Multigrid One-Shot Methods and Design Strategy

Shlomo Ta'asan
Carnegie Mellon University

1 Introduction

In this lecture we discuss the basic ideas regarding fast multigrid solutions of optimization problems governed by large systems. We give a short review of the essential multigrid concepts and discuss their adaption to the solution of optimization problems. We discuss the necessity of using adjoint variables (Lagrange multipliers) for defining proper coarse grid minimization problem that accelerate the fine grid minimization process. A central issue is the design of relaxation schemes for optimization problem which have the smoothing property. We discuss $h - ellipticity$ and its role in fast solvers for optimization problems. Several examples demonstrating the main issues are included. We distinguish different cases of design spaces. A small dimensional design space, a moderate size design space and the infinite dimensional design space. The multigrid treatment of each of these cases is slightly different.

2 Review of Multigrid Basics

The two main elements of a multigrid solver are a relaxation scheme and a coarse grid correction. The relaxation is usually a local process that for elliptic problems it has the smoothing property. That is, high frequency errors decay fast as a result of relaxation. Low frequency errors, on the other hand, are slow to converge and an acceleration is required for them. The coarse grid correction serve exactly for that purpose. In standard multigrid method the coarse grid approximate the smooth part of the fine grid error. Since relaxation smoothes the error, applying a coarse grid correction after a few relaxation sweeps results in an effective reduction of the error for all components.

For nonlinear problems there is a version of multigrid called FAS (Full Approximation Scheme) that does not use linearization in the coarse grid correction. This version of multigrid will be used later for our optimization techniques.

2.1 Full Approximation Scheme (FAS)

Let the fine grid equations be written as

$$L^h(u^h) = f^h. \tag{1}$$

The FAS equations defined on the coarse grid are

$$L^H(u^H) = f^H \tag{2}$$

where,

$$f^H = I_h^H f^h + L^H(I_h^H \bar{u}^h) - I_h^H L^h(\bar{u}^h), \tag{3}$$

and where $\bar{u}^h$ denotes an approximation to u^h, the exact solution of the fine grid problem. The coarse grid correction is done as

$$\bar{u}^h \leftarrow \bar{u}^h + I_H^h[u^H - I_h^H \bar{u}^h]. \tag{4}$$

Note that $I_h^H u^h$ is a solution of the coarse grid equations. This means that once the solution of the fine grid problem was obtained, the coarse grid correction does not introduce any changes via interpolation. We regard this property as an essential one and in our derivation of the coarse grid optimization problem we make sure that it is satisfied.

2.2 Smoothing and h-ellipticity

One of the important concepts to understand with regard to efficient multigrid solvers is that of h-ellipticity Brandt [14]. Elliptic problems have the property that high frequencies are local. That is, if we make a change in the right hand side with a high frequency, the change in the solution is a high frequency change and is localized to the vicinity of the right hand side change. This property follows from the fact that the symbol of the operator L defined by

$$L \exp(i\mathbf{k} \cdot \mathbf{x}) = \hat{L}(\mathbf{k}) \exp(i\mathbf{k} \cdot \mathbf{x}) \tag{5}$$

satisfies

$$|\hat{L}(\mathbf{k})| \geq C|\mathbf{k}|^{2m}, \tag{6}$$

where $\mathbf{x} = (x_1, \ldots, x_n), \mathbf{k} = (k_1, \ldots, k_n)$. The meaning of this is that the symbol attains high values for high frequencies and small values for low frequencies.

A discrete symbol $\hat{L}^h(\theta)$ for a discretization L^h of L is defined by

$$L^h \exp(i\theta \cdot \mathbf{x}/h) = \hat{L}^h(\theta) \exp(i\theta \cdot \mathbf{x}/h) \tag{7}$$

where $\theta = (\theta_1, \ldots, \theta_n)$, and $|\theta| = max_j|\theta_j| \leq \pi$. Discretization of elliptic problems may lead to one of the following analog properties of ellipticity for the symbol of the discretized problem,

$$\texttt{h-ellipticity:} \qquad L^h(\theta)| \geq C \sum_{j=1}^{d} sin(\theta_j/2)^{2m} \tag{8}$$

or

$$\texttt{quasi-ellipticity:} \qquad L^h(\theta)| \geq C \sum_{j=1}^{d} sin(\theta_j)^{2m} \tag{9}$$

where on the discrete level we consider the discrete transform as

$$\hat{U}(\theta) = \sum_{\beta} \exp(i\beta \cdot \theta) U_\beta. \tag{10}$$

In (10) β stands for a multi-index and the grid function U_β is defined in infinite space.

It is well known that h-ellipticity is required for simple efficient multigrid methods, see Brandt [14]. Actually it can be shown that for such schemes one can construct efficient smoothers. Such smoothers together with a coarse grid correction will result in efficient multigrid solvers. We will come to this when we discuss optimization and the role of h-ellipticity there.

To give the simplest examples for these two types of discretization we take the Laplace equation in two dimensions and the following two discretization

h-elliptic discretization. The standard 5-point discretization of the Laplacian

$$L^h = \frac{1}{h^2} \begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} \tag{11}$$

has a symbol

$$\hat{L}^h(\theta) = -\frac{4}{h^2}(\sin^2(\theta_1/2) + \sin^2(\theta_2/2)) \tag{12}$$

which is certainly h-elliptic.

quasi-elliptic discretization. The skew Laplacian given by

$$L^h_x = \frac{1}{2h^2} \begin{bmatrix} 1 & & 1 \\ & -4 & \\ 1 & & 1 \end{bmatrix} \tag{13}$$

has a symbol

$$\hat{L}^h_x(\theta) = -\frac{2}{h^2}[\sin^2(\theta_1/2)\cos^2(\theta_2/2) + \sin^2(\theta_2/2)\cos^2(\theta_1/2)]. \tag{14}$$

Due to the terms $\cos(\theta_1/2), \cos(\theta_2/2)$ this symbol vanishes for the frequency (π, π) and hence it is not h-elliptic.

3 Multigrid Approaches for Optimization Problems

We focus our discussion on gradient based methods for constrained optimization problems. The expensive part in such methods is the equation solver per each optimization step (and in addition to this the solver for the adjoint equation in the adjoint method). Although these equations may be solved only approximately, still they consist the major part of the solution time.

Multigrid methods can be used in a variety of ways to accelerate gradient based optimizers. We distinguish between multigrid methods that are used for the acceleration of a step in some optimization procedure to genuinely multigrid methods that aim at the full optimization problem.

3.1 Multigrid Methods for the Constraints

Steepest Descent and BFGS methods. A natural way to use multigrid to accelerate optimization procedures is to accelerate the solver parts for both the state and the adjoint equations. In such a case the number of optimization iteration required to solve the problem will remain unchanged, but each optimization step will require significantly less computer time due to the fast solver. This approach is straightforward and does not requires additional analysis into the numerical processes. This approach may be viewed as a loosely coupled optimization procedure, in which the solvers are separated from the optimizer. It works with any gradient based method, either a steepest descent or a quasi-Newton method such as BFGS.

```
ALGORITHM

(1) Solve the State equation using a Multigrid solver
(2) Solve the adjoint equation using a Multigrid solver
(3) Calculate Gradients
(4) Update Design Variables
```

Multigrid for infinite dimensional preconditioners. Infinite dimensional preconditioners can be extremely efficient as was explained in a previous lecture. In that approach one uses a preconditioner which is a partial differential operator in 3D, or an ODE in 2D design problems. Sometime a pseudo-differential preconditioner may also result, such as in transonic calculations with either the full potential or the Euler equations. In any case, in order to apply the desired preconditioner one has to solve a large scale sparse system. This system is defined on the boundary of the original domain for boundary control and shape design problems. In addition to this there is also the state and costate equations that needs to be solved per each optimization iteration.

Multigrid can be used in these type of methods as a fast solver for the state equations, adjoint equations and the equation for the preconditioner step. These multigrid solvers for the different parts can be completely independent and actually one may be replaced by some other fast solver. A typical algorithm may looks as follows,

```
ALGORITHM

(1) Solve the State equation using a Multigrid solver
(2) Solve the adjoint equation using a Multigrid solver
(3) Calculate Gradients
(4) Apply Infinite dimensional Preconditioner Using a Multigrid Solver
(5) Update Design Variables
```

3.2 Multigrid for the Full Optimization Problem

Another approach to use multigrid method for the efficient solution of constrained optimization problems is to use it for the full problem. This means not to accelerate an already existing algorithm by accelerating one of its step but to use multigrid in a genuine way. The idea is to construct a relaxation process for the optimization problem that will have the smoothing property. That is, high frequency in the design variables will be fast to converge with this relaxation. Smooth components of the design variables will be accelerated by coarse grids. This requires the construction of a sequence of optimization problems given on a sequence of coarse grids and a proper interaction between the grids that will achieve the fast solution process required.

One way to look at the solution process for the optimization problem is as a solver for the necessary conditions. These are usually a set of nonlinear partial differential equation. It is expected that one can construct efficient solver for such a system. The general framework is therefore the following. We have a relaxation method for the state, costate and design variables which is a smoother (for all). The objective is to construct what we call a *one-shot multigrid method*. That is, a method that solves the full optimization problem in a computation cost which is 2-3 times that of solving the constraints. This efficiency should be independent of the number of design variables.

A two level algorithm has the form outlined below. A multilevel version of this method is essentially a recursive application of it.

```
   Algorithm:  Two Level
(1) Relax (smooth) the state, adjoint and design variables
(2) Accelerate convergence using a coarse grid optimization problem
```

The rest of this lecture will be devoted to the one-shot multigrid methods.

4 One-Shot Multigrid Methods

The main issues of focus in the development of the multigrid one-shot methods are h-elliptic discretization, relaxation with the smoothing property and a coarse grid acceleration. Discretization for the partial equation equations, such as the state and the adjoint equations is a well understood subject, and we assume that an h-elliptic discretization has been done for those equations. As far as discretization introduced for the design space there are some points to be discussed here.

4.1 Discretization Issues

In order to have a multigrid solver for the full optimization problem with high efficiency, one has to guarantee that certain operators on the discrete level are h-elliptic. In particular, the Hessian which is a symmetric positive definite operator should have a discretization which is h-elliptic. The Hessian may be a pseudo differential operator and its symbol may no longer be a trigonometric polynomial. So we have to change slightly the definition of h-ellipticity to include that case. Thus,

$$\texttt{h-ellipticity:} \qquad \hat{\mathcal{H}}^h(\theta) \geq C \sum_j |sin(\theta_j/2)|^m \tag{15}$$

$$\texttt{quasi-ellipticity:} \qquad \hat{\mathcal{H}}^h(\theta) \geq C \sum_j |sin(\theta_j)|^m \tag{16}$$

To demonstrate the difficulty that may arise with discretization we consider the following example.

Example I: A Control Problem Let ϕ be the state variable, α the design variable and consider the minimization problem,

$$\min_{\alpha} \frac{1}{2} \int_{\partial\Omega} (\frac{\partial\phi}{\partial n} - d)^2 ds \tag{17}$$

where ϕ satisfies

$$\begin{array}{ll} \Delta\phi = 0 & \Omega \\ \phi = \alpha & \partial\Omega \end{array} \tag{18}$$

and where $\Omega = \{(x,y)|y > 0\}$. A simple calculation shows that the necessary (optimality) conditions are given by

$$\begin{array}{ll} \Delta\phi = 0 & \Omega \\ \Delta\lambda = 0 & \Omega \\ \phi = \alpha & \partial\Omega \\ \lambda + \frac{\partial\phi}{\partial n} = d & \partial\Omega \\ \frac{\partial\lambda}{\partial n} = 0 & \partial\Omega. \end{array} \tag{19}$$

Following a calculation as in a previous lecture we obtain that the symbol of the Hessian is given by

$$\hat{\mathcal{H}}(\mathbf{k}) = |\mathbf{k}|^2, \tag{20}$$

which is an elliptic symbol. Consider now the following discretizations.

Discretization I: h-elliptic Hessian. A uniform grid with *vertex discretization* is used with the usual 5-point Laplacian

$$L^h = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}. \tag{21}$$

The discrete approximation to ϕ are parameterized as $\phi_{l,k}$ where (l,k) is a grid point. For discretization of the normal derivative at the boundary we use a ghost point with a central discretization

$$\frac{\partial \phi}{\partial n} \approx \frac{1}{2h}(\phi_{l,1} - \phi_{l,-1}) \tag{22}$$

where $k = 0$ stands for the boundary, and $k = -1$ represent ghost points. The above formula is used in discretization of the cost functional. The actual implementation is to add one equation for the ghost point, by approximating the interior equation also on the boundary. Thus, relating the ghost values to other values in the interior.

The design variables α are taken at the grid vertices. A Fourier change in α by $\exp(i\theta l)$ introduce the following changes

$$\phi_{l,j} = \exp(i\theta l) d(\theta)^j \tag{23}$$

$$\lambda_{l,j} = A\exp(i\theta l) d(\theta)^j \tag{24}$$

where $d(\theta) \leq 1$ is the solution of

$$d(\theta) + 1/d(\theta) - 4 + 2cos(\theta) = 0. \tag{25}$$

Note that if $d(\theta)$ is a solution, then so is $1/d(\theta)$. The first correspond to a bounded solution in half space while the other is unbounded. Define

$$\mu(\theta) = \frac{1}{2h}[1/d(\theta) - d(\theta)] \tag{26}$$

and observe that $\mu(\theta)$ is the symbol for normal derivative at the boundary. A simple calculation, as was done previously, shows that the symbol for the discrete Hessian is

$$\hat{\mathcal{H}}^h(\theta) = \mu^2(\theta). \tag{27}$$

To see that this is indeed h-elliptic notice that $d(\pi) = (6 - \sqrt{32})/2$, and thus it is bounded away from zero.

Discretization II: Quasi-elliptic Hessian. Here we consider a *cell-centered scheme*, i.e., the variables approximating ϕ are located at the center of the cells and the control variables α_l are located at the boundary grid points. The discretization for the interior points is the same as before. The boundary condition also here uses ghost points with a central discretization for Neumann boundary condition. If we denote the grid points by (l,k) then the ϕ variable are $\phi_{l+\frac{1}{2},k+\frac{1}{2}}$ The contribution of α to the boundary condition at a general point in the bottom boundary is

$$\frac{1}{2}(\phi_{l+\frac{1}{2},\frac{1}{2}} + \phi_{l+\frac{1}{2},-\frac{1}{2}}) = \frac{1}{2}(\alpha_l + \alpha_{l+1}) \tag{28}$$

where $k = 0$ are the boundary points and $\phi_{l,-1/2}$ are the ghost variables. The Neumann derivative is approximated by

$$(\frac{\partial \phi}{\partial n})_{l+1/2,0} \approx \frac{1}{h}(\phi_{l+1/2,1/2} - \phi_{l+1/2,-1/2}) \tag{29}$$

and the corresponding symbol is

$$\mu_c(\theta) = \frac{1}{h}(\frac{1}{\sqrt{d(\theta)}} - \sqrt{d(\theta)}). \tag{30}$$

Defining

$$C(\theta) = \frac{1}{\sqrt{d(\theta)}} + \sqrt{d(\theta)} \tag{31}$$

the solution is given by

$$\phi_{l,j} = \frac{\cos(\theta/2)}{C(\theta)} \exp(i\theta l) d(\theta)^j \tag{32}$$

$$\lambda_{i,j} = \mu_c(\theta)\frac{\cos(\theta/2)}{C(\theta)} \exp(i\theta l) d(\theta)^j. \tag{33}$$

Again a simple calculation shows that symbol of the Hessian is

$$\hat{\mathcal{H}}(\theta) = \mu_c^2(\theta)\frac{\cos(\theta/2)}{C(\theta)}. \tag{34}$$

Since $cos(\theta/2)$ vanishes at π the symbol is not h-elliptic, but only quasi-elliptic. This means that no local relaxation of the design variables will result in smoothing. That is, high frequency errors in the design variable will not be damped fast. A standard multigrid method will not work for this discretization. In this example one could guess the quasi-ellipticity of the Hessian from the fact that perturbation in the design variable of the form $(-1)^l$ do not cause any changes in ϕ.

Example II: Consider the optimization problem

$$\min_{\alpha} \frac{1}{2} \int_{\partial\Omega} (\phi - d)^2 ds \tag{35}$$

$$\tag{36}$$

subject to the equation,

$$\begin{array}{ll} \Delta\phi = 0 & \Omega \\ \frac{\partial\phi}{\partial n} = \alpha_x & \partial\Omega \end{array} \tag{37}$$

where $\Omega = \{(x, y)|y > 0\}$. A simple calculation shown that the necessary conditions are given by

$$\begin{array}{ll} \Delta\phi = 0 & \Omega \\ \Delta\lambda = 0 & \Omega \\ \frac{\partial\phi}{\partial n} = \alpha_x & \partial\Omega \\ \frac{\partial\lambda}{\partial n} = \phi - d & \partial\Omega \\ -\lambda_x = 0 & \partial\Omega \end{array} \tag{38}$$

and the symbol of the Hessian is

$$\hat{\mathcal{H}}(\mathbf{k}) = 1. \tag{39}$$

Discretization I: h-elliptic Hessian. We use here *cell-centered* discretization for the state and costate and the design variables are at the boundary nodes. The approximation for the tangential derivative is

$$(\frac{\partial\alpha}{\partial x})_{l+1/2} \approx \frac{1}{h}(\alpha_{l+1} - \alpha_l) \tag{40}$$

whose symbol is given by

$$s(\theta) = \frac{2i}{h}\sin(\theta/2) \tag{41}$$

The normal derivative is the same as in discretization II of example I. A change in the design variable by a Fourier component $\exp(i\theta l)$ results in the following changes in state and costate

$$\begin{array}{l} \phi_{l,k} = \frac{s(\theta)}{\mu_c(\theta)} \exp(i\theta l) d(\theta)^j \\ \lambda_{l,k} = \frac{s(\theta)}{\mu_c^2(\theta)} \exp(i\theta l) d(\theta)^j \end{array} \tag{42}$$

and the symbol of the Hessian is

$$\hat{\mathcal{H}}^h(\theta) = \frac{4\sin^2(\theta/2)}{\mu_c^2(\theta)}. \tag{43}$$

We gave this example to show that although the symbol of the Hessian for the differential level was just the constant 1, on the discrete level the Hessian is more complicated. Taylor expansion of this symbol at $\theta = 0$ shows that it indeed approximate the PDE. Note however, that the Hessian has a condition number which is independent of h. The ratio of largest to smallest value that the symbol attains is independent of h. This means that this problem presents no special difficulties even for very large number of design variables.

Discretization II: Quasi-elliptic Hessian. Here we discretized the problem using *cell-vertex* variables for the state and adjoint variables. The design variables are given at the boundary nodes. A simple calculation shows that

$$\hat{\mathcal{H}}^h(\theta) = \frac{\sin^2(\theta)}{\mu^2(\theta)} \tag{44}$$

and this is a quasi-elliptic symbol, due to the $sin^2(\theta)$ term, which vanishes at $\theta = \pi$.

4.2 Coarse Grid Optimization Problems

Having discussed the main issues regarding discretization and h-ellipticity for optimization problems we move to the next major issue. The formulation of appropriate coarse grid equation that will serve to accelerate the fine grid optimization problem. We show that it is necessary in general to work with the adjoint variables (Lagrange multipliers) in order to define a proper coarse grid problem.

Consider the minimization problem

$$\begin{gathered} \min_{\alpha^h} E^h(u^h, \alpha^h) \\ \text{subject to} \\ L^h(u^h, \alpha^h) = f^h. \end{gathered}$$

At the minimum the following equation holds (see the first lecture)

$$\begin{aligned} L^h(u^h, \alpha^h) &= f^h \\ L_u^{h*}(u^h, \alpha^h)\lambda + E_u^h &= 0 \\ L_\alpha^h\,(u^h, \alpha^h)\lambda^h + E_\alpha^h &= 0. \end{aligned}$$

Next consider an attempt to define a coarse grid minimization problem that will accelerate the convergence of the fine grid above. We will show that without the use of Lagrange multipliers this is in general impossible. An argument for a quadratic functional and linear constraint is given in Ta'asan [7]. Consider the coarse grid minimization problem

$$\begin{gathered}\min_{\alpha^H} E^H(u^H,\alpha^H)\\ \text{subject to}\\ L^H(u^H,\alpha^H)=f^H.\end{gathered}$$

The necessary conditions for this problem are

$$\begin{aligned}L^H(u^H,\alpha^H)&=f^H\\ L_u^{H*}(u^H,\alpha^H)\lambda^H+E_u^H(u^H,\alpha^H)&=0\\ L_\alpha^H(u^H,\alpha^H)\lambda^H+E_\alpha^H(u^H,\alpha^H)&=0.\end{aligned}$$

In order for this coarse grid equation to accelerate the fine grid solution process we must have the property that if the fine grid equations are solved, the coarse grid will not introduce any change to the fine grid solution. This means that the coarse grid function $I_h^H u^h, \bar{I}_h^H \alpha^h$ must satisfy the coarse grid equation (Recall that the interpolation step is using the correction $u^H - I_h^H u^h, \alpha^H - \bar{I}_h^H \alpha^h$). From the first equation we see that we must have $f^H = L^H(I_h^H u^h, \bar{I}_h^H \alpha^h)$, and from the second and third coarse grid necessary condition we see that there must exists a function λ^H that satisfies simultaneously the two equation $L_u^{H*}(I_h^H u^h, \bar{I}_h^H \alpha^h)\lambda^H = -E_u^H(I_h^H u^h, \bar{I}_h^H \alpha^h)$ and $L_\alpha^H(I_h^H u^h, \bar{I}_h^H \alpha^h)\lambda^H = -E_\alpha^H(I_h^H u^h, \bar{I}_h^H \alpha^h)$. But since the the two right hand sides are independent, in general, we have twice the number of equations than the number of unknowns in λ^H. Thus, in general $I_h^H u^h, \bar{I}_h^H \alpha^h$ will not be the solution of the coarse grid minimization problem.

The correct way to define the coarse grid minimization problem is to start with the FAS equations for the necessary conditions. Thus, we get

$$\begin{gathered}L^H(u^H,\alpha^H)=f^H\\ L_u^{H*}(u^H,\alpha^H)\lambda^H+E_u^H(u^H,\alpha^H)=g_1^H\\ L_\alpha^H(u^H,\alpha^H)\lambda^H+E_\alpha^H(u^H,\alpha^H)=g_2^H\end{gathered}$$

where f^H, g_1^H, g_2^H are the FAS transfers of the necessary conditions. The corresponding minimization problem whose necessary conditions are the above equation is the proper choice for the coarse grid optimization problem. This leads to the following coarse grid optimization problem

$$\begin{gathered}\min_{\alpha^H} E^H(u^H,\alpha^H) - <g_1^H,u^H> - <g_2^H,\alpha^H>\\ \text{subject to}\\ L^H(u^H,\alpha^H)=f^H.\end{gathered}$$

Note that in this formulation the functional has two extra terms g_1^H, g_2^H which depend on the fine grid adjoint variables (Lagrange multipliers) through the FAS transfers for g_1^H, g_2^H.

4.3 Relaxation

We come now to the last issue which is the construction of a relaxation method for the optimization problem, with a property that it smoothes the error. This together with a coarse grid correction which effectively reduce the low frequency errors will result in a one-shot multigrid method for the optimization problem.

For the solution of elliptic differential equations it is well known that the discrete system should obey the so called $h-ellipticity$ property. Roughly speaking it means that high frequency are magnified significantly by the action of the discrete operator. This may not be true for any discretization of elliptic equations as we saw in the examples of a previous section.

If we want to construct a multigrid method for the full optimization problem we have to understand the role of relaxation in that process and how to construct the proper smoothers. The relaxation will have in general three steps,

```
Relaxation

1.  Relax the state variable
2.  Relax the adjoint variable
3.  Relax the design variables
```

The first two steps here are some standard relaxation for the state and the adjoint variables which we do not discuss here. The last step which involves the update of the design variables is of a special interest. This is due to the requirement that we want to achieve a one-shot solver, meaning that the total computer time for the full optimization problem will not be larger than solving the state equations just a few times. Notice that this step requires updating the state variable as well as the adjoint variable.

In order to achieve an efficient solver we must have those updates being done very efficiently. For example, by using one or two relaxations only, and actually better if done with less than that. For a variety of problems this is indeed possible. One class of such problems is the one in which design variables introduce smooth changes in the state and adjoint variables, enabling their update on those coarse levels. Such updates on coarse levels require only a small fraction of the computer time to relax the state or costate variables on the finest level of discretization, thus resulting in fast updates for design variables. Another class of problems is that of boundary control and boundary shape design for elliptic problems. In such problems the changes in the design variables produce changes in the state and adjoint variable which are localized to a vicinity of the boundary. In that case the updates of the state and adjoint variables are done by local relaxation near the boundary only, and its cost is only a fraction of a full relaxation for the state or adjoint equation.

4.4 Design Space of a Small Dimension

This is the simplest of all cases and was developed in Ta'asan [7]. The assumption is that the design space is of a small dimension and moreover, it is such that its effect on the solution is smooth. That is, a change in the design variables produces a relatively smooth change in the solution, that can be approximated using the coarsest grid. If the design variables describe the shape of a boundary to be designed, we may have a representation of the form

$$\alpha(x) = \sum \alpha_j f_j(x), \tag{45}$$

where the functions f_j can be represented on the coarsest grid.

This suggests to update the design variables on the coarse grid only. The question is how to perform the optimization of the design variables on coarse levels but to have convergence toward the finest grid solution? The answer is in our coarse grid optimization problem. The coarse grid functional is such that, at convergence of the solution is that of the fine grid problem, due to the FAS formulation we have mentioned.

On all grids we perform relaxation to smooth the errors in the state and costate variables. On the coarsest grid *only*, we solve the optimization problems which depends, through the FAS transfers, on the finer level optimization problem. On that level we may perform any optimizer of out choice, for example, a quasi-Newton method such as BFGS. The coarse grid iterations are done until convergence of its optimization problem.

```
RELAXATION: Level l

(1) Relax State Equation
(2) Relax Adjoint Equation
(3) If level = 1 Update Design Variables by BFGS
```

4.5 Design Space of Moderate Size

In this case one consider more variables in the design space such that the assumption about the smoothness can no longer be true. That is, some design variables may have a smooth effect on the solution, but since there are sufficiently many design variables, the space of changes in the solution is also large and hence must contain higher frequency that cannot be represented on the coarsest level. Some intermediate levels, (still much coarse than the finest level) have to be included in the optimization process. This case was developed by Ta'asan, Salas and Kuruvila in [10],[11].

One approach to achieve this is to change the basis representing the design space such that one is using orthonormal decomposition. For the shape design problem it means that we use a basis of shape functions $g_{m,l}$ where the representation of the shape is

$$\alpha(x) = \sum_{m=1}^{m_l} \sum_{l=1}^{n_d} \beta_{m,l} g_{m,l}(x) \tag{46}$$

and where the index l stands for the multigrid level, and the changes of state and adjoint variables corresponding to changes in $\beta_{m,l}, m = 1, \ldots, m_l$ can be approximated on this level l.

The idea is that these new shape functions, $g_{m,l}$ are going to be of increasing oscillations, due to their orthogonality. This implies that their effect on the solution will also be of increasing oscillations. Thus, we achieve a decomposition of the design space into subspaces of different smoothness as far as its effect on the solution. The implication of this is that a natural preconditioner of the optimization process is done. Since each level of those coarse grids on which optimization is to be done works independently of the other, at the relaxation step, we have removed the bad conditioning from the problem using this decomposition. The relaxation of the design variables is done now for the $\beta_{m,l}$ instead of the original α_j. On level l we update m_l of the total design variables, $\beta_{m,l}, m = 1, \ldots, m_l$.

`Relaxation:  Level` l

`(1) Relax State Equation`
`(2) Relax Adjoint Equation`
`(3) Update design variables` $\beta_{m,l}, m = 1, \ldots, m_l$

4.6 Infinite Dimensional Design Space

The most challenging case is the infinite dimensional design case. It include all the possible difficulties one may encounter in an optimization problem. Our main goal here is to construct a relaxation that will smooth the design variables. That is, make high frequency changes in the design variables on fine grids, and low frequency changes on coarse grids. On each grid we want to update the design variables only in the scales that are high frequency for that level. In this way we split the design process on all levels of discretization. These ideas were developed by Arian and Ta'asan in [1],[2],[3].

In order to look for changes in the design variables we use the gradient, which is calculated using the adjoint method. Given an error in the design variables, one may ask the question how does the corresponding gradient look like. That is, suppose that our error is mainly a high frequency. Is it true or not that also the gradient will be dominated by high frequencies. The answer to this is given by the Hessian and we know how to analyze it using Fourier techniques as discussed in the previous lectures.

We consider example I of section 4.1. For the quasi-elliptic discretization we know that no local relaxation (i.e., a relaxation based on the gradient) can have the smoothing property, since the gradient does not 'feel' the oscillatory parts in the errors of the design variables. Hence such errors cannot be corrected efficiently by using the gradient. The h-elliptic discretization in that example had a symbol $\mu_c^2(\theta)$, and if we consider the expression

$$1 - \delta \hat{\mathcal{H}}^h(\theta) = 1 - \delta \mu_c^2(\theta) \tag{47}$$

it is easy to see that by choosing, for example,

$$\delta = 2./[\hat{\mathcal{H}}^h(\pi/2) + \hat{\mathcal{H}}^h(\pi)] \tag{48}$$

one achieve a good reduction for all high frequencies. It is important to notice that the low frequencies are not changes much in this relaxation since $\hat{\mathcal{H}}^h(\theta)$ vanishes for $\theta = 0$.

In solving equations it is not so important to keep the low frequencies unchanged in the relaxation. It just gives the coarse levels an easier job. Here is it crucial not to introduce low frequency changes on the fine grids. The reason is that the corresponding effect on the state and costate has to be computed. When these changes are smooth this is not an inexpensive computation. Only the high frequency effect can be computed fast, due to the smoothing and to its localization to a vicinity of the boundary in elliptic problems.

In example II, the quasi-elliptic case does not have an efficient treatment, as was in example I. The h-elliptic discretization in that example had a symbol

$$\hat{\mathcal{H}}^h(\theta) = \frac{\sin^2(\theta/2)}{\mu_c^2(\theta)} \tag{49}$$

with the property

$$0 < c_1 \leq \hat{\mathcal{H}}^h(\theta) \leq c_2 \qquad\qquad |\theta| \leq \pi \tag{50}$$

Thus one can choose

$$\delta = 2/(c_1 + c_2) \tag{51}$$

and get a relaxation that converge in a rate independent of the number of design variables. This may seem very good, however, as can be seen the Hessian does not filter out the low frequencies as was in the previous example. This implies that the change in the state and adjoint variables, as a result in the change in design variable can no longer be calculated efficiently. The total efficiency of a multigrid method based on this relaxation will be far from optimal. Some modification of the gradient descent direction has to be introduced here to achieve optimal performance. This lead us to the general treatment.

Preconditioners for Multigrid Relaxation. To overcome difficulties such as in the last example, we introduce a preconditioner that is applied to the gradient before using it to define a direction of change for the design variables. Call that preconditioner $\mathcal{R}$, with a symbol $\hat{\mathcal{R}}(\theta)$. Our task is to choose $\hat{\mathcal{R}}^h(\theta)$ such that

$$\hat{\mathcal{R}}^h(\theta)\hat{\mathcal{H}}^h(\theta) \geq C \sum_j \sin^\gamma(\theta_j/2) \tag{52}$$

with $\gamma > 0$. To achieve this we need to analyze the discrete symbol $\hat{\mathcal{H}}^h(\theta)$. This however, may be a difficult task in general and an alternative approach is adopted.

The idea is to analyze the behavior of $\hat{\mathcal{H}}(\mathbf{k})$ and $\hat{\mathcal{R}}(\mathbf{k})$ for the differential problem, since in that case the computation is much simpler. This is followed by a discretization that does not violate h-ellipticity leading to a proper choice for $\hat{\mathcal{R}}(\theta)$.

We therefore construct $\hat{\mathcal{R}}(\mathbf{k})$ such that

$$\hat{\mathcal{R}}(\mathbf{k})\hat{\mathcal{H}}(\mathbf{k}) \geq |\mathbf{k}|^\gamma \tag{53}$$

with $\gamma > 0$. For practical application a $\gamma = 2$ yields good smoothing properties. The smoothing of the problem is then given by considering the preconditioned relaxation

$$\alpha \leftarrow \alpha - \delta \mathcal{R} g \tag{54}$$

where g is the gradient and δ is found by a line search on coarse levels and by Fourier analysis for fine grids. Fourier analysis of iteration matrix for that relaxation is

$$1 - \delta \hat{\mathcal{R}}^h(\theta)\hat{\mathcal{H}}^h(\theta). \tag{55}$$

A choice for δ for fine grids is easily calculated as

$$\delta = 2/[\mathcal{R}^h(\pi/2)\hat{\mathcal{H}}^h(\pi/2) + \mathcal{R}^h(\pi)\hat{\mathcal{H}}^h(\pi)]. \tag{56}$$

Notice that the relaxation does not requires a line search for the fine grids. The coarsest grids for which the analysis of the Hessian is not accurate enough uses a standard BFGS.

For example I of section 4.1 with the h-elliptic discretization we obtain that the preconditioner $\mathcal{R}^h$ should have a symbol

$$\mathcal{R}^h(\theta) = \frac{h^2}{4\sin^2(\theta/2)}, \tag{57}$$

which means that $\mathcal{R}^h$ is the inverse of T^h given by

$$(T^h g)_k = \frac{1}{h^2}(g_{k+1} - 2g_k + g_{k-1}) \tag{58}$$

4.7 Full Multigrid

As in any efficient multigrid implementation, it is recommended to use the full multigrid method. This means to start solving the problem on coarse grids, and using these solutions as initial approximations for the next finer level. This process continues until it reaches the finest level where the solution of the problem is required. The order of interpolation has to be chosen in an appropriate way, Brandt [14].

References

[1] Arian E., Ta'asan S., Shape Optimization in One-Shot. Optimal Design and Control, Edited by J. Boggaard, J. Burkardt, M. Gunzburger, J. Peterson, Birkhauser Boston Inc. 1995

[2] Arian E., Ta'asan S., Multigrid One-Shot Methods for Optimal Control Problems: Infinite Dimensional Control. ICASE Report No. 94-52.

[3] E. Arian, S. Ta'asan, Analysis of the Hessian for Aerodynamics Optimization: Inviscid Flow. ICASE Report No. 96-28., submitted to Journal of Computational Physics

[4] Beux F., Dervieux A., A Hierarchical Approach for Shape Optimization, Inria Rapports de Recherche, N. 1868 (1993).

[5] Jameson A., Aerodynamics Design via Control Theory. J. Sci. Comp. Nov. 21, 1988

[6] Mandel B., Periaux J., Stoufflet B., Optimum Design Methods in Aerodynamics, AGARD-FDP-VKI Special Course, April 1994.

[7] Ta'asan S., One-Shot Methods for Optimal Control of Distributed Parameter Systems I: Finite Dimensional Control. ICASE Report No. 91-2, 1991

[8] S. Ta'asan, Trends in aerodynamics design and optimization: a mathematical view point. in Proceedings of the 12th AIAA Computational Fluid Dynamics Conference June 1 995, San Diego, CA. pp. 961-970 AIAA-95-1731-CP.

[9] Ta'asan S., Fast Solvers for MDO Problems. Multidisciplinary Design Optimization, state of the art. Edited by N.M. Alexandrov, M.Y. Hussaini, SIAM 1997.

[10] Ta'asan S., Salas M.D., Kuruvila G., Aerodynamics Design and Optimization in One-Shot. Proceedings of the AIAA 30th Aerospace Sciences and Exhibit, Jan 6-9 1992.

[11] Ta'asan S., Kuruvila G., Salas M.D., A New Approach to Aerodynamics Optimization. Proceedings of the First European Conference on Numerical Methods in Engineering. Sept 1992, Brussels Belguim.

[12] C. Hirsch, Numerical Computation of Internal and External Flows, Vol I, II, Wiley 1990.

[13] O. Pironneaux, Optimal Shape Design for Elliptic Systems, Springer Series in Computational Physics, 1983.

[14] A. Brandt, Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics. GMD-Studien.