_k, z_{k+1}) = 0$.
As there is a one to one correspondece between the fixed points of $T(z)$ and the zeros of $d(z_k, z)$ uniqueness is allright and it remains to verify that z_{k+1} satisfies (4.34) and (4.35). Consider $c_i(x)$.
Then, as x^* satisfies the Kuhn-Tucker conditions, either
$c_i(x^*) > 0$ or $c_i(x^*) = 0$.

If $c_i(x^*) > 0$,

we know

$Lc_i(x_k, x_{k+1}) > 0$ (theorem 4.2(iii))

and

$u_i = \mu_{k,i}$ (strict complementary slackness)

If $c_i(x^*) = 0$

we know

$u_i^* > 0$ (strict complementary slackness)

and

$\mu_{k,i} > 0$ (theorem 4.2(iv)

which yields

$Lc_i(x_k, x_{k+1}) = 0$ (strict complementary slackness)

Hence $\mu_{k,i} \geq 0$ and $Lc_i(x_k,x_{k+1}) \geq 0$ for all i and the lemma is proved.

□

This proposition guarantees the existence in $\bar{B}(z_k,\frac{1}{2}r)$ of a unique Kuhn-Tucker point of the reduced problem (4.29). The next proposition extends this result in that it shows that under certain conditions, this point $z_{k+1} \in S(z_k,I(z^*))$: to find z_{k+1} it suffices to solve a smaller reduced problem, with only equality constraints. Note that propositions 4.3 and 4.4 only concern the case that $\phi(x_k,x)$ is defined by (4.6).

PROPOSITION 4.4

Let z^* be a regular point, satisfying the sufficient 2nd order Kuhn-Tucker conditions of (4.21), under strict complementary slackness in (4.24) and (4.32). If $z_k \in B(z^*, \frac{1}{2}r)$ with $4\beta||f(z_k)|| \leq r$ and $I(z_k) = \{i \mid c_i(x^*) = 0,\ i = 1,\dots,m\}$, then there exists a unique $z_{k+1} \in B(z_k, \frac{1}{2}r)$ with $z_{k+1} \in S(z_k) = S(z_k, I(z^*))$ and $||z_{k+1} - z_k|| \leq 2\beta||f(z_k)||$.

Proof:

Proposition 4.3 implies the existence of a unique $z_{k+1} \in B(z_k, \frac{1}{2}r)$ such that $z_{k+1} \in S(z_k)$ and $||z_{k+1} - z_k|| \leq 2\beta||f(z_k)||$.
Under the conditions z_k, $z_{k+1} \in B(z^*,r)$ and strict complementary slackness, we conclude from the proofs of lemmas 4.3, 4.4 that $S(z_k) = S(z_k, I(z^*))$ which gives the lemma.

□

The next propositions 4.5, 4.6 demonstrate that under the same assumptions at z_k and assuming that $I(z^*)$ is known it suffices to solve the equality constrained reduced problem (4.43) with $I(z_k) = I(z^*)$ to obtain the next iteration point z_{k+1}. Thus the question remains how to recognise the set $I(z^*) = \{i \mid c_i(x^*) = 0,\ i = 1,\dots,m\}$ at $z_k \neq z^*$? A more thorough discussion on the determination of the correct active set is postponed to section IV.5.
For a given arbitrary index set I_0, the following result can be proven, where this time $\phi(x_k,x)$ can be any function which is continuously differentiable and satisfies (4.14). The proposition is an extension of a theorem of Bräuniger (1977).

PROPOSITION 4.5

If all problem functions and $\phi(x_k, x)$ are continuously differentiable and the sequence $\{\bar{z}_k\}$converges to $\bar{z}$ and:
$\lim_{k\to\infty}\left\{\inf_{z_{k+1}\in S(\bar{z}_k, I_0)} ||z_{k+1} - \bar{z}_{k+1}||\right\} = 0$ then $\bar{z}$ is a Kuhn-Tucker point of the reduced equality constrained problem (4.40) with $I(\bar{z}_k) = I_0$.

Proof:

First note that the proposition is formulated in terms of an arbitrary sequence $\{\bar{z}_k\}$ with $\lim_{k\to\infty} \bar{z}_k = \bar{z}$. Let z_{k+1} be the element of $S(\bar{z}_k, I_0)$ defined above which realizes $\inf||z_{k+1}-\bar{z}_{k+1}||$ for $z_{k+1} \in S(\bar{z}_k, I_0)$. Then $\lim_{k\to\infty} \bar{z}_k = \bar{z}$ and $\lim_{k\to\infty}||z_{k+1}-\bar{z}_{k+1}|| = 0$ yields $\lim_{k\to\infty} z_k = \bar{z}$. Then the Kuhn-Tucker conditions (4.44) and (4.45) yield for $I(\bar{z}_k) = I_0$:

(4.52) $$Lc_i(\bar{x}_k, x) = 0 \text{ for all } i \in I_0$$

and

(4.53) $$\nabla_x(F(x) + \phi(\bar{x}_k, x) + \sum_{i\in I_0} \nu_i Lc_i(\bar{x}_k, x)) = 0$$

Substituting $z = z_{k+1}$, and using (4.14), (4.52) and (4.53), the continuity of all functions and their gradients and the convergence of the sequence $\{z_k\}$ yields:

(4.54) $$c_i(\bar{x}) = 0 \text{ for all } i \in I_0$$

and

(4.55) $$\nabla_x F(\bar{x}) = \sum_{i\in I_0} \bar{\nu}_i \nabla_x c_i(\bar{x})$$

which proves the proposition.

□

REMARK

Bräunigers theorem concerns the function $\phi(x_k,x)$ as defined in (4.6). For the special case that $\phi(x_k,x)$ is defined by (4.6) propositions 4.4 and 4.5 uniquely define a sequence converging to the point z*. This is proved in the next proposition.

PROPOSITION 4.6

If the conditions of propositions 4.4 and 4.5 are met and $\lim_{k\to\infty} z_k = z^*$, a solution of (4.21), then z* is a Kuhn-Tucker point of the problem (4.40) with $I(z_k) = I(z^*)$ viz. min F(x) subject to $c_i(x) = 0$ for all $i \in I(z^*)$.

Proof:

Starting from a z_0 which meets the requirements of proposition 4.4, the sequence $\{z_k\}$ with $z_{k+1} = S(z_k,I(z^*))$ is uniquely determined and satisfies $||S(z_k,I(z^*)) - z_{k+1}|| = 0$. Hence proposition 4.5 canbe applied and we are done.

□

The meaning of this proposition is that though in practice z* and I(z*) are unknown, the algorithm still generates sequences $\{z_k\}$ and $\{I(z_k)\}$ which can be proved to converge to a Kuhn-Tucker point z* of (4.21) and I(z*) respectively.
To prove this a more general approach by means of point to set maps will be followed. It will turn out to be possible to prove that under suitable conditions the sequence of iteration points generated by the algorithm converges to a fixed point of a point-to-set mapping which characterizes the algorithm. This point will turn out to be a Kuhn-Tucker point of the considered problem. This theory will be followed by a proposition on Kuhn-Tucker points of the original and reduced problems. The ultimate convergence properties are considered in section IV.6.

This approach to obtain an algorithm viapoint-to-set maps originates with Zangwill (1969). Later Polak (1971), Luenberger (1973), Meyer (1979) and others followed Zangwills approach in describing algorithms and investigating their properties.

We need some definitions and results which are summarized in Meyer (1979).
We assume $z \in E^{n+m}$ and that A acts on E^{n+m}.
Let X and Y be subsets of E^{n+m}.

DEFINITION

A *point-to-set map* A from X to Y is a map which associates a subset $A(z) \subset Y$ with each $z \in X$.

We shall assume that $X \subset E^{n+m}$ is closed, for instance $X = \bar{B}(z^*, r)$: the closed ball with radius r around the Kuhn-Tucker point z^* of (4.21). Then given a point $z_0 \in X$, and $X = Y$, an *algorithm* is defined by any scheme of the following type:

(4.56) Step 0. Set $k = 0$
Step 1. Pick a point $z_{k+1} \in A(z_k)$
Step 2. Set $k = k+1$, and go to step 1.

In this scheme *no stopping rule* is included (only infinite sequences are generated). This algorithm (4.56) is called *non-deterministic* (z_{k+1} may be chosen arbitrarily in $A(z_k)$) and *autonomous* (A is independent of k).

DEFINITION

The *characteristic set C* of algorithm (4.56) is the set of all $z \in X$ such that scheme (4.56) admits a sequence $\{z_k\}$ with $z_k = z$ for all k larger than a certain number K.
It is easy to verify that

$$C = \{z \in X \mid z \in A(z)\} \tag{4.57}$$

DEFINITION

$z \in X$ is a *periodic point* of A of period p if

(i) $z \in A^p(z)$
(ii) $z \notin A^q(z)$ for all $q = 1, 2, \ldots, p-1$

From these definitions we directly see that the characteristic set C of algorithm (4.56) consists of all periodic points of A of

period 1: the *fixed points* of A.
As the algorithm generates infinite sequences we can define P as the set of all limit points of all convergent sequences which can be generated by it, and Q as the set of all cluster points of all sequences which can be generated by the algorithm.

LEMMA 4.6

$$C \subseteq P \subseteq Q.$$

Proof:

By definition, we have $P \subseteq Q$. Furthermore $C \subseteq P$ as for each $z_0 \in C$ the algorithm can generate the infinite sequence $\{z_k\}$ with $z_k = z_0$ for all k.

□

A further investigation of the asymptotic properties of algorithms requires a definition of continuity of the mapping A:

DEFINITION

The map A is *upper semicontinuous* (u.s.c.) at a point $z_0 \in X$ if for every neighbourhood $N(A(z_0))$ of $A(z_0)$ there exists a neighbourhood $N(z_0)$ of z_0 so that $A(z_1) \subset N(A(z_0))$ for every $z_1 \in N(z_0) \cap X$.

We say that A is u.s.c. on a subset $S \subset X$ if A is u.s.c. at every point in S.

DEFINITION (Zangwill, 1969)

The map A is *closed* at a point $z_0 \in X$ if the fact that $\{z_k\}$ converges to z_0 (all $z_k \in X$) and $\{y_k\}$ converges to y_0 with $y_k \in A(z_k)$ for all k, implies that $y_0 \in A(z_0)$.

We call A closed on a subset $S \subset X$ if A is closed at every point $z \in S$.

DEFINITION

The map A is *compact valued* on X if A(z) is compact for every $z \in X$.

In, e.g., Meyer (1979) there are some examples that closedness and upper semicontinuity are not equivalent. For instance he proved the following useful lemma:

LEMMA 4.7

If X is bounded and A is closed on X, then A is u.s.c. and compact valued on X.

□

With the aid of these definitions the following proposition and a corollary can be stated:

PROPOSITION 4.7

Suppose that A is u.s.c. and compact valued on X, and let $\{z_k\}$ be a specific sequence generated by the algorithm (4.56). If z^* is a cluster point of $\{z_k\}$, then for every p = 1, 2, ... the set $A^p(z^*)$ contains a cluster point of $\{z_k\}$.

□

COROLLARY

If A is u.s.c. and compact valued on X, then P = C : if a sequence generated by algorithm (4.56) converges, it converges to a fixed point of A.

□

After this corollary the next point of interest is to state a condition under which the desired convergence is guaranteed. This condition concerns the asymptotic behaviour of the algorithm.

DEFINITION

A sequence $\{z_k\}$ is *asymptotically regular* if $\lim_{k\to\infty} ||z_{k+1} - z_k|| = 0$.

DEFINITION

An algorithm is *asymptotically regular* if every infinite sequence it can generate is asymptotically regular.

PROPOSITION 4.8

If the map A is u.s.c. and compact valued on X-D, where $D \subset X$, and algorithm (4.56) is asymptotically regular, then $C \subset Q \subset C \cup D$. □

COROLLARY 1

> If A is u.s.c., compact valued on X - C and algorithm (4.56) is asymptotically regular, then P = Q = C. □

Ostrowski (1966) proved that an as.regular sequence on a bounded set X has either a *unique* cluster point or a *continuum* of cluster points. Hence

COROLLARY 2

> Suppose A u.s.c. and compact valued on X - C, X is bounded, algorithm (4.56) is asymptotically regular and C contains at most a countable number of points, then <u>every</u> sequence generated by (4.56) converges to a point in C : a fixed point of algorithm (4.56). □

In order to apply the above results to the algorithms considered, we have to make sure that they use mappings A which are u.s.c. and compact valued on X - C.

Then according to lemma 4.7 it suffices to verify whether X-C is bounded and A is closed on X - C.

As X - C $\subset$ X, which is assumed to be bounded, it only remains to investigate whether A is closed on X-C. Instead of investigating the general algorithm, given by (4.15) with a currently defined active set $I(z_k)$, we focus on the special case in which the final, correct active set $I(z^*) = I^*$ has already been found and $I(z_k) = I(z^*)$ is used from now on. This is the final stage of the general algorithm, which characterizes the convergence properties. The following lemma,cf. e.g., Meyer (1979) will be used in the next theorem. It guarantees for any cluster point of a sequence $\{z_k\}$ the existence of a subsequence converging to this cluster point.

LEMMA 4.8

> Let q be the set of all cluster points of the sequence $\{z_k\}$, generated by an algorithm (4.56), A u.s.c. and compact valued on X. If q is nonempty and bounded, then given any neighbourhood N(q) of q, there exists an index k, depending on N(q) such that $z_i \in N(q)$ for all $i \geq k$. □

Application of this lemma to a suitable region X such that $q = \{z^*\}$ yields the required subsequence.

THEOREM 4.4

Let the sequence $\{z_k\}$ be defined by (4.56) with $z_{k+1} = A(z_k)$ where A is given by (4.43) with $I(z_k) = I^*$. Let X be bounded, and closed with all problem functions and $\phi(x_k,x)$ continuously differentiable while (4.14) is satisfied. Then every cluster point z^* of $\{z_k\}$ is a fixed point of A.

Proof:

The infinite sequence $\{z_k\}$, thus generated by the feasible point algorithm (4.56) has at least one limit point $z^* \in X$. Then there exists a subsequence converging to z^*, which we shall denote again by $\{z_k\}$. It follows from the definition of z_{k+1} that (4.44) and (4.45) are satisfied by z_{k+1}. The continuity of the gradients, (4.14), (4.44), (4.45) and $\lim_{k\to\infty} z_k = z^*$ then imply that

$$Lc_i(x^*,x^*) = 0 \quad \text{for all } i \in I(z^*)$$

and

$$\nabla_x(F(x^*) - \sum_{i\in I(z^*)} \nu_i^* Lc_i(x^*,x^*)) = 0$$

which means that z^* satisfies the Kuhn-Tucker conditions of the reduced problem defined at z^*, so $z^* \in A(z^*)$: z^* is a fixed point of A.

□

PROPOSITION 4.9

The deterministic algorithm defined in theorem 4.4 is continuous at every $z \in X$, hence is closed on X.

Proof:

The continuity follows directly from the continuity assumptions on the problem functions. The closedness is an immediate consequence of this continuity.

□

COROLLARY

The deterministic algorithm defined in theorem 4.4 is u.s.c. and compact valued on X.

Proof:

This follows from the application of lemma 4.7.

□

As a consequence, all above derived statements for u.s.c. and compact valued mappings apply to the algorithms developed, for instance those using reduced problem formulations as (4.15), (4.29) and (4.43) with $I(z_k) = I^*$.
We make special mention of the corollary of proposition 4.7:
<u>If a sequence, generated by the algorithm (4.56) converges, it converges to a fixed point of</u> A.
and corollary 2 of proposition 4.8:
<u>If the applied mapping</u> A <u>is u.s.c. and compact valued on</u> X - C, X <u>is bounded, algorithm (4.56) is asymptotically regular and</u> C <u>contains at most a countable number of points, then every sequence generated by (4.56) converges to a point in</u> C<u>: a fixed point of the mapping</u> A.

Compared with theorem 4.3 we see that the convergence to a fixed point is now established under conditions as asymptotic regularity, upper-semi-continuity and compact valued, instead of the condition of contractiveness. The extra requirement that C should be countable will hopefully be met in most real-life problems.
Let A be defined by (4.15) and let z^* be a fixed point of the mapping A. Again we assume strict complementary slackness in all occuring Kuhn-Tucker conditions. Then we can prove the following theorem, following Rosen (1977).

THEOREM 4.5

A fixed point z^* of the mapping A is a Kuhn-Tucker point of (4.21)

Proof:

As z^* is a fixed point of A, we know that $A(z^*) = z^*$, $d(z^*,z^*) = 0$ while (4.34) and (4.35) are satisfied. (z^* is a first order Kuhn-Tucker point of (4.29) with $z_k = z^*$, hence lemma 4.2 can be applied). From (4.37) we see that $f(z^*) = d(z^*,z^*) = 0$. Further (4.26) is satisfied, as $c_i^*(x^*) < 0$ for at least one $i \in \{1, \ldots, p\}$ would yield $Lc_i(x^*,x^*) = c_i(x^*) < 0$ which contradicts (4.34). Finally (4.27) is met as can be seen from $u_i^* = \mu_i^* \geq 0$ (use (4.35)) for

$i = 1, \ldots, p$. Hence lemma 4.1 applies and z^* is a first order Kuhn-Tucker point of (4.21).

□

In the corollary of lemmas 4.3,4.4 we saw that $S(z_k) = S(z_k, I(z^*))$, which meant that if $I(z^*)$ is known at z_k, it suffices to solve the equality constrained reduced problem (4.43) with $I(z_k) = I(z^*)$. In that situation we can formulate an analogous theorem for the resulting equality constrained reduction method.

THEOREM 4.6

If z^* is a fixed point of the mapping A defined by the application of the reduced problem (4.43) with the correct active set $I(z_k) = I(z^*)$ in the algorithmic scheme given by (4.56), then z^* is a Kuhn-Tucker point of (4.21).

Proof:

The proof follows from combination of lemmas 4.3,4.4 and theorem 4.5, using strict complementary slackness in the Kuhn-Tucker conditions.

□

An important question which should be considered now is under which conditions the correct final active set $I(z^*)$ can be defined at $z_k \neq z^*$. Possible *active set strategies* to answer this question are treated in the next section.

IV.5 Active set strategies

In the preceding chapter IV.4 we derived conditions under which $S(z_k) = S(z_k, I(z^*))$ which means that the first-order Kuhn-Tucker points of the original problem (4.21) are first-order Kuhn-Tucker points of the pure equality constrained reduced problem (4.40) with $I(z_k) = I(z^*)$ and vice versa.

As $I(z^*)$ is not usually known beforehand, we are looking for criteria by which an algorithm can recognize $I(z^*)$. That is why a current active set $I(z_k)$ is defined, which is intended to contain the constraints that are most relevant, at least locally. Adjustment of $I(z_k)$ should be accomplished using the information gathered on the constraints, such as their current status: binding ($c_i(x_k) = 0$, $i = 1, \ldots, m$), violated ($c_i(x_k) < 0$ for $i \in \{1, \ldots, p\}$ or $c_i(x_k) \neq 0$ for $i \in \{p+1, \ldots, m\}$ or satisfied ($c_i(x_k) \geq 0$ or $c_i(x_k) = 0$ for the respective cases). Constraints will be added to the active set or dropped from this set, with the ultimate goal that $I(z_k) = I(z^*)$ holds for all k greater than or equal to some number K. By the application of such an 'active set strategy' all reduced problems are equality constrained problems in which all constraints with index $i \notin I(z_k)$ are deleted. Hence the reduced problems are simplifications of the original problem. Obviously the above approach is equally valid for both the original problem (4.21) and the linearized problem (4.29).

The convergence of the sequence iteration points $\{z_k\}$ is expected to be accelerated if $I(z^*)$ can be recognized in an early stage of the iteration process. In this section relations between $I(z_k)$ and $I(z^*)$ are derived and a lemma by Bräuniger (1977), which states $I(z_k) = I(z^*)$ for suitable z_k and his proof of this lemma are discussed. As a consequence of this lemma and the proposed phase I step, section IV.7 can deal with the problem of how to link phase I and phase II such that phase I produces a good starting point and a good initial active set for phase II.

The next results rely heavily on Robinson's theorem 4.2, parts (iii) and (iv) in which the fact is expressed that as soon as $z_k \in B(z^*,r)$ we have at all points $z_{k+1} \in B(z^*,r)$:

(iii) if $c_i(x)$ is passive at x^*, then its linearization at x_k is passive as well.

(iv) if $c_i(x)$ is active at x^* then its linearization at x_k is active as well.

But first we shall consider the decision to be taken in step 2 of the algorithm: if $S(z_k)$ contains more than one point, choose the Kuhn-Tucker point of (4.15) which is closest to z_k. This means that z_{k+1} solves

$$(4.58) \qquad \min_{z \in S(z_k)} ||z-z_k||$$

The points z_{k+1} satisfying (4.58) were characterized by Bräuniger as summarized below in (4.59(i)-(iii)). For the sake of simplicity of notation, the current iteration point z_k and its successor z_{k+1} are denoted by $z = (x,u,v)$ and $\bar{z} = (\bar{x},\bar{u},\bar{v})$ respectively.

Then $\bar{z}$, defined by (4.58) should be given by one of the following possibilities:

(4.59(i)) (i) $\bar{z} = z$ hence z is a solution of (4.21)

(4.59(ii)) (ii) $\bar{z} = \begin{bmatrix} x \\ \bar{u} \\ v \end{bmatrix}$ with $\bar{u}_j = \begin{cases} u_j & j \neq i \\ 0 & j = i \end{cases}$

for some $i \in \{1, \ldots, p\}$. This means a pure dual step, as the only difference between $\bar{z}$ and z is the value of the Lagrange multiplier of the i-th inequality constraint.

(4.59(iii)) (iii) $\bar{z} = \begin{bmatrix} \bar{x} \\ u \\ v \end{bmatrix}$ with $\bar{x} = \left(x - \dfrac{c_j(x)}{||\nabla c_j(x)||^2} \nabla c_j(x) \right)$

for some $j \in \{1, \ldots, p\}$. This is a pure primal step (x is projected on the linear constraint $Lc_j(x,.) = 0$).

The interpretation of this statement is as follows. It claims that once we have obtained a Kuhn-Tucker point z of a reduced problem (4.21) with $z \in B(z^*, r)$, a move to a neighbouring Kuhn-Tucker point with respect to the same active set but with a weakly active constraint will always be shorter than a step to a Kuhn-Tucker point corresponding to another active set. Bräuniger (1977) proved $I(z^*) = I(z_k)$ using this statement. However the next lemma states that it leads to a contradiction.

LEMMA 4.9

If z^* and z satisfy the conditions of theorem 4.2, $z \neq z^*$, z^* being a regular Kuhn-Tucker point of (4.21) with strict complementary slackness. Then, application of (4.59(i)) - (4.59(iii)) leads to a contradiction.

Proof:

In the point $\bar{z}$ under consideration generated by (4.59(i))-(4.59(iii)) at least

one constraint, say $Lc_j(x\ ,.)$ with $j \in \{1, \ldots, p\}$ is weakly active.
This means $Lc_j(x\ ,\bar{x}) = 0$ with $\mu_j = 0$.
If $Lc_j(x,\bar{x}\ ) = 0$ with $z\ \in B(z^*,r)$ we have either
$c_j(x^*) < 0$, which is impossible as z^* is a Kuhn-Tucker point, or
$c_j(x^*) > 0$, which means, by theorem 4.2(iii) that

$Lc_j(x,\bar{x})\ > 0$, (assuming that both z and $\bar{z} \in B(z^*,r)$)
which contradicts $Lc_j(x,\bar{x}\ ) = 0$.

Hence $c_j(x^*) = 0$, but then the strict complementary slackness at z^* gives $\bar{u}_j > 0$ which, in turn, implies $\mu_j\ > 0$ (theorem (4.2(iv)). This contradicts the construction of $\bar{z}$ as given by (4.59(i)) - (4.59(iii)).

□

In other words Lemma 4.9 states that it is impossible to create weakly active constraints in a neighbourhood of z^*. The proof relies on the assumption that the problem functions are twice continuously differentiable in an open neighbourhood of x^* and that at z^* no weakly active constraints occur. Though this means that Bräuniger's proof is not correct, the result $I(z_k) = I(z^*)$ can be proved as follows from the next two propositions, using $I(z^*) := \{i : c_i(x^*) = 0\}$ and $I(z_k) := \{i : Lc_i(x_k,x) = 0\}$.

PROPOSITION 4.10

Let z^* be a regular Kuhn-Tucker solution of problem (4.21) satisfying the conditions of theorem 4.2 under strict complementary slackness. If $z_k \in B(z^*, \tfrac{1}{2}r)$ with $4\beta||f(z_k)|| \le r$, then $I(z^*) \subset I(z_k)$.

Proof:

We know from proposition 4.3 that $||S(z_k) - z_k|| \le \tfrac{1}{2}r$. Hence for $z_{k+1} \in S(z_k)$

$$||z_{k+1} - z^*|| \le ||z_{k+1} - z_k|| + ||z_k - z^*|| < r$$

Now theorem 4.2 can be applied.
Let $i \in I(z^*)$ be arbitrary.
Then

$u_i^* > 0$ (strict complementary slackness in z^*)

which gives

$\mu_{k,i} > 0$ (theorem (4.2(iv))

hence $\qquad Lc_i(x_k, x_{k+1}) = 0$ (complementary slackness in (4.32))

which means

$$i \in I(z_k).$$

Thus we proved

$$I(z^*) \subset I(z_k).$$

□

The interpretation of proposition (4.10) is obvious: if we are close enough to z^*, all constraints of $I(z^*)$ belong to the current active set $I(z_k)$. Also a reverse statement can be proved:

PROPOSITION 4.11

In case of strict complementary slackness in (4.32) for all $i \in I(z_k)$, with $z_k \in B(z^*,r)$ while z^* and r satisfy the conditions of theorem 4.2, then $I(z_k) \subset I(z^*)$.

Proof:

For $i \in I(z_k)$ chosen arbitrary we know by assumption

$$\mu_{k,i} > 0$$

and

$$Lc_i(x_k,x) = 0 \quad \text{(complementary slackness)}$$

which yields

$$c_i(x^*) = 0 \qquad \text{(proved in lemma 4.11)}$$

hence

$$i \in I(z^*)$$

□

COROLLARY

Under the conditions on z^* and z_k given in proposition 4.10 and the condition of strict complementary slackness in both propositions we conclude: $I(z_k) = I(z^*)$: the currently defined active set equals the final active set.

The proof is obvious.

□

Remark. The earlier procedure discussed above, suggested by Bräuniger (1977), does not satisfy $\mu_{k,i} > 0$ for all $i \in I(z_k)$ as it generates weakly active constraints.

Summarizing we see that $I(z_k) = I(z^*)$ if z_k and z verify some reasonable conditions and if an appropriate active set strategy is applied.

IV.6. Convergence of the solutions of the reduced problems

Insection IV.4 conditions were discussed under which sequences of iteration points $\{z_k\}$, generated by the application of the reduction methods proposed, converge. The limit points turned out to be fixed points z^* of the applied mapping A. These points also appeared among the first-order Kuhn-Tucker points of the original problem (4.21) or among those of the reduced problem (4.40) with $I(z_k) = I(z^*)$.
Now that conditions for convergence have been established, the next point of interest is the *rate of convergence*. This means, given z^*, that the question arises whether it is possible to derive bounds on $||z_k-z^*||$. The next theorem deals with this problem. It is proved that the supremum (for alle sequences $\{z_k\} \to z^*$ generated by the algorithm) of

$$(4.60) \qquad \lim_{k\to\infty} \sup ||z_k-z^*||^{2^{-k}}$$

is an element of (0, 1).
This means in the terminology of Ortega and Rheinboldt (1970) that the convergence is R-quadratic. The theorem is closely related to theorems of Robinson (1972) and Bräuniger (1977), and the proof is a slight modification of theirs.

THEOREM 4.7

Let z^* be a regular Kuhn-Tucker solution of (4.21) satisfying the second order sufficiency conditions with strict complementary slackness in (4.24), while all problem functions are twice continuously differentiable in an open neighbourhood of z^*.
Then there exists a $\delta > 0$ such that if $z_0 \in B(z^*,\delta)$ the algorithm with the active set strategy defined in section IV.5 generates a sequence $\{z_k\}$ which converges R-quadratically to z^*. In particular:

$$(4.61) \qquad ||z_k-z^*|| \le \sum_{j=k}^{\infty} ||z_{j+1}-z_j|| \le \frac{1}{2\beta M} \sum_{j=k}^{\infty} \eta^{2^j} \le \frac{1}{2\beta M} (\tfrac{1}{2})^{2^k} \left[\sum_{j=0}^{\infty} (\tfrac{1}{2})^{2^j} \right]$$

where the scalars β,r and M were defined in th.4.2 and η is defined as

$$(4.62) \qquad \eta := \min(\tfrac{1}{2}, \tfrac{1}{4}\beta Mr)$$

Proof:

First we shall prove by induction a result on $||z_j - z_{j-1}||$ and $||f(z_j)||$ for $j = 1,2,\ldots$

From $f(z^*) = 0$ (z^* is a Kuhn-Tucker point of (4.21)) and lemma 4.1 and the continuity of $f(z)$ we see that there exists a δ: $0 < \delta < \frac{1}{4}r$ such that for all $z \in B(z^*,\delta)$ we have

(4.63) $$||f(z)|| \le \frac{\eta}{4\beta^2 M}$$

Let $z_0 \in B(z^*,\delta)$ be the starting point of the algorithm. Then

$$||z_0 - z^*|| < \delta < \tfrac{1}{2}r$$

and

$$4\beta||f(z_0)|| \le \frac{\eta}{\beta M} \qquad \text{(by (4.63))}$$

$$\le \frac{\frac{1}{4}\beta M r}{\beta M} \qquad \text{(by (4.62))}$$

$$\le \tfrac{1}{4}r$$

$$< r$$

Hence propositions 4.10, 4.11 now yield that $I(z_0) = I(z^*)$ and proposition 4.4 implies that there exists a unique point z, with $z_1 = S(z_0) = S(z_0, I(z^*))$ with

(4.64) $$||z_1 - z_0|| \le 2\beta||f(z_0)||$$

From $z_1 = S(z_0)$ we see that $d(z_0,z_1) = 0$ hence

$$||f(z_1)|| = ||f(z_1) - d(z_0,z_1)||$$

$$\le M||z_1 - z_0||^2 \qquad \text{(by theorem (4.2(ii))}$$

$$\le 4\beta^2 M||f(z_0)||^2$$

$$\le \frac{4\beta^2 M\eta^2}{(4\beta^2 M)^2} \qquad \text{(by (4.63))}$$

hence

(4.65) $$||f(z_1)|| \le \frac{\eta^2}{4\beta^2 M} = \frac{\eta^{2^1}}{4\beta^2 M}$$

We claim that inequalities analogous to (4.64) and (4.65) are valid for any $j = 1, 2, \ldots$. Now that this statement has been proved for $j = 1$, it remains to be proved for $j = k+1$ assuming the validity for $j = 1, 2, \ldots, k$. So for $1 \le j \le k$ we assume:

(4.66) $$||z_j - z_{j-1}|| \le 2\beta||f(z_{j-1})||$$

and

(4.67) $$||f(z_j)|| \le \frac{\eta^{2^j}}{4\beta^2 M}$$

Consider (4.66) for some $1 \le j \le k$:

$$||z_j - z_{j-1}|| \le 2\beta||f(z_{j-1})||$$

$$\le \frac{2\beta\cdot\eta^{2^{j-1}}}{4\beta^2 M} = \frac{\eta^{2^{j-1}}}{2\beta M} \qquad \text{(by (4.67) for j-1)}$$

Then $||z_j - z_{j-1}|| \le \frac{\eta.\eta^{j-1}}{2\beta M}$

$$\le \frac{(\frac{1}{4}\beta Mr)(\frac{1}{2})^{j-1}}{2\beta M} = \tfrac{1}{4}r(\tfrac{1}{2})^j \qquad \text{(by (4.62)}$$

Using this we obtain

$$||z_k - z^*|| \le ||z_0 - z^*|| + \sum_{j=1}^{k} ||z_j - z_{j-1}||$$

(4.68) $$\le \tfrac{1}{4}r + \tfrac{1}{4}r \sum_{j=1}^{k} (\tfrac{1}{2})^j$$

$$< \tfrac{1}{2}r$$

which means that $z_k \in B(z^*, \frac{1}{2}r)$

Further, inequality (4.67) for j=k gives

$$4\beta||f(z_k)|| \le \frac{4\beta\eta^{2^k}}{4\beta^2 M} = \frac{\eta.\eta^{2^k-1}}{\beta M}$$

$$\leq \frac{\frac{1}{4}\beta M r}{\beta M} \cdot \eta^{2^k - 1} \qquad \text{(by (4.62))}$$

$$< \tfrac{1}{4}r < r$$

Now that (4.68) and (4.69) are true, proposition 4.4 can be applied which yields the unique point $z_{k+1} = S(z_k) = S(z_k, I(z^*))$ with

$$||z_{k+1} - z_k|| \leq 2\beta||f(z_k)||,$$

which is equivalent to (4.66) for $j = k+1$.
Finally, (4.67) is true for $j = k+1$, as $z_{k+1} = S(z_k)$ yields $d(z_k, z_{k+1}) = 0$ and hence

$$\begin{aligned} ||f(z_{k+1})|| &= ||f(z_{k+1}) - d(z_k, z_{k+1})|| \\ &\leq M||z_{k+1} - z_k||^2 && \text{(use theorem 4.2(ii))} \\ &\leq M.4\beta^2||f(z_k)||^2 && \text{(just proved)} \\ &\leq \frac{\eta^{2^{j+1}}}{4\beta^2 M} && \text{(by (4.67) for j=k)} \end{aligned}$$

The sequence $\{z_k\}$, thus generated by the described application of the algorithm is an infinite sequence in $\bar{B}(z^*,\frac{1}{2}r)$ so it has at least one cluster point $z' \in \bar{B}(z^*, \frac{1}{2}r)$. Then theorem 4.4 implies that z' is a fixed point of A and theorem 4.5 implies that z' is a Kuhn-Tucker point of (4.21).
The uniqueness of z^* in $\bar{B}(z^*,\frac{1}{2}r)$ implies that necessarily $z' = z^*$.
Finally

$$\begin{aligned} ||z_k - z^*|| &\leq \sum_{i=k}^{\infty} ||z_{i+1} - z_i|| \leq \frac{1}{2\beta M} \sum_{i=k}^{\infty} \eta^{2^i} \\ &\leq \frac{1}{2\beta M} (\tfrac{1}{2})^{2^k} \left[\sum_{i=0}^{\infty} (\tfrac{1}{2})^{2^i} \right] \end{aligned} \tag{4.70}$$

□

COROLLARY 1

The algorithm is asymptotically regular on $B(z^*, \frac{1}{4}r)$.

Proof:

The proof is obvious from (4.70).

□

COROLLARY 2

If C contains at most a countable number of points, then every sequence generated by (4.56), with $z_{k+1} = A(z_k)$ and A is the reduction method developed in this chapter, converges to a point in C: a fixed point of the algorithm which is a first-order Kuhn-Tucker point of (4.21).

Proof:

The proof follows from the application of corollary 2 of proposition 4.8.

□

COROLLARY 3

The effect of only using equality constrained reduced problems is reflected in the fact that $||f(z)||$ for problem (4.40) is smaller than or equal to $||f(z)||$ for problem (4.21).

Proof:

The proof follows from a comparison of proposition 4.3 and 4.4.

□

IV.7. Coordinating phase I and phase II

The proofs of convergence in the preceding chapters rely heavily on theorem 4.2 which assumes that we succeeded in defining a starting point z_0 close enough to z^*. As mentioned above, a first phase, called phase I, is used to provide us with a suitable starting point for phase II, which consists of the solution of reduced problems such as (4.29) or (4.43). In phase I an exterior point penalty function is minimized subject to the given linear constraints ((4.18), (4.19)). The question now is how the quality of the generated starting point z_0 for phase II can be improved and how the results of phase I can be used to define a reasonable approximation of the final active set $I(z^*)$. It may be expected that a penalty parameter in phase I that is too high creates an undesired ill conditioned reduced problem or that this early stage of the iteration process pays too much attention to feasibility at the expense of optimality.
On the other hand a weighing factor of the penalty term that is too low may yield a starting point $z_0 \notin B(z^*,r)$.
It seems reasonable to assert that the above sketched problem depends on the function $F(x)$ to be minimized, the constraint functions $c_i(x)$, $i = 1, \ldots, m$ and their respective gradients and Hessian.
The concept of defining a hybrid, 2-phase algorithm in this way was first suggested by Rosen (1976) (who proposed <u>one</u> exterior penalty minimization as phase I).
It gives rise to the following class algorithms.

Step 1. Initialization: choose a starting point z_0, a penalty parameter t_0, etc.

Step 2. Solve one (or more) problems of the form (4.18) as phase I to obtain $z_1 = (x_1, u_1, v_1)$. Put $z_0 := z_1$ and $k = 0$ to initialize phase II.

Step 3. Solve a reduced problem as developed in this chapter (e.g. (4.29) or (4.43)) as phase II, to obtain $z_{k+1} = (x_{k+1}, u_{k+1}, v_{k+1})$.

Step 4. Apply convergence criteria. In case of convergence: stop. Otherwise $k := k+1$ and return to step 3.

Concerning step 2 we remark that the theorems on the convergence of exterior penalty methods (see Fiacco & Mc Cormick (1968)) guarantee convergence to a Kuhn-Tucker point of the original problem under reasonable, relatively weak conditions. Hence theoretically we might expect a phase I step, which

consists of one exterior penalty step to provide us with a starting point z_0 for phase II which satisfies $z_0 \in B(z^*,r)$. However, neither z^* nor r will be known before starting the iterations, and this is a serious complication! To prevent the occurrence of the problems sketched above, which can arise if phase I consists of only <u>one</u> exterior penalty step (by lack of information t_0 might be too high or too low!), we also investigated the effect of a phase I consisting of more than one exterior penalty step (for instance $t_0 = 0.05$, $t_1 = 100$ and $t_2 = 100t_1$).
Beside this, we also investigate the sensitivity for the choice of t_0 if only one exterior penalty step is performed in phase I. The results of these investigations are summarized in table 5.1 of chapter V.

Besides the choice of the penalty parameter the coordination between the 2 phases of the algorithm is expected to be improved by using the Lagrange multiplier estimates obtained from phase I to define the active set at the starting point z_0 of phase II.
We propose to define this active set as

(4.71) $$I(z_0) := \{p+1, \ldots, m\} \cup \{i \in \{1, \ldots, p\} \mid u_{0,i} > 0\}$$

This means that $I(z_0)$ consists of all equality constraints and those inequality constraints with positive Lagrange multiplier estimates at the end of phase I.
As i-th Lagrange multiplier estimate we use

(4.72) $$\tilde{u}_i = -2t_0 c_i^-(x_0) \qquad i = 1, \ldots, p$$

with $c_i^-(x_0) := \min(0, c_i(x_0))$ (see e.g., Fiacco and Mc Cormick (1968)).
This estimate is based on the fact that in the optimum both the gradient of the Lagrangian function and the gradient of the penalty function (4.19) will vanish.
As soon as $z_0 \in B(z^*,r)$ and the active set strategy satisfies $\mu_{0,i} > 0$ for all $i \in I(z_0)$, propositions 4.10, 4.11 state that $I(z_0) = I(z^*)$ under some mild further conditions.
For instance $I(z_0)$ as defined in (4.71) is intended to meet these requirements. As it remains difficult to check $z_0 \in B(z^*,r)$ we also investigated

some alternative active set strategies to get more insight in the sensitivity of the algorithm on this point.

We considered the active set strategy proposed by Bräuniger (1977) which defines $I(z_0)$ as

$$I(z_0) := \{p+1, \dots, m\} \cup \{i \mid \mu_{0,i} > \frac{|c_i(x_0)|}{||\nabla c_i(x_0)||},\ i=1,\dots,p\} \tag{4.73}$$

Another active set strategy arises from a combination of (4.72) and (4.73) For the (violated) i-th constraint we estimate $\mu_{0,i}$ by $\tilde{u}_i$.

Then we obtain

$$\tilde{u}_i > \frac{|c_i(x_0)|}{||\nabla c_i(x_0)||} \tag{4.74}$$

Using (4.70) this yields

$$-2t_0\bar{c}_i(x_0) > \frac{|c_i(x_0)|}{||\nabla c_i(x_0)||}$$

from which we obtain

$$t_0 > \frac{1}{2||\nabla c_i(x_0)||} \tag{4.75}$$

(Note that $\bar{c}_i(x_0) = c_i(x_0) < 0$ in this case.)

From (4.75) we see that if

$$t_0 > \max\left\{\frac{1}{2||\nabla c_i(x_0)||}\ \middle|\ i=1,\dots,p \text{ with } c_i(x_0) < 0\right\} \tag{4.76}$$

then every constraint which is violated at x_0 will be put into the initial active set of phase II.

IV.8. Convergence of the composite algorithm

Now that the 2 phases of the algorithm have been discussed we conclude this chapter with some closing remarks concerning the convergence of the composite algorithm. In essence the theorem formulated below states that if the penalty parameter t_0 is large enough, the convergence of the exterior penalty methods (see Fiacco and Mc Cormick (1968)) assures $z_0 \in B(z^*,r)$: the final point of phase I is close enough to z^*. Assuming that all other conditions of the theorem on the convergence are satisfied (section IV.6) this means that the convergence is R-quadratic.

THEOREM 4.8

Consider the nonlinear programming problem (4.21) with a regular point z^* which satisfies the second order sufficiency conditions with strict complementary slackness. Assume further that all problem functions are twice continuously differentiable in an open neighbourhood of z^*. Then there exists a t_0^* such that if we use $t_0 \geq t_0^*$ in phase I of the algorithm the generated sequence $\{z_k\}$ of iteration points converges R-quadratically to z_0^*.

Proof:

As mentioned above there exists a t_0^* such that for $t_0 \geq t_0^*$ phase I provides us with a starting point z_0 which meets the requirements of theorem 4.7 so that if the other conditions such as differentiability of the problem functions are satisfied and an appropriate active set strategy is used theorem 4.7 can be applied to prove this theorem.

$\square$

V. ALGORITHMIC AND NUMERICAL ASPECTS OF 2-PHASE REDUCTION METHODS

Now that some properties of reduction methods using linearly constrained reduced problems have been derived, we focus our attention on the implementation of these reduction methods. After a short description of alternative 2-phase algorithms, a specially adapted algorithm for linearly constrained nonlinear programming is described together with a simplified presentation of the implementation of the 2-phase algorithms. Then the LU- and Cholesky matrix decompositions are discussed, together with known and new updating rules to modify the Cholesky-factors during the iteration process.

V.1 Introduction

The 2-phase algorithms developed in the preceding chapter rely heavily on the efficiency of an algorithm for the linearly constrained reduced problems, as both phase I and phase II require the solution of linearly constrained nonlinear programming problems. Here the literature offers an ample choice: for instance, algorithms proposed by Rosen (1960), Goldfarb (1966), Murtagh and Sargent (1969) and Gill and Murray (1974a) could be used.

Usually algorithms for linearly constrained nonlinear programming exploit the given linearity of the constraint functions by applying projections of unconstrained search directions on the intersection of currently active constraints. For instance Rosen (1960) projects the steepest descent direction, in the metric defined by the Euclidean distance function. Murtagh and Sargent (1969) project a quasi-Newton search direction in a metric induced by a distance function which depends on the second order information of the objective function (see e.g., Householder (1964)).

In this chapter we discuss an adapted version of Murtagh and Sargent's algorithm. The adaptation concerns the use of matrix factorizations (ch. V.4) and their updates (ch. V.5). Furthermore the active set strategy is replaced by the strategy described in ch. V.3.

As this chapter only deals with the solution of linearly constrained nonlinear programming problems, our problem formulation can be simflified to:

$$(5.1)\qquad \left\{\begin{array}{l}\text{minimize } F'(x)\\ \text{subject to}\\ A^T x \geq b\end{array}\right.$$

where the objective function $F'(x)$ is supposed to be a sufficiently often differentiable convex function of $(x_1,\ldots,x_n)$. The objective function $F'(x)$ of the reduced problem is defined by the algorithm in terms of $F(x)$, $LF(x_k,x)$, $c_i(x)$ and $Lc_i(x_k,x)$. The $n\times m$ matrix A has as its i-th column the vector a_i which is the gradient of the i-th linear constraint. The $m\times 1$ vector b contains the right hand side elements of the constraints.

Furthermore the matrix N (N_k) will be the matrix of active constraint normals (at step k) while B_k and H_k denote the k-th approximation of the Hessian and the inverse Hessian of $F(x)$ respectively.

V.2 Stepwise description of some 2-phase algorithms

Once we accept the idea to develop an algorithm consisting two separate phases, various designs appear to be possible. In all cases the first phase is intended to provide a good starting point for phase II, and to meet other initial conditions for phase II.

We only mention Rosen (1976,1977), Mayne and Polak (1978), Ballintijn, van der Hoek and Hooykaas (1978), Best, Bräuniger, Ritter and Robinson (1979) and Van der Hoek (1979).

A typical design of the algorithms discussed in the present work is

PHASE I:

```
Step 1 Initialization: $z_0, t_0$.

Step 2 Solve: minimize $F(x) + P(x)$
               $x \in L$
              where $P(x)$ is defined by (4.19)
```

Define z_0, the starting point of phase II, to be equal to the solution of phase I, put $k = 0$ and start

PHASE II

```
Step 1  At $z_k$, find a first order Kuhn-Tucker point $z_{k+1}$ of the reduced
        problem

            $\min_{x \in L \cap LNL(x_k)} F(x) + \phi(x_k, x)$

        If $z_{k+1}$ is not unique, choose the Kuhn-Tucker point closest to
        $z_k$.

Step 2  Apply convergence tests on $z_{k+1}$.
        In case of convergence: stop
        Otherwise, put $z_k := z_{k+1}$, define the active set $I(z_k)$ and go to
        step 1 of phase II.
```

A simple variation of this theme is to use a phase I which consists of a number of exterior penalty minimizations (e.g.: 3), starting with a relatively 'safe' penalty parameter t_0 and multiplying it by a constant factor every succeeding step. In this way hybrid algorithms composed of an exterior penalty algorithm and the reduction methods of ch. IV could be designed, in the hope that no reduced problems are created that are too ill-conditioned. This is reasonable as only a few penalty steps are applied.

Table 5.1 gives the results of some preliminary experiments, performed on the nonlinearly constrained test problems, mentioned in appendix A. The first four columns of the table correspond with the values 0.05, 0.5, 5.0, 50.0 and 500.0 of the penalty parameter t_0 of phase I, while the sixth column concerns a phase I consisting of 3 succeeding exterior penalty steps with values 0.5, 5.0 and 50. of the penalty parameter. The last column gives the results for the algorithm in which the constraint values $c_i(x_0)$ of the nonlinear constraints are used to define t_0 as

(5.2) $$t_0 = \sum_{i \in I_v} \frac{1}{|c_i(x_0)|}$$, where I_v concerns all violated constraints.

This parameter is put to unity if the value thus defined is less than 10^{-4}. The elements of the table are the number of objective function evaluations required for convergence of the particular test problem. (The second phase of the algorithms reported here is the reduction method based on the application of the reduced problem (4.29) with $\phi(x_k,x)$ defined by (4.6)).

Table 5.1 The influence of the penalty parameter of phase I on the convergence of the 2-phase algorithm

Penalty parameter / test problem	0.05	0.5	5.0	50.0	500.0	0.5,5.0 and 50.	formula (5.2)
12	81	108	95	82	82	168	94
13	397	417	474	541	912	696	376
14	211	209	179	183	173	248	178
15	260	256	253	253	212	296	255
16	475	686	576	488	517	882	532
17	394	394	394	394	394	555	394
18	197	197	197	197	197	245	197
19	392	392	392	392	392	410	392
20	5262	5262	5262	5262	5262	5544	5262
21	1413	F	F	F	F	F	1410
22	1458	771	953	6739	6157	6827	1255
23	802	634	566	708	943	1401	643
24	908	F	390	549	940	F	365

F = Failure

Conclusion: Besides the observation that the penalty parameter t_0 of phase I should not be too high, no further conclusion about a preferred fixed penalty parameter can be drawn from the figures of table 5.1. We decided to use the value of t_0 as defined by formula (5.2) in the further experiments. Note that this means that t_0 is based on the constraint violations in the starting point x_0. As a consequence the penalty function used is a combination of problem function values at x_0. Indeed it is a sum of squares of constraint violation amounts, weighted by a penalty parameter which is inverse proportional to the constraint values in question. The resulting 'balanced' penalty function may prevent an undesired step in phase I of the algorithm.

After discussing the choice of the penalty parameter of phase I, chapter IV.7 continued with the presentation of different approaches to improve the *coupling* of phase I and phase II. Again the value of the penalty parameter appeared to be important, together with the definition of the initial active set of phase II. Different coupling mechanisms evolving from the definitions (4.71),(4.73) and (4.76) of this active set were discussed in ch. IV.7.

A main result of chapter IV was the proof that, once an iteration point is in a sufficiently small neighbourhood of a 2nd order Kuhn-Tucker point, it suffices to consider merely equality constrained reduced problems in phase II of the algorithm. This means that the linearly constrained nonlinear programming algorithm which solves the reduced problems, no longer applies an active set strategy; all constraints involved are equality constraints, hence elements of the active set. The active set $I(z_k)$ is chosen *before* defining the reduced problem. For this algorithm, phase II becomes

Step 1 Arrived at z_k, find a first order Kuhn-Tucker point z_{k+1} of the reduced problem

$$(5.3) \quad \begin{cases} \min F(x) + \phi(x_k,x) \\ \text{subject to} \\ Lc_i(x_k,x) = 0 \qquad \text{for all } i \in I(z_k) \end{cases}$$

If z_{k+1} is not unique, choose the Kuhn-Tucker point which is closest to z_k.

Step 2 Apply convergence tests on z_{k+1}.
In case of convergence: stop.
Otherwise put $z_k := z_{k+1}$, define the active set $I(z_k)$ and go to step 1 of phase II.

Numerical experiments were performed on the same test set of 13 nonlinearly constrained test problems under the same circumstances as the experiments reported above on two implementations of the 2-phase algorithm, which apply for phase II a fixed active set or an adjustable active set respectively. The results of the experiments are summarized in table 5.2. The quotients in this table have the following meaning:

(5.4) $$\frac{\text{number of major iterations required to detect } I(z^*)}{\text{number of major iterations required for convergence}} .$$

Here each major iteration involves the definition and the solution of a reduced problem of phase II. Column 1 concerns the algorithm in which the penalty parameter of phase I is defined by (5.2) and phase II redefines the active set if necessary. Column 2 concerns an implementation with the same phase I, while the active set $I(x_k)$ of phase II remains unchanged. The set $I(x_k)$ is defined in the feasibility step which precedes the application of the linearly constrained nonlinear programming algorithm which solves phase II.

Table 5.2 $\frac{\text{number of major iterations required to detect } I(z^*)}{\text{number of major iterations required for convergence}}$

algorithm / test problem	1	2
12	1 : 4	3 : 4
13	0 : 4	0 : 4
14	0 : 19	17 : 19
15	2 : 7	F
16	0 : 3	2 : 3
17	0 : 3	0 : 3
18	3 : 4	1 : 4
19	0 : 3	0 : 3
20	9 : 10	F
21	1 : 4	1 : 4
22	0 : 3	0 : 3
23	4 : 7	2 : 7
24	0 : 3	0 : 3

F : Failure

Conclusion: If the convergence is obtained, then the applied definition of the active set does not influence the total number of required major iterations. However, the application of a fixed active set in phase II gives a less robust algorithm: 2 failures are reported. Apparently the second implementation is less flexible in adapting new information on the (in)-activity of constraints, which is expressed in the failures and a higher number of major iterations required to detect $I(z^*)$.
A further computational comparison of these two implementations is postponed to chapter VI.

Another obvious hybrid algorithm arises from a combination of the recursive quadratic programming algorithms of ch. III and the algorithms of ch. IV as phase I and phase II respectively. This yields the following algorithm:

PHASE I

Step 1 Initialization: z_0, t_0

Step 2 solve:

$$(5.5)\quad \begin{cases} \min\limits_{x} \tfrac{1}{2}(x-x_0)^T\nabla^2F(x_k)(x-x_0) + (x-x_0)^T\nabla F(x_k) + F(x_k) \\ \text{subject to} \\ A^Tx = b \end{cases}$$

where A and b are defined in ch. III.

Define z_0, the starting point of phase II, to be equal to the solution of phase I and start

Phase II. This phase can be any of the above presented phase II algorithms.

Finally we mention the independently developed and recently reported 2-phase algorithm of Best, Bräuniger, Ritter and Robinson (1979), which also applies linearly equality constrained reduced problems, together with an active set strategy which, if necessary, redefines the penalty parameter to obtain global convergence.

V.3 An adapted algorithm for linearly constrained nonlinear programming and the structure of the 2-phase algorithm

As both phase I and phase II of the algorithm reduce the solution of the problem to the solution of a linearly constrained nonlinear programming problem, special attention must be paid to the algorithms used to solve these problems. We choose Murtagh and Sargent's (1969) algorithm because of its reported computational efficiency in Himmelblau (1972) and Lenard (1979), and because of the possibility to derive easily the necessary update formulae for the applied Cholesky-decompositions.

We mentioned in Ch. IV, that usually methods for linearly constrained nonlinear programming are based on the use of adapted (e.g., projected) unconstrained search directions. This leads to algorithms which are composed of steps which define an unconstrained search direction, project it on the feasible set given by the active constraints, and then update the iteration matrices (if convergence is not yet achieved). Murtagh and Sargent's algorithm is an implementation of the ideas originally suggested by Davidon (1959), and later extended by Goldfarb (1966) and Davies (1968). Convergence theorems which prove the convergence of the algorithm can be found in Murtagh and Sargent (1969). The modifications and extension of the algorithm as implemented by us concern the following points:

1. The active set strategy. We only test whether a constraint has to be dropped from the set of active constraints in case of:
   (i) convergence with respect to the current active set,
   (ii) reinitialization of the approximation of the Hessian matrix, because of accumulation of calculation errors,
   (iii) any other constraint enters or leaves the active set.
2. For the criterion which constraint (if any) to delete we followed the suggestion of Gill and Murray (1974c) (step 7 of the algorithm).
3. For reasons of numerical stability B_k and $N_k^TH_kN_k$ are used instead of the matrices H_k and $(N_k^TH_kN_k)^{-1}$. The matrices B_k and $N_k^TH_kN_k$ are stored in the form of their Cholesky decompositions.

The resulting algorithm is as follows:

Step 1. Initialization: a feasible starting point x_0 is generated. Take $B_0 = I_n$ and determine $I(x_0)$, the set of active constraints at x_0

and the corresponding matrix N_0 of active constraint normals. Compute the Cholesky decomposition of $N_0^T H_0 N_0$ and set $k := 0$.

Step 2. $k := k+1$.

Determine the search direction

$$p_k = -P_k H_k \nabla F(x_k) \tag{5.6}$$

where the projection matix P_k is given by

$$P_k = I - H_k N_k (N_k^T H_k N_k)^{-1} N_k^T. \tag{5.7}$$

Find the maximum steplength, α_k^m, along p_k from

$$\alpha_k^m = \min_{i \notin I(x_k)} \left\{ \frac{b_i - a_i^T x_k}{a_i^T p_k} \;\middle|\; a_i^T p_k < 0 \right\}. \tag{5.8}$$

Calculate the vector of approximate Lagrange multipliers λ_k as

$$\lambda_k = (N_k^T H_k N_k)^{-1} N_k^T H_k \nabla F(x_k). \tag{5.9}$$

If $\| N_k^T p_k \| > \varepsilon$, where $\varepsilon > 0$ is a small, pregiven constant, the search direction is no longer parallel to the intersection of the active constraints, go to step 6.

If $\| p_k \| < \varepsilon$ or if in the preceding iteration the set of active constraints was changed, go to step 7. Otherwise go to step 3.

Step 3. Find the steplength $0 < \alpha_k \le \alpha_k^m$ that solves:

$$\min_{0 < \alpha \le \alpha_k^m} F(x_k + \alpha p_k). \tag{5.10}$$

Go to step 4.

Step 4. Set $x_{k+1} := x_k + \alpha_k p_k$ and modify the Cholesky-factors of B_k and $N_k^T H_k N_k$ to obtain B_{k+1} and $N_{k+1}^T H_{k+1} N_{k+1}$. If $\alpha_k = \alpha_k^m$ go to step 5, otherwise go to step 2.

Step 5. A new constraint, whose index was found in the solution of (5.8) has become active. Add its normal to N_k to obtain N_{k+1} and modify the Cholesky factors of $N_k^T H_k N_k$ accordingly. Go to step 2.

Step 6. Reset $B_k := I_n$ and adjust $N_k^T H_k N_k$ accordingly. Go to step 7.

Step 7. Select the largest Lagrange multiplier, say $\lambda(j)$, and calculate

$$\beta = \frac{\lambda^2(j)}{2b(j,j)} ,$$

where $b(j,j)$ is the jth diagonal element of $(N_k^T H_k N_k)^{-1}$. β can be interpreted as the expected improvement of the objective function if constraint j is dropped from $I(z_k)$. Stop the algorithm if both $\|p_k\| < \varepsilon$ and $\beta < \varepsilon$. If $-g_k^T p_k \leq \beta$, drop the jth constraint from $I(z_k)$. Update N_k and modify the Cholesky factors of $N_k^T H_k N_k$ accordingly to obtain $N_{k+1}^T H_{k+1} N_{k+1}$. If no change in $I(z_k)$ occurred, continue with step 3. Otherwise set $x_k := x_{k+1}$ and go to step 2.

We saw in ch.IV that if all problem functions satisfied certain differentiability and concavity conditions and if linearizations were performed around a feasible point x_k, then x_k was feasible with respect to the linearized constraints as well. Hence x_k can be used as a feasible starting point for the linearly constrained algorithm. We also found that as soon as the problem functions no longer satisfy the above conditions, the feasibility of the reduced problem could not be guaranteed. Besides that we know that the solution of the linearized problem will not necessarily satisfy the original nonlinear constraints. That is the reason why the application of the algorithm is preceded, if necessary, by a feasibility-step, which transforms an infeasible initial point into a feasible starting point for the application of the linearly constrained algorithm.

The procedure starts with transforming the infeasible starting point into a point which satisfies the linear equality constraints. Then an auxiliary problem is solved in which the magnitude of the violations is decreased so that the satisfied constraints are kept as such. A stepwise description of this feasibility step is:

Step 1. If the current point x_k is feasible, start the linearly constrained algorithm; otherwise, transform x_k into a solution, x_k', of the linear equality constraints (by orthogonal projection). Go to step 2.

Step 2. Define the index sets I_v and I_s corresponding with the currently violated and satisfied constraints respectively. Go to step 3.

Step 3. Find x'_{k+1} which solves

$$(5.11)\qquad \begin{cases} \min \sum_{i \in I_v} (a_i^T x - b_i)^2 \text{, where the } a_i^T \text{ and } b_i \text{ are as in (5.1),} \\ \text{such that all constraints with } i \in I_s \text{ are met} \end{cases}$$

Go to step 4.

Step 4. Adjust I_v and I_s with respect to x'_{k+1}. Go to step 5.

Step 5. If $I_v = \phi$, enter the linearly constrained routine. Otherwise go to step 2.

Obviously this feasibility step provides us with a feasible point by applying the linearly constrained algorithm itself (in step 3). A similar strategy was proposed earlier by Fiacco (1961) to start Carrolls 'created response surface algorithm', the first version of an interior point penalty algorithm. The convergence of this feasibility algorithm is based on the fact that the constraint violations are penalized in the objective function of (5.11), hence the number of elements of I_v will decrease monotonically until $I_v = \phi$: a feasible point has been achieved. Of course the rate of convergence of this algorithm depends on the particular choice of the loss-function in (5.11), together with the diameter of the feasible region. If this algorithm does not succeed in finding a feasible point, the problem is considered to be infeasible.

We conclude this section with a simplified statement of the structure of the implementation of the 2-phase algorithm (a detailed description can be found in appendix D).

In figure 5.1 NLPSOL is a subroutine which controls the procedure to solve the given nonlinear programming problem. NLPSOL applies a subroutine which solves a linearly constrained nonlinear programming problem (LINSOL) and the other subroutines mentioned. The auxiliary calculation subroutines concern the linearization of constraint functions, the solution of a system of linear equality constraints and the line search. The matrix decomposition subroutines calculate the necessary LU- and Cholesky decompositions and perform the updating of the Cholesky factors as discussed in the remaining part of this chapter.

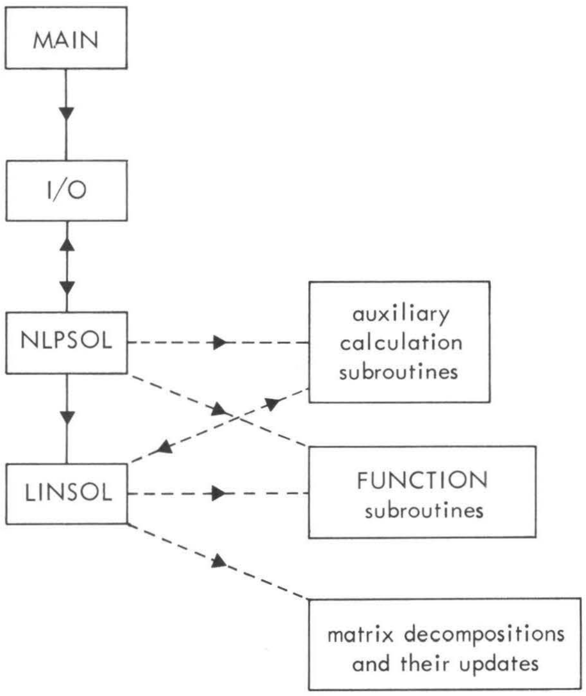

MAIN
I/O
NLPSOL
LINSOL
auxiliary calculation subroutines
FUNCTION subroutines
matrix decompositions and their updates

Figure 5.1 Simplified flowchart of the 2-phase algorithm

V.4 Decomposition methods for matrices

Especially in the last decade much attention has been paid to the development of efficient and numerically stable solution procedures to solve sets of linear equations. As the 2-phase algorithms require these solution procedures (to find a solution to a set of linear equality constraints and to define the constrained search direction), we decided to apply appropriate matrix decompositions with the general objectives

(i) to perform fast calculations

(ii) to obtain great numerical stability, especially by using permutation matrices to reorder diagonal elements before solving a system of equations

(iii) to recognize non-positive definiteness of B_k and $N_k^T H_k N_k$.

We shall now give a brief description of these LU- and Cholesky decompositions. The next chapter, V.5, will be concerned with updating procedures for the Cholesky-factors of the above mentioned iteration matrices. For a description of the used 'special' matrices, the reader should consult appendix C or Wilkinson (1965).

V.4.1 LU-decomposition following Peters and Wilkinson (1970)

This method constructs the factors L and U, which are m×n-lower trapezoidal and n×n-unit upper triangular matrices, of a m×n-matrix A ($m \geq n$) such that

(5.12) $$A = LU.$$

The use of LU-decompositions goes back to Bartels and Golub (1969) who use them in the context of linear programming. A motivation to apply this decomposition is that it replaces the solution of the n×n system $Ax = b$ by the solution of the two triangular systems $Ly = b$ and $Ux = y$. The calculation of L and U together with the solution of $Ly = b$ is usually termed a forward elimination, while the solution of $Ux = y$ is called a back substitution. Every nonsingular n×n matrix A has an LU decomposition, provided that interchanges of rows of A are introduced if necessary, to calculate these factors.

Application to our m×n matrix A, with $m \geq n$, means that we have to replace the lower triangular matrix L by a lower trapezoidal matrix L to be able to find the solution of the first n of the given m linear equations in n variables. We elected to use an algorithm due to Peters and Wilkinson which generates a sequence of pairs (L_0,U_0), (L_1,U_1),...,(L_n,U_n) such that

$$L = L_n \text{ and } U = U_n \quad \text{and} \quad A = L_iU_i \quad \text{for } i = 1,2,\ldots,n.$$

A description of their algorithm is:

Step 1. Initialization: $L_0 = A$, $U_0 = I$, hence $L_0U_0 = A$.
Go to step 2.

Step 2. Put $k = 1$. Form a special matrix E_1 (as defined in appendix C.6) such that $L_1 = L_0E_1$ with $L_1(1,j) = 0$ for $j > 1$. Compute E_1^{-1}, which is again such a special matrix and calculate $U_1 = E_1^{-1}U_0$. Hence

$$\begin{aligned} L_1U_1 &= L_0E_1E_1^{-1}U_0 \\ &= A. \end{aligned}$$

Go to step 3.

Step 3. Put $k := k+1$.
Form the special matrix E_k such that $L_k = L_{k-1}E_k$ and $L_k(i,j) = 0$ for $j > i$, $i \leq k$. Go to step 4.

Step 4. Calculate the special matrix E_k^{-1} and define $U_k = E_k^{-1}U_{k-1}$. Then $L_kU_k = A$. Go to step 5.

Step 5. If $k = n$, stop with $L = L_n$, $U = U_n$ and $A = LU$. Otherwise, go to step 3.

Note that for U_k, as defined in step 4, the following holds:

$$\begin{aligned} &U_k(i,i) = 1 \\ &U_k(i,j) \neq 0 \text{ only for } i < j \text{ and } i \leq k \\ &U_k(i,j) = 0 \text{ else.} \end{aligned}$$

For reasons of numerical stability we do not calculate a decomposition of the matrix A itself, but of the matrix PA.
Here P is the permutation matrix, which is defined during the execution of the algorithm in the following way:

(5.13) $$P = \prod_{k=1}^{n} P_k$$

where P_k is the permutation matrix that provides us with an adjusted matrix L'_{k-1} instead of L_{k-1}:

(5.14) $$L'_{k-1} = P_k.P_{k-1}...P_1.L_0.E_1.E_2...E_{k-1}$$

with as k-th diagonal element the absolutely largest element of the k-th column. The implementation of this algorithm requires $.5mn^2 - \frac{1}{6}n^3 + O(n^2)$ multiplications for making the LU decomposition of an m×n matrix A.

V.4.2 Cholesky-decomposition

A Cholesky-decomposition of a positive definite, symmetric n×n matrix A is the factorization

$$A = LDL^T \tag{5.15}$$

where L is a unit lower triangular matrix and D is a diagonal matrix. The proof of the existence and uniqueness of this decomposition is by induction on n and follows from the construction below. For n = 1 we take L = 1, D(1,1) = A(1,1), which clearly gives the required factors. Note that D(1,1) > 0.

A positive definite symmetric matrix A_n can be written as

$$A_n = \left[\begin{array}{c|c} A_{n-1} & b \\ \hline b^T & a_{nn} \end{array}\right] \tag{5.16}$$

where A_{n-1} is positive definite and symmetric. Hence, by induction hypothesis, A_{n-1} can be decomposed as

$$A_{n-1} = L_{n-1}D_{n-1}L_{n-1}^T \tag{5.17}$$

where L_{n-1} and D_{n-1} are its Cholesky-factors.

Then the Cholesky-factors for A_n can be constructed in the following way. Define

$$L_n = \left[\begin{array}{c|c} L_{n-1} & 0 \\ \hline c^T & 1 \end{array}\right] \tag{5.18}$$

and

$$D_n = \left[\begin{array}{c|c} D_{n-1} & 0 \\ \hline 0 & x \end{array}\right] \tag{5.19}$$

Then

(5.20) $$L_nD_nL_n^T = \left[\begin{array}{c|c} L_{n-1}D_{n-1}L_{n-1}^T & L_{n-1}D_{n-1}c \\ \hline c^TD_{n-1}L_{n-1}^T & c^TD_{n-1}c + x \end{array}\right].$$

The unknowns c(a vector) and x (a scalar) in (5.18) and (5.19) are found from the requirement that (5.16) and (5.20) are equal, which means

(5.21) $$L_{n-1}D_{n-1}c = b$$

and

(5.22) $$c^TD_{n-1}c + x = a_{nn}.$$

The vector c is uniquely determined by (5.21) and requires one back substitution. The unknown x is given by (5.22). Taking determinants in $A_n = L_nD_nL_n^T$ further gives

$$0 < \det(A_n) = \det(L_n)\det(D_n)\det(L_n^T),$$

so that

(5.23) $$\det(A_n) = x \det(D_{n-1}).$$

As $\det(D_{n-1}) > 0$ by the induction hypothesis, we see that x is real and positive.

This proof of the existence of a Cholesky decomposition gives at the same time a method to calculate the factors and these factors are calculated iteratively in n steps.

As in each step the newly determined vector c (in (5.18) and (5.21)) and the new element x (in (5.19) and (5.22),(5.23)) are uniquely defined we conclude that the Cholesky-factors of a positive definite symmetric matrix are unique as well.

Concerning the implementation we remark that for reasons of numerical stability the diagonal elements of A are ordered according to their magnitude, hence we calculate the Cholesky factors of the matrix PAP' where P is a permutation matrix. The implemented subroutine requires

(5.24) $\frac{1}{3}n^3 + O(n^2)$

multiplications to calculate the Cholesky-factorization of a matrix of order n.

V.5 Updating of the Cholesky factors

The two matrices B_k and $N_k^T H_k N_k$ which are stored in their Cholesky decompositions require different updating strategies. For B_k, the current approximation of the Hessian of the objective function, a rank 1 correction formula is applied for updating. This rank 1 formula, which is simpler than the rank 2 formulas treated in ch. II, has the advantage that the corresponding corrections of the Cholesky factors of B_k are relatively simple and available from the literature.

For the matrix $N_k^T H_k N_k$ updating of its Cholesky factors is necessary in case of a change in the active set $I(z_k)$, as both adding and deleting a constraint results in a new matrix N_k. Moreover the updating of the inverse Hessian approximation H_k has its consequences for the factors of $N_k^T H_k N_k$. All the necessary updating rules will be discussed in this chapter, in the order in which they are mentioned above.

V.5.1 The rank 1 correction of B_k and H_k

We are looking for a simple correction matrix C_k for updating B_k:

(5.25) $B_{k+1} = B_k + C_k$

such that the 'quasi-Newton property'

(5.26) $B_{k+1}s_k = y_k$, where s,y are as before in section II.2.1.

holds. From (5.25) and (5.26) we derive

(5.27) $C_k s_k = y_k - B_k s_k$,

which becomes

(5.28) $C_k s_k = v_k$

with

(5.29) $$v_k = y_k - B_k s_k .$$

Equation (5.28) consists of n equations, whereas a symmetric matrix C_k has $\frac{1}{2}n(n+1)$ elements to be determined. The simplest choice for a symmetric matrix C_k seems to be a rank 1 matrix

(5.30) $$C_k = r_k w_k w_k^T$$

where the scalar r_k and the n-vector w_k are still to be determined. Equations (5.30) and (5.28) yield

(5.31) $$r_k w_k w_k^T s_k = v_k$$

and, consequently,

(5.32) $$w_k = q_k v_k$$

with

(5.33) $$q_k = \frac{1}{r_k w_k^T s_k} .$$

Further (5.32) and (5.31) give rise to

(5.34) $$r_k q_k^2 v_k v_k^T s_k = v_k$$

from which we see

(5.35) $$r_k q_k^2 = \frac{1}{v_k^T s_k} .$$

Combining these results we obtain

$$C_k = r_k w_k w_k^T \qquad \text{(use (5.30))}$$

$$= r_k q_k^2 v_k v_k^T \qquad \text{(use (5.32))}$$

(5.36) $$= \frac{v_k v_k^T}{v_k^T s_k} \qquad \text{(use (5.35).}$$

Formula (5.36) gives the, unique, rank 1 updating formula which originates from Broyden (1967).

Analogously the corresponding rank 1 update formula for modifying the inverse Hessian approximation H_k can be derived to be

$$(5.37) \qquad H_{k+1} = H_k + \frac{t_k t_k^T}{t_k^T y_k}$$

with

$$(5.38) \qquad t_k = s_k - H_k y_k.$$

It can be proved that if $H_k B_k = I_n$ and these rank 1 formulae are applied to define H_{k+1} and B_{k+1}, then $H_{k+1} B_{k+1} = I_n$ holds as well.

As already mentioned our main rationale for choosing the rank 1 update was that it allows the possibility to apply Cholesky decompositions in a simple way. Besides that a number of advantages and disadvantages are known (see e.g., Powell (1969), Himmelblau (1972), Gill and Murray (1974c)), such as:

(i) no exact line search is required, though this might lead to a singular updated matrix. This advantage is even greater as we are concerned with a feasible point algorithm for linearly *constrained* reduced problems, where the unconstrained minimum along a search direction need not be feasible.

(ii) positive definiteness might be lost, in which case no updating should be performed, as the subsequent search direction will not be initially downhill. This phenomenon can be recognized easily from the Cholesky factors.

V.5.2 Updating of the Cholesky factors of B_k and H_k after a rank 1 correction

The algorithms to modify the Cholesky factors of a symmetric positive definite matrix A after a rank 1 correction come from the theory presented in Gill, Golub, Murray and Saunders (1974) and Gill, Murray and Saunders (1975). We only provide a brief description of their algorithms.

The question is how to calculate in a fast and numerically stable manner the Cholesky factors of a positive definite symmetric matrix, say A_{k+1} defined by

(5.39) $$A_{k+1} = A_k + r_k w_k w_k^T$$

where w_k is a non-trivial n-vector. We shall assume that both A_{k+1} and A_k are positive definite with Cholesky decompositions $L_{k+1}D_{k+1}L_{k+1}^T$ and $L_kD_kL_k^T$ respectively. After appropriate scaling of w_k (thus obtaining v_k) and distinguishing between $r_k > 0$ and $r_k < 0$ we have to consider the following two cases:

(i) $$L_{k+1}D_{k+1}L_{k+1}^T = L_kD_kL_k^T + v_kv_k^T$$

(ii) $$L_{k+1}D_{k+1}L_{k+1}^T = L_kD_kL_k^T - v_kv_k^T$$

(i) Choose p, such that $L_kp = v_k$. Then case (i) is:

$$\begin{aligned} A_{k+1} = L_{k+1}D_{k+1}L_{k+1}^T &= L_kD_kL_k^T + v_kv_k^T \\ &= L_kD_kL_k^T + L_kpp^TL_k^T \\ &= L_k(D_k + pp^T)L_k^T. \end{aligned} \quad (5.40)$$

Now lemma A3 of Gill, Murray and Saunders can be applied to give the Cholesky factors of $D_k + pp^T$ as

(5.41) $$D_k + pp^T = MDM^T$$

where M and D are special matrices, defined in loc.cit.
From (5.40) and (5.41) and using the known structure of M and D, we get:

(5.42) $$A_{k+1} = L_kMDM^TL_k^T$$

hence

(5.43) $$L_{k+1} = L_kM \quad \text{and} \quad D_{k+1} = D$$

where L_{k+1} is a unit lower triangular matrix and D_{k+1} is a diagonal matrix. For further details and suggestions for an efficient implementation we refer to the paper cited above.

(ii) Again using $L_k p = v_k$ in the second case

$$L_{k+1}D_{k+1}L_{k+1}^T = L_k D_k L_k^T - v_k v_k^T$$

$$= L_k D_k L_k^T - L_k pp^T L_k^T$$

$$= L_k (D_k - pp^T) L_k^T. \qquad (5.44)$$

Considering the quantity $\alpha = 1-p^T D_k p$ in conjunction with (5.44) gives

$$\det(A_{k+1}) = (\det(L_k))^2 \det(D_k - pp^T)$$

which means, using that L_k is unit triangular,

$$\det(A_{k+1}) = \det(D_k - pp^T)$$

$$= \alpha \det(D_k). \qquad (5.45)$$

As A_{k+1} and D_k are assumed to be positive definite, $\det(A_{k+1})$ and $\det(D_k)$ are positive hence α is positive. Now lemma A4 of Gill, Murray and Saunders can be applied to obtain the factorization

$$D_k - pp^T = MDM^T \qquad (5.46)$$

where M and D are special matrices defined in loc.cit. Analogously to the preceding case the Cholesky factors L_{k+1} and D_{k+1} of A_{k+1} can now be given in terms of L_k, M and D.

In the implementation a problem can arise after calculation of $\alpha = 1 - p^T D^{-1} p$. This calculation is to be followed by a test on the sign of this quantity. In case of $\alpha \leq 0$ we decided to set $A_{k+1} = A_k$, which means that no updating takes place. Another approach could be to replace a negative α by a small positive number e.g., the machine precision, in order to be able to continue with the updating. For further details and suggestions for an efficient implementation we refer to the paper cited above.

V.5.3 Updating of the Cholesky factors of $N_k^T H_k N_k$

During the iteration process both H_k and N_k change from time to time and the matrix $N_k^T H_k N_k$ and its Cholesky factors should be adjusted accordingly. We preferred to update the available factors, instead of performing a complete recalculation of the decomposition after every change of H_k or N_k. In that way the information gathered in the past is used to accelerate convergence. Necessary updating strategies for the cases treated in V.5.3.1 and V.5.3.3 were discussed earlier, e.g., in Gill and Murray (1974a) and Goldfarb (1975). In case V.5.3.2 the strategy developed in Ballintijn, Van der Hoek and Hooykaas (1978) was implemented.

Updating of $N_k^T H_k N_k$ and its factors is necessary if:

V.5.3.1 a constraint is added to $I(z_k)$, the current active set of constraints.

V.5.3.2 a constriant is dropped from $I(z_k)$

V.5.3.3 the rank 1 correction is applied to H_k.

The following strategies were applied in these cases.

V.5.3.1 A constraint is added to $I(z_k)$

If $I(z_k)$ consists of m_k constraints, this means that the matrix N_k^T is augmented with a row vector, say n_k^T, which is the gradient vector of the added constraint.

Then

$$N_{k+1}^T H_k N_{k+1} = \left[\begin{array}{c} N_k^T \\ \hline n_k^T \end{array}\right] \left[H_k \right] \left[\begin{array}{c:c} N_k & n_k \end{array}\right]$$

(5.47) $$= \left[\begin{array}{c:c} N_k^T H_k N_k & N_k^T H_k n_k \\ \hdashline n_k^T H_k N_k & n_k^T H_k n_k \end{array}\right]$$

Equation (5.47) means that $N_{k+1}^T H_k N_{k+1}$ arises from its predecessor $N_k^T H_k N_k$ by adding one row and one column to it, which means that the new Cholesky factors are obtained by simply taking one more step in the calculation of the Cholesky factors, this requires $m_k^2 + \frac{1}{2}n^2 + O(n)$ multiplications.

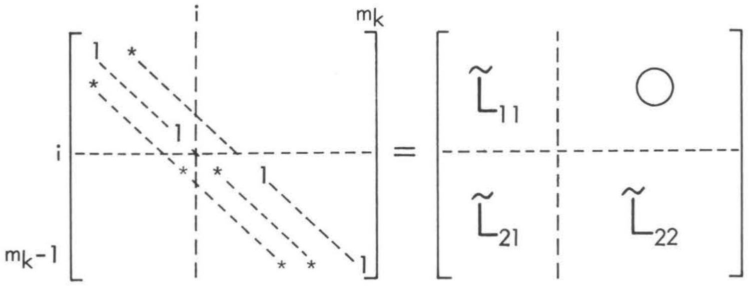
V.5.3.2 <u>A constraint is deleted from $I(z_k)$</u>

Let the Cholesky decomposition of the matrix $N_k^T H_k N_k$ be given by $L_k D_k L_k^T$. Deletion of the ith constraint from the active set means that its gradient vector is to be deleted from N_k, yielding N_{k+1}. Then the ith row and the ith column of $N_k^T H_k N_k$ is to be deleted. If we drop the ith row in L_k we obtain $\tilde{L}_k$ and the relation

$$\tilde{L}_k D_k \tilde{L}_k^T = N_{k+1}^T H_k N_{k+1} \tag{5.48}$$

still holds, but $\tilde{L}_k$ is no longer a unit lower triangular matrix, hence (5.48) does not represent the Cholesky decomposition of $N_{k+1}^T H_k N_{k+1}$. The matrix $\tilde{L}_k$ has the following structure:

(5.49)



In (5.49) we partitioned the right hand side matrix into:

$\tilde{L}_{11}$: $(i-1)\times(i-1)$ unit lower triangular matrix
$\tilde{L}_{21}$: $(m_k-i)\times(i-1)$ matrix
$\tilde{L}_{22}$: $(m_k-i)\times(m_k-i+1)$ matrix.

In a similar way the matrix D_k can be partitioned into:

$\tilde{D}_{11}$: $(i-1)\times(i-1)$ diagonal matrix

$\tilde{D}_{22}$: $(m_k-i+1)\times(m_k-i+1)$ diagonal matrix.

Application of these partitions to $N_{k+1}^T H_k N_{k+1}$ gives

$$N_{k+1}^T H_k N_{k+1} = \tilde{L}_k D_k \tilde{L}_k$$

(5.50)
$$= \left[\begin{array}{c|c} \tilde{L}_{11}\tilde{D}_{11}\tilde{L}_{11}^T & \tilde{L}_{21}\tilde{D}_{11}\tilde{L}_{11}^T \\ \hline \tilde{L}_{11}\tilde{D}_{11}\tilde{L}_{21}^T & \tilde{L}_{21}\tilde{D}_{11}\tilde{L}_{21}^T + \tilde{L}_{22}\tilde{D}_{22}\tilde{L}_{22} \end{array}\right]$$

On the other hand the Cholesky decomposition of $N_{k+1}^T H_k N_{k+1}$ is

(5.51)
$$N_{k+1}^T H_k N_{k+1} = L_{k+1} D_{k+1} L_{k+1}^T .$$

Partitioning of the $(m_k-1)\times(m_k-1)$ matrices L_{k+1} and D_{k+1} in a similar way as L_k and D_k leads to

(5.52)
$$L_{k+1}D_{k+1}L_{k+1}^T = \left[\begin{array}{c|c} L_{11}D_{11}L_{11}^T & L_{21}D_{11}L_{i1}^T \\ \hline L_{11}D_{11}L_{21}^T & L_{21}D_{11}L_{21}^T + L_{22}D_{22}L_{22}^T \end{array}\right]$$

Combining (5.50),(5.51) and (5.52) with the uniqueness of the Cholesky decomposition we conclude that

(5.53)
$$L_{11} = \tilde{L}_{11}$$

(5.54)
$$L_{21} = \tilde{L}_{21}$$

(5.55)
$$D_{11} = \tilde{D}_{11}$$

(5.56)
$$L_{22}D_{22}L_{22}^T = \tilde{L}_{22}\tilde{D}_{22}\tilde{L}_{22}^T$$

Relations (5.53)-(5.55) express that the first (i-1) columns of L_k and D_k remain unchanged. Equation (5.56) means that as soon as the Cholesky decomposition of $\tilde{L}_{22}\tilde{D}_{22}\tilde{L}_{22}^T$ is known, which takes $\frac{1}{3}(m_k-i)^3 + O((m_k-i)^2)$ multiplications, the required Cholesky decomposition (5.51) of $N_{k+1}^T H_k N_{k+1}$ is given by the factors

(5.57)
$$L_{k+1} = \left[\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & L_{22} \end{array}\right]$$

$$D_{k+1} = \left[\begin{array}{c|c} D_{11} & 0 \\ \hline D_{21} & D_{22} \end{array}\right] . \tag{5.58}$$

V.5.3.3 A rank 1 correction is applied to H_k

The rank 1 correction of H_k given by

$$H_{k+1} = H_k \pm v_k v_k^T \tag{5.59}$$

means that $N_k^T H_k N_k$ should be replaced by

$$N_k^T H_{k+1} N_k = N_k^T (H_k \pm v_k v_k^T) N_k$$

which gives

$$N_k^T H_{k+1} N_k = N_k^T H_k N_k \pm (N_k^T v)(N_k^T v)^T . \tag{5.60}$$

The interpretation of (5.60) is that the rank 1 correction of H_k using v_k yields a rank 1 correction of $N_k^T H_k N_k$ using $N_k^T v_k$. Hence the theory of section V.5.2 on the modification of Cholesky factors in case of a rank 1 correction applies and one and the same algorithm can be used for the correction of the Cholesky factor of H_k and of $N_k^T H_k N_k$.

CHAPTER VI. A COMPUTATIONAL COMPARISON OF 2-PHASE ALGORITHMS AND RECURSIVE QUADRATIC PROGRAMMING ALGORITHMS

VI.1. The design of computational experiments and the selection of test problems

Theoretically the performance of a general nonlinear programming algorithm can be described in terms of *convergence* and *rate of convergence*. Then for suitably chosen problems the guaranteed convergence and/or bounds on the rate of convergence can be calculated. See, e.g., the theory developed in chapter III and chapter IV of this monograph.
However, it is not advisable to compare general algorithms solely on a theoretical basis, as the actual convergence behaviour is clearly problem dependent. In practical problems convergence is obtained frequently under violation of theoretically imposed conditions, such as convexity or differentiability of the problem functions. Besides that up to now no single nonlinear programming algorithm has proved to be superior to all other nonlinear programming algorithms for *every* testproblem and for *every* required accuracy. For instance algorithms based on penalty functions do not use explicitly the given linearity of constraints, hence we expect them to be inferior to projection-like methods in case of application to linearly constrained problems. On the other hand penalty function like methods are expected to behave better on problems which contain (high) nonlinearities.
The comparison of the algorithms of chapters III and IV is still less simple: the recursive quadratic programming algorithms apply linearizations of the first order conditions of penalty functions, while the 2-phase algorithms apply a quadratic loss function in phase I and a linear penalty-like term together with linearizations in phase II.
We argue that a theoretical comparison of algorithms should be supplemented by a *computational comparison*: the algorithms are applied to a set of carefully chosen representative test problems and the efficiency of various methods in solving these test problems is measured in terms of some performance indicators. The representativity of the set of test problems means that both theoretical and practical problems are considered; there should be a significant difference in degree of nonlinearity, dimension, number of constraints and number of active constraints at x^*. To meet all these requirements, we selected a number of test problems from the standard literature such as Himmelblau (1972), Colville (1968), Bus (1976), Cornwell,

Hutchison, Minkoff and Schulz (1978) and Hock and Schittkowski (1979). The main advantage of such a test battery is that a comparison with computational experiments of other researchers is possible at acceptable costs, which seems not to be realizable with randomly generated test problems (Schittkowski (1978a), Rosen-Suzuki (1965), Hillstrom (1977)).
Besides that a drawback of generating test problems randomly is that the resulting 'random' problems can have special properties. This means that the randomly generated problems may not be as random as they are supposed to be. Recently Van Dam and Telgen (1979) reported on this phenomenon for the case of randomly generated polytopes.
The set of test problems selected can be found in appendix A. It is divided into 2 classes: 11 linearly constrained nonlinear programming problems (class LC) and 13 nonlinearly constrained nonlinear programming problems (class NLC). The class LC represents the majority of programming problems evolving from business applications while the class NLC especially corresponds with programming problems in research, development and engineering. See e.g., Bracken and Mc Cormick (1968), Beale (1968) and Lootsma (1976). Within the classes LC and NLC the problems vary in degree of difficulty with respect to the criteria mentioned above. The results obtained illustrate that it is not trivial to predict the influence of these problem characteristics on the performance of algorithms: a few highly nonlinear constraints may cause more difficulties than many active (almost) linear constraints at x^*.
Each of the test problems in appendix A is supplemented with some additional information: a classification number (in accordance with the proposal in Bus (1977)), source reference, number of variables, number of (non) linear constraints, statement of the problem, initial and final value of x and $F(x)$ respectively and special properties of the problem. This information is summarized in tables 6.1 and 6.2.

Table 6.1. Characteristics of the linearly constrained test problems

Problem number	Classification number	# var.	# NL constr.	# LIN constr.	# active constr. at x*
1	GLR-T0-1	3	0	7	1
2	GLR-T0-2	10	0	20	0
3	QLR-T0-1	4	0	7	2
4	SLR-T0-1	2	0	2	0
5	QLR-T0-2	5	0	5	1
6	QLR-T0-3	4	0	10	4
7	GLR-P0-1	5	0	15	4
8	GLR-T0-3	2	0	5	2
9	GLR-P0-2	10	0	13	3
10	SLR-T0-2	5	0	3	3
11	GLR-T0-4	6	0	14	6

Table 6.2. Characteristics of the nonlinearly constrained test problems

Problem number	Classification number	# var.	# NL constr.	# LIN constr.	# active constr. at x*
12	SNR-P0-1	2	1	1	2
13	GNR-T0-1	3	1	3	1
14	LNR-T0-1	2	1	2	2
15	QNR-P0-1	5	6	10	5
16	GNI-P0-1	3	14	6	2
17	SNR-T0-1	2	1	0	0
18	QNR-P0-2	5	2	9	5
19	SNR-P0-2	4	1	3	0
20	GNR-P0-1	15	5	15	11
21	SNR-P0-3	9	12	6	6
22	SNR-P0-4	9	6	0	6
23	QNR-T0-1	4	3	0	2
24	GNR-T0-2	5	3	10	3

VI.2. Termination criteria

In order to compare the robustness and the efficiency of competetive algorithms for constrained nonlinear programming, we have to apply the same termination criteria to all those algorithms. Then convergence to the same point x^* has the same meaning, for instance that the algorithms located x^* with the same relative accuracy. As a consequence of this requirement we cannot apply algorithm dependent termination criteria, such as the norm of the gradient of a penalty function. A choice remains to be made from the following general criteria:

i	(relative) objective function improvement
ii	(relative) stepsize
iii	acceptable constraint violation
iv	satisfaction of the first- or second order Kuhn-Tucker conditions
v	satisfaction of the Jacobian uniqueness conditions.

The criteria i and ii directly concern the iteratively generated sequence $\{x_k\}$ and the corresponding function values. As these criteria are closely related for the problem functions considered, we decided to apply ii, together with iii. The latter criterion is self-evident and especially concerns possible constraint violations in the iteration points of the recursive quadratic programming algorithms. Conditions iv and v are theoretical criteria. For well-behaved problem functions these criteria will be met to within a certain precision as soon as ii and iii are satisfied. Hence we decided to apply ii and iii as termination criteria, which means that x^* is located with a prescribed relative accuracy and that the constraint violations, if any, do not exceed a predesigned bound.
This means for the implementation that we shall define an algorithm to be convergentif both the following criteria are satisfied:

ii $\quad ||x_k - x_{k-1}|| \leq \varepsilon_1(||x_k|| + 1)$

for some pregiven precision parameter $\varepsilon_1 > 0$

and

iii $\quad |c_i(x_k)| \leq \varepsilon_2(||x_k|| + 1)$, for all currently violated

constraints and for some pregiven precision parameter $\varepsilon_2 > 0$

The precision parameters used are $\varepsilon_1 = \varepsilon_2 = 10^{-5}$. The application of iii is preferred as it expresses the accepted violation per constraint. An alternative criterion could be $[\sum_{i\in I(x_{k+1})} |c_i(x_{k+1})|^2]^{\frac{1}{2}} \leq \varepsilon_3$, which, however, allows for different violations of constraints and is only an acceptable criterion if the Lagrange multipliers of the active constraints at x^* are of the same order of magnitude.
As overall convergence is a result of convergence of the algorithms that solve the reduced problems generated, we have to augment the criteria mentioned above by termination criteria for the line search, for the algorithm which solves a linearly constrained reduced problem etc.
The line search in the recursive quadratic programming approach is terminated as soon as the Goldstein and Price test of formulae (3.24), (3.25) is satisfied with $\sigma = 0.01$ or if the distance of the successively generated iteration points along the line is smaller than or equal to ε_4, with $\varepsilon_4 = 10^{-2}$.
This last criterion is also applied in the line search of the 2-phase algorithms, with $\varepsilon_4 = 10^{-4}$. Note that in that case the Goldstein and Price test cannot easily be applied, as the algorithm requires feasible iteration points.
As mentioned in chapter V, the linearly constrained algorithm which solves the reduced problems of the 2-phase algorithm, is terminated if the norm of the search direction p_k, which is a projection of $\nabla F(x_k)$, is small enough: $||p_k|| \leq \varepsilon_5$, with $\varepsilon_5 = 10^{-4}$ *and* at the same time the expected improvement if a constraint is dropped is less than ε_5.

VI.3. Performance indicators

The computational experiments with the algorithms were performed on the DEC 2050 of the Erasmus University Rotterdam, using the FORTRAN-20, version 5 compiler under the operating system TOPS 20, version 3.
However, the computational comparison should be based on generally applicable, preferably machine independent performance indicators.
The general applicability means that indicators such as the number of iterations of the 2-phase algorithms or the number of line searches applied in a recursive quadratic programming algorithm are not suited for our purpose, as they are based on the special structure of a group of closely related algorithms. Furthermore, the required machine independence is pursued to simplify comparison with experiments on other computers. A prerequisity is that all successful computer runs terminate on reaching the same degree of precision, as discussed in section VI.2. Another disturbing factor in the measurement of the indicators is the use of prior information which is not included in the statement of the problem, such as an initial value of the penalty parameter, the initialization of the inverse Hessian approximation etc. We excluded these undesirable influences by using the same parameter values for *all* test problems, except 1 very unfavourable case. This means that all results concern one and the same implementation where no attempt is made to 'optimize' the parameter choices with respect to a particular problem. The parameters actually used derive from preliminary experiments on a larger testset than mentioned in appendix A. The obtained parameter choices are 'safe', in the sense that both the efficiency *and* the robustness of the resulting implementation are satisfactory with respect to the problems investigated.
Possible performance indicators are:

i "Ease of use" of the algorithm.
ii Number of failed runs or robustness of the algorithm.
iii The (standardized) number of CPU-secs, needed to solve problems.
iv The number of problem function evaluations needed to solve problems.

The first mentioned indicator "ease of use", could be measured in terms of: preparation time for executing a problem, the difficulties met in diagnostic work to find the causes of failures, the possibility of human errors (e.g., in analytically supplied derivatives). Though these aspects should

be reweighted in comparing algorithms, they are really programmer dependent as well, hence we consider them to be qualitative indicators which are hard to measure objectively.
The second indicator, *the number of failed runs*, is indicated by simply marking those runs by the character F in the corresponding position of the tables of results. Footnotes are supplied to explain the nature of the failures.
The (standardized) number of CPU-secs can be regarded as an indicator of the 'total effort' to solve a problem. A part of this effort will be reflected in the number of problem function evaluations, the remaining part mainly concerns all kinds of calculations performed during the execution of the program. Usually the time required for I/O-generation is excluded from these figures.
The rationale for using *standardized* CPU-times was to eliminate machine dependent and environment dependent influences such as access to memory and multiprogramming facilities. Usually the standardization of CPU-time is performed by dividing by the number of CPU-secs required for the execution of Colville's standard timing program. This program consists of 10 times inverting a given 40×40 matrix (Colville (1968)). Recent research shows that the desired machine independence of the resulting figures is not realized in this way, mainly because of factors such as the workload of the machine, methods of timing and the use of optimizing compilers (Eason (1977) and Hoffmann (1979)). Moreover Himmelblau (1972) points out that Colville's program is not representative as a 'meaningful standard timing program would be one that somehow takes into account the polymorphic factors of the arithmetic logic, access to memory, storage capacity, allocation of central processing vs. peripheral processing times'.
As a result we decided to apply as performance indicator the *number of problem function evaluations*.
Obviously the execution of an algorithm requires the computation of the value of both the objective function F(x) and the constraint functions $c_i(x)$ at intermediate points. These computations constitute the main costs of solving practical nonlinear programming problems, as these problem function evaluations tend to be expensive compared with the other calculations performed during the execution of the program. Instead of presenting all counted problem function evaluations separately, we shall present the number of *equivalent objective function evaluations*, as suggested in

Staha (1973). This means that all constraint function evaluations are to be converted into objective function evaluations. This conversion is realised using the estimated ratios of the costs of the constraint function evaluations and the concerning objective function evaluation at the point $x^T = (1, \ldots, 1)$. These estimated ratios evolve from the comparison of the required number of CPU-secs. to perform 10^6 function evaluations.
Hence to each successful run a unique number of equivalent objective function evaluations can be assigned. The evaluations of the linear constraints will <u>not</u> be counted at all, as they are much simpler to perform (as an inner product) than the nonlinear constraints.
Hence the ratios needed will only concern the nonlinearly constrained problems 12-24 of appendix A. Table 6.3 contains the estimated ratios.

Table 6.3. Estimated ratios to convert constraint function evaluations into equivalent objective function evaluations.

Problem	Constraint					
	1	2	3	4	5	6
12	.393					
13	2.870					
14	7.517					
15	1.200	1.200	1.249	1.249	1.244	1.244
16	.958	.958	.958	.958	.958	etc.
17	.418					
18	.969	.969				
19	.001					
20	.281	.285	.295	.286	.284	
21	2.674	2.723	2.663	2.656	2.845	2.894
22	2.674	2.723	2.663	2.656	2.845	2.894
23	.697	.784	.701			
24	.263	.128	.158			

Note: Restrictions of the form $l_i \le c_i(x) \le u_i$ are implemented as $c_i(x) - l_i \ge 0$ and $u_i - c_i(x) \ge 0$ respectively. The calculation of the latter expression requires one extra multiplication as compared to the first. We mentioned in the table the arithmetic mean of the ratio's.

VI.4. Results and conclusions

The computational experiments concern the following algorithms:

I Recursive Quadratic Programming with the Oren-Spedicato switch II update formulae

II 2-Phase algorithm with complete linearization

III 2-Phase algorithm with restricted linearization

The Recursive Quadratic Programming implementation applied numerical differentiation using forward difference quotients with step size $\varepsilon_i = 10^{-8}(|x_i| + 0.001)$ for i = 1, ..., n. The 2-Phase implementations behaved better with central difference quotients with $\varepsilon_i = 10^{-4}$. The results obtained are mentioned in table 6.4, in the columns indicated by I - III. The columns indicated by COMET, Lootsma, GREG and GPMNLC are taken from Staha (1973) who used Himmelblau's set of testproblems. Our problems 2, 7, 9, 12, 15, 16, 18, 19 and 20 are the same as problems 17, 10, 4, 24, 11-1, 7, 13, 9 and 18-1 respectively of Himmelblau's collection. Staha's figures concern experiments with numerical derivatives as well. The *COMET* algoritm was proposed in Staha (1973). It is a penalty function algorithm which applies moving truncations of the constraint set to control the convergence towards the optimum. *Lootsma's* implementation concerns a mixed interior point-exterior point penalty function algorithm which applies extrapolations to accelerate the convergence towards the optimum. *GREG* denotes the Generalized Reduced Gradient algorithm of Abadie and Guigou (1969). It is based on linearizations using constrained derivatives to project conjugate directions on the feasible set defined by the linearized constraints. Newton iterations are used as a restoration procedure. *GPMNLC* is an implementation of Rosen's Generalized Projection Method for Nonlinear Constraints. It applies Goldfarb's projection formulae for linear constraints with a quadratic loss penalty function for the nonlinear constraints. The computational results of Staha are mentioned to get more insight in the relative behaviour, the weaknesses and strengths of the algorithms mentioned.

Table 6.4

Numbers of equivalent objective function evaluations

Algorithm	I	II	III	COMET	Lootsma	GREG	GPMNLC
Test problem							
1	53	108	108				
2	58	164	164	338	130	39	46
3	51	91	91				
4	319	341	341				
5	353	186	186				
6	38	42	42				
7	255	109	109	7235	2496	148	20
8	41	29	29				
9	525	727	727	9732	2861	350	134
10	159	350	350				
11	38d	43	43				
12	61	130	127	1326	959	314	599
13	282	1444	1371				
14	1150	1501	1598				
15	982	1463	a	13482	7347	159	7098
16	11755g	7016	6958	3733	12614	699	8368
17	112e	541	545				
18	578	463	392	b	3865	285	12462
19	162	392	392	388	a	1656	c
20	5759	10630	a	110775	46120	2876	12251
21	2621f	44475	43968				
22	4930	21609	21609				
23	210	1846	1660				
24	76	561	561				

Explanation of the abbreviations used

a: no solution reached

b : terminated at infeasible point

c : argument of exponent too large

d : local convergence to $(1, 1\frac{2}{3}, \frac{1}{3}, 0, \frac{1}{3}, 1\frac{2}{3})$

e : local convergence to (-.4536, 0.2105)

f : local convergence to (428.9, -31.1, -.47, 28.7, 149.5, 0.0, 11.5, 38.6, 0.0)

g : no Goldstein/Price test used because of discontinuities in problem functions.

Discussion of the computational results

The results summarized in table 6.4 are an indication of the robustness and the efficiency of the implementations of the algorithms.
The RQP-algorithm seems to have more difficulties to solve the test set in a satisfactory way: two less attractive local solutions were reached (problems 11 and 21). However, it succeeded in determining in problem 17 a local solution that is overlooked by most algorithms. Probably this behaviour of RQP is due to the fact that it allows for infeasible iteration points. On the other hand, RQP is *more efficient*, both for the linearly constrained and the nonlinearly constrained problems. This result is rather surprising as far as the linearly constrained problems are concerned. It might be caused by the linearization of the first order conditions of the exterior penalty function, which forms the basis of this algorithm. For the special case of testproblems with (almost) n active constraints at x^* (problems 3, 7, 8, 10, 11, 12, 14, 15, 16, 18, 20, 22 and 24) RQP clearly improves on the 2-phase algorithms.

The 2-phase algorithm II, which linearizes *all* non linear constraints at every major iteration of phase II, is *more robust* than RQP. This can be explained by the fact that it only uses feasible iteration points. The efficiency of this 2-phase algorithm is clearly improved if only the constraints of the current active set contribute to the definition of the linearly constrained reduced problem of phase II. This can be seen from the 2nd and 3rd column, especially for problems 13, 16, 18, 21 and 23. However, the use of such a simplified reduced problem gives rise to failures on problems 15 and 20. An explanation of this phenomenon is that now the reduced problem has a *fixed* active set. Hence it is not possible to use collected information on the status of the linearized constraints to adjust the active set of the reduced problem.
We recall from the conclusion of table 5.2 that *if* convergence is obtained, then algorithm III needs the *same* number of major iterations as algorithm II in which all nonlinear constraints are linearized. Algorithm III requires more major iterations *to detect* $I(x^*)$ but once it obtains $I(x_k) = I(z^*)$ its convergence is faster. This is in accordance with corollary 3 of theorem 4.7. This observation provides additional motivation to look for a phase I which ends up with $I(z^*)$ as an active set.

The next point of interest is to compare the results of algorithms I, II and III with those given in Staha (1973). When comparing our results

with his numbers, we should be aware of some small differences in the definition of equivalent objective function evaluations. The ratios which Staha uses to convert constraint function evaluations into objective function evaluations concern *groups* of constraints (e.g., *all* nonlinear constraints) whereas we calculated the ratios for *individual* constraints. Another disturbing effect is that Staha's ratios are based on the comparison of 1000 problem function evaluations. This leads to less accurate ratios (in our experiments with 1000 evaluations the ratios varied up to 10%). More stable ratios are obtained by using 10^6 problem function evaluations. A last difference is that we did not count the evaluations of linear constraints.
As a result we have to be careful in drawing conclusions from a comparison with Staha's numbers. But it still seems to be justified to state that algorithms I, II and III, developed and discussed in chapters II - V, behave very well in comparison with COMET, Lootsma and GPMNLC. This conclusion seems not to be valid as far as GREG is concerned. Certainly for this case the relative results are disturbed by one more factor: algorithms I, II and III use *one* fixed set of parameter values to solve the *whole* test set whereas Staha reports that extended diagnostic work was required to prevent failures for GREG. He reports that sometimes an artificial constraint, such as $\sum_{i=1}^{n} x_i \geq -1.0 \times 10^{10}$ had to be added or that the bounds in the problem formulation had to be narrowed.
Hence our general conclusion is that both the Recursive Quadratic Programming algorithm with Self Scaling Update's for the 2nd order information and the two 2-phase algorithms are robust and efficient algorithms with respect to the test set used.

Final remarks

Some final experiments were performed to get better insight in the effect of the suggestion made in Powell (1977) to force positive definiteness of the approximating Hessian matrices, see section III.3.4. Table 6.5 gives some results in terms of numbers of major iterations required to solve some test problems. The test problems selected are expected to be illustrative because of their non convex nature. The intuitive idea behind these experiments is, that it may not be profitable to force positive definite - ness in the earlier iterations, when $I(x^*)$ is not yet determined.

Table 6.5

Iterations required to detect I(x*)

algorithm	Powell			OR-SPED		
Problem	A	B	B-A	A	B	B-A
1	3	11	8	3	12	9
3	11	14	3	3	10	7
4	6	34	28	2	88	86
7	49	∞	∞	23	35	12
9	80	139	59	40	42	2
13	12	22	10	2	16	14
15	4	8	4	19	23	4
19	7	∞	∞	5	32	27

A : # of iterations required to detect I(x*)

B : # of iterations required to obtain convergence

The following tentative conclusions can be drawn from table 6.5. We compare the use of Powell's update with the use of the self scaling update (Oren-Spedicato, switch II) in the context of RQP.

(i) Powell's update necessitates *more iterations* to detect the active set I(x*) (this emerges from the results with test problems 3, 4, 7, 9, 13 and 19, the numbers under the heading A);

(ii) Once I(x*) is known, use of Powell's update gives *faster* convergence (numbers under B-A for test problems 1, 3, 4 and 13);

(iii) Failures can occur if the use of Powell's update forces the algorithm to keep a wrong active set (test problems 7 and 19, where 19 has an unconstrained solution);

(iv) Powell's update should be used as soon as it may be expected that the algorithm detected I(x*), for instance if $I(x_{k+1}) = I(x_k)$.

Appendix A CONSTRAINED NONLINEAR TEST PROBLEMS

This appendix contains the test problems, divided into two classes: 11 linearly constrained nonlinear programming problems of class LC (test problems 1-11) and 13 nonlinearly constrained nonlinear programming problems constituting class NLC (test problems 12-24).

The list of test problems is preceded by a short description concerning the assignment of classification numbers. The classification numbers are in accordance with Bus (1977). Generally, the classification number of a problem has the following form:

O C D - K I - s,

where the symbols have the following meaning:

O - reflects properties of the objective function.

O = S: the objective function is a sum of squares
L: " " " is linear
Q: " " " is quadratic
G: " " " is nonlinear, nonquadratic and no sum of squares.

C - reflects the properties of the constraints.

C = U: unconstrained problem
L: linear constraints
N: at least one nonlinear constraint

D - reflects the differentiability of the problem functions.

D = R: the problem is "regular" in the sense that at least the first and second derivatives of the problem functions exist in the feasible region.
I: the problem is "irregular": there are points in the feasible region where the first and (or) second derivative of one of the problem functions do (does) not exist.

K - reflects the nature of the problem.

K = T: a "theoretical" and well-analysed problem; the solutions are given.

P: a "practical" problem; this means a problem that does not belong to the class categorised by K = T.

I - reflects the highest order of analytically calculated derivatives.

I = 2: first and second order partial derivatives are calculated analytically

1: first order partial derivatives are calculated analytically

0: no partial derivatives are calculated analytically.

s - gives a serial number within the class of test problems identified by OCD-KI.

Test problem 1 Classification number: GLR-T0-1

Source: Box (1966)

Number of variables: 3

Number of nonlinear constraints: 0

Number of linear constraints: 7

Number of active constraints at x^*: 1

Special properties: a nonconvex objective function F(x)

Starting point: $x_0^T = (10,10,10)$ with $F(x_0) = -1000$.

Solution: $x^{*T} = (24,12,12)$ with $F(x^*) = -3456$.

Statement of the problem

$$\text{minimize } F(x) = -x_1x_2x_3$$

subject to

$$42 \geq x_i \geq 0 \qquad i = 1,2,3$$

$$72 - x_1 - 2x_2 - 2x_3 \geq 0.$$

<u>Test problem 2</u> Classification number: GLR-T0-2

Source: Paviani (1969)

Number of variables: 10

Number of nonlinear constraints: 0

Number of linear constraints: 20

Number of active constraints at x^*: 0

Special properties: objective function undefined outside feasible region, free optimum

Starting point: $x_0(i) = 9.0,\ i = 1,\dots,10$ with $F(x_0) = -43.134$

Solution: $x^*(i) = 9.351,\ i = 1,\dots,10$ with $F(x^*) = -45.778$.

Statement of the problem:

$$\text{minimize: } F(x) = \sum_{i=1}^{10} \{[\ln(x_i - 2)]^2 + [\ln(10 - x_i)]^2\} - \left(\prod_{i=1}^{10} x_i\right)^{0.2}$$

subject to:

$$2.001 < x_i < 9.999 \qquad i = 1,\dots,10$$

<u>Test problem 3</u> Classification number: QLR-T0-1

Source: Murtagh and Sargent (1969)

Number of variables: 4

Number of nonlinear constraints: 0

Number of linear constraints: 7

Number of active constraints at x^*: 2

Special properties: a nonconvex objective function $F(x)$

Starting point: $x_0^T = (0.5,0.5,0.5,0.5)$ with $F(x_0) = -1.25$

Solution: $x^{*T} = (0.272,2.090,0.000,0.545)$ with $F(x^*) = -4.682$

Statement of the problem:

$$\text{minimize } F(x) = -x_1-3x_2+x_3-x_4 + \tfrac{1}{2}(2x_1^2-2x_1x_3+x_2^2+2x_3^2+2x_3x_4+x_4^2)$$

subject to:

$$x_i \geq 0,\ i = 1,\dots,4$$
$$-x_1 - 2x_2 - x_3 - x_4 + 5 \geq 0$$
$$-3x_1 - x_2 - 2x_3 + x_4 + 4 \geq 0$$
$$x_2 + 4x_3 - 1.5 \geq 0.$$

Test problem 4 Classification number: SLR-T0-1

Source: Schweigman (1974)

Number of variables: 2

Number of nonlinear constraints: 0

Number of linear constraints: 2

Number of active constraints at x^*: 0

Special properties: this is a linearly constrained version of the well-known Rosenbrock problem

Starting point: $x_0^T = (-1.2, 1.0)$ with $F(x_0) = 24.2$

Solution: $x^{*T} = (1.0, 1.0)$ with $F(x^*) = 0.0$

Statement of the problem:

$$\text{minimize } F(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

subject to:

$$\frac{1}{3} x_1 + x_2 \geq -\frac{1}{10}$$

$$-\frac{1}{3} x_1 + x_2 \geq -\frac{1}{10} .$$

Test problem 5 Classification number: QLR-T0-2

Source: Stoer (1971)

Number of variables: 5

Number of nonlinear constraints: 0

Number of linear constraints: 5

Number of active constraints at x^*: 1

Special properties: -

Starting point: $x_0^T = (1,1,1,1,1)$ with $F(x_0) = 12048$

Solution: $x^{*T} = (1,2,-1,3,-4)$ with $F(x^*) = 0$

Statement of the problem:

minimize: $F(x) = x^T D^T Dx - 2d^T Dx + d^T d$

subject to:

$$x_1 + x_2 + x_3 + x_4 + x_5 \le 5,$$
$$10x_1 + 10x_2 - 3x_3 + 5x_4 + 4x_5 \ge 20,$$
$$8x_1 - x_2 + 2x_3 + 5x_4 - 3x_5 \le 40,$$
$$8x_1 - x_2 + 2x_3 + 5x_4 - 3x_5 \ge 11,$$
$$4x_1 + 2x_2 - 3x_3 + 5x_4 - x_5 \le 30,$$

where D and d are given as

$$D = \begin{pmatrix} -74 & 80 & 18 & -11 & -4 \\ 14 & -69 & 21 & 28 & 0 \\ 66 & -72 & -5 & 7 & 1 \\ -12 & 66 & -30 & -23 & 3 \\ 3 & 8 & -7 & -4 & 1 \\ 4 & -12 & 4 & 4 & 0 \end{pmatrix} \qquad d = \begin{pmatrix} 51 \\ -61 \\ -56 \\ 69 \\ 10 \\ -12 \end{pmatrix}$$

<u>Test problem 6</u> Classification number: QLR-T0-3

Source: Konno (1976)

Number of variables: 4

Number of nonlinear constraints: 0

Number of linear constraints: 10

Number of active constraints at x^*: 4

Special properties: 4 active constraints at x^*, nonconvex objective function

Starting point: $x_0^T = (0,0,0,0)$ with $F(x_0) = 0$

Solution: $x^{*T} = (0,3,0,4)$ with $F(x^*) = -15$.

Statement of the problem:

$$\text{minimize } F(x) = x_1 - x_2 - x_3 - x_1x_3 + x_2x_3 - x_2x_4 + x_1x_4$$

subject to:

$$8 - x_1 - 2x_2 \geq 0$$
$$12 - 4x_1 - x_2 \geq 0$$
$$12 - 3x_1 - 4x_2 \geq 0$$
$$8 - 2x_3 - x_4 \geq 0$$
$$8 - x_3 - 2x_4 \geq 0$$
$$5 - x_3 - x_4 \geq 0$$
$$x_i \geq 0 \quad i = 1,2,3,4.$$

Test problem 7 Classification number: GLR-P0-1

Source: Shell Development Co., cited in Colville (1968)

Number of variables: 5

Number of nonlinear constraints: 0

Number of linear constraints: 15

Number of active constraints at x^*: 4

Special properties: -

Starting point: $x_0^T = (0.0,0.0,0.0,0.0,1.0)$ with $F(x_0) = 20.000$

Solution: $x^{*T} = (0.3000, 0.3335,0.4000,0.4285,0.224)$ with $F(x^*) = -32.349$

Statement of the problem:

$$\text{minimize } F(x) = \sum_{j=1}^{5} e_j x_j + \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_i x_j + \sum_{j=1}^{5} d_j x_j^3$$

subject to:

$$\sum_{j=1}^{5} a_{ij} x_j \geq b_i \qquad i = 1,...,10$$

$$x_j \geq 0 \qquad j = 1,...,5.$$

The coefficients are given in the next tables

j		1	2	3	4	5
e_j		-15	-27	-36	-18	-12
c_{ij}	1	30	-20	-10	32	-10
	2	-20	39	-6	-31	32
	3	-10	-6	10	-6	-10
	4	32	-31	-6	39	-20
	5	-10	32	-10	-20	30
d_j		4	8	10	6	2

a_{ij}	1	-16	2	0	1	0	-40
	2	0	-2	0	0.4	2	-2
	3	-3.5	0	2	0	0	-.25
	4	0	-2	0	-4	-1	-4
	5	0	-9	-2	1	-2.8	-4
	6	2	0	-4	0	0	-1
	7	-1	-1	-1	-1	-1	-40
	8	-1	-2	-3	-2	-1	-60
	9	1	2	3	4	5	5
	10	1	1	1	1	1	1
							b_i

<u>Test problem 8</u> Classification number: GLR-T0-3

Source: Betts (1977)

Number of variables: 2

Number of nonlinear constraints: 0

Number of linear constraints: 5

Number of active constraints at x^*: 2

Special properties: nonconvex objective function

Starting point: $x_0^T = (1,0.5)$ with $F(x_0) = -0.01336459$

Solution: $x^{*T} = (3,\sqrt{3})$ with $F(x^*) = -1.$

Statement of the problem:

$$\text{minimize } F(x) = \frac{1}{27\sqrt{3}}((x_1-3)^2-9)x_2^3$$

subject to:

$$\frac{x_1}{\sqrt{3}} - x_2 \geq 0$$

$$x_1 + x_2\sqrt{3} \geq 0$$

$$-x_1 - x_2\sqrt{3} + 6 \geq 0$$

$$x_1 \geq 0,\ x_2 \geq 0.$$

Test problem 9 Classification number: GLR-P0-2

Source: Bracken and McCormick (1968)

Number of variables: 10

Number of nonlinear constraints: 0

Number of linear constraints: 13

Number of active constraints at x^*: 3

Special properties: this is a problem in the chemical equilibrium at constant temperature and pressure, infeasible starting point

Starting point: $x_0(i) = 0.1 \qquad i = 1,\dots,10$

with $F(x_0) = -\ 20.961$

Solution: $x^{*T} = [0.0406 \quad 0.1477 \quad 0.7832 \quad 0.0014 \quad 0.4853$
$0.0007 \quad 0.0274 \quad 0.0180 \quad 0.0375 \quad 0.0969]^T$

with $F(x^*) = -\ 47.761$

Statement of the problem

$$\text{minimize: } F(x) = \sum_{i=1}^{10} x_i \left(c_i + \ln \frac{x_i}{\sum_{j=1}^{10} x_j} \right)$$

subject to:

$$x_1 + 2x_2 + 2x_3 + x_6 + x_{10} - 2 = 0$$
$$x_4 + 2x_5 + x_6 + x_7 - 1 = 0$$
$$x_3 + x_7 + x_8 + 2x_9 + x_{10} - 1 = 0$$
$$x_i \geq 0 \qquad i = 1,\dots,10$$

where

$c_1 = -6.089$	$c_2 = -17.164$	$c_3 = -34.054$	$c_4 = -5.914$
$c_5 = -24.721$	$c_0 = -14.986$	$c_7 = -24.100$	$c_8 = -10.708$
$c_9 = -26.662$	$c_{10} = -22.179$		

Test problem 10 Classification number: SLR-T0-2

Source: Huang and Aggerwal (1975)

Number of variables: 5

Number of nonlinear constraints: 0

Number of linear constraints: 3

Number of active constraints at x^*: 3

Special properties: -

Starting point: $x_0^T = (35,-31,11,5,-5)$ with $F(x_0) = 17416$

Solution: $x^{*T} = (1,1,1,1,1)$ with $F(x^*) = 0$

Statement of the problem:

$$\text{minimize } F(x) = (x_1-x_2)^2 + (x_2-x_3)^2 + (x_3-x_4)^4 + (x_4-x_5)^2$$

subject to:

$$x_1 + 2x_2 + 3x_3 - 6 = 0$$
$$x_2 + 2x_3 + 3x_4 - 6 = 0$$
$$x_3 + 2x_4 + 3x_5 - 6 = 0$$

Test problem 11 Classification number: GLR-T0-4

Source: Hsia (1975)

Number of variables: 6

Number of nonlinear constraints: 0

Number of linear constraints: 14

Number of active constraints at x^*: 6

Special properties: one constraint is redundant

Starting point: $x_0^T = (1,2,0,0,0,2)$ with $F(x_0) = 6$

Solution: $x^{*T} = (0, \frac{4}{3}, \frac{5}{3}, 1, \frac{2}{3}, \frac{1}{3})$ with $F(x^*) = \frac{19}{3}$

Statement of the problem:

$$\text{minimize } F(x) = x_1 + 2x_2 + 4x_5 + e^{x_1 x_4}$$

subject to:

$$\begin{aligned}
& x_1 + 2x_2 + 5x_5 - 6 = 0 \\
& x_1 + x_2 + x_3 - 3 = 0 \\
& x_4 + x_5 + x_6 - 2 = 0 \\
& x_1 + x_4 - 1 = 0 \\
& x_2 + x_5 - 2 = 0 \\
& x_3 + x_6 - 2 = 0 \\
& x_1 \le 1 \\
& x_4 \le 1 \\
& x_i \ge 0 \qquad i = 1,\dots,6.
\end{aligned}$$

Test problem 12 Classification number: SNR-P0-1

Source: Bracken and McCormick (1968)

Number of variables: 2

Number of nonlinear constraints: 1

Number of linear constraints: 1

Number of active constraints at x^*: 2

Special properties: -

Starting point: $x_0^T = (2,2)$ (infeasible) with $F(x_0) = 1$

Solution: $x^{*T} = (1,1)$ with $F(x^*) = 1$

Statement of the problem:

$$\text{minimize } F(x) = (x_1 - 2)^2 + (x_2 - 1)^2$$

subject to:

$$- x_1^2 + x_2 \geq 0$$

$$- x_1 - x_2 + 2 \geq 0$$

Test problem 13 Classification number: GNR-T0-1

Source: Davies (1968)

Number of variables: 3

Number of nonlinear constraints: 1

Number of linear constraints: 3

Number of active constraints at x^*: 1

Special properties: a nonconvex objective function F(x)

Starting point: $x_0^T = (1,1,1)$ with $F(x_0) = - 1.$

Solution: $x^{*T} = (4.00, 2.828, 2.0)$ with $F(x^*) = - 22.627$

Statement of the problem:

$$\text{minimize } F(x) = - x_1 x_2 x_3$$

subject to:

$$x_i \geq 0 \qquad i = 1,2,3$$

$$48 - x_1^2 - 2x_2^2 - 4x_3^2 \geq 0.$$

Test problem 14 Classification number: LNR-T0-1

Source: Fiacco and McCormick (1968)

Number of variables: 2

Number of nonlinear constraints: 1

Number of linear constraints: 2

Number of active constraints at x^*: 2

Special properties: x^* is an irregular point of the feasible region

Starting point: $x_0^T = (0.25, 0.25)$ with $F(x_0) = -0.25$

Solution: $x^{*T} = (1,0)$ with $F(x^*) = -1$.

Statement of the problem:

$$\text{minimize } F(x) = -x_1$$

subject to:

$$x_1 \geq 0$$

$$x_2 \geq 0$$

$$(1-x_1)^3 - x_2 \geq 0.$$

Test problem 15 Classification number: QNR-PO-1

Source: Proctor and Gamble Co., cited in Colville (1968)

Number of variables: 5

Number of nonlinear constraints: 6

Number of linear constraints: 10

Number of active constraints at x^*: 5

Special properties: singular Hessian matrix of $F(x)$

Starting point: $x_0^T = (78.62, 33.44, 31.07, 44.18, 35.22)$ with $F(x_0) = -30367$

Solution: $x^{*T} = (78.000, 33.000, 29.995, 45.000, 36.776)$ with $F(x^*) = -30665.5$

Statement of the problem:

$$\begin{aligned}\text{minimize } F(x) = {} & 5.3578547x_3^2 + 0.8356891x_1x_5 \\ & + 37.293239x_1 - 40792.141\end{aligned}$$

subject to:

$$92 \ge 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 \ge 0$$

$$110 \ge 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 \ge 90$$

$$25 \ge 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 \ge 20$$

$$78 \le x_1 \le 102$$

$$33 \le x_2 \le 45$$

$$27 \le x_3 \le 45$$

$$27 \le x_4 \le 45$$

$$27 \le x_5 \le 45$$

Test problem 16 Classification number: GNI-PO-1

Source: Colville (1968)

Number of variables: 3

Number of nonlinear constraints: 14

Number of linear constraints: 6

Number of active constraints at x^*: 2

Special properties: a test problem in which functions are described by a self-contained computer subroutine.

Starting point: $x_0^T = (1745,12000,110)$ with $F(x_0) = 868.6458$

Solution: $x^{*T} = (1728.37,16000,9813)$ with $F(x^*) = 1162.036$

Statement of the problem:

maximize $F(x) = 0.063y_2y_5 - 5.04x_1 - 3.36y_3 - 0.035x_2 - 10x_3$

subject to:

$0 \le x_1 \le 2000$

$0 \le x_2 \le 16,000$

$0 \le x_3 \le 120$

$0 \le y_2 \le 5000$

$0 \le y_3 \le 2000$

$85 \le y_4 \le 93$

$90 \le y_5 < 95$

$3 \le y_6 \le 12$

$0.01 \le y_7 \le 4$

$145 \le y_8 \le 162$

The variables y_2 to y_8 are calculated in the following subroutine:

```
    Y(2) = 1.6*X(1)
 10 Y(3) = 1.22*Y(2) - X(1)
    Y(6) = (X(2) + Y(3))/X(1)
    Y(2)CALC = X(1)*(112. + 13.167*Y(6) - 0.6667*Y(6)**2)/100.
    IF(ABS(Y2CALC - Y(2)) - 0.001)30,30,20
 20 Y(2) = Y2CALC
    GO TO 10
 30 CONTINUE
    Y(4) = 93.
100 Y(5) = 86.35 + 1.098*Y(6) - 0.038*Y(6)**2 + 0.325*(Y(4) - 89.)
    Y(8) = -133. + 3.*Y(5)
    Y(7) = 35.82 - 0.222*Y(8)
    Y4CALC = 98000.*X(3)/Y(2)*Y(7) + X(3)*1000.)
    IF(ABS(Y4CALC - Y(4)) - 0.0001)300,300,200
200 Y(4) = Y4CALC
    GO TO 100
300 CONTINUE
```

Test problem 17

Classification number: SNR-T0-1

Source: Schweigman (1974)

Number of variables: 2

Number of nonlinear constraints: 1

Number of linear constraints: 0

Number of active constraints at x^*: 0

Special properties: this test problem is a nonlinearly constrained Rosenbrock function with a local minimum and a free global minimum

Starting point: $x_0^T = (-1.2, 1.)$ with $F(x_0) = 24.2$

Solution: $x^{*T} = (1.0, 1.0)$ with $F(x^*) = 0.0$

Statement of the problem:

$$\text{minimize } F(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

subject to:

$$x_1^2 + x_2^2 \geq 0.25$$

<u>Test problem 18</u> Classification number: QNR-P0-2

Source: Box (1965)

Number of variables: 5

Number of nonlinear constraints: 2

Number of linear constraints: 9

Number of active constraints at x^*: 5

Special properties: this is an example of determining parameters in highly nonlinear differential equations from experimental data. The objective function was the sum of squared residuals between experimental data and numerically integrated solutions of the differential equations.

Starting point: $x_0^T = (2.52, 2., 37.5, 9.25, 6.8)$ with $F(x_0) = 2,351,243.5$

Solution: $x^{*T} = (4.538, 2.400, 60.000, 9.300, 7.000)$ with $F(x^*) = 5,280,254$

Statement of the problem:

$$\text{maximize } F(x) = (c_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_5)x_1 - 24,345$$

subject to:

$$0 \le y \le 277,200$$

$$y = (c_6 + c_7x_2 + c_8x_3 + c_9x_4 + c_{10}x_5)x_1$$

$$0 \le x_1$$

$$1.2 \le x_2 \le 2.4$$

$$20 \le x_3 \le 60$$

$$9 \le x_4 \le 9.3$$

$$6.5 \le x_5 \le 7$$

Calculation of the c_i's:

$c_1 = -8,720,288.849$	$c_6 = -326,669.5104$
$c_2 = 150,512.5253$	$c_7 = 7,390.68412$
$c_3 = -156.6950325$	$c_8 = -27.8986976$
$c_4 = 476,470.3222$	$c_9 = 16,643.076$
$c_5 = 729,482.8271$	$c_{10} = 30,988.146$

Test problem 19 Classification number: SNR-P0-2

Source: Himmelblau and Yates (1968)

Number of variables: 4

Number of nonlinear constraints: 1

Number of linear constraints: 3

Number of active constraints at x^*: 0

Special properties: the constraint $x_3 \geq 0$ is added, free optimum

Starting point: $x_0^T = (2.,4.,0.04,2.)$ with $F(x_0) = 0.9819$

Solution: $x^{*T} = (12.277,4.632,0.313,2.029)$ with $F(x^*) = 0.0075$

Statement of the problem:

minimize: $$F(x) = \sum_{i=1}^{19} (y_{i,cal} - y_{i,obs})^2$$

$$y_{i,cal} = \frac{x_3 \beta^{x_2} \left(\frac{x_2}{6.2832}\right)^{\frac{1}{2}} \left(\frac{c_i}{7.658}\right)^{(x_2-1)} \exp\left(x_2 - \beta \frac{c_i x_2}{7.658}\right)}{1 + \frac{1}{12x_2}}$$
$$+ \frac{(1-x_3)\left(\frac{\beta}{x_4}\right)^{x_1} \left(\frac{x_1}{6.2832}\right)^{\frac{1}{2}} \left(\frac{c_i}{7.658}\right)^{x_1-1} \exp\left(x_1 - \beta \frac{c_i x_1}{7.658 x_4}\right)}{1 + \frac{1}{12x_1}}$$

where $\beta = x_3 + (1 - x_3)x_4$. (Note: The c_i and $y_{i,obs}$ are given in the accompanying table).

subject to:

$x_3 + (1 - x_3)x_4 \geq 0$

$x_4 \geq 0$

$0 \leq x_3 \leq 1$

c_i and $y_{i,obs}$ for Test Problem 19

i	c	$y_{i,obs}$	i	c	$y_{i,obs}$
1	0.1	0.00189	11	10	0.702
2	1	0.1038	12	11	0.528
3	2	0.268	13	12	0.385
4	3	0.506	14	13	0.257
5	4	0.577	15	14	0.159
6	5	0.604	16	15	0.0869
7	6	0.725	17	16	0.0453
8	7	0.898	18	17	0.01509
9	8	0.947	19	18	0.00189
10	9	0.845			

<u>Test problem 20</u> Classification number: GNR-P0-1

Source: Shell Development Co., cited in Colville (1968)

Number of variables: 15

Number of constraints: 5

Number of linear constraints: 15

Number of active constraints at x^*: 11

Special properties: this problem is the dual of problem 7

Starting point: $x_0(i) = 0.0001 \quad i = 1,\dots,15,\ i \neq 7$

$x_0(7) = 60$

with $F(x_0) = 2400.01$

Solution: $x^{*T} = (0.0000, 0.0000, 5.1740, 0.0000, 3.0611, 11.8395, 0.0000, 0.0000, 0.1039, 0.0000, 0.3000, 0.3335, 0.4000, 0.4283, 0.2240)$ with

$F(x^*) = 32.386$

Statement of the problem:

$$\text{minimize } F(x) = -\sum_{i=1}^{10} b_i x_i + \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_{10+i} x_{10+j} + 2 \sum_{j=1}^{5} d_j x_{10+j}^3$$

subject to:

$$2 \sum_{i=1}^{5} c_{ij} x_{10+i} + 3 d_j x_{10+j}^2 + e_j - \sum_{i=1}^{10} a_{ij} x_j \geq 0, \quad j = 1,\dots,5$$

$$x_i \geq 0, \quad i = 1,\dots,15$$

The coefficients e_j, c_{ij}, d_j, a_{ij} and b_j are given in problem 7.

Test problem 21 Classification number: SNR-P0-3

Source: Ballintijn, van der Hoek and Hooykaas (1978)

Number of variables: 9

Number of nonlinear constraints: 12

Number of linear constraints: 6

Number of active constraints at x^*: 6

Special properties: singular Hessian matrix, infeasible starting point

Starting point: $x_0^T = (300.,-100.,-.1997,-127.,-151.,379.,421.,460.,426.)$
with $F(x_0) = 752,888.0$

Solution: $x^{*T} = (523.3,-156.9,-.1997,29.60,86.61,47.32,26.23,22.91,39.47)$
with $F(x^*) = 13,390.1$

Statement of the problem:

$$\text{minimize } F(x) = \sum_{i=4}^{9} x_i^2$$

subject to:

$$\begin{aligned}
&x_i \geq 0, \quad i = 4,\dots,9\\
&x_1 + x_2 \exp(-5x_3) + x_4 - 127 \geq 0\\
&x_1 + x_2 \exp(-3x_3) + x_5 - 151 \geq 0\\
&x_1 + x_2 \exp(-x_3) + x_6 - 379 \geq 0\\
&x_1 + x_2 \exp(x_3) + x_7 - 421 \geq 0\\
&x_1 + x_2 \exp(3x_3) + x_8 - 460 \geq 0\\
&x_1 + x_2 \exp(5x_3) + x_9 - 426 \geq 0\\
&-x_1 - x_2 \exp(-5x_3) + x_4 + 127 \geq 0\\
&-x_1 - x_2 \exp(-3x_3) + x_5 + 151 \geq 0\\
&-x_1 - x_2 \exp(-x_3) + x_6 + 379 \geq 0\\
&-x_1 - x_2 \exp(x_3) + x_7 + 421 \geq 0\\
&-x_1 - x_2 \exp(3x_3) + x_8 + 460 \geq 0\\
&-x_1 - x_2 \exp(5x_3) + x_9 + 426 \geq 0
\end{aligned}$$

<u>Test problem 22</u> Classification number: SNR-PO-4

Source: Ballintijn, van der Hoek and Hooykaas (1978)

Number of variables: 9

Number of nonlinear constraints: 6

Number of linear constraints: 0

Number of active constraints at x^*: 6

Special properties: the same problem statement as problem 21, now without bounds and with the first six constraints as equality constraints

Starting point: $x_0^T = (300.,-100.,-.1997,-127.,-151.,379.,421.,460.,426.)$ with $F(x_0) = 752,888.0$

Solution: $x^{*T} = (523.3,-156.9,-.1997,29.60,-86.61,47.32,26.23,22.91,-39.47)$ with $F(x^*) = 13,390.1$

Statement of the problem:

$$\text{minimize } F(x) = \sum_{i=4}^{9} x_i^2$$

subject to:

$$x_1 + x_2 \exp(-5x_3) + x_4 - 127 = 0$$
$$x_1 + x_2 \exp(-3x_3) + x_5 - 151 = 0$$
$$x_1 + x_2 \exp(-x_3) + x_6 - 379 = 0$$
$$x_1 + x_2 \exp(x_3) + x_7 - 421 = 0$$
$$x_1 + x_2 \exp(3x_3) + x_8 - 460 = 0$$
$$x_1 + x_2 \exp(5x_3) + x_9 - 426 = 0$$

Test problem 23 Classification number: QNR-T0-1

Source: Betts (1977)

Number of variables: 4

Number of nonlinear constraints: 3

Number of linear constraints: 0

Number of active constraints at x^*: 2

Special properties: feasible starting point

Starting point: $x_0^T = (0,0,0,0)$ with $F(x_0) = 0$

Solution: $x^{*T} = (0,1,2,-1)$ with $F(x^*) = -44$.

Statement of the problem:

$$\text{minimize } F(x) = x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4$$

subject to:

$$8 - x_1^2 - x_2^2 - x_3^2 - x_4^2 - x_1 + x_2 - x_3 + x_4 \geq 0$$
$$10 - x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 + x_1 + x_4 \geq 0$$
$$5 - 2x_1^2 - x_2^2 - x_3^2 - 2x_1 + x_2 + x_4 \geq 0$$

Test problem 24 Classification number: GNR-T0-2

Source: Powell (1978)

Number of variables: 5

Number of nonlinear constraints: 3

Number of linear constraints: 10

Number of active constraints at x^*: 3

Special properties: infeasible starting point

Starting point: $x_0^T = (-2,2,2,-1,-1)$ with $F(x_0) = -0.49966$

Solution: $x^{*T} = (-1.7171,1.5957,1.8272,-0.7636,-0.7636)$ with $F(x^*) = 0.053976$

Statement of the problem:

$$\text{minimize } F(x) = e^{x_1x_2x_3x_4x_5} - 0.5(x_1^3 + x_2^3 + 1)^2$$

subject to:

$$x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$$
$$x_2x_3 - 5x_4x_5 = 0$$
$$x_1^3 + x_2^3 + 1 = 0$$
$$-2.3 \leq x_i \leq 2.3 \qquad i = 1,2$$
$$-3.2 \leq x_i \leq 3.2 \qquad i = 3,4,5.$$

Appendix B UNCONSTRAINED NONLINEAR TEST PROBLEMS

Test problems for ch. II: A computational comparison of self scaling variable metric algorithms.

Problem 1,2,3,4 Classification number: SUR-T1-1a,b,c,d

Source: Rosenbrock (1961) (a family of Rosenbrock functions)

Number of variables: 2

Number of nonlinear constraints: 0

Number of linear constraints: 0

Special properties: nonconvex, ill conditioned at x^*, $c = 1, 10^2, 10^4, 10^6$

Starting point: $x_0^T = (-1.2, 1)$ with $F(x_0) = .1936c + 4.84$

Solution: $x^{*T} = (1,1)$ with $F(x^*) = 0$

Statement of the problem:

$$\text{minimize } F(x) = c(x_2 - x_1^2)^2 + (1 - x_1)^2$$

Problem 5,6 Classification number: SUR-T1-2a,b

Source: Rosenbrock (1961) (multidimensional banana functions)

Number of variables: n

Number of nonlinear constraints: 0

Number of linear constraints: 0

Special properties: nonconvex, high dimensional, n = 10,30

Starting point: $x_0^T = (-1.2, 1, -1.2, 1, \ldots, -1.2, 1)$

Solution: $x^{*T} = (1, 1, \ldots, 1)$ with $F(x^*) = 0$

Statement of the problem:

$$\text{minimize } F(x) = \sum_{k=1}^{n-1} \{100(x_{k+1} - x_k^2)^2 + (1 - x_k)^2\}$$

Problem 7,8,9 Classification number: GUR-T1-1a,b,c

Source: Oren (1973) (Oren's quartic function)

Number of variables: n

Number of nonlinear constraints: 0

Number of linear constraints: 0

Special properties: convex, various dimensions, n = 2,10,30

Starting point: $x_0^T = (1,1,\ldots,1)$ with $F(x_0) = \frac{1}{4}n^2(n+1)^2$

Solution: $x^{*T} = (0,0,\ldots,0)$ with $F(x^*) = 0$.

Statement of the problem:

$$\text{minimize } F(x) = (x^T A x)^2,$$

with

$$A = \begin{pmatrix} 1 & & & & 0 \\ & 2 & & & \\ & & 3 & & \\ & & & \ddots & \\ 0 & & & & n \end{pmatrix}$$

Problem 10,11,12 Classification number: QUR-T1-1a,b,c

Source: Oren (1973) (Hilbert problems)

Number of variables: n

Number of nonlinear constraints: 0

Number of linear constraints: 0

Special properties: convex, quadratic, ill-conditioned, n = 2,4,6

Starting point: $x_0(k) = \frac{-4}{k}$, $k = 1,\ldots,n$

Solution: $x^{*T} = (1,1,\ldots,1)$ with $F(x^*) = 0$

Statement of the problem:

$$\text{minimize } F(x) = x^T A_n x$$

with

$$A_n(i,j) = \frac{1}{i+j-1}, \qquad i,j = 1,\ldots,n.$$

The matrix A is an n×n segment of the Hilbert matrix, whose condition number increases rapidly with n.

Appendix C SPECIAL MATRICES

1. A lower *trapezoidal* matrix L is a m×n (m≥n) matrix $\ell(i,j)$, for which the following relation holds:

$$\ell(i,j) = 0 \text{ for } j = i+1 \text{ to } n, \text{ and } i = 1 \text{ to } n.$$

2. A *lower triangular* matrix L is a n×n matrix $\ell(i,j)$ for which the following relation holds:

$$\ell(i,j) = 0 \text{ for } j = i+1 \text{ to } n, \text{ and } i = 1 \text{ to } n.$$

3. An *upper triangular* matrix is a transposed lower triangular matrix.
4. A *unit lower (or upper) triangular matrix* is a lower (or upper) triangular matrix with all diagonal elements equal to 1.
5. A *special triangular matix* $M(\underline{p},\underline{b},\underline{c})$ is a triangular matrix $m(i,j)$ for which the following relation holds:

$$m(i,j) = 0 \text{ for } j = i+1 \text{ to } n, \text{ and } i = 1 \text{ to } n$$
$$m(i,j) = c(i) \text{ for } j = i, \text{ and } i = 1 \text{ to } n$$
$$m(i,j) = p(i)b(j) \text{ for } j = 1 \text{ to } i-1, \text{ and } i = 1 \text{ to } n.$$

6. A *special matrix* E_k is a n×n triangular matrix $e(i,j)$, with as only non-zero off diagonal elements $e(i,k)$, $i = k+1$ to n. Hence:

$$e(i,j) = 1 \text{ for } j = i, \text{ and } i = 1 \text{ to } n$$
$$e(i,j) = 0 \text{ for } j = i+1 \text{ to } n, \text{ and } i = 1 \text{ to } n$$
$$e(i,j) = 0 \text{ for } i = j+1 \text{ to } n, \text{ and } j = 1 \text{ to } k-1$$
$$e(i,j) = 0 \text{ for } i = j+1 \text{ to } n, \text{ and } j = k+1 \text{ to } n.$$

7. A *diagonal matrix* D is a n×n square matrix $d(i,j)$, for which the following relation holds:

$$d(i,j) = 0 \text{ for } j \neq i, \text{ and } i,j = 1 \text{ to } n.$$

The *unit matrix* I or I_n is a n×n diagonal matrix with all diagonal elements equal to 1.
8. A *permutation matrix* P is a n×n matrix $p(i,j)$, with $p(i,j) \in \{0,1\}$ and precisely one 1 in every row and column. Hence:

$$\sum_{i=1}^{n} p(i,j) = 1 \qquad j = 1 \text{ to } n$$
$$\sum_{j=1}^{n} p(i,j) = 1 \qquad i = 1 \text{ to } n$$

$$p(i,j) = 0 \text{ or } 1.$$

Appendix D

STRUCTURE OF THE IMPLEMENTATION OF THE 2-PHASE ALGORITHM

This appendix contains a detailed description of the last phase in the development of algorithms: the implementation as a computer code.

Point of departure for the implementation of the discussed 2-phase algorithms was the VANOP-code, presented in Ballintijn, Van der Hoek and Hooykaas (1978), which was further developed and adjusted in accordance with the theory of Ch. IV and Ch. V. Figure D.1 gives a flowchart of the implementation, followed by a description of the mentioned subroutines.

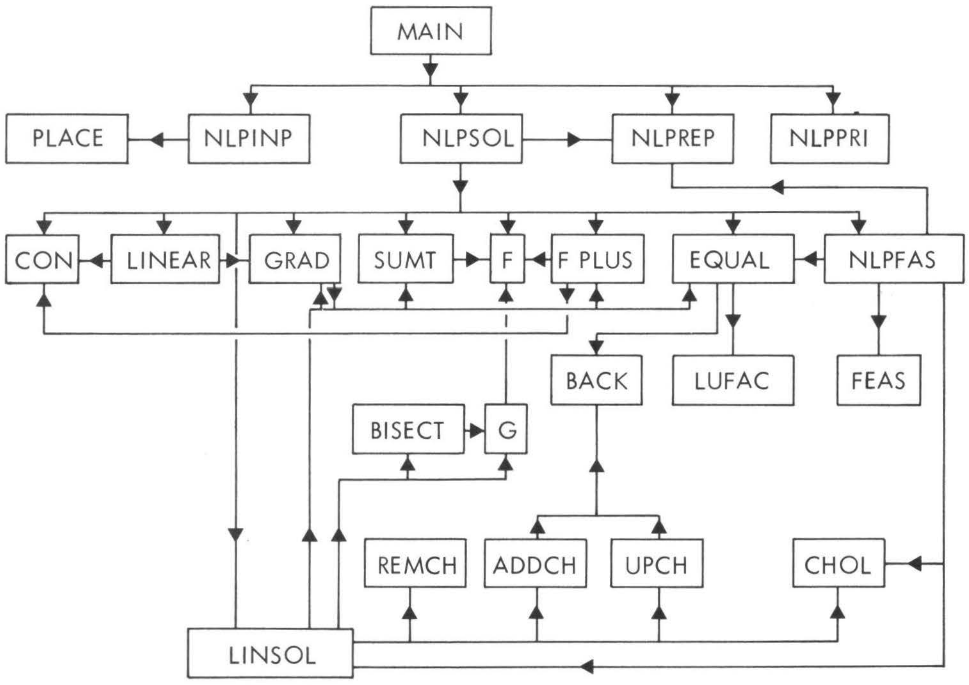


Figure D.1 Structure of the computer program for 2-phase algorithms

Description of the subroutines

NLPINP	- reads the input: information on functions, names of constraints and variables, precision parameters
NLPPRI	- prints the data that are read by NLPINP
NLPSOL	- controls the algorithm. First it finds a feasible point with respect to the linear constraints, then phase I and phase II as described in Ch. IV are executed
LINSOL	- solves the linearly constrained nonlinear reduced problems
NLPREP	- prints intermediate and final reports of the problem statistics
NLPFAS	- performs the calculations for phase I
LINEAR	- linearizes nonlinear constraints
CON	- function subroutine containing the nonlinear constraints
SUMT	- function subroutine in which the penalty function of phase I is calculated
FPLUS	- function subroutine that calculates the objective function of the phase II - reduced problems
PLACE	- recognizes the names of the variables and constraints of the input subroutine
BISECT	- a bisection-like linesearch subroutine
F	- function subroutine to calculate the objective function
G	- function subroutine that evaluates function - values along the search direction
GRAD	- calculates derivatives numerically
CHOL	- determines the Cholesky decomposition of a positive definite matrix
ADDCH	- updates the Cholesky-factors of the matrix $N_k^T H_k N_k$ if a constraint is added to the active set
REMCH	- updates the Cholesky-factors of $N_k^T H_k N_k$ if a constraint is removed from the active set
UPCHK	- updates the Cholesky-factors of a matrix if a rank 1 matrix of the form vv^T is added or substracted
BACK	- solves x from the linear system Lx = b where L is triangular
FEAS	- constructs the penaly function of (5.11)
EQUAL	- projects a point in the intersection of a set of equality constraints, using LU-decomposition
LUFAC	- calculates the LU-factors of a matrix

Appendix E

STRUCTURE OF THE IMPLEMENTATION OF THE RECURSIVE QUADRATIC PROGRAMMING ALGORITHM

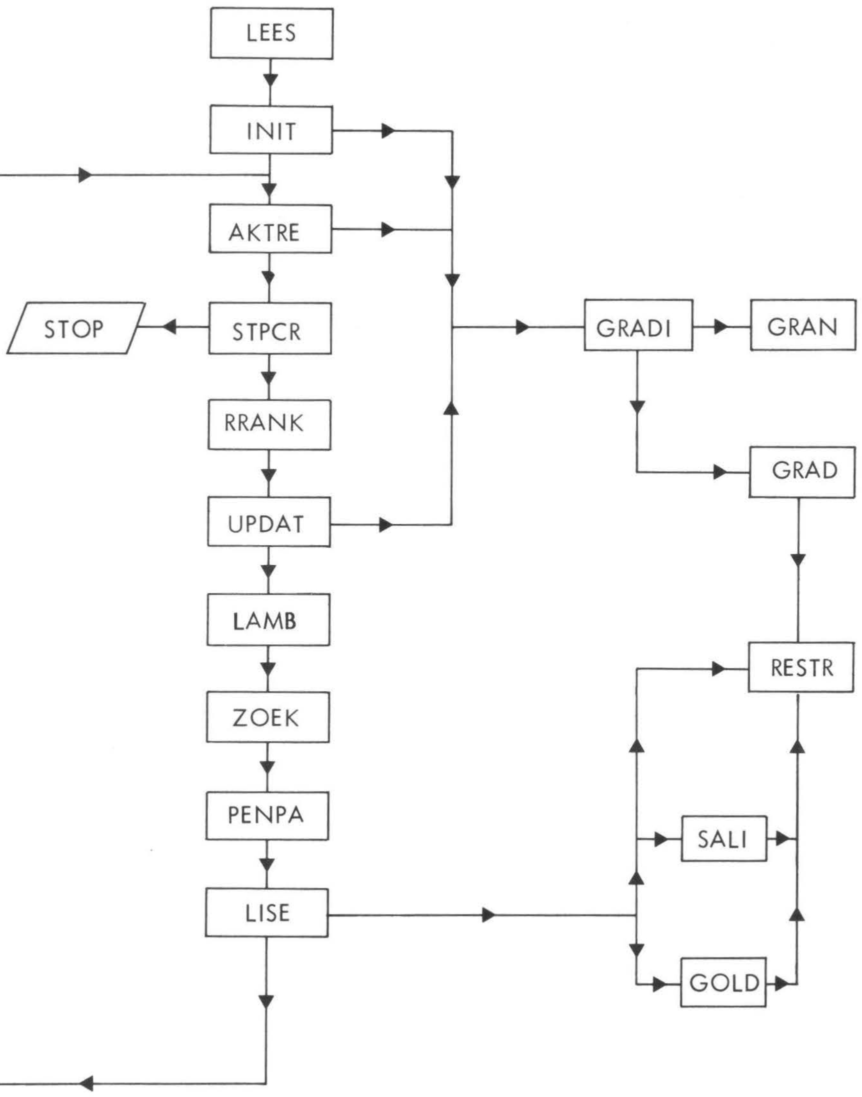


Figure E.1 Flowchart of the Recursive Quadratic Programming implementation

Description of the subroutines mentioned in the flowchart.

LEES	- reads input
INIT	- initializes parameters/variables which are not given as input
AKTRE	- determines the set of active constraints and its Jacobian matrix
STPCR	- applies the stopping criteria
RRANK	- determines the row rank of a matrix
UPDAT	- updates the inverse Hessian approximation
LAMB	- calculates the approximating Lagrange multipliers
ZOEK	- calculates the search direction
PENPA	- determines the penalty parameter
LISE	- determines the steplength along the search direction, calls for SALI and GOLD
SALI	- safeguarded line search
GOLD	- golden section line search
GRAN	- contains analytic expressions of the gradients of the problem functions
GRAD	- calculates numerical derivatives
GRADI	- gives the gradients, calls for GRAN or GRAD
RESTR	- function subroutine in which both the problem function- and the penalty function values are calculated.

List of symbols

Symbols that are used only locally, will be defined locally.

We shall consider the problem

$$\begin{cases} \text{minimize} \quad F(x) \\ \text{subject to} \\ \qquad c_i(x) \begin{Bmatrix} \geq \\ = \end{Bmatrix} 0 \qquad i = 1, 2, \ldots, m \end{cases}$$

where $x \in E^n$, the n-dimensional Euclidian space. The problem functions $F(x)$ and $c_i(x)$, $i = 1, \ldots, m$, are functions of $x^T = (x_1, \ldots, x_n)$, and the minimization is performed with respect to x.

$c(x)$	is the $m \times 1$ vector of constraint functions
$g(x)$	is the $n \times 1$ gradient vector of $F(x)$
$a_i(x)$	is the $n \times 1$ gradient vector of $c_i(x)$
A	is the $n \times m$ matrix with $a_i(x)$ as i-th column.

In the special case that all $c_i(x)$ are linear functions of x, the constraints $c(x) \geq 0$ are denoted by $A^T x - b \geq 0$, where b is an $m \times 1$ vector of given scalars.

x_k is the k-th approximation to x^*, a solution of our problem.
$F_k = F(x_k)$
$g_k = g(x_k)$
$y_k = g_{k+1} - g_k$
$s_k = x_{k+1} - x_k = \alpha_k p_k$, where $\alpha_k \in \mathbb{R}$ and p_k is a search direction.
B_k is an $n \times n$ matrix that is a k-th approximation to some Hessian matrix.
H_k is an $n \times n$ matrix that is a k-th approximation to some inverse Hessian matrix.
$I(x) = I$ is the index set of active constraints at x

$L(x,u,v) = F(x) - \sum_{i=1}^{p} u_i c_i(x) - \sum_{i=p+1}^{m} v_i c_i(x)$ is the Lagrangian function associated with the problem considered. The scalars u_i, $i = 1,\ldots,p$ and v_i, $i = p+1,\ldots,m$ denote the Lagrangian multipliers of the inequality and equality constraints respectively.

$P(x,r)$	is a penalty function with parameter r
$\|\|x\|\|$	is some, locally defined, norm of x.

REFERENCES

Abadie, J. and J. Guigou , 1969, Gradient réduit géneralisé, Note HI 069/02, Electricité de France.

Abadie, J. and A. Haggag, Performance du gradient réduit géneralisé avec une méthode quasi Newtonienne pour la programmation non linéaire, R.A.I.R.O., Recherche Opérationelle, 13, (2), 209-216.

Ballintijn, J.F., G. van der Hoek and C.L. Hooykaas, 1978, Optimization methods based on projected variable metric search directions, Report 7821/O, Econometric Institute, Erasmus University Rotterdam.

Bard, Y., 1968, On a numerical instability of Davidon-like methods, Maths. of Comp., 22, 665-666.

Bartels, R.H. and G.H. Golub, 1969, The simplex method of linear programming using LU decomposition, Comm. ACM, 12, (5), 266-268.

Beale, E.M.L., 1959, On quadratic programming, Naval Res. Log. Qu., 6.

Beale, E.M.L., 1967, Numerical methods; in: Nonlinear programming by J. Abadie (ed), North Holland Publishing Co, Amsterdam.

Beale, E.M.L., 1968, Mathematical programming in practice, Pitman, London.

Best, M.J., J. Bräuniger, K. Ritter and S.M. Robinson, 1979, A globally and quadratically convergent algorithm for general nonlinear programming problems, MRC Technical report # 1973, University of Wisconsin.

Betts, J.T., 1977, An accelerated multiplier method for nonlinear programming, JOTA, 21, (2), 137-174.

Biggs, M.C., 1972, Constrained minimization using recursive equality constrained quadratic programming, in: F.A. Lootsma (ed), Numerical methods for non-linear optimization, Academic Press, London.

Biggs, M.C., 1974, The development of a class of constrained minimization algorithms and their applications to the problem of electric power scheduling, Thesis, University of London, March.

Biggs, M.C., 1978, On the convergence of some constrained minimization algorithms based on recursive quadratic programming, J. Inst. Maths. Applics., 2, 67-81.

Box, M.J., 1965, A new method of constrained optimization and a comparison with other methods, Computer J., 8, 42-52.

Box, M.J., 1966, A comparison of several current optimization methods and the use of transformations in constrained problems, Computer J., 9, 67-78.

Bracken, J. and G.P. McCormick, 1968, Selected applications of nonlinear programming, John Wiley & Sons, Inc., New York.

Bräuniger, J., 1977, A modification of Robinson's algorithm for general nonlinear programming problems requiring only approximate solutions of subproblems with linear equality constraints, in: J. Stoer (ed), Optimization techniques, Proc. 8th IFIP conf. on opt. techn. Würzburg, Lecture notes in Control and Inf. Sciences 7, (2), Springer, Heidelberg.

Broyden, C.G., 1967, Quasi-Newton methods and their application to function minimization, Math. Comp., 21, 368-381.

Broyden, C.G., 1970, The convergence of a class of double-rank minimization algorithms part 1 and part 2, J. Inst. Maths. Applics., 6, 76-90, 222-231.

Bus, J.C.P., 1976, Private communication.

Bus, J.C.P., 1977, A proposal for the classification and documentation of testproblems in the field of nonlinear programming, Mathematical Centre, Amsterdam, report NN 9/77.

Carroll, C.W., 1961, The created response surface technique for optimizing nonlinear restrained systems, Operations Res., 9 (2), 169-184.

Colville, A.R., 1968, A comparative study on nonlinear programming codes. I.B.M. New York Scientific Centre, Tech. rep. 320-2949.

Cornwell, L.W., P.A. Hutchison, M. Minkoff and H.K. Schulz, 1978, Testproblems for constrained nonlinear mathematical programming algorithms, Argonne National Laboratory, Applied Mathematics Division, Technical Memorandum 320.

Courant, R., 1943, Variational methods for the solution of problems of equilibrium and vibrations, Bull. Am. Math. Soc., 49, 1-23.

Davidon, W.C., 1959, AEC Doc, ANL 5990 (rev).

Davies, D., 1968, The use of Davidon's method in nonlinear programming, ICI report MSDH/68/110.

Dixon, L.C.W., 1974, in: Software for numerical mathematics, D.J. Evans (ed), Ac. Press.

Eason, E.D., 1977, Validity of Colville's time standardization for comparing optimization codes, ASME Design Engineering Technical Conference, Chicago, September, 77-DET-116.

Fiacco, A.V., 1961, Comments on a paper of Carroll, Ops. Res., 9, (2).

Fiacco, A.V. and G.P. McCormick, 1963, Programming under nonlinear constraints by unconstrained minimization: A primal-dual method, Technical paper RAC-TP 96, Research Analysis Corporation, McLean.

Fiacco, A.V. and G.P. McCormick, 1964a, The sequential unconstrained minimization technique for nonlinear programming: A primal-dual method, Man. Sci., 10, (2), 360-366.

Fiacco, A.V. and G.P. McCormick, 1964b, Computational algorithm for the sequential unconstrained minimization technique for nonlinear programming, Man. Sci., 10, (2), 601-617.

Fiacco, A.V. and G.P. McCormick, 1966, Extensions of SUMT for nonlinear programming: Equality constraints and extrapolation, Man. Sci., 12, (11), 816-829.

Fiacco, A.V. and G.P. McCormick, 1968, Nonlinear programming: Sequential unconstrained minimization techniques, J. Wiley & Sons, New York.

Fletcher, R., 1969, UKAEA research report TP 368.

Fletcher, R., 1970a, A new approach to variable metric algorithms, Computer J., 13, 317-322.

Fletcher, R., 1970b, A class of methods for nonlinear programming with termination and convergence properties, in: J. Abadie (ed), Integer and nonlinear programming, North Holland, Amsterdam.

Fletcher, R., 1971, A general quadratic programming algorithm, J. Inst. Maths. Appls., 7, 76-91.

Fletcher, R. and S.A. Lill, 1970, A class of methods for nonlinear programming, II, Computational experience, in: J.B. Rosen, O.L. Mangasarian and K. Ritter (ed), Nonlinear Programming, Academic Press, London.

Fletcher, R. and M.J.D. Powell, 1963, A rapidly convergent descent method for minimization. Computer J., 6, 163-168.

Frisch, K.R., 1954, Principles of linear programming - With particular reference to the double gradient form of the logarithmic potential method, Memorandum of October 18, University Institute of Economics, Oslo.

Frisch, K.R., 1955, The logarithmic potential method of convex programming Memorandum of May 13, University Institute of Economics, Oslo.

Gill, P.E., G.H. Golub, W. Murray and M.A. Saunders, 1974, Methods for modifying matrix factorizations, Math. of Comp., 28, (126), 505-535.

Gill, P.E. and W. Murray, 1972, Quasi-Newton methods for unconstrained optimization, J. Inst. Maths. Applics., 9, 91-108.

Gill, P.E. and W. Murray, 1974a, Newton-type methods for unconstrained and linearly constrained optimization, Math. Prog., 7, 311-350.

Gill, P.E. and W. Murray, 1974b, Safeguarded steplength algorithms for optimization using descent methods, report NAC 37, Nat. Phys. Lab., Teddington, England.

Gill, P.E. and W. Murray, 1974c, Numerical methods for constrained optimization, Academic Press, London.

Gill, P.E. and W. Murray, 1979, The computation of Lagrange multiplier estimates for constrained minimization, Math. Prog., 17, (1).

Gill, P.E., W. Murray and M.A. Saunders, 1975, Methods for computing and modifying the LDV factors of a matrix, Math. of Comp., 29, (132), 1051-1077.

Goldfarb, D., 1966, Ph.D. Dissertation, Princeton University, Princeton, New York.

Kelley, H.J. and J.L. Speyer, 1970, Accelerated gradient projection, in: Proceedings of colloquium on optimization. Lectures on Mathe-

Lenard, M.L., 1976, Convergence conditions for restarted conjugate gradient methods with inaccurate linesearches, Math. Prog., 10, 32-52.

Lenard, M.L., 1979, A computational study of active set strategies in non-linear programming with linear constraints, Math. Prog., 16, (1), 81-98.

Lill, S.A., 1972, Generalization of an exact method for solving equality constrained problems to deal with inequality constraints, in: F.A. Lootsma (ed), Academic Press, London.

Loewner, C., 1957, Advanced matrix theory, Lecture notes, Stanford University.

Lootsma, F.A., 1969, Hessian matrices of penalty functions for solving constrained-optimization problems, Philips Res. Repts., 24, 322-331.

Lootsma, F.A., 1970, Boundary properties of penalty functions for constrained minimization, Ph.D. Thesis, Eindhoven University.

Lootsma, F.A., 1976, Nonlinear optimization in industry and the development of optimization programmes, Paper presented at the IXth International Symposium on Mathematical Programming, Budapest.

Luenberger, D.G., 1973, Introduction to linear and nonlinear programming, Addison-Wesley, Massachusetts.

Mayne, D.Q. and E. Polak, 1978, A superlinearly convergent algorithm for constrained optimization problems, Publication 78/52, Imperial College, London.

Mc Cormick, G.P., 1967, Second order conditions for constrained minima, SIAM J. Appl. Math., 15, (3), 641-652.

Mc Cormick, G.P., 1971, Penalty function versus nonpenalty function methods for constrained nonlinear programming problems, Math. Prog. 1, 217-238.

Mc. Cormick, G.P. and J.D. Pearson, 1969, Variable metric methods and unconstrained optimization, in: Optimization, R. Fletcher (ed), Academic Press, London, 307-325.

Meyer, G.G.L., 1979, Asymptotic properties of sequences iteratively generated by point-to-set maps, Math. Prog. Study 10, North Holland.

Murray, W., 1967, Ill-conditioning in barrier and penalty functions arising in constrained non-linear programming, the 6th Math. Prog. Symp., Princeton University.

Murray, W., 1969, An algorithm for constrained minimization, in: Optimization, R. Fletcher (ed), Academic Press, London.

Murray, W. and M.H. Wright, 1976, Efficient linear search algorithms for the logarithmic barrier function, Systems optimization laboratory, Technical report SOL 76-18, Stanford University.

Murtagh, B.A. and R.W.H. Sargent, 1969, A constrained minimization method with quadratic convergence, in: Optimization, R. Fletcher (ed) Academic Press, London.

Nazareth, L., 1979, Why is BFGS so good? Paper presented at the Xth International Symposium on Math. Prog., Montréal.

Oren, S.S., 1973, Self scaling variable metric algorithms without line-search for unconstrained minimization, Maths. of Comp., 27, (124), 873-885.

Oren, S.S., 1974a, Self scaling variable metric algorithms, part II: Implementation and experiments, Man. Sci., 20, (5).

Oren, S.S., 1974b, On the selection of parameters in self scaling variable metric algorithms, Math. Prog., 7, (3).

Oren, S.S., 1978, Perspectives on self scaling variable metric algorithms, Xerox, Analysis Research Group, Technical report 78-4.

Oren, S.S. and D.G. Luenberger, 1974, Self scaling variable metric algorithms, part I, Criteria and sufficient conditions for scaling a class of algorithms, Man. Sci., 20, (5).

Oren, S.S. and E. Spedicato, 1976, Optimal conditioning of self scaling variable metric algorithms, Math. Prog., 10, 70-90.

Ortega, J.M. and W.C. Rheinboldt, 1970, Iterative solution of nonlinear equations in several variables, Academic Press, London.

Osborne, M.R., 1972, Topics in optimization, Stan-CS-72, Stanford University.

Ostrowski, A.M., 1966, Solutions of equations and systems of equations, Academic Press, New York.

Paviani, D.A., 1969, Ph.D. Thesis, University of Texas, Austin.

Peters, G. and J.H. Wilkinson, 1970, The least squares problem and pseudo-inverses, Comp. J., 13, 309-316.

Pietrzkowsky, T., 1962, On a method of approximate final conditional maxima, Inst. Maszyn Matematcyoznych PAN, Algorythmy VI.

Polak, E., 1971, Computational methods in optimization: a unified approach Academic Press, New York.

Powell, M.J.D., 1969a, Rank one methods for unconstrained optimization, AERE Rept. TP 372.

Powell, M.J.D., 1969b, A method for nonlinear constraints in minimization problems, in: R. Fletcher (ed), Optimization, Academic Press, London.

Powell, M.J.D., 1977a, A fast algorithm for nonlinearly constrained optimization calculations, in: G.A. Watson (ed), Numerical analysis Proceedings Biennal Conference Dundee, Lecture notes in Mathematics, 630, Springer, Heidelberg.

Powell, M.J.D., 1977b, Restart procedures for the conjugate gradient method, Math. Prog., 12, 241-254.

Powell, M.J.D., 1978, Algorithms for nonlinear constraints that use Lagrangian functions, Math. Prog., 14, (2).

Ragsdell, K.M., 1978, On some experiments which delimit the utility of nonlinear programming algorithms, Working paper, School of Mechanical Engineering, Purdue University.

Rinnooy Kan, A.H.G., 1979, Paper presented at the Xth International Symposium on Math. Prog., Montréal.

Robinson, S.M., 1972, A quadratically convergent algorithm for general nonlinear programming problems, Math. Prog., 3, 145-156.

Rockafellar, R.T., 1974, Augmented Lagrange multiplier functions and duality in non-convex programming, SIAM J. Control and Optimization, 12, 268-285.

Rosen, J.B., 1960, The gradient projection method for nonlinear programming, part I, Linear constraints, J. Soc. Ind. Appl. Math. 8, (1), March.

Rosen, J.B., 1961, The gradient projection method for nonlinear programming part II, Nonlinear constraints, J. Soc. Ind. Appl. Math. 9, (4), December.

Rosen, J.B., 1963, Convex partitioning programming, in: R.L. Graves and P. Wolfe (eds), Recent advances in Mathematical Programming, Mc Graw Hill.

Rosen, J.B., 1976, A two-phase method for nonlinear constraint problems, Paper presented at the IXth International Symposium on Math. Prog., Budapest.

Rosen, J.B., 1977, Two-phase algorithm for nonlinear constraint problems, Technical report 77-8, University of Minnesota.

Rosen, J.B. and J. Kreuser, 1972, A gradient projection algorithm for nonlinear constraints, in: F.A. Lootsma (ed), Numerical methods for non-linear optimization, Academic Press, London.

Rosen, J.B. and S. Suzuki, 1975, Construction of nonlinear programming testproblems, Comm. ACM., 8, 113.

Rosenbrock, H.H., 1961, An automatic method for finding the greatest or least value of a function, Computer J., 3, 175-184.

Ryan, D.M., 1971, Transformation methods in nonlinear programming, Thesis, Australian National University.

Ryan, D.M., 1974, Penalty and barrier functions, in: P.E. Gill and W. Murray (eds), Numerical methods for constrained optimization, Academic Press, London.

Sargent, R.W.H. and D.J. Sebastian, 1972, Numerical experience with algorithms for unconstrained minimization, in: F.A. Lootsma (ed), Numerical methods for nonlinear optimization, Academic Press, London.

Schittkowski, K., 1978a, Randomly generated NLP test problems with predetermined solutions, Preprint 35, Institut für Angewandte Mathematik und Statistik, Universität Würzburg.

Schittkowski, K., 1978b, A numerical comparison of optimization software using randomly generated test problems - An intermediate balance - Preprint 43, Institut für Angewandte Mathematik und Statistik, Universität Würzburg.

Schweigman, C., 1974, Constrained minimization: handling of linear and nonlinear constraints, Thesis, University of Technology Delft.

Shanno, D.F. and Kang-Hoh Phua, 1978a, Matrix conditioning and nonlinear optimization, Math. Prog., 14, 149-160.

Shanno, D.F. and Kang-Hoh Phua, 1978b, Numerical comparison of several variable metric algorithms, JOTA, 25, (4), August.

Staha, R.L., 1973, Constrained optimization via moving exterior truncations Ph.D. Thesis, University of Texas.

Stoer, J., 1971, On the numerical solution of constrained least squares problems, SIAM J. Numer. Anal., 8, (2), 382-411.

Van Dam W.B. and J. Telgen, 1979, Randomly generated polytopes for testing mathematical programming algorithms, Report 7929/O, Econometric Institute, Erasmus University Rotterdam.

Van der Hoek, G., 1978, Experiments with a reduction method for nonlinear programming, based on a restricted Lagrangian, Report 7804/O, Econometric Institute, Erasmus University Rotterdam.

Van der Hoek, G., 1979, Asymptotic properties of reduction methods applying linearly equality constrained reduced problems, Report 7933/O, Econometric Institute, Erasmus University Rotterdam.

Van der Hoek, G. and M.W. Dijkshoorn, 1979, A numerical comparison of self scaling variable metric algorithms, Report 7910/O, Econometric Institute, Erasmus University Rotterdam.

Van der Hoek, G. and C.L. Hooykaas, 1979, A reduction method for nonlinear programming, based on a restricted Lagrangian (RESLA), Methods of operations research, band 31, Proceedings of the 3rd Symposium on Operations Research, Mannheim.

Van der Hoek, G. and R.Th. Wymenga, 1980, Aspects of recursive quadratic programming, Report Econometric Institute, Erasmus University Rotterdam, forthcoming.

Wilkinson, J.H., 1965, The algebraic eigenvalue problem, Clarendon Press, London.

Wilson, R.B., 1963, A simplicial algorithm for concave programming, Ph.D. Thesis, Graduate School of Business Administration, Harvard University, Boston.

Wolfe, Ph., 1961, Accelerating the cutting plane method for nonlinear programming, J. Soc, Ind. Appl. Math., 9, (3).

Zangwill, W.I., 1969, Nonlinear programming, a unified approach, Prentice Hall Inc., Englewood Cliffs.

Zoutendijk, G., 1960, Methods of feasible directions, Elsevier Publishing Company, Amsterdam.

AUTHOR INDEX

SUBJECT INDEX

Samenvatting

In dit proefschrift worden *niet lineaire programmeringsproblemen* beschouwd. Dat zijn problemen waarin een *doelfunktie* wordt *geoptimaliseerd* en de *beslissingsvariabelen* slechts waarden aannemen die voldoen aan m.b.v. *beperkende funkties* gedefinieerde restricties. De niet lineariteit van het probleem houdt in dat minstens één van de beslissingsvariabelen in een niet lineaire gedaante in het probleem voorkomt, hetzij in de doelfunktie dan wel in minstens één van de beperkende funkties.

De klasse van in dit proefschrift gedefinieerde *reductie methoden voor niet lineaire programmering* bestaat uit die oplossingsmethoden welke een gegeven beperkt niet lineair programmeringsprobleem oplossen door een *rij* 'eenvoudiger' niet lineaire programmeringsproblemen op te lossen.

Een voorbeeld van een 'eenvoudiger' probleem is een *vrij* of *onbeperkt* niet lineair programmeringsprobleem.
Deze vrije problemen worden behandeld in hoofdstuk II. De recentelijk ontwikkelde zogenaamde *zelf schalende variabele metriek* methoden (Oren en Luenberger (1974), Oren en Spedicato (1976) en Shanno en Phua (1978)) zijn in het bijzonder bedoeld om slecht geschaalde funkties zonder restricties te optimaliseren. Na een beknopte uiteenzetting van de theoretische achtergronden worden *numerieke experimenten* besproken die opgezet zijn om langs experimentele weg deze eigenschap te verifiëren. Tegelijkertijd worden de betrouwbaarheid en de efficiency van deze algoritmen onderling vergeleken. Uit de experimenten met een ontworpen set van geparametriseerde testproblemen blijkt duidelijk dat zowel deze algoritmen als de bekende BFGS-algoritme kunnen volstaan met een inexacte, dus goedkope, lijnzoeker. Opvallend zijn de goede resultaten die verkregen zijn met de door Oren en Spedicato voorgestelde 'switch II' variant en de (eventueel initieel geschaalde) BFGS algoritme. Voorgesteld wordt dan ook deze gevonden gunstige combinaties van lijnzoeker en bijwerken van 2e orde informatie toe te passen in het kader van de in hoofdstuk III besproken *Recursief Kwadratische Programmerings* reductie methoden.

Deze reductie methoden zijn gebaseerd op het gebruik van een kwadratisch programmeringsprobleem met gelijkheidsrestricties als gereduceerd probleem. Na een korte samenvatting van met name door Bartholomev-Biggs gepubliceerde resultaten betreffende de convergentie en de convergentiesnelheid van deze reductie methoden wordt een aantal modificaties besproken. Deze betreffen de toepassing van een efficiënte en betrouwbare lijnzoekmethode en een zelf schalende variabele metriek formule voor het bijwerken van de tweede orde informatie.

Hoofdstuk IV handelt over reductie methoden die een *lineair beperkt niet lineair programmeringsprobleem* als gereduceerd probleem toepassen. Een verbetering van de algoritme van Robinson (1972) wordt voorgesteld, die er op neer komt dat slechts de restricties die behoren tot de in het bereikte iteratiepunt gedefinieerde aktieve verzameling van restricties een bijdrage leveren tot de definitie van het gereduceerde probleem. Bewezen wordt dat onder zekere voorwaarden de convergentiesnelheid *R-kwadratisch* is. De bereikte verbetering ten opzichte van de oorspronkelijke algoritme wordt aangegeven. Voorts wordt bewezen dat onder vrij algemene voorwaarden de door de algoritme gegenereerde rij Kuhn-Tucker punten van gereduceerde problemen convergeert naar een *dekpunt* van de afbeelding die de algoritme definieert. Ook wordt bewezen dat zo'n dekpunt een Kuhn-Tucker punt is van het oorspronkelijke probleem.

Een resultaat van Bräuniger (1977) betreffende het vinden van de in het optimum aktieve restricties wordt besproken en verbeterd.

Aan de bovengenoemde voorwaarden kan worden voldaan door toepassing van enige voorafgaande berekeningen in een zogenaamde eerste fase. Deze *'fase I'*, die bestaat uit de toepassing van een uitwendige boetefunktietechniek op een geschikt gekozen hulpprobleem, moet zorgen voor een startpunt dat voldoende dicht bij het gezochte optimum ligt. In dit kader worden ook enige suggesties gedaan om met behulp van de in fase I verkregen informatie een goede, eerste aktieve verzameling van restricties voor de erop volgende fase II te definiëren. Het hoofdstuk wordt besloten met een stelling over de convergentie en de convergentiesnelheid van de resulterende 2-fase algoritme.

Hoofdstuk V handelt over algoritmische en numerieke aspekten van de in hoofdstuk IV besproken 2-fase reductie methoden. Een verbeterde aktieve verzamelingsstrategie wordt voorgesteld voor de algoritme die de lineair beperkte gereduceerde problemen oplost. Vervolgens wordt een *numeriek stabiele implementatie* van deze algoritme besproken, welke is gebaseerd op het gebruik van Cholesky- en LU-decomposities. Voor het bijwerken van de Cholesky factoren in het geval van een verandering van de aktieve verzameling van restricties wordt een nieuwe strategie afgeleid.

Hoofdstuk VI geeft eerst een discussie van de opzet van numerieke experimenten om de geïmplementeerde algoritmen van de hoofdstukken III-V te vergelijken. Ter sprake komen: *de keuze van testproblemen, stopcriteria* en

indicatoren om efficiency en betrouwbaarheid te meten. Gekozen wordt voor een set van 24 testproblemen met de (bij de optimalisatie)problemen veroorzakende eigenschappen, zoals: slechte conditionering, sterke niet-lineariteit van probleemfunkties, afhankelijke normaalvectoren van restricties, niet convexiteit van het probleem en een relatief groot aantal (t.o.v. de dimensie) aktieve restricties in het optimum.
Onder toepassing van het stopcriterium dat de stapgrootte en de eventuele schending van restricties relatief klein genoeg zijn, wordt als indicator voor de kwaliteit gekozen het benodigde aantal *equivalenten van doelfunktieberekeningen* om een probleem op te lossen. De resultaten worden vergeleken voor de recursief kwadratische programmerings reductie methode en een tweetal 2-fase reductie methoden (met linearisatie van alle dan wel een deel van de restricties). De algemene conclusie is dat al deze drie algoritmen op betrouwbare en efficiënte wijze de testproblemen oplossen, zij het dat de recursieve algoritme iets efficiënter en de 2-fase algoritmen iets betrouwbaarder zijn. Het voorstel om slechts de restricties van de bekende aktieve verzameling te laten bijdragen tot de definitie van het gereduceerde probleem leidt tot een efficiëntere 2-fase algoritme, die echter soms geneigd is een niet correcte aktieve verzameling te lang vast te houden. In een 2-tal gevallen leidt dit tot een onjuist antwoord. Klaarblijkelijk verdient het aanbeveling nader onderzoek te verrichten naar een verbetering van de fase I en een verbetering van de onderlinge afstemming van fase I en fase II. Alle ontwikkelde algoritmen blijken betere resultaten te boeken dan een 4-tal bekende, in Staha (1973) geteste algoritmen. In enkele slotopmerkingen worden voorlopige conclusies getrokken betreffende de toepassing van een in Powell (1977) voorgestelde wijze van bijwerken van de 2e orde informatie in het kader van recursief kwadratische programmerings reductie methoden.