Parallel Analytico–Computational Methods for Multicomponent Media:
Application to Thermal Composites and Porous-Media Flows

by
Chahid Kamel Ghaddar

B.S., Kuwait University (1989)
S.M., Massachusetts Institute of Technology (1991)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1995

© Chahid Kamel Ghaddar. All rights reserved.

The author hereby grants to MIT permission to reproduce and distribute publicly
paper and electronic copies of this thesis document in whole or in part, and to grant
others the right to do so.

Author

Department of Mechanical Engineering
September 14, 1994

Certified by

Anthony T. Patera
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by

Ain A. Soain
Chairman, Departmental Committee on Graduate Students

ARCHIVES

APR 16 1995
Parallel Analytico-Computational Methods for Multicomponent Media: Application to Thermal Composites and Porous-Media Flows

by

Chahid Kamel Ghaddar

Submitted to the Department of Mechanical Engineering on September 14, 1994, in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Abstract

We develop a general first-principle methodology for the determination of macroscopic transport properties of random heterogeneous media. In particular, we address two problems of engineering interest: heat conduction in fibrous composites; viscous and inertial flows through fibrous porous media. More specifically, we exploit a variational hierarchical decomposition procedure to recast the original multiscale problem as a sequence of three scale-decoupled micro-, meso-, and macroscale subproblems. At the microscale level we develop analytical models to treat the nearfield behavior of clustered inclusions. These analytical models are then incorporated into the mesoscale problem using a new variational-bound nip-element technique. At the mesoscale level, we introduce a periodic supercell containing many inclusions (and associated microscale-modeled clusters) randomly dispersed in a continuous matrix. The associated partial differential equations, derived from homogenization theory, are solved by parallel iterative finite element methods to determine the (tensorial) configuration effective transport properties. Monte-Carlo techniques are then used to predict the average effective properties. Finally, at the macroscale level, we solve the now-homogenized problem to predict engineering quantities.

Our approach departs from earlier analytical and computational approaches in several aspects. First we extend the usual macro-, mesoscale decomposition to a macro-, meso-, microscale analysis. The latter, which provides both accurate approximations for the effective properties and rigorous error estimates, is necessary to first, mitigate the numerical difficulties associated with the geometric stiffness of the mesoscale problem, and second, to improve the statistical analysis by incorporating otherwise numerically intractable realizations. Second, we adopt an entirely parallel numerical treatment of the computationally-intensive mesoscale problem which, in turn, permits consideration of more complex physics and better statistics. Third, our numerical approach facilitates the investigation of nonlinear problems (e.g., inertial flows), which are not amenable to, say, analytical techniques.

We apply our methodology, in conjunction with a systematic surrogate framework developed specifically for computational problems with noisy outputs, to construct and validate practical engineering models for the effective conductivity and creeping-flow permeability. Limited but illustrative inertial porous-media-flow results at moderate Reynolds numbers are also presented, and compared against available data for both regular and random arrays.

Thesis Supervisor: Anthony T. Patera
Title: Professor of Mechanical Engineering
Thesis Committee Members:
  Prof. Chiang C. Mei
  Prof. Borivoje Mikic
  Prof. Ronald F. Probstein
Acknowledgments

This work was supported by the Advanced Research Projects Agency under Grant N00014-91-J-1889, by the Office of Naval Research under Grants N00014–90–J–4124 and N00014–89–J–1610, and by the Air Force Office of Scientific Research under Grant F49620–94–1–0121.

The author is indebted to Professor Anthony T. Patera for his support and responsible supervision throughout this work. The author would like also to acknowledge discussions and collaborative work with Dr. Manuel Cruz on parts of the theoretical microscale formulation presented in Chapter 3; the help of Dr. Einer Rönquist of Nektonics Inc.; to thank research fellows Mariano Gurfinkel and Serhat Yeşilyurt for comments and their assistance in solving some of the Latex difficulties encountered during the writing of this manuscript; and to express his gratefulness for all those who have helped him in one way or another during this research effort.
Dedication

This work is dedicated to my father, whose deep faith, advice, and continuous encouragement, made this work possible.

My deep gratitude to my wife whose support, and understanding, have been critical.
# Contents

1 Introduction ................................................... 9
   1.1 Problems of Interest ...................................... 9
   1.2 Previous Work ............................................. 9
   1.3 Objectives ................................................. 11
   1.4 Outline of Thesis .......................................... 11

2 Hierarchical Variational Decomposition ................. 13
   2.1 Decomposition Procedure of Multiscale Problem ........ 13
   2.2 The Thermal Composite Problem ......................... 18
   2.3 The Viscous Porous Media Problem ................. 19
   2.4 The Inertial Porous Media Problem ........... 21

3 Microscale Analysis and Bounds .................. 25
   3.1 Introduction ................................................ 25
   3.2 The Thermal Composite Problem ......................... 26
      3.2.1 Nip Geometry ........................................ 27
      3.2.2 Crude Upper Bound .................................. 28
      3.2.3 Crude Lower Bound .................................. 30
      3.2.4 Sharp Upper Bound .................................. 31
      3.2.5 Sharp Lower Bound .................................. 34
      3.2.6 Extension to Multiple Nips ......................... 37
   3.3 The Porous Media Problem ............................. 38
      3.3.1 Viscous Flows ....................................... 38
      3.3.2 Inertial Flows ...................................... 40
   3.4 A Posteriori Analysis Using Sensitivity Derivatives ... 41
      3.4.1 The Sensitivity Derivative ......................... 41
      3.4.2 Application to the Effective Conductivity Problem 43

4 Parallel Numerical Methods (Mesoscale) ............ 46
   4.1 Domain Decomposition Procedure .................. 46
      4.1.1 Grid Adjustment .................................... 46
      4.1.2 Geometry Description ............................... 49
      4.1.3 Nip Generation ..................................... 52
      4.1.4 Tensor-Product Versus Voronoi-Based Partition ..... 53
   4.2 Mesh Generation ........................................ 55
   4.3 Finite Element Treatment ............................ 59
      4.3.1 The Conduction Problem ......................... 61
B  Example of Voronoi/Delaunay Mesh Generation  150
   B.1 Introduction ............................................. 150
   B.2 Definition of Voronoi/Delaunay Diagram ............... 150
   B.3 Mesh Construction Procedure ............................ 151
      B.3.1 Domain Definition ................................. 151
      B.3.2 Layout of Interior Nodes ........................... 152
      B.3.3 Preservation of Boundary Integrity ................ 152
      B.3.4 Construction of the Voronoi/Delaunay Diagram .... 154
   B.4 Examples ................................................. 154

C  Triangular Finite Elements  156
   C.1 Linear Triangular Elements ............................ 156
   C.2 Elemental Laplacian ..................................... 158
   C.3 Quadratic Triangular Elements ......................... 159
      C.3.1 Isoparametric Case ................................. 160
      C.3.2 Quadrature ......................................... 161
      C.3.3 Stokes Operators .................................... 161

D  3rd–Order Adams–Bashforth with Variable Time Stepping  163

E  Direct Stiffness Summation Procedure ...................... 166

F  Raw Data  172
   F.1 Random Number Generators and Acceptance–Rejection Algorithm .... 172
   F.2 Effective Conductivity Surrogate Validation Raw Data .......... 172
   F.3 Creeping–Flow Permeability Surrogate Validation Raw Data .... 175
   F.4 Inertial Porous–Media Flow Raw Data ...................... 176
      F.4.1 Regular Array Results ............................. 176
      F.4.2 Random Array Results .............................. 178

Bibliography  183
Chapter 1

Introduction

1.1 Problems of Interest

Transport phenomena in heterogeneous systems play a critical role in many man-made and natural processes. Examples of engineering interest that will be addressed in this thesis include heat conduction in thermal fibrous composites, and viscous and inertial flows through porous media composed of randomly distributed cylindrical obstacles. Understanding of the former problem is important, for example, in the design of more economical high performance insulating and conducting materials [Kelly 1987], whereas the latter is indispensable in the analysis of ground water flows, fluidization [Davidson et al. 1985], convective heat exchangers [Hetsroni 1982], and aerosol filtration [Shapiro and Brenner 1990].

Analytical treatment of such systems is plagued by the nonlinearity of the problem, and the statistical nature of the microstructure; numerical treatment is complicated by the excessive degrees of freedom, the large range of disparate length scales, and the stiff geometries. Moreover, the question of "microstructure characterization" is an essential problem that plagues all theoretical analyses. From an engineering point of view, these difficulties are somewhat mitigated because we are primarily concerned with the effective properties which characterize the macroscopic behavior of the media. We review previous efforts to determine effective properties, and then describe our own approach.

1.2 Previous Work

We will describe the key trends in the large body of available literature. Most of the previous work has been dedicated to the linear problems of conductivity and viscous porous media flow and, hence, the discussion of inertial flows is deferred to the end of the section. The essential problem in reconciling any proposed study with experimental results is the difficulty in characterizing the microstructure; this has led to the development of two complementary approaches:

Method of Bounds
The method of bounds focuses on the derivation of upper and lower limits to the macroscopic transport coefficient. Initial work by Hashin and Shtrikman [1962] for conductivity and Prager [1961] for (Stokes) permeability required as input only the component concentration. This limited microstructure information yields simple, easily applied, results but the bounds are, perforce, less than precise. For example, the structure-independent lower bound for a conducting matrix with insulating inclusions
is zero. Improving the bounds by providing higher statistical information characterizing the microstructure [Milton 1981, Torquato and Lado 1988] is practically difficult since it requires the measurement of higher-order correlation functions of real heterogeneous materials. (In fact, the measurement of five-point and higher-order correlation functions of real heterogeneous materials is beyond presently available technology [Torquato 1991].) Furthermore, the higher-order bounds are not tight for widely different components properties. A comprehensive survey of these techniques for both the conductivity and permeability problems can be found in the review by Torquato [1991].

**Structure Dependent Analyses**

The structure-dependent analyses typically use the "cell" model, in which the microstructure is an input to the problem. This approach, which considers a periodic cell containing a single or many randomly positioned particles dispersed in a continuous matrix, can treat very accurately cases in which the structure is either known or can be self-consistently computed (e.g., sedimentation). Furthermore, it can be systematically exercised to validate practical models resulting from structure-independent or phenomenological predictions.

Homogenization theory, by means of multiple scale analysis, allows for the rigorous deduction of the macroscopic coefficients from the knowledge of the microscopic, or cell, fields. Analytical solutions of these fields has been limited to structured (square or hexagonal) periodic arrays both for heat conduction [Perrins et al. 1979] and creeping flow [Sangani and Acrivos 1982] problems. These exact results allow for rigorous validation of numerical solutions. Semi-analytical solutions [Hashimoto 1959, Karila et al. 1989, Sangani and Yao 1988a, 1988b] have been proposed based on superposition of fundamental solutions followed by numerical treatment of truncated sums. While this approach may be successful in treating random supercells [Sangani and Yao 1988a, 1988b], its extension to nonlinear problems (e.g., inertial flow) is certainly not feasible. Until recently, numerical solution of the cell equations has been typically serial with limited statistical treatment [Winterfeld et al. 1981].

Combining the relative strengths of earlier analytical and computational approaches, a general first-principle framework has been proposed by Cruz and Patera [1993]. The methodology exploits a variational hierarchical decomposition procedure to recast the original problem as a sequence of three scale-decoupled micro-, meso- and macroscale subproblems. The mesoscale equations are then solved numerically by parallel finite element methods. The methodology has considered the two problems of conductivity and (Stokes) permeability, and by defining a correlation length, the results obtained are applicable to truly random media. Furthermore, the numerical framework can be extended to treat nonlinear effects where analytical strategies fail. The method is, however, very inefficient in cases in which clusters of close particles induce significant geometric stiffness. Moreover, the parallel performance of the implementation was unsatisfactory, and more efficient algorithms are required before the procedure could be extended to computationally-intensive nonlinear problems.

**Inertial Porous-Media Flows**

There are relatively few theoretical results for inertial porous-media flows, except for a few studies for structured arrays. Eidsath et al. [1983] and Edwards et al. [1989] performed numerical calculations using the Galerkin finite element method for various regular arrays.
Dybbas and Edwards [1982] performed flow visualization studies through media consisted of plexiglass spheres in hexagonal packing, and plexiglass rods arranged in a random three dimensional geometry. Their study shed light on the details of the flow regime as a function of Reynolds number, and on the onset of unsteadiness.

To date, no systematic studies have been published on inertial flows through random arrays. However, many empirical correlations – typically presented in terms of a pressure drop versus an appropriately defined Reynolds number – have been proposed, as compiled by Scheidegger [1974] and Davidson et al. [1985]. The most widely accepted relation for practical engineering application for regimes not described by the Carman–Kozeny equation is the Ergun relation [Ergun 1952].

1.3 Objectives

The theme of this research is to further develop the general methodology proposed by Cruz and Patera [1993] for the calculation of effective properties and correlation lengths:

![Diagram](attachment:image.png)

In particular, the specific goals of this research are:

1. The development and implementation of new microscale variational “nip elements” that permit rigorous and efficient treatment of geometrically stiff problems [Cruz, Ghaddar and Patera 1993].

2. The use of sensitivity derivatives of the, say, effective conductivity with respect to the nip elements parameters (e.g., size) to accurately decide on the incorporation/elimination or sizing of critical nips.

3. The development and implementation of more efficient, general, and automatic domain decomposition and parallel isoparametric finite element algorithms for the solution of the computationally intensive mesoscale problem.

4. The validation of simplified engineering models for the effective conductivity [Hashin 1970] and the (Stokes) permeability problems by systematic surrogate techniques [Yeşilyurt et al. 1993].

5. The application of the methodology to (two-dimensional) regular and random fibrous porous media flows at both low and moderate Reynolds numbers.

1.4 Outline of Thesis

This thesis is divided into 9 chapters. The outline of the chapters is as follows:
• In Chapter 2 we first describe the variational decomposition procedure developed in Cruz and Patera [1993]. We then revisit the mesoscale formulation of both the effective conductivity and Stokes permeability problems which is necessary for our subsequent developments. Lastly we extend the Stokes formulation to account for the inertial effects with all the necessary modifications.

• Chapter 3 is dedicated to the microscale analysis of the three problems of interest. Here we develop rigorous crude/sharp upper and lower bounds for the effective properties of interest which is necessary to mitigate the difficulties associated with the geometric stiffness of the mesoscale problem (here the nip region between close particles). We also formulate the use of sensitivity derivatives as a valuable aid in the design of nip elements.

• In Chapter 4 we describe the algorithmic ingredients required for the efficient parallel solution of the mathematical formulations. In particular we discuss: a new tensor-product domain partitioning strategy and related geometry and topology description; automatic nip generation; mesh generation; discretization and numerical solution of associated partial differential equations; special numerical treatment of nip-element boundary conditions; parallel implementation issues, and intra- and inter-processor improvements.

• In Chapter 5 we present selective examples to demonstrate the numerical benefits associated with the nip element method. We also present illustrative examples on the use of sensitivity derivative to assess the relative criticality of individual nip elements.

• In Chapter 6 we present a surrogate framework developed specifically for computational problems with noisy outputs typical of our measurements. The noisy effects that contaminate our data points arise due to the Monte-Carlo techniques used for the evaluation of deterministic integrals. The surrogate techniques are necessary because of the expensive, decoupled, and discontinuous nature of the simulation data points.

• In Chapter 7 we apply the surrogate framework presented in Chapter 6 to rigorously validate a structure–independent model for the effective conductivity.

• In Chapter 8 we present Stokes permeability results and discuss the Carman–Kozeny model as a potential surrogate for our fibrous media. We also construct a simple exponential model for prediction of the Stokes permeability for fibrous media.

• In Chapter 9 we present inertial porous–media flow results for regular and random arrays for selected concentrations and Reynolds numbers of practical relevance. We conclude with suggestions for future work.
Chapter 2

Hierarchical Variational Decomposition

In this chapter we describe the first-principle computational methodology for the prediction of effective properties and statistical correlation lengths of multicomponent random media. The purpose here is not to repeat the analysis which is described by [Cruz & Patera 1993], but rather to summarize the relevant details which are necessary for the developments to follow. We first present the hierarchical variational decomposition procedure of the original multiscale problem in section 2.1. Next we revisit the mathematical formulation of the thermal effective conductivity and creeping-flow permeability problems in sections 2.2 and 2.3, respectively. Lastly, we consider the new problem of inertial-flow permeability in section 2.4.

2.1 Decomposition Procedure of Multiscale Problem

The Decomposition Procedure of the original multiscale problem is best illustrated referring to the schematic shown in Figure 2-1. We consider an original multicomponent domain $\Omega_{or}$ of large scale (macroscale) $L$, consisting of two phases: a continuous phase of constant physical property (e.g., thermal conductivity); an isotropic dispersed phase of volumetric concentration $c$ consisting of co-oriented cylindrical inclusions of diameter $d$, $\epsilon = d/L \ll 1$, randomly distributed in the continuous phase. The aspect ratio of the cylindrical inclusions, length/diameter, is sufficiently large that the problem can be treated, effectively, as 2-dimensional. These inclusions represent insulation surfaces for the thermal composite problem or obstacles to the flow for the permeability problem.

The medium is assumed to be statistically homogeneous, that is, we postulate the existence of an intermediate length (mesoscale) $\lambda$, $d < \lambda < L$, of a representative volume element which is statistically identical to the whole specimen [Hashin 1983]. (Note that the statistical homogeneity assumption does not require that the concentration has to be constant throughout the original medium, but that the variation in the concentration occurs over a length scale $\gg \lambda$.) We shall also assume, for simplicity, that the composite is statistically isotropic, that is, on the average, the composite is characterized by scalar effective properties.

A potential difference (e.g., temperature, pressure) is imposed over the large scale $L$ of $\Omega_{or}$, along with boundary conditions prescribed on $\partial \Omega_{or}$ (e.g., adiabatic boundaries, walls). This is illustrated in for the thermal composite problem in Figure 2-2 in which the boundary
Figure 2-1: Decomposition of the original multiscale problem into the micro-, meso-, and macroscale subproblems.

$\partial \Omega_{or}$ is divided into 3 segments $\Gamma_0$, $\Gamma_1$, and $\Gamma_{ad}$ on which, respectively, uniform temperatures $T_0$, $T_1$, and adiabatic conditions are imposed. With this description of the original problem, the scale-decoupled micro-, meso-, and macroscale subproblems are described next, starting with the latter:

**Macroscale**

In the macroscale problem, the original multiscale domain is replaced with a (potentially anisotropic) homogenized medium characterized by effective properties. The governing equations of the transport phenomenon of interest (e.g., heat conduction, flow resistance) and associated boundary conditions are solved to compute the quantities of engineering interest (e.g., heat flux, flow rate). The computational requirement of this problem is only
a fraction, $O(\epsilon^2)$, of the cost required for the original complex problem.

**Mesoscale**

The mesoscale analysis supply the effective transport property characterizing the original problem for use in the macroscopic problem. The analysis encompasses four nested levels. In *level 1* we introduce a periodically replicated supercell, $\mathbf{y} \in [0, \lambda] \times [0, \lambda] \subset \mathbb{R}^2$, containing $N = 4c\lambda^2/\pi$ cylindrical inclusions (disks), as shown in Figure 2-3. The *mesoscale* region of interest, $\Omega$, is the region of the supercell occupied by the matrix component only: $\Omega = [0, \lambda]^2 \setminus \bigcup_{i=1}^N I_{\mathbf{y}_i}$, where $I_{\mathbf{y}_i}$ is the (periodically extended) inclusion disk with center $\mathbf{y}_i$ and unit diameter $d$. The placement of the cylinder centers within the supercell, $\{\mathbf{y}\}_N = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, is prescribed in terms of an assumed isotropic and homogeneous joint probability density function, $f_{\{\mathbf{y}\}_N}(\{\mathbf{y}\}_N ; c, \lambda)$. (Note that a particular configuration, in general, may be anisotropic, but, on the average, the properties are isotropic (scalar).)

The mesoscale partial differential equations governing the transport problem are deduced from periodic homogenization theory [Bensoussan et al. 1978]. The solution of these equations yields the (tensorial) *configuration* effective property of the particular realization under consideration. Here we are interested in the transverse configuration effective property (here either the effective conductivity $k$, or the permeability $\kappa$) which is defined as the actual flux across (say) $y_1$ plane of the supercell normalized by the product of the imposed potential gradient and the superficial area/depth, $\lambda$. (Note that for heat conduction, the longitudinal effective conductivity is trivial, in fact, structure-independent for our co-orientation fibrous medium.) For clarity, the following discussion is presented in terms of the effective conductivity, $k$; the treatment of the permeability, $\kappa$, is completely analogous.

The transverse configuration effective conductivity, $k(\{\mathbf{Y}\}_N, c, \lambda)$ is now a random variable governed by the postulated joint probability density function. The task of *level 2* is to determine the (by isotropy, scalar) supercell effective conductivity, $k_e(c, \lambda)$. (The analogous notation for the permeability is $\kappa_e$.) This is accomplished by computing the ensemble
Figure 2-3: Periodic supercell \([0, \lambda]^2\) containing \(N\) cylindrical inclusions of unity diameter \(d\), with centers located at \(y_i, i = 1, \ldots, N\).

average of \(k([Y]_N, c, \lambda)\) defined as

\[
k_e(c, \lambda) = \int_{([0, \lambda] \times [0, \lambda])^N} k([y]_N, c, \lambda) f([y]_N | ([y]_N, c, \lambda) dy.
\]

(2.1)

In order to compute (2.1) we need to know the inclusion joint probability density function, \(f([Y]_N)\). Unfortunately, in most physical systems, details of the inclusion distribution are, at best, costly to obtain, and, more typically, simply unavailable. Our arguably physically relevant inclusion joint probability density function is based on a random sequential addition process [Widom 1966, Cruz & Patera 1993]

\[
f([y]_N) = \left( \prod_{i=2}^N f_{y_i | [y_{i-1}, \ldots, y_1]} \right) f_{y_1},
\]

(2.2)

where

\[
f_{y_1} = \frac{1}{\lambda}, \quad y_1 \in [0, \lambda] \times [0, \lambda],
\]

(2.3)

and

\[
f_{y_j | [y_{j-1}, \ldots, y_1]}(y_j | [y_{j-1}, \ldots, y_1]) = \begin{cases} 0 & y_j \notin \mathcal{P}_j, \\ 1/ \int_{\mathcal{P}_j} dy & y_j \in \mathcal{P}_j, \end{cases}
\]

(2.4)

\[
\mathcal{P}_j = [0, \lambda] \times [0, \lambda] \setminus \bigcup_{n=1}^{j-1} D_{y_n},
\]

(2.5)

for \(j = 2, \ldots, N\). Here \(D_{y_n}\) is the (periodically extended) disk center \(y_n\) and twice the inclusion diameter.
In level 3 the assumed-to-exist $k_e^\infty(c)$ defined as

$$k_e^\infty(c) = \lim_{\lambda \to \infty} k_e(c, \lambda),$$  \hspace{1cm} (2.6)

and the correlation length function, $\Lambda(c)$, are determined. The latter, which is introduced to approximate this limiting result, is defined such that for $\lambda \geq \Lambda(c)$, (i) $k_e(c, \lambda)$ no longer changes appreciably, and (ii) the standard deviation of the random variable

$$K = k(\{Y\}_N, c, \lambda),$$

$\sigma_K(c, \lambda)$, is sufficiently small (see Figure 2-4).

In practice, $k_e(c, \Lambda(c))$ serves as an accurate effective conductivity for any macroscale (homogenized) problem for which the domain is large compared to $\Lambda(c)$. (More generally, we may consider any macroscale problem for which the concentration and inclusion distribution vary only over a lengthscale large compared to $\Lambda(c)$.)

![Figure 2-4: Illustration of the correlation length $\Lambda(c)$. I-shaped bars represent $\sigma_K(c, \lambda)$.](image)

Finally, in level 4, the functional dependencies of the effective property and correlation length on various relevant parameters, are determined.

Microscale

The calculation of the configuration effective property, $k$, is frequently plagued by the geometrical stiffness that arises when disparate length scales are present. The latter is caused by the excessive closeness of the inclusions which from very small gaps – here called nip regions – which in severe instances may prevent subsequent numerical treatment. The microscale analysis is concerned with developing analytical models for the nearfield behavior of clustered inclusions to be incorporated in the mesoscale problem. In particular, we develop a variational–bound nip–element method for geometrically stiff problems which combines variational bounding techniques and the finite element method. This method is discussed in detail in the next chapter.
2.2 The Thermal Composite Problem

Here we assume the cylindrical inclusions to be insulating fibers; the continuous phase to possess a thermal conductivity, $k_{co}$, and a global temperature gradient $\frac{\Delta T}{L}$ is imposed across the macroscale length $L$. The configuration effective property of interest is now the effective conductivity, $k$.

The partial differential equation for the mesoscale temperature perturbation, $\chi(y)$, is given by

$$-\frac{\partial}{\partial y_i}(k_{co} \frac{\partial \chi}{\partial y_i}) = 0 \quad \text{in } \Omega, \tag{2.7}$$

with Neumann boundary condition,

$$- k_{co} \frac{\partial \chi}{\partial y_i} n_i = -k_{co} \frac{\Delta T}{L} n_1 \quad \text{on } \partial \Omega, \tag{2.8}$$

and $\lambda$-periodicity of $\chi$ in both $y_1$ and $y_2$. It is further required for uniqueness that $\int_{\Omega} \chi \, dy = 0$. The Neumann boundary condition can be realized by recognizing that macroscale and the mesoscale temperature components will contribute to the adiabatic inclusion boundary condition to the same order.

In the following presentation we shall pursue a nondimensional formulation by assuming that $\chi$ is normalized by the characteristic temperature $\frac{\Delta T_d}{L}$; the configuration effective conductivity, $k$ is normalized by $k_{co}$; and all lengths present are scaled by the diameter $d$. The nondimensional forms of equations (2.7) and (2.8) are respectively

$$-\frac{\partial^2 \chi}{\partial y_i \partial y_i} = 0 \quad \text{in } \Omega \tag{2.9}$$

and

$$-\frac{\partial \chi}{\partial y_i} n_i = -n_1 \quad \text{on } \partial \Omega. \tag{2.10}$$

(Although we have used the same variable notation for sake of simplicity, it should be understood that all variables are hereafter nondimensional unless otherwise indicated.)

Introducing the functional $J_\Omega(w)$

$$J_\Omega(w) = \int_{\Omega} \frac{\partial w}{\partial y_k} \frac{\partial w}{\partial y_k} \, dy - 2 \int_{\Omega} \frac{\partial w}{\partial y_1} \, dy, \tag{2.11}$$

it can be shown [Cruz & Patera 1993] that

$$\chi = \arg \min_{w \in H^1_{#,0}(\Omega)} J_\Omega(w), \tag{2.12}$$

where $H^1_{#,0}(\Omega) = \{ w \in H^1_{#}(\Omega) \mid \int_{\Omega} w \, dy = 0 \}$, and $H^1_{#}(\Omega)$ is the space of all $\lambda$-doubly periodic functions (subscript #) for which both the function and derivative are square-integrable over $\Omega$ [Adams 1975]. Here summation over repeated indices is assumed, and $dy = dy_1, dy_2$. The weak form for $\chi$ then derives from the first variation of the functional $J_\Omega(w)$: Find $\chi \in H^1_{#,0}(\Omega)$ such that

$$\int_{\Omega} \frac{\partial v}{\partial y_k} \frac{\partial \chi}{\partial y_k} \, dy = \int_{\Omega} \frac{\partial v}{\partial y_1} \, dy \quad \forall v \in H^1_{#,0}(\Omega). \tag{2.13}$$
We shall refer to this problem over $\Omega$, in which we pursue no microscale preparation, as the “bare” problem.

The configuration effective conductivity, $k$, is expressed in terms of the mesoscale temperature perturbation, $\chi$ as

$$
  k = (1 - c) - \frac{1}{\lambda^2} \int_{\Omega} \frac{\partial \chi}{\partial y_1} \, dy ,
$$

which, from (2.11), (2.12), and (2.13), can be expressed as

$$
  k = (1 - c) + \frac{1}{\lambda^2} \min_{w \in H_0^1(\Omega)} J_\Omega(w) .
$$

Finally, defining

$$
  a_\Omega(w) = \frac{1}{\lambda^2} \int_{\Omega} \frac{\partial}{\partial y_k} (w - y_1) \frac{\partial}{\partial y_k} (w - y_1) \, dy ,
$$

we can write equation (2.15) in the equivalent form

$$
  k = a_\Omega(\chi) = \min_{w \in H_0^1(\Omega)} a_\Omega(w) .
$$

For future reference, we note that, for any domain $\Psi \in \mathbb{R}^n$, admissible function space $S(\Psi)$, and $\phi$ satisfying

$$
  \phi = \arg \min_{w \in S(\Psi)} J_\Psi(w) ,
$$

then

$$
  a_\Psi(\phi) = \min_{w \in S(\Psi)} a_\Psi(w) ;
$$

we shall refer to (2.18)–(2.19) as “Property $A$.” For the general anisotropic case, analogous relations to (2.13) and (2.16) can be derived more directly by applying periodic homogenization theory [Bensoussan, Lions & Papanicolaou 1978, Auriault 1983, Bendsøe & Kikuchi 1988]. It is then readily demonstrated that the effective conductivity tensor is symmetric and positive–(semi)– definite; furthermore, the extremizing property (here, minimum) (2.17) extends to the full conductivity tensor in a form similar to the inequalities derived in [Nir, Weinberger & Acrivos 1975] for the components of the shearing tensor.

### 2.3 The Viscous Porous Media Problem

Here we assume the continuous phase to be a fluid with viscosity $\mu$ and density $\rho$; the inclusions are obstacles to the flow; and a global pressure gradient $\frac{\Delta P}{L}$ is imposed across the macroscale length $L$. The configuration effective property of interest is now the permeability, $\kappa$. We further assume that Reynolds number, $Re$, based on the average mesoscale velocity and the inclusion diameter is much less than unity such that the mesoscale velocity vector $u(y) = (u_1(y_1, y_2), u_2(y_1, y_2))$ and the mesoscale pressure, $p = p(y)$, satisfy the steady Stokes equations,

$$
  - \frac{\partial}{\partial y_j} (\mu \frac{\partial u_i}{\partial y_j}) + \frac{\partial p}{\partial y_i} = \frac{\Delta P}{L} \delta_{ij} \text{ in } \Omega \text{ for } i = 1, 2 ,
$$

(2.20)
\[- \frac{\partial u_i}{\partial y_i} = 0 \text{ in } \Omega, \quad (2.21)\]

where $\delta_{ij}$ is the Kronecker delta. The velocity must vanish on all obstacle boundaries and both the velocity and pressure must be $\lambda$-periodic in both $y_1$ and $y_2$. It is further required for uniqueness that $\int_\Omega p \, dy = 0$. The configuration permeability, $\kappa(c, \lambda, \{y\}_N)$ is expressed in terms of the mesoscale velocity $u_1$, through the well-known Darcy's law

\[< u_1 > = -\frac{\kappa}{\mu} < \frac{\Delta P}{L} >, \quad (2.22)\]

where $< >$ denotes the spatial average. (Note that $< u_2 > = 0$.)

In order to pursue a nondimensional formulation we shall assume that the pressure is normalized by $\frac{\Delta P}{L}$; the velocity components by the velocity scale

\[U_\mu = \frac{\Delta P}{\mu L} \frac{d^2}{L} \quad (2.23)\]

(the subscript $\mu$ is used to indicate viscous scaling, and to distinguish it from an alternative scaling (denoted $U_p$) based on the inertia which is used for the Navier–Stokes problem in Section 2.4); the permeability by the square of the inclusion diameter, $d^2$; and all the lengths present are scaled by the inclusion diameter $d$. The nondimensional forms of equations (2.20) and (2.21) are respectively

\[- \frac{\partial^2 u_i}{\partial y_j \partial y_j} + \frac{\partial p}{\partial y_i} = \delta_{ii} \text{ in } \Omega \text{ for } i = 1, 2 \quad (2.24)\]

\[- \frac{\partial u_i}{\partial y_i} = 0 \text{ in } \Omega. \quad (2.25)\]

(Again here we have used the same variable notation for sake of simplicity. It should be, however, understood that all variables are hereafter nondimensional unless otherwise indicated.)

We introduce the functional $J^P_\Omega(v)$ (related to the "excess dissipation"),

\[J^P_\Omega(v) = 2 \int_\Omega v_1 \, dy - \int_\Omega \frac{\partial v_i}{\partial y_i} \frac{\partial v_i}{\partial y_i} \, dy. \quad (2.26)\]

The correct mesoscale velocity field maximizes the functional $J^P_\Omega(v)$ over all admissible incompressible velocity fields [Girault & Raviart 1986, pp. 80-84]

\[u = \arg \max_{v \in Z} J^P_\Omega(v), \quad (2.27)\]

where $Z = \{(v_1, v_2) \in (H^1_0(\Omega), H^1_0(\Omega)) | \text{div } v = 0\}$, and $H^1_0(\Omega)$ is the space of all $\lambda$-doubly periodic functions which vanish on $\partial \Omega$, and for which both the function and derivative are square-integrable over $\Omega$. Equating the first variation of the functional $J^P_\Omega(u)$ to zero, we get

\[\int_\Omega \frac{\partial v_i}{\partial y_i} \frac{\partial v_i}{\partial y_i} \, dy = \int_\Omega v_1 \, dy. \quad (2.28)\]
From (2.27) and (2.28) we derive

$$J^P_{\Omega}(u) = \int_{\Omega} \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy . \tag{2.29}$$

In order to arrive at the standard velocity–pressure weak form, we first transform the constrained maximization (2.27) into an unconstrained saddle problem: that is, we enlarge the velocity space to include all functions \((v_1, v_2)\) in \((H^{1\#}_0(\Omega), H^{1\#}_0(\Omega))\), and introduce a Lagrange multiplier — the pressure, \(p\) — to impose the incompressibility constraint. Taking the first variation of the resulting Lagrangian, we obtain the weak form: Find \((u_1, u_2, p) \in (H^{1\#}_0(\Omega), H^{1\#}_0(\Omega), L^2_{\#.0}(\Omega))\) such that

$$\int_{\Omega} \frac{\partial v_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy - \int_{\Omega} \frac{\partial v_i}{\partial y_i} p \, dy = \int_{\Omega} v_1 \, dy \quad \forall (v_1, v_2) \in (H^{1\#}_0(\Omega))^2 , \tag{2.30}$$

$$- \int_{\Omega} q \frac{\partial u_i}{\partial y_i} \, dy = 0 \quad \forall q \in L^2_{\#.0}(\Omega) , \tag{2.31}$$

where \(L^2_{\#.0}(\Omega)\) is the space of all \(\lambda\)-doubly periodic functions \(q(y)\) which are square–integrable over \(\Omega\) (note that candidate pressures need not be continuous), and for which \(\int_{\Omega} q \, dy = 0\).

Based on Darcy’s definition, (2.22), and the scaling (2.23), the configuration permeability, \(\kappa\), takes the form

$$\kappa = \langle u_1 \rangle = \frac{1}{\lambda^2} \int_{\Omega} u_1 \, dy , \tag{2.32}$$

which, from (2.26) – (2.29), can be expressed as

$$\kappa = \frac{1}{\lambda^2} \int_{\Omega} \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy = \frac{1}{\lambda^2} J^P_{\Omega}(u) \, dy , \tag{2.33}$$

or equivalently

$$\kappa = \frac{1}{\lambda^2} \max_{v \in Z} J^P_{\Omega}(v) . \tag{2.34}$$

(We remind here that the definition of the supercell permeability, \(\kappa_\lambda\), is analogous to (2.1).)

The expression (2.34) is, of course, related to the Helmholtz stationary–dissipation principle, see for example [Batchelor 1967, Hill & Power 1956, Kim & Karrila 1991]. Again, the extremizing property (2.34) extends to the full permeability following the development in [Nir, Weinberger & Acrivos 1975].

### 2.4 The Inertial Porous Media Problem

In the event of important convective inertial effects, the fluid motion in the mesoscale supercell is governed by the (possibly unsteady) Navier–Stokes equations

$$\rho \left[ \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial y_j} \right] - \frac{\partial}{\partial y_j} \left( \mu \frac{\partial u_i}{\partial y_j} \right) + \frac{\partial p}{\partial y_i} = \frac{\Delta P}{L} \delta_{ii} \quad \text{in } \Omega \quad \text{for } i = 1, 2 , \tag{2.35}$$

$$- \frac{\partial u_i}{\partial y_i} = 0 \quad \text{in } \Omega , \tag{2.36}$$
subject to the no-slip condition at the obstacles boundaries, and the \( \lambda \)-periodicity of both the velocity and pressure in both \( y_1 \) and \( y_2 \) directions. Again, since the pressure can be determined only up to an arbitrary constant, we also need the constraint \( \int_{\Omega} p \, dy = 0 \). It should be pointed out that the mesoscale equations are actually “rationalized” rather than rigorously derived using periodic homogenization theory as has been possible for the Stokes problem. Here the nonlinear convective term precludes complete decoupling between the tensorial components of the macro-, and mesoscale subproblems (save for the special case of \( Re \ll 1 \) [Mei & Auriault 1990]) which poses series difficulties for subsequent solution. The adopted model, widely accepted in the literature [Eidsath et al. 1983, Edwards et al. 1990], is justified by a “control-volume” type argument in which the obvious contribution of the macroscale problem to the mesoscale problem is the linearized macroscale pressure appearing locally as \( \rho + \frac{\Delta P}{L} y_1 \). Furthermore, the interest here is not in the details of the flow field of the macroscale problem but rather the bulk permeability relating the pressure drop to the unidirectional flow rate, which is precisely what we calculate from the mesoscale problem. The latter should be, due to spatial averaging, less sensitive to the flow field details.

In order to pursue a nondimensional formulation we assume the velocity vector to be normalized by the velocity scale

\[
U^2_\rho = \frac{\Delta P d}{L \rho} ;
\]

(2.37)

the pressure to be normalized by \( \frac{\Delta P d}{L} \), the time by the time scale \( d/U \); the permeability by the diameter squared, \( d^2 \); and all the lengths present by the diameter \( d \). The nondimensional strong form can be written as

\[
\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial y_j} - \frac{1}{Re_\rho} \frac{\partial^2 u_i}{\partial y_j \partial y_j} + \frac{\partial p}{\partial y_i} = \delta_{ii} \text{ in } \Omega \text{ for } i = 1, 2 ,
\]

(2.38)

\[- \frac{\partial u_i}{\partial y_i} = 0 \text{ in } \Omega ,
\]

(2.39)

where Reynolds number \( Re_\rho \) is defined as

\[
Re_\rho = \frac{\rho U_\rho d}{\mu} .
\]

(2.40)

An alternative scaling for the velocity may be based on (2.23) which leads to a different Reynolds number \( Re_\mu = \frac{\rho U_\mu d}{\mu} \) in the momentum equation

\[
Re_\mu \left[ \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial y_j} \right] - \frac{\partial^2 u_i}{\partial y_j \partial y_j} + \frac{\partial p}{\partial y_i} = \delta_{ii} \text{ in } \Omega \text{ for } i = 1, 2 ,
\]

(2.41)

From (2.37) and (2.23), it can be easily deduced that

\[
Re_\mu = Re_\rho^2
\]

(2.42)

for the two velocity scalings to yield identical flow conditions. The choice between the two scalings is a matter of convenience, and we will adopt the first scaling (2.37).

The definition of the configuration permeability, equation (2.22), remains valid, although temporal averaging may also be required in unsteady situations. The perme-
ability, however will also depend on the Reynolds number, \(Re_\rho\), characterizing the flow, \(\kappa = \kappa(y_1, \cdots, \kappa, \lambda, Re_\rho)\). Based on our nondimensionalization choice, the expression for the temporally averaged configuration permeability, \(\kappa\), as defined by Darcy’s equation takes the form

\[
\kappa = \frac{1}{Re_\rho} \frac{1}{T} \int_0^T <u_1(t)> \, dt = \frac{1}{Re_\rho} \frac{1}{\lambda^2} \frac{1}{T} \int_0^T \int_\Omega u_1(t) \, dy \, dt ,
\]

where \(T\) is the period of the unsteady flow.

In general, other forms describing the flow–rate/pressure–drop relation can be used. These latter relations are typically in the form

\[
< \frac{\Delta P}{L} >= a < u > + b < u >^m ,
\]

first proposed by Forchheimer (1901) which extends classical Darcy’s law by introducing the last nonlinear term to account for inertial losses. The coefficients \(a\) and \(b\) are constants that depends on the porosity and pore characteristics, and \(m\) is typically (or close to) 2. Many attempts have been made to accurately define the parameters in (2.44) to fit the body of experimental data, among which the Ergun equation [Ergun 1952] is most widely used. In our formulation we adopt the basic definition of the permeability through equation (2.22) which involves no modeling assumptions. The data can then be systematically used to validate the other engineering relations.

We also define a Reynolds number based on the average velocity, \(< u_1 >\) (note that \(< u_2 >= 0\))

\[
Re_{<u_1>} = \frac{\rho < u_1 > d}{\mu}.
\]

Using (2.43) and (2.45), we derive the following relation

\[
Re_{<u_1>} = \kappa Re_\rho^2 .
\]

The weak form of equations (2.38) and (2.39) takes the form:

Find \((u_1, u_2, p) \in (H^1_0(\Omega), H^1_0(\Omega), L^2_{\#}(\Omega))\) such that

\[
\int_\Omega \frac{\partial u_i}{\partial t} \, dy + \int_\Omega u_j \frac{\partial u_i}{\partial y_j} \, dy + \frac{1}{Re_\rho} \int_\Omega \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy - \int_\Omega \frac{\partial u_i}{\partial y_j} \, dy - \int_\Omega q \frac{\partial u_i}{\partial y_i} \, dy = 0 \quad \forall (u_1, u_2) \in (H^1_0(\Omega))^2 , \quad (2.47)
\]

\[
\int_\Omega \frac{1}{2} u_1^2 \, dy + \int_\Omega u_i u_j \frac{\partial u_i}{\partial y_j} \, dy + \frac{1}{Re_\rho} \int_\Omega \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy = \int_\Omega u_1 \, dy , \quad (2.49)
\]

where we have used the incompressibility condition (2.39). Due to \(\lambda\)–periodicity of the
velocity, the convective term, \( \int_{\Omega} u_i u_j \frac{\partial u_i}{\partial y_j} \, dy \), vanishes. Indeed

\[
\int_{\Omega} u_i u_j \frac{\partial u_i}{\partial y_j} \, dy = \frac{1}{2} \int_{\Omega} \frac{\partial u_j u_i^2}{\partial y_j} \, dy = \frac{1}{2} \int_{\partial \Omega} u_j u_i^2 \cdot n_j \, ds = 0 ,
\]

(2.50)

where \( ds \) is a differential element of \( \partial \Omega \) defined by the outer unit normal \( n \). Here we again use (2.39) followed by application of the divergence theorem. Using (2.50) and averaging (2.49) over one period, \( T \), we get

\[
\frac{1}{T} \int_0^T \int_{\Omega} \frac{1}{2} \frac{\partial u_i^2}{\partial t} \, dy \, dt + \frac{1}{T} \int_0^T \int_{\Omega} \frac{1}{Re} \int_{\Omega} \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy \, dt = \frac{1}{T} \int_0^T \int_{\Omega} u_1 \, dy \, dt .
\]

(2.51)

The first term on the left–hand–side of (2.51) vanishes

\[
\frac{1}{T} \int_0^T \int_{\Omega} \frac{1}{2} \frac{\partial u_i^2}{\partial t} \, dy \, dt = \int_{\Omega} \frac{1}{T} \int_0^T \frac{1}{2} \frac{\partial u_i^2}{\partial t} \, dt \, dy = \int_{\Omega} 0 = 0 .
\]

(2.52)

From (2.43), (2.51), and (2.52), we conclude that

\[
\kappa Re^2 = \frac{1}{T} \int_0^T \frac{1}{\lambda^2} \int_{\Omega} \frac{\partial u_i}{\partial y_k} \frac{\partial u_i}{\partial y_k} \, dy \, dt = \frac{1}{T} \int_0^T \frac{1}{\lambda^2} J^{P}_{\Omega}(u) \, dt ,
\]

(2.53)

which is identical to the "temporally–averaged" (2.33) despite the presence of convective contributions. Now given that \( u \) is not the Stokes solution as defined by (2.30) and (2.31), and hence property (2.34) is lost, it follows that \( J^{P}_{\Omega}(u) \) and hence, \( \kappa Re^2 \) can only decrease relative to the Stokes permeability. (We remark that the factor \( Re^2 > 0 \) appears only because of our nondimensionalization choice. In fact, if the alternative choice based on (2.41) is used, equation (2.53) would not involve the factor \( Re^2 \).)
Chapter 3

Microscale Analysis and Bounds

This chapter is dedicated to the microscale analysis which is necessary to mitigate the numerical difficulties associated with calculation of the configuration effective properties ($k$, and $\kappa$). We present in Section 3.1 a general introduction to the new approach. In Section 3.2 we develop the microscale analysis for the thermal composite problem: we present the variational inner–outer problem decomposition and associated weak forms; the extremizing properties of the configuration effective conductivity bounds are motivated physically and proven mathematically. In Section 3.3 we develop the microscale analysis for both the viscous and inertial porous–media–flow problem. Lastly in Section 3.4 we formulate the use of sensitivity derivatives for a posteriori analysis of the nip–element method. We indicate that all the present formulation here and in later chapters are based on the nondimensional problems as defined in Chapter 2. We also remark that the inclusion diameter $d$ has been set to unity through out this work; however, we shall often keep the symbol $d$ to indicate a length unit and avoid any confusion in nondimensionality.

3.1 Introduction

The microscale analysis can take two forms: the development of appropriate models for the interfacial phenomena between the continuous and dispersed phases; the development of analytical approximate models for the nearfield behavior in the tiny gaps, nips, formed by very close inclusions. The former is important when internal inclusion dynamics are important. For example, for the problem of thermal composite with conducting fibers, contact resistance may play an important role in the overall composite conduction capability [Hasselman & Johnson 1987]. For the permeability problem, these interfacial effects can be due to surface tension phenomenon as in the study of drops and bubbles [Clift, et al. 1978], dust accumulation in filtration [Friedlander 1977], or even chemical reactions. In our context, the inclusions are either perfect insulators or solid wall and hence we shall not pursue the interfacial effects.

Instead, we shall develop the nip–region–based analytical models, and variationally integrate these models into a unified numerical procedure. This latter analysis is necessary to mitigate the numerical difficulties associated with the geometric stiffness of the mesoscale problem, which have been frequently encountered in the previous calculations [Cruz & Patra 1993] in which no cluster models were included. This analysis will provide both accurate approximations for the effective properties and rigorous a posteriori error estimates, and will enhance the statistical analysis by incorporating otherwise numerically impossible cases.
More generally, we develop a hybrid framework for geometrically stiff problems which exploits the complementary advantages of variational bounding techniques and the finite element method; the method is rigorously applicable to problems in which the quantity of interest is the extremum of a (quadratic, symmetric, positive–(semi)definite) functional. The method proceeds by decomposing the geometrically stiff problem into an inner problem, defined on the smallest scales (nip regions), and an outer problem, defined on the larger scales. Simplifying assumptions allow the inner Euler equation to be solved analytically; next, this inner solution is reflected in “coupling” boundary conditions on a now well-conditioned outer problem, which can then be readily solved numerically by “almost-standard” finite element methods. Thanks to the functional–maximizing property of the effective quantities sought, convergent crude/sharper upper and lower bounds can rigorously be constructed by simple/sophisticated models in the nip regions; the accuracy of the bounds is directly controlled by the number and extent of the excised nip regions. Note that our method is not appropriate for finding uniform approximations for (local) field variables.

The merger of analytical and numerical techniques to overcome the stiffness difficulties has been attempted before with significantly improved performance over solely analytical and solely numerical approaches. For example, for the class of problems in which the stiffness arises due to the presence of a small parameter, Bar–Yoseph and Israeli [1986] successfully applied the singular perturbation theory to develop asymptotic finite element methods. Similarly, the simulation of external high-Reynolds-number flows routinely incorporates the inner–outer problem decomposition technique of boundary-layer theory to mitigate convection–diffusion stiffness.

We should point out that the variational bounds we construct are, for the most part, not new (note also that we do not pursue minimum complementary energy principles, as minimum energy principles are more readily married to standard finite–element formulations). In particular, our estimates derive: generally, from the classical variational estimation techniques of space expansion, space restriction, and domain embedding and inclusion [Hill & Power 1956, Keller et al. 1967, Kim & Karrila 1991, Maday & Patera 1993]; and, more specifically, from earlier work on heat transfer variational bounds [Elrod 1974, Magen et al. 1988]. The techniques by which we integrate these bounds into a numerical procedure are, however, new.

3.2 The Thermal Composite Problem

Our crude/sharper upper and lower bounds for the effective conductivity are constructed based on simple/sophisticated superconducting and insulating heat conduction models in the nip regions. These models are summarized in Figure 3-1. We begin the analysis by a formal description of the geometry of the nip region between two close inclusions in Section 3.2.1. Then in Sections 3.2.2 and 3.2.3, respectively, we consider the crude upper and lower bounds involving rather draconian assumptions on the form of the solution in the nip regions. The more sophisticated sharper upper and lower bounds, which will permit a form of convergence as the nip size shrinks, are covered in Sections 3.2.4 and 3.2.5 respectively. To simplify the derivations, a single pair of close inclusions is considered. Extension of the single–pair analysis to multiple inclusions is discussed in Section 3.2.6. Although the bounds are derived for the scalar configuration effective conductivity, $k$, extension to the full configuration effective conductivity tensor can be pursued following the treatment described in [Nir et al. 1975].
3.2.1 Nip Geometry

Consider a (initially single) pair of inclusions, shown in Figure 3-2, with centers, say, $y_1$ and $y_2$ in $[0, \lambda]^2$ separated by a distance $1 + \alpha$ ($\alpha$ typically small), for which the segment connecting the inclusion centers, $\overline{y_1y_2}$, is inclined by an angle $\theta$ from the unit–vector in the $y_1$ direction. Denoting the unit–vector perpendicular to $\overline{y_1y_2}$ as $r$, we define the closed “nip” region, $\mathcal{D}$ (shaded area in Figure 3-2), as the intersection of $\Omega$ with the rectangle described by the four corners $y_1 + \beta r$, $y_2 + \beta r$, $y_2 - \beta r$, and $y_1 - \beta r$, where $\beta \leq \frac{1}{2}$. (In three space dimensions the nip region will be, for example, a cylinder with concave “spherical” endcaps.) The parts of the segments $(y_1 + \beta r) (y_2 + \beta r)$ and $(y_1 - \beta r) (y_2 - \beta r)$ which lie within $\Omega$ are denoted $\Gamma_1$ and $\Gamma_2$, respectively. We now introduce the modified mesoscale domain, $C = \Omega \setminus \mathcal{D}$, in which the nip region has been removed.

In the nip region we define $\eta(y'_2)$, the variable inter–inclusion distance parallel to $\overline{y_1y_2}$,

$$\eta = 1 + \alpha - \sqrt{1 - 4y'_2^2}, \quad (3.1)$$

where the nip–transformed coordinates $(y'_1, y'_2)$ are obtained by translating the origin of $(y_1, y_2)$ to $O'$ (Figure 3-2), and rotating the axes through the angle $\theta$. The affine isometric transformation between the system $(y_1, y_2)$ and the system $(y'_1, y'_2)$ is thus given by

$$\begin{pmatrix} y'_1 \\ y'_2 \end{pmatrix} = \begin{pmatrix} l_1 \\ l_2 \end{pmatrix} + \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad (3.2)$$

where $l_1, l_2$ are real constants. As (3.2) is volume preserving,

$$\int_{\psi} f(y_1, y_2) \, dy_1 \, dy_2 = \int_{\psi} f(y_1[y'_1, y'_2], y_2[y'_1, y'_2]) \, dy'_1 \, dy'_2 \quad (3.3)$$

for any real function $f$ and any domain $\psi \in \Omega$. 

27
Figure 3-2: Geometry of the nip region $\mathcal{D}$ (shaded).

In the next two Sections, we construct rigorous lower and upper bounds for the configuration effective conductivity,

$$k_{LB} \leq k \leq k_{UB},$$

(3.4)

based only on solutions defined over $\mathcal{C}$: we avoid the computationally intensive nip region, while maintaining strict control over the resulting error.

### 3.2.2 Crude Upper Bound

A crude upper bound for $k$, $k^c_{UB}$, for the one-nip configuration shown in Figure 3-2 can be obtained by assuming that the entire nip region $\mathcal{D}$ has infinite conductivity, or equivalently, that the nip region is isothermal. This is readily motivated physically: by replacing a subregion of the matrix of finite conductivity with a hypothetical infinitely conducting material (Figure 3-1a), a relatively more conducting composite is expected, as heat flow is facilitated in the nip region. We now derive the weak form for the modified outer problem, and then turn to the simple proof of the claimed bounding property.

To begin, we consider the modified mesoscale problem

$$x^c_{UB} = \arg \min_{w \in W_{\#, UB}(\Omega)} J_\Omega(w),$$

(3.5)

where $x^c_{UB}$ is a (new) mesoscale temperature field, and $W_{\#, UB}(\Omega)$ is the space of all functions $w(y) \in H^1_{\#,0}(\Omega)$ for which $w|_{\mathcal{D}} = C$, $C \in \mathbb{R}$ part of the solution; note that $W_{\#, UB}(\Omega) \subset H^1_{\#,0}(\Omega)$. Expressing the functional $J_\Omega(w)$ as

$$J_\Omega(w) = J_C(w|_\mathcal{C}) + J_\mathcal{D}(w|_\mathcal{D}),$$

(3.6)

where $w|_\mathcal{C}$ and $w|_\mathcal{D}$ refer to the restrictions of $w(y)$ to $\mathcal{C}$ and $\mathcal{D}$, respectively, we can break
problem (3.5) into an inner problem,
\[ \chi_{UB, in} \{ \overline{C} \} = \arg \min_{w \in W_{UB}(D)} J_D(w), \]  
and an outer problem,
\[ \chi_{UB, out} = \arg \min_{w \in W_{\#, UB}(C)} \left[ J_C(w) + J_D(\chi_{UB, in} \{ w|_{\Gamma_1 \cup \Gamma_2} \}) \right]. \]  

Here \( \chi_{UB, in} \{ \overline{C} \} \) is an abbreviated notation for \( \chi_{UB, in} \{ y_1, y_2; \overline{C} \} \); \( W_{UB}(D) \) is the (rather trivial) set of all functions \( w(y) \in H^1(D) \) for which \( w = \overline{C}, \overline{C} \in \mathbb{R} \) given (isothermal); \( W_{\#, UB}(C) = \{ w \in H^1_{\#, 0}(C), w|_{\Gamma_1} = w|_{\Gamma_2} = C \}, C \in \mathbb{R} \) part of the solution; and
\[ \chi_{UB, out} = \chi_{UB} |_{\mathbb{R}} + s, \quad \chi_{UB, in} \{ \chi_{UB, out} |_{\Gamma_1 \cup \Gamma_2} \} = \chi_{UB} |_{\mathbb{R}} + s, \]  

where \( s \in \mathbb{R} \) is a shift such that both \( \int_{\Omega} \chi_{UB} \, dy = 0 \) and \( \int_{\overline{C}} \chi_{UB, out} \, dy = 0 \) may obtain. The global minimization procedure (3.5)--(3.9) is somewhat analogous to minimizing a function \( g(\rho_1, \rho_2) \) over two variables: we first fix \( \rho_2 = \overline{\rho}_2 \) and find the minimum of \( g \) with respect to \( \rho_1 \) in terms of the constraint (inner: \( h(\overline{\rho}_2) = \min_{\rho_1} g(\rho_1, \overline{\rho}_2) \)); next, we allow the artificial constraint to vary, and minimize the now single-variable function (outer: \( \min_{\rho_1, \rho_2} g(\rho_1, \rho_2) = \min_{\rho_2} h(\overline{\rho}_2) \)).

The inner problem constitutes the microscale problem in our macro--meso--microscale decomposition of the original multiscale problem (Chapter 2). Solution of the inner problem is trivially obtained in this case, since by assumption \( \chi_{UB, in} \{ \overline{C} \} = \overline{C} \). The outer problem thus becomes
\[ \chi_{UB, out} = \arg \min_{w \in W_{\#, UB}(C)} J_C(w), \]  

since
\[ J_D(\chi_{UB, in} \{ \overline{C} \}) = 0. \]  

Taking the first variation of \( J_C(w) \), we obtain the weak form: Find \( \chi_{UB, out} \in W_{\#, UB}(C) \) such that
\[ \int_{\overline{C}} \frac{\partial v}{\partial y_k} \frac{\partial \chi_{UB, out}}{\partial y_k} \, dy = \int_{\overline{C}} \frac{\partial v}{\partial y_1} \, dy \quad \forall v \in W_{\#, UB}(C). \]  

Equation (3.12) differs from equation (2.13) in two aspects: the domain \( C \) replaces \( \Omega \); and the space \( W_{\#, UB}(C) \) replaces \( H^1_{\#, 0}(\Omega) \).

Finally, the configuration effective conductivity crude upper bound is then given by
\[ k_{UB} = a_\Omega(\chi_{UB}), \]  

which, from (2.16), (3.9), (3.11), and (3.12), can be expressed as
\[ k_{UB} = (1 - c) - \frac{1}{\chi^2} \int_{\overline{C}} \frac{\partial \chi_{UB, out}}{\partial y_1} \, dy, \]  

which involves only the outer solution over the modified mesoscale supercell. The mathematical proof that \( k_{UB} \) is, indeed, an upper bound for \( k \) follows from standard function
space restriction arguments:

\[ k_{\text{UB}}^e = a_\Omega(\chi_{\text{UB}}^e) = \min_{w \in W_{#,0}^1(\Omega)} a_\Omega(w) \]
\[ \geq \min_{w \in H_{#,0}^1(\Omega)} a_\Omega(w) = a_\Omega(\chi) = k, \]  

(3.15) \hspace{1cm} (3.16)

where \( \chi \) is the solution of (2.13) of Section 2.2 of Chapter 2. Equality (3.15) follows from Property \( A, (2.18)-(2.19) \); the inequality (3.16) follows from \( W_{#,0}^1(\Omega) \subset H_{#,0}^1(\Omega) \).

### 3.2.3 Crude Lower Bound

A crude lower bound for \( k, k_{\text{LB}}^c \), can be obtained by assuming that the entire nip region is filled with an insulator, or equivalently, that the (new) mesoscale "outer" temperature field, \( \chi_{\text{LB}}^c \), satisfies homogeneous natural (adiabatic) boundary conditions on the \( \Gamma_l, l = 1, 2 \). This is readily motivated physically: by replacing a subregion of the matrix of finite conductivity with an isotropic insulating material (Figure 3-1b), a relatively less conducting composite is expected, as heat flow is impeded in the nip region. Note that, in practice, at low and moderate concentrations, this locally crude bound may be globally rather sharp, since the close–particle nip regions will typically not be the preferred conduits of heat.

The lower bound mesoscale problem is given by

\[ \chi_{\text{LB}}^c = \arg_{w \in X_{#,\text{LB}}(C)} \min_{w \in X_{#,\text{LB}}(C)} J_C(w), \]

(3.17)

where \( X_{#,\text{LB}}(C) \) is the space of all \( \lambda \)-doubly periodic functions \( w(y) \) for which both the function and derivative are square–integrable over \( C \), and for which \( \int_C w \, dy = 0 \); note that, in fact, \( X_{#,\text{LB}}(C) = H_{#,0}^1(C) \). Since the problem is defined on \( C \), the inner, or microscale, problem over \( D \) is irrelevant.

Taking the first variation of \( J_C(w) \) we obtain the weak form: Find \( \chi_{\text{LB}}^c \in X_{#,\text{LB}}(C) \) such that

\[ \int_C \frac{\partial v}{\partial y_k} \frac{\partial \chi_{\text{LB}}^c}{\partial y_k} \, dy = \int_C \frac{\partial w}{\partial y_1} \, dy \quad \forall v \in X_{#,\text{LB}}(C). \]

(3.18)

The configuration effective conductivity crude lower bound is then given by

\[ k_{\text{LB}}^c = a_C(\chi_{\text{LB}}^c), \]

(3.19)

which, from (2.16) and (3.18), can also be written as

\[ k_{\text{LB}}^c = (1 - \tilde{c}) - \frac{1}{\lambda^2} \int_C \frac{\partial \chi_{\text{LB}}^c}{\partial y_1} \, dy, \]

(3.20)

where

\[ \tilde{c} = 1 - \frac{\int_C dy}{\lambda^2} \]

(3.21)

is the effective concentration (that is, volume fraction of the insulating component) of the modified mesoscale domain.

The physically plausible claim that \( k_{\text{LB}}^c \) is a lower bound for \( k \) is readily proven mathematically by embedding (or domain inclusion):

\[ k_{\text{LB}}^c = a_C(\chi_{\text{LB}}^c) = \min_{w \in X_{#,\text{LB}}(C)} a_C(w) \]

(3.22)
\[ \leq a_c(\chi|c + s') = a_c(\chi|c) \] (3.23)
\[ \leq a_N(\chi) = k, \] (3.24)

where \( \chi|c \) here refers to \( \chi \), the solution of (2.13), restricted to \( C \), and \( s' \in R \) is the shift required such that \( \int\int (\chi|c + s') dy = 0 \). The equality (3.22) follows from Property \( A \); the inequality (3.23) derives directly from \( \chi|c + s' \in X_{\#*,LB}(C) \); the inequality (3.24) follows from the positive-(semi)definiteness of the quadratic form (2.16).

3.2.4 Sharp Upper Bound

A sharper upper bound for \( k \), \( k_{UB}^* \), can be obtained by assuming that the temperature, \( \chi_{UB}^* \), in the nip region is only a function of the direction perpendicular to the inter–center segment \( \overline{y_1y_2} \). This is equivalent to replacing the matrix component in the nip by a hypothetical anisotropic material with infinite continuous–phase conductivity parallel to \( \overline{y_1y_2} \) (Figure 3-1c); physically, this replacement leads to a sharper upper bound than \( k_{UB}^* \) because heat flow is artificially facilitated by a lesser degree than with an isotropic infinitely conducting nip.

The mesoscale problem associated with the sharper upper bound is given by

\[ \chi_{UB}^* = \arg \min_{w \in Y_{\#*,UB}(\Omega)} J_{\Omega}(w), \] (3.25)

where \( Y_{\#*,UB}(\Omega) \subset H^1_{\#*,0}(\Omega) \) is the space of all functions \( w(y) \in H^1_{\#*,0}(\Omega) \) for which \( w|_{\Gamma_l} = C_l \), \( l = 1, 2 \), \( C_l \in R \) part of the solution, and for which

\[ \frac{\partial w}{\partial y'} = 0 \quad \forall y \in D. \] (3.26)

Proceeding as in (3.6), we break problem (3.25) into an inner problem

\[ \chi_{UB,in}^*(\overline{C_1}, \overline{C_2}) = \arg \min_{w \in Y_{UB}(D)} J_D(w), \] (3.27)

and an outer problem,

\[ \chi_{UB,out}^* = \arg \min_{w \in \overline{Y}_{\#*,UB}(C)} [J_C(w) + J_D(\chi_{UB,in}^*(w|_{\Gamma_1}, w|_{\Gamma_2})], \] (3.28)

where: \( Y_{UB}(D) \) is the set of all functions \( w(y) \in H^1(D) \) which satisfy (3.26) and for which \( w|_{\Gamma_l} = \overline{C_l} \), \( l = 1, 2 \), \( \overline{C_l} \in R \) given; \( \overline{Y}_{\#*,UB}(C) = \{ w \in H^1_{\#*,0}(C), w|_{\Gamma_l} = C_l, l = 1, 2 \}, C_l \in R \) part of the solution; and

\[ \chi_{UB,out} = \chi_{UB}^*|c + s'', \chi_{UB,in}^*(\chi_{UB,out}^*|_{\Gamma_1}, \chi_{UB,out}^*|_{\Gamma_2}) = \chi_{UB}^*|D + s'', \] (3.29)

where \( s'' \in R \) is the requisite shift such that both \( \int \chi_{UB}^* dy = 0 \) and \( \int C \chi_{UB,out}^* dy = 0 \) may obtain.

The inner problem is solved by taking the first variation of the functional \( J_D(w) \). First, from (2.11), (3.2), (3.3), and (3.26), we obtain

\[ \int_D \frac{\partial w}{\partial y_2} \frac{\partial \chi_{UB,in}^*}{\partial y_2} dy' + \sin \theta \int_D \frac{\partial w}{\partial y_2} dy' = 0 \quad \forall w \in Y_{UB}'(D), \] (3.30)
where \( dy' = dy_1' \ dy_2' \), and \( Y_{UB}(D) \) is the space of all functions \( w(y) \in Y_{UB}(D) \) for which \( w|_{\Gamma_1} = w|_{\Gamma_2} = 0 \) (\( \nu = 0 \) at \( y_2' = \pm \beta \), since the \( \chi_{UB, in}^{l} |_{\Gamma_1} = \overline{C}_l \), \( l = 1, 2 \), are fixed for the inner problem). Performing the integrations in the \( y_1' \) direction, we obtain

\[
\int_{-\beta}^{\beta} \frac{\partial u}{\partial y_2'} \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \eta \ dy_2' + \sin \theta \int_{-\beta}^{\beta} \frac{\partial u}{\partial y_2'} \eta \ dy_2' = 0 \quad \forall \nu \in Y_{UB}^{l}(D),
\]

(3.31)

where \( \eta \) is given by (3.1). Next, integrating both terms in (3.31) by parts (assuming sufficient regularity), we obtain

\[
\int_{-\beta}^{\beta} \nu \left[ \frac{\partial}{\partial y_2'} \left( \eta \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \right) + \sin \theta \frac{\partial \eta}{\partial y_2'} \right] \ dy_2' = 0 \quad \forall \nu \in Y_{UB}^{l}(D).
\]

(3.32)

Finally, as equation (3.32) holds for any \( \nu \in Y_{UB}^{l}(D) \), we derive

\[
\frac{\partial}{\partial y_2'} \left( \eta \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \right) + \sin \theta \frac{\partial \eta}{\partial y_2'} = 0,
\]

(3.33)

which is the Euler equation for the inner problem. Integrating (3.33) once we obtain

\[
\frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} = \frac{C_{UB}}{\eta} - \sin \theta,
\]

(3.34)

where \( C_{UB} \) is a real constant; now integrating (3.34) from \( y_2' = -\beta \) to \( y_2' = +\beta \), we find

\[
C_{UB} = \frac{(\overline{C}_1 - \overline{C}_2) + 2\beta \sin \theta}{\int_{-\beta}^{\beta} (1/\eta) \ dy_2'}.
\]

(3.35)

Defining the inner-solution nip conductance \( G(\alpha, \beta) \),

\[
G(\alpha, \beta) = \left( 2 \int_{0}^{\beta} \frac{dy_2'}{1 + \alpha - \sqrt{1 - 4y_2'^2}} \right)^{-1},
\]

(3.36)

we obtain from (3.34), (3.35) and (3.36)

\[
\frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \{ \overline{C}_1, \overline{C}_2 \} = \frac{1}{\eta} G(\alpha, \beta) \left[ (\overline{C}_1 - \overline{C}_2) + 2\beta \sin \theta \right] - \sin \theta.
\]

(3.37)

We now fold this into the outer problem.

The outer problem is given by (3.28), with \( \chi_{UB, in}^{l} \) given (implicitly) by (3.37). We first rewrite the functional \( J_D(\chi_{UB, in}^{l} \{ w|_{\Gamma_1}, w|_{\Gamma_2} \}) \) from (2.11), (3.1), (3.2), (3.3), and (3.26),

\[
J_D(\chi_{UB, in}^{l} \{ w|_{\Gamma_1}, w|_{\Gamma_2} \}) = \int_{-\beta}^{\beta} \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \eta \ dy_2' + \sin \theta \int_{-\beta}^{\beta} \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \eta \ dy_2' + \sin \theta \int_{-\beta}^{\beta} \frac{\partial \chi_{UB, in}^{l}}{\partial y_2'} \eta \ dy_2';
\]

(3.38)
integrating-by-parts the first two terms in (3.38) and using (3.33), we obtain

\[
J_D(\chi_{UB,in}^t \{w|\gamma_1, w|\gamma_2\}) = \left[ \chi_{UB,in}^t \frac{\partial \chi_{UB,in}^t}{\partial y_2} \right]_{-\beta}^{\beta} + \left[ \sin \theta \chi_{UB,in}^t \eta \right]_{-\beta}^{\beta} \\
+ \sin \theta \int_{-\beta}^{\beta} \frac{\partial \chi_{UB,in}^t}{\partial y_2} \eta \, dy_2 ;
\]

(3.39)

finally, identifying \(\chi_{UB,in}^t(+\beta) = w|\gamma_1, \chi_{UB,in}^t(-\beta) = w|\gamma_2\), and substituting (3.37), we find

\[
J_D(\chi_{UB,in}^t \{w|\gamma_1, w|\gamma_2\}) = (w|\gamma_1 - w|\gamma_2) G(\alpha, \beta) (w|\gamma_1 - w|\gamma_2) \\
+ 4\beta \sin \theta G(\alpha, \beta) (w|\gamma_1 - w|\gamma_2) + 4\beta^2 \sin^2 \theta G(\alpha, \beta) - \sin^2 \theta V(D),
\]

(3.40)

where

\[
V(D) = \int_{-\beta}^{\beta} \eta \, dy_2
\]

(3.41)

is the volume of the nip region \(D\). Taking the first variation of the sum in (3.28), using (2.11) and (3.40), and recognizing that the last two terms in (3.40) do not contribute to the first variation, we obtain the desired weak form: Find \(\chi_{UB,out}^t \in \overline{Y}_{#,UB}(C)\) such that

\[
\int_C \frac{\partial u}{\partial y_k} \frac{\partial \chi_{UB,out}^t}{\partial y_k} \eta \, dy_k + (v|\gamma_1 - v|\gamma_2) G(\alpha, \beta) (\chi_{UB,out}^t|\gamma_1 - \chi_{UB,out}^t|\gamma_2) \\
= \int_C \frac{\partial u}{\partial y_1} \eta \, dy - 2\beta \sin \theta G(\alpha, \beta) (v|\gamma_1 - v|\gamma_2) \quad \forall v \in \overline{Y}_{#,UB}(C).
\]

(3.42)

In effect, our assumptions as to the form of the inner solution, (3.26), permit us to analytically "statically condense" [Przemieniecki 1963] the nip region \(D\). Note that the variational problems posed above involve slightly non-standard "torsion" boundary conditions, in which the solution is constrained to be, say, constant, over certain boundary segments; these boundary conditions can be shown to lead to well-posed problems [Magen 1988].

The configuration effective conductivity sharper upper bound is then given by

\[
k_{UB}^t = a_N(\chi_{UB}^t) = a_C(\chi_{UB}^t|C) + a_D(\chi_{UB}^t|D),
\]

(3.43)

which, from (2.11), (2.16), (2.29), (3.40), and (3.42), can be reduced to an evaluation solely over \(C\),

\[
k_{UB}^t = (1 - c) - \frac{1}{\lambda^2} \int_C \frac{\partial \chi_{UB,out}^t}{\partial y_1} \eta \, dy + \frac{1}{\lambda^2} 2\beta \sin \theta G(\alpha, \beta) (\chi_{UB,out}^t|\gamma_1 - \chi_{UB,out}^t|\gamma_2) \\
+ \frac{1}{\lambda^2} \sin^2 \theta (4\beta^2 G(\alpha, \beta) - V(D)).
\]

(3.44)

The mathematical proof that \(k < k_{UB}^t < k_{UB}^l\) is analogous to the proof (3.15)–(3.16) for the crude upper bound: we evoke Property \(A\), and identify the function space nesting \(W_{#,UB}(\Omega) \subset Y_{#,UB}(\Omega) \subset H_{#,0}^1(\Omega)\).
3.2.5 Sharp Lower Bound

A sharper lower bound for $k, k_{LB}^y$, can be obtained by introducing a large (in fact, infinite) number of insulating strips, or cuts, in the nip region, oriented perpendicularly to the nip centerline $\overline{y_1 y_2}$ (Figure 3-1d). The insulating cuts allow heat flow in the direction perpendicular to $\overline{y_1 y_2}$ only, and the temperature may, thus, be discontinuous across the cuts. Physically, the introduction of the cuts leads to a sharper lower bound because heat flow is not as severely restricted as in the isotropically insulating nip of the crude lower bound; in effect, the sharper-lower-bound nip region is filled with an anisotropic insulator, in which heat flow is prohibited in only one direction.

![Figure 3-3: Sharper-lower-bound nip-region decomposition.](image)

The geometry of the sharper-lower-bound nip is shown in detail in Figure 3-3. Note that we have further decomposed the nip region as $\mathcal{D} = \mathcal{D}' \cup (\mathcal{D} \setminus \mathcal{D}')$, $\mathcal{D}' = y' \in [-\gamma_L, \gamma_R] \times y_2' \in [-\beta, \beta]$; the two end regions that comprise $\mathcal{D} \setminus \mathcal{D}'$ are filled with an isotropic insulator. In theory, we should choose these end regions to be as small as possible, however mesh generation exigencies sometimes require slightly larger regions (though still small compared to $\mathcal{D}'$). The insulating cuts are introduced in $\mathcal{D}'$, in the direction perpendicular to $\overline{y_1 y_2}$; that is, in $\mathcal{D}'$, the conductivity parallel to $\overline{y_1 y_2}$ is zero.

The sharper lower bound mesoscale problem is then given by

$$\chi_{LB} = \arg \min_{\tilde{J}_{C \cup \mathcal{D}'}(w')} \tilde{\mathcal{J}}_{C \cup \mathcal{D}'}(w'),$$

where

$$\tilde{\mathcal{J}}_{C \cup \mathcal{D}'}(w) = J_C(w|C) + J_{\mathcal{D}'}(w|\mathcal{D}'),$$

and

$$J_{\mathcal{D}'}(w) = \int_{\mathcal{D}'} \frac{\partial w}{\partial y_2} \frac{\partial w}{\partial y_2'} \, dy' + 2 \sin \theta \int_{\mathcal{D}'} \frac{\partial w}{\partial y_2} \, dy'.
$$

Here $Z_{\#, LB}(C \cup \mathcal{D}')$ is the space of all $\lambda$-doubly periodic functions $w(y)$ which satisfy the following conditions: $w$ is continuous over $C \cup \mathcal{D}'$; $w|_C \in H^1(C)$; $w|_{\mathcal{D}'} \in L^2(\mathcal{D}')$. 

34
\[(\partial u/\partial y')_{|D'} \in L^2(D')\] (here \(L^2(D')\) is the space of functions which are square-integrable over \(D')\); \(\int_{CUD'} w\,dy = 0\). The modified functional approach, (3.47), in which only the \(y'\) derivatives are preserved, is based on techniques developed in Magen et al. [1988] for one-dimensional conduction estimates.

Problem (3.45) is now broken into the inner problem

\[\chi_{LB,\text{in}}^{g_+ , g_-} = \arg \min_{w \in Z_{LB}(D')} J_{D'}^i (w), \tag{3.48}\]

and the outer problem

\[\chi_{LB,\text{out}} = \arg \min_{w \in \mathcal{Z}_{\# LB}(C)} [J_C(w) + J_{D'}(\chi_{LB,\text{in}}^i \{w|_{\Gamma_1} , w|_{\Gamma_2}\})], \tag{3.49}\]

where

\[\chi_{LB,\text{out}} = \chi_{LB}^i |c + s''', \quad \chi_{LB,\text{in}}^i \{\chi_{LB,\text{out}}^i |_{\Gamma_1} , \chi_{LB,\text{out}}^i |_{\Gamma_2}\} = \chi_{LB}^i |_{D'} + s'''', \tag{3.50}\]

and \(s'''' \in \mathbb{R}\) is the shift required such that both \(\int_{CUD'} \chi_{LB}^i \,dy = 0\) and \(\int_C \chi_{LB,\text{out}}^i \,dy = 0\).

In (3.48), \(Z_{LB}(D')\) is the set of all functions \(w(y)\) which satisfy the following conditions:

\(w \in L^2(D')\) and \((\partial u/\partial y') \in L^2(D')\); \(w|_{\Gamma_1} = g_+ (y'_1)\), and \(w|_{\Gamma_2} = g_- (y'_1)\), where \(g_+\) and \(g_-\) are given functions of \(y'_1\) defined over the "top" \((y'_2 = +\beta)\) and "bottom" \((y'_2 = -\beta)\) boundaries of the nip, respectively. In (3.49), \(\mathcal{Z}_{\# LB}(C)\) is simply \(H^1_{\# C}(C)\). (Here, \(g_+\) and \(g_-\) must also satisfy certain technical conditions related to the trace of functions in \(H^1_{\# C}(C)\).)

Solution of the inner problem for \(\chi_{LB,\text{in}}^i\) proceeds by taking the first variation of the functional \(J_{D'}(w)\),

\[\int_{D'} \frac{\partial v}{\partial y'_2} \frac{\partial \chi_{LB,\text{in}}^i}{\partial y'_2} \,dy' + \sin \theta \int_{D'} \frac{\partial v}{\partial y'_2} \,dy' = 0 \quad \forall v \in \mathcal{Z}_{LB}(D') , \tag{3.51}\]

where \(\mathcal{Z}_{LB}(D')\) is the space of all functions in \(Z_{LB}(D')\) which vanish on \(\Gamma_1 \cup \Gamma_2\) (as \(\chi_{LB,\text{in}}^i (y'_1, \pm \beta) = g_\pm (y'_1)\) — fixed as regards the inner problem —, \(v(y'_1, \pm \beta) = 0\)). We can then integrate (3.51) by parts in \(y'_2\) (assuming sufficient regularity) to obtain

\[\int_{-\gamma_L}^{\gamma_R} \left\{ v \left( \frac{\partial \chi_{LB,\text{in}}^i}{\partial y'_2} + \sin \theta \right) \right|_{-\beta}^{\beta} - \int_{-\beta}^{\beta} v \frac{\partial}{\partial y'_2} \left( \frac{\partial \chi_{LB,\text{in}}^i}{\partial y'_2} \right) \,dy'_2 \right\} \,dy'_1 = 0 \quad \forall v \in \mathcal{Z}_{LB}(D') . \tag{3.52}\]

We thus deduce the simple Euler equation

\[\frac{\partial}{\partial y'_2} \left( \frac{\partial \chi_{LB,\text{in}}^i}{\partial y'_2} \right) = 0 , \tag{3.53}\]

the solution of which is the constant-in-\(y'_2\) gradient

\[\frac{\partial \chi_{LB,\text{in}}^i}{\partial y'_2} (g_+ , g_-) = \frac{g_+(y'_1) - g_-(y'_1)}{2\beta}, \quad \forall y'_1 \in (-\gamma_L, \gamma_R) . \tag{3.54}\]

We now incorporate this inner solution into the outer problem.

The outer problem is given by (3.49), with \(\chi_{LB,\text{in}}^i\) given (implicitly) by (3.54). We first
rewrite the functional \(J_D'(x_{LB,in}^t (w|\Gamma_1, w|\Gamma_2))\) as

\[
J_D'(x_{LB,in}^t (w_+, w_-)) = \frac{1}{2\beta} \int_{-\gamma_l}^{\gamma_R} [w_+(y'_1) - w_-(y'_1)]^2 dy'_1 + 2 \sin \theta \int_{-\gamma_l}^{\gamma_R} [w_+(y'_1) - w_-(y'_1)] dy'_1, \tag{3.55}
\]

where \(w_+ = w|\Gamma_1, w_- = w|\Gamma_2\). Then, taking the first variation of the functional in (3.49), and using (2.11) and (3.55), we obtain the weak form: Find \(x_{LB,out}^t \in \mathcal{Z}_{\#.LB}(C)\) such that

\[
\int_C \frac{\partial}{\partial y_k} \frac{\partial x_{LB,out}^t}{\partial y_k} dy + \frac{1}{2\beta} \int_{-\gamma_l}^{\gamma_R} [w_+(y'_1) - w_-(y'_1)] [x_{LB,out,+}^t(y'_1) - x_{LB,out,-}^t(y'_1)] dy'_1 = \int_C \frac{\partial v}{\partial y_1} dy - \sin \theta \int_{-\gamma_l}^{\gamma_R} [w_+(y'_1) - w_-(y'_1)] dy'_1 \forall v \in \mathcal{Z}_{\#.LB}(C), \tag{3.56}
\]

where \(x_{LB,out,+}^t = x_{LB,out}^t|\Gamma_1, x_{LB,out,-}^t = x_{LB,out}^t|\Gamma_2\), \(v_+ = v|\Gamma_1\), and \(v_- = v|\Gamma_2\). Equation (3.56) is very similar to the sharper–upper–bound equation (3.42): roughly, \((\gamma_R + \gamma_L)/2\beta\) in the former replaces \(G(\alpha, \beta)\) in the latter.

To express the effective conductivity, we introduce, for \(w \in \mathcal{Z}_{\#.LB}(C \cup \mathcal{D}')\),

\[
\tilde{a}_{CD'}(w) = a_C(w|C) + a_{D'}(w|\mathcal{D}'), \tag{3.57}
\]

where, for \(w' \in \mathcal{Z}_{LB}(\mathcal{D}')\),

\[
a_{D'}(w') = \frac{1}{\lambda^2} \int_{\mathcal{D}'} \frac{\partial}{\partial y_2} (w' - y'_1 \cos \theta + y'_2 \sin \theta) \frac{\partial}{\partial y_2} (w' - y'_1 \cos \theta + y'_2 \sin \theta) dy'. \tag{3.58}
\]

The configuration effective conductivity sharper lower bound is then defined by

\[
k_{LB}^t = \tilde{a}_{CD'}(x_{LB}^t), \tag{3.59}
\]

which, from (3.47), (3.50), (3.55)–(3.58), can be reduced to an evaluation solely over \(C\),

\[
k_{LB}^t = (1 - \tilde{c}) + \frac{\sin^2 \theta}{\lambda^2} 2\beta (\gamma_L + \gamma_R) - \frac{1}{\lambda^2} \int_C \frac{\partial x_{LB,out}^t}{\partial y_1} dy + \frac{\sin \theta}{\lambda^2} \int_{-\gamma_l}^{\gamma_R} [x_{LB,out,+}^t(y'_1) - x_{LB,out,-}^t(y'_1)] dy'_1, \tag{3.60}
\]

where \(\tilde{c}\) is given by (3.21). We now show that (3.59) is a lower bound for \(k\), and a better lower bound than \(k_{LB}^t\).

The mathematical proof that \(k_{LB}^t\) is indeed a lower bound for \(k\) follows from domain embedding: first, we write

\[
k_{LB}^t = \tilde{a}_{CD'}(x_{LB}^t) = \min_{w \in \mathcal{Z}_{\#.LB}(CD')} \tilde{a}_{CD'}(w), \tag{3.61}
\]

which follows from Property A (applied not to \([J_\Psi(w), a_\Psi(\phi)]\) as in (2.18)–(2.19), but, rather, to \([\bar{J}_{CD'}(w), \tilde{a}_{CD'}(x_{LB}^t)]\)); then

\[
k_{LB}^t = \min_{w \in \mathcal{Z}_{\#.LB}(CD')} \tilde{a}_{CD'}(w)
\]

36
\begin{align}
&\leq \bar{a}_{\text{CUD'}}(\chi_{\text{CUD'}} + s^{(iv)}) = \bar{a}_{\text{CUD'}}(\chi_{\text{CUD'}}) \quad (3.62) \\
&\leq \tilde{a}_{\text{CUD'}}(\chi_{\text{CUD'}}) + a_{D\setminus D'}(\chi_{D\setminus D'}) \quad (3.63) \\
&\leq a_{\Omega}(\chi) = k. \quad (3.64)
\end{align}

The inequality (3.62) follows from the fact that \( \chi_{\text{CUD'}} + s^{(iv)} \in Z_{\#, \text{LB}}(C \cup D') \) (\( s^{(iv)} \in \mathcal{R} \) an appropriate shift); the inequality (3.63) follows from the positive–(semi)definiteness of the quadratic form \( a_{D\setminus D'}(w) \) given in (2.16); the inequality (3.64) obtains because the “difference” between \( a \) and \( \tilde{a} \) is positive–(semi)definite. We can also show that the sharper lower bound is indeed sharper than the crude lower bound derived in Section 3.2.3. We write

\[
k_{LB}^k = \min_{w \in X_{\#, \text{LB}}(C)} a_c(w)
\leq a_c(\chi_{LB}^k | c + s^{(v)}) = a_c(\chi_{LB}^k | c) \quad (3.65)
\leq a_c(\chi_{LB}^k | c) + \frac{1}{\lambda^2} \int_{D'} \left[ \frac{\partial}{\partial y'_1}(\chi_{LB}^k - y'_1 \cos \theta + y'_2 \sin \theta) \right]^2 dy' 
\quad (3.66)
= \bar{a}_{\text{CUD'}}(\chi_{LB}^k) = k_{LB}^k, \quad (3.67)
\]

where the inequality (3.65) follows from the fact that \( \chi_{LB}^k | c \) (appropriately shifted by \( s^{(v)} \in \mathcal{R} \)) is in \( X_{\#, \text{LB}}(C) \). Physically, there is “more conducting domain” in the sharper lower bound case than in the crude lower bound case.

### 3.2.6 Extension to Multiple Nips

Equations (3.12), (3.18), (3.42), and (3.56), of Sections 3.2.2, 3.2.3, 3.2.4, and 3.2.5, respectively, consider only one nip. When \( N_{\text{nip}} > 1 \) nips are present, there are, correspondingly, \( N_{\text{nip}} \) inner problems. We currently adopt a particularly simple multiple–nip strategy in which two nip–defining geometrical constraints are imposed: \( \alpha < \alpha_c < 0.20 \) and \( \beta < 0.20 \), where \( \alpha_c \) is a threshold parameter such that a nip is introduced (for which \( \beta < .20 \)) for each pair of inclusions whose centers are separated by less than \( 1 + \alpha_c \). These (and many other possible) geometrical constraints are shown in Appendix A to guarantee that no inclusion will intersect any of the resulting \( N_{\text{nip}} \) nips, and that no two nips will intersect. With these non–intersection conditions satisfied: the nip effects become additive, in that each nip contributes an “independent” term to the outer problem weak form; the single–nip bound proofs extend directly to the multiple–nip case.

The general formulation presented in Sections 3.2.2 – 3.2.5 then allows precise analytical microsolutions to be directly incorporated into the mesoscale solution process in a rigorously quantifiable fashion: the nip regions of close–particle pairs (note \( n \)–tuples are treated pairwise) are eliminated, and the resulting bounds calculated through (3.14) and (3.20) or (3.44) and (3.60) (in practice, the latter). If the bounds are sufficiently sharp (that is, the bound “gap” is on the order of the discretization error), the calculation is concluded; if the bounds are not sufficiently sharp, some or all of the nip regions are reduced in size by decreasing \( \beta \) (or perhaps eliminated by decreasing \( \alpha_{\text{crit}} \)), and the calculation is repeated. In Section 3.4 we present the use of sensitivity derivatives [Newman, Hou, Jones, Taylor & Korivi 1992] to adaptively control nip parameters.

The combined analytico–computational procedure yields significant computational economies, and is both systematic and “convergent” (Chapter 5): although for the suggested crude lower and upper bounds (Sections 3.2.2 and 3.2.3) the bound “gap” may remain order
unity even as $\beta$ tends to zero, for the more sophisticated conductor and insulator choices (Sections 3.2.4 and 3.2.5) the bound gap will decrease to $O(\alpha_{cr\alpha})$ as $\beta \to 0$. In practice, the method performs even better than might be expected, as, on physical grounds, the very-close inclusions — which dominate the computational effort — are often (though not always) not major contributors to transport.

In closing Section 3.2, we make several comments. First, our weak forms have, of course, analogous strong formulations. In the latter, however, we must supplement the space description with the natural boundary conditions automatically imposed by the variational statement. For example, for the sharper upper bound (Section 3.2.4) strong statement we must explicitly require not only the essential boundary conditions $\chi_{U_{B,\text{out}}} | r_1 = \chi_{U_{B,\text{in}}} | r_1 = C_1$, $l = 1, 2$, but also that the integral of the outer flux must match the "lumped" inner flux, $G(\alpha, \beta)(C_1 - C_2)$. Second, we note that the sharper lower bound can be even further improved if we first map the nip region $\mathcal{D}$ into a rectangle and then insert the insulator cuts, thereby eliminating the endcaps $\mathcal{D} \setminus \mathcal{D}'$; the associated complexity is, however, probably not warranted given the good performance of (3.60) (Chapter 5). Finally, we note that the variational–bound nip–element method presented here can also be viewed as a heterogeneous–equation domain decomposition technique (Quarteroni 1991): the physics imposed by the geometry of the nip region permits us to pursue different (simpler) equations in $\mathcal{D}$, (3.11), (3.33), or in $\mathcal{D}'$, (3.53); the variational framework then permits us to quantify the resulting error.

### 3.3 The Porous Media Problem

#### 3.3.1 Viscous Flows

The (Stokes) permeability, $\kappa$, enjoys the variational principle (2.34) which is analogous to the Dirichlet principle (2.17) associated with the effective conductivity, $k$. Therefore, allowing for rigorous construction of nip–region–based lower and upper bounds in an analogous way to the the effective conductivity bounds. The two problems, of course, differ in certain aspects, particularly in the physical motivation of the nip–region models, and, more generally in the numerical treatment. In our development, we pursue only conceptually crude bounds for the permeability, that is, strategies which are analogous to the crude models illustrated in Figures 3-1a and 3-1b. To generate a crude upper bound, we simply enlarge the nip region as shown in Figure 3-4a, thus allowing for a larger "flux" of fluid across the gap than would actually pass. To obtain a crude lower bound we simply replace the nip region with a blockage as shown in Figure 3-4b, preventing flow across the gap. These models will not change the weak variational statement of the original problem as given by equations (2.30) and (2.31), except for replacing the original domain $\Omega$ by the new modified domain $\mathcal{C}$, defined as $\mathcal{C} = \Omega \setminus \mathcal{D}$ for the lower bound, and as $\mathcal{C} = \Omega \cup \mathcal{D}$ for the upper bound, where $\mathcal{D}$ is the excised nip region as defined by Figure 3-5 for the two different nips.

Although, these models seem physically plausible to generate the desired bounds, one could argue that the nip strategies change the total surface area exposed to the fluid and consequently the drag. Furthermore, the blockage strategy will eliminate entire regions of otherwise, relatively high dissipation. While the upper bound strategy would positively reduce the drag, it may not be readily persuasive that the lower bound would still obtain due to the two aforementioned effects, particularly if the location of the nip is not critical to the flow [Johnson 1994]. We shall only present the proof here, and will discuss the objections in the examples presented in Section 5.3.2 of Chapter 5.

38
The proofs for the claimed hierarchy of the bounds are based on variational embedding arguments, in which a maximizing solution on a smaller domain is extended (by zero, due to no slip boundary condition) to construct an admissible but non-maximizing candidate on a larger domain. To translate this statement mathematically let $u_\Omega$ be the correct Stokes solution for the bare (no nip) problem defined over the domain $\Omega$; $u_C$ be the correct Stokes solution for the nip-augmented problem defined over $C$; and $0|_D$ be the zero field extending over the region $D$. For the upper-bound permeability, $\kappa_{UB}$, we can write (equations (2.34 and 2.27)):

\[
\kappa = \frac{1}{\lambda^2} J^P_\Omega(u_\Omega) \\
= \frac{1}{\lambda^2} J^P_\Omega(u_\Omega) + J^P_D(0) \\
= \frac{1}{\lambda^2} J^P_C(u_\Omega \cup 0|_D) \\
\leq \frac{1}{\lambda^2} J^P_C(u_C) \\
= \kappa_{UB} \tag{3.68}
\]

where the last inequality follows because the constructed field $u_\Omega \cup 0|_D$, although admissible, is not the correct Stokes solution over the domain $C$ and thus will not maximize the
functional \( J^P_v \). Similarly for the lower-bound permeability, \( \kappa_{LB} \), we write:

\[
\kappa = \frac{1}{\lambda^2} J^P_\Omega (u_\Omega) \\
\geq \frac{1}{\lambda^2} J^P_\Omega (u_C \cup 0|_D) \\
= \frac{1}{\lambda^2} J^P_C (u_C) + J^P_D (0) \\
= \frac{1}{\lambda^2} J^P_C (u_C) \\
= \kappa_{LB}
\]  

(3.69)

Thus \( \kappa_{LB} < \kappa < \kappa_{UB} \) as claimed.

Although we have considered only the (scalar) isotropic permeability, lower and upper bounds for the components of the anisotropic permeability tensor can be constructed following the development in [Nir et al. 1975]. We also remark that sharper bounds analogous to their conduction counterparts illustrated in Figures 3-1c and 3-1d can be constructed. For example, to obtain a sharper lower bound, we can assume a one dimensional Poiseuille flow in the nip gap region in which the candidate velocity profiles are reduced to parallel parabolic ones. The lower bound will obtain physically because the model implies an infinite viscosity in the direction normal to the flow, and mathematically because of the additional restriction on the functional space in which we search for a solution. The mathematical treatment is, of course, more complicated with the presence of the Lagrange multiplier, the pressure.

The Poiseuille flow assumption, in fact, is very close to reality in the nip regions. Therefore, we expect our crude models to be rather sharp. This is because of the \( O(\alpha^2) \) dependence of the velocity, and hence, the permeability on the interparticle spacing as opposed to the \( O(\alpha) \) dependence of the effective conductivity. This has been observed numerically as we will see in Chapter 5.

### 3.3.2 Inertial Flows

In this section we discuss the implications on the creeping-flow-based nip models when inertia effects are important. As shown in Section 2.4, solution to the Navier–Stokes problem defined by equations (2.47) and (2.48) will not satisfy the variational principle (2.27), and the functional–maximizing property of the permeability, equation (2.34), will therefore not obtain at finite Reynolds numbers. It thus follows that our microscale bounds for the Stokes problem are no longer rigorous for the Navier–Stokes problem. However, we can still apply the Stokes bounds, heuristically, based on: the physical plausibility of the bounds even in the presence of inertia; and the effectively inertia-free flows expected to obtain in the narrow nip regions even when the global Reynolds number may be large. (Note, however, the presence of counterexamples [Karniadakis, Mikić & Patera 1988] ) Sensitivity derivatives, which is discussed in the next section, could be used (e.g., with respect to say, \( \beta \)) to assess the merits of this “naive” approach.
3.4 A Posteriori Analysis Using Sensitivity Derivatives

Sensitivity derivatives are the derivatives of an output function, which involves the solution of partial differential equations (PDEs), with respect to a change in input design variables. Examples of output functions are the configuration effective conductivity and permeability whose determination requires the solution of associated mesoscale PDEs. Examples of input design variables include the nip parameter $\beta$; the location of inclusions. It is important to note that the sensitivity derivatives are meaningful only with respect to a smooth continuous output function. For example, we can not use $\alpha_c$ as input design variable since the effective property is not continuous with respect to $\alpha_c$. (Recall that $\alpha_c$ decides the number of nips and a change in this number would cause an abrupt change in the effective property.)

Sensitivity derivatives can be used in an informal way, or by a formal gradient-based optimization algorithm to search in the design space for an improved design. Sensitivity analysis is necessary when there are many design variables and multidisciplinary output requirements which preclude, for example, parametric studies because of prohibitive cost [Sobieski 1989].

In the following section we present the two different methods for calculating the sensitivity derivative of an output function, namely the direct differentiation method and the adjoint variable method. Then in Section 3.4.2 we illustrate use of the sensitivity derivative to assess the nip element design for the effective conductivity problem, that is, to determine which nips most greatly affect the accuracy of $k$. For clarity and practicality of the presentation we shall found the formulation assuming a discretized form of the continuous mathematical problem.

3.4.1 The Sensitivity Derivative

The presentation here is parallel to that of Newman et al. [1992]. In order to simplify the discussion we will consider a single (scalar) output function and a single design variable. Extension to multiple design variables and general output functions can be carried out systematically and is discussed at the end of the section. Let $Q$ be the solution state vector of the problem of interest (e.g., temperature field) which is defined on a grid $X$. Let $D$ be a single design variable and $F = F(Q, D, X)$ be a required output function (e.g., effective conductivity). It is assumed that $Q$ satisfies a set of (possibly nonlinear) algebraic equations symbolically expressed as

$$R(Q, D, X) = 0.$$  \hspace{1cm} (3.70)

(Note that for our linear effective conductivity problem, $R$ is the discrete equivalent to the first variation of a functional whose maximization is achieved at $Q$.)

The sensitivity derivative, SD, of $F$ with respect to $D$ is defined by

$$\frac{dF}{dD} = \lim_{\delta D \to 0} \frac{F(Q(D + \delta D), D + \delta D, X(D + \delta D)) - F(Q(D), D, X(D))}{\delta D}.$$  \hspace{1cm} (3.71)

Using (3.71) to calculate $\frac{dF}{dD}$ by "brute force" finite differencing is not an economical approach for it requires reappealing to the code with the change in the design variable in place, which requires inordinate computational work, particularly if to be used in an optimization loop with many inputs; in practice, more efficient alternative approaches are used. To wit,
the SD can be equivalently expressed using the total derivative of \( F \) with respect to \( D \)
\[
\frac{dF}{dD} = \frac{\partial F}{\partial D} + \frac{\partial F}{\partial Q} Q' + \frac{\partial F}{\partial X} X',
\]
(3.72)
where the sensitivity functions \( Q' \) and \( X' \) are taken to mean \( \frac{\partial Q}{\partial D} \) and \( \frac{\partial X}{\partial D} \), respectively. Thus in order to compute the SD, one needs to determine the sensitivity functions \( Q' \) and \( X' \). Typically, the dependence of \( F \) on \( X \) is implicit and it is more advantageous to treat the last product in (3.72) as a whole by finite differencing
\[
\frac{\partial F}{\partial X} X' = \frac{F(Q, D + \delta D, X(D + \delta D)) - F(Q, D, X(D))}{\delta D},
\]
(3.73)
or by more sophisticated analytical or symbolic procedures. Note also that equation (3.73) does not require reappealing to the code since \( X \) is independent of \( Q \). On the other hand, any direct approximation of the sensitivity function \( Q' \) share the same difficulties associated with using (3.71) directly which is undesirable. Therefore two complementary methods have been developed to circumvent this difficulty.

The Method of Sensitivity Functions

The method of sensitivity functions works directly with equation (3.72), and is concerned with efficiently obtaining \( Q' \). Following the quasi-analytical approach suggested by Sobieski [1987], \( Q' \) can be computed from the solution of
\[
- \frac{\partial R}{\partial Q} Q' = \frac{\partial R}{\partial D} + \frac{\partial R}{\partial X} X'.
\]
(3.74)
Equation (3.74) follows from differentiating equation (3.70) with respect to \( D \) and recognizing that
\[
\frac{d}{dD} R(Q, D, X) = 0,
\]
(3.75)
since \( R(Q, D, X) \equiv 0 \). The last term in the right-hand-side (RHS) of equation (3.74) can be approximated, for example, by finite differencing
\[
\frac{\partial R}{\partial X} X' = \frac{R(Q(D), D + \delta D, X(D + \delta D)) - \{R(Q(D), D, X(D)) = 0\}}{\delta D}.
\]
(3.76)
In the case of multiple design variables \( D_i \), a new \( Q'_i \) must be solved for, for each design variable. Note that \( Q'_i \) is independent of the number of output functions required which makes this method more appropriate when there are more output functions than design variables (not the case which interests us).

The Method of Adjoint Variable

The method of adjoint variable relies on the idea of constructing an adjoint output function \( F \) whose SD is identical to that of \( F \) but easier to obtain. Following the procedure of Hou et al. [1991], \( F \) is constructed by introducing an adjoint variable, \( A \), as a multiplier of \( R \) and adding it to \( F \)
\[
F(Q, D, X) = F(Q, D, X) + A^T R(Q, D, X),
\]
(3.77)
where the superscript $^T$ denotes the transpose operation. Note that, from equation (3.70), $\mathcal{F} = F$. Taking the total derivative of $\mathcal{F}$ with respect to $D$

$$
\frac{d\mathcal{F}}{dD} = \frac{dF}{dD} + A^T \frac{dR}{dD} + \frac{dA^T}{dD} R.
$$

(3.78)

The last two terms on the RHS of equation (3.78) vanish due to (3.75) and (3.70) and thus $\frac{d\mathcal{F}}{dD} = \frac{dF}{dD}$ as required. Using (3.72), we write

$$
\frac{d\mathcal{F}}{dD} = \frac{\partial F}{\partial D} + \frac{\partial F}{\partial X} X' + A^T \left( \frac{\partial R}{\partial D} + \frac{\partial R}{\partial X} X' \right) + (A^T \frac{\partial R}{\partial Q} + \frac{\partial F}{\partial Q}) Q'.
$$

(3.79)

To eliminate the sensitivity function $Q'$, we define $A$ by the solution of the system

$$
-(\frac{\partial R}{\partial Q})^T A = (\frac{\partial F}{\partial Q})^T.
$$

(3.80)

Using (3.80) in (3.79) we finally obtain

$$
\frac{d\mathcal{F}}{dD} = \frac{\partial F}{\partial D} + \frac{\partial F}{\partial X} X' + A^T \left( \frac{\partial R}{\partial D} + \frac{\partial R}{\partial X} X' \right),
$$

(3.81)

where $\frac{\partial F}{\partial X} X'$ and $\frac{\partial R}{\partial X} X'$ can be evaluated as suggested by (3.73) and (3.76) respectively.

In the case of multiple output functions $F_i$, a new adjoint variable $A_i$ need be defined and solved for each output. Note however, that $A_i$ is independent of the number of design variables, which makes this method more appropriate when the number of design variables exceeds the number of output functions (the case which interests us here).

### 3.4.2 Application to the Effective Conductivity Problem

In this section we apply the adjoint method to calculate the sensitivity derivative of the configuration effective conductivity bound with respect to a change in the parameter $\beta$ defining the nip height. Our choice of the method is dictated by the fact that we have one single output function but typically many design variables (multiple nips). For the illustrative purpose, we shall consider only the simpler case of crude upper bound for which there is no explicit dependence of the output function $k_{UB}$, on the design variable here $\beta$, or the grid $X$.

In order to proceed, we need to introduce the discrete version of the weak form (3.12) and the expression (3.14) for $k_{UB}$ which is identical to that of the bare problem, equation (2.14). In Chapter 4, we show that the discrete vector, $\chi_{UB,\text{part}}$, which we shall denote simply as $\chi_{UB}$, is defined by the solution of the linear system

$$
A(X) \chi_{UB} = G(X),
$$

(3.82)

where the matrix $A$ is the discrete Laplacian operator and the vector $G$ is the discrete equivalent to the RHS of (3.12). The numerical expression for $k_{UB}$ takes the form

$$
k_{UB} = (1 - c) - \frac{1}{\lambda^2} \chi_{UB}^T A \chi_{UB}.
$$

(3.83)
Identifying

\[ Q \rightarrow \chi_{UB} \]
\[ R(Q) \rightarrow A\chi_{UB} - G \]
\[ F(Q) \rightarrow k_{UB} \]
\[ D \rightarrow \beta , \]

we find

\[ \frac{\partial R}{\partial Q} = A \]
\[ \frac{\partial F}{\partial Q} = -\frac{2}{\lambda^2} A\chi_{UB} , \]

and hence, the adjoint variable \( A \) follows from the following Poisson problem

\[ AA = \frac{-2}{\lambda^2} A\chi_{UB} , \]

where we have used the fact that \( A^T = A \). The finite difference approximations (3.73) and (3.76) then take, respectively, the forms

\[ \frac{\partial F}{\partial X} X' = \frac{-1}{\lambda^2} \frac{\chi_{UB}^T A(X(\beta + \delta \beta))\chi_{UB} - \chi_{UB}^T A(X(\beta))\chi_{UB}}{\delta \beta} \]

and

\[ \frac{\partial R}{\partial X} X' = \frac{A(X(\beta + \delta \beta))\chi_{UB} - G(X(\beta + \delta \beta))}{\delta \beta} . \]

Finally noting that \( \frac{\partial F}{\partial D} \) and \( \frac{\partial R}{\partial D} \) are zero, equation (3.81) can be readily used to calculate the sensitivity derivative of \( k_{UB} \) with respect to a change in \( \beta \) for a given nip element. The numerical choice of \( \delta \beta \) and its effect on the SD is discussed in the test case presented in Section 5.5.1 of Chapter 5.

The algorithm for computing the sensitivity derivatives of \( k_{UB} \) in the case of multiple nips is outlined below:

1. Given initial states \( X^0, \beta^0, A^0, G^0 \) and \( k_{UB} \).
2. Compute \( A \) from equation (3.86) and save.
3. Loop over present nips
   1. Modify \( X^0 \rightarrow X(\beta + \delta \beta) \) by displacing the nip edge nodes by \( \delta \beta \) as shown in Figure 3-6.
   2. Modify the operator \( A^0 \rightarrow A(X(\beta + \delta \beta)) \) and the vector \( G^0 \rightarrow G(X(\beta + \delta \beta)) \).
   3. Compute \( \frac{\partial F}{\partial X} X' \) and \( \frac{\partial R}{\partial X} X' \) from equations (3.87) and (3.88) respectively.
   4. Use equation (3.81) to calculate the SD of \( k_{UB} \) with respect to the current nip element.
5. Retrieve \( X^0, A^0 \) and \( G^0 \).
End

The information is then used to resize or eliminate nips to improve the approximation.
Figure 3-6: Changing the design variable $\beta$. 
Chapter 4

Parallel Numerical Methods
(Mesoscale)

In this Chapter we describe the numerical procedure for the efficient parallel solution of the mathematical formulation(s) of the mesoscale problem(s). The algorithmic components of this procedure encompass four main tasks: domain partitioning; mesh generation; finite element discretization and solution; parallel implementation. These tasks are discussed in detail, respectively, in Sections 4.1 – 4.4.

4.1 Domain Decomposition Procedure

The task of the decomposition procedure is to break the original supercell domain into a specific number of smaller non-overlapping subdomains, which we shall refer to as subcells, for distribution on several processors. Typically, the number of subcells is the number of processors, \( P \). For reasons related to parallel performance (discussed later), it is more advantageous however, to divide up the supercell into \( M \times P \) subcells, that is, each processor \( p \) will have \( M \) smaller subcells instead of one larger subcell. We shall refer to the \( M \) smaller subcells as one collection (if \( M = 1 \) then one subcell is simply a collection). The multiplicative factor \( M \) (typically 4) is introduced to permit subsequent reshuffling of subcells among processors for better and dynamic load balancing.

In this work we employ a new tensor–product domain partitioning strategy, rather than the Voronoi domain partitioning technique pursued in Cruz & Patera [1993], for reasons discussed in Section 4.1.4. In the following subsections we describe the underlying steps, and the various algorithmic components of the new strategy required to arrive at a subcell–level geometrical description such as the one shown in Figure 4-1. It is important to note that the decomposition procedure is performed in serial.

4.1.1 Grid Adjustment

The starting point in the tensor–product partitioning procedure is to break the supercell into \( M_x P_x \times M_y P_y \) structured grid lines, \( (P_x P_y = P, M_x M_y = M) \), as shown in Figure 4-2. (The subcells are identified with the bricks formed by the grid lines.) For a given \( P \), \( P_x \) and \( P_y \) are chosen to produce the best (most square) subcells. \( M \) is typically set to 4 with \( M_x = M_y = 2 \). On the iPSC/860 Hypercube the number of processors is always of the form \( 2^\ell \), where \( \ell \), the cube dimension, is an integer (typically, 4 or 5). Table 4.1 lists the best
choices of $P_x$ and $P_y$ assuming the i860 environment. The choice of $P$ and $M$ is restricted by the problem size, namely the number of cylinders. A rule of thumb is that the number of subcells be at most on the order of the number of cylinders. A more rigorous condition will be provided shortly.

The initial grid-based partitioning, can potentially pose a number of degeneracies for the following stages of geometry description and more importantly, boundary meshing. These problems arise since the grid lines can lie arbitrary close to other objects in the supercell like circles and/or nip edges, which can cause numerical instabilities at severe cases or at "best" result in finite elements with very acute angles. To alleviate these problems we reposition the grid lines within a specified moving range around the initial location such that to minimize the degeneracies. In order to implement this idea, we need to address two issues: a measure for the goodness of grid line position; conditions on the line moving range. These two issues are discussed below.

By inspection of Figure 4-3 we can distinguish between two types of interactions: grid line–cylinder interaction; and, grid line–nip edge interaction. The first type can be in the form of a grid line passing close to a cylinder surface or a grid vertex lying close to a cylinder surface. Ideally in such cases, the distance, $a$, defined as the closest distance between the
Figure 4-2: Grid lines for the case $P = 4$, $M = 4$ at a concentration of 0.5 and 50 inclusions.

Figure 4-3: Summary of line-object interactions.
grid line or grid vertex and the cylinder should be at least on the order of the boundary mesh spacing, \( h \), at the closest point on the cylinder (see Figure 4-4). The second type involves the interaction between two line segments. Ideally, the two segments should intersect at a moderate angle and well conditioned proportions or be sufficiently clear from each other. Given that a uniform mesh spacing, \( h \), is defined along the nip edge and that nip edge length is typically a few mesh spacings, it suffices, for acceptable interaction, to ensure a distance, \( a \), between the grid line and the end points of the nip edge on the order of \( h \) (see Figure 4-4). It thus follows, that an appropriate measure of goodness of grid-line/object interaction can be quantified by the value \( a/h \), for properly defined \( a \) and \( h \).

![Figure 4-4: Details of line-object interactions.](image)

Next we consider the line moving range. In order to avoid scenario b) of Figure 4-5 in which a subcell is entirely inside a cylinder as a result of lines shifting, a minimal spacing of \( 2(d + h_{\text{nom}}) \) is required between the original grid lines. Here \( h_{\text{nom}} \) is a nominal mesh spacing. (Note that this restricts the possible choices of \( P \) and \( M \) according to the supercell size \( \lambda \).) Each grid line is assigned a moving range of \( \pm d/2 \) as shown in part a) of Figure 4-5. If two neighboring grid lines are displaced by \( +d/2 \) and \( -d/2 \) respectively, then a subcell width of \( d + h_{\text{nom}} \) will exist at no risk of running into scenario b) of Figure 4-5.

The grid-line adjustment process then proceeds as follow:

- Identify all close cylinders interacting with the line moving range.
- For the current line position sum up all the values, \( a/h < 1 \), and normalize by the number of the interactions. Record this value as ErrSum.
- If ErrSum is zero stop, else increment the line position and repeat.
- If no zero value of ErrSum was recorded in the line moving range, reposition the line at the location with minimum ErrSum value.
- Stop.

### 4.1.2 Geometry Description

The next step following the layout and adjustment of the initial grid lines is to describe the boundary of the individual subcells, which is necessary for the following stage of boundary meshing. More precisely, we need to describe the geometry and connectivity of the individual components making up the boundary of the subcell. The subcell boundary will be composed
of some or all of the entities shown in Figure 4-6 namely: straight line segments, originating from grid lines or nip edges; and circular arc segments. Figure 4-6 also shows the information required for each entity to describe the geometry and connectivity. For the connectivity

Figure 4-6: Geometrical entities and the relevant information. The subscripts \( s \) and \( e \) denote start and end respectively. The integer \( n \) is an identifier designated to the point.

purpose we designate to the starting and ending point of each entity, save circles, a number, which will be unique at each shared point. This numbers will serve as a pointer identifying the preceding or following entity as illustrated in Figure 4-7. For the geometrical purpose we keep track of the coordinates of the defining points for line segments, and of the starting and ending angles for arc segments in addition to the center location.

The algorithm for extracting the boundary of the individual subcells, that is, describing the geometry and connectivity of the boundary components as explained above is divided
Figure 4-7: Geometry description for the case $P = 1$, $M = 1$ at a concentration of 0.5 and 3 inclusions.

into two tasks: extracting the boundary of the bare (no nip) subcells; modifying the bare description to account for the nip edges. The latter task is described in the next section. The former task proceeds by first identifying the cylinders that are entirely or partially in the grid brick identifying the subcell in question. Next the intersection point(s) of the circles and the brick sides are recorded and sorted. The data structures of the individual resulting components are then filled and their connectivity is determined.

Figure 4-8: An example of a subcell with disconnected domains.

Generally this tensor–product based partition will not yield a connected domain. Disconnected domains are likely to be generated because of nip edges (see Figure 4-7) or at particular distributions such as the one shown in Figure 4-8. This fact will only press more demands in the meshing process but will not affect the subcell integrity nor the structured
topology associated with the bricks. One feature of this strategy is the elimination of grid vertices lying inside cylinders. Moreover, for certain configurations, a complete subcell side may be eliminated if it passes through nips and cylinders. Elimination of vertices and possibly edges has a significant impact on reducing the communication needs in the parallel computing environment and hence boosting the performance.

4.1.3 Nip Generation

In this section we outline the algorithm designed for augmenting the bare subcell geometry with nip elements. The nip depicted in Figure 3-2 of Section 3.2.1 represents a special case when the forming inclusions lie entirely in the subcell and are not part of a cluster. Generally an inclusion may accommodate as many as six nips, and a nip edge can be broken into as many as 3 segments, each residing in a different subcell (see Figure 4-9). However, each nip will be associated with one and only one periodic pair of inclusions.

The tensor–product–based partition poses some complexity for the construction of nips since there is no systematic pattern for the intersection of the grid lines and nip elements. This makes it a tedious if not a formidable task to augment the subcells with nips on a case by case basis. Instead a unified strategy is required independent of any particular inclusions distribution. This strategy is outlined below.

Figure 4-9: Example of a nip edge split into three segments each residing in a different subcell.

1. Define $\alpha_c$ and $\beta$.

2. Collect the topological information. This include constructing lists for each inclusion indicating the number of nips it forms, the global number of each nip, and the partner inclusion.

3. Modify the subcell geometry. This is split into three tasks: modifying/eliminating arcs; modifying subcell sides; adding nip edges.

4. Identify the new connectivity information.
5. Identify the present disconnected domains making up the subcell geometry. This is accomplished using the spanning tree algorithm and the line–intersection–insideness test [Preparata & Shamos 1985].

The essence of the algorithm is to decouple the tasks as far as possible dealing with each subcell independently from the rest. The robustness of the algorithm is demonstrated in the random example of Figure 4-10.

Figure 4-10: Example showing nips for the case $P = 4$, $M = 4$ at a concentration of 0.5 and 100 inclusions; 97 nips are present.

4.1.4 Tensor–Product Versus Voronoi–Based Partition

In the section we discuss the advantages and drawbacks of the new tensor–product strategy adopted in this thesis in comparison to the Voronoi based strategy employed by Cruz & Patera [1993]. Figures 4-11 and 4-12 respectively, show the two different decompositions for an identical realization of 24 inclusions at a concentration of 0.3. There are two main advantages of the tensor product partition strategy over the Voronoi–based decomposition strategy. The first one is related to the resulting load balance which is discussed in detail in Section 4.4.2. Here we simply note that the tensor product strategy partitions the domain into precisely the target number of processors whereas the Voronoi–based strategy decomposes the domain into the number of inclusions which is typically not a multiple of the number of processors (limited to $2^\ell$ on the iPSC/860 hypercube where $\ell$ is the cube dimension). Thus in the latter, some processors will have to bear the burden of multiple collections which negatively affects the load balance. In the example of Figure 4-12, using
Figure 4-11: Tensor-product partition for a sample of 24 inclusions at a concentration of 0.3 into 16 collections.

Figure 4-12: Voronoi partition for the sample of Figure 4-11.
16 processors, leads to 8 processors with two collections. The second main advantage of the tensor product strategy is clearly in the fewer number of edges and vertices, and, hence, fewer communication requirements. For a given number of collections the tensor product partition, yields, in fact, the minimum possible number of edges and vertices. This number is even further reduced due to elimination of vertices lying inside cylinders and perhaps complete edges in certain configurations. Note that, for purpose of communication, an edge is a common boundary between two subcells. The details of the boundary are irrelevant for the definition. In Figure 4-11 we count (respecting periodicity) 20 global edges and 11 global vertices. (Note that 7 vertices have been eliminated.) The Voronoi partition, on the other hand, yields 88 edges and 45 vertices.

On the other hand, the tensor product strategy is certainly more complex to implement; requires a nontrivial algorithm for robust nip generation; demands complex data structures; and is more difficult to mesh compared to the regular connected polygonal domains of the Voronoi. We remark, however, that the significant benefits associated with the dramatic reduction in the communication warrant the complexity of the new approach.

4.2 Mesh Generation

One of the primary tasks of finite element analysis is the subdivision of the physical domain into a finite number of non-overlapping subdomains, or simply elements, a process referred to as mesh generation. For two dimensional domains, the ones being considered here, finite elements are typically either quadrangles or triangles. The latter type—the simplex element in 2-D—provides greater geometric flexibility and, hence, is more appropriate for complex domains; requires fewer number of nodes for a given interpolant degree order and hence less elemental operations. Furthermore, general robust triangulation codes are more widely available. On the other hand, triangles do not permit tensor-product discretization techniques, which are essential for spectral element methods [Maday & Patera 1989]. To the end user, most triangulation codes, require only the domain boundary mesh as an input. Triangular algorithms employ one of the two strategies: advancing-front technique; or generalized-Voronoi technique.

Schemes belonging to the first strategy [Mavriplis 1992, D-Rakhshandeh 1990, Sadek 1980] are centered around working inward from the boundaries, inserting new nodes in such a way to optimize the created elements with respect to size and shape, extracting out boundary layers and shrinking the domain until three nodes are left. One of the features of such methods is the automatic placement of interior nodes, which generally results in high-quality elements with smooth transition in size and density throughout most of the domain due to the optimum positioning of the nodes. Other features include: the guaranteed integrity of the boundary, since the boundary mesh constitutes the initial condition; the good robustness since elements are created on a one-by-one basis; and the simplicity of use since the user generally needs only provide the discretized boundary which makes the scheme suitable for complete automation. The disadvantages of the schemes are mainly related to their efficiency as they require expensive searching operations and intersection checking during the construction process.

The generalized-Voronoi techniques differs fundamentally in that they lay a background grid or set of nodes throughout the domain prior to triangulation. Schemes in this category [Hecht & Saltel 1990, Fukuda & Suara 1972, Cavendish 1974, Lo 1985] consists mainly of three phases: node placement; linking; smoothing. The node placement phase can be
accomplished by random generation from a prescribed density function, or by systematic positioning using regular grids. In the linking phase either a Delaunay triangulation is constructed or a searching algorithm is designed with some optimal conditions. The Delaunay triangulation, although more expensive, produces triangles that are as close to equilateral as possible as proven by Sibson [1978] and, hence, is optimal in that sense. The last smoothing phase is required only for non-Delaunay based linking. One algorithm suggested by Cavendish [1974] is to move each node to the centroid of the polygon formed by the triangles which meet at this node. One difficulty associated with the Voronoi methods is preserving the boundary integrity, which is typically remedied by increasing the boundary point resolution or by post-triangulation operations to recover the boundary.

In Appendix B we present an algorithm based on constructing the Voronoi diagram. Our algorithm preserves boundary integrity without the need to increase boundary mesh resolution nor postprocessing operations. Along with the algorithm, we provide a few illustrative examples. This algorithm, however, was not used in our actual calculations. Instead we employed the same program used by Cruz & Patera [1993] namely, MSHPTG - developed in INRIA, France, by Hecht & Saltel [1990] - with its demonstrated robustness, satisfactory results, and ease of use. MSHPTG is a Voronoi-based algorithm and can be fully automated given only the available boundary description, provided from the preceding stage.

The main problem is then reduced to how to discretize the boundary curves. The boundary resolution which is propagated smoothly to the interior, must therefore respect regions of potentially high gradients if discretization errors are not to dominate or even destroy the numerical solution. These high gradients are expected to occur at regions of clustered inclusions. The procedure used for gridding the boundaries makes use of the minimal distance as a guide in the node distribution process. The minimal distance between a point \( y \) and a surface \( S \), denoted as \( d_H(y, S) \) is defined by

\[
d_H(y, S) = \min_{x \in S} |y - x|,
\]

where \( |y - x| \) is the usual Euclidean distance between two points. The supercell boundary curves are composed of arc segments, nip edges, and side segments. In the following we will describe how the minimal distance is used in the meshing procedure of these components starting with former.

**Arc Segments**

Meshing of the arc segments is carried out first, adapting the procedure used by Cruz & Patera [1993]. More precisely, to determine the mesh spacing, \( h \), at a given point \( y \), on a particular arc segment, the following function [Cruz & Patera 1993] is used

\[
h(y) = \left[ \frac{1}{m} \left( \frac{1}{d_H(y, S)} + \frac{1}{h_{nom}} \right) \right] \cdot r^{-1},
\]

where \( S \) is defined as the nip-free supercell boundary \( \partial \Omega \) excluding the arc in question. That is the perimeters of all the inclusions in the supercell except the one being gridded. \( h_{nom} \) is a nominal mesh spacing parameter which is defaulted to, for sufficiently large \( d_H \); \( m \) is a refinement parameter that control the near–cylinder density; and \( r \) is a global control refinement parameter typically set to unity. In effect equation (4.2) honors the relative
closeness of neighboring inclusions when distributing the nodes along the inclusion surface. The resolution will be denser in clustered regions and lighter in well-conditioned regions. The algorithm for meshing a given arc segment then takes the following structure:

Starting from \( y_1 \) at the beginning of a given arc segment

1. Identify the surrounding inclusions to the arc in question. This is done by sorting the inclusions according to center-to-center distance with the arc.

2. Search for a point \( x \) on the inclusions surfaces that is closest to the point \( y_j \) and compute \( d_H \) using (4.1). This task may appear to be expensive, but thanks to the circular inclusions shape, \( d_H \) is achieved only if \( x \) lies on the segment connecting \( y_j \) and an inclusion center. This reduces the search to, at worst, the number of inclusions less one. Note that \( x \) need not be on the closest inclusion to the arc (see Figure 4-13).

![Figure 4-13: The minimal distance \( d_H(y_j, S) \) is achieved by the point \( x \) on the circle with center \( y_3 \).](image)

3. Place a new node, \( y_{j+1} \), at a distance \( dh/2 \) on the arc moving in a pre-defined direction, where \( h \) is computed using (4.2).

4. Repeat until the arc it fully traced. Here the last computed node position will not coincide with the starting point (if a complete circle) or the ending point (if an arc segment). This node is then forced into the right position making the appropriate shift of the remaining nodes.

Note that the number of nodes on the arc segments are not known a priori. In fact, it can vary by many order of magnitudes on different arcs in the same supercell. Dynamic memory allocation proves useful in such situations.

### Nip Edges

Meshing of nip edges which are part of \( \partial \Omega \) can be easily carried out once the arcs have been meshed. The strategy we adopt here is to use a uniform mesh spacing along the nip edge. The value of the mesh spacing is dictated by the average value of mesh spacings at the connecting nodes to the two forming inclusions. In situations where the nip edge is
broken into smaller segments due to partitioning, it may not be possible to use uniform spacing. Here we simply reduce \( h \) in accordance with shortest segment length present. It is important to mention also, for the case of sharper lower bound, one must honor the regions of isotropic insulator (see Figure 3-3) during the node placement. This is done by ensuring the existence of few symmetric insulator-region—defining nodes at the corresponding locations. The interior nodes need not be symmetric although symmetry would simplify the numerical treatment, it is, unfortunately, difficult to enforce due to broken nip edges.

**Side Segments**

Side segments define the boundary of the periodic supercell and interior partitions. These lines are not part of \( \partial \Omega \). In the case of the Voronoi–based partition [Cruz & Patera 1993], these lines possess symmetry features facilitating their meshing treatment. In our partitioning, these lines possess no symmetry features. They cut arbitrarily through the domain passing in some instances appreciably close to inclusion surfaces, and creating gaps with no physical significance. Our line-shifting remedy during the partition process minimizes these effects as possible. The algorithm we use for meshing these line segments employs the same function (4.2) in the following way: to find the mesh spacing at a given point \( y \) on a line segment, we determine the two smallest minimal distances, \( d_{H}^{1} \) and \( d_{H}^{2} \), between the point \( y \) and the inclusions surfaces. Based on \( d_{H}^{1} \) and \( d_{H}^{2} \), we then compute two different mesh spacings \( h_{1} \) and \( h_{2} \) respectively. The mesh spacing \( h \) at the point \( y \) is then defined by a weighted average of the two spacings \( h_{1} \) and \( h_{2} \)

\[
h(y) = \frac{h_{1}}{h_{1} + h_{2}} h_{1} + \frac{h_{2}}{h_{1} + h_{2}} h_{2},
\]

(4.3)

Figure 4-14 illustrates this procedure. In effect we try to define the mesh density along the line segment in harmony with the natural transition in the mesh density in the region between the two affected inclusions. Again here, the number of nodes is not known a priori.

We conclude this section by presenting two important remarks along with an illustrative example. First, concerning the parallel implementation, since subcells are meshed independently in parallel, it is important to ensure that periodic and geometrically coincident side segments possess identical number and distribution of nodes. This is required to avoid special mapping techniques in the imposition of periodic boundary conditions and the direct stiffness procedure (discussed in Section 4.4). In theory, this requirement should obtain by simply meshing the segments in the same direction on different processors, but due to round off errors this is not warranted. Therefore, an initial meshing of the boundary edges is performed in serial to determine the number of nodes a priori to the actual meshing phase. Figure 4-15 shows the boundary mesh for a realization of 17 inclusions at 0.55 concentration partitioned into 4 subcells. The actual mesh produced in parallel is shown in Figure 4-16.

Second, it is important to note that the mesh generation code provides a linear triangulation (1\textsuperscript{st}-order elements) to an approximate numerical domain \( \tilde{\Omega} \) and not the actual domain \( \Omega \). This is because linear elements can not represent the curved boundary of the inclusions. The difference between \( \Omega \) and \( \tilde{\Omega} \) is that the circular inclusions in the former are now composed of rectilinear segments in the latter. This error due to subparametric representation of the boundary is known as "skin effect". To eliminate this error, we used a 2\textsuperscript{nd}-order isoparametric representation of the boundary. This is achieved by mapping the linear mesh to a quadratic mesh and forcing the 2\textsuperscript{nd}-order nodes to match the circular inclusion boundaries (see Figure C-3 in Appendix C).
4.3 Finite Element Treatment

This section is dedicated to the Galerkin finite element treatment of the conduction and porous media problems. The finite element approach is the natural choice given our variational formulation. Furthermore, it provides optimal order accuracy for the energy–norm effective properties of interest [Strang & Fix 1973, Cruz & Patera 1993], and is more easily implemented for irregular and complex geometries than finite–difference–based approaches. There are two basic ingredients in the finite element analysis: discretization of the weak variational form; solution of the resulting discrete system of equations. For the first task we use isoparametric 2nd–order triangular finite elements for better geometry resolution and flexibility. For the second task, we adopt iterative solution algorithms since they require both dramatically less memory and significantly fewer operations than direct methods, and they are more readily and efficiently parallelized.

In the following subsections we deal with the three problems considered in this thesis: the conduction; Stokes; and Navier–Stokes problems respectively. Both of the aforementioned tasks are described including the treatment of the non-standard “Robin” boundary conditions associated with the nip–element method. We note that, in order to simplify the discussion, parallel implications associated with the iterative solution algorithms are deferred until Section 4.4.
Figure 4-15: Boundary mesh for a realization of 17 inclusions at 0.55 concentration, partitioned into 4 subcells. 11 nips are present.

Figure 4-16: The Finite element mesh generated from the boundary grid shown in Figure 4-15. The mesh which is composed of 4243 triangular elements is generated in parallel.
4.3.1 The Conduction Problem

Discretization

We consider first the bare mesoscale problem governed by the weak variational form (2.13) expressed here in the form

\[ a(\chi, v) = \ell(v) \quad \forall v \in H_{\#,0}^1(\Omega). \]  

(4.4)

The self-adjoint (symmetric), positive−definite, bilinear operator \( a(\cdot, \cdot) \) is also present in the more complex saddle problem of Stokes equations and the general Navier−Stokes equations. Discretization of (4.4) proceeds by projecting the infinite−dimensional space \( H_{\#,0}^1(\Omega) \) into a finite−dimensional subspace \( X_h \subset H_{\#,0}^1(\Omega) \) defined by

\[ X_h = \{ v|_{\Omega^k} \in P_n(\Omega^k) \cap H_{\#,0}^1(\Omega) \}, \]  

(4.5)

where \( P_n(\Omega^k) \) is the space of all polynomials of total degree less than or equal to \( n \), defined over triangular element \( \Omega^k \). Here it is assumed that the mesh generation phase subdivides \( \Omega \) into \( K \) non-overlapping isoparametric elements, \( \Omega \approx \bigcup_{k=1}^{K} \Omega^k \), defined by \( N_g \) total global grid nodes. The discrete statement corresponding to (4.5) then takes the form: Find \( \chi_h \in X_h \) such that

\[ a(\chi_h, v) = \ell(v) \quad \forall v \in X_h(\Omega). \]  

(4.6)

A set of \( n^{th} \)−order Lagrangian interpolants (known as basis functions) \( \phi_i(\Omega) \) \( i = 1, \ldots, N_g \) is introduced such that

\[ \phi_i(y_j) = \delta_{ij}, j = 1, \ldots, N_g. \]  

(4.7)

This global set which spans the space \( X_h, X_h = \text{span}(\phi_i i = 1, \ldots, N_g) \) is never formed in practice; instead elemental interpolants (shape functions) are defined and constructed locally on the elemental level based on a prototype reference element and mapping functions. The shape functions represent the traces of the global interpolants restricted to the elemental domains. Direct stiffness summation procedures allow for the recovery of the global basis. For the sake of simplicity however, we will continue the discussion with the global basis functions. The Galerkin method expands the solution \( \chi_h \) and the test function \( v \) using the same basis functions

\[ \chi_h(y) = \sum_{j=1}^{N_g} \chi_{hj} \phi_j(y), \]  

(4.8)

\[ v(y) = \sum_{i=1}^{N_g} v_i \phi_i(y). \]  

(4.9)

Substitution of equations (4.8) and (4.9) into (4.6) leads to

\[ v_i [A_{ij} \chi_{hj} = F_i], \]  

(4.10)

where \( A_{ij} \) and \( F_i \) are defined respectively by the integrals

\[ A_{ij} = \int_{\Omega} \frac{\partial \phi_i}{\partial y_k} \frac{\partial \phi_j}{\partial y_k} \, dy, \]  

(4.11)
and
\[
F_i = \int_\Omega \frac{\partial \phi_i}{\partial y_1} \, dy .
\] (4.12)

Defining the coefficients \( v_i = \delta_{i\ell} \) we finally arrive at the discrete system of equations
\[
A \, \chi_h = F .
\] (4.13)

where \( A = A_{ij} = a(\phi_i, \phi_j), \) \( i, j = 1, ..., N_p \) is the system stiffness matrix (discrete Laplacian operator), and \( F = F_i = \ell(\phi_i), \) \( i = 1, ..., N_p \) is the system forcing vector.

The evaluation of the quadratures (4.11) and (4.12) are performed using the standard finite element procedures by summing elemental contributions. In Appendix C we review the triangular finite elements and present a detailed description of the isoparametric evaluation of the elemental Laplacian. It should be pointed out that \( A \), as constructed, does not incorporate the associated boundary conditions of \( \lambda \)-periodicity and zero average of the vector \( \chi_h \). In the context of iterative solver, these boundary conditions are imposed as part of the evaluation procedure.

**Numerical Effective Conductivity, \( k_h \)**

The discrete equivalent of the energy–norm of the effective conductivity, equation (2.17), takes the form
\[
k_h = a_\Omega(\chi_h) = 1 - c - \frac{1}{\lambda^2} a(\chi_h, \chi_h) = 1 - c - \frac{1}{\lambda^2} \chi_h^T A \chi_h,
\] (4.14)

where \( ^T \) denotes the transpose operation. Due to the functional dependence on \( A \), \( k_h \) inherits the convergence properties associated with the bilinear operator [Strang & Fix 1973]:
\[
a(\chi - \chi_h, \chi - \chi_h) \leq C \, h^{n+1} \| \chi \|_{H^{n+1}}^2
\] (4.15)

where \( C \) is a constant independent of \( h \), and \( \| \cdot \|_{H^{n+1}} \) is the square of the \( H^{n+1} \) energy norm. Given the smoothness of the solution \( \chi \) and recognizing that \( a(\chi - \chi_h, \chi - \chi_h) \) is the \( H^1 \)-seminorm of \( k_h - k \) [Cruz 1993], we deduce that the error measured in the \( H^1 \)-seminorm in the numerical effective conductivity must decrease proportional to \( h^{n+1} \). We thus expect a 3\(^{rd}\)-order convergence with 2\(^{nd}\)-order finite elements. It is important to note however, that skin effects (which are committed when curved boundaries are represented by linear segments) due to use of subparametric finite elements will hinder the the predicted rate of convergence [Strang and Fix 1973]. We shall demonstrate this effect in Chapter 5.

**Conjugate Gradient Iterative Solution**

The best known iterative algorithm for the elliptic system (4.13) is the conjugate gradient method. The conjugate gradient method, in fact, constitutes the backbone of the iterative algorithms for the more complex problems of Stokes and Navier–Stokes problems. It derives its name from the fact that it generates a sequence of conjugate vectors – the residuals of the iterate – which are also the gradients of a quadratic functional, the minimization of which is achieved by solving the linear system. It requires only a limited number of storage which makes it superior in 3D, and is extremely efficient when the coefficient matrix, \( A \), is very well conditioned. The conjugate gradient method is well–documented (see, for example, Golub and Van Loan [1983], Strang [1986]) and we shall not elaborate any further on the
theory behind it.

The preconditioned conjugate gradient iteration takes the form

\[
\begin{align*}
\text{initialization:} & \quad m = 0, r^0 = F - A \chi^0_h, z^0 = M^{-1} r^0, w^0 = z^0 \\
\alpha^m = (z^m, r^m)/(w^m, A w^m) \\
\chi^m_{h+1} = \chi^m_h + \alpha^m w^m \\
r^{m+1} = r^m - \alpha^m A w^m \\
z^{m+1} = M^{-1} r^{m+1} \\
\beta^m = (z^{m+1}, r^{m+1})/(z^m, r^m) \\
w^{m+1} = r^{m+1} + \beta^m w^m
\end{align*}
\tag{4.16}
\]

here \( m \) is the iteration number, \( r \) is the residual vector, \( z \) and \( M \) are intermediate vector and conditioning matrix respectively, \( w \) is the search direction vector, \((\cdot, \cdot)\) denotes the inner product operation, and \( \alpha \) and \( \beta \) are scalar quantities. In the absence of round-off error the conjugate gradient iteration will converge to the exact solution in at most \( N_{\text{dof}} \) iterations, where \( N_{\text{dof}} \) is the number of global degrees of freedom. (We note here that although we have presented the preconditioned version of the conjugate gradient iteration, we have not used any preconditioners in the actual implementation and hence, \( M = I \) the identity matrix.)

In practice, the iterations are terminated based on a stopping criterion and an incomplete iteration error is tolerated. Normally, this error is on the order of the discretization error. In Cruz and Patera [1983], a stopping criterion is derived based on estimation of the minimum eigenvalue of the matrix \( A \) and the dependence of \( k_h \) on \( A \) to directly control the error on the numerical effective conductivity. The final expression takes the form

\[
\left\{ \sum_{i=1}^{N_{\text{dof}}} \frac{(r^m_i)^2}{B_i} \right\}^{\frac{1}{2}} < \frac{\pi}{\lambda} R_m \left( \sqrt{1 + \epsilon_{\text{tol}}} - 1 \right).
\tag{4.17}
\]

where \( R_m \) is known quantity given by

\[
R_m^2 = \min(\lambda^2(1 - c) - |\chi^m_h|_{H^1}^2, |\chi^m_h|_{H^1}^2),
\tag{4.18}
\]

\( B_i \) is the \( i^{\text{th}} \) element of the diagonal lumped mass matrix (the mathematical definition of the lumped mass matrix is given in Section 4.3.4); and \( \epsilon_{\text{tol}} \) is a desired accuracy. (Recall \( \lambda \) is the supercell size.) Note that this incomplete iteration error is independent of the discretization error. Controlling the latter requires a posteriori discretization error estimation and adaptive mesh refinement [Bank and Weiser 1985]. This was not pursued in this thesis.

One important remaining piece of information is the imposition of the boundary conditions associated with \( A \). The uniqueness condition on the continuous solution \( \chi \) is imposed with the simple zero average requirement

\[
\sum_{i=1}^{N_{\text{dof}}} \chi_{hi} = 0.
\tag{4.19}
\]

The \( \lambda \)-periodicity of the solution \( \chi \) is discretely imposed on the iterate after each operation with \( A \) by identifying appropriate pairs of nodes as the same node along periodic edges
[Sæz and Carbonell 1985, Borne et al. 1985], that is, summing the contributions and writing the sum over the old values. We shall elaborate further on this point in the parallel implementation section.

4.3.2 Finite Element Treatment of Nip–Element Boundary Conditions

In this section we consider the special finite element treatment required to effect the non-standard boundary conditions associated with the nip–element method for the effective conductivity problem. We describe this treatment within the context of both a direct and iterative solution strategies. We begin with the simpler crude upper and crude lower bounds before discussing the more sophisticated “Robin–condition” procedures for the sharper upper and lower bounds.

Crude Upper Bound

For the crude upper bound outer problem, the temperature $\chi^U_B$ on the nip boundaries is equal to a constant, $C$, that is not known a priori; $C$ is, thus, a single degree–of–freedom on $\Gamma_1 \cup \Gamma_2$. In order to effect this boundary condition within the structure of the standard finite element mesh and data structure, we employ ideas similar to those developed in the context of the (nonconforming) “mortar” element method [Anagnostou, Maday, Mavriplis & Patera 1990]. In particular, we identify the actual (master) degree–of–freedom associated with the nip–boundary temperature, $C$, with all the (slave) finite element nodes that, on the basis of a standard mesh generation process, reside on $\Gamma_1 \cup \Gamma_2$ (see Figure 4-17). In order to determine the variation in the functional associated with variation in this single degree–of–freedom (that is, the weak form (3.12) for $v = \delta C$): we first compute the variation due to each individual slave node following standard evaluation and summation techniques; we then sum the contributions from all slave nodes. Equivalently, we can assign a single global (master) node number to all local (slave) nodes on $\Gamma_1 \cup \Gamma_2$. This “restriction” procedure is based on the simple identity that the constant function (in fact, 1) over $\Gamma_1 \cup \Gamma_2$ is exactly reproduced by a nodal finite element representation in which all slave nodal values are unity.

The technique is readily implemented in either an iterative framework or a direct approach; the final set of linear equations will, of course, be both symmetric and positive–(semi)definite. In a direct–solver approach, prior to inversion of the system Laplacian, $A$, all the rows (simultaneously the right–hand–side) and columns associated with the slave nodes are summed. This will produce a single row corresponding to a variation of value 1 over $\Gamma_1 \cup \Gamma_2$ and zero elsewhere, and a single column corresponding to the single master node. In the iterative–solver approach, the same effect is achieved by summing all the contributions from the individual slave nodes after each operation with the system Laplacian followed by writing the sum onto the slave nodal values to obtain the correct master node value.

Crude Lower Bound

Solution of the crude lower bound outer problem is analogous to the solution of the bare mesoscale problem and requires no special treatment. The natural adiabatic boundary condition on the nip edges is automatically taken care of in the variational formulation.
Figure 4-17: Master and slave nodes representation.

Sharp Upper Bound

For the sharp upper bound outer problem, the temperature $\chi_{\partial B}$ on the nip boundaries $\Gamma_1$ and $\Gamma_2$ is equal respectively to different constants, $C_1$ and $C_2$, that are not known a priori; $C_1$ and $C_2$ are, thus, each single degrees-of-freedom on $\Gamma_1$ and $\Gamma_2$ respectively. The treatment of the sharp upper bound is hence, similar to the crude upper bound and the ideas described above extends directly to this case. The main difference however, comes from the inner problem contribution to the weak variational statement of the outer problem. This contribution is in the form of the non-standard term in (3.42) which couples the two constants $C_1$ and $C_2$ through multiplying their difference by the conductance, $G(\alpha, \beta)$, of the nip. The product is then added to the two master nodes each with the correct sign. The forcing vector at the master node locations needs also be augmented by the new additional constants appearing on the right-hand-side of equation (3.42) due to the inclination of the nip.

More specifically, in the case of direct-solver approach, the rows and columns in the system Laplacian, $A$, associated with the slave nodes on $\Gamma_1$ and $\Gamma_2$ are first summed respectively to produce two rows (say with numbers $i_1$ and $i_2$) and columns associated with master nodes 1 and 2. (The corresponding entries in the forcing vector are also summed in accordance with the row-summing operation on the system Laplacian.) Next a value of $+G(\alpha, \beta)$ and $-G(\alpha, \beta)$ are added to the entries $(i_1, i_1)$ and $(i_2, i_2)$ respectively in the system Laplacian. Similarly a value of $-G(\alpha, \beta)$ and $+G(\alpha, \beta)$ are added to the entries $(i_2, i_1)$ and $(i_2, i_2)$ respectively in the system Laplacian. These additions follows from the non-standard boundary condition appearing on the left-hand-side of the weak variational form of the sharp upper bound outer problem (equation (3.42)). Finally a value of $-2\beta \sin \theta G(\alpha, \beta)$ and a value of $+2\beta \sin \theta G(\alpha, \beta)$ are added to the forcing vector at locations $i_1$ and $i_2$ respectively, to account for the constant appearing on the right-hand-side of the weak form.

On the other hand, for the case of iterative-solver approach – the one implemented in this thesis – the contributions from the slave nodes on each side of the nip are first
summed as $S_1$ and $S_2$ respectively, after each operation with the system Laplacian. Then the quantities $S_1 + G(\alpha, \beta)(S_1 - S_2)$ and $S_2 - G(\alpha, \beta)(S_1 - S_2)$ are written back to the slave nodes on $\Gamma_1$ and $\Gamma_2$, respectively. The constants on the right-hand-side of (3.42) are added once and for all to the global slave nodal values in the forcing vector. Since data are stored locally in the forcing vector upon passing to the conjugate gradient iterative solver, these constants can then be added to the initial residual after the direct stiffness summation procedure so that the slave nodal values are global.

**Sharp Lower Bound**

The sharp lower bound is the most sophisticated nip, requiring special projection techniques to effect the non-standard "Robin-condition" due to the inner problem contribution which we recall here

$$\frac{1}{2\beta} \int_{-\gamma_L}^{\gamma_R} [v_+(y_1') - v_-(y_1')] \left[ \chi_{LB, out+}(y_1') - \chi_{LB, out-}(y_1') \right] dy_1'. \quad (4.20)$$

Unlike the upper bound nips, all the nodes on the nip edges are now global. In order to simplify the presentation we will treat first the simpler case of identical grids on both edges of the nip. The general case of independent discretization of the nip edges is considered second.

**Case 1: Identical Grids**

We consider a nip with symmetric distribution of $N$ nodes on both $\Gamma_1$ and $\Gamma_2$. We next expand the outer solution $\chi_{LB, out+}$ on $\Gamma_1$, and $\chi_{LB, out-}$ on $\Gamma_2$ using identical nodal Lagrangian interpolants, $\phi_i, i = 1, \ldots, N$ (here the same set is used because $\Gamma_1$ and $\Gamma_2$ have identical grids):

$$\chi_{LB, out+} = \sum_{j=1}^{N} \chi_j^+ \phi_j, \quad (4.21)$$

and

$$\chi_{LB, out-} = \sum_{j=1}^{N} \chi_j^- \phi_j \quad (4.22)$$

respectively. Note that the Lagrangian interpolants, $\phi_i$, used here are one dimensional since the nip edge grid is a one dimensional mesh. These interpolants are, in fact, the traces of the two–dimensional nodal interpolants defined in (4.7) restricted to the nip edge. Now to find the variation due to node $i$ on $\Gamma_1$ using the Galerkin method we set

$$v_{+, h} = \delta \phi_i = \phi_i, \quad (4.23)$$

for which the inner contribution (4.20) becomes:

$$\frac{1}{2\beta} \int_{-\gamma_L}^{\gamma_R} \phi_i \phi_j dy_1' \left[ \chi_j^+ - \chi_j^- \right] \quad (4.24)$$

where $[\chi_j^+ - \chi_j^-]$ is evaluated at the corresponding nodes on both sides. We recognize that this integral is nothing more than the system mass matrix $B$ associated with the 1–dimensional nip edge mesh. Therefore the contribution due to variations on $\Gamma_1$ is $\frac{1}{2\beta} B [\chi^+ - \chi^-$

66
\( \chi^- \), and due to variations on \( \Gamma_2 \) is \(-\frac{1}{2\beta} B [\chi^+ - \chi^-] \).

In a direct-solver approach, we will then have to add \( \frac{1}{2\beta} B_{ij} \) to entries \( a(n_1^1, n_j^1) \) and \( a(n_1^2, n_j^2) \), \( i,j \in 1,..,N \) of the system Laplacian, \( A \), where \( n_1^1 \) is the global node number associated with local node number \( \ell \) on \( \Gamma_1 \) mesh, and \( n_2^2 \) is the global node number associated with local node number \( \ell \) on \( \Gamma_2 \). Similarly, we need to add \(-\frac{1}{2\beta} B_{ij} \) to to entries \( a(n_1^2, n_j^2) \) and \( a(n_2^1, n_j^1) \), \( i,j \in 1,..,N \) of the system Laplacian, \( A \). In an iterative-solver approach, we add the vector \( \frac{1}{2\beta} B [\chi^+ - \chi^-] \) to the nodal values on \( \Gamma_1 \), and the vector \(-\frac{1}{2\beta} B [\chi^+ - \chi^-] \) to the nodal values on \( \Gamma_2 \) following each operation with the system Laplacian.

**Case 2: Different Grids**

Next we consider the case of different grids on \( \Gamma_1 \) and \( \Gamma_2 \). We assume there are respectively \( N_1 \) and \( N_2 \) grid points on \( \Gamma_1 \) and \( \Gamma_2 \), positioned independently. In this case we no longer can use the same set of nodal interpolants and must introduce two sets of (1-D) nodal bases: \( \phi^1_i, i = 1,..,N_1 \) and \( \phi^2_j, i = 1,..,N_2 \) associated with \( \Gamma_1 \) and \( \Gamma_2 \) grids respectively. Similarly we expand the outer solution \( \chi^0_{LB, out,+h} \) on \( \Gamma_1 \), and \( \chi^0_{LB, out, -h} \) on \( \Gamma_2 \):

\[
\chi^0_{LB, out,+h} = \sum_{j=1}^{N_1} \chi^+_j \phi^1_j ,
\]

and

\[
\chi^0_{LB, out, -h} = \sum_{j=1}^{N_2} \chi^-_j \phi^2_j
\]

respectively.

The inner problem contribution (4.20) due to a variation at node \( i \) on \( \Gamma_1 \) \((v_{+,h} = \delta_{ul} \phi^1_i = \phi^1_i) \) then becomes:

\[
\frac{1}{2\beta} \left[ \int_{\gamma_L} \phi^1_i \varphi^1_j dy_i \chi^+_j - \int_{\gamma_L} \phi^1_i \varphi^2_j dy_i \chi^-_j \right].
\]

(The expression due to a variation, \( \phi^2_j \), at node \( i \) on \( \Gamma_2 \) is analogous.) While the first integral is the standard mass matrix entry \( B_{ij} \), the second integral is complicated due to the nonsystematic overlapping of the nodal interpolants on one edge with respect to the other. This in turn requires the integral to be broken into several smaller integrals according to the regions of overlap (Figure 4-18 illustrates this scenario using linear interpolants). This procedure was not adopted in the actual implementation. Instead a new projection technique which takes advantage of the fact that an \( n^{th} \)-order Lagrangian interpolants can exactly represent any \( n^{th} \)-order polynomial was developed as follows:

1. The grids of \( \Gamma_1 \) and \( \Gamma_2 \) are projected to a common reference grid, \( \Gamma_r \), composed of the original nodes. The new grid can thus have at most \( N_r = N_1 + N_2 - 2 \) nodes (see Figure 4-19).

2. The nodal values \( \chi^+ \) and \( \chi^- \) are then mapped to the new grid using standard Lagrangian interpolation:

\[
\chi^+_{i,r} = \sum_{j=1}^{N_1} \chi^+_j \phi^1_j(y_i^r),
\]

(4.28)
Figure 4-18: The integral of the product $\phi_i^1 \phi_j^2$ is complicated due to the non-symmetry of nodes $i$ and $j$. Subdivisions over regions of overlap is required.

Figure 4-19: Projecting $\Gamma_1$ and $\Gamma_2$ onto a single grid $\Gamma_r$. 
\[ \chi_{i}^{-,r} = \sum_{j=1}^{N_r} \chi_{j} \phi_{j}^{r}(y_{i}^{r}) , \]  

(4.29)

where \( y_{i}^{r} \) is the coordinate of node \( i \) on the reference grid, and \( \chi_{i}^{+,r} \) and \( \chi_{i}^{-,r} \) are the new nodal values at node \( i \) on the reference grid.

3. The integral (4.27) is now readily computed on the reference grid as \( \pm \frac{1}{2\beta} B^{r}[\chi^{+,r} - \chi^{-,r}] \) where \( B^{r} \) is the system mass matrix corresponding to the reference grid, that is with a new set of Lagrangian interpolants, \( \phi_{i}^{r}, i=1,..,N_r \). However, it is important to realize that performing the integral on the reference grid is not consistent with the original problem for a variation \( \phi_{i}^{r} \) is not identical with a variation \( \phi_{i} \) on the original \( \Gamma_1 \) (similarly on \( \Gamma_2 \)). Therefore the computed nodal values can not be used directly and must be projected onto the original edges such that the required original variation is honored. In the following steps we will construct the correct projection operators to map \( \pm \frac{1}{2\beta} B^{r}[\chi^{+,r} - \chi^{-,r}] \) back to the original nip edges \( \Gamma_1 \) and \( \Gamma_2 \).

4. We first use the fact that a variation \( \phi_{i}^{1} \) on \( \Gamma_1 \) (similarly \( \phi_{i}^{2} \) on \( \Gamma_2 \)) can be exactly represented by the nodal Lagrangian interpolants \( \phi^{r} \) defined on the reference grid:

\[ \phi_{i}^{1} = \sum_{\ell=1}^{N_r} a_{i}^{\ell} \phi_{\ell}^{r} = \sum_{\ell=1}^{N_r} \phi_{\ell}^{r}(y_{\ell}^{r}) \phi_{i}^{r}, \quad i=1,..,N_1 , \]  

(4.30)

\[ \phi_{i}^{2} = \sum_{\ell=1}^{N_r} b_{i}^{\ell} \phi_{\ell}^{r} = \sum_{\ell=1}^{N_r} \phi_{\ell}^{r}(y_{\ell}^{r}) \phi_{i}^{r}, \quad i=1,..,N_2 . \]  

(4.31)

5. Next we plug (4.30) and (4.31) into (4.27), and recognize that \( \int_{-\gamma_{\ell}}^{\gamma_{\ell}} \phi_{i}^{1} \phi_{j}^{r} dy_{j} \chi_{j}^{+} - \int_{-\gamma_{\ell}}^{\gamma_{\ell}} \phi_{i}^{2} \phi_{j}^{r} dy_{j} \chi_{j}^{-} \) is a contribution to variation on \( \Gamma_1 \). The treatment due to a variation on \( \Gamma_2 \) is completely analogous.

\[ \frac{1}{2\beta} \left[ \int_{-\gamma_{\ell}}^{\gamma_{\ell}} \phi_{i}^{1} \phi_{j}^{r} dy_{j} \chi_{j}^{+} - \int_{-\gamma_{\ell}}^{\gamma_{\ell}} \phi_{i}^{2} \phi_{j}^{r} dy_{j} \chi_{j}^{-} \right] = \frac{1}{2\beta} \phi_{i}^{r}(y_{i}^{r}) \left[ B_{\ell m}^{r} \phi_{j}^{r}(y_{m}^{r}) \chi_{j}^{+} - B_{t n}^{r} \phi_{j}^{r}(y_{n}^{r}) \chi_{j}^{-} \right] . \]  

(4.32)

6. We identify the projection matrices \( C^{1} \) and \( C^{2} \) as follows

\[ C_{i\ell}^{1} = \phi_{i}^{1}(y_{\ell}^{r}) \]  

(4.33)

\[ C_{i\ell}^{2} = \phi_{i}^{2}(y_{\ell}^{r}) . \]  

(4.34)

Expression (4.32) can be expressed as

\[ \frac{1}{2\beta} C_{i\ell}^{1} \left[ B_{\ell m}^{r} C_{j m}^{1} \chi_{j}^{+} - B_{t n}^{r} C_{j n}^{2} \chi_{j}^{-} \right] . \]  

(4.35)

(By analogy the expression due to a variation on \( \Gamma_2 \) is

\[ -\frac{1}{2\beta} C_{i\ell}^{2} \left[ B_{\ell m}^{r} C_{j m}^{1} \chi_{j}^{+} - B_{t n}^{r} C_{j n}^{2} \chi_{j}^{-} \right] . \]  

(4.36)

7. Recognizing, from (4.28) and (4.29), that \( \chi_{i}^{+,r} = C_{j \ell}^{1} \chi_{j}^{+} \) and \( \chi_{i}^{-,r} = C_{j \ell}^{2} \chi_{j}^{-} \), expres-
sions (4.35) and (4.36) simplify to
\[ \frac{1}{2\beta} C_{1t}^1 [B_{tm}^r \chi_m^{+r} - B_{tn}^r \chi_n^{-r}] , \] (4.37)
and
\[ -\frac{1}{2\beta} C_{2t}^2 [B_{tm}^r \chi_m^{+r} - B_{tn}^r \chi_n^{-r}] \] (4.38)
respectively. It then follows that the correct projection from the reference grid back to \( \Gamma_1 \) is accomplished by
\[ C^1 B^r [\chi^{+,r} - \chi^{-,r}] , \] (4.39)
and back to and \( \Gamma_2 \), by
\[ -C^2 B^r [\chi^{+,r} - \chi^{-,r}] . \] (4.40)
(Note that if a direct-solver approach is used, the operators in (4.35) and (4.36) must be added correctly to the system Laplacian (see description for Case 1 above). In an iterative-solver approach the vectors (4.39) and (4.40) are added to the correct locations in the iterate.)

4.3.3 The Stokes Problem

The space-restricted statement of the weak variational equations (2.30) and (2.31) takes the form: Find \( u_h \in [V_h(\Omega)]^2 \) and \( p_h \in P_h(\Omega) \) such that
\[
a(u_h, v) - b(p_h, v) = (\delta_{ij}, v) \quad \forall v \in [V_h(\Omega)]^2 \] (4.41)
\[ -b(q, u_h) = 0 \quad \forall q \in P_h(\Omega) , \] (4.42)
where the bilinear form \( b \) is defined by
\[ b(q, v) = \int_{\Omega} q \nabla \cdot v \, d\Omega , \] (4.43)
and the subspaces \( V_h \) and \( P_h \) are defined as
\[
V_h = \{ v|_{\Omega^s} \in P_2(\Omega^k) \} \cap H_{\#,0}^1(\Omega) \] (4.44)
\[
P_h = \{ q|_{\Omega^s} \in P_1(\Omega^k) \} \cap L_{\#,0}^2(\Omega) . \] (4.45)

The choice of the approximation spaces for both the velocity and pressure is made such that the inf-sup condition [Gunzburger 1989] is satisfied. In particular, our choice is designed on the use of the Taylor–Hood [Taylor and Hood 1973] \( P_2-P_1 \) quadratic and linear triangular elements pair that are known to honor the inf-sup condition. Note that now two meshes are required, a linear one for the pressure and a quadratic one for the velocity. Here it is convenient to use the same geometrical mesh for the pressure by simply ignoring the 2nd-order nodes reserved for the velocity.

Proceeding with the discretization we arrive at the following linear system
\[
\begin{bmatrix}
A & 0 & -D_1^T \\
0 & A & -D_2^T \\
-D_1 & -D_2 & 0
\end{bmatrix}
\begin{bmatrix}
u_{1,h} \\
u_{2,h} \\
p_h
\end{bmatrix} = \begin{bmatrix} B \ 1 \ 0 \ 0 \end{bmatrix} ,
\] (4.46)
where $A$ is the standard Laplacian, $D^T_i = D_{nmi}$ is the derivative operator corresponding to the bilinear form $b$

$$D_{nmi} = \int_\Omega \hat{\phi}_m \frac{\partial \phi_n}{\partial y_i} \, d\Omega ,$$  \hspace{1cm} (4.47)

where $\hat{\phi}_m$ and $\phi_n$ are the nodal basis associated with the linear and quadratic meshes respectively, and $B = B_{ij}$ is the mass matrix defined by

$$B_{ij} = \int_\Omega \phi_i \phi_j \, d\Omega .$$  \hspace{1cm} (4.48)

The Numerical Permeability, $\kappa_h$

The dissipation-related expression for $\kappa_h$ corresponding to (2.33) is given by

$$\kappa_h = \frac{1}{\lambda^2} \int_\Omega \frac{\partial u_{i,h}}{\partial y_k} \frac{\partial u_{i,h}}{\partial y_k} \, dy = \frac{1}{\lambda^2} \left[ \sum_{i=1}^2 u_{i,h}^T A, u_{i,h} \right] .$$  \hspace{1cm} (4.49)

Again here, due to the functional dependence of $\kappa_h$ on $A$ we expect $O(h^3)$ convergence rate with the mesh spacing $h$. This is verified in Chapter 5 with the use of isoparametric elements.

Solution Strategy

The system of equations in (4.46) is symmetric but indefinite and the conjugate gradient can not be applied directly. The solution is based on an iterative scheme, known as the Uzawa algorithm [Maday, Meiron, Patera & Rønquist 1993, Cruz & Patera 1993, Fischer & Patera 1994], which decouples the pressure and velocity (via application of block Gaussian elimination to the system matrix) into two equations

$$S p_h = \sum_{i=1}^2 -D_i A^{-1} B \delta_{i1} ,$$  \hspace{1cm} (4.50)

$$A u_{i,h} = D^T_i p_h + B \delta_{i1} ,$$  \hspace{1cm} (4.51)

where $S$ is defined by

$$S = \sum_{i=1}^2 D_i A^{-1} D^T_i .$$  \hspace{1cm} (4.52)

The first equation (4.50) is first solved for the pressure by a nested conjugate gradient iteration followed by two elliptic solves for the velocity components.

The operator $S$ is symmetric and positive–definite. Its conditioning is strongly influenced by the geometrical distortion of the mesh. For a well–conditioned mesh, (a mesh with a homogeneous element sizes), the outer pressure solve should require few conjugate gradient iterations. This number is significantly increased due to geometrical stiffness; we shall observe this fact in Chapter 5. Note that $S$ is not formed explicitly; only its action on the search direction $w$ for the pressure is required, that is, the vector $Sw$. This vector is determined from an inner elliptic conjugate gradient solve as follows:

$$q_i = D^T_i w ; \quad A z_i = q_i ; \quad Sw = \sum_{i=1}^2 D_i z_i .$$  \hspace{1cm} (4.53)
Note that the same boundary conditions associated with the velocity are imposed in the inner solve. The requirement of zero-mean value for the pressure is effected using a similar relation to (4.19). Finally we remark that no special treatment is required in the presence of nip elements other than enforcing the no slip condition for the velocity along the nip edges.

4.3.4 The Navier–Stokes Problem

Discretization

Following the discussion for the Stokes equations, the spatial discretization of the variational, unsteady, incompressible, Navier–Stokes equations (2.47) and (2.48), takes the form (for clarity we drop the subscript \(h\)):

\[
B \frac{d u_i}{dt} + C(u)u_i = -\frac{1}{Re} A u_i + D_i^T p + B \delta_{i1} \quad (4.54)
\]

\[-D_i u_i = 0, \quad (4.55)\]

for \(i = 1, 2\). For reasons related to stability (will be discussed shortly), the discrete convection operator \(C(u) = C_{ij}(u)\) is defined based on the skew–symmetric form of the continuous convection operator [see Rönquist 1988]:

\[
C_{ij}(u) = \frac{1}{2} \int_{\Omega} \phi_i u_t \frac{\partial \phi_j}{\partial y_t} - \frac{1}{2} \int_{\Omega} \phi_j u_t \frac{\partial \phi_i}{\partial y_t}. \quad (4.56)
\]

In general there are two numerical strategies for the temporal discretization which lead to two different approaches: a “coupled” approach, in which a full–implicit temporal scheme yields a nonlinear non-symmetric elliptic system that must be solved at each timestep; a “decoupled” approach, in which semi–implicit fractional timestepping methods reduce the Navier–Stokes equations to a sequence of more manageable subproblems. (For a comprehensive survey of the different solution methodologies for the incompressible Navier–Stokes equations, the reader is referred to Fischer and Patera [1994].)

We proceed with the second approach by treating the viscous term implicitly using \(1^{st}\)–order backward Euler integration scheme, and the nonlinear convection term explicitly by the \(3^{rd}\)–order Adams–Bashforth. This leads to the following semi–implicit timestepping scheme:

\[
\left[ \frac{1}{Re} A + \frac{1}{\Delta t^n} B \right] u_i^{n+1} - D_i^T p^{n+1} = \frac{1}{\Delta t^n} B u_i^n - \sum_{q=0}^{2} \alpha_q C(u^{n-q}) u_i^{n-q} + B \delta_{i1} \quad (4.57)
\]

\[-D_i u_i^{n+1} = 0, \quad (4.58)\]

where \(\Delta t^n\) is the time step at iteration \(n\). The coefficients \(\alpha_q\) are defined by an algebraic functions of the three last time steps (see Appendix D for derivation):

\[
\alpha_0 = \frac{1}{12} \left[ \frac{12 \Delta t^{n-1} (\Delta t^{n-1} + \Delta t^{n-2}) + 6 \Delta t^n (2 \Delta t^{n-1} + \Delta t^{n-2}) + 4 \Delta t^{n+2}}{\Delta t^{n-1} (\Delta t^{n-1} + \Delta t^{n-2})} \right] \quad (4.59)
\]

\[
\alpha_1 = -\frac{1}{12} \left[ \frac{6 \Delta t^n (\Delta t^{n-1} + \Delta t^{n-2}) + 4 \Delta t^{n+2}}{\Delta t^n (\Delta t^{n-1} + \Delta t^{n-2})} \right] \quad (4.60)
\]
\[ \alpha_2 = \frac{1}{12} \left[ \frac{6 \Delta t^2 \Delta t^{n-1} + 4 \Delta t^n}{(\Delta t^{n-1} + \Delta t^{n-2}) \Delta t^{n-2}} \right]. \quad (4.61) \]

For the case of a constant \( \Delta t \), the coefficients \( \alpha_q \) reduce to the classical constants documented with 3\textsuperscript{rd}-order Adams–Bashforth namely: \( \alpha_0 = \frac{23}{12}; \alpha_1 = -\frac{16}{12} \) and \( \alpha_3 = -\frac{16}{12} \) [Hoffman 1992]. (Note that if a 2\textsuperscript{nd}-order backward Euler is used, these coefficients must assume different constants to retain the 2\textsuperscript{nd}-order accuracy [Ho 1989].)

The merit of this temporal scheme mixing is twofold: first by treating the nonlinear term explicitly, the equations become linear at every time step which would significantly simplify the solution; and second, the implicit treatment of the diffusion operator alleviates the stringent time step requirement associated with the stability of the diffusion operator. The time step is then governed solely by the maximum eigenvalue of the convection operator, \( \lambda_{\max}(C) \), which must lie inside the stability region of the explicit scheme. The eigenvalues of the skew-symmetric convection operator are pure imaginary which motivates the choice of the 3\textsuperscript{rd}-order Adams–Bashforth scheme because of the relatively large portion of the imaginary axis included in its stability region (see Figure D-1 in Appendix D). In fact, the stability boundary of the 3\textsuperscript{rd}-order Adams–Bashforth scheme intersects the imaginary axis at a value of 0.723 implying the stability constraint \( \Delta t^n \lambda_{\max}(C) < 0.723 \). For the one dimensional center–difference–discretized convection operator \( \lambda_{\max}(C) \) is \( i \frac{U}{h} \) where \( U \) is the maximum convective velocity. This leads to the condition \( C_r < 0.723 \) where \( C_r = U \Delta t^n \), which is the well-known Courant condition.

In multidimensional, finite element discretization, this condition is extended rather directly by properly defining the Courant number over the elements spatial directions [Giles 1987]. For our case of triangular finite elements, we define three Courant numbers, based on the last time step \( n \), per element as follows: for each side we take the projection of the velocity vector onto the side divided by half the side length. The new stable \( \Delta t^{n+1} \) is then dictated by the largest Courant number over the entire elements. (The division by half the side length takes into account the quadratic mesh in which the effective mesh spacing is half the actual side length.) In our implementation we used a Courant condition \( C_r \leq 0.5 \) which proved a stable limit for the range of Reynolds numbers considered.

Solution Strategy

One possible solution strategy would be to apply the Uzawa algorithm to (4.58) – (4.58) at each time step. While this is an accurate method it suffers from the very ill-conditioned pressure–solve operator; in addition, the expensive nested Helmholtz inverse solution makes this approach very inefficient. Instead we used a recent operator splitting method which breaks the original system into a sequence of three decoupled subproblems integrated in a new intermediate variable [Maday, Patera & Rønquist 1994, NEKTON V2.85 Manual \(^1\)] as follows:

\[ H \ddot{u}^{n+1}_i = D^T_i p^n + \frac{1}{\Delta t^n} B u^n_i - \sum_{q=0}^{2} \alpha_q C(u^{n-q}) u^{n-q}_i + B \delta_i \quad (4.62) \]

\[ E (p^{n+1} - p^n) = -\frac{1}{\Delta t^n} D_i \ddot{u}^{n+1}_i \quad (4.63) \]

\(^1\)The NEKTON V2.85 Manual, Fluent, Inc., Centerra Resource Park, 10 Cavendish Court, Lebanon, NH 03766.
\[ \mathbf{u}^{n+1}_i = \mathbf{u}^{n+1}_i + \Delta t^n \mathbf{B}^{-1} D^T (\mathbf{P}^{n+1} - \mathbf{p}^n), \]  
(4.64)

where \( \mathbf{u} \) is the intermediate velocity field, \( H = \left[ \frac{1}{\mathcal{L}} A + \frac{1}{\mathcal{L}t^n} B \right] \) is the Helmholtz operator, and \( E \) is the consistent Poisson operator given by

\[ E = \sum_{i=1}^{2} \mathbf{D}_i \mathbf{B}^{-1} D^T_i. \]  
(4.65)

It is important to note that this 1st-order split formulation incurs additional temporal errors of order \( \Delta t \). However, if steady time-independent solutions are reached, these errors vanish completely.

The first step (4.62) is just an elliptic solve for the intermediate velocity variable which is effected by the standard conjugate gradient iteration. Note that at this point, the current guess for the pressure field is incorrect and the incompressibility constraint is not enforced. In the second step (4.63), the pressure field is corrected towards enforcing the divergence-free constraint. In the last step (4.64), the velocity field is updated based on the pressure field correction. Of the three steps, the elliptic solve for the pressure by the unpreconditioned conjugate gradient iteration is the most time consuming since the conditioning of the operator \( E \) deteriorates significantly in the presence of mesh inhomogeneity.

Unlike the spectral element mass matrix, the finite element mass matrix is not diagonal. This in turn, implies the need for a nested conjugate gradient loop in the pressure solve. The mass matrix however is very well-conditioned and when pre-conditioned by the diagonal lumped mass matrix, the number of conjugate gradient iterations is reduced to order unity, which significantly mitigates the otherwise series drawback of the algorithm. To circumvent this inner solve completely we replace the mass matrix, \( \mathbf{B} \), by the diagonal lumped mass matrix \( \mathbf{B}_i \) defined elementally as

\[ \mathbf{B}_i^k = \frac{B_{ii}^k}{\text{Trace}(B^k)} |\Omega^k| \]  
(4.66)

where \( |\Omega^k| \) represents the area of element \( \Omega^k \). Note that this substitution also eliminates the iterations requirement for the last step (4.64). Using the lumped mass matrix is a common practice in finite element analysis. In essence, this is equivalent to replacing the exact integration by the mass matrix with another integration rule based on a different set of weights and collocation points. The error incurred due to this approximation vanishes completely if a steady state time-independent solution is obtained, as is evident from equation (4.55). For the case of unsteady or steady periodic solution, a slight damping effect was observed when using the lumped mass matrix as shown in Chapter 9.

We remark that that this splitting scheme is performed on the well-posed discrete original equations based upon a consistent approximation spaces and thus requires no artificial or special treatment of boundary conditions, which plagues the more classical fractional splitting method based on the continuous original equations [Karniadakis et al. 1990]. Furthermore, this splitting is applicable to, for example, free surface flows, whereas the classical one fails.

74
Inertial Permeability, $\kappa_h$

The temporally–averaged numerical permeability is calculated using the dissipation–related expression, (4.49)

$$\kappa_h = \frac{1}{Re_p^2} \frac{1}{T} \int_0^T \frac{1}{\lambda^2} \left[ \sum_{i=1}^{2} u_{i,h}^T A, u_{i,h} \right] dt .$$  \hspace{1cm} (4.67)

or, equivalently, from the Darcy's definition:

$$\kappa_h = \frac{1}{Re_p} \frac{1}{T} \int_0^T \sum B u_1 dt .$$ \hspace{1cm} (4.68)

where $T$ is the period of oscillations. The sum in (4.68) is over the entire length of the global vector $Bu_1$, where $B$ is the system mass matrix. (The presence of $Re_p$ is due to the nondimensionalization as discussed in Section 2.4 of Chapter 2.) The two expressions should be equivalent which provides a useful check for the implementation. The temporal averaging in unsteady situations is effected by averaging the permeability over a full time period.

4.4 Parallel Considerations

This section describes some of the essentials for the efficient implementation of the conjugate gradient iteration on a distributed-memory, message-passing architecture, here the Intel iPSC/860 32-node hypercube. In particular, we do not intend to discuss the broader context of parallel processing [Fischer and Patera 1994], but rather to describe the parallel constructs necessary for performing the direct-stiffness summation procedure and other communication-related tasks. Then in Section 4.4.2 we discuss the parallel performance aspects of our algorithms.

4.4.1 Parallel Constructs and the DSS Procedure

The parallel constructs are best explained by referring to the schematic shown in Figure 4-20. The supercell is partitioned into, here 4, subcells ($M = 1$) following the tensor–product procedure described in Section 4.1. This phase is carried out on a serial workstation and all the relevant information describing the geometry, topology and associated parallel constructs (discussed shortly) of the individual subcells are dumped to a file. The information associated with each subcell is then read by the assigned processor and the subcell domain is meshed. (The meshing phase is followed by several checks to ensure the integrity of the supercell mesh.) The mesh is composed of non-overlapping elements. Each element is defined by three local nodes forming a triangle. Geometrically coincident local nodes of several elements define a global node. Elements and global nodes are identified by unique numbers. The global node numbers must also be known locally on the elemental level. With this description of the mesh, we discuss next the data structures in conjunction with the conjugate gradient solution.

The data structures are based on the standard technique of elemental evaluation of operators, in which no assembling of the system operator is required. Instead, the quadratures are performed only to assemble elemental matrices (which may or may not be stored). (In the actual implementation, the elemental operators are stored if a memory request is successful, otherwise the operators are computed on the fly as needed.) All the iterates
supercell

Figure 4-20: Definitions of essential parallel constructs
(residuals, search directions, solution vectors) are stored locally based on the individual meshes of the subcells. For example, in Figure 4-20, the solution vector is split into 4 locally defined data structures. At the end of the solution stage, each subcell data structure will hold the correct information based on its mesh, and hence the solution is completely known over the entire supercell.

The action of the system operator is achieved by performing the elemental matrix-vector multiplications, followed by evoking the direct stiffness summation procedure, DSS. The latter procedure maintains the correct global nodal values at the elemental local nodes, that is, it sums all the contributions from all the elemental local nodes associated with a global node, and redistributes the sum back to the elemental local nodes.

By virtue of the partitioning procedure and also due to periodicity, certain global nodes will appear on more than one (and as many as four) processors. This will require in addition to the local DSS on each processor parallel communication (exchange of data) among processors either to attain the correct global values locally, or to impose periodic boundary conditions. Since the time to send \( n \) (32-bits) words, \( t_{\text{comm}}(n) \), is a decreasing function of \( n \), and the communication time \( t_{\text{comm}} \) is 60–100 times slower than the calculation time \( t_{\text{calc}} \), \( t_{\text{comm}}(1) \gg t_{\text{comm}}(\infty) > t_{\text{calc}} \), it becomes critical for parallel efficiency to minimize the number of messages (sends/receives) and to maximize their lengths. For this purpose, we classify and group boundary nodes into either a vertex or edge defined as follows (see Figure 4-20):

**vertex** a node that is shared between more than two subcells. (According to our partitioning strategy, a vertex will always be shared between 4 subcells. This is not true, for example, in the Voronoi–based partition.)

**edge** an ordered set of boundary nodes that is geometrically or periodically shared between two and only two subcells. Note that by definition an edge excludes end vertices and that geometrically, the edge need not be a continuous segment. In fact, our edges are typically a union of a number of line segments separated by gaps and arc segments which are not part of the edge.

The parallel part of the DSS (in addition to the serial part performed on each processor), is then to sum the contributions from all local nodes associated with edges and vertices and redistribute the correct sum back to the contributing local nodes. The parallel constructs developed to accomplish this task are described below for edges and vertices respectively. (In Appendix E, we describe in detail, the implementation of the full DSS algorithm and the associated parallel constructs.)

**DSS Procedure for Edges**

We assign a global edge (buffer) to every pair of geometrically–coincident or periodic edges. This buffer resides on both processors and is identified by a unique number to distinguish it from other global edges residing on the same processor. For each local edge on a processor we construct mappings that identify: the elements and their local nodes of which the edge is composed; the number of the global edge associated with this local edge; the target processor on which the other coincident or periodic edge exists. In fact, local edges are defined by these mappings which are external to the field variable buffer based on the standard finite element data structures.

After the serial-type DSS is performed locally on the iterate on all processors, the nodal values on the edges will be incorrect, lacking the contribution from the other local edge. The
following steps are performed: nodal values from the iterate field are extracted using the local edge mappings; a copy of the nodal values is saved in the reserved buffer; a copy is sent to the target processor and the complementary part of the target processor is received; the local copy in the buffer is summed with the received string; the correct nodal values in the buffer are redistributed to the iterate field local nodes using the mappings. This strategy of simultaneous send/receive communication pattern is designed, in fact, to take advantage of the two-way message-passing capability of the iPSC/860 hypercube architecture for reasons of efficiency. Note also that this parallel constructs requires no modification of the serial–type elemental–based data structures.

**DSS Procedure for Vertices**

The parallel constructs for vertices are very similar to those of edges except that we use a vector reduction, or gather/scatter operation to effect the final summation instead of the local–message–passing algorithm employed for edges. In this strategy, a buffer is allocated for all global vertices on all processors. Each processor fills specific locations of this buffer which correspond to the vertices residing on the processor (the locations of non-residing vertices are left blank). The gather/scatter operation is evoked to sum the buffers from all processors and the correct global values of the vertices are obtained which are then redistributed to the local vertices. The *simpler* adopted strategy is motivated by the fact that there are typically few vertices, and relatively small number of processors, \( P \), and that the time dominated by the gather/scatter – proportional to \( t_{\text{comm}}(1) \log_2(P) \) – is on the order of the time dominated by an alternative more complex local–message–passing strategy similar to the one employed for edges.

Before moving on to the next section we remark that because our partitioning enjoys simple, invariant topology we, could have easily eliminated one of the main problems associated with message passing, *contention*, – a hardware-related concern due to the presence of the same physical channel in the communication paths linking more than one pair of potentially simultaneously communicating processors – by simply effecting the exchange/summation of nodal values on the edges first to the east and west, then to the north and south. Any four-node vertex summation will automatically correctly be effected by this two-stage edge summation procedure (Fox et al 1988). (Note in this procedure the vertices are part of the edges.) The periodic edges, in fact, are not an obstacle to this simple efficient strategy on a hypercube topology, which permits embedding of periodic grids. This can be illustrated for 16 processors using binary gray codes:

\[
\begin{align*}
0000 & \quad 0001 & \quad 0011 & \quad 0010 \\
0100 & \quad 0101 & \quad 0111 & \quad 0110 \\
1100 & \quad 1101 & \quad 1111 & \quad 1110 \\
1000 & \quad 1001 & \quad 1011 & \quad 1010
\end{align*}
\]

where each 4 digit code represents a processor number (e.g., 0000 \( \rightarrow \) 0, 0111 \( \rightarrow \) 7) and a direct communication link exists between each two codes (processors) that differ by one digit (e.g., 0006 and 0100). For example, processor 0000 is linked to processors 0001, 0010, 0100, and 1000 which are exactly the processors on which geometric or periodic edges reside.

### 4.4.2 Parallel Performance

The parallel performance is characterized by the following four interrelated quantities: speed up, \( S_p \); parallel efficiency, \( \eta_p \); load balance, \( \Delta_b \); MFLOPS. We will define these measures
and provide numerical measured and estimated results.

Speed up is defined as follows:

\[ S_p = \frac{t_{\text{calc}}(1)}{t_{\text{calc}}(P)} , \]  

(4.69)

where \( t_{\text{calc}}(n) \) refers to the calculation time on \( n \) processors. Ideally in the limit of no communications and perfect load balance, the speed up should increase linearly with the number of processors.

The parallel efficiency is defined as follows:

\[ \eta_p = \frac{S_p}{P} , \]  

(4.70)

a high \( \eta_p \) implies light communication overhead and good load balance.

There are various ways to define the load balance. Here we define the load balance as follows

\[ \Delta_b = \frac{\max_{p \in P} N_p^{\text{dof}} - \bar{N}^{\text{dof}}}{\bar{N}^{\text{dof}}} + 1 = \frac{\max_{p \in P} N_p^{\text{dof}}}{\bar{N}^{\text{dof}}} , \]  

(4.71)

where \( N_p^{\text{dof}} \) is the number of degrees of freedom, residing on processor \( p \) and \( \bar{N}^{\text{dof}} \) is the average number of degrees of freedom which is equal to the total number of degrees of freedom divided by the number of processors. Note \( \Delta_b \geq 1 \), and \( \Delta_b = 1 \) is the ideal value. Our definition leads directly to the parallel efficiency in the limit of no communication overhead:

\[ \bar{\eta}_p = \frac{1}{\Delta_b} , \]  

(4.72)

where \( \bar{\eta}_p \) refers to the no-communication-overhead case which is, by definition, larger than \( \eta_p \). Relation (4.72) follows by recognizing that, in the limit of no communication, \( t_{\text{calc}}(P) \) scales with \( \max_{p \in P} N_p^{\text{dof}} \) and \( t_{\text{calc}}(1) \) scales with the total number of degrees of freedom.

Lastly, the overall MFLOPS can be estimated by multiplying the typical MFLOPS capability of a processor by the number of processors and the parallel efficiency.

These defined measures are generally problem-dependent, and may vary significantly from one case to another. We conclude this section by determining these performance measures for a small problem \( O(16,000) \) degrees of freedom conduction calculation that could be fit on one (8MB) processor of the i860 hypercube. In Figure 4-21 we plot the speed up and parallel efficiency versus the number of processors used.

We make several remarks concerning Figure 4-21. First, these results are based on the PDE-solution time only and exclude the size–nonlinear meshing time since the latter will dominate the calculation time for the single processor case, and will thus artificially boost the efficiency to a much higher values. Second, on higher processor numbers, these results are somewhat communication-dominated due to the relatively small problem size. (For typical problems, \( O(50,000) \) d.o.f, a parallel efficiency of 93% was achieved on 16 processors. Here \( t_{\text{calc}}(1) \) for the larger problem, which is required to calculate the efficiency, is estimated by linear extrapolation based on the known \( t_{\text{calc}}(1) \) for the smaller problem, the numbers of conjugate gradient iterations, and the degrees of freedom for both problems as follows:

\[ t^\ell_{\text{calc}}(1) = \left( \frac{N^\ell_{\text{dof}}}{N_{\text{dof}}} \right) \left( \frac{t^\ast_{\text{calc}}(1)}{N^\ast_{\text{itr}}} \right) N^\ell_{\text{itr}} , \]  

(4.73)
where the superscripts \( ^{\ell}, ^{s} \) indicates larger and smaller problems respectively; \( N_{dof}, N_{itr} \) indicates the number of degrees of freedom, and the number of iterations respectively.) Third, the decrease of \( \eta_p \) with the number of processors is a consistent attribute of parallel processing due to increased communication overhead. Again here, it is prominent due to the small problem size. Fourth, although, these measurements are accurate since no extrapolation is used to estimate \( t_{calc}(1) \), they are not precise. This is because of the pre-mesh partitioning strategy which yields, due to independent meshing, different total degrees of freedom and different mesh conditionings for different number of processors, which is reflected in the corresponding number of conjugate gradient iterations. This last fact, in addition to other nonlinear hardware–related effects obscure verification that \( \bar{\eta}_p > \eta_p \) through use of (4.72). In fact, for this exercise, only on 8 and 16 processors did we observe that \( \bar{\eta}_p > \eta_p \). (The load balance, \( \Delta_b \), for the 8 and 16–processor runs were 1.236 and 1.503, respectively.) To overcome this measurement artifact, one would need to use an element–based partitioning strategies [Simon 1991] to preserve exactly the number of elements and hence the number of conjugate gradient iterations. Lastly the i860 node performed 5.4 MFLOPS which is estimated with respect to the known CRAY-XMP performance. For this problem the absolute MFLOPS on 16 processors would be about 52. For a typical problem, \( O(50,000) \) d.o.f, \( O(81) \) MFLOPS was achieved on 16 processors.
Chapter 5

Illustrative Examples Using the Nip–Element Method

5.1 Computational Advantages

We begin by remarking that certain (though not all) computational improvements afforded by the nip–element approach are a function of the solution strategy pursued. For example, we expect that domain–decomposition [Quarteroni 1991] (or even diagonal) preconditioners will reduce the sensitivity of the conjugate–gradient convergence rate to geometry–induced conditioning problems; this, in turn, will moderate the computational savings effected by nip–removal strategies. Similarly, the performance of direct solution procedures will reflect geometry–induced conditioning effects only indirectly, through stability considerations. We therefore present not only the “observables” — reduced memory requirements, reduced computation time — for our particular (unpreconditioned) conjugate gradient iteration procedure, but also the sources of these improvements — reduced degrees-of-freedom, reduced geometric distortion, improved conditioning (as reflected in the conjugate gradient iteration count), and better load balancing due to elimination of element–dense nip regions which are the prime reason of imbalance; the latter will hopefully serve as more general metrics, applicable to a wider range of solution approaches.

5.2 The Effective Conductivity Bounds

The numerical implementation of all aspects of the mesoscale–microscale procedure described in Chapter 4 is first validated against known analytical results for the square array of cylinders [Perrins et al. 1979]. Then, we will present and discuss bound results for two configuration effective conductivities for single realizations of random arrays. The bound results presented here, and for the permeability, are for a given concentration, and for a given set of mesh parameters and conjugate gradient tolerances; we remark however, that in virtually all cases presented, discretization error and incomplete–iteration error contributions are small compared to the bound gaps of interest. This is demonstrated for the first random realization example. For purpose of clarity, the subscript $\lambda$ is, for the most part, dropped from the numerical effective property symbol.
5.2.1 Square Array

Validation and Convergence of the Bare Effective Conductivity

In Figure 5-1 we plot the normalized absolute error defined by

$$E_h = k_h - k_{\text{exact}} ,$$

(5.1)

versus the normalized mesh spacing $\hat{h}$ defined as

$$\hat{h} = \frac{h_{\text{nom}}/d}{r} ,$$

(5.2)

where $r$ is the global control refinement parameter appearing in equation (4.2), using $P_1$ linear triangles, $P_2$ subparametric quadratic triangles and $P_2$ isoparametric quadratic triangles. Note that $E_h > 0$, since, from property (2.17), and using the symmetry of the bilinear form $a(u,v)$, $E_h$ can be expressed as the $H^1$-seminorm $k_h - k_{\text{exact}}$

$$E_h = \frac{i}{x^2} a(x - x_h, x - x_h) .$$

(5.3)

The convergence test was done for a square array at the particular concentration of 0.5 for which $k_{\text{exact}} = 0.52465$ [Perrins et al. 1979]. Here $h_{\text{nom}}$ was fixed at 0.5, the near-cylinder refinement parameter $m$, in equation (4.2), was set to two, and $r$ was varied from 1 to 4 resulting in a number of degrees of freedom ranging from 400 to 7000. The calculations were performed on a high-end workstation, here the HP9000 series 735, with a CPU time ranging from a second to a minute.

For the linear elements, second order convergence is obtained with mesh spacing as expected. For quadratic elements, third-order convergence is recovered only when using isoparametric elements consistent with statement (4.15). The skin effect obscure the third-order convergence expected with subparametric quadratic elements. These effects are analyzed in detail in Strang and Fix [1973], pp. 105-116 and pp. 192-204.

Bound Results

The square array represents a severe test for the nip-element approach as the $N_{\text{nip}} = 2$ nips are the only passages for heat flow, as illustrated in Figure 5-2. The bound results for the square array for the concentration $c = 0.780$ are presented in Tables 5.1 and 5.2. Note that the maximum packing for the square array is $c_{\text{max}} = 0.785$; for concentrations $c \geq 0.784$, nip-free meshes can no longer be generated in single precision with our mesh generation procedure. The crude lower bound $k_{LB}^c$ is zero for all values of $\beta$ (see Figure 5-2), and is, therefore, not included in Table 5.1. The following quantities are presented in Tables 5.1 and 5.2: the relative error $E_r$ (dependent on the exact solution $k$),

$$E_r = \frac{|k_B - k|}{k} \times 100 ,$$

(5.4)

where $k_B$ is any of the bounds $k_{UB}, k_{LB}, k_{UB}^c, k_{LB}^c$; or $k_{LB}^c$; the relative error $\bar{E}_r^{c,s}$ (independent of $k$, and thus used in practice),

$$\bar{E}_r^{c,s} = \frac{(k_{UB}^{c,s} - k_{LB}^{c,s})/2}{k^{c,s}} \times 100 ,$$

(5.5)
Figure 5-1: Convergence of the effective conductivity with mesh spacing for a square array at concentration 0.5.

where

\[ \overline{k}^{c,s} = \frac{k_{UB}^{c,s} + k_{LB}^{c,s}}{2}, \]

(5.6)

depending on whether crude or sharper bounds are considered; the total number of finite-element degrees-of-freedom, \( N_{dof} \); and the total conjugate-gradient (wall clock) processing time in seconds, CPU\( (p) \) (all results are for \( p = 16 \) nodes of the Intel iPSC/860 unless otherwise indicated).

We make several remarks for the square-array results. First, the hierarchy of the bounds obtains (for all \( \beta \)):

\[ 0 = k_{LB}^{c} \leq k_{LB}^{s} \leq k \leq k_{UB}^{c} \leq k_{UB}^{s}. \]

(5.7)

Second, “convergence” of the bounds as \( \beta \) is reduced is clearly observed: the sharper bounds gap decreases, and the corresponding average \( \overline{k}^{s} \) approaches the exact analytical result \( k_{exact} = 0.02783 \) (equal to \( k \), our “bare” finite-element solution, to four significant digits). Although the crude upper bound also approaches \( k \) as \( \beta \) is decreased, the residual in \( \overline{k}^{c} \) remains large, as \( k_{LB}^{c} \) is identically zero. Third, the total number of degrees-of-freedom, \( N_{dof} \), is reduced considerably: the savings in \( N_{dof} \) ranges from roughly 15% to 50% as \( \beta \) increases from 0.01 to 0.0. Finally, for this particular problem, the advantages of the nip approach as regards wall-clock time are, at best, minimal: only for the imprecise crude average, \( \overline{k}^{c} \), and the \( \beta = 0.04 \) sharper average, \( \overline{k}^{s} \), do we observe any reduction in CPU (see Table 5.2).
5.2.2 Random Realizations

Realistic Case

We now consider application of the nip-approach to a “real” case — the configuration effective conductivity of a single realization of a random array of cylinders — in order to demonstrate the full generality of our procedure. (We remark that all random-array realizations presented here are drawn from a random sequential addition process [Widom 1966, Torquato 1991, Cruz & Patera 1993].) The particular realization \( \{ y \}_N \), shown in Figure 5-3, has concentration \( c = 0.500 \) and contains \( N = 50 \) inclusions; the finite-element mesh and associated parallel partition for the outer problem (\( \alpha_c = 0.1, \beta = 0.075, N_{nip} = 26 \)) are shown in Figure 5-4. Table 5.3 presents, in addition to the quantities already defined for Tables 5.1 and 5.2: the geometrical stiffness, \( GS \), defined as the ratio of the maximum edge-length of all finite elements present to the minimum edge-length of all finite elements present; the number of conjugate gradient iterations, \( N_{it} \); the load balance, \( \Delta_b \), the parallel efficiency, \( \eta_p \), and absolute MFLOPS. We observe the very good accuracy of the sharper bounds: in random arrays, the nips are typically not the preferred passages for heat flow (the accuracy is, therefore, even better at lower concentrations, as relatively more alternative heat-flow routes are available). Second, although there is only about a 15\% reduction in \( N_{dof} \) when the nips are introduced, the wall-clock time to compute \( \bar{\kappa}^s \) drops by 32\%. This savings in CPU (approximately a factor of three for a particular sharper bound) is primarily due to: improved parallel efficiency and overall absolute MFLOPS; improved conditioning — engendered by the dramatically decreased geometrical stiffness, \( GS \) — as evidenced by the reduction in \( N_{it} \) by a factor of two; improved load balancing, as evidenced by the significant decrease in \( \Delta_b \). We remark that the sharper bounds demand extra communication overhead which is evidenced by the slight drop in \( \eta_p \) relative to the
Figure 5-3: Single realization of a random array of cylinders for the concentration $c = 0.500$ and $N = 50$ inclusions ($\lambda = 8.862$).

Figure 5-4: Outer problem geometry ($N_{\text{nip}} = 26$ nips), parallel partition (for 16 processors), and finite-element mesh for the realization shown in Figure 5-3.
<table>
<thead>
<tr>
<th>$\beta$</th>
<th>$k_0$</th>
<th>$E_r$</th>
<th>$N_{dof}$</th>
<th>CPU(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>0.02783</td>
<td></td>
<td>6948</td>
<td>35</td>
</tr>
<tr>
<td>$k_{LB}$</td>
<td>0.04</td>
<td>0.02344</td>
<td>16</td>
<td>3701</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.02504</td>
<td>16</td>
<td>5193</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>0.02631</td>
<td>5.5</td>
<td>5989</td>
</tr>
<tr>
<td>$k_{UB}$</td>
<td>0.04</td>
<td>0.04917</td>
<td>77</td>
<td>3667</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.03697</td>
<td>33</td>
<td>5115</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>0.03188</td>
<td>15</td>
<td>5859</td>
</tr>
<tr>
<td>$k_{UB}$</td>
<td>0.04</td>
<td>0.02822</td>
<td>1.4</td>
<td>3669</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.02799</td>
<td>0.57</td>
<td>5117</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>0.02790</td>
<td>0.25</td>
<td>5861</td>
</tr>
</tbody>
</table>

Table 5.1: Bound results for the square array of cylinders for the concentration $c = 0.780$.

<table>
<thead>
<tr>
<th>$\beta$</th>
<th>$\bar{k}$</th>
<th>$E_{r}^{c,s}$</th>
<th>CPU(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{k}$</td>
<td>0.04</td>
<td>0.02459</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.01849</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>0.01594</td>
<td>100</td>
</tr>
<tr>
<td>$\bar{k}$</td>
<td>0.04</td>
<td>0.02583</td>
<td>9.3</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.02652</td>
<td>5.6</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>0.02711</td>
<td>2.9</td>
</tr>
</tbody>
</table>

Table 5.2: Crude- and sharper-bound average results for the square array of cylinders ($c = 0.780$).

cruider bounds. Finally, we emphasize the utility and rigor of the approach: $\bar{k}$ and $E_{r}$ can be computed without any knowledge of the “bare” finite element solution, $k$.

We conclude this example by verifying the small discretization and incomplete iteration error contributions in comparison to our bounds of interest. We re-do the calculations with a much finer mesh and a more stringent incomplete iteration error control. For the cruder bounds there have been no change in the calculated effective conductivities for four decimal places and thus are not included with the new results presented in Table 5.4. The appreciable decrease in the bound gap of the finer bounds (see Table 5.4) reflects convergence of the effective properties with mesh spacing. The maximum relative change in the properties themselves with respect to the coarser mesh results in Table 5.3 is less than one percent, (0.84%), however. This small improvement obtained is probably not warranted given the significant increase in the CPU time.

**Necessary Case**

The necessity of the nip-approach is illustrated for the “critical” case, shown in Figure 5-5, in which a single realization of a random array exhibits a pair of extremely close inclusions. This realization, for concentration $c = 0.375$ and $N = 17$ inclusions, is part of a sample of 20 realizations generated in the process of validating a “surrogate model” — the two-
Table 5.3: Bound results for a random array realization for the concentration $c = 0.500$.

<table>
<thead>
<tr>
<th>$k$, $\bar{k}$</th>
<th>$E_r$</th>
<th>$\bar{E}_r$</th>
<th>$N_{dof}$</th>
<th>CPU</th>
<th>GS</th>
<th>$N_{it}$</th>
<th>$\Delta_b$</th>
<th>$\eta_p$</th>
<th>MFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>0.2903</td>
<td>—</td>
<td>—</td>
<td>55776</td>
<td>459</td>
<td>2245</td>
<td>2013</td>
<td>2.2</td>
<td>71%</td>
</tr>
<tr>
<td>$k^p_{LB}$</td>
<td>0.2251</td>
<td>22</td>
<td>—</td>
<td>46018</td>
<td>184</td>
<td>70</td>
<td>1412</td>
<td>1.4</td>
<td>98%</td>
</tr>
<tr>
<td>$k^p_{LP}$</td>
<td>0.3046</td>
<td>4.7</td>
<td>—</td>
<td>46018</td>
<td>193</td>
<td>70</td>
<td>1430</td>
<td>1.4</td>
<td>94%</td>
</tr>
<tr>
<td>$k^p$</td>
<td>0.2646</td>
<td>15</td>
<td>—</td>
<td>377</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>$k^p_{LB}$</td>
<td>0.2829</td>
<td>2.5</td>
<td>—</td>
<td>46783</td>
<td>161</td>
<td>49</td>
<td>1155</td>
<td>1.4</td>
<td>93%</td>
</tr>
<tr>
<td>$k^p_{LP}$</td>
<td>0.2934</td>
<td>1.1</td>
<td>—</td>
<td>46018</td>
<td>151</td>
<td>70</td>
<td>1081</td>
<td>1.4</td>
<td>92%</td>
</tr>
<tr>
<td>$k^p$</td>
<td>0.2882</td>
<td>1.8</td>
<td>—</td>
<td>312</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 5.4: Effect of discretization and incomplete iteration error on the bound results of Table 5.3.

<table>
<thead>
<tr>
<th>$k$, $\bar{k}$</th>
<th>$E_r$</th>
<th>$\bar{E}_r$</th>
<th>$N_{dof}$</th>
<th>CPU</th>
<th>$N_{it}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>0.2903</td>
<td>—</td>
<td>—</td>
<td>125791</td>
<td>3742</td>
</tr>
<tr>
<td>$k^p_{LB}$</td>
<td>0.2853</td>
<td>1.75</td>
<td>—</td>
<td>107597</td>
<td>2490</td>
</tr>
<tr>
<td>$k^p_{LP}$</td>
<td>0.2924</td>
<td>0.72</td>
<td>—</td>
<td>107597</td>
<td>3000</td>
</tr>
<tr>
<td>$k^p$</td>
<td>0.2889</td>
<td>1.2</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

dimensional structure-independent bound of Hashin [1970] — for the effective conductivity which is presented in Chapter 6. The extremely close inclusions in Figure 5-5 are separated by a center-to-center distance of $1 + 9.6 \times 10^{-5}$, which, for our mesh procedure and machine precision, precludes successful mesh generation (even though we generate the mesh for each processor domain independently, thereby reducing the range of scales). The finite-element mesh and parallel partition for the outer problem ($\alpha_c = 0.1$, $\beta = 0.075$, $N_{nip} = 2$) for the realization shown in Figure 5-5, are shown in Figure 5-6, and the associated sharper bound predictions are summarized in Table 5.5. Note the very good accuracy obtained for the estimate $\bar{k}^p$, and the reasonable load balance, $\Delta_b$, obtained for this ostensibly very heterogeneous problem. Most importantly, the variational-bound nip-element procedure presented in this paper permits us to complete the ensembles for the random effective conductivity calculations [Cruz & Patera 1993, Yesilyurt et al. 1993]. Simple elimination of sampled realizations from the attempted ensemble could substantially bias the statistics of the averaged results, since systematic rejection of potentially much less conducting realizations might occur.

Table 5.5: Bound results for a random array realization for the concentration $c = 0.375$.

<table>
<thead>
<tr>
<th>$k$, $\bar{k}$</th>
<th>$\bar{E}_r$</th>
<th>$\Delta_b$</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k^p_{LB}$</td>
<td>0.4101</td>
<td>1.3</td>
<td>141</td>
</tr>
<tr>
<td>$k^p_{LP}$</td>
<td>0.4112</td>
<td>1.2</td>
<td>136</td>
</tr>
<tr>
<td>$k^p$</td>
<td>0.4107</td>
<td>0.13</td>
<td>277</td>
</tr>
</tbody>
</table>
Figure 5-5: Single realization of a random array of cylinders for the concentration $c = 0.375$ and $N = 17$ inclusions ($\lambda = 5.965$); the pair of extremely close inclusions in the upper right-hand corner precludes successful mesh generation.

Figure 5-6: Outer problem geometry ($N_{\text{nip}} = 2$ nips), parallel partition (for 4 processors), and finite-element mesh for the realization shown in Figure 5-5.
5.3 The (Stokes) Permeability Bounds

5.3.1 Validation and Convergence of the Permeability

We begin by verifying the third order convergence of the permeability with mesh spacing expected with the 2\textsuperscript{nd}–order isoparametric finite element implementation. In Figure 5-7 we plot the normalized absolute error defined by

$$E_h = \kappa_h - \kappa_{exact}$$  \hspace{1cm} (5.8)

versus the normalized mesh spacing $\hat{h}$ given in equation (5.2) using both subparametric and isoparametric implementations. The tests were performed for a hexagonal array at concentration 0.6 (shown in Figure 5-8) for which $\kappa_{exact} = .001452$ [Sangani and Acrivos 1982a].

![Figure 5-7: Convergence of the Permeability with mesh spacing for a hexagonal array at concentration 0.6. The third order convergence is observed with isoparametric finite elements.](image)

Similar to the effective conductivity convergence test, $h_{nom}$ was fixed at 0.5, $m$ set to two, and $r$ was varied from 1 to 4 resulting in a a total (velocity and pressure) number of degrees of freedom ranging from from 1400 to 12000. Test calculations were performed on 4 processors of the Intel i860 hypercube and consumed 30 to 1000 seconds of computation time. We again observe the 3\textsuperscript{rd}–order convergence of the permeability with isoparametric implementation. The skin effects due to the subparametric implementation obscure the 3\textsuperscript{rd}–order convergence.
Figure 5-8: A particular finite element mesh comprising 1965 finite triangle elements for a hexagonal array at concentration 0.6.

5.3.2 Illustrative Examples

We present two examples for the (Stokes) Permeability Bounds: the first one is for a random realization and is intended to demonstrate the significant numerical savings associated with the crude but numerically rather sharp Stokes bounds. The second one is intended to verify the rigor of the hierarchy of the bounds for an interesting single-nip example.

In Table 5.6 we present bound results for the configuration permeability of a random array of cylinders shown in Figure 5-9. The particular realization has concentration $c = 0.500$ and contains $N = 25$ inclusions; the parallel partition of the outer domain, $C \ (\alpha_c = 0.1, \ \beta = 0.1, \ \mathcal{N}_{nip} = 13)$, is shown in Figure 5-10. In Table 5.6, $N_{dof,v}$ refers to the degrees-of-freedom per velocity component, and $N_{it,p}$ refers to the number of (outer) pressure conjugate gradient iterations; all other symbols are analogous to their conduction counterparts. As anticipated in Section 3.3.1 of Chapter 3, these bounds, although conceptually crude, are, in fact, numerically rather sharp because of the $O(\alpha^2)$ dependence of the permeability on interparticle spacing, as opposed to the $O(\alpha)$ dependence of the effective conductivity.

The very significant savings in CPU is primarily due to improved conditioning, as reflected in the factor of ten reduction in outer pressure iterations relative to the "bare" case. (In this particular example we imposed progressively stringent tolerances for incomplete iteration of the nested solve by requiring an $O(1000)$ reduction in the initial residual — in practice, a fixed tolerance is specified and the iteration time for the "bare" case can be reduced by at least a factor of two with relatively little effect on the discrete solution.) The dramatic reduction in requisite pressure iterations reflects the sensitive dependence of the Uzawa pressure operator to significant geometric distortion; earlier tests for moderate distortion (see Figure 10 of Maday, Meiron, Patera & Rønquist [1993], already indicate some
Figure 5-9: Single realization of a fibrous porous medium for the concentration $c = 0.500$ and $N = 25$ cylindrical (solid) inclusions ($\lambda = 6.266$).

Figure 5-10: Upper bound outer–problem geometry ($N_{nip} = 13$ nips) and parallel partition (for 16 processors) for the porous medium realization shown in Figure 5-9.
<table>
<thead>
<tr>
<th>$\kappa$, $\bar{\kappa}$</th>
<th>$E_r$</th>
<th>$\bar{E}_r$</th>
<th>$N_{doft}$</th>
<th>$CPU(16)$</th>
<th>$GS$</th>
<th>$N_{itp}$</th>
<th>$\Delta_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa$</td>
<td>0.001913</td>
<td>—</td>
<td>30391</td>
<td>1007587</td>
<td>1932</td>
<td>2890</td>
<td>3.3</td>
</tr>
<tr>
<td>$\kappa_{LB}$</td>
<td>0.001764</td>
<td>7.8</td>
<td>21818</td>
<td>4313</td>
<td>34</td>
<td>290</td>
<td>1.4</td>
</tr>
<tr>
<td>$\kappa_{UB}$</td>
<td>0.001962</td>
<td>2.6</td>
<td>21958</td>
<td>4487</td>
<td>36</td>
<td>275</td>
<td>1.4</td>
</tr>
<tr>
<td>$\bar{\kappa}$</td>
<td>0.001863</td>
<td>—</td>
<td>5.3</td>
<td>8800</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 5.6: Bound results for the permeability of the random array realization shown in Figure 5-9, ($c = 0.500$).

<table>
<thead>
<tr>
<th>$\kappa$, $\bar{\kappa}$</th>
<th>$E_r$</th>
<th>$\bar{E}_r$</th>
<th>$N_{doft}$</th>
<th>$CPU(16)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa$</td>
<td>0.7439011</td>
<td>—</td>
<td>—</td>
<td>13697</td>
</tr>
<tr>
<td>$\kappa_{LB}$</td>
<td>0.7439008</td>
<td>4.0E-5</td>
<td>—</td>
<td>13214</td>
</tr>
<tr>
<td>$\kappa_{UB}$</td>
<td>0.7439019</td>
<td>1.5E-4</td>
<td>—</td>
<td>13357</td>
</tr>
<tr>
<td>$\bar{\kappa}$</td>
<td>0.7439013</td>
<td>—</td>
<td>3.4E-5</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 5.7: Bound results for the permeability of the symmetric realization shown in Figure 5-11 ($c = 0.1$).

degradation with aspect ratio. Improved load balancing and reduced number of degrees-of-freedom also contribute to the reduction in CPU, the former having a larger influence than the latter. (Unfortunately, we failed to precondition the Uzawa pressure operator with the (lumped) finite-element pressure mass matrix (Maday et al. 1993). The mass matrix preconditioning would reduce the number of pressure iterations for both the ”bare” and the nip calculations, but would have a greater effect on the more ill-conditioned ”bare” discretization. The reduction in CPU time reported in Table 5.6 is, thus, exaggerated.)

In the second example we consider the realization shown in Figure 5-11 which has $N = 2$ cylindrical inclusions at a concentration of 0.1, in which the center-line connecting the two inclusions is parallel to the flow direction. The motivation behind this example follows the discussion presented in Section 3.3.1 of Chapter 3, in which the physical motivation behind the blockage strategy to produce a lower bound could be obscured by the counter effects of reduced drag due to reduced surface area exposed to fluid and elimination of relatively high dissipative regions, particularly in this symmetric flow situation in which the nip region has very minor flow-passing role. The rigor of the bound–hierarchy proof is verified by the numerical results shown in Table 5.7. Indeed, these counter effects will only render the bounds very sharp. We remark that these results were obtained for the case $\beta = .1$ using a relatively very fine grid and stringent incomplete iteration control. (The discretization error is expected to be $O(10^{-7})$ based on the convergence test of Figure 5-7. This error is smaller than the reported bound gap.) We also considered $\beta = 0.5$ for the lower bound case (see Figure 5-12) and obtained a consistent lower value of 0.7237034 for the permeability.

### 5.4 Inertial Porous–Media Flow Example

As discussed in Section 3.3.2 of Chapter 3, inertial flows do not possess permeability-maximizing properties, which renders our creeping-flow microscale bounds no longer rigor-
Figure 5-11: Special realization of a fibrous porous medium for the concentration $c = 0.1$ and $N = 2$ cylindrical (solid) inclusions ($\lambda = 3.963$).

Figure 5-12: Lower-bound geometry for the realization of Figure 5-11.
<table>
<thead>
<tr>
<th>$\kappa$, $\overline{\kappa}$</th>
<th>$E_r$</th>
<th>$E_r^c$</th>
<th>$Re_{&lt;u_1&gt;}$</th>
<th>$N_{dof, u}$</th>
<th>CPU(16)</th>
<th>GS</th>
<th>$\Delta_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa$</td>
<td>0.006873</td>
<td>—</td>
<td>—</td>
<td>68.7</td>
<td>15508</td>
<td>872510</td>
<td>84.9</td>
</tr>
<tr>
<td>$\kappa_{LB}$</td>
<td>0.006811</td>
<td>0.9</td>
<td>—</td>
<td>68.1</td>
<td>13743</td>
<td>278324</td>
<td>18.7</td>
</tr>
<tr>
<td>$\kappa_{UB}$</td>
<td>0.007029</td>
<td>2.3</td>
<td>—</td>
<td>70.2</td>
<td>13783</td>
<td>245500</td>
<td>14.7</td>
</tr>
<tr>
<td>$\overline{\kappa}$</td>
<td>0.006920</td>
<td>—</td>
<td>0.7</td>
<td>69.2</td>
<td>—</td>
<td>523824</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 5.8: Bound results for the permeability of the random array realization shown in Figure 5-13 at $Re_p = 100$.

ous. However, we still expect the hierarchy to obtain based on the physical plausibility of the bounds, and the effectively inertia-free flows expected to occur in the narrow nip regions even when the global Reynolds number may be large. We substantiate this reasoning with a random realization Navier–Stokes calculation containing 14 inclusions at concentration of 0.3 shown in Figure 5-13. In this run, $Re_p$, is chosen to be 100, at which a moderate

Figure 5-13: Geometry for the inertial porous-media realization containing 14 inclusions at a concentration of 0.3, ($\lambda = 6.054$).

flow Reynolds number, $Re_{<u_1>}$ is expected to obtain (based on known regular-array calculations). The geometries of the lower and upper bound calculations are shown in Figure 5-14 in which $N_{nip} = 4$ nips are present. Table 5.8 presents the bound results for the permeability calculations. The hierarchy of the bounds is indeed obtained despite the relatively important convective contributions, $(Re_{<u_1>} = O(70)$, see Table 5.8). That is because the nip gaps form percolation "valves"; the flow is effectively inertia-free through these passages, even though the global Reynolds number is large. This is made evident in Figure 5-15 which contours the speed over the nip-free domain. The clustered inclusions in the middle almost block the flow entirely, forming a stagnation region.

94
Figure 5-14: Lower-bound and upper-bound geometries and parallel partition of the realization of Figure 5-13.

Given the relatively moderate geometrical stiffness of the original problem (see Table 5.8), the savings for this particular realization are somewhat moderate. (We only obtain a factor of 3 reduction in CPU time for both the lower-bound and upper-bound calculations compared to the original problem.) More typically, for worse geometrical distortion situations, the conditioning of the consistent Poisson operator associate with the pressure solve deteriorates rapidly, resulting in dramatic observed savings.

These simulations were started from rest and marched in time completing 20,000 time steps at a fixed Courant condition of 0.5. The steady-state final time step $\Delta t^{20,000}$ was on the order of 0.001. It is important to mention that the flow is found to be unsteady at this Reynolds number for all three calculations, and the presented data are, thus, temporally averaged. The unsteadiness is in the form of periodic oscillations. We illustrate this using the lower-bound calculation. In Figure 5-16 we plot the time history of the $y_1$-component of the velocity at a selected point in the lower-bound domain along with the time history of the permeability. We observe almost-steady-periodic oscillations which appears to be composed of several subharmonics. The frequency content of the wave could be determined by applying discrete Fourier transform methods but is not analyzed any further since it is beyond the purpose of this example. We remark also that the possibility of non-physical origin of these oscillations has not been ruled out. The latter requires revising the grid to establish mesh-independent results which is not undertaken here. We comment, however, that these oscillations are very likely physical since they appear at a later stage when the maximum speed has already been reached, and not as early accompanying wiggles. (Note that the initial portion of the history plot up to 5 time units is oscillation-free.) In any event, the effects of unsteadiness on the spatially-averaged permeability is minimal and will not alter the established hierarchy of the bounds (See Figure 5-16). We shall discuss the unsteadiness phenomena in greater detail in Chapter 9.
Figure 5-15: A plot showing speed contours for the Navier–Stokes calculation at Reynolds number $Re_{\phi} = 68.7$ in the fibrous porous medium of Figure 5-13.

5.5 Demonstrative Examples Using Sensitivity Derivatives

In this section we present two examples to illustrate the use of sensitivity derivatives to infer information on the relative importance of nips which can then be used to guide "mesh" design. The results presented here are for the effective conductivity crude upper bound and are based on the material discussed in Section 3.4 of Chapter 3.

5.5.1 Test Case

The first example we consider is for a square array at concentration of 0.7 with $N_{nip} = 2$ numbered as shown in Figure 5-17. This example serves both as validation check on the implementation of the sensitivity derivative algorithm presented in Section 3.4.2 of Chapter 3, and as an educational exercise since it is known a priori that the normal nip should have a higher sensitivity derivative for a horizontal imposed potential gradient.

Presented in Table 5.9 are the following quantities: the "exact" sensitivity derivative, $\frac{dE}{dD}$, calculated from (3.71) by finite differencing; the equivalent $\frac{dF}{dD}$ calculated using the
Figure 5-16: Time history plots for \( u_1 \) (upper two curves) and \( \kappa_{LB} \) (bottom curve) for the lower-bound calculation. The associated history point is at \( y = (0.8587, 0.1446) \) with reference at the lower left corner of the lower-bound realization shown in Figure 5-14.

adjoint variable method; the relative percentage error, \( E \), defined by

\[
E = \left| \frac{\frac{d\mathcal{F}}{dD} - \frac{d\mathcal{F}}{d\delta D}}{\frac{d\mathcal{F}}{dD}} \right| \times 100.
\]  

(5.9)

In the first test we choose finite step value of \( \delta D = \frac{\beta}{100} \). The nip size parameter, \( \beta \), is set equal to 0.1. The results obtained, although consistent with the physics of the problem, indicate a somewhat large discrepancy between the exact and the adjoint–method predictions. The source of these large errors is mainly the incomplete iteration error (set here to 0.0001) which results in \( R \neq 0 \) and hence \( \mathcal{F} \neq F \), in addition to the relatively coarse design step, \( \delta D \). (Note that discretization error is not a prime factor in this discrepancy and influence the results only indirectly since the formulation (see Section 3.4 of Chapter 3) departs from the discretized equations, and the same error is present in both calculations.) We repeat the calculations using a more stringent incomplete iteration error control (set here to 0.000001) and a smaller design step of \( \delta D = \frac{\beta}{10000} \). Table 5.10 presents the new results for which the error has diminished significantly (although less sharply for nip number 2).

To complete the illustration, we solve the square array problem retaining only one nip at a time. We observe 15% relative change in the effective conductivity with respect to the nip–free problem when only nip number 2 is present (i.e., nip 1 is removed), and a value of 8.4% relative change when only nip 1 is present (i.e., nip 2 is removed) consistent with the sensitivity derivatives information.
Figure 5-17: Square array with $N_{nip} = 2$.

<table>
<thead>
<tr>
<th>nip no.</th>
<th>$\frac{dF}{dD}$ exact</th>
<th>$\frac{dF}{dD}$ AVM</th>
<th>$E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2493</td>
<td>0.2901</td>
<td>16.3</td>
</tr>
<tr>
<td>2</td>
<td>0.1291</td>
<td>0.1576</td>
<td>22.0</td>
</tr>
</tbody>
</table>

Table 5.9: Sensitivity derivative results for the square array with large incomplete iteration error effect. AVM indicates result obtained by adjoint variable method.

5.5.2 A Random Realization

Our next example considers a random realization containing 9 inclusions at a concentration of 0.5. Using $\alpha_c = 0.05, \beta = 0.1$, 5 nips are formed and numbered as shown in Figure 5-18. The relative criticality of the nips in this example is not readily decided by human judgment. By examining Figure 5-18, one can expect that nip number 5 is the least critical given its location and parallel orientation with respect to the imposed temperature gradient. On the other hand, nip number 3 has the smallest $\alpha$, (see Table 5.11), and given the $O(\alpha)$ dependence of the heat flux, one could anticipate that nip number 3 is the most critical one (the very small gap almost blocks the heat flow). Turning to the remaining nips 1,2 and 4, we see that $\alpha$ is relatively large and on the same order (see Table 5.11) so it is not the prime factor. In such instances, the sensitivity derivatives are valuable information to guide the design process.

<table>
<thead>
<tr>
<th>nip no.</th>
<th>$\frac{dF}{dD}$ exact</th>
<th>$\frac{dF}{dD}$ AVM</th>
<th>$E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3178</td>
<td>0.2995</td>
<td>5.7</td>
</tr>
<tr>
<td>2</td>
<td>0.1809</td>
<td>0.1537</td>
<td>15.8</td>
</tr>
</tbody>
</table>

Table 5.10: Sensitivity derivative results for the square array with small incomplete iteration error effect. AVM indicates result obtained by adjoint variable method.
Figure 5-18: Sensitivity derivatives random realization example containing 5 nips at concentration $c = 0.500$ and 9 inclusions ($\lambda = 3.759$).

Table 5.12 presents the sensitivity derivatives calculated using the adjoint variable method based on the algorithm presented in Section 3.4.2 of Chapter 3. The results indicate that nip number 3/5 is the most/least critical one as anticipated. The order of criticality of the remaining nips is 1, 2 and 4 respectively. This information can then be used to obtain sharper bounds by reducing $\beta$ on the most critical nips or even eliminating certain nips. In Table 5.13 we show the percentage decrease in $k_U$ and the percentage increase in the associated CPU time due to elimination of individual nips one at a time. Again, we observe that the relative improvement in $k_U$ is in harmony with the sensitivity derivatives information.

<table>
<thead>
<tr>
<th>nip no.</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.02657</td>
</tr>
<tr>
<td>2</td>
<td>.02557</td>
</tr>
<tr>
<td>3</td>
<td>.00343</td>
</tr>
<tr>
<td>4</td>
<td>.01983</td>
</tr>
<tr>
<td>5</td>
<td>.02814</td>
</tr>
</tbody>
</table>

Table 5.11: The distance $\alpha$ associated with the nips numbered in Figure 5-18.
<table>
<thead>
<tr>
<th>nip no.</th>
<th>$\frac{dF}{dD}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10412</td>
</tr>
<tr>
<td>2</td>
<td>0.06655</td>
</tr>
<tr>
<td>3</td>
<td>0.17231</td>
</tr>
<tr>
<td>4</td>
<td>0.05011</td>
</tr>
<tr>
<td>5</td>
<td>0.01890</td>
</tr>
</tbody>
</table>

Table 5.12: Sensitivity derivative results for the random array shown in Figure 5-18.

<table>
<thead>
<tr>
<th>eliminated nip no.</th>
<th>% decrease in $k_{UB}$</th>
<th>% increase in CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.1</td>
<td>6.9</td>
</tr>
<tr>
<td>2</td>
<td>1.9</td>
<td>7.6</td>
</tr>
<tr>
<td>3</td>
<td>6.1</td>
<td>49.7</td>
</tr>
<tr>
<td>4</td>
<td>1.4</td>
<td>7.9</td>
</tr>
<tr>
<td>5</td>
<td>0.2</td>
<td>6.9</td>
</tr>
</tbody>
</table>

Table 5.13: Effect of eliminating individual nips on the effective conductivity and the associated CPU time. Results are for the random array shown in Figure 5-18.
Chapter 6

The Surrogate Framework

6.1 Introduction

In this chapter we describe a Bayesian–validated statistical framework developed in Yeşilyurt, Ghaddar, Cruz, and Patera [1993] for validation of noisy computer simulation surrogates. We shall then apply this framework to validate microstructure–independent proposed and constructed models respectively, for the dependence of effective conductivity (Chapter 7) and seepage permeability (Chapter 8) on the inclusion concentration. These simplified input–output models can then substitute for the original simulation in ensuing design and optimization studies. The surrogate framework is readily motivated by the limited resources and lack of automation flexibility in large-scale simulations. These facts place the surrogate approach at a competitive advantage for complete optimization results over the more–accurate but expensive simulation–based approach. Additional positive attributes of the cost–effective surrogate approach include: permitting significant flexibility in updating the design parameters during the optimization process without a need to appeal to the original simulation; readily integrating prior information to sharpen the surrogate.

On the other hand, developing surrogates for the effective properties, $k$ and $\kappa$, given a set of limited noisy data points is not a trivial task. We can assume little regularity information particularly at higher concentrations at which the effective properties may vary rapidly with concentration. The process is further complicated by the expensive, decoupled, and discontinuous nature of the simulation data points. It is therefore necessary, for reliable results, to be able to make a precise statement regarding the discrepancy between the simulation and any proposed or constructed surrogate given a fixed validation sample size, $N^v$. Proceeding with invalidated models leaves no assurance as to the accuracy of the surrogate at points different from the ones used for construction, which may also lead to inappropriate course of action during the optimization process. It is also of interest, to obtain information on the simulation noise contribution to the discrepancy estimator between the surrogate and the noise–free simulation. Note that the noisy effects that contaminate our data points arise primarily due to the Monte–Carlo techniques used for the evaluation of the deterministic integral (2.1). This information can be geared towards minimizing the computational effort associated with the individual simulation data points by controlling the Monte–Carlo sample size. Additional concerns include the effect of the surrogate–based optimization on the reliability and optimality of the obtained design points.

The following sections are dedicated for the first two considerations, namely the discrepancy estimator further referred to as the model prediction error estimator, $u$, and the
noise contribution to \( u \). A more comprehensive coverage of these considerations and the surrogate framework in general is developed and presented in Yeşilyurt [1994].

6.2 The Surrogate Definition

For purpose of simplicity, we shall assume a unidimensional input–output relationship. We take as given a design variable \( p \) (e.g., concentration) in an admissible domain \( \phi \subset \mathcal{R} \), and an output \( s \in \mathcal{R} \) (e.g., \( k_e \)) which is described by the input–output function \( \mathcal{S}(p) \)

\[
\mathcal{S}(p) : \phi \rightarrow \mathcal{R}, \quad \mathcal{S}(p) \in L^\infty(\phi).
\]  

Here \( \mathcal{S}(p) \) is assumed to contain no simulation noise: \( \mathcal{S}(p) \) is the input–output function for both the "continuous" mathematical system and the numerical simulation in the limit of no measurement noise. The noisy simulation output it then a random variable \( r \), expressed as the sum of the noise-free input–output function \( \mathcal{S}(p) \) and a (possibly unbounded) absolutely continuous measurement noise \( w \). Equivalently

\[
r = \mathcal{S}(p) + w.
\]  

The measurement noise random variable \( w \) is characterized by a finite variance \( \sigma_w(p) \), and a scale–parameterized probability density function

\[
f_{w|p}(w|p) = \frac{1}{\sigma_w(p)} g\left(\frac{w}{\sigma_w(p)}\right),
\]  

were the subscript \( w|p \) indicates the probability of \( w \) given \( p \). For simplicity we shall assume that the noise is symmetric, \( g(w) = g(-w) \).

We next introduce the simulation surrogate \( \tilde{\mathcal{S}}(p) \)

\[
\tilde{\mathcal{S}}(p) : \phi \rightarrow \mathcal{R}, \quad \tilde{\mathcal{S}}(p) \in L^\infty(\phi),
\]  

which should first, admit considerably simpler evaluation than the original input–output function, \( \mathcal{S}(p) \), and second, conservatively but effectively exploit priori information to provide a reasonable approximation to \( \mathcal{S}(p) \) over the design domain \( \phi \). Typically, \( \tilde{\mathcal{S}}(p) \) is constructed either mathematically based on simplifying assumptions on the continuous mathematical problem or numerically based on actual measurements and interpolation algorithms. A general discussion on the classes of models can be found in Yeşilyurt & Patera [1993].

The interest now is to find a validation algorithm which yields a precise probabilistic statement regarding the surrogate bias, which, in turn, is necessary to understand how errors in the surrogate will be reflected in the surrogate–based predicted system configuration and performance. This algorithm is presented in the following section.

6.3 Surrogate Validation Statement and Algorithm

We take as given \( 0 < \varepsilon_1 < 1, \ 0 < \varepsilon_2 < 1, \ 0 \leq q \leq 1 \), and define \( \nu_q(p) \) and \( N^v \) as follows

\[
\int_{-\infty}^{-\nu_q(p)} f_{w|p}(w|p) \, dw = q
\]  

102
and
\[ N^v = \frac{\ln\varepsilon_2}{\ln[1 - \varepsilon_1(1 - q)]}, \tag{6.6} \]
respectively. Here \( \varepsilon_1 \) specifies desired validation error which reflects that information from a finite number of points cannot be valid everywhere on \( \phi \); \( \varepsilon_2 \) is related to uncertainty which reflects that information cannot be certain on any region in \( \phi \) except at data points which is "measure" zero; \( \nu_q \) is a noise–cutoff value such that \( \text{Probability}\{\text{noise} > \nu_q\} = q \).

We next define an a strictly positive Bayesian importance function \( \rho(p) \)
\[ \rho(p) : \phi \rightarrow \mathbb{R}_+, \quad \int_{\phi} \rho(p) \, dp = 1, \tag{6.7} \]
which is best interpreted as a prior "density" on the input design variable [Yeşilyurt & Patera 1993]. We then construct the sample of independent identically distributed ordered pairs
\[ \{(p_1, r_1), \ldots, (p_{N^v}, r_{N^v})\} \tag{6.8} \]
where \( (p_j, r_j) \) has joint probability density function
\[ f_{p,w}(p, w) = \rho(p) f_{w|p}(w|p). \tag{6.9} \]
The validation statement then takes the from
\[ \text{Pr}\{\int_{\{p \in \phi | |\mathcal{S}(p) - \mathcal{S}(p)\| \leq u\}} \rho(p) \, dp \geq 1 - \varepsilon_1\} \geq 1 - \varepsilon_2, \tag{6.10} \]
where
\[ u = \max_{j \in \{1, \ldots, N^v\}} \{|r_j - \mathcal{S}(p_j)| + \nu_q(p_j)\} \tag{6.11} \]
is the noisy model prediction error estimator, or the surrogate bias. In (6.10), \( \text{Pr}\{\text{event}\} \) refers to the probability of event, and \( dp \) is a differential element of \( \phi \).

In words, the validation error statement (6.10) indicates that with probability \( \geq 1 - \varepsilon_2 \) the largest discrepancy between the noisy simulation and the surrogate in a sample of size \( N^v(\varepsilon_1, \varepsilon_2) \) is bounded by \( u \) over some subregion of input domain \( \phi \) of \( \rho(p) \)-weighted volume \( \geq 1 - \varepsilon_1 \). The proof of this statement is simply a binomial argument in a tensor–product space comprising both the input and the noise contribution to the simulation result [Yeşilyurt et al. 1993].

The Monte–Carlo algorithm associated with (6.10) and (6.11) is transparent: we sample \( N^v \) values of the design variable over \( \phi \), \( p_j, j = 1, \ldots, N^v; \) we compute a noisy simulation result, \( r_j \) at each design point \( p_j \); we calculate the model prediction error estimator, \( u \) according to (6.11).

### 6.4 Noise Contribution to Model Prediction Error Estimator

This section aims at understanding the effect of the simulation noise on the model prediction error estimator, \( u \). In particular, improved validation (larger \( N^v \)) implies that \( u \) will select larger noise contributions. The validation algorithm, however, is not dependent on estimation of the noise. The latter will serve only to minimize the computational effort or
to sharpen the bound. In this sense, we can proceed less stringently, looking at expectations rather than confidence intervals.

### 6.4.1 Bounded Noise

Bounded noise is a reasonable model for the measurement noise associated with finite element methods for partial differential equations. In this case we have \( f_{w|p}(w|p) = 0 \) for \( |w| > (c_w/2) \sigma_w(p) \), where \( c_w > 0 \) is independent of \( p \). For bounded noise it is interesting to consider \( q = 0 \) which implies from (6.5) and (6.6) that

\[
\nu_q(p) = c_w/2 \sigma_w(p),
\]  

(6.12)

and

\[
N^u = \frac{\ln \varepsilon_2}{\ln[1 - \varepsilon_1]},
\]

(6.13)

respectively. Finally from (6.2) and (6.11) we derive that

\[
u = \max_{j \in \{1, \ldots, N^u\}} \left| r_j - \tilde{S}(p_j) \right| + c_w \sigma_w^{\text{max}}
\]

(6.14)

where \( \sigma_w^{\text{max}} = \max_{p \in \Phi} \sigma_w(p) \).

### 6.4.2 Unbounded Noise

Unbounded noise – the case when the probability density for the noise is potentially nonzero for all values of \( w \in \mathbb{R} \) – is relevant to Monte–Carlo simulations in which the simulation random output is the result of a standard sample mean calculation. To make \( u \) independent of \( \nu_q \), we pick \( q = 1/2 \) so that \( \nu_q = 0 \), and

\[
N^u = \frac{\ln \varepsilon_2}{\ln[1 - \frac{\varepsilon_1}{2}]}.
\]

(6.15)

We measure the effect of the noise by the random variable \( y = \mathcal{Y}(p, w) \)

\[
\mathcal{Y}(p, w) = \max_{j \in \{1, \ldots, N^u\}} \left| S(p_j) + w_j - \tilde{S}(p_j) \right| - \max_{j' \in \{1, \ldots, N^u\}} \left| S(p_{j'}) - \tilde{S}(p_{j'}) \right|,
\]

(6.16)

where \( p = (p_1, \ldots, p_N^u) \) and \( w = (w_1, \ldots, w_N^u) \). Note that, even in the absence of noise, our validation statement does not permit a priori estimation of the deterministic error \( |S(p_j) - \tilde{S}(p_j)| \) without further assumptions on \( S(p) \) and \( \tilde{S}(p) \); the goal here is to understand the incremental effect of noise on \( u \). Denoting

\[
\hat{j} = \arg \max_{j \in \{1, \ldots, N^u\}} \left| S(p_j) + w_j - \tilde{S}(p_j) \right|
\]

(6.17)

we deduce that

\[
y \leq \left| S(p_j) + w_j - \tilde{S}(p_j) \right| - \left| S(p_j) - \tilde{S}(p_j) \right|
\]

\[
\leq \left| w_j \right| \leq z,
\]

(6.18)
where $z = Z(p, w)$ is the random variable

$$z = \max_{j \in \{1, ..., N^v\}} (|w_j|).$$  \hspace{1cm} (6.19)

We note that all the estimates for the expectation of $y$, $E_{p,w}(y)$, with respect to $p$ and $w$ are conditioned on the event of successful validation statement. It follows that [Yeşilyurt, et al. 1993]

$$E_{p,w}(y) \leq \frac{E_{p,w}(z)}{1 - \varepsilon_2}.$$  \hspace{1cm} (6.20)

We now present several a priori density–independent and density–dependent bounds for the expectation of $y$. The priori analysis is made possible through relation (6.20) since all the dependence on the uncharacterized functions $S(p)$ and $\tilde{S}(p)$ has been eliminated from the definition of $z$, equation (6.19).

**A priori Noise Estimates**

**Estimation for Unknown Density**

When only the variance, $\sigma^2_w(p)$ of the noise conditional density $f_{w|p}$ is known, it can be shown [Yeşilyurt, et al. 1993] that

$$E_{p,w}(z) \leq \sqrt{N^v} \sigma_w^p$$  \hspace{1cm} (6.21)

where

$$\sigma_w^p = (\int_\phi \rho(p)\sigma^2_w(p) \, dp)^{1/2}.$$  \hspace{1cm} (6.22)

In cases where $\sigma_w(p)$ is known only at the validation sample points, $\{p_1, ..., p_{N^v}\}$, the integral (6.22) is approximated by an $N^v$–point Monte–Carlo quadrature,

$$\sigma_w^p \approx \left(\frac{1}{N^v} \sum_{j=1}^{N^v} \sigma^2_w(p_j)\right)^{1/2}.$$  \hspace{1cm} (6.23)

In the case where $\sigma_w(p)$ is, in fact, constant, the prefactor in (6.21) is easily improved to $\sqrt{N^v/2}\sqrt{2N^v/(2N^v - 1)}$.

**Estimation for known Density**

In the event that we know (or can approximate) the noise conditional density, $f_{w|p}$, the expectation of $z$ can be evaluated [Yeşilyurt, et al. 1993] as

$$E_{p,w}(z) = n^v \int_0^\infty I_1(v)^{N^v - 1} I_2(v) \, dv,$$  \hspace{1cm} (6.24)

where

$$I_1(v) = \int_\phi \rho(p)(1 - 2F_{w|p}(-v|p)) \, dp,$$  \hspace{1cm} (6.25)

and

$$I_2(v) = \int_\phi \rho(p)(2f_{w|p}(-v|p)) \, dp.$$  \hspace{1cm} (6.26)
$F_{w|p}$ is the cumulative distribution function associated with $f_{w|p}$. In cases where $f_{w|p}$ is known only at the validation sample points, the integrals (6.25) and (6.26) are effected by an $N^v$-point Monte–Carlo quadrature,

$$I_1(v) \approx \left( \frac{1}{N^v} \sum_{j=1}^{N^v} \rho(p) \left( 1 - 2F_{w|p}(-v|p) \right) \right), \quad (6.27)$$

and

$$I_2(v) \approx \left( \frac{1}{N^v} \sum_{j=1}^{N^v} \rho(p) \left( 2f_{w|p}(-v|p) \right) \right). \quad (6.28)$$

**Estimation for Normal Parent**

When the noise density is normal, which is a realistic model in many Monte–Carlo calculations, it can be shown [Yeşilyurt et al. 1993] for the simple situation of uniform $\rho(p)$ and $\sigma_w(p) = \sigma_\nu$ constant, for which (6.24) admits symbolic integration, that if the noise contribution is not to dominate $u$, $\sigma_\nu$ must decrease as $(2\ln N^v)^{-1/2}$. More precisely, it is required that

$$\frac{\sqrt{2} \sigma_\nu}{1 - \epsilon_2} \left( \ln(-\ln \epsilon_2) + \frac{2}{\epsilon_1} \right)^{1/2} \quad (6.29)$$

remains bounded as $\epsilon_1 \to 0$, $\epsilon_2 \to 0$.

We conclude this section by mentioning that it is also possible with a simple posteriori guideline to better reflect the form of $|S(p) - \bar{S}(p)|$ in the estimates of the noise contribution [Yeşilyurt et al. 1993].

### 6.5 Optimization Purposiveness

For sake of completion we will briefly describe the surrogate–based optimization framework. (A detailed discussion is presented in [Yeşilyurt and Patera 1993, Yeşilyurt 1994].) Let us presume that the target value of the output $s$ is prescribed as $\tau$, and that the best design point $p^*$ is defined by

$$p^* = \arg \min_{p \in \Phi} |S(p) - \tau|, \quad (6.30)$$

We next replace $S(p)$ in (6.30) with $\bar{S}(p)$ to construct the approximate design problem

$$\hat{p}^* = \arg \min_{p \in \Phi} |\bar{S}(p) - \tau|, \quad (6.31)$$

It can be readily shown [Yeşilyurt and Patera 1993] that, with probability greater than $1 - \epsilon_2$, for any region $\mathcal{R} \subset \Phi$ for which

$$\int_{\mathcal{R}} \rho(p) \, dp > \epsilon_1, \quad (6.32)$$

there exists a (in fact, many) $\hat{p}^* \in \mathcal{R}$ such that

$$|S(p^*) - \tau| \leq |\bar{S}(\hat{p}^*) - \tau| + \omega \quad (6.33)$$
where

$$\omega = u + \delta_R$$

(6.34)

is the predictability gap and

$$\delta_R = \max_{\bar{p} \in \mathcal{R}} |\hat{S}(p) - \hat{S}(\bar{p}^*)|.$$ 

(6.35)

In words, (6.33) indicates that for sufficiently accurate \(\hat{S}(p)\) (small \(u\)) and sufficiently small \(\varepsilon_1\) (small predictability gap), there exists a design point \(\bar{p}^*\) near \(\bar{p}^*\) at which actual system performance – as measured by \(|S(\bar{p}^*) - \tau|\) – is close to the optimal system performance predicted by the surrogate. If it is further assumed that \(|S(p) - \tau|\) and \(|\hat{S}(p) - \tau|\) are quasi-convex, one can construct, based solely on the surrogate \(\hat{S}(p)\), a random region \(\mathcal{K}\) that shrinks to \(\bar{p}^*\) as \(u \to 0\), \(\varepsilon_1 \to 0\), and that with probability greater than \(1 - \varepsilon_2\), contains the actual design point, \(p^*\).
Chapter 7

Validated Surrogate for the Effective Conductivity

In this chapter we validate a structure–independent model for the prediction of the effective conductivity as a function of the inclusion concentration. We then briefly consider applying the optimization purposive analysis summarized in Section 6.5 of Chapter 6.

7.1 Formulation

Calculation of the scalar supercell effective conductivity of a random fibrous composite, $k_e(c, \lambda)$, based on the individual configuration calculations, $k(\{y\}_N, c, \lambda)$, has been discussed in detail, in Sections 2.1 and 2.2 of Chapter 2. In particular, $k_e(c, \lambda)$, is calculated via integral (2.1):

$$k_e(c, \lambda) = \int_{(0,\lambda) \times (0,\lambda)} k(\{y\}_N, c, \lambda) f(\{y\}_N | \{y\}_N; c, \lambda) \, dy,$$

where the inclusion joint probability density function, $f(\{y\}_N | \{y\}_N; c, \lambda)$ is given by (2.2)–(2.5).

For the purpose of this validation exercise we will not consider the limiting process in $\lambda$, that is $\Lambda(c)$, at which transition to the truly random medium is achieved (Section 2.1); rather we shall compute the supercell effective conductivity for a particular $\lambda$, specifically $\lambda_0 = 6$, which is sufficiently large that size effects are reasonably small [Cruz & Patera 1993]. (Note that, in practice, for each concentration $c$ we choose that $\lambda$ closest to $\lambda_0$ for which $N$ is integral.)

Next, as a surrogate for the effective conductivity, we adopt the best possible structure–independent upper bound, $\tilde{k}_e(c)$ of Hashin [1970]

$$\tilde{k}_e(c) = \frac{1 - c}{1 + c}.$$  \hspace{1cm} (7.1)

As discussed in Chapter 1, the method of bounds, when based on low–order moments characterizing the microstructure, yields simple, easily applied results, but the bounds are often not sufficiently tight for a priori application. In fact, for our insulating fibrous medium, the structure–independent lower bound is, perforce, zero. The merit of our surrogate approach, however, is to exploit the complementary attributes of structure–dependent and
structure-independent approaches by adopting the latter as a surrogate for the former; if the surrogate proves an accurate estimate for $k_c(c, \lambda_0)$ for the presumed physically relevant inclusion density function, then: (i) we will have expanded the prognostic range of the upper bound; (ii) we have obtained a simple, robust, expression for the effective conductivity for the particular random media of interest.

### 7.2 Simulation Subproblems

In Chapter 4 a detailed description of the numerical treatment of the configuration effective conductivity calculation is presented. Figure 7-1 shows a typical representative realization comprising 26,000 finite element degrees-of-freedom which requires 2 minutes of computation on 16 nodes of the Intel iPSC/860 hypercube. It is important to mention that our nip-element procedure has been evoked only in extreme situations in which mesh generation was no longer possible (see example 2 of Section 5.2.2). In virtually all the calculations, the nip-element, discretization, and incomplete-iteration errors are sufficiently small that the configuration conductivity simulation solution is effectively exact.

We now turn to the Monte-Carlo component of the algorithm. For any particular concentration, $c$, we approximate the integral (2.1) as a sample mean

$$\overline{K} = \frac{1}{N_r} \sum_{i=1}^{N_r} K_i,$$

(7.2)
Figure 7-2: Acceptance–rejection method: a) shaded area represents the uniform conditional \( f_{y_2|y_1}(y_2|y_1) \), and b) shaded area represents the uniform conditional \( f_{y_3|y_1, y_2}(y_3|y_1, y_2) \).

where \( K_i = k(\{Y\}_N, c, \lambda_0), \ i = 1, \ldots, N_r. \) The \( N_r \) microstructure realizations are generated by an acceptance–rejection Monte–Carlo sampling procedure [Rubinstein 1981, Cruz 1993 pp. 98-101] based on the conditional probability density function representation given by (2.2)–(2.5) (see Figure 7-2 for illustration). The details of the acceptance–rejection algorithm and information on the actual random number generator used can be found in Appendix F. We approximate the variance of \( K, \sigma_K^2(c, \lambda_0) \), by the sample variance,

\[
\sigma_K^2(c, \lambda_0) = \frac{1}{N_r - 1} \sum_{i=1}^{N_r} (K_i - \bar{K})^2. \tag{7.3}
\]

Finally, we assume that \( N_r \) is sufficiently large that the density of the random variable

\[
V = \frac{\sqrt{N_r} [\bar{K} - k_e(c, \lambda_0)]}{\sigma_K} \tag{7.4}
\]

is approximately normal with zero mean and unity variance [Mood, Graybill, and Boes 1974].

Before turning to the validation results, we make several final remarks. First, our assumptions on \( V \) in (7.4) in no way affect our validation statement (6.10), nor, in fact, the density–independent noise bound (6.23). Second, although the measurement noise is, in fact, bounded, the range of \( \bar{K} \) is large compared to the standard deviation of \( \bar{K} \), and thus the unbounded assumption is more profitable. Third, \( \sigma_K(c, \lambda_0) \) will decrease with increasing \( \lambda \) due to spatial averaging effects. It is however, more economical to effect variance reduction by increasing \( N_r \) than by increasing \( \lambda \), as the former, first requires less
memory, and second, leads to better conditioned problems. Fourth, for computational efficiency, \( N_r \) should be chosen so as to control – but not unnecessarily eliminate – the noise contribution to the model prediction error. We select, based on earlier calculations [Cruz & Patera 1993] \( N_r = 20 \), which reflects the relatively small variance of \( K \). Fifth, and finally, we point out the relative advantage of the parallel approach by noting that \( N_r = 20 \) calculations at (relatively time–consuming) concentration, \( c = 0.492 \), require approximately 27 minutes and costs $10 on 16 processors of Intel iPSC/860 hypercube. A similar calculation on a workstation would require significantly more time. (The time to complete \( N^v N_r \) calculations would be very lengthy on a workstation.)

### 7.3 Surrogate Validation Results

We first identify our problem in terms of the general variables of the surrogate framework introduced in Chapter 6:

\[
\begin{align*}
    p & \rightarrow c \\
    \phi & \rightarrow [0.05, 0.5] \\
    \rho(p) & \rightarrow 1/45 \text{ (uniform)} \\
    S(p) & \rightarrow k_e(c, \lambda_0) \\
    \tilde{S}(p) & \rightarrow \tilde{k}_e(c) = \frac{1 - c}{1 + c} \\
    r & \rightarrow \bar{K} \\
    \sigma_w(p) & \rightarrow \approx \frac{\tilde{\sigma}_K(c, \lambda_0)}{\sqrt{N_r}} \\
    g(v) & \rightarrow \approx \sqrt{\frac{1}{2\pi}} e^{-v^2/2}
\end{align*}
\]

(Although the most "interesting" behavior of the effective conductivity occurs near maximum packing, \( c \approx 0.82 \) [Berryman 1983]; we do not consider this case here, as our joint probability density function and associated Monte–Carlo sampling procedure is prohibitively inefficient for high concentrations. Note, however, that our nip–element procedure can handle efficiently and accurately the severe geometrical stiffness characteristic of random high–concentration realizations.) Finally, we specify \( \varepsilon_1 = .1, \varepsilon_2 = .1 \), and thus, from (6.6), \( N^v = 45 \). (The validation row data points are listed in Section F.2 of Appendix F.)

Performing the validation procedure of Section 6.3 in Chapter 6, we find for the model prediction error estimate \( u = 0.0574 \). In words: with confidence level greater than 99%, \(|k_e(c, \lambda_0) - \tilde{k}_e(c)|\) is less than 0.0574 over more than 99% of the concentration range [0.05, 0.5]. In Figure 7.3 we plot the surrogate–simulation discrepancy, \(|\bar{K} - \tilde{k}_e(c_j)|\), and the estimated noise standard deviation, \( \tilde{\sigma}_K(c_j, \lambda_0) \sqrt{N_r} \), at the validation input points, \( c_j, j = 1, \ldots, N^v \). Our bound procedures for the noise contribution yields the following:

Estimation for unknown density

\[ E_{p,w}(u) \leq 0.0273 \] (7.5)

Estimation for known density

\[ E_{p,w}(u) \leq 0.0110 \] (7.6)
Figure 7-3: Surrogate-simulation discrepancy, ($|\bar{K}_j - \bar{k}_e(c_j)|$ — ○), and estimated noise standard deviation, ($\tilde{\sigma}_K(c_j, \lambda_0)/\sqrt{N_r}$ — □), at validation input points, $c_j$, $j = 1, \ldots, N^v$.

the latter suggests that the noise contribution to the model prediction error estimator is, relatively, not too large.

7.4 Surrogate-Based Optimization

We briefly illustrate the purposive analysis summarized in Section 6.5 of Chapter 6. (For a detailed presentation see Yeşilyurt [1994].) We take the target value for the effective conductivity to be $\tau = 0.5$, for which our surrogate predicts the design point $\bar{c}^* = 0.333$. Then from (6.33), (6.34), and (6.332) we can construct design intervals, $\mathcal{R}$, which contain concentration values, $c$, for which the actual effective conductivity, $k_e(c, \lambda_0)$ is within a known tolerance of $\tau$; for example, with confidence level greater than $1 - \epsilon_2 = 0.9$, on over half the interval $c \in \mathcal{R} = [0.288, 0.378]$, the actual effective conductivity $k_e(c, \lambda_0)$ is within $\omega = 0.110$ ($\delta_{\mathcal{R}} = 0.0528$) of the target value $\tau = 0.5$. 


Chapter 8

Creeping–Flow Permeability Results

In this chapter we evaluate the well-known Carman–Kozeny equation as a potential surrogate for the creeping–flow permeability of fibrous random media over a wide range of inclusion concentrations. A simple empirical model for the dependence of the permeability on the inclusion concentration is also proposed.

8.1 Simulation Subproblems

We are interested in the permeability behavior over a wide range of concentrations $\phi = [1, 6]$. The supercell permeability, $\kappa_e(c, \lambda)$, is calculated for 25 deterministic concentration values of $c_j = 1 + j \cdot 0.02$, $j = 0, \ldots, 24$. (The evaluation raw data points are listed in Section F.3 of Appendix F.)

The supercell permeability, $\kappa_e(c, \lambda)$, is calculated for the particular $\lambda$, specifically $\lambda_0 = 7$ which is sufficiently large that size effects are reasonably small [Cruz & Patera 1993]. Solution of the saddle Stokes problem by the nested Uzawa procedure (Section 4.3.4) requires, typically, $O(70)$ more computational effort than a comparable elliptic conduction calculation comprising the same degrees of freedom per velocity component. Our nip–element procedure is no longer a “luxury” but rather a necessity, dictated by the dramatic negative effects of the geometrical stiffness on the computation time (Section 5.3.2). Our presented data for the sample mean and variance at virtually all the concentrations are averages of the lower and upper bound results. In order to minimize the bound gap error, a stringent $\alpha_c$, $\alpha_c = 0.05$, has been defined. The worst obtained nondimensional relative bound gap error (see equation (5.5) in Chapter 5) was for the highest concentration considered, $c = 0.58$, and amounted to 0.0520. (Note that this value is based on the sample mean lower and upper bounds and not individual configuration calculations which can possibly yield a higher value.) This bound gap error, although small, is not on the order of discretization and incomplete–iteration errors which are sufficiently small and can be safely neglected. If rigorous validation statement is to be made, one conservative way to account for this error, is to simply add it to the model prediction error estimator, $u$. (In this case, $u$ need be redefined as a nondimensional relative error rather than a dimensional absolute error.)

Finally, for the sample size, we choose $N_r = 20$ which, based on the effective conductivity results of Chapter 7, is expected to reasonably control the noise contribution. However, larger noise effects have been reported for the permeability problem relative to the con-
ductivity problem [Cruz 1993] and the noise is expected to be significantly higher. The corresponding Monte–Carlo components of the algorithm are analogous to their effective conductivity counterparts. (Here we compute a lower–bound and an upper–bound sample mean and variance by redefining \( K_i = \kappa_{LB}(\{Y\}_N, c, \lambda_0) \) and \( K_i = \kappa_{UB}(\{Y\}_N, c, \lambda_0) \) respectively, in the mean (7.2), and sample variance (7.3) calculations. Our reported mean and variance at each concentration in then the average of the lower–bound and upper–bound sample mean and variance.) For \( N_p = 20 \) realizations, the most time–consuming, \( c = 0.58 \), calculation require 42 hours of computation and costs roughly $600 on 16 processors at relatively good parallel performance (\( \eta_p \approx 0.9\% \) and 80 MFLOPS). This calculation is simply prohibitive on a workstation.

8.2 Carman–Kozeny Model

In this section we consider the well–known Carman–Kozeny Model, further referred to simply as the Kozeny equation, as a potential surrogate in the concentration range of interest. The simplest form of the Kozeny equation takes the form

\[
\kappa = \frac{\epsilon m^2}{C}
\]

(8.1)

where \( \epsilon \) is the porosity = \( 1 - c \); \( m \) is the mean hydraulic radius defined as the ratio of free volume to wetted area, which is, for a fibrous medium, equal to

\[
m = \frac{d \epsilon}{4(1 - \epsilon)}
\]

(8.2)

and \( C \) is a constant supposedly independent of \( \epsilon \). the Kozeny equation is discussed in detail in many recent text books [Scheidegger 1974, Probstein 1989, Dullien 1992] and we shall not elaborate on its derivation. Rather we are interested in testing the validity of the model for 2-D fibrous media over the range of concentrations [.1,.6], or equivalently, the hypothesis of a “nearly” constant \( C \).

For beds of randomly packed uniform spheres, Carman [1937] showed experimentally that the Kozeny equation, within the range, \( .26 \leq \epsilon \leq .48 \) gives excellent correlation with a Kozeny constant \( C \approx 4.8 \). Carman has also found that the Kozeny equation is inapplicable when changes in porosity are caused by extreme variations in particle size, or when the bed is composed of highly irregular particles. On the other hand, there is a large volume of data for 3-D beds consisting of a variety of nonspherical particles which indicate that a value of \( C \approx 5.0 \) independent of shape and porosity in the range \( .26 \leq \epsilon \leq .8 \) yields a good agreement of the Kozeny model (see, for example, Happel and Brenner [1983]).

The author is not aware of any data for 2-D random fibrous beds (flow perpendicular to randomly cooriented fibrous), with microstructure that could be adequately modeled by a random sequential addition process, the one that has been employed to generate the realizations in this thesis. However, there are data for 3-D random fibrous media, that is with random orientation of fibrous, involving mostly air as the working fluid and porosities larger then 0.7 [Kyan, Wasan and Kintner 1970, Lord 1955, Brown 1950]. These data indicate that, in the porosity range \( .7 \leq \epsilon \leq .9 \), the Kozeny equation with a value of \( C \approx 5.0 \) is also a valid model for 3-D random fibrous media.

There has been some earlier theoretical work for prediction of the permeability of 3-D random fibrous beds based on knowledge of the transverse and parallel permeability of
simplified cell models [Happel 1959, Kuwabara 1959, Spielman and Goren 1968]. The models consider an isolated fiber enclosed within an imaginary cell whose dimensions are chosen so that the porosity within the cell boundary equals the bulk porosity of the medium. The analysis proceeds by assuming that the influence of the surrounding porous medium upon the isolated fiber can be lumped into appropriate boundary condition on the cell boundaries. For example, in the models of Happel and Kuwabara the cell is chosen, for mathematical convenience, to be a concentric circle enclosing the fiber, with the assumption of no shear at the outer boundary in the former model and no vorticity at the outer boundary in the latter model. The flowfield equations can then be solved analytically to obtain an expression relating the drag to the bulk velocity, and by comparing this expression to the Darcy’s law, an expression for the permeability is obtained for both parallel and perpendicular flow configurations. The 3-D random fibrous bed permeability is then obtained by averaging the drag over all possible orientations of the fiber (a simple weighted sum of the two normal directions), and the models do predict a Kozeny constant of $C \approx 5$ in the porosity range $0.26 \leq \epsilon \leq 0.8$.

8.2.1 Results and Discussion

In Figure 8-1 we plot, on a semi-log coordinate system, values of $C_i = C$, versus the concentration points $c_i$, where $C$ is calculated from (8.1) using our simulation permeability values. We also plot the error bars which represent one standard deviation plus the relative bound gap error due to use of nips. (It is important to note here that the way we calculated the $C_i$ based on the sample mean permeability values does not give us directly information on the standard deviations associated with the $C_i$. However, due to the inversely linear relation of the permeability and the Kozeny constant (for a fixed porosity) one can easily show that the mean–normalized standard deviations are, in fact, equivalent ($\frac{\Delta C}{C} = \frac{\Delta K}{K}$).) On the same figure we also plot the Kozeny constant predicted form the cell models of Happel [1959] and Kuwabara [1959] for flow perpendicular to cylinders. We make several remarks on the figure:

First, the noise contribution to the data is certainly high, as anticipated earlier. Similar high noise levels have also been reported by Cruz [1993], but it is not clear why the noise is significantly pronounced in the creeping–flow problem relative to the effective conductivity problem. Furthermore, the higher concentrations possess a larger number of inclusions for the specified $\lambda_0$ and thus, due to spatial averaging effects, one would expect smaller associated variances, however this is not observed here.

Second, the plot shows that, up to a concentration of 0.44 ($\epsilon \geq 0.56$), $C$ can be accurately modeled as a constant. For concentrations higher than 0.44 ($\epsilon < 0.56$), $C$ doubles in magnitude and a constant model is not valid. In the range $c \in [0.1, 0.44]$, in which the Kozeny equation appears to be a valid model, the least square fit (equivalent here to the simple algebraic average) predicts a value of $C = 10.167$, and the Kozeny equation takes the form

$$\kappa = \frac{\epsilon^3}{10.167 \times 16(1-\epsilon)^2}$$  (8.3)

Based on the average value $C = 10.167$, we plot in Figure 8-2 the relative error $E$ defined as

$$E = \left|\frac{K - \tilde{K}}{\tilde{K}}\right| \times 100$$  (8.4)
Figure 8-1: Predicted values of the Kozeny constant based on the 2-D fibrous medium permeability calculations. The error bars represent one standard deviation plus the bound gap error. Also shown the Kozeny constant predicted form the cell models of Happel [1959] and Kuwabara [1959] for flow perpendicular to cylinders.

at the concentrations \( c \) in the range .1 to .44, where \( \bar{K} = \kappa_e(c, \lambda_0) \) and \( \bar{K} \) is the predicted permeability value from the Kozeny equation. The maximum relative error, \( E \approx 19\% \), occurs at the lowest concentration, and, for the most part, is less than 14\%. This indicate that the Kozeny equation is, practically, a valid model for 2-D random fibrous porous media in the concentration range [.1, .44].

Third, the behavior of our 2-D fibrous media data at the lower porosities deviates markedly from the behavior of 3-D porous beds. In the latter, the Kozeny constant behaves very much like the curves of Happel and Kuwabara which are shown on the plot with approximately a constant value of \( C = 5 \), whereas in the former the predicted values of the Kozeny constant appears to increase indefinitely as the maximum packing is approached. This behavior is a pure geometrical artifact that is specific only to 2-D porous beds, that is, beds with invariant cross section in the direction parallel to the flow. This behavior at lower porosities is best elucidated by considering regular arrays: as the maximum packing is approached, a percolation mechanism for the flow prevails until the flow is totally blocked at maximum packing at which the permeability tends to zero and consequently the Kozeny constant tends to infinity. This physical picture
can be proven rigorously based on the available asymptotic expression for the drag due to flow through square and hexagonal arrays near the maximum packing [Sangani and Acrivos 1982b]. The same behavior applies to the case of 2-D random arrays although the author is not aware of any data or asymptotic expressions, beside the current data, near the maximum packing that support this claim. In 3-D beds, a path for the flow is always guaranteed even at the maximum packing and the drag will not increase indefinitely. Here it is important to note that the agreement of the 2-D cell models of Happel and Kuwabara (which are inconsistent with our 2-D data) with the behavior of true 3-D beds is probably due to the no shear or no vorticity assumptions which lower the drag and, in turn, the Kozeny constant.

Fourth, at the higher porosities, \( \epsilon \to 1 \), for 3-D beds, the permeability is known to vary as \( 1/\epsilon \) [Batchelor 1972]. Using this fact in the Kozeny equation, it is easily shown that the Kozeny constant varies also as \( 1/\epsilon \) [Ethier 1982, pp. 35–38] and the model is not valid in that limit. The cell models of Happel and Kuwabara are consistent with this fact as can be seen in Figure 8-1. For 2-D fibrous media similar behavior can also be shown to obtain based on the asymptotic expressions for the drag available for regular arrays [Sangani and Acrivos 1982b], although the dependence on the concentration is different: for square arrays, it can be shown that the Kozeny constant varies as \(-[c^2 \ln c]^{-1}\). Unfortunately, we do not have data in that limit to demonstrate this behavior.
It is interesting to consider the applicability of the Kozeny equation to model the permeability of regular arrays as well. In Figure 8-3 we compare the permeability of the square array and the corresponding predicted Kozeny constant to those of the random arrays. We make two observations: first, a crossover between the behaviors of the regular and random arrays occurs at approximately $c = .3$ after which the permeability of the random array becomes lower than for the regular array. This observed crossover, which is also reported by Cruz [1993], may result from a percolation phenomenon in the random arrays. At low concentrations, the flow through a random array encounters "large" and "small" passages at any given cross section: flow resistance combines in parallel, and thus the permeability of random arrays will be larger than the permeability of regular arrays. At high concentrations, flow conduits become scarcer, and the "small" passages become a bottleneck: resistance then combines in series and the permeability of random arrays is, thus relatively smaller. Second, the Kozeny equation (8.3) can satisfactory model the permeability of the square array over the wide range of concentrations $[.14, .5]$. (In fact, the square array data in the range $[.14, .5]$ predicts an average value $C = 10.104$. ) The maximum relative error as defined by (8.4), for the square array is 16.7%, which occurs at the lower concentration $c = .14$. For the higher concentrations the relative error is less than 10%.

![Graph showing permeability comparison](image)

Figure 8-3: Comparison between the permeability of the square and random arrays (upper plot), and the associated predicted values of the Kozeny constant (lower plot).

We conclude this section by suggesting a qualitative explanation for the higher values of the predicted Kozeny constant, $O(10)$, for the 2-D random arrays of cylinders compared to the value of 5 associated with 3-D random porous media. This can be attributed to the hypothesis that the array of cylinders provide a larger surface area in contact with non-stagnant fluid than do comparable random porous media, which, in turn, leads to larger drag and hence lower permeability. Note that we do not claim that a fibrous porous medium
possesses a total larger surface area than, say, a comparable random bed of spheres at the same volume, which is, in fact, not true. (The specific surface area of a cylinder is $4/d$ which is less than $6/d$ associated with a sphere.) What we claim here, is that, due to the less complex interaction between the fibers and hence the smaller likelihood of forming stagnation hidden (trapping) regions, there will be relatively more energetic (non-stagnant) fluid in contact with available surface area in the case of a 2-D fibrous medium.

8.3 Simple Exponential Model for Random Fibrous Media

In this section we construct a simple empirical model for the behavior of the permeability of random arrays throughout the entire concentration range $[.1,.6]$. Figure 8-4 shows the behavior of the permeability data of the 2-D random arrays as a function of concentrations, along with error bars representing one standard deviation plus the 5% bound gap error. The plot suggests an exponential decay with increased concentration of the form

$$\tilde{\kappa}_e(c) = \alpha_0 e^{-\beta_0 c} \tag{8.5}$$

where $\alpha_0, \beta_0$ are constant to be determined. We emphasize the blackbox nature of our modeling approach which is not intended to reflect any prior information on the anticipated form of behavior of the permeability with concentration. We also emphasize that our simple model is only applicable in the indicated range of concentrations; in the limit $c \to c_{\text{max}}$, the model will not predict a zero value of the permeability which is not consistent with the physics of the problem.

The sharp diminishing of permeability with increased inclusion concentration is a well-known fact on account of many factors. The most prominent of which is the increased drag due to increased fluid-solid interaction and the likelihood of percolation-type flow mechanism. This sharp decrease in the permeability can also be interpreted in terms of simple channel-flow considerations. The average velocity in plane channel flow is proportional to the square of the plate separation; in porous medium, the average nearest-neighbor inter-particle spacing, $O(c^{-1/2})$, plays the role of the channel separation. Predicting the form of dependence on the concentration is not a trivial task and has been proposed analytically only in the limit of dilute concentrations [Batchelor 1972, Childress 1972]. Here we will not attempt to consolidate our empirical model with any analytical analysis. (The author is not aware of a structure-independent bound for the permeability analogous to the Hashin’s bound [1970] used as a surrogate for the effective conductivity in the previous chapter.)

The constants $\alpha_0$ and $\beta_0$ are determined from a standard-deviation-weighted least square fit to the data, [see Press et al. 1992 pp. 655-660, Strang 1986 pp. 137-146]. This fit will yield, for the case of normally distributed errors, the maximum likelihood parameter estimation. It is important here to realize that the permeability varies over 3 decades and the standard deviations associated with the sample means must be normalized by the sample means prior to incorporation in the weighted least square fit. Performing the fit we find for the parameters $\alpha_0 = 1.171$ and $\beta_0 = 12.736$. Our simple model can then be expressed as

$$\tilde{\kappa}_e(c) = 1.171 e^{-12.736 c} \tag{8.6}$$

In Figure 8-5 we plots the surrogate as given by equation (8.6) and the permeability
Figure 8-4: The behavior of the permeability as a function of concentrations. The error bars represent one standard deviation plus the bound gap error. The semi-log plot suggests a simple exponential decay.

points $\bar{K}_j$. The corresponding relative error $E_e$ defined as

$$E_e = \left(\frac{|\bar{K} - \bar{\kappa}(c)|}{\bar{\kappa}(c)}\right) \times 100$$  \hspace{1cm} (8.7)

is plotted in Figure 8-6 which indicate that for virtually all the data points the relative error is less that 15% except for one data point at concentration $c = .12$ where the relative error is approximately 31%. The latter data point, in fact, has a larger variance relative to the neighboring points and hence, is relatively less reliable.
Figure 8-5: A plot showing the permeability points $\bar{K}_j$ and the surrogate equation.

Figure 8-6: Relative nondimensional error between $\kappa_e(c, \lambda_0)$ and the values predicted by the exponential model given in (8.6).
Chapter 9

Inertial Porous–Media Flow Results and Future Work

In this last chapter we present the limited results collected for the permeability of both regular and random arrays under conditions of moderate–Reynolds–number flow. The theme of the presentation is not to provide a comprehensive analysis on the subject, but rather to illustrate the capability of the methodology in treating nonlinear problems; to shed light on some of the aspects characteristic of nonlinear problems like oscillatory response to non-oscillatory driving force; to discuss some interesting (dangerous) pure numerical effects induced by the convection–diffusion stiffness; to demonstrate the decrease of the permeability with increased Reynolds number as predicted in Section 2.4 of Chapter 2; to compare our data against the limited available experimental and numerical studies; to pave the way for future systematic investigation of the physics of inertial flows in true granular beds.

9.1 Regular Arrays

Regular arrays share many qualitative aspects of the physics of the more complex random arrays. Their simple geometries allow for efficient numerical treatment. Analysis of moderate and large Reynolds number flow through regular arrays is valuable for many engineering applications, perhaps the most relevant of which (for our microstructure and incompressibility condition) is convective heat exchangers [Saunders 1988]. We shall treat here two types of periodic arrangement: the square and the hexagonal arrays.

9.1.1 Square Array Results

For the sake of making a comparison with the limited available numerical data of Edwards et al. [1989], Eidsath et al. [1983], and the experimental data of Bergelin et al. [1952], we restrict our selected concentrations and Reynolds numbers mainly to the comprehensive set reported in Edwards et al. [1989]. These concentrations and Reynolds numbers are

- \(c_j \in \{2.2, 3, 4, 5, 6\}\)
- \(Re_{<u_1>} \in \{0, 20, 40, 95, 180\}\)

That is a total of 25 runs. \(Re_{<u_1>}\) is based on the average velocity and inclusion diameter as defined in (2.45). For clarity we shall refer to \(Re_{<u_1>}\) simply as \(Re\). Since our solution strategy does not permit defining \(Re\) a priori, the actual \(Re\) is calculated a posteriori based
on equation (2.46). Here we can only approximate the desired $Re$ based on estimation of $Re_p$ according to our nondimensionalization (see Section 2.4 of Chapter 2). Our estimation of the latter is based on the permeability results provided in Edwards et al. [1989].

Figure 9-1 shows our permeability results (marked with o's) for the concentration $c = .2$ at approximately the Reynolds numbers listed above and 4 additional values. On the same Figure is also plotted the data reported by Edwards et al. [1989] (marked with x's). The data for the remaining set of concentrations .3, .4,.5 and .6 are plotted in Figure 9-2 along with those reported by Edwards et al. [1989]. (The data points are listed in Table F.4.) Figures 9-1 and 9-2 suggest, generally, a reasonable, if not accurate for certain data points, agreement between our calculations and those of Edwards et al. [1989]. However, there are conspicuous discrepancies that deserve to be investigated to establish the possible sources of errors and, consequently, the relative reliability of the two calculations. We shall therefore discuss each concentration individually starting with Figure 9-1.

![Figure 9-1: Square array permeability results at concentration c = .2 for several Reynolds numbers.](image)

Referring to Figure 9-1 we first note the relatively large discrepancy at the Stokes solution ($Re = 0$). Comparing the two calculations to the exact solution reported by Sangani and Acirivos [1982a] we find that our Stokes solution is identical to 5 decimal places, whereas Edwards et al. larger solution deviates by 2.5%. This indicates an underresolved solution due possibly to large iteration or/and discretization errors. In the following we will try to point out the sources of errors more accurately.

Edwards et al. use a penalty finite element solution [Fried 1979] based on the steady formulation of the cell problem. No iteration error control is reported for the Newton iteration scheme employed for the solution of the resulting system of nonlinear algebraic equations. They consider only one fourth of the square cell for the Stokes problem, and
Figure 9-2: Square array permeability results at 4 concentrations for several Reynolds numbers. The o's indicate current data; the x's are data reported in Edwards et al. [1989].
half the cell for the finite-Reynolds-number runs, with symmetry assumptions on both the velocity and pressure across the $y_1$ and $y_2$ axes in the former, and across the $y_1$ axis in the latter. (The symmetry assumptions in the case of finite-Reynolds-number runs, although consistent with a steady solution strategy, is, in fact, incorrect if the flow is unsteady which has been for the higher Reynolds numbers. We shall elaborate on this point shortly.) Edwards et al. report using typically 400 isoparametric biquadratic square finite elements in half the cell, which would imply that $O(200)$ elements have been used for the Stokes problem.

We use a mixed finite element solution based on the unsteady formulation as discussed in Section 4.3.4 in Chapter 4. We also impose a stringent error control on both the velocity and pressure for the steady runs. More precisely, we require that the $L_2$-norm of the residual in both the velocity and pressure be less than $1 \times 10^{-6}$ in virtually all the steady runs reported (see Table F.4). In our calculations we consider the complete square array cell with periodic boundary conditions for both the Stokes and inertial runs, and use about 1600 2nd-order isoparametric triangular finite elements, which is twice the number of elements reported by Edwards et al. (Note, the above is also true for the hexagonal array as well.)

For the steady runs, which we found to be stable solutions for Reynolds numbers, $Re$, less than approximately 150, the source of error between our data points and theirs, which is consistently higher than ours (see Figure 9-1), is thus, more likely due to lack of resolution in Edwards et al. runs given the Stokes discrepancy (assuming that a comparable iteration error control is used). For Reynolds numbers higher than approximately 150 the flow was found to be unsteady. The unsteadiness was in the form of steady periodic oscillations. This, in turn, results in larger dissipation and thus lower permeability which explains the sudden drop in our temporally-averaged permeability values for the last two largest Reynolds numbers. The value reported by Edwards et al. at $Re = 180$ is incorrect because they solve the problem as steady when, in fact, it is unsteady and hence, their solution is on another (unstable) branch. Note that the $y_1$ symmetry assumption on the velocity and pressure due to consideration of half the cell only, although consistent with Edwards et al. steady approach, is certainly erroneous for unsteady oscillatory flow. This is evident in Figures 9-3 and 9-4 which contour the pressure and the $v_1$-component of the velocity at a particular instant for $Re = 168.4$.

Figure 9-5 plots the history curves of the $v_1$-component of the velocity at a selected point for the two unsteady calculations at Reynolds numbers $Re = 150.2$ and $Re = 168.4$ respectively. We first note that the onset of unsteadiness at about $Re = 150$ is consistent with limited previous visual studies in regular arrays of spheres [Wenger, Karabelas and Hanratty 1971, Dybbs and Edwards 1982] which concluded that the steady flow assumption requires that the Reynolds number be less than about 150. (At $Re = 138.3$ no oscillations were observed up to 30 nondimensional time units, which indicates the flow is steady at this Reynolds number.) Second, we also observe the expected increase in the amplitude of oscillations at the higher Reynolds number which is also accompanied with a slight increase of measured frequency, $f$, $f = \frac{2\pi}{T}$, where $T$ is the period of oscillations. We measure for the Strouhal number, $St$, defined as

$$St = \frac{f L_f}{<u_1>}$$

values of 4.046 and 4.021 respectively for $Re = 150.2$ and $Re = 168.4$, where the flow length scale, $L_f$, is defined as $\lambda - d$ for the square array (recall $\lambda$ is the cell size). The unsteadiness phenomenon here in periodic arrays is fundamentally different form the periodic wake
Figure 9-3: An instantaneous unsteady pressure contours for a the square array at concentration .2, for Reynolds number, \( Re = 168.4 \). Clearly, the assumption of symmetry across the center line is incorrect at this Reynolds number.

Figure 9-4: The corresponding \( u_1 \) contours for a the square array shown in Figure 9-3. Again, the assumption of symmetry across the center line is incorrect at this Reynolds number.
Figure 9-5: History plots of the $\gamma_1$-component of the velocity at Reynolds numbers, $Re = 150.2$, and $Re = 168.4$ respectively, for the square array at concentration $0.2$.

(vortex shedding) phenomenon behind bluff cylinders. In the latter, the Strouhal number is experimentally shown [Roshko 1954] to increase (albeit slightly) with increasing Reynolds numbers until the vortex shedding disappears in the completely turbulent flow regime. (The corresponding Strouhal number for the vortex shedding at $Re = 168.4$ is .19 [Roshko 1954] compared to 4.096 for the oscillatory flow in the periodic square array at $c = .2$.) Although here we notice a slight drop in Strouhal number with increased Reynolds number, our two adjacent data points are insufficient to withdraw any conclusion on the behavior of Strouhal number with Reynolds numbers in periodic arrays (which is beyond the scope of this study), and additional investigation is required.

The unsteadiness raises an important question concerning the effect of the lumped mass matrix employed in place of the actual mass matrix for reasons of computational savings. (The lumped mass matrix has virtually no effect on steady time–independent solutions (see Section 4.3.4).) To investigate this effect, we re-compute the flow for the case $Re_p = 57.2$ (yields $Re = 168.4$ with the lumped mass method) using the exact mass matrix. In Figure 9-6 we plot the time history of the $u_1$-component of the velocity resulting from both strategies at an identical selected history point. The result indicate that the lumped mass matrix has a slight damping effect as we observe the slight increase in the oscillations amplitude when using the exact mass matrix. This effect imply that the lumped mass matrix which is equivalent to a quadrature based on the collocation points and a set of weights underestimate the integrand or in finite–difference context, introduces additional numerical diffusion into the system of equations. The exact mass matrix method will thus yield a lower permeability due to the higher–amplitude oscillations and thus higher dissipation. The actual reduction in the permeability however, was found to be inappreciable (the relative change is less than .33%). This slight improvement in the measurement does not warrant the $O(5) - O(10)$
Figure 9-6: Time history plots for the $u_1$-component of the velocity at $Re_p = 57.2$ using both the lumped mass matrix (upper) and the exact mass matrix (lower) for the square array at concentration $c = .2$. 
Table 9.1: Maximum reported values for the absolute divergence as a function of the mesh density for two square arrays at the indicated Reynolds numbers.

| $c$ | $N^{el}$ | $Re$  | $|\nabla \cdot u|_{max}$ | $\kappa$ | % change |
|-----|----------|-------|--------------------------|---------|----------|
| .2  | 618      | 168.4 | 16.46                    | .05323  | 3.1      |
|     | 1550     | 158.4 | 7.18                     | .05164  |          |
| .3  | 568      | 169.9 | 13.92                    | .01695  | 2.4      |
|     | 1202     | 169.9 | 7.96                     | .01667  | .7       |
|     | 1800     | 169.9 | 6.06                     | .01656  |          |

more expensive exact matrix method.

It is important to note that an $O(6)$ maximum absolute divergence value was noted at isolated high-gradient regions in the domain for the high-Reynolds-number runs. This divergence has little effect on the permeability calculation which is a spatial average, but can certainly lead to incorrect conclusions particularly in relation to unsteadiness phenomena [Gresho et al. 1993] (see also the section on Hexagonal arrays). To significantly reduce the maximum divergence reported, in an economic way, one need to pursue an adaptive mesh refinement and error estimation [Bank and Weiser 1985] rather than a global refinement which can be prohibitive. We merely state here, that the divergence has been checked against the mesh fineness and determined to drop with refined mesh (albeit significant cost increase). This is illustrated in Table 9.1 which reports the maximum absolute divergence, $|\nabla \cdot u|_{max}$ for different mesh densities (measured by the number of elements $N^{el}$), at a high Reynolds numbers for two square arrays. We also report in the table, the calculated permeability for each mesh and the percentage (insignificant) change with respect to the fine mesh result.

Finally, to support our conclusions, we computed the flow at the highest Reynolds number considered ($Re = 168.4$) using the computational spectral-element-based package NEKTON

\footnote{NEKTON Version 2.85, Fluent Inc. Lebanon, NH.}, using order-seven 64 spectral elements (see Figure 9-7). The spectral element code predicted results agreeing with those predicted by our finite-element code. The same oscillatory response was observed with virtually the same measured frequency of oscillations. Moreover, the onset of the oscillations occurred after the elapse of approximately the same time units starting from zero initial conditions. The amplitude of the oscillations in the spectral element code was again slightly higher than our lumped-mass finite element code. These effects can be seen in Figure 9-8 which shows the time history of the $y_{2}$-component of the velocity from both codes at approximately the same history point location. We also note that although the mesh used in NEKTON is relatively fine, we still observed a lower but $O(6)$ isolated high divergence regions. (The maximum absolute divergence was 5.99.)

Next, we return to Figure 9-2 and comment on the discrepancy between our data points and those obtained by Edwards et al., for the higher concentrations indicated on the Figure. At $c = .3$ and $c = .4$, we obtain relatively good agreement for the lower Reynolds numbers (steady flows). At the highest Reynolds number considered the flow is determined to be unsteady which explains the notable drop in the permeability. At the concentrations $c = .5$ and $c = .6$ a large disagreement is observed. Edwards et al. data points are significantly
Figure 9-7: Spectral element mesh used in NEKTON for the square array at concentration .2. The x's indicate location of history points.

<table>
<thead>
<tr>
<th>c</th>
<th>$Re_p$</th>
<th>$Re$</th>
<th>$St$</th>
</tr>
</thead>
<tbody>
<tr>
<td>.2</td>
<td>53.0</td>
<td>150.2</td>
<td>4.046</td>
</tr>
<tr>
<td></td>
<td>57.2</td>
<td>168.4</td>
<td>4.021</td>
</tr>
<tr>
<td>.3</td>
<td>101.3</td>
<td>169.9</td>
<td>3.601</td>
</tr>
<tr>
<td>.4</td>
<td>175.7</td>
<td>172.5</td>
<td>3.029</td>
</tr>
<tr>
<td>.5</td>
<td>344.9</td>
<td>208.2</td>
<td>7.533</td>
</tr>
<tr>
<td>.6</td>
<td>700.8</td>
<td>190.4</td>
<td>5.931</td>
</tr>
</tbody>
</table>

Table 9.2: Strouhal numbers associated with the unsteady flow fields for the square array at the indicated concentrations and Reynolds numbers.

lower than ours for all the Reynolds numbers considered. Their Stokes solution is again at a high relative error of 2.5% and 6.2% respectively at the concentrations .5 and .6, from the known exact solutions. The lack of resolution as evident by the large error in the Stokes solution, and the incorrect steady assumption at the high Reynolds numbers, in Edwards et al. data, render their data less reliable than our presented data for the permeability of square arrays.

Table 9.2 presents the Strouhal numbers associated with the unsteady flows for the various concentrations considered at the obtained flow Reynolds numbers. It is important to note that our choice of the flow length scale $L_f = (\lambda - d)$ (as opposed to, say, $d$) in the definition of the Strouhal number (9.1) takes into consideration (albeit nonlinearly) the concentration dependence, which renders the Strouhal number a function only of the Reynolds number. We remark, however, that sharper analysis is needed before any conclusions could be withdrawn on the behavior of the Strouhal number for unsteady flows through periodic arrays. (For example, the oscillatory flow in the gap of width $\lambda - d$ may be analyzed in the light of a jet instability. Here $<u_1>$ may also need to be scaled to better reflect the flow.
Figure 9-8: Time history plots for the $u_2$-component of the velocity at $Re_p = 57.2$ using both the lumped mass finite element code (upper) and the spectral–element code NEKTON (lower) for the square array at concentration $c = .2$.

We next compare our results against the available experimental data of Bergelin et al. [1952], and the numerical data of Eidsath et al. [1983] at $c = .5$. The comparison is shown in Figure 9-9. We first remark that data are presented in terms of a dimensionless pressure drop, $\overline{\nabla P}$, defined as

$$\overline{\nabla P} = \frac{1}{\lambda/d} \frac{\Delta P}{\rho <u_1>^2},$$

(9.2)

rather than the diameter–squared normalized permeability, $\kappa$, used in the previous comparisons. The two are, however, related by the following relation

$$\overline{\nabla P} = \frac{1}{\kappa^2 Re_p^2}.$$  

(9.3)

For the first two Reynolds numbers, the figure suggests that our data fall closer to the experimental measurements of Bergelin et al. [1952], than the data reported by Edwards et al. [1989] and Eidsath et al. [1983]. Our last two data points happen to be at higher Reynolds numbers than the rest of the data which precludes accurate comparison.

We conclude this section by describing the flow structure at different Reynolds numbers. In Figure 9-10 we plot the streamlines for the square array at concentration .3 for several Reynolds numbers. At finite Reynolds number a pair of symmetric vortices is seen to form in the gaps between cylinders. As Reynolds number is increased, the vortices intensify with their centers pushed apart resulting in parallel streamlines through the free path. At the
Figure 9-9: Comparison of dimensionless pressure drop, $\nabla P$, versus Reynolds number, for the square array at concentration 0.5 between the experimental data of Bergelin et al. [1952], the numerical data of Eidsath et al. [1983] and Edwards et al. [1989], and current data.

The highest Reynolds number the flow is unsteady and the plot represents an instantaneous streamlines configuration.

A deeper qualitative picture of the flow pattern with increasing Reynolds numbers can be described in conjunction with the visual studies of Dybbs and Edwards [1982]. The latter considered porous media consisting of plexiglass spheres in hexagonal packing and plexiglass rods arranged in a complex, fixed three dimensional geometry. Although the pore structure is different than our simple square array, there is still a close parallel in the flow structure between the different structures. The visualization studies showed that at $Re = 1$ boundary layers begin to develop near the solid boundaries of the pores. At $Re > 10$, the boundary layers become more pronounced and an “inertial core” appears. As the Reynolds number is increased, the core flows enlarge in size and their influence become more and more significant on the overall flow picture. The first evidence of unsteady flow is observed at $Re \approx 150$, in the form of laminar oscillations in the pores. These oscillations take the form of traveling waves characterized by distinct periods and amplitudes.

### 9.1.2 Hexagonal Array Results

The next type of regular array considered, is that of a hexagonal arrangement as shown in Figure 5-8. Again we restrict our selected concentrations and Reynolds numbers mainly to the set reported in Edwards et al. [1989] to allow for constructive comparison. These concentrations and Reynolds numbers are
Figure 9-10: Streamlines contours for flow through a square array of cylinders at .3 concentration at the indicated Reynolds numbers.
• $c_j \in \{.3, .4, .5, .6\}$
• $Re_{ru1} \in \{0.20, 40, 120, 180\}$

which is a total of 20 data points. Unlike the square array, the permeability of the hexagonal array differs along any two orthogonal directions. Here, we compute the permeability along the $y_1$-direction for the hexagonal configuration shown in Figure 5-8, which we shall denote simply as $\kappa$. (Our $y_1$-direction matches $y_2$-direction with respect to Edwards et al. hexagonal cell.) It is important to note here that, due to the nonlinear Navier–Stokes equations, knowledge of the permeability for any two orthogonal directions can not be used (by simply adding vectors) to calculate the permeability along a different oblique direction. This statement is not true in the case of linear Stokes flow in which knowledge of the permeability tensor $\kappa_{ij}$ will uniquely specify the permeability along any desired direction.

Figure 9-11 shows our permeability results (marked with o’s) and those reported by Edwards et al. [1989] (marked with x’s) for the combination of concentrations and Reynolds numbers considered. (The data points are listed in Table F.5.) Again here we observe notable discrepancies which we shall elaborate on by treating each concentration individually.

We note first, that our computational domain is four times larger than that considered in Edwards et al. (Our hexagonal cell contains 4 inclusions whereas Edwards et al. cell contains 2 inclusions but they consider only half of it with symmetry assumptions.) We have used typically $O(3000)$ finite elements, that is, $O(800)$ elements per quarter the domain which is twice the number reported in Edwards et al. (Note that Edwards et al. solution is technically incorrect at unsteady flows, which are found to exist at Reynolds numbers larger than about 150 for all concentrations, and, hence, a valid comparison can not be made in such cases. Therefore we will limit the comparison to the steady cases.)

At concentration $c = .3$, the data points are in good agreement for the (steady) Reynolds numbers considered. At concentration $c = .4$, a noticeable discrepancy is observed at $Re \approx 40$. We note first, that Edwards et al. Stokes solution deviates from the exact solution of Sangani and Acrivos [1982a] by 2.3% compared to our virtually identical solution, and second, at $Re \approx 40$ there appears a nonsmooth change in Edwards et al. data as suggested by the plot which is likely due to numerical errors. (Note that Edwards et al. incorrect solution at the highest Reynolds number, happen to lie below our temporally–averaged value, contrary to what have been typically observed.) At $c = .5$ we find a very good agreement between the two measurements for all the reported Reynolds numbers. Nevertheless, we remark that the agreement at the unsteady data point (highest $Re$) does not imply that Edwards et al. approach is correct. We have, in fact, targeted this point using a finer mesh with virtually insignificant improvement in the reported value. The good agreement here is attributed to the fact that the permeability, due to spatial averaging, is insensitive to the flow details. A large discrepancy is observed at the last concentration considered, $c = .6$. Again here, Edwards et al. Stokes solution deviates by relative error of 1.3%, and there is a noticeable abrupt drop in Edwards et al. permeability value at $Re \approx 40$, perhaps indicating some numerical errors in the solution which can be potentially responsible for the discrepancy at the two neighboring data points.

It is interesting to illustrate the effect of $\sqrt{x}$ of resolution on the output flow characteristics [Gresho et al. 1993]. In Figure 9-12 we compare the time history of the $y_1$–component of the velocity for the same $Re_p$ using two meshes with respectively 1430 and 3840 finite elements for the hexagonal array at concentration $c = .4$. For the coarser mesh, the flow switched from a steady flow to complete chaos after introducing a small numerical disturbance due to a restart. The finer mesh solution assumed smoothly a stable periodic state
Figure 9-11: Hexagonal array permeability results at 4 concentrations for several Reynolds numbers. The o’s indicate current data; the x’s are data reported in Edwards et al. [1989].
$$
<table>
<thead>
<tr>
<th>c</th>
<th>Re_\rho</th>
<th>Re</th>
<th>St</th>
</tr>
</thead>
<tbody>
<tr>
<td>.3</td>
<td>124.7</td>
<td>171.2</td>
<td>3.910</td>
</tr>
<tr>
<td>.4</td>
<td>245.3</td>
<td>207.1</td>
<td>6.386</td>
</tr>
<tr>
<td>.5</td>
<td>345.8</td>
<td>182.3</td>
<td>7.690</td>
</tr>
<tr>
<td>.6</td>
<td>476.3</td>
<td>165.9</td>
<td>6.439</td>
</tr>
</tbody>
</table>

Table 9.3: Strouhal numbers associated with the unsteady flow fields for the hexagonal array at the indicated concentrations and Reynolds numbers.

which is insensitive to numerical disturbances.

![coarse mesh](image1)

![fine mesh](image2)

Figure 9-12: Two history plots of the $y_1$-component of the velocity at Reynolds number $Re_\rho = 245.3$ using coarse and fine meshes for the hexagonal array at concentration $c = .4$. Clearly, lack of resolution can lead to erroneous conclusions about the flow characteristics [Gresho et al. 1993].

The Strouhal numbers associated with the unsteady hexagonal-array flows are listed in Table 9.3. For the hexagonal array, the flow length scale, $L_f$, in (9.1) is defined as the gap width along the center line between two neighboring cylinders of unity diameters (recall that $d$ is set to 1 for all the presented results in this thesis).

It is interesting to compare the permeability results for the square and hexagonal arrays. Figure 9-13 provides this comparison, which indicate the existence of a crossover between the two arrays at all the concentrations shown, expect the highest $c = .6$, at which no crossover
occurs in the range of Reynolds numbers considered. At the Stokes limit, the hexagonal arrays possess higher permeability than the square arrays. The Reynolds number at which a crossover occurs appears to increase with increasing concentrations. The crossover Reynolds numbers are approximately 12, 50, and 125 for the concentrations \( c = .3, c = .4, \) and \( c = .5 \) respectively. It is very likely that a crossover will also occur for the concentration \( c = .6 \) at a higher Reynolds number value. The reasoning behind this behavior can be explained as follows: the hexagonal array possesses less surface area relative to a square array at the same concentration. (In a unit volume the surface area exposed to the fluid is proportional to the \( \sqrt{3c/2\pi} \) for the hexagonal arrangement and to \( \sqrt{cn} \) for the square arrangement.) This explains why the hexagonal array has higher permeability for lower Reynolds numbers at which inertial effects are insignificant. At higher Reynolds numbers, the “tortuous” character of the hexagonal array leads to higher dissipation than does the “straight-through” character of the square array. In other words, in the square array, one inclusion shields the next as the wake grows, an effect which is not as pronounced in the hexagonal array. This, leads to a more streamlined flow pattern (less dissipation) in the square array compared to the hexagonal array, which could explain the crossover as Reynolds number is increased. As the concentration is increased, the surface–related drag effects become stronger (the square array has larger area), which could explain the delay in the crossover with increasing concentration.

We conclude this section with a brief description of the flow structure in the hexagonal array at various Reynolds numbers. Figure 9-14 depicts the streamlines contours for the hexagonal array at concentration \( c = .3 \) for several Reynolds numbers. At \( Re = 0 \) the flow is symmetric and no vortices are present (or at least that could be detected numerically). As the Reynolds number is increased, a symmetric pair of vortices is seen to appear in the gaps between successive inclusions. These vortices grow with increasing Reynolds number with their centers being drawn further apart forcing the streamlines to be parallel in a channel–like flow (see the flow at \( Re = 124.1 \)). At \( Re = 171.2 \) the flow is unsteady and the picture represents an instantaneous configuration of the flow pattern.

### 9.2 Random Arrays

The ultimate goal of this research effort is to understand the permeable character of truly random porous media under practical flow situations. Although this problem is three dimensional in nature, simpler two–dimensional models that can be realized computationally, could afford valuable insights into the physics of the more complex systems, and should be viewed as a first step in the investigation process. The purpose of this section is to present the few results collected along this line of thought to help shape the way for future work.

We present, here, configuration upper–bounded permeability results which are calculated for randomly–generated realizations at selected concentrations and flow Reynolds numbers:

1. \( c = .2, \) \( Re \in \{18, 31, 59\} \)
2. \( c = .3, \) \( Re \in \{16, 28, 60, 70\} \)
3. \( c = .4, \) \( Re \in \{24, 44, 87\} \)
4. \( c = .5, \) \( Re \in \{15, 23, 57, 91\} \)

that is a total of 14 runs. (We remark that the above Reynolds numbers have been rounded up to an integral value. The actual Reynolds numbers and the permeability values are listed
Figure 9-13: Comparison between square and hexagonal arrays permeability results at 4 concentrations for several Reynolds numbers. The o’s indicate square array data; the x’s indicate hexagonal array data.
Figure 9-14: Streamlines contours for flow through a hexagonal array of cylinders at .3 concentration at the indicated Reynolds numbers.
in Table F.6.) The size, $\lambda$, of the supercells is chosen sufficiently large, $\lambda = \lambda_0 = 7$, to insure low variances, and, hence, a higher likelihood that our configuration values do represent the supercell (multiple-realization ensemble) permeability values. (We note that, due to computational considerations, only upper-bound nip elements are used ($\alpha_c = .1, \beta = .1$). However, based on the results of Section 5.4, we do not expect the one-sided bound error to be significant.) For each of the first three concentrations, we have only varied the Reynolds number using the same realization. The three upper-bound random realizations are shown in Figures F-1, F-2, and F-3 respectively, with the corresponding inclusion centers listed in Table F.7. For the concentration $c = .5$ different random realizations are used for the 4 Reynolds numbers considered. The corresponding inclusions centers are listed in Table F.8.

For the mesh generation procedure, we specified the following parameters (see Section 4.2): $h_{nom} = .2, m = 3, r = 1$, which resulted in a number of elements ranging from 7,200 ($c = .2$) to 11,200 ($c = .5$). (The number of degrees of freedom per velocity component is approximately twice the number of elements.) We believe, based on wiggle-free history plots and the moderate Reynolds numbers, that this resolution is adequate for accurate permeability prediction. (Figure 9-15 shows selected higher-Reynolds-number steady history plots for the $u_1$-components of the velocity.) The Courant condition has been set to 0.5 in all these calculations.

![Figure 9-15: Time history plots of $u_1$ for selected steady Navier–Stokes calculations at the indicated Reynolds numbers and concentrations.](image)

The flow has been found to be steady for all the Reynolds numbers listed, except for the highest one in each group. Unfortunately, for the unsteady runs, except the case ($c = .3, Re = 70$), our permeability values represent spatial averages only at the final integration times which are also listed in Table F.6. Figure 9-16 shows the unsteady history plots of
the $u_1$-components of the velocity for the temporally-unaveraged three runs, which indicate that a steady state has not been reached yet (although significant time has elapsed). We anticipate, however, based on the results of Section 5.4, that the unsteadiness will have little effect on the spatially-averaged permeability, and therefore have decided to include those three data points as well.

![Graphs showing $u_1$ components for different concentrations and Reynolds numbers](image)

$c = .2, \, Re = 58.9$

$c = .4, \, Re = 86.7$

$c = .5, \, Re = 90.5$

time unit

Figure 9-16: Time history plots of $u_1$ for unsteady Navier–Stokes calculations at the indicated Reynolds numbers and concentrations.

Finally, it is worthy to note that these calculations are computationally intensive. The higher Reynolds-number calculations require a few days each to reach a steady state. (Had the upper-bound nips not been used, this time would have significantly increased.) Figure 9-17 shows the $y_1$-component of the velocity contours for a particular unsteady calculation at Reynolds number, $Re = 90.5$, in a random array at concentration $c = .5$.

The author is not aware of any published systematic studies on random arrays of cylinders, and hence no data are available for constructive validation and comparison. We shall however, compare our results to the Ergun correlation [Ergun 1952] for true 3-D granular beds. The correlation is expressed in the form:

$$f' = \frac{150}{Re'} + 1.75,$$

(9.4)

where

$$f' \equiv \nabla_P \frac{\epsilon^3}{1 - \epsilon},$$

(9.5)

and

$$Re' \equiv Re \frac{1}{1 - \epsilon},$$

(9.6)
Figure 9-17: A plot showing $y_1$-component of the velocity contours for a Navier–Stokes calculation at Reynolds number, $Re = 90.5$, in a random array at concentration $c = .5$. 
are a modified friction factor and Reynolds number as defined by Ergun. Here \( \epsilon \) is the porosity, \( 1 - c \), and \( \nabla \bar{P} \) is the nondimensional pressure drop defined in (9.3). The Ergun relation is a widely accepted relation for practical engineering application for regimes not amenable to Darcy's law. Ergun studied the effect of void fraction and particle size on the pressure drop through unconsolidated granular bed. He concluded, based on 640 experiments involving various-sized spheres, sand, pulverized coke and several gases, that the total pressure drop in fixed porous beds can be treated as the sum of viscous (\( \sim < u_1 > \)), and kinetic (\( \sim < u_1^2 > \)) energy losses with a coefficient proportional to \( \frac{(1-\epsilon)^2}{\epsilon^4} \) for the former, and to \( \frac{1}{\epsilon^4} \) for the latter.

In Figure 9-18 we plot our data points in terms of the modified friction factor, equation (9.5), at the selected Reynolds numbers. Ergun relation is also plotted on the same figure. This figure raises more questions than one can provide answers to based on the limited data. Given the profound differences between the Ergun relation which represents three-dimensional granular beds measurements at porosities around \( \epsilon = .4 \), and our calculations which are for two-dimensional circular microstructure at several porosities, one can only remark that the agreement is, for the most part, reasonable. We shall attempt, however, to seek plausible reasoning for some of the observations. The data points at porosity 0.5 which fall in the porosity range 0.43 \( \leq \epsilon \leq \: 0.52 \) characteristic of the Ergun relation happen to fall above the curve. This suggests that our 2D model predicts lower permeability than a comparable granular bed. (Note that our permeability values are already upper-bounded.) One factor (among many others) contributing to this phenomenon may, arguably, be based on the surface drag effects as follows: although the granular porous media possess larger surface area exposed to the fluid compared to a fibrous random media of identical volume (e.g., a spherical inclusion has larger specific surface area than a cylindrical inclusion), the complex structure of the former causes a large portion of this area to be in contact with only stagnant fluid due to trapping geometrical effects. In other words, the fibrous random media will have a larger net surface area, due to its more-structured microstructure, in contact with energetic non-stagnant fluid, and hence, will have more pronounced drag effects. We note that similar observation has been made for creeping flows where the random array of cylinders predicted a lower permeability than a comparable array of spheres (see Section 8.2).

The agreement between Ergun relation and our data points at porosity 0.7 is indeed remarkable although nonintuitive, and no plausible explanation could be provided for this behavior. Note that the discrepancy between our data points and the Ergun equation is not systematic in the porosity, or one would have expected a better agreement with data points at porosity 0.6 (given the behavior at porosity 0.5).

We conclude this section by comparing the random array results to those of the square array. Figure 9-19 shows the permeability results of both the random array and the square array at the indicated concentrations for several Reynolds numbers. We observe a crossover after which the random array permeability becomes significantly lower, than the square array permeability, with increasing Reynolds number. No crossover is observed at concentration \( c = .5 \) in which the random array permeability is consistently lower. (In the limit of vanishing Reynolds number, a similar crossover is observed in Cruz [1993] between random arrays and regular arrays at around \( c = .3 \) which is consistent with our result.) The explanation of this crossover near the Stokes limit at this concentration follows the argument provided in Cruz [1993] in which a larger net passage is available to the flow in the random array (see Chapter 8). Once inertial effects are important, the tortuous nature
Figure 9-18: Comparison of the modified friction factor (equation (9.5)) predicted by Ergun equation [Ergun 1952] versus values calculated for random 2–dimensional arrays of cylindrical inclusions at selected concentrations and Reynolds numbers.

of the random array leads to significantly higher dissipation and thus lower permeability.

9.3 Future Work

Related tasks to be targeted in future work include the following:

- On the numerical side: implementation of a posteriori error estimation and adaptive mesh refinement [Bank and Weiser 1985]; implementation of localized direct–stiffness vertex collection methods [Fox el al. 1988]; development of efficient algorithms for shuffling of subcells for better and dynamic load balancing (this mechanism has been implemented in the code, but was not activated given the good results with the presence of nip elements); implementation of more efficient Monte–Carlo algorithms for generation of realizations with high concentrations; implementation of alternative solution algorithms for the Navier–Stokes equations based on Newton iteration and GMRES iterative solvers for nonsymmetric systems [Saad and Schultz 1986];

- On the nip element method: formulation and implementation of sharper bounds for the Stokes problem, analogous to the sharper bounds for the effective conductivity; application of sensitivity derivatives to the Navier–Stokes bounds to assess their applicability at moderate and high Reynolds numbers; development of smarter algorithms for the placement of nips based on geometrical considerations; formulation of a more general strategy in which nips can have variable sizes (in this work the nip height, $\beta$
Figure 9-19: Comparison of the permeability of the random array and the square array at the indicated concentrations for several Reynolds numbers. The x indicates random array, the o indicates square array.
was constant for all nips).

• On the model development side: completion of the inertial flow results for regular arrays, and presentation of the final results in terms of validated accurate models; investigation of unsteadiness–related effects; consideration of (multiple–realization ensembles of) inertial flow through random arrays to provide plausible explanations for experimentally observed behavior, with the ultimate goal of validating an accurate surrogate to predict the permeability of such arrays (this task also include determination of the associated correlation lengths as a function of Reynolds numbers); investigation of the inclusion shape on the surrogate by attempting other, possibly nonconvex, inclusion geometries.

• Extension of the methodology to three spatial dimensions: this task will certainly place new demands on the numerical treatment, in particular the geometry and nip definition, domain partitioning, mesh generation, parallel direct stiffness procedure, and, generally, memory management. It is also anticipated that more powerful parallel computing power will be necessary.

• Extension of the methodology to Stokes sedimentation [Cruz 1993], and to fluidized beds, both as suspensions and static fluidization problems.
Appendix A

Nip–Defining Geometrical Constraints

In this appendix we derive sufficient conditions on the nip element dimensions to prevent intersection of multiple nips. The nip size is controlled by the parameters $\alpha$, the distance between the two forming cylinders, and $\beta$ the distance between the nip edge and the center line connecting the two cylinders. Central to what follows is the assumption that $\beta$ is constant for all nips. The general case of variable $\beta$ was not considered in this thesis. The problem then becomes: for a specified critical $\alpha_c$, for which any two cylinders with distance equal to or less than $\alpha_c$ are considered forming a nip, what is the maximum allowable $\beta_c$ such that no two nips intersect? It turns out that the answer to this question can be determined in two steps: first finding a condition on $\beta$ as a function of the angle between two neighboring nips sharing a joint cylinder to prevent their intersection; and second finding the minimum angle that can be obtained in a cluster of three nip-forming cylinders and hence $\beta_c$.

Consider the three cylinders of unit diameter shown in Figure A-1 with centers $y_1, y_2$, and $y_3$ with $y_1$ positioned conveniently at the origin. The segment $\overline{y_1y_2}$ is inclined by an angle $\theta < \frac{\pi}{2}$ from the unit vector in the $y_1$ direction. We also assume that the pairs $(y_1, y_2)$ and $(y_1, y_3)$ are separated by a distance less than or equal to $1 + \alpha_c$, and thus, are considered forming nips $a$, and $b$, respectively. We comment that the third pair $(y_2, y_3)$ may or may not form a nip depending on the angle $\theta$. The latter case however, will lead to the most stringent condition on $\beta_c$ as we will see shortly.

We first derive a condition on $\beta$ as a function of the angle between two neighboring nip edges such that to prevent intersection of the two edges. We consider $\Gamma_1^a$ and $\Gamma_2^b$ and simply require the intersection point $p = (x_p, \beta)$ between the two line segments passing through $\Gamma_1^a$ and $\Gamma_2^b$ as shown in Figure A-2, to lie inside the circle with center $y_1$, that is:

$$x_p^2 + \beta^2 \leq \frac{1}{4}.$$ \hspace{1cm} (A.1)

We have

$$\Gamma_1^a(x) = \tan \theta \left( x - \frac{\beta}{\sin \theta} \right),$$ \hspace{1cm} (A.2)

and

$$\Gamma_2^b(x) = \beta.$$ \hspace{1cm} (A.3)
Figure A-1: Definition of nips $a$ and $b$.

Figure A-2: The intersection of $\Gamma_1^q$ and $\Gamma_2^b$ is prevented by requiring the intersection point $p$ to lie inside the circle with center $y_1$. 
Figure A-3: The minimum angle $\theta_{\min}$ in a triangle whose sides are greater than or equal to unity but less than or equal to $1 + \alpha_c$ is achieved by the isosceles case shown above.

Solving for $x_p$ by letting $\Gamma_i^1 = \Gamma_i^2$, we derive the following condition on $\beta$:

$$\beta \leq \frac{1}{2} \left[ 1 + \frac{1}{\tan^2 \theta} \left( 1 + \frac{1}{\cos \theta} \right)^2 \right]^{-\frac{1}{2}} \tag{A.4}$$

By appropriately redefining $\theta$, relation A.4 applies as well to any two neighboring nip edges.

As expected, $\beta$ is an increasing function of the angle $\theta$ between the two nip edges. Thus the second step would be to determine the minimum possible value, $\theta_{\min}$, for the angle $\theta$ in the configuration of Figure A-1 when three nips are formed. The argument here is that by picking $\beta_c$ according to $\theta_{\min}$ no intersection among all the nip edges will be possible since the remaining angles will be equal or greater than $\theta_{\min}$, and hence, will yield equal or larger $\beta_c$'s.

We consider the triangle formed by the centers $y_1, y_2,$ and $y_3$ whose sides are greater than or equal to unity but less than or equal to $1 + \alpha_c$. By inspection, the minimum angle is achieved by an isosceles triangle having sides $1, 1 + \alpha_c$ and $1 + \alpha_c$ as shown in Figure A-3. In fact, if the center $y_2$ is rotated clockwise about the center $y_3$, $\theta_{\min}$ will decrease but the segment $\bar{y_1}y_2$ will exceed $1 + \alpha_c$ in length and hence, no nip is formed. On the other hand rotating $y_2$ counterclockwise will increase $\theta_{\min}$ and hence $\beta_c$. (Similar argument applies to the center $y_3$.) For numerical values we take $\alpha_c = 0.20$ for which $\theta_{\min} = 49.24$ degree. From equation A.4 we calculate $\beta_c = 0.21$. 

149
Appendix B

Example of Voronoi/Delaunay Mesh Generation

B.1 Introduction

In the following appendix we describe a method for triangulating two dimensional arbitrary domains. The method is based solely on construction of the Delaunay triangulation for a given set of nodes, and should, therefore, also be applicable to three dimensional domains.

Typically, in Delaunay–based triangulations [Weatherill 1985, Baker 1987, Holmes and Snyder 1988], the integrity of the boundaries is achieved either by increasing the boundary point resolution, or by performing an edge swapping clean-up phase as a post–processing operation to recover the boundary edges [George, Hecht and Saltel 1990].

The new feature of our method is a simple procedure for preservation of domain boundary integrity which requires neither modification of the Delaunay triangulation, nor special attention in the layout of boundary and background nodes.

B.2 Definition of Voronoi/Delaunay Diagram

We begin by defining the Voronoi diagram, and its dual the Delaunay triangulation, for a set of points (generators) in the Euclidean plane. Given a set of \( N_g \) generators in a plane, the Voronoi diagram identifies \( N_g \) convex regions (called Voronoi tiles) whose interior points are closest to the corresponding generator. (The outermost tiles may be bounded or unbounded depending on whether or not an artificial box is place around the region of interest.) Points in the plane that are equidistant form two generators define a Voronoi edge. A Voronoi vertex is defined by the intersection of three (or more) Voronoi edges. (The case when more than three Voronoi edges meet at one vertex corresponds to a degeneracy which poses some stability–related difficulty in the construction algorithms [Preparata and Shamos 1988, Sugihara and Iri 1989a, 1989b].) One property of the Voronoi diagram (known as the empty circle) is that any Voronoi vertex is the center of the circle that goes through the three (or more) equidistant generators around at, and that no other generator lies inside this circle. Figure B-1 shows a representative Voronoi diagram.

The dual Delaunay diagram can be constructed from the Voronoi diagram by identifying each three generators associated with a Voronoi vertex as one triangle as shown in Figure B-2. (Again, a degenerate Voronoi vertex would lead to a quadrangle.) A Delaunay edge, by definition, connects two Voronoi generators and is perpendicular to a Voronoi edge. It
should be noted that Delaunay diagram can be constructed directly for a given set of nodes without the need to go through the Voronoi diagram a priori.

The issue of preserving the boundary integrity of a given domain to be triangulated, arise since one can not guarantee, without careful attention in the lay out of generators, that boundary segments will constitute part of the Delaunay diagram. In the next section we present a procedure that can guarantee this requirement based on a simple circle test.

B.3 Mesh Construction Procedure

B.3.1 Domain Definition

A Domain $\Omega$ to be triangulated need not be simply connected. The $k^{th}$ boundary curve of $\partial \Omega$, $M_k^{\partial \Omega}$, is described by a periodic chain of consecutive nodes $x_j^k$, which are spaced according to the desired element-size distribution on the boundary. These nodes will constitute
the boundary mesh:

\[ M^\Omega_k = \{ \ldots, x^k_{j-1}, x^k_j, x^k_{j+1}, \ldots \} \].

**B.3.2 Layout of Interior Nodes**

A structured background grid \( M^{BG} \) of size \( h \) is laid out such that the domain is completely covered. The pattern of the grid controls the type of the elements not lying on a boundary curve of \( \Omega \), and is chosen by the user. As an example, a square grid will generate right triangles and a hexagonal grid will create equilateral elements. Other background placement procedures (e.g., random) are also possible. Figure B-3 depicts two different domains with appropriate background grids.

**B.3.3 Preservation of Boundary Integrity**

Before passing the set of nodes to the Voronoi/Delaunay construction, a test is carried out to discard certain grid nodes in order to ensure integrity of the boundary segments. This test is explained and accomplished through the following two steps:

1. \( M^\Omega_k \) center points and radii are defined.

\[ M^\Omega_k |_{cp} = \{ \ldots, x^k_{j-1/2}, x^k_{j+1/2}, x^k_{j+3/2}, \ldots \} \]

\[ M^\Omega_k |_{r} = \{ \ldots, r^k_{j-1/2}, r^k_{j+1/2}, r^k_{j+3/2}, \ldots \} \]

where

\[ x^k_{j+1/2} = \frac{1}{2} (x^k_j + x^k_{j+1}) \]

152
In the case of a uniform boundary mesh, with spacing $h_B$, $\rho_{j+1/2}^k$ is simply $h_B/2$.

2. The second step is to loop over all boundary curves of $\partial \Omega$, and to eliminate all grid points $x_{BG}^k \in M_{BG}$ that lie in the closed disk $D$ of radius $\rho_{j+1/2}^k$ centered at $x_{j+1/2}^k$, as illustrated in Figure B-4. This step is essential in preserving the integrity of each boundary segment $x_j^k x_{j+1}^k$. In particular, by virtue of our circle-exclusion test, $x_j^k x_{j+1}^k$ must be a Delaunay edge, since, by construction, $x_{j+1/2}^k$ will lie on a Voronoi edge shared by the Voronoi tiles associated with $x_j^k$ and $x_{j+1}^k$. This is shown in Figure B-5.

It is also recommended, but not necessary, to perform the circle test centered at the boundary nodes themselves, in order to prevent overly distorted elements near the boundary (see Figure B-6).
Figure B-6: The circle test is also recommended at the boundary grid points (filled circles) in order to eliminate background grid nodes giving rise to acute angle. Performing the circle test only at the center points (empty circles) will not predict ill-located background grid nodes.

B.3.4 Construction of the Voronoi/Delaunay Diagram

The Voronoi diagram is then constructed for the remaining grid and boundary nodes. (One can proceed directly with a Delaunay triangulation without the need to go through the dual Voronoi diagram.)

The triangles of the Delaunay are retrieved from the Voronoi data structure by identifying the three generators, VG's, associated with each Voronoi vertex \( V \) (see Figure B-2). The Voronoi is assumed to contain no degeneracies. This is realized, for example, by the Sugihara and Iri [1989a, 1989b] program, which was used to generate the examples given below.

Finally, the triangulated domain is extracted by eliminating all the triangles with centroid outside \( \Omega \). This step is performed using the line-intersection insideness test [Preparata and Shamos 1988].

B.4 Examples

In this section we present two simple examples, one with a square background grid, and another with a polar background grid. The domains are shown in Figures B-7 and B-8 respectively. We note that these examples were generated using the program VORONOI2 developed by Sugihara & Iri [1989a, 1989b].
Figure B-7: Rectilinear background grid.

Figure B-8: Polar background grid.
Appendix C

Triangular Finite Elements

Triangular finite elements enjoy a number of attractive features which make them popular in finite element analysis. They are more suitable for irregular geometries yielding generally fewer deformed elements, which might lead to larger numerical errors; they possess fewer degrees of freedom per element compared for example, to same order quadrangular elements (the latter has a larger polynomial space though, which is the tensor product of two same-order spaces); and, triangulation programs are widely more available than for other types of elements. In this appendix we will review linear and quadratic triangular finite elements, and derive the elemental integrals for the standard Laplacian and Stokes operators.

C.1 Linear Triangular Elements

![Diagram of mapping a triangular element from $\mathbb{R}^2$ plane to the $\eta$ plane via the affine transformation $F^k$.]

Figure C-1: Mapping a triangular element from $\mathbb{R}^2$ plane to the $\eta$ plane via the affine transformation $F^k$.

Figure C-1 depicts a typical element $\Omega^k$, of area $A$, with generic point, $x = (x, y)$. The element vertices, $v_i = (x_i, y_i)$, $i \in \{1, 2, 3\}$, are numbered periodically in counterclockwise direction. We map the element in $\mathbb{R}^2$ to a reference element, $\bar{\Omega}$, defined in the barycentric coordinate system, $\bar{\eta} = (\eta_1, \eta_2, \eta_3)$, via the mapping

$$x = F^k(\eta).$$  \hspace{1cm} (C.1)
The barycentric coordinates, \( \eta_i, i \in \{1, 2, 3\} \), are defined as follows

\[
\eta_i(x) = \frac{\text{area}(\Delta_{x, y_{i+1}, y_{i+2}})}{A},
\]

where \( \Delta_{,,} \) denotes the triangle defined by the given points. Note that by definition \( 0 \leq \eta_i \leq 1 \), and that the barycentric coordinates are not independent, \( \eta_1 + \eta_2 + \eta_3 = 1 \). This should be the case since only two independent coordinates are needed in \( R^2 \). Generally only \( \eta_1 \) and \( \eta_2 \) are used in the elemental constructs with the substitution

\[
\eta_3 = 1 - \eta_1 - \eta_2.
\]

(To avoid confusion in this presentation, unless otherwise noted, \( \eta = (\eta_1, \eta_2) \).) By construction we have

\[
\eta_i(v_j) = \delta_{ij},
\]

where \( \delta_{ij} \) is the Kronecker delta function. Thus, in the linear case, \( \eta_i \) will simply serve as the shape functions, \( h_i \), for the element.

To obtain the inverse mapping \( \eta = F^{k-1}(x) \), we use definition (C.2). For example:

\[
\eta_2(x, y) = \frac{1}{2} \begin{vmatrix} x_3 - x & y_3 - y \\ x_1 - x & y_1 - y \end{vmatrix} / A,
\]

which simplifies to

\[
\eta_2 = a_2 + b_2 x + c_2 y,
\]

where

\[
\begin{align*}
a_2 &= -(x_1 y_3 - x_3 y_1) / 2A \\
b_2 &= (y_3 - y_1) / 2A \\
c_2 &= (x_1 - x_3) / 2A
\end{align*}
\]

Similarly we obtain for \( \eta_1 \) and \( \eta_3 \), respectively:

\[
\begin{align*}
a_1 &= (x_2 y_3 - x_3 y_2) / 2A \\
b_1 &= -(y_3 - y_2) / 2A \\
c_1 &= -(x_2 - x_3) / 2A \\
a_3 &= (x_1 y_2 - x_2 y_1) / 2A \\
b_3 &= -(y_2 - y_1) / 2A \\
c_3 &= -(x_1 - x_2) / 2A.
\end{align*}
\]

An alternative way to obtain the algebraic expressions for \( \eta_i \) or, equivalently, \( h_i \), is by using the definition of the shape function. For a linear interpolant we write

\[
h_i = a_i + b_i x + c_i y.
\]
Based on the interpolation property, equation (C.4), we construct the linear system

\[
\begin{bmatrix}
1 & x_1 & y_1 \\
1 & x_2 & y_2 \\
1 & x_3 & y_3 \\
\end{bmatrix}
\begin{bmatrix}
a_j \\
b_j \\
c_j \\
\end{bmatrix}
= \delta_{ij} .
\]  
(C.17)

Inverting system (C.17) for each node \(i\) yields the same expressions for the coefficients, \(a_i, b_i\) and, \(c_i\) derived above using the definition of barycentric coordinates.

The mapping \(x = F^k(\eta)\) takes the form:

\[
(x, y) = \sum_{i=1}^{3} (x_i, y_i) \eta_i ,
\]  
(C.18)

which can be intuited geometrically simply by using the interpolating property of the barycentric coordinates, or derived algebraically by inverting the system

\[
\begin{bmatrix}
a_1 & b_1 & c_1 \\
a_2 & b_2 & c_2 \\
a_3 & b_3 & c_3 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
x \\
y \\
\end{bmatrix}
= \begin{bmatrix}
\eta_1 \\
\eta_2 \\
\eta_3 \\
\end{bmatrix} .
\]  
(C.19)

Remark The mapping \(F^k\) and the inverse mapping \(F^{k-1}\) are, in fact, affine mappings meaning that a straight line in the \(\eta\) plane maps to a straight line in the \(R^2\) plane and, more generally, polynomials of total degree \(Q\) in \(\eta_i\) maps to polynomials of total degree \(Q\) in \((x, y)\) and visa versa. This statement will not hold for the isoparametric mapping in which \(\eta_i\) are no longer simple polynomials in \(x\) and \(y\).

### C.2 Elemental Laplacian

All the elemental integrals/derivatives are performed on the reference element using the affine transformation, \(F^k\), defined by (C.18). The Jacobian of the transformation, \(J\), is defined by the determinant

\[
J = \frac{\partial(x, y)}{\partial(\eta_1, \eta_2)} = \begin{vmatrix}
\frac{\partial x}{\partial \eta_1} & \frac{\partial y}{\partial \eta_1} \\
\frac{\partial x}{\partial \eta_2} & \frac{\partial y}{\partial \eta_2} \\
\end{vmatrix} = \left| \frac{\partial x}{\partial \eta_1} \frac{\partial y}{\partial \eta_2} - \frac{\partial y}{\partial \eta_1} \frac{\partial x}{\partial \eta_2} \right| .
\]  
(C.20)

Using equation (C.3) we find

\[
J = \begin{vmatrix}
x_1 - x_3 & x_2 - x_3 \\
y_1 - y_3 & y_2 - y_3 \\
\end{vmatrix} = 24 .
\]  
(C.21)

Remark For a right-handed system, that is, counterclockwise numbering of vertices (the one adopted here), the Jacobian will always be positive. In fact, for a consistent mesh (the elements either all right-handed or all left-handed), the Jacobian has always the same sign (either positive or negative).

The elemental stiffness matrix is then computed as follows

\[
\int_{\Omega^k} \nabla h_i \nabla h_j dx = \int_0^1 \int_0^{1-\eta^2} \frac{\partial h_i}{\partial \eta_k} \frac{\partial h_j}{\partial \eta_l} \frac{\partial h_i}{\partial \eta_l} + \frac{\partial h_i}{\partial \eta_k} \frac{\partial h_j}{\partial \eta_l} \frac{\partial h_i}{\partial \eta_k} \frac{\partial h_j}{\partial \eta_l} |J| d\eta_1 d\eta_2 ,
\]  
(C.22)
Figure C-2: Mapping a 2nd-order triangular element from $\mathbb{R}^2$ plane to the $\eta$ plane.

where summation is assumed over repeated indices. The integral in equation C.22 simplifies in the case of linear elements to the simple algebraic expression

$$\int_{\Omega^k} \nabla h_i \nabla h_j \, dx = A(b_i b_j + c_i c_j) . \quad (C.23)$$

### C.3 Quadratic Triangular Elements

Here we look for shape functions in the elemental space $P_2(\Omega^k)$ where $P_2(\Omega^k)$ is the set of polynomials of total degree 2 defined over the elemental domain $\Omega^k$. Note that the interpolant has six coefficients and thus six nodes per element are required. To conserve continuity and conformity of the functional space over which the trial function is defined, the three additional nodes must be inserted on the sides between the element vertices (see Figure C-2). (The functional space would be the same no matter where we position the mid nodes as long as they are on the element sides, however the discrete system, and associated conditioning would be different. The mid way position is a normal practice.)

We now look for six shape functions, $h_i(\eta_j) = \delta_{ij}$, $i, j = 1, \ldots, 6$. By inspection we find:

\begin{align*}
h_1 &= \eta_1(2\eta_1 - 1) \\
h_2 &= \eta_2(2\eta_2 - 1) \\
h_3 &= 2(1 - \eta_1 - \eta_2)^2 - (1 - \eta_1 - \eta_2) \\
h_4 &= 4\eta_1\eta_2 \\
h_5 &= 4\eta_2(1 - \eta_1 - \eta_2) \\
h_6 &= 4\eta_1(1 - \eta_1 - \eta_2)
\end{align*} \quad (C.24-29)

The mapping $P^k$ takes the form

$$\begin{align*}
(x, y) &= \sum_{i=1}^{6} (x_i, y_i) h_i(\eta_j) .
\end{align*} \quad (C.30)$$

The evaluation of the elemental Laplacian entries as given by equation (C.22), can be reduced to algebraic expressions using symbolic integration packages.
Figure C-3: The mid point of the triangular finite element is displaced to lie on the curved boundary for isoparametric accuracy.

C.3.1 Isoparametric Case

In the isoparametric case the mid nodes between the element vertices are forced to match the curved boundary of the domain (see Figure C-3). (In our supercell calculation, curved boundaries come from the circular inclusions; mid nodes of adjacent finite triangles are displaced to match the circular boundaries.) The mapping $F^k$ as given by (C.30) will uniquely determine the trace of the curved boundary based only on nodal values, but the mapping will no longer be affine since the inverse mapping, $\eta = F^{k-1}(x)$, will not be polynomial in $(x, y)$, and we no longer have explicit expressions for $\eta_i = \eta_i(x, y)$. Moreover, the Jacobian of the transformation is no longer a constant scaling factor but a variable function.

In order to form the elemental matrices we need to calculate the now-variable Jacobian and the derivatives of the inverse mapping with respect to $(x, y)$, that is, $\frac{\partial \eta_i}{\partial (x, y)}$. Since we do not have an explicit formula for the Jacobian, $J(x, y)$, we approximate it using the same shape functions

$$J(x, y) = \sum_{i=1}^{6} J(x_i, y_i) h_i(x, y).$$

(C.31)

The mapping derivatives needed to calculate the Jacobian at the collocation points, $J_i = J(x_i, y_i)$, are obtained using (C.30)

$$\frac{\partial (x, y)}{\partial \eta_j} |_{\eta_i} = \sum_{i=1}^{6} (x_i, y_i) \frac{\partial h_i}{\partial \eta_j} (\eta_i).$$

(C.32)

To find the derivatives $\frac{\partial \eta_i}{\partial (x, y)}$ we write

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \eta_1} & \frac{\partial x}{\partial \eta_2} \\ \frac{\partial y}{\partial \eta_1} & \frac{\partial y}{\partial \eta_2} \end{bmatrix} \begin{bmatrix} d\eta_1 \\ d\eta_2 \end{bmatrix}.$$  

(C.33)

Inverting the system (C.33) we get

$$\begin{bmatrix} d\eta_1 \\ d\eta_2 \end{bmatrix} = \frac{1}{J} \begin{bmatrix} \frac{\partial y}{\partial \eta_2} & -\frac{\partial x}{\partial \eta_2} \\ -\frac{\partial y}{\partial \eta_1} & \frac{\partial x}{\partial \eta_1} \end{bmatrix} \begin{bmatrix} dx \\ dy \end{bmatrix}. $$

(C.34)

From (C.34) the partial derivatives $\frac{\partial \eta_i}{\partial (x, y)}$ are readily obtained simply by allowing only the
spatial direction in question to vary while holding the other constant

\[
\begin{align*}
\frac{\partial \eta_1}{\partial x} &= + \frac{1}{J} \frac{\partial y}{\partial \eta_2} ; \\
\frac{\partial \eta_2}{\partial x} &= - \frac{1}{J} \frac{\partial y}{\partial \eta_1} \\
\frac{\partial \eta_1}{\partial y} &= \frac{1}{J} \frac{\partial y}{\partial \eta_1} ; \\
\frac{\partial \eta_2}{\partial y} &= + \frac{1}{J} \frac{\partial y}{\partial \eta_1} 
\end{align*}
\] (C.35)

(C.36)

Here again, the derivatives \( \frac{\partial (x, y)}{\partial \eta} \) are calculated using (C.32).

### C.3.2 Quadrature

The elemental integral (C.22) can no longer be evaluated symbolically and must be effected numerically. This is done based on Gauss quadrature in which the integrand is evaluated at specific points in the triangular domain and multiplied by certain weights (see, for example, [Kikuchi 1986]). Let \( \eta_q, \omega_q, q = 1, \ldots, Q \), be the \( q^{th} \) quadrature point coordinates and associated weight respectively. Let

\[
G_{qki} = \frac{\partial h_i}{\partial \eta_k}(\eta_q),
\] (C.37)

\[
b_{qk} = (-1)^{k+1} \sum_{i=1}^{6} G_{qki} y_i,
\] (C.38)

\[
c_{qk} = (-1)^{k} \sum_{i=1}^{6} G_{qki} x_i,
\] (C.39)

then the integral (C.22) can be evaluated as follows

\[
\int_{\Omega_k} \nabla h_i \nabla h_j dx = \frac{1}{2} \sum_{q=1}^{Q} w_q \left[ \frac{1}{J_a h_a(\eta_q)} [G_{qki} b_{qk} G_{qti} b_{qt} + G_{qmi} c_{qm} G_{qn} c_{qn}] \right]
\] (C.40)

where summation over repeated indices is assumed. The \( \frac{1}{2} \) factor appearing in front is based on a convention according to which the weights are multiplied by factor of 2. The origin of this convention, the author believes, is to avoid multiplying by 2 in the subparametric case when using Gauss integration to evaluate the integrals (recall that the jacobian in that case is 2A).

**Remark** It is important to note that one can, in fact, evaluate the Jacobian directly at the quadrature points without the need to use the approximation (C.31), which should also be more accurate. In this work, however, we have implemented (C.40) identically.

### C.3.3 Stokes Operators

In addition to the Laplacian operator, the (pressure) derivative operators \( D_i^T, i = 1, 2 \), defined in (4.47), are also needed for the Stokes problem. The elemental isoparametric discretizations of the derivative operators, \( D_i^T \), and \( D_2^T \) take the forms:

\[
D_{ij1} = \int_{\Omega_k} \frac{\partial h_i}{\partial x} \hat{h}_j dx = \frac{1}{2} \sum_{q=1}^{Q} w_q G_{qmi} b_{qm} (\eta_j)_q
\] (C.41)
and

\[ D_{ij} = \int_{\Omega^k} \frac{\partial h_i}{\partial y} \hat{h}_j \, dx = \frac{1}{2} \sum_{q=1}^{Q} w_q G_{qmi} c_{qm} (\eta_j)_q \]  

(C.42)

where summation over repeated indices is assumed. Here \( \hat{h}_j \) are the linear shape functions or simply the barycentric variables \( \eta_i, i = 1, 2 \). Note that we have used the fact that \( \hat{h}_j (\eta_q) = (\eta_j)_q \), where \( (\eta_j)_q \) is the \( j^{th} \) coordinate of the quadrature point \( \eta_q \). Note also that the Jacobian cancels out from the integrals. In fact, we get \( |J| \) which is unity due to our right-handed elements.
Appendix D

3rd–Order Adams–Bashforth with Variable Time Stepping

The family of Adams–Bashforth, A-B, schemes are explicit multipoint methods. A k–order accurate A-B scheme requires k function evaluations. The 3rd–order A-B scheme can be derived in a variety of ways. For example, in the case of uniform time step Δt, by fitting a third–degree Newton backward-difference polynomial to data points at n – 2, n – 1, and n and integrating from point n to point n + 1 (see, for example, Hoffman [1992]). This yields the formula:

\[ y_{n+1} = y_n + \frac{\Delta t}{12} (23f_n - 16f_{n-1} + 5f_{n-2}) \]  \hspace{1cm} (D.1)

The corresponding stability diagram for \( y_t = \lambda y \) is shown in Figure D-1. It intersects the imaginary axis at approximately 0.723. This fact makes the scheme suitable for solving the pure convection equation \( u_t = u u_x \) provided that

\[ |\lambda_{\max}| \Delta t = \frac{\max(u)}{h} \Delta t < 0.723 \]  \hspace{1cm} (D.2)

which is known as the Courant condition. Here \( |\lambda_{\max}| \) denotes the magnitude of the maximum eigenvalue of the discrete convection operator, and h is the mesh spacing associated with the spatial discretization of the 1st–order derivative operator \( d/dx \). The scheme is also preferred over, say, the leap–frog whose stability region includes a larger portion of the imaginary axis, \((+t \to -t)\) because: first it is third order accurate whereas the latter is only second order; and second, the spurious (numerical) modes associated with the higher-order discretization are order \( \lambda_{\max} \Delta t \) in the former, but order \((-1 + \lambda_{\max} \Delta t)\) in the latter, and hence, the start up is not crucial for A-B as it is for the leap–frog.

In practice, uniform time stepping is not always economical. Particularly if the convective velocity varies significantly over time. The use of a constant time step based on the maximum value of the velocity over the entire history is unnecessarily restrictive, and it is more beneficial to use variable timestepping as dictated by the Courant condition (D.2). This is done by computing a new \( \Delta t \) based on (D.2) and comparing against the current time step. If (D.2) predicts a smaller stable value then the current time step is updated for the next iteration. (In practice, this check need only be done every few, say 5, iterations.) In the latter case we need to derive new coefficients for 3rd–order A-B based on variable timestepping, in order to retain the 3rd–order accuracy.

To begin, we consider the nonuniform finite difference grid depicted in Figure D-2). We
Figure D-1: Stability diagram for 3rd-order Adams–Bashforth with uniform time-stepping, equation D.1.

write a forward Taylor series expansion around $y_n$:

$$y_{n+1} = y_n + \Delta t^n f_n + \frac{\Delta t^{n^2}}{2} f'_n + \frac{\Delta t^{n^3}}{6} f''_n + O(\Delta t^4) \quad (D.3)$$

next we use a backward difference to approximate $f'_n$ and $f''_n$:

$$f'_n = \frac{f_n - f_{n-1}}{\Delta t^{n-1}} + \frac{\Delta t^{n-1}}{2} f''_n + O(\Delta t^2) \quad (D.4)$$

$$f''_n = \frac{f_{n-2} - f_n + (\Delta t^{n-1} + \Delta t^{n-1}) f'_n}{(\Delta t^{n-1} + \Delta t^{n-1})^2/2} + O(\Delta t) \quad (D.5)$$

equations D.4 and D.5 are obtained by using a backward Taylor series expansions around the point $n$ to find $f_{n-1}$ and $f_{n-2}$, then solving for $f'_n$ and $f''_n$ respectively. Inserting equations D.4 and D.5 into equation D.3, retaining order $\Delta t^3$ and less terms only, and
simplifying, we arrive at the following general expression for 3rd-order A-B scheme:

\[
y_{n+1} = y_n + \frac{\Delta t^n}{12} \left[ \frac{12\Delta t^{n-1}(\Delta t^{n-1} + \Delta t^{n-1}) + 6\Delta t^n(2\Delta t^{n-1} + \Delta t^{n-1}) + 4\Delta t^{n+1}}{\Delta t^{n-1}(\Delta t^{n-1} + \Delta t^{n-1})} f_n \\
- \frac{6\Delta t^n(\Delta t^{n-1} + \Delta t^{n-1}) + 4\Delta t^{n+2}}{\Delta t^{n-1}\Delta t^{n-1}} f_{n-1} \\
+ \frac{6\Delta t^n\Delta t^{n-1} + 4\Delta t^{n+2}}{(h_1 + h_2)h_2} f_{n-2} \right]
\]

(D.6)

As a check, by letting \( \Delta t^{n-1} = \Delta t^{n-1} = \Delta t^n = \Delta t \), we retrieve equation D.1.
Appendix E

Direct Stiffness Summation Procedure

In this appendix we provide a detailed description of the direct stiffness summation, DSS, algorithm implemented in our calculations. We will also address some aspects of our data structure which make it more efficient to an alternative approach [Cruz 1993] in terms of memory requirement and operation count.

Remark This algorithm is designed for a distributed memory parallel machine, in which each processor has access to its own memory. Therefore, in order to avoid confusion, we shall assume that all the presented data structures are allocated on all the processors without explicitly mentioning this point any further.

We begin by defining the mesh nomenclature:

- \( g_i^k \): global number of local node \( i \) of element \( k \).
- \( K^m \): number of elements in a subcell \( m \). (Recall that \( M \) subcells are distributed on each processor. In the actual runs, \( M \) was set to 1, however we shall present here the general algorithm based on \( M > 1 \).
- \( n_t \): 3 for a linear mesh; 6 for a quadratic mesh.

A general data structure for a field variable is denoted \( u \) and is locally defined as:

- \( u(m = 1, M; i = 1, n_t; k = 1, K^m) \) value of field at local node \( i \) of element \( k \) in subcell \( m \).

We remark here that for the purpose of exposition we will use multidimensional arrays. Naive copying of the arrays, however, can result in excessive waste of memory. For example, not all subcells have the same number of elements. In the actual implementation C–based structures which are dynamically allocated are employed for the contiguous and efficient use of memory. Also the order of any nested loops is listed in the most readable fashion. Shuffling of loops is required to take advantage of pipe lining, vector considerations, as well as memory access time in general.

The serial–type DSS implementation can be performed with the following loops (here we need an additional working array, \( G \), to hold the correct global values):

\[
\text{zero } G \\
\text{for } (k = 1; k \leq K^m) \\
\]
for \((i = 1; i \leq n_i)\) {
    \(G(g^k_t) = G(g^k_t) + u(1, i, k)\)
}
\}
write back to \(u\)
for \((k = 1; k \leq K^m)\) {
    for \((i = 1; i \leq n_i)\) {
        \(u(1, i, k) = G(g^k_t)\)
    }
}\)

Next we describe the nomenclature and the data structures for local edges and vertices at the subcell level (see Figure 4-20):

- \(le_m, lv_m\): local edge and local vertex of subcell \(m\), respectively.
- \(N_{le_m}, N_{lv_m}\): number of local edges and local vertices of subcell \(m\) respectively. Note that \(N_{le_m}\) and \(N_{lv_m}\) are normally four for a regular square. Here they can take any value between zero and four depending on the particular geometry.

For each subcell \(m\), we construct for each \(le_m\) and \(lv_m\) the following tables: (will limit the list to local edges, since local vertices have analogous tables)

- \(nNodes(m = 1, M; i = 1, N_{le_m})\) gives number of global nodes on local edge \(i\).
- \(gikList(m = 1, M; i = 1, N_{le_m}; j = 1, nNodes(m, i))\) gives the global node number of local node \(j\) on local edge \(i\). (This array is stored temporarily to construct the next three tables and then destroyed.)
- \(ndMult(m = 1, M; i = 1; N_{le_m}; j = 1, nNodes(m, i))\) gives the number of elements that have a coincident node with node \(j\) of local edge \(i\).
- \(eleList(m = 1, M; i = 1, N_{le_m}; j = 1, nNodes(m, i); k = 1, ndMult(m, i, j))\) gives the global element number of the local \(k^{th}\) element that have a coincident node with node \(j\) of local edge \(i\).
- \(eleNdN(m = 1, M; i = 1; N_{le_m}; j = 1, nNodes(m, i); k = 1, ndMult(m, i, j))\) gives the local node number of the local \(k^{th}\) element which is coincident with node \(j\) of edge \(i\).

For purpose of communications, we assign a unique global number to each couple of shared local edges and another one for each quadruple shared local vertices. Thus for each subcell \(m\), we construct the following mappings:

- \(M_{le2ge}(m = 1, M; i = 1, N_{le_m})\) gives the global edge number of local edge \(i\).
- \(M_{lv2gv}(m = 1, M; i = 1, N_{lv_m})\) gives the global edge number of local vertex \(i\).

One last crucial piece of information to successfully complete message transfer is to identify locally for a given global edge number the processor number on which the partner subcell sharing the same global edge resides. Therefore for each subcell \(m\) we need the additional mapping:

- \(M_{ge2ge}(i = 1, N_{ge})\) gives the target processor number contributing to the same global edge \(i\). \(N_{ge}\) is the total number of global edges.
Equipped with the aforementioned mappings and tables we are in a position to describe the complete parallel DSS procedure, first we require two more working buffers in addition to $G$ used in the serial DSS

- $geBuf(i = 1, N_{ge}; j = 1, No. of Nodes on i)$ this buffer will hold the correct value for each node $j$ on global edge $i$ to be redistributed to the field local nodes. Due to our counterclockwise numbering of local edges, local edges contributing to a global edge will have opposite directions. We adopt the convention that $geBuf$ matches the direction of local edges 1 and 2, Thus for edges 3 and 4 it will hold the correct values in a reverse order.

- $geStatus(i = 1, N_{ge})$ this flag is zero for untouched global edge $i$ buffer, one of partially filled buffer (one local edge contribution only), and two for completely correctly filled buffer.

On each processor $p$, the parallel DSS procedure comprises the following steps:

1. direct stiffness local subcells.

   for $(m = 1; m \leq M) \{
       \text{invoke serial DSS}
   \}

2. direct stiffness global vertices. Since the number of vertices is typically small (upper bounded by the grid size) we distribute all vertices to all processors, with global gather/scatter summation which is dominated by a constant times $t_{comm(1)log_2 P}$.

   zero $G$
   for $(m = 1; m \leq M) \{
       \text{for (i = 1; N_{tv_m}) \{}
           \begin{itemize}
           \item get $g^k_n, n, k$ from $M_{tv2gv}$ and tables $eleN\!d\!N, eleList$.
           \item $G(g^k_n) = G(g^k_n) + u(m, n, k)$
           \end{itemize}
       \}
   \}
   - invoke global gather/scatter summation for $G$.

   for $(m = 1; m \leq M) \{
       \text{for (i = 1; N_{tv_m}) \{}
           \begin{itemize}
           \item get $g^k_n$
           \item write back $G(g^k_n)$ to the field $u$ using $ndMult, eleList, eleN\!d\!N$.
           \end{itemize}
       \}
   \}

3. direct stiffness global edges.

   zero $geStatus$
   for $(m = 1; m \leq M) \{
       \text{for (i = 1; N_{le_m}) \{}
           \begin{itemize}
           \item extract field values at edge nodes and store in $G$ in forward
order for $i < 3$ and in reverse order for $i > 2$ using tables
$nNodes, eleNdN, eleList$ for $le_m$.
- get global edge no, $gen$, using mapping $M_{le2ge}$
- check $geStatus(gen)$:
  - if zero: copy $G$ to $geBuf(gen,:)$, set $geStatus(gen)$ to one.
  - if one: add $G$ to $geBuf(gen,:)$, set $geStatus(gen)$ to two.
- check mapping $p = M_{ge2p}(gen)$:
  - if $p ≠ myProcessor$:
    - send a message of type $gen$ to $p$ containing buffer $G$,
    - check for an incoming message of type $gen$.
      - if available:
        - receive in buffer $G$,
        - add it to $geBuf(gen,:)$, set $geStatus(gen)$ to two.
  
}synchronize all processors.

At this point all messages have been sent but not all have been received, a second loop is necessary to receive remaining messages.

for $(m = 1; m ≤ M)$ {
  for $(i = 1; N_{le_m})$
    - get $gen$ using mapping $M_{le2ge}$
    - check $geStatus(gen)$:
      - if one:
        - receive an incoming message of type $gen$, in $G$,
        - if no message available: an error has occurred.
        - add it to $geBuf(gen,:)$, set $geStatus(gen)$ to two.
  
}

The $geStatus$ buffer should contain two for all global edges, this can be used to check for any errors. At this point all the global edges buffers contain the correct values at the local edges nodes. It remains to redistribute the values to the field.

for $(m = 1; m ≤ M)$ {
  for $(i = 1; N_{le_m})$
    - get $gen$ using mapping $M_{le2ge}$
    - if $(i < 3)$:
      - write $geBuf(gen,:)$ to $u$ in forward order
        using $nNodes, eleNdN, eleList$ for $le_m$.
    - if $(i > 2)$:
      - write $geBuf(gen,:)$ to $u$ in reverse order
        using $nNodes, eleNdN, eleList$ for $le_m$.
  
}
Before we conclude this Appendix we will present briefly an alternative data structure for the field variable \( u \), that has been implemented by Cruz [1993]. To simplify the discussion we will assume a regular square domain with a structured mesh (see Figure E). The domain has four edges each with \( N - 2 \) degrees of freedom, four vertices that delimit the edges but are not included in the edges, and \( K \) triangular elements each with \( n_t \) local nodes. The proposed data structure then takes the form \( u(i = 1, N - 2 ; j = 1, 8 + K) \) where

\[
\begin{align*}
  &i = 1, 1; j = 1, 4 \\
  &i = 1, N - 2; j = 5, 8 \\
  &i = 1, n_t; j = 9, K
\end{align*}
\]

: locations for four vertices.

: locations for four edges.

: locations for \( K \) elements.

![Figure E-1: Regular square domain.](image)

For each node, \((i, j)\), the following mappings are constructed:

- \( \tilde{n}(i, j) \): gives number of geometrically coincident nodes at node \((i, j)\).

- \( i^*(p = 1, \tilde{n}(i, j); i; j) \): gives local node number of element \( p \) that coincides with node \((i, j)\).

- \( j^*(p = 1, \tilde{n}(i, j); i; j) \): gives global element number of local element number \( p \) which has a geometrically coincident node at node \((i, j)\).

The serial DSS is performed as follows: (here the working array \( G \) needs to be of the same dimension of the field \( u \))

\[
\text{zero } G \\
\text{for } (k = 1; k \leq K) \{ \\
  \text{for } (n = 1; n \leq 4) \{ \\
    G(n, k) = u(n, k) \\
    \text{for } (p = 1; \tilde{n}(n, k)) \{ \\
      n_o = i^*(p, n, k) \\
      k_o = j^*(p, n, k) \\
      G(n, k) = G(n, k) + u(n_o, k_o)
  \}
\}\}
\]
copy $G$ back to $u$

The motivation behind this data structure is to group communicating structures vertices/edges in a contiguous memory at the beginning of the field variable. There are two serious drawbacks however, associated with such implementation:

- The memory requirement, assuming a typical multiplicity of 5 per node, is at least 12 times the size of the field variable $u$. This can significantly limit the maximum size of the problem that can solved with a limited available memory. In our implementation, no major memory is required beside the field variable and the working array $G$. The tables and buffers we construct are on the order of the boundary nodes which is only a fraction of the entire field.

- The serial DSS procedure requires 2 indirect addressing while the $g^k_i$-based DSS requires only one indirect addressing. For larger problems this can be reflected negatively on the overall performance.
Appendix F

Raw Data

F.1 Random Number Generators and Acceptance–Rejection Algorithm

The random number generator employed in the realization generation throughout most of this thesis is called RANDOM which comes with the standard Unix implementation (see the manual page for the Digital Ultrix random function). The RANDOM subroutine uses a non-linear additive feedback random number generator algorithm which is described in Park and Miller [1988]. The algorithm employs a default table of size 31 long integers to return successive pseudo-random numbers in the range from 0 to $(2^{**31})-1$ (which is normalized to $[0, 1]$). The period of this random number generator is very large, approximately $16*(2^{**31})-1$, and the random number generator is known to pass the random statistical tests (see Park and Miller [1988]).

For the effective conductivity surrogate validation, we employed a different random number generator: “RAN2” of the Numerical Recipes book [Press et al. 1992], which is based on the L’Ecuyer algorithm with Bays-Durham shuffle and added safeguards [L’Ecuyer 1988]. RAN2 has a long period ($> 2 \times 10^{18}$), is known to pass the random statistical tests [Press et al. 1992], and is highly recommended by the Numerical Recipes authors. RAN2 returns a random number in the open interval $(0, 1)$. To draw the concentration validation points in the range $[.05, .5]$, the withdrawn random number is multiplied by .45 and added to .05.

The generation of a random realization is performed as follows:

1. A pair of random numbers is generated in the box $(0, 1) \times (0, 1)$ by two successive calls to the random number generator.

2. The pair is scaled by the supercell size $\lambda$, and the corresponding point $y_i$ is now a candidate inclusion center in $\Omega$.

3. The newly generated point $y_i$ is accepted if the distance between $y_i$ and each of the previously accepted points is larger or equal to the inclusion diameter $d$. If not, the point is rejected.

4. The procedure is repeated until the required number of inclusions is obtained.

F.2 Effective Conductivity Surrogate Validation Raw Data
<table>
<thead>
<tr>
<th>$c$</th>
<th>$\bar{K} = k(c, \lambda_0 = 6)$</th>
<th>$\sigma^2_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.052E-01</td>
<td>5.018E-01</td>
<td>3.500E-04</td>
</tr>
<tr>
<td>3.985E-01</td>
<td>3.858E-01</td>
<td>2.053E-04</td>
</tr>
<tr>
<td>4.920E-01</td>
<td>2.830E-01</td>
<td>2.752E-04</td>
</tr>
<tr>
<td>9.212E-02</td>
<td>8.280E-01</td>
<td>1.919E-04</td>
</tr>
<tr>
<td>4.503E-01</td>
<td>3.333E-01</td>
<td>3.513E-04</td>
</tr>
<tr>
<td>3.091E-01</td>
<td>4.935E-01</td>
<td>2.856E-04</td>
</tr>
<tr>
<td>1.150E-01</td>
<td>7.896E-01</td>
<td>8.696E-05</td>
</tr>
<tr>
<td>7.123E-02</td>
<td>8.671E-01</td>
<td>5.033E-05</td>
</tr>
<tr>
<td>3.334E-01</td>
<td>4.681E-01</td>
<td>7.276E-04</td>
</tr>
<tr>
<td>1.018E-01</td>
<td>8.099E-01</td>
<td>9.157E-05</td>
</tr>
<tr>
<td>3.716E-01</td>
<td>4.132E-01</td>
<td>4.271E-04</td>
</tr>
<tr>
<td>4.524E-01</td>
<td>3.345E-01</td>
<td>2.249E-04</td>
</tr>
<tr>
<td>4.641E-01</td>
<td>3.191E-01</td>
<td>2.629E-04</td>
</tr>
<tr>
<td>3.751E-01</td>
<td>4.141E-01</td>
<td>3.831E-04</td>
</tr>
<tr>
<td>6.678E-02</td>
<td>8.738E-01</td>
<td>5.108E-05</td>
</tr>
<tr>
<td>3.935E-01</td>
<td>3.956E-01</td>
<td>4.017E-04</td>
</tr>
<tr>
<td>4.709E-01</td>
<td>3.099E-01</td>
<td>2.456E-04</td>
</tr>
<tr>
<td>4.606E-01</td>
<td>3.238E-01</td>
<td>3.781E-04</td>
</tr>
<tr>
<td>4.234E-01</td>
<td>3.601E-01</td>
<td>4.256E-04</td>
</tr>
<tr>
<td>1.156E-01</td>
<td>7.890E-01</td>
<td>1.773E-04</td>
</tr>
<tr>
<td>4.804E-01</td>
<td>2.974E-01</td>
<td>3.632E-04</td>
</tr>
<tr>
<td>4.832E-01</td>
<td>2.946E-01</td>
<td>5.168E-04</td>
</tr>
<tr>
<td>1.017E-01</td>
<td>8.110E-01</td>
<td>1.199E-04</td>
</tr>
<tr>
<td>2.407E-01</td>
<td>5.984E-01</td>
<td>2.822E-04</td>
</tr>
<tr>
<td>8.401E-02</td>
<td>8.461E-01</td>
<td>7.721E-05</td>
</tr>
<tr>
<td>3.612E-01</td>
<td>4.279E-01</td>
<td>1.582E-04</td>
</tr>
<tr>
<td>7.520E-02</td>
<td>8.587E-01</td>
<td>9.600E-05</td>
</tr>
<tr>
<td>4.348E-01</td>
<td>3.504E-01</td>
<td>3.830E-04</td>
</tr>
<tr>
<td>4.987E-01</td>
<td>2.771E-01</td>
<td>3.457E-04</td>
</tr>
<tr>
<td>3.087E-01</td>
<td>5.044E-01</td>
<td>3.665E-04</td>
</tr>
<tr>
<td>2.981E-01</td>
<td>5.179E-01</td>
<td>2.667E-04</td>
</tr>
<tr>
<td>4.156E-01</td>
<td>3.643E-01</td>
<td>2.666E-04</td>
</tr>
<tr>
<td>2.441E-01</td>
<td>5.851E-01</td>
<td>4.354E-04</td>
</tr>
<tr>
<td>2.242E-01</td>
<td>6.190E-01</td>
<td>1.166E-04</td>
</tr>
<tr>
<td>1.447E-01</td>
<td>7.419E-01</td>
<td>1.733E-04</td>
</tr>
</tbody>
</table>

Table F.1: Concentrations, sample means and variances for the 45 random validation points of the effective conductivity surrogate presented in Chapter 7. Only the first 35 points are listed here. The remaining 10 validation points are continued in Table F.2.
<table>
<thead>
<tr>
<th>$c$</th>
<th>$\bar{K} = k_e(c, \lambda_0 = 6)$</th>
<th>$\sigma^2_K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.578E-01</td>
<td>7.116E-01</td>
<td>3.031E-04</td>
</tr>
<tr>
<td>2.731E-01</td>
<td>5.522E-01</td>
<td>1.833E-04</td>
</tr>
<tr>
<td>2.312E-01</td>
<td>6.048E-01</td>
<td>2.319E-04</td>
</tr>
<tr>
<td>1.370E-01</td>
<td>7.476E-01</td>
<td>2.089E-04</td>
</tr>
<tr>
<td>2.374E-01</td>
<td>5.995E-01</td>
<td>4.623E-04</td>
</tr>
<tr>
<td>2.342E-01</td>
<td>5.994E-01</td>
<td>2.037E-04</td>
</tr>
<tr>
<td>9.195E-02</td>
<td>8.277E-01</td>
<td>1.498E-04</td>
</tr>
<tr>
<td>4.617E-01</td>
<td>3.214E-01</td>
<td>2.235E-04</td>
</tr>
<tr>
<td>4.415E-01</td>
<td>3.406E-01</td>
<td>2.141E-04</td>
</tr>
<tr>
<td>4.306E-01</td>
<td>3.484E-01</td>
<td>2.890E-04</td>
</tr>
</tbody>
</table>

Table F.2: Continuation of Table F.1.
### F.3 Creeping-Flow Permeability Surrogate Validation Raw Data

<table>
<thead>
<tr>
<th>( c )</th>
<th>( K = \hat{K}(c, \lambda_0 = 6) )</th>
<th>( \sigma_{K}^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10</td>
<td>3.622E-01</td>
<td>1.263E-02</td>
</tr>
<tr>
<td>0.12</td>
<td>3.337E-01</td>
<td>1.784E-02</td>
</tr>
<tr>
<td>0.14</td>
<td>1.980E-01</td>
<td>1.321E-03</td>
</tr>
<tr>
<td>0.16</td>
<td>1.438E-01</td>
<td>9.891E-04</td>
</tr>
<tr>
<td>0.18</td>
<td>1.257E-01</td>
<td>1.981E-03</td>
</tr>
<tr>
<td>0.20</td>
<td>8.793E-02</td>
<td>7.080E-04</td>
</tr>
<tr>
<td>0.22</td>
<td>6.690E-02</td>
<td>8.792E-04</td>
</tr>
<tr>
<td>0.24</td>
<td>5.259E-02</td>
<td>2.476E-04</td>
</tr>
<tr>
<td>0.26</td>
<td>3.715E-02</td>
<td>1.135E-04</td>
</tr>
<tr>
<td>0.28</td>
<td>3.161E-02</td>
<td>7.885E-05</td>
</tr>
<tr>
<td>0.30</td>
<td>2.424E-02</td>
<td>8.609E-05</td>
</tr>
<tr>
<td>0.32</td>
<td>1.767E-02</td>
<td>2.142E-05</td>
</tr>
<tr>
<td>0.34</td>
<td>1.516E-02</td>
<td>3.847E-05</td>
</tr>
<tr>
<td>0.36</td>
<td>1.189E-02</td>
<td>1.069E-05</td>
</tr>
<tr>
<td>0.38</td>
<td>9.943E-03</td>
<td>7.553E-06</td>
</tr>
<tr>
<td>0.40</td>
<td>7.179E-03</td>
<td>3.512E-06</td>
</tr>
<tr>
<td>0.42</td>
<td>6.372E-03</td>
<td>3.171E-06</td>
</tr>
<tr>
<td>0.44</td>
<td>4.964E-03</td>
<td>2.911E-06</td>
</tr>
<tr>
<td>0.46</td>
<td>3.363E-03</td>
<td>5.249E-07</td>
</tr>
<tr>
<td>0.48</td>
<td>2.972E-03</td>
<td>1.312E-06</td>
</tr>
<tr>
<td>0.50</td>
<td>1.969E-03</td>
<td>4.030E-07</td>
</tr>
<tr>
<td>0.52</td>
<td>1.720E-03</td>
<td>4.947E-07</td>
</tr>
<tr>
<td>0.54</td>
<td>1.099E-03</td>
<td>2.040E-07</td>
</tr>
<tr>
<td>0.56</td>
<td>8.101E-04</td>
<td>1.908E-07</td>
</tr>
<tr>
<td>0.58</td>
<td>6.428E-04</td>
<td>7.614E-08</td>
</tr>
</tbody>
</table>

Table F.3: The table presents the sample mean permeability values and associated variances at the listed concentrations for \( \lambda = 7 \) used in Chapter 8.
### F.4 Inertial Porous–Media Flow Raw Data

#### F.4.1 Regular Array Results

<table>
<thead>
<tr>
<th>$c$</th>
<th>$Re_\rho$</th>
<th>$\kappa$</th>
<th>$Re_{&lt;u_1&gt;}$</th>
<th>Steady/Unsteady</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7.620E-2</td>
<td>0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>17.2</td>
<td>6.602E-2</td>
<td>19.5</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>25.2</td>
<td>6.141E-2</td>
<td>40.0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>30.0</td>
<td>6.055E-2</td>
<td>54.5</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>.2</td>
<td>5.752E-2</td>
<td>93.9</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>40.4</td>
<td>5.644E-2</td>
<td>114.3</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>45.0</td>
<td>5.532E-2</td>
<td>138.3</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>50.0</td>
<td>5.346E-2</td>
<td>150.2</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>53.0</td>
<td>5.164E-2</td>
<td>168.4</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>57.2</td>
<td>2.544E-2</td>
<td>0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>29.9</td>
<td>2.221E-2</td>
<td>19.9</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>.3</td>
<td>2.042E-2</td>
<td>39.9</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>71.5</td>
<td>1.895E-2</td>
<td>96.9</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>101.3</td>
<td>1.656E-2</td>
<td>169.9</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>3.009E-3</td>
<td>0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>50.4</td>
<td>7.963E-3</td>
<td>20.2</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>.4</td>
<td>7.221E-3</td>
<td>40.5</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>122.2</td>
<td>6.404E-3</td>
<td>95.6</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>175.7</td>
<td>5.587E-3</td>
<td>172.5</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>2.949E-3</td>
<td>0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>89.2</td>
<td>2.617E-3</td>
<td>20.8</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>.5</td>
<td>2.315E-3</td>
<td>42.1</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>134.9</td>
<td>2.055E-3</td>
<td>105.3</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>226.4</td>
<td>1.750E-3</td>
<td>208.2</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>7.424E-4</td>
<td>0</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>181.3</td>
<td>6.742E-4</td>
<td>22.2</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>.6</td>
<td>5.885E-4</td>
<td>44.1</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>460.6</td>
<td>4.867E-4</td>
<td>103.2</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>700.8</td>
<td>3.878E-4</td>
<td>190.4</td>
<td>U</td>
<td></td>
</tr>
</tbody>
</table>

Table F.4: Permeability results for the square array at the indicated concentrations and Reynolds numbers.
<table>
<thead>
<tr>
<th>c</th>
<th>$Re_p$</th>
<th>$\kappa$</th>
<th>$Re_{&lt;u_1&gt;}$</th>
<th>Steady/Unsteady</th>
</tr>
</thead>
<tbody>
<tr>
<td>.3</td>
<td>31.1</td>
<td>2.081E-2</td>
<td>0</td>
<td>S</td>
</tr>
<tr>
<td>.3</td>
<td>48.4</td>
<td>1.748E-2</td>
<td>20.1</td>
<td>S</td>
</tr>
<tr>
<td>.3</td>
<td>97.3</td>
<td>1.311E-2</td>
<td>40.9</td>
<td>S</td>
</tr>
<tr>
<td>.3</td>
<td>124.7</td>
<td>1.102E-2</td>
<td>124.1</td>
<td>S</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1.056E-2</td>
<td>0</td>
<td>S</td>
</tr>
<tr>
<td>.4</td>
<td>48.4</td>
<td>8.713E-3</td>
<td>20.4</td>
<td>S</td>
</tr>
<tr>
<td>.4</td>
<td>70.0</td>
<td>7.481E-3</td>
<td>36.6</td>
<td>S</td>
</tr>
<tr>
<td>.4</td>
<td>161.6</td>
<td>4.784E-3</td>
<td>124.9</td>
<td>S</td>
</tr>
<tr>
<td>.4</td>
<td>245.3</td>
<td>3.442E-3</td>
<td>207.1</td>
<td>S</td>
</tr>
<tr>
<td>.5</td>
<td>0</td>
<td>4.108E-3</td>
<td>0</td>
<td>S</td>
</tr>
<tr>
<td>.5</td>
<td>75.0</td>
<td>3.653E-3</td>
<td>20.5</td>
<td>S</td>
</tr>
<tr>
<td>.5</td>
<td>112.8</td>
<td>3.160E-3</td>
<td>40.2</td>
<td>S</td>
</tr>
<tr>
<td>.5</td>
<td>246.6</td>
<td>2.025E-3</td>
<td>123.1</td>
<td>S</td>
</tr>
<tr>
<td>.5</td>
<td>345.8</td>
<td>1.525E-3</td>
<td>182.3</td>
<td>U</td>
</tr>
<tr>
<td>.6</td>
<td>0</td>
<td>1.451E-3</td>
<td>0</td>
<td>S</td>
</tr>
<tr>
<td>.6</td>
<td>123.9</td>
<td>1.354E-3</td>
<td>20.8</td>
<td>S</td>
</tr>
<tr>
<td>.6</td>
<td>194.6</td>
<td>1.200E-3</td>
<td>45.4</td>
<td>S</td>
</tr>
<tr>
<td>.6</td>
<td>356.2</td>
<td>9.092E-4</td>
<td>115.3</td>
<td>S</td>
</tr>
<tr>
<td>.6</td>
<td>476.3</td>
<td>7.314E-4</td>
<td>165.9</td>
<td>U</td>
</tr>
</tbody>
</table>

Table F.5: Permeability results for the hexagonal array at the indicated concentrations and Reynolds numbers.
### F.4.2 Random Array Results

<table>
<thead>
<tr>
<th>$c$</th>
<th>$Re_p$</th>
<th>$\kappa(c, Re_{&lt;u_1&gt;}, \lambda = 7)$</th>
<th>$Re_{&lt;u_1&gt;}$</th>
<th>Steady/Unsteady</th>
<th>$t_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>9.5861E-2</td>
<td>0</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>6.2602E-2</td>
<td>18.1</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>.2</td>
<td>25</td>
<td>4.9754E-2</td>
<td>31.1</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>3.6794E-2</td>
<td>58.9</td>
<td>U</td>
<td>16.96</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>2.7473E-2</td>
<td>0</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>1.7954E-2</td>
<td>16.2</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>.3</td>
<td>45</td>
<td>1.3931E-2</td>
<td>28.2</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>1.0161E-2</td>
<td>60.0</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>100</td>
<td>7.0291E-3’</td>
<td></td>
<td>70.3</td>
<td>U</td>
<td>29.18</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>1.3925E-2</td>
<td>0</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>9.5538E-3</td>
<td>23.9</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>.4</td>
<td>75</td>
<td>7.7745E-3</td>
<td>43.7</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>122</td>
<td>5.8290E-3</td>
<td></td>
<td>86.7</td>
<td>U</td>
<td>10.79</td>
</tr>
<tr>
<td>.5</td>
<td>90</td>
<td>1.8017E-3</td>
<td>14.6</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>135</td>
<td></td>
<td>1.2816E-3</td>
<td>23.4</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>225</td>
<td></td>
<td>1.1223E-3</td>
<td>56.8</td>
<td>S</td>
<td>–</td>
</tr>
<tr>
<td>345</td>
<td></td>
<td>7.5997E-4</td>
<td>90.5</td>
<td>U</td>
<td>11.01</td>
</tr>
</tbody>
</table>

Table F.6: Upper-bound permeability results for random arrays at the indicated concentrations, supercell sizes, and Reynolds numbers. For the unsteady runs, the results represent spatial averages only at the indicated final integration times, $t_f$, except for the calculation marked with ' which is temporally averaged as well.
Figure F-1: The upper-bound realization used for inertial permeability calculations at concentration $c = 0.200$ and various Reynolds numbers.

Figure F-2: The upper-bound realization used for inertial permeability calculations at concentration $c = 0.200$ and various Reynolds numbers.
Figure F-3: Realization of a random array of cylinders for the concentration $c = 0.300$ and $N = 17$ inclusions.
<table>
<thead>
<tr>
<th>$c = .2$</th>
<th>$c = .3$</th>
<th>$c = .4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>1.526</td>
<td>5.473</td>
<td>2.540</td>
</tr>
<tr>
<td>6.162</td>
<td>5.350</td>
<td>5.929</td>
</tr>
<tr>
<td>1.091</td>
<td>1.809</td>
<td>2.534</td>
</tr>
<tr>
<td>2.837</td>
<td>1.053</td>
<td>.7931</td>
</tr>
<tr>
<td>1.397</td>
<td>.7726</td>
<td>4.115</td>
</tr>
<tr>
<td>6.572</td>
<td>3.515</td>
<td>5.266</td>
</tr>
<tr>
<td>3.350</td>
<td>3.199</td>
<td>2.903</td>
</tr>
<tr>
<td>4.422</td>
<td>.8605</td>
<td>1.260</td>
</tr>
<tr>
<td>5.363</td>
<td>2.897</td>
<td>6.457</td>
</tr>
<tr>
<td>6.626</td>
<td>.6082</td>
<td>5.869</td>
</tr>
<tr>
<td>5.125</td>
<td>5.726</td>
<td>5.340</td>
</tr>
<tr>
<td>.05270</td>
<td>1.715</td>
<td>.7269</td>
</tr>
<tr>
<td>1.928</td>
<td>2.565</td>
<td>4.702</td>
</tr>
<tr>
<td>4.434</td>
<td>2.961</td>
<td>2.013</td>
</tr>
<tr>
<td>.9132</td>
<td>3.501</td>
<td></td>
</tr>
</tbody>
</table>

Table F.7: Inclusion-center locations for the realizations of Table F.6 at the indicated concentrations. In these runs, the geometry was fixed and Reynolds number was varied. The upper-bound realizations are shown in Figures F-1, F-2, and F-3.
<table>
<thead>
<tr>
<th>( \text{Re}_p = 90 )</th>
<th>( \text{Re}_p = 135 )</th>
<th>( \text{Re}_p = 225 )</th>
<th>( \text{Re}_p = 345 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>( y )</td>
<td>( x )</td>
<td>( y )</td>
</tr>
<tr>
<td>4.324</td>
<td>4.376</td>
<td>1.895</td>
<td>5.667</td>
</tr>
<tr>
<td>5.453</td>
<td>6.380</td>
<td>2.087</td>
<td>1.178</td>
</tr>
<tr>
<td>2.213</td>
<td>5.441</td>
<td>1.065</td>
<td>4.641</td>
</tr>
<tr>
<td>.03193</td>
<td>.9897</td>
<td>5.092</td>
<td>5.118</td>
</tr>
<tr>
<td>.1047</td>
<td>2.568</td>
<td>4.173</td>
<td>4.527</td>
</tr>
<tr>
<td>2.224</td>
<td>4.219</td>
<td>2.593</td>
<td>.1784</td>
</tr>
<tr>
<td>1.402</td>
<td>1.630</td>
<td>.7129</td>
<td>1.257</td>
</tr>
<tr>
<td>.3677</td>
<td>5.821</td>
<td>4.526</td>
<td>.08577</td>
</tr>
<tr>
<td>5.555</td>
<td>3.039</td>
<td>.9198</td>
<td>3.503</td>
</tr>
<tr>
<td>1.962</td>
<td>2.489</td>
<td>1.713</td>
<td>2.162</td>
</tr>
<tr>
<td>3.239</td>
<td>4.283</td>
<td>4.750</td>
<td>2.922</td>
</tr>
<tr>
<td>5.291</td>
<td>4.662</td>
<td>1.353</td>
<td>.1362</td>
</tr>
<tr>
<td>3.654</td>
<td>5.673</td>
<td>3.920</td>
<td>.9298</td>
</tr>
<tr>
<td>5.788</td>
<td>1.987</td>
<td>.5447</td>
<td>2.575</td>
</tr>
<tr>
<td>3.317</td>
<td>1.797</td>
<td>3.113</td>
<td>2.680</td>
</tr>
<tr>
<td>4.414</td>
<td>6.354</td>
<td>6.416</td>
<td>4.826</td>
</tr>
<tr>
<td>1.135</td>
<td>3.253</td>
<td>4.037</td>
<td>2.037</td>
</tr>
<tr>
<td>2.391</td>
<td>1.390</td>
<td>5.506</td>
<td>.6533</td>
</tr>
</tbody>
</table>

Table F.8: Inclusion–center locations for the realizations of Table F.6 at .5 concentration and the indicated \( \text{Re}_p \). In these runs, different random geometries were used for the various Reynolds numbers.
Bibliography


