

AUTOMATICALLY ASSESSING THE PERFORMANCE OF AN OPTIMIZATION-BASED MULTILEVEL METHOD

STEPHEN G. NASH* AND ROBERT MICHAEL LEWIS†

Abstract. Many large nonlinear optimization problems are based upon a hierarchy of models, corresponding to levels of discretization or detail in the problem. Optimization-based multilevel methods—that is, multilevel methods based on solving coarser versions of an optimization problem—are designed to solve such multilevel problems efficiently by taking explicit advantage of the hierarchy of models. The methods are generalizations of more traditional multigrid methods for solving partial differential equations. These multilevel methods are a powerful tool, but they will not lead to improved performance over traditional algorithms for all optimization problems. We develop techniques whereby a particular multilevel method can assess the properties of the optimization problem, with the goal of automatically determining whether the optimization problem is well suited for the multilevel algorithm. We also show that our diagnostic tests are sufficient to measure the properties of the optimization problem relevant to the performance of the multilevel method.

Key words. Multilevel methods, multigrid methods, optimization of systems governed by differential equations, PDE-constrained optimization

March 27, 2009

1. Introduction. A multilevel algorithm can be a powerful tool for solving optimization problems that correspond to a hierarchy of models. For example, such algorithms can be appropriate when the variables represent an adjustable discretization of some underlying function [1, 16], or when solving a multi-level optimization problem corresponding to a family of models of increasing complexity and fidelity [5]. However, multilevel algorithms are not well suited to all optimization problems, and it can be difficult to predict whether they will work effectively on a particular optimization problem. The goal of this paper is to develop diagnostic tests that can assess the performance of a multilevel method as the optimization problem is being solved, and identify whether the optimization problem has the necessary properties.

The multilevel algorithm applies an optimization algorithm recursively to ever-coarser levels of the optimization problem. The recursion produces search directions for the finer levels that are used to improve the estimates of the solution. Since the computations on the coarse problems are usually much less costly, the multilevel algorithm may be able to compute a solution far more rapidly than a traditional optimization algorithm applied directly to the fine-resolution problem.

Multilevel algorithms arise from multigrid algorithms, which were first developed to solve elliptic partial differential equations. In that setting, the differential operator is stated explicitly and its properties can be studied directly. In contrast, we will be solving optimization problems with an *optimization-based multilevel method*, where on each level an optimization problem is solved, not a system of equations. For the constrained optimization problems that we consider, the corresponding properties of the relevant operator will be unknowable except in the simplest of cases. It may be difficult or impossible to deduce even qualitative information about the operator in advance of solving the problem. Since multilevel algorithms are not general-purpose optimization methods, they are not guaranteed to perform effectively on all problems.

*Systems Engineering and Operations Research Department, Mail Stop 4A6, George Mason University, Fairfax, VA 22030, snash@gmu.edu.

†Department of Mathematics, College of William & Mary, P.O. Box 8795, Williamsburg, Virginia, 23187-8795, buckaroo@math.wm.edu.

So the question arises: How do you determine if it is appropriate to use a multilevel algorithm?

We are not asking about the *convergence* of the multilevel algorithm. The algorithms we study have guarantees of convergence analogous to traditional optimization algorithms [15, 23], assuming that the optimization problems satisfy standard assumptions. Rather, we are asking about the *performance* of the multilevel algorithm. Is the multilevel algorithm likely to perform better than a traditional optimization algorithm on a particular optimization problem?

A simplistic approach would be to apply the multilevel algorithm and see how well it works. This might be appropriate if a single property of the optimization problem would be enough to guarantee good performance. But it is sufficient to examine the following four properties:

- degree of nonlinearity of the optimization problems
- consistency of the optimization problems across levels
- complementarity of the optimization problems and the solver
- separability of the reduced Hessians of the optimization problems across levels

These properties can be intertwined. For example, the reduced Hessian of an optimization problem might share some of the spectral characteristics of the Laplacian (an ideal operator for multigrid), but convergence could be slow because of nonlinearities. In addition, the optimization problems could change character depending on the values of the design variables.

Our aim is to demonstrate that a particular optimization-based multilevel algorithm [16] is capable of assessing these four properties of the optimization problem, as a by-product of finding the solution. Moreover, these diagnostic tests rely on information that can be readily extracted during the normal course of the algorithm at little additional cost. A multilevel algorithm including these tests would be capable of solving challenging hierarchical optimization problems or of suggesting the reasons why it failed to do so. It would not be perfect—there would be problems that would confound tests as simple as those we describe—but it would provide a level of “self awareness” rare among optimization algorithms. In addition, the diagnostic tests would provide guidance if the user wished to investigate further, using more sophisticated tests fine-tuned to the specific optimization problem.

The multilevel algorithm would not always work well, and would not always be appropriate. But if the algorithm had the sort of “self awareness” we describe, it might be possible to identify the reasons why multilevel were not appropriate, and either provide guidance for modifying the optimization problem or identifying an alternative algorithm as necessary. If this could be automated then it would be possible to develop an adaptive optimization environment, where the adaptation would be based on using a variety of optimization algorithms, and not on having a single algorithm adapt to a particular problem (as in adaptive quadrature or mesh refinement). With this strategy, there would be occasions when the optimization took a long time (such as cases where only an algorithm as general as pattern search [13] could be used) but for most problems more effective algorithms would be identified. Hence, on average, optimization would be efficient.

Our tests and our analysis are specific to a particular optimization-based multilevel algorithm called ML/Opt (a modification of our earlier algorithm MG/Opt[16]). ML/Opt is derived from a truncated-Newton optimization algorithm, and uses the conjugate-gradient method for computing search directions. These details make some of our diagnostic tests feasible and practical. It may be possible to derive analogous

diagnostic tests for other multilevel algorithms, but we have not attempted this.

Here is an outline of the paper. Section 2 gives background material on the optimization problems and the multilevel algorithm. Section 3 develops the diagnostic tests. Section 4 demonstrates why the tests are sufficient to assess the performance of the multilevel algorithm. Section 5 offers guidance on what to do when a diagnostic test is not satisfied. Section 6 describes computational experiments. Conclusions are in Section 7.

The paper builds on a variety of research results. It owes a great debt to the vast research on multigrid methods (e.g., [2, 19]). More specifically, one of the diagnostic tools for assessing the coarse models derives from algebraic multigrid methods (e.g., [4]). However, those results cannot be used directly. Algebraic multigrid is typically used to solve linear equations, and assumes that the coefficient matrix is sparse and that its entries are known. This is not true for the optimization-based multilevel methods that we describe.

Related diagnostic tests have been developed for detecting nonlinearity in the context of truncated-Newton methods [10]. Various approaches have been considered for testing convexity (see, e.g., [7, 26]). Finally, analogous tests have been developed for other problem classes, such as branch-and-bound methods for integer programming [9] and iterative solvers for linear equations [12, 17].

2. Background. Much of our discussion applies to a general optimization problem

$$(2.1) \quad \underset{a_h}{\text{minimize}} F_h(a_h)$$

where the objective function F_h and the variables a_h represent a family of problems of varying fidelity or complexity. The subscript h might represent a discretization, but other choices are possible.

Our prior work and our intuition are based on a more specific form of optimization problem

$$(2.2) \quad \underset{a}{\text{minimize}} F(a) = f(a, u(a)),$$

where a is a set of design variables, and $u = u(a)$ is a set of state variables. Given a set of parameter values a , the state variables are defined implicitly by a system of partial differential equations

$$(2.3) \quad S(a, u(a)) = 0$$

in a and u . Here we assume that $S(a, u) = 0$ is solved for u given a . However, even though this more specific problem has provided inspiration, our results apply more generally and not just to discretized problems corresponds to PDEs.

For simplicity, we assume that there are no other constraints on the variables, although this is not essential. The presence of additional constraints, including inequalities, makes the application of multigrid to optimization a true generalization of the multigrid approach [16].

For problems of the form (2.2)–(2.3), particularly in the case where additional constraints are present, the behavior of the multilevel method will depend on the reduced Hessian, an operator derived from the Hessian of the objective function and the Jacobian of the constraint functions. We do not assume that we have explicit information about this matrix. It may not be sparse even in cases where the underlying Hessian and Jacobian are sparse. For these reasons, certain analytical and

computational tools from traditional multigrid (e.g., the choice of smoother) are not appropriate. Although this paper often makes reference to the unconstrained problem (2.1), we want our results to apply also to constrained problems. As a result, the more complicated character of the reduced Hessian underlies many of our choices and approaches.

The problem (2.1) represents a family of optimization problems, each corresponding to a particular instance h of the optimization model F_h . We will solve the problem using the multilevel algorithm ML/Opt described below.

When traditional multigrid algorithms are applied to PDEs, it is common to use terms such as “grid”, “coarse”, “fine”, “frequency”, etc. This is a convenient shorthand used to refer to the various sub-problems. Our optimization-based multilevel algorithm is designed to be applicable to more general multi-level optimization problems where the individual models may have higher or lower fidelity or resolution, but do not necessarily correspond to discretizations or to PDEs. The algorithm has only limited knowledge of the optimization problem and the meaning of the variables. In particular, when we use our algorithm ML/Opt with the TN truncated-Newton method as the underlying optimization algorithm, the software has the following information available:

- A procedure to compute the objective function and its gradient for given values of the design variables on any of the sub-problems.
- The dimensions of the vectors of design variables on the various sub-problems.
- Update and downdate procedures to transform vectors from one sub-problem to another.

ML/Opt does not know what the various sub-problems represent, nor does it know what the design variables represent.

For convenience, we will describe the sub-problems as being representations of a particular optimization problem on a given “level”. The downdate procedure will move from a “fine level” to a “coarse level”, and the update procedure will do the reverse. We will describe the fine level as representing a “finer resolution” capable of representing the solution at “higher fidelity”. And so on. For example, the various sub-problems could correspond to a network with various degrees of aggregation, as is appropriate for certain VLSI problems [15].

We assume throughout that the user-provided information is correct. More specifically, we assume that the software to evaluate the objective function, the constraints, and their first derivatives, has been correctly programmed, and that the update and downdate operators are correct and effective (i.e., that the output is a good approximation to the input from the other level). Our diagnostic tests are intended to augment basic error checking done by many software systems.

2.1. The Multilevel Algorithm. The algorithm ML/Opt given here is a realization of the more general algorithmic framework in [14], and corresponds to the algorithm used in the computational tests in that paper. We apply ML/Opt to the problem (2.1).

Given coarser and finer level parameters H and h , respectively, let I_H^h denote a prolongation operator that transfers information from the coarser level to the finer level, and let I_h^H denote a restriction operator that transfers information from the finer level to the coarser level. We make the standard assumption that

$$I_H^h = C_I \times (I_h^H)^T .$$

for some constant C_I .

The algorithm is built upon the truncated-Newton optimization algorithm TN / TNBC [21, 24], which in turn uses a conjugate-gradient (CG) algorithm to compute a search direction.

Here is a description of the ML/Opt algorithm. A call to the function corresponds to one iteration of the multilevel algorithm, and takes a step from $a_h^{(k)}$, the current estimate of the solution on the finest level, to $a_h^{(k+1)}$. At the outermost level the lower and upper bounds on the variables are set to $\pm\infty$, and the shift v_h on the objective function is set to zero. (The algorithm does not assume that the initial bounds are $\pm\infty$, but for the problem (2.1) this is the appropriate choice.) The algorithm calls $\text{TN}(a_0, a_{low}, a_{up}, v, it_{max})$ which minimizes $\tilde{F}(a) \equiv F(a) - v^T a$ subject to the bounds $a_{low} \leq a \leq a_{up}$ with initial guess a_0 ; the optional parameter it_{max} specifies an upper bound on the number of iterations of the TN algorithm.

- Initialize: Set $v_h \leftarrow 0$, $a_{h,low} \leftarrow -\infty$, $a_{h,up} \leftarrow \infty$. Specify an initial guess of the solution $a_h^{(0)}$. Set $k \leftarrow 0$. Specify it_{max} .
- While (not converged)
 - Compute $[a_h^{(k+1)}] \leftarrow \text{ML/Opt}(a_h^{(k)}, a_{h,low}, a_{h,up}, v_h)$
 - Set $k \leftarrow k + 1$
- end

The function ML/Opt is defined as follows.

- function $[a_h^{(1)}] \leftarrow \text{ML/Opt}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h)$
- If on coarsest level, compute $a_h^{(1)} \leftarrow \text{TN}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h)$.
 - Partially minimize: compute $a_{h,1} \leftarrow \text{TN}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h, it_{max})$.
Downdate the result to obtain $a_{H,1} \leftarrow I_h^H a_{h,1}$.
 - Compute $v_H \leftarrow \nabla F_H(a_{H,1}) - I_h^H \nabla F_h(a_{h,1})$.
 - Apply the multilevel recursion: $a_{H,2} \leftarrow \text{ML/Opt}(a_{H,1}, a_{H,low}, a_{H,up}, v_H)$ which solves

$$\underset{a_H}{\text{minimize}} \tilde{F}_H(a_H) \equiv F_H(a_H) - v_H^T a_H$$

subject to the bound constraints

$$a_{H,low} \leq a_H \leq a_{H,up}.$$

(See below for a definition of the bounds.)

- Compute the search direction $e_h \leftarrow I_H^h(a_{H,2} - a_{H,1})$.
- Use a line search to obtain $a_{h,2} \leftarrow a_{h,1} + \alpha e_h$.
- Partially minimize: compute $a_h^{(1)} \leftarrow \text{TN}(a_{h,2}, a_{h,low}, a_{h,up}, v_h, it_{max})$.

Algorithm ML/Opt is a multilevel algorithm with a V-cycle template for traversing the levels. Other templates could be used by making simple modifications to the recurrence. The algorithm is initialized with a specified estimate on the finest level. An alternative approach is to use a full multigrid initialization scheme (see, for example, [20]).

Under appropriate assumptions, ML/Opt is guaranteed to converge to a stationary point of the optimization problem [15, 23]. Also, if the multilevel recursion reduces the value of the coarse model, the search direction e_h at each iteration will be a descent direction, ensuring that the estimate of the solution improves at every iteration of the multilevel algorithm. The line search used to obtain $a_{h,2} \leftarrow a_{h,1} + \alpha e_h$ is the same as the line search in algorithm TN.

The idea behind the convergence guarantees for ML/Opt is straightforward. On the finest level the optimization algorithm TN is guaranteed to converge to a local solution under standard assumptions. In the worst case, the multilevel recursion will fail to identify an improved estimate of the solution on the finest level and the line search will set $a_{h,2} \leftarrow a_{h,1}$. In this situation the multilevel recursion will waste computational effort but will not spoil the convergence guarantees of TN on the finest level.

The recursion requires ML/Opt to solve a shifted version of the optimization problem subject to the bound constraints. The bounds used are the same as those in [14], namely

$$\begin{aligned} a_{H,low} &= a_{H,1} - \gamma e \\ a_{H,up} &= a_{H,1} + \gamma e \end{aligned}$$

where

$$\begin{aligned} e &= (1, \dots, 1)^T \\ \gamma &= \max\{\|v_H\|, \|\nabla F_H(a_{H,1})\|, \|I_h^H \nabla F_h(a_{h,1})\|\}. \end{aligned}$$

The shifted problem on the coarse level is a first-order approximation to the optimization problem on the fine level, and thus will only be accurate in a neighborhood of the point $a_{H,1}$. The bounds restrict the algorithm to such a neighborhood, analogous to a trust-region approach for optimization [8].

2.2. The Reduced Hessian. When multigrid is applied to a PDE, its effectiveness is determined by the properties of the differential operator. When our multilevel algorithm is applied to the optimization problem (2.1), its effectiveness is determined by the properties of the Hessian for this problem or, for the special case (2.2)–(2.3), by the properties of the *reduced Hessian*. It is far more challenging to analyze the reduced Hessian than it is to analyze the differential operator. The goal of this section is to explain those challenges. There are three aspects: (a) the formula for the reduced Hessian is complicated, (b) it may not be possible to determine the components of this formula, since they depend on the inverse operator for the (perhaps nonlinear) state equation (2.3), (c) the properties of the reduced Hessian may be dramatically different from those of the state equation (2.3), and hence contrary to intuition and expectation. The discussion here is condensed from [16].

The Hessian of the Lagrangian for (2.2)–(2.3) is

$$\nabla_{(a,u)}^2 L(a, u; \lambda) = \nabla_{(a,u)}^2 f(a, u) + \nabla_{(a,u)}^2 S(a, u) \lambda.$$

The Hessian of L with respect to both a and u has the block structure

$$\nabla^2 L = \begin{pmatrix} M_{aa} & M_{au} \\ M_{ua} & M_{uu} \end{pmatrix}.$$

We then have the following expression for the Hessian of F with respect to a [16]:

$$\nabla^2 F = M_{aa} + M_{au} S_u^{-1} S_a + S_a^* S_u^{-*} M_{ua} + S_a^* S_u^{-*} M_{uu} S_u^{-1} S_a,$$

where S_a and S_u are the derivatives of S with respect to a and u , respectively. In this formula, $*$ denotes the adjoint of an operator. This is the *reduced Hessian* [25] of $f(a, u)$ with respect to the equality constraints $S(a, u) = 0$.

When applying multigrid to the solution of Poisson’s equation, for example, we have direct access to the representation A_h of the Laplacian on any grid. For our optimization problem, on the other hand, we do not directly have $\nabla^2 F(a)$ at our disposal.

Clearly the Hessian has a complicated structure. A further complication is that the reduced Hessian depends on S_u^{-1} , i.e., on the inverse of a Jacobian of the state equation. If the state equation is nonlinear it may be impossible to derive a formula for S_u^{-1} , and hence challenging to analyze the properties of the reduced Hessian.

The final complication is that the constraint operator $S(a, u)$ and the reduced Hessian can have qualitatively different properties. In [16] we study a simple problem with a linear differential equation

$$u_t + cu_x = 0$$

and a least-squares objective function. In this example, the constraint is a *hyperbolic* differential equation, and is not well-suited to multigrid. However the reduced Hessian for the problem is an *elliptic* operator and is similar to the Laplacian, which is an ideal operator for multigrid. Thus, even for simple problems, it may be challenging to guess if the optimization problem is well suited to our multilevel algorithm.

On more general optimization problems additional difficulties can arise. The optimization algorithms ML/Opt and TN do not explicitly form the Hessian or reduced Hessian, and hence do not have knowledge of its coefficients. Further, for a problem of the form (2.2)–(2.3) there is no guarantee that the reduced Hessian will be sparse. In summary, the approaches commonly used to analyze multilevel methods for PDEs may not be appropriate in the context of an optimization-based multilevel algorithm.

3. The Diagnostic Tests. The performance of ML/Opt depends on four properties of the optimization problem. As our subsequent analysis will show, if these properties are satisfied, we can expect that ML/Opt will display various aspects of ideal performance, as enumerated in the previous section. Our assessment techniques will attempt to identify if these properties are satisfied, and hence attempt to diagnose the performance of ML/Opt as it is solving a particular optimization problem.

1. *Nonlinearity*—ML/Opt relies on an underlying optimization algorithm, in this case the truncated-Newton method TN. TN, like many optimization algorithms, is based on a quadratic approximation to the optimization problem derived from the Taylor series. If this is not a good approximation to the problem, i.e., if there are significant higher-order terms (“nonlinearities”), then it will be more difficult for ML/Opt to find the solution to the optimization problem.
2. *Consistency*—The heuristic idea behind the multilevel algorithm is to use computations on coarse problems to improve the solution on fine problems. This is only sensible if the optimization problems are all approximations to the same underlying problem. If this is not true, then ML/Opt will not work effectively. For example, it is possible to over-coarsen the problems, choosing a model so coarse that the solution of this problem does not provide useful information about the finer problems; or it is possible to choose a coarse-level model that is qualitatively different than the fine-level model.
3. *Complementarity*—ML/Opt is designed with the hope that the smoother (that is, the underlying optimization method) will complement the multilevel recursion. That is, that the components of the solution resolved by the smoother will be distinct from the components resolved by the recursion. (In

traditional multigrid the smoother resolves high-frequency components and the recursion low-frequency components.) This is only possible if the coarse models are chosen so they are complementary to the fine models, i.e., so they represent “algebraically smooth” components of the solution (components that are not effectively resolved by the smoother).

4. *Separability*—If the multilevel algorithm is to be effective, the components of the solution resolved on the fine model must not be significantly influenced by the components of the solution on the coarse model. For this to happen, the reduced Hessian must be approximately separable across model levels. For example, if the reduced Hessian were equal to the Laplacian (an ideal operator for traditional multigrid), then the reduced Hessian would be separable—in fact, diagonal—in the Fourier (frequency) basis.

Our goal is to identify diagnostic tests that can be used to determine if the problem has the properties that would make it a good candidate for ML/Opt. The diagnostic tests will be required to either:

- Use information normally computed by the algorithm.
- Use information about the optimization problem that can be deduced from information provided by the user (update and downdate procedures, function and gradient procedures) with limited amounts of auxiliary computation.

Here “limited” means that the cost of the diagnostic tests must be small compared with the cost of optimization.

It would be reasonable to consider diagnostic tests that were more expensive but that were only invoked at the request of the user, but we do not discuss such tests here. We think it likely that, if ML/Opt discovered evidence suggesting that a particular property was not satisfied, the user would be better able to investigate further using information about the problem that is not provided to ML/Opt. For example, the user might be willing to compute and analyze the reduced Hessian of the optimization problem, at least for selected values of the design variables.

Because the tests that we consider are simple, and use such limited information about the optimization problem, they are not definitive. They may misdiagnose properties of the problem, giving either false positives or false negatives. By choosing tolerances appropriately, they could be made more or less sensitive.

Despite these limitations, it is our hope that the diagnostic tests would be able to automatically identify properties of the optimization problem that influence the behavior of ML/Opt, and provide useful information that could guide a more thorough investigation of the properties of the optimization problem.

The following subsections discuss the diagnostic tests associated with each of the above properties.

3.1. Nonlinearity. Most standard optimization methods are based on a Taylor series approximation. If this is not a good approximation to the nonlinear function, that is, if the higher order terms in the series are significant, then the optimization method may not perform effectively. In addition, if these higher-order terms are significant, the other tests that we describe may not be reliable. Thus, a user may choose not to perform the other tests under these circumstances.

The Taylor series approximation is an approximation to

$$F_h(a + p)$$

for some value of a , typically the current estimate of the solution. The optimization method will return a value of p that will be used as a search direction in the algorithm.

We will be concerned with the quality of the Taylor series approximation for $F_h(a+\alpha p)$ when $0 \leq \alpha \leq 1$. The difference between the Taylor series and $F_h(a + \alpha p)$ will be referred to as the “nonlinearity” in the problem.

In our TN optimization algorithm a quadratic Taylor series approximation is used. This is already a nonlinear problem, which may make the term “nonlinearity” confusing. In fact, the term will refer to nonlinearity of the *first-order optimality conditions*. For a quadratic approximation $Q(a)$, the optimality condition is $\nabla Q(a) = 0$, and if Q is quadratic then this condition is a linear equation. This is consistent with the terminology when multigrid is applied to PDEs. The solution to a linear elliptic PDE (i.e., a linear equation with a positive-semi-definite operator) is equivalent to the minimization of a quadratic function involving the same operator.

In general, it will not be possible to measure the magnitude of the nonlinearity directly, particularly given the limited information available to the algorithm. If $\|p\|$ is small, then the higher-order terms may well be small, since as $\|p\|$ goes to zero the higher-order terms become negligible. But if $\|p\|$ is not small, we may be able to test for nonlinearity indirectly via the line search in the optimization algorithm.

The TN algorithm, like Newton’s method, is based on a quadratic Taylor series approximation to $F_h(a + p)$. The TN optimization algorithm computes the search direction p using a conjugate-gradient (CG) method using a finite-difference approximation to $\nabla^2 F_h$ for the required matrix-vector products. If k iterations of the CG method are performed, and if the CG method does not detect degeneracy or indefiniteness, then p will be a solution to

$$\underset{p=P_i v_i}{\text{minimize}} F_h(a) + p^T \nabla F_h(a) + \frac{1}{2} p^T \nabla^2 F_h(a) p,$$

where P_i is a basis for the i -th Krylov subspace generated by the CG algorithm:

$$\{-\nabla F_h(a), -[\nabla^2 F_h(a)]\nabla F_h(a), \dots, -[\nabla^2 F_h(a)]^{i-1}\nabla F_h(a)\}.$$

Thus p is of the form

$$p = -P_i [P_i^T \nabla^2 F_h(a) P_i]^{-1} P_i^T \nabla F_h(a)$$

for some i .

A line search is then performed to determine the new estimate of the solution: $a_+ \leftarrow a + \alpha p$, where α is an approximate solution to

$$\underset{\alpha}{\text{minimize}} s(\alpha) \equiv F_h(a + \alpha p).$$

If $F_h(a)$ were a quadratic function, then p would minimize $F_h(a + p)$ over some Krylov subspace, and $\alpha = 1$ would minimize $s(\alpha)$. If the Taylor series is a good approximation to $F_h(a + p)$, then we would expect that $\alpha = 1$ would approximately minimize $s(\alpha)$, and hence that $s'(1) \approx 0$. If we expand $s(\alpha)$ in a Taylor series about $\alpha = 0$, and substitute the above formula for p for any value of $i > 0$, we obtain

$$s'(1) = O(\|p\|^3).$$

That is, $s'(1)$ is the value of the higher-order terms in the Taylor series for this particular p , and hence is a measure of nonlinearity. For additional details, see [22]. As the solution is approached and $\|p\| \rightarrow 0$, a step of $\alpha = 1$ is likely to be accepted

in the line search. Whenever $\alpha = 1$ is an acceptable search direction, we will say that the search direction p is *well scaled*.

Since TN is likely to produce well-scaled search directions when the nonlinearities are not severe, we can use the step length as a diagnostic for nonlinearity:

- *Nonlinearity Test*—Is $|s'(1)| \leq \text{tolerance}$?

If this test is satisfied, then we will assume that the nonlinearities are relatively benign, and that it will be appropriate to consider relying on the other tests discussed in the article. The nonlinearity test is analogous to the “over solving” test from [10], although that test was developed for a different purpose.

A simpler test could be based on whether the line search used a step length equal to one, and this might be desirable if the line search algorithm did not use derivative values to determine the step length.

The nonlinearity test here applies to the optimization problem (2.1), or to (2.2)–(2.3) where the state equation (2.3) is solved for u given a . In more general settings there might be additional constraints on the variables, or the state equation might only be solved approximately. In such settings the nonlinearity test would have to be modified to take these changes into account.

3.2. Consistency. Because computations on the coarser problems are typically much cheaper than computations on finer problems, it is tempting to use coarser problems extensively. This is only reasonable if the optimization models on the various levels are good approximations to each other. For example, they might all be discrete approximations to a single continuous problem. In this section we develop a test to determine if the optimization problems are indeed consistent, in the sense that they are all good approximations to each other.

Problems can be inconsistent even if they are all approximations to the same underlying continuous problem. For example, if too coarse a model is chosen, the corresponding optimization problem will be a poor approximation to the original problem, and the results of these computations will be of little value. So, while cheap, they may be almost useless. This is at the least a waste of computational effort, and may possibly hinder the overall algorithm by suggesting confounding search directions. How do we determine if the optimization problems are consistent?

Our test will compare problems on two successive levels. Let us assume that the problem on the coarser level is sensible as an optimization problem, i.e., that it has a solution and that its derivatives have the necessary smoothness. Then the optimization algorithm applied to this problem will be effective, and will work in an unremarkable fashion. Thus, any test of consistency will focus on the behavior of the multilevel recursion.

If the problems are not consistent, then the coarse solution $a_{H,2}$ will not produce a good search direction e_h for the fine problem. What does “not a good search direction” mean? We are using the coarse problem to identify a search direction for the fine-level problem. We obtain a search direction by applying an optimization method to the coarse problem. In other words, the optimization method produces a reduction in the coarse problem. Since the coarse problem is assumed to be an approximation to the fine problem, we would hope that the reduction *predicted* by the coarse problem would be approximately equal to the *actual* reduction in the fine problem obtained via the line search.

Before stating the test, it will be helpful to derive approximations for the actual and predicted reductions based on

$$\langle \text{predicted reduction} \rangle \equiv R_p = \tilde{F}_H(a_{H,1}) - \tilde{F}_H(a_{H,2})$$

$$\langle \text{actual reduction} \rangle \equiv R_a = F_h(a_{h,1}) - F_h(a_{h,2}).$$

(For the definition of \tilde{F} , see the description of algorithm ML/Opt in Section 2.1.) The formula for the actual reduction assumes that the multilevel line search uses a step length of $\alpha = 1$ in the multilevel line search. The diagnostic test can be performed even if the line search does not accept $\alpha = 1$. There are theoretical reasons to expect that the search direction from the multilevel recursion will be well scaled (see [16] and Section 3.1), at least as the algorithm approaches the solution.

Taylor series expansions give

$$\begin{aligned} -R_p &= \tilde{F}_H(a_{H,2}) - \tilde{F}_H(a_{H,1}) \\ &= F_H(a_{H,1} + e_H) - F_H(a_{H,1}) - v_H^T a_{H,1} - v_H^T e_H + v_H^T a_{H,1} \\ &= F_H(a_{H,1} + e_H) - F_H(a_{H,1}) - e_H^T \nabla F_H(a_{H,1}) + e_H^T I_H^H \nabla F_h(a_{h,1}) \\ &= e_H^T \nabla F_H(a_{H,1}) + \frac{1}{2} e_H^T \nabla^2 F_H(a_{H,1}) e_H + O(\|e_H\|^3) - e_H^T \nabla F_H(a_{H,1}) + e_H^T I_H^H \nabla F_h(a_{h,1}) \\ &= e_H^T I_H^H \nabla F_h(a_{h,1}) + \frac{1}{2} e_H^T \nabla^2 F_H(a_{H,1}) e_H + O(\|e_H\|^3), \end{aligned}$$

and

$$\begin{aligned} -R_a &= F_h(a_{h,1} + I_H^h e_H) - F_h(a_{h,1}) \\ &= (I_H^h e_H)^T \nabla F_h(a_{h,1}) + \frac{1}{2} (I_H^h e_H)^T \nabla^2 F_h(a_{h,1}) I_H^h e_H + O(\|I_H^h e_H\|^3) \\ &= e_H^T (I_H^h)^T \nabla F_h(a_{h,1}) + \frac{1}{2} e_H^T (I_H^h)^T \nabla^2 F_h(a_{h,1}) I_H^h e_H + O(\|e_H\|^3) \\ &= C_I \left[e_H^T I_h^H \nabla F_h(a_{h,1}) + \frac{1}{2} e_H^T I_h^H \nabla^2 F_h(a_{h,1}) I_H^h e_H + O(\|e_H\|^3) \right], \end{aligned}$$

where C_I is the constant that relates I_h^H and I_H^h (see section 2.1). Thus

$$C_I^{-1} R_a - R_p = \frac{1}{2} e_H^T \left[\nabla^2 F_H(a_{H,1}) - I_h^H \nabla^2 f^h(a_0^h) I_H^h \right] e_H + O(\|e_H\|^3).$$

This formula involves two terms. There is a second-order term involving the difference between $\langle \text{coarse Hessian} \rangle$ and $\langle \text{restricted fine Hessian} \rangle$. And there is a third-order term measuring “nonlinearity”. As discussed in Section 3.1, this is a generic issue for nonlinear optimization. We can use the nonlinearity test to determine if the third-order term is likely to be significant.

Assuming that the higher-order terms can be ignored, i.e., that the nonlinearity test is satisfied, the difference between predicted and scaled actual reduction can be used to measure the difference between the coarse Hessian and the restricted fine Hessian. If these two matrices are dissimilar, we cannot expect the coarse problem to be a useful approximation to the fine problem.

For these reasons, our test for consistency will be of the form

- *Consistency Test*—Is

$$\frac{|\langle \text{predicted reduction} \rangle - C_I^{-1} \langle \text{actual reduction} \rangle|}{|\langle \text{actual reduction} \rangle|} \leq \text{tolerance?}$$

Since in many applications this will be a test for over-coarsening, the test may only need to be applied on the coarsest levels.

It is possible that this test would fail for a simple reason that might be considered a programming error. Consider an optimization problem with objective function

$$F(a) = \int_0^T \int_0^1 (u(x, t) - \phi(x, t))^2 dx dt.$$

If we were to discretize in space with mesh size h_x , and in time with mesh size h_t , then we could use the following discrete approximation to the objective function:

$$F_h(a_h) = h_x h_t \sum_{i=0}^{N_x} \sum_{j=0}^{N_t} C_{i,j} (u_{i,j} - \phi_{i,j})^2$$

where the constants $C_{i,j}$ depend on the quadrature rules used to estimate the integrals.

From the point of view of optimization on a single level, the same solution would be obtained using the scaled objective function

$$\hat{F}_h(a_h) = \frac{1}{h_x h_t} F_h(a_h).$$

However, if ML/Opt is used, the corresponding coarse problems would be different. Consider a coarse problem with mesh sizes $H_x = 2h_x$ and $H_t = 2h_t$. Then $F_h(a_h) \approx F_H(I_h^H a_h)$ since they both are approximations to the same continuous problem. However $\hat{F}_h(a_h) \approx 4\hat{F}_H(I_h^H a_h)$. Since v_H in the coarse optimization problem in ML/Opt mixes quantities derived from both the fine and coarse levels, ML/Opt would perform differently on F_h than on \hat{F}_h .

When using ML/Opt it is important that the optimization problems on the various levels be *level invariant*, i.e., that they are all approximations to the same underlying problem. It is easy to overlook this issue. Typically, an optimization problem would first be tested on a single level using a standard optimization method. The optimization method would behave much the same way on both F_h and \hat{F}_h , except perhaps for rounding errors. If ML/Opt is then applied, it may well solve the problem, but not with the efficiency expected of a multilevel algorithm, and the cause of the inferior performance might be difficult to detect.

Our diagnostic test compares predicted and actual reduction, and would reveal that the optimization problems were not level invariant, but that requires completing a multilevel recursion before using the test. It makes sense to use a simple test for model invariance before using ML/Opt:

- *Model Invariance Test*—Is $|F_h(a_h) - F_H(I_h^H a_h)| \leq \text{tolerance}$?

This test would only need to be performed during the first iteration of ML/Opt.

3.3. Complementarity. ML/Opt is designed with the hope that the smoother (that is, the underlying optimization method) will complement the multilevel recursion. That is, that the components of the solution resolved by the smoother will be distinct from the components resolved by the recursion. This is only possible if the coarse models are chosen so they are complementary to the fine models, i.e., so they represent “algebraically smooth” components of the solution (components that are not effectively resolved by the smoother). In a sense, our diagnostic test will attempt to determine if the coarse level corresponds to algebraically smooth components of the solution.

Multigrid methods were first developed based on geometric principles, for example, the coarsening of a discretization of a simple region like a line segment. In such a

setting, it is typically straightforward to select the coarser grids. Subsequent research has led to the development of algebraic multigrid methods which attempt to identify the coarse grids automatically, so problems can be solved by a multigrid method without any effort or understanding on the part of the user. Much of this research is focused on linear equations arising from the discretization of partial differential equations. The coefficient matrices are sparse, and many algebraic multigrid methods are defined using knowledge of the sparsity pattern and the entries of the coefficient matrix.

In our setting, the relevant coefficient matrix is the (reduced) Hessian. Typically we will not know its entries nor its sparsity pattern. It may even be a dense matrix, even when the components of the reduced Hessian are sparse. For these reasons we base our diagnostic test on some general principles of algebraic multigrid, rather than specific algebraic multigrid methods.

We refer only to the choice of models, but the choice of the interpolants I_h^H and I_H^h can also affect the algorithm’s ability to approximate algebraically smooth components of the solution [4]. Because ML/Opt has such limited information available, it is not clear that ML/Opt can isolate the properties of the interpolant from the properties of the coarse model. For this reason we conflate the two issues, and refer only to the choice of the coarse model when discussing the diagnostic test.

In ML/Opt we use TN as an underlying optimization method, which is in turn based on the conjugate-gradient (CG) method. To simplify the explanation, let us assume that we are working with a quadratic optimization problem and that the reduced Hessian is positive definite.

It is well known that CG typically reduces the error in components of the solution corresponding to large eigenvalues of the coefficient matrix [11]. Then the complementary components of the solution will be those corresponding to the small eigenvalues. In the context of algebraic multigrid, such components are referred to as the “near null space” of the coefficient matrix because they correspond to small (near zero) eigenvalues [3]. (CG is not typically used within traditional multigrid methods, but the same reasoning applies.) One way to measure whether the coarse model is appropriate is to compute a generalized Rayleigh quotient for the error η_h in the variables a_h :

$$RQ_h(\eta_h) = \frac{(G^h \eta_h)^T (G^h \eta_h)}{\|G^h\|_2 (G^h \eta_h)^T \eta_h},$$

where G^h is the (reduced) Hessian of the optimization problem. In the context of algebraic multigrid the coefficient matrix is known and, if the algorithm is applied to a linear system with zero right-hand side, the current estimate of the solution is also the error. In that setting this is a practical formula. For ML/Opt neither η_h nor G^h is available.

There are two steps to obtaining a practical test. The first is to replace the error η_h with the search direction e_h from the multilevel recursion. If ML/Opt is working effectively, the multilevel recursion should produce an accurate approximation to the components of the solution in the near null space. If that is the case, then the search direction e_h should be an accurate approximation to the error η_h . We can then approximate $G^h \eta_h$ with the matrix-vector product $G^h e_h$, which can be estimated by finite-differencing of gradient values (a technique already used by TN in its CG algorithm).

That leaves $\|G^h\|_2 = \rho(G^h)$, the largest eigenvalue of G^h . This can be estimated

as a by-product of the CG method. The CG method is equivalent to the Lanczos method for computing eigenvalues [27]. The Lanczos method iteratively computes a tridiagonal matrix T^h whose eigenvalues approximate those of the coefficient matrix G^h . Typically the largest eigenvalues are the first to converge. It is possible to use quantities from the CG method to construct this tridiagonal matrix. We can use $\rho(T^h) = \|T^h\|_2$ as an approximation to $\|G^h\|_2$. (We will not compute the full matrix T^h , but just use the tridiagonal matrix generated at the final iteration of the CG method as it computes a search direction for TN.)

Using these approximations, our test will be based on the following approximate Rayleigh quotient:

$$\hat{R}_h(e_h) = \frac{(G^h e_h)^T (G^h e_h)}{\rho(T^h) (G^h e_h)^T e_h}.$$

Thus our diagnostic test for the appropriateness of the coarse model will be

- *Complementarity Test*—Is $|\hat{R}(e_h)| \leq \text{tolerance}$?

Within a traditional algebraic multigrid method, it is possible to use techniques such as compatible relaxation to determine an appropriate coarse grid and interpolant for use within a multigrid method [3]. ML/Opt is not provided with enough information about the optimization problems to do this. However, the user might be able to take advantage of these techniques if the complementarity test were not satisfied.

3.4. Separability. The multilevel algorithm will be an effective strategy if the components of the solution resolved by the smoother on the fine model are not significantly influenced by the components of the solution determined by the multilevel recursion on the coarse model. For this to happen, the reduced Hessian must be approximately separable across model levels. If we write the problem in a basis that isolates the coarse-model components of the solution, then the reduced Hessian for the optimization problem will have the following block form

$$G^h = \begin{pmatrix} G_{hh} & G_{Hh}^T \\ G_{Hh} & G_{HH} \end{pmatrix}$$

where H and h are used to designate coarse- and fine-level components, respectively. (If the optimization model corresponded to a PDE then these would be low- and high-frequency components.) For the multilevel algorithm to be successful, the off-diagonal block must be small compared to the diagonal blocks. Ideally, the off-diagonal block will be zero. Our goal is to develop a practical test to determine if the off-diagonal block of the reduced Hessian is small. We call this a test for *separability*, i.e., separability of high- and low-level components of the solution.

This is challenging for several reasons. First, the ML/Opt algorithm only uses function and gradient values from the optimization problem. It does not have Hessian information available, except in the sense that it approximates Hessian-vector products using finite differences of gradient values. Second, for constrained problems we are asking for information about the reduced Hessian, and the formulas for the reduced Hessian are complicated and involve the inverse operator for the linearized state equation (see Section 2.2). Third, the reduced Hessian is not likely to be represented in a basis that isolates high and low levels, such as a Fourier basis in the case of a PDE.

To determine separability, we would like to know if the off-diagonal block $G_{Hh} \approx 0$, when the reduced Hessian is represented in a basis that separates high- and low-level components. This is a challenging analysis even for simple problems where all

the component operators are available explicitly. How then can ML/Opt with its limited problem information have any hope of assessing this property?

Suppose that we were to use update and downdate operators I_H^h and I_h^H that perfectly isolated low- and high-level components, and consider a fine-level vector z^h with fine-level and coarse-level components z_h and z_H , respectively. Then

$$I_h^H z^h = I_h^H \begin{pmatrix} z_h \\ z_H \end{pmatrix} = z_H \quad \text{and} \quad I_H^h z_H = \begin{pmatrix} 0 \\ z_H \end{pmatrix}.$$

Using these operators, a test for separability can be obtained using a Hessian-vector product for an appropriately chosen vector. Hessian-vector products are estimated by TN as a step in the computation of the TN search direction, so this calculation can be performed using the information normally available to ML/Opt.

Let $z^h = (z_h, 0)^T$ with $z_h \neq 0$, i.e., z^h consists only of high-level components. Such a vector can be obtained via

$$z^h = w^h - I_H^h I_h^H w^h$$

for some arbitrary initial vector $w^h = (w_h, w_H)^T$. Then we compute

$$I_h^H G^h z^h = I_h^H \begin{pmatrix} G_{hh} z_h \\ G_{Hh} z_h \end{pmatrix} = G_{Hh} z_h.$$

Thus, a downdated Hessian-vector product of this form is a measure of separability.

Unfortunately, this test is not useful in its raw form because the update and downdate operators need not perfectly isolate low- and high-level components. They may be based on interpolation schemes that cause some mixing of these components. (This issue is illustrated in the computational tests in Section 6.)

Nevertheless, it is possible to refine this idea to produce a useful test. While the downdate operator provided by the user is unlikely to isolate low-level components precisely, it should be an approximate low-pass filter. That is, high-level components should be damped, and low-level components should undergo little change. If we begin with some random vector w^h on the fine level, then repeated application of the downdate and update operators

$$\hat{w}^h = (I_H^h I_h^H)^k w^h$$

should result in a vector \hat{w}^h that is dominated by low-level terms. Then $z^h = w^h - \hat{w}^h$ should be dominated by high-level terms. Likewise, we must process the resulting Hessian-vector product $G^h z^h$ with $(I_H^h I_h^H)^k$ to isolate the low-level components.

One additional point deserves comment. The test below uses $\|G^H\|$. An estimate for the norm of the Hessian is computed by the complementarity test (see Section 3.3) and is used here. We divide by $\|G^H\|$ and not $\|G^h\|$ since we are trying to assess the relative magnitude of the diagonal and off-diagonal blocks.

In summary, here is the test for separability:

1. Generate a random vector w^h on the fine model. Let $\bar{w}^h = w^h$.
2. Iterate k_1 times: $\bar{w}^h = I_H^h I_h^H \bar{w}^h$ [this vector is dominated by low-level terms]
3. Define $z^h = w^h - \bar{w}^h$. [this vector is dominated by high-level terms]
4. Compute the Hessian-vector product: $(Gz)^h = G^h z^h$. Let $(Gz)^H = I_h^H (Gz)^h$
5. Iterate k_2 times: $(Gz)^H = I_h^H I_H^h (Gz)^H$ [this vector is dominated by low-level terms]

6. Test for separability:

- *Separability Test*—Is $\|(Gz)^H\|/(\|z^h\| \|G^H\|) \leq \text{tolerance}$?

This test only uses information available to ML/Opt. The software does not know what “coarse”, “fine”, “high level”, and “low level” mean. Instead it uses the update and downdate procedures as a surrogate. If this test revealed that the problem might not be separable across levels, it might be possible to ask the user to provide a more precise analysis (e.g., using a Fourier transform) to confirm the diagnosis.

If the reduced Hessian is positive definite, then there is some expectation that the separability test will be automatically satisfied. This is a consequence of the results in [18]. Consider two symmetric matrices

$$A = \begin{pmatrix} M & R \\ R^T & N \end{pmatrix} \quad \text{and} \quad \tilde{A} = \begin{pmatrix} M & 0 \\ 0 & N \end{pmatrix}.$$

For any matrix A , let $\{\lambda_k(A)\}$ be the set of eigenvalues A ordered from smallest to largest, and let $\sigma(A)$ be the spectrum of A .

Using this notation, Theorem 1 in [18] states (in part): If $\lambda_1(A) \notin \sigma(N)$, then

$$|\lambda_1(A) - \lambda_1(\tilde{A})| \leq \frac{\|R\|_2^2}{\min_i |\lambda_1(A) - \lambda_i(N)|}.$$

We will apply this theorem with $M = G_{hh}$, $N = G_{HH}$, and $R = G_{hH}$. Hence $A = G^h$ and

$$\tilde{A} = \tilde{G}^h = \begin{pmatrix} G_{hh} & 0 \\ 0 & G_{HH} \end{pmatrix}.$$

We will assume that \tilde{G}^h is positive definite. (If \tilde{G}^h has a negative eigenvalue then so does G^h .) In applying the theorem, our main concern is whether G^h remains positive definite as $\|G_{hH}\|$ increases. Thus, we assume that $\min_i |\lambda_1(G^h) - \lambda_i(G_{HH})| = |\lambda_1(G^h) - \lambda_1(G_{HH})|$.

Then to guarantee that G^h is positive definite, the theorem implies that it is sufficient that $\|G_{hH}\| < \lambda_1(G_{HH})$. If the complementarity test is satisfied then the smallest eigenvalue of G_{HH} will be small. Thus, if G^h is positive definite, then $\|G_{hH}\|$ is expected to be small and G^h will be nearly separable.

This suggests that lack of separability may be a more significant issue at points where the Hessian is not positive definite, typically at points far from the solution.

4. Sufficiency of the Diagnostic Tests. Our goal in this section is to demonstrate that our diagnostic tests address the relevant issues for the performance of ML/Opt. We have identified four properties that are relevant: (a) nonlinearity, (b) consistency, (c) complementarity, and (d) separability.

Since ML/Opt is based on quadratic approximations to the optimization problem, the performance of ML/Opt will deteriorate if there is significant nonlinearity, since in that case the quadratic problems will be poor approximations to the optimization problem. For this reason, the nonlinearity test is clearly necessary in assessing the performance of ML/Opt.

To assess the importance of the other tests, we will assume now that the nonlinearity test is satisfied, and that the optimization problem (expressed in terms of the Hessian or reduced Hessian) can be approximated effectively by a quadratic problem

$$\underset{a_h}{\text{minimize}} F_h(a_h) = \frac{1}{2} a_h^T G^h a_h - b_h^T a_h$$

where b_h is some vector. We assume that the optimization problem is well defined, i.e., that G^h is a positive-definite matrix. This is not an unreasonable assumption. The (reduced) Hessian is guaranteed to be positive-semi-definite at a local minimizer. Also, standard convergence theory assumes that the reduced Hessian will be positive definite at a local minimizer; without this assumption we will not be able to guarantee that either ML/Opt or TN converges to a local solution of (2.1).

In analyzing ML/Opt we will only refer to two levels and accept that only that an approximate solution is found to the coarse problem. Such an approximate solution could be obtained by applying the ML/Opt recursion. Thus our description in terms of only two levels applies to the case when more than two levels are used in ML/Opt, and is not a restrictive assumption.

On each level we will apply TN to the quadratic problem. This is equivalent to applying the CG method with restarts to the quadratic problem. It is well known that the CG method reduces the error as measured in the G^h -norm, and it is straightforward to show that the line search in TN further reduces the error in the G^h -norm. Asymptotically TN will also reduce the error in the 2-norm, although this norm of the error will not decrease monotonically.

We will write the problem in terms of fine-level and coarse-level components. In the ideal case $I_h^H = (0 \ I)$, but typically we will have

$$I_h^H = (0 \ I) + \Delta_I$$

for some matrix Δ_I where $\|\Delta_I\|$ will be small if the downdate operator is effective at identifying low-level components. (As discussed in Section 2, we have assumed that the user has done this.)

We will denote the solution by a^* , with fine- and coarse-level components a_h^* and a_H^* , respectively.

In terms of fine-level and coarse-level components, we have

$$G^h = \begin{pmatrix} G_{hh} & G_{hH} \\ G_{hH}^T & G_{HH} \end{pmatrix}$$

and the minimizer of the quadratic optimization problem is the solution to

$$G^h a_h = b_h \quad \text{or} \quad \begin{pmatrix} G_{hh} & G_{hH} \\ G_{hH}^T & G_{HH} \end{pmatrix} \begin{pmatrix} (a_h)_h \\ (a_h)_H \end{pmatrix} = \begin{pmatrix} (b_h)_h \\ (b_h)_H \end{pmatrix}.$$

Suppose that the diagnostic tests are all satisfied. Then an iteration of ML/Opt will behave as follows:

1. ML/Opt will apply TN to the fine level problem to obtain $a_h^{(1)}$, reducing the error in the G^h -norm.
2. In the multilevel recursion, ML/Opt will solve the coarse level problem. If the separability and consistency tests are satisfied, this will determine an approximation to $(a_h)_H$. If the complementarity test is satisfied, then this approximation will be in the near-null space of G^h , i.e., will represent components of the solution not well resolved by the TN method on the fine level.
3. ML/Opt will apply TN to the fine level problem, further reducing the error in the G^h -norm.

This gives a rough idea of why the diagnostic tests are sufficient to assess the performance of ML/Opt. We can make this argument more precise using perturbation analysis. The central issue will be how well ML/Opt approximates a_H^* via

the multilevel recursion. The structure of the ML/Opt algorithm ensures the overall convergence of the algorithm, with TN taking care of the fine-level components of a_h^* .

The value of a_H^* can be found by solving the linear system

$$(4.1) \quad (G_{HH} - G_{hH}^T G_{hh}^{-1} G_{hH}) a_H = (b_h)_H - G_{hH}^T G_{hh}^{-1} (b_h)_h.$$

We will demonstrate that our diagnostic tests are sufficient by showing that ML/Opt solves a perturbation of this equation, and derive bounds on the perturbation that are related to our diagnostic tests. To do this we find formulas for some of the terms that will arise in our discussion.

If we apply ML/Opt to the quadratic optimization problem, then the recursion will approximately solve a perturbed equation. We can derive a formula for that equation, involving a residual term η_H corresponding to the residual that results from the approximate solve. That is, we can determine that the multilevel recursion estimates a_H^* by solving for \hat{a}_H^* in the coarse-level equation

$$\begin{aligned} G^H \hat{a}_H &= b_H + v_H + \eta_H \\ &= b_H + [G^H (I_h^H a_h^{(1)}) - b_H] - [I_h^H (G^h a_h^{(1)} - b_h)] + \eta_H \\ &= G^H (a_h^{(1)})_H + G^H \Delta_I a_h^{(1)} - [(0 \quad I) + \Delta_I] (G^h a_h^{(1)} - b_h) + \eta_H \\ &= (b_h)_H - G_{hH}^T (a_h^{(1)})_h + (G^H - G_{HH}) (a_h^{(1)})_H \\ &\quad + G^H \Delta_I a_h^{(1)} - \Delta_I (G^h a_h^{(1)} - b_h) + \eta_H. \end{aligned}$$

We can assess the performance of ML/Opt by finding a bound for $\|a_H^* - \hat{a}_H^*\|$.

We can write the coarse-level equation as

$$G^H \hat{a}_H = \bar{b}_H$$

where

$$\begin{aligned} \bar{b}_H &= b_H - G_{hH}^T (a_h^{(1)})_h + (G^H - G_{HH}) (a_h^{(1)})_H \\ &\quad + G^H \Delta_I a_h^{(1)} - \Delta_I (G^h a_h^{(1)} - b_h) + [(b_h)_H - b_H] + \eta_H. \end{aligned}$$

Then we can write (4.1) as

$$(G^H + \delta G)(\hat{a}_H + \delta a) = \bar{b}_H + \delta b$$

where

$$\begin{aligned} \delta G &= (G_{HH} - G^H) - G_{hH}^T G_{hh}^{-1} G_{hH} \\ \delta b &= -G_{hH}^T G_{hh}^{-1} (b_h)_h + G_{hH}^T (a_h^{(1)})_h - (G^H - G_{HH}) (a_h^{(1)})_H \\ &\quad - G^H \Delta_I a_h^{(1)} + \Delta_I (G^h a_h^{(1)} - b_h) - \eta_H. \end{aligned}$$

Then standard perturbation analysis for linear equations [11] gives

$$\frac{\|\delta a\|}{\|\hat{a}_H^*\|} \leq \kappa(G^H) \left(\frac{\|\delta G\|}{\|G^H\|} + \frac{\|\delta b\|}{\|\bar{b}_H\|} \right) + (\text{higher-order terms}).$$

Here $\kappa(G^H)$ is the condition number of G^H : $\kappa(G^H) = \|G^H\| \cdot \|(G^H)^{-1}\|$.

It is straightforward to conclude that

$$\begin{aligned}
\|\delta G\| &\leq \|G_{hH}\|^2 \|G_{hh}^{-1}\| + \|G^H - G_{HH}\| \\
\|\delta b\| &\leq \|G_{hH}\| \cdot \|G_{hh}^{-1}\| \cdot \|(b_h)_h\| + \|G_{hH}\| \cdot \|(a_h^{(1)})_h\| \\
&\quad + \|G^H - G_{HH}\| \cdot \|(a_h^{(1)})_H\| + \|G^H \Delta_I a_h^{(1)}\| \\
&\quad + \|\Delta_I(G^h a_h^{(1)} - b_h)\| + \|\eta_H\|.
\end{aligned}$$

These terms will be small if our diagnostic tests are satisfied. If the consistency test is satisfied then $\|G^H - G_{HH}\|$ will be small. If the separability test is satisfied then $\|G_{hH}\|$ is small. If the update and downdate operators are chosen appropriately (a responsibility of the user) then $\|\Delta_I\|$ will be small. Also, G_{hh} will correspond to the large eigenvalues of G^h , so the multilevel recursion and TN methods will be complementary, and $\|G_{hh}^{-1}\|$ will be as small as possible. The quantities b_h , $(a_h^{(1)})_h$, and $(a_h^{(1)})_H$ are computed explicitly by ML/Opt, and can be monitored directly. Finally, the structure of ML/Opt will ensure that η_H will go to zero as the solution is approached.

Thus the performance of ML/Opt is governed by the properties associated with our diagnostic tests.

5. What to Do if the Diagnostic Tests are not Satisfied. We offer here guidance on how a user might respond if one or more of the diagnostic tests were not satisfied. It is not possible to be comprehensive since the appropriate response might require sophisticated understanding of the model. For example, it might require using an alternative approximation to a derivative term to remove an instability, or some other modification that is problem-specific.

Another difficulty is that some of the tests are influenced by more than one property of the optimization problem. For example, the complementarity test is influenced by the choice of the coarse-level model as well as by the update and downdate operators. Given the limited problem-related information provided to ML/Opt, it is not possible to use more focused approaches such as [4]. However, based on the guidance provided by our diagnostic tests, the user would be able to apply these other techniques using all available information about the optimization problem.

Finally, because of the simplicity of the diagnostic tests, there may be false positives, i.e., occasions where the diagnostic test fails but the optimization problem is satisfactory. A user may wish to use more sophisticated tests to confirm the diagnosis.

So, what should be done if a diagnostic test is violated? Assuming that the optimization problem is well defined, a simplistic response is always to just use a traditional optimization method on the finest level. Another option may be to do nothing. Assuming that the optimization problem satisfies the necessary assumptions, ML/Opt is guaranteed to converge to a local solution. Convergence may be slow, however.

In some cases it may be possible to provide more nuanced guidance. We discuss the diagnostic tests in turn.

If the nonlinearity test is violated then the optimization problem is not well approximated by the quadratic Taylor series approximation that is the foundation for the truncated-Newton method, and hence for ML/Opt. It is an indication that the algorithm is far from the solution, since near the solution the norm of the search direction will be small, and the higher-order terms in the Taylor series will be negligible. If the test only fails at a small number of early iterations, then no action may

be necessary. However, if ML/Opt is not able to make satisfactory progress toward the solution, it will be worthwhile to find a better estimate of the solution to use as an initial guess for the algorithm. That might mean using coarser models to develop an initial guess. Alternatively it might mean temporarily using an optimization method with less overhead, e.g., adjusting the truncated-Newton method to severely limit the number of conjugate-gradient iterations used to compute a search direction. This latter suggestion is offered because the truncated-Newton method is based on the Taylor series approximation and, if that is not useful, a lower-overhead algorithm may produce adequate search directions.

If the optimization problem has significant nonlinearities, then its behavior may change as the solution is approached. For example, it is possible that the problem may not satisfy the separability test far from the solution, but may have better properties near the solution. In discussing the remaining three diagnostic tests, we are assuming that the performance of ML/Opt is not ideal, and that the difficulties warrant further investigation by the user.

If the consistency test is violated at every level, then there may be an error in the software that defines the model. If it only occurs at coarse levels, then there might be over-coarsening. For example, the coarse models may have qualitatively different behavior than the fine models. We include a test problem of this type in section 6.3. To remedy this it may be sufficient to not use the coarse models in the multilevel recursion. Alternatively, it may be necessary to reformulate the coarse models.

If the separability test is violated, then there is significant interaction between the high-level and low-level components of the solution. If this is an inherent aspect of the optimization problem, then it may be inappropriate to use a multilevel algorithm. Alternatively, it may be possible to reformulate the problem or identify an appropriate preconditioner to handle this interaction.

If the complementarity test is violated then the smoother is not well matched with the choice of model levels. There may also be concerns about the update and downdate operators. This could be fixed by choosing an appropriate preconditioner, or by using techniques from algebraic multigrid [4].

6. Computational Tests. We apply the ML/Opt algorithm together with the diagnostic tests to a set of test problems. Some of the test problems were chosen to isolate the various properties that are of interest. Others were chosen to illustrate how the tests might perform on more realistic problems. We do not include in this paper detailed computational experiments for the nonlinearity test, since this topic is addressed in [10], for example. We focus on the remaining tests which are less familiar. The multilevel software used is an adaption of the software from [16] with the addition of the diagnostic tests. It is a Matlab implementation of the ML/Opt algorithm discussed in Section 2.1. In our computational tests, we use $it_{max} = 1$, i.e., “partially minimize” means apply one iteration of the TN optimization algorithm.

One of our goals in this Section is to demonstrate that the proposed diagnostic tests isolate the relevant properties of the optimization problem. For that reason, some of our test problems may seem simple, but they are useful because they isolate these properties. All of the test problems are based on discretizations, even though our work applies to more general multi-level problems. This is because discretized problems are simpler to describe. We do not attempt to perform experiments of a production-grade multilevel algorithm on realistic applications. That is beyond the scope and aims of this paper.

Before discussing the computational tests, we first present the revised ML/Opt

algorithm including the diagnostic tests.

6.1. Details of the Algorithm. Below is a revised description of the ML/Opt algorithm, indicating how the diagnostic tests are incorporated. As before, a call to the function corresponds to one iteration of the multilevel algorithm, and takes a step from $a_h^{(k)}$, the current estimate of the solution on the finest level, to $a_h^{(k+1)}$. At the outermost level the lower and upper bounds on the variables are set to $\pm\infty$, and the shift v_h on the objective function is set to zero. The function ML/Opt now computes $\hat{\rho}_H$, an estimate of $\|G^H\|$ on the next coarser model.

- Initialize: Set $v_h \leftarrow 0$, $a_{h,low} \leftarrow -\infty$, $a_{h,up} \leftarrow \infty$. Specify an initial guess of the solution $a_h^{(0)}$. Set $k \leftarrow 0$.
- While (not converged)
 - $[a_h^{(k+1)}, \hat{\rho}_H] \leftarrow \text{ML/Opt}(a_h^{(k)}, a_{h,low}, a_{h,up}, v_h)$
 - $k \leftarrow k + 1$
- end

The function ML/Opt is defined as follows.

- $[a_h^{(1)}, \hat{\rho}_h] \leftarrow \text{function ML/Opt}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h)$
- If on coarsest level, compute $a_h^{(1)} \leftarrow \text{TN}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h)$; compute $\hat{\rho}_h$, an estimate of the norm of the reduced Hessian $\|G^h\|$, for use in the separability test on the next finer level.
 - Partially minimize: compute $a_{h,1} \leftarrow \text{TN}(a_h^{(0)}, a_{h,low}, a_{h,up}, v_h, it_{max})$; estimate $\|G^h\|$, the norm of the reduced Hessian, for use in the complementarity test. Downdate the result to obtain $a_{H,1} \leftarrow I_h^H a_{h,1}$.
 - Compute $v_H \leftarrow \nabla F_H(a_{H,1}) - I_h^H \nabla F_h(a_{h,1})$.
 - Apply the multilevel recursion: $[a_{H,2}, \hat{\rho}_H] \leftarrow \text{ML/Opt}(a_{H,1}, a_{H,low}, a_{H,up}, v_H)$ which solves

$$\underset{a_H}{\text{minimize}} \tilde{F}_H(a_H) \equiv F_H(a_H) - v_H^T a_H$$

subject to the bound constraints

$$a_{H,low} \leq a_H \leq a_{H,up}.$$

(See Section 2.1 for a definition of the bounds.)

- Perform separability test.
- Compute the search direction $e_h \leftarrow I_H^h(a_{H,2} - a_{H,1})$.
- Use a line search to obtain $a_{h,2} \leftarrow a_{h,1} + \alpha e_h$. Perform nonlinearity test.
- Perform consistency test and complementarity test.
- Partially minimize: compute $a_h^{(1)} \leftarrow \text{TN}(a_{h,2}, a_{h,low}, a_{h,up}, v_h, it_{max})$; compute $\hat{\rho}_h$, an estimate of the norm of the reduced Hessian $\|G^h\|$, for use in the separability test on the next finer level.

6.2. Complementarity. In assessing the complementarity test, it is revealing to consider two related computational examples that isolate this property.

Both are quadratic optimization problems:

$$F(a) = \frac{1}{2} a^T G a - b^T a,$$

for some positive-definite matrix G and vector b .

The first has a uniformly discretized one-dimensional Laplacian as its Hessian (“Laplacian”). The second uses the same Laplacian, but with a random permutation

TABLE 6.1
Complementarity Test

n_H	n_h	Laplacian	Permuted	Ratio
7	15	0.03	0.34	11.7
15	31	0.04	0.21	4.8
31	63	0.04	0.57	13.6
63	127	0.07	0.30	4.5
127	255	0.13	0.25	1.9
255	511	0.20	0.34	1.7
511	1023	0.26	0.77	3.0

of the assignment of eigenvalues to eigenvector (“Permuted”). That is, if $G = V^T D V$ is the spectral decomposition of G , then the diagonal entries in the diagonal matrix D are randomly permuted.

The first example is ideal for ML/Opt. It satisfies separability (it is separable in the Fourier basis), consistency, and complementarity. The second also satisfies separability and consistency, but not complementarity since the coarse level does not correspond to the near null-space for the problem.

For the first example, the vector b in the linear term is obtained by summing the eigenvectors of the Hessian, and then artificially constraining the components to be between -2 and 2 . (Any entry outside this range is set to -2 if it is too small, and to 2 if it is too big.) Appropriate transformations are used to form the corresponding vector for the other example in such a way as to keep the solution of the optimization problem the same. The initial guess for the Laplacian problem is chosen randomly as follows: In Matlab, the random number generator is initialized with `rand('twister',5489)` and then we use the initial guess `v0 = 20*rand(n,1)`; where n is the dimension of the problem. We transform this vector to get the equivalent initial guess for the Permuted problem.

In these two test problems we use a uniform discretization of the interval $[0, 1]$ with $n = 2^m - 1$ interior points, for $m = 3, 4, \dots, 10$.

Table 6.2 shows the results of the complementarity test for these two problems. For each entry we give values of n_H and n_h , the dimensions of the two levels used in the test. It is clear that the results for the permuted Laplacian are worse than for the Laplacian, and the difference grows ever more dramatic as the dimensions of the levels get smaller. This is not surprising. For larger dimensions there is not a dramatic difference in magnitude between the norms of G^h and G^H . It is only as the dimensions get smaller that there are significant reductions in G^H .

6.3. Consistency. If the underlying optimization problem is sensible, and if the optimization problems on each level are consistent discretizations of the same underlying optimization problem, then in many circumstances the consistency test will be satisfied. Even in this situation, however, it is possible to over-coarsen the discretization, and in that case the consistency test may be violated. But in general, a “sensible” optimization problem is likely to satisfy the consistency test.

For these reasons, we construct a set of artificial optimization problems to illustrate and isolate the consistency test. The problems will be based on the original

Laplacian test problem from the previous subsection. We will apply ML/Opt with two levels: $n = 1023$ and $n = 511$. For $n = 1023$ the optimization problem is the same as before. But for $n = 511$ we artificially distort the Hessian using an orthogonal transformation. Because of the way in which the optimization problems are constructed, they satisfy the nonlinearity test (the problem remains quadratic), the separability test (the fine-level Hessian is unchanged), and the complementarity test (the orthogonal transformation does not alter the overall magnitude of the eigenvalues). However, as the distortion becomes greater, the consistency test is violated.

To define the optimization problems, we need only describe how the coarse-level Hessian G^H is modified. We first generate a random skew-symmetric matrix R using the following Matlab commands: `randn('state',1000)`, `R = randn(511,511)`, `R = R-R'`, `R = R/norm(R)`. The resulting matrix satisfies $\|R\| = 1$. Then for various values of $\alpha \in [0, 1]$ we define an orthogonal matrix Q via

$$Q = (I + \alpha R)(I - \alpha R)^{-1}.$$

It is straightforward to verify that Q is orthogonal. Finally we define the distorted Hessian by

$$G^H \leftarrow Q^T G^H Q.$$

For $\alpha = 0$, the Hessian G^H is unchanged, but as α increases, the distortions grow ever greater. For each value of α we ran one iteration of ML/Opt, with the same initial guess each time, and computed the value of the consistency test after the recursion.

The results are in Figure 6.1. As expected, the consistency test is satisfied for $\alpha \approx 0$, but for larger values of α it is clearly violated. The test is sensitive to distortions in the Hessian.

We illustrate the consistency test with a second test problem derived from the one-dimensional convection-diffusion equation. As discussed in [28], for example, consider the differential equation

$$-\epsilon u'' + au' = f, \quad 0 < x < 1$$

with boundary conditions $u(0) = f_0$ and $u(1) = f_1$. Here a and ϵ are constants. With a uniform discretization h , we use the approximations

$$u_i'' = (u_{i-1} - 2u_i + u_{i+1})/h^2 \quad \text{and} \quad u_i' = (u_{i+1} - u_{i-1})/2h.$$

Then if h is too large, more specifically if $h|a| > 2\epsilon$, the solution has artificial oscillations and is a poor approximation to the solution of the differential equation. Thus an overly coarse discretization should be a poor approximation to a higher-level model, and we would hope that the consistency test would be violated.

The discretization of the convection-diffusion equation is not directly suitable as a test problem, however. The coefficient matrix is non-symmetric, so it is not the Hessian of some equivalent quadratic problem. One possibility would be to form the normal equations and then minimize the equivalent quadratic. This leads to an artificially ill-conditioned optimization problem that is difficult to solve with or without a multilevel algorithm. (However, the consistency test is violated when the discretization is too coarse, as we would expect.)

Instead we have constructed a closely related quadratic optimization problem that gives rise to similarly oscillating solutions when h is too large. It corresponds to using

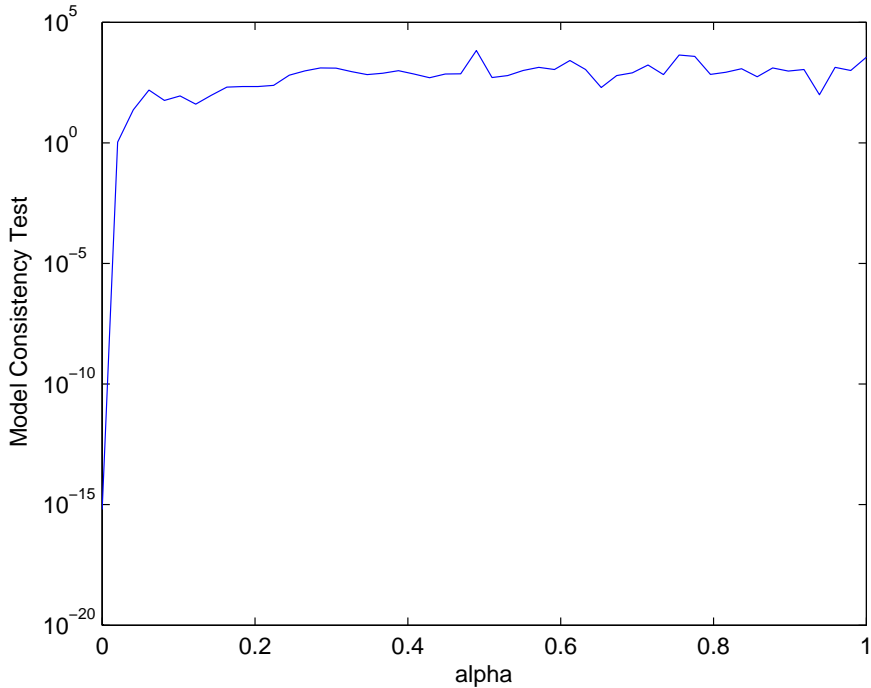


FIG. 6.1. *Consistency Test*

the “approximation” $u'_i = (u_{i+1} + u_{i-1})/2h$. Then the coefficient matrix is symmetric and (for appropriate choices of the parameters a , ϵ , and h) positive definite as well.

We apply ML/Opt to this problem with the following parameters: $f_0 = 0$, $f_1 = 1$, $a = 1$, and $\epsilon = 2 \times 10^{-3}$. The discretization is $h = 1/N + 1$, and we used the models corresponding to $N = 1023, 511, 255, 127$. The solutions for these values of N are illustrated in Figure 6.2, where we have zoomed in on the portion of the solution near $x = 1$.

If we run ML/Opt, the consistency test at the coarsest level has value 16.8, i.e., the predicted reduction is about 16 times larger than the actual reduction.

If we choose $\epsilon = 1.954 \times 10^{-3}$ then the Hessian on the coarsest model is close to being singular and the results are more dramatic. In this case the consistency test at the coarsest level has value 522.5.

6.4. Separability. To demonstrate the separability test we will use a set of test matrices derived from the one-dimensional Laplacian problem described in Section 6.2. We use two levels, corresponding to $n = 63$ and $n = 31$. To construct the test matrices, we first use the matrix of eigenvectors V to diagonalize the Laplacian with the eigenvalues ordered from large to small. We will write the resulting diagonal matrix as

$$V^T G^h V = \begin{pmatrix} D_h & 0 \\ 0 & D_H \end{pmatrix}.$$

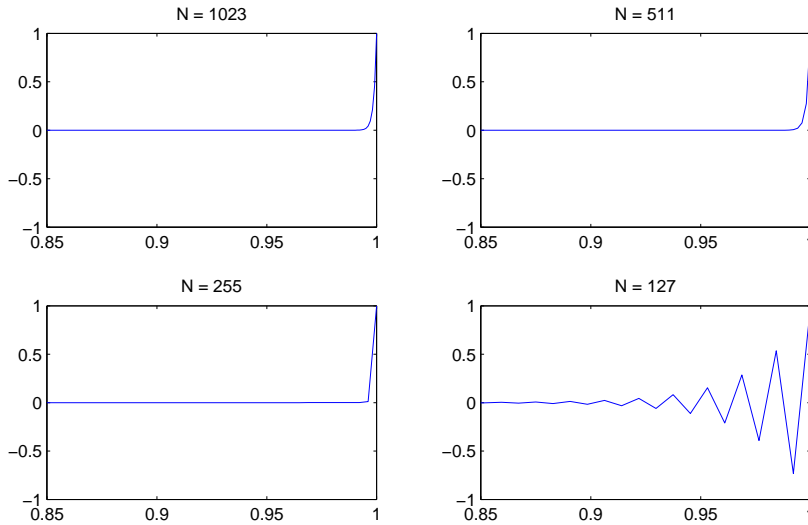


FIG. 6.2. *Oscillatory Model*

The test matrices will have the form

$$G^h(\alpha) \equiv V \begin{pmatrix} D_h & \alpha R \\ \alpha R^T & D_H \end{pmatrix} V^T,$$

where α is a scalar, and R is a random matrix obtained using the Matlab commands `randn('state',1000)`, `R = randn(n1,n2)`, `R = R/norm(R)`. Here $n1$ and $n2$ are the dimensions of the off-diagonal block of $G^h(\alpha)$. We perform a series of experiments for an increasing sequence of positive values of α .

As α increases, at some point $G^h(\alpha)$ will no longer be positive definite. As mentioned in Section 3.4, if we limit our attention to positive-definite matrices, then that will seriously limit the size of the off-diagonal entries in $G^h(\alpha)$, and limit our ability to evaluate this diagnostic test. Large off-diagonal entries may only occur at points far from the solution of the optimization problem, but we still believe that it is valuable to have information on separability even under these circumstances.

We will perform two sets of tests. One will use the user-provided update and downdate procedures, the other will use Fourier transforms to perform “exact” update and downdate operations. As mentioned in Section 3.4, the user-provided procedures are likely to be based on interpolation, and will result in some mixing of frequencies. This mixing limits the sensitivity of the separability test. In some applications it may be feasible to use Fourier transforms or comparable techniques to isolate frequencies and produce a more sensitive diagnostic test, if desired.

The results are illustrated in Figure 6.3. As can be seen, the separability test using “exact” update and downdate operators is ideal, in the sense that there is a near-linear relationship between the norm of the off-diagonal perturbation and the value of the separability measure. If the user-supplied update and downdate operators are used, the test is not quite as sensitive. It is less able to detect very small off-diagonal perturbations, but is successful at detecting larger off-diagonal perturbations. In this case, the user-supplied update and downdate operators can detect off-diagonal

perturbations of a magnitude corresponding to about 2% of the norm of the matrix.

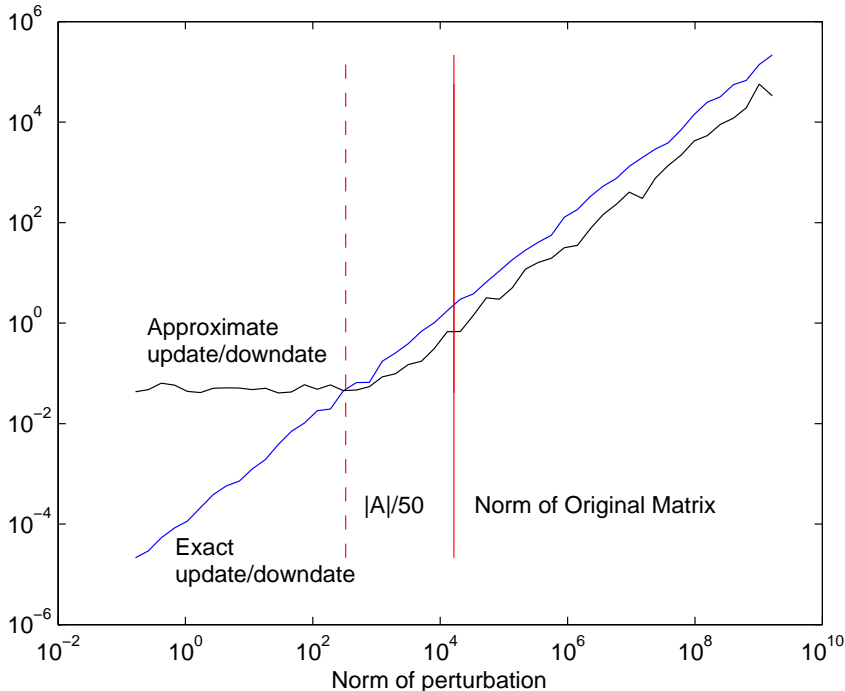


FIG. 6.3. *Separability Test*

6.5. Nonlinearity. Our final test problem illustrates the usefulness of the nonlinearity test. This particular problem does not isolate the nonlinearity test, but it does show how the nonlinearity test might be triggered by a more realistic optimization problem if a poor initial guess is used. The test problem has the general form

$$\min_u \int_{\Omega} \rho(|\nabla u|) + fu \, dx$$

where $\rho(t) = \sqrt{t^2 + \alpha^2}$ with $\alpha \ll 1$. When $\alpha = 0$ the function $\rho(t)$ has a derivative discontinuity at $t = 0$, and even for small α this makes the optimization problem difficult to solve. Specialized techniques are required to make it appropriate for multigrid algorithms [6]. This problem has similarities to the minimal-surface problem which MG/Opt can solve effectively [23].

In our tests we solve a 1-dimensional version of this problem with $\Omega = [0, 1]$. We used a uniform discretization of this interval. We chose $f(t) = 0.6$ (a constant), and used the values $\alpha = 0.01$ and $\alpha = 0.001$. We used the values $n = 511, 255, 127,$ and 63 in MG/Opt, with the initial guess being the vector of all ones. This initial guess is far from the solution.

If we run this problem with $\alpha = 0.01$, then MG/Opt works effectively. Using full multigrid initialization, at the end of the first iteration the solution has been found, with the norm of the gradient equal to 10^{-5} . The results of the assessment tests

are satisfactory, the separability test gives values about 10^{-1} , the complementarity test about 10^{-2} , and the consistency test about 10^{-1} . At the first few iterations of the underlying truncated-Newton method, the nonlinearity test gives nontrivial results (as large as 10^7 at inner iterations 1 and 2), but after that the value of the nonlinearity test is essentially zero. So MG/Opt works well, and the assessment tests are consistent with this behavior.

In contrast, regular optimization is not particularly efficient. The underlying truncated-Newton method requires about 3700 iterations to reduce the norm of the gradient to 4×10^{-4} .

If we run this problem with $\alpha = 0.001$, however, the performance of MG/Opt deteriorates. After the full multigrid initialization the norm of the gradient is 2×10^0 , and subsequent multigrid iterations do not reduce this significantly. The assessment tests for separability and complementarity still give small values. However, the consistency test gives large values (e.g., sometimes on the order of 10^4 or 10^5). And the nonlinearity test has significant values at almost every iteration of the underlying truncated-Newton method.

Using the regular optimization method does not alleviate the difficulties. It requires about 17,000 iterations to reduce the norm of the gradient to 10^{-2} .

Note that, if a better initial guess is used (such as the vector of all zeroes), the performance of MG/Opt is ideal. This true even for smaller values of α . With this better initial guess, the assessment tests give benign values, as we would expect.

7. Conclusions. Optimization-based multilevel algorithms such as ML/Opt can be a powerful technique for solving large and challenging optimization problems, but they are not appropriate for all optimization problems. We have demonstrated that the performance of ML/Opt depends on four properties of the optimization problem: nonlinearity, complementarity, consistency, and separability. In addition, we have developed computational tests that can be used to assess whether the optimization problem has the desired properties. These computational tests are “cheap” since they only require modest computational effort beyond that required by the optimization method itself, and do not require any additional information about the optimization problem beyond that already provided by the user. Further, the tests can be applied while the optimization problem is being solved, and thus can be useful even if the characteristics of the optimization problem change from one point to another. In addition to determining whether a multilevel algorithm is appropriate for a particular optimization problem, the tests identify relevant properties of the optimization problem, and thus might be used to guide a user to an optimization algorithm that is better suited to the particular optimization problem. In the long term, it might be possible to automate this process, and have optimization software automatically identify an appropriate algorithm to apply to a particular optimization problem to achieve good performance and efficiency.

8. Acknowledgements. We would like to thank Ray Tuminaro for pointing us toward relevant results from algebraic multigrid. We are grateful to the referees for their helpful comments. The contributions of Stephen G. Nash were based on work supported by the National Science Foundation, while he was working at the Foundation.

REFERENCES

- [1] A. BORZI AND U. HOHENESTER, *Multigrid optimization schemes for solving Bose-Einstein condensates control problems*, SIAM Journal on Scientific Computing, 30 (2008), pp. 441–462.
- [2] A. BRANDT, *Multi-level adaptive solutions to boundary-value problems*, Mathematics of Computation, 31 (1977), pp. 333–390.
- [3] J. BRANNICK AND L. ZIKATANOV, *Algebraic multigrid methods based on compatible relaxation and energy minimization*, Tech. Report AM304, Center for Computational Mathematics and Applications, The Pennsylvania State University, University Park, PA, 2006.
- [4] M. BREZINA, R. FALGOUT, S. MACLACHLAN, T. MANTEUFFEL, S. MCCORMICK, AND J. RUGE, *Adaptive algebraic multigrid*, SIAM Journal on Scientific Computing, 27 (2006), pp. 1261–1286.
- [5] T. F. CHAN, J. CONG, T. KONG, J. R. SHINNERL, , AND K. SZE, *An enhanced multilevel algorithm for circuit placement*, in Proceedings of the International Conference on Computer-Aided Design, 2003, pp. 299–306.
- [6] T. F. CHAN AND W. L. WAN, *Robust multigrid methods for nonsmooth coefficient elliptic linear systems*, Journal of Computational and Applied Mathematics, 123 (2000), pp. 323–352.
- [7] J. CHINNECK, *Analyzing mathematical programs using MProbe*, Annals of Operations Research, 104 (2001), pp. 33–48.
- [8] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Trust-Region Methods*, vol. 1 of MPS/SIAM Series on Optimization, SIAM, Philadelphia, 2000.
- [9] G. CORNUÉJOLS, M. KARAMANOV, AND Y. LI, *Early estimates of the size of branch-and-bound trees*, INFORMS Journal on Computing, 18 (2006), pp. 86–96.
- [10] S. C. EISENSTAT AND H. F. WALKER, *Choosing the forcing terms in an inexact newton method*, SIAM Journal on Scientific Computing, 4 (1996), pp. 16–32.
- [11] G. H. GOLUB AND C. V. LOAN, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1996.
- [12] L. HAGEMAN AND R. B. KELLOGG, *Estimating optimum overrelaxation parameters*, Mathematics of Computation, 22 (1968), pp. 60–68.
- [13] T. G. KOLDA, R. M. LEWIS, AND V. TORCZON, *Optimization by direct search: New perspectives on some classical and modern methods*, SIAM Review, 45 (2003), pp. 385–482.
- [14] R. M. LEWIS AND S. G. NASH, *A multigrid approach to the optimization of systems governed by differential equations*. AIAA Paper 2000-4890, 2000.
- [15] ———, *Practical aspects of multiscale optimization methods for vlsicad*, in Multiscale Optimization and VLSI/CAD, J. Cong and J. R. Shinnerl, eds., Kluwer Academic Publishers, Boston, 2002, pp. 265–291.
- [16] ———, *Model problems for the multigrid optimization of systems governed by differential equations*, SIAM Journal on Scientific Computing, 26 (2005), pp. 1811–1837.
- [17] T. MANTEUFFEL, *Adaptive procedure for estimating parameters for the nonsymmetric tchebyshev iteration*, Numerische Mathematik, 31 (1978), pp. 183–208.
- [18] R. MATHIAS, *Quadratic residual bounds for the Hermitian eigenvalue problem*, SIAM Journal on Matrix Analysis, (1998), pp. 541–550.
- [19] S. F. MCCORMICK, ed., *Multigrid Methods*, Frontiers in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, 1987.
- [20] ———, *Multilevel Adaptive Methods for Partial Differential Equations*, Society for Industrial and Applied Mathematics, 1989.
- [21] S. G. NASH, *TN/TNBC software*. <http://www-fp.mcs.anl.gov/OTC/Guide/SoftwareGuide/Blurbs/tn.tnbc.html>. ■
- [22] ———, *Newton-type minimization via the Lanczos method*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 770–788.
- [23] ———, *A multigrid approach to discretized optimization problems*, Journal of Computational and Applied Mathematics, 14 (2000), pp. 99–116.
- [24] S. G. NASH AND J. NOCEDAL, *A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization*, SIAM Journal on Optimization, 1 (1991), pp. 358–372.
- [25] S. G. NASH AND A. SOFER, *Linear and Nonlinear Programming*, McGraw–Hill, New York, 1996.
- [26] I. NENOV, D. FYLSTRA, AND L. KOLEV, *Convexity determination in the Microsoft Excel solver using automatic differentiation techniques*, tech. report, Frontline Systems Inc., Incline Village, NV, 2004.
- [27] C. C. PAIGE AND M. A. SAUNDERS, *Solution of sparse indefinite systems of linear equations*, SIAM Journal on Numerical Analysis, 12 (1975), pp. 617–629.
- [28] U. TROTTEMBERG, C. OOSTERLEE, AND A. SCHÜLLER, *Multigrid*, Academic Press, London, 2001.