

# Chapter 12

## A LOCAL MINIMIZER

### 12.1 Introduction

In this chapter, we repeat the computations of Chapter 11, but now, in the computation of the cost gradients, the sensitivities are approximated by finite difference quotients rather than with discretized sensitivities.

We expect this approximation to be better, and indeed, the optimization code now converges properly to a minimizer. However, the minimizer is not the global minimizer corresponding to the target data, but is instead a local minimizer. We exhibit graphical evidence of a “barrier” between the this local minimizer and the global minimizer, which allows the local minimizer to trap the optimization code.

We then consider whether this local minimizer is a *spurious solution*, that is, a mathematical solution for the discrete problem which does not correspond to any physical solution. We carry out this investigation by refining the finite element mesh and examining the behavior of the refined finite element solution.

Table 12.1: Results for  $\mathcal{J}_2^h$ , using finite difference gradients.

<b>TolOpt</b>	Steps	$\lambda$	$\alpha_1$	$\alpha_2$	$\alpha_3$	$\mathcal{J}_2^h$	$\ \nabla \mathcal{J}_2^h\ _\infty$
1.0E-09	44	0.507739	0.1407	0.5394	0.0599	0.311E-06	0.2E-09

## 12.2 Optimization Using Finite Difference Gradients

The experiments described in this chapter were all carried out on the same four-parameter problem used in Chapter 11. The algorithm was the same, except for one difference. Where previously we had estimated the sensitivities  $(u^h)_\beta$  with the discretized sensitivities,  $(u_\beta)^h$ , we now computed direct finite difference quotient approximations  $\frac{\Delta u^h}{\Delta \beta}$ .

This is an expensive process, of course. Before, each step of the optimization iteration required one flow solve. Now, each step will require an additional flow solve for every parameter. In general, we would dearly like to avoid such wasteful computation by using discretized sensitivities. But for now, we need to do the extra work in order to understand whether the discretized sensitivities failed because of their limited approximation power.

The sensitivities, as estimated by finite differences, were used in Equation (9.11) to produce an approximation to the cost gradients  $\frac{\partial \mathcal{J}_2^h}{\partial \beta_i}$ . It was hoped that  $\frac{\Delta u^h}{\Delta \beta}$  would give a better estimate of  $(u^h)_\beta$ , and that we would therefore arrive at a reliable approximation of the cost gradients.

As before, the optimization code was given a zero starting estimate for the parameters, and the cost functional at this starting point was  $\mathcal{J}_2^h(0,0,0,0) = 0.430$ .

We present the results of these computations in Table 12.1. The results presented are for an optimization tolerance of **TolOpt**=1.0E-9, but essentially identical results were achieved for tolerances of 1.0E-10, 1.0E-11 and 1.0E-12.

The first thing we check in these results is the parameter values, and we see that, as with

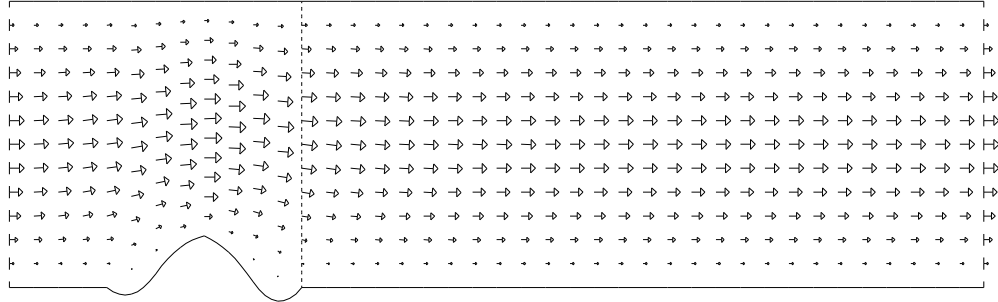


Figure 12.1: The local optimizing flow.  
 Gradients computed with finite difference sensitivities.  
 The mesh parameter  $h$  is 0.25.

This flow may be compared with the target flow, in Figure 11.1.

the discretized sensitivity calculation, we have been unable to reproduce the original target parameters.

We then turn to the cost functional information, and here we see that the optimization has probably been successful, at least in terms of returning a local minimizer. The cost functional has dropped by 6 orders of magnitude, from 0.430 to  $0.311E - 06$ , and the gradient has dropped extremely close to zero, signaling that very little “local” improvement can be made to this solution point.

Thus, it seems we may have stumbled upon an unexpected solution to the problem. The parameters don’t seem far from the target values. We see in Figure 12.1 that the computed bump is not too far from the target bump, although it still has the “gutters” we saw in the previous chapter.

But before we accept this solution, we would like to repeat the tests of the previous chapter, and convince ourselves that what we are seeing this time is a real, acceptable, local minimizer for the optimization problem, and that this minimizer tells us something about the continuous physical problem whose behavior we are approximating.

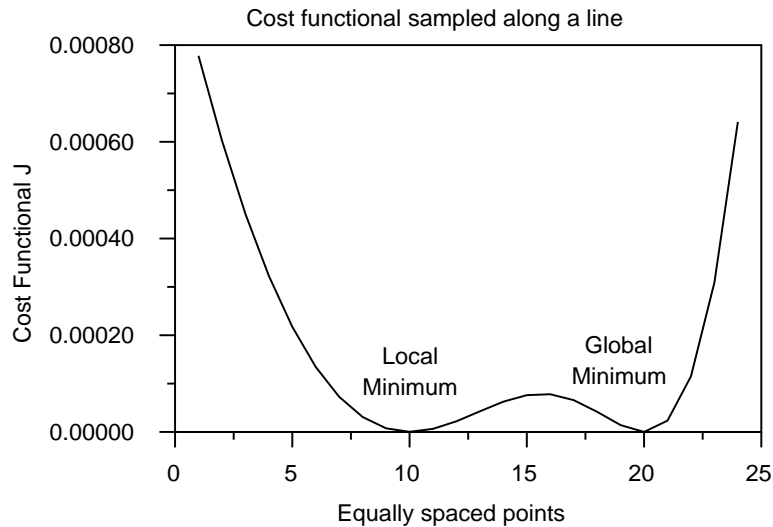


Figure 12.2: Values of  $\mathcal{J}_2^h$  on a line between the local and global minimizers.

### 12.3 Checking for Local Minimization

While it was natural to assume that the optimization code had halted at a local minimizer, there were some simple checks that could be made to verify this. For instance, it was possible that the optimization code had halted at the point because the functional was so flat that there was no discernible difference between functional values at that point and the global minimizer, or any points in a “plain” nearby. But this possibility was quickly dispelled by computing the cost functional at a series of intermediate points on the line between the optimization code’s solution and the global minimizer. The graph in Figure 12.2 shows how the value of the cost functional rises from  $\mathcal{J}_2^h = 0.3E - 06$  at the optimization code’s solution to an intermediate value of  $\mathcal{J}_2^h = 0.7E - 04$  before falling to  $\mathcal{J}_2^h = 0$  at the global minimizer.

The graph strongly suggests that a local minimizer has been found. However, this is not guaranteed by any means. We have only sampled the functional along a direct line, whereas the space of parameters has four dimensions, meaning there could yet be a curve connecting the two points along which the functional is strictly decreasing. We would not expect the

optimization software to miss an obvious descent direction, but we will feel more confident that we have found a local minimizer if we sample the functional in a plane and still see a barrier of higher functional values blocking access from the optimization code’s solution to the global minimizer. Figure 12.3 displays such a plot, and the barrier is clearly visible. The solution returned by the optimization code lies in the “well” formed in the lower left hand corner, and the global minimizer is in the well in the upper right. Once the optimization code falls into the well in the lower left, it will not be able to rise out. Instead, its goal should be to seek the local minimizer in the well, which it does.

This graph, in which the computed minimizer correctly lies in the bottom of a local depression, should be compared with the unsatisfactory situation portrayed in Figure 11.5, where the optimization code stopped in confusion on a ridge, moving neither to the local nor global minimizers.

Finally, to convince ourselves that the cost gradient data is compatible with the cost functional, we display the (negative) normalized gradient vectors over the contour plot, in Figure 12.4. The plot makes it evident that the gradients computed using finite difference sensitivities correspond properly to the contour lines of  $\mathcal{J}_2^h$ , crossing them perpendicularly, and pointing inward toward the local and global minimizers.

## 12.4 The Possibility of Spurious Solutions

Solving the discrete problem has given us two solutions to the minimization problem: the global minimizer that we had built into the problem, and the unexpected local minimizer. We have no doubt that the global minimizer found for the discrete problem corresponds precisely to the global minimizer that would be found if we could optimize the continuous problem. But it is not certain whether the local minimizer really has physical meaning, or

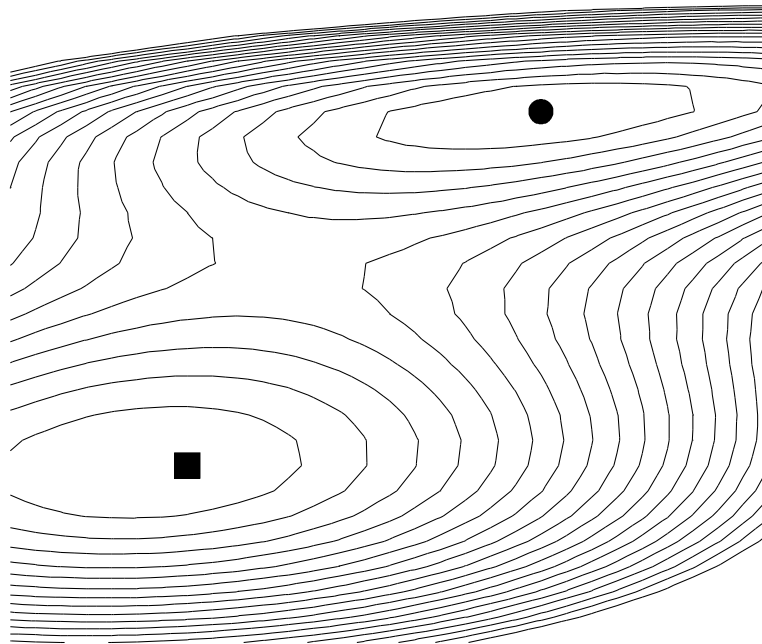


Figure 12.3: Contours of  $\mathcal{J}_2^h$  on a plane containing the two minimizers.  
The global minimizer is the black dot.  
The local minimizer is the black square.

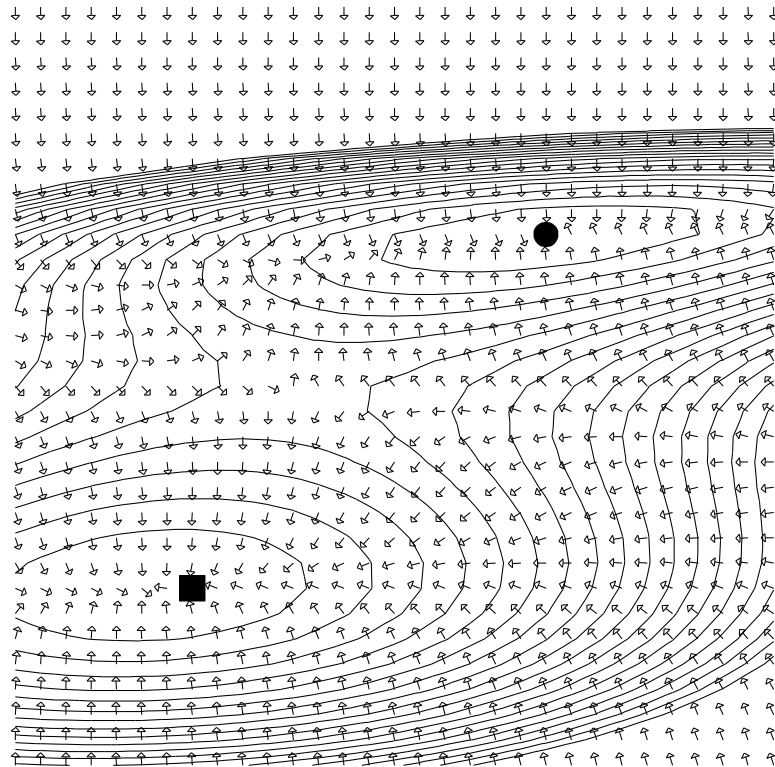


Figure 12.4: Contours and normalized gradients of  $\mathcal{J}_2^h$ .  
 Gradients computed using finite difference sensitivities.  
 The global minimizer is the black dot.  
 The local minimizer is the black square.

Table 12.2: Investigating the local minimizer with finer meshes.

$h$	$\lambda$	$\alpha_1$	$\alpha_2$	$\alpha_3$	$\mathcal{J}_2^h(\lambda, \alpha)$
0.250	0.507739	0.14079	0.539404	0.059978	0.311E-06
0.166	0.508730	-1.46097	0.359595	-0.084067	0.104E-04
0.125	0.511107	-1.91279	0.298398	-0.290631	0.126E-04

is merely a *spurious solution*, that is, a mathematical artifact, a solution to the discrete problem that does not correspond to any physical solution.

It can be quite difficult to tell whether a given solution to a discretized problem has a physical meaning. For instance, we can convince ourselves that the “gutters” in the local minimizer’s bump shape might perform the function of separating off regions of recirculating flow, so that some streamline actually comes close to reproducing the target bump shape.

On the other hand, we have grounds for disbelieving this solution. The guttering causes a substantial distortion of the relatively coarse finite elements, which makes the computational results in those elements less reliable.

One way to investigate this question is to examine the same problem on a sequence of finer meshes. If there actually is a true, physically meaningful, locally optimizing solution to the continuous problem corresponding to the discrete local minimizer we found, then we expect that each time we refine the mesh, we will again find a locally minimizing discretized solution, and that the sequence of these solutions will converge to the physical solution. Without actually knowing whether there is a true, limiting solution, we are testing the solutions by performing a rough sort of *Cauchy convergence test*, demanding that the successive iterates begin to cluster together.

The original solution was computed with mesh parameter  $h = 0.25$ , corresponding to 41 nodes along the  $x$  direction and 13 along  $y$ . To investigate the solution we found on this mesh, we set up two finer meshes, starting these optimizations at the local minimizer  $\lambda = 0.507$ ,

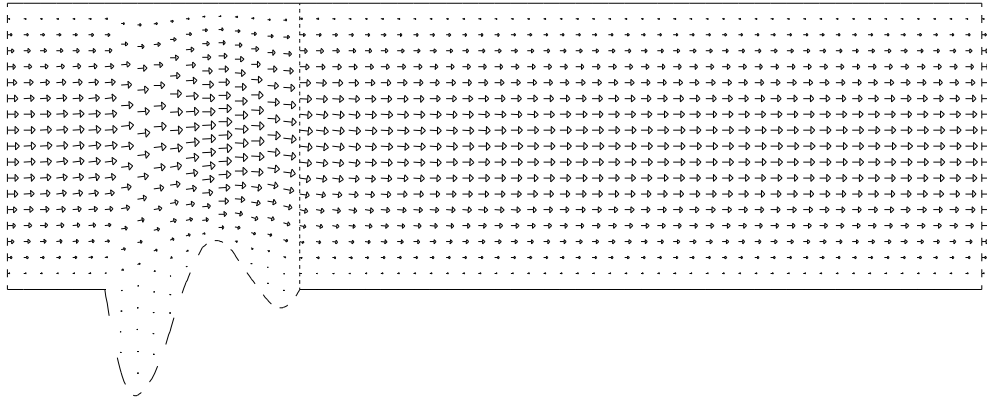


Figure 12.5: The flow for  $h = 0.166$ .

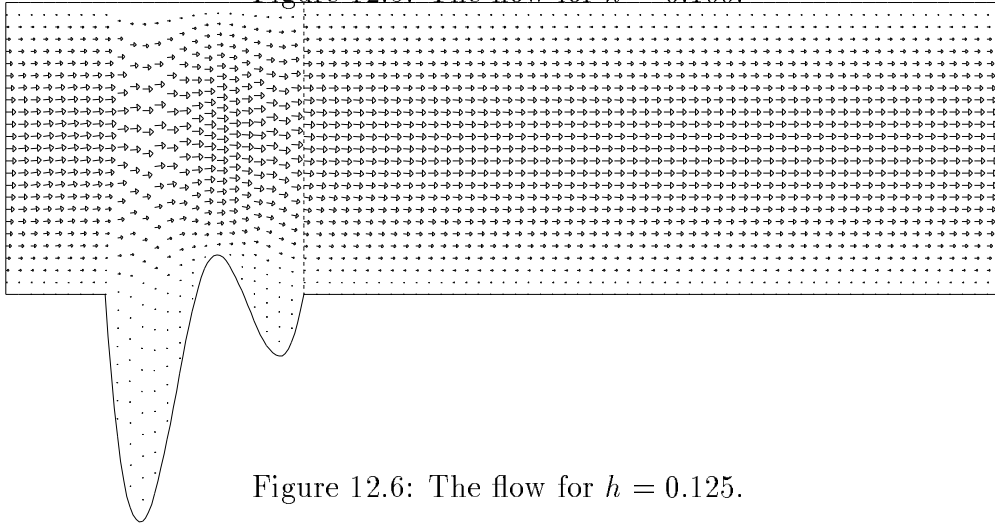


Figure 12.6: The flow for  $h = 0.125$ .

$\alpha = (0.140, 0.539, 0.059)$ , and allowing up to 65 optimization steps. The numerical results are displayed in Table 12.2. In each case, the optimization code concluded that it had converged to an acceptable minimizer, but in both cases this minimizer had much deeper “gutters” than the original solution.

The solutions for  $h = 0.166$  and  $h = 0.125$  are shown in Figures 12.5 and 12.6. These flows may be compared with the flow solution for  $h = 0.125$  in Figure 12.1.

The results from these runs on finer meshes suggest that the local minimizer is a spurious solution. Instead of staying put, the gutters have gotten much deeper. The fact that the gutters are so deep and narrow also means that all the elements above the gutters are

severely distorted, so that we cannot place much confidence in the computed flow results either. Since this local minimizer doesn't seem to represent an interesting physical solution, we are no longer interested in it, except as a trap which we wish to avoid. We now consider how to modify the approach so that the optimization can reach the global minimizer, avoiding such meaningless solutions as we have just encountered.