Multi-Secant Equations, Approximate Invariant Subspaces and Multigrid Optimization

by S. Gratton[1] and Ph. L. Toint[2]

Report 07/11 20th November 2007

[1] C.N.E.S. and C.E.R.F.A.C.S.,
Toulouse, France
Email: gratton@cerfacs.fr

[2] Department of Mathematics,
University of Namur - FUNDP,
61, rue de Bruxelles, B-5000 Namur, Belgium.
Email: philippe.toint@fundp.ac.be

Multi-Secant Equations, Approximate Invariant Subspaces and Multigrid Optimization

Serge Gratton* and Philippe L. Toint†

20th November 2007

Abstract

New approximate secant equations are shown to result from the knowledge of (problem dependent) invariant subspace information, which in turn suggests improvements in quasi-Newton methods for unconstrained minimization. It is also shown that this type of information may often be extracted from the multigrid structure of discretized infinite dimensional problems. A new limited-memory BFGS using approximate secant equations is then derived and its encouraging behaviour illustrated on a small collection of multilevel optimization examples. The smoothing properties of this algorithm are considered next, and automatic generation of approximate eigenvalue information demonstrated. The use of this information for improving algorithmic performance is finally investigated on the same multilevel examples.

Keywords: large-scale optimization, multigrid techniques, quasi-Newton methods, limited memory algorithms, discretized problems.

1 Introduction

The history of quasi-Newton methods for optimization is rich and long. Starting with the Davidon-Fletcher-Powell (DFP) method (Davidon, 1959, Fletcher and Powell, 1963), examplified by the famous Broyden-Fletcher-Goldfarb-Shanno (BFGS) update (Broyden, 1970, Fletcher, 1970, Goldfarb, 1970 and Shanno, 1970), and excellently explained in the classical book by Dennis and Schnabel (1983), they have played an important role in the solution of practical problems. In the context of unconstrained minimization, i.e. the solution of problems of the form

$$\min_{x\in\mathbb{R}^n} f(x) \tag{1.1}$$

for a smooth objective function f from $\mathbb{R}^n$ into $\mathbb{R}$, they attempt to construct, around a given point $x \in \mathbb{R}^n$, a second-order model of this function of the form

$$m(x+s) = f(x) + \langle g(x), s\rangle + \tfrac{1}{2}\langle s, Bs\rangle$$

where $g(x) \stackrel{\rm def}{=} \nabla_x f(x)$, and where B is an (often positive-definite) approximation of the Hessian matrix $\nabla_{xx} f(x)$, capturing information about the curvature of the objective function around x. Quasi-Newton methods then proceed to exploit a sequence of models of this type in an iterative manner. In this process, the curvature information at iterate x_{k+1} is obtained by updating the approximate Hessian matrix B_k to obtain the new approximation B_{k+1} such that the *secant equation*

$$B_{k+1}s_k = y_k, \tag{1.2}$$

*CNES and CERFACS, Toulouse.

†Department of Mathematics, University of Namur - FUNDP, Namur, Belgium.

holds, where

$$s_k \stackrel{\rm def}{=} x_{k+1} - x_k \;\; \text{and} \;\; y_k \stackrel{\rm def}{=} g(x_{k+1}) - g(x_k), \tag{1.3}$$

The pair (s_k, y_k) is then said to be the quasi-Newton pair associated with equation (1.2). If positive-definiteness of the matrix B_k is also maintained throughout the iterations (as can be enforced for instance with the BFGS or DFP updates), the search direction at iteration k is then computed from

$$d_{k+1} = -B_{k+1}^{-1} g(x_{k+1}) \tag{1.4}$$

and a linesearch is performed along this direction (see Dennis and Schnabel, 1983, for details). In order to avoid the cost of solving the linear system in (1.4), the matrix $H_{k+1} \stackrel{\rm def}{=} B_{k+1}^{-1}$ is typically recurred instead of B_{k+1}, using the *inverse secant equation*

$$H_{k+1} y_k = s_k \tag{1.5}$$

as an alternative to (1.2), and

$$d_{k+1} = -H_{k+1} g(x_{k+1}) \tag{1.6}$$

instead of (1.4). Note that (1.5) uses the same pair as (1.2) but in the reverse order. For the BFGS update, which is particular interest to us in this paper, the relevant updating formula is then given by

$$H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k} \right) H_k \left(I - \frac{y_k s_k^T}{y_k^T s_k} \right) + \frac{s_k s_k^T}{y_k^T s_k} \tag{1.7}$$

It readily follows from this formula that H_{k+1} remains positive-definite if H_k is positive-definite and if

$$y_k^T s_k > 0, \tag{1.8}$$

a condition one can always enforce in the linesearch procedure if the objective function is bounded below (again see Dennis and Schnabel, 1983).

We are especially interested in the application of quasi-Newton to large-scale problems, in which case it is often impractical to store the (dense) matrices B_{k+1} or H_{k+1} explicitly. In such a context, a "limited-memory" version of the quasi-Newton method has been pioneered by several authors, in which the matrix H_{k+1} is assembled at every iteration as a product of finitely many low-rank updates, each involving a pair (s_j, y_j) (see Liu and Nocedal, 1989 and Byrd, Lu and Nocedal, 1993, for the most famous algorithm of this type and further references).

Our purpose in the present paper is to show that additional knowledge about the eigenstructure of the local Hessian matrix $\nabla_{xx} f$ can be used to advantage in order to capture more information on the local curvature. We discuss in particular how this can be achieved when limited-memory BFGS updates are considered, and illustrate the practical motivation for this analysis in the case of large-scale multilevel/multigrid unconstrained optimization. Examples of this type are presented, and it is shown that our proposal may improve their numerical solution substantially.

The resulting multigrid optimization method is a linesearch algorithm, at variance with the trust-region based techniques discussed in Gratton, Sartenaer and Toint (2007*b*, 2006) or Gratton, Mouffe, Toint and Weber-Mendonça (2007*a*). It also differs from the proposals by Nash (2000), Lewis and Nash (2002) and Wen and Goldfarb (2007) in that none of these techniques makes explicit use of limited-memory quasi-Newton Hessian approximations.

Section 2 introduces the use of invariant subspaces in the derivation of secant information, presents the resulting quasi-Newton algorithm and discusses the specialization of this new algorithm to the multigrid case. Section 3 briefly describes our test problems, and reports our first numerical results. Section 4 discusses further consequences of the use of the new algorithm to multigrid optimization, introduces approximate eigenvalue equations and their use within the new algorithmic framework. Section 5 presents the associated numerical tests. Some conclusions and perspectives are finally outlined in Section 6.

2 Invariant Subspaces in Quasi-Newton Methods

2.1 Invariant Subspaces and Approximate Secant Equations

We start by considering the case where the objective function f is a convex quadratic, that is

$$f(x) = f + \langle g, x\rangle + \tfrac{1}{2}\langle x, Gx\rangle$$

where $f \in \mathbb{R}$, $g \in \mathbb{R}^n$ and G is a positive-definite symmetric matrix. In this case, it is easy to verify that, if we consider a step s_k at iteration k of a quasi-Newton algorithm, then

$$Hy_k = s_k \tag{2.1}$$

where $H = G^{-1}$. Assume now that we also know a decomposition of $\mathbb{R}^n$ in a collection of invariant subspaces $\{\mathcal{S}_i\}_{i=1}^p$ related to G, i.e. subspaces such that, for each i, $Gd \in \mathcal{S}_i$ whenever $d \in \mathcal{S}_i$. Since the eigenvectors of H are identical to those of G, the subspaces $\mathcal{S}_i$ are also invariant for H. Now consider S_i the orthogonal projectors onto $\mathcal{S}_i$. Since these projectors share a common system of eigenvectors with H, we know that they must commute, that is $HS_i = S_iH$. Using this very simple observation, we then obtain, for a step s_k at iteration k of a quasi-Newton algorithm, that

$$HS_iy_k = S_iHy_k = S_is_k, \tag{2.2}$$

thereby yielding a new secant equation with the pair (S_is_k, S_iy_k). Repeating the procedure for $i \in \{1, \ldots, p\}$, we therefore obtain p additional secant equations (2.4) (in addition to the original equation (2.1)) provided we know the projections S_i.

If we now consider general twice differentiable, possibly non-convex, objective functions, then (2.1) remains valid for

$$G = \int_0^1 \nabla_{xx} f(x_k + t(x_{k+1} - x_k))s_k \, dt$$

and the same reasonning still holds if the invariant subspaces are now associated with this latter matrix. Finally, if the subspaces $\mathcal{S}_i$ are only approximately invariant, or if the operators S_i are only approximately equal to projectors onto these subspaces, then our secant equations stop being exact but can be expected to hold approximately. We therefore refer to secant equations of the type (2.2) as *approximate*, as opposed the the *exact* equation (1.5). If we are interested in the size of the perturbation of G that would necessary for the approximate secant equations (2.2) to hold exactly, then we must investigate the norm of the symmetric perturbation E_i such that

$$(G + E_i)S_is_k = S_iy_k \quad (1 \le i \le p),$$

which in turn gives that, for some compatible matrix and vector norms,

$$\frac{\|E_i\|}{\|G\|} \le \frac{\|GS_is_k - S_iy_k\|}{\|S_is_k\| \, \|G\|} \tag{2.3}$$

for $k \ge 0$ and $0 \le i \le p$. If the relative perturbation on the left-hand side of this inequality is modest, then relation (2.2) (with $H = G^{-1}$) holds exactly for a symmetric Hessian approximation relatively close to G. It may thus be reasonable to use it for extracting curvature information, which may then be included in our current Hessian approximation.

2.2 Multi-Secant (Limited-Memory) Quasi-Newton Algorithms

Using the simple derivation exposed above, and assuming, crucially, that a collection of (possibly approximate) projectors S_i are known, we may then outline a multi-secant quasi-Newton algorithm as in Algorithm 2.1 on the following page.

Algorithm 2.1: Multi-Secant Quasi-Newton Algorithm (outline)

Step 0: Initialization. An initial point $x_0 \in \mathbb{R}^n$ and an initial (positive-definite) Hessian H_0 are given. The operators S_i $(i = 1, \ldots, p)$ are also given, as well a a small tolerance $\epsilon \geq 0$. Compute $f(x_0)$ and $g(x_0)$, and set $k = 0$.

Step 1: If $\|g(x_k)\| \leq \epsilon$, stop.

Step 2: Compute the search direction. Define $d_k = -H_k g(x_k)$.

Step 3: Linesearch. Perform a linesearch ensuring

$$f(x_{k+1}) \leq f(x_k) + \alpha \langle g(x_k), d_k \rangle \text{ and } \langle g(x_{k+1}), d_k \rangle \geq \beta \langle g(x_k), d_k \rangle$$

for some $\alpha \in (0, 1)$ and $\beta \in (\alpha, 1)$, yielding $f(x_{k+1})$, $g(x_{k+1})$, s_k and y_k satisfying (1.8).

Step 4: Update the Hessian approximation. Compute H_{k+1} such that (1.2) holds and

$$H_{k+1}(S_i y_k) = (S_i s_k) \quad i \in \{1, \ldots, p\}. \tag{2.4}$$

Step 5: Loop. Set $k = k + 1$ and return to Step 1.

In this outline, we have not specified the linesearch procedure in detail, but this well understood technique is described in detail in Section 6.3 of Dennis and Schnabel (1983) or in Moré and Thuente (1994), for instance. We haven't either indicated how we can impose (1.2) and (2.4) together. While updating simultaneaously for more than one secant equation is indeed possible (see Byrd, Nocedal and Schnabel, 1994), we focus here on the incorporation of the information in the Hessian in a sequential manner, by performing first p BFGS updates of the form (1.7) using the p pairs $(S_i s_k, S_i y_k)$, followed by a final update using the pair (s_k, y_k). Of course the order in which the p "subspace updates" are performed is then significant.

If the problem size is large, which is the case in the applications described below, it is often better to avoid the explicit updating of B_{k+1} or H_{k+1}, mostly because the low rank update (1.7) typically results in H_{k+1} being dense. To circumvent this problem, variants of the classical quasi-Newton algorithm have been proposed that replace (1.6) by a recursion in which H_{k+1} is implicitly reconstructed from some simple initial approximation (typically a multiple of the identity matrix: we use the matrix

$$\frac{\langle y_k, s_k \rangle}{\|y_k\|_2^2} I$$

in the experiments discussed below) and a modest number of the most recent pairs (s_j, y_j). No matrix is ever assembled in the process, which only requires the storage of a small number of vectors. Because only the most recent pairs are used, these methods are called "limited-memory" methods, compared to the usual (full-memory) quasi-Newton algorithms where H_{k+1} includes information derived from all past pairs. The best-known method of this type is the limited-memory BFGS method pioneered by Byrd, Lu, Nocedal and Zhu, 1995. The update (1.7) is (implicitly) used in this algorithm to compute the step. An efficient technique to perform the calculation of the step s_k is described on page 225 of Nocedal and Wright (1999), where m pairs (s_j, y_j) are used sequentially in the implicit update of a diagonal initial approximation. We refer below to this technique as the implicit-secant-updating algorithm.

This technique can readily be adapted to our context where each secant pair generates p additional approximate ones corresponding to its images in p approximate invariant

subspaces. Instead of considering only the last m pairs in the limited-memory updating procedure, we may now consider $m \times p$ pairs, or any selection we care to make amongst them. An extreme case is when we select only the $p+1$ secant pairs corresponding to (s_k, y_k) and its images (2.4) onto $\{\mathcal{S}_i\}_{i=1}^p$, giving a "memory-less" BFGS method.

2.3 Collinearity and Curvature Control

Nothing in the algorithm we have described so far prevents secant pairs to be linearly dependent. While not a major issue in the usual context where secant pairs are generated at different iterations, this might be of an issue in our case, where one expects some dependency amongst the pairs generated from different, but possibly nested, invariant subspaces. Fortunately, some control of the possible collinearity of the secant pairs is easy, and we have chosen to include a provision in our algorithm that considers the angle between approximate secant pairs generated at iteration k and the "exact" pair (s_k, y_k). More formally, we have decided to ignore the secant pair $(S_i s_k, S_i y_k)$ whenever

$$|\langle S_i s_k, s_k\rangle|_2 \leq \tau \|S_i s_k\|_2 \, \|s_k\|_2 \;\; \text{or} \;\; |\langle S_i s_k, S_j s_k\rangle| \leq \tau \|S_i s_k\|_2 \, \|S_j s_k\|_2$$

for some $\tau \in (0,1]$ (typically 0.999 when this feature is active) and some $j < i$. Note that this last condition depends on the order in which the secant pairs are considered, a choice which we discuss below.

Because we wish to preserve the positive-definite nature of the approximate Hessian H_{k+1}, we also ignore approximate secant pairs for which

$$\langle S_i y_k, S_i s_k\rangle < \mu \|S_i y_k\|_2 \, \|S_i s_k\|_2$$

for some $\mu \in (0,1)$.

2.4 Invariant Subspaces and Multigrid Optimization

Of course, the above discussion and the proposed multi-secant quasi-Newton methods may only be of practical interest when the operators S_i are known for a collection of (approximate) invariant subspaces. The purpose of this paragraph is to show that this desirable situation may occur in an important practical case: multigrid optimization.

Let us assume that the optimization variables of the problem under consideration represent the discretization of some continuous field defined on some spatial or temporal domain, a very common situation in engineering applications or physical modelling. For example, the variables may stand for coordinates of a design surface, atmospheric pressure over some part of the ocean, or position of a spacecraft along a controlled trajectory. The main characteristics of these problems is that it is possible to define discretization of the field of interest with varying degree of coarseness, from the very coarse to the very fine. For the sake of the argument, suppose that we consider $r+1$ such different field discretizations, which we number from 0 (coarsest) to r (finest). In this case, it is very often reasonable to assume that, for each $i \in \{1, \ldots, r\}$, there exist a full-rank linear operator R_i from $\mathbb{R}^{n_i}$ into $\mathbb{R}^{n_{i-1}}$ (the restriction from the fine grid i to the coarser grid $i-1$) and another full-rank operator P_i from $\mathbb{R}^{n_{i-1}}$ into $\mathbb{R}^{n_i}$ (the prolongation from the coarse grid $i-1$ to the fine grid i). Moreover, these grid-transfer operators are typically computationaly cheap to apply: the prolongation is for instance often chosen as the linear interpolation operator and the restriction as some multiple of its transpose, sometimes called the full-weighting operator.

When the problem to be solved for the (field) variables is a linear or nonlinear system of equations (instead of an optimization problem), multigrid techniques often yield the computationally most efficient algorithm, as their cost typically grows only linearly with the number of variables. The main characteristics of multigrid algorithms (we refer the reader to Briggs, Henson and McCormick, 2000 for an excellent introduction, or to Trottenberg, Oosterlee and Schüller, 2001, for a fuller treatment) are based on the observation

that different "frequencies" are present in the solution of the finest grid problem (or even of the infinite-dimensional one), and become only progressively visible in the hierarchy from coarse to fine grids. Low frequencies are visible from coarse grids and up, but higher ones can only be distinguished when the mesh-size of the grid becomes comparable to the inverse of the frequency in question. In multigrid strategies, specific algorithms, called *smoothers*, are known to very efficiently reduce the high frequency components of the error on a grid (that is, in most cases, the components whose "wavelength" is comparable to the grid's mesh-size), but have little effect on the low frequency error components. It is observed however that such components on a fine grid appear more oscillatory on a coarser grid. They may thus be viewed as high frequency components on some coarser grid and be in turn efficiently reduced by a smoother. Moreover, this is done at a lower cost since computations on coarse grids are typically much cheaper than on finer ones. The multigrid strategy consists therefore in alternating between solving the problem on coarse grids, essentially annihilating low frequency components of the error, and on fine grids, where high frequency components are reduced (at a higher cost) by a smoother. This last operation is often called *smoothing* and the associated method a smoother because the effect of reducing high frequency components without altering much the low frequency ones has a "smoothing effect" of the error's behaviour.

In other words, the multigrid strategy exploits the fact that the considered operator is (approximately) separable in the frequency domain, and that restrictions from fine to progressively coarser grids followed by prolongations to the fine grid isolate the corresponding nested invariant subspaces frequency-wise. A very well-understood example is that of the linear Poisson equation in unbounded domains, given in its discretized[(1)] variational form by

$$\min_u \tfrac{1}{2}\langle u, \Delta u\rangle - \langle f, u\rangle, \tag{2.5}$$

where $u = u(x)$ is the unknown temperature distribution at position x of the underlying spatial domain and Δ is the discretized Laplacian operator. It is easy (see Briggs et al., 2000, page 18) to verify that the eigenvalues of the unidimensional discretized operator in $n-1$ variables are (when one takes the Dirichlet boundary condition $u_0 = u_n = 0$ into account)

$$\lambda_i = 4\sin^2\left(\frac{i\pi}{2n}\right) \quad (i = 1, \ldots, n-1)$$

and that its eigenvectors are given componentwise by

$$z_{i,j} = \sin\left(\frac{ij\pi}{n}\right) \quad (i = 1, \ldots, n-1,\ j = 0, \ldots, n).$$

It is remarkable that the eigenvectors z_i $(i = 1, \ldots, \frac{1}{2}n)$ on the fine grid are exactly representable on a coarser grid with double mesh-size. If we choose the commonly used full weighting restriction operator (the transpose of the linear interpolator), it is possible to verify (see Briggs et al., 2000, pages 80–81) that

$$R_r z_i = \theta_i \sin\left(\frac{ij\pi}{n/2}\right), \quad \text{where } \theta_i = \begin{cases} \cos^2\left(\frac{i\pi}{2n}\right) & \text{for } i = 1, \ldots, \frac{1}{2}n, \\ -\sin^2\left(\frac{i\pi}{2n}\right) & \text{for } i = \frac{1}{2}n, \ldots, n. \end{cases}$$

It would be ideal if the prolongation $P_r R_r z_i$ would be exactly a linear combination of

$$\left\{\sin\left(\frac{ij\pi}{n}\right)\right\}_{i=1}^{n/2}, \tag{2.6}$$

the eigenvectors associated with the smooth modes on the fine grid, since then the image of $P_r R_r$ would be identical to the invariant subspace spanned by the vectors (2.6). Unfortunately, this often not the case, due to a (often modest) contamination of the prolongated vector by oscillatory modes. Thus the image of $P_r R_r$ is only approximately equal to $\mathcal{S}_{r-1}$, but this operator is typically very cheap to apply.

[(1)] Using a simple finite-difference scheme.

2.5 A Multi-Secant Multigrid Limited-Memory BFGS Algorithm

We may now combine all the ingredients of our discussion so far into a single minimization algorithm: it suffices to use the multi-secant limited-memory BFGS method described above with the definition of our multigrid approximate operators

$$S_i u_r \stackrel{\rm def}{=} P_r \ldots P_{i+1} R_{i+1} \ldots R_r u_r \quad \text{for} \quad 0 \leq i \leq r-1. \tag{2.7}$$

Algorithm 2.2: Multi-Secant Multigrid Limited-Memory BFGS

Initialization. An initial point $x_0 \in \mathbb{R}^{n_r}$ and an initial (positive-definite) Hessian H_0 are given. The restriction and prolongation operators R_i and P_i are given for $i = 1, \ldots, r$, and the operators S_i are given by (2.7) for $i = 0, \ldots, r-1$. Choose a memory size $m \geq 1$, as well a a small tolerance $\epsilon \geq 0$. Compute $f(x_0)$ and $g(x_0)$, define the initial set of secant pairs $\mathcal{P}_0 = \emptyset$ and set $k = 0$.

Step 1: If $\|g(x_k)\| \leq \epsilon$, stop.

Step 2: Compute the search direction. Apply the implicit-secant-updating algorithm to compute $s_k = -H_k g(x_k)$ using the secant pairs in $\mathcal{P}_k$.

Step 3: Linesearch. Perform a linesearch ensuring

$$f(x_{k+1}) \leq f(x_k) + \alpha \langle g(x_k), s_k \rangle \ \text{ and } \ \langle g(x_{k+1}), s_k \rangle \geq \beta \langle g(x_k), s_k \rangle$$

for some $\alpha \in (0,1)$ and $\beta \in (\alpha, 1)$, yielding $f(x_{k+1})$, $g(x_{k+1})$ and y_k satisfying (1.8).

Step 4: Generate secant pairs. Apply the operators to compute

$$s_{k,i} = S_i s_k \ \text{ and } \ y_{k,i} = S_i y_k$$

for $i = 0, \ldots, r-1$ and set

$$\mathcal{P}_k^+ = \mathcal{P}_k \cup \{(s_{k,0}, y_{k,0}), \ldots, (s_{k,r-1}, y_{k,r-1}), (s_k, y_k)\}.$$

Step 5: Select the next set of secant pairs. Select m pairs in $\mathcal{P}_k^+$ to form $\mathcal{P}_{k+1}$.

Step 6: Loop. Set $k = k+1$ and return to Step 1.

We obviously expect multi-secant updates to work at their best potential when the eigensystem of the objective function's Hessian $\nabla_{xx} f(x)$ is well-aligned with the grid, in the sense that

$$S_i \nabla_{xx} f(x) \approx \nabla_{xx} f(x) S_i.$$

Note that many algorithmic variants are possible in selecting $\mathcal{P}_{k+1}$ (in Step 5). One may for instance give priority to the most recent information by selecting the pairs $(s_{k,0}, y_{k,0}), \ldots, (s_{k,r-1}, y_{k,r-1}), (s_k, y_k)$ or to exact secant equations (as in the usual limited-memory BFGS) by including pairs $(s_{k-m+1}, y_{k-m+1}), \ldots, (s_k, y_k)$ instead. Any combination of the above is also possible.

3 Numerical Experience with Multi-Secant Equations

We now illustrate the performance of the multi-secant multigrid limited-memory BFGS algorithm on test problems exhibiting the multigrid structure which is of interest to us.

3.1 Test Examples

We now briefly describe the problems on which our multi-secant limited-memory BFGS algorithm has been applied.

3.1.1 DN: A Dirichlet-to-Neumann Transfer Problem

Let S be the square $[0,\pi]\times[0,\pi]$ and let Γ be its lower edge defined by $\{(x,y), 0\leq x\leq \pi, y=0\}$. The Dirichlet-to-Neumann transfer problem (Lewis and Nash, 2005) consists in finding the function $a(x)$ defined on $[0,\pi]$, that minimizes

$$\int_0^\pi \left[\partial_y u(x,0)-\phi(x)\right]^2\,dx,$$

where $\partial_y u$ is the partial derivative of u with respect to y, and where u is the solution of the boundary value problem

$$\begin{array}{lcll}\Delta u & = & 0 & \text{in } S,\\ u(x,y) & = & a(x) & \text{on } \Gamma,\\ u(x,y) & = & 0 & \text{on } \partial S\backslash\Gamma.\end{array}$$

The problem is a one-dimensional minimization problem, but the computations of the objective function and gradient involve a partial differential equation in two dimensions. To introduce oscillatory components in the solution, we set $\phi(x)=\sum_{i=1}^{15}\left[\sin(i\,x)+\sin(40\,x)\right]$. The discretization of the problem is performed by finite differences with the same grid spacing in the two directions. The discretized problem is a linear least-squares problem.

3.1.2 Q2D: A Simple Quadratic Example

We consider here the two-dimensional model problem for multigrid solvers in the unit square domain $S_2=[0,1]\times[0,1]=\{(x,y), 0\leq x\leq 1, 0\leq y\leq 1\}$:

$$\begin{array}{rcl}-\Delta u(x,y) & = & f \text{ in } S_2\\ u(x,y) & = & 0 \text{ on } \partial S_2,\end{array}$$

where f is such that the analytical solution to this problem is $u(x,y)=2y(1-y)+2x(1-x)$. This problem is discretized using a 5-point finite-difference scheme, giving linear systems $A_i x=b_i$ at level i where each A_i is a symmetric positive-definite matrix. The optimization is carried out on the variational minimization problem

$$\min_{x\in\mathbb{R}^{n_r}} \tfrac{1}{2}x^T A_r x - x^T b_r, \tag{3.1}$$

which is obviously equivalent to the linear system $A_r x=b_r$. The main purpose of this example is to illustrate that our algorithm is able to exploit the best multigrid structure.

3.1.3 MS: A Minimum Surface Problem

We consider the minimum surface problem

$$\min_{v\in K}\int_0^1\int_0^1 \sqrt{1+(\partial_x v)^2+(\partial_y v)^2}\,dx\,dy,$$

where $K=\left\{v\in H^1(S_2)\mid v(x,y)=v_0(x,y) \text{ on } \partial S_2\right\}$. The boundary condition v_0 is chosen as

$$v_0(x,y)=\begin{cases} f(x), & y=0, \quad 0\leq x\leq 1,\\ 0, & x=0, \quad 0\leq y\leq 1,\\ f(x), & y=1, \quad 0\leq x\leq 1,\\ 0, & x=1, \quad 0\leq y\leq 1,\end{cases}$$

where $f(x) = x(1-x)$. This convex problem is discretized using a finite element basis defined using a uniform triangulation of S_2, with same grid spacing h along the two coordinate directions. The basis functions are the classical P1 functions which are linear on each triangle and take value 0 or 1 at each vertex. The starting point is a random vector uniformly distributed in $[0,1]$.

3.1.4 BR: Bratu's Problem in Two Dimensions

We consider the minimization problem

$$\min_{v\in K} \int_0^1 \int_0^1 (\partial_x v)^2 + (\partial_y v)^2 + R\, v\, e^v \; dx\, dy,$$

where $K = \left\{v \in H^1(S_2) \mid v(x,y) = 0 \text{ on } \partial S_2\right\}$, corresponding to a variationnal formulation of the partial differential equation

$$\begin{aligned} -\Delta u(x,y) + Re^u &= 0 \text{ in } S_2 \\ u(x,y) &= 0 \text{ on } \partial S_2, \end{aligned}$$

where $R = 6.8$, as advocated in Moré and Toraldo (1991). This variational problem is discretized using the same finite element basis as that used in the MS problem. The starting point is a random vector uniformly distributed in $[0,1]$.

3.1.5 IP: An Inverse Problem from Image Processing

We consider the image deblurring problem stated on page 130 of Vogel (2002). In this problem, the columns of the unknown deblurred image are stacked into a vector f. A doubly block Toeplitz matrix T is computed using the `blur` function of Hansen's toolbox (Hansen 1994), which also yields the blurred image d. The image deblurring problem uses the total variation principle and can be written as

$$\min_f \left[\tfrac{1}{2}\|Tf - d\|_2^2 + \tfrac{1}{10} \int_0^1 \int_0^1 \sqrt{1 + (\partial_x f)^2 + (\partial_y f)^2}\, dx\, dy \right].$$

The problem is convex. The discretization scheme is the same as for the MS problem, and the starting point for the minimization is chosen as $f = 0$.

3.1.6 MT: A Membrane Tracking Problem

We consider the minimization problem

$$\min_{v\in K} \int_0^1 \int_0^1 (\partial_x v)^2 + (\partial_y v)^2 + v \; dx\, dy.$$

The set of contraints is defined by

$$K = \left\{v \in H^1(S_2) \mid v(x,y) = 0 \text{ on } \Gamma_1 \text{ and } v(x,y) \geq l(x,y) \text{ on } \Gamma_2\right\},$$

where boundary conditions are defined on the line segments $\Gamma_1 = \{0\} \times [0,1]$ and $\Gamma_2 = \{1\} \times [0,1]$, and where $l(x,y) = \sqrt{1 - (y-0.5)^2} - 1.3$. This problem corresponds to the displacement of a membrane under a traction of unit density (see Hlavacek, Haslinger, Necas and Lovisek, 1998). To fit into our unconstrained optimization framework, this bound-constrained problem is transformed into a problem with Dirichlet boundary conditions by replacing K with

$$K' = \left\{v \in H^1(S_2) \mid v(x,y) = 0 \text{ on } \Gamma_1 \text{ and } v(x,y) = l(x,y) \text{ on } \Gamma_2\right\}.$$

The discretization of the above problem again uses the same finite element basis as for the MS problem. The starting point is a random vector uniformly distributed in $[0,1]$.

3.1.7 DAHNL: Data Assimilation for the Nonlinear Heat Equation

Data assimilation problems constitute an important class of parameter estimation. Their purpose is to reconstruct the initial conditions at $t = 0$ of a dynamical system based on knowledge of the system evolution laws and on observations of the state at times t_i. More precisely, consider a dynamical system described by the equation $\dot{x} = f(t, x)$ whose solution operator is given by $x(t) = \mathcal{M}(t, x_0)$. Assume that the system state is observed (possibly only in parts) at times $\{t_i\}_{i=0}^N$, yielding observation vectors $\{y_i\}_{i=0}^N$, whose model is given by $y_i = \mathcal{H}x(t_i) + \epsilon$, where ϵ is a noise with covariance matrix R_i. Assume finally that one knows B, an *a priori* error covariance matrix on x_0. We are then interested to find x_0 which minimizes

$$\frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{i=0}^{N} \|\mathcal{H}\mathcal{M}(t_i, x_0) - y_i\|^2_{R_i^{-1}}.$$

The first term in this cost function is the often called the background term, the second the observation term.

The first problem of this type in our test examples is a relatively simple case where the dynamical system is the nonlinear heat equation in a two-dimensional domain, defined on S_2 by

$$\frac{\partial u}{\partial t} = \Delta u - \sin(u) \quad \text{in } S_2, \tag{3.2}$$

No background term is considered in this example. The data of our problem is the computed by imposing a solution $x_0(x, y, 0) = \frac{1}{4}\sin(\frac{1}{4}x)(x-1)\sin(5y)(y-1)$, computing the exact system trajectory and observing it at every point in the spatial domain and at every time step, but with a random measurement error uniformly distributed in $[0, 10^{-4}(1+\sin(\pi x))(1+\sin(\pi y))]$. The problem DAHNL is then to reconstruct $x_0(x, y, 0)$. The system is integrated for ten timesteps of length 2 using an implicit Euler scheme. The initial point is a random vector uniformly distributed in $[0, 100]$.

3.1.8 DASW: Data Assimilation for the Shallow-Water System

A more complex and more realistic example is given by a system governed by the (two-dimensional) shallow-water equations. This system is often considered as a good approximation of the dynamical systems used in ocean modeling, themselves a crucial element of climatic evolution scenarii (Griffies, 2004). The system's equations are

$$\left\{\begin{array}{rcl} \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial v}{\partial y} - fv + g\frac{\partial z}{\partial x} & = & 0, \\ \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + fu + g\frac{\partial z}{\partial y} & = & 0, \\ \frac{\partial z}{\partial t} + u\frac{\partial z}{\partial x} + v\frac{\partial z}{\partial y} + z\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) & = & 0, \end{array}\right. \tag{3.3}$$

where u, v and z are functions of (x, y, t). The domain is the rectangle $[0, L_x] \times [0, L_y]$ (with $L_x = 32 \times 10^6$ and $L_y = 8 \times 10^6$) and the integration horizon is 50 timesteps of 400 seconds. The boundary condition are assumed to be periodic in y and of Dirichlet type in x. Following the suggestion by Weaver and Courtier (2001) we have modelled the *a priori* term x_b using a diffusion operator. As is recommended in the climate modelling community (we refer the reader to Griffies, 2004, for further details), our formulation uses initial geostrophic winds and a β-plane formulation for the Coriolis force; we integrate this system using a leapfrog scheme and a Laplacian spatial damping (which introduces a right-hand side of the form $\mu(\Delta u, \Delta v, \Delta z)^T$ in the system (3.3), where $\mu = 3 \times 10^6$). We also considered using an Asselin time-filter (see Asselin, 1972), but gave this up as it had little effect on our results. In our problem, we assume to observe the state at every 5 points in the spatial domain and every 5 time steps. The true initial geopotential height,

which we seek to reconstruct, is assumed to be

$$z(x,y,0) = 5000 + 50\sin\left(\frac{2\pi x}{L_x}\right)\cos^2\left(\frac{\pi(y-\frac{1}{2}L_y)}{L_y}\right).$$

The starting point of this non-convex problem is chosen as the initial state perturbed by normally distributed random noise with mean and standard deviation equal to 10^{-2}.

3.2 The First Results

We now turn to the numerical experiments themselves, where we considered the test examples described above for the sizes and algorithmic parameters presented in Table 3.1. In this table, m is the total number of secant pairs (exact and approximate) memorized by our algorithm and "# levels" is the number of levels exploited in the multi-secant algorithm. All experiments were run in Matlab. We used the full-weighting restriction and the linear-interpolation prolongation operators in all cases, and μ was set to 10^{-6}. Convergence of the algorithm was declared as soon as $\|g(x_k)\| \leq \epsilon$, the values of this tolerance being also specified in Table 3.1.

		DN	Q2D	MS	BR
size	(n)	255	$1046529 = 1023^2$	$1046529 = 1023^2$	$261121 = 511^2$
memory	(m)	10	9	9	7
# levels	(r)	7	8	7	3
accuracy	(ϵ)	10^{-5}	10^{-5}	10^{-5}	10^{-5}

		IP	MT	DAHNL	DASW
size	(n)	$66049 = 257^2$	$262143 = 511 \times 513$	$3969 = 63^2$	$3969 = 63^2$
memory	(m)	9	8	5	5
# levels	(r)	4	6	4	4
accuracy	(ϵ)	10^{-5}	10^{-5}	10^{-3}	10^{-3}

Table 3.1: The parameters in our numerical tests

The algorithmic variants tested differ according to the mechanism used in Step 5 to select the secant pairs, the order in which these pairs are used in the implicit updating procedure and the level of collinearity control τ. We have defined three different strategies for the selection of secant pairs:

full: all secant pairs (exact and approximate) are considered for each iteration, but the pairs generated at the iterations further in the past are dropped first, and, amongst those corresponding to the same iteration, the approximate pairs are dropped before the exact one;

local: only the approximate secant pairs generated at the current iterations are considered for updating, in addition to as many past exact pairs as allowed by the memory;

mless: all information from previous iterations is discarded and only the (exact and approximate) pairs generated at the current iteration are considered. This corresponds to using the multiple-secant updates in a purely "memoryless" manner.

We have also defined two different strategies for the order in which the approximate secant pairs are used for updating :

coarse-first: the approximate inverse Hessian is updated for the pairs corresponding to the coarser levels first;

fine-first: the approximate inverse Hessian is updated for the pairs corresponding to the finer levels first.

Each algorithmic variant is thus characterized by the triplet specifying its the pair selection strategy, its pair ordering strategy and its value of τ. We also consider the standard limited-memory BFGS method (L-BFGS) for comparison. Table 3.2 on the current page presents the results of our first tests, expressed in terms of number of objective function evaluations (nf) and iterations (nit).

Multi-secant LM variant			DN		Q2D		MS		BR	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	152	122	381	304	2130	1716	472	381
full	coarse-first	0.999	131	108	515	432	2541	2194	547	465
full	fine-first	1.0	130	105	366	273	2097	1815	428	367
full	fine-first	0.999	120	98	638	550	2361	2099	396	337
local	coarse-first	1.0	94	84	563	427	2003	1707	440	372
local	coarse-first	0.999	110	92	501	385	2033	1725	450	384
local	fine-first	1.0	120	100	304	233	2267	1943	397	339
local	fine-first	0.999	90	76	335	278	1867	1605	400	341
mless	coarse-first	1.0	125	100	563	427	1724	1364	527	422
mless	coarse-first	0.999	113	89	501	385	2207	1832	401	316
mless	fine-first	1.0	137	100	304	233	1679	1371	420	313
mless	fine-first	0.999	140	107	335	278	2204	1844	445	338
L-BFGS			330	319	1505	1471	2671	2644	1074	1053

Multi-secant LM variant			IP		MT		DAHNL		DASW	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	196	186	1542	1277	25	24	73	70
full	coarse-first	0.999	204	206	1262	1041	25	24	73	70
full	fine-first	1.0	213	200	1070	883	29	27	64	62
full	fine-first	0.999	214	210	1272	1077	29	27	64	62
local	coarse-first	1.0	165	161	929	781	39	35	73	70
local	coarse-first	0.999	165	162	1070	882	39	35	73	70
local	fine-first	1.0	185	184	1027	861	37	33	64	62
local	fine-first	0.999	180	179	1138	959	37	33	64	62
mless	coarse-first	1.0	355	353	1536	1221	28	27	73	70
mless	coarse-first	0.999	280	279	1307	1051	31	30	73	70
mless	fine-first	1.0	265	256	862	650	28	27	64	62
mless	fine-first	0.999	232	225	1461	1149	31	30	64	62
L-BFGS			181	176	1419	1386	27	25	60	57

Table 3.2: Performance of multi-secant limited-memory BFGS algorithms

These results indicate that using approximate secant pairs associated with grid levels is potentially useful, although not uniformly for every problem nor across variants. Further variations were also observed for other choices of m and r, but leave the overall picture unchanged. In the results reported here, the improvement is especially noticeable for problems (like Q2D and BR) where one expects the multigrid method to perform well. One also notes that the "local" and "mless" variants seem to be most efficient, and that collinearity control often helps somewhat the first of these strategies. The memoryless variants ("mless") are especially efficient on the Laplacian (Q2D) and Bratu (BR) problems. We also note that the memory-less and local variants give identical number of function evaluations and iterations for problem Q2D, which we believe results from the fact that $r \approx m$, which leaves little room for past iteration history. The same comment applies to the two data assimilation problems (DASW and DAHNL), for which the multisecant variants are clearly less successful. This is probably because the inherent time integration mixes the frequency components of the errors to a larger extent than for the other problems, blurring the invariant subspace interpretation. We finally observe that collinearity control produces fairly mixed results, and seems to irrelevant for the data assimilation problems.

In order to verify the analysis of Section 2.1, we have also computed the relative perturbations (2.3) (with $G = \nabla_{xx} f(x_{k+1})$ and $\|\cdot\| = \|\cdot\|_\infty$) during runs of the variant [local,coarse-first,1.0] on problems Q2D and MS. Typical results are shown in Figure 3.1.

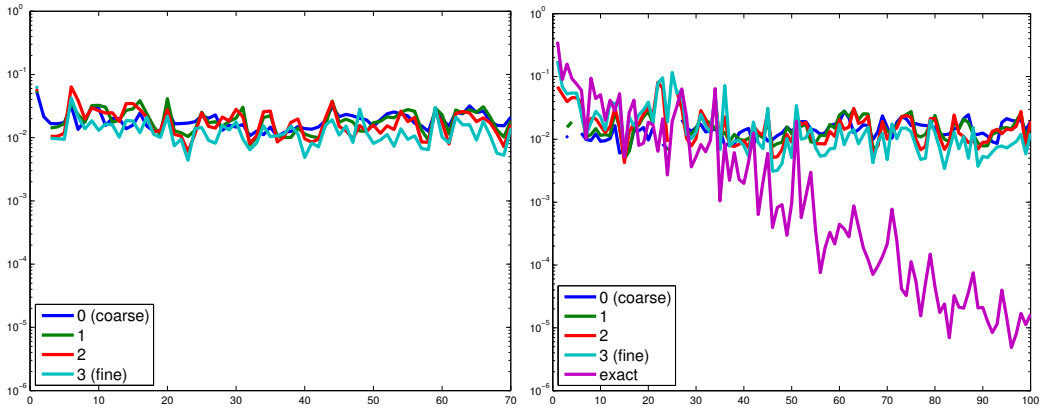

Figure 3.1: Evolution of the relative Hessian perturbation sizes (2.3) with k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ for problems of size $n = 63^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

This figure shows that the size of the relative perturbation of the true Hessian needed to make the approximate secant exact is very modest (a few percents of $\|\nabla_{xx} f(x_{k+1})\|$, typically). The relative size of the Hessian perturbation necessary make the exact secant equation (1.5) hold exactly is shown in the right figure as the curve ultimately decreasing to the order of 10^{-5}. Clearly, this perturbation is of the same order as that for the approximate secant equations in the early iterations of the algorithm. It is invisible on the left figure, because it is always tiny (between 10^{-15} and 10^{-10}) on a quadratic function. Further analysis (not illustrated here) indicates that the perturbation corresponding to past exact secant equations follows the same pattern as that corresponding to the current one, both for quadratic and nonquadratic problems.

4 Asymptotic Approximate Eigenvalue Equations

These numerical experiments prompt another discussion. It was observed in the numerical test-runs that the multiple-secant limited-memory BFGS algorithm also acts as a smoother on the original problem, in the sense that convergence often occurs much faster for the oscillatory modes of the solution than for the smooth modes. This is illustrated in Figure 4.2 which shows the decomposition of the step s_k along the subspaces $\mathcal{S}_0, \mathcal{S}_1^C, \mathcal{S}_2^C, \ldots, \mathcal{S}_r^C$, where, for $i > 0$, $\mathcal{S}_i^C$ is the orthogonal complement of $\mathcal{S}_{i-1}$ in $\mathcal{S}_i$.
In this figure, one observes that, for both problems, s_k is nearly entirely contained in $\mathcal{S}_0$ (the coarse subspaces corresponding to very smooth modes), for k sufficiently large, which we express by writing that

$$s_k \approx S_0 s_k \tag{4.1}$$

for large k. Now observe that the eigenvalues associated with $\mathcal{S}_0$ are typically the smallest ones: in the case of the one-dimensional Laplacian model problem analyzed above, we have that

$$S_0 = \text{span}\left\{ \sin\left(\frac{ij\pi}{n}\right) \right\}_{i=1}^{n_0}$$

and the eigenvalues associated with this invariant subspace are

$$\lambda_i = 4\sin^2\left(\frac{i\pi}{2n}\right) \quad (i = 1, \ldots, n_0). \tag{4.2}$$

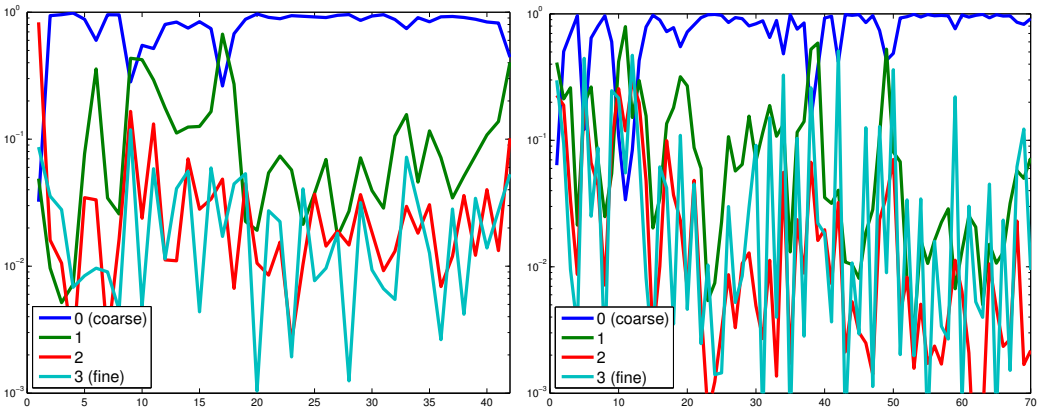

Figure 4.2: Evolution of the decomposition of the step s_k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ along $\mathcal{S}_0, \mathcal{S}_1^C, \mathcal{S}_2^C, \ldots, \mathcal{S}_r^C$ for problems of size $n = 31^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

If we assume, for instance, that $n = 255$ and that we exploit five levels, we verify that $n_0 = 7$. The eigenvalues (4.2) are therefore all contained in the interval $[0.00015, 0.0074]$. The norm of the Laplacian operator is however given by its maximal eigenvalue, equal to 4. Thus the deviation of the eigenvalues associated to $\mathcal{S}_0$ from any ρ_0 in the interval, relative to the operator norm, is bounded by

$$\frac{0.0074 - 0.00015}{4} \approx 0.0018,$$

that is slightly less than 0.2%. Modifying the Laplacian operator by imposing that its restriction to S_0 is $\rho_0 I$ therefore amounts to a perturbation of the operator of at most 0.2%, which is again very modest. Returning to the case discussed in Section 2.1, this indicates that

$$Gs_k \approx GS_0 s_k \approx \rho_0 Gs_k \approx \rho s_k,$$

and $s_k \approx S_0 s_k$ is an approximate eigenvector of G, the value ρ_0 being the Rayleigh quotient along this direction, given by

$$\rho_0 \|S_0 s_k\|^2 = \langle S_0 s_k, GS_0 s_k\rangle \approx \langle S_0 s_k, S_0 G s_k\rangle \approx \langle S_0 s_k, S_0 y_k\rangle.$$

Moreover,

$$S_0 y_k = S_0 G s_k = GS_0 s_k \approx Gs_k = y_k. \tag{4.3}$$

If we now consider $\mathcal{S}_1$ (a superset of $\mathcal{S}_0$ of dimension 15 in our example), the size of relative perturbation Hessian remains at most 9%. Expressing the above reasoning for an a generic invariant subspace $\mathcal{S}_i$ (note that $\mathcal{S}_0 \subset \mathcal{S}_i$ in our multigrid context), we may thus expect that

$$GS_i s_k \approx \frac{\langle GS_i s_k, S_i s_k\rangle}{\|S_i s_k\|_2^2} S_i s_k \approx \frac{\langle y_k, S_i s_k\rangle}{\|S_i s_k\|_2^2} S_i s_k \approx \frac{\langle S_i y_k, S_i s_k\rangle}{\|S_i s_k\|_2^2} S_i s_k,$$

which gives that

$$HS_i s_k \approx \frac{\|S_i s_k\|^2}{\langle S_i y_k, S_i s_k\rangle} S_i s_k. \tag{4.4}$$

Alternatively,

$$\langle S_i y_k, HS_i y_k\rangle \approx \langle S_i y_k, S_i H y_k\rangle \approx \langle S_i y_k, S_i s_k\rangle \approx \langle y_k, S_i s_k\rangle,$$

yielding that

$$HS_i s_k \approx \frac{\langle S_i y_k, S_i s_k\rangle}{\|S_i y_k\|_2^2} S_i s_k. \tag{4.5}$$

Using the approximations (4.1) and (4.3), we see that equations of the form

$$Hp_{i,k} \approx \rho_{i,k} q_{i,k} \quad (i = 0, \ldots, r-1), \tag{4.6}$$

may therefore be of interest when updating our Hessian approximation, where we have the choice to select $p_{i,k}$ and $q_{i,k}$ among $S_i s_k$ $(0 \leq i < r-1)$ or s_k, and $\rho_{i,k}$ among

$$\frac{\|S_i s_k\|_2^2}{\langle y_k, S_i s_k \rangle}, \quad \frac{\|S_i s_k\|_2^2}{\langle S_i y_k, S_i s_k \rangle}, \quad \frac{\langle y_k, S_i s_k \rangle}{\|S_i y_k\|_2^2} \text{ or } \frac{\langle S_i y_k, S_i s_k \rangle}{\|S_i y_k\|_2^2} \tag{4.7}$$

Further alternative forms may be obtained by replacing $S_i s_k$ by s_k or $\|S_i s_k\|_2^2$ by $\langle s_k, S_i s_k \rangle$ or $\|s_k\|_2^2$, or $S_i y_k$ by y_k in (4.7)...

We finally note that Algorithm 2.2 was stated above using only secant equations of the form (2.4), but it is now possible to incorporate equations of the form (4.6) (possibly in combination with (2.4)) into the updates. This only requires the redefinition of the vectors $y_{k,i}$ to be generated at Step 4: for instance, we redefine $y_{k,i}$ to be

$$\frac{\langle y_{k,i}, s_{k,i} \rangle}{\|s_{k,i}\|_2^2} s_{k,i}$$

if one wishes to use (4.4).

5 Numerical Experience with Eigenvalue Equations

Our present purpose is not to conduct a full investigation on which of the alternative forms mentioned in the previous section is numerically preferable and when. We focus in this section on verifying the main lines of our analysis of approximate eigenvalue equations and on illustrating the potential of a few choices in a simple algorithmic setting. We leave the more complete experimentation and the design of a suitable selection mechanism for further research.

In this first exploration, we have chosen to experiment with four variants of the many possible choices of the type (4.6)-(4.7), namely the "eigenvalue equations" given by (4.4), (4.5),

$$HS_i s_k \approx \frac{\|S_i s_k\|_2^2}{\langle y_k, S_i s_k \rangle} S_i s_k, \tag{5.1}$$

and

$$HS_i s_k \approx \frac{\langle y_k, S_i s_k \rangle}{\|S_i y_k\|_2^2} S_i s_k. \tag{5.2}$$

The results of the numerical experience using these equations in the limited memory framework already detailed for approximate secant equations are given in Tables 5.3 to 5.10.

Qualitatively speaking, we obtain a conclusion very similar to that reached for approximate secant equations: there exists a clear algorithmic potential in using the information specified by the approximate eigenvalue equations, but this potential is uniform neither across problems nor across algorithmic variants. We only note that some of the best performances are obtained with these techniques, in particular for the data assimilation problems.

As for the multiple secant equations, we may complete our picture by an estimation of the approximate nature of the considered eigenvalue equations. This estimation is again obtained by measuring the size of the Hessian perturbation that would ensure (4.6) exactly, relative to the Hessian norm. In this case ,the relative perturbation size is then given by

$$\frac{\|E_i\|_\infty}{\|G\|_\infty} \leq \frac{\|Gp_{i,k} - \rho_{i,k} q_{i,k}\|_\infty}{\|p_{i,k}\|_\infty \|G\|_\infty} \tag{5.3}$$

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	219	129	158	126	132	118	139	114
full	coarse-first	0.999	222	125	129	106	177	160	153	129
full	fine-first	1.0	529	272	176	134	129	117	145	106
full	fine-first	0.999	243	127	157	118	145	129	113	89
local	coarse-first	1.0	156	96	121	101	233	217	138	117
local	coarse-first	0.999	164	106	122	99	195	178	126	103
local	fine-first	1.0	159	100	104	86	116	104	101	85
local	fine-first	0.999	159	103	127	105	135	124	123	105
mless	coarse-first	1.0	199	112	169	136	246	223	138	111
mless	coarse-first	0.999	224	128	178	132	239	219	153	125
mless	fine-first	1.0	229	128	122	90	229	192	168	135
mless	fine-first	0.999	216	119	155	117	188	159	156	125

Table 5.3: Performance of eigenvalue limited-memory BFGS algorithms on problem D2N (L-BFGS uses 330 function evaluations and 319 iterations on this problem)

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	707	564	806	608	689	534	552	440
full	coarse-first	0.999	592	470	868	672	704	592	689	595
full	fine-first	1.0	414	316	439	336	565	467	842	675
full	fine-first	0.999	908	749	1068	853	1185	966	770	637
local	coarse-first	1.0	551	437	771	576	630	482	686	551
local	coarse-first	0.999	405	313	524	410	532	410	602	481
local	fine-first	1.0	690	528	479	366	558	433	520	428
local	fine-first	0.999	579	479	389	313	699	582	518	413
mless	coarse-first	1.0	551	437	771	576	630	482	686	551
mless	coarse-first	0.999	405	313	524	410	532	410	532	481
mless	fine-first	1.0	690	528	479	366	558	433	520	428
mless	fine-first	0.999	579	479	389	313	699	582	518	413

Table 5.4: Performance of eigenvalue limited-memory BFGS algorithms on problem Q2D (L-BFGS uses 1505 function evaluations and 1471 iterations on this problem)

for the appropriate choices of $p_{i,k}$, $q_{i,k}$ and $\rho_{i,k}$. The results obtained during runs of the variant [local,coarse-first,1.0] on problems Q2D and MS are illustrated in Figures 5.3 to 5.6. These pictures confirm our analysis that the limited-memory framework may generate useful approximate eigenvalue information. The quality and relevance of this information varies according to the precise choice of approximate eigenvalue equation used and problem considered. Once more, this encourages further investigation in methods exploiting this information .

6 Conclusions

We have shown that the *a priori* knowledge of approximate invariant subspaces associated with the Hessian of an unconstrained optimization problem allows a more efficient exploitation of the secant information, and thus more efficient minimization algorithms. We have also described how this knowledge can often be extracted from the multigrid structure of discretized infinite dimensional problems. Using the asymptotic smoothing properties of the limited-memory BFGS method, we have indicated how the steps generated by this algorithm often generate approximate eigenvalue information, which may also be used to improve the efficiency of the optimization. Preliminary numerical experience on a small collection of test problems suggests that this approach is promising and further

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	2123	1672	2663	2027	2277	1998	2627	2326
full	coarse-first	0.999	2674	2145	2780	2170	2595	2295	2466	2187
full	fine-first	1.0	2300	1895	2192	1772	1957	1683	2101	1817
full	fine-first	0.999	3097	2561	2610	2117	3042	2588	3259	2792
local	coarse-first	1.0	2780	2270	2869	2227	2511	2213	2375	2135
local	coarse-first	0.999	2565	2094	2275	2214	2369	2078	2367	2099
local	fine-first	1.0	2147	1872	2168	1851	2147	1917	2029	1808
local	fine-first	0.999	1698	1470	1637	1379	1848	1654	2316	2080
mless	coarse-first	1.0	2840	2107	2246	2246	2356	2043	2180	1931
mless	coarse-first	0.999	2429	1889	2753	1997	2346	1997	2546	2182
mless	fine-first	1.0	1494	1207	1833	1477	1992	1718	1853	1643
mless	fine-first	0.999	2036	1974	1922	1627	3075	2674	3397	3045

Table 5.5: Performance of eigenvalue limited-memory BFGS algorithms on problem MS (L-BFGS uses 2671 function evaluations and 2644 iterations on this problem)

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	511	425	532	443	555	484	552	490
full	coarse-first	0.999	424	352	580	478	471	405	622	544
full	fine-first	1.0	321	251	420	326	469	392	593	494
full	fine-first	0.999	377	291	557	460	413	342	573	479
local	coarse-first	1.0	383	334	584	486	494	441	390	348
local	coarse-first	0.999	427	358	647	522	523	462	363	322
local	fine-first	1.0	365	329	548	466	527	465	315	282
local	fine-first	0.999	381	339	449	396	419	364	415	359
mless	coarse-first	1.0	660	525	518	391	575	473	548	468
mless	coarse-first	0.999	405	314	432	320	605	486	490	404
mless	fine-first	1.0	432	343	535	407	419	339	547	431
mless	fine-first	0.999	301	234	428	330	338	282	665	522

Table 5.6: Performance of eigenvalue limited-memory BFGS algorithms on problem BR (L-BFGS uses 1074 function evaluations and 1053 iterations on this problem)

investigation worthwhile.

The exploitation of invariant subspace information opens, from the authors' point of view, a number of immediate possibilities and more long-term perspectives. The first is that, although algorithmic variants have been outlined in the present paper, much remains to be done for obtaining a well-tested, robust and optimized multi-secant/eigenvalue quasi-Newton code. But other questions also merit further research. One may wonder, for instance, if the BFGS updates could performed at the different grid levels and the resulting matrices prolongated to the fine grid. One may also consider the effect of using simultaneous updates or other quasi-Newton updates in the context of multi-secant algorithms, or the use of the resulting approximations in multilevel optimization algorithms of the type suggests by Nash (2000), Gratton et al. (2007*a*) or Wen and Goldfarb (2007). Alternative (possibly iterative) ways to obtain approximate invariant subspace information are obviously also of interest. Yet further research questions include the effect of the particular choice of the restriction and prolongation operators, the impact of grid refinement strategies, and the use of multi-secant/eigenvalue approaches in algorithms for bound- and more generally constrained optimization.

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	330	303	282	274	256	252	248	241
full	coarse-first	0.999	367	314	206	199	237	229	233	228
full	fine-first	1.0	2007	1015	186	181	610	609	189	187
full	fine-first	0.999	1738	881	178	609	610	609	198	196
local	coarse-first	1.0	223	220	164	174	166	165	165	164
local	coarse-first	0.999	229	226	164	163	169	168	169	168
local	fine-first	1.0	680	405	179	178	211	210	184	183
local	fine-first	0.999	727	444	184	183	219	218	184	183
mless	coarse-first	1.0	285	261	195	189	344	343	191	189
mless	coarse-first	0.999	371	314	246	195	344	343	200	197
mless	fine-first	1.0	959	484	198	245	3061	3060	236	235
mless	fine-first	0.999	1292	695	240	239	3061	3060	233	230

Table 5.7: Performance of eigenvalue limited-memory BFGS algorithms on problem IP (L-BFGS uses 181 function evaluations and 176 iterations on this problem)

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	1219	1060	1363	1099	1219	1072	1292	1140
full	coarse-first	0.999	1370	1167	1295	1034	1588	1384	1469	1294
full	fine-first	1.0	974	805	935	749	1063	934	1232	1075
full	fine-first	0.999	1064	847	1334	1074	1605	1377	1593	1367
local	coarse-first	1.0	1544	1299	1221	987	1334	1190	1340	1205
local	coarse-first	0.999	1512	1279	1275	1033	1279	1124	1277	1128
local	fine-first	1.0	1129	964	977	823	1293	1113	1205	1043
local	fine-first	0.999	1262	1079	931	786	1312	1146	1496	1375
mless	coarse-first	1.0	1311	1050	1515	1531	1311	1300	1469	1256
mless	coarse-first	0.999	1304	1050	1682	1292	1775	1471	1417	1188
mless	fine-first	1.0	1106	881	1145	880	1423	1136	1064	866
mless	fine-first	0.999	890	703	1349	1094	1981	1649	1540	1270

Table 5.8: Performance of eigenvalue limited-memory BFGS algorithms on problem MT (L-BFGS uses 1419 function evaluations and 1386 iterations on this problem)

Acknowledgements

The authors thank Michael and Stefan Ulbrich for their interest in this research at the ICCOPT II conference. They are also indebted to Anthony Weaver for his friendly advice on the numerical aspects of the shallow-water system.

References

R. A. Asselin. Frequency filter for time integration. *Monthly Weather Review*, **100**, 487–490, 1972.

W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, Philadelphia, USA, 2nd edn, 2000.

C. G. Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and its Applications*, **6**, 76–90, 1970.

R. H. Byrd, P. Lu, and J. Nocedal. A limited memory algorithm for bound constrained optimization. Technical Report NAM-08, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois, USA, 1993.

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	29	24	30	28	30	29	28	27
full	coarse-first	0.999	29	24	30	28	30	29	28	27
full	fine-first	1.0	30	27	27	26	43	32	30	25
full	fine-first	0.999	20	27	27	26	43	32	30	25
local	coarse-first	1.0	42	39	36	34	42	43	48	43
local	coarse-first	0.999	42	39	36	34	42	43	48	43
local	fine-first	1.0	39	34	36	33	46	30	34	30
local	fine-first	0.999	39	34	36	33	46	30	34	30
mless	coarse-first	1.0	27	26	26	24	29	28	27	26
mless	coarse-first	0.999	27	26	26	24	29	28	27	26
mless	fine-first	1.0	29	26	23	22	37	30	29	25
mless	fine-first	0.999	29	26	23	22	37	30	29	25

Table 5.9: Performance of eigenvalue limited-memory BFGS algorithms on problem DAHNL (L-BFGS uses 27 function evaluations and 25 iterations on this problem)

Eigenvalue LM variant			(5.1)		(4.4)		(5.2)		(4.5)	
Pairs	upd. order	τ	nf	nit	nf	nit	nf	nit	nf	nit
full	coarse-first	1.0	65	62	83	80	47	45	79	71
full	coarse-first	0.999	65	62	83	80	47	45	79	71
full	fine-first	1.0	61	58	93	92	81	76	59	51
full	fine-first	0.999	61	58	93	92	81	76	59	51
local	coarse-first	1.0	65	63	83	80	47	45	79	71
local	coarse-first	0.999	65	63	83	80	47	45	79	71
local	fine-first	1.0	61	58	91	88	91	84	57	51
local	fine-first	0.999	61	58	91	88	91	84	57	51
mless	coarse-first	1.0	65	63	83	80	47	45	79	71
mless	coarse-first	0.999	65	63	83	80	47	45	79	71
mless	fine-first	1.0	61	58	91	88	91	84	57	51
mless	fine-first	0.999	61	48	91	88	91	84	57	51

Table 5.10: Performance of eigenvalue limited-memory BFGS algorithms on problem DASW (L-BFGS uses 60 function evaluations and 57 iterations on this problem)

R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, **16**(5), 1190–1208, 1995.

R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representation of quasi-newton matrices and their use in limited meory methods. *Mathematical Programming*, **63**, 129–156, 1994.

W. C. Davidon. Variable metric method for minimization. Report ANL-5990(Rev.), Argonne National Laboratory, Research and Development, 1959. Republished in the SIAM Journal on Optimization, vol. 1, pp. 1–17, 1991.

J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1983. Reprinted as *Classics in Applied Mathematics 16*, SIAM, Philadelphia, USA, 1996.

R. Fletcher. A new approach to variable metric algorithms. *Computer Journal*, **13**, 317–322, 1970.

R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Computer Journal*, **6**, 163–168, 1963.

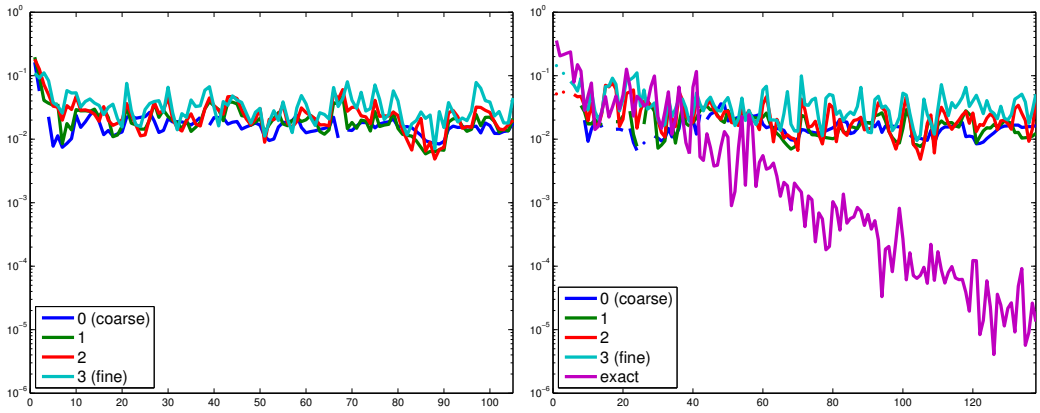

Figure 5.3: Evolution of the relative Hessian perturbation sizes (2.3) with k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ using equation (5.1) for problems of size $n = 63^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

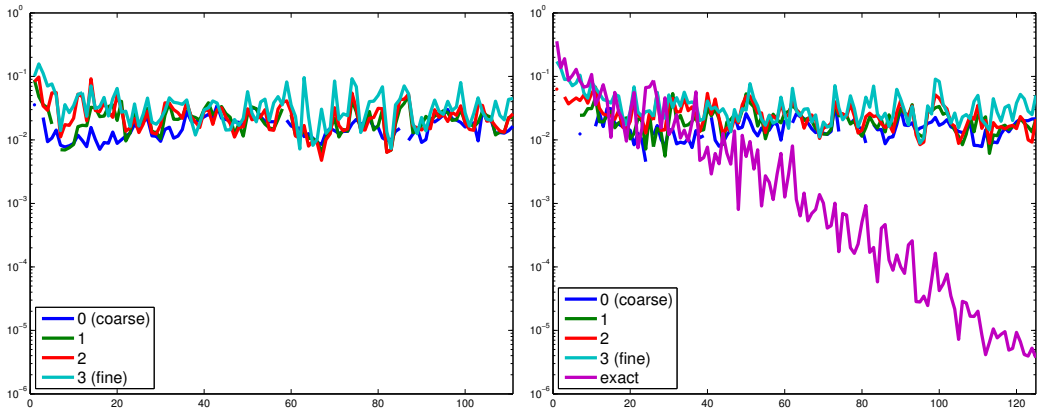

Figure 5.4: Evolution of the relative Hessian perturbation sizes (2.3) with k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ using equation (4.4) for problems of size $n = 63^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

D. Goldfarb. A family of variable metric methods derived by variational means. *Mathematics of Computation*, **24**, 23–26, 1970.

S. Gratton, M. Mouffe, Ph. L. Toint, and M. Weber-Mendonça. A recursive trust-region method in infinity norm for bound-constrained nonlinear optimization. Technical Report 07/01, Department of Mathematics, University of Namur - FUNDP, Namur, Belgium, 2007*a*.

S. Gratton, A. Sartenaer, and Ph. L. Toint. Second-order convergence properties of trust-region methods using incomplete curvature information, with an application to multigrid optimization. *Journal of Computational and Applied Mathematics*, **24**(6), 676–692, 2006.

S. Gratton, A. Sartenaer, and Ph. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, **(to appear)**, 2007*b*.

S. Griffies. *Fundamentals of Ocean Climate Models*. Princeton University Press, Princeteon, USA, 2004.

P. C. Hansen. Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms*, **6**, 1–35, 1994.

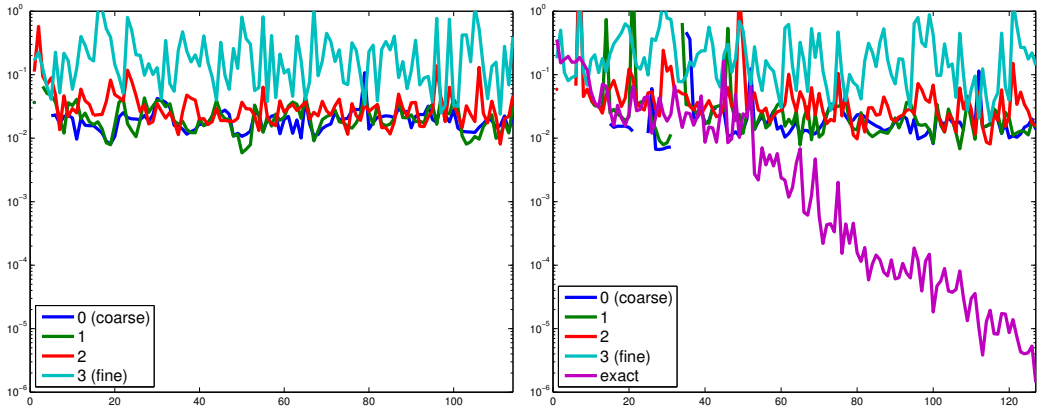

Figure 5.5: Evolution of the relative Hessian perturbation sizes (2.3) with k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ using equation (5.2) for problems of size $n = 63^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

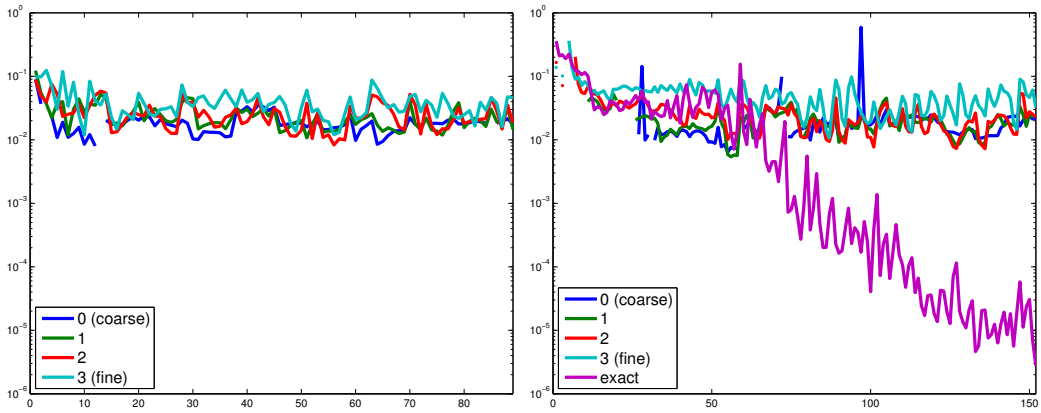

Figure 5.6: Evolution of the relative Hessian perturbation sizes (2.3) with k for the variant [local,coarse-first,1.0] with $m = 7$ and $r = 4$ using equation (4.5) for problems of size $n = 63^2$ (Q2D on the left, MS on the right, logarithmic vertical scale)

I. Hlavacek, J. Haslinger, J. Necas, and J. Lovisek. *Solution of Variational Inequalities in Mechanics*. Springer Verlag, Heidelberg, Berlin, New York, 1998.

M. Lewis and S. G. Nash. Practical aspects of multiscale optimization methods for VLSI-CAD. *in* J. Cong and J. R. Shinnerl, eds, 'Multiscale Optimization and VLSI/CAD', pp. 265–291, Dordrecht, The Netherlands, 2002. Kluwer Academic Publishers.

M. Lewis and S. G. Nash. Model problems for the multigrid optimization of systems governed by differential equations. *SIAM Journal on Scientific Computing*, **26**(6), 1811–1837, 2005.

D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming, Series B*, **45**(1), 503–528, 1989.

J. J. Moré and J. D. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software*, **20**, 286–307, 1994.

J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, **1**(1), 93–113, 1991.

S. G. Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, **14**, 99–116, 2000.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Series in Operations Research. Springer Verlag, Heidelberg, Berlin, New York, 1999.

D. F. Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, **24**, 647–657, 1970.

U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Elsevier, Amsterdam, The Netherlands, 2001.

C. R. Vogel. *Computational Methods for Inverse Problems*, Vol. 23 of *Computational Frontiers in Applied Mathematics*. SIAM, Philadelphia, USA, 2002.

A. Weaver and Ph. Courtier. Correlation modelling on the sphere using a generalized diffusion equation. *Quarterly Journal of the Royal Meteorological Society*, **127**, 1815–1846, 2001.

Z. Wen and D. Goldfarb. A linesearch multigrid methods for large-scale convex optimization. Technical report, Department of Industrial Engineering and Operations Research, Columbia University, New York, July 2007.