Optimization-based Multigrid Applied to Aerodynamic Shape Design

Markus P. Rumpfkeil* and Dimitri J. Mavriplis †

University of Wyoming, Laramie, 82071, USA

In this paper an optimization-based multigrid algorithm is applied to the optimization of systems governed by differential equations. The multigrid algorithm relies on nonlinear optimization models as sub-problems on coarser levels and employs a line search globalization technique. The application of the algorithm to an elliptic model problem and aerodynamic shape design shows the potential for large improvement in the overall computational cost for high fidelity optimizations which typically involve many design and state variables.

Nomenclature

α	Angle of attack
β_l	Line search step length
C_d	Drag coefficient
C_l	Lift coefficient
D	Design variables
$\mathbb{D}$	Total number of design variables
D_l^{low}, D_l^{up}	Lower and upper bounds on design variables
e_l	Line search direction
f	Objective function
$\frac{df}{dD_j}$	Gradient of objective function
$\frac{d^2f}{dD_j dD_k}$	Hessian of objective function
I_{l+1}^l, I_l^{l+1}	Prolongation and Restriction operator
K_1, K_2	Number of pre- and post-smoothing iterations
l	Multigrid level
N	Finest multigrid level
N_0	Coarsest multigrid level
q	Flow variables
R	Flow residual
s	Grid residual
v_l	Fine-to-coarse gradient correction
x	Grid variables

I. Introduction and Motivation

The optimization problems arising in aerodynamic (shape) design involve systems of partial differential equations (PDEs) which model the fluid dynamics. Through extensive research in the last decades, efficient computational fluid dynamics (CFD) codes have become available to solve these PDEs. However, for solving an entire optimization problem the cost is determined by the level of discretization and thus number of state variables used to solve the governing PDEs since one has to solve the field equations (and the adjoint equations for gradient based methods[1,2]) quite accurately several times. Experience has also shown that

*Postdoctoral Research Associate, Department of Mechanical Engineering; mrumpfke@uwyo.edu

†Professor, Department of Mechanical Engineering; mavripl@uwyo.edu

convergence of the optimization problem degrades as the number of design variables increases.[3,4] Thus, the number of times the complete set of field equations must be solved in an optimization problem can be expected to scale with the number of design variables. This makes the overall computational cost for high fidelity optimization problems, where many state and design variables are employed, very high.

In a fairly recent approach Nash and Lewis[5–7] proposed a multigrid optimization framework for solving (convex) optimization problems which they called MG/OPT. MG/OPT recursively uses coarse resolution optimization problems to generate search directions for finer-resolution optimization problems. Since the sub-problems on the different levels are of similar structure one can use the same algorithms and software modules to solve them. MG/OPT is inspired by multigrid methods for elliptic PDEs,[5,7] however, the underlying system of state equations of the optimization problem need not be elliptic. It can also be related to more traditional optimization techniques, in particular to the steepest descent and Newton's method for optimization.[7] Nash[5] also discussed extensions to non-convex problems and the concept of automatically determining the finest grid level such that a measure of error for the converged solution is below a certain threshold.

Expanding on Nash's approach Gratton, Sartenaer and Toint[8–10] proposed a recursive trust region method that also converges for non-convex problems to a first-order optimal point. Wen and Goldfarb[11] proposed a line search multigrid method that adopts some of the features of both Nash's method and the method of Gratton, Sartenaer and Toint. Another promising approach is the multigrid one-shot method in which simultaneous pseudo-time-stepping is used to solve the entire optimization problem at once.[12,13]

The good performance of MG/OPT for model problems[7] encouraged us to apply this algorithm to CFD inspired optimization problems. In Section II the basic algorithm for a full multigrid optimization is outlined. Sections III and IV show the application of the algorithm to an elliptic model problem and to an aerodynamic shape design optimization problem, respectively. Finally, Section V concludes this paper.

II. Basic Algorithm

We consider nonlinear optimization problems of the form

$$\min_D \; f(D) = F(D, x(D), q(D)), \quad f, F \in \mathbb{R} \tag{1}$$

with respect to the independent design variables $D \in \mathbb{R}^{\mathbb{D}}$ such that the grid variables $x(D)$ and flow variables $q(D)$ satisfy the grid residual equation

$$s(D, x(D)) = 0, \tag{2}$$

and flow residual equation

$$R(D, x(D), q(D)) = 0. \tag{3}$$

Nash and Lewis[5–7] proposed to recursively use coarse resolution problems to generate search directions for finer-resolution problems and then to apply a line search to refine the solution. The line search globalization technique makes it possible to prove convergence results for their multigrid optimization algorithm MG/OPT.[5] Also, at least one iteration of an optimization algorithm must be performed either before going to or after returning from a coarser level during a multigrid cycle. These iterations are similar to prior smoothing or post smoothing steps in multigrid methods for PDEs. We adopt the following notation in this paper: $f_{l,k} := f_l(D_{l,k})$, $\nabla f_{l,k} := \nabla f_l(D_{l,k})$, $R_l := R_l(D_l, x_l(D_l), q_l(D_l))$, and $s_l := s_l(D_l, x_l(D_l))$, where the first subscript l denotes the discretization level of the multigrid and the second subscript k denotes the iteration count. I^l_{l+1} denotes a prolongation operator, while I^{l+1}_l denotes a restriction operator with the standard assumption that $I^l_{l+1} = \text{const} \cdot (I^{l+1}_l)^T$. N is reserved for the index of the finest level and N_0 for the coarsest. The notation suggests that a coarsening is applied to both the design and state variables. However, it is not necessary to coarsen both and either the number of design or state variables can remain constant.

One cycle of the algorithm MG/OPT can be expressed as follows (assuming a simple V-cycle):

MG/OPT Algorithm $D_{l,k} = \text{MG/OPT}(l, D_{l,0}, v_l)$

- If $l = N_0$ solve

$$\min_{D_l} \left(f_l(D_l) - v_l^T D_l \right) \tag{4}$$

subject to

$$\begin{aligned} R_l &= 0 \\ s_l &= 0 \\ D_l^{low} &\leq D_l \leq D_l^{up} \end{aligned} \tag{5}$$

and return $D_{l,k}$

- Otherwise $l < N_0$,

1. *Pre-optimization*
   Apply K_1 iterations of an optimization algorithm to the problem (4)-(5) and obtain D_{l,K_1}
2. *Coarse grid problem*
   Restrict the solution of Step 1, $D_{l+1,K_1} = I_l^{l+1} D_{l,K_1}$
   Compute the fine-to-coarse gradient correction,

$$v_{l+1} = I_l^{l+1} v_l + \tau_{l+1} \quad \text{with} \quad \tau_{l+1} = \nabla f_{l+1}(D_{l+1,K_1}) - I_l^{l+1} \nabla f_l(D_{l,K_1}) \tag{6}$$

   Apply one cycle of MG/OPT $D_{l+1,k} = \text{MG/OPT}(l+1, D_{l+1,K_1}, v_{l+1})$
3. *Coarse-to-fine minimization*
   Prolongate the error, $e_l = I_{l+1}^l [D_{l+1,k} - D_{l+1,K_1}]$
   Perform a line search to obtain $D_{l,K_1+1} = D_{l,K_1} + \beta_l e_l$
4. *Post-optimization*
   Apply K_2 iterations of an optimization algorithm to the problem (4)-(5) and obtain D_{l,K_1+1+K_2}

The algorithm is considered to have converged once a gradient norm on the finest level $||\nabla f_N||$ is below a certain tolerance. The coarse grid optimization problem is a first-order approximation to the fine grid optimization problem since

$$\begin{aligned} \nabla[f_l(D_{l,0}) - v_l^T D_{l,0}] &= \nabla f_l(D_{l,0}) - v_l \\ &= \nabla f_l(D_{l,0}) - I_{l-1}^l v_{l-1} - \nabla f_l(D_{l,0}) + I_{l-1}^l \nabla f_{l-1}(D_{l-1,0}) \\ &= I_{l-1}^l [\nabla f_{l-1}(D_{l-1,0}) - v_{l-1}], \end{aligned}$$

that is, the gradient of the coarse grid problem at $D_{l,0}$ is the restriction of the gradient of the fine-grid problem at $D_{l-1,0}$.[5–7] This ensures that line search steps based on the coarse grid correction will yield improvement for the fine grid problem. It is also possible to make the objective function values match at the initial point by adding a constant to the coarse grid objective function.[6] However, adding a constant to the objective function has no significant effect on the overall behavior of any optimization algorithm. The line search direction e_l is a descent direction for convex optimization problems.[5] However, there is no a priori estimate of a good step length β_l which is why Lewis and Nash[6] included bound constraints

$$D_l^{low} \leq D_l \leq D_l^{up}$$

for the coarser grid optimization problems. These bounds are heuristic, analogous to trust-region strategies[14] and defined as follows[7]

$$\gamma_1 = ||v_l||$$

$$\begin{aligned}
\gamma_2 &= ||\nabla f_l(D_l)|| \\
\gamma_3 &= ||I_{l-1}^{l}[\nabla f_{l-1}(D_{l-1}) - v_{l-1}]|| \\
\delta &= \max(\gamma_1, \gamma_2, \gamma_3) \\
D_l^{low} &= D_l - \delta \cdot (1 \ldots 1)^T \\
D_l^{up} &= D_l + \delta \cdot (1 \ldots 1)^T.
\end{aligned}$$

If there are bounds, D_N^{low} and D_N^{up}, imposed on the finest level N then

$$\begin{aligned}
D_l^{low} &= \max(D_N^{low}, D_l^{low}) \\
D_l^{up} &= \min(D_N^{up}, D_l^{up}).
\end{aligned}$$

The basic MG/OPT algorithm solves problem (1)-(3) by calling the multigrid cycle $D_{N,k} = \text{MG/OPT}(N, D_{N,0}, v_N = 0)$ as many times as it takes to converge. Since starting from a good initial point usually reduces the total number of cycles required, the performance of MG/OPT can be improved by using a full multigrid method, that is, the use of the multilevel approach itself to provide a good initial point. The resulting algorithm FMG/OPT for one cycle can be expressed as:

FMG/OPT Algorithm $D_{N,k} = \text{FMG/OPT}(N_0, D_{N,0})$

---

1. Set $v_N = 0$ and $D_{N,0^*} = D_{N,0}$, restrict $D_{N,0^*}$ to coarser levels to obtain $D_{l,0^*}$ for $l = N+1, N+2, \ldots, N_0$ and compute v_l for $l = N+1, N+2, \ldots, N_0$ using equation (6)
2. Set $D_{N_0,0} = D_{N_0,0^*}$
3. For $l = N_0, N_0 - 1, \ldots, N$
   - 3.1. Call $D_{l,k} = \text{MG/OPT}(l, D_{l,0}, v_l)$
   - 3.2. If $l > N$, prolongate the error $e_{l-1} = I_l^{l-1}[D_{l,k} - D_{l,0}]$ and perform a line search to obtain an initial point $D_{l-1,0} = D_{l-1,0^*} + \beta_{l-1} e_{l-1}$ for level $l-1$

FMG/OPT requires very little from the user: the restriction and prolongation operators as well as an optimization algorithm must be specified and the user must be able to provide $f_l(D_l)$ and $\nabla f_l(D_l)$ (and $\nabla^2 f_l(D_l)$ if a full Newton optimizer is used) for any multigrid level, l. The rest of the algorithm is independent of the particular optimization problem being solved.[5] Thus, FMG/OPT isolates the optimization techniques from the techniques used to solve the underlying PDEs which allows considerable flexibility for the implementation and very general applicability. FMG/OPT is applied to examples in the following sections.

III. Elliptic Model Problem

The elliptic model problem is the Dirichlet-to-Neumann map for the Laplacian on a square $\Omega = \{(x_1, x_2) \mid 0 \le x_1 \le \pi, 0 \le x_2 \le \pi\}$ which is very similar to the problem considered by Gratton, Sartenaer and Toint.[8] The region Γ is defined to be the bottom edge of the square. The governing equations are given by

$$\begin{aligned}
\Delta u(x_1, x_2) &= 0 \quad \text{in } \Omega, \\
u(x_1, x_2) &= 0 \quad \text{on } \partial\Omega \setminus \Gamma, \\
u(x_1, 0) &= D(x_1).
\end{aligned} \tag{7}$$

In the physical analogy where equations (7) represent a model of heat conduction in the square, the design variables D represent the temperature distribution along Γ. The objective function is defined as

$$f(D) = \int_0^\pi \left(\frac{\partial u}{\partial x_2}(x_1, 0) - \phi(x_1) \right)^2 dx_1,$$

where ϕ is a prescribed target for the heat flux through the lower edge of the square.

The governing equations are discretized using M grid points in both directions yielding $\Delta x_1 = \Delta x_2 = \frac{\pi}{M-1}$. The discretized objective function takes the form

$$f(D) = \sum_{j=1}^{M} \left(\frac{u_{j,2} - u_{j,1}}{\Delta x_1} - \phi_j \right)^2,$$

and the governing equations are solved using successive over relaxation (SOR) for the interior grid points as follows

$$\begin{aligned} \delta u^n_{j,k} &= (u^n_{j+1,k} + u^n_{j-1,k} + u^n_{j,k+1} + u^n_{j,k-1})/4 - u^n_{j,k} \\ u^{n+1}_{j,k} &= u^n_{j,k} + \omega \delta u^n_{j,k} \end{aligned}$$

with the optimal relaxation factor $\omega = 2/[1 + \sin(\frac{\pi}{M+1})]$. The SOR algorithm is iterated until $||\delta u^n||_2 \leq 10^{-12}$. A highly oscillatory target heat flux is defined as

$$\phi_j = \sin[40 \cdot (j–1) \cdot \Delta x_1] + \sum_{k=1}^{15} \sin[k \cdot (j–1) \cdot \Delta x_1] \quad \text{for} \quad j = 1, \ldots, M$$

and for $M = 129$ we show the solution u which yields this heat flux in Figure 1.

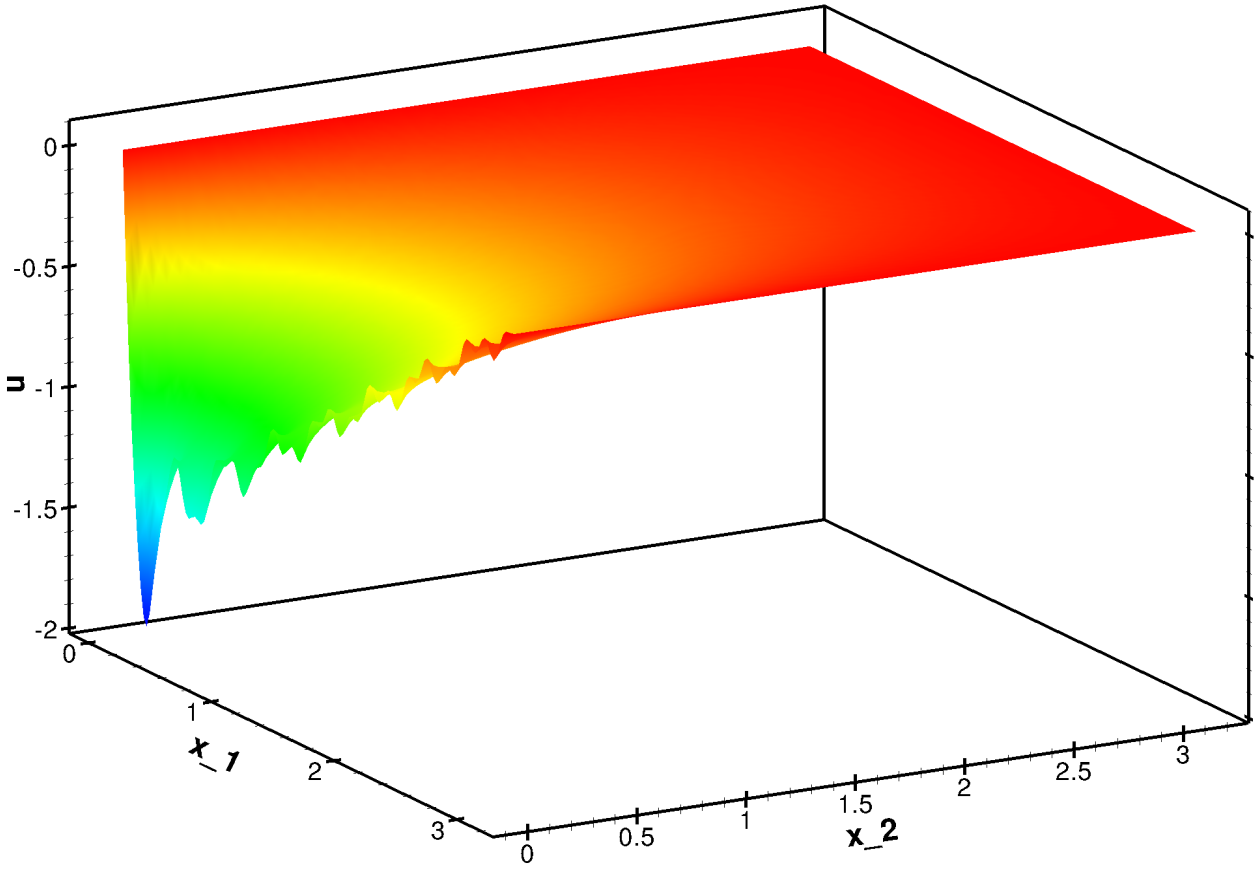

Figure 1. The solution u that yields the target heat flux ϕ ($M = 129$).

The gradient of the objective function f is computed via an adjoint computation and the Hessian, if required, can be calculated as follows. The first derivative of f with respect to one individual component of D is given by

$$\frac{df}{dD_s} = \frac{2}{\Delta x_1} \sum_{j=1}^{M} \left(\frac{d(u_{j,2} - u_{j,1})}{dD_s} \right)^T \left(\frac{u_{j,2} - u_{j,1}}{\Delta x_1} - \phi_j \right).$$

Differentiating again yields

$$\begin{aligned}\frac{d^2 f}{dD_s dD_r} &= \frac{2}{(\Delta x_1)^2}\sum_{j=1}^{M}\left(\frac{d(u_{j,2}-u_{j,1})}{dD_s}\right)^T \frac{d(u_{j,2}-u_{j,1})}{dD_r} \\ &+ \frac{2}{\Delta x_1}\sum_{j=1}^{M}\left(\frac{u_{j,2}-u_{j,1}}{\Delta x_1}-\phi_j\right)\frac{d^2(u_{j,2}-u_{j,1})}{dD_s dD_r} \\ &= \frac{2}{(\Delta x_1)^2}\sum_{j=1}^{M}\left(\frac{d(u_{j,2}-u_{j,1})}{dD_s}\right)^T \frac{d(u_{j,2}-u_{j,1})}{dD_r}. \end{aligned} \tag{8}$$

The last equality is true since the u-values depend only linearly on D. Equation (8) implies that we can compute the Hessian by only determining the first derivatives $\frac{d(u_{j,2}-u_{j,1})}{dD_s}$ for $j = 1, \ldots, M$ and $s = 1, \ldots, \mathbb{D}$, which is accomplished by using the adjoint as well.

The restriction and prolongation operators are most easily defined by way of an example. Let D represent a set of coarse grid design variables and d a set of fine level variables

$$D_{l+1} = (D_1\ D_2\ D_3) \quad \text{and} \quad d_l = (d_1\ d_2\ d_3\ d_4\ d_5).$$

Then

$$I_{l+1}^{l} D_{l+1} = \begin{pmatrix} D_1 & \frac{1}{2}(D_1+D_2) & D_2 & \frac{1}{2}(D_2+D_3) & D_3 \end{pmatrix} \tag{9}$$

and

$$I_{l}^{l+1} d_l = \frac{1}{2}\begin{pmatrix} d_1+\frac{1}{2}d_2 & \frac{1}{2}d_2+d_3+\frac{1}{2}d_4 & \frac{1}{2}d_4+d_5 \end{pmatrix}. \tag{10}$$

The last parameters that need to be specified are the optimization algorithm, the number of multigrid levels, the convergence criterion, the initial point, and the number of pre- and post-optimization iterations, K_1 and K_2, respectively. We use quasi-Newton optimizers (L-BFGS-B[15,16] or a simple steepest descent algorithm (SD) with a line search), which need only function and gradient evaluations, on all multigrid levels except for the coarsest level. Here, we use a full Newton optimizer (KNITRO[17]) which additionally requires the evaluation of the Hessian. All optimizers can handle simple bound constraints on the design variables, which are required for the implementation of FMG/OPT. The reasoning behind this choice is that the Hessian is pretty expensive to evaluate on all but the coarsest level. However, on the coarsest level the Hessian information aids to fully solve the optimization problem, whereas on all other levels we only require a few optimization iterations. We use three multigrid levels where the optimization on level l is considered to be converged if $||\nabla f_l||_2 \leq 10^{-4}/10^{l-N}$ and the initial point is taken as $D_{N,0} = 0$.

We did extensive testing for the optimal choice of K_1 and K_2 and we decided to use the following strategy for L-BFGS-B as smoother: K_1 is always taken to be zero and K_2 is initialized as two on the finest level and for the first FMG/OPT cycle. For each new FMG/OPT cycle K_2 is increased by one and for each coarser multigrid level K_2 is decreased by one. The reason for this strategy is as follows: At the beginning of the optimization the initial point is presumably far away from the solution and we do not want to waste too many iterations on the finest level. However, as the number of cycles increases we approach the solution and increase the number of iterations on the finest level to allow L-BFGS-B to build a better Hessian approximation and pinpoint the solution. At the same time the pre- and post-optimization iterations on all the intermediate multigrid levels are for smoothing purposes only and a few iterations can be saved by decreasing K_2 on those levels. For SD as smoother the best strategy seems to be to simply set K_1 to zero and K_2 to one.

Tables 1 and 2 give an overview of our results for coarsening both the design and state variables ($\mathbb{D}_l = M_l$, for $l = N_0, N_0 - 1, \ldots, N$) using L-BFGS-B and SD as optimizers as well as smoothers, respectively. The term "tnfg" refers to total number of function and gradient evaluations and "tnh" means total number of Hessian evaluations. For quadratic problems like this the Hessians on all the required levels can be precomputed, however, we still show the total number of Hessian "evaluations" in the tables.

One can clearly see that FMG/OPT is superior to the use of L-BFGS-B or SD alone, since it shifts much of the computational effort to coarser grids. It is also apparent that L-BFGS-B is much more effective than SD, both as single level optimizer and smoother for FMG/OPT. Furthermore, FMG/OPT's performance is almost independent of the number of design and state variables because it always takes roughly the same

Table 1. Results for the elliptic model problem using L-BFGS-B ($\mathbb{D}_l = M_l$).

		$M = 513$	$M = 257$	$M = 129$	$M = 65$	$M = 33$	$M = 17$
Pure L-BFGS-B	iter	206	106	64	40	23	16
	tnfg	215	118	70	44	25	18
FMG/OPT	cycles	5	-	-	-	-	-
(start at M=513)	tnfg	34	84	30	-	-	-
	tnh	0	0	15	-	-	-
FMG/OPT	cycles	-	5	-	-	-	-
(start at M=257)	tnfg	-	33	63	30	-	-
	tnh	-	0	0	15	-	-
FMG/OPT	cycles	-	-	5	-	-	-
(start at M=129)	tnfg	-	-	34	66	30	-
	tnh	-	-	0	0	15	-
FMG/OPT	cycles	-	-	-	5	-	-
(start at M=65)	tnfg	-	-	-	35	69	30
	tnh	-	-	-	0	0	15

Table 2. Results for the elliptic model problem using steepest descent ($\mathbb{D}_l = M_l$).

		$M = 513$	$M = 257$	$M = 129$	$M = 65$	$M = 33$	$M = 17$
Pure SD	iter	153346	39363	10340	2460	633	151
	tnfg	472732	118087	31020	7380	1899	453
FMG/OPT	cycles	8	-	-	-	-	-
(start at M=513)	tnfg	43	194	48	-	-	-
	tnh	0	0	24	-	-	-
FMG/OPT	cycles	-	7	-	-	-	-
(start at M=257)	tnfg	-	35	73	42	-	-
	tnh	-	0	0	21	-	-
FMG/OPT	cycles	-	-	7	-	-	-
(start at M=129)	tnfg	-	-	35	123	42	-
	tnh	-	-	0	0	21	-
FMG/OPT	cycles	-	-	-	7	-	-
(start at M=65)	tnfg	-	-	-	36	75	42
	tnh	-	-	-	0	0	21

amount of design cycles (i.e. the same number of fine level function and gradient evaluations) to converge. As a side note, KNITRO takes only two function and gradient evaluations as well as one Hessian evaluation on any level to converge since the objective function is a quadratic function of the design variables.

In our next example we keep the number of design variables constant which requires a few very simple changes to FMG/OPT. This approach can be referred to as semi-coarsening in the state variables. We choose $\mathbb{D}_l = 33$ design variables (temperature values) which are imposed in equal distanced locations along Γ with $\Delta x_1 = \frac{\pi}{32}$. We use simple linear interpolation if intermediate values for finer meshes are required. The results using L-BFGS-B as the optimizer and smoother are shown in Table 3 and the results using SD are displayed in Table 4.

The performance of FMG/OPT is not quite as impressive as using coarsening in both design and state variables. However, it is still independent of the number of state variables which makes the use of FMG/OPT for a large number of state variables worthwhile. However, FMG/OPT using SD as smoother outperforms the single level SD optimizer by a large margin.

In the last example we keep the number of state variables constant. This approach can be referred

Table 3. Results for the elliptic model problem using L-BFGS-B ($\mathbb{D}_l = 33$).

		$M = 513$	$M = 257$	$M = 129$	$M = 65$	$M = 33$
Pure L-BFGS-B	iter	48	42	38	31	23
	tnfg	61	46	40	34	25
FMG/OPT	cycles	4	-	-	-	-
(start at M=513)	tnfg	33	133	24	-	-
	tnh	0	0	12	-	-
FMG/OPT	cycles	-	4	-	-	-
(start at M=257)	tnfg	-	33	64	24	-
	tnh	-	0	0	12	-
FMG/OPT	cycles	-	-	4	-	-
(start at M=129)	tnfg	-	-	33	62	24
	tnh	-	-	0	0	12

Table 4. Results for the elliptic model problem using steepest descent ($\mathbb{D}_l = 33$).

		$M = 513$	$M = 257$	$M = 129$	$M = 65$	$M = 33$
Pure SD	iter	3429	2982	2294	1450	633
	tnfg	10287	8946	6882	4350	1899
FMG/OPT	cycles	5	-	-	-	-
(start at M=513)	tnfg	32	79	30	-	-
	tnh	0	0	15	-	-
FMG/OPT	cycles	-	5	-	-	-
(start at M=257)	tnfg	-	28	117	30	-
	tnh	-	0	0	15	-
FMG/OPT	cycles	-	-	5	-	-
(start at M=129)	tnfg	-	-	29	60	30
	tnh	-	-	0	0	15

as semi-coarsening in the design variables. The results using both L-BFGS-B and SD as optimizers and smoothers are shown in Tables 5 and 6, respectively.

FMG/OPT is worse than in the previous case. In fact, FMG/OPT cannot outperform the single level L-BFGS-B optimizer. This is partly due to the fact that coarse level function and gradient evaluations take roughly the same amount of time as fine level evaluations, since the number of state variables is constant on all levels. However, in this case FMG/OPT also does not substantially reduce the number of fine level function and gradient evaluations compared to L-BFGS-B. On the other hand, FMG/OPT does result in a large reduction of fine level evaluations when using a weaker optimizer such as SD. In both cases, the number of optimization cycles varies much more slowly with the number of design cycles using FMG/OPT compared to the equivalent single level optimizer. Thus, this approach may be useful for cases with thousands of design variables. However, convergence of the optimization process is not entirely independent of the number of design variables as in the previous cases, and this dependence remains an area of further investigation. In the next section we describe the application of FMG/OPT to an aerodynamic shape optimization problem.

IV. Aerodynamic Shape Optimization

We consider the inviscid steady flow around a NACA 0012 airfoil as a flow example which is described in more detail in Mani and Mavriplis.[18,19] The finest mesh has about $20,000$ triangular elements and is shown in the top of Figure 2. The required coarser meshes are built by repeated agglomeration or merging of neighboring control volumes to form a single control volume.[18] Two coarser levels are shown in the middle and bottom of Figure 2 with $7,900$ and $2,600$ elements, respectively. The deformation and movement of the

Table 5. Results for the elliptic model problem using L-BFGS-B ($M_l = \mathbb{D}_N$).

		$\mathbb{D} = 513$	$\mathbb{D} = 257$	$\mathbb{D} = 129$	$\mathbb{D} = 65$	$\mathbb{D} = 33$	$\mathbb{D} = 17$
Pure L-BFGS-B	iter	206	106	64	40	23	16
	tnfg	215	118	70	44	25	18
FMG/OPT	cycles	17	-	-	-	-	-
(use M=513)	tnfg	213	581	102	-	-	-
	tnh	0	0	51	-	-	-
FMG/OPT	cycles	-	14	-	-	-	-
(use M=257)	tnfg	-	158	312	84	-	-
	tnh	-	0	0	42	-	-
FMG/OPT	cycles	-	-	12	-	-	-
(use M=129)	tnfg	-	-	116	247	72	-
	tnh	-	-	0	0	36	-
FMG/OPT	cycles	-	-	-	9	-	-
(use M=65)	tnfg	-	-	-	78	157	54
	tnh	-	-	-	0	0	27

Table 6. Results for the elliptic model problem using steepest descent ($M_l = \mathbb{D}_N$).

		$\mathbb{D} = 513$	$\mathbb{D} = 257$	$\mathbb{D} = 129$	$\mathbb{D} = 65$	$\mathbb{D} = 33$	$\mathbb{D} = 17$
Pure SD	iter	153346	39363	10340	2460	633	151
	tnfg	472732	118087	31020	7380	1899	453
FMG/OPT	cycles	21	-	-	-	-	-
(use M=513)	tnfg	86	264	126	-	-	-
	tnh	0	0	63	-	-	-
FMG/OPT	cycles	-	19	-	-	-	-
(use M=257)	tnfg	-	77	205	114	-	-
	tnh	-	0	0	57	-	-
FMG/OPT	cycles	-	-	17	-	-	-
(use M=129)	tnfg	-	-	69	153	102	-
	tnh	-	-	0	0	51	-
FMG/OPT	cycles	-	-	-	16	-	-
(use M=65)	tnfg	-	-	-	63	144	96
	tnh	-	-	-	0	0	48

mesh is performed via a linear tension spring analogy[18–20] on the finest level.

The free-stream Mach number is $M_\infty = 0.755$ with an angle of attack of 1.25 degrees. The non-dimensionalized pressure contours for the flow at this angle of attack are shown in Figure 3 leading to a lift coefficient of $C_l = 0.268$ and a drag coefficient of $C_d = 0.00521$.

The optimization example consists of an inverse design given by the following objective function:

$$f = \frac{1}{2}(C_l - C_l^*)^2 + \frac{100}{2}(C_d - C_d^*)^2, \tag{11}$$

where a star denotes a target lift or drag coefficient and the factor of one hundred is introduced since the drag coefficient is about an order of magnitude smaller than the lift coefficient in this particular flow example. The objective function is always scaled such that its initial value is unity.

We use $\mathbb{D} = 258$ design variables, half of which are placed at upper and the other half at lower surface points which control the magnitude of Hicks-Henne sine bump functions.[21] This steady inverse design problem is initialized with the NACA 0012 airfoil profile and the target lift and drag coefficients are set to

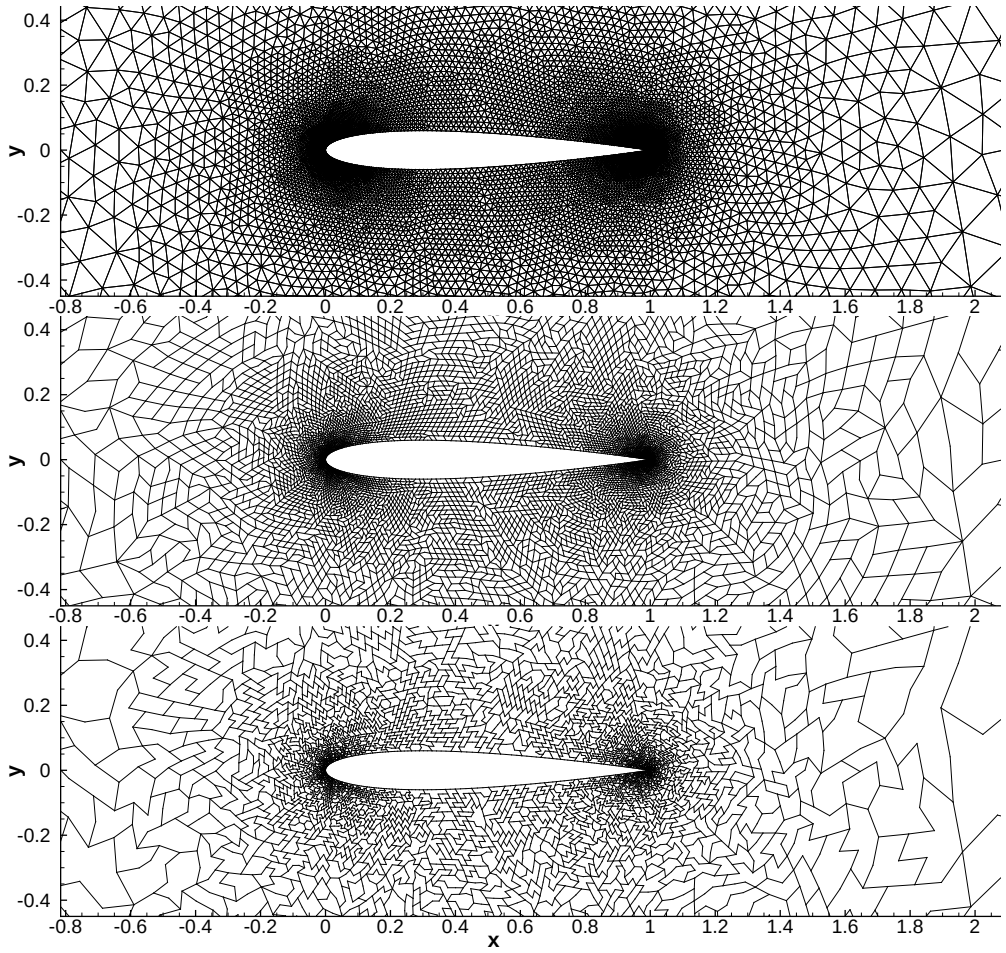

Figure 2. The triangular finest mesh with about $20,000$ elements (top) and agglomerated coarser meshes (middle and bottom).

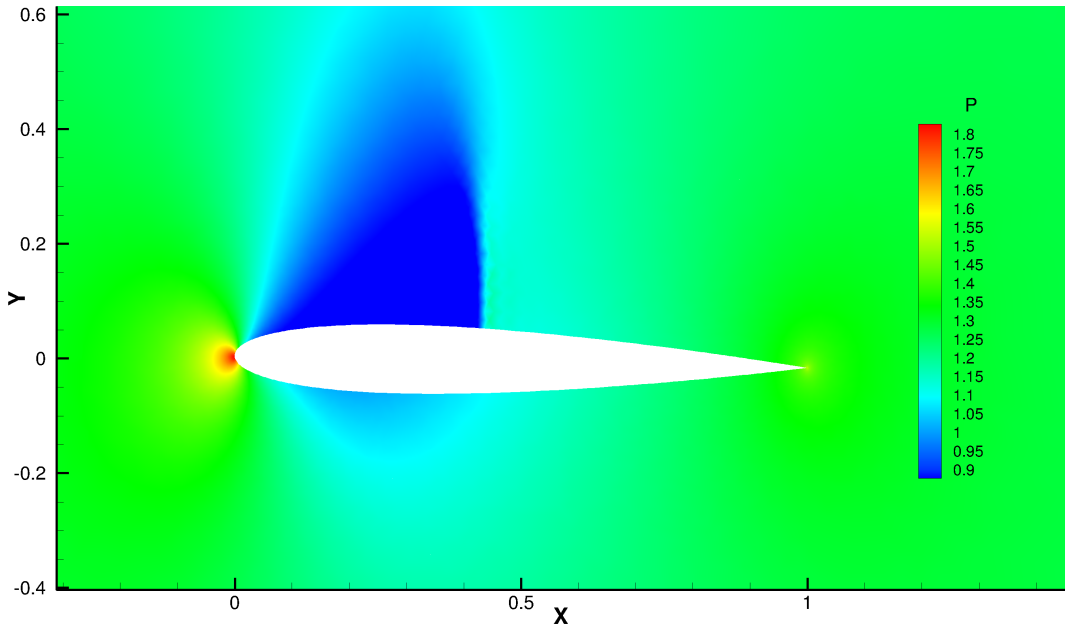

Figure 3. Non-dimensionalized pressure contours for $M_\infty = 0.755$ and $\alpha = 1.25$.

0.3 and 0.00045, respectively, that is, we attempt a lift constrained drag reduction. We must use bound constraints on the design variables to prevent the generation of invalid geometries from the mesh movement algorithm. Using L-BFGS-B as a single level optimizer requires 254 function and gradient evaluations to converge and the initial NACA 0012 airfoil and optimized airfoil are shown in Figure 4.

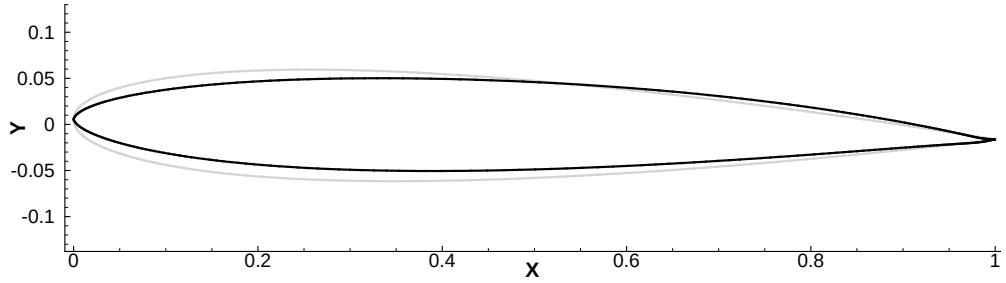

Figure 4. Initial NACA 0012 airfoil (in gray) and inverse designed airfoil with $C_l^* = 0.3$ and $C_d^* = 0.00045$.

The settings for FMG/OPT are exactly the same as described in Section III, except that we use the smoother for the optimization on the coarsest level as well with a maximum of ten iterations. Using FMG/OPT, we are not able to reduce the number of function and gradient evaluations on the finest level below the number required by the single level L-BFGS-B optimizer. However, if we compare the results using SD as single level optimizer and as smoother/optimizer in FMG/OPT (using three multigrid levels) one can see in Figure 5 that FMG/OPT is clearly superior in terms of total number of function and gradient evaluations on the finest level ("tnfg"). In addition, FMG/OPT takes about three times "tnfg" function and gradient evaluations on the medium level and about twenty times "tnfg" on the coarsest level. Since the computational costs are approximately a factor of four and sixteen cheaper, respectively, this adds "only" about 200 per cent to the overall computational cost.

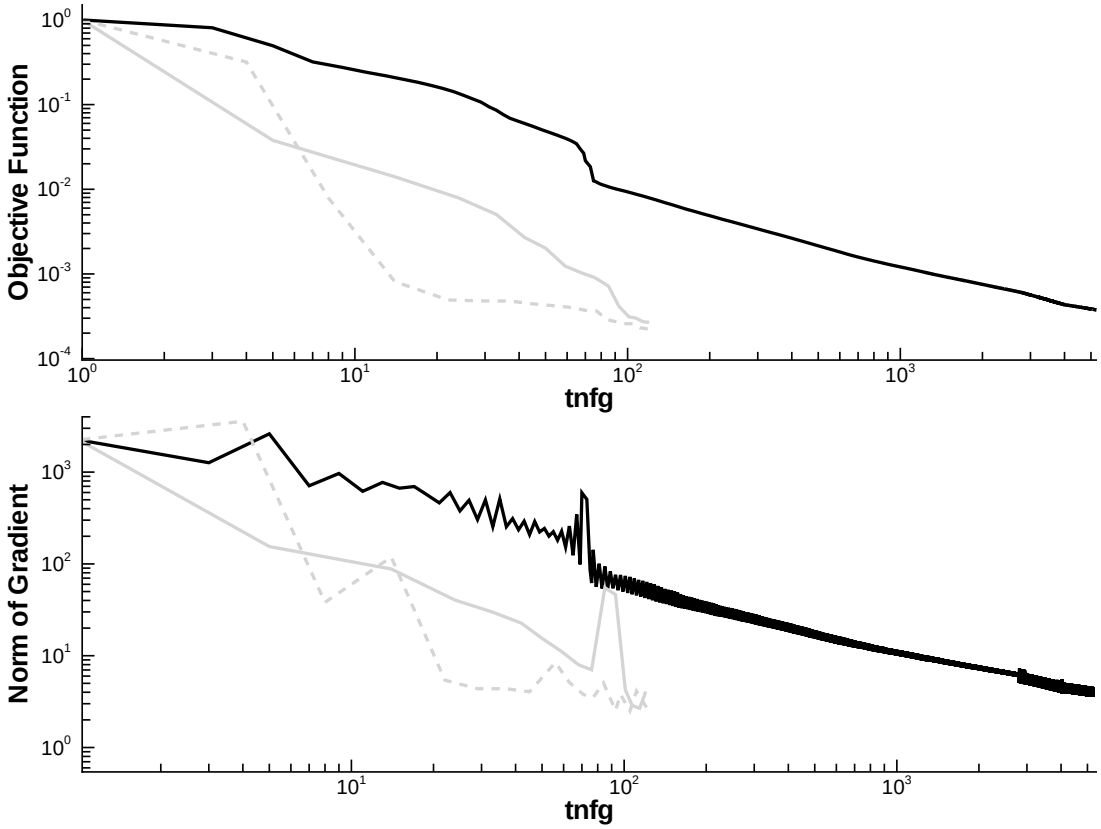

Figure 5. Convergence plots for pure SD (solid black) and FMG/OPT with 3 multigrid levels and SD as smoother/optimizer (solid gray using constant number of design variables and dashed gray using multigrid for the number of design variables).

V. Conclusions

An efficient general optimization-based multigrid algorithm is successfully applied to an elliptic model problem and to an aerodynamic shape design. The application of the algorithm to the elliptic model problem shows a great improvement in the overall computational cost as compared to only using L-BFGS-B or a steepest descent optimizer with a line search. When applied to a steady-state aerodynamic optimization problem, the multigrid optimization approach shows substantial improvement over simple single level optimizers such as steepest-descent, although the performance of more sophisticated optimizers such as L-BFGS-B remain difficult to improve upon. Future work will investigate the high-frequency smoothing properties of various single level optimizers and examine alternate design variable restriction and prolongation strategies with the goal of developing a robust multilevel optimization approach which delivers convergence for the optimization problem which is independent of the number of design variables.

Acknowledgments

We are very grateful to Karthik Mani for making his flow and adjoint solver available to us.

References

[1]Jameson, A., "Optimum Aerodynamic Design Using Control Theory," *Computational Fluid Dynamics Review, Hafez, M., Oshima, K. (eds)*, Wiley: New York, pp. 495-528, 1995.

[2]Nadarajah, S. and Jameson, A., "Optimum Shape Design for Unsteady Flows with Time-Accurate Continuous and Discrete Adjoint Methods," *AIAA Journal*, Vol. Vol. 45 No. 7, 2007, pp. 1478–1491.

[3]Nemec, M. and Zingg, D. W., "Multipoint and Multi-Objective Aerodynamic Shape Optimization," *AIAA Journal*, Vol. Vol. 42, No. 6, 2004, pp. 1057–1065.

[4]Rumpfkeil, M. P. and Zingg, D. W., "The Optimal Control of Unsteady Flows with a Discrete Adjoint Method," *Optimization and Engineering*, doi:10.1007/s11081-008-9035-5, 2008.

[5]Nash, S. G., "A Multigrid Approach to Discretized Optimization Problems," *Journal of Optimization Methods and Software*, Vol. Vol. 14, 2000, pp. 99 – 116.

[6]Lewis, R. M. and Nash, S. G., "A Multigrid Approach to the Optimization of Systems Governed by Differential Equations," AIAA, 2000-4890, 2000.

[7]Lewis, R. M. and Nash, S. G., "Model Problems for the Multigrid Optimization of Systems Governed by Differential Equations," *SIAM Journal on Scientific Computing*, Vol. Vol. 26, No. 6, 2005, pp. 1811 – 1837.

[8]Gratton, S., Sartenaer, A., and Toint, P., "Numerical experience with a recursive trust-region method for multilevel nonlinear optimization," Tech. rep., Dept of Mathematics, FUNDP, Namur (B), 2006.

[9]Gratton, S., Sartenaer, A., and Toint, P., "Second-order convergence properties of trust-region methods using incomplete curvature information, with an application to multigrid optimization," *Journal of Computational Mathematics*, Vol. Vol. 24, 2006, pp. 676 – 692.

[10]Gratton, S., Sartenaer, A., and Toint, P., "Recursive trust-region methods for multiscale nonlinear optimization," *SIAM Journal of Optimization*, Vol. Vol. 19, 2008, pp. 414 – 444.

[11]Wen, Z. and Goldfarb, D., "A Line Search Multigrid Method for Large-Scale Convex Optimization," Tech. rep., Department of IEOR, Columbia University, 2007.

[12]Hazra, S. B. and Schulz, V., "Simultaneous pseudo-timestepping for PDE-model based optimization problems," *BIT*, Vol. Vol. 44, 2004, pp. 457 – 472.

[13]Hazra, S. B. and Jameson, A., "One-Shot Pseudo-Time Method for Aerodynamic Shape Optimization Using the Navier-Stokes Equations," AIAA, 2007-1470, 2007.

[14]Nash, S. G. and Sofer, A., *Linear and Nonlinear Programming*, McGraw-Hill, New York, 1996.

[15]Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C., "A Limited Memory Algorithm for Bound Constrained Optimization," *SIAM Journal on Scientific Computing 16*, Vol. Vol. 5, 1995, pp. 1190–1208.

[16]Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J., "L-BFGS-B: A Limited Memory FORTRAN Code for Solving Bound Constrained Optimization Problems," Tech. Rep. NAM-11, EECS Department, Northwestern University, 1994.

[17]Byrd, R. H., Nocedal, J., and Waltz, R. A., "KNITRO: An Integrated Package for Nonlinear Optimization," *Large-Scale Nonlinear Optimization, di Pillo, G., Roma, M.(eds)*, Springer-Verlag, pp. 35-59, 2006.

[18]Mani, K. and Mavriplis, D. J., "An Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," AIAA, 2007-60, 2007.

[19]Mani, K. and Mavriplis, D. J., "Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," *AIAA Journal*, Vol. Vol. 46 No. 6, 2008, pp. 1351–1364.

[20]Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," *AIAA Journal*, Vol. Vol. 28, No. 8, 1990, pp. 1381 – 1388.

[21]Hicks, R. and Henne, P., "Wing Design by Numerical Optimization," *Journal of Aircraft*, Vol. Vol. 15, No. 7, 1978, pp. 407 – 412.