Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2015.w13si

Vol. **8**, No. 2, pp. 149-167
May 2015

Using Discrete and Continuous Models to Solve Nanoporous Flow Optimization Problems

Paul T. Boggs[1], David M. Gay[2], and Stephen G. Nash[3*]

[1] *Sandia National Laboratories (retired), M.S. 9159, P.O. Box 969, Livermore, CA 94551, U.S.A.*
[2] *AMPL Optimization, Inc., 900 Sierra Place SE, Albuquerque, NM 87108-3379 U.S.A.*
[3] *George Mason University, M.S. 5C8, Fairfax, VA 22030 U.S.A.*

Received 20 May 2014; Accepted 10 December 2014

Abstract. We consider using a discrete network model in combination with continuous nonlinear optimization models to solve the problem of optimizing channels in nanoporous materials. The problem and the hierarchical optimization algorithm are described in [2]. A key feature of the model is the fact that we use the edges of the finite element grid as the locations of the channels. The focus here is on the use of the discrete model within that algorithm. We develop several approximations to the relevant flow and a greedy algorithm for quickly generating a "good" tree connecting all of the nodes in the finite-element mesh to a designated root node. We also consider Metropolis-Hastings (MH) improvements to the greedy result. We consider both a regular triangulation and a Delaunay triangulation of the region, and present some numerical results.

AMS subject classifications: 49M37, 65K04, 90C08

Key words: Multigrid Optimization, Multilevel Optimization, Nonlinear Optimization, Complex Hierarchical Optimization.

1. Introduction

We consider using a discrete network model in combination with continuous nonlinear optimization models to solve the problem of optimizing channels in nanoporous materials. The problem and the hierarchical optimization algorithm are described in [2]. The focus here is on the use of the discrete model within that algorithm. We begin by summarizing the physics problem and our hierarchical approach, thus providing the necessary context for the ensuing discussion.

Nanoporous materials have the desirable property that within the nanopores a gas can be stored at much higher density than, say, in a tank with comparable pressure.

*Corresponding author. *Email addresses:* paulboggs@comcast.net (Paul T. Boggs), dmg@ampl.com (David M. Gay), snash@gmu.edu (Stephen G. Nash)

 ©2015 Global-Science Press

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

Many important applications, including the storage of electric charge, are noted in [2]. The problem is that it is difficult to charge, i.e., fill, these materials rapidly. It has been suggested that a series of channels be constructed in the material to act like a vascular system that facilitates the charge and discharge of the gas. Thus there are tradeoffs among the storage capacity, the volume of the channels, and the time needed to charge or discharge the materials. One version of the design problem is to choose the channel structure, including the topology and the widths of the channels, to minimize some property of the system, e.g., the average pressure, subject to the condition that the total volume of the channels divided by the total volume of the material (called the porosity) be bounded and that the rates of charge and discharge dictated by the particular application be achieved. Other versions are clearly possible.

As noted above, our approach to solving these problems is described in [2]. Briefly, we specify a PDE on the flow's domain and allow channels to be built along the edges of the finite element grid. Building the channels in this way restricts the channels somewhat, but is much more efficient than solving a general topology optimization problem. We comment more on this below. One node on the boundary of the domain is designated as the outlet node. The optimization problem is to determine the widths of the channels to solve the optimization problem described above. In a typical vascular system, wide channels split into successively narrower channels to cover the domain sufficiently to achieve the design criteria. It is thus natural think about formulating the problem on different scales, starting with a coarse scale to get the general shape of the channel system, followed by refinements of the scale to achieve better coverage. The problem on the finest scale captures all of the variables of the optimization problem and is where the ultimate problem is formulated. This problem, however, can be huge; exploiting this hierarchical structure, as shown in [2], can lead to significant computational savings.

The resulting optimization problems are challenging to solve. There are many design variables (the widths of the numerous channels within the material) and many more state variables (from the finite-element discretizations of the PDEs for the flows in the channel and bulk material). The dramatic difference in flow rates between the channels and bulk gives rise to significant ill conditioning. In addition, because the numerous small channels individually have only a small and subtle effect on the objective, the solution to the optimization problem is nearly degenerate. Finally, it is not at all clear how to obtain quality initial approximations that lead to reasonable solution times.

The hierarchical optimization algorithm developed in [2] is a multilevel method based on the multigrid optimization (MG/Opt) framework from [14]. The implementation is called χOpt and is described below in more detail. The MG/Opt framework is designed for solving continuous nonlinear optimization problems in cases where a hierarchy of optimization models is available. The hierarchy consists of a high-fidelity (fine) model that is typically expensive to evaluate, along with lower-fidelity (coarser) models that are used to update the estimate of the solution to the fine model. Typically the coarser models will be cheaper to evaluate, leading to computational savings. It

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

has been demonstrated in a variety of applications [2, 4, 6, 10, 12, 16] that MG/Opt has the potential to dramatically accelerate convergence to a solution.

Our hierarchical approach ameliorates some of the above difficulties, but not all. Although the multigrid methods often require many fewer evaluations of the functions on the fine grids, the problem of generating a good initial approximation is not addressed. Our response to this difficulty is to add to the hierarchy of models a simple model that is not based on PDEs.

Fig. 2 shows sample solutions to one of these problems. Domain experts in physics expect the channels to form a tree network that connects all of the nodes in the finite-element mesh to the outlet node. Here we use this intuition to construct a discrete network model of the flow in addition to the finite-element PDE models. This will allow us to create excellent initial approximations and to maintain a tree structure. Our goal in this paper is to demonstrate how to use a pure network model as part of a multilevel optimization algorithm such as MG/Opt. It is not straightforward to include such a pure network model in the MG/Opt framework.

In this paper we describe how we use the discrete network model to initialize the continuous model. We begin in Section 2 by giving a formal statement of the optimization problem. Then we state the general multigrid optimization algorithm and describe its convergence properties. For our algorithm to be effective, we need a good starting approximation. As shown in [2], uniform guesses may not produce good solutions and, even if they do, the convergence can be much slower than with a better starting guess. We also show that a network that has a tree structure can perform much better than one that does not. We have continued to observe this in other experiments and it is in accord with the intuition of domain experts. In Section 3 we present a result that provides strong justification for our use of trees—in fact, a tree structure is optimal under certain conditions that often hold in the problems of interest. Thus we want an initial condition that is a tree, rooted at the outlet node, that spans all of the nodes in the finite element grid. Obviously there are many such spanning trees, so we need an appropriate objective function to guide us. We examine some choices for the objective function in Section 3.

We have experimented with greedy heuristics to choose good initial trees and with a Metropolis-Hastings (MH) algorithm to improve the initial guesses. Both the initial greedy heuristics and the improvement algorithm use pseudorandom number generators for breaking ties in the heuristics and for deciding whether to accept a tentative move in the improvement algorithm. These details are also discussed in Section 3.

In almost all of the problems reported on here and in [2], this network initialization procedure was employed and worked well. The only cases where the network initialization was not used are the tiny problems in [2] created to demonstrate some of the properties of the solution. By starting with an initial guess that was not a tree structure, we obtained convergence to a solution that was also not a tree structure; when we added and enforced the tree constraint, we obtained a significantly better solution, i.e., lower objective function. The same is true of some larger problems that we ran but did not publish. As mentioned above, this accords with the expectations of experts in

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

the physics of the problem.

In Section 4 we discuss how we transition the discrete model to the continuous one and we consider the issue of the tree constraint, which creates a mixed-integer optimization problem. In Section 5 we present the results of computational experiments that demonstrate the effectiveness of our approach. (A more extensive set of computational experiments is contained in the technical report [3].) In Section 6, we discuss extensions to several interesting and practical situations. Our conclusions appear in Section 7.

2. The General Multigrid Optimization Algorithm

As mentioned above, the optimization problem can take several forms. Here we state an abstract form of the discretized problem based on the model described in [2]. Let N_n be the number of nodes and N_a be the number of arcs in the discrete domain at the finest level. Our objective function is the average pressure over the domain, which is a linear function of the pressure vector p, where p_i is the pressure at each node in the finite element grid. Let $c \in R^{N_n}$ be the resulting linear operator, so the objective function is $c^T p$. Let w_i be the channel width corresponding to each arc i. From [2] we see that the PDE describing the pressure can be represented abstractly as

$$A(w)p = g(w),$$

where, for each value of w, $A(w)$ is a linear operator and $g(w)$ is a vector. If ℓ_i is the length of arc i, then the constraint that the porosity be bounded is that $w^T \ell \leq z$ for some specified z. Since there is no reason not to use all available porosity, this becomes an equality constraint in the final formulation:

$$\begin{aligned} \min_{w,p} \quad & c^T p \\ \text{subject to:} \quad & A(w)p = g(w) \\ & w^T \ell = z \\ & w \geq 0. \end{aligned} \tag{2.1}$$

As the model is refined, the finite-element subdomains are divided and channels added along the new edges. Note that we have not included the constraint that the final network structure be a tree. We will discuss this issue after we describe how we construct our network model. We show in [2] that (2.1) can be solved by creating a simple bound-constrained problem; see also Section 2.1.

To solve the PDE-constrained optimization problem above we rely on MG/Opt. The idea behind MG/Opt is to use the coarse models to define a *search direction* for the fine-level solution. We include here an unconstrained version of MG/Opt adapted from the constrained algorithm in [13]. To make this notationally consistent with [13] we

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

use x as the vector of optimization variables. We will switch back to w in the sequel. MG/Opt assumes the availability of a convergent optimization algorithm Opt defined as a function of the form

$$x^+ \leftarrow \mathsf{Opt}(f(\cdot), v, \bar{x}, k)$$

which applies k iterations of a convergent optimization algorithm to a shifted or "surrogate" problem

$$\min_x \quad f_s(x) \equiv f(x) - v^T x \tag{2.2}$$

with initial guess $\bar{x}$ to obtain x^+. If the parameter k is omitted, the optimization algorithm continues to run until its termination criteria are satisfied.

Within MG/Opt the shift v is used to adjust a coarse model so that it matches a fine model to first order at the current estimate of the solution. Thus the surrogate model acts like a Taylor series approximation to the fine model. But unlike in a Taylor series where the approximation is based on a generic polynomial function, here the approximation is based on a coarse model that has the potential to be a far better approximation to the fine model.

The algorithm MG/Opt attempts to reduce the number of iterations of Opt on the finest grids by doing only a relatively few iterations before and after a so-called recursion step that exploits the coarser grids. Specifically, MG/Opt performs k_1 steps in the "pre-smoothing" step and k_2 steps in the "post-smoothing" step. Here is the complete algorithm for MG/Opt:

Given an initial estimate of the solution x_h^0, set $v_h = 0$. Let I_h^H and I_H^h be downdate and update operators, respectively, that map between successive models in the hierarchy. Then for $j = 0, 1, \ldots$, set

$$x_h^{j+1} \leftarrow \mathsf{MG/Opt}(f_h(\cdot), v_h, x_h^j)$$

where the function MG/Opt is defined as follows:

- *Coarse-level solve:* If on the coarsest level,

  $$x_h^{j+1} \leftarrow \mathsf{Opt}(f_h(\cdot), v_h, x_h^j).$$

  Otherwise,

- *Pre-smoothing:*

  $$\bar{x}_h \leftarrow \mathsf{Opt}(f_h(\cdot), v_h, x_h^j, k_1)$$

- *Recursion:*

  – Compute

  $$\bar{x}_H = I_h^H \bar{x}_h$$
  $$\bar{v} = I_h^H v_h + \nabla f_H(\bar{x}_H) - I_h^H \nabla f_h(\bar{x}_h)$$

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

– Apply MG/Opt recursively to the surrogate model:

$$x_H^+ \leftarrow \mathsf{MG/Opt}(f_H(\cdot), \bar{v}, \bar{x}_H)$$

– Compute the search directions $e_H = x_H^+ - \bar{x}_H$ and $e_h = I_H^h e_H$.

– Use a line search to determine $x_h^+ = \bar{x}_h + \alpha e_h$ satisfying $f_h(x_h^+) \leq f_h(\bar{x}_h)$.

- *Post-smoothing:*

$$x_h^{j+1} \leftarrow \mathsf{Opt}(f_h(\cdot), x_h^+, k_2)$$

As reported in [2], MG/Opt is significantly better than Opt at solving the nanoporous material problem. For a problem where the fine-level model has 24,704 edges and where two coarser models were used in the hierarchy, MG/Opt took 287 seconds to converge to a solution while Opt took 3546 seconds, about 12 times longer. In addition, MG/Opt found a better solution. The final objective value with MG/Opt was 7.2×10^{-7} while the final objective value with Opt was 2.2×10^{-6}.

We use the discrete network model to initialize MG/Opt, but not as part of the model hierarchy in the MG/Opt framework. Since the network model is discrete, it is not sensible to talk about a search direction. Also, it is not obvious how a solution to the network model can be used to update the solution to a finite-element model. For these reasons we use the network model as part of a full-multigrid initialization, i.e., to help obtain the initial solution to the PDE model on the coarsest finite-element mesh. Interestingly, we can also use a network approximation, but one based on the finest level, to ensure that a tree structure is maintained, i.e., to help to satisfy the constraint that we maintain a tree. These details are in Section 4. Viewed in this way our fine level has two coarse approximations, one based on the grid and one based on the network. Thus the overall hierarchy is now a tree structure, rather than a conventional linear hierarchy. This is discussed further in Section 5.

The discrete model is a crude approximation to the PDE-based model. Diffusion within the material is reduced to simple source terms at the nodes of the network. And flow within the channels is reduced to network flow-balance equations (flow in equals flow out). What is remarkable is that such a crude model can be of practical value as part of the model hierarchy.

2.1. Convergence and Descent

There are two phases to the overall algorithm: the initialization using the discrete network model, and the subsequent application of MG/Opt to the PDE model. The discrete network model is discussed in Section 3.1; here we discuss MG/Opt.

To address descent for MG/Opt in our case, we need to provide more detail on our optimization model. As explained in [2], we are solving a steady-state optimization problem that consists of finding the channel widths w that minimize the mean pressure within the medium subject to a constraint on the fraction of the overall volume, or

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

equivalently the porosity, that is allocated to the transport channels. The optimization problem has the form

$$\begin{aligned} \min_{w} \quad & \int_{\Omega} p(w) + \frac{1}{2}\theta \left[\sum_{i=1}^{N_e} \int_{e_i} w_i - z \right]^2 \\ \text{subject to} \quad & \text{[the weak form of the equations of flow]}, \\ & 0 \le w_i \le u_i, \end{aligned}$$

where the variables are the channel widths w. Here $p(w)$ is the pressure (obtained from the solution to the equations of flow), and z is a specified target value for the proportion of the domain devoted to channels. In our implementation the flow equations are used to solve for $p(w)$ given a set of channel widths w. Thus this is a bound-constrained nonlinear optimization model once the network has been specified.

This target porosity z is formulated as a penalty term with penalty parameter θ, a decision based on our choice of Opt (see below); with a different choice of Opt it could have been formulated as an explicit constraint. The domain consists of bulk material in the region Ω and channels (edges) e_i. A Dirichlet boundary condition is imposed at the outlet, and a no-flow (i.e., zero flux) Neumann condition is imposed elsewhere on the boundary.

As is proven in [13], if Opt is a convergent optimization algorithm, then MG/Opt will be convergent as well. In our computational results, Opt is chosen as the limited-memory quasi-Newton method L-BFGS [11], which has the necessary convergence properties. In addition, at every iteration of L-BFGS, there is a guarantee of descent, i.e., there is progress toward the solution of the optimization problem. Thus both Opt and MG/Opt are convergent in our implementation, and descent is guaranteed for Opt.

For MG/Opt, the issue of descent only arises in the line search based on the search direction for the multilevel recursion:

$$w_h^+ = \bar{w}_h + \alpha e_h.$$

Because L-BFGS maintains feasibility with respect to the bound constraints, the search direction e_h will be a descent direction if the following theorem [13] is satisfied.

Theorem 2.1. *Assume that*

- $\nabla f_h(w_h)$ *is defined for all values of* w_h.
- $\nabla^2 f_h(w_h)$ *is continuous for all choices of* w_h.
- $(I_H^h)^T = C_I I_h^H$ *for some constant* $C_I > 0$.

In addition, assume that $\nabla f_h(\bar{w}_h) \neq 0$. *Then the search direction* e_h *from the recursion step of MG/Opt will be a descent direction for* f_h *at* $\bar{w}_h$ *if*

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

(a) $f_s(w_H^+) < f_s(\bar{w}_H)$, *and*

(b) $e_H^T \nabla^2 f_H(\bar{w}_H + \eta e_H) e_H > 0$ *for* $0 \leq \eta \leq 1$.

To assess whether the conditions in the above theorem are satisfied by our model, it is necessary to examine the flow equations in more detail. In the two-dimensional case, the weak form of these equations (taking into account some simplifications and re-normalizations described in [2]) has the form

$$\int_\Omega [\bar{u}\beta + \nabla\bar{u} \cdot \nabla p]\, \xi \, dy + \sum_{i=1}^{N_e} \int_{e_i} w_i \left[\bar{u} + \Gamma w_i^2 \frac{\partial \bar{u}}{\partial l} \frac{\partial p}{\partial l}\right] dl - \int_{\partial\Omega} \bar{u}(\nabla p) \cdot \mathbf{n}\, dS = 0. \quad (2.3)$$

Here, $\bar{u}$ is a test function, β is the capacitance ratio (a constant), ξ is a volume element (a scalar), and Γ is a large constant, perhaps $\approx 10^{12}$. Hence this equation is nonlinear, with cubic terms in the channel widths w, and products of terms involving w and the pressures p.

Let us now examine the above theorem for our optimization model. Our algorithms are designed only to evaluate the objective function and the PDE model at points that satisfy the bound constraints. For these points, ∇f_h is defined, and $\nabla^2 f_h$ is continuous. In addition, we have chosen update and downdate operators that satisfy $(I_H^h)^T = C_I I_h^H$ for some constant $C_I > 0$. If Opt performs at least one iteration on the coarse model, then $f_s(w_H^+) < f_s(\bar{w}_H)$. This leaves only the condition

$$e_H^T \nabla^2 f_H(\bar{w}_H + \eta e_H) e_H > 0$$

for $0 \leq \eta \leq 1$. Thus we have to assess the convexity of the coarse-model version of

$$f(w) \equiv \int_\Omega p(w) + \frac{1}{2}\theta \left[\sum_{i=1}^{N_e} \int_{e_i} w_i - z\right]^2.$$

Because of the nature of the nonlinearities of the flow equations, the Hessian of the first term is not guaranteed to be positive definite, and hence the final condition in the theorem may not be satisfied for all choices of w.

However, because this is an optimization model, the Hessian will be positive semi-definite at the solution. In addition, the Hessian of the second term has the form θE for some positive-definite diagonal matrix E whose diagonal entries are proportional to the lengths of the channels. Hence, if the penalty parameter θ is sufficiently large, it is possible to guarantee that the theorem will be satisfied and thus to guarantee that MG/Opt will produce descent directions in the line-search step.

More can be said for special cases of the network. For example, if a two-dimensional rectilinear network is used, and if restrictions on the channel widths are imposed, then the model is convex [15].

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

3. Objective Functions and Algorithms

In defining the discrete model we must relate the continuous (PDE-based) optimization model to the objective function and constraints in the discrete model. In this section we focus on the objective function. We examine several choices and describe our algorithm for obtaining an approximate solution to the discrete model. In Section 5 we demonstrate the behavior of the algorithm and its components.

The constraints in the discrete model correspond to the network structure and are handled implicitly within our algorithm by maintaining a particular network structure (a spanning tree). Flow satisfies network-flow constraints: at each node the flow in must match the flow out. In the PDE model, diffusion of the gas within the bulk material results in flow of gas into the channels. In the network model, source terms in the network are included at nodes internal to the bulk material and represent a simplistic approximation of the diffusion of the gas in the bulk material. There is a sink at the outlet.

Since our goal is to develop a low-cost model, we do not attempt to capture all the physics of the overall problem, but we use our knowledge of the problem to formulate simple yet effective objectives. As stated in Section 1, we are often interested in either minimizing the porosity or bounding it as a constraint. We formulate the porosity as follows.

Let u be the velocity of the material in a given arc, let w be the channel width, let μ be the viscosity, and let p be the pressure. The physics of viscous flow in two space dimensions imply that

$$u = -\frac{w^2}{12}\,\frac{1}{\mu}\,\frac{\partial p}{\partial x}, \tag{3.1}$$

in which x represents distance along an arc. The volumetric flow rate is then given by

$$\dot{v} = |u|A = |u|w, \tag{3.2}$$

where A is the cross-sectional area (just w in 2-d) and we have assumed that all pressure gradients are approximately the same. From this, we obtain that

$$\dot{v} \propto w^3. \tag{3.3}$$

Our goal is to minimize the porosity while being sure that all nodes are connected to the outlet node. This means that we want to minimize the volume of the channels. We can easily compute the channel volume (in 2-d) as

$$V_{\text{channel}} = \sum_{i=1}^{N_a} \ell_i w_i, \tag{3.4}$$

where N_a is the number of arcs in the graph, and ℓ_i is the length of the i-th arc. Combining (3.3) and (3.4), we obtain

$$V_{\text{channel}} = \sum_{i=1}^{N_a} \ell_i (\dot{v}_i)^{\frac{1}{3}}. \tag{3.5}$$

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

An interesting result by Banavar et al. [1] shows that if (3.5) is used as the objective function, then the optimal structure will always be a tree. In particular, they consider a general transportation network with one outlet and an objective function of the form

$$\sum_{i=1}^{N_a} \alpha_i |w_i|^\gamma,$$

where α_i is a given constant associated with edge i, $\gamma > 0$ is a constant, and w_i is to be determined. They show that if $\gamma < 1$ then the optimal network is a tree. This of course means that there can be no loops in the optimal structure and many edges may not be present in the optimal network, i.e., if edge i is not in the final tree, w_i will be zero. On the other hand, if $\gamma > 1$ then it is most efficient to use all of the edges available and thus loops emerge. In our case, $\gamma = 1/3$ (or $1/4$ in three dimensions), so a network minimizing the porosity will be a tree. Since our goal is either to minimize porosity or bound it, we will consider only trees in our formulations.

The volumetric flow rate along a given arc a_j can be thought of as the total flow into a_j that comes from the part of the tree "above" it. Let f_j be the total flow along arc a_j. The contribution to V_{channel} of a_j is $\ell_j f_j^{1/3}$. Let T be the set of the indices of the arcs in the tree. Then

$$V_{\text{channel}} = \sum_{j \in T} \ell_j f_j^{\frac{1}{3}} \tag{3.6}$$

and our goal is to find a T that minimizes V_{channel}.

We must account for flow from the bulk into the spanning tree and can use suitable choices of f_j for this purpose. There are two obvious choices. We can assume that the flow from the bulk into an arc is proportional to the arclength and use the total arclength "above" node j in the tree, or we can assume a source of one on each node and use the total number of nodes "above" node j in the tree. To be specific, we define f_j^a as the choice using the arclength and f_j^n as the choice using the number of nodes and we obtain the two objective functions

$$V_{\text{channel}} = \sum_{j \in T} \ell_j (f_j^a)^{\frac{1}{3}} \tag{3.7}$$

and

$$V_{\text{channel}} = \sum_{j \in T} \ell_j (f_j^n)^{\frac{1}{3}}. \tag{3.8}$$

In addition to these choices, we also consider simple objective functions that involve the flow variables f_j. If f_i is the current flow in the incomplete spanning tree containing i arcs, and arc k is added to the tree, then for the f_j^a variables, the resulting flow is

$$f_{i+1}^a = f_i^a + \text{length}(a_k), \tag{3.9}$$

and for the f_j^n variables, the resulting flow is

$$f_{i+1} = f_i + 1. \tag{3.10}$$

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

The obvious trouble with (3.10) is that all arcs look the same.

Since we will eventually bound or minimize the porosity, we constrain our greedy algorithm to construct trees. Our heuristics yield good, but typically not optimal, spanning trees.

These network models are admittedly crude approximations to the underlying physics, and they do not have the fidelity or realism of the PDE models described in [2]. The PDE models are based on more sophisticated physics, and in addition it is the fine-level PDE model that ultimately is used to identify the optimal solution. We selected the network models to be (a) crude enough to be inexpensive to use, but (b) sophisticated enough to give us a useful initial guess for the PDE model. The network models and the coarser PDE models are used to accelerate convergence to the solution of the high-fidelity model.

3.1. A Greedy Algorithm

Our greedy heuristics choose a spanning tree that connects all nodes to a specified root node. To do so, they maintain a *frontier* of nodes connected with the root. As long as unconnected nodes remain, a "best" one that can be connected to the frontier is chosen, and a most attractive arc from the chosen node to a node in the frontier is added to the spanning tree. We used the objective functions above for our definitions of "best" and "attractive" and found randomization useful in breaking ties.

It became clear early in testing (see Section 5) that this simple version of the greedy algorithm would often generate very long channels. To avoid this, we implemented Dijkstra's shortest path algorithm to compute the minimum distance, $\text{Dist}_i^{\min}$, from node i to the root. We then picked a factor, $D_{\max} > 1$, and attempted to ensure that the path from node i to the root was no longer than $D_{\max} \times \text{Dist}_i^{\min}$. It is clearly possible that this may not be attainable in the context of the greedy algorithm, so we put in a procedure to calculate the smallest value of $D_{\max}$ that we could use to complete the greedy procedure. It is not surprising that since the value of $D_{\max}$ affects the progress of the greedy algorithm, the final values will depend on the initial value. For example, in one test the initial value was 1.6 and the final value was computed to be 1.68 whereas for an initial value of 1.2, the final value remained at 1.2.

The computational complexity of Dijkstra's algorithm is, in general, $O(N_a+N_n log N_n)$, where N_n is the number of vertices [8]. The greedy algorithm can be seen to be $O(N_a)$ and thus the entire procedure is quite fast. We emphasize that this procedure is only performed on the coarsest mesh, making it even more efficient.

3.2. Metropolis-Hastings Improvements

As mentioned above, we experimented with several variants of a Metropolis-Hastings (MH) algorithm for improving the results of the greedy algorithm just described. The basic idea is simple: randomly select a vertex in the network and then randomly select an edge from that vertex. Replace the current edge from that vertex with the randomly

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

selected edge and evaluate the objective function. If the objective is improved, accept the change. Otherwise accept the change with some probability. Repeat while progress seems worthwhile.

To make this effective in our context, we added some modifications. First, any change must result in a legal network, so we check that the change did not cause a loop or otherwise disconnect the vertex from the root. This saves time in eliminating the need to calculate the objective function. Second, a large fraction of the proposed changes do not improve the objective function, so one has to choose the probability of acceptance conservatively to allow some exploration, but not so much that the objective never recovers. Finally, to improve the speed, we implemented a parallel version in which we randomly select a vertex and, in parallel, evaluate all edges from that vertex.

It is clear that each iteration of the MH algorithm requires a bounded number of operations, so the cost is dominated by the number of MH iterations actually performed. In our case, we simply limit the number of such iterations, but even though this is only performed on the coarsest grid, it depends on N_a and can be quite expensive. We comment more on this in Section 5 and Section 7.

3.3. Algorithm Summary

We conclude this section by summarizing our initialization scheme in algorithmic form.

1. Generate a coarse finite element grid on the domain representing the bulk material.
2. Pick a node on the boundary as the designated outlet node.
3. Pick an objective function from (3.7 – 3.10) and a random number seed.
4. Run the greedy algorithm with this objective to obtain a tree connecting all of the nodes in the finite element grid to the designated outlet node.
5. Pick a number $q \geq 0$ and a random number seed.
6. Perform q iterations of the MH algorithm starting from the result of the greedy algorithm.
7. Store the total flow through each arc in the final tree.

In the next section we discuss how to use this tree and the flow through each arc of the tree to create an initial approximation for the optimization problem.

4. Use of the Network Model

Having developed efficient techniques to generate a tree structure that connects all of the nodes in the finite element grid to the outlet node, we consider here how

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

to use this discrete model as an adjunct to our continuous hierarchical models. The original motivation for creating the network model was to be able to generate a good initial approximation for the optimization problem; we start with a description of our procedure of translating the network approximation to the continuous model.

From the procedures in the previous section we obtain the discrete flow through each arc. Let $\phi_i,\ i = 1, \cdots, N_a$, be the flow through arc i. We have assumed a source with a value of 1 at each node, so it is easy to keep track of the total flow through each arc. To accommodate this flow, in two dimentions we assume that the width of each arc is proportional to the flow. Thus we initialize each width w_i to ϕ_i. Note that any arc not part of the tree T is set to zero. Then we scale w so the porosity constraint is satisfied, i.e., the area of the channels is

$$\gamma = \sum_{i=1}^{N_a} w_i \ell_i.$$

Let Δ be the total area of the domain. Then set the initial condition $w_0 = w * \beta$, where

$$\beta = \frac{z\Delta}{\gamma}.$$

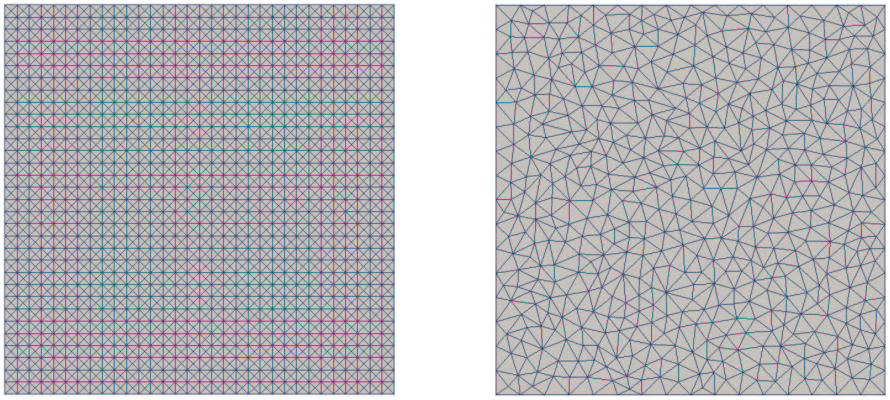
This initialization is for the coarse grid. We use a full multigrid method to extend this to the fine grid. Specifically, we do an automatic grid refinement that systematically partitions each triangle of the finite element grid into four subtriangles. This allows us to maintain the tree structure on the coarse grid and we can use the update operator to initialize the variables on this level. After doing a multigrid V-cycle on these two levels, we add the next level in the same manner.

Recall that the tree constraint is not part of the optimization problem (2.1). We now add this constraint. Let $\mathcal{T}$ be the set of all possible trees connecting all nodes of the finite element grid to the outlet node and let $T(w) = \{i : w_i \neq 0\}$. The tree constraint is simply $T \in \mathcal{T}$.

Solving a nonlinear optimization problem having many integer variables would be difficult. Instead, we use the following heuristic: after completing the first complete MG/Opt cycle, we again use the network approximation to enforce the tree structure. In this case, the network is based on the finest grid. This network is checked to see if more than one outlet arc is associated with any node; if so, only the arc with the largest pressure drop is kept and the others are constrained to be zero. This is done only once; the tree structure is then frozen. This means that after the first multigrid cycle our optimization problem is continuous.

An alternative to this approach would be to model the tree structure using explicit constraints in the optimization model. This would result in a mixed-integer nonlinear programming (MINLP) problem. Such problems are challenging to solve. For a convex model, a variety of approaches are available, such as branch and bound [9], or the outer approximation method [7], although they are computationally intensive. However, as we discuss in Section 2, our model will in general be non-convex. Solving MINLPs in

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press



(a) Rectangular mesh (b) Delaunay mesh

Figure 1: Rectangular and Delaunay meshes

the non-convex case is significantly more challenging, and heuristics must be used in an attempt to find good solutions [5]. Our approach is a heuristic designed around our specific application, and it avoids the large computational overhead associated with solving a MINLP problem.

5. Numerical Results

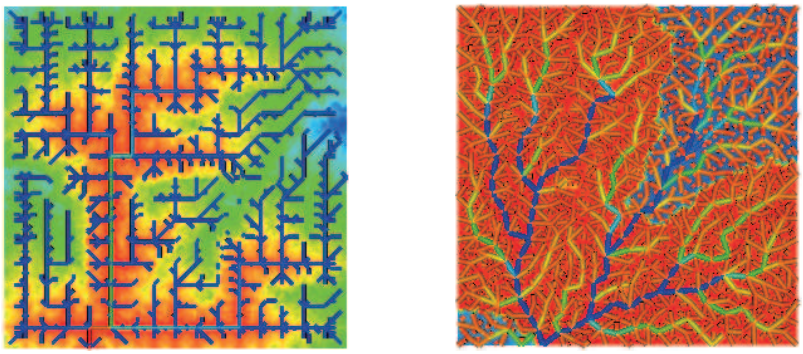
In this section we discuss numerical tests we performed using the objectives and algorithms of Section 3 in conjunction with χOpt for solving the complete optimization problem. We conducted a broad range of numerical experiments with each of the objective functions, both with and without the MH improvement. Since the idea of bounding the pathlength came later, we also tested the effects of bounding. For each category, we performed three tests, each with a different starting seed for the random number generator used in the algorithm.

As mentioned above, our procedures in Section 3 produce remarkably good initial approximations for χOpt. This holds true for all of the options considered, with some slight variations. The complete results are in [3]. Rather than show all of these results, which would need to be digested in detail, we summarize them and point out certain of the more interesting observations.

The most significant differences in the results depend on the underlying finite element grid. The first set of tests was done using a regular triangulation based on square grid with all of the diagonals added (Fig. 1(a)). Thus all of the arcs are either 0, 45, 90 or 135 degrees. The second set of tests used a more general Delaunay triangulation (Fig. 1(b)), which yields a richer variation in arc angles. For the Delaunay triangulation, the results for problems of similar sizes are substantially better in terms of the final average pressure, often by more than two orders of magnitude. Specifically, for the square grids, the final average pressure varied between 3.5×10^{-7} and 5.7×10^{-7}; for the

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

Delaunay grids, the final average pressure varied between 6.6×10^{-11} and 7.6×10^{-7}, a considerably wider variation than for the square grids. It should be noted, however, that there were only 3 of the 21 tests in the range of 10^{-7}, 2 in the range of 10^{-8}, but 15 in the range of 10^{-9} and 10^{-10}. Only 1 was in the range of 10^{-11}. Also, the number of iterations to convergence for the Delaunay triangulations was very small, typically only about 6 evaluations of the function and gradient on the finest grid. Fig. 2 shows the pressure distributions for both the rectangular and Delaunay grids. One can also see the final network of channels.



(a) Final pressure distribution for rectangular grid

(b) Final pressure distribution for Delaunay grid

Figure 2: Pressure distributions with rectangular and Delaunay grids

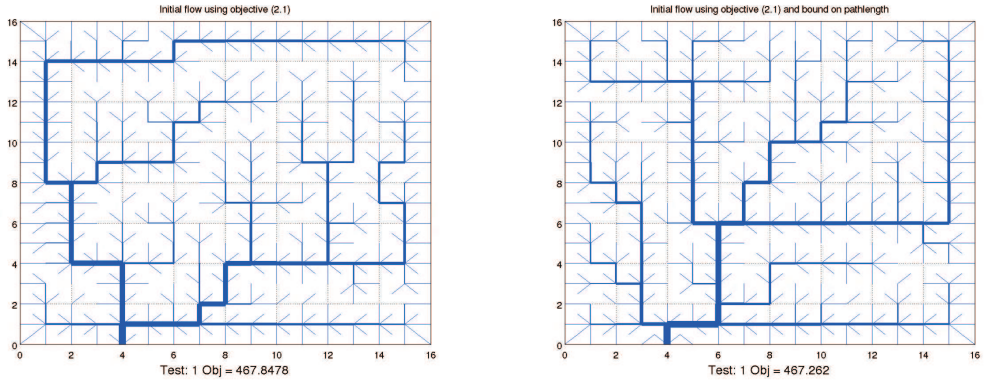
Some specific observations are as follows:

- In early testing using (3.9) and (3.10) as objective functions, we found that bounding the pathlength made the networks "nicer" visually and often, but not always, produced better final solutions. In fact, the highest average pressure in the square grid case came from a long path solution, but so did the lowest average pressure. In a number of other tests, the bounded pathlength results were sufficiently better that we used bounding in all subsequent tests. Fig. 3(a) shows a run without the pathlength bound; Fig. 3(b) is with the bound. Note the very long path to the outlet from the nodes in the upper right hand corner in Fig. 3(a).

- χ*Opt* was run using three levels, i.e., three MG/Opt levels with the network approximation used to define the starting point for the full multigrid initialization. From the point of view of the hierarchy, we can consider the network approximation to be the fourth, and coarsest, level of the hierarchy for the purposes of the full-multigrid initialization.

- As noted above, on the first set of tests on the regular triangulation, all of these approaches yielded final objective values from χ*Opt* that were between 3.5×10^{-7} and 5.7×10^{-7}. By simply changing the random number seed for the greedy algorithm or the MH improvement, however, the results can vary by about this much.

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press



(a) Network with no path bound (b) Network with path bound

Figure 3: Networks with and without path bound

This suggests, but does not prove, that many initial networks can be optimized to yield near-optimal average pressure for this grid.

- MH can change the networks quite a bit, but, again, it does not always yield better final solutions. For example, in using (3.7) as the objective MH reduced the value obtained by the greedy algorithm by approximately 10%, but the final values of the average pressure for each run showed no clear pattern.

- The final networks from χOpt on the coarsest grid maintain the look of the initial networks. This concurs with our expectation that the initial guess determines the basic coarse channel network, which is then refined to produce the final optimized value, i.e., the particular local solution found.

Although the focus of this paper is on the use of a network approximation in the context of an MG/Opt algorithm, a few comments on the overall performance of χOpt are in order. The problems reported here and in [3] are of moderate size and all of them converged in three or fewer multigrid cycles. Thus, for these problems, the network approximation indeed produced a remarkably good initial approximation for the continuous model and the multigrid recursion steps were effective. In larger problems reported in [2] and in unpublished other experiments, the network approximation also produced good starting approximations, but the number of MG/Opt cycles to achieve final convergence was often substantially larger. Since these problems are nonlinear, it is not surprising that the recursion step did not always reduce the objective or reduced it only by a small amount. It did, however, often make a substantial difference. In [2] we ran a test without the tree constraints to make a consistent comparison between Opt and χOpt and, as noted above, found that χOpt was about 12 times faster. This can also be observed by the fact that we typically obtained our best results using a small number of iterations on the finest grid, thus allowing the recursion steps to be more fully

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

exploited. Finally, we note that the triangulation of the region will, of course, have an effect on the final solution. Since we assume uniform properties of the bulk material, we were careful to ensure that the triangulation was also uniform in the sense that all of the triangles were of roughly the same size with comparable edge lengths. Clearly, having a significantly nonuniform mesh would lead to certain regions of the bulk not being effectively used in the final solution.

6. Extensions

Here we briefly describe some simple extensions. Sometimes it is appropriate to consider multiple inlet/outlet nodes for charging and discharging the material. Suppose, for example, that we want all of the nodes on a given edge to be inlet/outlet nodes. In this case, we create a fictitious node outside of this edge and add fictitious arcs connecting each node on the edge to the new fictitious node. We have successfully performed tests using this approach.

Sometimes one must restrict the flow on some of the edges. In fact, in all of the problems reported in this paper, the flow on any boundary edge was constrained to be zero. In other experiments, we wanted to test networks with the flow restricted to either vertical or horizontal arcs to be able to compare with analytic solutions. To do so, we simply constrained the flow on the other edges to be zero, thus removing them from consideration from both the greedy algorithm and from χOpt. Again, simple experiments have shown this to be effective.

Related to these extensions is the problem of allowing an entire edge to be an outlet. Here, we repeat the above procedure of allowing all nodes on that edge to be outlets and then restrict all flow along the boundary arcs to be zero.

7. Conclusion

We have shown that a network approximation to the flow in nanoporous materials can be a valuable complement to continuous models based on PDEs. Our view that the network approximation is just another member of the hierarchy leads to a consistent and more general way of thinking about such approximations. Since our problem has the non-smooth constraint that the solution must be a tree, we use this model at the coarsest level to set an initial condition and then use the model at the fine level to enforce the tree structure. Viewed in this way, the hierarchy is represented as a tree rather than the typical linear structure.

Our method begins with a greedy algorithm, based on an appropriate objective function, to generate an initial network along with edge widths that can be appropriately scaled to yield a quality initial approximation. We found it useful to restrict the pathlength from any node to the root node to be a modest multiple of the shortest path to the root. We also developed a Metropolis-Hastings procedure to improve the objective function from the greedy algorithm. This improvement did not generally yield

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

lower average pressures from the χOpt optimization runs, but the MH procedure may be useful in other contexts.

Our approach provides a powerful and flexible way to generate quality starting approximations to the optimal network. Our results confirm that there are many local solutions that give similar values for the final average pressure. We found Delaunay triangulation to be much better than a regular triangulation. Other considerations, e.g., manufacturability, may dictate the best choice of the mesh and the initial network.

Acknowledgments The authors would like to thank our colleague, Robert Nilson, for his help with the physics of the network approximations in Section 3. The authors would also like to acknowledge support by the Department of Energy Office of Advanced Scientific Computing Research under contract 10-014804 and award DESC-0001691.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[1] J. R. BANAVAR, F. COLAIROI, A. FLAMMINI, A. MARITAN, AND A. RINALDO, *Topology of the Fittest Transportation Network*, Physics Review Letters, 84 (2000), pp. 4745–4748.

[2] P. T. BOGGS, D. M. GAY, S. GRIFFITHS, R. M. LEWIS, K. R. LONG, S. NASH, AND R. H. NILSON, *Optimization Algorithms for Hierarchical Problems, with Application to Nanoporous Materials*, SIAM Journal on Optimization, 22 (2012), pp. 1285–1308.

[3] P. T. BOGGS, D. M. GAY, AND R. H. NILSON, *Network heuristics for initial guesses to nanoporous flow optimization problems*, Tech. Report 2012–3681, Sandia National Laboratories, Albuquerque NM and Livermore CA, 2012.

[4] A. BORZÌ AND U. HOHENESTER, *Multigrid optimization schemes for solving Bose-Einstein condensate control problems*, SIAM Journal on Scientific Computing, 30 (2008), pp. 441–462.

[5] S. BURER AND A. N. LETCHFORD, *Non-convex mixed-integer nonlinear programming: A survey*, Surveys in Operations Research and Management Science, 17 (2012), pp. 97–106.

[6] Z. DI, M. EMELIANENKO, AND S. G. NASH, *Truncated newton-based multigrid algorithm for centroidal Voronoi diagram calculation*, Numerical Mathematics: Theory, Methods & Applications, 5 (2012), pp. 242–259.

[7] M. DURAN AND I. GROSSMANN, *An outer-approximation algorithm for a class of mixed-integer nonlinear programs*, Mathematical Programming, 36 (1986), pp. 307–339.

[8] M. L. FREDMAN AND R. E. TARJAN, *Fibonacci heaps and their uses in improved network optimization algorithms*, J. ACM, 34 (1987), pp. 596–615.

[9] O. GUPTA AND V. RAVINDRAN, *Branch and bound experiments in convex nonlinear integer programming*, Management Science, 31 (1985), pp. 1533–1546.

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press

[10] R. M. LEWIS AND S. G. NASH, *Factors affecting the performance of an optimization-based multigrid method*, in Multiscale Optimization Methods and Applications, W. W. Hager, S.-J. Huang, P. M. Pardalos, and O. A. Prokopyev, eds., Springer, Berlin, 2005, pp. 151–172.
[11] D. C. LIU AND J. NOCEDAL, *On the limited memory method for large scale optimization*, Mathematical Programming B, 45 (1989), pp. 503–528.
[12] S. G. NASH, *A multigrid approach to discretized optimization problems*, Optimization Methods and Software, 14 (2000), pp. 99–116.
[13] ——, *Properties of a class of multilevel optimization algorithms for equality-constrained problems*, Optimization Methods and Software, 29 (2014), pp. 137–159.
[14] S. G. NASH AND R. M. LEWIS, *Assessing the performance of an optimization-based multigrid method*, Optimization Methods and Software, 26 (2011), pp. 693–717.
[15] R. H. NILSON AND S. K. GRIFFITHS, *Optimizing transport in materials having two scales of porosity*, Phys. Rev. E, 79 (2009), p. 036304.
[16] M. P. RUMPFKEIL AND D. J. MAVRIPLIS, *Optimization-based multigrid applied to aerodynamic shape design*, tech. report, Department of Mechanical Engineering, University of Wyoming, Laramie, 2009.

https://doi.org/10.4208/nmtma.2015.w13si Published online by Cambridge University Press