Copyright © 2000, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

CONSISTENT APPROXIMATIONS AND APPROXIMATE FUNCTIONS AND GRADIENTS IN OPTIMAL CONTROL

by

Olivier Pironneau and Elijah Polak

Memorandum No. UCB/ERL M00/14

21 March 2000

CONSISTENT APPROXIMATIONS AND APPROXIMATE FUNCTIONS AND GRADIENTS IN OPTIMAL CONTROL

by

Olivier Pironneau and Elijah Polak

Memorandum No. UCB/ERL M00/14

21 March 2000

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Consistent Approximations and Approximate Functions and Gradients In Optimal Control[*]

Olivier Pironneau[†]& Elijah Polak[‡]

March 21, 2000

Abstract

As shown in [7], optimal control problems with either ODE or PDE dynamics can be solved efficiently using a setting of consistent approximations obtained by numerical discretization of the dynamics together with master algorithms that adaptively adjust the precision of discretization (in an outer loop) and call finite dimensional optimization algorithms as subroutines (in an inner loop). An important fact overlooked in [7] is that in many discretized optimal control problems both the value and the gradient of the cost function cannot be computed exactly because they involve the solution of a large linear or nonlinear system at some stage. As a result, the master algorithms presented in [7] cannot be implemented efficiently for such problems.

In [7] we find also a master algorithm for solving *finite dimensional* optimization problems when both the cost function value and its gradient can only be computed approximately. In this paper we present a new master algorithm model that combines the features of this master algorithm with those of one intended for infinite dimensional problems and establish conditions for its convergence.

We implement this new master algorithm using an approximate steepest descent method for the solution of two problems: a two point

[*]This work was supported in part by the National Science Foundation under Grant No. ECS-9900985 and by the Institut Universitaire de France

[†]LAN, University of Paris 6.

[‡]EECS, University of California, Berkeley.

boundary value problem where the linear system corresponding to the ODE is solved approximately only, and a distributed control problem in which the discretized dynamics are solved using a Domain Decomposition algorithm which can be implemented on parallelized computers.

1 Introduction

In [6], [7] a theory of consistent approximations is presented for optimization problems as a way of dealing with infinite dimensional problems, such as optimal control problems with either ODE or PDE dynamics. The theory provides conditions for a set of discretized problems to be a family of consistent approximations together with master algorithms that adaptively adjust the precision of discretization (in an outer loop) and call finite dimensional optimization algorithms as subroutines (in an inner loop).

While attempting to solve some optimal control problems with distributed dynamics (see Lions[3]) using the consistent approximations framework, we came across a new difficulty which stems from the fact that even the discretized state equation cannot be solved with adequate precision in reasonable time. In such problems there are *two* precision parameters to control: the mesh size h, which defines the approximating problem, and the number of iterations N used by a "solver" in solving the discretized state equations. Since the parameter N seriously impacts the behavior of optimization algorithms as well as the total work needed to solve a problem, it is desirable to control the two precision parameters individually. We will present an efficient scheme for doing this in the form of a Master Algorithm Model.

To illustrate the source of the difficulty mentioned above, consider an optimization problem of the form

$$\textbf{(P)} \qquad \min_{v \in V} f(v) \tag{1}$$

where, for example, $V = L^2(0,1)$,

$$f(v) = J(u(v), v) = \int_0^2 |u(v) - u_d|^2 dx, \tag{2}$$

and $u(v)$ is the solution of an equation of the form

$$Cu = Bv \tag{3}$$

such as

$$-u''(x) = v(x)I_{(0,1)}, \quad \forall x \in (0,2), \quad u(0) = u(2) = 0 \tag{4}$$

where u_d is given and I_D is the characteristic function of the set D. $\diamondsuit$

This problem can be approximated by a finite dimensional problem of the form

$$(\mathbf{P}_h) \qquad \min_{v \in V_h} f_h(v) \tag{5}$$

where V_h the space of piecewise constant functions defined on a mesh for $(0,2)$, $h > 0$ is the mesh size,

$$f_h(v) = J(u_h(v), v_h), \tag{6}$$

and $u_h(v)$ is the solution of a discretized equation of the form

$$C_h u_h = B_h. \tag{7}$$

It is not difficult to show that the problems $(\mathbf{P}_h)$ epi-converge [1] to the problem $(\mathbf{P})$, as $h \to 0$, and that if $\{v_h\}$ is a sequence of points such that $v_h \in V_h$, $h \to 0$, and $v_h \to v$, as $h \to 0$, then $\mathrm{grad} f_h(v_h) \to \mathrm{grad} f(v)$ as $h \to 0$. These facts show that the pairs $(\mathbf{P}_h, -\|\mathrm{grad} f_h(\cdot)\|)$ form a family of *consistent approximations* for the pair $(\mathbf{P}, -\|\mathrm{grad} f(\cdot)\|)$, in the sense defined in Section 3.3 of [7]. The important consequence of this is that any accumulation point of global optimizers of the problems $(\mathbf{P}_h)$ is a global optimizer of the problem $(\mathbf{P})$. It also follows from the above that if $\{v_h\}$ is a sequence of points such that $v_h \in V_h$, $h \to 0$, $v_h \to v$ and $\mathrm{grad} f_h(v_h) \to 0$, then $\mathrm{grad} f(v) = 0$ also.

The fact that the approximating pairs $(\mathbf{P}_h, -\|\mathrm{grad} f_h(\cdot)\|)$ are a family of consistent approximations for the pair $(\mathbf{P}, -\|\mathrm{grad} f(\cdot)\|)$ lays a basis for the solution of $\mathbf{P}$ by the type of algorithm outlined in Section 3.3 of [7]. Unfortunately, for large problems C_h is a large sparse matrix and it is quite

[1]See Sec. 3.3 in [7] or chap. 7 in [8] for a definition of epi-convergence.

possible that all efficient methods for solving the linear system for $u_h(v)$ are iterative and, realistically, only a reasonable number of iterations of an iterative "solver" can be contemplated. Similar facts apply to the computation of $f_h(v)$ and $\text{grad} f_h(v)$. Hence, as we have already mentioned, we have to establish new algorithms that can use such approximations.

We will denote by $u_{h,N}(v)$ the result of N iterations of an iterative "solver" applied to the linear system, (7), and we will denote by $f_{h,N}(v)$ the associated approximation to $f_h(v)$. Similarly, we will denote by $\text{grad}_N f_h(v)$ the result of N iterations of an iterative "solver" applied to the defining equations for $\text{grad} f_h(v)$. For instance, if the Gauss-Seidel relaxation algorithm is used to solve (7), then $u_{h,N}(v)$ is the N-th iterate of recursion

$$L_h u^p = B_h v - U_h u^{p-1}, \quad p = 1, ..., N, \quad u^0 \text{ given}, \tag{8}$$

where L_h is the lower diagonal part of C_h and U_h its upper part.

Thus we see that in this case the discretized functions $f_h(v)$ are not computable exactly and, for obvious reasons, neither are their gradients. We will see later that this is also the case when Domain Decomposition is used to solve discretized PDE's. A quick reference to Section 3.3 of [7] shows that the master algorithm models outlined there are not applicable to these cases, because there are no standard nonlinear programming algorithms that use approximate function and gradient values, necessitating the development of a new computational scheme, which we will present in the next section.

2 A Master Algorithm Model

We will construct a new master algorithm model for solving problems of the form (**P**), that uses only approximations $f_{h,N}(v)$ and $\text{grad}_N f_h(v)$ to the cost function $f_h(v)$ and its gradient $\text{grad} f_h(v)$, by making use of some existing results in [7]. The relevant results are as follows: First on page 406 in [7], we find the following Master Algorithm Model 3.3.17 for solving problems of the for (**P**), in (2) above, which uses the iteration functions $A_h : V_h \to V_h$, $h \in (0, h_{-1}]$:

Master Algorithm Model 1: Solves (**P**).

Parameters. $\omega \in (0, 1)$, $\sigma > 0$.

Data. $h_{-1} \in \Re_+$, and $v_0 \in V_{-1}$.

Step 0. Set $i = 0$.

Step 1. Compute the largest h_i, of the form $h_i/2^k$, and v^{i+1}, such that $h_i \leq h_{i-1}$ and

$$v^{i+1} \in A_{h_i}(v^i), \tag{9}$$

and

$$f_{h_i}(v^{i+1}) \; - \; f_{h_i}(v^i) \leq -\sigma h_i^{\omega} \; . \tag{10}$$

Step 2. Replace i by $i \; + \; 1$, and go to **Step 1.** $\diamondsuit$

Unfortunately, as we have explained in the Introduction, we may not have explicit formulas for computing $f_h(v)$ and $\text{grad} f_h(v)$ and hence we may be forced to use the limited precision results of N iterations of an iterative solver for computing these quantities. Consequently, a high precision evaluation of even a simple iteration map $A_h(v)$, such as

$$A_h(v) = v - \lambda \text{grad} f_h(v) \tag{11}$$

with the step-size λ determined by the Armijo rule or by one-dimensional minimization, can be prohibitively expensive.

Defining, as before, $u_{h,N}(v)$, to be the result of N iterations of a solver applied to the defining equation (7), we find that

$$f_{h,N}(v) := J(u_{h,N}(v), v). \tag{12}$$

As we will see later, $\text{grad} f_h(v)$ is usually determined as a solution of an adjoint equation. Hence $\text{grad}_N f_h(v)$ is defined as the result of N iterations of a solver applied to the adjoint equations. This leads to an approximation $A_{h,N}(v)$ to the ideal iteration map $A_h(v)$. For example, the ideal iteration map $A_h(v)$ defined in (11) has to be replaced by

$$A_{h,N}(v) = v - \lambda \text{grad}_N f_h(v) \tag{13}$$

where the step-size λ determined either by a modified Armijo rule of by one-dimensional minimization.

There is obviously any number of ways of making the parameter N a function of h, which result in a new approximation to the cost function

$$\hat{f}_h(v) := f_{h,N_h}(v) \tag{14}$$

and iteration map

$$\hat{A}_h(v) := A_{h,N(v)}(v), \tag{15}$$

which, hopefully, can be used within the structure of Master Algorithm Model 1. One can classify the rules for making N a function of h as *open-loop* or *closed-loop*. An example of an open-loop rule is to set $N = \text{int}(1/h)$, the integer part of $1/h$. A closed-loop rule can be made more subtle, and designed to produce as small a parameter N as is compatible with the convergence of the overall solution scheme in the form of a master algorithm.

We will now show that one iteration of Master Algorithm Model 1.2.36 in [7], which can be used for constructing algorithms for solving the *finite dimensional* problem $(\mathbf{P}_h)$, in (5), provides a reasonable closed-loop technique for defining the the number N of iterations to be used by the solver in terms of the mesh size h. This master algorithm model has the form, with $\mathcal{N} := \{0, 1, 2, ...\}$:

Master Algorithm Model 2: Solves $(\mathbf{P}_h)$.

Parameters. $N_0, K \in \mathcal{N}$, $N_0 > 0$, $\sigma > 0$, $\omega \in (0,1)$, $\Delta : Re_+ \to Re_+$.

Data. $v_0 \in X$.

Step 0. Set $i = 0$.

Step 1. Set $N = N_0$.

Step 2. Compute a $y \in A_{h,N}(v^i)$

Step 3. If

$$f_{h,N}(y) - f_{h,N}(v^i) \leq -\frac{\sigma}{N^\omega}, \tag{16}$$

set $v^{i+1} = y$, replace i by $i + 1$, and go to **Step 2**.

Else, replace N by $N + K$, and go to **Step 2**. ◊

Proceeding formally from this point on, we assume that for every $h > 0$ we can construct an iteration map $A_{h,N} : V_h \to V_h$, of the type required by Algorithm Model 2.

We will depend on the following assumption:

Assumption 1. We will assume as follows:

(i) The function $f(\cdot)$ is continuous and bounded from below, and for all $h \in (0, h_{max}]$, the functions $f_h(\cdot)$ are continuous and bounded from below.

(ii) For every bounded set $B \subset V$, there exists $\kappa < \infty$, a function $N^* : \Re_+ \to \mathcal{N}$, and functions $\varphi : \Re_+ \times \Re_+ \to \Re_+$, $\Delta : \Re_+ \to \Re_+$ with the properties

$$\lim_{h\to 0} N^*(h) = \infty, \tag{17}$$

$$\lim_{N\to\infty} \varphi(h, N) = 0, \quad \forall h > 0, \tag{18}$$

$$\lim_{h\to 0} \varphi(h, N_h) = 0, \quad \forall N_h \geq N^*(h), \tag{19}$$

$$\lim_{h\to 0} \Delta(h) = 0, \tag{20}$$

such that for all $h \in (0, h_{max}]$, $v \in V_h \cap B$,

$$|f_h(v) - f(v)| \leq \kappa\Delta(h), \tag{21}$$

and for all $h \in (0, h_{max}]$, $N \in \mathcal{N}$, $v \in V_h \cap B$,

$$|f_{h,N}(v) - f_h(v)| \leq \kappa\varphi(h, N). \tag{22}$$

(iii) For every $v^* \in V$ such that $\text{grad} f(v^*) \neq 0$, there exist $\rho^* > 0$, $\delta^* > 0$, $h^* > 0$, $N^* < \infty$, such that

$$f_{h,N}(A_{h,N}(v)) - f_{h,N}(v) \leq -\delta^*, \ \forall v \in V_h \cap B(v^*, \rho^*), \ \forall h \leq h^*, \ \forall N \geq N^*. \tag{23}$$

For any positive real number α, we define ceil$[\alpha]$ to be the smallest integer larger than α.

Master Algorithm Model 3: Solves (**P**).

```
Parameters. $h_0 > 0$, $\omega \in (0,1)$, $C_1 \geq 1$, $C_2, C_3 > 0$, $K \in \mathcal{N}$, $N^*(\cdot)$, $\varphi(\cdot,\cdot)$
    verifying (17), (18), (19), (20).

Data. $v^0$.

Begin Outer Loop

Step 0. Set $i = 0$, $h = h_0$.

    Begin Inner Loop (defines $N_h(v^i)$, $\hat{f}_h(v^i)$ and the iteration function
        $\hat{A}_h(v^i)$).
    Step 1. Set $N = C_1 N^*(h)$.
    Step 2. Compute a point $v^* = A_{h,N}(v^i)$.
    Step 3. Compute
            $f_{h,N}(v^*) - f_{h,N}(v^i)$.        (24)
    Step 4. If
            $f_{h,N}(v^*) - f_{h,N}(v^i) > -C_2 \varphi(h,N)^\omega$,        (25)
        replace $N$ by $N + K$ and go to Step 2.
        Else, set
            $N_h(v^i) := N$,        (26)
        and
            $\hat{A}_h(v^i) := A_{h,N_h(v^i)}(v^i)$        (27)
    End Inner Loop

Step 5. If
        $f_{h,N}(v^*) - f_{h,N}(v^i) > -C_3(\Delta(h) + \varphi(h, N_h(v^i)))^\omega$,        (28)
    replace the mesh-size $h$ by $h/2$ and go to Step 1.
    Else, set
        $v^{i+1} = \hat{A}_h(v^i)$,        (29)
    replace $i$ by $i + 1$ and go to Step 1.

End Outer Loop        ◇
```

Remark 1 The main function of the test (25) is to increase N over the initial value of $N = N^*(h)$ if that is necessary. It gets reset to $N = N^*(h)$ whenever h is halved.

Note that the faster $\varphi(h, N) \to 0$ as $N \to \infty$, the easier it is to satisfy the test (25) at a particular value of N. Thus, when the solver is fast, the precision parameter N will be increased more slowly than when it is slow. A similar argument applies to the test in (19). In the context of dynamics defined by differential equations, the integration mesh size will be refined much faster when the Euler method is used for integration than when a Runge-Kutta method is used for integration. $\diamondsuit$

In view of the definition (26), for every $h \in (0, h_{max}]$ and $v \in V_h$, we define

$$\hat{f}_h(v) := f_{h,N_h(v)}(v). \tag{30}$$

We can define problems

$(\hat{\mathbf{P}}_\mathbf{h})$

$$\inf_{v \in V_h} \hat{f}_h(v). \tag{31}$$

It is possible to show that these problems epi-converge to $(\mathbf{P})$, as $h \to 0$. In order to deduce the convergence properties of Master Algorithm Model 3 from Theorem3.3.19 in [7], we need the following result.

Lemma 1

(a) *For every bounded set $B \subset V$, there exists a $\kappa < \infty$ and a function $\hat{\Delta} : \Re_+ \times V \to \Re_+$, such that (i) $\hat{\Delta}(h, v) \to 0$, as $h \to 0$, uniformly in $v \in B$, and (ii) for all $h \in (0, h_{max}]$, $v \in V_h \cap B$,*

$$|\hat{f}_h(v) - f(v)| \leq \kappa \hat{\Delta}(h, v). \tag{32}$$

(b) *For every $\hat{v} \in V$ such that* $\text{grad} f(\hat{v}) \neq 0$*, there exist $\hat{\rho} > 0$, $\hat{\delta} > 0$, $\hat{h} > 0$ such that*

$$\hat{f}_h(\hat{A}_h(v)) - \hat{f}_h(v) \leq -\hat{\delta}, \quad \forall v \in B(\hat{v}, \hat{\rho}), \quad \forall h \leq \hat{h}, \tag{33}$$

where $\hat{A}_h(v)$ is defined by (27).

Proof. *(a)* It follows from (21) and (22) that for all $h \in (0, h_{max}]$, $v \in V_h$ and $N \in \mathcal{N}$,

$$\begin{aligned} |f_{h,N}(v) - f(v)| &\leq |f_{h,N}(v) - f_h(v)| + |f_h(v) - f(v)| \\ &\leq \kappa\varphi(h, N) + \kappa\Delta(h). \end{aligned} \tag{34}$$

Hence we have that

$$|\hat{f}_h(v) - f(v)| = |f_{h,N_h(v)}(v) - f(v)| \leq \kappa(\varphi(h, N_h(v)) + \Delta(h)) \equiv \kappa\hat{\Delta}(h, v). \tag{35}$$

Since

$$\hat{\Delta}(h, v) = \varphi(h, N^*(h)) + \Delta(h), \tag{36}$$

and $N_h(v) \geq N^*(h)$, it follows that $\hat{\Delta}(h, v) \to 0$, as $h \to 0$, uniformly in $v \in V \cap B$.

(b) Suppose that $v \in V$, is such that $\mathrm{grad} f(v) \neq 0$. Then, by Assumption 1 (iii), there exist a $\rho^* > 0$, $\delta^* > 0$, and $h^* > 0$, and $N^* < \infty$ such that (23) holds. Let $\hat{h} \in (0, h^*]$ be such that $N^*(h) \geq N^*$ for all $h \in (0, \hat{h}]$. Then, because $N_h(v) \geq N^*(h)$ by construction, it follows from (23) that

$$\begin{aligned} \hat{f}_h(\hat{A}_h(v)) - \hat{f}_h(v) &= f_{h,N}(A_{h,N_h(v)}(v)) - f_{h,N_h(v)}(v) \\ &\leq -\delta^*, \quad \forall v \in V_h \cap B(v^*, \rho^*), \quad \forall h \leq \hat{h}, \end{aligned} \tag{37}$$

which completes our proof. ◇

The following theorem is a direct consequence of Lemma 1 and Theorem 3.3.19 in [7] for the case where the cost function $f(v)$ is strictly convex.

Theorem 1 *If $f(\cdot)$ is strictly convex and $\{v^i\}_{i=0}^{\infty}$ is a sequence constructed by Master Algorithm Model 3, in solving the problem* (**P**)*, then it converges to the unique solution of* (**P**)*.*

Remark 2 If $f(\cdot)$ is not strictly convex but only continuously differentiable, then the conclusion of Theorem 1 has to be changed to read that all accumulation points of the sequence $\{v^i\}_{i=0}^{\infty}$ are stationary points. ◇

Remark 3 The following Master Algorithm Model differs from Master Algorithm Model 3 in two respects: first the integer N is never reset and hence increases monotonically, and second the test for reducing h is based on the magnitude of the norm of the approximate cost-gradient. As a result, the proof its of convergence is substantially simpler than for Master Algorithm Model 3. However, convergence can be established only for the diagonal subsequence $\{v^{i_j}\}_j$ at which h is halved. ◇

Master Algorithm Model 4: Solves (**P**).

Parameters. $h_0 > 0$, $\omega \in (0,1)$, $\epsilon > 0$, $C > 0$, $K \in \mathcal{N}$, $N^*(\cdot)$, $\varphi(\cdot,\cdot)$ verifying (17), (18), (19).

Data. $v^0 \in V_h$.

```
Begin Outer Loop

Step 0. Set $i = 0$, $h = h_0$, $N = N^*(h)$.

    Begin Inner Loop
    Step 1. Compute a point $v^* = A_{h,N}(v^i)$.
    Step 2. Compute
                $$f_{h,N}(v^*) - f_{h,N}(v^i).  \qquad (38)$$
    Step 3. If
                $$f_{h,N}(v^*) - f_{h,N}(v^i) > -C\varphi(h,N)^\omega,  \qquad (39)$$
        replace $N$ by $N + K$ and go to Step 1.
        Else, set $v^{i+1} = v^*$, and go to Step 4.
    End Inner Loop

Step 4. If
                $$\|\text{grad}_N f_h(v^{i+1})\| \le \epsilon \text{ and } N \ge N^*(h),  \qquad (40)$$
        replace $h$ by $h/2$, $\epsilon$ by $\epsilon/2$, $i$ by $i + 1$, and go to Step 1.
    Else, replace $i$ by $i + 1$ and go to Step 1.

End Outer Loop
```
◇

3 A Two-Point Boundary Value Control Problem

Consider again the two-point boundary-value control problem first stated in the introduction:

$$\textbf{(P)}\quad \left|\begin{array}{l} \min\limits_{v\in L^2(0,1)} f(v) := J(u(v)) := \int_0^2 |u-u_d|^2 dx \quad \text{subject to} \\ \qquad -u''(x) = v(x)I_{(0,1)}, \ \ \forall x \in (0,2), \ \ u(0)=u(2)=0. \end{array}\right. \tag{41}$$

The gradient of $f(\cdot)$ with respect to v can be expressed in terms of p, the solution of the adjoint equation

$$-p'' = 2(u-u_d) \quad p(0)=p(2)=0. \tag{42}$$

Thus,

$$\delta f = 2\int_0^2 (u-u_d)\delta u = -\int_0^2 p''\delta u = -\int_0^2 p\delta u'' = \int_0^1 p\delta v, \tag{43}$$

which shows that $\text{grad} f(v) = p$.

To approximate the problem (**P**), we use a finite difference method with uniform mesh of size $h = 1/M$, to solve the differential equation. This results in the approximating problems

$$(\mathbf{P}_h)\quad \left|\begin{array}{l} \min\limits_{v\in V_h} f_h(v) := \sum_1^{2M-1} |u_j - u_d(jh)|^2 \quad \text{subject to} \\ -\frac{1}{h^2}(u_{j+1} - 2u_j + u_{j-1}) = v_j I_{j\le M}, \ \ i = 1,...,2M-1 \\ \qquad u_0 = u_{2M} = 0. \end{array}\right. \tag{44}$$

where V_h is the set of piecewise constant functions on the intervals $(jh, (j+1)h]$. Note that the coefficient u_j define a piecewise constant function $u(\cdot)$ on $[0,2]$.

As in the continuous case

$$\delta f = \sum_1^{2M-1} 2(u_j - u_d(jh))\delta u_j \tag{45}$$

and if

$$-\frac{p_{j+1}-2p_j+p_{j-1}}{h^2} = 2(u_j - u_d(jh)), \quad j = 1, ..., 2M-1 \quad p_0 = p_{2M} = 0. \tag{46}$$

then

$$\begin{aligned} \textstyle\sum_1^{2M-1} 2(u_j - u_d(jh))\delta u_j &= -\textstyle\sum_1^{2M-1} \frac{p_{j+1}-2p_j+p_{j-1}}{h^2}\delta u_j \\ &= -\textstyle\sum_1^{2M-1} \frac{\delta u_{j+1}-2\delta u_j+\delta u_{j-1}}{h^2} p_j. \end{aligned} \tag{47}$$

Therefore

$$\delta f = \sum_1^M p_j \delta v_j, \tag{48}$$

and hence the gradient grad$f_h(v)$ is the piecewise constant function $p_h(\cdot)$, on $[0, 2]$, defined by the coefficients $p_0, p_1, ..., p_M$.

3.1 Verification of the Hypotheses

Master Algorithm Model 3 depends on Assumption 1 to be satisfied and, in particular, on the existence three appropriate functions $\phi(h, N)$, $N^*(h)$, and $\Delta(h)$, and of an appropriate iteration map $A_{h,N}(\cdot)$.

We begin by showing that parts (i) and (ii) of Assumption 1 are satisfied. When the ODE for u is multiplied by u and integrated in x, we obtain, after an integration by parts

$$-\int_0^2 (uu'') = \int_0^1 uv = \int_0^2 (u'^2). \tag{49}$$

Applying the Schwarz inequality to the middle integral leads to $\|u'\|_0 \leq \|v\|_0$. Next, it follows from the Poincaré inequality that $\|u\|_0 \leq C\|u'\|_0$, for some $C < \infty$, and hence we conclude that u is Lipschitz continuous with respect to v in L^2:

$$\|u\|_0 \leq C\|v\|_0. \tag{50}$$

Now the function $u \to J(u)$ is obviously continuous in u, and hence $f(\cdot)$ is continuous in v.
Using similar arguments we find that p is continuous in v and hence grad$f(\cdot)$ exists and is continuous.

For the discrete problem we note that $(u_1, u_2, ..., u_{2M-1})^T$ is the solution of a linear system with right hand side $(v_1, .., v_M, 0.., 0)^T$ and the matrix of the linear system is tridiagonal with $2/h^2$ on the main diagonal and $-1/h^2$ on the diagonals below and above the main one. This is a positive definite matrix so u is continuous with respect to v. Similarly, it is possible to show that p is also continuous with respect to v.

Next, it follows from the error analysis for the finite difference scheme that for some $C < \infty$,

$$\|u_h - u\|_0 < Ch^2, \quad |J_h(u,v) - J(u,v)| < Ch^2 \tag{51}$$

which implies that

$$|f_h(v_h) - f(v)| < Ch^2. \tag{52}$$

Now the Gauss-Seidel algorithm is linearly convergent but the constant of convergence is proportional to the condition number of the linear system. In particular, for some $C, c < \infty$

$$\|u_{h,N} - u_h\| \le C(1 - ch^2)^N \quad \forall N \in \mathcal{N}. \tag{53}$$

By inspection, the bound function φ is $\varphi(h, N) = (1 - ch^2)^N$. However, it contains an unknown constant. We have the choice of either guessing this constant or replacing the function φ with a conservative estimate, such as $\varphi(h, N) = (1 - h^{2+\epsilon})^N$, with $\epsilon < 1$, small, i.e, we replace c with h^ϵ. In either event, and to satisfy the hypothesis we may take

$$N^*(h) = \frac{C}{h^{2+2\epsilon}}, \tag{54}$$

with $C \in (0, \infty)$. Indeed,

$$(1 - ch^{2+\epsilon})^{\frac{C}{h^{2+2\epsilon}}} = \exp \frac{C \log(1 - ch^{2+\epsilon})}{h^{2+2\epsilon}} \approx e^{-\frac{C}{h^\epsilon}} \to 0, \text{ as } h \to 0. \tag{55}$$

We have thus shown that parts (i) and (ii) of Assumption 1 are satisfied.

To conclude, we must show that part (iii) of Assumption 1 is satisfied. We will derive the iteration map $A_{h,N}(\cdot)$ from the the standard steepest descent algorithm with exact step-size. We recall that for the problems $(\mathbf{P}_h)$, this algorithm is defined by the following iteration function:

$$A_h(v) = v - \lambda(v) \text{grad} f_h(v), \tag{56}$$

where

$$\lambda(v) := \arg\min_{\lambda} f_h(v - \lambda \text{grad} f_h(v)). \tag{57}$$

Note that for our problem $\lambda(v)$ can be computed exactly because $f(v - \lambda \text{grad} f_h(v))$ is a quadratic function of λ.

Next, we define $A_{h,N}$ as follows:

$$A_{h,N}(v) := v - \lambda(v) \text{grad}_N f_h(v), \tag{58}$$

with

$$\lambda(v) := \arg\min_{\lambda} f_{h,N}(v - \lambda \text{grad}_N f_h(v)), \tag{59}$$

where $f_{h,N}(v)$ and $\text{grad}_N f_h(v)$ are computed using N iterations of the Gauss-Seidel algorithm on the difference equation in (44) and the adjoint equation (46), respectively.

Now, it follows from the properties of the method of steepest descent that given any $v^* \in V$ such that $\text{grad} f(v^*) \neq 0$, there exists a $\rho^* > 0$, a $\delta^* > 0$, λ^*, and an $h^* > 0$, such that for all $v \in V \cap B(v^*, \rho)$, (i) $\text{grad} f_h(v) \neq 0$ and (ii)

$$f(v - \lambda(v) \text{grad} f(v)) - f(v) \leq f(v - \lambda^* \text{grad} f(v)) - f(v) \leq -\delta^*, \tag{60}$$

where $\lambda(v)$ is the exact step-size computed by the Steepest Descent Algorithm. It now follows from (18, 19, 21, 22) that there exist an $h^* > 0$ and an $N^* < \infty$, such that for all $h \leq h^*$, $N \geq N^*$, and $v \in V_h \cap B(v^*, \rho)$

$$f_{h,N}(v - \lambda(v) \text{grad}_N f_h(v)) - f_{h,N}(v) \leq -\delta^*/2, \tag{61}$$

which shows that part (iii) of Assumption 1 is satisfied.

3.2 Implementation of Master Algorithm Model 3

Making use of the maps defined in the preceding subsection, we now obtain the following

Implementation of Master Algorithm Model 3

Data. $C_1 > 0$, $C_2 > 0$, $C_3 > 0$, $\epsilon > 0$, $h > 0$, $K \in \mathcal{N}$, $v_0 \in V_h$.

Step 0. Set $i = 0$.

Step 1. Set $M = 1/h$, $N = \text{ceil}(\frac{C_1}{h^{2+2\epsilon}})$.

Step 2. Compute $\{u_j^i\}$ using N Gauss-Seidel iterations.

Step 3. Compute $\{p_j^i\}$ using N Gauss-Seidel iterations.

Step 4. Compute $\lambda = \text{argmin}_\lambda f_{h,N}(v^i - \lambda p^i)$ using N Gauss-Seidel iterations.

Step 5. Set $v_j^{i+1} = v_j^i - \lambda p_j^i, \quad j = 1 ... M$.

Step 6. **If** $f_{h,N}(v^{i+1}) - f_{h,N}(v^i) > -C_2(1 - h^{2+\epsilon})^N$, replace N by $N+K$ and go to **Step 2**.

Else, go to **Step 7**.

Step 7. **If** $f_{h,N}(v^{i+1}) - f_{h,N}(v^i) > -C_3[h^2 + (1 - h^{2+\epsilon})^N]$, replace h by $h/2$ and go to **Step 1**.

Else, replace i by $i + 1$ and go to **Step 2**. $\diamondsuit$

3.3 Numerical results

Problem (41) was solved with $u_d = \sin(\pi x)$ starting from $v = 0$, first using the standard steepest descent method, with a fixed mesh of 256 points and solving the linear system using 500 Gauss-Seidel iterations. Then it was solved using an implementation of Master Algorithm Model 3.

In the second case the initial mesh had 8 points and the final mesh had 512. We have used $\varphi(h, N) = (1 - C_4h^2)^N$ instead of $(1 - h^{2+\epsilon})^N$ and we set $N^*(h) = 0.1\text{ceil}(1/h^2)$. Finally, we used $\Delta(h) = 1/h^2$. The algorithm constants were

$$C_1 = 1 \quad C_2 = 0.1 \quad C_3 = 2\ 10^{-4} \quad C_4 = 5 \quad \epsilon = 0.1 \quad K = 20.$$

Figure 1 shows the convergence history of the cost function for both tests (left) and the history of the number of Gauss-Seidel iterations for the second case (right).

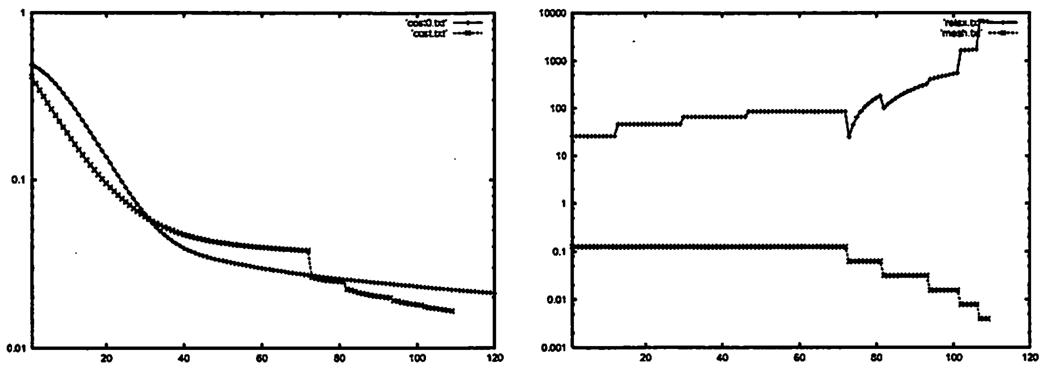

Figure 1
Cost function (left), mesh size and number of Gauss-Seidel iterations (right).

We have also tested a number of other values and other functions N_h, $\varphi(h, N)$. Most of the time similar computational behavior to the one describe here was obtained, however, some time the mesh was refined too fast and some time the number of Gauss-Seidel iterations became too large too soon, etc. Fine tuning the values of the parameters is not an easy task but it is C_3 which is the most important.

Our overall observation is that when the parameters of our algorithm are reasonably well selected, it computes a solution to problem **P** roughly 10 times faster than the algorithm that uses the same precision in all its iterations.

4 A Distributed Control Problem

Let S be a given subset of the boundary Γ of an open bounded subset Ω of $\Re^d$ and consider the boundary control problem

$$\textbf{(P)}\quad \left| \begin{array}{l} \min_{v \in L^2(S)} \{ f(v) = \int_\Omega [(u-u_d)^2 + |\nabla(u-u_d)|^2] \quad \text{subject to} \\ \qquad u - \Delta u = 0 \text{ in } \Omega, \ \ \frac{\partial u}{\partial n}|_S = \xi v \quad u_{\Gamma - S} = u_d \end{array} \right. \tag{62}$$

The gradient of $f(\cdot)$ can be obtained by making use of the fact that

$$\delta f = 2\int_\Omega((u-u_d)\delta u + \nabla(u-u_d)\cdot\nabla\delta u + o(|v|) = \int_S \xi(u-u_d)\delta v, \quad (63)$$

which follows from the fact that the PDE in variational form is

$$\int_\Omega(uw+\nabla u\cdot\nabla w) = \int_S \xi vw \quad \forall w \in H^1_{0_{\Gamma-S}}(\Omega). \quad (64)$$

So, by inspection of (63), we see that the gradient of $f(\cdot)$ with respect to the $L^2(S)$ norm is

$$\mathrm{grad}_v f(v) = \xi(u-u_d)|_S. \quad (65)$$

To approximate the problem (65), we propose to use a finite Element Method with $u \in V_h$, continuous and piecewise linear on the triangles of a triangulation of Ω. This results in the discretized, finite dimensional optimization problem below:

$$(\mathbf{P})_h \quad \left| \begin{array}{l} \min\limits_{v\in V_h} f_h(v) = \int_\Omega[(u-u_{dh})^2 + |\nabla(u-u_{dh})|^2] \quad \text{subject to} \\ \qquad \int_\Omega(uw+\nabla u\cdot\nabla w) = \int_S \xi vw \ \forall w \in V_h \end{array} \right. \quad (66)$$

where V_h is the approximation of $H^1_{0_{\Gamma-S}}(\Omega)$ consisting of continuous piecewise linear functions on the triangulation which are zero on $\Gamma-$.

The gradient of the discrete cost function $f_h(\cdot)$ can be obtained using the fact that

$$\delta f_h = \int_S \xi(u-u_{dh})\delta v \quad (67)$$

This formula is obtained exactly as in the continuous case. Therefore

$$\mathrm{grad}_v f_h(v) = P_h(u-u_{dh})|_S, \quad (68)$$

where P_h is the projection operator from $L^2(S)$ into $V_h \cap L^2(S)$.

Strictly speaking (68) holds only if Ω is a polygonal domain, but this is a standard technical problem with the finite element method which can be dealt with easily.

4.1 The Schwarz Algorithm

Now for some reason suppose that we want to solve the discrete Partial Differential Equation (i.e. it's equivalent sparse linear system) by a Domain Decomposition Method.

Let $\Omega = \Omega_1 \cup \Omega_2$, let $\Gamma = \partial\Omega$, and let $\Gamma_{ij} = \partial\Omega_i \cap \Omega_j$. The multiplicative Schwarz algorithm for the Laplace equation starts from a guess u_1^0, u_2^0 and computes the solution of

$$u - \Delta u = f \text{ in } \Omega, \quad u|_\Gamma = u_\Gamma \tag{69}$$

as the limit as $n \to \infty$ of the sequence $u_{i,n}, i = 1,2$ defined by

$$\left.\begin{array}{l}
u_{1,n+1} - \Delta u_{1,n+1} = f \text{ in } \Omega_1, \\
u_{1,n+1}|_{\Gamma\cap\bar\Omega_1 - S} = u_\Gamma \quad u_{1,n+1}|_{\Gamma_{12}} = u_{2,n} \quad \frac{\partial u_{1,n+1}}{\partial n}|_S = \xi v, \\
u_{2,n+1} - \Delta u_{2,n+1} = f \text{ in } \Omega_2, \\
u_{2,n+1}|_{\Gamma\cap\bar\Omega_2 - S} = u_\Gamma \quad u_{2,n+1}|_{\Gamma_{21}} = u_1^n \quad \frac{\partial u_{2,n+1}}{\partial n}|_S = \xi v.
\end{array}\right\} \tag{70}$$

4.2 The Doubly Discretized Problem

The introduction of the Schwarz algorithm leads to a doubly discretized problem, as follows. Let $\mathcal{T}_h$ be a triangulation of Ω of average edge size h such that by removing triangles we obtain also proper triangulations $\{\mathcal{T}_{jh}\}_{j=1,2}$ of Ω_1 and Ω_2.
Let V_{1h} and V_{2h} be the finite element spaces of continuous piecewise affine functions on $\{\mathcal{T}_{jh}\}_{j=1,2}$. Let V_{jh}^0 be the subspaces of continuous piecewise linear functions which are zero on the Dirichlet boundaries Γ_{ij}.

Then the doubly discretized problem is

$$(\mathbf{P})_{h,N} \quad \left| \begin{array}{l}
\min_{v\in V_h} f_{h,N}(v) = \|u^N - u_d\|_\Omega^2 : \quad u_j^0 = 0, \quad n = 1..N \\
u_j^n \in V_{jh} \ \forall w \in V_{jh}^0 : u_j^n|_{\Gamma_{ij}} = u_j^{n-1} \quad \int_{\Omega_j}[u_j^n w + \nabla u_j^n \nabla w] = \int_S \xi v w,
\end{array}\right. \tag{71}$$

where N is the number of Schwarz iterations applied to the the discretized PDE in (66).

Consider the mapping from $V_{1h} \times V_{2h}$ onto itself which defines u^n in terms of u^{n-1} by means of the recursion

$$\forall w \in V_h \quad u_j^n|_{\partial\Omega_{ij}} = u_j^{n-1} \quad \int_{\Omega_j} [u_j^n w + \nabla u_j^n \nabla w] = \int_S \xi v w \tag{72}$$

for $i = 1, 2$. Let $\{A, B, C\}$ be the finite element matrices associated with this operation:

$$AU^n = BU^{n-1} + CV \tag{73}$$

where U denotes the vector of values of u_1 at the vertices of $\mathcal{T}_1 h$ and of u_2 at the vertices of $\mathcal{T}_{2h}$ and V is the vector of values of v at the vertices of S. The doubly discretized problem (71) can now be rewritten as

$$\min_v (U^N - U_d)^T G (U^N - U_d) : \begin{pmatrix} A & 0 & 0 & \dots & 0 & 0 \\ B & A & 0 & \dots & 0 & 0 \\ 0 & B & A & .. & 0 & 0. \\ \dots & & & & & \dots \\ \dots & & & & B & A \end{pmatrix} \begin{pmatrix} U^1 \\ U^2 \\ U^3 \\ \dots \\ U^N \end{pmatrix} = \begin{pmatrix} BU^0 + CV \\ CV \\ CV \\ CV \\ CV \end{pmatrix} \tag{74}$$

where G is the finite element mass matrix (see Ciarlet [1] for more details).

We can express the exact gradient $\text{grad} f_{h,N}(v)$ of $f_{h,N}(v)$ in terms of the solution of the adjoint equation

$$\begin{pmatrix} A & B^T & 0 & \dots & 0 & 0 \\ 0 & A & B^T & \dots & 0 & 0 \\ 0 & 0 & A & .. & 0 & 0. \\ \dots & & & & \dots & B^T \\ \dots & & & & 0 & A \end{pmatrix} \begin{pmatrix} P^1 \\ P^2 \\ P^3 \\ \dots \\ P^N \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 2G(U - U_d) \end{pmatrix} \tag{75}$$

by making use of the fact that $\delta f_{h,N} = P^{1^T} C \delta V$. Thus we see that $\text{grad} f_{h,N}(v) = C^T P^1$.

The interpretation is that P, like U, is the set of values at vertices of the Schwarz system

$$p^N - \Delta p^N = 2(u^N - u_d) \quad p^{N-1} - \Delta p^{N-1} = 0 \quad p_{\Gamma_{ij}}^{N-1} = p^N \dots \tag{76}$$

These equations are difficult to implement because we must store all intermediate functions generated by the Schwarz algorithm and integrate the system for p^n in the reverse order. Hence we will we will use approximations to the gradients $\text{grad} f_{h,N}(v)$ defined by

$$\text{grad}_N f_h(v) := \mathcal{P}(\xi(u_{h,N}(v) - u_d))|_S, \tag{77}$$

where $\mathcal{P}$ is the interpolation operator ($\mathcal{P}g$ is the piecewise linear function which coincides with g at the vertices of S) and $u_{h,N}$ is computed by N iterations of the Schwarz algorithm with the convention that on $\Omega_1 \cap \Omega_2$, $u_{h,N} = \frac{1}{2}(u_{1h,N} + u_{2h,N})$.

4.3 Verification of the Hypotheses

We proceed exactly as in the one dimensional case to show that Assumption 1 is satisfied.

(i) Continuity of $f(\cdot)$ with respect to the control is established in Lions[3]. Continuity of $f_h(\cdot)$ with respect to the control is obvious from (74).

(ii) It follows from the finite element error estimates given in [1] that the error estimates (51, 52) hold for this case as well. Hence we can set $\Delta(h) = h^2$.

(iii) The Schwarz algorithm converges linearly with rate $(1 - d/D)$ where d is the diameter of $\Omega_1 \cap \Omega_2$ and D is the diameter of $\Omega - 2 \cup \Omega - 2$, so instead of (51) we have the bound

$$\|u_{h,N} - u_h\| \leq C(1 - \frac{d}{D})^N \quad \forall N \in \mathcal{N}, \tag{78}$$

for some $C \in (0, \infty)$, which implies that we can set $\phi(h, N) = (1 - \frac{d}{D})^N$. Note that in this case $\phi(h, N)$ is actually independent of h. In view of this, we can take $N^*(h) = C\text{ceil}(1/h)$, where $C > 0$ is arbitrary.

(iv) The relation (60) obviously holds for this case as well. To show that the relation (61) also holds, we make use of the facts that (a)

$$\text{grad} f_h(v) = P_h(\xi(u_h(v) - u_{dh}))|_S, \quad \text{grad}_N f_h(v) = \mathcal{P}_h(\xi(u_{h,N}(v) - u_d))|_S, \tag{79}$$

(b) both P_h and $\mathcal{P}_h$ tend to the identity operator at the rate $O(h)$ at least, and (c) the bound functions $\Delta(h)$, $\phi(h, N)$, and $N^*(h)$ have the required properties.

4.4 Example

In this example Ω_1 is the unit circle centered at the origin and Ω_2 is the rectangle $(0,1) \times (0,1)$ minus the unit triangle with vertices (0,0),(0,1),(1,0) and minus a disk of boundary S . The control boundary is S (see figures).

The function which is to be recovered by the optimization process is $u_d = e^{-x\sqrt{2}} sin(y)$. The weight on the control has been deliberately chosen to have oscillations: $\xi = \sin(30*(x-1.15)) + \sin(30*(y-0.5))$. We have used an automatic mesh generator controlled by a parameter n, the number of vertices on the boundaries, so, for practical reasons, we initialized $h = 1/(8n)$. The number of Schwarz iterations was initialized at 1.

The tests (25) (in Master Algorithm Model 3) and (39) (in Master Algorithm Model 4) for increasing the number of Schwarz iterations were determined by setting $\phi(h, N) = (0.8)^N$, and $C_1 = 0.1$. The mesh refinement test (28), in Algorithm Model 3 was implemented with the right hand side being set to $-0.001[10^{-4}h^2 + (0.8)^{1/(8h)}]$, which corresponds to $N^*(h) = 0.1/(8h)$, $\varphi(h, N) = 0.8^N$, $\Delta(h) = h^2$. Naturally other choices of coefficients and bound functions are possible.

The mesh refinement test (40) in Algorithm Model 4 was implemented by setting $\epsilon(n) = 10^{-n}$, where $n = 1/8h$.

We have used the code freefem+ [2] which is a matlab like environment for partial differential equations developped for the purpose of testing parallel algorithms, among other things.

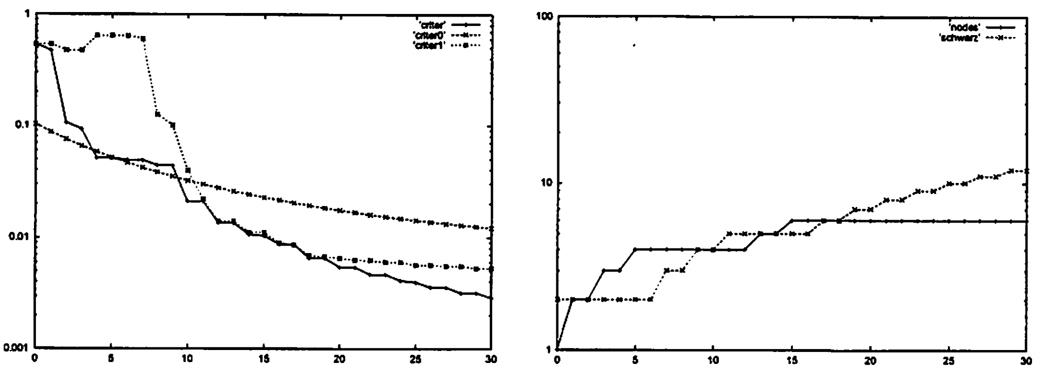

Figure 2

In the left part of Figure 2, we plot the values of the cost function $f(\cdot)$ v/s the iteration number for two cases. The first corresponds to optimization using a fixed mesh and a fixed number of Schwarz iterations, i.e., without adaptive precision refinement (curve 'criter0'), and the second one was obtained using adaptive refinement based either on the norm of the gradient (case (i), curve 'criter1'), or on the decrease of the cost function (case (ii), curve 'criter'). The right part of Figure 2 shows the number of Schwarz iterations N and the mesh parameter n versus the iteration number for case (i). After 30 iterations the gradient is 10^{-6} times its initial value, while without mesh refinement it is only 10^{-2} times its initial value (multigrid effect).

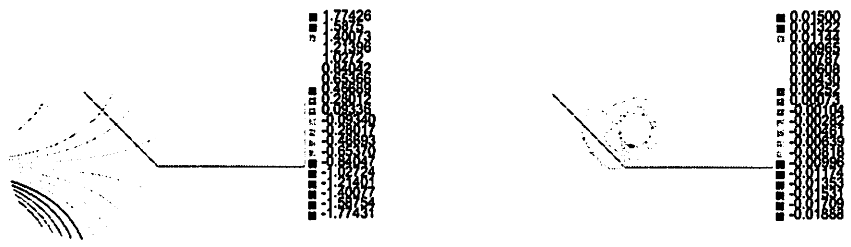

Figure 3

Figure 3 shows the computed solution u (left) and the error $u - u_d$ (right).

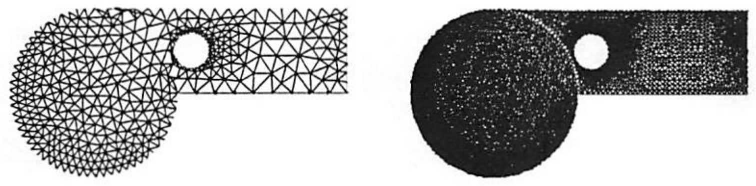

Figure 4

Figure 4 shows the second finite element mesh on the left and the 7th finite element mesh (the last is the 9th) on the right. Both are generated automatically by a Delaunay-Voronoi mesh generator from a uniform distribution of points on the boundaries.

5 Conclusion

It is a well known rule in optimization that, in solving infinite dimensional problems via discretization, one must use the exact gradient of the discretized problem rather than the approximate gradient of the exact problem. In this paper, we have shown that mesh refinement within the optimization loop enables us to relax the above rule. Our motivation is two fold: first, there are problems where the exact gradient of the discretized problem cannot be easily computed; secondly there is a multi-grid effect in combining mesh refinement with a descent algorithm which results in an order of magnitude in speed-up.

References

[1] CIARLET, P.G, The Finite Element Method,Prentice Hall, 1977.

[2] BERNARDI D., HECHT, F., OTSUKA K., PIRONNEAU O. : freefem+, a finite element software to handle several meshes.Dowloadable from ftp://ftp.ann.jussieu.fr/pub/soft/pironneau/, 1999.

[3] LIONS J.L.,*Contrôle Optimal de systèmes gouvernés par des équations aux dérivées partielles.* Dunod-Gauthier Villars, 1968.

[4] LIONS P.L. : On the Schwarz alternating method. I,II,III. Int Symposium on Domain decomposition Methods for Partial Differential Equations. SIAM, Philadelphia, 1988,89,90.

[5] LIONS J.L., PIRONNEAU O., Algorithmes parallèles pour la solution de problèmes aux limites, C.R.A.S., 327, pp 947-352, Paris 1998.

[6] E. Polak, "On the Use of Consistent Approximations in the Solution of Semi-Infinite Optimization and Optimal Control Problems", *Mathematical Programing*, Series B, Vol. 62, No.2, pp 385-414, 1993.

[7] E. Polak, Optimization: *Algorithms and Consistent Approximations*, Springer-Verlag, New York, 1997.

[8] R. T. Rockafellar and R. J-B. Wets, *Variational Analysis*, Springer-Verlag, Heidelberg, 1997.