MANAGEMENT SCIENCE
Vol. 20, No. 5, January, 1974
Printed in U.S.A.

SELF-SCALING VARIABLE METRIC (SSVM) ALGORITHMS

Part II: Implementation and Experiments*†

SHMUEL S. OREN

Xerox Corporation, Palo Alto, California and Stanford University

This part of the paper introduces some possible implementations of Self-Scaling Variable Metric algorithms based on the theory presented in Part I. These implementations are analyzed theoretically and discussed qualitatively. A special class of SSVM algorithms is introduced, which has the additional property of being invariant under scaling of the objective function or of the variables. Experimental results are provided for a particular case of this class. This case has been tested in comparison to the DFP algorithm on a variety of functions with up to 50 variables. The results indicate that the new method has substantial advantage for functions with a large number of variables.

1. Introduction

Variable metric methods such as the DFP algorithm tend to be very sensitive to factors that weaken their approximation to the conjugate gradient method (e.g., nonquadratic terms in the objective function, line-search inaccuracy, roundoff errors). It has been known for a long time that this sensitivity is substantially affected by the scaling of the objective function. In Part I of this paper it was demonstrated that this sensitivity actually depends on the "single step convergence rate" which is a bound on the stepwise decrease in function value. This bound is a function of the condition number of the matrix $R = D^{1/2}\nabla^2 f(x)D^{1/2}$ where D is the current inverse Hessian approximation. Poor scaling of the objective function (through multiplication by a scalar) may cause this condition number to increase during the process of minimization. Consequently, this will cause deterioration in the single-step convergence rate, and will increase the algorithm's sensitivity to the destructive factors mentioned above.

The self-scaling variable metric algorithms mentioned in Part I of this paper form a two-parameter family of variable metric algorithms where the parameters are restricted so as to guarantee monotonic decrease in the condition number of R when the algorithms are applied to a quadratic function. This will consequently insure good scaling.

Algorithm 1 below describes the general family of self-scaling variable metric algorithms, based on the theory presented in Part I of this paper.

Algorithm 1 (*SSVM*).

Begin with any starting point x_0.

Step 1. Set $i = 0$ and choose D_0 to be a positive definite matrix.

Step 2. Set $d_i = -D_i g_i$.

Step 3. Minimize $f(x_i + \alpha d_i)$ with respect to $\alpha \geqq 0$ to obtain

α_i, $p_i = \alpha_i d_i$, $x_{i+1} = x_i + p_i$, $g_{i+1} = \nabla f(x_{i+1})$ and $q_i = g_{i+1} - g_i$.

Step 4. Choose $\theta_i \in [0, 1]$ and γ_i such that $\gamma_i \in [1/\lambda_n{}^i, 1/\lambda_1{}^i]$, $\gamma_i > 0$.

($\lambda_n{}^i$, $\lambda_1{}^i$ are the largest and the smallest eigenvalues of

$$R_i = H_i^{1/2} D_i H_i^{1/2} \quad \text{where} \quad H_i = \int_0^1 \nabla^2 f(x_i + \rho p_i)\, d\rho).$$

* Received June 1972; revised January 1973.

† This research was supported by the National Science Foundation, Grant No. NSF-GK-29237.

Step 5. Set

$$v_i = (q_i' D_i q_i)^{1/2}(p_i/p_i' q_i - D_i q_i/q_i' D_i q_i), \tag{1}$$

$$D_{i+1} = [D_i - D_i q_i q_i' D_i/q_i' D_i q_i + \theta_i v_i v_i']\gamma_i + p_i p_i'/p_i' q_i . \tag{2}$$

Step 6. Add one to i and return to Step 2.

In summary of Part I of the paper we list below the main properties of the above algorithm.

1. D_i is positive definite for any i provided that D_0 is positive definite and $p_i' q_i > 0$.

2. In the quadratic case the algorithm is a conjugate gradient algorithm, provided $D_0 = I$, and converges in n steps.

3. In the quadratic case the condition number of R_i is monotonically decreasing.

In this part of the paper we are concerned with the practical implementation of SSVM algorithms. The practical value of Algorithm 1 depends on the possibility of implementing Step 4 without evaluating the eigenvalues of R_i. Fortunately, it is possible to generate scaling factors γ_i using only readily available information. We develop a family of formulae for generating γ_i and investigate its properties. We also discuss the implications of varying γ_i in Algorithm 1. Experimental results are provided for a particular SSVM algorithm that has been tested in comparison with the DFP method on several functions.

2. A Convex Class of Scaling Factors

In this section we introduce a convex class of formulae for computing γ_i that satisfy the requirement of Step 4 in Algorithm 1. These formulae use only information that is already generated in the algorithm for other purposes.

DEFINITION 1. Let D be a nonsingular symmetric matrix and p, q two nonzero vectors in E^n. Then the scalar $\gamma^\varphi(D, p, q)$ is defined by the formula

$$\gamma^\varphi(D, p, q) = (1 - \varphi)p'q/q'Dq + \varphi p'D^{-1}p/p'q. \tag{3}$$

We are going to show that the scalars $\gamma^\varphi(D_i, p_i, q_i)$ with $\varphi \in [0, 1]$ satisfy the conditions required in Step 4 of Algorithm 1. It immediately follows from (3) that $\gamma^\varphi(D, p, q)$ is strictly positive if D is positive definite, $p'q > 0$ and $\varphi \in [0, 1]$. Thus we have only to show that $1/\gamma^\varphi(D_i, p_i, q_i)$ is in the interval spanned by the eigenvalues of R_i. This is proved in the following theorem.

THEOREM 1. *Let p, q be nonzero vectors in E^n such that $p'q > 0$, D a positive definite symmetric matrix and H, R positive definite matrices such that*

$$q = Hp, \tag{4}$$

and

$$R = H^{1/2}DH^{1/2}. \tag{5}$$

Let $\gamma^\varphi(D, p, q)$ be as in Definition 1. Then for any $\varphi \in [0, 1]$ there holds

$$1/\lambda_n \leqq \gamma^\varphi(D, p, q) \leqq 1/\lambda_1, \tag{6}$$

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

PROOF. Since for any $\varphi \in [0, 1]$, $\gamma^\varphi(D, p, q)$ is a convex combination of its extreme values, $\gamma^0(D, p, q)$ and $\gamma^1(D, p, q)$, it is sufficient to prove the theorem for these two extreme values.

We define

$$z = H^{1/2}p \tag{7}$$

and

$$r = H^{-1/2}q. \tag{8}$$

For $\varphi = 0$, substituting (4), (5) and (7) into (3) yields

$$\gamma^0(D, p, q) = z'z/z'Rz. \tag{9}$$

Since $\lambda_1 z'z \leqq z'Rz \leqq \lambda_n z'z$, we have

$$1/\lambda_n \leqq \gamma^0(D, p, q) \leqq 1/\lambda_1 . \tag{10}$$

For $\varphi = 1$ substituting (4), (5) and (8) into (3) yields

$$\gamma^1(D, p, q) = r'R^{-1}r/r'r. \tag{11}$$

Since $r'r/\lambda_n \leqq r'R^{-1}r \leqq r'r/\lambda_1$, we have

$$1/\lambda_n \leqq \gamma^1(D, p, q) \leqq 1/\lambda_1 . \tag{12}$$

The result follows from (10) and (12).

For the general case where p is an arbitrary vector, the computation of $\gamma^1(D, p, q)$ involves inversion of D. Thus it may seem impractical to use $\gamma^\varphi(D_i, p_i, q_i)$ for $\varphi \neq 0$ as a scaling factor in Algorithm 1. Fortunately, for the p_i used in Algorithm 1, $\gamma^1(D_i, p_i, q_i)$ can be generated directly without inverting D_i. This is shown in the following proposition.

PROPOSITION 1. *Let* $p_i = -\alpha D_i \nabla f(x_i)$ *and* $q_i = \nabla f(x_i + p_i) - \nabla f(x_i)$ *for some* $x_i \in E^n$, $f \in C^2$, *a positive scalar* α *and a positive definite matrix* D_i. *Let* $g_i = \nabla f(x_i)$ *and* $\gamma^\varphi(D, p, q)$ *be as in Definition* 1. *Then there holds*

$$\gamma^1(D_i, p_i, q_i) = g_i'p_i/g_i'D_iq_i = -\alpha g_i'p_i/q_i'p_i . \tag{13}$$

If in addition $\alpha = \alpha_i$ *where* α_i *minimizes* $f(x_i - \alpha D_i g_i)$ *over* $\alpha \geqq 0$ (*assuming the minimum exists*) *then*

$$\gamma^1(D_i, p_i, q_i) = \alpha_i . \tag{14}$$

PROOF. Substituting $p_i = -\alpha_i D_i g_i$ and $\varphi = 1$ in (3) yields

$$\gamma^1(D_i, p_i, q_i) = \frac{p_i' D_i^{-1} p_i}{p_i' q_i} = -\frac{\alpha g_i' p_i}{p_i' q_i} = \frac{g_i' p_i}{g_i' D_i q_i} . \tag{15}$$

If $\alpha = \alpha_i$ then $p_i'\nabla f(x_i + p_i) = 0$ and hence $p_i'q_i = -p_i'g_i$. Substituting the above into (15) yields (14).

Proposition 1 provides two alternative ways for obtaining $\gamma^1(D_i, p_i, q_i)$. From an economical point of view it seems better to use the value of α_i (as suggested by (14)), which is readily available. However, the error in this value may be quite large even for accurate line-searches (for example, if the function is very flat). For that reason we prefer to use (13) which does not depend on the line-search accuracy. Thus for the case $p_i = -\alpha D_i g_i$ we can write

$$\gamma^\varphi(D_i, p_i, q_i) = (1 - \varphi)p_i'q_i/q_i'D_iq_i + \varphi g_i'p_i/g_i'D_iq_i . \tag{16}$$

Using (16) as a scaling-factor generator in Algorithm 1 has the additional advantage of making the algorithm invariant under scaling.

The expression $\gamma^{\varphi}(D, p, q)$ has two additional properties the first of which is a "duality" relationship that follows immediately from (3). This duality can be described in the form

$$\gamma^1(D, p, q) = [\gamma^0(D^{-1}, q, p)]^{-1}. \tag{17}$$

The second property is set forth in the following proposition.

PROPOSITION 2. *Let p, q, H, D, R and $\gamma^{\varphi}(D, p, q)$ be as in Theorem* 1. *Then there holds*

$$1 \geqq \gamma^0(D, p, q)/\gamma^1(D, p, q) \geqq 4\kappa(R)/(\kappa(R) + 1)^2 \tag{18}$$

where $\kappa(R)$ is the condition number of R.

PROOF. By (3),

$$\frac{\gamma^0(D, p, q)}{\gamma^1(D, p, q)} = \frac{(p'q)^2}{(p'D^{-1}p)(q'Dq)} = \frac{(z'z)^2}{(z'R^{-1}z)(z'Rz)} \tag{19}$$

where $z = H^{1/2}p$. Applying Schwarz inequality to (19) yields the upper bound in (18), while the Kantorovich inequality [4] yields the lower bound.

3. A Class of SSVM Algorithms

The scaling-factors generator developed in the last section can be incorporated in Algorithm 1 to form the following class of SSVM algorithms.

Algorithm 2.

Begin with any starting point x_0.

Step 1. Set $i = 0$ and choose D_0 to be a positive definite matrix.

Step 2. Set $d_i = -D_i g_i$.

Step 3. Minimize $f(x_i + \alpha d_i)$ with respect to $\alpha \geqq 0$ to obtain α_i, $p_i = \alpha_i d_i$, $x_{i+1} = x_i + p_i$, $g_{i+1} = \nabla f(x_{i+1})$ and $q_i = g_{i+1} - g_i$

Step 4. Choose $\varphi_i \in [0, 1]$ and $\theta_i \in [0, 1]$.

Step 5. Set

$$\gamma_i = \frac{p_i' q_i}{q_i' D_i q_i}(1 - \varphi_i) + \frac{g_i' p_i}{g_i' D_i q_i}\varphi_i, \tag{20}$$

$$v_i = (q_i' D_i q_i)^{1/2}\left(\frac{p_i}{p_i' q_i} - \frac{D_i q_i}{q_i' D_i q_i}\right), \tag{21}$$

$$D_{i+1} = \left[D_i - \frac{D_i q_i q_i' D_i}{q_i' D_i q_i} + \theta_i v_i v_i'\right]\gamma_i + \frac{p_i p_i'}{p_i' q_i}. \tag{22}$$

Step 6. Add one to i and return to Step 2.

An additional favorable property of Algorithm 2 is its invariance, under scaling of the objective function or of the variables. This is proved in the following proposition. By possessing this property, the algorithm is immune to numerical instabilities of the type indicated by Bard [1].

PROPOSITION 3. *Let D_i, x_i, φ_i and θ_i be as in Algorithm* 2 *and let $\{D_i\}$, $\{x_i\}$ and $\{\hat{D}_i\}$, $\{\hat{x}_i\}$ be the corresponding sequences generated by Algorithm* 2 *when it is applied to $f(x)$ and $af(b\hat{x})$ respectively. [$f \in C^2$ and a, b are positive scalars.] If $\hat{D}_0 = cD_0$ [c positive scalar], $b\hat{x}_0 = x_0$ and the sequences $\{\theta_i\}$, $\{\varphi_i\}$ are identical for both cases, then $\hat{D}_i = D_i/ab^2$ and $\hat{x}_i = x_i/b$ for $i > 0$.*

PROOF. For any $\hat{x}_i \in E^n$ $\hat{g}_i = \nabla_{\hat{x}}[af(b\hat{x}_i] = ab\nabla_x f(x_i) = abg_i$ where $x_i = b\hat{x}_i$. The first direction of search is clearly the same for both cases. Thus $\hat{x}_1 = x_1/b$, since both minimize $f(\cdot)$ along the same line. Consequently, $\hat{p}_0 = p_0/b$, $\hat{g}_0 = abg_0$ and $\hat{q}_0 = abq_0$. Substituting the above $\hat{p}_0$, $\hat{g}_0$, $\hat{q}_0$ and $\hat{D}_0$ in (20), (21) and (22) yields $\hat{D}_1 = D_1/ab^2$. The result follows by induction.

4. Optimal Scaling Factors

In view of the large number of possibilities for choosing the scaling factors in Algorithm 1 and Algorithm 2, one may ask to what extent different choices of γ_i may affect the convergence of an SSVM algorithm. We focus our attention on this question from the aspect of improving the single-step convergence rate (see Part I). More explicitly, we are concerned with how the decrease in $\kappa(R_i)$ at a certain iteration will be affected by updating D_i using self-scaling updating formulae with some fixed $\theta_i \in [0, 1]$ and different $\gamma_i \in [1/\lambda_n^{\,i}, 1/\lambda_n^{\,i}]$. For this purpose we reconsider in the next theorem the eigenvalue structure of the matrices R_i.

For reading convenience, we redefine the necessary notation introduced in Part I of this paper. It was shown there that if D_i is updated by equations (1) and (2) and if the objective function is quadratic with Hessian H, then $R_{i+1} = \mathfrak{D}^\theta(R_i, \gamma, z, z)$ where

$$\mathfrak{D}^\theta(R, \gamma, z, z) = [R - Rzz'R/z'Rz + \theta uu']\gamma + zz'/z'z, \tag{23}$$

$$u = (z'Rz)^{1/2}(z/z'z - Rz/z'Rz), \tag{24}$$

and

$$z = H^{1/2}p.$$

THEOREM 2. *Let $\hat{R}^\theta(\gamma) = \mathfrak{D}^\theta(R, \gamma, z, z)$ be defined by (23) and (24) for some fixed positive definite matrix R and a nonzero vector z. Then for any fixed θ the eigenvalues of $\hat{R}^\theta(\gamma)$ are $\{\gamma\eta_2^{\,\theta}, \gamma\eta_3^{\,\theta}, \cdots, \gamma\eta_n^{\,\theta}, 1\}$ where $\eta_2^{\,\theta} \leqq \eta_3^{\,\theta} \leqq \cdots \leqq \eta_n^{\,\theta}$ are the nonzero eigenvalues of P^θ defined as*

$$P^\theta = R - Rzz'R/z'Rz + \theta uu', \tag{25}$$

with u defined by (24). Furthermore, if the eigenvalues of R are denoted by $\lambda_1 \leqq \lambda_2 \leqq \cdots \leqq \lambda_n$ then for any $\theta \in [0, 1]$ there holds

$$\lambda_1 \leqq \eta_2^{\,\theta} \leqq \lambda_2 \leqq \eta_3^{\,\theta} \leqq \cdots \leqq \eta_n^{\,\theta} \leqq \lambda_n. \tag{26}$$

PROOF. From (23) and (25),

$$\hat{R}^\theta(\gamma) = \gamma P^\theta + zz'/z'z. \tag{27}$$

From (24) and (25) it follows that $P^\theta z = 0$ and consequently, by (27), $\hat{R}^\theta(\gamma)z = z$. Since z is an eigenvalue of P^θ, adding $zz'/z'z$ to γP^θ moves its eigenvalue corresponding to z from zero to unity, while the other eigenvalues remain unchanged. Thus the eigenvalues of $\hat{R}^\theta(\gamma)$ denoted by $\mu_1^{\,\theta}(\gamma) \leqq \mu_2^{\,\theta}(\gamma) \leqq \cdots \leqq \mu_n^{\,\theta}(\gamma)$ are all contained in the set $\{1, \gamma\eta_2^{\,\theta}, \gamma\eta_3^{\,\theta}, \cdots, \gamma\eta_n^{\,\theta}\}$.

Let $\bar{\gamma}$ be a constant such that $\bar{\gamma} > 1/\lambda_1$. It was proved in Part I of this paper (Theorem 4) that for this choice of $\bar{\gamma}$, $\mu_1^{\,\theta}(\bar{\gamma}) = 1$ and $\bar{\gamma}\lambda_{k-1} \leqq \mu_k^{\,\theta}(\bar{\gamma}) \leqq \bar{\gamma}\lambda_k$ for $k = 2, 3, \cdots, n$. It follows that $\bar{\gamma}\eta_k^{\,\theta} = \mu_k^{\,\theta}(\bar{\gamma})$ for $k = 2, 3, \cdots, n$ and therefore

$$\bar{\gamma}\lambda_{k-1} \leqq \bar{\gamma}\eta_k^{\,\theta} \leqq \bar{\gamma}\lambda_k \quad \text{for } k = 2, 3, \cdots, n. \tag{28}$$

Dividing (28) by $\bar{\gamma}$ yields (26).

COROLLARY 1. *Let $\hat{R}^\theta(\gamma)$ and $\eta_k^{\,\theta}$ be as defined in Theorem 2 with some fixed positive θ.*

Then

(a) *For any positive* γ *the largest eigenvalue of* $\hat{R}^{\theta}(\gamma)$ *is* $\max[1, \gamma\eta_n^{\theta}]$ *and the smallest is* $\min[1, \gamma\eta_2^{\theta}]$.

(b) *For any positive* γ,

$$\kappa(\hat{R}^{\theta}(\gamma)) \leqq \eta_n^{\theta}/\eta_2^{\theta}. \tag{29}$$

Equality in (29) *is achieved if, and only if*

$$\eta_2^{\theta} \leqq \frac{1}{\gamma} \leqq \eta_n^{\theta}. \tag{30}$$

PROOF. (a) Follows directly from Theorem 2.

(b) From part (a),

$$\kappa(\hat{R}^{\theta}(\gamma)) = \left\{\frac{\max[1, \gamma\eta_n^{\theta}]}{\min[1, \gamma\eta_2^{\theta}]}\right\} \geqq \frac{\eta_n^{\theta}}{\eta_2^{\theta}}. \tag{31}$$

If (30) holds then, by (31), $\kappa(\hat{R}^{\theta}(\gamma)) = \eta_n^{\theta}/\eta_2^{\theta}$. If (30) does not hold, there are two possible cases: If $1/\gamma > \eta_n^{\theta}$ then, by (31), $\kappa(\hat{R}^{\theta}(\gamma)) = 1/\gamma\eta_2^{\theta} > \eta_n^{\theta}/\eta_2^{\theta}$. If $1/\gamma < \eta_2^{\theta}$ then again, by (31), $\kappa(\hat{R}^{\theta}(\gamma)) = \gamma\eta_n^{\theta} > \eta_n^{\theta}/\eta_2^{\theta}$.

DEFINITION 2 (OPTIMAL SCALING FACTOR). Let $\hat{R}^{\theta}(\gamma)$ be as in Theorem 2 for a fixed positive θ. γ_i is an optimal scaling factor if

$$\kappa(\hat{R}^{\theta}(\gamma_i)) = \min_{\gamma>0} \kappa(\hat{R}^{\theta}(\gamma)).$$

In view of the above definition it follows that condition (30) is a necessary and sufficient condition for γ to be an optimal scaling factor. This condition is given in terms of the eigenvalues of P^{θ}. Using the results of Theorem 2 we obtain a sufficient condition for optimality of γ in terms of the eigenvalues of R.

COROLLARY 2. *Let* $\hat{R}^{\theta}(\gamma)$ *be as in Theorem* 2 *and the eigenvalues of* R *be* $\lambda_1 \leqq \lambda_2 \leqq \cdots \leqq \lambda_n$. *If* $\theta \in [0, 1]$ *then any* $\gamma \in [1/\lambda_2, 1/\lambda_{n-1}]$ *is an optimal scaling factor.*

PROOF. By Theorem 2, $\eta_2^{\theta} > \lambda_2$ and $\lambda_{n-1} > \eta_n^{\theta}$ for all $\theta \in [0, 1]$. Therefore if $\lambda_2 \leqq 1/\gamma \leqq \lambda_{n-1}$, then $\eta_2^{\theta} \leqq 1/\gamma \leqq \eta_n^{\theta}$ and hence, by Corollary 1 and Definition 2 γ is optimal.

COROLLARY 3.

Let $\hat{R}^{\theta}(\gamma)$, R, λ_k, η_k^{θ} *be as in Theorem* 2, *and* κ^{θ} *be defined as*

$$\kappa^{\theta} = \min_{\gamma>0} \kappa(\hat{R}^{\theta}(\gamma)). \tag{32}$$

For any γ *such that* $\lambda_1 \leqq 1/\gamma \leqq \lambda_n$ *and all* $\theta \in [0, 1]$,

$$\kappa(\hat{R}(\gamma))/\kappa^{\theta} \leqq \max[\lambda_n/\lambda_{n-1}, \lambda_2/\lambda_1]. \tag{33}$$

PROOF. By Corollary 1,

$$\kappa(\hat{R}^{\theta}(\gamma)) = \max[1, \gamma\eta_n^{\theta}]/\min[1, \gamma\eta_2^{\theta}] \quad \text{and} \quad \kappa^{\theta} = \eta_n^{\theta}/\eta_2^{\theta}.$$

Since $\lambda_1 \leqq 1/\gamma \leqq \lambda_n$, then, for all $\theta \in [0, 1]$,

$$\kappa(\hat{R}^{\theta}(\gamma)) \leqq \max[\eta_n^{\theta}/\lambda_1, \lambda_n/\eta_2^{\theta}, \eta_n^{\theta}/\eta_2^{\theta}]. \tag{34}$$

Thus

$$\kappa(\hat{R}^{\theta}(\gamma))/\kappa^{\theta} \leqq \max[\eta_2^{\theta}/\lambda_1, \lambda_n/\eta_n^{\theta}, 1]. \tag{35}$$

By Theorem 2, $\eta_2^{\theta} \leqq \lambda_2$ and $\eta_n^{\theta} \geqq \lambda_{n-1}$ for all $\theta \in [0, 1]$. Hence (33) follows from (35).

We see from Corollary 3 that, when the separation between the eigenvalues of the R_i is relatively small, only minor improvement can be achieved by using a better scaling factor. In such cases any choice of $\gamma \in [1/\lambda_n, 1/\lambda_1]$ is satisfactory. However there may be cases where the separation between the eigenvalues of R_i is large such as for penalty or barrier functions. If, for example, there is only one constraint, λ_n/λ_{n-1} may be quite large. Unfortunately, no general formula for γ, such as (16), has yet been found that will guarantee $\lambda_2 \leqq 1/\gamma \leqq \lambda_{n-1}$.

5. Alternative Strategies and Related Properties

In certain cases difficulties associated with poor scaling can be eliminated merely by initially scaling the problem. This corresponds to choosing a sequence $\{\gamma_i\}$ such that $\gamma_i = 1$ for $i > 0$. For the quadratic case such a strategy can be justified theoretically by the results given in Part I (Theorem 4). It was shown that in the quadratic case R_i has a unit eigenvalue for all $i > 0$; therefore, the choice $\gamma_i = 1$ for all $i > 0$ satisfies $\lambda_1^i \leqq 1/\gamma_i \leqq \lambda_n^i$. Assuming that γ_0 is chosen such that $\lambda_1^0 \leqq 1/\gamma_0 \leqq \lambda_n^0$, the above strategy in a quadratic case satisfies the conditions of Step 4 in Algorithm 1. An algorithm resulting from this strategy may seem attractive since it maintains the property $D_n = H^{-1}$ for the quadratic case (see Corollary 4 in Part I). Such a strategy which will be referred to as "initial scaling" can be implemented by modifying Algorithm 2 so that γ_0 is generated by (20) while for $i > 1$ $\gamma_i = 1$.

The initial scaling strategy is clearly superior to the traditional way of choosing $\gamma_i = 1$ for all i, since it retains all the former properties while benefiting from the freedom of choosing γ_0. This was also confirmed in our numerical experiments.

Unfortunately, in a general purpose algorithm scaling only on the first iteration may be insufficient. When considering nonquadratic functions one may expect that the changes in the Hessian, as the algorithm proceeds, may cause the eigenvalues of the matrix $H_i^{1/2} D_i H_i^{1/2}$ to drift away from unity unless the problem is rescaled. Thus in these cases the initial scaling algorithm may eventually lose the properties associated with self-scaling. Cases in which initial scaling is insufficient are not always predictable. Thus using the above strategy may not always overcome the deficiencies of classical variable metric algorithms. An example which illustrates such difficulties is discussed in §6.

It is clear that the changes in the Hessian could be compensated by using occasionally $\gamma_i \neq 1$. However this may raise the question of how often to scale. Algorithm 2 resolves this problem by readjusting γ_i at each step. Unfortunately, by allowing the scaling factor γ_i to vary we lose the property $D_n = H^{-1}$ in the quadratic case ("Property 1"). On the other hand, we ensure monotonic improvement in the single-step convergence rate which is implied by monotonic decrease in $\kappa(R_i)$ ("Property 2"). In the remainder of this section we compare the "initial scaling" strategy with that used in Algorithm 2 by discussing the implications of trading Property 1 for Property 2. This discussion also applies to comparison between SSVM and classical variable metric algorithms and is the key to understanding the results of the experiments presented in §6.

The trading of Property 1 for Property 2 is meaningless in the quadratic case where $D_0 = I$ and exact line search is performed. This is because in this case Algorithm 1 will become the conjugate gradient algorithm independently of Step 4, generating a unique sequence of points that converge to the minimum in n steps. Property 1 in that case would provide a Newton step on the $(n+1)$th iteration which yields the minimum. However it is redundant, since the minimum is reached in n steps by virtue of

the conjugacy. Property 2, on the other hand, ensures that the decrease in the function at each step will be at least as good as for steepest descent. This superiority to steepest descent, however, is again secured by the fact that the algorithm is conjugate gradient (see Luenberger [5]).

In order to understand the meaning of trading Property 1 for 2 and to predict the consequences in practical cases, we have to consider in a more heuristic manner a perturbation from the ideal that will destroy the conjugacy and the n-step convergence. Such a perturbation may be associated with inaccurate line search, nonquadratic terms in the objective function, or roundoff errors. It may be argued that in such a case Property 1 will still tend to provide a good step every $(n + 1)$th iteration if the progress at intermediate steps is poor. If for example the algorithm jams for n steps in a small neighborhood, the D_{n+1} will be a good local approximation of the inverse Hessian, yielding an approximate Newton step. Property 2, on the other hand, will improve the progress on each step.

On the basis of the above argument, we may anticipate an advantage for Property 1 in the case of a difficult function with few variables. In such a case having a good step every $(n + 1)$th iteration will compensate for the poor intermediate steps. As the

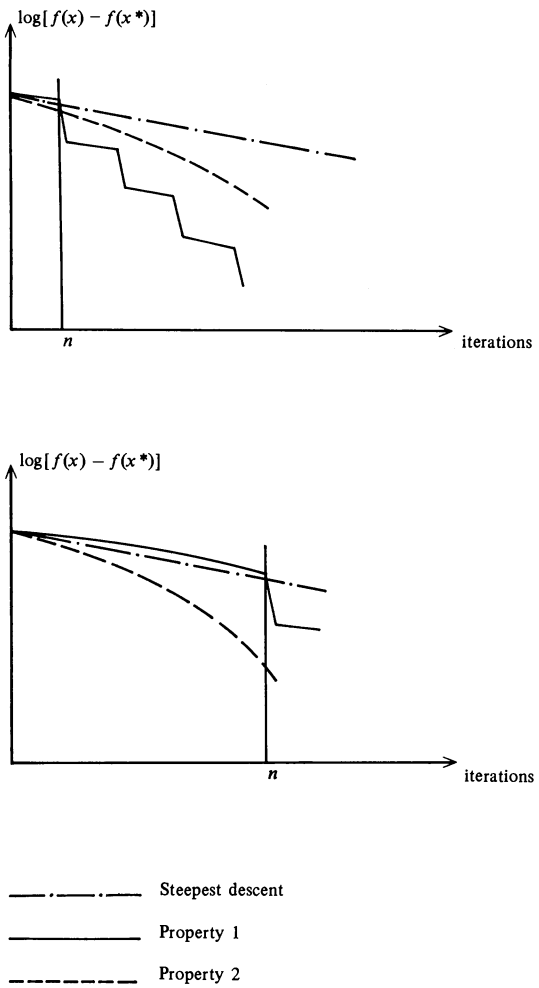

FIGURE 1. Convergence Pattern of Algorithms Having Property 1 versus Those Having Property 2.

number of variables becomes larger the importance of Property 1 decreases while that of Property 2 increases. This can be explained by the infrequency of the good steps provided by Property 1 and the increase in the cumulative contribution of the good intermediate steps provided by Property 2. Furthermore when the number of variables n is large, the number of iterations required to obtain a reasonable approximation to the solution may be expected to be a low multiple of n or even less than n. The above argument is illustrated in Figure 1 in which the steepest descent method that converges linearly is used as a reference.

6. Numerical Experiments and Discussion

The numerical experiments were aimed at testing the effect of self-scaling and at verifying the theory presented so far. Since testing the effect of varying the parameters φ_i and θ_i in Algorithm 2 was out of the scope of this paper, we used only a special case of that algorithm where $\theta_i = \varphi_i = 0$ for all i. This algorithm was compared with the DFP algorithm, which corresponds to $\theta_i = 0$ and $\gamma_i = 1$ for all i. Both the SSVM and the DFP algorithms were run with and without restarting after every n steps.

These algorithms were already compared for two test problems in Part I of this paper. In the first example, it was demonstrated that the SSVM algorithm is far less sensitive to nonquadratic terms in the objective function than the DFP method. Furthermore, in contract to the DFP algorithm, the SSVM is always better than steepest descent. In a second example for which detailed results are given in [5], it was shown that the SSVM algorithm is far less sensitive to the line search inaccuracy than the DFP algorithm. It was also demonstrated that the DFP method may become inferior to steepest descent in the presence of a small error in line search, while the SSVM algorithm is always at least as good as steepest descent. In the quadratic case with exact line search both algorithms performed identically, as is implied by the theory.

In the rest of the experiments described below we use several nonquadratic test problems with the variables ranging in number from two up to fifty. The results of these experiments are summarized in Table 1. More detailed results are given in [6]. In a few cases we also tested the initial scaling strategy mentioned in §5 where $\gamma_0 = p_0' q_0 / q_0' D_0 q_0$ and $\gamma_i = 1$ for $i > 0$.

Test functions. (a) Helical valley [3]—3 variables.

$$f(x_1, x_2, x_3) = 100\{[x_3 - 10\theta(x_1, x_2)]^2 + [r(x_1, x_2) - 1]^2\} + x_3^2.$$

$$\theta(x_1, x_2) = \frac{1}{2\pi} \arctan (x_2/x_1) \qquad \text{for } x_1 > 0,$$

$$= \frac{1}{2} + \frac{1}{2\pi} \arctan (x_2/x_1) \quad \text{for } x_1 < 0.$$

$$r(x_1, x_2) = (x_1^2 + x_2^2)^{1/2}.$$

Minimum point: $x^* = (1, 0, 0)$.

Starting point: $x_0 = (-1, 0, 0)$.

(b) Wood's function [2]—4 variables.

$$f(x_1, x_2, x_3, x_4) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1).$$

Minimum point: $x^* = (1, 1, 1, 1)$.

Starting point: $x_0 = (-3, -1, -3, -1)$.

TABLE 1

Summary of Numerical Experiments

Function	N	DFP		DFP Restarted		SSVM		SSVM Restarted		Initial Scaling Strategy	
		IT	*NF*	*IT*	*NF*	*IT*	*NF*	*IT*	*NF*	*IT*	*NF*
a	3	23	52	31	69	28	71	45	119	—	—
b	4	31	93	26	74	60	183	>70(1)	291	—	—
c	2	23	74	26	88	33	137	50	218	—	—
c	6	67	175	77(2)	206	68	197	80(3)	286	—	—
c	10	>88(2)	281	131	316	85	249	133	427	90	261
c	16	>200(1)	632	200(1)	477	139	395	171	507	131	379
c	30	—	—	—	—	259	749	—	—	—	—
c	50	—	—	—	—	437	1319	—	—	—	—
d	6	29(3)	88	15(3)	57	9	36	7(3)	31	28	104
d	10	42(3)	136	33(3)	93	13	48	11(3)	43	67(3)	210
d	20	78(3)	236	45(3)	131	17	58	17	58	—	—
d	30	89(3)	270	74(3)	211	21	63	21	63	—	—
d	50	134(3)	381	141(3)	365	29	88	29	88	—	—

IT: Number of Iterations to Convergence.
NF: Cumulative Number of Function and Gradient Evaluations.
(1) Program stopped due to excessive number of iterations.
(2) Program stopped due to error exit.
(3) Required accuracy not achieved.

(c) Multidimensional banana function—N variables.

$$f(x) = \sum_{k=1}^{N} [100\,(x_{k+1} - x_k^2)^2 + (1 - x_k)^2].$$

Minimum point: $x^* = (1, 1, \cdots, 1)$.

Starting point: $x_{0k} = -1.2 \quad \text{for } k = 1, 3, 5 \cdots,$

$= 1 \qquad \text{for } k = 2, 4, 6 \cdots.$

(For $N = 2$ this is Rosenbrock's function [7].)

(d) Quartic—N Variables.

$$f(x) = (x'Qx)^2.$$

$Q \equiv$ diagonal matrix with diagonal elements,

$q_{kk} = k \quad \text{for } k = 1, 2, \cdots, N.$

Minimum Point: $x^* = (0, 0, \cdots, 0)$.

Starting Point: $x_0 = (1, 1, 1, \cdots, 1)$.

The "multidimensional banana" function (defined above) is used as an extreme case which is not exactly typical in practical applications. Even for a modest number of variables it usually severely challenges standard algorithms. In our experiments the DFP algorithm was tried on this function with up to 16 variables, in which case it failed to converge. For 16 variables the DFP algorithm was quite far from the solution even after 200 iterations when it was stopped. For that reason and because of limited computer time only the SSVM algorithm was tried with 30 and 50 variables. (The 50-variables case took 4.5 minutes CPU time.)

The results given in Table 1 show that for a small number of variables the DFP algorithm has an advantage. However, as the number of variables increases, the SSVM algorithm becomes better. In the cases we ran, the SSVM algorithm was better than DFP for functions with more than six variables. This behavior can be explained by the fact that the DFP algorithm has Property 1 mentioned in §5 while the SSVM algorithm has Property 2. The argument given in §5 regarding the tradeoffs between these two properties is supported by the results.

The limited experiments with the initial-scaling strategy were successful with the banana function. However the quartic function is a classical example where this strategy performs poorly. In the latter case the Hessian matrix at the minimum is identically zero and therefore as the algorithm progresses the eigenvalues of the Hessian as well as the eigenvalues of R drift constantly toward zero. Thus unless γ_i is adjusted constantly, as occurs in the SSVM algorithm, $1/\gamma_i$ does not remain in the span of the eigenvalues of R_i and the self-scaling property is lost. Nevertheless, this strategy was better than the DFP method except for one case in which the eigenvalues of the Hessian at the starting point were large so that the function had to be scaled down. This initial down-scaling proved disadvantageous in later stages when unscaling became necessary.

The quartic functions with large numbers of variables also illustrate cases where convergence can be achieved with the SSVM algorithms in fewer iterations than the number of variables. This again illustrates the advantages of Property 2 compared to Property 1 for a large number of variables. In fact, the detailed numerical results in [6] show that the SSVM algorithm always was better than the DFP algorithm on the first n steps even in cases where the DFP method won. Thus if we stopped the algorithms before n steps, the SSVM would always yield a better approximation to the minimum than the DFP algorithm.

For some reason the number of gradient and function evaluations per line-search was slightly higher for the SSVM algorithm in all cases. This can only be explained by the fact that in the program used the heuristic parameters in the line-search were adjusted for the DFP algorithm which gave this method an advantage.

Computational Remarks

The numerical experiments were done on an IBM 360/65 computer with single precision. The program used was a modiffied version of the algorithm FELPOMIN (see [9]) programmed in ALGOL W. The same program was used to test all the methods and only the updating formula was altered accordingly. The line-search was the same in all cases, and was based on cubic interpolation. The stopping rule for the line-search was: Stop if the new point is obtained between the last two points, and the new function is lower than at the last two points. The new point was also required to satisfy the condition $p_i'q_i > 0$ to guarantee positive definiteness of the D_i matrices. The stopping rule for the complete algorithm was based on the stepsize and the norm of the gradient. However, successful convergence always resulted in function values within 10^{-9} or less (depending on the function) from the minimum.

A feature added to the algorithms on a heuristic basis was to reset the inverse Hessian approximation to a diagonal matrix with random elements in the interval [0.1, 2] if the denominators in the updating formula became too small. However this was not done more than once in succession.

The algorithms had several error exits for the following cases:

(1) The direction of search was not a direction of descent.

(2) The number of iterations exceeded some predetermined number.
(3) The step size became too small.
(4) The denominators in the updating formula became too small twice in succession.

7. Conclusions

The focus in this part of the paper is on possible implementations of SSVM algorithms based on the theory presented in Part I. Algorithm 2 contains a special family of such implementations, which have the following properties.

(1) The inverse Hessian approximations are positive definite.

(2) For an n-dimensional quadratic function the directions of search are conjugate (if $D_0 = I$, conjugate gradient), and the algorithms converge in n steps.

(3) The single-step convergence rate decreases monotonically when the algorithm is applied to a quadratic function.

(4) The algorithms are invariant under the scaling of the objective function or uniform scaling of the variables.

The last two properties, which are new, are responsible for the superior performance of the SSVM algorithm in high-dimensional problems, demonstrated in the numerical experiments, and promise low sensitivity to line-search inaccuracy and roundoff errors.

Further investigation is required to analyze the effect of varying the parameters φ_i and θ_i in Algorithm 2, and to obtain criteria for choosing these parameters. A possibility that should be considered is to choose φ_i so that γ_i is as close as possible to unity. γ_i will then automatically become unity if the problem is already well-scaled. The selection $\varphi_i = \theta_i = 0$ used in this paper for demonstrating the virtues of self scaling, is by no means an optimal choice of these parameters. Further experiments (to be reported elsewhere) actually indicate that different values of φ_i and θ_i may lend to much better performance of Algorithm 2. Additional improvement may also be obtained by relaxing the line-search (see Oren [7]).

References

1. Bard, Y., "On a Numerical Instability of Davidon-like Methods, *Maths. of Comp.*, Vol. 22, pp. 665–666, 1968.
2. Colville, A. R., *A Comparative Study on Nonlinear Optimization Techniques*, Oliver and Boyd Ltd. (Edinburgh), 1969.
3. Fletcher, R., and Powell, M. J. D., "A Rapidly Convergent Descent Method for Minimization," *Comp. Jnl.*, Vol. 6, pp. 163–168, 1963.
4. Kantorovitch, L. V., and Akilov, G. P., *Functional Analysis in Normed Spaces*, New York, Macmillan, 1964.
5. Luenberger, D. G., *Introduction to Linear and Nonlinear Programming, Addison-Wesley*, 1973.
6. Oren, S. S., *Self-Scaling Variable Metric Algorithms for Unconstrained Minimization*, Ph.D. Thesis, Department of Engineering-Economic Systems, Stanford University, 1972.
7. Oren, S. S., "Self-Scaling Variable Metric Algorithm Without Line-Search for Unconstrained Minimization", *Maths. of Comp.* 27 (1973) No. 124.
8. Rosenbrock, H. H., "An Automatic Method for Finding the Greatest or Least Value of a Function," *Comp. Jnl.*, Vol. 3, p. 175, 1960.
9. Wells, M., "Algorithm 251, Function Minimization," [E4] *Collected Algorithms from CACM*, 251-P1-0, 1964 (see also: Certification 251-P2-0, 1966, and Remarks 251-P3-R1, 1960 and 1970.).