EVALUATION OF LEVEL SET TOPOLOGY OPTIMIZATION FORMULATIONS FOR DESIGN OF MINIMUM-DISPERSION MICROFLUIDIC DEVICES

A.R. TERREL* AND K.R. LONG†

Abstract. Applying topology optimization to minimize the sample dispersion of a microfluidic device is challenging in part because the fictitious permeability used to classify topological regions tends to a smooth non-Boolean field, in contradiction to the requirement that a valid solution be Boolean. Imposing the Boolean condition must be done carefully in order to avoid spurious local minimizers or an excessively smooth solution. Here we investigate the behavior of two different formulations of level-set based topology optimization on problems where we must find the Boolean field that best approximates a specified non-Boolean field. We introduce a formulation based on inequality constraints that can effectively satisfy the Boolean condition.

1. Introduction. The goal in shape optimization is to identify the shape of a domain $\mathcal{I}$ such that when a physical problem is solved on that domain, an objective function is minimized. The most general form of shape optimization is topology optimization [2], in which the topology of the domain is allowed to vary; *i.e.,* topological holes and/or islands can appear or disappear. By its nature, topology optimization is a large-scale boolean programming problem, because each point in a larger, embedding domain is either in $\mathcal{I}$ or not. However, in practice it is rarely approached directly in that form; rather, each point in the domain is characterized by a real number to be driven, somehow, to the boolean extremes of 0 or 1. Much of the art in topology optimization is choosing this reparametrization in such a way closely approximates a boolean field without introducing spurious local minimizers.

In many problems, there is a natural reparametrization that is simple and works well. In structural topology optimization, for example, if the relationship between material density and stiffness is taken to be nonlinear and monotonic, then for certain objective functions there will be no payoff to intermediate material densities. In other words, in such problems the nature of the physical problem, objective function, and constraints drives the material density naturally to a boolean solution [2]. In this case, nothing special has to be added to the method to produce a boolean solution.

Here, though, we are interested in problems where no such lucky accident happens. The motivating problem for this study is the design of minimum-dispersion microfluidic sensors; by minimum-dispersion we mean that a sample of particles that enters the device together will not be dispersed by the flow. We represent the shape of the channel by introducing an approximately-boolean permeability. The challenging feature of this problem is that for the physics and objective function under consideration, the solution to this problem is a permeability that varies smoothly between the boolean values of zero and one, with most of the domain being at intermediate permeabilities. Such a device cannot be manufactured. In this problem, then, we must somehow externally – by means of a penalty or constraint – enforce the boolean condition. The obvious way to do this is with a penalization on any non-boolean values, but that is easily seen to be non-convex, and indeed, in practice it is found that such a penalization introduces numerous artificial local minimizers.

An improved formulation was introduced by Cunha in his thesis [4]. Cunha penalizes the *slope* of a level set function, which, as will be discussed below, allows

*University of Chicago, aterrel@uchicago.edu
†Sandia National Laboratories, krlong@sandia.gov

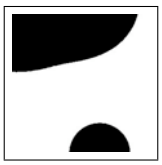


FIG. 2.1. *Example of a domain Ω with the black representing the exterior $\mathcal{E}$ and the white the interior $\mathcal{I}$.*

fine control of the transition region where non-boolean values can occur. We will refer to this method as the slope penalty method. Penalizing the gradient of a function is less restrictive than penalizing the function itself, so that we expect (and find) fewer artificial local minimizers with this method than with a direct penalization of non-boolean permeabilities. However, the formulation still does have artificial minimizers, and it is difficult to choose a penalty parameter loose enough to avoid these yet tight enough to enforce the boolean condition.

We have thus introduced another method, which we call the slope barrier method, in which a certain inequality constraint (to be described below) is imposed on the level set function and its slope. This inequality constraint is an even looser restriction on the problem, suggesting less problem with local minimizers, yet it can more strongly enforce the boolean condition.

In this paper we will compare the slope penalty and slope barrier method on model shape-matching problems having the difficult characteristics of minimum-dispersion microflow. In section 2 we show a level set formulation of the shape optimization problem. We motivate and describe the two methods to be compared in sections 2.1 and 2.2. We give some details in order to implement and optimize the objective function in section 3. Finally we give testing methods and results of our comparison in sections 4 and 5.

2. Formulation of the Problem. The formulation of our problem is the same as outlined in Cunha's thesis [4]. For topology optimization, we have a domain, Ω, in our case this is a square array of pixels, and want to partition it into two different subsets: the exterior $\mathcal{E}$ and the interior $\mathcal{I}$, see Figure 2.1. The interior, $\mathcal{I}$, is used to represent portion of Ω that will be the domain of a PDE equation such as the flow in our microfluidics application. The exterior, $\mathcal{E} = \Omega - \mathcal{I}$, is the rest of Ω. The interface between the two regions is given by the zero contour of a level set function, $\phi(\mathbf{x})$,

$$\phi : \mathbf{x} \in \Omega \to \mathbb{R}$$

The level set function defines a geometric shape one dimension higher than Ω, for our purposes $\phi(\mathbf{x}) \geq 0$ indicates $\mathbf{x} \in \mathcal{E}$, and $\mathbf{x} \in \mathcal{I}$ otherwise. Thus our shape is determined by the distribution χ:

$$\chi(\phi) = \begin{cases} 0 & \text{if } \phi < 0 \\ 1 & \text{otherwise} \end{cases}$$

To reach the goal of our topology optimization, we are concerned with the values of the χ distribution. Optimizing on a large boolean problem is quite hard, so we relax the problem by replacing it with a sigmoid function that maps to the interval $[0, 1]$ on the reals. The sigmoid function is a smooth interpolation of χ.

$$\sigma(\phi) = \frac{1}{2}\left(1 + \tanh\left(\frac{\phi}{\Delta}\right)\right)$$

where Δ is a given parameter. The parameter Δ gives some control over the transition of the sigmoid from values near 0 to values near 1, but this will be discussed further in Section 2.1 with the introduction of the slope penalty method.

This formulation offers several advantages over a more traditional shape optimization with parametric curves. First, the design variable that we manipulate in our optimization algorithm is ϕ and not a boundary of our mesh. This allows us to avoid remeshing because of topology changes. This also allows us to use a very rich design space without limits on the curves we produce. Finally, using a smooth sigmoid function insures that derivatives exist and give us the opportunity to use faster optimization algorithms.

While there are several advantages with this formulation, the problem is ill posed. The problem statement uses only the zero contour of the level set function ϕ, yet there are infinitely many level set functions that have the same zero contour. Second, the level set function has no restriction to be smooth. It could be a zig zag of an infinite number of sinusoidal functions or anything else as perverse. By adding some regularization to ϕ, we are able to guide the optimization to use certain types of functions. We use a Tikhonov regularization for ϕ, to produce a smooth function [1]. For our applications this regularization will be added by the term:

$$\frac{\alpha_1}{2}\int_\Omega |\nabla\phi|^2 d\Omega$$

We add another term for the regularization of σ. In our microfluidics application, it also was important that we have a smooth shape in order to have a shape that is manufacturable. To achieve this we use a total variation diminishing regularization on σ:

$$\alpha_2\int_\Omega (\nabla\sigma^2)^{\frac{1}{2}}$$

The total variation diminishing regularization is used because Tikhonov is restrictive of jumps in slope [1].

2.1. Slope Penalty Method. Because we relaxed χ to a sigmoid function, there is a range for which our results will not be near 0 or 1. Let us call this range the bandwidth, Λ. In other words Λ is the measure of the set $\{\mathbf{x}\in\Omega|\varepsilon<\sigma(\phi(\mathbf{x}))<1-\varepsilon\}$ for some given ε. This means that Λ is dependent on both Δ and $|\nabla\phi|$. To enforce the criteria that the resulting shape be as boolean as possible, we prefer to have Λ small. In order to control this at runtime we wish Λ dependent only on Δ. This could be achieved by the constraint $|\nabla\phi|=1$, but such a constraint would be inhibitive of the Tikhonov regularization. To balance between these two controls on $\nabla\phi$, we use a penalty method to give an approximate constraint,which is characterized in the objective function by adding the following term:

$$\frac{\beta}{4}\int_\Omega (|\nabla\phi|^2-1)^2 d\Omega$$

This approximate constraint is more a concession that we want $|\nabla\phi|\approx 1$. The β term allows us to control how much to enforce this approximate constraint, when $\beta>\alpha_1$ we enforce the approximate constraint more, and when $\beta<\alpha_1$ we enforce Tikhonov regularization more. This penalty gives a method to increase or decrease the bandwidth of σ at runtime with the Δ parameter, with respect to our level of Tikhonov regularization.

2.2. Slope Barrier Method. The slope penalty method controls Λ by setting a restriction of the slope of ϕ over all of Ω. However, the slope outside the bandwidth does not affect the value of σ since it will be closer to 0 or 1 than our tolerance; any work done in meeting the unit slope condition outside the transition band is therefore wasted. We therefore introduce an alternative method, in which instead of a penalty we use an inequality constraint:

$$\left(\frac{\phi}{\Delta}\right)^2 + (\nabla\phi)^2 \geq 1 - \varepsilon$$

This term strictly regulates the slope of ϕ near the zero contour yet lets it vary freely elsewhere.

To enforce this inequality constraint we use a barrier method, in which we put a singular barrier at the boundary between the feasible and infeasible regions. This lets us use an unconstrained optimization algorithm. Rather than the more conventional log barrier [7], we use a piecewise but differentiable barrier that goes to zero a small distance from the constraint surface. The sharpness of the curve for the barrier is a controlled by the weight factor γ. The larger we allow γ the larger the tail leading up to the barrier, which will possibly put some control on the size of Λ allowed.

3. Implementation. In the previous section, we have introduced ways of describing shapes in terms of a level set function, and means of regulating the smoothness and the sharpness of the boolean transition. We also need to add some measure of quality of the shape produced, *i.e.* an objective function. For testing purposes we are only going to look at matching a specified target shape, $\mathcal{E}^*$, and thus we will use a Heaviside Distance function to compare our shape with the target shape, that is:

$$d(\mathcal{E}, \mathcal{E}^*) = \frac{1}{2}\int_\Omega (\sigma - \sigma^*)^2 d\Omega$$

In addition, we will have the terms described above.

Now our full objective function consists of the following terms: a Heaviside Distance, a Tikhonov Regularization for ϕ, a total variation regularization for σ, a slope penalty term. and a slope barrier term. Or in equation form:

minimize the function $\mathcal{F}$ where:

$$\mathcal{F} = \frac{1}{2}\int_\Omega (\sigma - \sigma^*)^2 d\Omega + \frac{\alpha_1}{2}\int_\Omega |\nabla\phi|^2 d\Omega + \alpha_2 \int_\Omega (\nabla\sigma^2)^{\frac{1}{2}} d\Omega + \\ + \frac{\beta}{4}\int_\Omega (1.0 - \nabla\phi^2)^2 d\Omega$$

augmented to the inequality constraint:

$$\nabla\phi^2 + \left(\frac{\phi}{\Delta}\right)^2 \geq (1 - \varepsilon)$$

To test our different controls of the slope penalty and the barrier weight we can adjust β and γ, respectively. Since we are not concerned so much about the level of the regularization terms, we have fixed both α_1 and α_2 at 1.0×10^{-4}. Doing this we have a baseline configuration, $\beta = \gamma = 0$, for when the Tikhonov regularization will be the dominant contribution to the control of ϕ. Now we turn our attention the optimization algorithms that we used.

3.1. Local Optimization. For local optimization we use the adaptive limited-memory BFGS algorithm of Byrd and Boggs [3].

We tried to use ways of controlling ϕ and σ so that there would be few artificial minimizers, that is minimizers that are not physically significant but that have been introduced by our slope control methods. But it must be noted that there is a trade off between the matching of our shape and keeping a boolean result. Furthermore, in many problems there will also be physically significant local minimizers. We have therefore found it necessary to embed the local optimization routine in an outer global optimization loop.

3.2. Global Optimization. Generally speaking, our global optimization scheme is that upon each successful local minimization, we try to "tunnel" in a randomly-chosen direction to a lower function value. However, doing so blindly would be a hopeless task with very low probability of finding a fruitful search direction; the reason for this is that the vast majority of random perturbations one could make to the function ϕ are at high spatial frequency, having little macroscopic effect on the objective function. To find search directions giving a macroscopic effect, we favor perturbations at low spatial frequencies, which is easily done by means of a truncated Fourier series.

We generate search directions by updating our design variable ϕ as follows:

$$\phi \leftarrow \phi + \sum_{m=-M,n=-N}^{M,N} A_{m,n} e^{-i\left(\frac{\pi m}{L}x + \frac{\pi n}{L}y\right)}.$$

Here L is the size of one side of our square grid of pixel and we take an M and N much smaller than the number of pixels per axis. The coefficients $A_{m,n}$ are chosen at random from different points on the complex plane. This give us a new direction that is chosen from the subspace of smoothly-varying random fields.

With a direction computed in this way, we can try to tunnel to a better minimizer by doing a line search in that direction. In practice, searching even in this restricted subspace has a low probability of finding a better minimizer. Therefore, in practice we often conditionally accept an uphill step; the probability for accepting an uphill step can be determined by any convenient distribution; we use a Boltzmann factor. As in simulated annealing [8], the temperature in the Boltzmann factor can be reduced as the procedure progresses, thereby allowing frequent uphill steps early on, but more stringently rejecting them later.

3.3. Software Implementation. The code is implemented using the Sundance [6] and Trilinos [5] software packages.

4. Testing the methods. In order to test the two methods, we start from a variety of images and attempt to match a set of target images, of different degrees of difficulty. To demonstrate some properties of the two methods, we use a host of target images, see Figure 4.1. These images provide a set of complex and simple boolean and non-boolean values to compare. One of the target images is a smooth, non-boolean picture based on the unconstrained solution to a microflow optimization problem; a challenge will be to satisfy the boolean constraint when matching this decidedly non-boolean image.

Another factor of interest is the dependence on the initial guess. We tested this property by running the tests on a series of 1x1, 2x2, 3x3, 4x4, 5x5 grids of circles, see Figure 4.2. We have discovered that to an extent the more disconnected shapes

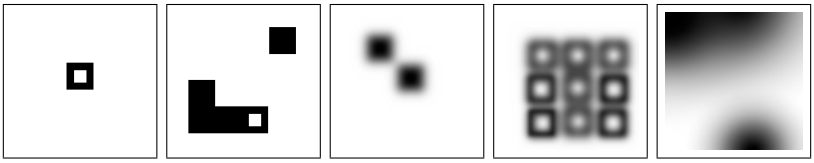


FIG. 4.1. *The target images. (a) Simple boolean image, (b) Complex boolean image, (c) Simple non-boolean image, (d) Complex non-boolean image, (e) Smooth solution to Unconstrained Minimum Dispersion Problem.*

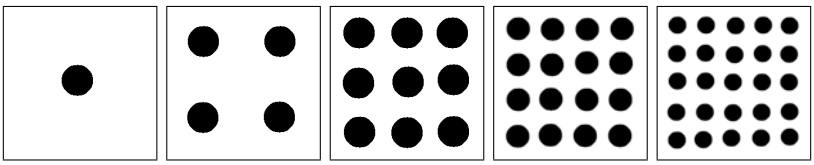


FIG. 4.2. *The input images. (a) 1x1, (b) 2x2, (c) 3x3, (d) 4x4, (e) 5x5.*

initially given to the algorithm, the more accurate the output thus prompting these initial shapes.

The final property studiedis the speed of convergence of the method to a better solution. For each test we see how well it is able to handle a single line search to give a local minimum and a number of additional line searches to find a better global minimum. The first local minimum gives a sense of how the method responds in a head to head comparison, whereas using a method of getting an new direction for the global search will produce quite different directionss to search for each configuration. One problem with a global search is always how long to let it run, whereas the first local line search can be give a stopping criteria based on the convergence of the objective function. For our experiments, we decided to stop the global search if one of the following conditions were met: 1) the objective function was evaluated more than 5000 times, 2) there were more than one hundred directions searched, or 3) after testing 100 directions no direction was found that further reduced the objective function. These criteria were chosen to simulate an engineer optimizing with limited time and computational resources.

5. Results. We will see from our experiments that both methods do well at matching the boolean images. As might be expected, the slope penalty method does better at matching non-boolean shapes; however, recall that the purpose is *not* to match a smooth shape perfectly, but rather to match it well while satisfying the boolean condition. We expect the output to produce an image that is both a boolean and conforms, at least in coarse outline, to the shape of the target. As we will show sometimes it is easy to fulfill either part of this requirement but not both at the same time. In our results we can differentiate the performance of the different methods by the number of iterations required to yield a given reduction in objective function.

5.1. Boolean targets. With boolean targets we see that both methods match the shape very well. The variation between methods comes in the form of global minimizers, convergence rate, see Figure 5.1, and the transition, Λ, between $\mathcal{I}$ and $\mathcal{E}$.

The results show that using only the Tikhonov regulation, see Figure 5.2, pro-

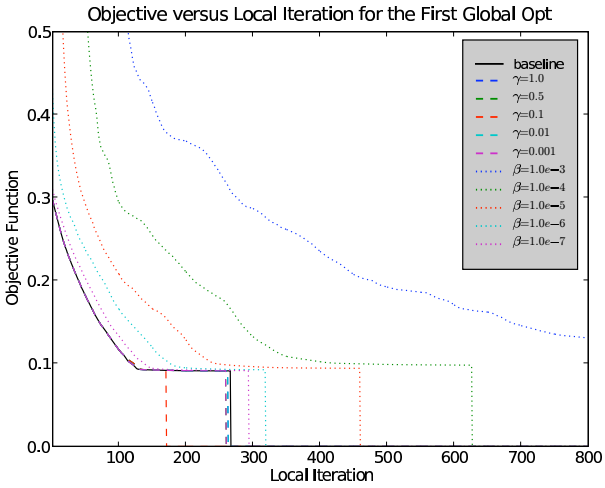
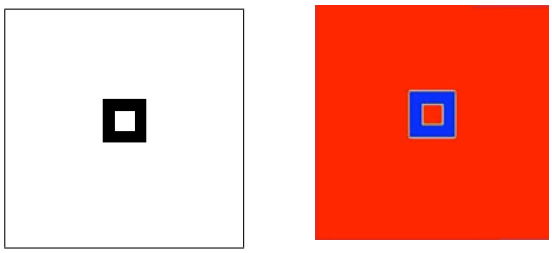


FIG. 5.1. *Sample convergence speed comparison*

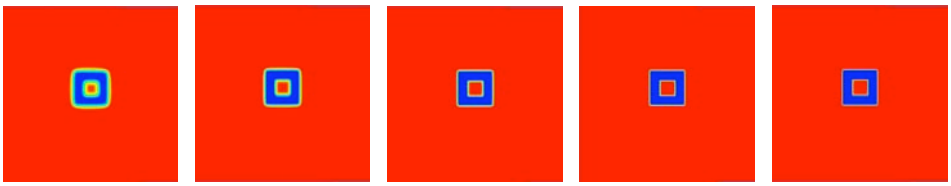


(a) target (b) baseline

FIG. 5.2. *Sample Target and baseline configuration*

duced a well matched shape and a small Λ. At the same time it did not produce a large amount of global minimizers.

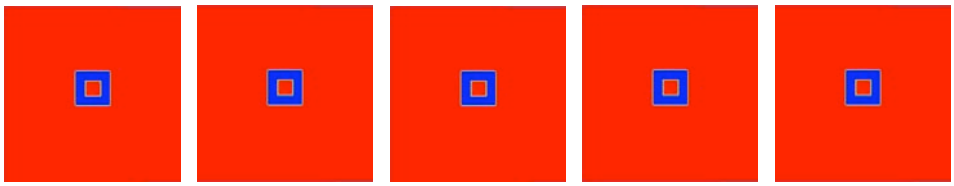
The slope penalty method, see Figure 5.3, tended to produce less shapely boolean contours. Superficially, it also appears to converge more slowly, although that is difficult to assess because the objective function is altered by the introduction of the penalty term. The slope barrier method, see Figure 5.4, successfully reproduced the sharp boolean images.



(a) $\beta = 1.0e-3$ (b) $\beta = 1.0e-4$ (c) $\beta = 1.0e-5$ (d) $\beta = 1.0e-6$ (e) $\beta = 1.0e-7$

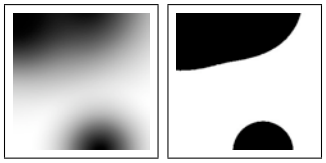
FIG. 5.3. *Sample slope penalty configurations*

5.2. Non-boolean targets. The most important test for our purposes is to be able to find good boolean approximations to non-boolean images. We use as a target



(a) $\gamma = 1.0$ (b) $\gamma = 0.5$ (c) $\gamma = 0.1$ (d) $\gamma = 0.01$ (e) $\gamma = 0.001$

FIG. 5.4. *Sample slope barrier configurations*



(a) target (b) solution

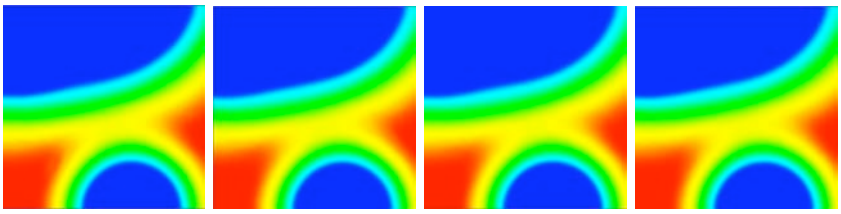
FIG. 5.5. *Smooth target and boolean solution*

image the smooth solution to our unconstrained microfluidics problem. The "true" boolean solution can be found by taking everything that is at least 50% black and giving it the value 1 and the rest 0, see Figure 5.5. This example almost reversed all the results of the boolean targets, where the narrow Λ was true to the shape of the target. We see in Figure 5.6 the results with a slope penalty method. As can be seen, it fits the target image quite well, but does *not* satisfy the boolean condition.

For non-boolean problems the slope barrier method converged more slowly than the slope penalty method and (as expected) for a small Δ it resulted in a large Heaviside distance from the boolean solution. The global optimization process also a large number of global minimizers giving the optimization even more problems to overcome. As can be seen in Figure 5.8, the solutions are, while sharply boolean, also quite ugly and plagued by local minimizers.

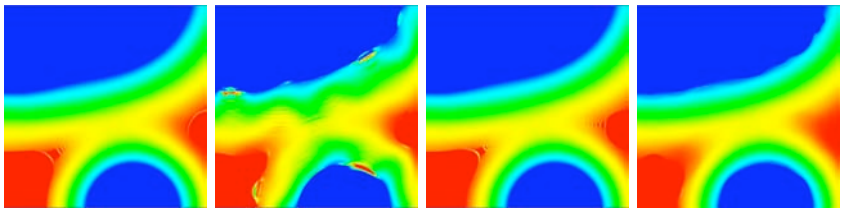
This performance showed that the small values of Δ, usually 12 pixels, that we used for the boolean targets were unsuitable for the non-boolean targets. By increasing Δ we were able to get performance closer to the slope penalty method, but this still did not result in a boolean image that we would be able to use in our application. Therefore we implemented a variable Δ in the optimization loop, in which we gradually reduce Δ after a number of outer iterations. To give a fair comparison we used a total number of global optimization steps equal to the non-variable Δ. This method did considerable better that the static version, and was able to achieve a more boolean shape that was closer to the shape of the solution, see Figure 5.9.

6. Conclusions. For boolean targets, the methods are closely comparable in performance and in effectiveness. There is a tradeoff between faithfully matching a non-boolean image and meeting a boolean constraint. The slope penalty method provides a better "fit", but does not easily meet the boolean condition. The slope barrier method does a better job of meeting the boolean condition; in our microfluidics applications that is the more important consideration. Finally, we observed that a gradual reduction in bandwidth parameter Δ is essential to getting clean results from the slope barrier method.

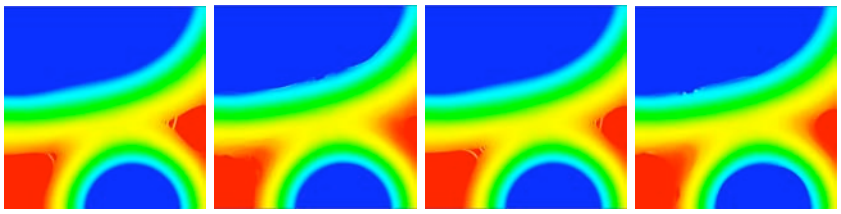


(a) 4 circles (b) 9 circles (c) 16 circles (d) 25 circles

FIG. 5.6. *Larger slope penalty ($\beta = 1.0e-4$) Final Global iteration*

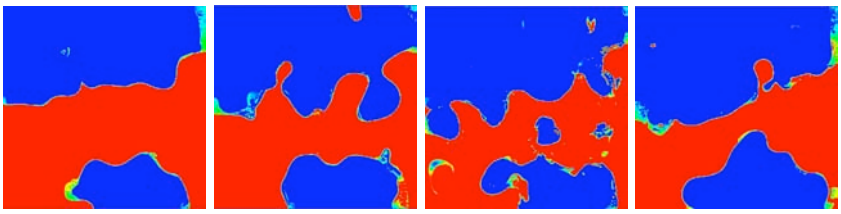


(a) 4 circles, first (b) 4 circles, final (c) 9 circles, first (d) 9 circles, final



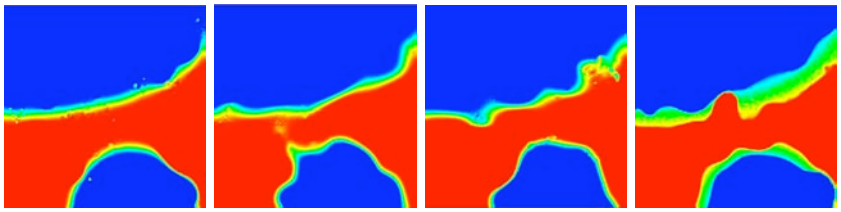
(e) 16 circles, first (f) 16 circles, final (g) 25 circles, first (h) 25 circles, final

FIG. 5.7. *Smaller slope penalty $\beta = 1.0e-6$, First and Final Global iteration*



(a) 4 circles (b) 9 circles (c) 16 circles (d) 25 circles

FIG. 5.8. *Slope Barrier $\gamma = 0.001$, Final Global iteration*



(a) 4 circles (b) 9 circles (c) 16 circles (d) 25 circles

FIG. 5.9. *Slope Barrier Reduction Method ($\gamma = 0.001$, reduces = 20, opts = 5, reduction = 0.9)*

REFERENCES

[1] V. Akcelik, G. Biros, O. Ghattas, K. Long, and B. vanBloemenWaanders, *A variational finite element method for source inversion for convective-diffusive transport*, Finite Elements in Analysis and Design, 39 (2003), pp. 683–705.
[2] M. P. Bendsøe and O. Sigmund, *Topology Optimization: Theory, Methods and Applications*, Springer-Verlag, 2003.
[3] R. Byrd and P. Boggs, *The alm-bfgs method*. A work in progress.
[4] A. Cunha, *A Fully Eulerian Method for Shape Optimization with Applications to Navier-Stokes Flows*, PhD thesis, Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA, 2004.
[5] M. Heroux, R. Bartlett, V. Howle, R. Heokstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, and A. Williams, *An overview of the trillinos project*, ACM Transactions on Mathematical Software, 31 (2005).
[6] K. Long, *Sundance 2.0 tutorial*, Tech. Report SAND2004-4793, Sandia National Labs, 2004.
[7] J. Nocedal and S. Wright, *Numerical Optimization*, Springer-Verlag, 2000.
[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++: The Art of Scientific Computing*, Cambridge University Press, 2 ed., 2002.