

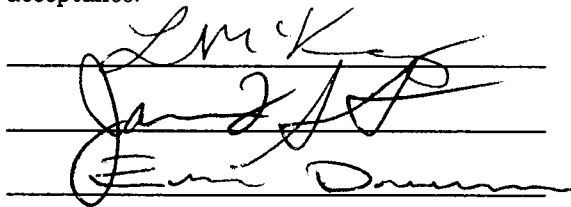
To the Graduate Council:

I am submitting herewith a thesis written by David M. Doolin entitled "Bed-normal permeability in fractured stratified rock masses." I have examined the final copy of this thesis for form and content and recommend it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Civil Engineering.



Matthew Mauldon, Major Professor

We have read this thesis and recommend its acceptance:



Accepted for the Council:



Associate Vice Chancellor
and Dean of the Graduate School

Bed-normal permeability in fractured stratified rock masses

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

David Michael Doolin
December 1996

DEDICATION

For God so loved the world, that he gave his only begotten son, that whosoever believeth in him should not perish, but have everlasting life. For God sent not his son into the world to condemn the world; but that the world through him might be saved.

John 3:16

That if thou shalt confess with thy mouth the Lord Jesus, and shalt believe in thine heart that God hath raised him from the dead, thou shalt be saved. *Romans 10:9*

ACKNOWLEDGMENTS

I would like to thank my wife, Bethany Coleman Doolin, my parents, Mr. and Mrs. Jon S. Nordvik, and my in-laws, Mr. and Mrs. Thomas E. Coleman for their gracious forbearance while I was suffering from the interlayer effect. Many thanks to Dr. Allen J. Baker with the Department of Aerospace and Engineering Science and Mechanics at the University of Tennessee, Knoxville, (UTK), who provided critical support for the finite element modeling conducted to verify the interlayer equation, and, in general, enthusiastically served as an unofficial committee member. Thanks to Drs. J. Rosinski and C. Sundberg of the UTK Dept. of Mathematics for conversation and instruction. Many thanks to Dr. Matthew Mauldon for hours of constructive criticism and collaboration, and to committee members Dr. Eric Drumm, Dr. Larry McKay and Dr. James Smoot.

Some initial portions of the analysis were funded by DOE cooperative agreement DE-FC05-92OR22056. The author thanks Drs. RaNaye Dreier and Peter Lemiszki of Oak Ridge National Laboratory for introduction to the rocks which stimulated the research.

ABSTRACT

Bed-normal permeability of a two-dimensional, fractured, stratified rock masses with steady-state, laminar, saturated flow is investigated. Fracturing is assumed to be orthogonal to stratification, and the permeability is assumed to be a function of fracture aperture and flow path length in an otherwise impermeable matrix. The flow path length through fractures takes the value of bed thickness for bed-confined fractures. Bedding planes are modeled as fictitious layers, termed *interlayers*, that conduct flow through "virtual" fractures. The aperture of virtual fractures in the interlayer is assumed equal to the bedding plane aperture. Fracture spacing in the interlayer is assumed equal to the fracture spacing in a fluid source layer. The interlayer thickness is a random variable derived from the mean length of the shortest paths from a fracture in a source layer to a fracture in a sink layer, along the bedding plane. Two fracture spacing assumptions are investigated, one where fractures have constant spacing, the other where fractures are negative exponentially spaced. An analytic expression, the *interlayer* equation, is found to quantify bed-normal permeability.

Investigation of the interlayer equation indicates the fracture spacing in a layered rock mass influences bed-normal permeability more than previous derivations using tensor methods. The bed-normal permeability is found to be highly sensitive to the magnitude of the bedding plane aperture. The bed-normal permeability is also found to be scale dependent, implying that a statistically homogeneous rock mass does not imply the existence of a representative elemental volume.

The analytic solution was verified numerically using the computer program *bnpflow*, written expressly for this purpose. For fractures with constant spacing, the analytical prediction and the numerical solution agree with 0.6%. For fractures with negative exponential spacing, the analytical prediction and numerical solution agree within 25%. The larger difference for negative exponentially spaced fractures is assumed to be a result of the assumptions used in the analytical derivation.

A potential application for the interlayer method is investigated, using rocks from the Cambrian-

aged Conasauga Group in Eastern Tennessee. Only an upper and lower bound for bed-normal permeability is found, because current techniques for collecting fracture data overlook important parameters necessary to accurately employ the interlayer equation.

Contents

1	Introduction	1
2	Analytical model of bed-normal permeability	7
2.1	Conceptual model	7
2.2	Interlayer analyses	10
2.2.1	Bed-normal permeability	11
2.2.2	Lattice model	12
2.2.3	Poisson model	14
2.2.4	The bed-normal permeability	17
2.3	Fractures crossing multiple beds	18
3	The Interlayer Effect	22
3.1	Comparison with disaggregate methods	23
3.2	Effect of fracture spacing	26
3.2.1	Measurement of fracture spacing	27
3.2.2	Effect of fracture spacing on permeability	29
3.3	Effect of bedding plane aperture	34
3.4	Interlayer scale effect	38

4 Numerical verification	40
4.1 Introduction	40
4.2 Mathematical development	41
4.2.1 Approximating a solution	42
4.2.2 Discretizing the weak statement	44
4.2.3 Assembly of elements	46
4.3 Implementation	47
4.4 Finite element modeling	49
4.4.1 Verification of the Lattice model	49
4.4.2 Verification of the Poisson model	50
4.4.3 Width of sampling window	51
4.4.4 No-flow boundary effects	53
4.5 Discussion	53
5 Illustrative example	56
5.1 Geologic setting	56
5.2 Estimating bed-normal permeability	59
6 Conclusions	62
List of References	65
Appendices	72
A Mean Path Length	73
B Construction of permeability tensor	78
C Finite element code	81

List of Figures

2.1	Two models of stratified porous media	9
2.2	The Interlayer concept	10
2.3	The Lattice model	13
2.4	The Poisson model	15
2.5	Termination probability for non bed-delimited fracturing	19
2.6	Termination probability for non bed-delimited fracturing	20
3.1	Correction factor ρ	25
3.2	A interlayer permeability approximation.	26
3.3	Schematic diagram of fractured layered rock	28
3.4	Schematic diagram of fractured layered rock	31
3.5	Interlayer comparison factor ρ, β varies.	34
3.6	Permeability vs. number of small aperture bedding planes.	35
3.7	Log-normal bedding plane apertures	37
3.8	Exponential bedding plane apertures	37
3.9	The interlayer effect on non bed-delimited fracturing	38
4.1	A linear basis constructed from a first order Lagrange interpolant.	46
4.2	Poisson mesh.	48

4.3	Increasing layers vs. permeability.	50
4.4	Increasing layers vs. permeability.	51
4.5	Effect of spacing bias on permeability	52
4.6	Bias from sampling window	53
4.7	Effect of no flow boundary	54
5.1	Location map of study area	57
5.2	Stratigraphy of the Conasauga Group	58
A.1	Number of flow paths	74
A.2	Mean path length model, even flow paths.	75
A.3	Mean path length model, odd paths.	76
A.4	Mean length of flow paths as a function of the S_j/S_i spacing ratio.	77
B.1	Permeability tensor example	79

List of Symbols and Acronyms

EPM	Equivalent porous medium.
REV	Representative elemental volume.
GWS	Galerkin weak statement.
α	Number of fractures in a source layer that occur between two fractures in a sink layer.
S_i	Expected value of fracture spacing in a given layer i .
T_i	Bed thickness of i th layer.
T	Fracture termination (2.3).
e_i^3	Expected value of cubed fracture aperture for layer i .
j	$i + 1$.
e_{ij}	Aperture of bedding plane bounded by beds i and j .
β	Proportionality constant relating bed thickness to mean fracture spacing.
c	Flow path length constant.
ΣB	Harmonic sum of permeabilities of rock layers.
ΣT	Arithmetic sum of bed thicknesses.

ΣI	Interlayer correction term.
$k_{\perp}$	Intrinsic, bed-normal fracture permeability, computed from the interlayer equation.
k_{DA}	Intrinsic, bed-normal fracture permeability, computed using a dis-aggregate method.
$E(X)$	Expected value of X .
T_{ij}	"Thickness" of the interlayer.
S_{ij}	Expected value of "fracture spacing" in interlayer.
$P(X)$	Probability of event X .
f_n	Fraction of joints in a rock mass crossing n beds; independent of bed thickness.
ρ	Comparison factor relating interlayer permeability to permeability determined by Snow's method ($\rho = k_{\perp}/k_{DA}$).
Ψ_i	Trial function on some element i .
w_i	Weight function for some element i .
Φ	Head, a potential quantity.
$\{b\}$	Column vector b ; b represents known, constant head boundary conditions.
Ω	Problem domain for discretizing a numerical solution.
$\mathbf{n}$	Vector normal to domain boundary.
$\partial\Omega$	Domain boundary.
$d\sigma$	Differential over domain boundary.
$d\tau$	Differential over spatial dimension of the problem.
$\{N_1(\zeta_j)\}$	Linear basis function in natural coordinates.

W, w	Width of domain for modeling fractured layered rock.
$\mathbf{k}$	Permeability tensor.
$\mathbf{K}$	Conductivity tensor.
$q, \mathbf{q}$	Flow, flow vector.
L	Flow path length, a random variable.
ℓ	A specific value of the random variable L (chapter 2), or element length (chapter 4).

Chapter 1

Introduction

Characterizing the permeability of intensely fractured sedimentary rocks, such as thin-bedded limestones, siltstones and shales in tectonically deformed regions, is important for groundwater resources planning and groundwater contaminant remediation. Since the permeability of the rock matrix is usually orders of magnitude lower than the permeability of fractures, quantifying the fracture geometry and its effect on fluid flow is a necessary first step. According to *Dershowitz and Einstein* (1988), two main approaches have been followed to characterize the fracture geometry of rock masses: the “disaggregate” approach, and the “aggregate,” or joint system model approach. In the disaggregate approach, fracture parameters such as spacing, orientation, fracture aperture, etc., are quantified independently, providing statistical information on individual joints or joint sets. In a *joint system model*, the same joint parameter distributions are described but, in addition, the interdependence of joint parameters upon one another is specified. A joint system model includes rules for describing characteristics of the joint network geometry such as abutting relationships and termination styles among joints of different sets. In other words, the joint system model includes information about the joint network formed by connected fractures as well as information about individual joints and joint sets.

Joint system models are differentiated by their distributions of fracture parameters and hierarchy of rules correlating fracture network properties. Many joint system models have been developed. *Snow* (1969) simplified fracture systems to countable sets of infinite planes to model permeability in granitic rocks. *Baecher* (1977) and *Barton* (1978) modeled fractures in crystalline rocks as being randomly located, with random sizes and orientations. *Veneziano* (*Dershowitz* 1984) modeled coplanar but impersistent fractures by removing random polygons bounded by intersection lines of randomly oriented, randomly located infinite planes. *Dershowitz* (1984) extended the *Veneziano* model by restricting the polygon removals to correspond to polygons that would create polyhedral rock blocks.

Other joint system models have been developed to model site specific fracturing. *Barton and Larsen* (1985) modeled fractures in a welded tuff as a self-similar or fractal network, noting that the fracture patterns appeared similar independent of the observation scale. *Billaux et al.* (1989) modeled jointing at the Fanay-Augères uranium mine in France by a "parent-daughter" technique. Certain fractures (parents) were noted to be centers of prominent clusters, other fractures (daughters) were distributed conditioned on the location of the parent fracture. *Gervais, Chiles and Gentier* (1992) built a joint system model of fracture traces exposed on a limestone bedding plane from a master joint set by successively conditioning each additional fracture set upon preceding sets.

The bulk permeability of joint system models is usually quantified by either discrete methods, where every fracture is explicitly modeled, or by continuum methods, where the fractured rock mass is idealized as an equivalent porous medium (EPM). Bulk permeability of an EPM may be determined by analysis (e.g., *Snow* 1969; *Oda* 1982; *Oda* 1985), numerical methods (e.g., *Wilson* 1970; *Long et al.* 1982), by empirical methods, using electric resistor networks (*Marcus and Evenson* 1961; *Bear* 1972; *Hudson and La Pointe* 1980) or field tests (*Hsieh et al.* 1985; *McKay et al.* 1993). Analytical models generally assume that the permeability of the rock mass is controlled

by the average properties of the fractures. If the fractured rock mass is assumed homogeneous in the domain of interest, many researchers assume the rock mass to behave as an equivalent porous medium. Equivalent porous medium, or continuum models, rely on the construction of a representative elemental volume (REV). The fracture geometry of the REV is assumed statistically similar to the fracture geometry of the rock mass. The fracture parameters are then used to construct a permeability tensor to quantify the permeability of the REV (*Snow* 1969; *Oda* 1982). Numerical models for determining bulk permeability construct synthetic sets of fracture traces, then compute flow through each fracture across the domain, and finally back calculate an equivalent bulk permeability from the flow volume and head difference using Darcy's law. Empirical models using networks of electrical resistors proceed similarly to the numerical method: a potential difference across an array of resistors is used to compute current through each resistor, which is used to back-calculate the electrical conductivity from the current and potential difference. Field testing involves methods such as pump testing (e.g., *Hsieh and Neuman* 1985; *Jancin and Ewart* 1995) or injection tests (e.g., *Dreier et al.* 1987; *McKay et al.* 1993).

Two EPM models of fractured rock masses in which the effect of the joint system model on bulk permeability has been in studied detail are: *Snow's* (1969) model of infinitely extensive fractures grouped into countable sets, and *Baecher and Lanney's* model of randomly located, randomly oriented, circular fractures with exponential spacing and log-normally distributed radii (*Baecher* 1977; *Baecher and Lanney* 1978). The permeability of *Snow's* (1969) model was numerically investigated by *Wilson* (1970), using finite element methods, who concluded that *Snow's* analysis of a rock mass containing infinitely extensive fractures accurately predicted permeability when fracture apertures were constant. *Long et al.* (1982) numerically investigated the equivalent permeability of a 2-D *Baecher* fracture network using finite elements, concluding that finite length fracture systems may not produce an equivalent porous medium useful for computing bulk permeability. As pointed out by both *Long et al.* (1985) and *Wilson* (1970), the ability of a rock mass to conduct fluid depends

in part on the nature of the fracture connections. For instance, the flow through two wide aperture fractures connected by a narrow aperture fracture is controlled by the narrow aperture of the connecting fracture. In an EPM of a joint system, if the permeability tensor does not model joint system connectivity, an inaccurate estimate of the bulk permeability will result.

The field study (*Doolin and Mauldon 1995*) that motivated this thesis was concerned with attempting to quantify the degree of anisotropy and heterogeneity of the permeability of Cambrian-aged Conasauga Group rocks similar to rocks hosting groundwater contaminants on the Oak Ridge Reservation, Tennessee. Fracture intensity in these rocks, previously described by *Dreier et al. (1987)* and *Lemiszki (1992)*, can approach 50 fractures per meter. For predicting groundwater flow and contaminant transport, such intense fracturing precludes the use of discrete fracture models, (e.g., *Dershowitz and LaPointe 1994*), suggesting a continuum approach where the rock mass is idealized as an EPM and fracture flow is characterized by a permeability tensor. Determining the equivalent continuum permeability of a fractured rock mass requires constructing a joint system model from field observations, determining the scale of the REV, then calculating a permeability tensor from the REV.

Inspection of Conasauga Group rocks exposed in a road cut near the Oak Ridge Reservation reveals that the joint system in stratified rocks often consists of bedding planes, which can be assumed to be of infinite extent, and fractures that terminate on bedding planes, commonly occurring in regular orthogonal patterns. Bed-delimited fractures render a joint system model scale dependent, ruling out infinite plane and fractal geometry models. Previous studies of the permeability of fractured stratified rocks have focused on permeability parallel to bedding, modeling the fracture trace network exposed on bedding planes (e.g., *LaPointe 1985*; *Gervais et al. 1992*; *Odling 1992*). This thesis describes a new joint system model, appropriate for modeling fracture permeability normal to bedding. The new model, termed the *interlayer model*, combines features of the models of Snow and Baecher.

The interlayer model consists of two orthogonal sets of fractures. The first is a set of infinitely extensive, parallel fractures corresponding to bedding planes in layered rock. The second set is confined to a single layer of rock, oriented perpendicular to the first set, and contains fractures located according to a stochastic point process. Since fractures are assumed bed-delimited, the size of fractures in the second set is conditioned on the bed thickness, which is modeled as a random variable. The occurrence of fractures in each bed is assumed independent of fractures in other beds. Therefore, with probability one, fractures in adjacent beds do not terminate at the same point on the same bedding plane. Since flow along the bedding plane is difficult to model directly, a fictitious or virtual bed, called an "interlayer," rather than virtual fractures (*Dershowitz and Einstein 1988*), is introduced to approximate the head loss incurred by laminar fluid flow along a bedding plane, from fractures in a source layer to fractures in a sink layer.

The interlayer model makes the following assumptions: (1) the rock matrix is impermeable, (2) flow is conducted solely through fractures, (3) fracturing in the rock mass is assumed to be statistically homogeneous, (4) while bedding planes are infinitely extensive fractures, all other fractures are assumed to ultimately terminate on bedding planes. The first three assumptions are shared by most analytical models for rock mass permeability. The fourth assumption allows the quantitative effect of the fracture connectivity on bed-normal permeability to be analyzed. The fourth assumption is easily supported by field observation: fractures are often confined to a single layer of rock, or terminate within the rock of an adjacent bed, thereby truncating the flow path at the previous bedding plane. Confining fractures to beds allows the derivation of novel analytic solutions for the bed-normal permeability of a stratified, fractured rock mass. Unlike other analytical models, statistical homogeneity is not assumed to produce a scale independent representative elemental volume (REV) for modeling continuum flow.

The present investigation is limited to analytical and numerical modeling of bulk permeability in a direction perpendicular to layering. The *interlayer equation*, an analytic derivation of bulk per-

meability in a direction perpendicular to bedding of fractured, stratified rock masses, is presented for two different fracture spacing assumptions: constant and negative exponential. The behavior of the interlayer equation is explored by varying one parameter while holding other parameters constant. The magnitudes and distribution of bedding plane apertures are found to greatly influence the bed-normal permeability. The bed-normal permeability is found to be scale dependent, invalidating the concept of a statistically identical REV for fractured, layered rock. A finite element model was constructed to verify the analytic solution. The interlayer equation was found to agree exactly with numerical results when fractures were assumed to have constant spacing. When fractures were assumed to have negative exponential spacing, the interlayer equation over-predicted the numerical result by 25%, on average. Lastly, a practical example for the interlayer equation is presented, using a thin-bedded, Cambrian-aged limestone from East Tennessee.

In all following material, *italicized text* other than references will (nearly) always indicate that something has just been defined, or is being defined, or will be defined immediately. The terms permeability, rock mass permeability or bulk permeability refer to the intrinsic permeability of the rock mass, in units of length squared, resulting from a network of connected fractures. The matrix, or intact rock permeability is assumed to be zero. The intrinsic permeability of an individual fracture will be noted by context. The words "fracture" and "joint" are synonymous here and refer to finite length discontinuities not parallel with stratification. The words "bed" and "layer" are synonymous, as are "bedding" and "layering." The term bedding plane is used to refer to infinitely extensive discontinuities separating discrete rock strata. These terms are purely mechanical; no genetic description is implied.

Chapter 2

Analytical model of bed-normal permeability

2.1 Conceptual model

The permeability k of a rock mass with a single set of parallel, infinitely extensive fractures, in the direction parallel to fractures, is directly proportional to the expected value $E(e^3)$ of the cubed fracture aperture e , and inversely proportional to the expected value $E(S)$ of the spacing S between parallel fractures,

$$k = \frac{1}{12} \frac{E(e^3)}{E(S)}, \quad (2.1)$$

a result heuristically derived by *Snow* (1969), and familiar as “Snow’s Law.” The notion is mathematically correct only under the assumption that fracture spacing and aperture are statistically independent, implicit in *Snow’s* (1969) derivation, and when fracture apertures are modeled as parallel plates. For convenience in the following derivations, we define $e^3/S \equiv E(e^3)/E(S)$.

Snow’s heuristic result is quickly and rigorously proven by application of the law of large numbers.

Consider an impermeable rock mass divided by parallel fractures. Under the assumption of plane Poiseuille flow, that is, stationary and parallel fracture walls and laminar, slow velocity flow with no inertial forces, the permeability of a single fracture with linearly varying head drop is proportional to the cubed aperture e^3 per unit thickness (*Malvern 1969*). For some width W of rock containing m fractures, the permeability is proportional to $\sum_{k=1}^m e_k^3$, where e_k is the aperture of the k th fracture in the layer. The permeability per unit width is then $(e_1^3 + e_2^3 + \dots + e_m^3)/W$. But W is the sum of the spaces S separating each fracture: $W = S_1 + S_2 + \dots + S_m$. By the strong law of large numbers, $P(|W/m - \mu_S| > \epsilon) \rightarrow 0$, where μ_S is the expected value of the fracture spacing and ϵ is an arbitrarily small number greater than zero. It follows that

$$\frac{e_1^3 + e_2^3 + \dots + e_m^3}{S_1 + S_2 + \dots + S_m} \rightarrow \frac{E(e^3)}{E(S)}, \quad (2.2)$$

m increasing without bound. In a fractured stratified rock mass (Fig. 2.1b) where the cross-section is subdivided by bedding planes and orthogonal fractures, Snow's Law also may be assumed to hold for the bed-normal permeability of any particular rock bed i that has constant (bed) thickness over the domain, that is, $k_i = e_i^3/(12S_i)$.

In a stratified porous medium of n layers, on the other hand, (Fig. 2.1a), the equivalent bed-normal permeability $k_\perp$ of unlithified sediments is the weighted harmonic mean of n layer permeabilities k_i (*Marcus and Evenson 1961; Freeze and Cherry 1979*):

$$k_\perp = \frac{\Sigma T}{\sum_{i=1}^n \frac{T_i}{k_i}}, \quad (2.3)$$

where $\Sigma T = \sum_{i=1}^n T_i$ is the total thickness, and T_i, k_i are the thickness and permeability, respectively, of the i th layer. The model described in Eq. 2.3 assumes mass continuity of flow, zero head loss at the interface between layers, and that the permeability of each layer is homogeneous. We

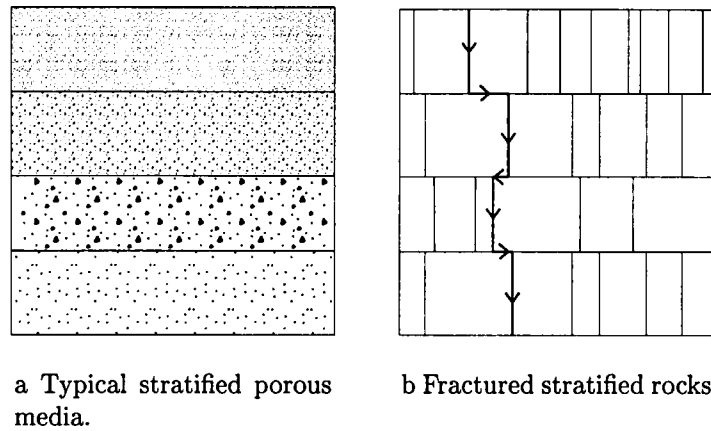


Figure 2.1: Two models of stratified porous media: (a) the layers conduct through a porous granular medium; (b) layers only conduct through discrete fractures.

now suppose that the individual layers in the porous medium model (Eq. 2.3) are layers of fractured rock rather than layers of porous media.

In a layered rock mass, Eq. 2.1 is valid for the bulk permeability in a direction parallel to bedding, noting that bed thickness is the spacing between conductive bedding planes. When fractures are confined between and terminate on bedding planes, Eq. 2.1 is not valid for bulk fracture permeability normal to bedding. Unlike flow through layered granular porous media (Fig. 2.1a), layer-perpendicular flow through a fractured layered rock mass (Fig. 2.1b) necessarily involves flow along bedding planes from fractures in one bed to fractures in the next bed. The bed-normal permeability of a fractured, layered rock mass cannot be determined by simple substitution of Eq. 2.1 into Eq. 2.3 because Eq. 2.1 neglects head losses incurred by flow along the bedding plane between fractures in adjacent beds. Consequently, a correction term is necessary to quantify the extra head loss. In this thesis, the interconnecting flow along bedding planes is modeled as flow through a virtual rock layer, or *interlayer*.

Figure 2.2 illustrates the interlayer model: a sequence of layers (i.e., beds) and interlayers (i.e., bedding planes between beds). The interlayer may be considered analogous to a bed of fractured

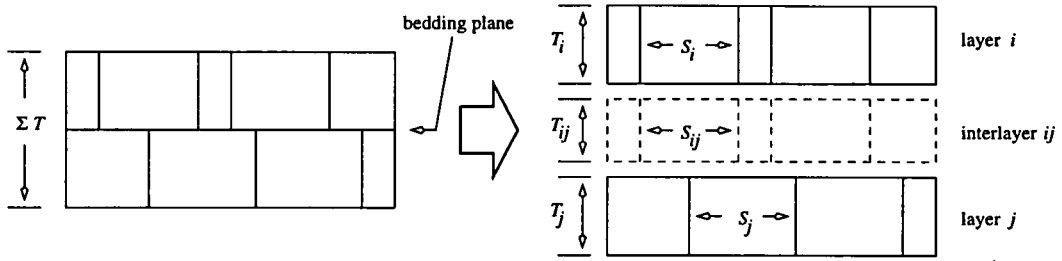


Figure 2.2: Model of layered fractured rock illustrating the interlayer concept.

rock, with a “thickness” equal to the mean path length $E(L)$ of flow along the bedding plane, i.e., the mean flow path connecting fractures in one bed to fractures in the next bed, a “fracture aperture” equal to the bedding plane aperture, and a “fracture spacing” derived from the fracture spacing of the adjacent rock layers. Using the interlayer model, a fractured layered rock mass is rendered mathematically equivalent to the granular porous media model by assuming there is no head loss between layers and interlayers.

2.2 Interlayer analyses

The following assumptions were made to derive an analytical approximation for bed normal permeability. Steady-state, saturated, creeping, laminar flow along a bedding plane from a source fracture in one layer to a sink fracture in an adjacent layer is modeled as a collection of streamlines comprising a *flow path*. In laminar, creeping flow, the velocity is assumed to be slow enough that the momentum of the flow is essentially zero. Thus, no head loss is assumed to occur at fracture intersections, an assumption supported by laboratory tests (Wilson 1970). Fractures are assumed to extend completely through one bed and terminate on the bedding planes bounding that bed. Fractures confined to one layer are assumed to be located independently of fractures in other layers. Assuming no source or sink, mass continuity requires equal flow through every layer and every layer interface. The permeability of an interlayer is assumed proportional solely to the mean length of the flow path in the interlayer; not to flow rate or head. To simplify the derivation, the intrinsic

fracture permeability is assumed equal to $\frac{1}{12}e^3$ (units of cubed length per unit length), which may be replaced by a fracture permeability determined from experimental data (e.g., *Witherspoon et al.* 1980; *Gale* 1990; *Zimmerman et al.* 1992; *Piggott and Elsworth* 1993) or numerical modeling data (e.g., *Pyrak-Nolte et al.* 1990) with no loss of generality.

2.2.1 Bed-normal permeability

To determine equivalent bed-normal permeability of the beds in a fractured, layered rock mass, substitute Eq. 2.1 for k_i in Eq. 2.3, defining ΣB as the sum of n bed thicknesses divided by the bed permeabilities:

$$\Sigma B = 12 \sum_{i=1}^n \frac{T_i S_i}{e_i^3}, \quad (2.4)$$

where the bed thickness T_i , the mean fracture spacing S_i and the mean aperture e_i of the i th layer are estimated from field data. Let T_{ij} , S_{ij} and e_{ij} , denote, respectively, the equivalent thickness, equivalent fracture spacing and equivalent fracture aperture of the interlayer between layer i and layer j (see Fig. 2.2). By analogy with the layer permeability, an interlayer correction term ΣI may now be defined as the sum of interlayer thicknesses divided by interlayer permeabilities

$$\Sigma I = 12 \sum_{i=1}^{n-1} \frac{T_{ij} S_{ij}}{e_{ij}^3}, \quad (2.5)$$

where $j = i + 1$. The aperture e_{ij} of the bedding plane bounded by layers i and j represents the fracture aperture of the interlayer. To model the appropriate thickness T_{ij} and fracture spacing S_{ij} of the interlayers, note that the layer thickness T_i is also the length of the flow path through a fracture contained in some i th layer of rock. By analogy with an actual layer of rock, the “thickness” T_{ij} of the interlayer is assumed to be the expected value $E(L_{ij})$ of the flow path length along the bedding plane, that is,

$$T_{ij} \equiv E(L_{ij}). \quad (2.6)$$

By substitution,

$$\Sigma I = 12 \sum_{i=1}^{n-1} \frac{E(L_{ij})S_{ij}}{e_{ij}^3}. \quad (2.7)$$

A complete expression for bed-normal permeability may now be obtained. From Eq. 2.3, let

$$\sum_{i=1}^n \frac{T_i}{k_i} \equiv \Sigma B + \Sigma I. \quad (2.8)$$

Letting ΣT denote the sum of the individual bed thicknesses, and substituting Eq. 2.8 into Eq. 2.3 yields the *interlayer equation* for bed-normal permeability in fractured stratified rocks:

$$k_{\perp} = \frac{\Sigma T}{\Sigma B + \Sigma I}. \quad (2.9)$$

In layered, granular porous media, the bed-permeability is the weighted harmonic mean of the layer permeabilities (Eq. 2.3). In layered, fractured porous media, on the other hand, the bed-permeability is the weighted harmonic mean of the layer and interlayer permeabilities (Eq. 2.9).

In the following analyses, two different fracture spacing models are investigated for fractures confined to a single bed. The first model assumes fractures of constant spacing (Lattice model) and the second assumes negative exponentially spaced fractures (Poisson model). Since ΣT and ΣB are quantified in terms of rock mass data such as bed thickness and mean fracture spacing, collected during field study, it is convenient to derive ΣI in terms of field data also. In both cases, ΣI is shown to be a function of the product of fracture spacing in adjacent beds divided by the aperture of the bedding planes.

2.2.2 Lattice model

Fracture spacing in the Lattice model is assumed to be constant (Fig. 2.3) in each layer, consistent with a first approximation to field observations and with linear elastic fracture mechanics (*Gross*

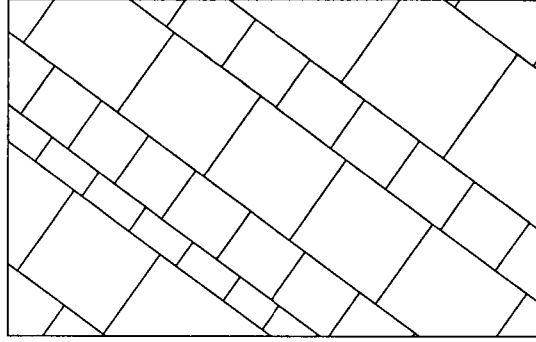


Figure 2.3: The Lattice model: bed confined fractures with constant spacing.

et al. 1995). Although the fracture spacing is assumed constant in each of the n layers, it may vary from layer to layer. By assumption, fractures in each layer are located independently of fractures in the next layer. Since, by Darcy's Law, the flow is inversely proportional to the length of the flow path ($q \propto 1/dx$), an assumption that the flow takes the shortest path along the bedding plane from one fracture to the next fracture is a reasonable approximation. Given independence of fracture location between layers, the length L_{ij} of the shortest path along the bedding plane from a source fracture i to a sink fracture j is uniformly distributed over the interval $(0, S_j/2)$, where S_j is the fracture spacing of layer j . The probability density function of L is

$$f_L(\ell) = \begin{cases} 2/S_j, & 0 < \ell < S_j/2; \\ 0, & \text{otherwise.} \end{cases} \quad (2.10)$$

Since, by assumption, fractures are confined to single beds, and fracture locations in successive beds are independent, $P(L = 0) = 0$.

Choosing any fracture in a source layer i at random, where all choices are equally likely, the expected value of the flow path length to the nearest fracture in layer j is

$$E(L) = \int_0^{S_j/2} \ell f_L(\ell) d\ell = \frac{S_j}{4}. \quad (2.11)$$

In an infinitely long bed, the expected value of any sample of randomly chosen fractures will also be $S_j/4$.

Using the expected value of the flow path length (Eq. 2.11), the interlayer thickness in the Lattice model is $T_{ij} = E(L) = S_j/4$ (Doolin and Mauldon 1996). By the shortest path length assumption, the number of flow paths in the interlayer equals the number of fractures in the source layer, which implies the fracture spacing in the interlayer equals the fracture spacing in the source layer: $S_{ij} = S_i$. The interlayer correction term ΣI for fractures with constant spacing is

$$\Sigma I = 12 \sum_{i=1}^{n-1} \frac{S_i S_j}{4e_{ij}^3}. \quad (2.12)$$

Equation 2.12 can also be derived under the assumption that flow is distributed uniformly from source fractures, instead of assuming the flow is directed solely towards the nearest fracture. Detailed analysis under this latter assumption is found in Appendix A.

2.2.3 Poisson model

If the occurrence of fractures in each layer is assumed to follow a Poisson process (Fig. 2.4), the length L between fractures in each layer is negative exponentially distributed: $F_L(\ell) = 1 - e^{-\lambda\ell}$, where $\lambda = 1/S$ is the reciprocal of the mean fracture spacing. Since fractures are assumed to be confined to single layers, a source fracture in layer i will terminate on the bedding plane in an interval between two sink fractures in layer j . Let L_1 be the connection length along the bedding plane from a source fracture to the next occurrence of a sink fracture (in a given direction), nL_2 the length to the previous occurrence of a sink fracture. Since the exponential distribution is independent of an origin, the cumulative distribution functions of L_1 and L_2 are

$$F_{L_1}(\ell) = P(L_1 \leq \ell) = 1 - e^{-\lambda_1\ell}, \quad \text{and} \quad F_{L_2}(\ell) = P(L_2 \leq \ell) = 1 - e^{-\lambda_2\ell}. \quad (2.13)$$

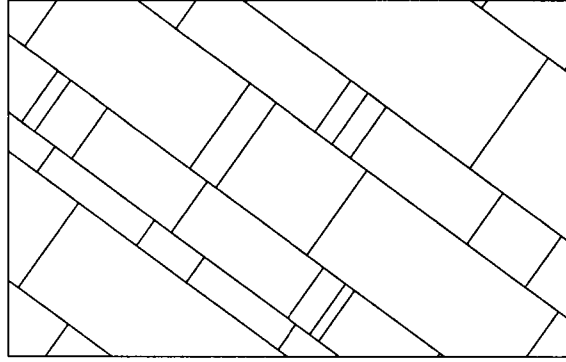


Figure 2.4: The Poisson model: bed confined fractures with negative exponential spacing.

Note that L_1 and L_2 have the same cumulative distribution function as the fracture spacings; a consequence of the lack of memory of the Poisson process. In other words, since the occurrence of any point in a one dimensional Poisson process is independent of the occurrence of the previous or next point, the distribution of lengths to the next or previous points are identical, i.e. $\lambda_1 = \lambda_2 = \lambda$. To determine the distribution of distances to the nearest fractures, define the events A_1 and A_2 as

$$A_1 = L_1 \leq \ell, \quad A_2 = L_2 \leq \ell$$

and $F_L(\ell)$ as the distribution of path lengths to the nearest fracture, either next or previous, such that

$$\begin{aligned} F_L(\ell) &= P(A_1 \cup A_2) \\ &= P(A_1) + P(A_2) - P(A_1 A_2). \end{aligned} \tag{2.14}$$

In general, $P(A_1 A_2) = P(A_1 | A_2) P(A_2)$. In a Poisson process, however, A_1 and A_2 are independent events, and $P(A_1 | A_2) = P(A_1)$, implying that $P(A_1 A_2) = P(A_1) P(A_2)$. Since $P(A_1) = P(A_2) =$

$1 - e^{-\lambda\ell}$, by substitution in Eq. 2.14 and some algebra,

$$P(A_1 \cup A_2) = 2(1 - e^{-\lambda\ell}) - (1 - e^{-\lambda\ell})^2, \quad (2.15)$$

which simplifies to the probability distribution

$$F_L(\ell) = 1 - e^{-2\lambda\ell}. \quad (2.16)$$

Observe that $F_L(0) = 0$ and $F_L(\infty) = 1$ as required. The probability density function of L is

$$f_L(\ell) = \frac{d}{d\ell} F_L(\ell) = 2\lambda e^{-2\lambda\ell}; \quad (2.17)$$

the expected value $E(\ell)$ of the flow path length is

$$E(L) = \int_0^\infty \ell f_L(\ell) d\ell = \frac{1}{2\lambda}. \quad (2.18)$$

Since $\lambda = 1/S$ for any arbitrary layer, $T_{ij} = E(L) = S_j/2$ for each layer j . Assuming the number of flow paths in the interlayer is equal to the number of fractures in the source layer, the “fracture spacing” in the interlayer equals the fracture spacing in the source layer: $S_{ij} = S_i$. By substitution in Eq. 2.7, the interlayer correction term for fractures following a Poisson process is

$$\Sigma I = 12 \sum_{i=1}^{n-1} \frac{S_i S_j}{2e_{ij}^3}. \quad (2.19)$$

What is implied here is that the average length of intervals that contain flow paths is twice the length of the mean spacing in the layer. The result is well known in renewal theory as length biasing, Feller’s Paradox, the waiting time paradox, or the renewal paradox. Details may be found in *Cox* (1962), *Feller* (1966) or *Brémaud* (1988).

An illustration of the waiting time paradox may be observed at a bus stop served by randomly arriving buses. Suppose 12 buses arrive at a bus stop at random times during a 12 hour period between 6 a.m. and 6 p.m. By definition, random means that a bus is equally likely to arrive in any arbitrarily small period of time during that 12 hours. The average rate of bus arrivals is one per hour. Now, the reader arrives at the bus stop at, say, 12 noon. What is the expected waiting time until the next bus? One hour. Surprisingly, the expected elapsed time since the previous bus is also one hour. This is the waiting time paradox. The length of the time interval between the last bus preceding 12 noon, and the first bus following 12 noon has an expected value of two hours. The paradox is resolved by recognizing that any particular time (such as the readers arrival time) is more likely to be contained in a large interval than in a small interval.

2.2.4 The bed-normal permeability

To express the interlayer equation in terms of fracture parameters, let a constant c take the value of 2 for the Poisson model (Eq. 2.19) and the value of 4 for the Lattice model (Eq. 2.12). Expand the sum of the layer thicknesses $\Sigma T = \sum_{i=1}^n T_i$, and substitute Eq. 2.4 and Eq. 2.12 or 2.19 into Eq. 2.9, yielding

$$k_{\perp} = \frac{1}{12} \left(\frac{\sum_{i=1}^n T_i}{\sum_{i=1}^n \frac{T_i S_i}{e_i^3} + \sum_{i=1}^{n-1} \frac{S_i S_j}{c e_{ij}^3}} \right), \quad c = 2, 4. \quad (2.20)$$

This expanded form (Eq. 2.20) of the interlayer equation is easily implemented by computer programming or in a spreadsheet. Note the symmetry of the product term $S_i S_j$; the interlayer equation can be used if the beds are counted from top-to-bottom (layer 1 to n) or bottom-to-top (layer n to 1). Also, as the spacing $S_j \rightarrow 0$ for any bed j , the interlayer effect becomes negligible, approaching a porous medium equivalent for bed j . Furthermore, $k \rightarrow 0$ as $e_i \rightarrow 0$ or as $e_{ij} \rightarrow 0$, for any layer $i, j = 1, 2, \dots, n$.

2.3 Fractures crossing multiple beds

The interlayer effect can also be shown to reduce the permeability in joint systems where fractures cross single or multiple beds. A similar reduction in k was noted by Wilson (1970), a result due to varying fracture aperture in segments of a numerical fracture flow study of Snow's joint system model: "... the effect of a rare single large fracture segment upon total flow is very small because it cannot penetrate the system." Wilson also notes that aperture distributions which fail to account for fracture location may be inadequate. The Poisson process, extended from joints confined to a single bed, to joints crossing 2, 3, or more beds (but not the whole domain), provides a convenient starting point to examine the interlayer effect in a rock mass where fractures cross more than a single bed, but do not cross every bed.

Assume that random length joints with constant aperture, perpendicular to and delimited by beds of constant thickness, are distributed throughout a fractured rock mass. Fracture locations in any given bed are statistically independent of fracture length or bed thickness. Fractures crossing any given bed are modeled as a one-dimensional Poisson process. Assume the total joint frequency λ is constant and independent of bed thickness, and represents a scan line measurement parallel to bedding and long enough to be unbiased by the domain boundary. Choosing a layer i at random, suppose that the probability $P(T)$ represents the fraction of all joints in the i th layer terminating on the lower bedding plane ij bounding that layer. The fraction $1 - P(T)$ of all joints in layer i extend through the next layer j (Fig. 2.5). The path length L along the bedding plane for the through-going joints is zero. The path length for the terminated joints is $S_j/2$, because distances between uniformly distributed events are negative exponentially distributed. The result was previously determined for the Poisson process (§2.2.3). From the definition of expectation, the expected value of the total path length is

$$E(L) = P(T) \cdot \frac{S_j}{2} + (1 - P(T)) \cdot 0$$

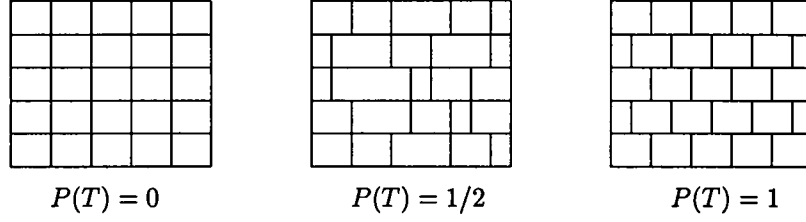


Figure 2.5: Three different examples for termination probability.

$$= P(T) \frac{S_j}{2}. \quad (2.21)$$

The interlayer correction term becomes

$$\Sigma I = \sum_{i=1}^{n-1} \frac{S_i S_j}{2e_{ij}^3} P(T). \quad (2.22)$$

As $P(T) \rightarrow 0$, the interlayer correction term approaches zero, and the interlayer model approaches Snow's model of infinitely extensive joints. As $P(T) \rightarrow 1$, all joints tend to be confined to a single layer, increasing the importance of the interlayer correction term ΣI .

The total frequency λ_{total} of joints in a bed can be expressed as the sum of the frequencies of joints crossing exactly one bed (λ_1), exactly two beds (λ_2), up to exactly n beds (λ_n):

$$\lambda_{\text{total}} = \lambda_1 + \lambda_2 + \cdots + \lambda_n, \quad (2.23)$$

where "exactly n " means " n and no more than n ." Dividing both sides through by λ_{total} yields

$$\frac{\lambda_{\text{total}}}{\lambda_{\text{total}}} = \frac{\lambda_1}{\lambda_{\text{total}}} + \frac{\lambda_2}{\lambda_{\text{total}}} + \cdots + \frac{\lambda_n}{\lambda_{\text{total}}} = 1, \quad (2.24)$$

Randomly selecting a fracture from a given bed, with all fractures equally likely, let the fraction

$f_i = \lambda_i / \lambda_{\text{total}}, i = 1, 2, \dots, n$ denote the probability that the fracture crosses exactly n beds, and substitute into Eq. 2.24:

$$f_1 + f_2 + \dots + f_n = 1. \quad (2.25)$$

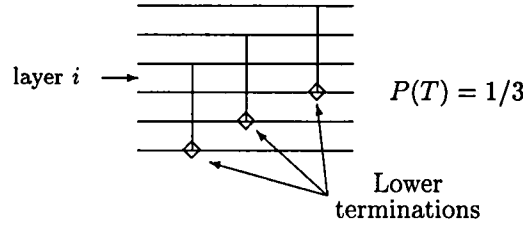


Figure 2.6: For beds of unit thickness, for any fracture of length 3 crossing bed i , the probability of termination $P(T) = 1/3$.

Consider flow in the downward direction. We define termination of a fracture on a layer i to be the *lower termination* (Fig. 2.6). Since the flow path length is zero for through-going fractures, the next step is to determine the probability that the intersected fracture terminates on the next lower bedding plane. Let F_n denote the event that a fracture crosses exactly n beds, and let T denote the event that a fracture terminates on the bottom of a given bed. For the fraction of joints that are confined to one layer, the probability of termination is 1. For fractures that cross exactly two beds (fraction f_2 of all joints), the probability of termination equals the probability that a fracture crossing two beds terminates on the next lower bedding plane, or $P(T|F_2) = 1/2$ (Fig. 2.5). For any particular fracture crossing exactly n beds, the probability of termination on the next lower bedding plane, given n , is $P(T|F_n) = 1/n$. Observing that $F_1, F_2, \dots, F_n$ form a mutually exclusive, collectively exhaustive set, the probability $P(T)$ of a (lower) termination is found from the total probability theorem

$$P(T) = P(T|F_1)P(F_1) + P(T|F_2)P(F_2) + \dots + P(T|F_n)P(F_n)$$

$$= \frac{f_1}{1} + \frac{f_2}{2} + \dots + \frac{f_n}{n}. \quad (2.26)$$

Recall that the mean flow path length along the interlayer is zero if the fracture does not terminate on the layer, and $S/2$ if the fracture does terminate on the layer. Substituting Eq. 2.26 into Eq. 2.21, the interlayer correction term becomes

$$\Sigma I = \sum_{i=1}^{n-1} \frac{S_i S_j}{2e_{ij}^3} \left(\frac{f_1}{1} + \frac{f_2}{2} + \dots + \frac{f_n}{n} \right). \quad (2.27)$$

where e_{ij} , S_i , S_j are bedding plane aperture and spacing for layers i and j , as previously defined.

Chapter 3

The Interlayer Effect

Equivalent porous medium models of fractured rock masses first quantify the permeability of a statistically similar sub-domain, or representative elemental volume (REV), of the rock mass. Existing models (*Snow* 1969; *Oda* 1985) use “disaggregate” analysis to determine a permeability tensor $\mathbf{k}$. The effect of fracture parameters on $\mathbf{k}$ are assumed independent of fracture connectivity. Neither the fracture connectivity nor the effect of fracture connectivity on permeability is explicitly modeled. In contrast, the interlayer model is a joint system model that quantifies not only the parameters of individual fractures or fracture sets, but also the interactions of parameters such as fracture spacing and aperture. These fracture parameter interactions, quantified in the interlayer correction term ΣI of Eq. 2.9, give rise to the *interlayer effect*. The interlayer effect is a reduction in permeability caused by interaction of bed-confined fractures on flow across a conductive (permeable) layer interface such as a bedding plane. The interlayer model also illustrates a scale effect on the bed-normal permeability as the number of layers n increases in a fractured rock mass.

Four aspects of the interlayer effect on bed-normal permeability are investigated:

1. Comparison with analytic expressions for continuum permeability such as the bed-normal component of permeability tensors derived by *Snow* (1969) and *Oda* (1982);

2. The permeability reduction from increasing fracture spacing relative to other parameters.
3. The permeability reduction from decreasing bedding plane apertures relative to other parameters.
4. Scale effects that control the magnitude of bed-normal permeability.

The results show that bed-normal permeability in fractured, stratified rocks is scale dependent, influenced by both fracture spacing, fracture aperture and distribution of bedding plane apertures.

3.1 Comparison with disaggregate methods

To examine the interlayer effect on bed-normal permeability in a fractured stratified rock mass, consider a rock mass containing constant spaced ($S_i = S$) fractures of constant aperture ($e_i = e_f$) confined by and orthogonal to beds of constant thickness ($T_i = T$). Let the bedding plane apertures be constant ($e_{ij} = e_b$). Assuming that e_f, e_b, T and S are constants allows the summation terms in the interlayer equation (Eq. 2.20) to be replaced by simple products, expressing k as a function of n layers:

$$k_{\perp}(n) = \frac{1}{12} \left(\frac{nT}{\frac{nTS}{e_f^3} + \frac{(n-1)S^2}{ce_b^3}} \right), \quad (3.1)$$

where, as before, $c = 4$ for the Lattice model and $c = 2$ for the Poisson model. Algebraic manipulation to clear fractions yields:

$$k_{\perp}(n) = \frac{1}{12} \left(\frac{nTce_b^3e_f^3}{nTSce_b^3 + e_f^3(n-1)S^2} \right). \quad (3.2)$$

Eq. 3.2 is useful for investigating the minimum number of layers necessary to approximate the permeability in an REV of a certain size (*Doolin and Mauldon 1996*), given a constant bedding plane aperture.

Applying L'Hôpital's rule and taking the limit as the number of layers grows without bound yields

$$\lim_{n \rightarrow \infty} k_{\perp}(n) = \frac{1}{12S} \left(\frac{cTe_f^3 e_b^3}{cTe_b^3 + Se_f^3} \right). \quad (3.3)$$

Using a disaggregate method such as derived by *Oda* (1985) to obtain an equivalent permeability tensor for a rock mass, under the same assumptions and without incorporating the interlayer effect, the bed-normal permeability would be (*Doolin and Mauldon* 1995)

$$k_{DA} = \frac{e_f^3}{12S}, \quad (3.4)$$

subscript DA denoting a disaggregate k value. We define a dimensionless number ρ relating k_{DA} to the corrected bed normal permeability determined from the interlayer equation, by dividing Eq. 3.4 into Eq. 3.3, i.e.,

$$\begin{aligned} \rho &= \frac{k_{\perp}}{k_{DA}} \\ &= \frac{cTe_b^3}{cTe_b^3 + Se_f^3}. \end{aligned} \quad (3.5)$$

If $S = \beta T$, i.e., if fracture spacing is proportional to bed thickness (*Gross et al.* 1995), ρ may be further simplified to

$$\rho = \frac{ce_b^3}{ce_b^3 + \beta e_f^3}, \quad c = 2, 4. \quad (3.6)$$

The comparison factor ρ is the value that a permeability estimate computed from a disaggregate method derivation such as *Oda* (1985) would have to be multiplied by to equal the bed-normal permeability estimate computed from the interlayer equation. Figure. 3.1 is a graph of Eq. 3.6 where ρ is determined as a function of the bedding plane aperture e_b . The fracture aperture and the spacing coefficient β are held constant at unit values. The effect of the bedding plane aperture

on the bulk conductivity normal to bedding is profound: each order of magnitude reduction of the bedding plane aperture results in a more than two order of magnitude reduction in bed-normal permeability. An estimate of bed-normal permeability based on disaggregate geometry could be grossly in error by failing to correct for the aperture of the bedding planes.

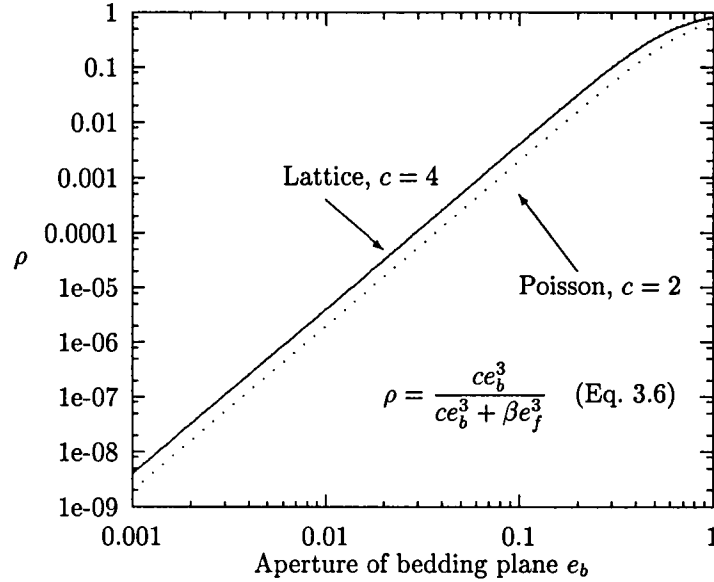


Figure 3.1: Comparison factor ρ (Eq. 3.6) is the amount that a bed-normal permeability computed from a disaggregate geometry (e.g., Snow's Law) would have to be multiplied by to result in the permeability predicted by the interlayer equation.

When the bedding plane aperture is much smaller than the fracture apertures ($e_b \ll e_f$), the bed-normal permeability may be approximated by substituting 0 for e_b and factoring out e_f in Eq. 3.3, yielding,

$$k_{\perp} \approx \frac{1}{12S} \frac{ce_b^3}{\beta}. \quad (3.7)$$

Figure 3.2 compares Eq. 3.3 with Eq. 3.7 for $S = T = 1$ and $\beta = 1$. Equation 3.7 is useful for rapid field estimation of bed-normal permeability, and as an order of magnitude check to validate computational results.

In contrast, assuming $S = T$ and $e_f = e_b = e$, the bed-normal permeability may be further

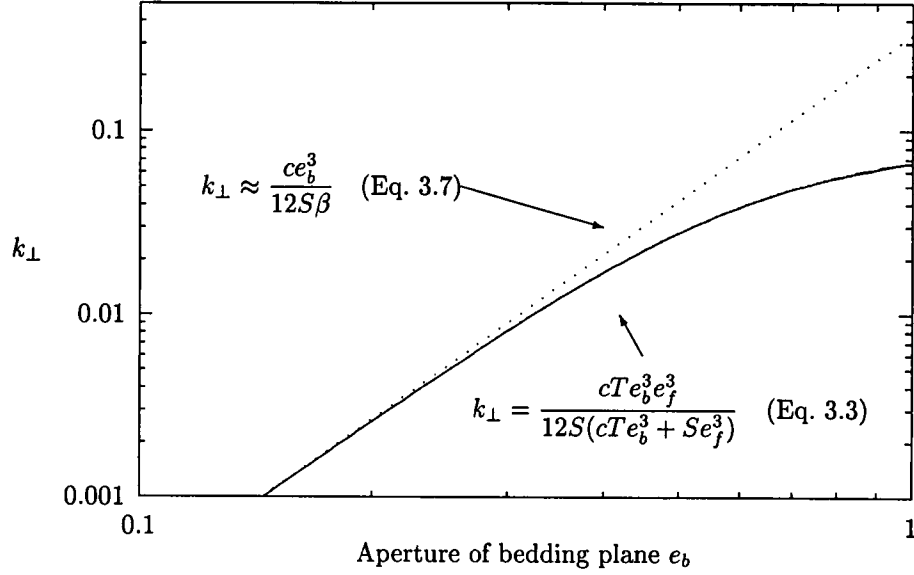


Figure 3.2: Eq. 3.7 closely approximates the bed-normal permeability when $e_b \ll e_f$.

simplified to

$$k_{\perp} = \frac{1}{12} \left(\frac{c}{c+1} \right) \left(\frac{e^3}{S} \right), \quad (3.8)$$

which implies that if all fracture and bedding plane apertures are everywhere equal, and the bed thickness and fracture spacing are also assumed equal to one another (i.e., $\beta = 1$), then the interlayer effect is slight and can be disregarded. For example, when $c = 2$, $k_{\perp} = \frac{2}{3} k_{DA}$; for $c = 4$, $k_{\perp} = \frac{4}{5} k_{DA}$. The practical significance is that, under the above conditions, fracture connectivity does not greatly influence the bed-normal permeability. In this case, uncorrected estimates of permeability derived by disaggregate methods are close approximations.

3.2 Effect of fracture spacing

However useful a comparison factor might be for estimating the extent of the interlayer effect on bed-normal permeability, a subtle but very important point must be observed: ρ is mathematically

correct only under the assumption that fracture spacing in each layer is equal. Disaggregate methods use a single mean fracture spacing S defined for the entire rock mass. The interlayer model uses the expected value S_i of the fracture spacing for each layer i . Before describing the role of fracture spacing in the interlayer effect, it is necessary to define fracture spacing.

3.2.1 Measurement of fracture spacing

Fracture spacing can be defined in different ways, two of which are considered here. In *Oda's* (1985) derivation of a permeability tensor, the permeability is found to be related to the *fracture intensity* P , defined as the mean area of fractures per unit volume of a rock mass (*Dershowitz and Herda* 1992; *Mauldon* 1994). To investigate the relationship between the fracture intensity P and a mean fracture spacing S for the entire rock mass, consider a domain of width w , containing n layers each of thickness t_i . Let the fracture area A equal the area of a planar region that bisects the fracture. As just noted, the fracture intensity P equals the mean area A_f of fractures per unit volume V :

$$P = \frac{A_f}{V}. \quad (3.9)$$

Assuming unit depth in the third dimension (e.g., Fig. 3.3), the total area A_f of fractures in the domain equals the number N_i of fractures in each layer i divided by the product of the width w and the layer thickness t_i , summed over n ,

$$A_f = N_1 t_1 + N_2 t_2 + \cdots + N_n t_n, \quad \text{and} \quad (3.10)$$

$$V = (t_1 + t_2 + \cdots + t_n)w. \quad (3.11)$$

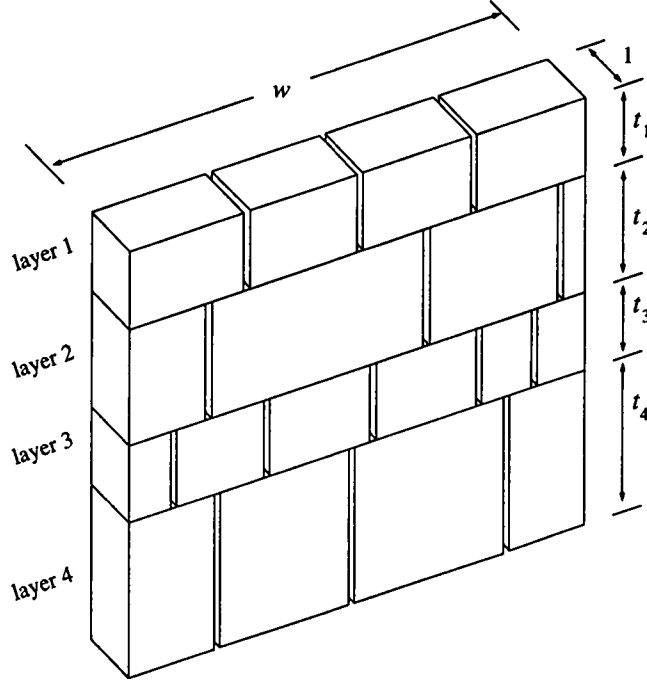


Figure 3.3: Schematic diagram of fractured layered rock. For this example, $w = 100$, $N_1 = 3$, $N_2 = 3$, $N_3 = 5$, $N_4 = 3$, $t_1 = 20$, $t_2 = 30$, $t_3 = 20$, and $t_4 = 40$. The bedding plane apertures e_{ij} all equal 0.1.

Noting that the expected number N_i of fractures in layer i equals the product of the layer fracture frequency λ_i and the domain width w , substitute $\lambda_i w$ into Eq. 3.10 and divide Eq. 3.10 by Eq. 3.11:

$$\begin{aligned}
 P &= \frac{A_f}{V} \\
 &= \frac{\lambda_1 w t_1 + \lambda_2 w t_2 + \cdots + \lambda_n w t_n}{(t_1 + t_2 + \cdots + t_n)w} \\
 &= \lambda_1 \frac{t_1}{\sum t_i} + \lambda_2 \frac{t_2}{\sum t_i} + \cdots + \lambda_n \frac{t_n}{\sum t_i} \quad i = 1, 2, \dots, n \text{ layers} \\
 &= \lambda_1 \frac{t_1}{n\bar{t}} + \lambda_2 \frac{t_2}{n\bar{t}} + \cdots + \lambda_n \frac{t_n}{n\bar{t}} \\
 &= \sum_{i=1}^n \lambda_i \frac{t_i}{n\bar{t}}, \tag{3.12}
 \end{aligned}$$

where $\bar{t}$ is the mean bed thickness, and P is the domain fracture intensity.

For this orthogonal case, the intensity P is exactly the one-dimensional *fracture frequency* λ , defined as the number of fractures per unit length, and reciprocal to the mean fracture spacing for the rock mass (*Dershowitz and Herda 1992; Mauldon 1994*):

$$S = \frac{1}{P}. \quad (3.13)$$

In the interlayer method, an estimate $\hat{S}_i$ of the expected fracture spacing S_i must be determined for each layer, without regard for layer thickness:

$$\hat{S}_i = \frac{w}{N_i}. \quad (3.14)$$

From consideration of Eq. 3.1, one may suppose that a reasonable approximation to the bed-normal permeability could be found using $n\bar{S}$, $\bar{S} = (\hat{S}_1 + \hat{S}_2 + \dots + \hat{S}_n)/n$, instead of the summations $\sum S_i$. However, this substitution will lead to significant error. Note that the layer term ΣB (Eq. 2.4) and interlayer term ΣI (Eqs. 2.12 and 2.19) contain ratios of the form “ S/e .” By expectation, $E(\sum(S/e)) = \sum E(S/e)$, but to equate $E(S/e)$ with $E(S)/E(e)$ is to commit the *fallacy of averages*. In words, the expected value of a ratio is, in general, not equal to the ratio of the expected values. Furthermore, as will be shown in §3.2.2, the fracture spacing ($\hat{S}_i, i = 1, 2, \dots, n$) used in the interlayer equation influences bed-normal permeability more than the fracture spacing ($S = 1/P$) used in a disaggregate method.

3.2.2 Effect of fracture spacing on permeability

Increasing the distance between fractures decreases the bulk permeability of a fractured rock mass in two ways: (1) Widely spaced fractures reduce layer permeability. The bed-normal permeability, without the interlayer effect, is the weighted, harmonic mean of all layer permeabilities; (2) The interlayer correction term ΣI is proportional to the product $S_i S_j$ of fracture spacing. For example,

using a disaggregate model of rock mass geometry, adding a thin rock layer with wide fracture spacing changes the bed-normal permeability of the rock mass very little. Wide spacing weighted by small bed thickness does not have a large quantitative effect on the rock mass fracture intensity. In the interlayer model, such a layer noticeably reduces the bed-normal permeability in the same rock mass.

For an illustrative example, we examine the change in permeability that occurs when a single layer containing widely spaced fractures is introduced into a joint system model. Consider the fractured rock mass schematics shown in Fig. 3.3 with four layers and Fig. 3.4 with five layers. First, the bed-normal permeability for the four layer joint system model must be determined using both the disaggregate method for k_{DA} and the interlayer method for $k_{\perp}$. Then, a layer containing widely spaced fractures (layer 3 of Fig. 3.4) is introduced into the previous, four layer joint system model. In the disaggregate method, k_{DA} decreases only 0.3% when layer 3 is added. In the interlayer method, $k_{\perp}$ decreases 18% when layer 3 is added.

4 Layers: Calculate the bed-normal permeability with disaggregate method for Fig. 3.3, using Eq. 3.4, and the parameters listed in the caption for Fig. 3.3. To simplify the analysis, $e = e_1 = e_3 = e_4 = 0.1$. By Eq. 3.12

$$\begin{aligned}
 P &= \sum_{i=1}^n \lambda_i \frac{t_i}{nt} \\
 &= \frac{N_1}{w} \frac{t_1}{nt} + \frac{N_2}{w} \frac{t_2}{nt} + \frac{N_3}{w} \frac{t_3}{nt} + \frac{N_4}{w} \frac{t_4}{nt} \\
 &= \frac{3}{100} \frac{20}{110} + \frac{3}{100} \frac{30}{110} + \frac{5}{100} \frac{20}{110} + \frac{3}{100} \frac{40}{110} \\
 &= \frac{370}{11000} \\
 &= 3.36 \times 10^{-2}.
 \end{aligned} \tag{3.15}$$

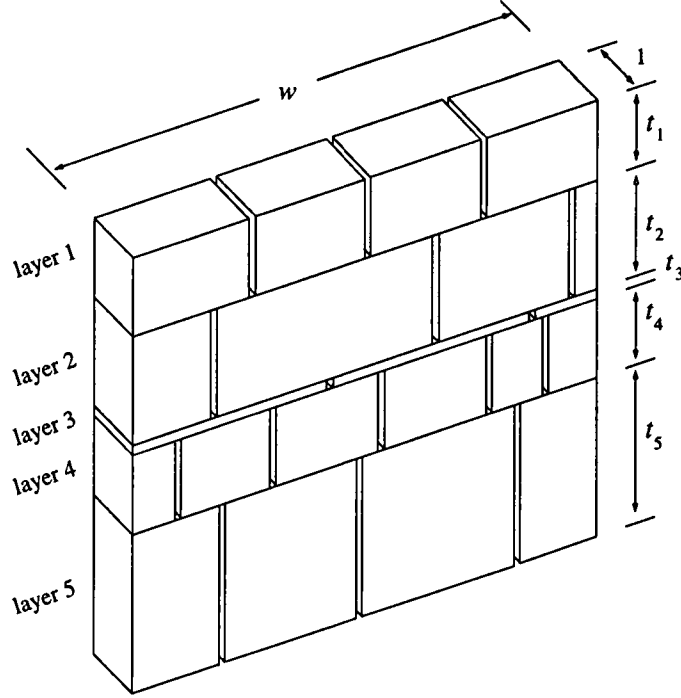


Figure 3.4: Schematic diagram of fractured layered rock. For this example, $w = 100$, $N_1 = 3$, $N_2 = 3$, $N_3 = 2$, $N_4 = 5$, $N_5 = 3$, $t_1 = 20$, $t_2 = 29$, $t_3 = 1$, $t_4 = 20$ and $t_5 = 40$. The bedding plane apertures e_{ij} all equal 0.1.

For the disaggregate solution, spacing $S = 1/P = 29.73$. By substitution,

$$k_{DA} = \frac{e^3}{12S} = \frac{0.1^3}{12 \cdot 29.73} = 2.803 \times 10^{-6}. \quad (3.16)$$

Using the interlayer equation (Eq. 2.9) note that $\Sigma T = 110$, and set the bedding plane apertures e_{ij} equal to the fracture apertures (0.1) to minimize their influence on the result. Next, compute the layer permeability term (Eq. 2.4), noting that $t_i \equiv T_i$:

$$\begin{aligned} \Sigma B &= 12 \sum_{i=1}^n \frac{T_i S_i}{e_i^3} \\ &= \frac{12}{e_i^3} \left(t_1 \frac{w}{N_1} + t_2 \frac{w}{N_2} + t_3 \frac{w}{N_3} + t_4 \frac{w}{N_4} \right) \end{aligned}$$

$$\begin{aligned}
&= \frac{12}{0.1^3} \left(20 \cdot \frac{100}{3} + 30 \cdot \frac{100}{3} + 20 \cdot \frac{100}{5} + 40 \cdot \frac{100}{3} \right) \\
&= 4.08 \times 10^7.
\end{aligned} \tag{3.17}$$

Then compute the interlayer correction term (Eq. 2.19):

$$\begin{aligned}
\Sigma I &= 12 \sum_{i=1}^{n-1} \frac{S_i S_j}{ce_{ij}^3} \\
&= \frac{12}{ce_{ij}^3} \left(\frac{w}{N_1} \frac{w}{N_2} + \frac{w}{N_2} \frac{w}{N_3} + \frac{w}{N_3} \frac{w}{N_4} \right) \\
&= \frac{12}{2 \cdot 0.1^3} \left(\frac{100}{3} \frac{100}{3} + \frac{100}{3} \frac{100}{5} + \frac{100}{5} \frac{100}{3} \right) \\
&= 1.47 \times 10^7.
\end{aligned} \tag{3.18}$$

Finally, compute the interlayer permeability (Eq. 2.20):

$$k_{\perp} = \frac{\Sigma T}{\Sigma B + \Sigma I} = \frac{110}{4.08 \times 10^7 + 1.47 \times 10^7} = 1.98 \times 10^{-6} \tag{3.19}$$

The interlayer permeability is slightly less than the disaggregate permeability.

5 layers: Now, introduce a layer with wide fracture spacing (i.e., layer 3 in Fig. 3.4). Using the same aperture assumption as for the four layer example, and the parameter values listed in the caption for Fig. 3.4, computing the bed-normal permeability k_{DA} by the disaggregate method:

$$\begin{aligned}
P &= \frac{3}{100} \frac{20}{110} + \frac{3}{100} \frac{30}{110} + \frac{2}{100} \frac{1}{110} + \frac{5}{100} \frac{20}{110} + \frac{3}{100} \frac{40}{110} \\
&= \frac{369}{11000}
\end{aligned}$$

$$= 3.35 \times 10^{-2}. \quad (3.20)$$

$S = 1/P = 29.81$. For the disaggregate permeability,

$$k_{DA} = \frac{0.1^3}{12 \cdot 2.75} = 2.795 \times 10^{-6}, \quad (3.21)$$

a 0.3% decrease from the four layer permeability (Eq. 3.16).

Using the interlayer method for five layers, $\Sigma T = 110$. Compute the bedding term:

$$\begin{aligned} \Sigma B &= \frac{12}{0.1^3} \left(20 \cdot \frac{100}{3} + 29 \cdot \frac{100}{3} + 1 \cdot \frac{100}{2} + 20 \cdot \frac{100}{5} + 40 \cdot \frac{100}{3} \right) \\ &= 4.10 \times 10^7. \end{aligned} \quad (3.22)$$

Now compute the interlayer correction term (Eq. 2.19):

$$\begin{aligned} \Sigma I &= \frac{12}{2 \cdot 0.1^3} \left(\frac{100}{3} \frac{100}{3} + \frac{100}{3} \frac{100}{2} + \frac{100}{2} \frac{100}{5} + \frac{100}{5} \frac{100}{3} \right) \\ &= 2.67 \times 10^7. \end{aligned} \quad (3.23)$$

Compute the bed-normal permeability:

$$k_{\perp} = \frac{\Sigma T}{\Sigma B + \Sigma I} = \frac{110}{4.10 \times 10^7 + 2.67 \times 10^5} = 1.63 \times 10^{-6}, \quad (3.24)$$

an 18% decrease from the four layer result. Adding a thin layer of widely spaced fractures has little effect on the bed-normal permeability determined from a disaggregate method, but significantly decreases the bed-normal permeability in the interlayer method.

In Fig. 3.5, the interlayer comparison factor ρ is graphed as a function of the fracture spacing to

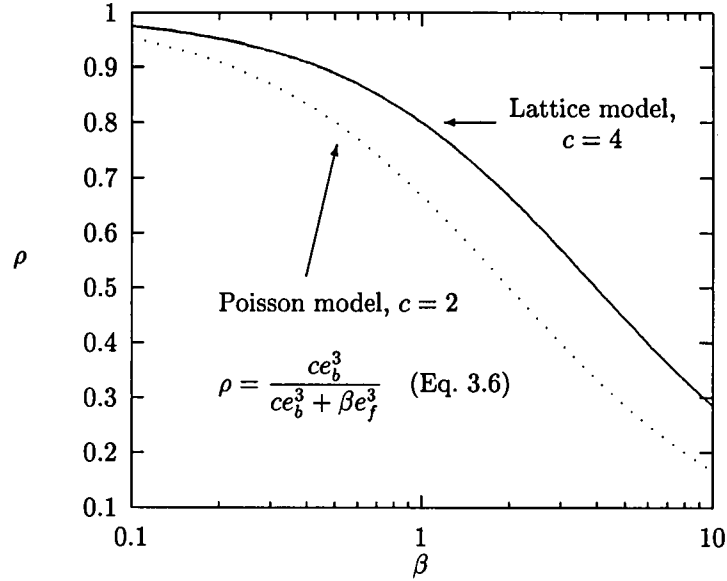


Figure 3.5: Comparison factor ρ as a function of β . Bed-normal permeability decreases as the spacing to thickness ratio β increases.

bed thickness ratio β , holding all other parameters constant and unit value. The graph shows the effect of fracture spacing on bed-normal permeability, with respect to permeability determined from a disaggregate method. As can be seen from Fig. 3.5, widely spaced fractures in relatively thin beds ($\beta > 1$) reduce the bed-normal permeability, relative to permeability predicted by a disaggregate method.

3.3 Effect of bedding plane aperture

Another feature of the interlayer effect is that bed-normal permeability converges to a value controlled by the lower bound of the bedding plane aperture distribution. If bedding plane apertures are variable with a lower limit $e_b > 0$, the bed-normal permeability converges to some value proportional to e_b (e.g., bedding plane apertures are constant and equal). When zero forms the lower limit of the bedding plane aperture distribution ($e_b \geq 0$), increasing the number of layers increases

the probability that an aperture very near zero will occur, greatly reducing permeability. In either case the bed-normal permeability is scale dependent. Three distributional forms of bedding plane apertures are used to investigate the scale dependence of bed-normal permeability. The aperture distributions considered are a two-value discrete distribution, negative exponential and log-normal distributions.

Each distribution differs qualitatively and quantitatively in its effect on bed-normal permeability. The two-value discrete distribution (Fig. 3.6) illustrates the magnitude of influence that the bedding plane aperture has on bed-normal permeability, the log-normal and negative exponential distributions demonstrate the interlayer scale effect.

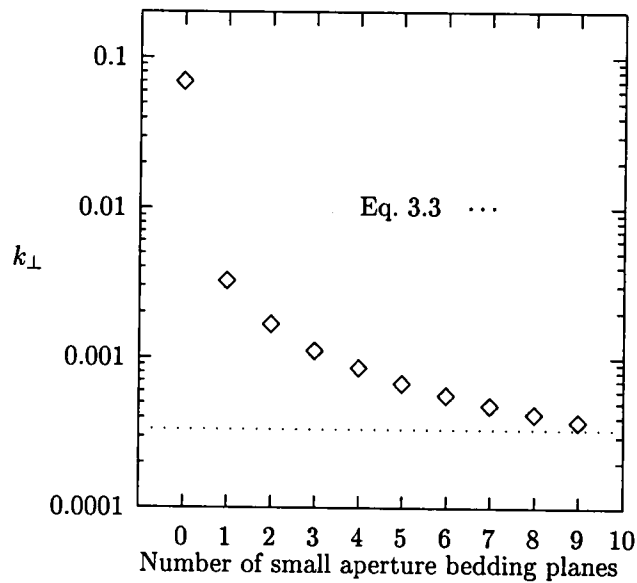


Figure 3.6: Bed-normal permeability $k_{\perp}$ as a function of the number n of layers in a 10 layer system. Permeability decreases radically as the number of small aperture bedding planes increases; the largest decrease occurring at the introduction of one smaller aperture.

First, for the two-value discrete distribution, we consider a case where the fracture apertures, spacings and bed thicknesses are, for convenience, set to some arbitrary constant, say 1, and the bedding plane apertures assume the value of either 1 or 0.1. For a 10 layer system there will be 9

bedding planes, each of which can have an aperture of either 1 or 0.1. Fig. 3.6 shows the change in bed-normal permeability when the number of bedding planes with aperture 0.1 is increased from zero to nine. Introducing just one small aperture produces the greatest change in the bed-normal permeability. In the example shown in Fig. 3.6, $k_{\perp}$ decreases by more than one order of magnitude with the introduction of a single small aperture of 0.1. Successively adding more small apertures reduces the bed-normal permeability further, but the rate of reduction decreases. The bed-normal permeability will converge to a value controlled by the small aperture, as shown by the dotted line in Fig. 3.6.

When zero forms the lower limit of the bedding plane aperture distribution ($e_b \geq 0$), increasing the number of layers increases the probability that an aperture very near zero will occur, greatly reducing permeability. Permeabilities computed for negative exponential and log-normal bedding plane apertures are plotted in Figs. 3.7 and 3.8. Each point on the plot represents the mean of 1000 trial values for permeability computed by the interlayer equation (Eq. 2.20) from synthetic distributions of rock mass parameters. In both figures, the bed thickness, mean fracture spacing and fracture apertures are constant and equal to 1. The exponentially distributed bedding plane apertures (Fig. 3.8) were generated using a parameter equal to 1; the log-normal apertures (Fig. 3.7) were generated with a mean of 1 and standard deviation of 0.1 ($\diamond$), 0.5 (+) and 0.9 ($\square$).

The domain of both distributions is from 0 to ∞ , thus in both cases the permeability ultimately converges to 0. The range of computed permeability differs markedly for the two distributions. The exponential distribution of bedding plane apertures results in permeability that is very scale dependent: the permeability continues to decrease rapidly as the number of layers increases. This scale sensitivity reflects the fact that for the negative exponential distribution the probability density $f(x)$ increases as $x \rightarrow 0$.

In contrast, log-normally distributed bedding plane apertures produced a range of permeabilities that decrease rapidly at first, then the rate of decrease slows quickly as the number of layers

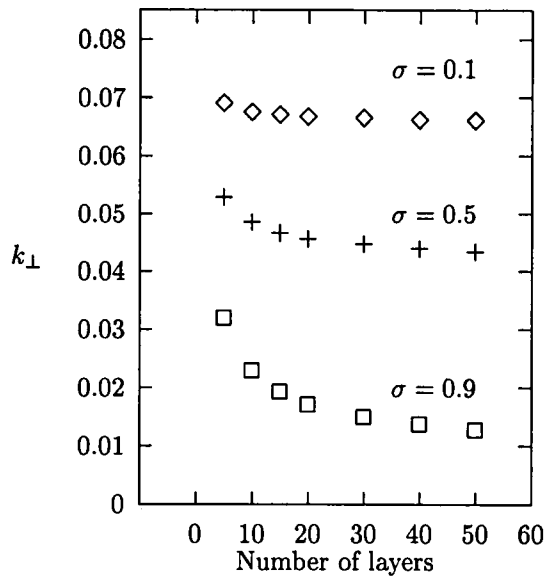


Figure 3.7: Permeability as a function of the number of layers, as computed from an log-normal distribution of bedding plane apertures. The change in bed-normal permeability ($\Delta k_{\perp}$) continues to decrease with increasing layers. The rate of change in bed-normal permeability ($\Delta k_{\perp}/\Delta n$) decreases with increasing n .

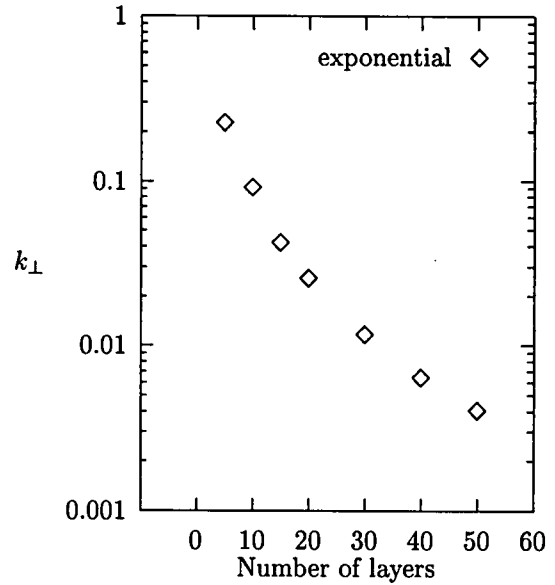


Figure 3.8: Permeability as a function of the number of layers, as computed from an exponential distribution of bedding plane apertures. The change in bed-normal permeability ($\Delta k_{\perp}$) continues to decrease with increasing layers. The rate of change in bed-normal permeability ($\Delta k_{\perp}/\Delta n$) decreases with increasing n .

increases. In other words, even though log-normally distributed apertures eventually produce a permeability indistinguishable from zero, the rate of convergence is much slower than the exponentially distributed apertures, a fact explained by the tendency of log-normal random variables to mass on some mean value greater than zero. Figure 3.7 also shows that as the variability increases, the magnitude of bed-normal permeability decreases, even though it maintains a similar convergent behavior.

For random length fractures with constant apertures, which are not confined to single beds, and where the fracture spacing is independent of bedding, the bedding plane aperture can still affect the bed-normal permeability. To illustrate, the graph shown in Figure 3.9 incorporates Eq. 2.27 into the derivation of the comparison factor ρ (Eq. 3.6) for several values of the bedding plane

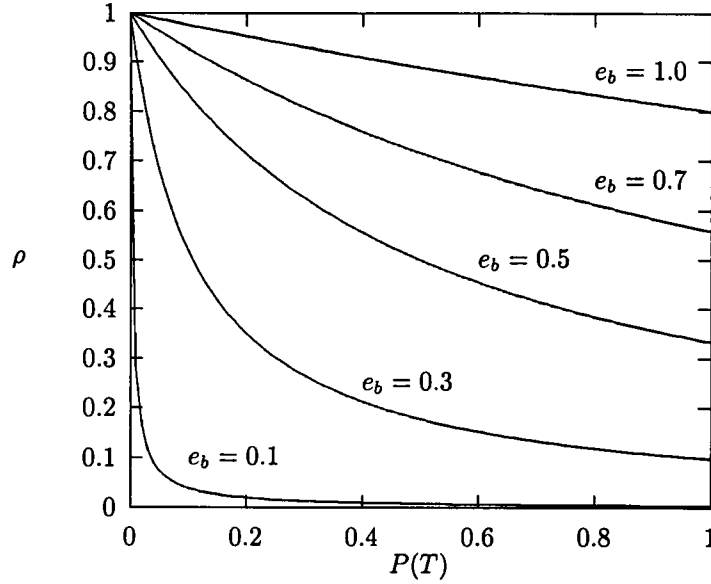


Figure 3.9: Comparison factor ρ as a function of the termination probability $P(T)$. The interlayer effect also applies when fractures extend through several beds but do not cross the entire domain.

aperture e_b . Recall from §2.3 that the probability of termination $P(T) = 0$ for the Snow infinite fracture model, and $P(T) = 1$ when all fractures are delimited by single beds. The values of all other parameters are held constant at unity. When the bedding plane aperture is equal to the fracture aperture, $P(T)$ does not greatly influence the bed-normal permeability. When the bedding plane aperture is an order of magnitude less than the fracture aperture, $P(T)$ greatly influences bed-normal permeability. As the bedding plane aperture decreases relative to fracture aperture, the interlayer effect increasingly reduces bed-normal permeability, even when fractures are not single bed-delimited.

3.4 Interlayer scale effect

The interlayer effect is the result of fracture parameters that influence the permeability of each layer and interlayer. The presence of a single layer with narrow fracture apertures or with widely spaced fractures reduces the bed-normal permeability (Fig. 3.5). Similarly, the presence of a single,

small bedding plane aperture greatly reduces the bed-normal permeability (Fig. 3.6). Each added layer increases the chance that the rock mass will contain very narrow fracture apertures, or very wide fracture spacing, or narrow bedding plane apertures (Figs. 3.7 and 3.8). Weibull's theory of weakest links (*Jaeger and Cook* 1969) provides an analogy. Weakest link theory assumes that the least value controls the behavior of a material. For example, an old cliché states that a "chain is only as strong as it's weakest link." Similarly, flow in the bed-normal direction is controlled by the least permeable bed.

In certain cases, such as shown in Eqs. 3.3 and 3.8, the bed-normal permeability converges to a minimum, non-zero value. This is analogous to a porous medium composed of fine-grained material such as sand. At the grain scale, porous media do not behave as continua (*Bear* 1972; *Freeze and Cherry* 1979). However, the bulk permeability satisfying Darcy's Law can be measured experimentally, or mathematically derived from the properties of connected pore spaces between grains of a porous medium (*Whitaker* 1986). Fractures are considered pore space in a rock mass (*Snow* 1968), implying that rock blocks bounded by fractures and bedding planes are analogous to "grains" in a porous medium. Given that fractures and bedding planes have non-zero apertures, the bed-normal permeability would be expected to have a lower bound controlled, in the limit, by (1), the size of the rock blocks, and (2), the apertures of connected fractures and bedding planes.

As an illustrative example of the interlayer scale effect, consider the following. It has been shown that the permeability is reduced when a layer with widely spaced fractures occurs. Suppose, for example, layer 2 in Fig. 3.3, occurs with a probability p . Then, with independence between layers, the probability P at least one wide spacing layer in a package of $n + 1$ layers is $1 - (1 - p)^n$. For example, if $p = 1/4$ and $n = 10$, $P(\text{at least one layer with wide spacing}) = 1 - (3/4)^{10} = 0.94$. But when $n = 4$, $P(\text{at least one layer with wide spacing}) = 1 - (3/4)^4 = 0.68$. Thus, the interlayer effect depends on the number of layers present.

Chapter 4

Numerical verification

4.1 Introduction

A finite element computer program, `bnpflow`, was written to numerically model the permeability of a two-dimensional fractured, layered rock mass. Numerical experiments using `bnpflow` were carried out with four objectives: (1) verification of the interlayer equation for fractures with constant spacing; (2) verification of the interlayer equation for fracture with negative exponential spacing; (3) investigation of the width of a sampling window necessary to employ the interlayer equation; and, (4) estimation of the error resulting from no-flow boundaries perpendicular to stratification. The program uses the Galerkin Weak Statement (GWS) method (*Baker and Pepper 1991*) to construct a mesh of linear elements simulating fracture traces in a two-dimensional cross-sectional area of a fractured, layered rock mass. The bed-normal permeability for constant head boundaries is calculated from the numerically determined head distribution and the process is repeated through a set of Monte Carlo trials to estimate the expected value of the bed-normal permeability for a given set of fracture parameters.

4.2 Mathematical development

Incompressible, steady-state, laminar fluid flow through a continuous medium can be modeled by the Darcy equation:

$$\mathbf{q} = -\mathbf{K}\nabla\phi. \quad (4.1)$$

The flow vector $\mathbf{q}$ is linearly proportional to a gradient $\nabla\phi$, that is, the potential (head) change $\partial\phi$ over a distance ∂x_i , by a conductivity tensor $\mathbf{K}$, the sign convention ensuring flow from greater to lesser head. The conductivity tensor

$$\mathbf{K} = \frac{g}{\nu} \mathbf{k} \quad (4.2)$$

is a function of the intrinsic permeability (tensor) $\mathbf{k}$ of the medium, the kinematic viscosity of the fluid and the constant of gravitational acceleration g (Bear 1972). In the present problem, we are interested in computing a scalar constant *permeability* coefficient k for the bed-normal direction, but have no information about either the flux, that is, the volume of flow through the volume, or the head distribution Φ .

Noting that incompressibility requires zero divergence, the continuity equation

$$\nabla \cdot \mathbf{q} = 0, \quad (4.3)$$

holds. When $\mathbf{K}$ is not dependent on spatial location, substitution of Eq. 4.1 into Eq. 4.3 produces the Laplace equation,

$$\mathcal{L}(\Phi) = -\mathbf{K}\nabla^2\phi = 0, \quad (4.4)$$

an elliptic partial differential equation, $\mathcal{L}$ denoting a linear differential operator. Note that the Laplace equation implies that the head distribution does not depend on the constant of gravitational acceleration g or the kinematic viscosity ν . We take this opportunity to dispose of these constant

factors by multiplying their inverses through both sides of Eq. 4.4, leaving us free to concentrate on the intrinsic permeability k . A solution of the Laplace equation under the appropriate boundary conditions can produce the head distribution Φ , whence flow is computed, leaving k as a single unknown in the Darcy equation (Eq. 4.1).

The Laplace equation can be solved analytically for an exact solution only in domains with simple geometry, such as rectangular or radial domains and with uniform boundary conditions. For domains with irregular boundary conditions or unusual geometry, Eq. 4.4 must be solved numerically for an approximate solution by discretizing the domain into a finite number of sub-domains, i.e., elements. Many different numerical methods are available, such as the finite difference method (*Bear 1972; Freeze and Cherry 1979; Wylie 1982*), variational methods, such as Rayleigh-Ritz (*Farlow 1992*) and weighted residual methods, such as collocation (*Seegerlind 1984*) and the Galerkin Weak Statement (*Becker et al. 1981; Baker and Pepper 1991*). The Galerkin Weak Statement method provides a general methodology, exactly reproducing the finite difference and variational methods for identically stated problems, but allowing solution of first derivative (e.g., convection) terms, which is not possible using variational methods (*Seegerlind 1984; Baker and Pepper 1991*). The following notation and derivation of the Galerkin method closely follows *Baker and Pepper (1991)*.

4.2.1 Approximating a solution

Since the analytic solution is intractable, an approximate, numerical solution $\Phi^N(x)$ of Eq. 4.4 may be constructed in the form

$$\Phi^N(x) = \sum_{i=1}^N \phi_i \Psi_i(x_1, x_2, \dots, x_n), \quad (4.5)$$

where n is the problem dimension, and N is the number of elements in a problem domain denoted by Ω . Finding an approximate solution requires finding a suitable set of trial functions $\Psi_i(x_n)$ and their coefficients, the head value on each element ϕ_i . The numerical solution is not exact; the exact

solution $\Phi(x)$ is composed of the numerical solution and the residual error e^N of the approximation:

$$\Phi(x) = \Phi^N(x) + e^N(x). \quad (4.6)$$

By linearity, $\mathcal{L}(\Phi) = \mathcal{L}(\Phi^N) + \mathcal{L}(e^N) = 0$. Since, by Eq. 4.4, $\mathcal{L}(\Phi) = 0$, $\mathcal{L}(\Phi^N) = -\mathcal{L}(e^N)$.

The weighted residuals can be forced to vanish by imposing functional orthogonality, minimizing the error in the solution when integrating over the problem domain Ω :

$$\int_{\Omega_i} w_i(x_n) \mathcal{L}(\Phi^N) d\tau = 0, \quad i = 1, 2, \dots, N, \quad (4.7)$$

where $d\tau$ denotes the volume differential $dx_1 \dots dx_n$, $\mathcal{L}(\Phi^N)$ is the approximated differential equation and $w_i(x_n)$ is a weight function. To exactly match solution values at the i th element endpoints x_0 and x_b , $w_i(x_0) = w_i(x_b) = 0$. We conclude that $\mathcal{L}(\Phi) = \mathcal{L}(\Phi^N) = 0$ identically for all $w_i(x_n), i = 1, 2, \dots, N$, the conclusion proven by contradiction (*Weinstock 1974*).

The Galerkin criterion requires each weight function to equal the corresponding trial function, i.e. $w_i(x) = \Psi_i(x)$. The trial function set approximates the solution over each of N elements in the problem domain Ω , for all $N \in \Omega$. Substituting the problem formulation (Eq. 4.4) into Eq. 4.7 under the Galerkin criterion results in

$$\int_{\Omega} \Psi_i(x) (-\mathbf{k} \nabla^2 \phi^N) d\tau = 0. \quad (4.8)$$

Using $\int v du = uv - \int u dv$ and rewriting results in the Galerkin Weak Statement,

$$\begin{aligned} GWS &= \int_{\Omega} \nabla \Psi_i \cdot \mathbf{k} \nabla \Phi^N d\tau - \int_{\partial\Omega} \Psi_i \mathbf{k} \nabla \Phi^N \cdot \mathbf{n} d\sigma, \\ &= \int_{\Omega} \nabla \Psi_i \cdot \mathbf{k} \nabla \Phi^N d\tau + \int_{\partial\Omega_1} \Psi_i q_n d\sigma - \int_{\partial\Omega_2} \Psi_i \mathbf{k} \nabla \phi^N \cdot \mathbf{n} d\sigma = 0. \end{aligned} \quad (4.9)$$

The $d\sigma$ indicates integration along the domain boundary $\partial\Omega$. Integrating Eq. 4.8 by parts “weakens” the required order of differentiability for the trial functions $\Psi_i(x)$, allowing the use of a linear basis interpolant for a second-order differential equation, and produces a mathematically symmetric statement. Equation 4.9 is identical to a functional whose minimum solution also satisfies Eq. 4.4, the Laplace equation (Weinstock 1974; Farlow 1992). Another benefit of the Galerkin method is that the flux ($\mathbf{q} = -k d\phi/d\mathbf{n}$) boundary conditions are intrinsic.

A Darcy experiment for bulk permeability can be conducted under constant head boundary conditions,

$$\Phi(x_0) = \phi_0 \quad \text{and} \quad \Phi(x_b) = \phi_b, \quad (4.10)$$

the subscripts 0 and b indicating the boundary locations of a flow domain Ω . This allows the flux boundary conditions in Eq. 4.9 to be disposed of and replaced by a column vector $\{B\}$ of known head boundary conditions. Furthermore, by substituting Eq. 4.5 into Eq. 4.9 and noting that a two-dimensional fracture flow problem is topologically one-dimensional, the weak statement for any element i may be simplified to

$$GWS = \int_{\Omega_i} \frac{d\Psi_i}{dx} \cdot k_i \frac{d\Psi_i}{dx} dx \{\Phi\}_i - \{B\}_i = \{0\}, \quad (4.11)$$

where $\{\Phi\}$ is a column vector of unknown head values and x is the length over an element.

4.2.2 Discretizing the weak statement

Solving the GWS numerically, to satisfy Eq. 4.5, requires a specific definition of the trial function set $\Psi(x)$ in terms of finite element basis functions of degree α , denoted by $\{N_\alpha(\zeta_j)\}$. Power series formulations and high-order polynomial interpolation, while possessing the necessary mathematical properties, oscillate wildly in higher order terms. A better function set is obtained by using lower order polynomial interpolation of element endpoints in Ω . The Lagrangian interpolation scheme

provides such a set of polynomials because the unknowns of Eq. 4.5 correspond exactly to the linear endpoints (knots) of the Lagrange polynomial. The basis functions $\{N_\alpha(\zeta_j)\}$ are composed of the functions ζ_j , locally and uniformly defined for each element $N \in \Omega$, allowing the *GWS* to be discretized and a global matrix statement (e.g., $[A]\{x\} = \{b\}$) to be assembled. In light of the *GWS* (Eq. 4.11) formulation, the finite element basis function set $\{N_\alpha(\zeta_j)\}$ must have at least one non-vanishing derivative, and the product of such derivatives must be integrable.

In Darcy flow, the change in head is a linear function of the length of the flow path ($k d\phi = q dx$), justifying the use of line elements (*Wilson 1970; Long et al. 1982*). Consider the line element shown in Fig. 4.1. Interpolating a quantity across the length of the element requires the linear polynomials ζ_1 and ζ_2 to exactly match the unknown quantity at x_i and x_{i+1} . The linear polynomials are written in terms of their slopes, using the element endpoint as a local origin. Using the local coordinate $\bar{x}$, and normalizing by the length of the element ℓ_e , provides the basis function set for one-dimensional, linear interpolation

$$\{N_1(\zeta_j)\} = \begin{Bmatrix} \zeta_1 \\ \zeta_2 \end{Bmatrix} = \begin{Bmatrix} 1 - \frac{\bar{x}}{\ell_e} \\ \frac{\bar{x}}{\ell_e} \end{Bmatrix}, \quad (4.12)$$

ℓ_e indicating the length for any element $N \in \Omega$. To simplify the notation, let $\{N_1\} = \{N_1(\zeta_j)\}$ and substitute the discretizations $\{N_1\} = \Psi(x)$ and $\Phi^N = \Psi_i(x)\phi_i = \{N_1\}^T \{\Phi\}_i$. The discretized Galerkin Weak Statement GWS^h for Eq. 4.11 is

$$GWS^h = \int_{\Omega_i} \frac{d\{N_1\}}{dx} \cdot k_i \frac{d\{N_1\}^T}{dx} dx \{\Phi\}_i - \{B\}_i = 0, \quad (4.13)$$

i indicating an element $N \in \Omega$ and the vector transpose $\{N_1\}^T$ reflecting the inner product resulting from interpolating the scalar coefficients Φ_i .

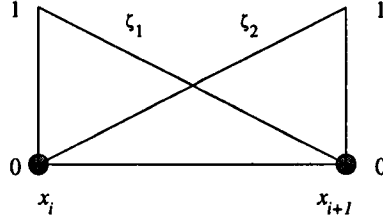


Figure 4.1: A linear basis constructed from a first order Lagrange interpolant.

4.2.3 Assembly of elements

The finite element discretization to compute an approximate solution for a value $\{\Phi\}$ at each of the interpolation nodes of Eq. 4.13 results in the symmetric matrix statement

$$[D]\{\Phi\} = \{B\}, \quad (4.14)$$

where $\{\Phi\}$ is head, $\{B\}$ is constant head boundary conditions, and $[D]$ is the so-called “diffusion” matrix, defined on each element i by

$$[D]_i = \frac{e_i^3}{12\ell_i} \frac{d\{N_1\}}{dx} \frac{d\{N_1\}^T}{dx}. \quad (4.15)$$

Note that $e_i^3/12 \equiv k_i$, and that $1/\ell_i$ is a common factor of all matrix entries in $[D]_i$. The discretized weak statement ($GW S^h$) is then assembled over the problem domain Ω :

$$GW S^h = \bigcup_{i=1}^N ([D]_i \{\Phi\}_i - \{B\}_{\partial i}) = \{0\}. \quad (4.16)$$

$\bigcup_i$ indicates the matrix assembly of all element solutions into a finite element *mesh*.

In most finite element models, a diffusion matrix $[D]_i$ is derived for one “standard” element i . $[D]$ is assembled from $[D]_i$ over the domain of N elements: $[D] = \bigcup_{i=1}^N [D]_i$. For bed-confined fractures located according to a stochastic process, and occurring in beds of random thickness, the fracture element lengths take the values of the bed thicknesses. The length of connecting elements

along a bedding plane is determined by the distance between adjacent fracture intersections with that bedding plane.

4.3 Implementation

The numerical model solving the Laplace equation follows the same assumptions as the analytic model: laminar, steady-state flow through an impermeable rock mass; the fractures are orthogonal to and confined to beds. The computer program `bnpflow` uses constant and uniformly distributed head on sides 2 and 4 of the problem domain, with no flow on sides 1 and 3 (Fig. 4.2), to simulate Darcy flow conditions. This is similar to discrete fracture modeling methods advocated by *Dershowitz and LaPointe* (1994). Furthermore, the no flow conditions on side 1 and 3 are *required* to preserve the mass continuity assumption explicitly stated in the analytic derivation (Eq. 2.20). The principal conductivities are assumed to be parallel and perpendicular to bedding, so that bulk flow parallels gradient normal to layers (*Bear* 1972; *Long et al.* 1982).

Fractures are represented as line elements and the flow through each fracture is determined by the fracture aperture cubed (page 8). Fracture locations within a layer are integer multiples of the mean spacing in the case of the Lattice model, or random variables following a negative exponential distribution in the case of the Poisson model. The negative exponential random numbers are generated using algorithms given by *Priest* (1993), implemented in C. The mesh is assembled by determining the fracture connectivity, then the assembled mesh is written to a file as a sparse matrix system in `matlab` format. The program invokes `matlab` with a system call and suspends program execution until `matlab` returns nodal solutions (head) into a file. `bnpflow` then reads the text file containing head values to compute the equivalent bed-normal permeability.

To calculate the equivalent analytical solution for any given trial, an estimate, $\hat{S}$, of the expected value of the fracture spacing $E(S)$ in each layer is calculated from the actual fractures in the layer,

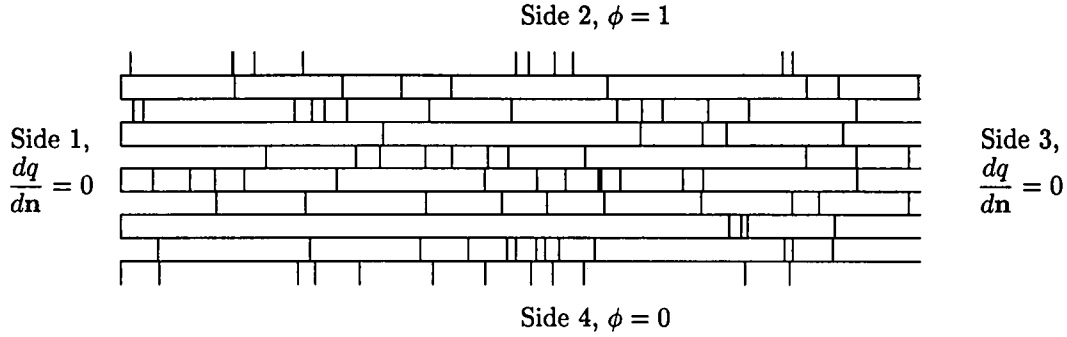


Figure 4.2: An example 10 layer mesh of the Poisson model. The fracture spacing is nominally 10. The permeability computed for this trial was $k = 1.65 \times 10^{-2}$; the analytic prediction is $k = 1.75 \times 10^{-2}$. The relative error between the analytic and computed values is 5.9%.

just as if the fractures had been sampled by a scan line or scan window. The number of fractures n are counted, then divided into the width of the domain L for that trial (*Priest* 1993): $\hat{S} = L/n_i$. The estimated mean fracture spacing for each layer is then used to calculate the equivalent analytic solution, according to Eq. 2.20, for comparison to the result from the numerical trial. To get the permeability of the numerical trial, the flow is calculated through each layer from the numerically determined head distribution. In some trials, there is a slight amount of variability in the flow through each layer; some layers gain a little, some layers lose a little (i.e., numerically, $\nabla \cdot \mathbf{q} \neq 0$). This violation of model assumptions is assumed due to round-off and truncation errors in a solution matrix where adjacent entries vary up to 6 orders of magnitude. A mean flow $\bar{q}$ is determined by averaging flows through each layer. The equivalent bulk permeability k for the fractured rock mass was determined by multiplying the mean flow $\bar{q}$ by the total layer thickness ΣT and dividing by the product of the domain width W and head difference across the domain $\Delta\phi$:

$$k_{\perp} = \frac{\bar{q}\Sigma T}{W\Delta\phi}. \quad (4.17)$$

4.4 Finite element modeling

Four sets of experiments using `bnpflow` were conducted to: (1) Verify the analysis for constant fracture spacing. (2) Verify the analysis of negative exponential fracture spacing. (3) Investigate the effect of the sampling width of the domain parallel to layers on the bed-normal permeability. (4) Determine the existence of boundary effects on bed-normal permeability. Experiments 2 and 3 estimated the expected value of the bed-normal permeability from the mean of a number of trials. Each trial computed the bed-normal permeability for a mesh with negative exponentially spaced fractures. The lesser number of trials for experiment 3 reflects the larger size of the experiment problem domain.

4.4.1 Verification of the Lattice model

To verify the analysis of the interlayer equation for the Lattice model, the bed thickness, fracture aperture, bedding plane aperture and fracture spacing were set to unity. The number of layers were varied from 2 to 10, in 2 layer increments. The width of the problem domain was set to 50. Fractures from one layer intersected intervals between fractures in successive layers at the midpoint of the interval. Figure 4.3 shows a comparison between analytical and numerical results for the Lattice model. The diamonds ($\diamond$) indicate the bed-normal permeability predicted by the interlayer equation. The crosses (+) represent the bed-normal permeability determined numerically using `bnpflow`. For constant fracture spacing, the numerical results agree very closely with the analytical predictions. The slight discrepancy between the analytic and numerical results is interpreted to be a consequence of the no-flow boundary (e.g., sides 1 and 3 of Fig. 4.2). Figure 4.3 also illustrates the interlayer scale effect (§ 3.4): the bed-normal permeability is reduced as the number of layers increases.

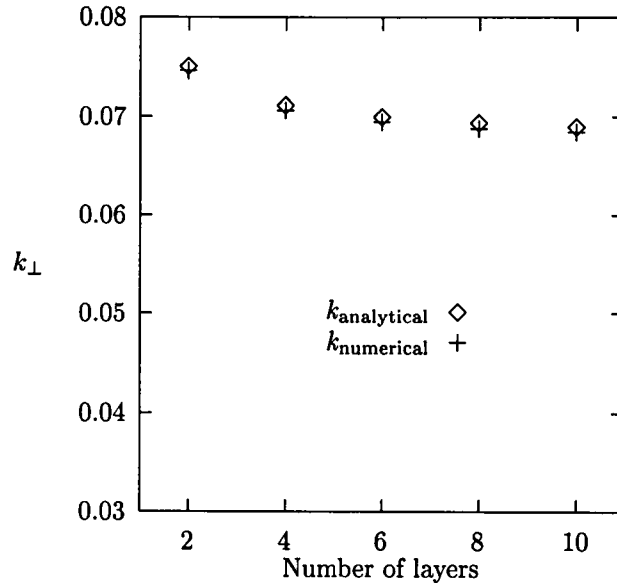


Figure 4.3: Results from finite element analysis of the Lattice model. Numerical and analytical results closely agree for the Lattice model.

4.4.2 Verification of the Poisson model

In the numerical analysis of the Poisson model, the parameter of negative exponential spacing was equal to 1, producing a mean spacing equal to 1. The bed thickness, fracture aperture, and bedding plane aperture were set equal to unity. The width of the domain was set to 50. The number of layers was varied from 2 to 10, in 2 layer increments. Figure 4.4 shows a comparison between the numerically determined bed-normal permeability obtained from the mean of 100 trials (shown with crosses (+)), and the analytical prediction using the interlayer equation (shown with diamonds ($\diamond$)). The analytical and numerical results appear to converge, but to different values. The relative error, defined as the difference between the analytical prediction and numerical result, normalized by the analytical prediction $(k_{\perp} - k_{\text{FE}})/k_{\perp}$, ranged from 8% to 25%, and increased with the number of layers. The interlayer equation systematically over-predicts the bed-normal permeability with respect to the numerically computed value (Fig. 4.4), although the order of magnitude of k in each case is similar. The difference may be due to the mean shortest path length assumption used in

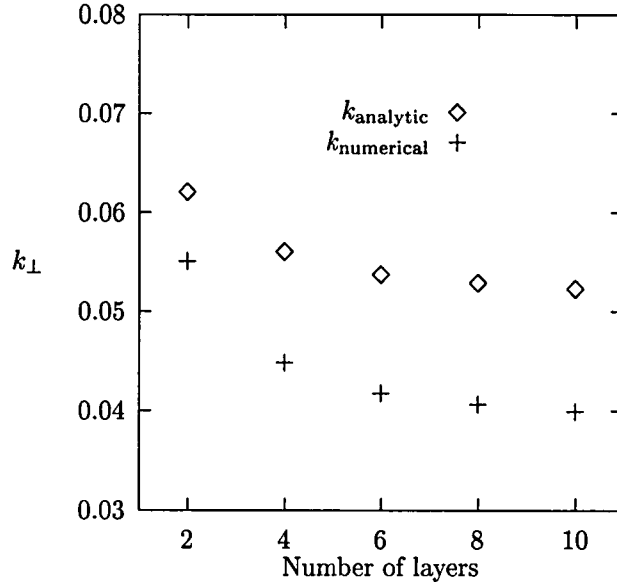


Figure 4.4: Results from the finite element analysis of the Poisson model. In the Poisson model, the analytic and numerical values for bed-normal permeability appear to converge to different values.

the analysis (2.2.3) to derive an expression for interlayer thickness. In addition, Fig. 4.4 illustrates the interlayer effect: the bed-normal permeability is reduced as the number of layers increases.

4.4.3 Width of sampling window

Figure 4.5 shows that in the Poisson model, the permeability increases as the domain width, or equivalently, the number of fractures, increases. The fracture spacing is an exponentially distributed random variable with parameter 0.1, for a mean fracture spacing of 10. The bedding plane aperture, bed thickness, and fracture aperture are set to unit values, and the number of layers equals 20. The diamonds ($\diamond$) represent an analytic prediction of permeability using fracture spacing data obtained from the numerical model. Each data point (+) represents the mean permeability obtained from 30 trials of the finite element model. The dotted line is the permeability value obtained from using the expected value of the spacing in Eq. 3.2. In Fig. 4.5, the rate of change of the permeability with respect to domain width can be seen to decrease rapidly at a domain width of 300, implying that

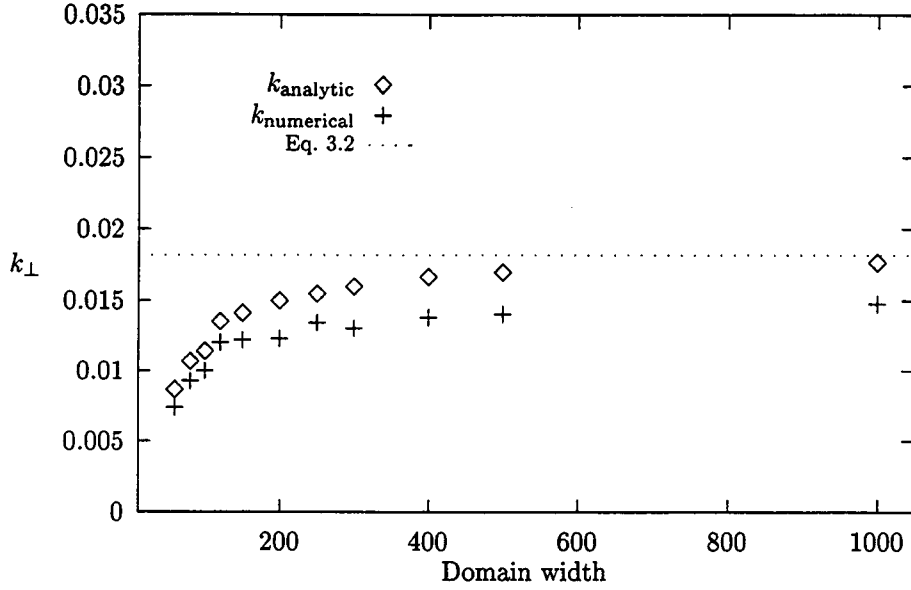


Figure 4.5: Graph of bed-normal permeability $k_{\perp}$ as a function of domain width. Permeability increases with the number of fractures sampled.

for a mean fracture spacing of 10 the sampling window should include 30 fractures on average.

The permeability increase parallel to layers is interpreted to be a result of sampling bias. Although the parameters used by the negative exponential generating function were identically 0.1 for each layer, the point estimate $\hat{S}$ of spacing in adjacent layers differed, in some cases, up to a factor of four. When the domain width is small compared to the fracture spacing, the estimated mean spacing $\hat{S}$ is too high, resulting in lower permeability. As the domain increases in width, the estimated mean spacing $\hat{S}$ decreases, producing an increase in permeability. The reason for the bias is illustrated in Fig. 4.6. The random variables X_L and X_R represent “leftover” spacing that is not counted in the population spacing. The leftover spacings X_L and X_R imply that the estimate $\hat{S}$ will, statistically, have a greater value than the expected value S_i , and that $\hat{S}$ will statistically converge to S_i from a higher value. It follows that decreasing the fracture spacing increases the bed-normal permeability.

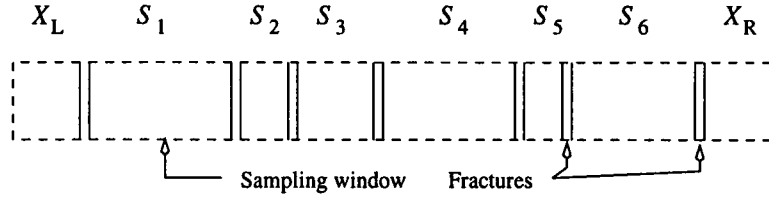


Figure 4.6: A sampling window to determine fracture spacing. Fracture spacing is over-estimated because of bias in sampling due to boundaries.

4.4.4 No-flow boundary effects

No-flow boundaries on sides 1 and 3 (Fig 4.2) produce a difference in permeability values, because the analytic solution assumes an unbounded domain. Each data point in Fig. 4.7 represents 1 trial on a mesh where the spacing was constant with fractures located at interval midpoints. The bed thicknesses, fracture spacing, fracture and bedding plane apertures were set to unity. Figure 4.7 shows that increasing the domain width decreases the relative error of the analytic solution with respect to the numerical solution. The no-flow boundary produces little more than 4% relative error at worst, decreasing to 0.6% in the widest domain that was tested.

4.5 Discussion

The formulation of the numerical problem is nodally exact because of the linear nature of the differential equation $k\nabla^2\phi = 0$, the linear basis functions $\{N_1\}$, the absence of interpolated source or other data, and the direct solution for the matrix statement $[D]\{\Phi\} = \{B\}$ resulting from the finite element mesh assembly. Nevertheless, the bed-normal permeability determined from the interlayer equation systematically over-predicts relative to the numerically computed bed-normal permeability. Systematic over-prediction could result from the presence of the no-flow boundary, from numerical errors accumulated while solving the matrix statement, or from differences in flow path length assumptions between the interlayer model and the finite element model.

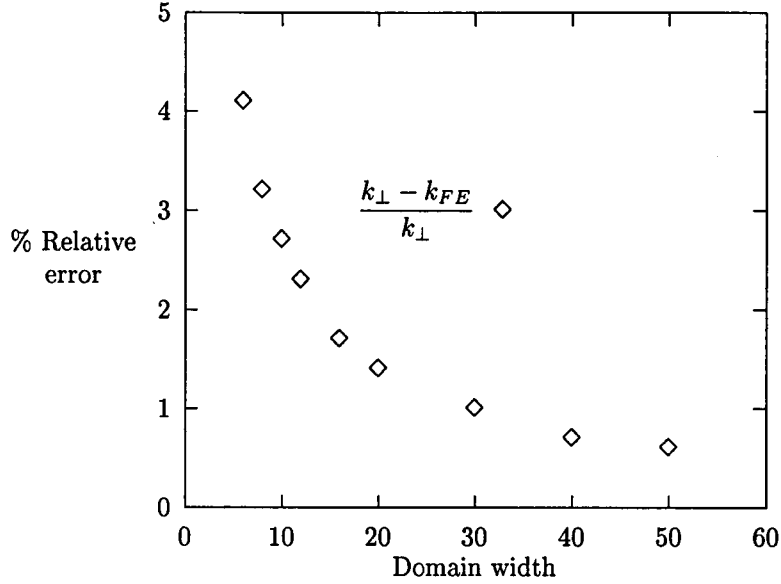


Figure 4.7: Graph shows the effects relative error $(k_{\perp} - k_{FE})/k_{\perp}$ decreasing in wider domains, an effect of the no-flow boundary condition.

In the Lattice model, where fractures were equally spaced and were located at interval midpoints, the percent relative error ranged from 4.2% to 0.6%, as shown in Fig. 4.7.

In the Poisson model, however, for any particular configuration of the fracture mesh (e.g., Fig. 4.2), the bed-normal permeability predicted using the interlayer equation may over- or under-predict relative to the numerically computed permeability. On average, for a statistically significant number of trials, the interlayer equation consistently over-predicts the permeability by 8% to 25% (Fig. 4.4), relative to the permeability computed by the numerical solution. The over-prediction is most likely due to the assumed distribution of path lengths along the bedding planes. Assuming flow always travels to the nearest fracture, either next or previous, neglects a small amount flow in the interlayer. In the finite element model, flow from some fractures may split and go to different sink fractures, or may join with flow from a different source fracture. Splitting flow paths increases the amount of head loss incurred by flow along the bedding plane. In terms of the interlayer, the analytic solution underestimates the interlayer thickness T_{ij} .

The finite element model must conserve flow between layers for comparing a numerically determined permeability to an interlayer equation prediction for a fracture network with identical geometry. Small differences (around 1%) in the computed flow through each layer are assumed to be a result of error occurring during the numerical solution of the matrix statement. The flow depends in part on the inverse length (ℓ_i) of the (line) element (Eq. 4.15) between adjacent fractures. The Poisson process can result in fractures that are very close to each other in successive beds. If two adjacent fractures are very close, the inverse distance between the fractures is a large number. Adjacent entries in the diffusion ($[D]$) matrix were observed to have up to six orders of magnitude difference, which could have caused imprecision errors (round off, etc.) to occur during matrix statement solution. Since the permeability is determined from the flow, this error has an effect on the bed-normal permeability calculated from the finite element model. In an extreme case, two fractures in successive beds terminate on the same point resulting in a zero length flow path. Theoretically, zero length flow paths occur with probability zero. In practice, finite precision computation cannot exclude the possibility of a zero length event, which would result in a division by zero (Eq. 4.15) and produce spurious results.

The effect of the no-flow boundary in the Poisson model is more difficult to measure than in the Lattice model. Unfortunately, increasing the domain width of the Poisson model to reduce the boundary influence, increases the likelihood of extremely short element lengths.

Chapter 5

Illustrative example

5.1 Geologic setting

The Cambrian-aged Conasauga Group of eastern Tennessee (USA) provides a potential application of the interlayer equation. The Conasauga Group outcrops in the Valley and Ridge Province of the Southern Appalachians in a belt about 60 km wide, between the western Blue Ridge and the Cumberland Plateau (Fig. 5.1). The Valley and Ridge is a fold and thrust belt topographically expressed as a series of long parallel ridges and valleys. Drainage is dendritic in headwater areas along ridges, feeding trellis streams. Rivers occupy the major valleys.

Cambrian-aged rocks are the earliest record of cratonic flooding in North America (*Weber* 1988). The Conasauga Group is an intra-shelf sequence deposited in a deltaic system fed from the present northwest. Conditions were periodically favorable for carbonate deposition. The Conasauga Group ranges from relatively pure carbonates in the northeast Valley and Ridge, becoming progressively more clastic towards Oak Ridge Reservation to the southwest. The Allegheny orogeny subsequently caused thrust faulting, folding and fracturing, characteristic of Valley and Ridge structural geology. Rapid stratigraphic changes and fractured deformation zones have created a groundwater system

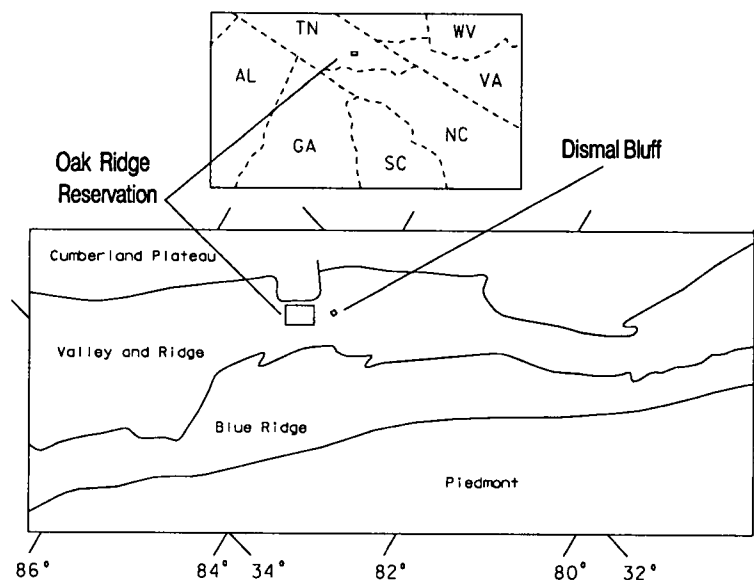


Figure 5.1: Location map showing proximity of Dismal Bluff study area to the Oak Ridge Reservation. After *Lemiszki* (1992).

with anisotropic and heterogeneous hydrogeologic behavior.

Conasauga Group rocks outcrop along the east and south side of Dismal Bluff road, near Clinton, Tennessee, for about one kilometer and are exposed in several road cuts. The road cuts are 3 to 8 meters high and from 20 to 150 meters in length. The exposure is located on the hanging wall of the Whiteoak Mountain thrust sheet and bedding dips between 30 to 90 degrees southeast. There are several small faults from 1–10 m long exposed in the road cut, and folding ranges from 10 cm to 3 m in amplitude.

The Conasauga Group is composed of six formations, listed from oldest to youngest (Figure 5.2): Pumpkin Valley Shale, Friendship Formation, Rogersville Shale, Dismal Gap Formation, Nolichucky Shale, and Maynardville Limestone (*Hasson and Haase* 1988; *Lemiszki* 1992). Field inspection of Conasauga Group rocks reveals that all formations can be generally described as composed of five units, divided according to lithology and fracture characteristics. The Dismal Gap Formation is exposed in the road cut, which is nearly perpendicular to strike, for approximately 45 meters. The three most prominent units in the Dismal Gap Formation are:

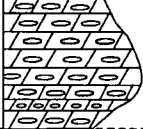
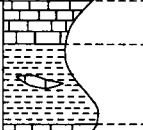
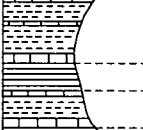
		Lithology	Thickness (m)	Formation	
CAMBRIAN	UPPER		244-355	€cr	Copper Ridge Dolomite
			100-110	€m	Maynardville Ls.
			150-180	€n	Nolichucky Sh.
	MIDDLE		98-125	€dg	Dismal Gap Fm. (Maryville Ls.)
			25-34	€rg	Rogersville Sh.
			31-37	€f	Friendship Fm. (Rutledge Ls.)
	LOWER		56-70	€pv	Pumpkin Valley Sh.
			122-183	€r	Rome Fm.

Figure 5.2: Stratigraphy of the Conasauga Group, after *Hatcher et al.* (1992). The names in parentheses are regional names for formations locally renamed specifically to characterize lithologic conditions in the Oak Ridge Reservation area.

- Shale: 0.1–10 m, very soft green to maroon shale that weathers dark brown to dark red, yellow limonite staining is locally common, lithologic bedding in this unit is either incipient, or bedding parallel slip surfaces, weathers by semi-fissile slaking becoming saprolitic near top of road cut.
- Thin limestone: 0.2–2 m, light to dark gray, calcareous siltstone to argillaceous limestone, laminar bedding, commonly interbedded with thin (1–2 cm) of green shale, fracturing irregular, polygonal networks, locally enlarged by limestone dissolution, or vein-filled with calcite.
- Intraclastic limestone: 0.1–1 m, light gray, oolitic, bioclastic, or intraclastic (flat-pebble conglomeratic) limestone, 0.1–1 m bedding, unlaminated, rare glauconite, vein swarms locally common, widely spaced (0.5–1.5 m), bed normal solutionally enlarged fractures (0.5 m) with apertures to 2 cm.

The lithology of the upper Dismal Gap Formation as well as the lower to middle Nolichucky

Shale was mapped at 10 cm scale (*Doolin and Mauldon 1995*). Mapping recorded 188 lithology transitions, with a mean unit thickness of 1.4 m. Also noted was the fact that many bedding plane partings existed within any particular lithology. Although individual beds within a lithologic unit contained bed-confined fractures, the thickness and number of intra-lithology beds was not recorded.

Lithology	Thickness T_i (meters)	Spacing S_i (meters)	e_f (meters)	e_b (meters)
Intraclastic ls.	0.2	0.5	0.01	0.001
Thin ls.	0.05	0.1	0.001	0.0001
Shale	0.05	0.025	0.0001	0.0001

Table 5.1: Representative fracture data for the Conasauga Group (*Doolin and Mauldon 1995*).

Table B.1 shows typical rock mass parameters in the Dismal Gap Formation (*Doolin and Mauldon 1995*). The thickness and fracture spacing are mean values. The aperture values should be regarded as order of magnitude estimates due to the difficulty in accurately measuring fracture apertures from a road cut exposure. It is reasonable to assume that fracture and bedding plane apertures are smaller in the subsurface. For a traditional disaggregate method, these parameters and easily collected orientation data provides all the information to construct an equivalent porous medium. Using *Snow's* (1969) method, the fractures would be approximated by an equivalent system of infinite planes. Using *Oda's* (1985) technique, an REV would be constructed to determine a permeability tensor. However, the scale dependent nature of bed-normal permeability renders a strictly defined REV useless. Also, the permeability tensor has been shown valid for a stratified porous medium only where the permeability of all strata are isotropic (*Bear 1972*).

5.2 Estimating bed-normal permeability

An "accurate" computation of bed-normal permeability requires the actual bed thickness and fracture spacing for each bed, regardless of lithologic change. As previously noted, mean values for the

domain are not accurate, and data for beds within lithologic units was not collected. Nevertheless, since that data is not available, an estimate of the least and greatest possible permeability may be made using Eq. 3.3.

Least permeability: For an estimate of a minimum value, assume that the overall bed-normal permeability is equal to or greater than the permeability of the least permeable bed (thin ls.). Then, using Eq. 3.3, the permeability of the rock mass should be greater than

$$k_{\perp} = \frac{1}{12S} \frac{cTe_f^3 e_b^3}{cTe_b^3 + Se_f^3} = \left(\frac{1}{12 \cdot 0.1} \right) \left(\frac{2 \cdot 0.05 \cdot 0.001^3 \cdot 0.0001^3}{2 \cdot 0.05 \cdot 0.0001^3 + 0.1 \cdot 0.001^3} \right) = 8.33 \times 10^{-13} \text{ m}^2. \quad (5.1)$$

In terms of conductivity (*Bear* 1972):

$$K = \frac{g}{\nu} k_{\perp} = \frac{9.81}{10^{-6}} 8.33 \times 10^{-13} \frac{86400 \text{ s}}{1 \text{ day}} \approx 0.73 \text{ m d}^{-1}, \quad (5.2)$$

or about a meter per day.

Greatest permeability To compute an upper bound of the permeability, consider a sequence of the most permeable layers. As previously assumed, let fractures in successive layers intersect at the same point on a common bedding plane with probability zero. Also, assume that a large number of layers n allows use of a limit equation (Eq. 3.3) and that the spacing in each layer is constant, so that $c = 4$.

$$k_{\perp} = \frac{1}{12S} \frac{cTe_f^3 e_b^3}{cTe_b^3 + Se_f^3} = \left(\frac{1}{12 \cdot 0.2} \right) \left(\frac{4 \cdot 0.2 \cdot 0.01^3 \cdot 0.001^3}{4 \cdot 0.2 \cdot 0.001^3 + 0.5 \cdot 0.01^3} \right) = 6.65 \times 10^{-10} \text{ m}^2. \quad (5.3)$$

In terms of conductivity:

$$K = \frac{g}{\nu} k_{\perp} = \frac{9.81}{10^{-6}} 6.65 \times 10^{-10} \frac{86400 \text{ s}}{1 \text{ day}} \approx 565 \text{ m d}^{-1}, \quad (5.4)$$

not unreasonable for a karsted limestone.

Actual conductivity values determined from well tests of Conasauga Group rocks on Oak Ridge Reservation range from $6.2 \times 10^{-3} \text{ m d}^{-1}$ (McKay *et al.* 1996) to 0.66 m d^{-1} (Wilson *et al.* 1989). Horton (1996), in a laboratory experiment using an excavated column of shale saprolite, found conductivities ranging from 5.3 to 11.8 m d^{-1} . The difference between these field and laboratory conductivities and the estimated conductivities determined in the illustrative examples could possibly be resolved by accurate fracture aperture measurements.

Chapter 6

Conclusions

The interlayer model, a new joint system model for predicting bed-normal permeability, has been introduced in this thesis. Rules describing the fracture geometry of a fractured, layered rock mass are directly incorporated into the mathematical analysis of bed-normal permeability. The interlayer model provides a specific, probability based estimate of bed-normal permeability in fractured, layered rocks. The interlayer analysis is founded on Snow's Law for permeability through a fractured rock mass cut by infinite fractures, and the principle of mass continuity of flow normal to bedding. Fractures are modeled as parallel plates, using the cubic law for intrinsic fracture permeability. Bedding planes are modeled as fictitious layers, termed *interlayers*, that conduct flow through "virtual" fractures. The aperture of virtual fractures in the interlayer is assumed equal to the bedding plane aperture. Fracture spacing in the interlayer is assumed equal to the fracture spacing in a fluid source layer. The interlayer thickness is a random variable derived from the mean length of the shortest paths from a fracture in a source layer to a fracture in a sink layer, along the bedding plane. An analytic solution, the interlayer equation, was derived for bed-normal permeability.

A finite element model computer program, `bnpflow` was written to numerically verify the interlayer equation. The interlayer model agrees with the finite element model within 0.6% for fractures

with constant spacing, and overestimates the permeability by no more than 25% for negative exponential spacing.

Some general conclusions for bed-normal permeability in fractured layered rocks can be drawn from the interlayer model:

1. The influence of fracture connectivity on bed-normal permeability can be modeled probabilistically using a fictitious interlayer.
2. Bed-normal permeability is scale dependent, and controlled by the least permeable layer or interlayer. Variability in parameters such as fracture spacing or bedding plane apertures increase the scale dependence. As more layers are added, the probability increases that a lower permeability layer or interlayer occurs.
3. Even such a simple model as brick-like fracturing produces a scale effect invalidating the equivalent porous medium concept of a representative elemental volume. Results from the interlayer model extend a conclusion drawn by *Long et al.* (1982) from numerical models of Baecher type fracture networks: a porous media equivalent for permeability may not result from a statistically homogeneous fracture network.
4. For a given number of layers and given bedding plane aperture distribution, sampling bias can result in consistently over-predicting mean fracture spacing, resulting in lower calculated permeability.
5. Standard fracture intensity measures determined by stereological techniques (e.g., *Dershowitz and Herda* 1992; *Mauldon* 1994), as used by *Oda* (1985) and *Kulatilake and Wu* (1984), are not directly applicable for determining bed-normal permeability with the interlayer equation unless the bed thickness and fracture spacing for each bed is constant. Fracture spacing must be determined explicitly for each layer. For example, sparsely fractured thin layers reduce more than what disaggregate methods would predict.

6. When the apertures of the fractures perpendicular to bedding are an order of magnitude greater than the apertures of the bedding planes, the bed-normal permeability is proportional to the bedding plane aperture and may be approximated by $k \approx ce^3/(12S\beta)$.
7. Current methods of describing and sampling fractured, stratified rock masses may not provide useful parameters for the interlayer equation. In geologic terms, formation may be composed of *units* of like lithology, which are composed of individual *beds*. The interlayer model depends on the properties of the beds, not the number and thickness of lithologic units
8. Even though a true value for permeability in a fractured layered rock mass may not be quantifiable, a useful estimate of the bed-normal permeability may be found using the interlayer equation.

List of References

List of References

- Baecher, G. B., Statistical description of rock properties, in *Energy, Resources and Excavation Technology* (Edited by F.-D. Weng and G. B. Clark), 5C1-1 – 5C1-8, 18th U. S. Symposium on Rock Mechanics, 1977.
- Baecher, G. B. and N. A. Lanney, Trace length biases in joint surveys, in *19th U. S. Symposium on Rock Mechanics* (Edited by Y. S. Kim), 56-65, U. S. National Committee for Rock Mechanics, 1978.
- Baker, A. J. and D. W. Pepper, *Finite Elements 1-2-3*, McGraw-Hill, Inc., New York, 1991.
- Barton, C. C. and E. Larsen, Fractal geometry of two-dimensional fracture networks at Yucca Mountain, Southwest Nevada, in *Proceedings of the International Symposium on Fundamentals of Rock Joints* (Edited by Bjorkliben), 77-94, International Society for Rock Mechanics, 1985.
- Barton, C. M., Analysis of joint traces, in *19th U. S. Symposium on Rock Mechanics* (Edited by Y. S. Kim), 38-42, U. S. National Committee for Rock Mechanics, 1978.
- Bear, J., *Dynamics of Fluids in Porous Media*, American Elsevier, New York, 1972.
- Becker, E. B., G. F. Carey and J. T. Oden, *Finite Elements, an Introduction*, University of Texas, Austin, Texas, 1981.
- Billaux, D., J. P. Chiles, K. Hestir and J. Long, Three-dimensional statistical modelling of a

- fractured rock mass—an example from the Fanay–Augères mine, *Int. J. Rock Mech. Min. Sci. & Geomech. Abstr.*, 26, 281–299, 1989.
- Brémaud, P., *An Introduction to Probabilistic Modeling*, Springer-Verlag, New York, 1988.
- Cox, D. R., *Renewal Theory*, Methuen & Co. LTD, London, 1962.
- Dershowitz, W. S., *Rock Joint Systems*, Ph.D. thesis, Massachusetts Institute of Technology, 1984.
- Dershowitz, W. S. and H. H. Einstein, Characterizing rock joint geometry with joint system models, *Rock Mechanics and Rock Engineering*, 21, 21–51, 1988.
- Dershowitz, W. S. and H. H. Herda, Interpretation of fracture spacing and intensity, in *Rock Mechanics* (Edited by Wawersik and Tillerson), 757–767, International Society for Rock Mechanics, 1992.
- Dershowitz, W. S. and P. LaPointe, Discrete fracture approaches for oil and gas applications, in *Rock Mechanics Models and Measurements, Challenges from Industry* (Edited by P. P. Nelson and S. E. Laubach), 19–30, North American Rock Mechanics Symposium, 1994.
- Doolin, D. M. and M. Mauldon, Anisotropy and heterogeneity in Conasauga Group rocks near Oak Ridge Reservation, Tennessee, Technical report DE-FC05-92OR22056, Department of Energy, Oak Ridge, Tennessee, 1995.
- Doolin, D. M. and M. Mauldon, Bed-normal permeability in fractured sedimentary rocks, in *Rock Mechanics, Tools and Techniques* (Edited by M. Aubertin, F. Hassani and H. Mitri), 1793–1800, 2d North American Rock Mechanics Symposium, 1996.
- Dreier, R. B., D. K. Solomon and C. M. Beaudoin, Fracture characterization in the unsaturated zone of a shallow land burial facility, in *Flow and Transport Through Unsaturated Fractured Rock*, 51–59, Washington D. C., American Geophysical Union, 1987.

- Farlow, S. J., *Partial Differential Equations for Scientists and Engineers*, Dover Publications, Inc., New York, 1992.
- Feller, W., *An Introduction to Probability Theory and its Applications, Volume 2*, John Wiley & Sons, Inc., New York, 2nd edition, 1966.
- Freeze, R. A. and J. A. Cherry, *Groundwater*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
- Gale, J., Hydraulic behavior of rock joints, in *Rock Joints* (Edited by N. Barton and O. Stephansson), 351–362, 1990.
- Gervais, F., J. P. Chiles and S. Gentier, Geostatistical analysis and hierarchical modelling of a fracture network in a stratified rock mass, in *Fractured and Jointed Rock Masses* (Edited by L. R. Meyer, C. Tsang, N. G. W. Cook and R. E. Goodman), 158–165, 1992.
- Gross, M. R., M. P. Fischer, T. Engelder and R. J. Greenfield, Factors controlling joint spacing in interbedded sedimentary rocks: Integrating numerical models with field observations from the Monterey Formation, USA, in *Fractography: fracture topography as a tool in fracture mechanics and stress analysis* (Edited by M. S. Ameen), 215–233, Geological Society Special Publication No. 92, 1995.
- Harton, A. D., *Influence of Flow Rate on Transport of Bacteriophage in a Column of Highly Weathered and Fractured Shale*, Master's thesis, University of Tennessee, Knoxville, 1996.
- Hasson, K. O. and C. S. Haase, Lithofacies and paleogeography of the conasauga group, (middle and late cambrian) in the valley and ridge province of east tennessee, *Geological Society of America Bulletin*, 100, 234–246, 1988.
- Hatcher, R. D., P. J. Lemiszki, R. B. Dreier, R. H. Ketelle, R. R. Lee, D. A. Leitzke, W. M. McMaster, J. Foreman and S. Y. Lee, Status report on the geology of the oak ridge reservation, TM-12074, ORNL, Oak Ridge, TN, 1992.
- Hsieh, P. A. and S. P. Neuman, Field determination of the three-dimensional hydraulic conduc-

- tivity tensor of anisotropic media. part 1. theory, *Water Resources Research*, 21, 1655-1665, 1985.
- Hsieh, P. A., S. P. Neuman, G. K. Stiles and E. S. Simpson, Field determination of the three-dimensional hydraulic conductivity tensor of anisotropic media. part 2. methodology and application to fractured rocks, *Water Resources Research*, 21, 1667-1676, 1985.
- Hudson, J. A. and P. R. La Pointe, Printed circuits for studying rock mass geometry, *Int. J. Rock Mech. Min. Sci. & Geomech. Abstr.*, 17, 297-301, 1980.
- Jaeger, J. C. and N. Cook, *Fundamentals of Rock Mechanics*, Methuen & Co., Ltd., London, 1969.
- Jancin, M. and J. Ewart, Prediction and testing of hydraulic parameters in interstratal karst of the Valley and Ridge Province, Pennsylvania, in *Karst Geohazards* (Edited by B. F. Beck), 125-130, A. A. Balkema, Rotterdam, 1995.
- Kulatilake, P. H. S. W. and T. H. Wu, The density of discontinuity traces in sampling windows, *Int. J. Rock Mech. Min. Sci. & Geomech. Abstr.*, 21, 345-347, 1984.
- LaPointe, P. R., *Characterization and Interpretation of Rock Mass Joint Patterns*, The Geological Society of America, Boulder, Co., 1985.
- Lemiszki, P. J., *Geology of Bethel Valley Quadrangle*, Ph.D. thesis, University of Tennessee, 1992.
- Long, J. C. S., P. Gilmour and P. A. Witherspoon, A model for steady fluid flow in random three-dimensional networks of disc-shaped fractures, *Water Resources Research*, 21, 1105-1115, 1985.
- Long, J. C. S., J. S. Remer, C. R. Wilson and P. A. Witherspoon, Porous media equivalents for networks of discontinuous fractures, *Water Resources Research*, 18, 645-658, 1982.

- Malvern, L. E., *Introduction to the Mechanics of a Continuous Medium*, Prentice-Hall, Englewood Cliffs, New Jersey, 1969.
- Marcus, H. and D. E. Evenson, Directional permeability in anisotropic media, Contribution no. 31, Water Resources Center, University of California, Berkeley, 1961.
- Mauldon, M., Intersection probabilities of impersistent joints, *Int. J. Rock Mech. Min. Sci. & Geomech. Abstr.*, 31, 107-115, 1994.
- McKay, L. D., J. A. Cherry and R. W. Gillham, Field experiments in a fractured clay till 1. hydraulic conductivity and fracture aperture, *Water Resources Research*, 29, 1149-1162, 1993.
- McKay, L. M., P. L. Stafford and L. E. Toran, Epm modeling of a field scale tritium tracer test experiment in fractured weathered shale, *Submitted to Ground Water*, 1996.
- Oda, M., Fabric tensor for discontinuous geological materials, *Soils and Foundations— Journal of the Japanese Society of Soil Mechanics and Foundation Engineering*, 22, 96-108, 1982.
- Oda, M., Permeability tensor for discontinuous rock mass, *Geotechnique*, 35, 483-495, 1985.
- Odling, N., The development of network properties in natural fracture patterns, in *Fractured and Jointed Rock Masses* (Edited by L. R. Meyer, C. Tsang, N. G. W. Cook and R. E. Goodman), 42-48, 1992.
- Piggott, A. R. and D. Elsworth, Laboratory assessment of the equivalent apertures of a rock fracture, *Geophysical Research Letters*, 20, 1387-1390, 1993.
- Priest, S. D., *Discontinuity Analysis for Rock Mechanics*, Chapman & Hall, New York, 1993.
- Pyrak-Nolte, L. J., D. D. Nolte, L. R. Myer and N. G. W. Cook, Fluid flow through single fractures, in *Rock Joints* (Edited by N. Barton and O. Stephansson), 413-422, 1990.
- Segerlind, L. J., *Applied Finite Element Analysis*, John Wiley and Sons, New York, 1984.

- Snow, D., Anisotropic permeability of fractured media, *Water Resources Research*, 5, 1273–1289, 1969.
- Snow, D. T., Rock fracture spacings, openings, and porosities, *Journal of the Soil Mechanics and Foundations Division, ASCE*, 94(SM1), 73–91, 1968.
- Terzaghi, R., Sources of error in joint surveys, *Geotechnique*, 15, 286–304, 1965.
- Weber, J. L., *Paleoenvironmental Analysis and Test of Stratigraphic Cyclicity in the Nolichucky Shale and Maynardville Limestone (Upper Cambrian) in Central East Tennessee*, Ph.D. thesis, University of Tennessee, Knoxville, 1988.
- Weinstock, R., *Calculus of Variations*, Dover Publications, Inc., Don Mills, Ontario, 1974.
- Whitaker, S., Flow in porous media i: A theoretical derivation of darcy's law, *TIPM*, 1, 3–25, 1986.
- Wilson, C. R., *An Investigation of Laminar Flow in Fractured Porous Media*, Ph.D. thesis, University of California, Berkeley, 1970.
- Wilson, G. V., J. M. Alfonsi and P. M. Jardine, Spatial variability of saturated hydraulic conductivity of the subsoil of two forested watersheds, *Soil Sci. Am. J.*, 53, 679–685, 1989.
- Witherspoon, P. A., J. S. Y. Wang, K. Iwai and J. E. Gale, Validity of the cubic law for fluid flow in a deformable rock fracture, *Ground Water*, 16, 1016–1024, 1980.
- Wylie, C. R., *Advanced Engineering Mathematics*, McGraw-Hill, New York, 5th edition, 1982.
- Zimmerman, R. W., D. Chen and N. G. W. Cook, The effect of contact area on the permeability of fractures, *Journal of Hydrology*, 139, 79–96, 1992.

Appendices

Appendix A

Mean Path Length

In the analysis presented in §2.2.2, all the flow from any particular fracture was assumed to travel to the nearest fracture in the adjacent layer. This is a bit of a conundrum, implying that a gradient does not exist in certain parts of the bedding plane, a physically unlikely occurrence. A slightly different model can be posed which accounts assures that flow exists in every part of the bedding plane. Here, the *mean path length* of all flow paths is found, instead of the mean length of the *shortest paths*. The results are found to be very similar for both cases.

Flow is assumed to occur in all segments of the bedding plane, there is no region along the bedding plane that will not have flow and equal flow is assumed in each segment. For fractures with constant spacing, this implies that exactly one source fracture in every interval of sink fractures will split its contributing flow to the sink fractures bounding that interval. For all other source layer (i) fractures, the flow from each is assumed to travel along the bedding plane to the nearest fracture in the sink layer (j).

Mean spacing S_{ij}

By determining the total number of source and sink fractures in some arbitrary length, the number of nodes per unit length λ may be calculated, whence the spacing S of nodes from the reciprocal frequency $1/\lambda$. Assuming that the fracture spacing in each layer is constant, the beds can be assumed to be composed of integer multiples m of a fracture interval, say mS_j (Fig. A.1). Note that the multiple mS_j is bounded by $m + 1$ fractures. The mean number n_{Fi} of S_i fractures in the interval mS_j is

$$n_{Fi} = \frac{mS_j}{S_i}. \quad (\text{A.1})$$

The number of flow paths in such a system is

$$n = \frac{mS_j}{S_i} + m = \frac{m(S_i + S_j)}{S_i}. \quad (\text{A.2})$$

Then, the number of flow paths per unit length is

$$\lambda = \frac{m(S_i + S_j)}{S_i} \frac{1}{mS_j}, \quad (\text{A.3})$$

$$= \frac{S_i + S_j}{S_i S_j}. \quad (\text{A.4})$$

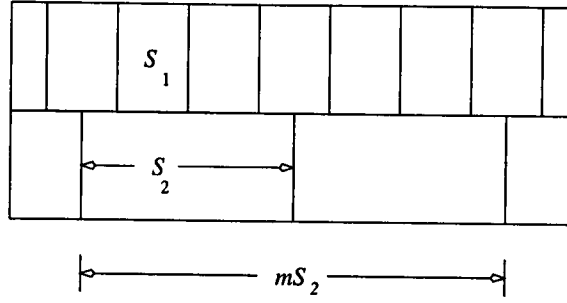


Figure A.1: Number of flow paths

Since by definition, $S = 1/\lambda$, the spacing S_{ij} of flow paths in the interlayer is

$$S_{ij} = \frac{S_i S_j}{S_i + S_j}. \quad (\text{A.5})$$

Using Fig. A.1 as an elementary example, let $S_i = 1$, $S_j = 3$, and $m = 2$. Then the mean number of S_i fractures in length mS_j is 6. The number of flow paths is 8; paths per unit length: $8/6 = 4/3$; flow path spacing equals $3/4$.

Mean thickness T_{ij} of the interlayer

Recall that the thickness of the interlayer is assumed equal to the expected value length $E(L)$ of the flow paths. $E(L)$ may be determined by averaging the summed lengths of the individual flow paths. For bed-normal permeability, flow paths are constructed to minimize the total length of the flow paths, while still allowing flow to be possible along every part of the bedding plane. The graph model presents flow as taking the shortest possible path from one layer to the next, allowing flow to split and join, ensuring flow along every segment of the interface. In this particular flow model, the mean path length is a function of the spacing of the fractures. There are two cases; the first when an even number of fractures exists in an interval, the second when an odd number exists.

Consider α fractures spaced S_i in an interval defined by S_j . From graph theory, the number of flow paths is $\alpha + 1$. The sum L of all flow paths L_i is

$$L = \sum_{i=1}^{\alpha} L_i.$$

Since $E(\sum L_i) = \sum E(L_i)$, the mean length of flow paths $E(L)$ along the interlayer is

$$E(L) = \frac{1}{\alpha + 1} \sum_{i=1}^{\alpha} E(L_i). \quad (\text{A.6})$$

Odd case: An odd number α of fractures occurs in an interval S_j , resulting in an even number $\alpha + 1$ of flow paths (Fig. A.2). The expected value of the flow path length for α odd is

$$E(L)_{\text{odd}} = \frac{1}{\alpha + 1} E(p_1 + p_2 + \cdots + p_{\alpha} + p_{\alpha+1}).$$

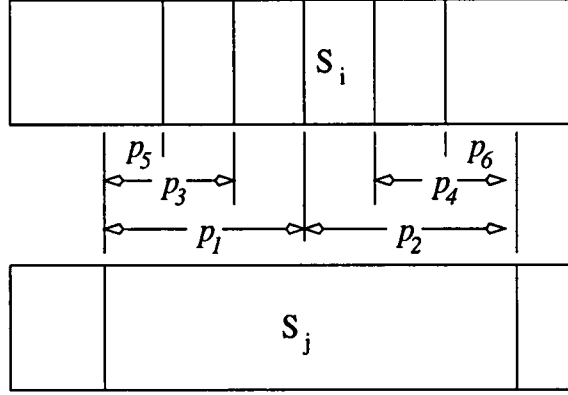


Figure A.2: Model of shortest flows paths in laminar, non-crossing flow, even number of flow paths.

Grouping flow paths from the outside in allows expressing the expected value of pairs of flow path lengths as functions of S_i and S_j :

$$E(p_1 + p_2) = \frac{2(S_j)}{2}, E(p_3 + p_4) = \frac{2(S_j - 2S_i)}{2}, \dots, E(p_\alpha, p_{\alpha+1}) = \frac{2(S_j - (\alpha - 1)S_i)}{2}. \quad (\text{A.7})$$

By Eq. A.6, summing the expected values of in Eq. A.7 yields

$$E(L)_{\text{odd}} = \frac{1}{\alpha + 1} \left\{ \left(\frac{\alpha + 1}{2} \right) S_j - (2 + 4 + \dots + \alpha - 1) S_i \right\}. \quad (\text{A.8})$$

Substituting $\left(\frac{\alpha-1}{2} \right) \left(\frac{\alpha+1}{2} \right)$ for $(2 + 4 + \dots + \alpha - 1)$ in Eq. A.8 yields

$$E(L)_{\text{odd}} = \frac{1}{\alpha + 1} \left\{ \frac{\alpha + 1}{2} S_j - \left(\frac{\alpha - 1}{2} \right) \left(\frac{\alpha + 1}{2} \right) S_i \right\},$$

which may be algebraically simplified to

$$E(L)_{\text{odd}} = \frac{S_j}{2} + \frac{S_i}{4} - \frac{\alpha S_i}{4}. \quad (\text{A.9})$$

When S_i and S_j are integer multiples (i.e. $S_j = \alpha S_i, \alpha = 1, 2, \dots$), Eq. A.9 reduces to $(S_i + S_j)/4$.

Even case: An even number α of fractures occurs in an interval S_j , resulting in an odd number $\alpha + 1$ of flow paths (Fig. A.3). The expected value of the flow path length for α even is

$$E(L)_{\text{even}} = \frac{1}{\alpha + 1} E(p_1 + p_2 + \dots + p_\alpha + p_{\alpha+1}).$$

Grouping flow paths from the outside in, and summing the flow paths term by term for any odd integer α yields

$$E(L)_{\text{even}} = \frac{1}{\alpha + 1} \left\{ \frac{\alpha}{2} S_j - (1 + 3 + \dots + \alpha - 1) S_i + \left(\frac{\alpha}{2} \right) S_i + \delta \right\}, \quad (\text{A.10})$$

where δ is length of a leftover path that cannot be paired; defined as $\delta < S_i/2$. Since fractures in each layer are located independently of fractures in the adjacent layers, the position of the fracture

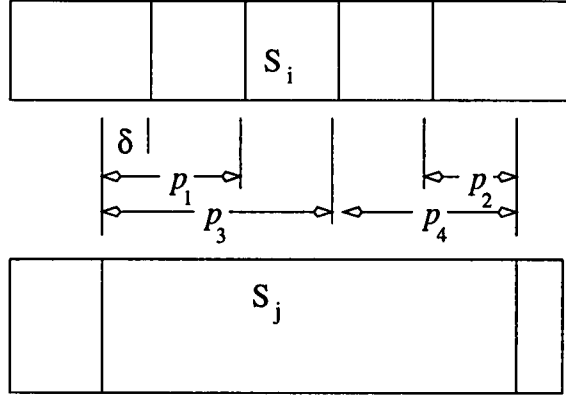


Figure A.3: Model of shortest flows paths in laminar, non-crossing flow, odd number of flow paths.

contributing δ is uniformly distributed over an interval of length $S_i/2$

$$f(\delta) = \begin{cases} 2/S_i & 0 \leq \delta \leq S_i/2, \\ 0 & \text{otherwise.} \end{cases} \quad (\text{A.11})$$

The mean path length of the remaining length δ :

$$E(\delta) = \int_0^{S_i/2} \delta f(\delta) d\delta = \frac{S_i}{4}. \quad (\text{A.12})$$

Substituting $(\alpha/2)^2$ for $(1 + 3 + \dots + \alpha - 1)$ and $\delta = S_i/4$ in Eq. A.10 yields

$$\begin{aligned} E(L)_{\text{even}} &= \frac{1}{\alpha + 1} \left\{ \frac{\alpha}{2} S_j - \left(\frac{\alpha}{2} \right)^2 S_i + \left(\frac{\alpha}{2} \right) S_i + \frac{S_i}{4} \right\}. \\ (\text{Simplifying}) &= \frac{-S_i \alpha^2 + 2S_j \alpha + 2S_i \alpha + S_i}{4(\alpha + 1)}. \end{aligned} \quad (\text{A.13})$$

Sadly, Eq. A.13 factors no further, but when S_i and S_j are integer multiples (i.e., $S_j = \alpha S_i, \alpha = 1, 2, \dots$), Eq. A.13 reduces to $E(L) = (S_i + S_j)/4$, just as did the odd case of Eq. A.9.

Equations A.9 and A.13 are unwieldy, impractical to use, and it is not immediately apparent how to determine the number of intervals containing even and odd flow paths for arbitrary spacings S_i and S_j . Buffon's needle problem inspires a probabilistic solution to this potential impasse. Buffon considered the problem of computing the value of π from the probability that a needle of length ℓ dropped randomly on a field of straight parallel lines separated by the distance a will intersect one of the lines (Brémaud 1988). If Buffon's needle is considered in one dimension, the relative probabilities of even and odd flow path cases can be computed. Let the ratio of fracture spacings $S_j/S_i = a + b, S_j > S_i$; where a and b are the respective integer and decimal parts of the ratio. Let the field lines have spacing S_i and the needle length S_j . The number a of S_i fractures ("lines") in an interval of S_j (covered by the "needle") is either a , with probability $1 - b$, or $a + 1$, with probability b . The expected value of the flow path lengths for an interlayer is the sum of the expected values

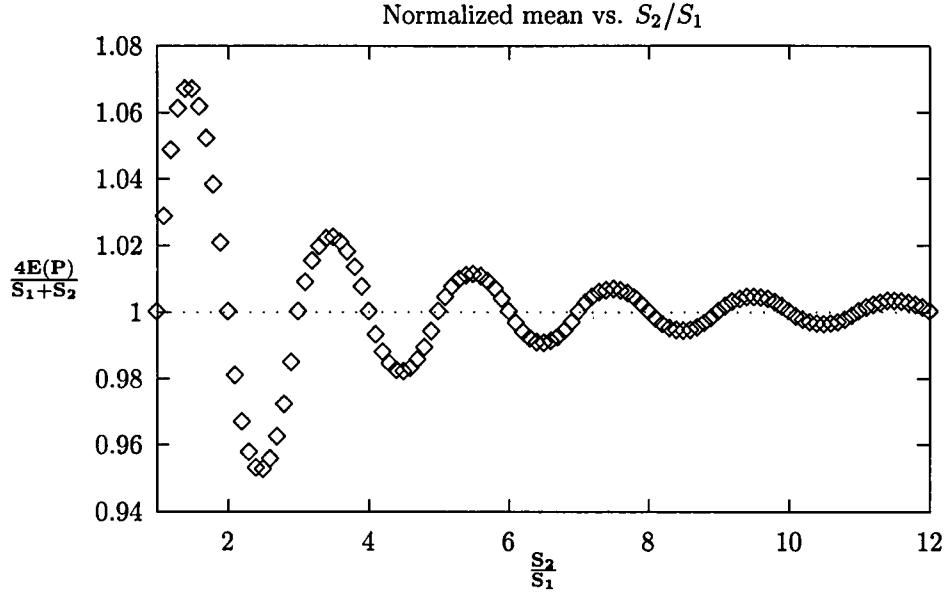


Figure A.4: Mean length of flow paths as a function of the S_j/S_i spacing ratio.

of intervals with α even or α odd. There are two cases:

$$E(L) = \begin{cases} E(L)_{\text{even}}(1-b) + E(L)_{\text{odd}}(b), & \alpha \text{ is even;} \\ E(L)_{\text{odd}}(1-b) + E(L)_{\text{even}}(b), & \alpha \text{ is odd.} \end{cases} \quad (\text{A.14})$$

If $S_j/S_i < 1$, then the ratio $S_i/S_j = a + b$ is used. The expected values of flow path lengths, normalized against integer values, are plotted as a function of the spacing ratio S_j/S_i in Fig. A.4. The plot shows that Eq. A.14 is nowhere greater than 7% from $(S_i + S_j)/4$. For practical purposes, $T_{ij} \equiv (S_i + S_j)/4$. Using $T_{ij} \equiv (S_i + S_j)/4$ for flow path length, and substituting Eq. A.5 for flow path spacing, the interlayer correction term ΣI is approximated by

$$\begin{aligned} \Sigma I &= 12 \sum_{i=1}^{n-1} \frac{S_i + S_j}{4} \frac{\frac{S_i S_j}{S_i + S_j}}{4e_{ij}^3} \\ &= 12 \sum_{i=1}^{n-1} \frac{S_i S_j}{4e_{ij}^3}, \end{aligned} \quad (\text{A.15})$$

a result identical to Eq. 2.12.

Appendix B

Construction of permeability tensor

The following develops a permeability tensor according to *Snow's*(1969) method. Snow's tensor is then modified according to the interlayer concept, under the assumption of a large number of layers. The method is illustrated with an example.

Snow presents the following equation for a permeability tensor as Eq. 24 (*Snow* 1969).

$$k_{ij} = \frac{2}{3} \sum \frac{b^3}{|n_i D_i|} (\delta_{ij} - m_{ij}). \quad (\text{B.1})$$

We are told to take the summation “over all the members of a joint set intersected by sampling line D_i , each of which has a particular aperture $2b$, uniform over its extensive surface and constant in orientation n_i .” In other words, sum over the apertures of fracture intersected by a scan line of length D . The denominator $|n_i D_i|$ is an absolute value of the inner product formed by the sampling line D_i and the normal to the fracture plane n_i . It serves to weight the permeability by orientation, i.e., that Terzaghi correction (*Terzaghi* 1965). Note that b is the *half aperture*. In more modern notation, $\frac{2}{3}b^3 = \frac{2}{3}(\frac{1}{2}e)^3 = \frac{2}{3}(\frac{1}{8}e^3) = \frac{1}{12}e^3$. The kronecker delta is denoted by δ_{ij} . The matrix m_{ij} is equal to the dyad $n_i n_j$ representing the matrix of direction cosines formed by projecting the components of the fracture normal onto the coordinate axes.

$$n_i n_j = m_{ij} = \begin{bmatrix} \cos^2 \theta & \cos \theta \sin \theta \\ \sin \theta \cos \theta & \sin^2 \theta \end{bmatrix}. \quad (\text{B.2})$$

Fracture set	Orientation (of normal)	Spacing S_i (meters)	e_f (meters)	e_b (meters)
Set 1	90	1	0.0001	0.0001
Set 2	0	2	0.0001	0.0001

Table B.1: Synthetic fracture data for two-dimensional, orthogonal system illustrated in Fig. B.1. The orientation of the normal vectors n_i are with respect to the x axis.

To calculate a permeability tensor for the rock mass illustrated in Fig. B.1, we make the following

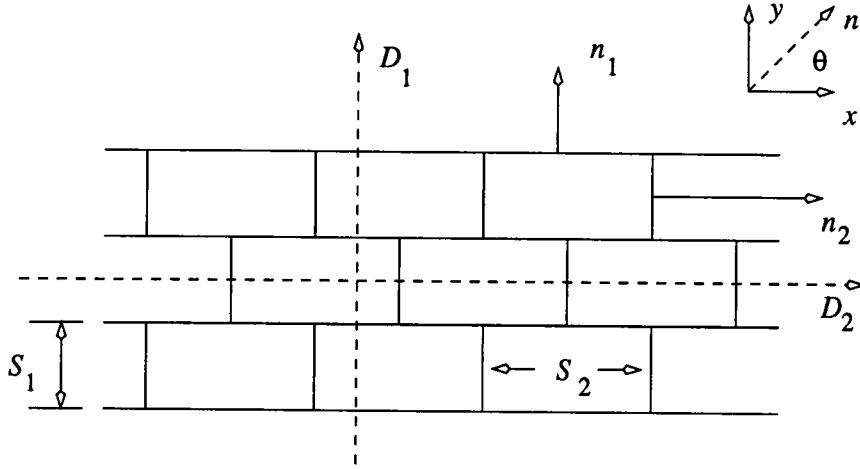


Figure B.1: Synthetic rock mass geometry, resembling bricks. D_i are sampling (scan) lines; n_i are normal vectors to the fracture sets.

substitution in Eq. B.1:

$$\frac{e^3}{12S} \equiv \frac{2}{3} \sum \frac{b^3}{|n_i D_i|}. \quad (\text{B.3})$$

We don't need any weighting in this example because the sampling (scan) lines D_1 and D_2 are parallel to the normal vectors $n_i, i = 1, 2, \dots$. The direction cosines m_{ij} are easy to calculate for this example: $\cos 90 = 0, \sin 90 = 1, \cos 0 = 1, \sin 0 = 0$. For set 1:

$$\begin{aligned} k_{ij}^{(\text{set1})} &= \frac{e_1^3}{12S_1} (\delta_{ij} - m_{ij}) \\ &= \frac{0.0001^3}{12 \cdot 1} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right) \\ &= \frac{0.0001^3}{12 \cdot 1} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 8.33 \times 10^{-14} & 0 \\ 0 & 0 \end{bmatrix}. \end{aligned} \quad (\text{B.4})$$

For set 2:

$$\begin{aligned} k_{ij}^{(\text{set2})} &= \frac{e_2^3}{12S_2} (\delta_{ij} - m_{ij}) \\ &= \frac{0.0001^3}{12 \cdot 2} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \right) \end{aligned}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 4.17 \times 10^{-14} \end{bmatrix}. \quad (\text{B.5})$$

By assumption,

$$k_{ij} = k_{ij}^{(\text{set1})} + k_{ij}^{(\text{set2})} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix} = \begin{bmatrix} 8.33 \times 10^{-14} & 0 \\ 0 & 4.17 \times 10^{-14} \end{bmatrix}. \quad (\text{B.6})$$

As would be expected, the permeability in the y direction (k_y) is half the value of the permeability in the x direction (k_x). Note that k_y would actually be less than the computed value due to the interlayer effect. Snow's method assumes infinitely extensive jointing and cannot account for permeability reduced by fracture connectivity relationships. By analysis presented in chapter 3, for this rock mass geometry, as the number of layers $n \rightarrow \infty$, the bed-normal permeability (k_y) converges to 4/5 of the value computed in Eq. B.5:

$$= \begin{bmatrix} 8.33 \times 10^{-14} & 0 \\ 0 & 3.33 \times 10^{-14} \end{bmatrix}. \quad (\text{B.7})$$

Snow's work with fractured, jointed rock masses was pioneering. However, terminology and notation has advanced over the last 30 years to the point that his work may be difficult to follow from a more modern perspective.

Appendix C

Finite element code

The finite element program `bnpflow` was written in ANSI C, and should compile with the public domain GNU C compiler (`gcc`), on any supported architecture (DOS, unix, etc.). The program produces a `matlab` input file, then calls `matlab` to solve for the head distribution. The permeability is computed from head data produced by `matlab`.

```
#include<stdio.h>
#include "dlist.h"

/* To make sure any changed defs get propagated through
   all functions, 'make clean', then compile from scratch.
   */
#define LINE_BUF 100
#define LBC 50.
#define RBC 0.0
#define C 2
#define POISSON
#define GRAPHICS
#define x 0
#define y 1
#define l 0
#define r 1
#define M 0 /* Mean. */
#define S 1 /* Standard deviation. */

typedef struct fracture_tag {
    double    aperture;
    double    center;
    double    length;      /* Bed thickness. */
    double    lcoord[2];   /* Cumulative coordinates. */
    double    rcoord[2];
```

```

double      lhead;
double      rhead;
double      flow;
int         element;      /* Element number. */
int         leftnode;
int         rightnode;
int         layer;
}           FRACTURE;

```

/ This is a rewrite of the above structure to try and get the io a little more organized. The 'Beds' struct was written quickly and uses arrays. This one will be part of a linked list. */*

```

typedef struct bed_tag {
double      width;
int         layer_number;
double      xcoord[2];
double      thickness;
double      mean_spacing;
double      mean_cubed_fracap;
double      interapers;
double      spacing[2];
double      fracaps[2];
}           BEDS;

```

/ Tex stuff. */*

```

typedef struct tex_title {
char        title[180];
char        author[180];
char        date[30];
}           TITLE;

```

```

typedef struct stats_tag {
int seed;
int layers;
int numnodes;
double width;
double parameter;
double spacing[100];
double sumI;
double sumB;
double sumT;
double K_pred;
double K_calc;
double domain_width;
double domain_length;

```

```

char outfile[100];
} STATS;

Dlist      merge(Dlist, Dlist);
int        fracture_comp(void *, void *);
void       assembly(Dlist, Dlist);
double     value(double, double);
Dlist      listbeds();
FRACTURE   *fracload(BEDS *, double);
double     exponential(double, double);
double     lognormal(double, double);
Dlist      makefracturelist(Dlist, STATS *);
Dlist      makeinterlayerlist(Dlist);
Dlist      makebedlist(STATS *);
double     value(double, double);
void       predictedk(Dlist, STATS *);

/*
 * frac6.c
 *
 * David M. Doolin
 *
 * 31 July, 1996. Attempting a complete rewrite of the input scheme to get
 * distributions of fracture parameters as input. So far, I am having to
 * extensively comment the code to figure out what I did in the past. Also
 * removed some redundant or commented out code.
 */

#include <stdio.h>
#include <math.h>
#include "fracture.h"

#define FRAC6

main(int argc, char **argv)
{
    Dlist      bedlist;      /* List of beds with data for each bed. */
    Dlist      jointssystem; /* Tree of beds with lists of fractures. */
    Dlist      interlayerlist; /* Tree with lists of interlayers. */
    STATS      *stats;      /* Data for fracture statistics. */
    double     width;
    int        seed;

    if (argc < 3) {
        /* Trap bad command line. */
        fprintf(stderr, "Usage: frac6 <width> <seed>\n");
        printf("\n");
        exit(-1);
    }
    width = atof(argv[1]); /* Process command line arguments. */

```

```

seed = atoi(argv[2]);
srandom(seed);                                /* Initialize the random number generator. */

/* Initialize the statistics structure. */
stats = (STATS *) malloc(sizeof(STATS));
stats->seed = seed;
stats->width = width;
strcpy(stats->outfile, "runs");

bedlist = makebedlist(stats);/* Read in data from frac6.c. */
jointssystem = makefracturelist(bedlist, stats);
interlayerlist = makeinterlayerlist(jointssystem);
fracturestatistics(jointssystem, bedlist, stats);
residuals(jointssystem, stats->numnodes);      /* Start the matlab script. */
assembly(interlayerlist, bedlist);
head_distribution(jointssystem, bedlist);
flow(jointssystem, bedlist);
predictedk(bedlist, stats);
output(stats);
#ifdef GRAPHICS
    gnuout(jointssystem, bedlist);/* Output to gnuplot for graphics. */
    system("gnuplot < setup.gnu");
#endif GRAPHICS
    return 1;
}
/* End of main(). * */
/*
 * io.c
 *
 * David M. Doolin
 *
 * March 9, 1996.
 *
 * Input/output routines for for the fracture mesh generator.
 */

#include <stdio.h>
#include <math.h>
#include "fracture.h"

/*
 * The control procedure takes a pointer to the STATS structure and appears
 * to use it only to provide the number of layers in the system. The
 * procedure returns a double linked list where the value of each list points
 * to the head of another doubly linked list containing values for each bed
 * in the domain. Read in the frac6.dat file that has bedding parameters,
 * etc. in it. I need to rewrite this block of code to support comments in
 * the data file. Also, there is no provision for input range checking.
 */

```

```

Dlist
makebedlist(STATS * stats)
{
    FILE          *ifp;
    BEDS          *bedding;
    /*
     * M = mean, S = standard deviation, BT = bed thickness, FA = fracture
     * aperture, IA = interlayer aperture, FS = fracture spacing.
     */
    double        MBT, SBT, MFA, SFA, MIA, SIA, MFS, SFS;
    double        xcoord;
    int           checkval;
    char          line[LINE_BUF];
    Dlist         bed_list, tmp;
    int           layer;

    layer = 0;
    xcoord = 0;
    bed_list = make_dl();

    ifp = fopen("frac6.dat", "r");

    /*
     * This loop reads in data for each bed from the 'control.dat' file. There
     * is no provision for fracture location in the file or in the input
     * routine. Fracture locations are dealt with in main().
     */
    while (fgets(line, LINE_BUF, ifp)) {
        find_comment(line);
        bedding = (BEDS *) malloc(sizeof(BEDS));
        checkval = sscanf(line, "%lf%lf%lf%lf%lf%lf%lf%lf",
                           &MBT, &SBT, &MFA, &SFA, &MFS, &SFS, &MIA, &SIA);

        /*
         * Get values for a BED structure (bedding) that will be inserted into
         * a list of layers.
         */
        if (checkval == 8) {
            bedding->interapers = lognormal(MIA, SIA);
        } else if (checkval == 7) {
            bedding->interapers = 0;
        } else {
            /* Exits when input file contains comments. */
            fprintf(stderr, "Problem with frac6.dat file\n");
            exit(-1);
        }
        /* Close checkval for scanf. */
        bedding->thickness = lognormal(MBT, SBT);
        bedding->fracaps[0] = MFA;
        bedding->fracaps[1] = SFA;
        bedding->spacing[0] = MFS;
        bedding->spacing[1] = SFS;
    }
}

```

```

    bedding->xcoord[l] = xcoord;
    xcoord += bedding->thickness;
    bedding->xcoord[r] = xcoord;
    layer++;
    bedding->layer_number = layer;
    bedding->width = stats->width;
    /* Insert the node into a list of layers, continue. */
    dl_insert_b(bed_list, (void *) bedding);
}
fclose(ifp); /* Close the control file. */
stats->layers = layer; /* Only place in this routine where stats is
                        * used in this function. */
return bed_list;
} /* Close makebedlist(). */

void
gnuout(Dlist bedlist, Dlist layerlist)
{
    FRACTURE      *fracture;
    Dlist         tmp, tmp1, head;
    BEDS          *bed;
    FILE          *gnuftp;

    gnuftp = fopen("mesh.dat", "w");

    /* Traverse lists to write out all of the fractures. */
    for (tmp = bedlist->flink; tmp != bedlist; tmp = tmp->flink) {
        head = (Dlist) tmp->val;
        dl_traverse(tmp1, head) {
            fracture = (FRACTURE *) tmp1->val;
            /* Previous; vertical beds.
            fprintf(gnuftp, "%.2lf %.2lf\n", fracture->lcoord[x], fracture->lcoord[y]);
            fprintf(gnuftp, "%.2lf %.2lf;\n\n", fracture->rcoord[x], fracture->rcoord[y]);
            */
            fprintf(gnuftp, "%.2lf %.2lf\n", fracture->rcoord[y], fracture->rcoord[x]);
            fprintf(gnuftp, "%.2lf %.2lf;\n\n", fracture->lcoord[y], fracture->lcoord[x]);

        }
    }
    /*
    * This traverse draws in the bedding planes (vertically).
    */
    for (tmp = layerlist->flink; tmp != layerlist->blink; tmp = tmp->flink) {
        bed = (BEDS *) tmp->val;
        /* Previous; vertical beds.
        fprintf(gnuftp, "%.2lf 0.0\n", bed->xcoord[r]);
        fprintf(gnuftp, "%.2lf %.2lf;\n\n", bed->xcoord[r], bed->width);
        */
    }
}

```

```

/* Horizontal beds. */
    fprintf(gnufp, "0.0 %.21f\n", bed->xcoord[r]);
    fprintf(gnufp, "%.21f %.21f;\n\n", bed->width, bed->xcoord[r]);

}
fclose(gnufp);
}

void
tecout(Dlist bedlist, Dlist layerlist)
{
    FRACTURE      *fracture;
    Dlist          tmp, tmp1, head;
    FILE           *tecfp;

    tecfp = fopen("tecmesh.dat", "w");

    /* Traverse lists to write out all of the fractures. */
    for (tmp = bedlist->flink; tmp != bedlist; tmp = tmp->flink) {
        head = (Dlist) tmp->val;
        dl_traverse(tmp1, head) {
            fracture = (FRACTURE *) tmp1->val;
            fprintf(tecfp, "%.41f %.41f %.41f\n", fracture->lcoord[x], fracture->lcoord[y], fracture->lhead);
            fprintf(tecfp, "%.41f %.41f %.41f\n", fracture->rcoord[x], fracture->rcoord[y], fracture->rhead);
        }
    }
    fclose(tecfp);
}

/*
 * This routine sets up the Dirichlet boundary conditions, writing them out
 * to a matlab script file. It is currently segfaulting.
 */
void
residuals(Dlist bedlist, int nnodes)
{
    FILE           *matfp;
    Dlist          leftside, rightside, tmp;
    FRACTURE       *fracture;

    matfp = fopen("fracture.m", "w");
    fprintf(matfp, "residuals = zeros(1,%d);\n", nnodes);
    leftside = (Dlist) bedlist->flink->val;
    dl_traverse(tmp, leftside) {
        fracture = (FRACTURE *) tmp->val;
        fprintf(matfp, "residuals(%d) = %1f;\n", fracture->leftnode, LBC);
    }
}

```

```

    rightside = (Dlist) bedlist->blink->val;
    dl_traverse(tmp, rightside) {
        fracture = (FRACTURE *) tmp->val;
        fprintf(matfp, "residuals(%d) = %lf;\n", fracture->rightnode, RBC);
    }
    fprintf(matfp, "COND = sparse(%d,%d);\n", nnodes, nnodes);
    fclose(matfp);
}

```

void

head_distribution(Dlist bedlist, Dlist layerlist, STATS * stats)

```

{
    BEDS          *bedding;
    FILE          *potentfp, *matfp;
    double        total_thickness;
    double        potential;
    Dlist          sentinel, node, tmp;
    FRACTURE      *fracture;
    /*
     * #undef HUH #ifdef HUH matfp = fopen("fracture.m","a"); fprintf(matfp,
     * "for i = 2 : 2 : 14\n"); fprintf(matfp, "diff( (i+1)/2 ) = head (i-1) -
     * head(i);\n"); fprintf(matfp, "end\n"); fclose(matfp); #endif
     */

    system("runmatlab");
    potentfp = fopen("head.txt", "r");

    /*
     * Get the head at each end of the fractures. This code is going to mess
     * me up later when I decide I want to calculate the flow in all of the
     * interlayers.
     */
    for (tmp = bedlist->flink; tmp != bedlist; tmp = tmp->flink) {
        sentinel = (Dlist) tmp->val;
        dl_traverse(node, sentinel) {
            fracture = (FRACTURE *) node->val;
            fscanf(potentfp, "%le", &potential);
            fracture->lhead = potential;
            fscanf(potentfp, "%le", &potential);
            fracture->rhead = potential;
        }
    }
    fclose(potentfp);

    /* Calculate the total thickness of the layers. */
    dl_traverse(tmp, layerlist) {
        bedding = (BEDS *) tmp->val;
        total_thickness += bedding->thickness;
    }
}

```

```

    }
}

```

```

void
flow(Dlist bedlist, Dlist layerlist, STATS * stats)
{
    FILE          *statfp;
    double         perm, total_thickness, average_flow, gradient;
    Dlist          sentinel, node, tmp;
    FRACTURE       *fracture;

    double         flow, total_flow;
    flow = 0;
    total_flow = 0;

    statfp = fopen("statistics", "a");

    /* Calculate the flow through each of the fractures. */
    /* This just sets up the headers. */
    fprintf(statfp, "Fracture\tLeft head\tRight head\tFlow:\n");

    for (tmp = bedlist->flink; tmp != bedlist; tmp = tmp->flink) {
        sentinel = (Dlist) tmp->val;
        dl_traverse(node, sentinel) {
            fracture = (FRACTURE *) node->val;
            perm = pow(fracture->aperture, 3) / 12;
            gradient = (fracture->lhead - fracture->rhead) / fracture->length;
            fracture->flow = perm * gradient;
            flow += fracture->flow;
            fprintf(statfp, "\t%d\t%lf\t%lf\t%lf\n", fracture->element, fracture->lhead, \
                    fracture->rhead, fracture->flow);
        }
        fprintf(statfp, "Bulk flow through layer %d: %lf\n", fracture->layer, flow);
        total_flow += flow;
        flow = 0;
    }

    /*
     * Average the flow through each layer.
     */
    average_flow = total_flow / (double) stats->layers;
    fprintf(statfp, "Average bulk flow = %lf\n", average_flow);
    /*
     * Compute the numermerically estimated permeability..
     */
    stats->K_calc = (total_thickness * average_flow) / (stats->width * (LBC - RBC));
    fclose(statfp);
}

```

```

}

void
output(STATS * stats)
{
    FILE          *ofp;
    char          outfile[81]; /* Output file name: "runs." */
    int           i;

    strcpy(outfile, stats->outfile);
    ofp = fopen(outfile, "a");
    fprintf(ofp, "%.11f %d ", stats->width, stats->seed);
    for (i = 0; i < stats->layers; i++) {
        fprintf(ofp, "%.31f ", stats->spacing[i]);
    }
    fprintf(ofp, "%e %e ", stats->K_pred, stats->K_calc);
    fprintf(ofp, "%.11f\n", 100 * (stats->K_pred - stats->K_calc) / stats->K_pred);
    fprintf(stdout, "%e %e ", stats->K_pred, stats->K_calc);
    fprintf(stdout, "%.11f\n", 100 * (stats->K_pred - stats->K_calc) / stats->K_pred);

    fclose(ofp);
}
/*
 * initialize.c
 *
 * Sept. 27, 1996.
 *
 * File contains numerous routines that were factored out of the main()
 * function
 * in frac6.c. That code was getting to big to handle.
 *
 */

#include<stdio.h>
#include "fracture.h"

/*
 * This function needs to be fully commented.
 */

Dlist
makefracturelist(Dlist bed_list, STATS * stats)
{
    /* Scope for initializing fracture data. */
    Dlist tmp; /* Increment nodes. */
    Dlist list_of_fractures; /* List of fractures for each bed. */
    Dlist joint_system; /* Tree containing one list of fractures for
                        * each bed. */
    BEDS *bedding;

```

```

    FRACTURE      *fracture;
    double        ypos = 0; /* Position of fracture. */
    list_of_fractures = make_dl(); /* list of fractures in a bed. */
    joint_system = make_dl(); /* rockmass is a list of initialized beds
                                * containing fractures. */
    for (tmp = bed_list->flink; tmp != bed_list; tmp = tmp->flink) {
        bedding = (BEDS *) tmp->val;
        /*
         * Accumulate the position of the y coordinates of the fractures. Don't
         * want fractures outside of the domain.
         */
        if(!(bedding->layer_number%2))
#ifdef CONSTANT
            ypos = ypos + bedding->spacing[0]/2;
#endif
#ifdef POISSON
            ypos = ypos + exponential(bedding->spacing[0], .001);
#endif
        while (ypos <= (bedding->width)) {

            /*
             * fracload returns a pointer to a FRACTURE structure that is
             * initialized from data in the bedding structure (read from
             * control()) and the random variable 'ypos.'
             */
            fracture = fracload(bedding, ypos);
            /*
             * Insert the *fracture (pointer to FRACTURE) as the value ('val')
             * at the back of the list.
             */
            dl_insert_b(list_of_fractures, (void *) fracture);
#ifdef CONSTANT
            ypos = ypos + lognormal(bedding->spacing[0], bedding->spacing[1]);
#endif
#ifdef POISSON
            ypos = ypos + exponential(bedding->spacing[0], .001);
#endif
        }
        dl_insert_b(joint_system, (void *) list_of_fractures);
        ypos = 0; /* 'ypos' reinitialized. */
        /* Start a new list for the next bedding plane. */
        list_of_fractures = make_dl();
    } /* Close for() {loop}. */
    return joint_system;
} /* Close fracturesystem(). */

/*
 * Not completely sure exactly how this works. Evidently, each fracture

```

```

* makes a call here to load up the parameters for that particular fracture.
* Returns a pointer to a FRACTURE structure for each individual fracture.
*/
FRACTURE
fracload(BEDS * bedding, double ypos)
{
    FRACTURE      *fracture;
    static int     element_number = 1, node_number = 1;
    fracture = (FRACTURE *) malloc(sizeof(FRACTURE));
    fracture->layer = bedding->layer_number;
    fracture->lcoord[y] = ypos; /* ypos an RV. */
    fracture->lcoord[x] = bedding->xcoord[l];
    fracture->rcoord[y] = ypos; /* ypos an RV. */
    fracture->rcoord[x] = bedding->xcoord[r];
    fracture->element = element_number;
    element_number++;
    fracture->leftnode = node_number;
    fracture->rightnode = ++node_number;
    node_number++;
    fracture->length = bedding->thickness;
    fracture->aperture = lognormal(bedding->fracaps[0], bedding->fracaps[1]);
    return fracture;
} /* Close fracload() */

/*
* Combine the two dlists into one sorted dlist for each bedding plane.
* First, some experiments to get the pointers lined up and operating
* correctly. Use the first two nodes in the main bedding list. The code
* works as written. What need to be done is to extend it so that it will
* march down the bedding plane list. After that, send each pair to get
* merged, and then relinked to the bedding plane list.
*/
Dlist
makeinterlayerlist(Dlist joint_system)
{
    Dlist          inter_layer_list, tmp, headl, headr, head1, head2, headi;
    inter_layer_list = make_dl(); /* Create an interlayer list. This is
                                   * probably what I want a function here to
                                   * return. */

    /*
    * First have to insert fractures in the left end layer of the domain into
    * the interlayer list.
    */
    dl_insert_b(inter_layer_list, (void *) joint_system->flink->val);
    /* Now insert all of the middle fractures. */
    for (tmp = joint_system->flink; tmp != joint_system->blink; tmp = tmp->flink) {
        headl = (Dlist) tmp->val;

```

```

    headr = (Dlist) tmp->flink->val;
    /*
     * Each sublist must be copied because the merge routine frees the two
     * lists that are passed in, but I need to merge the lists twice: once
     * to the left, then to the right.
     */
    head1 = dlist_copy(headl);
    head2 = dlist_copy(headr);
    /* Now merge the two lists. */
    headi = merge(head1, head2);
    dl_insert_b(inter_layer_list, (void *) headi);
}
/*
 * Now stick the rightmost layer fractures in the interlayer list.
 */
dl_insert_b(inter_layer_list, (void *) headr);
return inter_layer_list;
}                                     /* Close makeinterlayerlist(). */

/*
 * This block prints out the fracture center locations for every list on
 * the bedding plane list, and sums the total. First loop traverses the
 * bedding plane list. The fractures located in a list pointed to by
 * 'val' on each node of the bedding plane list.
 * This routine needs to be rewritten to provide a table of complete
 * element data: x and y coordinates of the nodes, node numbers,
 * apertures, spacings, lengths etc. Also should be stuck in the stats
 * code and have stats passed in.
 */

/*
 * I have no real clue what this routine is used for, but I need to find out
 * fast!
 */
int
fracture_comp(void *v1, void *v2)
{
    /* Compares two elements, and returns -1, 0, or 1, like strcmp */
    int value;
    FRACTURE *frac1, *frac2;
    frac1 = (FRACTURE *) v1;
    frac2 = (FRACTURE *) v2;
    if (frac1->lcoord[y] <= frac2->lcoord[y])
        value = -1;
    else
        value = 1;
    return value;
}                                     /* Close fracture_comp(). */

```

```

/*
 * stats6.c
 *
 * David M. Doolin
 *
 * 12 March, 1996
 *
 * Statistics and analytic solutions fracture modeling.
 */
#include <stdio.h>
#include <math.h>
#include "fracture.h"

/*
 * fracturestatistics() prints out the fracture center locations for every
 * list on the bedding plane list, and sums the total. First loop traverses
 * the bedding plane list. The fractures located in a list pointed to by
 * 'val' on each node of the bedding plane list. This routine needs to be
 * rewritten to provide a table of complete element data: x and y coordinates
 * of the nodes, node numbers, apertures, spacings, lengths etc.
 */
int
fracturestatistics(Dlist joint_system, Dlist layerlist, STATS * stats)
{
    FILE          *statfp;
    int            numfracs, totalfracs;
    double         sumcubedapertures = 0;
    Dlist          spacing;
    Dlist          tmp, tmp1, head;
    BEDS           *layer;
    FRACTURE       *fracture;
    statfp = fopen("statistics", "w");

    totalfracs = 0;
    numfracs = 0;
    fprintf(statfp, "Element Left\tRight\tLeft coord\t"
            "Right coord\tLayer\tLength\tAperture\n");
    fprintf(statfp, "number\tnode\tnode\n");
    /*
     * This code prints out statistics on individual fractures located in each
     * layer. The function predictedk() depends on this block to calculate
     * overall stats; any change here will need to be followed up with changes
     * below to make sure that the code runs.
     */
    for (tmp = joint_system->flink; tmp != joint_system; tmp = tmp->flink) {
        head = (Dlist) tmp->val; /* Assign val for 2d traverse. */
        dl_traverse(tmp1, head) { /* Second traverse, for each layer. */
            fracture = (FRACTURE *) tmp1->val;

```

```

        /* Start an output list of all the fracture parameters. */
        fprintf(statfp, "%d\t", fracture->element);
        fprintf(statfp, "%d\t%d\t", fracture->leftnode, fracture->rightnode);
        fprintf(statfp, "(%.21f,%.21f)\t", fracture->lcoord[x], fracture->lcoord[y]);
        fprintf(statfp, "(%.21f,%.21f)\t", fracture->rcoord[x], fracture->rcoord[y]);
        fprintf(statfp, "%d\t", fracture->layer);
        fprintf(statfp, "%.21f\t", fracture->rcoord[x] - fracture->lcoord[x]);
        fprintf(statfp, "%.21f\n", fracture->aperture);
        sumcubedapertures += pow(fracture->aperture, 3);
        numfracs++;
    }
    fprintf(statfp, "Number of fractures in the layer: %d\n\n", numfracs);
    totalfracs = totalfracs + numfracs;
    /*
     * Store some fracture statistics data such as spacing, layer numbers,
     * etc. for later use calculating the analytic solution for a
     * particular geometry.
     */
    layer = (BEDS *) layerlist->flink->val;
    layer->mean_spacing = layer->width / numfracs;
    layer->mean_cubed_fracap = sumcubedapertures / numfracs;
    /*
     * The layer number is pulled from the _last_ fracture read from the
     * whole list of fracture in a layer. If the code above is changed,
     * this will probably have to change also.
     */
    numfracs = 0;
    sumcubedapertures = 0;
    layerlist = layerlist->flink;
}
layerlist = layerlist->flink;
stats->numnodes = totalfracs * 2;
fprintf(statfp, "The total number of nodes is %d\n\n", stats->numnodes);
fclose(statfp);
}
/* Close fracturestatistics(). */

/*
 * Print some fracture spacing data. Calculate the analytic, predicted
 * solution.
 */
void
predictedk(Dlist bed_list, STATS * stats)
{
    FILE          *statfp;
    Dlist         tmp;
    BEDS          *layer, *nlayer;
    int           j = 0;
    double        sumT, sumB, sumI, product;

```

```

sumB = 0;
sumT = 0;
sumI = 0;

statfp = fopen("statistics", "a");
dl_traverse(tmp, bed_list) {

    layer = (BEDS *) tmp->val;
    nlayer = (BEDS *) tmp->flink->val;
    fprintf(statfp, "Layer %d spacing = %lf\n", layer->layer_number, layer->mean_spacing);
    stats->spacing[j] = layer->mean_spacing;
    j++;
    sumT += layer->thickness;
    sumB += ((layer->thickness * layer->mean_spacing) / (layer->mean_cubed_fracap));
    if (tmp->flink != bed_list) {
        product = (layer->mean_spacing) * (nlayer->mean_spacing);
        sumI += (product / (pow(layer->interapers, 3)));
        fprintf(statfp, "sumT = %lf\n", sumT);
        fprintf(statfp, "sumB = %lf\n", sumB);
        fprintf(statfp, "sumI = %lf\n", sumI);
    }
}
fprintf(statfp, "sumT = %lf\n", sumT);
fprintf(statfp, "sumB = %lf\n", sumB);
fprintf(statfp, "sumI = %lf\n", sumI / C);
stats->K_pred = (1./12.) * sumT / (sumB + (sumI / C));
fprintf(statfp, "Analytic prediction of permeability: %lf\n", stats->K_pred);

fclose(statfp);
}                                     /* Close predictedk(). */
/*
* assemble.c
*
* David M. Doolin
*
* Stochastic mesh assembly for checking bednormal permeability in fractured,
* layered rocks.
*
*/

#include <stdio.h>
#include <math.h>
#include "fracture.h"

#define MATLAB "COND(%d, %d) = %lf;\n"
#define BC
#define DIRI

void

```

```

assembly(Dlist mesh, Dlist layerlist)
{
#define F FRACTURE

    FILE          *matfp;
    Dlist          tmp, node, tmp1;
    Dlist          interlayer;    /* A list of fracture connections on a
                                   * particular interlayer. */

    BEDS           *bedding;
    FRACTURE       *fracture;
    FRACTURE       *pfracture, *nfracture;    /* i.e. previous fracture,
                                                * next fracture. */

    double         v, vf, vp, vn, lp, ln;    /* Values of fracture
                                                * permeability. */

    int            layer;
    int            i;
    i = 0;
    layer = 0;

    matfp = fopen("fracture.m", "a");

    /*
     * This huge block of code is the mesh assembly routine. It is long,
     * poorly written, and kludgy. Conditional statements control which layer
     * is traversed in what manner.
     */

    tmp1 = layerlist->flink;

    for (tmp = mesh->flink; tmp != mesh; tmp = tmp->flink) {
        interlayer = (Dlist) tmp->val;

        /*
         * Leftmost layer traversal, write out the fracture nodes on the
         * leftmost side of the domain.
         */
        if (tmp->blink == mesh) {
            bedding = (BEDS *) tmp1->val;
            dl_traverse(node, interlayer) {
                fracture = (F *) node->val;
                vf = value(fracture->aperture, fracture->length);
                /* Fractures on the leftmost boundary. */
#ifdef BC
                fprintf(matfp, MATLAB, fracture->leftnode, fracture->leftnode, 1.0);
#else
                fprintf(matfp, MATLAB, fracture->leftnode, fracture->leftnode, vf);
                fprintf(matfp, MATLAB, fracture->leftnode, fracture->rightnode, -vf);
#endif
            }
        }
    }
#endif
}

```

```

    }
}
/*
 * This block runs down the rightmost boundary. It is the last piece
 * traversed before the loop exits, but I put it up front because it
 * was small and I wrote it before the middle interlayer routines.
 */
else if (tmp->flink == mesh) {
    bedding = (BEDS *) tmp1->val;
    /* printf("Right boundary\n"); */
    dl_traverse(node, interlayer) {
        fracture = (F *) node->val;
        vf = value(fracture->aperture, fracture->length);
#ifdef DIRI
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->rightnode, 1.0);
#else
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->rightnode, vf);
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->leftnode, -vf);
#endif
    }
}
/*
 * Every thing below here writes out the interlayer fractures. The
 * code is clumsy and huge, and exceeding hard to follow.
 */
else {
    bedding = (BEDS *) tmp1->val;
    dl_traverse(node, interlayer) {

        /*
         * At the start of the list, the 'bottom' node does not connect 3
         * ways; it only connect two ways. An extension will need to be
         * made to allow a three way connection for nodes across the
         * bottom, to allow for linearly decreasing head across the
         * domain.
         */
        if (node->blink == interlayer) { /* i.e. head of dlist. */
            fracture = (F *) node->val;
            nfracture = (F *) node->flink->val;
            vf = value(fracture->aperture, fracture->length);
            ln = nfracture->lcoord[y] - fracture->lcoord[y];
            vn = value(bedding->interapers, ln);
            v = vf + vn;
            /* Check to see which layer the fracture is in. */
            if (fracture->layer == layer) {
                fprintf(matfp, MATLAB, fracture->rightnode, fracture->rightnode, vf + vn);
                fprintf(matfp, MATLAB, fracture->rightnode, fracture->leftnode, -vf);
            }
        }
    }
}

```

```

    * Check the next fracture on the list to see if it is the
    * same layer or an adjacent layer.
    */
    if (nfracture->layer == layer) {
        fprintf(matfp, MATLAB, fracture->rightnode, nfracture->rightnode, -vn);
    } else {
        fprintf(matfp, MATLAB, fracture->rightnode, nfracture->leftnode, -vn);
    }
}
/* It is in the adjacent layer. */
else {
    fprintf(matfp, MATLAB, fracture->leftnode, fracture->leftnode, vf + vn);
    fprintf(matfp, MATLAB, fracture->leftnode, fracture->rightnode, -vf);
    if (nfracture->layer == layer) {
        fprintf(matfp, MATLAB, fracture->leftnode, nfracture->rightnode, -vn);
    } else {
        fprintf(matfp, MATLAB, fracture->leftnode, nfracture->leftnode, -vn);
    }
}
}
/*
 * At the end of an interlayer list there is another exception,
 * the 'top' node only connects two ways.
 */
else if (node->flink == interlayer) {
    fracture = (F *) node->val;
    pfracture = (F *) node->blink->val;
    vf = value(fracture->aperture, fracture->length);
    lp = fracture->lcoord[y] - pfracture->lcoord[y];
    vp = value(bedding->interapars, lp);
    v = vf + vp;
    /* Check to see which layer the fracture is in. */
    if (fracture->layer == layer) {
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->rightnode, v);
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->leftnode, -vf);
    }
    /*
     * Check the previous fracture on the list to see if it is
     * the same layer or an adjacent layer.
     */
    if (pfracture->layer == layer) {
        fprintf(matfp, MATLAB, fracture->rightnode, pfracture->rightnode, -vp);
    } else {
        fprintf(matfp, MATLAB, fracture->rightnode, pfracture->leftnode, -vp);
    }
} else {
    /* It is in the next layer. */
    fprintf(matfp, MATLAB, fracture->leftnode, fracture->leftnode, v);
    fprintf(matfp, MATLAB, fracture->leftnode, fracture->rightnode, -vf);
    if
        (pfracture->layer == layer) {

```

```

        fprintf(matfp, MATLAB, fracture->leftnode, pfraction->rightnode, -vp);
    } else {
        fprintf(matfp, MATLAB, fracture->leftnode, pfraction->leftnode, -vp);
    }
}
}
/*
 * The rest are all the non-exception fractures in the middle
 * of the domain. They are easily written out by simply
 * traversing the interlayer lists.
 */
else {
    fracture = (F *) node->val;
    pfraction = (F *) node->blink->val;
    nfraction = (F *) node->flink->val;
    vf = value(fracture->aperture, fracture->length);
    lp = fracture->lcoord[y] - pfraction->lcoord[y];
    vp = value(bedding->interapers, lp);
    ln = nfraction->lcoord[y] - fracture->lcoord[y];
    vn = value(bedding->interapers, ln);
    v = vf + vp + vn;
    /*
     * Check and see which layer the fracture is in. This controls
     * how the nodal connectivity is written out. This whole block
     * of code is one huge kludge, the result of a conceptual
     * error in implementation. A better way to have implemented
     * this would have been to make list of the nodes instead of
     * the fractures.
     */

    /*
     * Because of the initial decision alluded to above, each
     * fracture on the list has to be checked to see which layer
     * it is in before it can be written out by nodes. Otherwise
     * there is no way to know whether to write out the left node
     * or the right node for a particular fracture. This first
     * block tests for fractures in the 'same' layer; else the
     * fracture is in the next layer.
     */
    if (fracture->layer == layer) {
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->rightnode, v);
        fprintf(matfp, MATLAB, fracture->rightnode, fracture->leftnode, -vf);
        if (pfraction->layer == layer) {
            fprintf(matfp, MATLAB, fracture->rightnode, pfraction->rightnode, -vp);
        } else {
            fprintf(matfp, MATLAB, fracture->rightnode, pfraction->leftnode, -vp);
        }
        if (nfraction->layer == layer) {

```

```

        fprintf(matfp, MATLAB, fracture->rightnode, nfracture->rightnode, -vn);
    } else {
        fprintf(matfp, MATLAB, fracture->rightnode, nfracture->leftnode, -vn);
    }

    /*
     * As mentioned previously, this else statement means that
     * the fracture on the list is in the 'next' layer, not the
     * 'same' layer. 'Next' layers mean write out the left
     * fracture nodes.
     */

    } else {
        fprintf(matfp, MATLAB, fracture->leftnode, fracture->rightnode, -vf);
        fprintf(matfp, MATLAB, fracture->leftnode, fracture->leftnode, v);
        if (pfracture->layer == layer) {
            fprintf(matfp, MATLAB, fracture->leftnode, pfracture->rightnode, -vp);
        } else {
            fprintf(matfp, MATLAB, fracture->leftnode, pfracture->leftnode, -vp);
        }
        if (nfracture->layer == layer) {
            fprintf(matfp, MATLAB, fracture->leftnode, nfracture->rightnode, -vn);
        } else {
            fprintf(matfp, MATLAB, fracture->leftnode, nfracture->leftnode, -vn);
        }
    }
}

    }
    tmp1 = tmp1->flink;
}
layer++;
}
fprintf(matfp, "head = COND\\residuals';\n");
fprintf(matfp, "save head.txt head -ascii\n");
fclose(matfp);
}

double
value(double aperture, double length)
{
    double        v1;

    v1 = pow(aperture, 3) / (12*length);
    return v1;
}

/* Doubly linked lists: These are routines that you can use to create, use,
and manipulate doubly linked lists. Each list is made of Dlist structures.
The first one is a "sentinel". In other words, it holds nothing but points

```

to the first and last elements in the list proper. Sentinels are used to make programming easier — in this case, it makes deleting elements, and dealing with empty lists much easier.

```
Dlist l, tmp;
```

```
l = make_dl();
```

When you call `make_dl()`, an empty list is returned. This list has one `Dlist` element: the sentinel. The "flink" and "blink" pointers point to itself, and the "val" pointer is null.

If you want to insert the two strings "1" and "2" into this list (suppose the list is in the variable `l`), you do the following

```
dl_insert_b(l, "1");
dl_insert_b(l, "2");
```

Now, the list looks as follows:

```
*/
#ifdef VERB
\begin{verbatim}
  l --> |-----| |-----| |-----|
----->| flink |-->| flink |-->| flink |-----
-----| blink |<--| blink |<--| blink |<-----
      | val=0 |   | val |   | val |
      |-----|   |----|----|   |----|----|
                      |         |
                      --> "1"   --> "2"
\end{verbatim}
#endif
/*
```

Suppose we want to print out the list. Then we could do the following:

```
for (tmp = l->flink; tmp != l; tmp = tmp->flink) {
    printf("%s\n", tmp->val);
}
```

Note how having a sentinel element makes life easier. If the list is empty, the loop works perfectly. In this header file, there is a macro defined: `dl_traverse`. Look at it and make sure you know how it works. The above loop can be rewritten using `dl_traverse`:

```
dl_traverse(tmp, l) printf("%s\n", tmp->val);
```

The following will also work:

```
    dl_traverse(tmp, l) {
        printf("%s\n", tmp->val);
    }

*/

typedef struct dlist {
    struct dlist *flink;
    struct dlist *blink;
    void *val;
} *Dlist;

/* Nil, first, next, and prev are macro expansions for list traversal
 * primitives. */

#ifndef nil
#define nil(l) (l)
#endif

#ifndef first
#define first(l) (l->flink)
#endif

#ifndef last
#define last(l) (l->blink)
#endif

#ifndef next
#define next(n) (n->flink)
#endif

#ifndef prev
#define prev(n) (n->blink)
#endif

/* These are the routines for manipulating lists */

extern Dlist make_dl(); /* This makes an empty list */

extern void dl_insert_b(Dlist, void *);
/* Makes a new node, and inserts it before
the given node -- if that node is the
head of the list, the new node is
inserted at the end of the list */
```

```

#define dl_insert_a(n, val) dl_insert_b(n->flink, val)
                                /* Makes a new node, and inserts it after
                                the given node -- if that node is the
                                head of the list, the new node is
                                inserted at the beginning of the list */

Dlist  dlist_copy(Dlist);

extern void dl_delete_node(Dlist); /* Deletes and free's a node -- do not
                                use on the head node */

extern void dl_delete_list(Dlist); /* Deletes the entire list from existence.
                                Use only on the head node */

extern void *dl_val(Dlist); /* Returns node->val (used to shut lint up).
                                Do not use on the head node */

/* This is a macro that traverses a list. Look this one over, and be
sure you're convinced of how it works */

#define dl_traverse(ptr, list) \
    for (ptr = first(list); ptr != nil(list); ptr = next(ptr))

/* This is a macro to test whether a list is empty. */
#define dl_empty(list) (list->flink == list)

/* This function counts nodes in a list. (dmd) */
extern int      dlist_length(Dlist);
/*
 * $Source: /n/fs/vd/jsp/src/jgraph/RCS/list.c,v $
 * $Revision: 5.4 $
 * $Date: 91/06/06 16:14:15 $
 * $Author: jsp $
 */

#include <stdio.h> /* Basic includes and definitions */
#include "dlist.h"

#define boolean int
#define TRUE 1
#define FALSE 0

/*-----
 * PROCEDURES FOR MANIPULATING DOUBLY LINKED LISTS
 * Each list contains a sentinel node, so that
 * the first item in list l is l->flink. If l is
 * empty, then l->flink = l->blink = l.
 *-----

```

```

Dlist make_dl()
{
    Dlist d;

    d = (Dlist) malloc (sizeof(struct dlist));
    d->flink = d;
    d->blink = d;
    d->val = (void *) 0;
    return d;
}

void dl_insert_b(node, val)      /* Inserts to the end of a list */
Dlist node;
void *val;
{
    Dlist last_node, new;

    new = (Dlist) malloc (sizeof(struct dlist));
    new->val = val;

    last_node = node->blink;

    node->blink = new;
    last_node->flink = new;
    new->blink = last_node;
    new->flink = node;
}

/* This routine puts a list in-line with an existing list. */
dl_insert_list_b(node, list_to_insert)
Dlist node;
Dlist list_to_insert;
{
    Dlist last_node, f, l;

    if (dl_empty(list_to_insert)) {
        free(list_to_insert);
        return;
    }
    f = list_to_insert->flink;
    l = list_to_insert->blink;
    last_node = node->blink;

    node->blink = l;
    last_node->flink = f;
    f->blink = last_node;
    l->flink = node;
    free(list_to_insert);
}

```

```

}

void dl_delete_node(item)                /* Deletes an arbitrary item */
Dlist item;
{
    item->flink->blink = item->blink;
    item->blink->flink = item->flink;
    free(item);
}

void dl_delete_list(l)
Dlist l;
{
    Dlist d, next_node;

    d = l->flink;
    while(d != l) {
        next_node = d->flink;
        free(d);
        d = next_node;
    }
    free(d);
}

void *dl_val(l)
Dlist l;
{
    return l->val;
}

int
dlist_length(Dlist dlist)
{
    Dlist      tmp;
    int        i = 0;
    for (tmp = dlist->flink; tmp != dlist; tmp = tmp->flink) {
        ++i;
    }
    return i;
}

/*
 * Copy the contents of one dlist into another.  Calls the insert routine to
 * insert and malloc the memory for the new list.  Needs to go into header
 * file
 */

Dlist
dlist_copy(Dlist d1)

```

```

{
    Dlist      f, d2;
    int        i = 0;      /* Debug. */
    d2 = make_dl();

    f = d1->blink;
    while (f != d1) {
        dl_insert_b(d2, (void *) d1->flink->val);
        /*      printf("i = %d\n", ++i);      /* Debug. */
        d1 = d1->flink;
    }
    return d2;
}

Dlist
merge(Dlist d1, Dlist d2)
{
    Dlist      d3;
    Dlist      todo;

    d3 = make_dl();
    /*
     * While 1st list doesn't point to itself or 2d list doesn't point to
     * itself, do the following:
     */
    while (d1->flink != d1 || d2->flink != d2) {
        /* If d1 is empty, todo gets d2. */
        if (d1->flink == d1) {
            todo = d2;
            /* Likewise, if d2 is empty, todo gets d1. */
        } else if (d2->flink == d2) {
            todo = d1;
        }
        /*
         * Else compare the values of d1 and d2. If d1 less than d2, todo gets
         * d1,
         */
        } else if (fracture_comp(d1->flink->val, d2->flink->val) < 0) {
            todo = d1;
            /* ...else todo gets d2. */
        } else {
            todo = d2;
        }
        /* Don't want it in reverse order. */
        /* dl_insert_a(d3, todo->flink->val); */
        /*
         * Now insert the value of the first node of todo at the back of list d3.
         */
        dl_insert_b(d3, todo->flink->val);
        /* Delete the first node in list todo. */

```

```

        dl_delete_node(todo->fink);
    }                               /* End of while loop. */
    dl_delete_list(d1);
    dl_delete_list(d2);
    return d3;
}

/*
 * fracshell.c
 *
 * The driver program for the frac[0-9] series of programs.
 *
 * Need to rename and clean up, maybe scrap this program or incorporate it into
 * frac.
 */
#include<stdio.h>
#include<ctype.h>

main(argc, argv)
    int      argc;
    char     **argv;
{
    FILE      *ofp;
    char      randomizer[100], fracture[100], commandline[100];
    int       seed, i, rn, runs, j;
    double    var;
    double    width;
    double    parameter;
    i = 1;

    if (argc < 4) {
        fprintf(stderr, "Usage: bnpflow <seed> <runs> <width>\n");
        printf("\n");
        exit(-1);
    }
    /*
     * Make a string out of the command line array to print into the output
     * file.
     */
    for (j = 0; j < argc; j++) {
        strcat(commandline, argv[j]);
        strcat(commandline, " ");
    }
    /* Set up the out put file to recieve data. */
    system("rm runs");
    system("cp frac6.dat runs");
    ofp = fopen("runs", "a");

```

```

fprintf(ofp, "# %s\n", cmdline);
fprintf(ofp, "# Width Seed Spacing Predicted Computed Rel. Error\n");
fclose(ofp);

seed = atoi(argv[1]);
runs = atoi(argv[2]);
width = atof(argv[3]);

srandom(seed);
while (i <= runs) {
    sprintf(fracture, "frac6 %s %d", argv[3], random());
    fprintf(stderr, "Run %d, %s\n", i, fracture);
    system(fracture);
    i++;
}
}

```

Appendix D

Output log

Output log file for the bnpflow computer program.

Element Left Right Left coord Right coord Layer Length Aperture
number node node

1	1	2	(0.00,0.00)	(1.00,0.00)	1	1.00	1.00
2	3	4	(0.00,0.48)	(1.00,0.48)	1	1.00	1.00
3	5	6	(0.00,2.20)	(1.00,2.20)	1	1.00	1.00
4	7	8	(0.00,2.41)	(1.00,2.41)	1	1.00	1.00
5	9	10	(0.00,2.97)	(1.00,2.97)	1	1.00	1.00
6	11	12	(0.00,3.89)	(1.00,3.89)	1	1.00	1.00
7	13	14	(0.00,4.55)	(1.00,4.55)	1	1.00	1.00
8	15	16	(0.00,5.12)	(1.00,5.12)	1	1.00	1.00
9	17	18	(0.00,5.39)	(1.00,5.39)	1	1.00	1.00
10	19	20	(0.00,5.78)	(1.00,5.78)	1	1.00	1.00
11	21	22	(0.00,7.80)	(1.00,7.80)	1	1.00	1.00
12	23	24	(0.00,8.36)	(1.00,8.36)	1	1.00	1.00

Number of fractures in the layer: 12

13	25	26	(1.00,0.46)	(2.00,0.46)	2	1.00	1.00
14	27	28	(1.00,2.35)	(2.00,2.35)	2	1.00	1.00
15	29	30	(1.00,3.73)	(2.00,3.73)	2	1.00	1.00
16	31	32	(1.00,4.33)	(2.00,4.33)	2	1.00	1.00
17	33	34	(1.00,4.82)	(2.00,4.82)	2	1.00	1.00
18	35	36	(1.00,4.93)	(2.00,4.93)	2	1.00	1.00
19	37	38	(1.00,5.18)	(2.00,5.18)	2	1.00	1.00
20	39	40	(1.00,5.29)	(2.00,5.29)	2	1.00	1.00
21	41	42	(1.00,5.47)	(2.00,5.47)	2	1.00	1.00
22	43	44	(1.00,5.91)	(2.00,5.91)	2	1.00	1.00
23	45	46	(1.00,8.29)	(2.00,8.29)	2	1.00	1.00
24	47	48	(1.00,8.38)	(2.00,8.38)	2	1.00	1.00
25	49	50	(1.00,8.91)	(2.00,8.91)	2	1.00	1.00

Number of fractures in the layer: 13

26	51	52	(2.00,0.00)	(3.00,0.00)	3	1.00	1.00
27	53	54	(2.00,7.60)	(3.00,7.60)	3	1.00	1.00

28 55 56 (2.00,7.75) (3.00,7.75) 3 1.00 1.00
29 57 58 (2.00,7.83) (3.00,7.83) 3 1.00 1.00
30 59 60 (2.00,8.92) (3.00,8.92) 3 1.00 1.00

Number of fractures in the layer: 5

31 61 62 (3.00,1.18) (4.00,1.18) 4 1.00 1.00
32 63 64 (3.00,2.29) (4.00,2.29) 4 1.00 1.00
33 65 66 (3.00,3.80) (4.00,3.80) 4 1.00 1.00
34 67 68 (3.00,4.75) (4.00,4.75) 4 1.00 1.00
35 69 70 (3.00,5.32) (4.00,5.32) 4 1.00 1.00
36 71 72 (3.00,6.03) (4.00,6.03) 4 1.00 1.00
37 73 74 (3.00,7.25) (4.00,7.25) 4 1.00 1.00
38 75 76 (3.00,8.39) (4.00,8.39) 4 1.00 1.00
39 77 78 (3.00,8.72) (4.00,8.72) 4 1.00 1.00
40 79 80 (3.00,9.84) (4.00,9.84) 4 1.00 1.00

Number of fractures in the layer: 10

41 81 82 (4.00,0.00) (5.00,0.00) 5 1.00 1.00
42 83 84 (4.00,0.40) (5.00,0.40) 5 1.00 1.00
43 85 86 (4.00,0.86) (5.00,0.86) 5 1.00 1.00
44 87 88 (4.00,1.17) (5.00,1.17) 5 1.00 1.00
45 89 90 (4.00,1.53) (5.00,1.53) 5 1.00 1.00
46 91 92 (4.00,2.68) (5.00,2.68) 5 1.00 1.00
47 93 94 (4.00,4.54) (5.00,4.54) 5 1.00 1.00
48 95 96 (4.00,5.19) (5.00,5.19) 5 1.00 1.00
49 97 98 (4.00,5.55) (5.00,5.55) 5 1.00 1.00
50 99 100 (4.00,5.96) (5.00,5.96) 5 1.00 1.00
51 101 102 (4.00,5.99) (5.00,5.99) 5 1.00 1.00
52 103 104 (4.00,6.23) (5.00,6.23) 5 1.00 1.00
53 105 106 (4.00,7.02) (5.00,7.02) 5 1.00 1.00
54 107 108 (4.00,7.28) (5.00,7.28) 5 1.00 1.00
55 109 110 (4.00,9.20) (5.00,9.20) 5 1.00 1.00

Number of fractures in the layer: 15

56 111 112 (5.00,1.80) (6.00,1.80) 6 1.00 1.00
57 113 114 (5.00,2.92) (6.00,2.92) 6 1.00 1.00
58 115 116 (5.00,3.22) (6.00,3.22) 6 1.00 1.00
59 117 118 (5.00,3.79) (6.00,3.79) 6 1.00 1.00
60 119 120 (5.00,4.13) (6.00,4.13) 6 1.00 1.00
61 121 122 (5.00,4.58) (6.00,4.58) 6 1.00 1.00
62 123 124 (5.00,4.83) (6.00,4.83) 6 1.00 1.00
63 125 126 (5.00,5.79) (6.00,5.79) 6 1.00 1.00
64 127 128 (5.00,8.56) (6.00,8.56) 6 1.00 1.00
65 129 130 (5.00,9.19) (6.00,9.19) 6 1.00 1.00
66 131 132 (5.00,9.84) (6.00,9.84) 6 1.00 1.00

Number of fractures in the layer: 11

67 133 134 (6.00,0.00) (7.00,0.00) 7 1.00 1.00
68 135 136 (6.00,3.26) (7.00,3.26) 7 1.00 1.00

69 137 138 (6.00,6.49) (7.00,6.49) 7 1.00 1.00
 70 139 140 (6.00,7.27) (7.00,7.27) 7 1.00 1.00
 71 141 142 (6.00,7.57) (7.00,7.57) 7 1.00 1.00
 72 143 144 (6.00,9.02) (7.00,9.02) 7 1.00 1.00

Number of fractures in the layer: 6

73 145 146 (7.00,0.15) (8.00,0.15) 8 1.00 1.00
 74 147 148 (7.00,0.16) (8.00,0.16) 8 1.00 1.00
 75 149 150 (7.00,0.28) (8.00,0.28) 8 1.00 1.00
 76 151 152 (7.00,2.16) (8.00,2.16) 8 1.00 1.00
 77 153 154 (7.00,2.37) (8.00,2.37) 8 1.00 1.00
 78 155 156 (7.00,2.53) (8.00,2.53) 8 1.00 1.00
 79 157 158 (7.00,2.81) (8.00,2.81) 8 1.00 1.00
 80 159 160 (7.00,3.84) (8.00,3.84) 8 1.00 1.00
 81 161 162 (7.00,4.87) (8.00,4.87) 8 1.00 1.00
 82 163 164 (7.00,6.19) (8.00,6.19) 8 1.00 1.00
 83 165 166 (7.00,6.51) (8.00,6.51) 8 1.00 1.00
 84 167 168 (7.00,6.77) (8.00,6.77) 8 1.00 1.00
 85 169 170 (7.00,7.34) (8.00,7.34) 8 1.00 1.00
 86 171 172 (7.00,7.85) (8.00,7.85) 8 1.00 1.00
 87 173 174 (7.00,9.99) (8.00,9.99) 8 1.00 1.00

Number of fractures in the layer: 15

88 175 176 (8.00,0.00) (9.00,0.00) 9 1.00 1.00
 89 177 178 (8.00,1.41) (9.00,1.41) 9 1.00 1.00
 90 179 180 (8.00,2.75) (9.00,2.75) 9 1.00 1.00
 91 181 182 (8.00,3.49) (9.00,3.49) 9 1.00 1.00
 92 183 184 (8.00,4.12) (9.00,4.12) 9 1.00 1.00
 93 185 186 (8.00,6.07) (9.00,6.07) 9 1.00 1.00
 94 187 188 (8.00,8.57) (9.00,8.57) 9 1.00 1.00
 95 189 190 (8.00,8.97) (9.00,8.97) 9 1.00 1.00
 96 191 192 (8.00,9.96) (9.00,9.96) 9 1.00 1.00
 97 193 194 (8.00,9.99) (9.00,9.99) 9 1.00 1.00

Number of fractures in the layer: 10

98 195 196 (9.00,0.12) (10.00,0.12) 10 1.00 1.00
 99 197 198 (9.00,1.38) (10.00,1.38) 10 1.00 1.00
 100 199 200 (9.00,1.40) (10.00,1.40) 10 1.00 1.00
 101 201 202 (9.00,1.66) (10.00,1.66) 10 1.00 1.00
 102 203 204 (9.00,2.26) (10.00,2.26) 10 1.00 1.00
 103 205 206 (9.00,4.93) (10.00,4.93) 10 1.00 1.00
 104 207 208 (9.00,5.09) (10.00,5.09) 10 1.00 1.00
 105 209 210 (9.00,5.41) (10.00,5.41) 10 1.00 1.00
 106 211 212 (9.00,5.64) (10.00,5.64) 10 1.00 1.00
 107 213 214 (9.00,8.27) (10.00,8.27) 10 1.00 1.00
 108 215 216 (9.00,8.40) (10.00,8.40) 10 1.00 1.00

Number of fractures in the layer: 11

The total number of nodes is 216

Fracture Left head Right head Flow:

1 50.000000 48.537087 0.121909
2 50.000000 47.918679 0.173443
3 50.000000 49.104325 0.074640
4 50.000000 49.149719 0.070857
5 50.000000 49.363981 0.053002
6 50.000000 49.229653 0.064196
7 50.000000 49.049055 0.079245
8 50.000000 48.712394 0.107300
9 50.000000 48.510051 0.124162
10 50.000000 48.130459 0.155795
11 50.000000 46.177744 0.318521
12 50.000000 43.844362 0.512970

Bulk flow through layer 1: 1.856041

13 47.867890 43.635278 0.352718
14 49.073020 47.634115 0.119909
15 49.172854 48.565305 0.050629
16 49.044210 48.607291 0.036410
17 48.804134 48.428538 0.031300
18 48.742416 48.345667 0.033062
19 48.624675 48.058164 0.047209
20 48.533954 47.881581 0.054364
21 48.377378 47.447722 0.077471
22 47.785565 45.984697 0.150072
23 43.889656 39.897916 0.332645
24 43.684605 40.023947 0.305055
25 42.011759 38.835151 0.264717

Bulk flow through layer 2: 1.855562

26 40.732081 34.385479 0.528884
27 37.401361 33.454191 0.328931
28 37.235437 34.023536 0.267658
29 37.406430 34.082005 0.277035
30 38.780942 33.346902 0.452837

Bulk flow through layer 3: 1.855345

31 26.924214 23.031696 0.324376
32 24.191616 22.420820 0.147566
33 23.161059 22.288875 0.072682
34 23.342164 22.362073 0.081674
35 24.007835 22.845453 0.096865
36 25.655443 23.857295 0.149846
37 30.721038 26.985393 0.311304
38 32.615865 29.744227 0.239303
39 32.701430 29.617035 0.257033
40 31.403903 29.310332 0.174464

Bulk flow through layer 4: 1.855114

41 22.461310 22.317941 0.011947
42 22.518469 22.260782 0.021474
43 22.704726 22.074526 0.052517

44 23.023356 21.755896 0.105622
 45 22.465592 20.931075 0.127876
 46 21.701700 19.346951 0.196229
 47 22.032341 20.128444 0.158658
 48 22.614836 21.432144 0.098558
 49 22.981669 21.949034 0.086053
 50 23.645173 22.740086 0.075424
 51 23.737887 22.856811 0.073423
 52 24.181916 23.401189 0.065061
 53 26.084177 24.615008 0.122431
 54 26.989366 24.631300 0.196506
 55 27.962832 22.405128 0.463142
 Bulk flow through layer 5: 1.854920
 56 19.885647 16.665527 0.268343
 57 18.635482 16.866616 0.147405
 58 18.284534 16.401386 0.156929
 59 18.678552 17.809236 0.072443
 60 19.204563 18.511025 0.057795
 61 20.140412 19.126186 0.084519
 62 20.484345 19.218694 0.105471
 63 22.045910 18.356564 0.307446
 64 21.706130 18.370769 0.277947
 65 22.365558 19.238394 0.260597
 66 21.523642 20.132920 0.115894
 Bulk flow through layer 6: 1.854788
 67 10.533788 7.134117 0.283306
 68 16.246546 9.832057 0.534541
 69 15.121862 11.337050 0.315401
 70 14.492344 12.369327 0.176918
 71 14.887014 12.666465 0.185046
 72 18.465113 14.151067 0.359504
 Bulk flow through layer 7: 1.854715
 73 6.617416 5.389833 0.102299
 74 6.598436 5.401663 0.099731
 75 6.485108 5.419913 0.088766
 76 6.654934 5.842661 0.067689
 77 6.836184 6.069137 0.063921
 78 7.106491 6.127481 0.081584
 79 7.862004 6.101392 0.146718
 80 8.665807 6.548832 0.176415
 81 8.780146 7.622808 0.096445
 82 10.460062 8.810759 0.137442
 83 11.320709 9.846191 0.122876
 84 11.480946 10.312061 0.097407
 85 12.347400 10.649295 0.141509
 86 12.426611 10.086191 0.195035
 87 11.397349 8.555373 0.236831
 Bulk flow through layer 8: 1.854668
 88 4.997495 2.416078 0.215118

89 4.387042 1.539283 0.237313
 90 5.991833 3.713552 0.189857
 91 6.032705 4.448684 0.132002
 92 6.374122 4.075149 0.191581
 93 8.235228 3.851712 0.365293
 94 7.592589 4.593964 0.249885
 95 7.418299 5.890724 0.127298
 96 8.495217 7.631104 0.072009
 97 8.545800 7.654225 0.074298
 Bulk flow through layer 9: 1.854654
 98 2.095750 0.000000 0.174646
 99 1.487303 0.000000 0.123942
 100 1.515130 0.000000 0.126261
 101 1.457626 0.000000 0.121469
 102 2.136528 0.000000 0.178044
 103 1.706017 0.000000 0.142168
 104 1.525969 0.000000 0.127164
 105 1.633618 0.000000 0.136135
 106 2.084774 0.000000 0.173731
 107 3.124950 0.000000 0.260413
 108 3.487823 0.000000 0.290652
 Bulk flow through layer 10: 1.854624
 Average bulk flow = 1.855043
 Layer 1 spacing = 0.833333
 sumT = 1.000000
 sumB = 0.833333
 sumI = 0.641026
 Layer 2 spacing = 0.769231
 sumT = 2.000000
 sumB = 1.602564
 sumI = 2.179487
 Layer 3 spacing = 2.000000
 sumT = 3.000000
 sumB = 3.602564
 sumI = 4.179487
 Layer 4 spacing = 1.000000
 sumT = 4.000000
 sumB = 4.602564
 sumI = 4.846154
 Layer 5 spacing = 0.666667
 sumT = 5.000000
 sumB = 5.269231
 sumI = 5.452214
 Layer 6 spacing = 0.909091
 sumT = 6.000000
 sumB = 6.178322
 sumI = 6.967366
 Layer 7 spacing = 1.666667
 sumT = 7.000000

sumB = 7.844988
sumI = 8.078477
Layer 8 spacing = 0.666667
sumT = 8.000000
sumB = 8.511655
sumI = 8.745144
Layer 9 spacing = 1.000000
sumT = 9.000000
sumB = 9.511655
sumI = 9.654235
Layer 10 spacing = 0.909091
sumT = 10.000000
sumB = 10.420746
sumI = 4.827117
Analytic prediction of permeability: 0.054652

VITA

David Michael Doolin was born in Memphis Tennessee on 17 May, 1961. He attended Anchorage (Kentucky) Public School through grade 4, then Taipei (Taiwan) American School, Dominican Catholic School (Taipei), Crim Elementary School (New Jersey), Bellarmine Jr. High (Louisville, Kentucky), Binford Middle School (Bloomington, Indiana), Bloomington High School South, and failed out of R. Nelson Snider High School, Fort Wayne, Indiana in December, 1978. After passing an equivalence class, he entered Indiana University in August, 1979, transferring to Tennessee Technological University (Cookeville, Tennessee) in August, 1981. Leaving Cookeville in late 1981, he worked in the Gulf of Mexico oil field offshore Louisiana, until the domestic oil market collapsed in 1982. He then worked for several years as a framing carpenter in Austin Texas, until overbuilding flattened real estate development, a result of the savings and loan scandal in the late eighties. In 1987, he enlisted in the United States Marine Corps, earning an assignment to the 2d Reconnaissance Battalion, and served as a reconnaissance team leader in Operation Sharp Edge and in Operations Desert Shield/Desert Storm. Upon completing his service obligation in 1991, he enrolled in Pellissippi State Technical Community College, transferring to University of Tennessee, Knoxville Tennessee to complete a Bachelor of Science (Summa Cum Laude) in Geology, May, 1994. In December 1994, he married Bethany Christine Doolin née Coleman. In December 1996, he completed a Master of Science in Civil Engineering (Geotechnical). He is licensed to practice geology in Tennessee.