

# Chapter 15

## APPROXIMATING NONFEASIBLE TARGETS

### 15.1 Introduction

The fundamental information defining our cost functional is the profile data, that is, the values  $u^T(x, y)$  (and possibly  $v^T(x, y)$  and  $p^T(x, y)$ ) specified along the profile line  $x = x_s$ . Usually, this data is generated by specifying some set of parameters  $\beta^T$ , solving for the corresponding flow  $(u^T(\beta), v^T(\beta), p^T(\beta))$ , and saving the flow values along the profile line.

In such a case, we speak of  $(u^T, v^T, p^T)$  as a *feasible target*. This terminology is meant to imply that the profile data was generated from a discrete fluid flow, that this fluid flow was generated by solving a parameterized flow problem, and that the original set of parameters is part of the feasible set of parameters to be examined by the optimization code. In other words, if the target is feasible, the generating parameters are buried somewhere in the feasible set, if only the optimizer is smart enough to find them.

Hence if we generate our problems in this way, we know beforehand that the minimum value of the discrepancy cost functional will be zero, and that this minimum value certainly occurs for the flow solution corresponding to the parameter values  $\beta^T$ , although it may

occur for other parameter values as well. For that matter, we could also have numerous local minimizers with higher values of the cost.

However, there is no requirement that we set up our cost functional data in this way. In a real world application, it is likely that we would not know the form of any global or local minimizer beforehand. It is also likely that we would prescribe what might be termed an *unfeasible target*; that is, profile data for which there is no corresponding set of parameters that would produce a flow with that profile.

We would like to investigate some simple problems involving unfeasible targets. We will consider some problems where the target data is generated from a flow that depends on parameters, but where the spaces spanned by the parameters are different than those used for the optimization. We will also consider some problems where the target data is simply produced, *data ex machina*, with presumably no corresponding flow solution.

In these cases, we will still expect the optimization code to produce a locally minimizing solution, but we have no idea beforehand what the minimal value will be, nor what the form of the solution will be. In cases where our target data is chosen too arbitrarily, we may find that the optimization results are counter-intuitive.

## 15.2 Example 1: Differing One-Parameter Bump Shapes

For our first example, let us suppose that the target flow is generated in a flow region whose bump is a piecewise quadratic, whereas the feasible space will only consider bumps which are piecewise linear. To keep things simple, we will only use a single bump parameter,  $\alpha = 0.5$ , and we will fix the inflow parameter  $\alpha$  at 0.5, and the Reynolds parameter at 1. When we give this problem to our program, the optimizer converges to the solution  $\alpha = 0.6498$  in 14 steps, with a cost functional value of  $\mathcal{J}(\alpha) = 0.62E - 4$ . Of course, we didn't expect the two

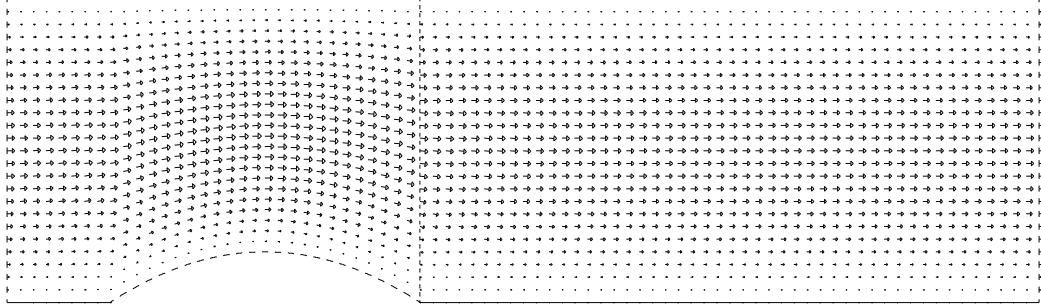


Figure 15.1: Example 1: The target flow.  
A piecewise quadratic bump is used.

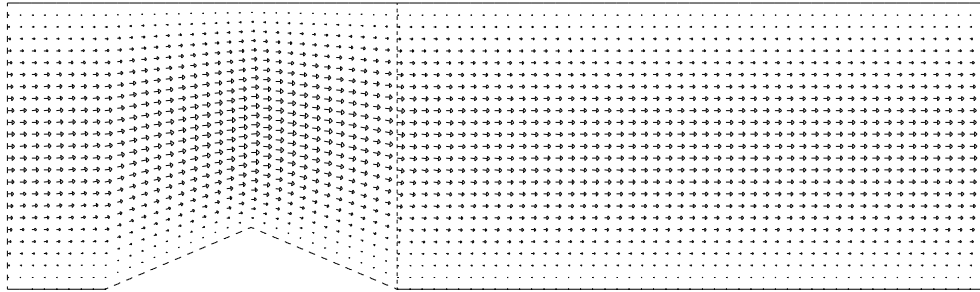


Figure 15.2: Example 1: The optimizing flow.  
Only piecewise linear bumps are feasible.

values of  $\alpha$  to agree, since they are coefficients for different sets of piecewise polynomials. The real tests are in the shape of the bump, and the functional match.

From Figures 15.1 and 15.2, it's easy to see that the piecewise linear bump is “trying” to match the shape of the piecewise quadratic, but it's nearly impossible to see and compare the horizontal velocities along the profile line. To rectify that problem, in Figure 15.3 we plot the horizontal velocity functions for the target and optimizing flows, along the profile line. The target values are drawn with a dashed line, but the match is so good between the two functions that it's hard to see.

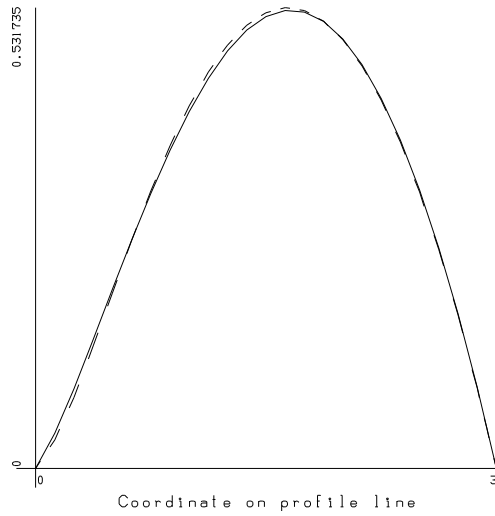


Figure 15.3: Example 1: Comparison of target and achieved flows.

Table 15.1: Example 1: Optimization results.

| Steps | $\alpha$ | $\mathcal{J}_2^h$ | $\ \nabla \mathcal{J}_2^h\ _\infty$ |
|-------|----------|-------------------|-------------------------------------|
| 0     | 0.000    | 0.00607           | –                                   |
| 15    | 0.649    | 0.00006           | 0.2E-07                             |

Table 15.2: Example 2: Optimization results.

| Steps | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\mathcal{J}_2^h$ | $\ \nabla \mathcal{J}_2^h\ _\infty$ |
|-------|------------|------------|------------|-------------------|-------------------------------------|
| 0     | 0.000      | 0.000      | 0.000      | 0.343E-01         | –                                   |
| 10    | 0.033      | 0.175      | 0.845      | 0.266E-03         | –                                   |
| 20    | 0.151      | 0.666      | 0.859      | 0.179E-03         | –                                   |
| 30    | 0.230      | 0.996      | 0.794      | 0.591E-05         | –                                   |
| 40    | 0.231      | 1.000      | 0.791      | 0.575E-05         | –                                   |
| 50    | 0.232      | 1.000      | 0.792      | 0.575E-05         | –                                   |
| 60    | 0.471      | 1.006      | 0.793      | 0.574E-05         | –                                   |
| 70    | 0.595      | 1.009      | 0.794      | 0.572E-05         | –                                   |
| 80    | 0.599      | 1.010      | 0.793      | 0.571E-05         | –                                   |
| 90    | 0.995      | 1.022      | 0.798      | 0.544E-05         | –                                   |
| 100   | 1.161      | 1.006      | 0.805      | 0.515E-05         | –                                   |
| 110   | 1.246      | 0.999      | 0.806      | 0.488E-05         | –                                   |
| 120   | 1.364      | 0.988      | 0.812      | 0.458E-05         | –                                   |
| 130   | 1.638      | 0.886      | 0.823      | 0.378E-05         | –                                   |
| 135   | 1.638      | 0.885      | 0.823      | 0.378E-05         | 0.2E-08                             |

### 15.3 Example 2: Differing Three-Parameter Bump Shapes

After the results of our first example, we are interested in pushing the problem a little further. We will essentially repeat the calculation, but with three parameters used to represent the bumps. Also, we will make the target bump higher, using the values  $\alpha = (0.75, 1.0, 0.75)$ .

From the optimization history in Table 15.2, it seems reasonable to believe that the minimal cost, and the values of the second and third bump parameters have been well approximated. Looking at the behavior of the bump parameters, it seems as though they are determined from right to left, with the bump parameter closest to the sampling line presumably having the biggest influence, and hence being determined first. We will accept our results, while

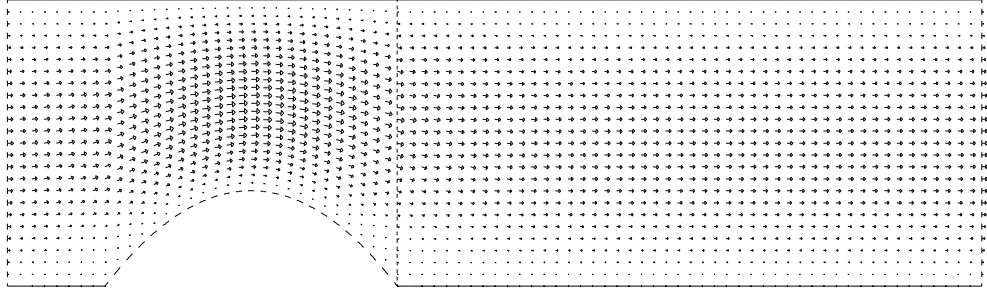


Figure 15.4: Example 2: The target flow.  
A piecewise quadratic bump is used.

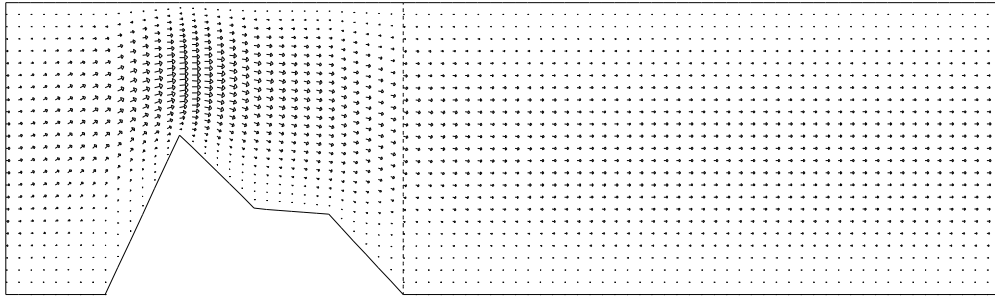


Figure 15.5: Example 2: The optimizing flow.  
Only piecewise linear bumps are feasible.

realizing that there may be some uncertainty in the value of  $\alpha_1$ .

From Figures 15.4 and 15.5, it's easy to see that the bumps are dissimilar in shape. The piecewise linear bump is by no means the interpolant of the quadratic bump, nor does it seem close to one.

Of course, since the target isn't feasible, we can't judge our results by how well the bumps match, but should instead refer to the match of horizontal velocities along the sampling line. In our plots which show the entire region, this is very difficult to see. To rectify that problem, in Figure 15.6 we plot the horizontal velocity functions for the target and optimizing flows,

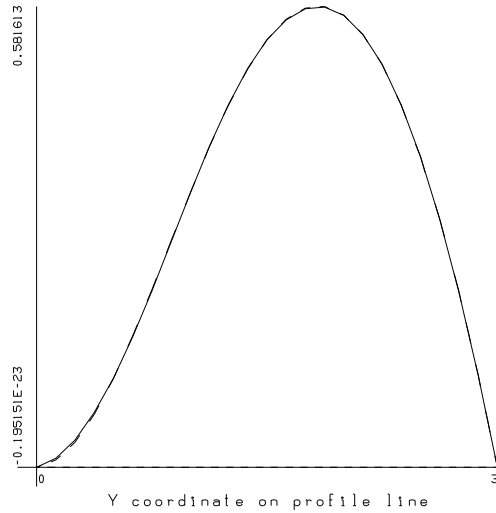


Figure 15.6: Example 2: Comparison of the target and minimizing flows.

along the profile line. The target values are drawn with a dashed line, but even now the discrepancy between the two is difficult to spot.

It may be of interest to compute the flow solution generated by the linear interpolant to the quadratic bump, which would have been a natural guess for the minimizing parameters. The results are shown in Figure 15.7, where the discrepancy between the target and computed flows is clear. The value of  $\mathcal{J}(\alpha)$  is 0.1E-03, which certainly shows that our minimizing solution has done better.

## 15.4 Example 3: Three-Parameter Inflows

In our next example, we leave the bump fixed, but allow the inflow to vary. We model the shape of the inflow by three parameters. Such a varying inflow might be generated by using

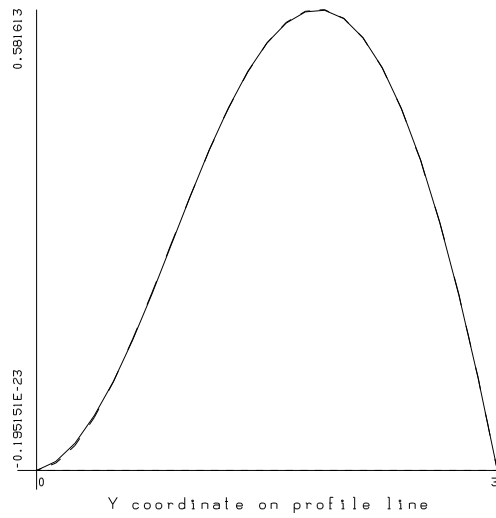


Figure 15.7: Example 2: Comparison of the target and linear interpolant flows.

Table 15.3: Example 3: Optimization results.

| Steps | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\mathcal{J}_2^h$ | $\ \nabla \mathcal{J}_2^h\ _\infty$ |
|-------|-------------|-------------|-------------|-------------------|-------------------------------------|
| 0     | 0.1000      | 0.1000      | 0.1000      | 5.974             | –                                   |
| 22    | 1.9608      | -2.0536     | 2.9955      | 0.128E-01         | 0.9E-08                             |

several separate fans or deflectors in a wind tunnel.

We generate an arbitrary curve for the horizontal flow at the profile line, using the formula:

$$\begin{aligned}
 u_1(x_s, y) &= (3 - y)y/m_1 \\
 u_2(x_s, y) &= (3 - y)y(1.5 - y)/m_2 \\
 u^T(x_s, y) &= \gamma_1 u_1(x_s, y) + \gamma_2 u_2(x_s, y)
 \end{aligned}$$

where  $m_1$  is chosen so that the maximum value of the parabolic flow  $u_1$  over  $[0, 3]$  is 1, and  $m_2$  is chosen similarly for  $u_2$ . The quantities  $\gamma_1$  and  $\gamma_2$  are chosen by the user at run-time. Since parabolic flow is the natural tendency for our problem, we may think of  $u_2$  as a perturbation to this natural tendency. The larger  $\gamma_2$  is, relative to  $\gamma_1$ , the more “unnatural” the target flow will be, and the harder to achieve. For our demonstration problem, we choose  $\gamma_1 = 2$ ,  $\gamma_2 = -0.5$ . Figure 15.8 is a graph of the resulting target profile.

Now the value of the target flow along the profile is all we need to define an optimization problem, so we can proceed to solve this problem, using a Reynolds number  $Re = 10$ . Because the target flow was not generated in the usual way, we expect that our optimizing flow will not achieve a zero cost functional, although we do want the cost gradients to be close to zero. A summary of our results are in Table 15.3.

In Figure 15.9 we show the optimizing velocity field near the profile line, and in Figure 15.10 the optimizing velocity field at the inflow, where it is plain that a certain amount of *outflow* is generated. Finally, in Figure 15.11 we show the excellent match between the target and optimizing velocities.

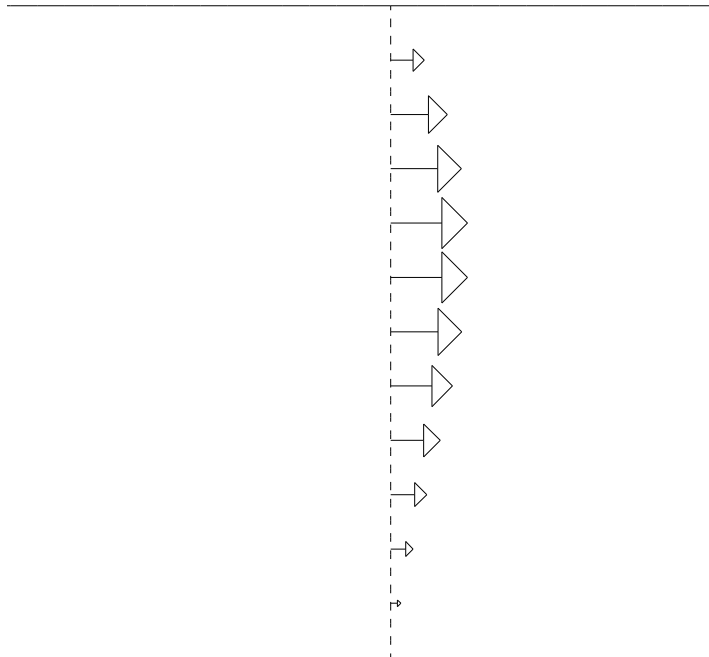


Figure 15.8: Example 3: The “artificial” flow.  
Values  $\gamma_1 = 2.0$ ,  $\gamma_2 = -0.5$  were used.

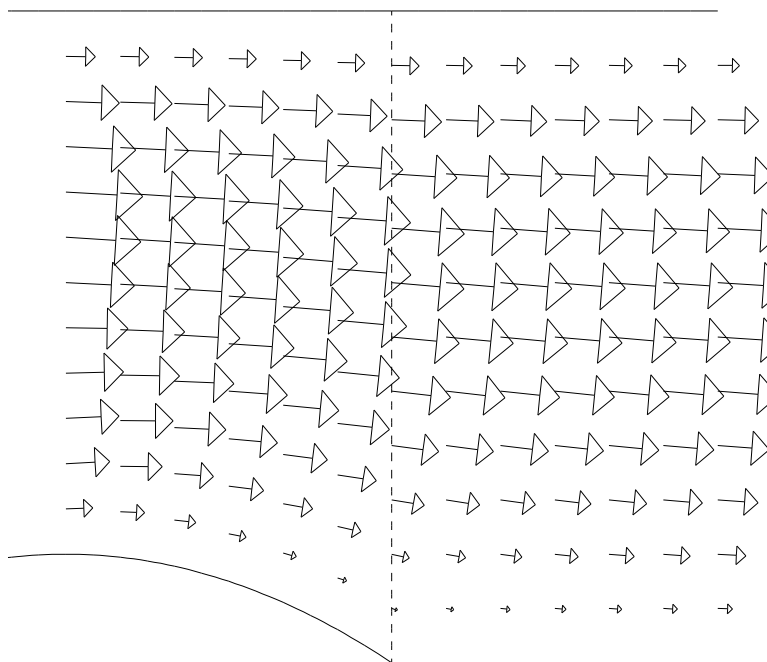


Figure 15.9: Example 3: The optimizing flow, near the profile line.

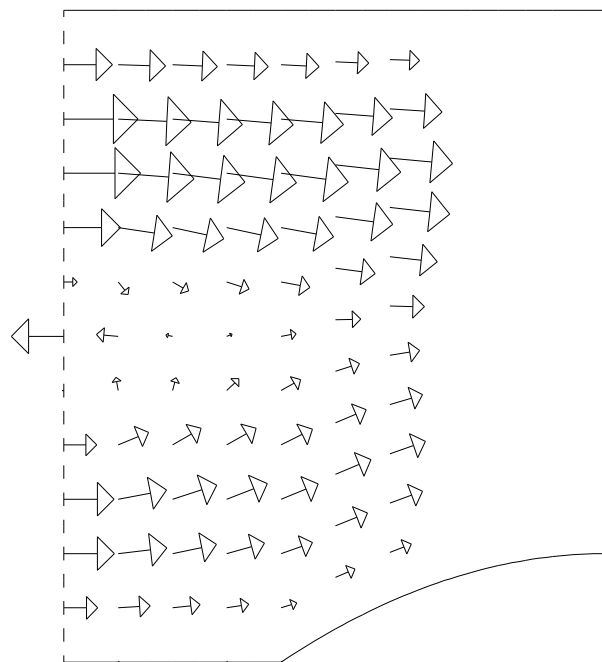


Figure 15.10: Example 3: The optimizing inflow (includes outflow!).

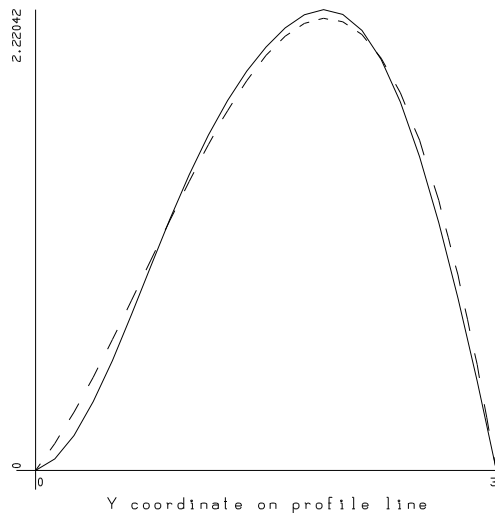


Figure 15.11: Example 3: The match between target and desired flows.