Netherlands school of economics.
~~econometrics~~
ECONOMETRIC INSTITUTE

A NUMERICAL COMPARISON OF SELF SCALING VARIABLE METRIC ALGORITHMS

G. van der HOEK and M. W. DIJKSHOORN

REPORT 7910/O

ERASMUS UNIVERSITY, P.O. BOX 1738, ROTTERDAM, THE NETHERLANDS

A NUMERICAL COMPARISON OF SELF SCALING VARIABLE METRIC ALGORITHMS

by

G. van der Hoek and M.W. Dijkshoorn

ABSTRACT

Recently developed quasi-Newton algorithms for unconstrained optimization focus on the solution of badly scaled problems. A uniform numerical comparison of these algorithms is performed to get insight in their relative behaviour and to verify emperically their ability to solve badly scaled problems. A suitable battery of testproblems is suggested. The design of the experiments and their results are preceded by a short description of the theoretical backgrounds as they were developed by Oren and Luenberger. Recent alternatives of Oren and Spedicato and of Shanno and Phua are mentioned The classical DFP- and BFGS-algorithms are considered as well.

Contents

June 1979

'Preliminary and confidential, comments invited'

1. INTRODUCTION

This study concerns the unconstrained minimization problem:

1.1. $\min_x f(x)$, $x \in E^n$, the n-dimensional Euclidian space.

The objective function is supposed to be a sufficiently differentiable convex function of x. The solution of unconstrained minimization problems is one of the main subjects in the field of nonlinear programming. The quasi-Newton methods (Davidon, [1959] called them Variable Metric Methods) constitute a class of algorithms which use the gradients $g_k = \nabla f(x_k)$ at the iteration point x_k to define the search direction p_k and to construct an approximation D_k to the inverse Hessian H^{-1} of f(x) in the optimal point x^*. A general scheme for quasi-Newton methods is:

1. Initialization: choose arbitrarily a starting point x_0 and a positive definite symmetric matrix D_0.
   Calculate $g_0 = \nabla f(x_0)$. Go to 2
2. Arrived at x_k, k = 0, 1, 2, ... define x_{k+1} from $x_{k+1} = x_k - \alpha_k D_k g_k$, where $\alpha_k > 0$ is the steplength, which is determined by a linesearch. Go to 3
3. In case of convergence, the algorithm stops. Otherwise go to 4
4. D_k is updated and k:= k+1. Go to 2.

Remark: The choice $D_k = I$, the (n * n) - identity matrix for k = 0,1,2, ... defines the method of steepest descent. Newton's method appears from $D_k = H^{-1}(x_k)$ with $\alpha_k = 1$, for k = 0,1,2, ...

The first quasi-Newton method is due to Davidon, [1959] while Fletcher and Powell, [1963] further developed Davidons method and supplemented convergence and stability theorems.

Usually this first representative of the class of quasi-Newton algorithms is called the DFP algorithm.

Three characteristics of this algorithm are:

1. The matrices D_k, k = 1, 2, 3, ... are positive definite, provided that D_0 is chosen to be positive definite.
2. If f(x) is a positive definite quadratic function and $D_0 = I_n$, the algorithm is a conjugate gradient method and thus converges in at most n steps.

3. If again $f(x)$ is a positive definite quadratic function and the algorithm requires all n steps, then D_n equals the inverse H^{-1} (this means that at x_n the direction $- D_n g_n$ equals the Newton step to the optimum x^*).

The convergence behaviour of the DFP-algorithm relies on these well-known properties. The 1963-paper of Fletcher and Powell was succeeded by a great number of publications on related algorithms, most of them presented alternative rules for the way of updating the matrices D_k.

Huang, [1970] formulated a general class of algorithms, with linesearch, satisfying the second characteristic on the conjugacy of the search directions. Later Osborne, [1972] suggested the following compact description of Huangs update-formulae:

$$(1.2) \qquad D_{k+1} = D_k - \frac{D_k q_k q_k' D_k}{q_k' D_k q_k} + \rho_k \frac{p_k p_k'}{p_k' q_k} + \theta_k v_k v_k' \quad \text{with}$$

$$(1.3) \qquad v_k = (q_k' D_k q_k).p_k - (p_k' q_k).D_k q_k$$

Here $q_k = g_{k+1} - g_k$, the gradient-difference vector,
$p_k = x_{k+1} - x_k$, the step,
ρ_k and θ_k are scalar parameters which can be chosen arbitrarily within some limits.

The original DFP-algorithm corresponds with the choice $\rho_k = 1$, $\theta_k = 0$ for $k = 0, 1, 2, \ldots$, while the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS) corresponds with $\theta_k = \rho_k = 1$ for $k = 0, 1, 2, \ldots$.
Besides their successes in applications, also less favourable properties of quasi-Newton methods were reported, such as:

1. The dependency of the accuracy achieved in the linesearch. Also this study shows that the influence of this accuracy varies per algorithm, which agrees with the results of earlier experiments by van der Hoek and Baardman, [1977].

2. Possible singularity of the inverse Hessian approximation D_k. A restart with $D_k = I_n$ is usually applied. See McCormick and Pearson, [1969], Lenard, [1976] and Powell, [1977].

3. The sensitivity of the DFP algorithm to scaling of the objective function. See e.g. Bard, [1968]. Bad scaling can give rise to a singular matrix D_k.

These above mentioned imperfections are confirmed by the results of Broyden, [1967] and Greenstadt, [1970]. A remarkable result is proved in Dixon, [1972] namely that for any differentiable function the quasi-Newton algorithms of Huang's family generate exactly the same iteration points, provided that they start from the same point x_0 and that the applied exact linesearch uniquely defines the steplength.

The self-scaling variable metric algorithms (SSVM) of Oren and Luenberger, [1974] do not suffer from the three above mentioned shortcomings. Oren and Luenberger constructed a two-parameter family of updates, belonging to Huangs family. They showed that efficient scaling of the objective function can be achieved by a correct choice of their parameters. In fact their results are extentions of the work of Fletcher,[1970] who showed the monotone convergence of eigenvalues of the matrices D_kH for a convex class of updating formulae. If we define the condition number of a matrix as: the ratio of the largest to the smallest eigenvalue, we see that as soon as the condition number of D_kH equals 1 there holds: $D_k = H^{-1}$. So this condition number is a measure for comparison of D_k with H^{-1}.
Oren and Luenberger succeeded in the construction of a sequence of matrices D_kH with decreasing condition numbers, by applying the update formulae:

$$(1.4) \qquad D_{k+1} = \left\{ D_k - \frac{D_k q_k q_k' D_k}{q_k' D_k q_k} + \theta_k v_k v_k' \right\} \gamma_k + \frac{p_k p_k'}{p_k' q_k} \quad \text{with}$$

$$(1.5) \qquad v_k = (q_k' D_k q_k)^{\frac{1}{2}} \cdot \left(\frac{p_k}{p_k' q_k} - \frac{D_k q_k}{q_k' D_k q_k} \right) ,$$

$$(1.6) \qquad \gamma_k = \frac{p_k' q_k}{q_k' D_k q_k} \cdot (1 - \phi_k) + \frac{g_k' p_k}{g_k' D_k q_k} \cdot \phi_k$$

Particular choices of the parameters in the formulae (1.4)-(1.6) yield:

1. The DFP-update for $\gamma_k = 1$, $\theta_k = 0$ for all k
2. The BFGS-update for $\gamma_k = 1$ and $\theta_k = 1$ for all k.
3. SSVM arises from all other combinations in which γ_k satisfies (1.6)

Basically the parameters ϕ_k and θ_k in (1.4)-(1.6) can be chosen arbitrarily between bounds which shall be derived in ch. 2.
In their first experiments Oren and Luenberger, [1974] merely considered the case $\phi_k = \theta_k = 0$ for all k. Later Oren, [1974b] discussed the choice of parameters in a separate paper.

We started our experiments by extending to 25 combinations of ϕ and θ thus performing a sensitivity analysis on the choice of the parameters. We shall further investigate in this study the required accuracy of the linesearch, which still quarantees overall convergence.

Denoting:

(1.7) $h_k(\alpha) = f(x_k - \alpha D_k g_k)$

with α as variable of the linesearch, we terminate this search as soon as:

(1.8) $| h_k(\alpha_{i+1}) - h_k(\alpha_i) | < \varepsilon$

where i counts the number of cubic interpolations in the linesearch.
We investigated $\varepsilon = 10^{-1}, 10^{-3}, 10^{-6}$.
Subsequently for the 3 best of these algorithms the efficiency of Oren's linesearch, Oren, [1974b], based on the application of the test of Goldstein and Price, [1967] was investigated.
Also we tested recent succeeders of the SSVM algorithms: the algorithms of Oren and Spedicato, [1976] which are a subclass of Optimally Conditioned Self-Scaling (OCSS) algorithms. These algorithms minimize a sharp bound on the condition number of D_k. The resulting algorithms are two switching rules which were tested together with two switching rules suggested by Oren, [1974b].

Finally the results are compared with those of the DFP and BFGS algorithms and with two algorithms suggested by Shanno and Phua, [1978], which consist of devices for scaling of H_0 before applying the BFGS algorithm.
The first part of this report summarizes the theoritical backgrounds of SSVM algorithms and their extentions as they appeared in literature in the years 1976-1978. This concerns chapters 2.1-2.3.
The second part, consisting of chapters 3.1-3.3, deals with the design and realisation of experiments to perform a mutual comparison of the above mentioned algorithms. Special attention is paid to the choice of testfunctions. Finally chapter 3.4 contents a discussion of the results of the experiments.

2.1. Self Scaling Variable Metric Algorithms

The main difference between SSVM-algorithms and other quasi-Newton algorithms is the choice of the updating formulae.
We restrict our analysis to the minimization of convex functions $f(x)$ on E^n which are at least twice continuously differentiable. Those functions can be approximated in a neighbourhood of their optimum x^* by the first three terms of their Taylor-series expansion:

$$(2.1.1) \qquad f(x) = f(x^*) + (x-x^*)'\nabla f(x^*) + \tfrac{1}{2}(x-x^*)'H(x-x^*)$$

$$= f(x^*) + \tfrac{1}{2}(x-x^*)'H(x-x^*).$$

An important theorem on the global convergence of quasi-Newton algorithms applied on a quadratic objective function is

Theorem 1. For a positive definite quadratic objective function $f(x)$ the quasi-Newton algorithms convergence to the unique optimum x^* of $f(x)$, for any initial point x_0.
At every step the following inequality holds:

$$(2.1.2) \qquad f(x_{k+1}) - f(x^*) \leq \left\{\frac{\kappa(R_k) - 1}{\kappa(R_k) + 1}\right\}^2 \cdot \left(f(x_k) - f(x^*)\right),$$

where $\kappa(R_k)$ is the condition number of the matrix $R_k = H^{\frac{1}{2}}D_kH^{\frac{1}{2}}$.

A proof of this theorem can be found in Luenberger, [1973] and de Jong, [1976]. The matrix R_k is used as an indicator for the difference between D_k and H^{-1}. Clearly $R_k = I$ indicates that $D_k = H^{-1}$. As R_k is similar to D_k, by definition, it has the same condition number as D_k.
It is clear from theorem 1 that convergence is accellerated if the quotients $\left\{\frac{\kappa(R_k) - 1}{\kappa(R_k) + 1}\right\}^2$ form a decreasing null sequence. Thus $\lim_{k\to\infty} \kappa(R_k) = 1$ is required.
Before proceeding with the theoretical backgrounds we illustrate the effect of scaling by an example which makes use of the SSVM updating formulae (1.4.)-(1.6.). We apply three algorithms on the function:

$$f(x) = 30x_1^2 + 20x_2^2 \text{ with } x_0' = (1,1).$$

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 scalingfactor 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 D_k, H and R_k for k = 0, 1. λ_1 and λ_2 are the eigenvalues of R_k for k = 0, 1.

Table I Iteration matrices at the starting point

	algorithm 1	algorithm 2	algorithm 3
D_0	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
H	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$	$\begin{pmatrix} 1\frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 60 & 0 \\ 0 & 40 \end{pmatrix}$
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
$\kappa(R_0)$	$1\frac{1}{2}$	$1\frac{1}{2}$	$1\frac{1}{2}$
$\kappa(H)$	$1\frac{1}{2}$	$1\frac{1}{2}$	$1\frac{1}{2}$

Exact lineminimization in the direction $-\alpha D_0 g_0$ and application of (1.4.)-(1.6.) yields:

Table II Iteration matrices after one iteration

D_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

Obviously $\kappa(R_1)$ for the scaled DFP algorithm improves the unscaled version and is comparable to the SSVM algorithm with $\theta = \phi = 0$.

2.1.1 Properties of the matrices D_k.

The matrices D_k generated by the application of the rules given in (1.4.)-(1.6.) satisfy the following quasi-Newton requirements:

1. For a positive definite matrix D_k, $\theta_k \geq 0$, $\gamma_k \geq 0$ and $p_k'q_k > 0$ the matrix D_{k+1} is positive definite as well.
2. For a quadratic objective function f(x) the SSVM-algorithm is a conjugate direction algorithm which converges to the minimum in at most n steps.
3. For a quadratic objective function f(x) the n-th approximation D_n equals H^{-1} in the case that $\gamma_k = 1$ for all k.

Properties 1, 2 and 3 are well-known for quasi-Newton algorithms. As an illustration only the first property will be proved below. For simplicity of the notation the index k is suppressed and the index k+1 is replaced by '-'. So $D_k = D$ and $D_{k+1} = \bar{D}$.

Proof of property 1.

The proof is by induction. Using $D_0 = I_n$ we only have to prove the positive definiteness of $\bar{D}$ from that of D. Let $x \in E^n$ be a nontrival vector, then:

$$x'\bar{D}x = \gamma x' \left[D - \frac{Dqq'D}{q'Dq} + \theta vv'\right] x + \frac{x'pp'x}{p'q}$$

This yields for $\theta \geq 0$

$$x'\bar{D}x \geq \gamma x'\left[D - \frac{Dqq'D}{q'Dq}\right]x + \frac{x'pp'x}{p'q}$$

Now we distinguish two cases: a. $x'p \neq 0$, b. $x'p = 0$

a. $x'\bar{D}x \geq \gamma \left[\frac{x'Dx.q'Dq - x'Dqq'Dx}{q'Dq}\right] + \frac{(x'p)^2}{p'q}$

As the square root of the positive definite symmetric matrix D exists, we define:

$r = D^{\frac{1}{2}}x$ and $s = D^{\frac{1}{2}}q$

So: $x'\bar{D}x \geq \gamma \left[\frac{r'r.s's - (r's)^2}{s's}\right] + \frac{(x'p)^2}{p'q}$

Using $s's > 0$, $r'r.s's \geq (r's)^2$ (Cauchy-Schwarz) and $\gamma > 0$, this yields $x'\bar{D}x \geq \frac{(x'p)^2}{p'q} > 0$ as $p'q > 0$ and $x'p \neq 0$.

b. $x'p = 0$. As x is nontrivial and $p = 0$ and $\alpha = 0$ correspond with $x = x^*$ we only need to consider $x'Dg = 0$.

Then we see that $x'Dq = x'D(\bar{g} - g)$

$= x'D\bar{g}$,

and

$q'Dq = (\bar{g} - g)'D(\bar{g} - g)$

$= \bar{g}'D\bar{g} + g'Dg$ (use that the exact linesearch provides $\bar{g}'Dg = 0$).

This means that

$$x'\bar{D}x \geq \gamma x'\left[D - \frac{Dqq'D}{q'Dq}\right] x$$

can be converted into:

$$x'\bar{D}x \geq \gamma \left[\frac{\bar{g}'D\bar{g}.x'Dx - (x'D\bar{g})^2 + g'Dg.x'Dx}{q'Dq}\right]$$

We define $u = D^{\frac{1}{2}}\bar{g}$ and $v = D^{\frac{1}{2}}x$.

Then $x'\bar{D}x \geq \gamma \frac{u'u.v'v - (v'u)^2}{q'Dq} + \gamma \frac{g'Dg.x'Dx}{q'Dq}$

$\geq \gamma \frac{g'Dg.x'Dx}{q'Dq}$ (Cauchy - Schwarz and $q'Dq > 0$)

> 0 ($\gamma > 0$ and D pos. def.)

2.1.2. The eigenvaluestructure of the updating formulae.

For compactness we follow Oren and Luenberger, [1974] in their notation of (1.4.) and (1.5.) as

2.1.3.

$$D^{\theta}(D,\gamma,p,q) = (D - \frac{Dqq'D}{q'Dq} + \theta vv')\,\gamma + \frac{pp'}{p'q}$$

with $$v = (q'Dq)^{\frac{1}{2}}.\,(\frac{p}{p'q} - \frac{Dq}{q'Dq})$$

Again we suppress the subscripts as only one iteration is considered. The following fundamental Lemma is due to Oren and Luenberger, [1974].

Lemma 1. Let $D^{\theta}(D,\gamma,p,q)$ be defined by relations (2.1.3.). Then for any symmetric non-singular matrix D, non-trival vectors $p,q \in E^n$ and scalars θ, $\gamma(\gamma \neq 0)$, there holds:

a. $D^{\theta}(D,\gamma,p,q) = D^{\theta}(\gamma D,1,p,q)$

b. $D^{\theta}(D,\gamma,p,q) = (1-\theta)D^{0}(D,\gamma,p,q) + \theta D^{1}(D,\gamma,p,q)$

c. $[D^{1}(D,\gamma,p,q)]^{-1} = D^{0}(D^{-1},1/\gamma,q,p)$

Relation (a) connects a scaled problem (γD) with a special update formula, (b) gives the update formula as a combination of elementary formulae (the restriction $\theta \in [0,1]$ will prove to be necessary) and (c) gives a 'duality' relation.

Proof:

a. $D^{\theta}(D,\gamma,p,q) = D^{\theta}(\gamma D,1,p,q)$

Substitution of γD in (2.1.5.) gives:

$$\bar{v} = \gamma^{\frac{1}{2}}.(q'Dq)^{\frac{1}{2}} \left\{\frac{p}{p'q} - \frac{Dq}{q'Dq}\right\}$$

$$= \gamma^{\frac{1}{2}}.v$$

Then (2.1.3) gives:

$$D^{\theta}(\gamma D,1,p,q) = (\gamma D - \frac{\gamma^{2}Dqq'D}{\gamma q'Dq} + \gamma\theta vv')\,\gamma + \frac{pp'}{p'q}$$

$$= (D - \frac{Dqq'D}{q'Dq} + \theta vv')\,\gamma + \frac{pp'}{p'q}$$

$$= D^{\theta}(D,\gamma,p,q)$$

b. $D^{\theta}(D,\gamma,p,q) = (1-\theta)D^{0}(D,\gamma,p,q) + \theta D^{1}(D,\gamma,p,q)$

Relation (2.1.3) yields:

(2.1.4) $D^{1}(D,\gamma,p,q) - D^{0}(D,\gamma,p,q) = \gamma vv'$

(2.1.5) $D^{\theta}(D,\gamma,p,q) - D^{0}(D,\gamma,p,q) = \gamma\theta vv'$

Subtracting θ times (2.1.4) from (2.1.5) yields the desired relation.

c. $[D^{1}(D,\gamma,p,q)]^{-1} = D^{0}(D^{-1},1/\gamma,q,p)$

This can be proved by direct multiplication showing that:

$D^{1}(D,\gamma,p,q)\ D^{0}(D^{-1},1/\gamma,q,p) = I$

The next Lemma deduces an updating formula for the matrix $R = H^{\frac{1}{2}}DH^{\frac{1}{2}}$ from the updating formula for D.

Lemma 2. Let $D^{\theta}(D,\gamma,p,q)$ be defined by (2.1.3) while H is a positive definite symmetric matrix. Assume $p'q > 0$ and $q = Hp$. Then for $R = H^{\frac{1}{2}}DH^{\frac{1}{2}}$ and $z = H^{\frac{1}{2}}p$ the following relation holds:

(2.1.6)

$$\bar{R} = D^{\theta}(R,\gamma,z,z).$$

Proof: Define $u = H^{\frac{1}{2}}v$ and use $H^{\frac{1}{2}}\bar{D}H^{\frac{1}{2}} = \bar{R}$

$$H^{\frac{1}{2}}\bar{D}H^{\frac{1}{2}} = \left\{ H^{\frac{1}{2}}DH^{\frac{1}{2}} - \frac{H^{\frac{1}{2}}DHpp'HDH^{\frac{1}{2}}}{p'HDHp} + \theta H^{\frac{1}{2}}vv'H^{\frac{1}{2}} \right\} \gamma + \frac{H^{\frac{1}{2}}pp'H^{\frac{1}{2}}}{p'Hp}$$

$$= \left\{ R - \frac{Rzz'R}{z'Rz} + \theta uu' \right\} \gamma + \frac{zz'}{z'z}$$

$$= D^{\theta}(R,\gamma,z,z)$$

Assuming that $\gamma \neq 0$ and D nonsingular, $\bar{R}$ is also nonsingular.
Hence Lemma 1 applies for $\bar{R}$ with z=p=q and R=D, thus yielding relations for the updating of R.
The intended eigenvaluestructure analysis of $D^{\theta}(R,\gamma,z,z)$ will be done in two steps.
First, in theorem 2, the relation is considered between the eigenvalues of two general matrices B and A which satisfy the relation.

(2.1.7)
$$B = A - \frac{Arr'A}{r'Ar} + \frac{rr'}{r'r} \quad ,$$

with $r \in E^n$, $r \neq 0$. Note that (2.1.7) means that the matrix B follows from the addition of two matrices of rank 1 to the matrix A.
In the second step those results are extended to $D^{\theta}(R,\gamma,z,z)$.
The theorems were formulated by Oren and Luenberger, [1974] and are extensions of the results of Fletcher, [1970].

The following Lemma, which is due to Loewener, [1957] will be used:

Lemma 3. (Interlocking eigenvalue theorem). Let S 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.
The matrix T is defined by: $T = S + aa'$ and has eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$.
Then: $\lambda_1 \leq \mu_1 \leq \lambda_2 \leq \mu_2 \leq \ldots \leq \lambda_n \leq \mu_n$.

Theorem 2. Let A 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 nontrivial vector. The matrix B is defined by (2.1.7) and has 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 $\lambda_1 \leq 1 \leq \lambda_n$ the index J is such that $\lambda_J \leq 1 \leq \lambda_{J+1}$ then

$\lambda_1 \leq \mu_1 \leq \lambda_2 \leq \mu_2 \leq \ldots \leq \lambda_J \leq \mu_J \leq 1 \leq \mu_{J+1} \leq \lambda_{J+1} \leq \ldots \leq \mu_n \leq \lambda_n$ and at least one of the two eigenvalues μ_J, μ_{J+1} equals unity.

Proof: First we consider the matrix P defined by:

2.1.8 $$P = A - \frac{Arr'A}{r'Ar} \quad .$$

Then $Pr = 0$, $r \neq 0$, means that r is eigenvector of P with eigenvalue 0.
Let $\zeta_1 \leq \zeta_2 \leq \ldots \leq \zeta_n$ be the eigenvalues of P. Then the interlocking eigenvaluetheorem gives:

2.1.9 $$0 = \zeta_1 \leq \lambda_1 \leq \zeta_2 \leq \lambda_2 \leq \ldots \leq \zeta_n \leq \lambda_n .$$

From (2.1.7) and (2.1.8) follows:

$$B = P + \frac{rr'}{r'r}$$

Let $(r=)w_1, w_2, \ldots, w_n$ denote the orthogonal eigenvectors of the symmetric matrix P, corresponding with the eigenvalues $0 = \zeta_1 \leq \zeta_2 \leq \ldots \leq \zeta_n$. Then $r'w_k = 0$ for $k = 2,3,\ldots,n$ yields $Bw_k = Pw_k = \zeta_k w_k$ for $k = 2,3,\ldots, n$. Further: $Br = r$, so the eigenvalues of B are $\{\zeta_2, \ldots, \zeta_n, 1\}$, which becomes the set of nondecreasing eigenvalues $\{\mu_1, \ldots, \mu_n\}$ of the matrix B. One of those μ_k equals 1 and corresponds with the eigenvector r of B. Now there are three cases:

(i) $1 \leq \lambda_1$. As $\lambda_1 \leq \zeta_2$, it follows that unity is the smallest eigenvalue of B, so $\mu_1 = 1$ and $\mu_i = \zeta_i$ for $i = 2,3,\ldots,n$ and part (i) of the theorem is proved.

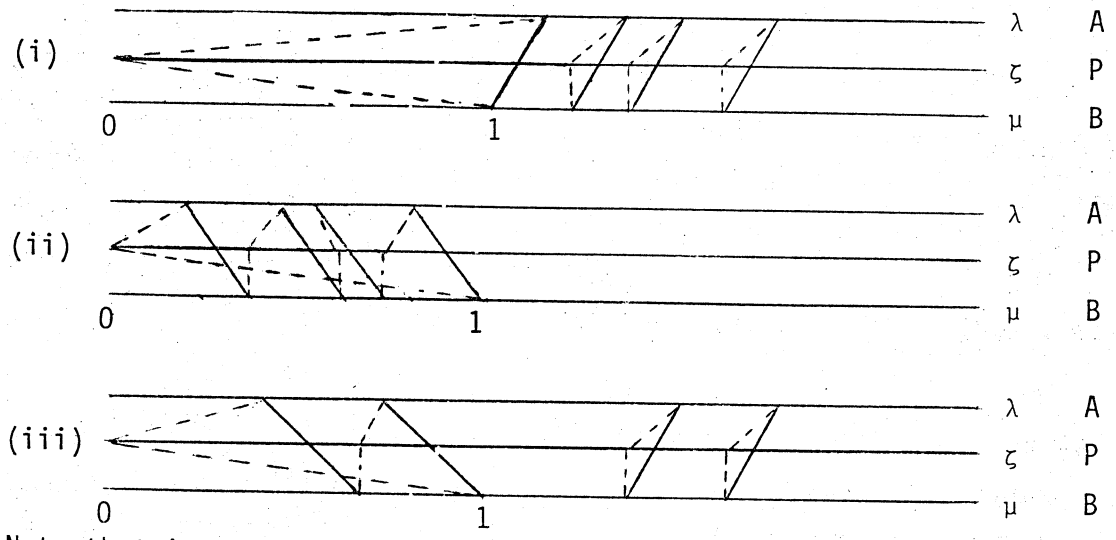
(ii) $\lambda_n \leq 1$. As $\zeta_n \leq \lambda_n$, it follows that unity is the largest eigenvalue of B, so $\mu_n = 1$ and $\mu_i = \zeta_{i+1}$ for $i = 1,2,\ldots,n-1$, and part (ii) of the theorem is proved.

(iii) $\lambda_J \leq 1 \leq \lambda_{J+1}$ for some index J. $1 \leq J \leq n-1$

The interval $[\lambda_J, \lambda_{J+1}]$ contains the eigenvalue 1 and ζ_{J+1}, as (2.1.9) shows. Then the applied rearrangement caused: $\mu_i = \zeta_{i+1}$ for $i = 1,2,\ldots,J-1$, $\mu_J = \min(1, \zeta_{J+1})$, $\mu_{J+1} = \max(1, \zeta_{J+1})$ and $\mu_i = \zeta_i$. for $i = J+2,\ldots,n$.

This means that at least one of the eigenvalues μ_J and μ_{J+1} equals unity. ∎

The way in which the eigenvalues (and consequently the condition numbers) change in the construction of the matrix B from P and A is illustrated in figure 1. The possible cases (i), (ii), and (iii) are given for n = 4



Note that in all three cases the smallest eigenvalue λ, of A is transformed into the eigenvalue $\zeta = 0$ of P, which in turn becomes the eigenvalue $\mu = 1$ of the resulting matrix B.

Figure 1.

It will be clear from theorem 2, especially part (iii), that in order to guarantee that B will have a lower condition number than A, the interval spanned by the eigenvalues of A must contain the element 1. This forms the basis of the development of the SSVM-algorithms.

The next theorem extends the results of theorem 2 to the updating formula (2.1.6) of R.

Theorem 3. Let $\bar{R}^{\theta}(\gamma) = D^{\theta}(R,\gamma,z,z)$ be given by (2.1.6) for some fixed positive definite matrix R and $z \in E^n$, $z \neq 0$. Let the eigenvalues of $\bar{R}^{\theta}(\gamma)$ be:
$\mu_1^{\theta}(\gamma) \leq \mu_2^{\theta}(\gamma) \leq \ldots\ldots \leq \mu_n^{\theta}(\gamma)$. Then, for $\theta \in [0,1]$ and $\gamma > 0$ there holds:
$\mu_i^{0}(\gamma) \leq \mu_i^{\theta}(\gamma) \leq \mu_i^{1}(\gamma)$ for $i = 1,2,\ldots,n$.

Proof: Lemma 2 yields for two parameters θ_1 and θ_2 with $0 \leq \theta_1 \leq \theta_2 \leq 1$:
$\bar{R}^{\theta_2}(\gamma) = \bar{R}^{\theta_1}(\gamma) + \gamma\,(\theta_2 - \theta_1)\,uu'$.
For this case Lemma 3 states:

(2.1.10) $\mu_i^{\theta_1}(\gamma) \leq \mu_i^{\theta_2}(\gamma) \leq \mu_{i+1}^{\theta_1}(\gamma)$ for $i = 1,2,\ldots,n-1$.

This means for an arbitrary $\theta \in [0,1]$:

(2.1.11) $\mu_i^{0}(\gamma) \leq \mu_i^{\theta}(\gamma) \leq \mu_{i+1}^{0}(\gamma)$ and

(2.1.12) $\mu_i^{\theta}(\gamma) \leq \mu_i^{1}(\gamma) \leq \mu_{i+1}^{\theta}(\gamma)$ for $i = 1,2,\ldots,n-1$

Combination of (2.1.11) and (2.1.12) gives the theorem ∎

Theorem 4. Let $\bar{R}^{\theta}(\gamma) = D^{\theta}(R,\gamma,z,z)$ be given by (2.1.6) for a fixed positive definite matrix R and $z \in E^n$, $z \neq 0$. The eigenvalues of R and $\bar{R}^{\theta}(\gamma)$ are respectively $\lambda_1 \leq \lambda_2 \leq \ldots\ldots \leq \lambda_n$ and $\mu_1^{\theta}(\gamma) \leq \mu_2^{\theta}(\gamma) \leq \ldots\ldots \leq \mu_n^{\theta}(\gamma)$.
Then, provided that $\theta \in [0,1]$ and $\gamma > 0$ there are three possible cases:

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

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

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

Proof. First we consider the case $\bar{R}^0(1)$ (the DFP-update). Then the theorem follows from the substitution A=R, $B=\bar{R}^0(1)$ and r=z in theorem 2. Secondly the case $\bar{R}^1(1)$ (BFGS). Lemma 1 states: $[\bar{R}^1(1)]^{-1} = D^0(R^{-1},1,z,z)$. The eigenvalues of R^{-1} and $[\bar{R}^1(1)]^{-1}$ are respectively,

$$1/\lambda_n \leq 1/\lambda_{n-1} \leq \ldots \leq 1/\lambda_1 \text{ and } 1/\mu_n^1(1) \leq 1/\mu_{n-1}^1(1) \leq \ldots \leq 1/\mu_1^1(1).$$

Application of theorem 2 for these eigenvalues and the inverse of these relations gives the proof for $\bar{R}^1(1)$.
Using the results for $\bar{R}^0(1)$ and $\bar{R}^1(1)$ we can extend the theorem to $\bar{R}^0(\gamma)$ and $\bar{R}^1(\gamma)$ with $\gamma>0$ using Lemma 1.

Lemma 1 gives: $D^\theta(R,\gamma,z,z) = D^\theta(\gamma R,1,z,z)$.

The eigenvalues of γR are $\gamma\lambda_1 \leq \gamma\lambda_2 \leq \ldots \leq \gamma\lambda_n$. Substitution of $\{\gamma\lambda_1,\ldots,\gamma\lambda_n\}$ for $\{\lambda_1,\ldots,\lambda_n\}$ in the preceding part of the proof extends the results to the cases $\theta = 0$ and $\theta = 1$ with $\gamma > 0$

Further theorem 3 gives for all $\theta \in [0,1]$ and $\gamma > 0$

$$\mu_i^0(\gamma) \leq \mu_i^\theta(\gamma) \leq \mu_i^1(\gamma) \quad , i = 1,2,\ldots,n.$$

Herefrom follows directly that any inequality satisfied by both $\mu_i^0(\gamma)$ and $\mu_i^1(\gamma)$ is also satisfied by $\mu_i^\theta(\gamma)$ and the theorem is proved for all $\theta \in [0,1]$ and $\gamma>0$.

Corollary 1. With $\bar{R}^\theta(\gamma)$, R, λ_i and $\mu_i^\theta(\gamma)$ as in theorem 4, then

$$|\mu_i^\theta(1) - 1| \leq |\lambda_i - 1| \quad \text{for } i = 1,2,\ldots,n.$$

Proof. In all cases of theorem 4 we have

$$\lambda_i \leq \mu_i^\theta(1) \leq 1 \text{ and /or } 1 < \mu_i^\theta(1) \leq \lambda_i \text{ for all } i,$$

which makes the result obvious.

Corollary 2. With the same notation as corollary 1 and $\kappa(.)$ as the condition of a matrix, then for $\theta \in [0,1]$ and $\gamma>0$ there are three cases:

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

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

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

The proof follows immediately from theorem 4.

As we are looking for matrices R_k with decreasing condition number, the last case of corollary 2 is most interesting. Hence we are interested in rules to find scaling factors γ_k which satisfy $1/\lambda_n \leq \gamma \leq 1/\lambda_1$. In chapter (2.1.3) will be shown that all γ_k as defined in (1.6) satisfy this relation.

Figure 2 gives an illustration of the difference of the eigenvalues of R_0 and R_1 in the example of ch. 2.1.

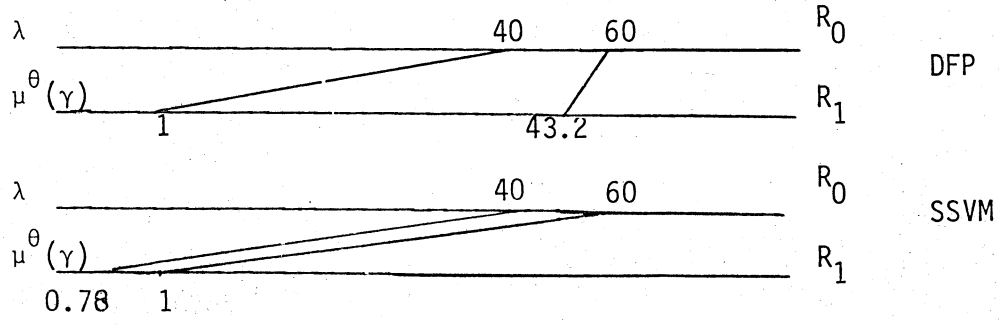


Figure 2

2.1.3 Restrictions on the parameters.

In the preceding paragraphs several times the sense of proper scaling of the objective function was mentioned. An alternative way to obtain scaling of the objective function by a constant ξ is to multiply the inverse Hessian approximation by ξ before updating. In both cases the matrix $R_k = H^{\frac{1}{2}} D_k H^{\frac{1}{2}}$ will be multiplied by ξ.

As Lemma 1 stated:

$$D_{k+1} = D^{\theta_k}(\xi D_k, 1, p_k, q_k) = D^{\theta_k}(D_k, \xi, p_k, q_k),$$

this scaling can be implemented in SSVM-algorithms by simply choosing $\gamma_k = \xi$. So γ_k can be interpreted as a scaling factor and varying γ_k has the effect of rescaling the objective function.
Updating formulae as defined by (1.4) and (1.5) are said to be self-scaling, if for any fixed positive definite quadratic function with Hessian H the parameters θ_k, γ_k are automatically selected such that $\kappa(R_{k+1}) \leq \kappa(R_k)$ for all k, independently of the updating vector p_k. (Oren and Luenberger, [1974]).
The parameter θ_k: the condition $\theta_k \in [0,1]$ was applied in a number of above mentioned theorems. The necessity of this condition follows from a counterexample due to Fletcher, [1970], in which both $\theta \leq -\varepsilon$ and $\theta \geq 1 + \varepsilon$ for $\varepsilon \in (0,1)$ lead to a contradiction.

Consider a problem with:

$$R = \begin{pmatrix} 1+\varepsilon & \varepsilon^{\frac{1}{2}} \\ \varepsilon^{\frac{1}{2}} & \varepsilon \end{pmatrix} \qquad z = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad , \; 0 < \varepsilon < 1$$

The eigenvalues of R are:

$$\lambda_1 = \zeta$$

$$\lambda_2 = 1 + 2\varepsilon - \zeta$$

$$\zeta = \tfrac{1}{2}\{(1+2\varepsilon)-(1+4\varepsilon)^{\frac{1}{2}}\}$$

As ζ is strictly positive of order ε^2 we know $\zeta < 1 < 1+2\varepsilon - \zeta$
First let $\gamma = 1$
Then substitution in (2.1.6) yields:

$\bar{R}^{\theta}(1) = \begin{pmatrix} \varepsilon+\theta & 0 \\ 0 & 1 \end{pmatrix}$, which means that for $\theta \leq -\varepsilon$ the matrix $\bar{R}^{\theta}(1)$ is either singular or has a negative eigenvalue.
This contradicts the positive definiteness of D_k for all k.

For $\gamma = 1/\zeta$ the relation $1/\lambda_n \leq \gamma \leq 1/\lambda_1$ still holds.
This time substitution in (2.1.6) gives:

$$\bar{R}^{\theta}(\gamma) = \begin{pmatrix} (\varepsilon+\theta)/\zeta & 0 \\ 0 & 1 \end{pmatrix}$$

Then for $\theta \geq 1+\varepsilon$ we have:

$\kappa(\bar{R}^{\theta}(\gamma)) \geq \frac{1+2\varepsilon}{\zeta} > \frac{1+2\varepsilon-\zeta}{\zeta} = \kappa(R)$, which contradicts corollary 2 of theorem 4.

The parameter γ_k. Our goal is to meet the requirement (iii) of corollary 2 of theorem 4, to ensure a decrease in the value of the condition number of the matrix R_k. As it is rather expensive to evaluate the eigenvalues themselves we are interested in scaling factors γ_k based on currently available information and still satisfying part (iii) of corollary 2. Therefore Oren, [1974a] introduces a convex class of scaling factors:
Let D be a nonsingular symmetric matrix and $p,q \in E^n$, $p \neq 0$, $q \neq 0$.
Then the scalar $\gamma^{\phi}(D,p,q)$ is defined by:

$$(2.1.13) \qquad \gamma^{\phi}(D,p,q) = (1-\phi)\,\frac{p'q}{q'Dq} + \phi\,\frac{p'g}{g'Dq} \; .$$

We intend to show that the thus defined scalars $\gamma^{\phi}(D,p,q)$ for $\phi \in [0,1]$ meet our requirements. It is clear from (2.1.13) that $\gamma^{\phi}(D,p,q)$ is strictly positive if D is positive definite, $p'q > 0$ and $\phi \in [0,1]$. There remains to be proved that

$$\lambda_1 \leq \frac{1}{\gamma^{\phi}(D,p,q)} \leq \lambda_n$$

Theorem 5. Let $p,q \in E^n, p \neq 0, q \neq 0$ with $p'q > 0$, D and H are positive definite symmetric matrices and R is a positive definite matrix, such that

$$q = Hp$$

$$R = H^{\frac{1}{2}}DH^{\frac{1}{2}}$$

then for all $\phi \in [0,1]$ there holds:

$$1/\lambda_n \leq \gamma^{\phi}(D,p,q) \leq 1/\lambda_1,$$

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

Proof. First we rewrite (2.1.13) as:

(2.1.14) $$\gamma^{\phi}(D,p,q) = (1-\phi)\frac{D'q}{q'Dq} + \phi\frac{p'D^{-1}p}{p'q}$$, using the relation

$$\frac{p'D^{-1}p}{p'q} = \frac{p'g}{g'Dq} \quad \text{for } p = -\alpha Dg$$

As $\gamma^{\phi}(D,p,q)$ is defined as a convex combination of $\gamma^{1}(D,p,q)$ and $\gamma^{0}(D,p,q)$ it suffices to prove the theorem for these two extreme values.

Let $z = H^{\frac{1}{2}}p$ and $r = H^{-\frac{1}{2}}q$. Then $\phi = 0$ in (2.1.14) yields:

$\gamma^{0}(D,p,q) = \frac{z'z}{z'Rz}$. Further: $\lambda_1 z'z \leq z'Rz \leq \lambda_n z'z$, which leads directly to

$1/\lambda_n \leq \gamma^{0}(D,p,q) \leq 1/\lambda_1$

The case $\phi = 1$ gives in (2.1.13)

$\gamma^{1}(D,p,q) = \frac{r'R^{-1}r}{r'r}$. Substitution in $\frac{r'r}{\lambda_n} \leq \frac{r'r}{\lambda_1}$, yields the result $1/\lambda_n \leq \gamma^{1}(D,p,q) \leq 1/\lambda_1$

Conclusion: $\gamma^{\phi}(D,p,q)$ as defined in (2.1.13) are suitable scaling factors for SSVM-algorithms. Those scaling factors can be found using currently available information about p,q,g and D, and make the algorithm invariant under scaling of the objective function and/or variables (numerical instabilities as reported by Bard, [1968] will not occur).
Theorem 6 proves this statement.

Theorem 6. Let D_k, x_k, θ_k and ϕ_k be defined as above. $\{D_k\}$, $\{x_k\}$ and $\{\hat{D}_k\}$, $\{\hat{x}_k\}$ are the sequences generated in the application of the algorithm on $f(x)$ and $\alpha f(\beta x)$ respectively ($\alpha > 0$, $\beta > 0$). For the initialisation we assume $\hat{D}_0 = \delta D_0$ ($\delta > 0$) and $\beta\hat{x}_0 = x_0$. Both applications use the same sequences $\{\theta_k\}$ and $\{\phi_k\}$. Then, for $f \in C^2$,

$$\hat{D}_k = D_k/\alpha\beta^2 \text{ and } \hat{x}_k = x_k/\beta \text{ for all } k$$

Proof. For all $\hat{x}_k \in E^n$ we have $\hat{g}_k = \nabla_{\hat{x}}(\alpha f(\beta\hat{x}_k))$

$$= \alpha\beta\nabla_x f(x_k)$$

$$= \alpha\beta\, g_k(x_k)$$

So the first search direction is the same and $\hat{x}_1 = x_1/\beta$

Hence $\hat{p}_0 = p_0/\beta$, $\hat{g}_0 = \alpha\beta g_0$ and $\hat{q}_0 = \alpha\beta q_0$

Substitution in the updating formulae (1.4.) - (1.6.) gives:

$\hat{D}_1 = D_1/\alpha\beta^2$. The remaining part of the proof is by induction. ■

Conclusion: we found that for $k = 1,2,\ldots$ there holds:

1) If $\gamma_k > 0$, $\theta_k \geq 0$, $p_k'q_k > 0$ for all k, then all matrices D_k are positive definite.

2) The derived scaling factors γ_k guarantee a sequence of matrices R_k, whose condition numbers form a decreasing sequence.

3) As the algorithm is self-scaling it is invariant under scaling of the objective function and/or the variables.

4) If $\theta_k \in \{0,1\}$ the inverse Hessian approximation constantly improves and the algorithm more and more resembles Newton's method. This will provide for a good local convergence rate even without linesearch. Thus we expect to find in our experiments good results with inexact linesearches while simultaneously the influence of roundoff errors will decrease.

2.2. Optimally Conditioned Self-Scaling Algorithms (OCSS)

It will be clear from ch. 2.1. that there is still a wide variety of possible choices of the parameters γ and θ. That's why Oren, [1974b] investigated the selection of these parameters. Moreover he suggested two rules for the parameters γ and θ. According to one of these rules the parameter γ, expressing scaling of the objective function is selected as close as possible to unity and θ is chosen such to offset an estimated bias in $\text{Det}(D_kH)$ relative to unity. The main result of Oren, [1974b] is that it is showed that still further improvement can be achieved by a proper selection of these parameters γ and θ.

In their paper Oren and Spedicato [1976] give a theory to impose a sharper bound on the condition number of the inverse Hessian approximation (D_k). For these algorithms we skip the theoretical backgrounds which can be found in Oren and Spedicato, [1976] and proceed with mentioning the results, using the following brief notation:

$$\sigma = p'q$$

$$\tau = q'Dq$$

$$\pi = p'D^{-1}p = p'q.\ g'p/g'Dq$$

An optimal parameter, $\hat{\theta}$ for which $\bar{D}$ is optimally conditioned, is:

(2.2.1) $$\hat{\theta} = \sigma(\pi - \gamma\sigma)/\{\gamma(\pi\tau - \sigma^2)\}$$

This optimal parameter $\hat{\theta}$ satisfies:

$$0 \leq \hat{\theta} \leq 1 \quad \text{for} \quad \sigma/\tau \leq \gamma \leq \pi/\sigma$$

$$\hat{\theta} = 0 \text{ for } \gamma = \pi/\sigma$$

$$\hat{\theta} = 1 \text{ for } \gamma = \sigma/\tau$$

One can easily see that still any value of $\gamma \in [\sigma/\tau, \pi/\sigma]$ can be chosen. Oren and Spedicato suggested the following two switches:

Switch 1

If $\pi/\sigma \leq 1$, choose $\gamma = \pi/\sigma$ and $\theta = 0$

If $\sigma/\tau \geq 1$, choose $\gamma = \sigma/\tau$ and $\theta = 1$

If $\sigma\ \tau \leq 1 \leq \pi/\sigma$, choose $\gamma = 1$ and

$$\theta = \sigma(\pi - \sigma)/(\pi\tau - \sigma^2)$$

Switch 2

$$\gamma = (\pi/\tau)^{\frac{1}{2}} \text{ and } \theta = 1/\{1 + (\tau\pi/\sigma^2)^{\frac{1}{2}}\}$$

This switch can be found by substituting $\gamma = (\pi/\tau)^{\frac{1}{2}}$ into 2.2.1.

This switch has the pleasant property that the update of the inverse equals the inverse of the update.

Beside these two switch algorithms two other switching algorithms (Oren, [1974b]) were implemented:

Switch 3

If $\pi/\sigma \leq 1$, choose $\gamma = \pi/\sigma$ and $\theta = 0$

If $\sigma/\tau \geq 1$, choose $\gamma = \sigma/\tau$ and $\theta = 1$

If $\sigma/\tau \leq 1 \leq \pi/\sigma$, choose $\gamma = 1$ and

$\theta = \sigma(\tau - \sigma)/(\pi\tau - \sigma^2)$

Switch 4

$\gamma = \pi/\tau$ and $\theta = \frac{1}{2}$

2.3. Shanno and Phua

In their paper Shanno and Phua, [1978] discuss matrix conditioning and non-linear optimization, focussed on scaling algorithms for unconstrained optimization. They state that SSVM algorithms perform poorly compared to initially scaled BFGS and Davidon's, [1975] algorithm with only one exception formed by the class of homogeneous functions. Using the definition of Jacobson and Oskman, [1970]: a homogeneous function can be written as:

$f(x) = \beta^{-1}(x - \hat{x})\ \nabla f(x) + f(\hat{x})$, in which $\hat{x}$ is the minimizer and β is called the degree of homogeneity. A typical homogeneous function of degree four is Oren's Quartic function:

$f(x) = (x'Ax)^2$, which we also used as a testfunction.

As mentioned before the BFGS algorithm is a SSVM algorithm with $\gamma_k = \theta_k = 1$ for all k.
Shanno and Phua proposed initial scalings of the BFGS algorithm. They meant by initial scaling that $H_0 = I_n$ is multiplied by a scalar before it is updated. The two strategies for initial scaling are:

1) $\gamma_0 = \alpha_0$ (α_0 is the stepsize found in the first linesearch)
As a motivation for choosing α_0 is that it corresponds with scaling the objective function with a factor $\gamma_0 = \alpha_0$. Note that we proved in theorem 6 that the scaling factor $\gamma_k = \alpha_k$ can also be deduced from the following composite divice for updating:
first the matrix D_k is replaced by $\alpha_k D_k$ and then D_{k+1} is calculated from (1.4) - (1.6).

2) $\gamma_0 = \frac{p_0'q_0}{q_0'D_0q_0}$, which means that $\phi_0 = 0$ in 1.6.

As stated before $\theta = 1$ in the BFGS update; assuming that this is the optimal value for θ then with 2.2.1 $\gamma_i = \frac{p'q}{q'Dq}$ for $i = 0$.

3. Numerical Experiments

The goal of the numerical experiments is to compare the efficiency of developed quasi-Newton algorithms for unconstrained optimization, which apply different rules to solve the problem of bad scaling. In the experiments special attention is paid to the effect of increasingly bad scaling, the influence of the accuracy of the linesearch and of the dimension of the problem.

Further reasons to design these experiments are that reported numerical results in literature concern more or less different test batteries. Surprisingly, up to now ecperiments are not focussed on the main goal of these algorithms: their ability to attack badly scaled problems where the spectrum of eigenvalues of the matrix R_1 does not contain the unit element.

The next chapter describes the design of the experiments performed.

3.1. Algorithms Implemented

The flowchart given in figure 3 is a general representation of the implementation of the considered algorithms.

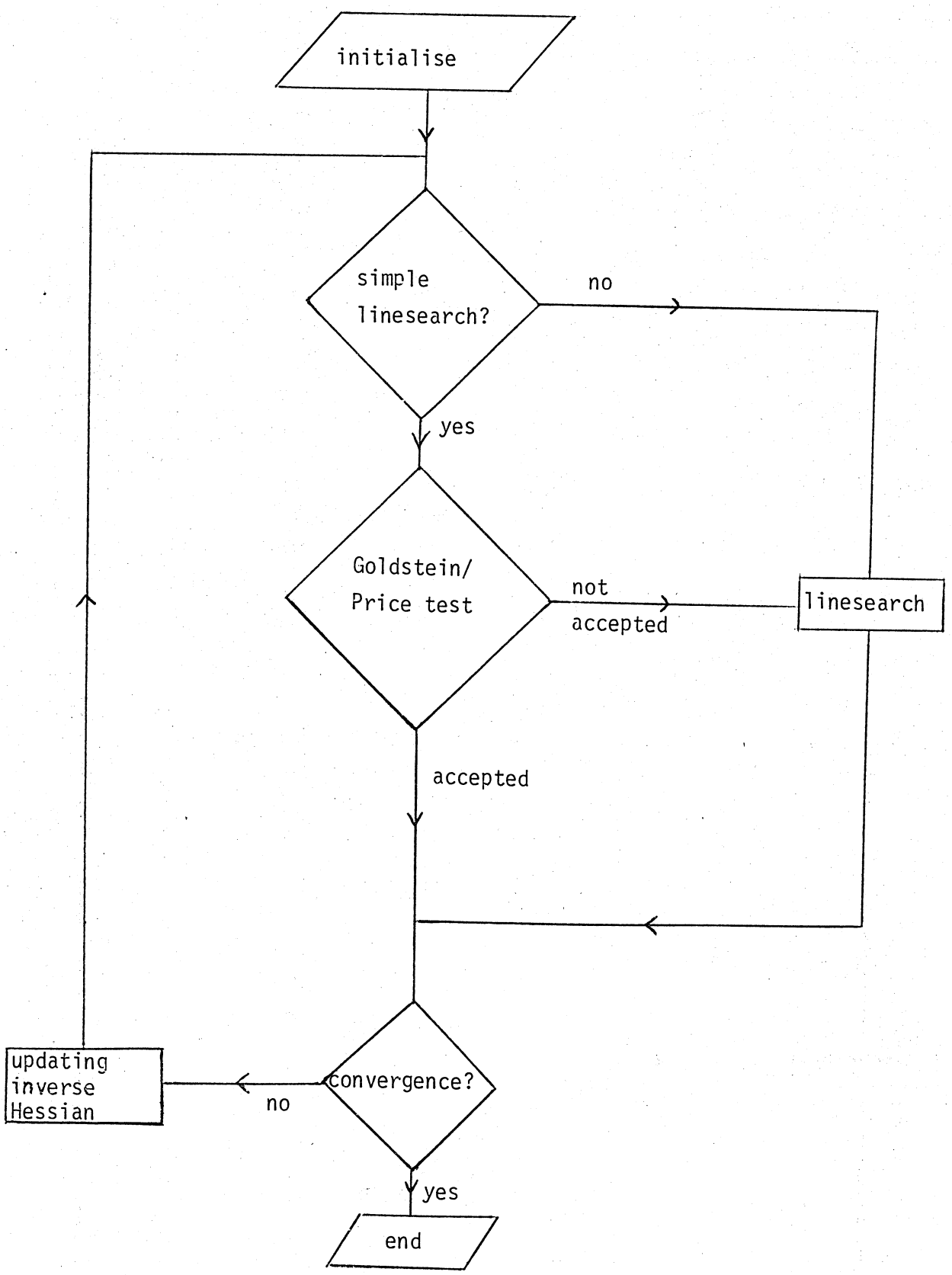

initialise
simple linesearch?
no
yes
Goldstein/ Price test
not accepted
linesearch
accepted
updating inverse Hessian
convergence?
no
yes
end

figure 3

The different algorithms are defined by particular choices for the linesearch 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, [1978]

For these algorithms the accuracy of the linesearch was varied. Also the effect of the test of Goldstein and Price, [1967] to avoid linesearches 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 program consists of a main program called SSVM which calls the subroutines CUBIC (linesearch) and UPDAT (updating inverse Hessian approximation).
Special remarks on the program:

SSVM: 1. The Goldstein and Price condition to test whether the Newton steplength'1' is acceptable or not is tested in the main program SSVM.

2. As theoretically D_{k+1} need no longer be positive definite if $p'q \leq 0$ we test this relation before calling UPDAT. If $p'q < 0$, D_{k+1} equals D_k (no updating). If this happens 10 times in executing one testproblem the run is terminated with a message. Only in the execution of testproblem 6 this occured.

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 distinct hardly solvable problems from unsolvable problems.

CUBIC This linesearch is described in appendix A. It is a bracketing process followed by cubic interpolations. Because of possible nonconvexity in the problems ($p'q < 0$ can occur!) the linesearch is safeguarded 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 $E = p'D^{-1}p$ which equals $E = \frac{p'g * p'q}{g'Dq}$, Oren, [1974a]
The latter expression in used in the computations as it is cheaper than the first. (In case of an exact linesearch we can use $E'' = \alpha^2 g'Dg$ or $E''' = \alpha p'q$).

3.2. The choice of testproblems, termination criteria and performance indicators

The subjects to be treated in this chapter are motivated by the necessity of a proper design of the experiments, in order to be able to draw correct conclusions from the figures which will be generated.
We treat these subjects in the order in which they occur in the title of this chapter.

Testproblems To meet our goal on the design of the numerical experiments, mentioned in the beginning of this chapter, we composed a collection of 12 testproblems, mentioned in appendix B. The testproblems, whose gradients are analytically given, are taken from literature. Necessary new problems are generated by varying parameters which influence the condition number of the testproblem and/or the dimension. Though the convergence properties of the developed algorithms are proved for convex minimization problems, usually test batteries, including ours, also contain nonconvex problems. For the moment we merely remark that recent research on global minimization algorithms to minimize nonconvex problems, Rinnooy Kan, [1979], provides for an entirely different, statistically oriented approach, in stead of simply applying one local search. The set of 12 testfunctions consists of:

1, 2, 3, 4: Increasingly badly scaled variations on 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 increasing extreme ill-conditioning on purely quadratic functions.

Terminationcriteria As a wide variety of these criteria is known and applied we had to make a choice and decided to stop iterating as soon as both following conditions are met:

$$|| g_k || \leq 10^{-6},$$

$$|| x_{k+1} - x_k || \leq 10^{-4}.$$

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

$$||f(x_{k+1}) - f(x_k)|| \leq || g_k || \, ||x_{k+1} - x_k||$$

so

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

Table III illustrates the still possible inaccuracy in x* under our stopping rules.

Table III

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	$.1102\ 10^{-2}$	$-.2708\ 10^{-3}$	$.185\ 10^{-13}$

A single component criterium as $|| f(x_{k+1}) - f(x_k) || \leq 10^{-10}$, as applied in Oren, [1974b], Oren and Spedicato [1976] and Shanno and Phua, [1978] locates x* still less accurately.
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. This means in the notation of (1.7) and (1.8):

$$| h_k(\alpha_{i+1}) - h_k(\alpha_i)| \leq \text{EPSCU}$$

Performance indicators Candidates for performance indicators are: number of function evaluations, number of iterations and required number of CPU-secs to solve a testproblem. In our terminology an iteration consists of the generation and exploration of a search direction. These three indicators are all mentioned in the tables with results. The number of required function evaluation was used as the main indicator. It corresponds directly with the number of iterations as the number of functionevaluations per iteration does not vary much. The main disadvantage of counting functionevaluations to solve the whole set of testproblems is that different objective functions are equally weighted though they may differ substantially in complexity: from the figures of tables IV and VI can be seen that one evaluation of the 30-dimensional Rosenbrock-function is approximately as expensive as five evaluations of the 2-dimensional Quartic function. This influence is compensated by considering separately classes of testfunctions, such as the higher dimensional ones, and drawing separate conclusions for those classes.
The required CPU-time gives additional information on the overhead of computations such as matrixmanipulations 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, which is more important, because of the multiprogramming facility.
We find 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 merely mention CPU-times in the next tables and do not draw further conclusions from them.

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 linesearch 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 evolving from experiment I and implementationsof the four Oren-Spedicato switches are compared. The parameter σ of the new applied 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 by Shanno and Phua, [1978].

Under the applied termination criteria 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 next tables.
The most relevant results are summarized in the next tables, using the following notation:

- # F : number of required function evaluations
- # IT : number of required iterations
- CPU : required CPU-time in secs
- F* : reached function value

The 25 algorithms evolving 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
```

Table IV: # F for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-1}

No Goldstein/Price test

algorithm / testfunction	1	2	3	4	5	6	7	8	9	10	11	12 B	13	14	15	16	17 C	18	19	20	21	22 A	23	24	25
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 F	1689	1925 F	1923 F	1944 F	1528	1367	1408	1478	1571	1451	1245	1297	1449	1485	1388	1247	1325	1389	1405	1386	1227	1310	1319	1584

29

Table V: # IT for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-1}

No Goldstein/Price test

algorithm / testfunction	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
Ros(c=1)	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
($c=10^2$)	24	23	22	23	24	23	21	23	26	23	22	23	22	23	22	23	26	23	23	23	24	23	22	23	24
($c=10^4$)	111	88	87	88	95	88	94	96	99	97	90	92	95	106	104	84	97	106	103	107	93	100	103	102	134
(n=10)	57	63	67	70	81	57	54	58	60	68	58	54	54	59	64	63	54	57	52	57	63	55	55	54	56
(n=30)	F	212	F	F	F	165	148	176	199	232	191	126	145	165	186	201	122	127	146	161	203	122	127	136	151
Quartic(n=2)	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
(n=10)	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	15	16	17	17	17	16	16	16	17	16
(n=30)	26	26	26	26	25	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26	26
Hilbert(n=2)	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
(n=4)	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
(n=6)	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5

31

Table VI: CPU for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-1}. No Goldstein/Price test

algorithm / testfunction	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
Ros(c=1)	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.06	0.03	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.03	0.03	0.04	0.03	0.04	0.03	0.03
($c=10^2$)	0.17	0.16	0.14	0.14	0.15	0.16	0.14	0.15	0.16	0.15	0.14	0.15	0.15	0.15	0.15	0.15	0.17	0.16	0.16	0.16	0.15	0.15	0.15	0.15	0.16
($c=10^4$)	0.85	0.48	0.51	0.48	0.45	0.45	0.45	0.48	0.47	0.45	0.46	0.43	0.45	0.51	0.56	0.44	0.51	0.55	0.55	0.59	0.48	0.53	0.49	0.49	0.73
(n=10)	1.53	1.59	1.60	1.70	1.96	1.38	1.32	1.37	1.45	1.70	1.35	1.37	1.29	1.42	1.60	1.51	1.33	1.42	1.31	1.44	1.48	1.38	1.36	1.29	1.32
(n=30)	17.89	32.03	35.52	35.43	37.19	24.13	22.77	25.95	29.30	34.82	27.89	18.94	21.60	24.50	27.37	29.24	18.16	18.98	21.65	23.80	29.17	18.39	18.94	20.29	22.55
Quartic(n=20)	0.06	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.07	0.05	0.05	0.05	0.06	0.06	0.06
(n=10)	0.46	0.46	0.46	0.45	0.48	0.49	0.46	0.43	0.43	0.45	0.43	0.43	0.43	0.48	0.48	0.41	0.41	0.46	0.43	0.43	0.41	0.41	0.41	0.43	0.41
(n=30)	3.72	3.76	3.79	3.81	3.69	3.74	4.01	3.92	3.78	3.78	3.73	3.83	3.79	3.97	3.98	3.87	3.82	3.96	3.78	3.85	3.82	3.82	3.91	3.81	3.77
Hilbert(n=2)	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
(n=4)	0.08	0.08	0.08	0.08	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.09	0.07	0.07	0.07	0.07
(n=6)	0.14	0.14	0.15	0.15	0.15	0.15	0.16	0.15	0.16	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.17	0.14	0.14	0.14	0.14	0.14

Table VII: F* for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-1}

No Goldstein/Price test

algorithm / testfunction	1	2	3	4	5	6	7	8	9	10	11	12
Ros(c=1)	$.744\ 10^{-18}$	$.245\ 10^{-16}$	$.140\ 10^{-16}$	$.562\ 10^{-17}$	$.853\ 10^{-18}$	$.245\ 10^{-16}$	$.700\ 10^{-17}$	$.916\ 10^{-20}$	$.197\ 10^{-16}$	$.103\ 10^{-16}$	$.140\ 10^{-16}$	$.916\ 20^{-20}$
$(c=10^2)$	$.789\ 10^{-30}$	$.783\ 10^{-19}$	$.133\ 10^{-27}$	$.115\ 10^{-22}$	$.415\ 10^{-25}$	$.783\ 10^{-19}$	$.197\ 10^{-17}$	$.342\ 10^{-24}$	$.170\ 10^{-24}$	$.783\ 10^{-23}$	$.185\ 10^{-18}$	$.342\ 10^{-24}$
$(c=10^4)$	$.209\ 10^{-25}$	$.508\ 10^{-19}$	$.105\ 10^{-28}$	$.436\ 10^{-24}$	$.675\ 10^{-22}$	$.127\ 10^{-25}$	$.462\ 10^{-20}$	$.147\ 10^{-17}$	$.425\ 10^{-20}$	$.149\ 10^{-23}$	$.979\ 10^{-22}$	$.222\ 10^{-23}$
(n=10)	$.543\ 10^{-17}$	$.204\ 10^{-17}$	$.234\ 10^{-17}$	$.823\ 10^{-18}$	$.127\ 10^{-16}$	$.174\ 10^{-18}$	$.104\ 10^{-17}$	$.892\ 10^{-17}$	$.731\ 10^{-17}$	$.108\ 10^{-17}$	$.220\ 10^{-17}$	$.313\ 10^{-17}$
(n=30)	F	$.404\ 10^{-16}$	F	F	F	$.128\ 10^{-16}$	$.856\ 10^{-17}$	$.614\ 10^{-16}$	$.609\ 10^{-16}$	$.902\ 10^{-16}$	$.538\ 10^{-17}$	$.110\ 10^{-16}$
Quartic(n=2)	$.117\ 10^{-16}$	$.113\ 10^{-16}$	$.110\ 10^{-16}$	$.107\ 10^{-16}$	$.103\ 10^{-16}$	$.113\ 10^{-16}$	$.110\ 10^{-16}$	$.107\ 10^{-16}$	$.103\ 10^{-16}$	$.999\ 10^{-17}$	$.110\ 10^{-16}$	$.107\ 10^{-16}$
(n=10)	$.728\ 10^{-19}$	$.903\ 10^{-19}$	$.102\ 10^{-18}$	$.108\ 10^{-18}$	$.111\ 10^{-18}$	$.490\ 10^{-19}$	$.684\ 10^{-19}$	$.859\ 10^{-19}$	$.984\ 10^{-19}$	$.106\ 10^{-18}$	$.222\ 10^{-19}$	$.459\ 10^{-19}$
(n=30)	$.127\ 10^{-15}$	$.112\ 10^{-15}$	$.118\ 10^{-15}$	$.125\ 10^{-15}$	$.183\ 10^{-14}$	$.187\ 10^{-15}$	$.137\ 10^{-15}$	$.115\ 10^{-15}$	$.115\ 10^{-15}$	$.128\ 10^{-15}$	$.265\ 10^{-15}$	$.177\ 10^{-15}$
Hilbert(n=2)	$.104\ 10^{-31}$	$.971\ 10^{-32}$	$.920\ 10^{-32}$	$.929\ 10^{-32}$	$.917\ 10^{-32}$	$.971\ 10^{-32}$	$.921\ 10^{-32}$	$.975\ 10^{-32}$	$.975\ 10^{-32}$	$.997\ 10^{-32}$	$.921\ 10^{-32}$	$.975\ 10^{-32}$
(n=4)	$.619\ 10^{-16}$	$.619\ 10^{-16}$	$.615\ 10^{-16}$	$.621\ 10^{-16}$	$.620\ 10^{-16}$	$.619\ 10^{-16}$	$.618\ 10^{-16}$	$.615\ 10^{-16}$	$.616\ 10^{-16}$	$.616\ 10^{-16}$	$.619\ 10^{-16}$	$.617\ 10^{-16}$
(n=6)	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$

32

33

Table VII: Continuation

algorithm / testfunction	13	14	15	16	17	18
Ros(c=1)	$.153\ 10^{-16}$	$.428\ 10^{-17}$	$.192\ 10^{-26}$	$.562\ 10^{-17}$	$.197\ 10^{-16}$	$.428\ 10^{-17}$
($c=10^2$)	$.105\ 10^{-24}$	$.145\ 10^{-21}$	$.158\ 10^{-27}$	$.115\ 10^{-22}$	$.169\ 10^{-24}$	$.145\ 10^{-21}$
($c=10^4$)	$.130\ 10^{-30}$	$.250\ 10^{-20}$	$.125\ 10^{-17}$	$.316\ 10^{-29}$	$.188\ 10^{-28}$	$.106\ 10^{-17}$
(n=10)	$.257\ 10^{-18}$	$.124\ 10^{-16}$	$.898\ 10^{-18}$	$.463\ 10^{-17}$	$.123\ 10^{-17}$	$.107\ 10^{-16}$
(n=30)	$.263\ 10^{-16}$	$.468\ 10^{-17}$	$.278\ 10^{-16}$	$.120\ 10^{-16}$	$.186\ 10^{-16}$	$.168\ 10^{-16}$
Quartic(n=2)	$.103\ 10^{-16}$	$.999\ 10^{-17}$	$.966\ 10^{-17}$	$.107\ 10^{-16}$	$.103\ 10^{-16}$	$.999\ 10^{-17}$
(n=10)	$.645\ 10^{-19}$	$.818\ 10^{-19}$	$.947\ 10^{-19}$	$.530\ 10^{-16}$	$.374\ 10^{-17}$	$.656\ 10^{-19}$
(n=30)	$.135\ 10^{-15}$	$.124\ 10^{-15}$	$.113\ 10^{-15}$	$.260\ 10^{-15}$	$.222\ 10^{-15}$	$.170\ 10^{-15}$
Hilbert(n=2)	$.950\ 10^{-32}$	$.958\ 10^{-32}$	$.109\ 10^{-31}$	$.929\ 10^{-32}$	$.105\ 10^{-31}$	$.958\ 10^{-32}$
(n=4)	$.619\ 10^{-16}$	$.618\ 10^{-16}$	$.617\ 10^{-16}$	$.614\ 10^{-16}$	$.611\ 10^{-16}$	$.613\ 10^{-16}$
(n=6)	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$

19	20	21	22	23	24	25
$.283\ 10^{-16}$	$.182\ 10^{-16}$	$.853\ 10^{-18}$	$.103\ 10^{-16}$	$.192\ 10^{-26}$	$.182\ 10^{-16}$	$.207\ 10^{-26}$
$.880\ 10^{-22}$	$.605\ 10^{-17}$	$.415\ 10^{-25}$	$.783\ 10^{-23}$	$.159\ 10^{-27}$	$.605\ 10^{-17}$	.0.0
$.470\ 10^{-25}$	$.825\ 10^{-23}$	$.272\ 10^{-24}$	$.146\ 10^{-24}$	$.103\ 10^{-29}$	$.259\ 10^{-17}$	$.120\ 10^{-30}$
$.804\ 10^{-18}$	$.127\ 10^{-16}$	$.314\ 10^{-17}$	$.170\ 10^{-16}$	$.522\ 10^{-17}$	$.126\ 10^{-16}$	$.294\ 10^{-17}$
$.675\ 10^{-17}$	$.130\ 10^{-16}$	$.223\ 10^{-16}$	$.707\ 10^{-17}$	$.147\ 10^{-16}$	$.852\ 10^{-17}$	$.136\ 10^{-16}$
$.966\ 10^{-17}$	$.933\ 10^{-17}$	$.103\ 10^{-16}$	$.999\ 10^{-17}$	$.966\ 10^{-17}$	$.933\ 10^{-17}$	$.900\ 10^{-17}$
$.616\ 10^{-19}$	$.787\ 10^{-19}$	$.174\ 10^{-16}$	$.833\ 10^{-17}$	$.860\ 10^{-18}$	$.125\ 10^{-18}$	$.106\ 10^{-16}$
$.133\ 10^{-15}$	$.120\ 10^{-15}$	$.264\ 10^{-15}$	$.273\ 10^{-15}$	$.226\ 10^{-15}$	$.167\ 10^{-15}$	$.131\ 10^{-15}$
$.969\ 10^{-32}$	$.991\ 10^{-32}$	$.988\ 10^{-32}$	$.933\ 10^{-32}$	$.107\ 10^{-31}$	$.100\ 10^{-31}$	$.937\ 10^{-32}$
$.614\ 10^{-16}$	$.619\ 10^{-16}$	$.624\ 10^{-16}$	$.611\ 10^{-16}$	$.620\ 10^{-16}$	$.620\ 10^{-16}$	$.617\ 10^{-16}$
$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$

34

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

algorithm / testfunction	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
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	430	429	543	433	436	434
(n=10)	250	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 F	2238 F	2218 F	2236 F	2212 F	1922	1773	1862	2203 F	2297 F	1866	1650	1792	1957	1900	1896	1646	1829	1734	1835	1934	1720	1645	1709	1776

Table IX: # F for 25 (ϕ,θ)-combinations. Accuracy linesearch 10^{-6}

No Goldstein/Price test

algorithm / testfunction	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
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	529	534	520	526	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 F	2370 F	2372 F	2381 F	2355 F	2341 F	2015	2147	2384 F	2346 F	2317 F	1908	1996	2056	2322 F	2307 F	2091	1898	1957	2084	2333 F	1897	1871	1942	1976

Table X: # F for the 3 best (ϕ,θ)-combinations and the 4 Oren-Spedicato switches. Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	A	B	C	SWI	SWII	SWIII	SWIV
Ros(c=1)	28	22	29	21	21	21	17
($c=10^2$)	110	106	114	111	111	98	493
($c=10^4$)	346	371	347	358	350	364	F
(n=10)	128	133	124	130	117	166	F
(n=30)	237	264	256	225	268	312	F
Quartic(n=2)	38	38	38	38	38	38	52
(n=10)	47	47	47	48	47	48	F
(n=30)	52	53	53	53	53	53	F
Hilbert(n=2)	13	13	13	13	13	13	14
(n=4)	28	28	28	28	28	28	18
(n=6)	27	27	27	27	27	27	16
Σ	1054	1102	1076	1052	1073	1168	

Table XI: # IT for the 3 best (ϕ,θ)-combinations and the 4 Oren-Spedicato switches. Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	A	B	C	SWI	SWII	SWIII	SWIV
Ros(c=1)	10	10	10	9	9	9	10
($c=10^2$)	29	31	32	33	33	31	54
($c=10^4$)	148	167	164	165	160	156	F
(n=10)	79	91	72	81	77	100	F
(n=30)	157	162	160	176	184	233	F
Quartic(n=2)	24	24	24	24	24	24	31
(n=10)	33	33	33	34	33	34	F
(n=30)	38	39	39	39	39	39	F
Hilbert(n=2)	3	3	3	3	3	3	3
(n=4)	4	4	4	4	4	4	5
(n=6)	5	5	5	5	5	5	5

Table XII: CPU for the 3 best (ϕ,θ)-combinations and the 4 Oren-Spedicato switches. Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	A	B	C	SWI	SWII	SWIII	SWIV
Ros(c=1)	0.05	0.06	0.04	0.03	0.04	0.03	0.03
($c=10^2$)	0.14	0.13	0.15	0.13	0.14	0.12	0.38
($c=10^4$)	0.54	0.61	0.58	0.54	0.60	0.53	F
(n=10)	1.39	1.60	1.30	1.46	1.32	1.71	F
(n=30)	19.69	20.28	20.79	20.47	21.71	26.62	F
Quartic(n=2)	0.09	0.07	0.09	0.09	0.09	0.08	0.11
(n=10)	0.57	0.66	0.67	0.62	0.63	0.65	F
(n=30)	4.52	4.70	4.81	4.45	4.37	4.49	F
Hilbert(n=2)	0.02	0.02	0.04	0.02	0.02	0.02	0.02
(n=4)	0.08	0.07	0.09	0.08	0.08	0.08	0.08
(n=6)	0.15	0.14	0.15	0.17	0.18	0.17	0.10

39

Table XIII: F*, obtained by the 3 best (ϕ,θ)-combinations and the 4 Oren-Spedicato switches.
Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	A	B	C	SWI	SWII	SWIII	SWIV
Ros(c=1)	$.179\ 10^{-17}$	$.528\ 10^{-17}$	$.376\ 10^{-18}$	$.513\ 10^{-19}$	$.513\ 10^{-19}$	$.513\ 10^{-19}$	$.687\ 10^{-20}$
($c=10^2$)	$.453\ 10^{-16}$	$.270\ 10^{-22}$	$.473\ 10^{-17}$	$.120\ 10^{-22}$	$.120\ 10^{-22}$	$.439\ 10^{-26}$	$.215\ 10^{-21}$
($c=10^4$)	$.132\ 10^{-26}$	$.586\ 10^{-21}$	$.596\ 10^{-20}$	$.210\ 10^{-22}$	$.123\ 10^{-13}$	$.724\ 10^{-21}$	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)	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$	$.151\ 10^{-10}$

Table XIV: # F for DFP, BFGS and 2 Shanno/Phua variants.
Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	DFP	BFGS	SH/PH I	SH/PH II
Ros(c=1)	22	17	15	15
($c=10^2$)	133	67	72	72
($c=10^4$)	300	227	225	228
(n=10)	262	112	116	113
(n=30)	F	381	244	231
Quartic(n=2)	60	41	58	58
(n=10)	479	128	172	172
(n=30)	716	253	415	414
Hilbert(n=2)	10	10	12	12
(n=4)	18	30	32	32
(n=6)	16	26	35	35
Σ	3016	1292	1396	1382

Table XV: # IT for DFP, BFGS and 2 Shanno/Phua variants.
Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	DFP	BFGS	SH/PH I	SH/PH II
Ros(c=1)	8	8	8	8
($c=10^2$)	32	34	34	34
($c=10^4$)	132	123	119	122
(n=10)	59	58	81	82
(n=30)	F	152	171	178
Quartic(n=2)	17	21	35	35
(n=10)	446	83	144	144
(n=30)	616	109	387	386
Hilbert(n=2)	3	3	3	3
(n=4)	6	6	4	4
(n=6)	6	6	5	5

Table XVI: CPU for DFP, BFGS and 2 Shanno/Phua variants.
Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$

algorithm / testfunction	DFP	BFGS	SH/PH I	SH/PH II
Ros(c=1)	0.03	0.02	0.02	0.02
($c=10^2$)	0.14	0.10	0.10	0.11
($c=10^4$)	0.45	0.36	0.35	0.47
(n=10)	1.30	0.97	1.28	1.35
(n=30)	F	17.99	19.87	21.04
Quartic(n=2)	0.08	0.07	0.11	0.10
(n=10)	7.17	1.46	2.45	2.38
(n=30)	69.91	13.14	44.14	43.49
Hilbert(n=2)	0.02	0.01	0.01	0.02
(n=4)	0.06	0.08	0.08	0.09
(n=6)	0.12	0.14	0.17	0.22

Table XVII: F^* , obtained by DFP, BFGS and 2 Shanno/Phua variants. Accuracy linesearch 10^{-1}. Goldstein/Price test with $\sigma = 10^{-1}$.

algorithm / testfunction	DFP	BFGS	SH/PH I	SH/PH II
Ros(c=1)	$.515\ 10^{-22}$	$.154\ 10^{-20}$	$.327\ 10^{-20}$	$.317\ 10^{-20}$
($c=10^2$)	$.481\ 10^{-20}$	$.262\ 10^{-21}$	$.323\ 10^{-21}$	$.323\ 10^{-21}$
($c=10^4$)	$.156\ 10^{-25}$	$.493\ 10^{-29}$	$.260\ 10^{-24}$	$.804\ 10^{-23}$
(n=10)	$.382\ 10^{-17}$	$.106\ 10^{-17}$	$.125\ 10^{-16}$	$.278\ 10^{-18}$
(n=30)	F	$.158\ 10^{-16}$	$.836\ 10^{-17}$	$.102\ 10^{-17}$
Quartic(n=2)	$.292\ 10^{-13}$	$.186\ 10^{-11}$	$.215\ 10^{-13}$	$.215\ 10^{-13}$
(n=10)	$.680\ 10^{-12}$	$.707\ 10^{-12}$	$.474\ 10^{-9}$	$.472\ 10^{-9}$
(n=30)	$.240\ 10^{-11}$	$.111\ 10^{-10}$	$.264\ 10^{-9}$	$.263\ 10^{-9}$
Hilbert	$.810\ 10^{-63}$	$.810\ 10^{-63}$	$.324\ 10^{-62}$	$.810\ 10^{-63}$
(n=4)	$.212\ 10^{-13}$	$.105\ 10^{-13}$	$.207\ 10^{-13}$	$.207\ 10^{-13}$
(n=6)	$.166\ 10^{-10}$	$.949\ 10^{-11}$	$.303\ 10^{-13}$	$.303\ 10^{-13}$

3.4 Discussion of the results

Experiment I

The numbers of function evaluations required by algorithms 1-25 for EPSCU = 10^{-1}, 10^{-3} and 10^{-6} are given in tables IV, VIII, IX. We selected the seven 'best' algorithms for respectively EPSCU = 10^{-1}, for 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 XVIII. We mention that obviously nontrivial values are to be preferred and that all three columns of table XVIII contain the same seven parameter combinations. From tables IV, VIII, IX and figure 4 can be deduced that increasing the accuracy makes all algorithms more expensive from which we conclude that EPSCU = 10^{-1} is to be preferred. This confirms our remarks in Ch. 2.1. on inexact linesearches. These arguments led to the following choice of three 'best' parameter combinations evolving from experiment I on our set of testproblems:

$\phi = 1.$, $\theta = .25$: algorithm 22
$\phi = .50$, $\theta = .25$: algorithm 12
$\phi = .75$, $\theta = .25$: algorithm 17.

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

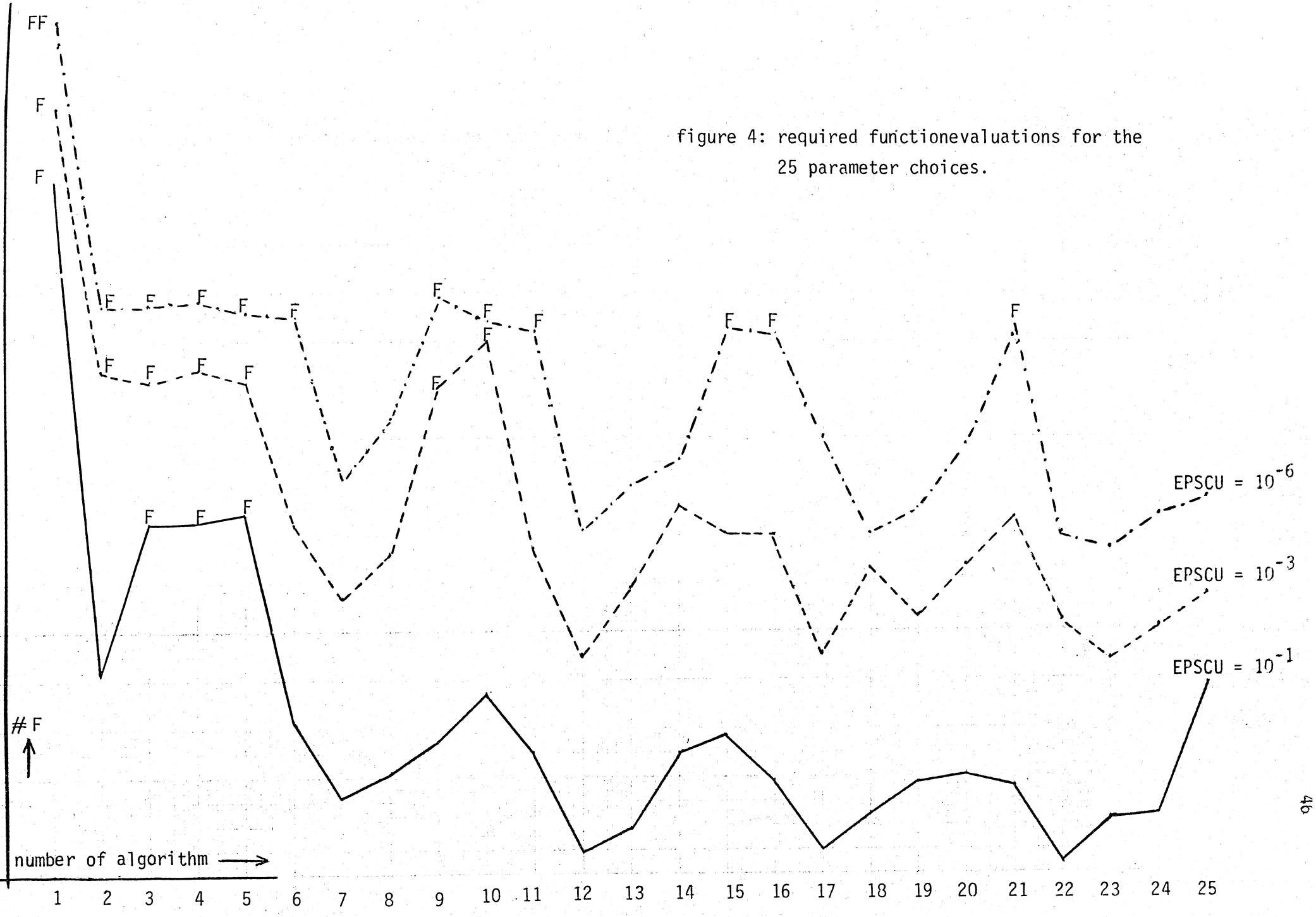
Table XVIII: # F for EPSCU = 10^{-1}
EPSCU = 10^{-1}, 10^{-3} (cumulative)
EPSCU = 10^{-1}, 10^{-3}, 10^{-6} (cumulative)

EPSCU = 10^{-1}		EPSCU = 10^{-1}, 10^{-3}		EPSCU = 10^{-1}, 10^{-3}, 10^{-6}	
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 4. In this figure the experimentally found points are connected by straight lines to simplify 'reading' of the figure. This is not intended to suggest an analytically proved continuity of number of function evaluations in terms of parameter combinations!

From figure 4 we see that $\phi = 0$ is unsatisfactorily while for any given nontrivial value of ϕ the algorithms get worse for higher values of θ. Obviously the parameter θ, which is the weighting factor of the correction term vv' in (1.2) is of more importance than the parameter ϕ which defines the scaling of the objective function!
The conclusion on the accuracy of the linesearch is further investigated in experiment II where the Goldstein and Price test is implemented which may lead to avoid linesearching completely.

figure 4: required functionevaluations for the 25 parameter choices.



Experiment II

We consider implementation of the algorithms A, B and C and the four Oren-Spedicato switches.

First the sensitivity for the choice of the parameter σ of the Goldstein and Price test is investigated. We tested σ = 0.01. 0.10, 0.25 and 0.49. For σ = 0.01 the Newton steplength '1' will often be accepted and no linesearch is performed. Increasing σ causes more linesearches, for σ = 0.49 almost all iterations use a cubic linesearch with EPSVU = 10^{-1}. In our experiment σ = 0.10 generally yielded the best results. The final results are given in tables X - XIII. Clearly switch IV is dominated by the competitive algorithms.

Experiment III

Implementation of DFP, BFGS and the two Shanno-Phua algorithms were run for σ = 0.10 and EPSCU = 10^{-1}. Clearly DFP prefers (requires) an exact linesearch, which confirm known results. Tables XIV - XVII present the relevant figures

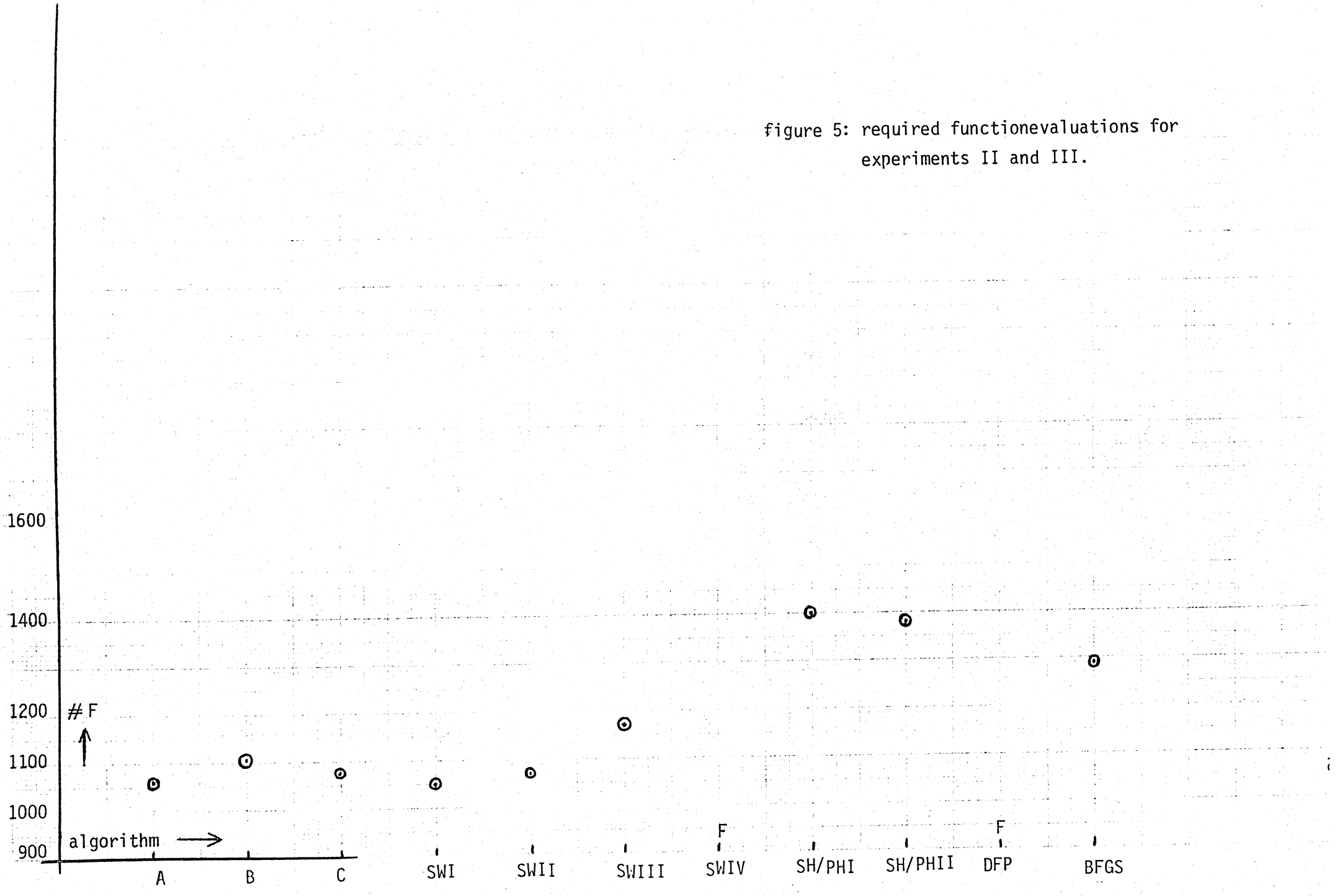
Figure 5 illustrates the results of the experiments on the resulting testset of 11 algorithms:

A, B, C, switches I, II, III, IV, Shanno/Phua I, II, DFP and BFGS.

Our general conclusion is that switches I, II and II are competitive with the (optimized) algorithms A, B and C.

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 realized 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 and C indicate to replace θ_k = 1 for all k in BFGS by θ_k = .25 for all k.

figure 5: required functionevaluations for experiments II and III.



The influence of the dimension of the testproblem and remarks on homogeneous testproblems.

Table XIX presents those figures from tables X and XIV which concern the 10- and 30-dimensional Rosenbrock and Quartic testfunctions. Clearly initial scaling of BFGS should not be recommended for homogeneous testproblems as the Quartics. This confirms Shanno and Phua, [1978]. Further table XIX suggests to apply Shanno/Phua I or switch 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.

Table XIX: # F for the 10- and 30-dimensional testproblems.
Goldstein/Price test with $\sigma = 0.10$. Accuracy linesearch 10^{-1}

algorithm / testfunction	A	B	C	SW I	SW II	SWIII	SWIV	SH/PH I	SH/PH II	DFP	BFGS
Ros(n=10)	128	133	124	130	117	166	F	116	113	262	112
(n=30)	237	264	256	225	268	312	F	244	231	F	381
Quartic(n=10)	47	47	47	48	47	48	F	172	172	479	128
(n=30)	52	53	53	53	53	53	F	415	414	716	253
Σ	464	497	480	456	485	579	4F	947	930	2457 F	874

Influence of the condition of the testproblem.

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 condition 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 testfunctions are summarized in table XX.

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 XX: # F for ill-conditioned testproblems

algorithm / testfunction	A	B	C	SW I	SW II	SW III	SW IV	SH/PH I	SH/PH II	DFP	BFGS
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 remarks

Recently developed self scaling algorithms for unconstrained minimization were described and compared in experiments.
All algorithms, except DFP, showed a good performance with an inexact linesearch (generally an iteration requires about 2 function evaluations). For reasons of robustness and simplicity in use (initially scaled) BFGS algorithms seem to be preferable in most practical situations. This conclusion is even more general: numerical comparisons by Grandinetti, [1978] and Shanno and Phua, [1978b] show that this classical quasi-Newton algorithm is competitive with sophisticated versions of quasi-Newton algorithms as those based on factorisations or projections of search directions.

Acknowledgement

The authors are indebted to Prof.Dr.Ir. van den Meerendonk, Drs. Bus and Dr.Ir. de Jong for discussions in various stages of this project.

Appendix A: Linesearches and the Goldstein and Price Test.

Linesearches

The efficiency of linesearches or one-dimensional optimization procedures, is frequently reported in literature. See e.g. Lootsma, [1972], Dixon, [1972], Sargent and Sebastian, [1972], Himmelblau, [1972], van der Hoek and Baardman, [1977], Biggs, [1971], Tamir, [1976], Fox R.L.c.s., [1975] and Walsh, [1975].

The available methods can be derived, roughly speaking, into two classes: Methods based on the splitting of an interval into two segments (Golden Section and Fibonacci search) and methods based on (polynomial) approximation followed by interpolation to a point x near the minimum x*

In the basic algorithm we implemented Davidon's cubic interpolation method as linesearch, but we also investigated the possibility to avoid relatively expensive linesearches by simply taking the Newton step '1'.

Usually linesearches are based on the assumption that the function f(x) is unimodal in the search direction d. The linesearch searches for the unique minimum of $h(\alpha) = f(x_k + \alpha d_k)$ along the ray $x_k + \alpha d_k$. The used cubic interpolations are applied after bracketing the minimum. They require function values f(x) and the directional derivative of f(x) in the point $x_k + \alpha d_k$, which is given by

$$G(\alpha) = \nabla' f(x_k + \alpha d_k).d_k.$$

We proceed now with the description of the linesearch, (the subscript k will be suppressed), which consists of the following steps:

step 1. Calculate $f_0 = f(x + 0.d)$ and $G(0) = \nabla' f(x + 0.d).d < 0$ go to step 2.

step 2. Find a value α_1 of α satisfying at least one of the following conditions:
i. $G(\alpha_1) = \nabla' f(x + \alpha_1.d).d > 0$
ii. $f_{\alpha_1} = f(x + \alpha_1.d) > f_0$,
go to step 3.

step 3. Approximate $f(x + \alpha.d)$ by the cubic polynomial $y(\alpha)$ given by the 4 conditions which express that $y(\alpha)$ and $f(x + \alpha.d)$ possess the same function value and the same directional derivate in both endpoints of the interpolation interval. Go to step 4.

step 4. Find the minimum α_m of $y(\alpha)$ on the considered interval and go to step 5.

step 5. Define, using α_m, which of the smaller intervals ($[0,\alpha_m]$ or $[\alpha_m, \alpha_1]$); brackets the minimum and apply convergence conditions. Stop if convergence is obtained, otherwise go to step 3.

Figures 6 and 7 illustrate the conditions of step 2. Note that figure 7 concerns the case that the function to be minimized is not unimodal, which happens frequently in practical problems.

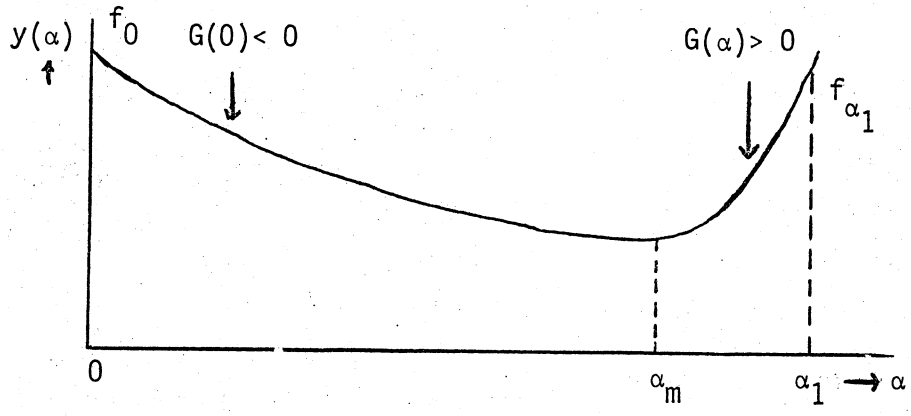


figure 6

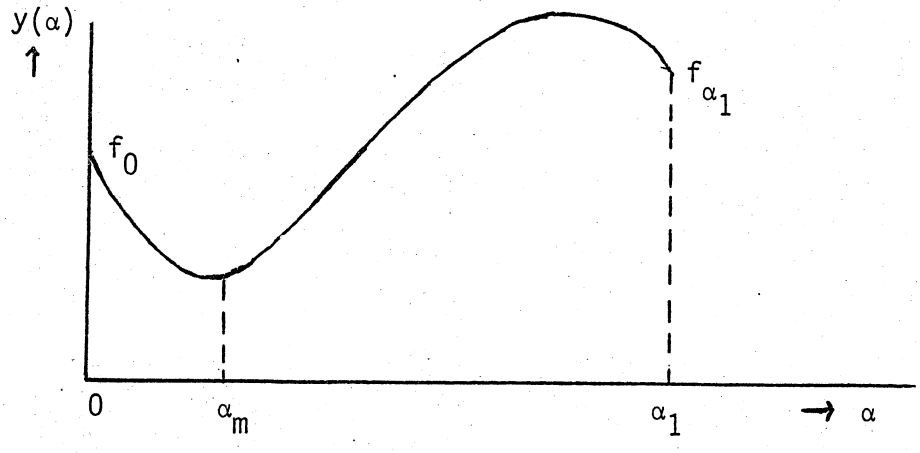


figure 7

Davidon proposed to define α_1 in step 2 by:

A.1.1 $$\alpha_1 = \min \{\alpha_0, -\frac{2(f_0 - f_e)}{G(0)}\},$$

where α_0 is some representative magnitude of the problem (usually $\alpha_0 = 2$) and f_e is a preliminary estimate of the minimum (we used in all cases $f_e = 0$)
It is easy to verify that $\frac{-2(f_0 - f_e)}{G(0)}$ is the value of α which minimizes a quadratic objective function along the ray $x + \alpha d$, substituting for f_e the exact minimum value of $f(x)$. We replaced Davidon's estimate by:

A.1.2 $$\alpha_1 = \min \{2, \operatorname{abs}(\frac{-2f_0}{G(0)})\},$$

where the absolute value of the quotient $\{\frac{-2f_0}{G(0)}\}$ is made necessary by the choice $f_e = 0$ and permitted because of $G(0) < 0$ (d is a descent direction). The choice of α_1 is followed by testing the conditions of step 2. If α_1 violates both conditions, α_1 is replaced by $2\alpha_1$, etc. thus performing a 'bracketing' of the minimum.
The polynomial $y(\alpha)$, given by step 3, has a unique minimum in α_m:

A.1.3 $$\alpha_m = \alpha_1 \left\{ 1 - \frac{G(\alpha_1) + w - z}{G(\alpha_1) - g(0) + 2w} \right\}$$

with

A.1.4 $$z = 3/\alpha_1 . (f_0 - f_{\alpha_1}) + G(0) + G(\alpha_1) \text{ and}$$

A.1.5 $$w = (z^2 - G(0).G(\alpha_1))^{\frac{1}{2}}.$$

See e.g. Walsh [1975] for a further treatment of (A.1.3) - (A.1.5).
Finally, step 5 needs a rule to choose the next interpolation interval: $G(\alpha_m) < 0$ gives rise to consider $[\alpha_m, \alpha_1]$ as next interpolation interval, otherwise $[0, \alpha_m]$ is taken.

The Goldstein and Price test (Goldstein and Price, [1967])

The effect of this test is that no linesearch is performed if the steplength "1" yields a point x_{k+1} with $f(x_{k+1})$ sufficiently close to the linear (Taylor) approximation of $f(x)$ in a neighbourhood of x_k.
The implementation of this test yields the following strategy for determination of the steplength:

step 1: set $\alpha = 1$ calculate x_{k+1} and $f(x_{k+1})$ and
go to step 2

step 2: test if the Goldstein and Price test is satisfied:

$$\sigma < \frac{f(x_{k+1}) - f(x_k)}{g'_k\, p_k} < 1 - \sigma \quad \text{for given } 0 < \sigma < \tfrac{1}{2}$$

If satisfied, accept x_{k+1} and return.
Otherwise go to step 3.

step 3: find α and $\hat{}_{k+1}$ from the available linesearch and return.

Apparently $\sigma = \frac{1}{2}$ permanently causes a usual linesearch, while σ close to zero almost completely avoids this linesearch. The best choice of σ depends on $f(x)$ and the algorithm in which this test is used. We obtained good results for $\sigma = 0.10$, which means that seldomly a linesearch is performed.

Appendix B: Testproblems.

1, 2, 3, 4 A family of Rosenbrock-functions:

$$f(x) = c(x_2 - x_1^2)^2 + (1 - x_1)^2$$

for $c = 1, 10^2, 10^4, 10^6$.

Initial point: $x_0' = (-1.2, 1)$

Solution: $x^{*'} = (1,1)$ with $f(x^*) = 0$.

5, 6 Multidimensional banana functions:

for N = 10, 30 the function

$$f(x) = \sum_{k=1}^{n-1} \left\{ 100(x_{k+1} - x_k^2)^2 + (1 - x_k)^2 \right\}$$

initial point $x_0' = (-1.2, 1, -1.2, 1, \ldots, -1.2, 1)$

solution $x^{*'} = (1,1,\ldots,1)$ with $f(x^*) = 0$.

7, 8, 9 Oren's Quartic function for N = 2, 10, 30

$f(x) = (x'Ax)^2$ with $A = \begin{pmatrix} 1 & 2 & & & \phi \\ & 3 & 4 & & \\ & & \ddots & \ddots & \\ \phi & & & \ddots & N \end{pmatrix}$

initial point $x_0' = (1,1,\ldots,1)$

solution $x^{*'} = (0,0,\ldots,0)$ with $f(x^*) = 0$

10, 11, 12 Hilbert problems for N = 2, 4, 6

$$f(x) = x'A_n x \quad \text{with} \quad (A_n)_{ij} = \frac{1}{i+j-1} \quad (i,j = 1,\ldots,n)$$

(A_n is an (n x n) segment of the Hilbert matrix)

REFERENCES

Bard, Y., [1968], On a Numerical Instability of Davidon-like Methods, Maths. of Comp., 22, pp. 665-666.

Biggs, M.C., [1971], Minimization Algorithm making use of non-quadratic Properties of the objective Function, J. Inst. Maths. Applics., 8, pp. 315-327 and 12, pp. 337,338.

Broyden, C.G., [1967], Quasi-Newton methods and their Application to Function Minimization, Maths. of Comp., 21, pp. 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, pp. 76-90, 222-231.

Colville, A.R., [1968], A comparative Study on nonlinear Programming Codes, IBM Tech. Rept. no. 320-8949.

Davidon, W.C., [1959], Variable Metric Method for Minimization, A.E.C. Research and Development Report ANL-5990 (Rev.).

Davidon, W.C., [1975], Optimally conditioned Optimization Algorithms without Linesearches, Math. Progr. 9, pp. 1-30.

Dixon, L.C.W., [1972a], Variable Metric Algorithms: Necessary and sufficient Conditions for Identical Behaviour on Non-quadratic Functions, J.O.T.A., 10, pp. 34-40.

Dixon, L.C.W., [1972b], The Choice of Steplength, a crucial Factor in the Performance of Variable Metric Algorithms, in F.A. Lootsma (ed), Numerical Methods for nonlinear Optimization, Ac. Press, London, 1972.

Fletcher, R., [1970], A new Approach to Variable Metric Algorithms, Computer J., 13, pp. 317-322.

Fletcher, R., and M.J.D. Powell, [1963], A rapidly convergent descent Method for Minimization. Computer J., 6, pp. 163-168.

Fox, R.L., L.S. Lasdon, A. Tamir and M.W. Ratner, [1975], An efficient 1-dimensional Searchprocedure, Man. Sci., 22, no. 1.

Goldstein, A.A., and J.F. Price, [1967], An effective Algorithm for Minimization, Numerische Mathematik, 10, pp. 184-189.

Grandinetti, L., [1978], Factorization versus non-Factorization in quasi-Newton Algorithms for Differentiable Optimization, Report N. 5, Dipartimento di Sistemi, Universita della Calabria, Italy.

Greenstadt, J., [1970], Variations on Variable Metric Methods, Maths. of Comp., 24, pp. 1-22.

Himmelblau, D.M., [1972], Applied nonlinear Programming, Mc Graw Hill, New York.

Hoek, G. van der and M. Baardman, [1977], Onderzoek naar de keuze van de staplengte op de werking van de algoritme van Fletcher en Powell, Dutch,Working Paper, Econometric Institute, Erasmus University Rotterdam.

Huang, H.Y., [1970], Unified Approach to Quadratically convergent Algorithms for Function Minimization, J.O.T.A., 5, pp. 405-423.

Jacobson, D.H., and W. Oksman, [1970], An Algorithm that minimizes homogeneous Functions of n Variables in n + 2 Iterations and rapidly minimizes general Functions, Technical Report 618, Division of Engineering and Applied Physics, Harvard University, Cambridge, MA.

Jong, J.L. de, [1976], Numerieke algoritmen voor niet-lineaire Optimaliseringsproblemen, Dutch, Technische Hogeschool Eindhoven.

Lenard, M.L., [1976], Convergence Conditions for restarted Conjugate Gradient Methods with Inaccurate Linesearches, Math. Progr., 10, pp. 32-52.

Loewner, C., [1957], Advanced Matrix Theory, Lecture Notes, Stanford University.

Lootsma, F.A., [1972], Penalty-Function Performance of several Unconstrained Minimization Techniques, Philips Res. Repts., 27, pp. 358-385.

Luenberger, D.G., [1973], Introduction to Linear and Nonlinear Programming, Addison-Wesley Publishing Company, Massachusetts.

Mc Cormick, G.P., and J.D. Pearson, [1969], Variable Metric Methods and Unconstrained Optimization, in 'Optimization', R. Fletcher (ed), Ac. Press, London, pp. 307-325.

Oren, S.S, [1973], Self Scaling Variable Metric Algorithms without Linesearch for Unconstrained Minimization, Maths. of Comp., 27, no. 124, pp. 873-885.

Oren, S.S., [1974a], Self Scaling Variable Metric Algorithms, Part II: Implementation and Experiments, Man. Sci., 20, no. 5.

Oren, S.S., [1974b], On the Selection of Parameters in Self Scaling Variable Metric Algorithms, Math. Prog. 7, no. 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, no. 5.

Oren, S.S. and E. Spedicato, [1976], Optimal Conditioning of Self Scaling Variable Metric Algorithms, Math. Prog. 10, pp. 70-90.

Osborne, M.R., [1972], Topics in Optimization, Stan-CS-72, Stanford University.

Powell, M.J.D., [1977], Restart Procedures for the Conjugate Gradient Method, Math. Progr. 12, pp. 241-254.

Rinnooy Kan, A.H.G., [1979], Paper to be presented at the X International Symposium on Mathematical Programming, Montreal.

Rosenbrock, H.H., [1961], An automatic Method for finding the greatest or least Value of a Function, Computer J. 3, pp. 175-184.

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, Ac. Press, London, 1972.

Shanno, D.F. and Kang-Hoh Phua, [1978a], Matrix Conditioning and Nonlinear Optimization, Math. Prog. 14. pp. 149-160.

Shanno, D.F. and Kang-Hoh Phua, [1978b], Numerical Comparison of Several Variable Metric Algorithms, J.O.T.A. 25, no. 4, Aug. 1978.

Tamir, A., [1976], Linesearch Techniques based on Interpolating Polynomials using Functionvalues only, Man. Sci. 22, no. 5.

Walsh, G.R., [1975], Methods of Optimization, J. Wiley & Sons, London

LIST OF REPORTS 1979

7900 "List of Reprints, nos 220-230; Abstracts of Reports Second Half 1978.

7901/S "Motorists and Accidents (An Empirical Study)", by B.S. van der Laan.

7902/S "Estimation of the Minimum of a Function Using Order Statistics", by L. de Haan.

7903/S "An Abel and Tauber Theorem Related to Stochastic Compactness", by L. de Haan.

7904/E "Effects of Relative Price Changes on Input-Output Ratios - An Empirical Study for the Netherlands", by P.M.C. de Boer.

7905/O "Preemptive Scheduling of Uniform Machines Subject to Release Dates", by J. Labetoulle, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan.

7906/S Π-Regular Variation", by J.L. Geluk.

7907/O "Complexity Results for Scheduling Chains on a Single Machine", by J.K. Lenstra and A.H.G. Rinnooy Kan.

7908/E "Input-Output and Macroeconomic Forecasting Through the Generalized Inverse", by K.P. Vishwakarma.

7909/E "An Application of the Generalized Inverse in Input-Output and Macroeconomic Analysis", by K.P. Vishwakarma.

7910/O "A Numerical Comparison of Self Scaling Variable Metric Algorithms", by G. van der Hoek and M.W. Dijkshoorn.