The Boundary Element Dual Reciprocity Method - Multidomain Approach for Solving 3D Potential Problems

Bruno Natalini

Thesis submitted in accordance with the requirements of the University of Wales for the degree of Doctor of Philosophy

October 2005

Wessex Institute of Technology

University of Wales

Acknowledgements

The thesis that is presented here would not have been possible without the help of many people, unfortunately there are more people than I can mention here.

I would like to thank to Viktor Popov, who supervised the present thesis, and Carlos Brebbia, who encouraged me to undertake the doctoral program.

Thanks to my wife, Clara, who during these five years had to cope with the stress that the doctoral program introduced into our family life.

Thanks to those colleagues at the Wessex Institute of Technology with whom I spent hours of fruitful discussions, among them (in order of appearance), Margarita Portapila, Farith Briceño, Ernesto Santana Díaz, Mariela Castillo, Maria Ibañez, Ivan Saavedra, Todorka Samardzioska, Andrés Peratta and Andrea Calaón; to the staff at WIT and BEASY and to Youssef Rashed and Roberto Torra.

I would like also to thank my friends Sara Willcocks, José Zabala, Margarita Peli, Margarita Portapila and their families, who were my substitute family during certain periods of my stay in England.

This thesis was made possible by the support of the Universidad Nacional del Nordeste, Argentina, and the Brebbia Foundation, UK.

Abstract

The Dual Reciprocity Method (DRM) is a BEM technique to approach domain dominant problems without loosing the boundary-only nature of the BEM. For this kind of problems, domain integrals are sometimes introduced in the integral formulation. The DRM converts the domain integrals into boundary integrals by means of approximation functions. The DRM is general and the number of applications solved using the procedure has been increasing in the literature since early 90s. However the DRM faces a serious drawback when applied to large problems: the resulting system of equations is dense and frequently ill-conditioned. A way to overcome this inconvenient feature is by using domain subdivision in the limiting case when the resulting internal mesh looks like a finite element grid. This technique is known as the Dual Reciprocity Method - Multi Domain approach (DRM-MD). The DRM-MD produces sparse and well-conditioned system of equations. It has been successfully applied to a variety of linear and non-linear fluid dynamics problems in 2D domains and showed good performance. The extension of the procedure to 3D cases, and especially to large 3D problems, is not straightforward since factors such as continuity of the elements, DRM approximation function, scaling, number of internal DRM nodes, etc., which largely affect the performance of the code, need to be selected.

In this thesis, several schemes were tested in order to have an insight on 3D DRM-MD implementation. A set of codes using quadratic tetrahedrons to discretize the domain was produced and ten DRM approximation functions, five globally supported and five compactly supported Radial Basis Functions (RBF), were tested for both Poisson and steady-state advection-diffusion problems using dis-continuous elements. Compactly supported RBFs showed the highest accuracy, while the augmented thin plate splines (ATPS) showed the highest consistency in terms of accuracy and convergence for the two examples considered. A problem for implementation of the compactly supported RBFs is the lack of guidelines in choosing the size of the support, which showed to have large influence on the accuracy and convergence when these types of RBFs are used. The ATPS showed satisfactory accuracy and since its use does not involve any extra parameters, at the moment it is this function the choice of the author of this thesis for use in the DRM-MD. Besides, the effect of scaling and internal DRM nodes were tested using both continuous and discontinuous elements. Results showed that scaling must be always considered to obtain an optimum performance of the code. Internal DRM nodes improve the accuracy of the codes though they are not as important as in the classical single-domain DRM.

The thesis includes a few contributions on computational implementation of the DRM-MD, among them, a general assembly procedure and an alternative way to represent partial derivatives. Finally, the simulation of the transport of a pollutant disposed in an underground repository under different scenarios, which is a large case, and a feasibility study of flow in unsaturated media, which is a strongly non-linear case, are presented and discussed. The results presented here show that the

DRM-MD is as versatile and efficient when applied to problems defined in 3D domains as it is when solving 2D problems.

Contents

Chapter 1

Introduction

1.1 The Boundary Element Method and dominant domain problems

Nowadays, the Boundary Element Method (BEM) is a well-established technique for the numerical analysis of problems governed by Partial Differential Equations (PDEs). The BEM converts the original PDE in an integral equation and solves it applying a numerical scheme such as Collocation or Galerkin methods.

One of the main advantages of the BEM is that, when a fundamental solution of the PDE is known, the integral representation is done in terms of boundary integrals, therefore only boundary discretization is required. This results in a considerable reduction in both data preparation and the number of degrees of freedom of the resulting system of equation in regard with other domain based approached such as Finite Element Method (FEM) or Finite Difference Method (FDM). The integral representation itself is a statement of the exact solution of the problem posed, and as it is defined in terms of boundary integrals, the numerical approximation is made only on the boundary of the domain.

Since mid-70s, when the fundamentals of the BEM were formulated in the way they are currently known, there had been much discussion whether the BEM is better than the FEM or other domain based approaches. In solid mechanics, most problems that lack strong gradients of stress or strain are better solved by the FEM, e.g.: problems involving non-fracture driven plasticity, materials with significant inhomogeneities, plates and shells [1]. But the FEM sometimes gives inaccurate results. This is particularly true in the cases of discontinuous functions, singularities or functions which vary rapidly. The FEM also has difficulties when modeling infinite regions and moving boundary problems. Conversely, the BEM requires a lower degree of

continuity, which admits the presence of discontinuities and helps to represent singularities in a better way, as it is the case in crack propagation problems, where the method remains unbeatable. This important characteristic results in a simpler representation of complex boundary conditions and constraints without the high degree of continuity needed in both FEM or FDM. The BEM is also preferred in problems with infinite or semi-infinite domains, such as those occurring in corrosion control, geomechanics, ocean engineering, foundations, and potential flow, because it does not require the imposition of fictitious external boundaries nor the meshing of vast external areas.

One of the main drawbacks of the BEM is that when the fundamental solution of the governing PDE is unknown or when the problem is non-linear, the boundary integrals have to be augmented by a volume integral. The early BEM analysis dealt with domain integrals through numerical cell integration. Cell integration is an effective and general technique, but it is computationally expensive. Partridge *et al.* [2] mentioned a number of methods that can be used to approach domain integrals: Analytical Integration, Fourier Expansions, the Galerkin Vector Technique, the Multiple Reciprocity Method and the Dual Reciprocity Method. Among them only DRM is completely general.

1.2 The Dual Reciprocity Method

The Dual Reciprocity Method (DRM) is acknowledged to be one of the most effective techniques in transforming domain integrals appearing in BEM formulations into boundary integrals. Introduced by Nardini & Brebbia [3] in the early 80s, the number of applications of the DRM in the BEM literature has grown steadily since early 90s. Recent developments have extended the method beyond the BEM field by combining DRM with the Method of Fundamental Solutions to create a meshless method [4]. Partridge *et al.* [2] compiled the early development of the DRM until 1990. The book describes thoroughly the computational implementation of the method in a variety of cases, and it is recommended to those interested in the details of the numerical implementation.

It is well known that it is possible to formulate a boundary-only integral representation for a given Poisson equation by means of both the fundamental solution of the Laplace equation and any particular solution of the given Poisson equation. This is called the Method of Particular Solutions. The DRM approximates a particular solution of the PDE by means of an approximation function in order to apply the Method of Particular Solutions. The DRM has been described by Partridge *et al.* as 'a generalized way of constructing particular solutions'. Indeed, it is not only a general procedure for Poisson but even for a wider range of equations, because after writing them as non-homogeneous Laplace equations and applying the DRM approximation on the non-homogeneous term, they are converted into Poisson equations. In this work we will deal with non-homogeneous Laplace PDEs, however, the DRM can be applied using particular solutions other than that of the Laplace equation.

The accuracy of the DRM is strongly dependent on the function used in the approximation. A variety of interpolation functions can be used. Until the 90s simple expansions such as $1 + R$, where R is the distance between a pre-specified fixed collocation point and a field point where the function is approximated, were preferred by users. It is not clear why this function was the most extensively used. Possibly because it is easy to implement and performs satisfactorily in a range of cases or simply because it was the choice of Nardini and Brebbia from the outset.

With the development of the theory of Radial Basis Functions (RBFs) in the context of the DRM the procedure gained a solid mathematical foundation. The origin of RBFs can be traced back to the 70s, when they were applied to geophysical data interpolation. In the 90s, Golberg and Cheng [5, 6], among others, introduced the theory in the DRM field. $1 + R$ turned out to be a particular case of RBFs. The fact that it was chosen as interpolation function in the earliest DRM developments is a bewildering example of the role that intuition can play in research.

A characteristic feature of the DRM is that a set of internal points can be used to improve the accuracy of the DRM approximation. The accuracy of the method is sensitive to both the number and distribution of the DRM points. These two factors constitute one of its main drawbacks as the criteria of how to deal with them has

been largely left to users. To the knowledge of the author there have been only two attempts to tackle the problem by using mesh refining algorithms. Schclar [7] developed a mesh refining algorithm which carries out a heuristic adaptation that eliminates points from a dense DRM mesh. It was tried in 2D potential problems and in 3D anisotropic elasticity problems. The second attempt is due to Rodriguez & Power [8], who developed and adaptive refinement scheme based on local error analysis. The scheme is valid only when collocation technique is used and it has been tried in 2D Poisson's problems.

The DRM has demonstrated to be a general and reliable procedure. However, as many of the RBFs used are globally supported, when applied to large problems the matrix of the resulting system of equations is dense and frequently ill conditioned. This makes the method computationally expensive if not instable when applied to large problems. Besides, it shall be seen in the next chapter that the method requires calculating the inverse of a matrix which size increases with the number of nodes (the **F** matrix), and for large problems this calculation constitutes another pitfall. Currently, there are two ways to avoid these problems: by using Compactly Supported RBFs (CS-RBFs) or by using Domain Decomposition.

Positive-definite CS-RBFs, which are locally supported, have been explicitly constructed and applied to multivariate safe reconstruction in mid-90s by Schaback [9]**,** Wendland [10] and Wu [11]. They founded their proposal in a firm theoretical basis. Since then, much effort has been focused in building efficient algorithms using CS-RBFs, as a result of which new functions have been proposed [10,12]. The implementation of these functions leads to a sparse system of equation that is free of the problem mentioned above, due to the local support. However, the current state-of-the-art CS-RBFs faces two main difficulties: a) the accuracy and efficiency depends on the scale of the support, being the determination of this scale uncertain and b) the convergence rate of CS-RBFs is low.

1.3 Multy domain methods

Domain subdivision is a technique that is commonly used in the BEM when the domain is piecewise homogeneous. The common practice is to consider areas where domain properties are homogeneous as separate subdomains. After applying the numerical formulation in every subdomain, the final system of equation is obtained by means of a set of matching conditions in the interfaces between subdomains. The resulting system of equation is not dense, and if many subdomains are used the system is sparse and the sparsity of the system increases with the number of subdomains.

Among the first efforts on the use of BEM with a large number of sub-regions resulting in a mesh that looks like a FEM mesh, is the work of Taigbenu who called the numerical scheme the Green Element Method (GEM) [13]. The method has been applied to 1D and 2D heterogeneous groundwater flow problems, Helmholtz and Boussinesq equations, diffusion and advection-diffusion problems using linear and quadratic interpolating functions [14] and also to the nonlinear unsaturated groundwater flow [15]. The GEM uses the fundamental solution of the Laplace equation and domain integration in each subdomain. The GEM ensures only one degree of freedom per node since the flux at the internal node is expressed in terms of the primary dependent variable, by using a finite difference scheme.

Another BEM sub-domain approach has been developed by Škerget and collaborators and named the Boundary-Domain Integral Method (BDIM). Among the other applications of the BDIM we mention here the use of the BDIM to solve convection-diffusion problems [16] using the fundamental solution of the convection-diffusion equation with constant coefficients, resulting in domain integrals with a convective term due to the perturbation velocity field. The formulation has been tested for high values of Pe number and shows good accuracy. We also mention the application of the BDIM to transient non-linear convection-diffusion problems [17] using the parabolic diffusion fundamental solution.

An alternative approach is to use domain subdivision combined with the DRM to avoid the domain integral. Popov and Power implemented a scheme of this kind and

called it the Dual Reciprocity Method – Multi-Domain approach (DRM-MD). The initial problem solved using this formulation was the flow of a mixture of gases through a porous media [18, 19, 20]. The DRM-MD has also been applied to linear and non-linear convection-diffusion problems [21, 22], driven cavity flow of a Newtonian fluid [23], pressure driven flow and the Couette flow of non-Newtonian fluids [24], and the flow of polymers inside mixers with complex geometries [25]. DRM-MD does not suffer the two main problems related to standard DRM; the systems of equations produced by DRM-MD are sparse and well conditioned, and the number and position of DRM nodes is usually not critical, since small sub-domains usually require no or few internal DRM nodes. Besides, DRM-MD is compatible with iterative solvers.

1.4 Aim and organization of the thesis

All the results that have been reported regarding the DRM-MD are encouraging in respect to both accuracy and versatility. So far, the reported DRM-MD results are obtained by solving 2D problems. Though application of DRM-MD to 3D problems should be straightforward, in reality this may not be the case since the method is very sensitive, for instance, to the choice of the DRM approximation function used. The chief aim of this thesis is to provide an insight on how to implement DRM-MD in 3D cases and show that the procedure is as accurate and versatile when solving 3D problems as it is when solving 2D cases. Furthermore, a few new schemes regarding the implementation of DRM-MD codes in both 2D and 3D cases are proposed.

The thesis is organized in seven chapters. In Chapter 1, the DRM, which appeared as a tool to overcome the difficulties arising when approaching domain-dominant problems in BEM analysis, is briefly introduced.

Chapter 2 presents the concepts that are essential to understand the subsequent chapters. It starts from the BEM and it is supposed that the reader has a background on it. It comprises a review on the DRM, the DRM-MD and details of the computational implementation of DRM-MD codes for Poisson´s problems.

Chapter 3 introduces a new way to represent partial derivatives in the DRM. The new scheme is tested in a variety of 2D linear and non-linear cases.

Chapter 4 is the core of the thesis. In this chapter, results of a set of 3D DRM-MD codes to solve the Poisson and the steady state advection-diffusion equations are presented and compared against the analytical solution. Several schemes that make use of different approximation functions, types of elements, numerical scaling and internal DRM nodes have been evaluated. Besides, the implementation of DRM-MD codes for Poisson and advection-diffusion problems is described.

The application to large practical cases is presented in Chapters 5, where results of the simulation of flow and transport of a chemical pollutant in a waste disposal considering several scenarios are displayed. The procedure showed to be a robust tool to approach these kinds of problems.

A feasibility study on the application of the DRM-MD to the solution of flow in unsaturated porous media, which is a strongly non-linear case when near saturation, is introduced in Chapter 6.

Finally, the conclusions of the thesis are presented in Chapter 7.

Chapter 2

Fundamental concepts

2.1 The direct BEM formulation for potential problems

In this thesis a range of linear and non-linear problems will be approached even though the BEM formulation will be restrained to potential problems. It shall be seen later how the different PDEs are converted into Poisson equations by means of a DRM approximation.

Let us consider the following standard Poisson equation, with constitutive coefficient equal to one

$$\nabla^2 u(\boldsymbol{x}) = b(\boldsymbol{x}) \tag{2.1}$$

where

$u(\boldsymbol{x})$ is a scalar field (potential field)

$b(\boldsymbol{x})$ is the non-homogeneous term

$\boldsymbol{x}$ is a position vector in the domain with components x_i

Eqn. (2.1) defines a family of problems known as potential problems. A characteristic feature of a potential problem is that a scalar field exists, the potential field, which depends only on the position, and which spatial derivatives describe the exchange of energy between the field and a point moving inside it.

Given a point $\boldsymbol{x}$ belonging to a domain Ω, which is enclosed by a contour Γ, the Green Integral Representation Formula for (2.1) gives the value of u at $\boldsymbol{x}$ in terms of integral equations involving the fundamental solution of the Laplace equation

$$\lambda(\boldsymbol{x})u(\boldsymbol{x}) + \int_\Gamma q^*(\boldsymbol{x},\boldsymbol{y})u(\boldsymbol{y})d\Gamma_y - \int_\Gamma u^*(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{y})d\Gamma_y = -\int_\Omega u^*(\boldsymbol{x},\boldsymbol{y})b(\boldsymbol{y})d\Omega_y \tag{2.2}$$

Here, $u^*(\boldsymbol{x},\boldsymbol{y})$ is the fundamental solution of the Laplace equation, $q(\boldsymbol{y}) = \partial u(\boldsymbol{y})/\partial n$, $q^*(\boldsymbol{x},\boldsymbol{y}) = \partial u^*(\boldsymbol{x},\boldsymbol{y})/\partial n$ and n is the unit vector normal to the boundary of the domain. The constant $\lambda(\boldsymbol{x})$ has values between 1 and 0, being equal to 1/2 on smooth parts of boundaries. It is also important to note that the above equation holds for points inside the domain, in which case $\lambda(\boldsymbol{x})$ is equal 1.

The fundamental solution of a PDE defined over an unbounded domain provides solution for a point source of unit strength. It represents a Green's function and is also known as the free-space Green's function of the problem [26]. The fundamental solution of the Laplace equation is

$$u^*(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2\pi}\ln\left(\frac{1}{r}\right) \tag{2.3}$$

for 2D problems, and

$$u^*(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{4\pi}\frac{1}{r} \tag{2.4}$$

for 3D problems, where r is the distance from the point of application of the concentrated unit source to any other point under consideration, i.e. $r = |\boldsymbol{x}\text{-}\boldsymbol{y}|$.

The equation (2.2) is the starting point of the so-called direct boundary element method for potential problems. There are several ways to derive eqn. (2.2) and all of them are thoroughly described in the BEM literature. It can be done trough weighted residuals consideration [27, 28], Green's third identity [26] or other theorems [29]. In order to solve (2.2), appropriate boundary conditions must be applied.

Notice that in equation (2.2) all the integrals are over the boundary of the domain except for the one corresponding to the term $b(\boldsymbol{y})$, which represents the sum of the non-homogeneous terms. This domain integral appears because the integral representation was done in term of the fundamental solution of the Laplace equation instead of using the one of the Poisson equation. If the fundamental solution of the Poisson equation is available, the integral representation can be done in terms of boundary-only integrals, then the full advantages of the BEM can be exploited. Domain integrals are unavoidable if the fundamental solution of the PDE is

unknown, which is the case of many kinds of potential problems, or if the governing PDE is non-linear. Early BEM analysis dealt with domain integrals through numerical cell integration. As mentioned in Chapter 1, cell integration is an effective and general technique, but it spoils the relative advantages of the BEM on other numerical domain-based approaches because it is computationally expensive. In this work, domain integrals are transformed into boundary integrals using the DRM.

2.2 The Dual Reciprocity Method

The basic idea of the DRM approximation is to expand the $b(\boldsymbol{y})$ term in (2.1) using approximation functions, i.e.:

$$b(\boldsymbol{y}) \cong \widetilde{b} = \sum_{k=1}^{J+I} \alpha_k f(\boldsymbol{y}, \boldsymbol{z}^k) \tag{2.5}$$

The functions $f(\boldsymbol{y},\boldsymbol{z}^k)$ are approximation functions that depend only on the geometry of the problem, and the constants α_k are unknown coefficients. The approximation is done at $(J+I)$ nodes, with J boundary nodes around the boundary of the domain and I nodes inside the domain.

With the DRM approximation for the non-homogeneous term, $b(\boldsymbol{y})$, the domain integral in equation (2.2) becomes:

$$\int_{\Omega} u^*(\boldsymbol{x}, \boldsymbol{y}) b(\boldsymbol{y}) d\Omega_y \cong \sum_{k=1}^{J+I} \alpha_k \int_{\Omega} u^*(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}, \boldsymbol{z}^k) d\Omega_y \tag{2.6}$$

Using (2.5), the value of the coefficients α_k can be expressed in terms of the nodal values of the function b

$$\boldsymbol{\alpha} = \mathbf{F}^{-1}\boldsymbol{b} \tag{2.7}$$

where **F** is a matrix with coefficients

$$F_{ij} = f(\boldsymbol{y}_i, \boldsymbol{z}_j) \tag{2.8}$$

A new auxiliary non-homogeneous Laplacian field, $\hat{u}$, for each collocation point $\boldsymbol{z}^k$, can be defined in the following way:

$$\frac{\partial^2 \hat{u}(\boldsymbol{y}, \boldsymbol{z}^k)}{\partial x_l \partial x_l} = f(\boldsymbol{y}, \boldsymbol{z}^k) \tag{2.9}$$

where $\hat{u}$ is an approximation of a particular solution. Applying the Green's formula to the non-homogeneous Laplacian field in (2.9) at a point $\boldsymbol{x}$, yields:

$$-\int_{\Omega} u^*(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}, \boldsymbol{z}^k) d\Omega_y = \lambda(\boldsymbol{x}) \hat{u}(\boldsymbol{x}, \boldsymbol{z}^k) + \int_{\Gamma} q^*(\boldsymbol{x}, \boldsymbol{y}) \hat{u}(\boldsymbol{y}, \boldsymbol{z}^k) d\Gamma_y - \int_{\Gamma} u^*(\boldsymbol{x}, \boldsymbol{y}) \hat{q}(\boldsymbol{y}, \boldsymbol{z}^k) d\Gamma_y \tag{2.10}$$

where $\hat{q} = \partial \hat{u} / \partial n$. Substituting the last equation into (2.6), the domain integral can be recast in terms of surface integrals. Using the resulting expression in (2.2), one finally arrives at a boundary only integral representation formula

$$\lambda(\boldsymbol{x}) u(\boldsymbol{x}) + \int_{\Gamma} q^*(\boldsymbol{x}, \boldsymbol{y}) u(\boldsymbol{y}) d\Gamma_y - \int_{\Gamma} u^*(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y}) d\Gamma_y \cong$$

$$\sum_{k=1}^{J+I} \left\{ \alpha_k \left(\lambda(\boldsymbol{x}) \hat{u}(\boldsymbol{x}, \boldsymbol{z}^k) + \int_{\Gamma} q^*(\boldsymbol{x}, \boldsymbol{y}) \hat{u}(\boldsymbol{y}, \boldsymbol{z}^k) d\Gamma_y - \int_{\Gamma} u^*(\boldsymbol{x}, \boldsymbol{y}) \hat{q}(\boldsymbol{y}, \boldsymbol{z}^k) d\Gamma_y \right) \right\} \tag{2.11}$$

For the numerical solution of the problem, the contour Γ is discretized in Γ_j elements and the density of the integrals in the above equation is defined in terms of nodal values by means of interpolation functions. In this way the equation (2.11) for the ith node can be written as:

$$\lambda_i u_i + \sum_{j=1}^{n}\left[\int_{\Gamma_j} q_{ij}^{*} u_j \mathrm{d}\Gamma_j - \int_{\Gamma_j} u_{ij}^{*} q_j \mathrm{d}\Gamma_j\right] = \sum_{k=1}^{n+m}\left\{\lambda_i \hat{u}_{ik} + \sum_{j=1}^{n}\left[\int_{\Gamma_j} q_{ij}^{*} \hat{u}_{jk} \mathrm{d}\Gamma_j - \int_{\Gamma_j} u_{ij}^{*} \hat{q}_{ik} \mathrm{d}\Gamma_j\right]\right\}\alpha_k \tag{2.12}$$

After application of collocation technique to all boundary nodes, the previous equation can be written in terms of four matrices, $\mathbf{H}$, $\mathbf{G}$, $\hat{\mathbf{U}}$ and $\hat{\mathbf{Q}}$ which depend only on the geometry of the problem

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = (\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\boldsymbol{\alpha} \tag{2.13}$$

In (2.13) the vector $\boldsymbol{\alpha}$ is unknown but it can be expressed as a function of the non-homogeneous term using (2.7), yielding

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = (\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}\boldsymbol{b} \tag{2.14}$$

2.3 The Dual Reciprocity Method - Multy Domain approach (DRM-MD)

The domain discretization in the BEM is commonly used when there are few parts of the domain that are with different properties. In that case the method of domain decomposition is often used, in which the original domain is divided into sub-regions, and on each of them the full integral representation formula is applied. A case of a domain, which is subdivided into four sub-domains, is shown in figure 2.1. Though in figure 2.1 a 2D domain is considered for the reason of simplicity, the conclusions can be extended to 3D cases as well. As the nodes at the interfaces between two subdomains introduce more unknowns than equations, every system has more unknowns than equations. However, when the matching conditions for the two subdomains are applied, the system of equations becomes closed.

Matching conditions for a potential problem establish that at every node at the interface:

a) the value of the potential is the same for both subdomains

$$u_1(\boldsymbol{x}_k) = u_2(\boldsymbol{x}_i) \tag{2.15}$$

b) the physical flux is the same for both subdomains

$$\Phi_1(\boldsymbol{u}_k, \boldsymbol{q}_k) = -\Phi_2(\boldsymbol{u}_i, \boldsymbol{q}_i) \tag{2.16}$$

where the form of the function Φ depends on the physical problem under consideration.

While the BEM matrices that arise in the single domain formulation are fully populated, the sub-region formulation leads to blocks banded matrix systems with one block for each sub-region and overlaps between blocks when sub-regions have a common interface. Equation (2.17) represents the structure of the system of equations in matrix form that corresponds to domain sub-division shown in figure 2.1.

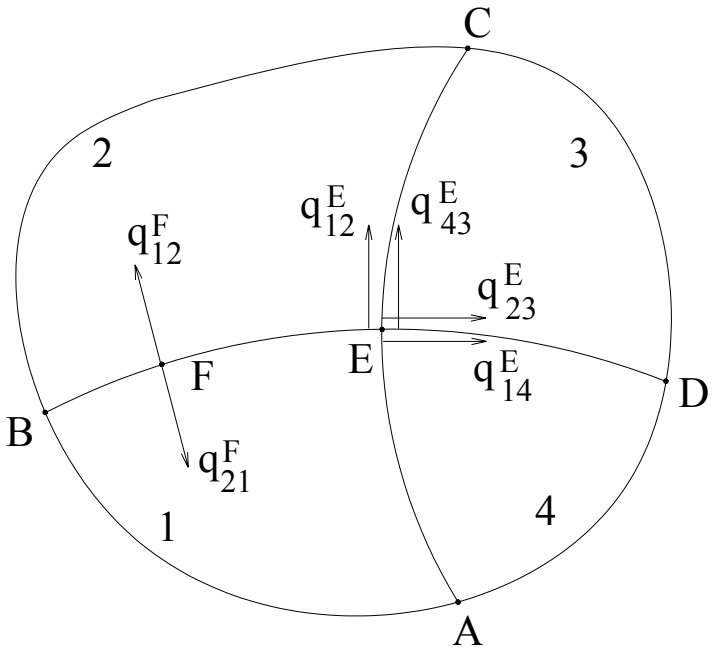


Figure 2.1: Subdivision of a domain into sub-domains.

$$\mathbf{A}^{\mathbf{MD}}\boldsymbol{x} = \begin{bmatrix} A_1 & A_{12}^i & 0 & A_{14}^i & 0 & 0 & 0 & 0 \\ 0 & A_{21}^i & A_2 & 0 & A_{23}^i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{32}^i & A_3 & A_{34}^i & 0 \\ 0 & 0 & 0 & A_{41}^i & 0 & 0 & A_{43}^i & A_4 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_{12}^i \\ x_2 \\ x_{14}^i \\ x_{23}^i \\ x_3 \\ x_{34}^i \\ x_4 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{Bmatrix} \tag{2.17}$$

A_j represents the influence coefficients obtained by integration over the external boundary that bounds the sub-domain j and x_j represents the unknown potentials and derivatives at the nodes on this part of the boundary. For example, for the sub-domain 1 the external part of the boundary is given with the curve from A to B. A_{kl}^i represents the influence coefficients obtained by integration over the interface of the sub-domains k and l and x_{kl}^i represents the unknown potentials and derivatives at the nodes on the interface. When considering nodes on the interface several different situations may occur, of which only the most characteristic two will be explained in this text.

The first one will be analysed using the node F on the interface between sub-domains 1 and 2. In this node there are four unknowns, two potentials and two normal derivatives. Two equations can be written collocating from the F node, one for the sub-domain 1 and one for the sub-domain 2. Using equations (2.15) and (2.16) the contribution of this node towards a closed system of equations is achieved.

The situation with node E is more complicated as this node is shared between four sub-domains. In each sub-domain there will be three unknowns, two derivatives and one potential, which would overall make twelve unknowns. However, as the potential is unique in this node, using equation (2.15) three of the unknowns are eliminated reducing the number of unknowns to nine. Further, by using equation (2.16) the number of unknowns will be reduced to five, that is, four normal

derivatives, for example the ones shown in figure 2.1, and the potential. With four equations that can be written collocating from the node into each of the sub-domains, the contribution of this node towards closed system of equations has not been yet achieved and unless the medium is homogeneous and the line/s B-E-D or/and A-E-C are smooth in the node E, this node would need to be converted to a discontinuous node in order that a closed system of equations is achieved. Node E when discontinuous will have four freedom nodes instead, moved for a small distance from the location of node E on the lines A-E-C, in the direction of A and C nodes, and on the line B-E-D, in the direction of B and D nodes. In each of the new freedom nodes a situation equivalent to the situation in node F will appear. When the medium is homogeneous and the intersection lines are smooth in E, it can be shown that $q_{12}=q_{43}$ and $q_{23}=q_{14}$, reducing the number of unknowns to three, making the final system of the equations over-determined. Therefore, when continuous, node E may have three degrees of freedom, contributing towards an over-determined system, or four degrees of freedom contributing towards a closed system of algebraic equations, depending on whether both or just one of the B-E-D and A-E-C lines are smooth in the node E. Node E can be continuous if all of the sub-domains 1 to 4 are with same properties, or two by two of the neighbouring domains are of same properties, i.e., 1-2 and 3-4, or, 1-4 and 2-3. In any other combination node E must be discontinuous, which produces eight degrees of freedom, contributing towards a closed system of algebraic equations. Similar analysis could be applied to nodes which are shared between three or more than four sub-domains in 2D or 3D.

Next, let us define degree of overdetermination of a continuous node, O_v, as

$$O_v = N_{eq} - N_{un} \tag{2.18}$$

where

N_{eq} is the number of equations introduced at the node

N_{un} is the number of unknowns at the node

Provided all the subdomains around the node have the same properties, the value of N_{eq} and N_{un} can be calculated as:

$$N_{eq} = N_{sub} + N_{co} \tag{2.19}$$

$$N_{un} = 1 + N_{in} \tag{2.20}$$

where

N_{sub} is the number of subdomains around the node

N_{co} is the number of independent conditions of collinearity (in 2-D problems) or coplanarity (in 3-D problems) feasible to be formulated about the interfaces joining the node.

N_{in} is the number of interfaces joining the node

2.4 Schematic view of a DRM code for Poisson's problems

Let us consider the Poisson´s equation with a constant source term

$$\nabla^2 u(\boldsymbol{x}) = d \tag{2.21}$$

The resulting system of equations that results from the application of the DRM comes straightforward from (2.14) as

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = (\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}\boldsymbol{d} \tag{2.22}$$

Calling **S** the matrix $(\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}$ yields

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = \mathbf{S}\boldsymbol{d} \tag{2.23}$$

A typical computer code that uses a DRM algorithm to solve the eqn. (2.21) follows the sequence of figure 2.2. This sequence is common to any code independently of particular features such as type of element or DRM approximation function. Since a thorough description of implementation of DRM codes is given by Partridge et al. [2], only general aspect shall be described here with emphasis in those aspect that are important to DRM-MD implementation.

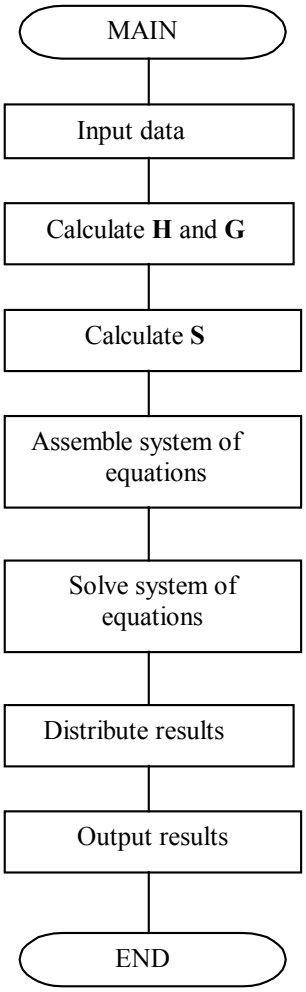


Figure 2.2: Flow chart of a DRM code for Poisson's problems.

In the 'Input data' process, information on the geometry of the problem and the mesh is introduced together with the boundary conditions and values of the constant *d*. The information on the geometry of the problem and the mesh comprises geometrical nodes coordinates, conectivities and some further information to identify whether the node is an internal DRM node or any particular feature such as being in a corner of the domain.

'Calculate **H** and **G**' process performs the integrals of the lefth-hand side of equation (2.12). The process will produce one pair of matrices **H** and **G**. To arrive at this point, the code requires some auxiliary processes to order the information, for instance: each row of the matrices corresponds to a source point and every column to a collocation point, which entails that nodes have been numbered with some criteria,

which depends on the programmer preferences. Another example: when dealing with continuous elements there will be nodes in the interface between adjacent elements; in this case, in order to assemble the matrices **H** and **G** the code must be able to identify which elements are around every node. The way in which the information is ordered depends on the particular features of the code and the preferences of the developer who wrote it.

'Calculate **S**' process computes $\mathbf{S} = (\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}$. There is only one **S** matrix for the problem and for large cases, **F** becomes ill-conditioned and $\mathbf{F}^{-1}$ can not be calculated, which is one of the main drawbacks of the DRM.

'Assemble system of equations' imposes the boundary conditions on equation (2.23) to reduce it to the form

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{y} \tag{2.24}$$

Unlike $\mathbf{A}^{\mathbf{MD}}$ in eqn. (2.17), the resulting matrix of coefficients **A** is fully populated. 'Solve system of equations' solves eqn. (2.24) using a suitable solver and gives back a vector of results, in which the formerly unknown potential and normal derivatives are contained. The results are taken from the vector of results and distributed into the vectors $\boldsymbol{u}$ and $\boldsymbol{q}$ by 'distribute results'. Finally, results are presented as output in a given format.

2.5 General aspects of DRM-MD implementation for Poisson's problems

Figure 2.3 displays the sequence that is proposed to implement DRM-MD codes for Poisson's problems. This scheme is general for 2D and 3D codes no matter what kind of elements are used. Some of the processes in figures 2.2 and 2.3 have features that are inherent to every procedure but still are conceptually the same. For instance, the 'Input data' process is different in every case since DRM-MD uses subdomains, which is not the case of the DRM procedure. This is especially true for the last three processes of the main program, and to some extent it is also for the 'Calculate **H** and

G' and 'Calculate **S**' processes. Other processes, such as 'Set boundary conditions and **NCOL**' (see section 2.5.3), which does not have an equivalent in the DRM, are conceptually different.

It shall be seen later that the scheme proposed in figure 2.3 (and its corresponding code) can be used for a wide range of equations with minor changes. Let us now do some remarks about the first four processes.

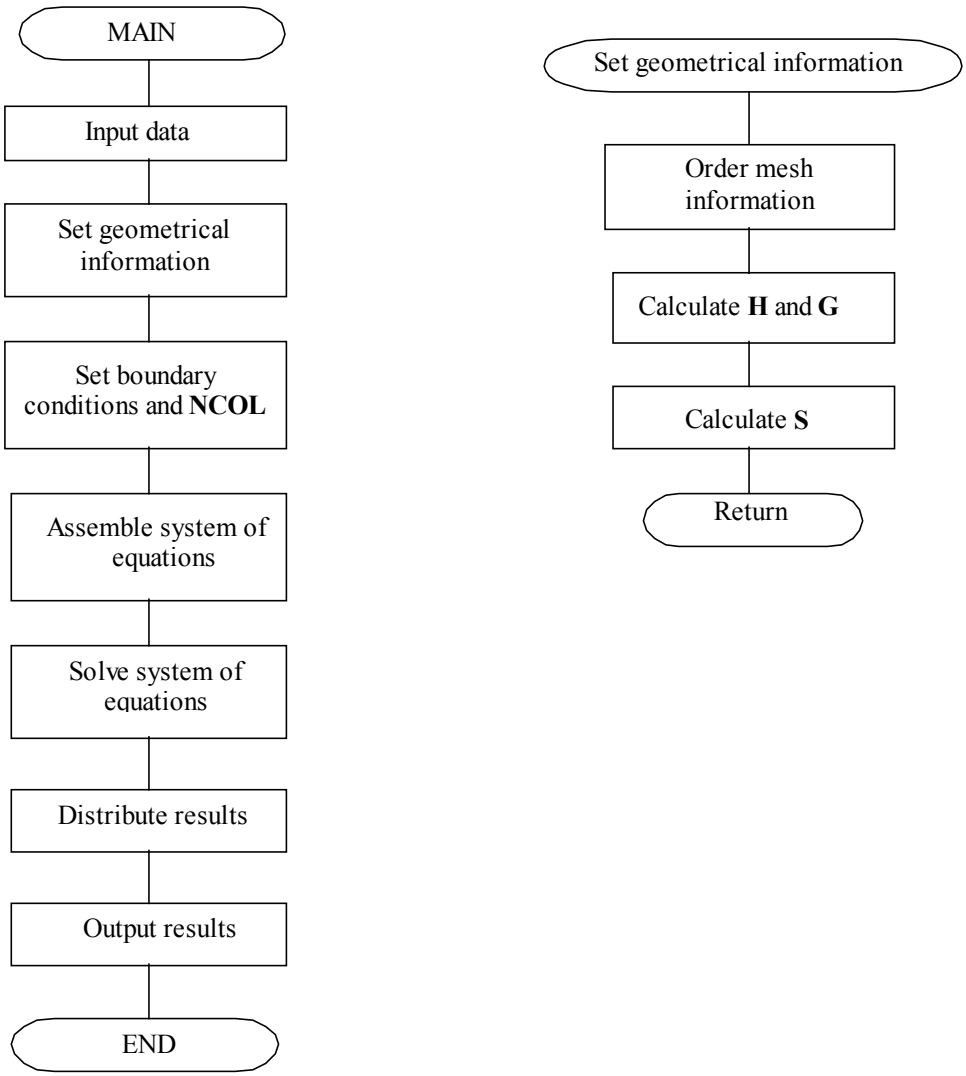


Figure 2.3: proposed scheme for DRM-MD codes for Poisson's problems.

2.5.1 Input data

It is conceptually the same than its counterpart of DRM. The difference is that the conectivities correspond to domain elements.

2.5.2 Set geometrical information

This process comprises three subprocesses in which all information that depends only on the mesh and the geometry of the domain is processed. It is independent of the boundary conditions or governing equation being considered.

2.5.2.1 Order mesh information: subsequent processes make use of a set of arrays containing all information that can be obtained from the geometrical data. For instance, given a subdomain which are the subdomains around it, or what are the normal versor components at a given node, etc. That set of arrays is created here.

2.5.2.2 Calculate **H** *and* **G**: It is conceptually similar to its counterpart of DRM, but, as the full integral representation formula is applied on each subdomain, there will be as many small **H** and **G** matrices as there are subdomains.

2.5.2.3 Calculate **S**: similar to its counterpart of DRM. There will be as many small **S** matrices as subdomains. Now, the problem of having an ill-conditioned matrix **F** has been eliminated no matter how large the problem under analysis is.

2.5.3 Set boundary conditions and NCOL

The assembly process requires specifying the boundary conditions in a format that is different than the format in which the boundary conditions are input. That task is performed here.

In eqn. (2.23) some coefficients multiply unknown values of u and q, and some other ones multiply values that have been specified as boundary conditions. In this process, those coefficients multiplying known values are identified, and a column in the matrix of coeficients $\mathbf{A}^{\mathbf{MD}}$ of eqn. (2.17) is assigned to every coefficient multiplying an unknown. This information, that is, the number of column that every coefficient has been assigned in the matrix $\mathbf{A}^{\mathbf{MD}}$, are stored in an array called **NCOL**. Note that

this process depend on the geometry, mesh and boundary conditions independently of the governing equation.

2.5.4 Assemble system of equations

This is the only process that is linked to the governing equation. The key concept in the assembly is that every node of freedom generates coefficients that multiply the variables involved. These coefficients depend on the governing equation. The assembly is done by going through the mesh subdomain by subdomain, and at every subdomain going through the nodes of freedom. At every node the values of the coefficients are calculated and assigned to the matrix $\mathbf{A}^{\mathbf{MD}}$ or incorporated into the right-hand-side according to the value of the array **NCOL**.

More details on the implementation of 3D codes will be discussed in Chapter 4.

Chapter 3

First and higher order derivatives in the DRM-MD approach

3.1 Introduction

The accuracy of both DRM and DRM-MD is strongly dependant on the approximation function used in the DRM approximation. Although a variety of functions can in principle be used as an interpolation function, previous work [2] has shown that good performances were obtained in a variety of cases with simple expansions, the most popular of which is $1+R$, where R is the distance between the source point $\boldsymbol{x}$ and a set of field points. However, this is not the case when dealing with equations where partial derivatives need to be represented using the DRM approximation. In the classical approach, the partial derivatives are approximated by means of partial derivatives of the interpolation function used. Zhu and Zhang [30] have shown that when performing partial derivatives of the $1+R$ interpolation function the procedure introduces singularities on all nodes, as a result of which large numerical errors are produced. Zhu and Zhang [30] and Chen [31] avoided these artificially created singularities by constructing a transformation, which leads to improved numerical results.

An important feature of the DRM-MD is that it offers the possibility of representing the spatial partial derivatives as a function of the normal derivatives that are calculated directly from the integral equation, instead of approximating them using the DRM representation of partial derivatives. In this way the partial derivatives of the interpolation function are reduced by one order avoiding artificial singularities that would otherwise appear in some cases. In this chapter, the proposed technique is

applied to several linear and non-linear advection-diffusion problems using different 2D DRM-MD codes in which various formulations for the first and second order partial derivatives have been implemented and tested.

3.2 The interpolation functions in the DRM

In this chapter, results of tests of three different radial basis functions in the DRM approximation are shown. The first one is the most often used in the past, 1+*R*, whose implementation in the DRM has been described in [2]. The second one is the augmented thin plate splines (ATPS). This alternative, proposed by Golberg and Chen [5], uses a linear combination of thin plate splines and linear functions, which for 2D problems are two-dimensional analogues of one-dimensional cubic splines. Using ATPS, $b(\boldsymbol{y})$ can be expanded as

$$b(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k R^2(\boldsymbol{y}, \boldsymbol{z}^k) \log R(\boldsymbol{y}, \boldsymbol{z}^k) + P(\boldsymbol{y}) \tag{3.1}$$

where

$$P(\boldsymbol{y}) = ax_1 + bx_2 + c \tag{3.2}$$

and $\boldsymbol{y} = (x_1, x_2)$. The three additional augmented equations that result in an optimum interpolant and assure the uniqueness of the solution of the above system of equations are

$$\sum_{k=1}^{J+I} \alpha_k = \sum_{k=1}^{J+I} \alpha_k x_1^k = \sum_{k=1}^{J+I} \alpha_k x_2^k = 0 \tag{3.3}$$

where $\boldsymbol{z}^k = (x_1^k, x_2^k)$. The corresponding particular solution is given by [32]

$$\begin{aligned} f_R &= R^2 \log R \quad \Rightarrow \quad \hat{u}_R = R^4 (\log R - 1/2)/16 \\ f_P &= ax_1 + bx_2 + c \quad \Rightarrow \quad \hat{u}_P = ax_1^3 + bx_2^3 + c(x_1^2 + x_2^2) \end{aligned} \tag{3.4}$$

Hardy [33] introduced another class of RBF, the multiquadric (MQ)

$$f(R) = \left(R^2 + c^2\right)^{1/2} \tag{3.5}$$

where c is a free parameter referred to as a shape parameter. The corresponding particular solution can be found in [32].

In a landmark study, Franke [34] tested 29 different algorithms for the scattered data interpolation problem on a variety of known data surfaces. He found that Hardy's Multiquadric method gave the most accurate results, followed by the Thin-Plate Spline of Duchon. An important constraint to using MQ in practical applications is that the shape parameter c needs to be chosen conveniently, and how to do this is still matter of research.

3.3 Treatment of partial derivatives in the DRM

In the DRM approach an algorithm is established to express partial derivatives in the vector $\boldsymbol{b}$ in (2.14) as functions of either $\boldsymbol{u}$ or $\boldsymbol{q}$.

In the classical approach the starting point is to express the potential at a point $\boldsymbol{x}$ in terms of the approximation function f, in a similar way as it was done for b in (2.5)

$$u(\boldsymbol{x}) \cong \sum_{k=1}^{J+I} f(\boldsymbol{x}, \boldsymbol{z}_k)\beta \tag{3.6}$$

or expressed in matrix form

$$\boldsymbol{u} = \mathbf{F}\boldsymbol{\beta} \tag{3.7}$$

where the **F** matrix is the one defined by (2.8) and $\boldsymbol{\beta}$ are unknown coefficients where $\boldsymbol{\beta} \neq \boldsymbol{\alpha}$.

Differentiation of equation (3.7) produces

$$\frac{\partial \boldsymbol{u}}{\partial x_i} = \frac{\partial \mathbf{F}}{\partial x_i}\boldsymbol{\beta} \tag{3.8}$$

After re-writing (3.7) as $\boldsymbol{\beta} = \mathbf{F}^{-1}\boldsymbol{u}$, equation (3.8) becomes

$$\frac{\partial \boldsymbol{u}}{\partial x_i} = \frac{\partial \mathbf{F}}{\partial x_i}\mathbf{F}^{-1}\boldsymbol{u} \tag{3.9}$$

Second order partial derivatives can be expressed in two ways. The first one is obtained by differentiating twice (3.7)

$$\frac{\partial^2 \boldsymbol{u}}{\partial x_i^2} = \frac{\partial^2 \mathbf{F}}{\partial x_i^2}\mathbf{F}^{-1}\boldsymbol{u} \tag{3.10}$$

The second one uses the DRM approximation for representing the first order partial derivatives, as was previously used for representing $\boldsymbol{b}$ in (2.5) and $\boldsymbol{u}$ in (3.7)

$$\frac{\partial \boldsymbol{u}}{\partial x_i} \cong \sum_{k=1}^{J+I} f(\boldsymbol{x}, \boldsymbol{z}_k)\gamma(\boldsymbol{z}_k) = \mathbf{F}\boldsymbol{\gamma} \tag{3.11}$$

Differentiating (3.11)

$$\frac{\partial^2 \boldsymbol{u}}{\partial x_i^2} = \frac{\partial \mathbf{F}}{\partial x_i}\boldsymbol{\gamma} \tag{3.12}$$

and replacing $\boldsymbol{\gamma}$ from (3.11) yields

$$\frac{\partial^2 \boldsymbol{u}}{\partial x_i^2} = \frac{\partial \mathbf{F}}{\partial x_i}\mathbf{F}^{-1}\frac{\partial \boldsymbol{u}}{\partial x_i} \tag{3.13}$$

Finally, expressing $\frac{\partial \boldsymbol{u}}{\partial x_i}$ through (3.9) produces the final expression

$$\frac{\partial^2 \boldsymbol{u}}{\partial x_i^2} = \frac{\partial \mathbf{F}}{\partial x_i}\mathbf{F}^{-1}\left(\frac{\partial \mathbf{F}}{\partial x_i}\mathbf{F}^{-1}\boldsymbol{u}\right) \tag{3.14}$$

As the diagonal elements of the $\frac{\partial \mathbf{F}}{\partial x_i}$ matrix are undefined when $f = 1+R$, equation (3.9) introduces singularities that lead to large numerical errors. The same happens to formulation (3.10) when f is ATPS in 2D because in this case the diagonal elements of the $\frac{\partial^2 \mathbf{F}}{\partial x_i^2}$ matrix are undefined. Zhu and Zhang [30], likewise Chen [31], proposed a functional transformation to avoid these singularities. Another possibility to avoid this problem is to use an interpolation function that does not introduce singularities when differentiated. The DRM-MD offers a third option that consists of expressing the partial derivatives through the normal derivatives q, which appear in (2.14). To introduce this algorithm let us consider a corner node on the boundary of a subdomain as shown in figure 3.1.

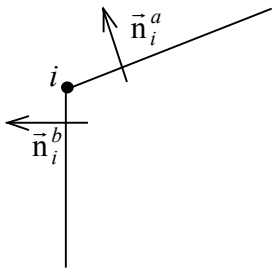


Figure 3.1: A corner node on the boundary.

In a single subdomain, at every corner node there will be three variables, some of which can be given as the boundary condition: the potential u, the normal derivative before the node q^b, and the normal derivative after the node q^a. The normal

derivatives can be represented as a scalar product of the gradient $\vec{\nabla} u_i$ and the outward unitary vector normal to the boundary

$$q^b{}_i = \vec{\nabla} u_i \cdot \vec{\mathrm{n}}^b{}_i \tag{3.15}$$

and

$$q^a{}_i = \vec{\nabla} u_i \cdot \vec{\mathrm{n}}^a{}_i \tag{3.16}$$

Eqns (3.15) and (3.16) can be re-written as

$$q^b{}_i = \frac{\partial u_i}{\partial x_{1_i}} n^b_{1\,i} + \frac{\partial u_i}{\partial x_{2_i}} n^b_{2\,i} \tag{3.17}$$

$$q^a{}_i = \frac{\partial u_i}{\partial x_{1_i}} n^a_{1\,i} + \frac{\partial u_i}{\partial x_{2_i}} n^a_{2\,i} \tag{3.18}$$

These two equations form a linear system that yields

$$\frac{\partial u_i}{\partial x_{1_i}} = u_{i,1}(q^b{}_i, q^a{}_i) = \left(q^b{}_i\, n^a_{2\,i} - q^a{}_i\, n^b_{2\,i}\right) / J_i \tag{3.19}$$

and

$$\frac{\partial u_i}{\partial x_{2_i}} = u_{i,2}(q^b{}_i, q^a{}_i) = \left(q^a{}_i\, n^b_{1\,i} - q^b{}_i\, n^a_{1\,i}\right) / J_i \tag{3.20}$$

where

$$J_i = n^b_{1\,i}\, n^a_{2\,i} - n^a_{1\,i}\, n^b_{2\,i} \tag{3.21}$$

In order to produce the second order partial derivatives, a similar scheme to (3.13) is applied on (3.19) and (3.20) to yield

$$\frac{\partial^2 \boldsymbol{u}}{\partial x_j^2} = \frac{\partial \mathbf{F}}{\partial x_j} \mathbf{F}^{-1}\left(\boldsymbol{u}_{,j}(\boldsymbol{q})\right); \quad (j = 1,2) \tag{3.22}$$

The formulations given by (3.19), (3.20) and (3.22) have two main advantages in respect to the ones given by (3.9) and (3.10). Firstly, they reduce the order of the PDs of the approximation function by one, in respect to the formulation given by (3.9) and (3.10), and secondly they involve less number of operations. This algorithm can be implemented in the corner nodes only and therefore for the DRM nodes, which are located in the interior of the domain, the classical approach must be applied. This scheme can be applied to 3D problems in a similar way. It is evident that this approach offers substantial benefits when the proportion of corner nodes is high, that is, in the DRM-MD meshes.

For a 2D case that shall be tested later, theses schemes are applied in the following way. If the advection-diffusion equation is considered

$$D\nabla^2 u - \vec{V} \cdot \vec{\nabla} u - ku = 0 \tag{3.23}$$

where

D is the coefficient of dispersion

$\vec{V}$ is the vector of flow velocity

k is the reaction constant, which is constant in the whole domain,

The equation can be written as a non-homogeneous Laplace equation

$$\nabla^2 u = \frac{1}{D}\left(\vec{V} \cdot \vec{\nabla} u + ku\right) \tag{3.24}$$

The eqn (2.14), which represents the system of equations resulting from the DRM formulation, becomes

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = (\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}\frac{1}{D}\left[\boldsymbol{V}_{\boldsymbol{x}}\frac{\partial \boldsymbol{u}}{\partial x} + \boldsymbol{V}_{y}\frac{\partial \boldsymbol{u}}{\partial y} + k\boldsymbol{u}\right] \tag{3.25}$$

where

$\mathbf{H}$ and $\mathbf{G}$ are the BEM matrices

$\hat{\mathbf{U}}$ and $\hat{\mathbf{Q}}$ are the DRM matrices

$\boldsymbol{V}_x$ and $\boldsymbol{V}_y$ are diagonal matrices containing the flow velocity components

Replacing the PDs by (3.9) and denoting the matrix $(\mathbf{H}\hat{\mathbf{U}} - \mathbf{G}\hat{\mathbf{Q}})\mathbf{F}^{-1}$ as $\mathbf{S}$ yields

$$\mathbf{H}\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = \frac{\mathbf{S}}{D}\left[\boldsymbol{V}_{\boldsymbol{x}}\frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1}\boldsymbol{u} + \boldsymbol{V}_{\boldsymbol{y}}\frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1}\boldsymbol{u} + k\boldsymbol{u}\right] \tag{3.26}$$

Denoting the matrix $\frac{\mathbf{S}}{D}\left[\boldsymbol{V}_{\boldsymbol{x}}\frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1} + \boldsymbol{V}_{\boldsymbol{y}}\frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1}\right]$ as $\mathbf{T}$ and reordering produces

$$\left(\mathbf{H} - \mathbf{T} - \frac{\mathbf{S}}{D}k\right)\boldsymbol{u} - \mathbf{G}\boldsymbol{q} = 0 \tag{3.27}$$

Eqn (3.27), after applying the boundary conditions, is the linear system of equations resulting from (3.23) when the classical formulation (3.9) for first partial derivatives is used.

In order to analyse the equivalent equation obtained when the new formulation is used, a case where the nodes in the domain are corner boundary nodes only will be considered. Replacing the partial derivatives by (3.19) and (3.20) in (3.25) yields in index notation

$$\sum_{j=1}^{J} h_{ij} u_j - \sum_{j=1}^{2J} g_{ij} q_j = \sum_{j=1}^{J} \frac{s_{ij}}{D} \left[\sum_{k=1}^{J} v_{x\,jk} \left(\frac{n^a_{y\,k}}{J_k} q^b{}_k - \frac{n^b_{y\,k}}{J_k} q^a{}_k \right) + \sum_{k=1}^{J} v_{y\,jk} \left(\frac{n^b_{x\,k}}{J_k} q^a{}_k - \frac{n^a_{x\,k}}{J_k} q^b{}_k \right) + k u_j \right] \tag{3.28}$$

By splitting the matrix **G** into two matrices containing the columns of **G** that multiply the normal derivatives before and after the node, and reordering the terms produces the following equation

$$\sum_{j=1}^{J} h_{ij} u_j - \sum_{j=1}^{J} g^b{}_{ij} q^b{}_j - \sum_{j=1}^{J} g^a{}_{ij} q^a{}_j = \sum_{j=1}^{J} \frac{s_{ij}}{D} \left[\sum_{k=1}^{J} \left(v_{x\,jk} \frac{n^a_{y\,k}}{J_k} - v_{y\,jk} \frac{n^a_{x\,k}}{J_k} \right) q^b{}_k \right] + \sum_{k=1}^{J} \left(v_{y\,jk} \frac{n^b_{x\,k}}{J_k} - v_{x\,jk} \frac{n^b_{y\,k}}{J_k} \right) q^a{}_k + k u_j \Bigg] \tag{3.29}$$

Denoting $\left(v_{y\,jk} \frac{n^b_{x\,k}}{J_k} - v_{x\,jk} \frac{n^b_{y\,k}}{J_k} \right)$ and $\left(v_{x\,jk} \frac{n^a_{y\,k}}{J_k} - v_{y\,jk} \frac{n^a_{x\,k}}{J_k} \right)$

as z^b_{jk} and z^a_{jk} respectively, and reordering the terms yields

$$\sum_{j=1}^{J} \left(h_{ij} - k s_{ij} \right) u_j - \sum_{j=1}^{J} \left(g^b{}_{ij} + \sum_{k}^{J} s_{ik} z^a_{kj} \right) q^b{}_j - \sum_{j=1}^{J} \left(g^a{}_{ij} + \sum_{k}^{J} s_{ik} z^b_{kj} \right) q^a{}_j = 0 \tag{3.30}$$

or in matrix notation

$$\left(\mathbf{H} - k \frac{\mathbf{S}}{D} \right) \boldsymbol{u} - \left(\mathbf{G}^b + \frac{\mathbf{S}}{D} \mathbf{Z}^a \right) \boldsymbol{q}^b - \left(\mathbf{G}^a + \frac{\mathbf{S}}{D} \mathbf{Z}^b \right) \boldsymbol{q}^a = 0 \tag{3.31}$$

Eqn (3.31), after applying the boundary conditions, is the linear system of equations resulting from application of the BEM to (3.23) and application of the new scheme (3.19) and (3.20) to first partial derivatives. Note that when there are boundary nodes on a smooth boundary, or DRM nodes, the resulting system of equations will be a combination of (3.27) and (3.31). When the corner node is internal, e.g. between four internal subdomains, the same analysis would apply and the matching conditions for potential and flux would have to be added to the system of equations.

3.4 Numerical examples

3.4.1 Linear case: a 1D advection-diffusion problem

In this section four different 2-D DRM-MD schemes are used to solve the advection-diffusion equation

$$D\nabla^2 u - \vec{V}\cdot\vec{\nabla}u - ku = 0 \tag{3.32}$$

governing a 1D problem with analytical known solution. The velocity in this 1-D problem is a function of the x-coordinate and of the reaction constant k

$$V(x,k) = \frac{1}{L}\ln\left(\frac{U_1}{U_0}\right) + k\left(x - \frac{L}{2}\right) \tag{3.33}$$

where $U_0 = u(x = 0)$ and $U_1 = u(x = L)$ are the boundary conditions. The analytical solution, when $D = 1$, is given by

$$u(x) = U_0 \exp\left\{\left(\frac{kx^2}{2}\right) + \left[\frac{1}{L}\ln\left(\frac{U_1}{U_0}\right) - L\frac{k}{2}\right]x\right\} \tag{3.34}$$

The four tested schemes differed in the way that the first order partial derivatives were evaluated. The first and the second schemes are based on (3.9) and use $1+R$ and

ATPS as interpolation function, respectively. The third and fourth ones use the new scheme given by (3.19) and (3.20) and 1+*R* and ATPS as interpolation functions, respectively.

In order to produce equivalent 1-D results from the 2-D codes, rectangular domains of length *L* in the *x* direction and width *W* in the *y* direction were used and the following boundary conditions were applied (see figure 3.2)

$$u(0,y)=U_0 \text{ and } u(L,y)=U_1$$

and

$$\left.\frac{\partial u}{\partial n}\right|_{y=W/2} = \left.\frac{\partial u}{\partial n}\right|_{y=-W/2} = 0 \tag{3.35}$$

The velocity field is defined as

$$V_x = V(x) \text{ and } V_y = 0 \tag{3.36}$$

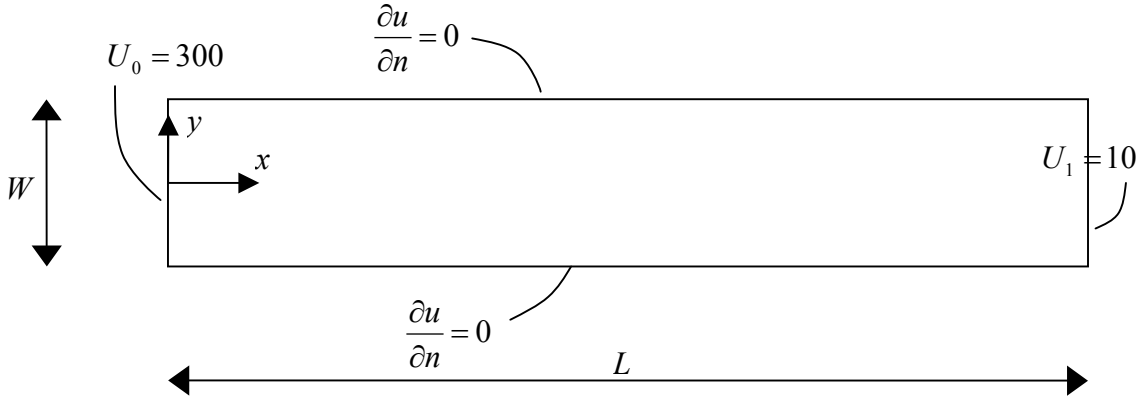


Figure 3.2: 2D domain and boundary conditions.

The domain was subdivided in rectangular subdomains and linear elements were used within each subdomain. The codes used a direct Gauss elimination with pivoting solver. Figure 3.3 shows the analytical solution for: (a) the potential, and (b) the normal derivative for three different values of the reaction constant *k*. Note that

the normal derivative in this 1D case is represented by $\partial u/\partial x$ and can be obtained by differentiating (3.34).

Figures 3.4 to 3.9 show the relative error in the potential and the normal derivative using the four codes for different values of the reaction constant, the relative error being defined as

$$\text{Error [\%]} = \frac{\text{numerical result} - \text{analytical result}}{\text{analytical result}} * 100 \qquad (3.37)$$

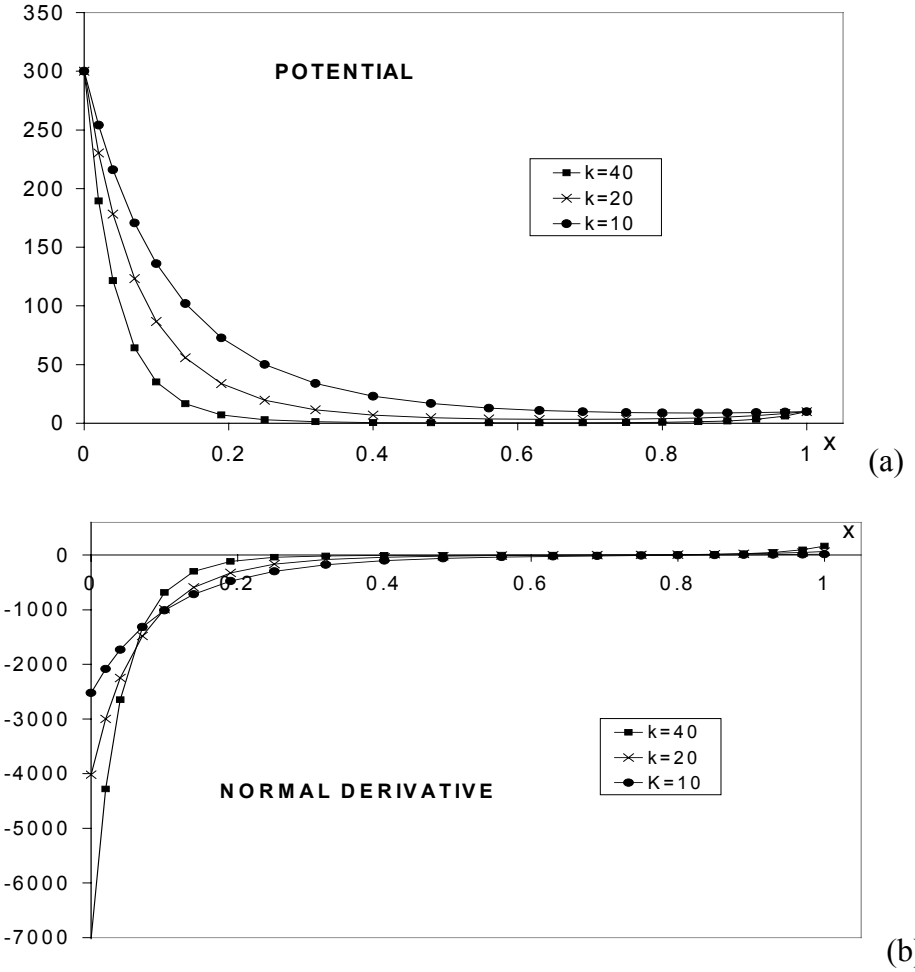
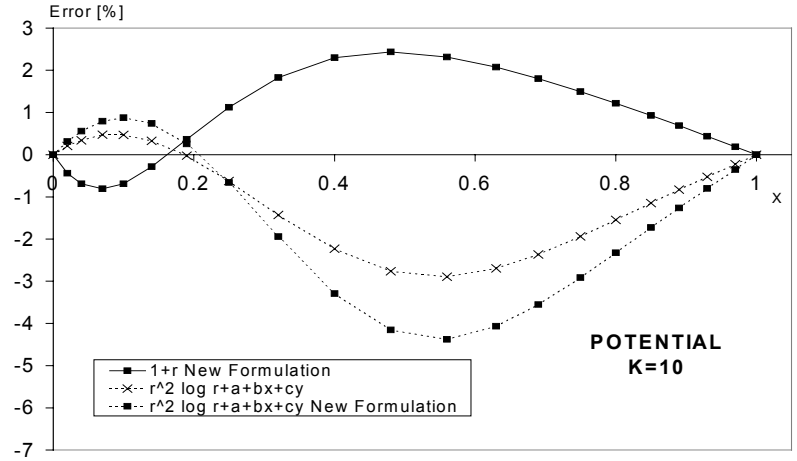


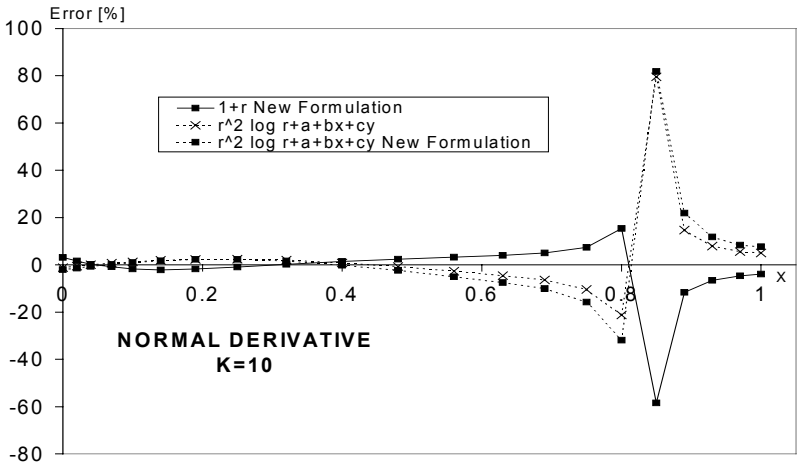
Figure 3.3: Analytical solution for the: (a) potential and (b) normal derivative, for the examples tested.

The length of the domain was L=1, U_0 = 300 and U_1 = 10. Three meshes were used: 4×20 (4 subdivisions in the *y* direction and 20 subdivision in the *x* direction), 4×40

and 2×80. In some examples a DRM node was placed at the centre of every subdomain. In further text to these nodes will be referred as central DRM nodes (CDRMN). In most of the figures the errors corresponding to the first code are not displayed because they are one or two orders of magnitude higher than those of the other codes. In those cases the maximum values of the errors obtained using the first code are given in the figure caption.



(a)



(b)

Figure 3.4: Error distribution for the: (a) potential and (b) normal derivative for W=0.2, k=10, mesh = 4×20, CDRMN used, $V_{x\ max}$= -8.4 (the maximum errors for the code 1 were 49 and 327% for the potential and normal derivative, respectively).

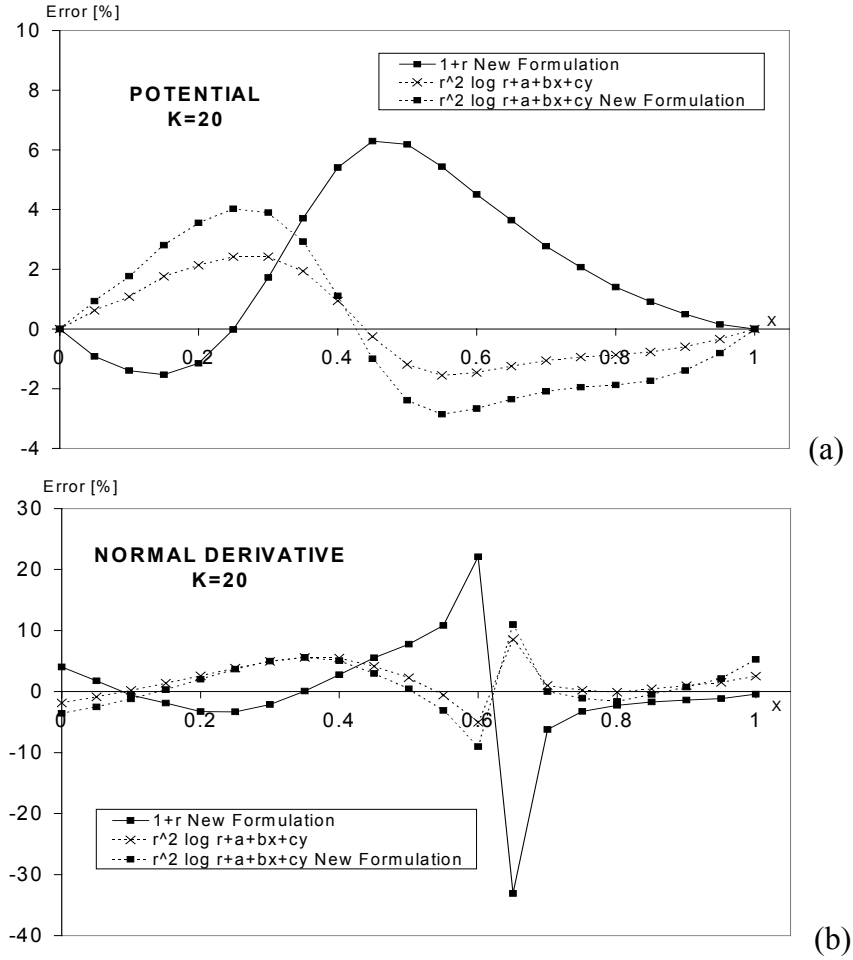
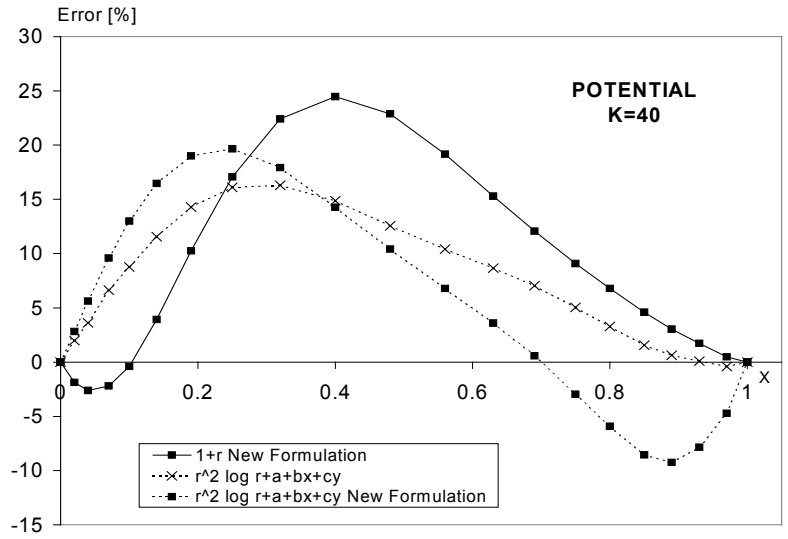
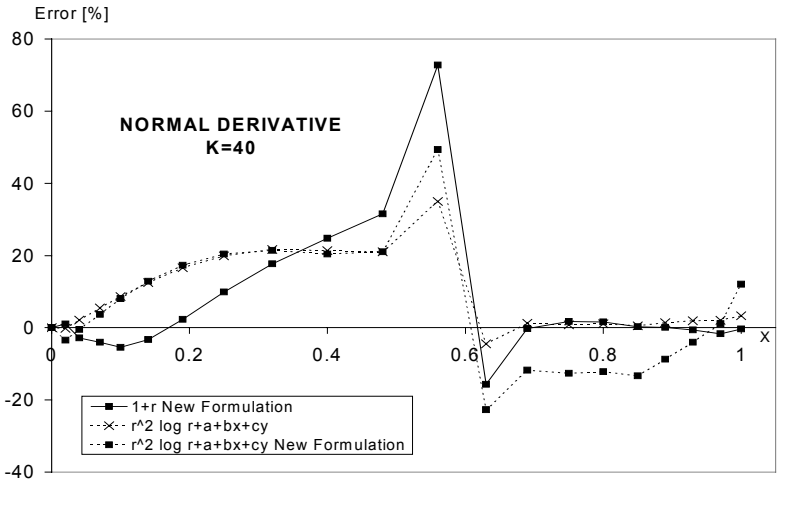


Figure 3.5: Error distribution for the: (a) potential and (b) normal derivative for W=0.2, k=20, mesh = 4×20, CDRMN used, $V_{x\ max}$= -13.4 (the maximum errors for the code 1 were 84 and 290% for the potential and normal derivative, respectively).



(a)



(b)

Figure 3.6: Error distribution for the: (a) potential and (b) normal derivative for W=0.2, k=40, mesh = 4×20, CDRMN used, $V_{x\ max}$=-23.4 (the maximum errors for the code 1 were 200 and 590% for the potential and normal derivative, respectively).

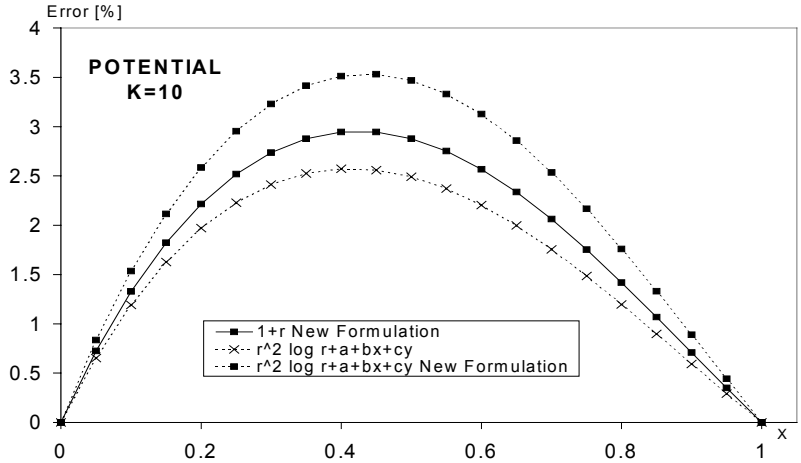


Figure 3.7: Error distribution for the: (a) potential and (b) normal derivative for W=0.2, k=10, mesh = 4×20, CDRMN not used, $V_{x\ max}$= -8.4 (the maximum errors for the code 1 were 80 and 547% for the potential and normal derivative, respectively).

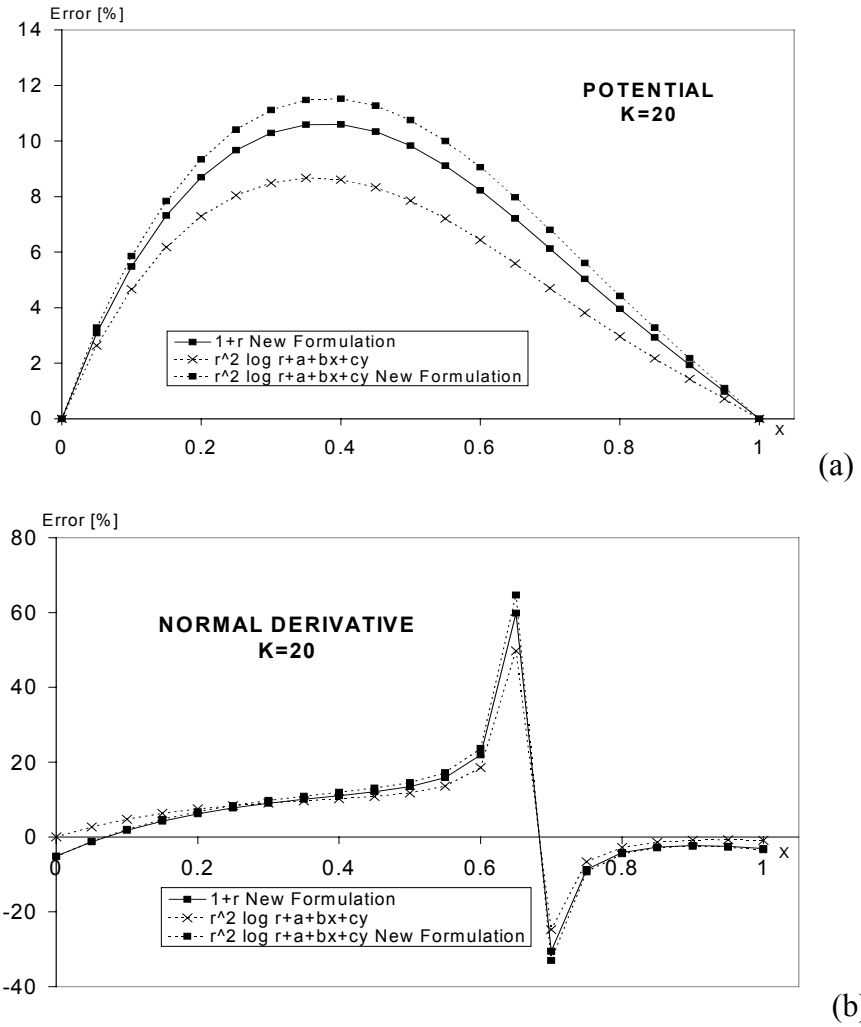
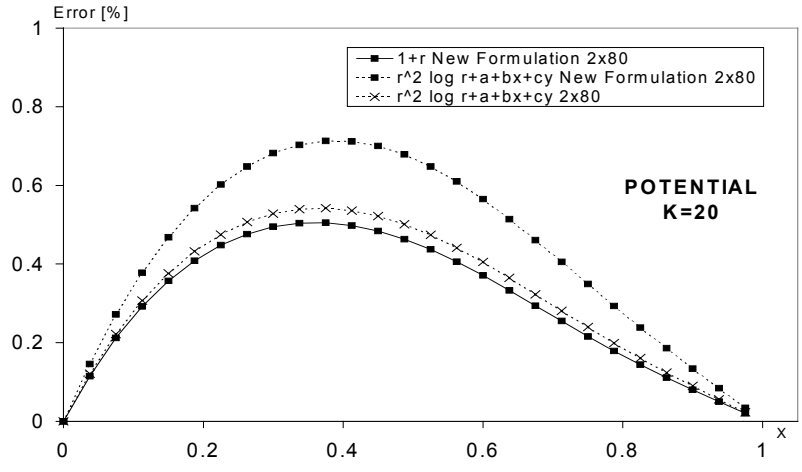
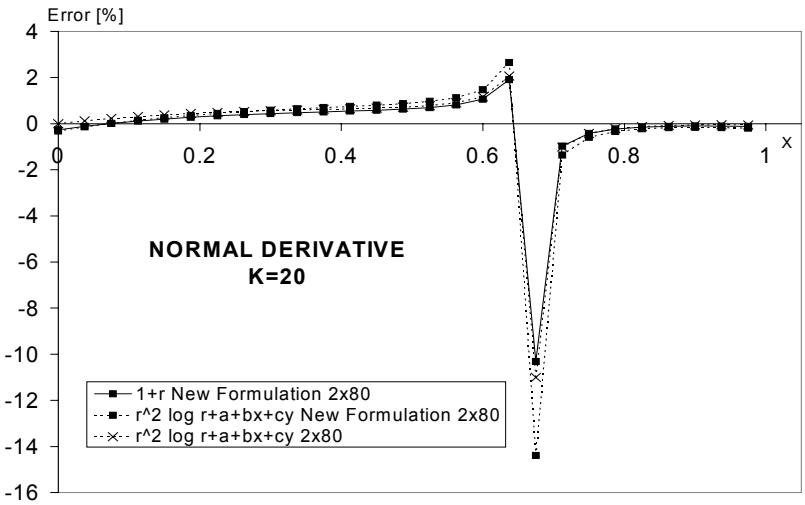


Figure 3.8: Error distribution for the: (a) potential and (b) normal derivative for W=0.2, k=20, mesh = 4×20, CDRMN not used, $V_{x\ max}$= -13.4 (the maximum errors for the code 1 were 141 and 922% for the potential and normal derivative, respectively).



(a)



(b)

Figure 3.9: Error distribution for the: (a) potential and (b) normal derivative for W=0.025, k=20, mesh = 2×80, CDRMN not used, $V_{x\ max}$= -13.4 (the maximum errors for the code 1 were 147 and 960% for the potential and normal derivative, respectively).

Note that as the velocity field is divergent, having a stagnation point at x = 0.83, 0.33 and 0.16 for k equal to 40, 20 and 10, respectively. In the middle of the domain, the analytical solution for potential drops from 300 to very low values, being the drop higher as the reaction constant, k, increases, because the flow velocity increases with k, being the flow directed toward the boundaries. The numerical results for the potential show the highest errors in the middle of the domain where the potential has the lower values. This is because the definition of the relative error, which was defined in eqn. (3.37). As the order of magnitude of the absolute error, that is, the

result of the code minus the result of the analytical solution, keeps constant along the domain, when it is divided by the analytical solution the highest values of relative errors appear where the analytical solution is near zero. In all the results for the normal derivative there exists a jump in the relative error after which the relative error changes sign. The position of this jump depends on the point where the analytical solution for normal derivatives crosses the x-axis, which is x = 0.59, 0.67 and 0.84 for k equal to 40, 20 and 10, respectively. Again, the jump and the change of sign are due to the definition (3.37) of the relative error. When moving along the x axis the analytical values of normal derivatives are negatives with decreasing absolute value until they cross the axis and become positive; and as the analytical value is in the denominator of (3.37) and the absolute error keeps its order of magnitude, the jump and the change of sign appear.

From the examples shown, it can be seen that the results from codes 2 to 4 contain errors that are of the same order of magnitude. Using $1+R$ in combination with (3.9) to represent the first order PDs produces large errors and therefore should not be used for this type of applications. When $1+R$ is used in combination with the new formulation, equations (3.19) and (3.20), the scheme produces a reduction of the error of one or two orders of magnitude. The approximation (3.9) provides a good estimation of first order PDs in 2D when ATPS is used. The new formulation when combined with ATPS does not produce an improvement of the results. Though first order PDs of the ATPS do not introduce singularities, representing the PDs using DRM introduces additional errors due to the DRM approximation and therefore it was expected that the new formulation would perform better for the ATPS as well. One possible explanation of why the new formulation does not give improvements in representation of the first order PDs for ATPS could be that the new formulation calculates the PDs as a function of the normal derivatives, which are, in the examples shown above, calculated with lower accuracy than the potentials.

The figures show that the magnitude of the errors increases as k increases. For comparison of the accuracy of the DRM-MD formulation with the accuracy of the DRM with single domain, the BEM formulation with domain integration and with Galerkin and Taylor-Galerkin finite element method we refer to [21]. When figures

3.5 and 3.9 are compared it is evident that refining the mesh can reduce the errors, which verifies the convergence of the method. A comparison of figures 3.4 and 3.7 and figures 3.5 and 3.8 shows that the results improve when DRM nodes are used. The last observation is not surprising, as the presence of DRM nodes improves the ability of the approximation function to represent the particular solution of governing equation.

3.4.2 Non-linear case: a 1D advection-diffusion equation governing the flow of a compressible fluid in a porous media

In order to test the representation of the second order PDs using the new scheme, the following example is considered. A set of six DRM-MD codes to solve 2D advection-diffusion problems of a compressible fluid were compared. These codes solve the equation

$$D\nabla^2 u + K\vec{\nabla}\cdot\left(u\vec{\nabla}u\right) + p = 0 \tag{3.38}$$

where

D is the coefficient of dispersion

p represents the source or the sink term

K represents the permeability of the porous media

Before introducing the features of the codes let us expand the second term in (3.38)

$$D\nabla^2 u + K\left(\vec{\nabla}u\right)\cdot\vec{\nabla}u + Ku\nabla^2 u + p = 0 \tag{3.39}$$

The governing equation can be re-written in terms of the Laplacian operator in two different ways

$$\nabla^2 u = \frac{1}{D+Ku}\left\{-K\left(\vec{\nabla}u\right)\cdot\vec{\nabla}u - p\right\} \tag{3.40}$$

or

$$\nabla^2 u = \frac{1}{D}\left\{-K\left(\vec{\nabla}u\right)\cdot\vec{\nabla}u - Ku\nabla^2 u - p\right\} \tag{3.41}$$

In order to apply the DRM to these equations, an iterative procedure, which requires an *a priori* known value of u is used. Therefore the DRM can be applied on the following equations

$$\nabla^2 u = \frac{1}{D + K\widetilde{u}} \left\{ - K\left(\vec{\nabla}\widetilde{u}\right) \cdot \vec{\nabla} u - p \right\} \tag{3.42}$$

or

$$\nabla^2 u = \frac{1}{D} \left\{ - K\left(\vec{\nabla}\widetilde{u}\right) \cdot \vec{\nabla} u - K\widetilde{u}\nabla^2 u - p \right\} \tag{3.43}$$

where $\widetilde{u}$ is the value of u from the previous iteration. Eqn. (3.42) is a better formulation than (3.43) since the second order partial derivatives are incorporated in the left-hand-side Laplacian and they do not need to be explicitly calculated. But equation (3.43) is used here to test the accuracy of the second order PDs when represented using the new scheme.

The features of the codes used are shown below:

Code A: solves (3.42) using the formulation (3.9) for representing the first order PDs and ATPS as interpolation function.

Code B: solves (3.43) using the formulation (3.9) for representing the first order PDs and the formulation (3.22) for representing the second order PDs and ATPS as interpolation function.

Code C: solves (3.43) using the formulation (3.9) for representing the first order PDs, the formulation (3.14) for representing the second order PDs and ATPS as interpolation function.

Code D: solves (3.43) using the formulation (3.9) for representing the first order PDs, the formulation (3.10) for representing the second order PDs and ATPS as interpolation function.

Code E: solves (3.43) using the formulation (3.9) for representing the first order PDs, the formulation (3.10) for representing the second order PDs and MQ as interpolation function.

Code F: solves (3.43) using the formulation (3.9) for representing the first order PDs and the formulation (3.10) for representing the second order PDs. Two different interpolation functions are used: $1+R^3$ for representing the second order PDs, following the idea of Partridge *et al*. [2], and ATPS for the remaining part.

The implementation of the procedures under analysis results in the following systems of equations:

Code A

$$(\mathbf{H}-\mathbf{T1})\boldsymbol{u}-\mathbf{G}\boldsymbol{q}=-\mathbf{S1}\,p \tag{3.44}$$

Where

$\mathbf{S1}$ is the matrix of components $s1_{ij}=\dfrac{s_{ij}}{D+K\tilde{u}_j}$

$$\mathbf{T1}=\mathbf{S1}\left[\mathbf{V1}_{\boldsymbol{x}}\frac{\partial\mathbf{F}}{\partial x}\mathbf{F}^{-1}+\mathbf{V1}_{\boldsymbol{y}}\frac{\partial\mathbf{F}}{\partial y}\mathbf{F}^{-1}\right] \tag{3.45}$$

$\mathbf{V1}_x$ is the matrix of velocities in the x-direction, which components are $v1_{xij}=-K\sum_{k=1}^{J}\sum_{l=1}^{J}\dfrac{\partial f_{il}}{\partial x}f_{lk}^{-1}\tilde{u}_k$ when $i=j$, and zero anywhere else;

$\mathbf{V1}_y$ is the analogous for the y direction

Code B

$$(\mathbf{H}-\mathbf{T2})\boldsymbol{u}-\left(\mathbf{G}^b+\frac{\mathbf{S}}{D}\mathbf{Z1}^a\right)\boldsymbol{q}^b-\left(\mathbf{G}^a+\frac{\mathbf{S}}{D}\mathbf{Z1}^b\right)\boldsymbol{q}^a=-\frac{\mathbf{S}}{D}p \tag{3.46}$$

$$\mathbf{T2}=\frac{\mathbf{S}}{D}\left[\mathbf{V1}_{\boldsymbol{x}}\frac{\partial\mathbf{F}}{\partial x}\mathbf{F}^{-1}+\mathbf{V1}_{\boldsymbol{y}}\frac{\partial\mathbf{F}}{\partial y}\mathbf{F}^{-1}\right] \tag{3.47}$$

$\mathbf{Z1}^b$ and $\mathbf{Z1}^a$ are respectively the matrixes of components

$$z1_{ij}^{b} = -K\widetilde{u}_i\left(\sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial y}f_{kj}^{-1}\frac{n_{x\,j}^{b}}{J_j} - \sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial x}f_{kj}^{-1}\frac{n_{y\,j}^{b}}{J_j}\right) \tag{3.48}$$

and

$$z1_{ij}^{a} = -K\widetilde{u}_i\left(\sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial x}f_{kj}^{-1}\frac{n_{y\,j}^{a}}{J_j} - \sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial y}f_{kj}^{-1}\frac{n_{x\,j}^{a}}{J_j}\right) \tag{3.49}$$

respectively.

Code C:

$$\left(\mathbf{H}-\mathbf{T2}-\mathbf{W1}\right)\boldsymbol{u}-\mathbf{G}\boldsymbol{q} = -\frac{\mathbf{S}}{D}\boldsymbol{p} \tag{3.50}$$

Where

W1 is the matrix of components

$$w1_{ij} = -K\widetilde{u}_i\left[\sum_{l=1}^{J}\left(\sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial x}f_{kl}^{-1}\sum_{k=1}^{J}\frac{\partial f_{lk}}{\partial x}f_{kj}^{-1}\right)+\sum_{l=1}^{J}\left(\sum_{k=1}^{J}\frac{\partial f_{ik}}{\partial y}f_{kl}^{-1}\sum_{k=1}^{J}\frac{\partial f_{lk}}{\partial y}f_{kj}^{-1}\right)\right] \tag{3.51}$$

Code D, E and F:

$$\left(\mathbf{H}-\mathbf{T2}-\mathbf{W2}\right)\boldsymbol{u}-\mathbf{G}\boldsymbol{q} = -\frac{\mathbf{S}}{D}\boldsymbol{p} \tag{3.52}$$

Where

W2 is the matrix of components

$$w2_{ij} = -K\widetilde{u}_i\left(\sum_{k=1}^{J}\frac{\partial^2 f_{ik}}{\partial x\partial x}f_{kj}^{-1}+\sum_{k=1}^{J}\frac{\partial^2 f_{ik}}{\partial y\partial y}f_{kj}^{-1}\right) \tag{3.53}$$

These codes were applied to 1-D multi-layer problems with boundary conditions $q(x=0)=(\partial u/\partial n)_{x=0} = 0$ and $u(x=L) = U_L$. The analytical solution of this problem is given in [35]. The domain was a rectangle of length $L = 12$ in the x direction and

width $W = 6$ in the y direction. In order to produce equivalent 1D results from the 2D codes, the following boundary conditions were used

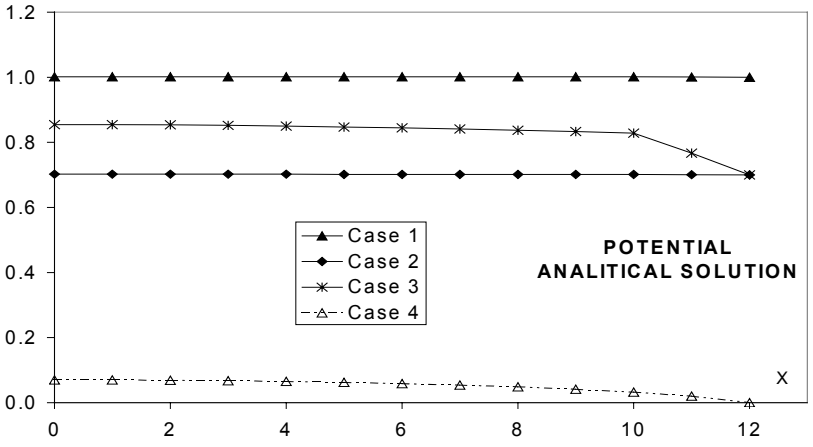
$$q(0,y) = 0 \text{ and } u(L,y) = U_L$$

$$\left.\frac{\partial u}{\partial n}\right|_{y=W/2} = \left.\frac{\partial u}{\partial n}\right|_{y=-W/2} = 0 \tag{3.54}$$

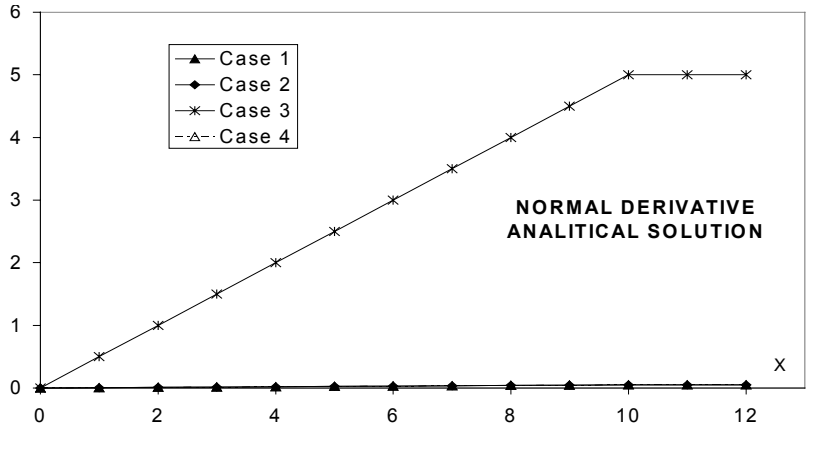
The vector $\tilde{\boldsymbol{u}}$ in the first iteration was set equal to zero for those nodes were the potential was not defined as boundary condition. All the examples used a mesh of 6×12 subregions (six subdivisions in the y direction and twelve subdivisions in the x direction) with no CDRMN. The subdomains were rectangular and linear elements were used within each subdomain. Likewise the previous codes, these ones used a direct Gauss elimination with pivoting solver. The domain considered had three layers of different properties except for the Case 4, which had two layers. Details of the four cases analysed are shown in Table 3.1. Figure 3.10 shows the analytical solution for the potential and normal derivative for the four cases studied. The relative errors for the potentials and normal derivatives for the Cases 1 to 4 are presented in figures 3.11 to 3.14, respectively.

Table 3.1: Description of the properties of the media and fluid used in the four cases studied

	Layer1				Layer2				Layer3				U_L
	Thickness	D	K	p	Thickness	D	K	p	Thickness	D	K	p	
Case1	5	1.5	1000	0.005	5	1.5	1200	0.005	2	1.5	100	0.0	**1.0**
Case2	5	1.5	1000	0.005	5	1.5	1200	0.005	2	1.5	100	0.0	**0.7**
Case3	5	1.5	1000	**0.5**	5	1.5	1200	**0.5**	2	1.5	100	0.0	0.7
Case4	10	1.5	100	0.005	2	1.5	100	0.0	-	-	-	-	0.0



(a)



(b)

Figure 3.10: Analytical solutions for the: (a) potential and (b) normal derivative, for the four examples tested.

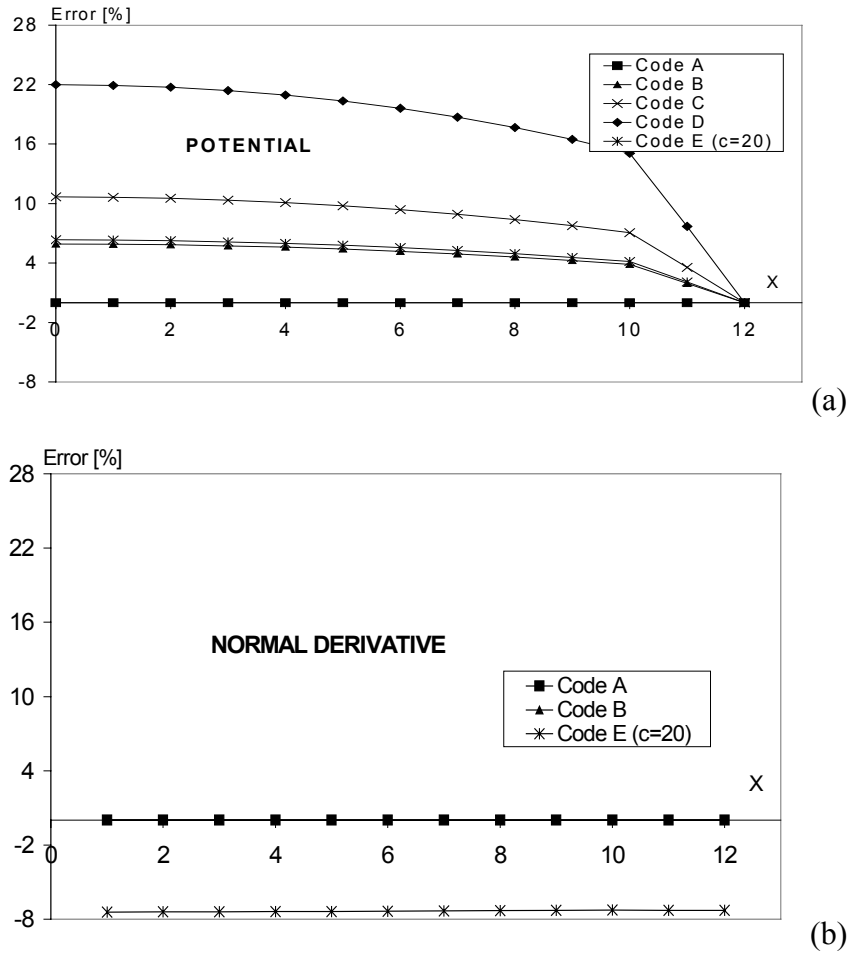


Figure 3.11: Error distribution for the: (a) potential and (b) normal derivative for the Case 1. Code F did not converge. Errors for the normal derivative for codes C and D were 14 and 260%, respectively.

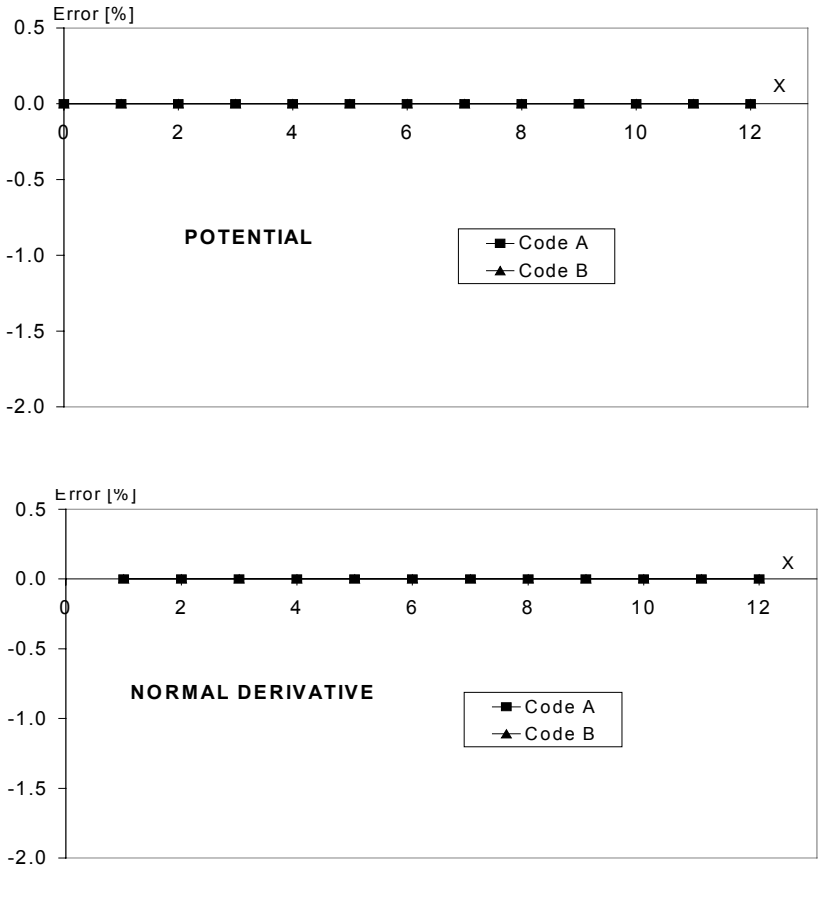


Figure 3.12: Error distribution for the: (a) potential and (b) normal derivative for the Case 2. Codes C, D, E and F did not converge.

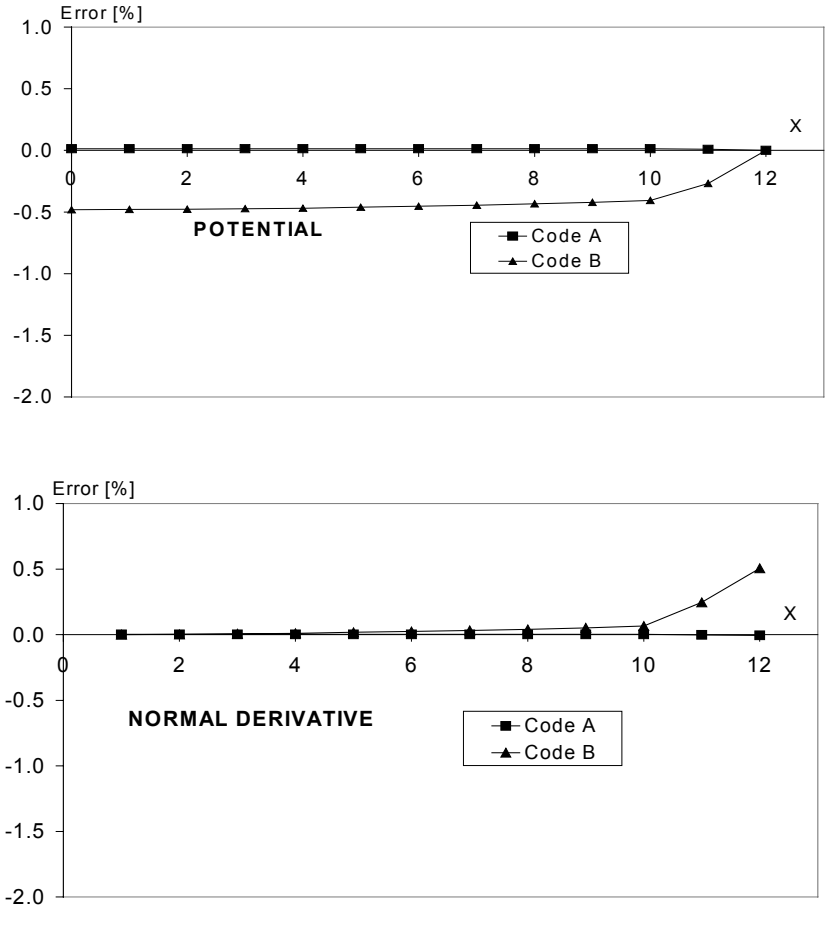


Figure 3.13: Error distribution for the: (a) potential and (b) normal derivative for the Case 3. Codes C, D, E and F did not converge.

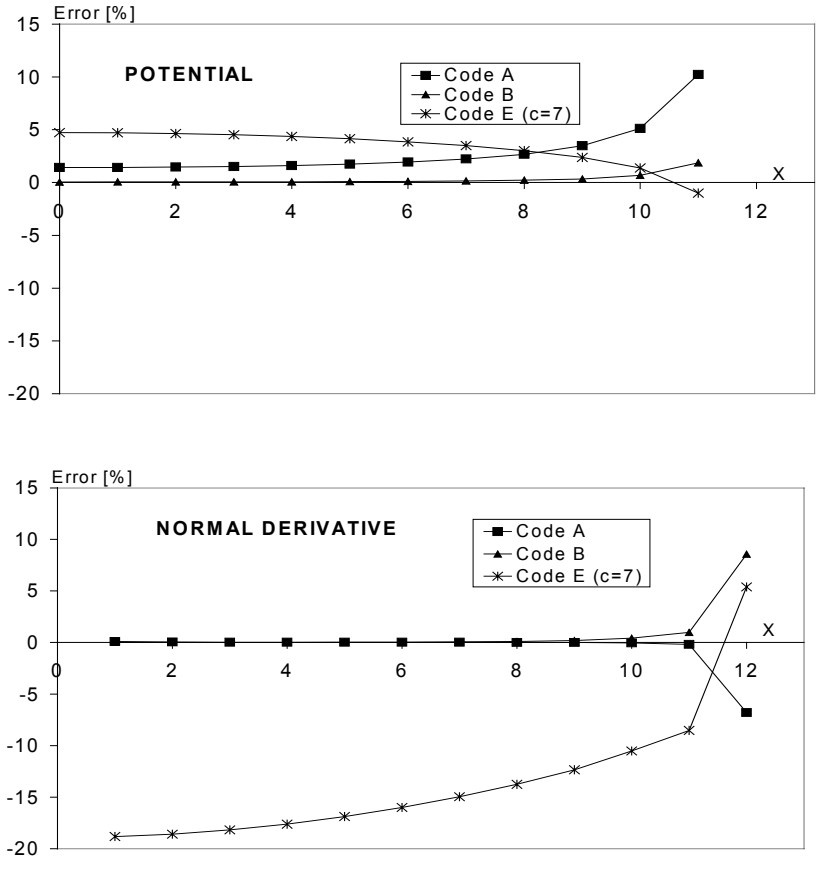


Figure 3.14: Error distribution for the: (a) potential and (b) normal derivative for the Case 4. Codes C, D and F did not converge.

The results show the advantage of formulation (3.22) in respect to both, formulation (3.10) and the new formulation (3.14) as a way of representing second order partial derivatives.

As it was expected, Code A showed the best performance, since it did not deal with the second order PDs as they were incorporated in the Laplacian on the left-hand-side of (3.42). Code B solved eqn. (3.43), which included the new DRM representation of the second order PDs, and converged in all four cases. This indicates that the iterative scheme used to approach the non-linearity of eqn. (3.43) works as long as the representation of second order partial derivatives is done with enough accuracy. Codes A and B produce errors of the same order of magnitude indicating that the new DRM representation of the second order PDs did not produce errors of higher order of magnitude than the rest of the code.

It was expected that Code D, which uses formulation (3.10) and ATPS, would perform poorly because the second order derivative of the 2D-ATPS is singular at $R = 0$. The results are in agreement with this expectation. Code D converged only in the simplest case, which is Case 1, and produced large errors.

Code C, which in regard with Code B differs only in the way of representing second order partial derivatives (with formulation (3.14)), performed almost as poorly as Code D even though formulation (3.14) does not introduce any singularity when using ATPS.

Results of Code E, which has similar features than Code D except by the fact that it uses MQ instead of ATPS and therefore is free of singularities, shows that the problem is the formulation (3.10) rather than the approximation function. It converged only in the simplest case.

Finally, Code F, which is similar to Code D but uses $1 + R^3$ as approximation function in the formulation (3.10), did not converge even in the simplest case. Considering that the only difference in regard with Code B is the way in which the second order partial derivatives are represented and that there are no reported results proving that this representation works in the kind of examples that have been approached in this section, it is concluded that this representation does not represent the second order partial derivates with enough accuracy as to allow the convergence of the iterative scheme.

Code E deserves a further remark. For every case studied, approximately 20 different values of the shape parameter c were tested in order to choose a suitable one, see figure 3.15. The criterion used was that convergence should be obtained in the lowest number of iterations. Apart from being a time consuming procedure, figure 3.16 shows that the procedure is not always accurate. Note the oscillation in the error for c equal 6, 7 and 8.

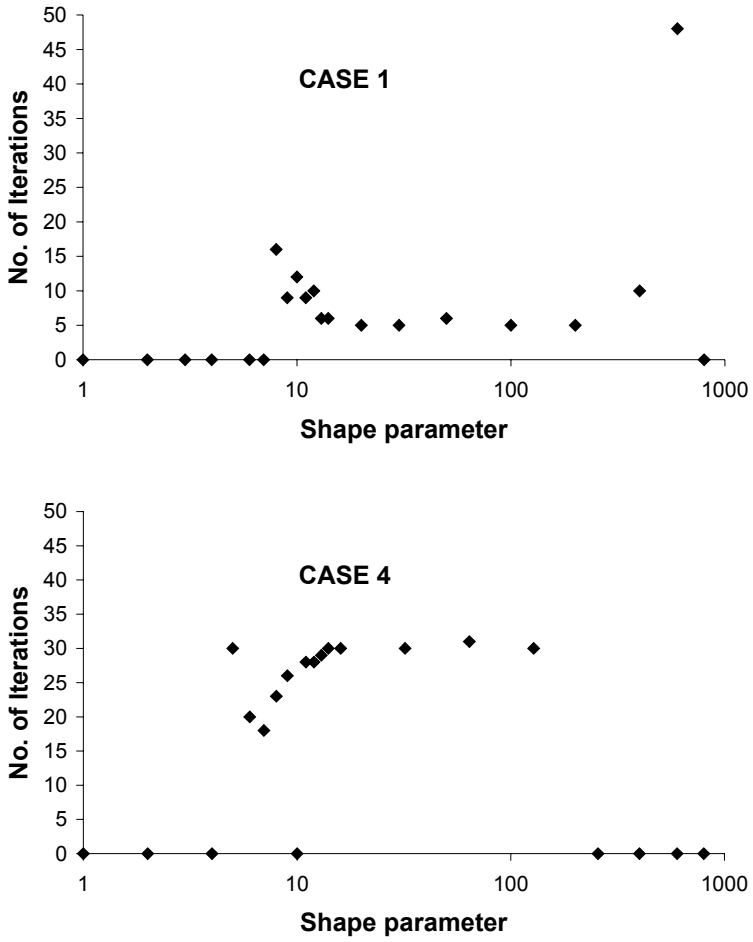


Figure 3.15: Different shape parameters *c* tested for the Cases 1 and 4, for which the code E converged. The dots on the *x*-axis indicate no convergence. For the Cases 2 and 3 a similar number of shape parameters were tried but convergence was not obtained in any case.

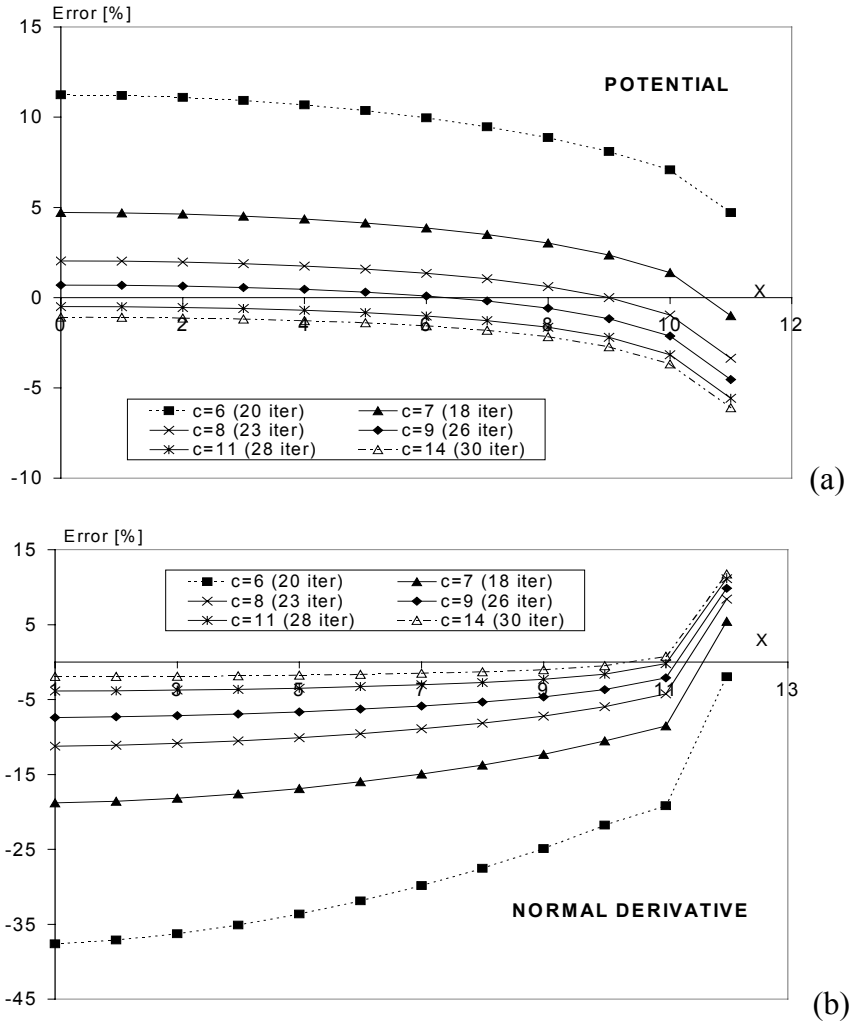


Figure 3.16: Error distribution for the: (a) potential and (b) normal derivative produced by code E when solving the Case 4 using different shape parameters *c* of the MQ *f* function. The number of iterations needed for each case to converge is shown in brackets.

3.5. Conclusions of the chapter

In this chapter, a new way of representing partial derivatives (PDs) in the DRM was proposed. This formulation can be implemented only for corner nodes where two boundary elements join under angle different from π and therefore is suitable for meshes with large proportion of corner nodes or for implementations where the large number of subdomains is part of the implementation strategy (DRM-MD). It is expected that the accuracy of the approach would deteriorate as the corner angle

approaches π. In the internal DRM nodes the classical formulation for partial derivatives must be used. When the scheme was implemented in the DRM-MD with internal DRM nodes, the results did not deteriorate.

The new formulation reduces the order of the PDs of the DRM approximation function by one in respect to the order of the PDs of the DRM approximation function when the classical DRM representation of PDs is used. In this way the introduction of the singularities at the nodes due to the PDs of the DRM approximation function can be avoided. However, the formulation did not improve the accuracy in the 2D case when compared to first PD representation using the ATPS, but it improved the accuracy when the ATPS were used to represent second PDs as the second PDs of the ATPS introduce singularities. The use of some other RBFs, which do not introduce singularities when higher derivatives are involved, e.g. $r^4 \log r$, may show good results, but such functions were not tested as the ATPS have shown to be the best option in the DRM-MD problems that do not involve higher derivatives than first order.

A number of linear and non-linear 2D case studies are shown where different DRM approximations are used for first and second order partial derivatives using four different radial basis functions. In all cases where PDs of the approximation function introduce singularities, the new formulation improved the accuracy of the solution by at least one order of magnitude.

Chapter 4

3D implementation

4.1 Introduction

In this chapter, the implementation of 3D DRM-MD codes is introduced and discussed. Preliminary tests on Laplace problems showed maximum errors of 10^{-8} and 10^{-11} % when using continuous and discontinuous elements respectively. These results are not shown here since they are too simple to offer any conclusions.
Numerical examples on the Poisson and the steady-state advection-diffusion equations tested under a variety of schemes, with different radial basis functions, continuity of elements, scaling and number of internal nodes, are presented.

4.2 A 3D DRM-MD implementation for Poisson's problems using discontinuous elements

The codes that are described in this and subsequent sections are intended to be used for solving general problems of arbitrary geometry. Consequently, the domain is subdivided using unstructured tetrahedrons with quadratic shape functions to represent the geometry of every subdomain, which are very flexible when used to discretize complicated geometries. Ten nodes that are called geometrical nodes describe every tetrahedron. Every tetrahedron is made up of four triangular faces, each of which has a mesh point at the corner and at the mid-side position (figure 4.1).

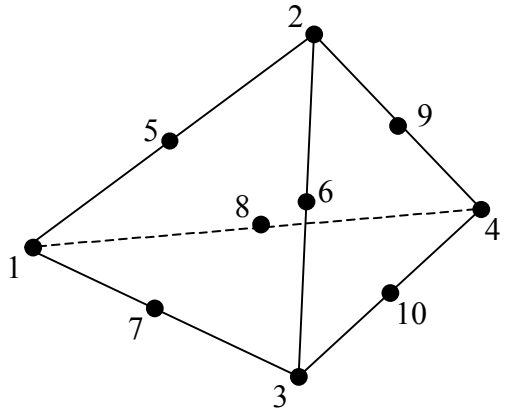


Figure 4.1: tetrahedral subdomain. The geometrical nodes are numbered from a SPV.

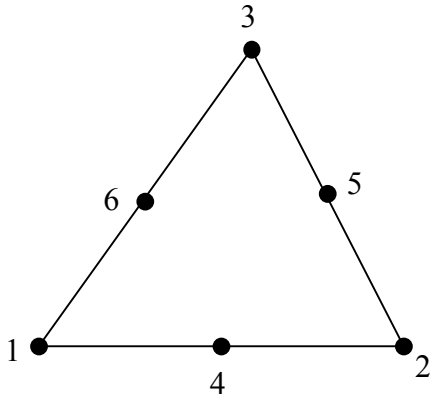


Figure 4.2: triangular boundary element. The geometrical nodes are numbered from an EPV.

At this point it is convenient to introduce a few conventions for a better description. Numeration can be described as belonging to a given side of a given tetrahedron. Let us call this description as Element Point of View (EPV). Subdomain Point of View (SPV) is a description that sees the node as belonging to a given subdomain of the mesh, while Mesh Point of View (MPV) considers the node as a part the mesh. For instance, a given node can be at the same time the node 6 (of a given tetrahedron side) from an EPV, node 8 (of a given subdomain) from a SPV and node 2648 (of the mesh) from a MPV.

The elements used in the code described in this section are quadratic, discontinuous, triangular boundary elements. As the triangular elements are quadratic, each of them is defined by 6 geometrical nodes (figure 4.2). The relative position between nodes from SPV and EPV, which is displayed in figures 4.1 and 4.2, is always the same. Table 1 shows the correspondence between the number of node from the EPV and the SPV.

Table 4.1: correspondence between the number of node from the EPV and the SPV.

Node from an EPV	Node from a SPV			
	Side I	Side II	Side III	Side IV
1	1	4	4	4
2	3	2	3	1
3	2	3	1	2
4	7	9	10	8
5	6	6	7	5
6	5	10	8	9

The BEM implementation for triangular elements used in the codes presented in this chapter follows the treatment proposed by Do Rêgo Silva [36]. As he described in detail the implementation, only general aspects will be mentioned here. Over the discontinuous triangular elements, 6 nodes of freedom, on which the eqn. (2.12) is applied, are distributed according to figure 4.3. At every node of freedom there are two variables, the values of which can be unknown or can have been specified among the boundary conditions: the potential and the normal derivative. Inside every tetrahedron, internal DRM nodes can be added. For the sake of simplicity, the code that is described here has no internal DRM nodes, being the total number of freedom nodes equal to 24 (SPV).

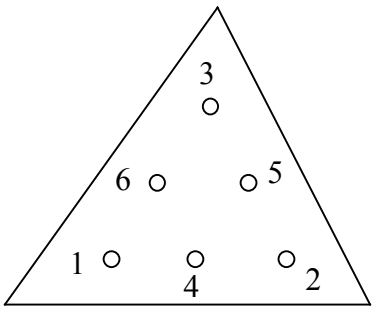


Figure 4.3: nodes of freedom on a discontinuous triangular boundary element. The nodes are numbered from an EPV.

The flow chart of a DRM-MD code to solve problems that are governed by the Poisson's equation (2.21) has been given in figure 2.3. Let us see now some features of every subroutine for this particular case.

4.2.1 Input data

The input data file needs to be generated by a preprocessor due to the large amount of information involved even in the simplest 3D meshes. Here it was successfully used GID™ with minor customisations. The input file has the following information: total number of subdomains, total number of geometrical nodes, length scale factor, list of geometrical nodes coordinates, list of conectivities, list of geometrical nodes that are in the boundary, boundary conditions and list of values of the sink term, *d*.

The geometrical nodes coordinates are given and read from a MPV. The conectivities of each subdomain is a sequence of ten numbers that identify the ten geometrical nodes defining the subdomain. The list of geometrical nodes that are in the boundary is given and read from a MPV. The boundary conditions are defined on the sides of subdomains. They are given as a list of four numbers which identify: subdomain (MPV), side of the subdomain (SPV), type of boundary condition (whether it is a potential or a normal derivative) and value of the boundary condition. Finally, *d* values are assigned to subdomains.

4.2.2 Set geometrical information

4.2.2.1 Order mesh information: Table 4.2 shows different auxiliary arrays that are set up in this process. Although the code has a sobroutine for this task, some of the arrays of Table 4.2 have been already set up during the input process because they were needed to read some parts of the input file. The arrays are listed in the order in which the code sets them up. The arrays of Table 4.2 depend on the mesh; setting them up requires loops to go through the mesh extracting the information needed in every particular case. Apart from these, there are other auxiliary arrays that contain information from the SPV and EPV. For instance, there is the array elv_{ij}, which gives the geometrical node number, from a SPV, of the node ‘j’ of the side ‘i’ of any tetrahedron. As they always have the same values independently of the mesh, they will not be explicitly mentioned in this thesis. During this process, the freedom nodes are generated and labelled.

Table 4.2: Auxiliary arrays that depend only on the mesh and geometry of the problem.

ARRAY	DESCRIPTION
$nshc1_{ij}$	Number of the cell sharing the side 'j' of the cell 'i'
$nshc2_{ij}$	Side (SPV) of the cell $nshc1_{ij}$ that shares the side 'j' of the cell 'i'.
$xgn1_{ij}$ $ygn1_{ij}$ $zgn1_{ij}$	Coordinates (MPV) of the geometrical node 'j' (SPV) of the cell 'i'
xnf_{ij} ynf_{ij} znf_{ij}	Coordinates (MPV) of the freedom node 'j' (SPV) of the cell 'i'
ne_i	Array indicating what rows of the $\mathbf{A}^{\mathbf{MD}}$ matrix correspond to the equations of the subdomain 'i': they occupy rows ne(I)+1 to ne(I)+24
nf_{ij}	Number of freedom node (MPV) of the freedom node 'j' (SPV) of the cell 'i'
$normverX_{ij}$ $normverY_{ij}$ $normverZ_{ij}$	Components of the outward (SPV) normal versor at the freedom node 'j' (SPV) of the cell 'i'

4.2.2.2 Calculate **H** *and* **G**: in this process, the full integral representation formula is applied on each subdomain producing $\mathbf{H}_{24x24}$ and $\mathbf{G}_{24x24}$ matrices. The triangular elements using here belong to a family of quadrilateral and triangular elements proposed by Do Rêgo Silva [36]. In order to have the capability of using both quadrilateral and triangular elements, the code applies a transformation of coordinates that maps the triangular elements into quadrilateral ones. When the integrals are regular, the standard Gaussian quadrature is used with a fixed 40×40 Gauss point number. The singular integrals are approached using polar coordinates, Gaussian quadrature and the self-adaptative coordinate transformation proposed by Telles [37]. Telles´ transformation is also used for regular integrals because it prevents the failure of the code when the source point is too near the collocation point. The Gauss point number for singular integrals was 32 for the radius and 40 for the angle. The diagonal elements of **H** are calculated through the rigid body motion hypothesis.

In the code, **H** and **G** matrices are stored in two h_{ijk} and g_{ijk} arrays, where the subindex 'i' stands for subdomain number.

4.2.2.3 Calculate **S**: a $\mathbf{S}_{24x24}$ is obtained for every subdomain. Similarly as the DRM, the DRM-MD is highly dependant on the approximation function appearing in equation (2.5). In section 4.6 a study on the performance of the method in 3D codes using 10 different Radial Basis Funtions shall be presented. The preferred choice of the authors is the Duchon's Augmented Thin Plate Splines. However, any other approximation function can be used whithout modifying a single sentence of a process other than this one. Similarly to the BEM matrices, in the code the **S** matrices are stored in a s_{ijk} array.

4.2.3 Set boundary conditions and NCOL

In the same way as 'order mesh information' process, this process sets a group of auxiliary arrays up (see Table 4.3) by going throughout the mesh extracting the information needed in every particular case. Note that there is a significant conceptual difference between both processes: the former process analyses only geometrical information while this one requires in addition information on the boundary conditions.

4.2.4 Assemble system of equations

Here the $\mathbf{A}^{\mathbf{MD}}$ matrix and the right-hand-side vector of eqn. (2.17) are assembled. As both the matrix and the vector are made up with the coefficients arising when applying eqn. (2.14) to every subdomain, the process does an outer loop on subdomains and two inner loops on collocation and source nodes (SPV), respectively. For each pair of source and collocation nodes there are a coefficient multiplying the potential at the collocation node and another one multiplying the normal derivative at the collocation node. The code calculates the value of these coefficients and asks whether they have been specified as boundary condition (this information has been previously stored in **NCOL**). Then, according to the answer, the value of the coefficients are stored in some position of the $\mathbf{A}^{\mathbf{MD}}$ matrix or incorporated into the right-hand side vector. In the former case, the position is given by the **NE** and **NCOL** arrays. Figure 4.4 shows the flow chart of the process.

Once reached this point, there is not too much to say about the remaining part of the code; the system of equations must be solved and the result distributed and given as output. The performance of iterative solvers in 2D DRM-MD codes has been exhaustively tried by Portapila and Power [38, 39]. Based in their work, here different solvers were tried with the Discontinuous Element Code (DEC) that has been presented, and those which performed the best were a direct solver using a Gaussian Elimination algorithm and an iterative solver using a Conjugate Gradient-Normal Residual (CGNR) algorithm [40].

Table 4.3: Auxiliary arrays that depend on the mesh, geometry and boundary conditions of the problem.

ARRAY	DESCRIPTION
$kfnp_{\mathrm{i}}$	Position of the freedom node 'i' (MPV). When it is equal to 1, the node is on the boundary; when it is equal to 0, the node is in between two cells; when it is equal to 2, the node is an internal (SPV) DRM node
c_{ij}	Concentration at the freedom node 'j' (SPV) of the cell 'i'. Only the known values (if any) are initially assigned.
der_{ij}	Normal derivative at the freedom node 'j' (SPV) of the cell 'i'. Only the known values (if any) are initially assigned.
$ncol_{\mathrm{ijk}}$	Column number of the $\mathbf{A}^{\mathbf{MD}}$ matrix assigned to the coefficients of the cell 'i'. The subindex 'j' indicates the collocation point number (SPV). The subindex 'k' indicates that, in the system (2.17), the coefficient multiplies the potential at the node 'j' (SPV) if k = 1, or the normal derivative at the node 'j' if k = 2. If the coefficient multiplies a value that is specified as a boundary condition, $ncol_{\mathrm{ijk}}$ is equal to -1.
$chsign_{\mathrm{ij}}$	In order to apply the matching condition (2.16), a precedence order must be defined between adjacent subdomains. The criterion used here is that the subdomain labelled with the lower number is the master and the other one the slave. $chsign_{\mathrm{ij}}$ stores this information in a format that is convenient for the assembly: it indicates whether the subdomain 'i' is master in regard with the subdomain adjacent to its node 'j' (SPV).
ia_{i}	The matrix of coefficient, $\mathbf{A}^{\mathbf{MD}}$, is stored in Compressed Sparse Row (CSR) format. IA_{i} is an integer array containing the pointers to the beginning of each row in the vector containing the non-zero coefficients of $\mathbf{A}^{\mathbf{MD}}$.

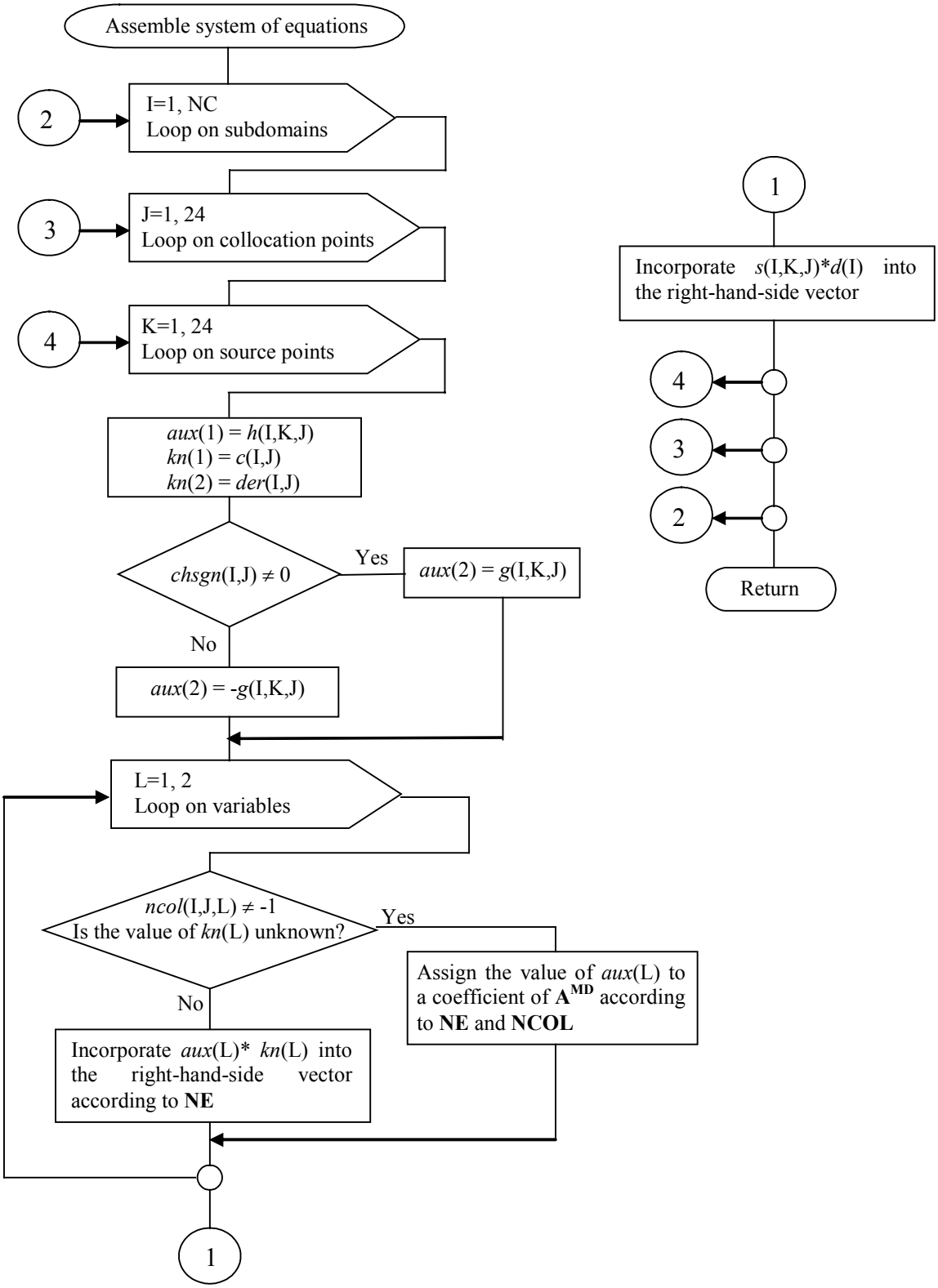


Figure 4.4: Flow chart of the 'assembling system of equations' process of the code for Poisson's problems using discontinuous elements.

4.3 A 3D DRM-MD implementation for Poisson's problems using continuous elements

In this section, a code using tetrahedrons, which sides are quadratic, continuous, triangular boundary elements, is described. In this case the freedom nodes are coincident with the geometrical nodes, as a result of which each tetrahedron produces ten equations, provided there are no internal DRM nodes. This is a noticeable advantage in regard with the discontinuous elements code. Among the relative disadvantages, it can be mentioned that the resulting system of equations is overdetermined, as it was seen in section 2.3, and also continuous elements cannot be used in between subdomains with different properties. In this case discontinuous elements must be used.

Another important factor must be considered when using continuous elements. Eqns. (2.18), (2.19) and (2.20) give the degree of overdetermination, O_v, in a continuous node. O_v must always be greater or equal than zero. According to eqns. (2.18), (2.19) and (2.20) this is possible only when at least one independent condition of coplanarity (in 3D problems) of the interfaces joining the node is existing. When the mesh is structured, that condition can be fulfilled, but in a general case the mesh is unstructured, therefore the condition is not fulfilled, unless the node is in the boundary. Consequently, the eqn. (2.23) was manipulated in such a way that the variables at nodes were potential and **partial derivatives** instead of potential and normal derivatives. The change of variables has been done in the following way. Let us apply eqn. (2.23) to a tetrahedral subdomain and express it in index notation

$$h_{ij}u_j - g_{ik}q_k = s_{ij}d_j \qquad \text{with} \qquad \begin{matrix} 1 \le i \le 10 \\ 1 \le j \le 10 \\ 1 \le k \le 24 \end{matrix} \tag{4.1}$$

Note that the subindex *k* goes from 1 to 24 because there are 24 normal derivatives in a single tetrahedron. At the same time, every normal derivative is the scalar product of the gradient vector and the normal versor:

$$q_k = \left(\frac{\partial u}{\partial x}\right)_k n_{x_k} + \left(\frac{\partial u}{\partial y}\right)_k n_{y_k} + \left(\frac{\partial u}{\partial z}\right)_k n_{z_k} \tag{4.2}$$

By replacing (4.2) into (4.1)

$$h_{ij}u_j - g_{ik}n_{x_k}\left(\frac{\partial u}{\partial x}\right)_k - g_{ik}n_{y_k}\left(\frac{\partial u}{\partial y}\right)_k - g_{ik}n_{z_k}\left(\frac{\partial u}{\partial z}\right)_k = s_{ij}d_j \tag{4.3}$$

In the previous equation, there are twenty-four partial derivatives with respect to x, y and z, respectively. However, some of them are repeated, since there are only ten freedom nodes where the partial derivatives are computed, thus, the products between the elements of the **G** matrix and the components of the normal versors can be assembled in three 10×10 matrices that we will denote $\mathbf{G^x}$, $\mathbf{G^y}$ and $\mathbf{G^z}$. The resulting system of equations is now

$$h_{ij}u_j - g_{ij}^x \frac{\partial u_j}{\partial x} - g_{ij}^y \frac{\partial u_j}{\partial y} - g_{ij}^z \frac{\partial u_j}{\partial z} = s_{ij}d_j \tag{4.4}$$

A further condition arises in order to apply (4.4): it must be at least four subdomains around every node in order to have a closed system of equations. This condition is not always fulfilled in apex nodes that are located in the boundary of the domain (MPV), where, instead, at least one condition of coplanarity can always be found. In those nodes the problem is defined in terms of potential and normal derivatives.

The Continuous Element Code (CEC) is conceptually identical to the one that has been described in the previous section. The structure is the same and the same set of auxiliary arrays is used, though some of them are set up in a different order than in the previous case. The main differences between both codes are:

a) The boundary conditions are specified as potential and partial derivatives, except for apex nodes, for which they are specified as potential and normal derivatives.

b) The size of the **H**, **G** and **S** matrices are 10×10, 10×24 and 10×10, respectively.

c) 'Setting **NCOL**' and the 'assembling system of equations' processes are a bit more complex because every collocation node must be processed in different ways according to whether they are inside the domain (from a MPV) or in the boundary; and if they are in the boundary, a distinction must be done between apex and vertex nodes.

d) The over determined system of the CEC is solved here in a lest-square sense by means of a LSQR algorithm [41], which is algebraically equivalent to applying the symmetric conjugate gradient method to the normalized equations.

4.4 A 3D DRM-MD implementation for advection-diffusion problems

The codes that have been described in the previous sections can be easily adapted to solve steady-state advection-diffusion problems, which governing equation is (3.23). The chief modifications are that the definition of the coefficient multiplying potential (*aux*(1) in figure 4.6) and the matching condition for derivatives are changed. For the discontinuous element code, these mean changing two sentences and deleting one in the assembling subroutine, as it can be seen in figure 4.6.

Figure 4.5 shows the flow chart of the codes; it is common to continuous and discontinuous element codes and it uses the formulation (3.9) for first order partial derivatives, which leads to the resulting system of equations (3.27). Most of the structure is similar to the codes which solve the Poisson's equation. Apart from the assembly process, the following processes need to be slightly modified:

- *Input data*: it has to incorporate the list of coefficients of dispersion, D, reaction constant, k, and porosities n.
- *Set geometrical information*: It has a subprocess more, in which the partial derivatives of the f function are calculated.
- *Distribute results*: because the matching condition for derivatives is different than in the Poisson case.

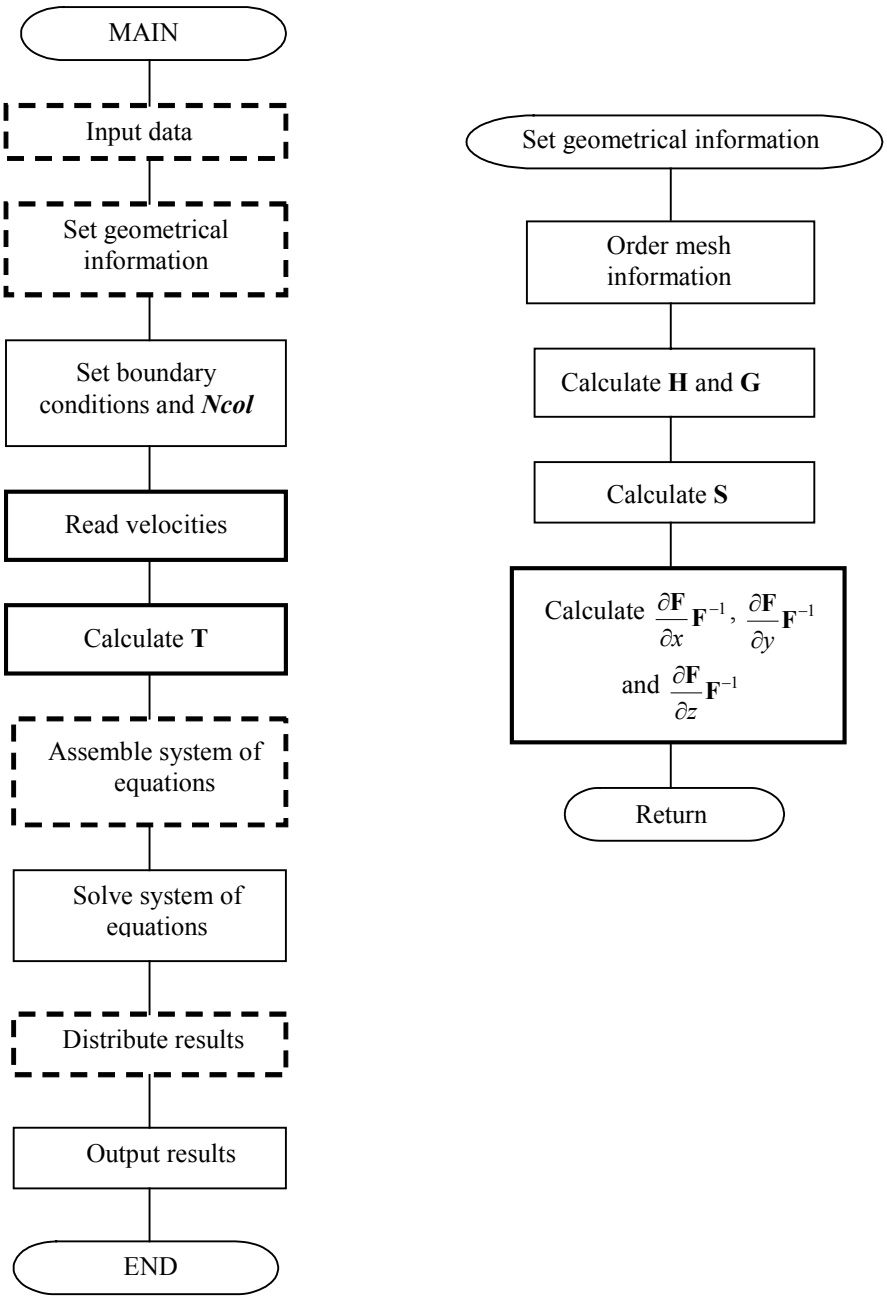


Figure 4.5: Flow chart of a DRM-MD code to solve the steady-state advection-diffusion equation. Dashed line indicates processes that have been modified and bold line indicates new processes in regard with the Poisson´s code.

In addition, as it can be seen in figure 4.5, two new processes have to be incorporated in the main program: ´read velocities´ and ´calculate **T**´. The former process read the velocities from a file and the latter calculates $\mathbf{T} = \frac{\mathbf{S}}{D}\left[\boldsymbol{V}_x \frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1} + \boldsymbol{V}_y \frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1} + \boldsymbol{V}_z \frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1} \right]$. Similarly to the BEM matrices, there is one **T** matrix for every subdomain, the values of which are stored in an array t_{ijk}.

4.5 A convenient arrangement

When working with the codes that have been previously described, it is convenient to split them out in two codes. Given a problem, the first code calculates the BEM and DRM matrices and the auxiliary arrays in order to save them in a file. The second code reads the file produced by the first code and produces the solution of the problem.

Let us suppose that there is a problem under study and it is intended to obtain the solution for a variety of cases with different parameters and values of boundary conditions. Provided that mesh, scale factor and kind of boundary conditions are not modified, this arrangement requires running the first code just once. Considering that the first code is several orders of magnitude more time-consuming than the second one, this scheme is highly convenient. Figure 4.7 illustrates this arrangement for the steady-state advection-diffusion code. All the subroutines, but the three which are in bold line, are the same ones of figure 4.5.

It is possible to use the PREPROCESS subroutine of the advection-diffusion case in codes to solve the Laplace and Poisson's equations, however, it is preferable to use a particular PREPROCESS subroutine for each kind of problem since some problems require the calculation of matrices which are not used in other ones, and in large problems this means a noticeable difference in CPU time.

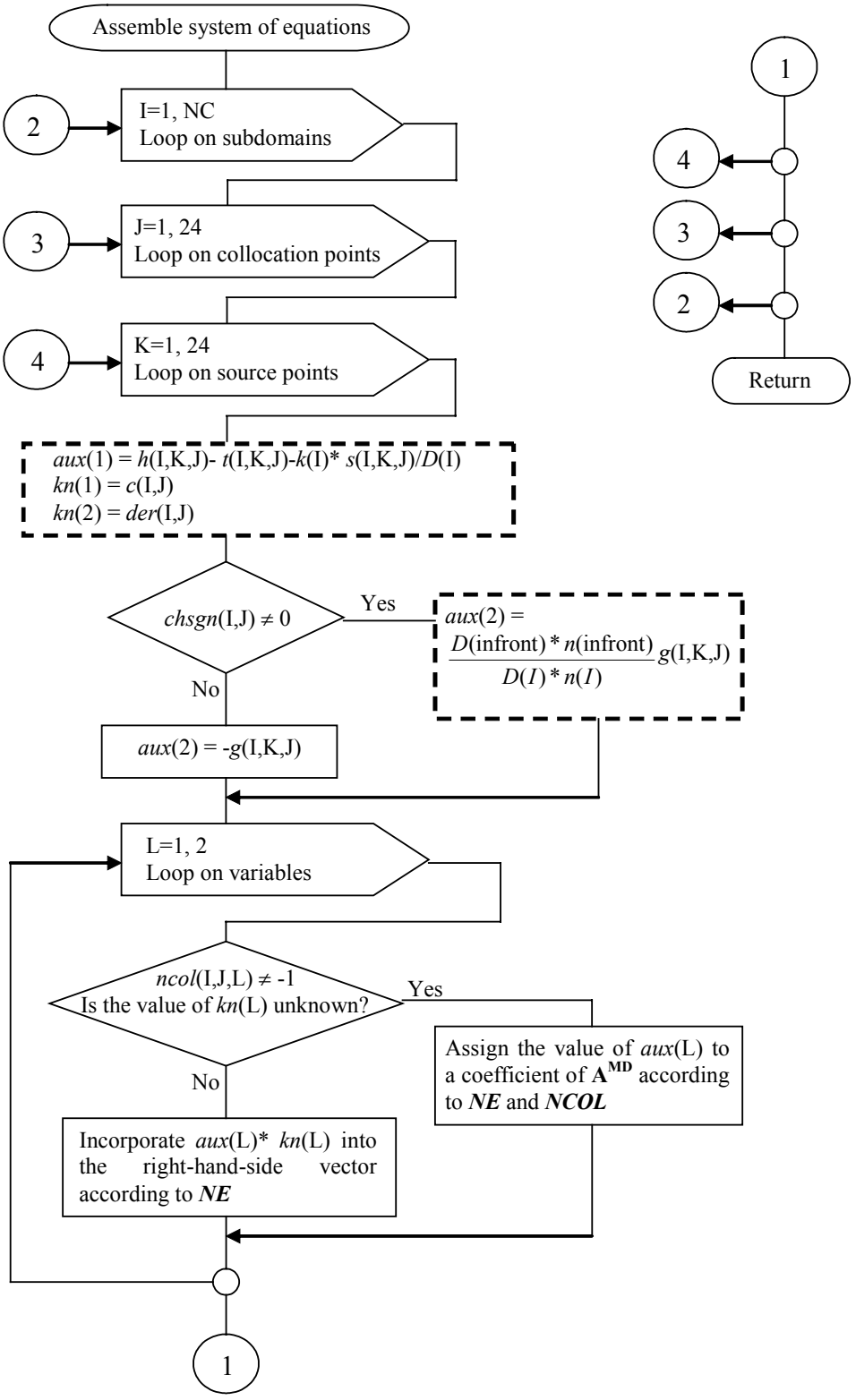


Figure 4.6: Flow chart of the 'assembling system of equations' process of the code for steady-state advection-diffusion equation problems using discontinuous elements.

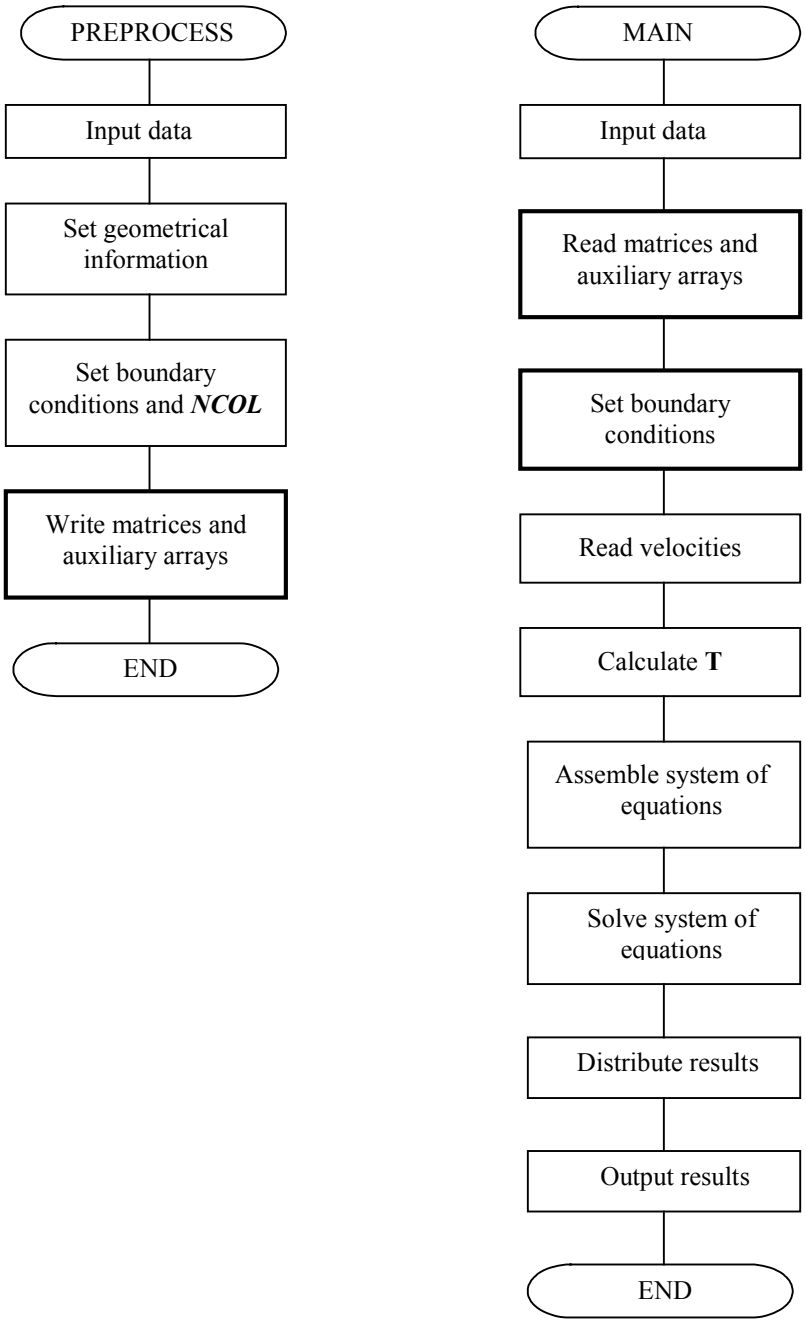


Figure 4.7: Flow chart of the DRM-MD code with pre-processing to solve steady-state advection-diffusion. Processes in bold line are new in regard with the flow chart of figure 4.5.

4.6 Tests on Radial Basis Functions

4.6.1 The tested Radial Basis Functions

The DRM is very sensitive to the approximation function used. Until the 90s, the most popular choice for the function $f(\boldsymbol{y},\boldsymbol{z}^k)$ in the DRM approach was

$$f(\boldsymbol{y},\boldsymbol{z}^k) = 1 + R(\boldsymbol{y},\boldsymbol{z}^k) \tag{4.5}$$

where $R(\boldsymbol{y},\boldsymbol{z}^k)$ is the distance between a pre-specified fixed collocation point $\boldsymbol{z}^k$, and a field point $\boldsymbol{y}$ where the function is approximated, i.e. $R(\boldsymbol{y},\boldsymbol{z}^k) = ||\boldsymbol{y} - \boldsymbol{z}^k||$.

It has been mentioned in section 1.2 that the f function given in (4.5) is a member of a special family of functions called Radial Basis Functions (RBFs), which are related to the theory of multivariate interpolation [42]. Let $\varphi : \mathbb{R}^+ \to \mathbb{R}$ be a continuous function with $\varphi(0) \geq 0$. If $\boldsymbol{z}^k \in \Omega$, then

$$\varphi_k = \varphi(||\boldsymbol{y} - \boldsymbol{z}^k||) \tag{4.6}$$

is called a Radial Basis Function. Micchelli [43] proved that for some additional conditions and when the nodal points are all distinct, the matrix resulting from a radial basis function interpolation is always non-singular. Therefore, as long as the function $b(\boldsymbol{y})$ is regular the coefficients α_k appearing in (2.5) are well defined.

So far, only globally supported RBFs have been used in the DRM-MD. In single domain DRM and in interpolation problems, globally supported RBFs produces dense **F** matrices, which makes difficult the reconstruction of the interpolated function in large problems. In the multidomain approach the size of the **F** matrix is given by the number of freedom nodes of every subdomain, hence the reconstruction of the interpolated function is good no matter how large the problem is. In this section five globally RBFs are tested in both Poisson and advection-diffussion problems.

Compactly supported RBFs (CS-RBF) appeared in the last years as a tool that overcomes the problem of having ill conditioned **F** matrices in large problem, since the contribution to **F** of every source point is limited to those field points which are in a radius equal to the support. As a matter of fact, from the point of view of DRM-MD this feature has no practical effect since the multidomain approach guarantee a fair recovery. But CS-RBFs are unconditionally definite positive and some of them are optimal interpolants. In this section, five CS-RBFs are tried for first time in DRM-MD codes.

The size of the support of CS-RBFs β affects the accuracy of the interpolation. This topic has been discussed within the framework of multiscattered interpolation theory by Floater & Iske [44] and Schaback [9] among others. A small value of β guarantee a safe recovery, but a large one reduces the error at expense of stability. Therefore a balance must be reached between these two factors. In sections 4.6.2 and 4.6.3 several results for every CS-RBF are presented because they have been tried with different supports. The support was set locally in every subdomain as

$$\beta = MaxR \cdot s \tag{4.7}$$

where *MaxR* is the maximum $R(\boldsymbol{y}, \boldsymbol{z}^k)$ of the subdomain and *s* is a scale factor. Every CS-RBF was tested with values of *s* equal to 1.05, 10, 10^2, 10^3, ..., 10^8. For every CS-RBF, sometimes the code fails to converge to the correct solution when *s* is below a certain level due to lack of accuracy in the interpolation, and it always fails when s is above an upper limit due to lack of stability in the interpolation. For instance, for the Poisson's case of section 4.6.2 the code fails when using Wendland-C2 function and values for *s* below 10 or above 10^4. The results when the code does not converge to the correct solution are not displayed.

In the remaining part of this section, the RBFs that are tested in a Poisson's problem (section 4.6.2) and in advection-diffusion problems (section 4.6.3) are listed together with its corresponding particular solution, $\hat{u}(\boldsymbol{y})$, and the normal derivative of the

particular solution, $\hat{q}(\boldsymbol{y})$. In every case these two functions were derived with symbolic mathematics software Mathcad.

a) 1+*R*

In this case the approximation of a particular solution, $\hat{u}(\boldsymbol{y})$, and its normal derivative, $\hat{q}(\boldsymbol{y})$, at a point $\boldsymbol{y}$ are

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left[\frac{R^3(\boldsymbol{y}, \boldsymbol{z}^k)}{12} + \frac{R^2(\boldsymbol{y}, \boldsymbol{z}^k)}{6} \right] \tag{4.8}$$

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left(\frac{R^2(\boldsymbol{y}, \boldsymbol{z}^k)}{4} + \frac{R(\boldsymbol{y}, \boldsymbol{z}^k)}{12} \right) \frac{\partial R(\boldsymbol{y}, \boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} \tag{4.9}$$

b) *R* +P_0

It has been demonstrated that $f = R$ is conditionally positive definite (CPD) of order one [45, 46]. This means that in order to assure a safe reconstruction in (2.7) a polynomial of degree zero must be added. Then

$$b(\boldsymbol{y}) \cong \sum_{k=1}^{J+I} \alpha_k R(\boldsymbol{y}, \boldsymbol{z}^k) + a \tag{4.10}$$

and α_k coefficients must satisfy the condition

$$\sum_{k=1}^{J+I} \alpha_k = 0 \tag{4.11}$$

In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R^3(\boldsymbol{y}, \boldsymbol{z}^k)}{12} + \frac{a}{6}\left(y_x^2 + y_y^2 + y_z^2\right) \tag{4.12}$$

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R^2(\boldsymbol{y}, \boldsymbol{z}^k)}{4} \frac{\partial R(\boldsymbol{y}, \boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} + \frac{a}{3}\left(y_x \frac{\partial y_x}{\partial n^{\boldsymbol{y}}} + y_y \frac{\partial y_y}{\partial n^{\boldsymbol{y}}} + y_z \frac{\partial y_z}{\partial n^{\boldsymbol{y}}} \right) \tag{4.13}$$

c) ATPS: *R* +P_1

Golberg [47] proposed the use of Duchon's Thin Plate Splines (TPS) [48], which in $\mathbb{R}^3$ are

$$\varphi(\boldsymbol{y},\boldsymbol{z}^k) = R(\boldsymbol{y},\boldsymbol{z}^k) \tag{4.14}$$

Duchon's theorem states that TPS are optimal interpolants in the following sense: let $s(\boldsymbol{y})$, $\boldsymbol{y} \in \mathbb{R}^3$, be $C^2(\mathbb{R}^3 - \{0\})$ such that

$$I(s) = \int_{\mathbb{R}^3} \sum_{(i,j)=1}^{3} \left[\frac{\partial^2 s(\boldsymbol{y})}{\partial z_i \partial z_j} \right]^2 \mathrm{d}\boldsymbol{y} \; < \; \infty \tag{4.15}$$

and let $f : \mathbb{R}^3 \to \mathbb{R}$ have values $f(\boldsymbol{z}^k)$, $k = 1,2,...,n\,(n \geq 4)$ on the points $\{\boldsymbol{z}^k\}_{k=1}^{n}$. Then there exist a unique s interpolanting to f at $\{\boldsymbol{z}^k\}_{k=1}^{n}$, and minimizing $I(s)$ if s is of the form

$$s(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k R(\boldsymbol{y},\boldsymbol{z}^k) + a y_x + b y_y + c y_z + d \tag{4.16}$$

where the α_k coefficients must satisfy the condition

$$\sum_{k=1}^{N} \alpha_k = \sum_{k=1}^{N} \alpha_k y_{x_k} = \sum_{k=1}^{N} \alpha_k y_{y_k} = \sum_{k=1}^{N} \alpha_k y_{z_k} = 0 \tag{4.17}$$

TPS functions plus the polynomial of degree one are called the Augmented Thin Plate Splines (ATPS). In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ become

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R^3(\boldsymbol{y},\boldsymbol{z}^k)}{12} + \frac{1}{6}\left(a y_x^3 + b y_y^3 + c y_z^3\right) + \frac{d}{6}\left(y_x^2 + y_y^2 + y_z^2\right) \tag{4.18}$$

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R^2(\boldsymbol{y},\boldsymbol{z}^k)}{4} \frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} + \frac{1}{2}\left(a y_x^2 \frac{\partial y_x}{\partial n^{\boldsymbol{y}}} + b y_y^2 \frac{\partial y_y}{\partial n^{\boldsymbol{y}}} + c y_z^2 \frac{\partial y_z}{\partial n^{\boldsymbol{y}}} \right)$$

$$+ \frac{d}{3}\left(y_x \frac{\partial y_x}{\partial n^{\boldsymbol{y}}} + y_y \frac{\partial y_y}{\partial n^{\boldsymbol{y}}} + y_z \frac{\partial y_z}{\partial n^{\boldsymbol{y}}} \right) \tag{4.19}$$

d) $R^2 \log R + P_1$

This is the ATPS in $\mathbb{R}^2$. In this case the implementation is given by

$$b(\boldsymbol{y}) \cong \sum_{k=1}^{J+I} \alpha_k R^2(\boldsymbol{y}, \boldsymbol{z}^k) \log\left(R(\boldsymbol{y}, \boldsymbol{z}^k)\right) + a y_x + b y_y + c y_z + d \tag{4.20}$$

where the α_k coefficient must satisfy the condition

$$\sum_{k=1}^{N} \alpha_k = \sum_{k=1}^{N} \alpha_k y_{x_k} = \sum_{k=1}^{N} \alpha_k y_{y_k} = \sum_{k=1}^{N} \alpha_k y_{z_k} = 0 \tag{4.21}$$

$\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left[\frac{R^4(\boldsymbol{y}, \boldsymbol{z}^k)}{20} \ln\left[R(\boldsymbol{y}, \boldsymbol{z}^k)\right] - \frac{9R^4(\boldsymbol{y}, \boldsymbol{z}^k)}{400} \right] \frac{R^3(\boldsymbol{y}, \boldsymbol{z}^k)}{12} +$$

$$\frac{1}{6}\left(a y_x^3 + b y_y^3 + c y_z^3\right) + \frac{d}{6}\left(y_x^2 + y_y^2 + y_z^2\right) \tag{4.22}$$

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left[\frac{R^3(\boldsymbol{y}, \boldsymbol{z}^k)}{5} \ln\left[R(\boldsymbol{y}, \boldsymbol{z}^k)\right] - \frac{R^3(\boldsymbol{y}, \boldsymbol{z}^k)}{25} \right] \frac{\partial R(\boldsymbol{y}, \boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} +$$

$$\frac{1}{2}\left(a y_x^2 \frac{\partial y_x}{\partial n^{\boldsymbol{y}}} + b y_y^2 \frac{\partial y_y}{\partial n^{\boldsymbol{y}}} + c y_z^2 \frac{\partial y_z}{\partial n^{\boldsymbol{y}}} \right) + \frac{d}{3}\left(y_x \frac{\partial y_x}{\partial n^{\boldsymbol{y}}} + y_y \frac{\partial y_y}{\partial n^{\boldsymbol{y}}} + y_z \frac{\partial y_z}{\partial n^{\boldsymbol{y}}} \right) \tag{4.23}$$

e) $R^2 \log R$

It can be found in any survey on RBFs that this function is conditionally definite positive of order two, that is, it needs a polynomial of order one to assure a safe reconstruction. This function will be tried in order to show how the solution is less accurate without the polynomial.

f) Wendland-C0

Wendland [10] proposed several positive definite piecewise polynomial CS-RBFs that have minimal degree for prescribed smoothness and space dimension.

Let us call Wendland-C0 the function defined by

$$f(\boldsymbol{y},\boldsymbol{z}^k)=\left(1-\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^2 \quad \text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.24}$$

and

$$f(\boldsymbol{y},\boldsymbol{z}^k)=0 \quad \text{when} \quad \frac{R}{\beta}>1 \tag{4.25}$$

where β is an scaling parameter to control the support.

$\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{1}{60}R^2(\boldsymbol{y},\boldsymbol{z}^k)\frac{\left(3R^2(\boldsymbol{y},\boldsymbol{z}^k)-10R(\boldsymbol{y},\boldsymbol{z}^k)\beta+10\beta^2\right)}{\beta} \quad \text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.26}$$

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left[\frac{\beta^2}{12}-\frac{\beta^3}{30R(\boldsymbol{y},\boldsymbol{z}^k)}\right] \quad \text{when} \quad \frac{R}{\beta}>1 \tag{4.27}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{1}{30}R(\boldsymbol{y},\boldsymbol{z}^k)\frac{\left(6R^2(\boldsymbol{y},\boldsymbol{z}^k)-15R(\boldsymbol{y},\boldsymbol{z}^k)\beta+10\beta^2\right)}{\beta^2}\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}}$$

$$\text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.28}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{\beta^2}{30R^2(\boldsymbol{y},\boldsymbol{z}^k)}\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n} \quad \text{when} \quad \frac{R}{\beta}>1 \tag{4.29}$$

g) Wendland-C2

Let us call Wendland-2 the function defined by

$$f(\boldsymbol{y},\boldsymbol{z}^k)=\left(1-\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^4\left[4\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)+1\right] \quad \text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.30}$$

and

$$f(\boldsymbol{y},\boldsymbol{z}^k)=0 \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.31}$$

In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{R^2(\boldsymbol{y},\boldsymbol{z}^k)}{6}-\frac{R^4(\boldsymbol{y},\boldsymbol{z}^k)}{2\beta^2}+\frac{2R^5(\boldsymbol{y},\boldsymbol{z}^k)}{3\beta^3}-\frac{5R^6(\boldsymbol{y},\boldsymbol{z}^k)}{14\beta^4}+\frac{R^7(\boldsymbol{y},\boldsymbol{z}^k)}{14\beta^5}\right)$$

$$\text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.32}$$

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{\beta^2}{14}-\frac{\beta^3}{42R(\boldsymbol{y},\boldsymbol{z}^k)}\right) \qquad \text{when} \quad \frac{R}{\beta}>1 \tag{4.33}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{R^6(\boldsymbol{y},\boldsymbol{z}^k)}{2\beta^5}-\frac{15R^5(\boldsymbol{y},\boldsymbol{z}^k)}{7\beta^4}+\frac{10R^4(\boldsymbol{y},\boldsymbol{z}^k)}{3\beta^3}\right.$$

$$\left.-\frac{2R^3(\boldsymbol{y},\boldsymbol{z}^k)}{\beta^2}+\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{3}\right)\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} \qquad \text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.34}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\,\frac{\beta^3}{42R^2(\boldsymbol{y},\boldsymbol{z}^k)}\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.35}$$

h) Wendland-C4

Let us call Wendland-C4 the function defined by

$$f(\boldsymbol{y},\boldsymbol{z}^k)=\left(1-\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^6\left[35\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^2+18\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}+3\right] \quad \text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.36}$$

and

$$f(\boldsymbol{y},\boldsymbol{z}^k)=0 \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.37}$$

In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{R^2(\boldsymbol{y},\boldsymbol{z}^k)}{330}\left(\frac{105R^8(\boldsymbol{y},\boldsymbol{z}^k)-704R^7(\boldsymbol{y},\boldsymbol{z}^k)\beta+1925R^6(\boldsymbol{y},\boldsymbol{z}^k)\beta^2}{\beta^8}+\right.$$
$$\left.\frac{-2640R^5(\boldsymbol{y},\boldsymbol{z}^k)\beta^3+1650R^4(\boldsymbol{y},\boldsymbol{z}^k)\beta^4-462R^2(\boldsymbol{y},\boldsymbol{z}^k)\beta^6+165\beta^8}{\beta^8}\right)$$
$$\text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.38}$$

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{\beta^2}{6}-\frac{8\beta^3}{165R(\boldsymbol{y},\boldsymbol{z}^k)}\right) \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.39}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{165}\left(\frac{525R^8(\boldsymbol{y},\boldsymbol{z}^k)-3168R^7(\boldsymbol{y},\boldsymbol{z}^k)\beta+7700R^6(\boldsymbol{y},\boldsymbol{z}^k)\beta^2}{\beta^8}+\right.$$
$$\left.\frac{-9240R^5(\boldsymbol{y},\boldsymbol{z}^k)\beta^3+4950R^4(\boldsymbol{y},\boldsymbol{z}^k)\beta^4-924R^2(\boldsymbol{y},\boldsymbol{z}^k)\beta^6+165\beta^8}{\beta^8}\right)\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}}$$
$$\text{when} \quad 0\le\frac{R}{\beta}\le 1 \tag{4.40}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{8\beta^3}{165R^2(\boldsymbol{y},\boldsymbol{z}^k)}\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.41}$$

i) Wendland-C6

Let us call Wendland-C6 the function defined by

$$f(\boldsymbol{y},\boldsymbol{z}^k)=\left(1-\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^8\left[32\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^3+25\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^2+8\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}+1\right]$$

$$\text{when}\quad 0\le\frac{R}{\beta}\le 1 \tag{4.42}$$

and

$$f(\boldsymbol{y},\boldsymbol{z}^k)=0 \qquad \text{when} \qquad \frac{R}{\beta}>1 \tag{4.43}$$

In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ will be

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\frac{R^2(\boldsymbol{y},\boldsymbol{z}^k)}{5460}\left(\frac{960R^{11}(\boldsymbol{y},\boldsymbol{z}^k)-8085R^{10}(\boldsymbol{y},\boldsymbol{z}^k)\beta+29120R^9(\boldsymbol{y},\boldsymbol{z}^k)\beta^2}{\beta^{11}}+\right.$$
$$\frac{-57330R^8(\boldsymbol{y},\boldsymbol{z}^k)\beta^3+64064R^7(\boldsymbol{y},\boldsymbol{z}^k)\beta^4-35035R^6(\boldsymbol{y},\boldsymbol{z}^k)\beta^5}{\beta^{11}}$$
$$\left.\frac{+8580R^4(\boldsymbol{y},\boldsymbol{z}^k)\beta^7-3003R^2(\boldsymbol{y},\boldsymbol{z}^k)\beta^9+910\beta^{11}}{\beta^{11}}\right)$$

$$\text{when}\quad 0\le\frac{R}{\beta}\le 1 \tag{4.44}$$

$$\hat{u}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{7\beta^2}{156}-\frac{16\beta^3}{1365R(\boldsymbol{y},\boldsymbol{z}^k)}\right) \qquad \text{when} \qquad \frac{R}{\beta}>1$$

(4.45)

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\,\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{1365}\left(\frac{3120R^{11}(\boldsymbol{y},\boldsymbol{z}^k)-24255R^{10}(\boldsymbol{y},\boldsymbol{z}^k)\beta+80080R^{9}(\boldsymbol{y},\boldsymbol{z}^k)\beta^2}{\beta^{11}}+\right.$$
$$\frac{-143325R^{8}(\boldsymbol{y},\boldsymbol{z}^k)\beta^3+144144R^{7}(\boldsymbol{y},\boldsymbol{z}^k)\beta^4-70070R^{6}(\boldsymbol{y},\boldsymbol{z}^k)\beta^5}{\beta^{11}}$$
$$\left.\frac{+12870R^{4}(\boldsymbol{y},\boldsymbol{z}^k)\beta^7-3003R^{2}(\boldsymbol{y},\boldsymbol{z}^k)\beta^9+455\beta^{11}}{\beta^{11}}\right)\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}}$$

$$\text{when}\quad 0\le\frac{R}{\beta}\le 1 \tag{4.46}$$

$$\hat{q}(\boldsymbol{y})=\sum_{k=1}^{J+I}\alpha_k\left(\frac{16\beta^3}{1365R^2(\boldsymbol{y},\boldsymbol{z}^k)}\right)\frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}}\qquad\text{when}\qquad\frac{R}{\beta}>1 \tag{4.47}$$

j) Bhumann

Bhumann [12] proposed a new family of CS-RBFs. Here it has been tried the following one

$$f(\boldsymbol{y},\boldsymbol{z}^k)=\frac{1}{6}\qquad\text{when}\ \frac{R}{\beta}=0 \tag{4.48}$$

$$f(\boldsymbol{y},\boldsymbol{z}^k)=2\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^4\ln\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)-\frac{7}{2}\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^4+$$
$$\frac{16}{3}\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^3-2\left(\frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{\beta}\right)^2+\frac{1}{6}\qquad\text{when}\ \ 0<\frac{R}{\beta}\le 1 \tag{4.49}$$

and

$$f(\boldsymbol{y},\boldsymbol{z}^k)=0\qquad\text{when}\qquad\frac{R}{\beta}>1 \tag{4.50}$$

In this case $\hat{u}(\boldsymbol{y})$ and $\hat{q}(\boldsymbol{y})$ are

$$\hat{u}(\boldsymbol{y}) = 0 \qquad \text{when } \frac{R}{\beta} = 0 \tag{4.51}$$

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R^2(\boldsymbol{y},\boldsymbol{z}^k)}{8820} \left[\frac{420R^4(\boldsymbol{y},\boldsymbol{z}^k)\ln\left(R(\boldsymbol{y},\boldsymbol{z}^k)\right) - 865R^4(\boldsymbol{y},\boldsymbol{z}^k)}{\beta^4} + \frac{-420R^4(\boldsymbol{y},\boldsymbol{z}^k)\ln(\beta) + 1568R^3(\boldsymbol{y},\boldsymbol{z}^k)\beta - 882R^2(\boldsymbol{y},\boldsymbol{z}^k)\beta^2 + 245\beta^4}{\beta^4} \right]$$

$$\text{when } \; 0 < \frac{R}{\beta} \le 1 \tag{4.52}$$

$$\hat{u}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left(\frac{\beta^2}{90} - \frac{8\beta^3}{2205R(\boldsymbol{y},\boldsymbol{z}^k)} \right) \qquad \text{when } \; \frac{R}{\beta} > 1 \tag{4.53}$$

$$\hat{q}(\boldsymbol{y}) = \frac{1}{18} \qquad \text{when } \; \frac{R}{\beta} = 0 \tag{4.54}$$

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \frac{R(\boldsymbol{y},\boldsymbol{z}^k)}{4410} \left[\frac{1260R^4(\boldsymbol{y},\boldsymbol{z}^k)\ln\left(R(\boldsymbol{y},\boldsymbol{z}^k)\right) - 1260R^4(\boldsymbol{y},\boldsymbol{z}^k)\ln(\beta)}{\beta^4} + \frac{-2385R^4(\boldsymbol{y},\boldsymbol{z}^k) + 3920R^3(\boldsymbol{y},\boldsymbol{z}^k)\beta - 1764R^2(\boldsymbol{y},\boldsymbol{z}^k)\beta^2 + 245\beta^4}{\beta^4} \right] \frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}}$$

$$\text{when } \; 0 < \frac{R}{\beta} \le 1 \tag{4.55}$$

and

$$\hat{q}(\boldsymbol{y}) = \sum_{k=1}^{J+I} \alpha_k \left(\frac{8\beta^3}{2205R^2(\boldsymbol{y},\boldsymbol{z}^k)} \right) \frac{\partial R(\boldsymbol{y},\boldsymbol{z}^k)}{\partial n^{\boldsymbol{y}}} \qquad \text{when } \; \frac{R}{\beta} > 1 \tag{4.56}$$

4.6.2 A 1D Poisson's problem with constant sink term

The following Poisson´s equation with a constant source term with $d = 40$

$$\nabla^2 u(\boldsymbol{x}) = d \tag{4.57}$$

was solved with the DEC using all the RBFs introduced in section 4.6.1.
The one-dimensional analytical solution of (4.57) is

$$u(x) = \frac{d}{2}x^2 + \left(\frac{U_l - U_0 - \frac{dL^2}{2}}{L} \right) x + U_0 \tag{4.58}$$

where $U_0 = u(x = 0)$ and $U_l = u(x = L)$ are the boundary conditions.
In order to produce equivalent 1-D results from the 3-D codes, prismatic domains of length $L = 1$ in the x direction and width $W = 0.2$ in the y and z direction were used, see figure 4.8, and the following boundary conditions were applied

$$u(0,y) = U_0 = 10 \quad ; \quad u(L,y) = U_l = 4 \tag{4.59}$$

and

$$\left.\frac{\partial u}{\partial n}\right|_{y=W/2} = \left.\frac{\partial u}{\partial n}\right|_{y=-W/2} = \left.\frac{\partial u}{\partial n}\right|_{z=W/2} = \left.\frac{\partial u}{\partial n}\right|_{z=-W/2} = 0 \tag{4.60}$$

Figure 4.10 shows the analytical solution for this example, while figure 4.9 shows the discretization of the considered domain.

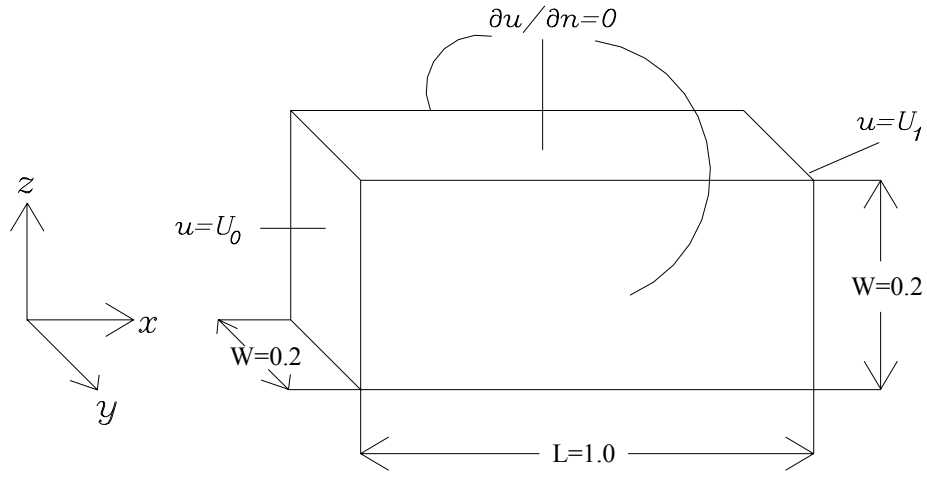


Figure 4.8: Geometry of the domain and the boundary conditions used in the numerical example.

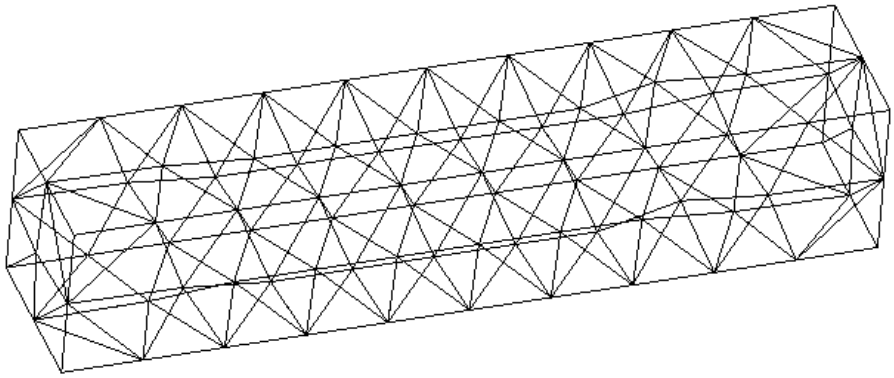


Figure 4.9: Mesh with 173 subdomains used in the examples.

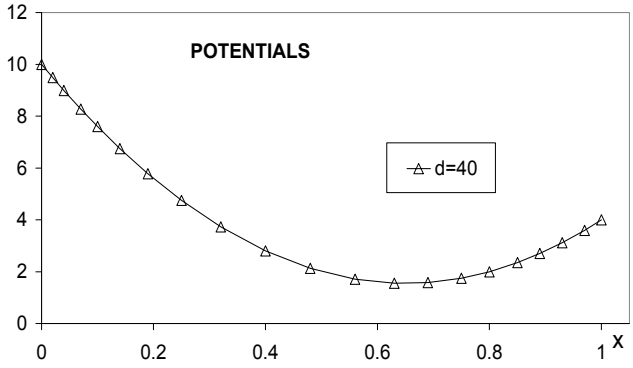


Figure 4.10: Analytical solution of the Poisson case.

Figures 4.11 to 4.15 present error distributions for this case using a mesh of 173 subdomains/tetrahedrons. The definition of the error is given in eqn. (3.37). The error distributions are grouped in different figures according to the order of magnitude of the error $\mathcal{O}$(Error). The number appearing in brackets to the right of the label identifying the CS-RBF used is the value of the scale factor *s*, which sets the size of the support β according to eqn. (4.7).

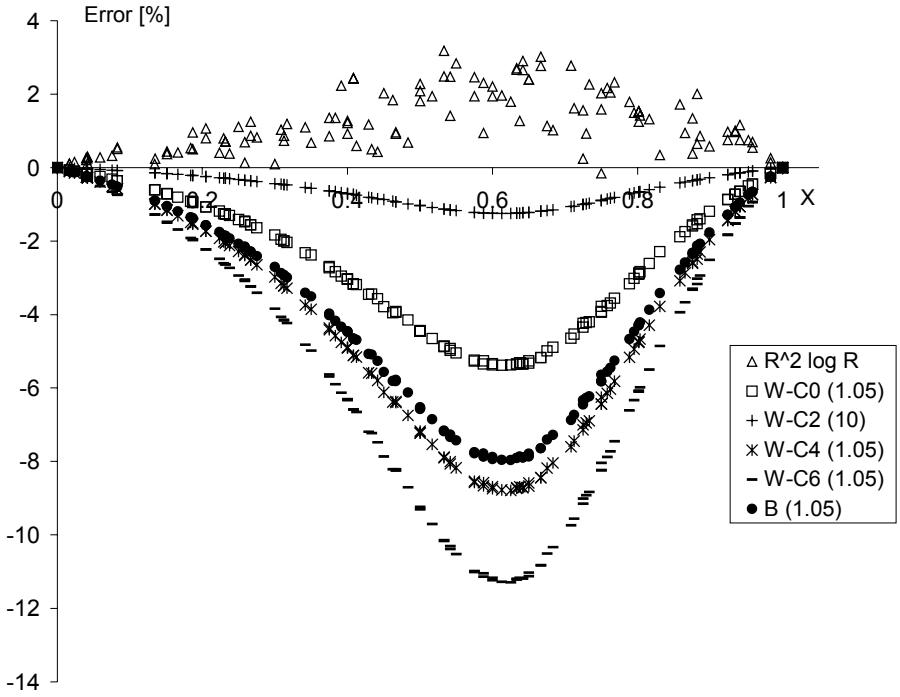


Figure 4.11: Error distribution along the x-axis using the DEC for the Poisson case and different RBFs (cases in which $\mathcal{O}$(Error) = 1).

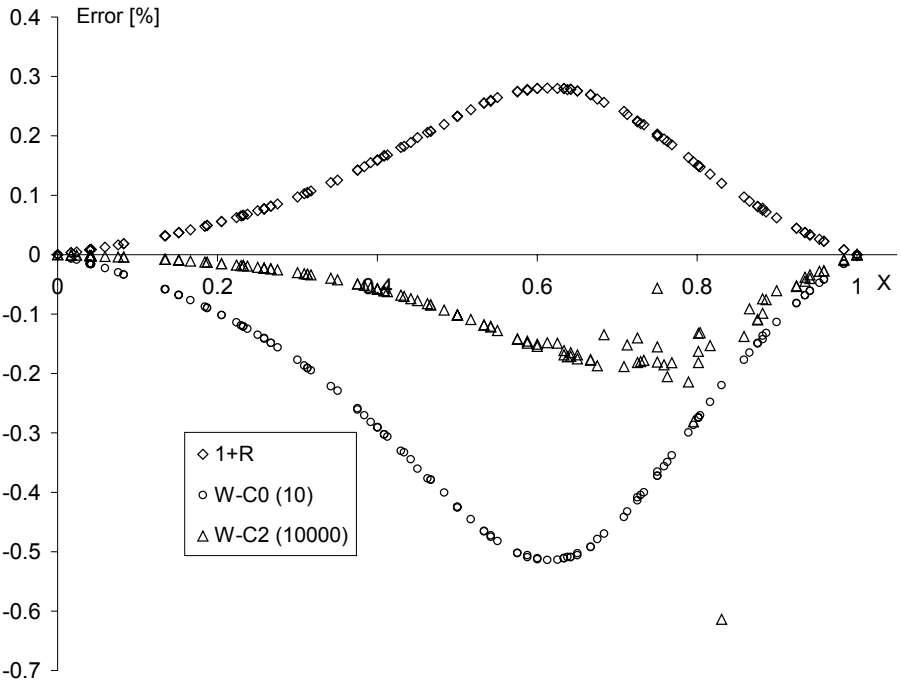


Figure 4.12: Error distribution along the x-axis using the DEC for the Poisson case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^{-1}$).

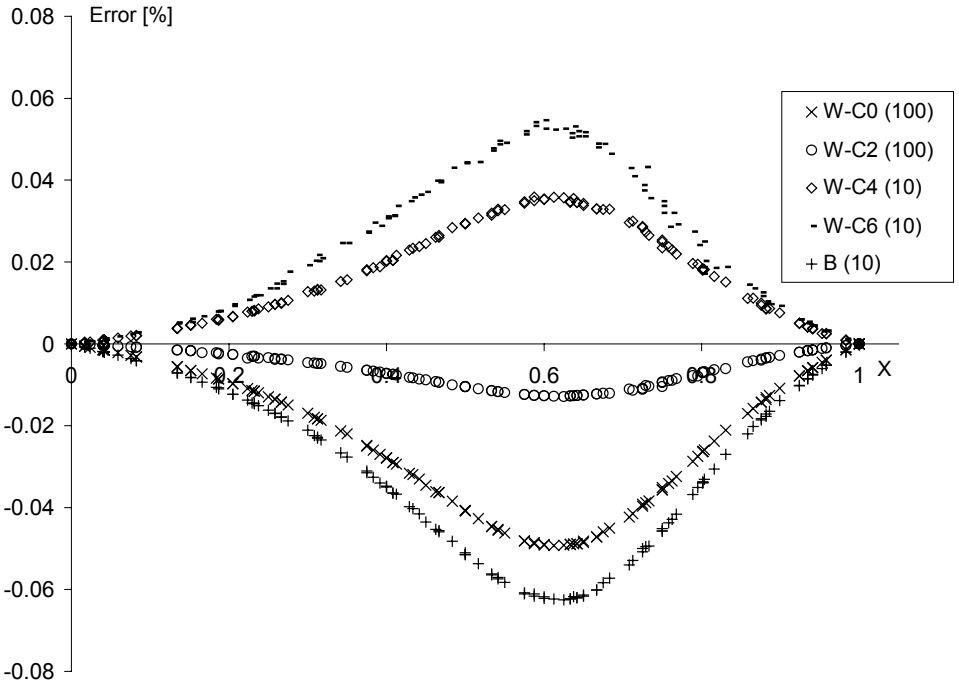


Figure 4.13: Error distribution along the x-axis using the DEC for the Poisson case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^{-2}$).

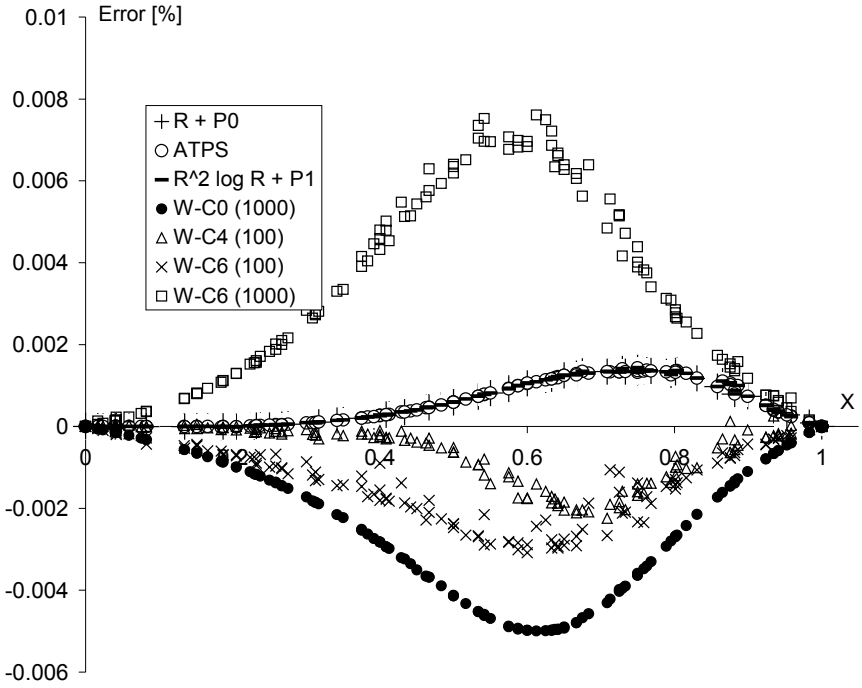


Figure 4.14: Error distribution along the x-axis using the DEC for the Poisson case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^{-3}$).

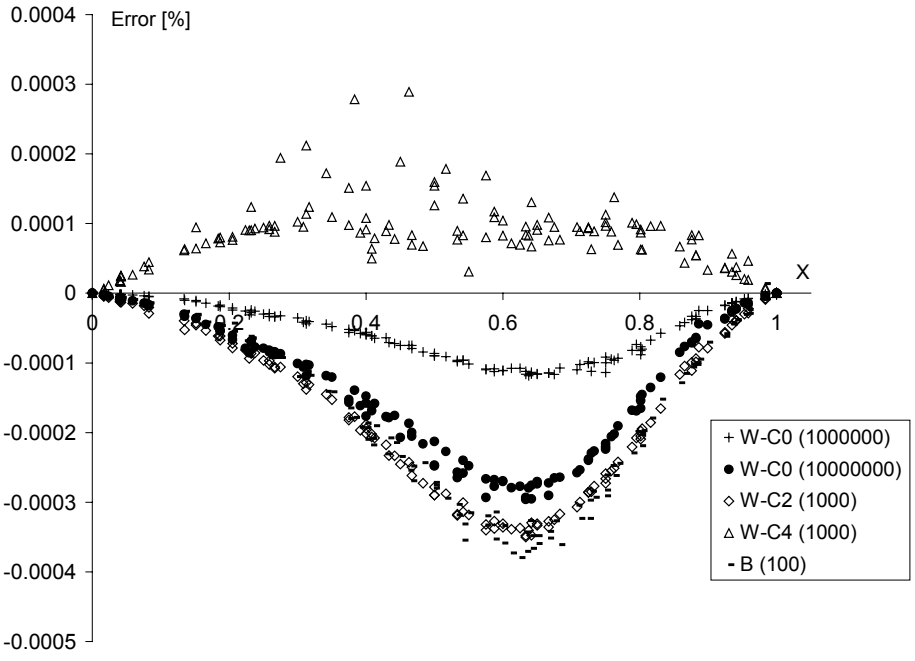


Figure 4.15: Error distribution along the x-axis using the DEC for the Poisson case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^{-4}$).

Figure 4.15 shows that the most accurate results are given by CS-RBFs, being the maximum errors of an order of magnitude 10^{-4} %. The relative error increases in all cases when values of the x coordinates are near 0.6 due to the definition of the relative error, because the analytical values have a minimum in that part of the domain and they are in the denominator of (3.37).

In section 4.6.1 it was pointed out that the accuracy is strongly influenced by the support β, which depends on s. It can be seen that for every CS-RBF the best accuracy is achieved for a value of s that it is neither the smallest nor the biggest of the tried values. This shall be discussed in more detail at the end of the next section.

As for the other functions, all of them being conditionally positive definite, it can be seen that the results agreed with what the theory predicts: a proper implementation requires a polynomial term. Once this is done, the results of ATPS, $R + P_0$ and $R^2 \log R + P_1$ gave almost identical results with good accuracy (figure 4.14), being the maximum error of order of magnitude 10^{-3}. The reason could be that the polynomial part of the RBF becomes dominant and fits well with the analytical solution.

Next, the approximation functions will be tested for the case of advection-diffusion problem.

4.6.3 A 1D steady state-advection-diffusion problem with non-uniform velocity field

The steady state advection-diffusion equation

$$D\nabla^2 u - \vec{V} \cdot \vec{\nabla} u - ku = 0 \tag{4.61}$$

where D is the coefficient of dispersion, $\vec{V}$ is the vector of flow velocity and k is the reaction constant, which is constant in the whole domain, was solved with the DEC using all the RBFs introduced in section 4.6.1. The 1-D problem that was solved is the same one of section 3.4.1, that is, with a velocity variable velocity field which is function of the reaction constant

$$V(x,k) = \frac{1}{L}\ln\left(\frac{U_1}{U_0}\right) + k\left(x - \frac{L}{2}\right) \tag{4.62}$$

The boundary conditions, geometry of the domain and mesh were identical to the Poisson case (figures 4.8 and 4.9). Figure 4.16 shows the analytical solution for the concentration for the different values of the reaction rate *k*.

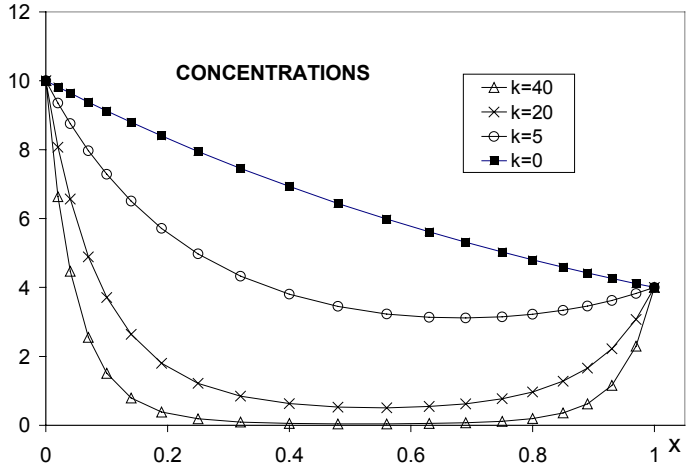


Figure 4.16: Analytical solution of the tested cases.

Figures 4.17 to 4.20 show the error distributions when k = 20. Figures 4.21 to 4.24 show the same but for k = 5, which is a less advective case. Note that as the velocity field is divergent, a stagnation point exists at x = 0.83, 0.33, and 0.18 for k equal to 40, 20 and 5, respectively. For k = 0 the velocity field is uniform. The analytical solution for potential drops from 10 to very low values in the middle of the domain, being the drop higher as the reaction constant, k, increases, because the flow velocity increases with k, as the flow is directed toward the boundaries.

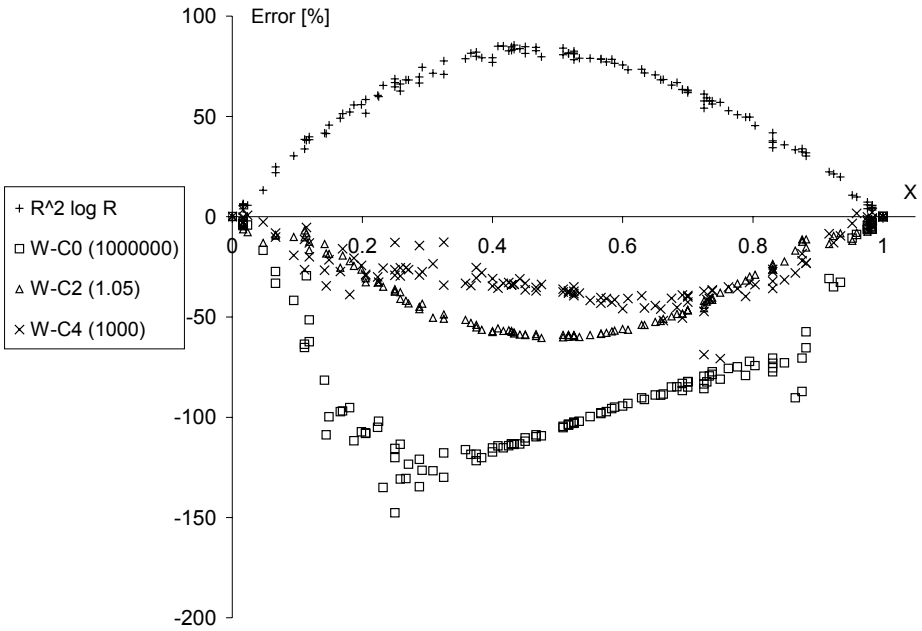


Figure 4.17: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 20$) case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^2$).

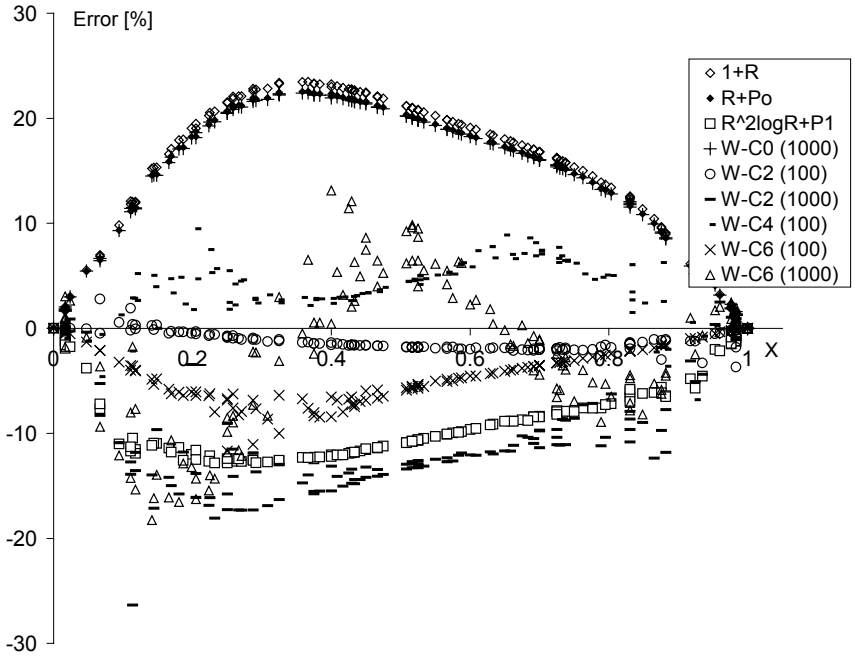


Figure 4.18: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 20$) case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10$).

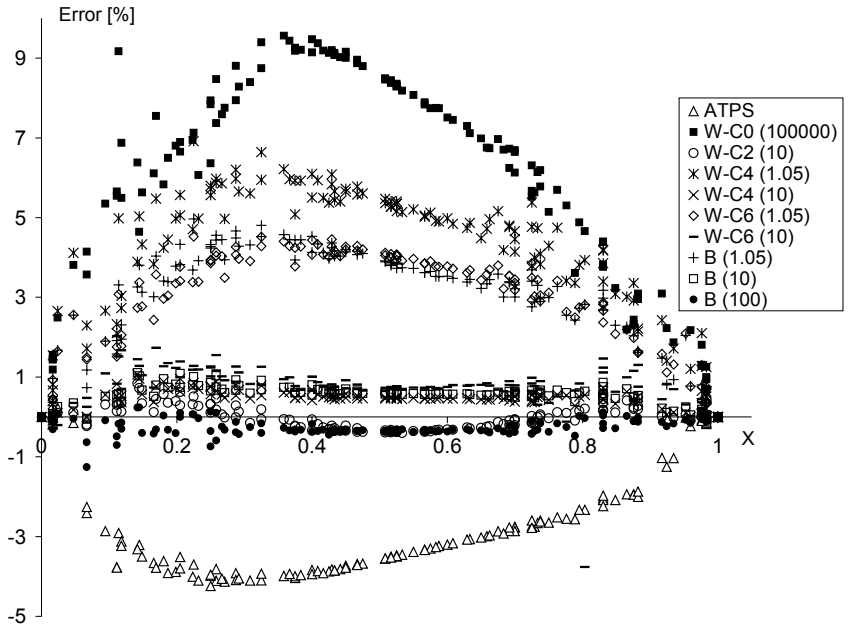


Figure 4.19: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 20$) case and different RBFs (cases in which $\mathcal{O}$(Error) = 1).

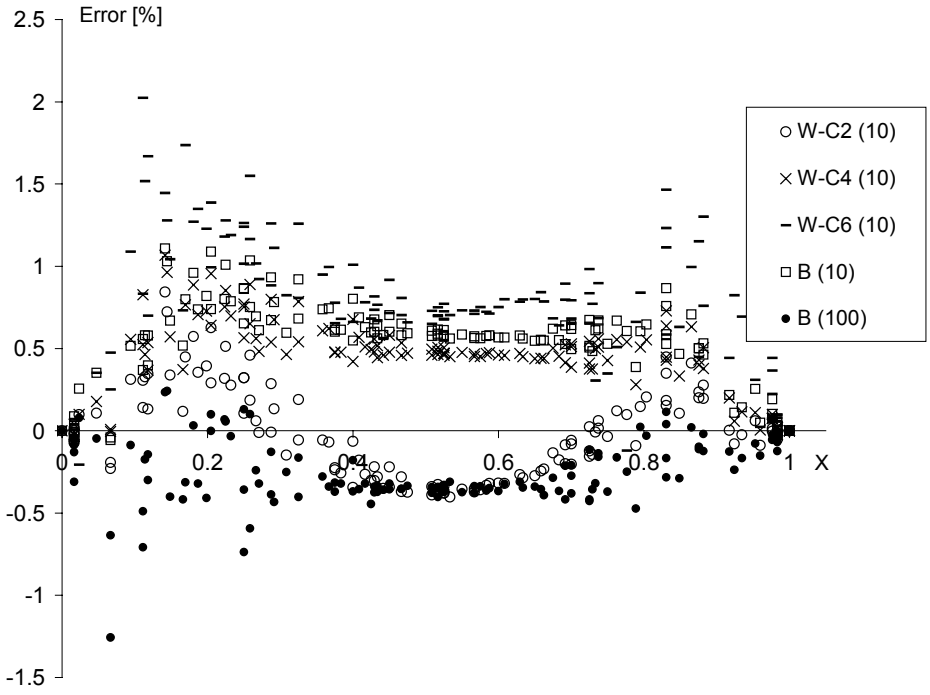


Figure 4.20: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 20$) case and different RBFs (cases in which $\mathcal{O}$(Error) = 1).

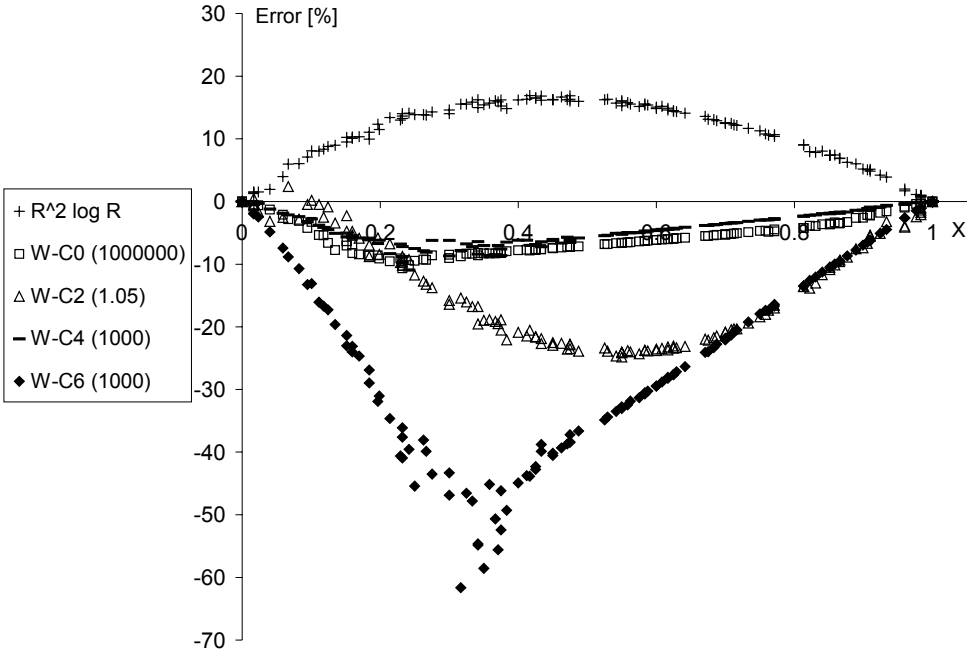


Figure 4.21: Error distribution along the x-axis using the DEC for the advection-diffusion (k = 5) case and different RBFs (cases in which $\mathcal{O}$(Error) = 10).

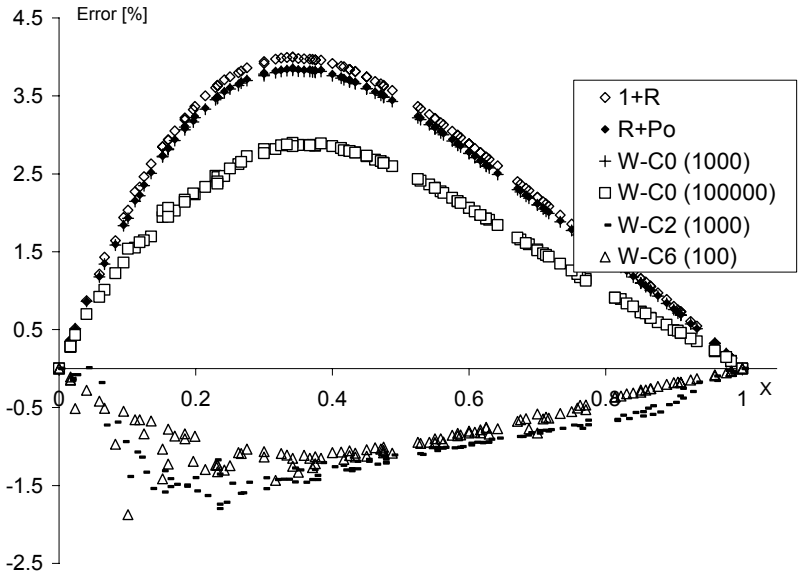


Figure 4.22: Error distribution along the x-axis using the DEC for the advection-diffusion (k = 5) case and different RBFs (cases in which $\mathcal{O}$(Error) = 1).

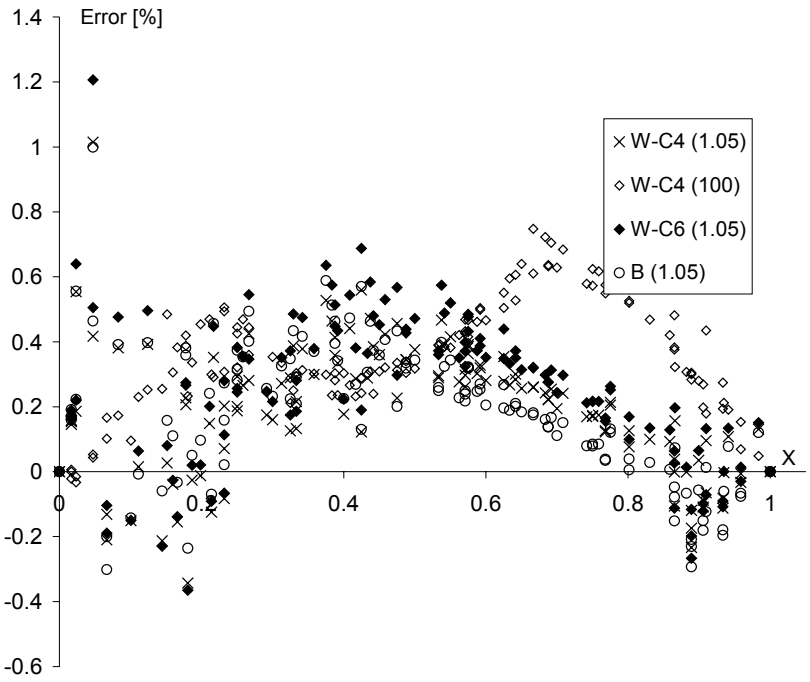


Figure 4.23: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 5$) case and different RBFs (cases in which $\mathcal{O}(\text{Error}) = 10^{-1}$).

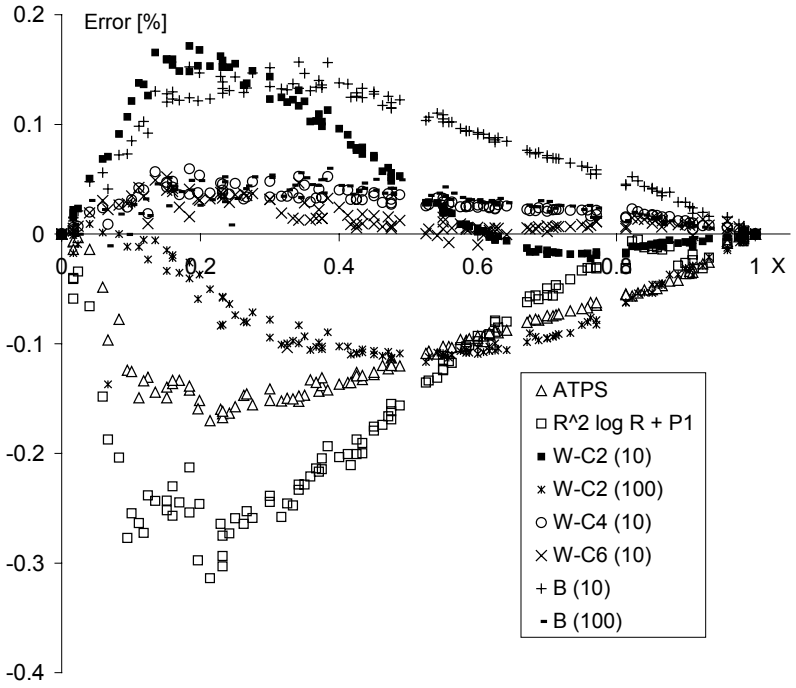


Figure 4.24: Error distribution along the x-axis using the DEC for the advection-diffusion ($k = 5$) case and different RBFs (cases in which $\mathcal{O}(\text{Error}) \leq 10^{-1}$).

Results for k =40 show that the errors are higher in the case of advection-diffusion equation than in the case of the Poisson equation, and this is mainly due to the advective term. In the case of advection-diffusion equation Wendland-C0 did not outperform ATPS, and although some other CS-RBFs did, their accuracy was of the same order of magnitude as the accuracy of the ATPS. $1 + R$, $R + P_0$ and $R^2 \log R + P_1$ were unable to represent accurately the solution of (4.61), mainly because they failed to approximate accurately the partial derivatives of the advective term, which for this case is the dominant term in (4.61). When k = 5, apart from the fact that the error decreases in two orders of magnitude, it can be seen that the trends regarding the performance of the RBFs are the same than the previous case, except that for k = 5, $R^2 \log R + P_1$ was almost as accurate as ATPS.

The problem of defining the optimum value of the scale of the support s becomes more obvious, since the optimum value of s varies from case to case. In here, optimum values of s have been found by trial and error. It remains for future research to establish guidelines to set the size of the support. So far this topic has been discussed within the framework of multiscattered interpolation theory by Floater & Iske [44] and Schaback [9] among others. It has never been considered related to DRM-MD since this is the first time that CS-RBFs are used in a DRM-MD algorithm. Floater & Iske proposed a strategy but it is to be used in a hierarchical scheme for smoothly interpolating scattered data, which cannot be applied in the codes presented here. The problem could be addressed by establishing indicators of both error and stability in terms of scaling of the support in order to establish criteria in solid mathematical grounds.

Considering all the examples, of the globally supported RBFs only ATPS performed satisfactorily in all the cases. In terms of accuracy CS-RBFs outperformed ATPS in both cases. However, the performance of CS-RBFs depends on the size of the support, which introduces an additional variable in the problem.

A standard procedure to determine the optimum value of the size of the support is not yet available, ATPS appears to be the best choice of the tested functions for the 3D

DRM-MD code at the moment, as it produced one of the most accurate and certainly the most consistent results and does not introduce any additional parameters.

4.7 Scaling effects

Another important issue in this kind of codes was analysed, which is the scale of the problem. A given problem can be converted into a similar one but with a different size if a scale factor is properly applied on both the boundary conditions and the parameters of the governing equations. Once a result corresponding to this new problem is obtained, using again the same scale factor the solution of the original problem can be retrieved. From a mathematical point of view the results should be identical no matter whether scaling has been applied or not. In the codes presented here the scale affects the time needed for the iterative solver to converge. The reason is that in BEM formulations some of the matrices have different dimensions. For instance, let us consider the resulting system of equation of the advection diffusion case (for more details on the derivation of this equation see section 3.3)

$$\left(\mathbf{H}-\mathbf{T}-\frac{\mathbf{S}}{D}k\right)\boldsymbol{u}-\mathbf{G}\boldsymbol{q}=0 \tag{4.63}$$

The term $\left(\mathbf{H}-\mathbf{T}-\frac{\mathbf{S}}{D}k\right)$ has no dimensions, but the matrix $\mathbf{G}$ has a dimension of length, consequently, the size at which the problem is solved affects the condition number of the resulting system of equations. Indeed, as the DRM formulation is applied locally to every subdomain, a change in the size of the mesh, that is, when refining the mesh, will change the condition number. Note that it is the size of the sobdomain what must be controlled rather than the size of the whole domain.

Figure 4.25 shows the variation of the condition number with the inverse of the scale factor for the advection-diffusion case that was presented in the previous section, while keeping the geometry of the mesh constant. The inverse of the scale factor indicates how many times the length of the problem has been enlarged. Both the DEC and the CEC have been tried using ATPS as DRM approximation function.

It can be seen that the condition number has a minimum for an inverse of the scale factor around two for the DEC, and around sixteen for the CEC. Note that the DEC is better conditioned than the CEC. Higher values of the inverse of the scale factor causes changes in the condition number that are more sudden for the DEC and depend on the value of the reaction constant k. Let us recall that in this problem the velocity field is a function of k (see eqn. (4.62)); the higher the value of k the more advective the problem. As both, the advective term and the reactive term are contained in the term $\left(\mathbf{H} - \mathbf{T} - \frac{\mathbf{S}}{D} k \right)$, it becomes dominant when k increases; and as it has no dimensions, changes in the scale have less influence for higher k than for lower ones. For instance, for $k = 0$ the rate of increment of the condition number with the inverse of the scale factor is higher than for $k = 5$ and $k = 20$.

Figure 4.26 shows the variation of the maximum error with the inverse of the scale factor. For the DEC the best accuracy is in the range where the condition number is minimum. For the CEC the situation is different depending on the value of k. When the term $\left(\mathbf{H} - \mathbf{T} - \frac{\mathbf{S}}{D} k \right)$ dominant, the accuracy is not noticeably affected by the scaling, but when it is small or null, the error reaches a minimum for a scale factor that it is not the best from the point of view of the condition number. At the same time these optima scales vary with the value of k.

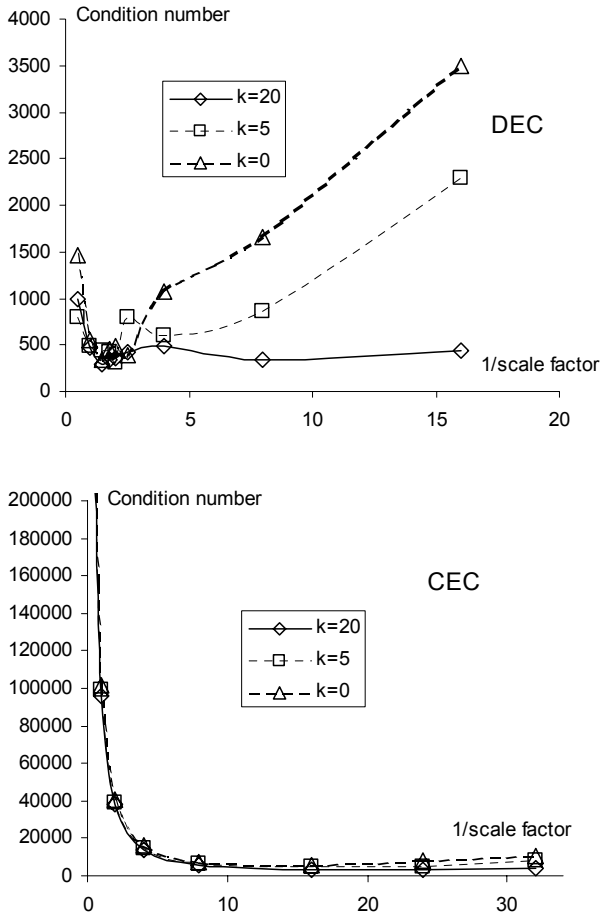


Figure 4.25: Variation of the condition number with the scale factor for both the DEC and the CEC.

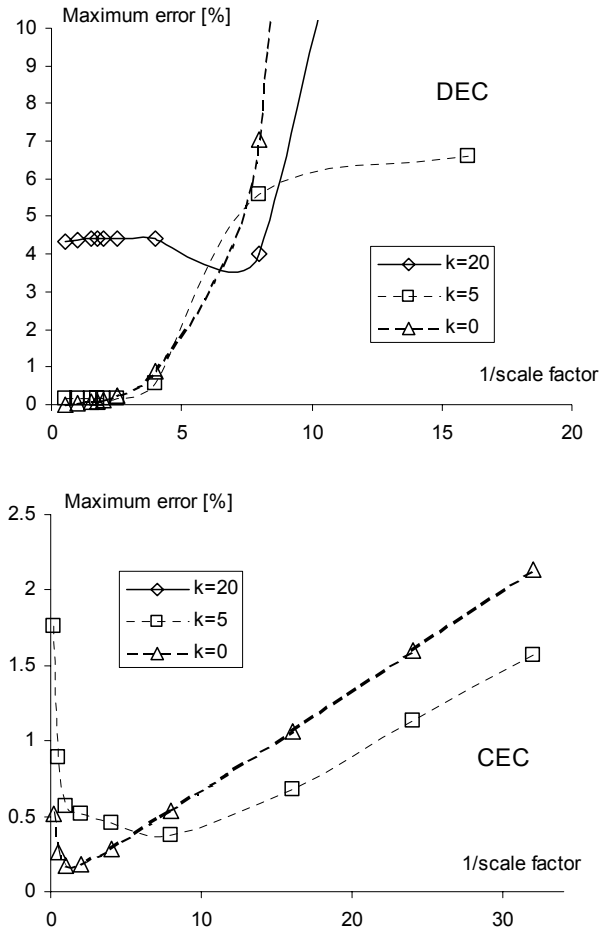


Figure 4.26: Variation of the maximum error with the scale factor for both the DEC and the CEC. The values corresponding to the CEC with k = 20 are not displayed because they stay uniformly around 14 %.

It is evident that it is convenient to scale the problem when using these codes. A single change in the unit system in which a given problem is being specified or a refinement of the mesh can cause the code to fail if scaling is not implemented.

Tables 4.4 and 4.5 compare the time that the solver needs to give the solution of the problem and the maximum error for both situations: with and without scaling. The problem solved was the advection-diffusion case with a refined mesh of 1456 subdomains, therefore, a better accuracy was expected with regard to the previous cases, in which a mesh of 173 subdomains was used. The scaling criterion was that the average size of the subdomains had to be kept the same as in the 173 subdomains case when the inverse of the scale factor was two, for the DEC, and fifteen, for the

CEC. As the scaling criterion is up to a certain point a matter of preference, it can be implemented with different degrees of sophistication.

Table 4.4: DEC. Time needed by the solver (the iterative one) to obtain a solution and maximum error with and without scaling. Refined mesh of 1456 subdomains.

DEC								
	$k = 40$		$k = 20$		$k = 5$		$k = 0$	
	Not scaled	Scaled	Not scaled	Scaled	Not scaled	Scaled	Not scaled	Scaled
Solver time [s]	2366	746	2666	823	2928	895	3005	923
Maximum error [%]	6.3	6.5	0.94	0.92	0.039	0.36	0.0078	0.52

Table 4.5: CEC. Time needed by the solver to obtain a solution and maximum error with and without scaling. Refined mesh of 1456 subdomains.

CEC								
	$k = 40$		$k = 20$		$k = 5$		$k = 0$	
	Not scaled	Scaled	Not scaled	Scaled	Not scaled	Scaled	Not scaled	Scaled
Solver time [s]	1451	103	1503	145	1621	158	1661	169
Maximum error [%]	33.2	23.0	2.0	1.9	0.13	0.78	0.066	1.0

The results displayed in Tables 4.4 and 4.5 show how the rate of convergence of the solver improves with the scaling. It reduces roughly three times the CPU time for the DEC and ten times for the CEC. The accuracy worsens when k is equal to five and zero, however, for these values of k the codes achieved good accuracy so the error was low. For the CEC and $k = 40$ the accuracy improves substantially with scaling. The refinement of the mesh clearly reduces the error which proves the convergence of the method

4.8 Internal DRM nodes

The DEC and the CEC, where tested using internal DRM nodes (IDRMN). Two situations were considered: a) when one IDRMN was added in the mesh in the middle of every subdomain, and b) when five IDRMN were added, preserving as

much as possible equal distance between the nodes in order to avoid ill conditioned system of equations. The distribution of the IDRMN is done automatically by the code, once the number of the IDRMN is defined in the input data.

It was expected that the accuracy would improve as more IDRMN were added. Table 4.6 and 4.7 show that this was true for both codes using a mesh of 173 subdomains. Table 4.8 shows that this is still the trend for the DEC when using a refined mesh of 1456 subdomains, but Table 4.9 shows that not always using IDRMN a better accuracy for the case of CEC is achieved. However, it can be seen that though Table 4.9 corresponds to results obtained using the mesh with 1456 subdomains, the accuracy is not better than the one obtained using 173 subdomains for the case of k equal to 5 or 0, which means that the code does not show convergence any more with the refinement of the mesh. The accuracy of this kind of codes can be improved by refining the mesh, but this procedure has a limit for the reason that the condition number worsens with the refinement of the mesh; further research on using IDRMNs should consider what happens when this limit is reached.

Adding 5 IDRMN seems to be a suitable tool to reduce significantly the error when using the DEC in cases in which the advective term is dominant, even though the solver time increases about 50%.

Table 4.6: DEC. Maximum error with 1 and 5 internal DRM nodes. Mesh of 173 subdomains. Direct solver.

	DEC								
	$k = 20$			$k = 5$			$k = 0$		
No. of IDRMN	NONE	1	5	NONE	1	5	NONE	1	5
Maximum error [%]	4.42	4.19	3.16	0.18	0.18	0.16	0.13	0.10	0.06

Table 4.7: CEC. Time needed by the solver to obtain a solution and maximum error with 1 and 5 internal DRM nodes. Mesh of 173 subdomains.

	CEC								
	$k = 20$			$k = 5$			$k = 0$		
No. of IDRMN	NONE	1	5	NONE	1	5	NONE	1	5
Solver time [s]	10	19	47	12	20	55	13	24	63
Maximum error [%]	14.4	12.0	9.2	0.63	0.43	0.33	0.99	0.65	0.34

Table 4.8: DEC. Time needed by the solver (the iterative one) to obtain a solution and maximum error with 1 and 5 internal DRM nodes. Refined mesh of 1456 subdomains.

	DEC							
	$k = 40$		$k = 20$		$k = 5$		$k = 0$	
No. of IDRMN	1	5	1	5	1	5	1	5
Solver time [s]	802	1115	885	1220	959	1304	986	1350
Maximum error [%]	6.08	4.41	0.90	0.67	0.24	0.050	0.36	0.07

Table 4.9: CEC. Time needed by the solver to obtain a solution and maximum error with 1 and 5 internal DRM nodes. Refined mesh of 1456 subdomains.

	CEC							
	$k = 40$		$k = 20$		$k = 5$		$k = 0$	
No. of IDRMN	1	5	1	5	1	5	1	5
Solver time [s]	201	637	231	707	306	972	325	1027
Maximum error [%]	26.3	30.1	1.61	1.37	0.51	0.67	0.65	0.71

4.9 Conclusions of the chapter

In this chapter, the sensitivity of the implementation of the DRM-MD to the choice of radial basis functions, continuity of the elements, scaling and number of internal nodes were tested on a set of DRM-MD codes to solve the Poisson and the steady-state advection-diffusion equations in 3D domains.

Ten different RBFs were tested with the code using discontinuous elements. The highest accuracy was achieved when using the Wendland-C2, Wendland-C4, Wendland-C6 and Bhumann CS-RBFs. However, when using these functions, a further variable is introduced: the size of the support. As there is no rigorous guideline on how to choose the size of the support, which affects largely the accuracy and the convergence of the code, this is an obstacle at the moment for using the CS-RBFs in the DRM-MD codes. On the other hand, the ATPS showed satisfactory accuracy and convergence, and since the implementation of the ATPS is straightforward, they are the preferred choice in this thesis.

Tests were made on the influence of length scaling when solving the advection-diffusion problem using ATPS and both dis-continuous and continuous elements. The size of the subdomains, which depends on the size of the problem and the degree of refinement of the mesh, affects the condition number of the resulting system of equations and consequently the time needed by the iterative solvers to converge. By using simple scaling of the equations, the solver time was reduced by three and by ten times for the DEC and the CEC, respectively. The tests has been limited to length scaling because for the advection diffusion case that has been analysed, length is the only dimension that affects the resulting system of equation. Other problems should consider scaling according to their nature.

For the examples tested, internal DRM nodes in every subdomain improved the accuracy of the DEC, specially when using 5 IDRMN in a problem with a dominant advective term. The same cannot be said about the CEC, which for some cases produced lower accuracy with the increase of number of IDRMN.

Regarding the use of continuous and discontinuous elements, it is difficult to say which strategy is the best. The DEC offers higher accuracy, especially for highly advective transport, but it is much slower than the CEC. At the present stage, the DEC has the advantage that it can solve non-homogeneous domains. The most convenient strategy is to have a choice to use both types of elements in a single mesh, which would preserve the versatility of dealing with non-homogeneous domains, while offering possibility for CPU time and memory usage reduction.

In Chapters 5 and 6 the problems that shall be presented are solved with the DEC using the ATPS as DRM approximation function, no internal DRM nodes and the length scaling criterion used in section 4.7.

Chapter 5

Flow and transport in saturated media

5.1 Equations to be solved

In this chapter, flow and transport of a pollutant disposed in a repository built in a host media constituted by limestone are modelled considering different scenarios.

The flow field has been considered to be steady, and the domain was piecewise homogeneous and isotropic. In these conditions the Laplace type equation governs the flow:

$$K\nabla^2\phi = 0 \tag{5.1}$$

where K is the hydraulic conductivity and ϕ the hydraulic head.

The velocity field is related to ϕ by the generalized Darcy's law

$$\vec{q} = -K\vec{\nabla}\phi \tag{5.2}$$

where $\vec{q}$ is the specific discharge vector, which is

$$\vec{q} = \vec{V}n_{eff} \tag{5.3}$$

where n_{eff} is the effective porosity. Note that the letter 'q' is used here to denote both specific discharge and normal derivatives. This notation is widely used in the BEM and the fluid mechanic literature, which can lead to some confusion. Throughout this thesis, the symbol $\vec{q}$ is used for the specific discharge vector while $\boldsymbol{q}$ and q symbols are used for normal derivatives. The governing equation of transport by advection, dispersion and diffusion, considering adsorption but excluding reaction, in a saturated, isotropic, homogeneous media is

$$R\frac{\partial c}{\partial t} = D\nabla^2 c - \vec{V}\cdot\vec{\nabla}c + \sum s \tag{5.4}$$

where c is the concentration of a given substance, R is the retardation factor, D is the coefficient of mechanical dispersion and s stands for a source or a sink. The theoretical considerations that lead to eqns. (5.1) and (5.4) have been extensively treated by Bear & Verruijt [49] among others.

After applying the BEM, the resulting system of equations in every subdomain for (5.1) is the well known

$$\mathbf{H}\boldsymbol{\phi} - \mathbf{G}\boldsymbol{q} = 0 \tag{5.5}$$

On the other hand, in order to apply the DRM formulation on (5.4) let us express it as a equation

$$\nabla^2 c = \frac{1}{D}\left(\vec{V}\cdot\vec{\nabla}c - \sum s + R\frac{\partial c}{\partial t}\right) \tag{5.6}$$

Then, replacing the right-hand-side into (2.14) yields

$$\mathbf{H}\boldsymbol{c} - \mathbf{G}\boldsymbol{q} = \frac{\mathbf{S}}{D}\left[V_x\frac{\partial \boldsymbol{c}}{\partial x} + V_y\frac{\partial \boldsymbol{c}}{\partial y} + V_z\frac{\partial \boldsymbol{c}}{\partial z} - \sum \boldsymbol{s} + R\frac{\partial \boldsymbol{c}}{\partial t}\right] \tag{5.7}$$

By using (3.9), (5.7) can be expressed as

$$\mathbf{H}\boldsymbol{c} - \mathbf{G}\boldsymbol{q} = \frac{\mathbf{S}}{D}\left[V_x\frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1}\boldsymbol{c} + V_y\frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1}\boldsymbol{c} + V_z\frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1}\boldsymbol{c} - \sum \boldsymbol{s} + R\frac{\partial \boldsymbol{c}}{\partial t}\right] \tag{5.8}$$

In this chapter, a simple two-level time integration scheme is employed together a linear approximation for the variation of c and q_c within each timestep, in the form

$$\boldsymbol{c} = (1-\theta_c)\boldsymbol{c}^{\mathrm{m}} + \theta_c\boldsymbol{c}^{\mathrm{m+1}} \tag{5.9}$$

$$\boldsymbol{q}_c = (1-\theta_{q_c})\,\boldsymbol{q}_c{}^{\mathrm{m}} + \theta_{q_c}\boldsymbol{q}_c{}^{\mathrm{m+1}} \tag{5.10}$$

$$\frac{\partial \boldsymbol{c}}{\partial t} = \frac{1}{\Delta t}\left(\boldsymbol{c}^{\mathrm{m+1}} - \boldsymbol{c}^{\mathrm{m}}\right) \tag{5.11}$$

where the super-script indicates the number of the time iteration and Δt the timestep. θ_c and θ_q are parameters which position the values of c and q_c, respectively, between time iterations m and m+1. The scheme represented by the last three equations was applied for first time in DRM by Wrobel et al. [50] to solve the transient diffusion equation. The same path was followed later by Partridge et al. [2]. Stability and accuracy of the scheme using DRM was tested in a variety of cases of 2D advection-diffusion problems by the IMF [51]. It was used by Popov & Power to modelling of flow of gases in porous media using DRM [20, 52] and DRM-MD [18] and by Peratta & Popov [22] to model transient transport in fracture porous media using DRM-MD. Following the guidelines of ref. [51], θ_{q_c} and θ_c were set equal to 1.

Substituting (5.9), (5.10) and (5.11) into (5.8) yields

$$\begin{aligned}\mathbf{H}\left[(1-\theta_c)\boldsymbol{c}^{\mathrm{m}} + \theta_c \boldsymbol{c}^{\mathrm{m+1}}\right] - \mathbf{G}\left[(1-\theta_{q_c})\,\boldsymbol{q}_c{}^{\mathrm{m}} + \theta_{q_c}\boldsymbol{q}_c{}^{\mathrm{m+1}}\right] = \\ \frac{\mathbf{S}}{D}\left[\boldsymbol{V}_x \frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1}\left[(1-\theta_c)\boldsymbol{c}^{\mathrm{m}} + \theta_c \boldsymbol{c}^{\mathrm{m+1}}\right] + \boldsymbol{V}_y \frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1}\left[(1-\theta_c)\boldsymbol{c}^{\mathrm{m}} + \theta_c \boldsymbol{c}^{\mathrm{m+1}}\right]\right. \\ \left. + \boldsymbol{V}_z \frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1}\left[(1-\theta_c)\boldsymbol{c}^{\mathrm{m}} + \theta_c \boldsymbol{c}^{\mathrm{m+1}}\right] - \sum \boldsymbol{s} + \frac{R}{\Delta t}\left(\boldsymbol{c}^{\mathrm{m+1}} - \boldsymbol{c}^{\mathrm{m}}\right)\right]\end{aligned} \tag{5.12}$$

Reordering produces

$$\begin{aligned}\left[\theta_c \mathbf{H} - \theta_c \frac{\mathbf{S}}{D}\left(\boldsymbol{V}_x \frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1} + \boldsymbol{V}_y \frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1} + \boldsymbol{V}_z \frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1}\right) - \frac{R\mathbf{S}}{D\Delta t}\right]\boldsymbol{c}^{\mathrm{m+1}} - \theta_{q_c}\mathbf{G}\boldsymbol{q}_c{}^{\mathrm{m+1}} = \\ \left[(1-\theta_c)\frac{\mathbf{S}}{D}\left(\boldsymbol{V}_x \frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1} + \boldsymbol{V}_y \frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1} + \boldsymbol{V}_z \frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1}\right) - (1-\theta_c)\mathbf{H} - \frac{R\mathbf{S}}{D\Delta t}\right]\boldsymbol{c}^{\mathrm{m}} + \\ + (1-\theta_{q_c})\mathbf{G}\,\boldsymbol{q}_c{}^{\mathrm{m}} - \frac{\mathbf{S}}{D}\sum \boldsymbol{s}\end{aligned} \tag{5.13}$$

Calling **T** the matrix $\mathbf{S}\left[\boldsymbol{V}_x \frac{\partial \mathbf{F}}{\partial x}\mathbf{F}^{-1} + \boldsymbol{V}_y \frac{\partial \mathbf{F}}{\partial y}\mathbf{F}^{-1} + \boldsymbol{V}_z \frac{\partial \mathbf{F}}{\partial z}\mathbf{F}^{-1}\right]$ yields

$$\left[\theta_c \mathbf{H} - \theta_c \frac{\mathbf{T}}{D} - \frac{R\mathbf{S}}{D\Delta t}\right]\boldsymbol{c}^{m+1} - \theta_{q_c}\mathbf{G}\boldsymbol{q}_c{}^{m+1} = \left[\left(1-\theta_c\right)\frac{\mathbf{T}}{D} - \left(1-\theta_c\right)\mathbf{H} - \frac{R\mathbf{S}}{D\Delta t}\right]\boldsymbol{c}^{m} + \left(1-\theta_{q_c}\right)\mathbf{G}\,\boldsymbol{q}_c{}^{m} - \frac{\mathbf{S}}{D}\sum \boldsymbol{s} \tag{5.14}$$

Equation (5.14), once the boundary and initial conditions are applied, is the system of equations produced by the DRM formulation on the transient advection-diffusion equation (5.4).

5.2 Comparison with semi-analytical solution

A 1D case of transport was solved using the DEC and results were compared with the semi-analytical solution given in ref. [53]. Several combinations of the parameters of eqn. (5.4) were tested, of which only one case will be shown in this section. The quality of the agreement was similar in all cases provided suitable timesteps were used.

The 1D case is a column of a porous material of $D = 1$ in which fluid is flowing at a uniform constant velocity equal to 1. The length of the column is 1 and the boundary conditions are concentration 1 at the inlet of the domain and zero normal derivative at the outlet. The reaction constant is equal to 1, no sources are present and the initial condition is zero concentration inside the domain.

In order to emulate the 1D case, a prismatic domain of 1×0.2×0.2 was subdivided in 173 subdomains and a constant, uniform, parallel-to-the-longitudinal-axis velocity equal to 1 was imposed. Apart from the boundary conditions corresponding to the 1D case, zero normal derivatives were specified on the lateral sides of the domain. The example was solved using the DEC with ATPS as DRM approximation function, the same scaling criterion than the one used in section 3.4 and no DRM nodes.

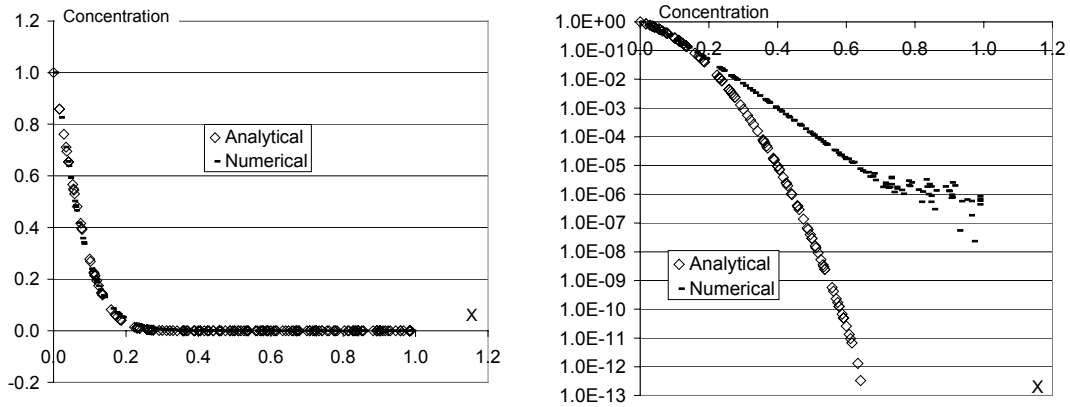


Figure 5.1: A 1D case. Profile of concentration at t = 0.004.

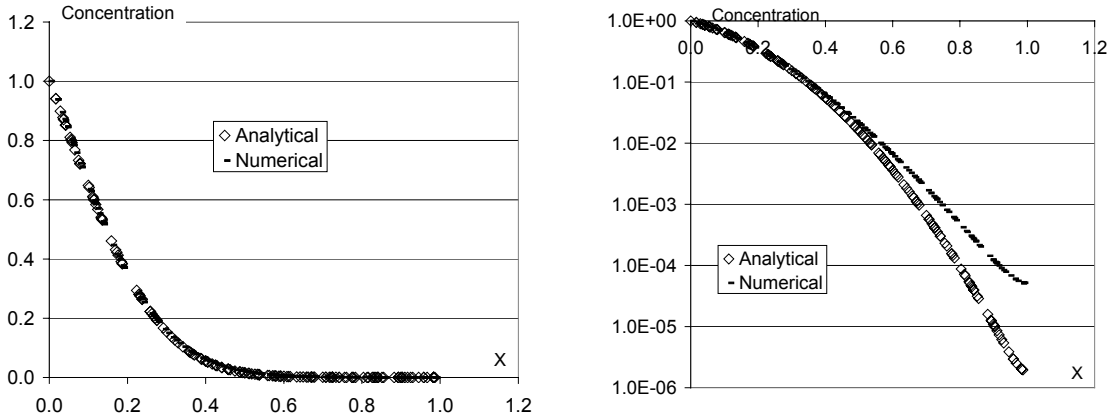


Figure 5.2: A 1D case. Profile of concentration at t = 0.02.

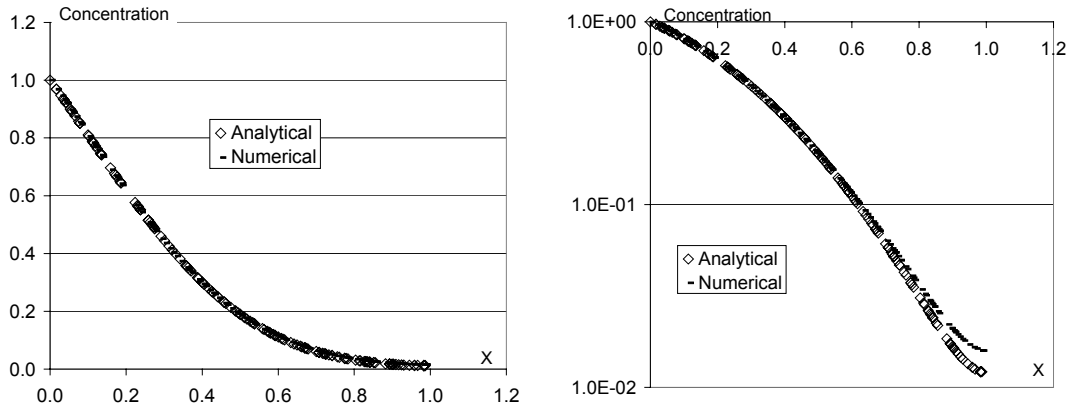
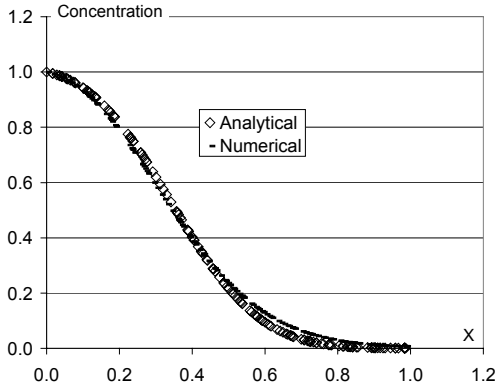


Figure 5.3: A 1D case. Profile of concentration at t = 0.06.

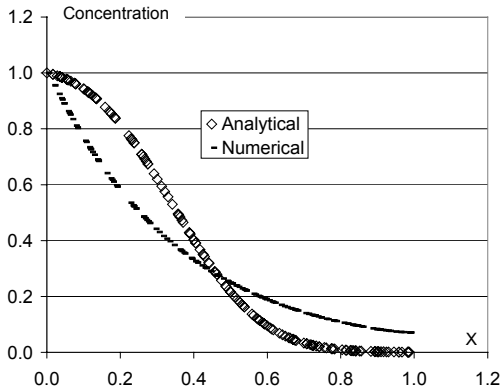
Figures 5.1 to 5.3 show results at times 0.004, 0.02 and 0.06. It was used 2 timesteps of 0.002 each to arrive at $t = 0.004$, then further 8 timesteps of 0.002 each to arrive at $t = 0.02$ and further 10 timesteps of 0.004 each to arrive at $t = 0.06$. Results are

compared with the ones of the analytical model. For every situation the results are presented in natural and semi-logarithmic axes.

When results are compared in natural axes, the agreement between semi-analytical and numerical values is good in all cases. The plot in semi-log axes in figure 5.1 shows that results have different orders of magnitude for values of the *x* coordinate higher than 0.22, where concentrations are very small. From that point on, the semi-analytical model loses accuracy and for values of *x* higher than 0.6 it gives negatives values, which is physically meaningless. The order of magnitude of the difference between analytical and numerical results decreases as the time (and concentrations) increases.



$$t = 0.002 \frac{1}{timestep} \text{x} 10\, timesteps$$



$$t = 0.02 \frac{1}{timestep} \text{x} 1\, timestep$$

Figure 5.4: A 1D case. Profile of concentration at t = 0.02 using different timesteps. The velocities in this case are equal to 15.

Two factors have a decisive influence on the accuracy of the code: mesh and timestep. Coarse meshes can prevent the code to converge to a correct solution and a exceedingly fine mesh can turn the problem too cumbersome. As for the timestep, if not chosen properly it can introduce errors. For instance, figure 5.4 compares results using 10 timesteps of 0.002 and 1 timestep of 0.02 when using the velocities equal to 15. It can be seen that the solution with one time step of 0.02 does not produce accurate solution.

The criterion to adopt a timestep started by accepting that the time scale at the level of subdomains is determined by the coefficient of mechanical dispersion, *D*, the

absolute value of the velocity, V, and a characteristic length representative of the subdomains size, L. By applying basic dimensional analysis, the following non-dimensional product is proposed

$$\frac{\Delta t^2 DV}{L^2} \tag{5.15}$$

which is the relevant variable to assess accuracy regarding timestep size. Calibration tests carried out on the problem presented in this section showed that that the product (5.15) must be around 0.02. Note that if scaling is being used, the product (5.15) must be made with the scaled variables.

As for the size of the subdomains, no guideline can be obtained from theory since there is no theoretical proof of convergence for the BEM direct method. Here it has been adopted by trial and error, by refining the mesh until maximum variations between results with one and other mesh were lower than 5%.

When approaching large problems, a mesh and a set of scales must be proposed for every particular case. Although the 1D case tests showed that the code can solve several combinations of parameters and scales, it seems advisable, when the geometry allows it, to run a 1D case using every particular mesh, timestep and scale in the domain to be studied with the properties of the case, in order to check whether the numerical model is properly set up.

5.3 Flow in limestone: case 1

The first case presented in this chapter is the steady flow field in a domain consisting of limestone 700 m in length, 600 m width and 800 m depth, which is saturated with water (see figures 5.5 and 5.6). The terrain has a 5% slope from West to East. Inside the domain there is an empty room of $50 \times 50 \times 100$ m^3 dimension.

In this chapter, results are displayed using contour fills, which is a friendly way to visualize results. Contour fills presented in this thesis have been obtained by using GID™, a pre and postprocessor originally conceived for FEM applications. GID™ builds the contour plots from values of the variable defined at every geometrical

node. As all the examples that are presented in chapters 5 and 6 have been simulated using discontinuous elements, for which the nodes of freedom and geometrical nodes are in different positions in space, in order to produce the contour plots the DEC has to infer the values of the variable at every geometrical node from the values at the freedom nodes, where the solution is originally found. The DEC calculates the values of the variables at a given geometrical node as an average of the contribution of every tetrahedron side joining the node. The contribution of every side is interpolated using the corresponding shape function. This procedure sometimes introduces noticeable distortions in the display of results. For instance, in figures 5.18 and 5.31 the plume of pollutant seems to follow the tetrahedrons in some parts, which does not represent the situation if the results at nodes of freedom are considered. Therefore, as a complement to displaying contour plots it is convenient to show the time history of the variables in certain points in the domain. In order to do so, five observation points have been defined in the domain, which are nodes of freedom of the mesh. The positions of the observation points are summarized in Table 5.1 and are shown in figure 5.6.

Table 5.1: Case 1, coordinates of the observation points.

	O_1	O_2	O_3	O_4	O_5
x [m]	539	532	534	265	184
y [m]	281	293	316	301	302
z [m]	406	341	20	396	400

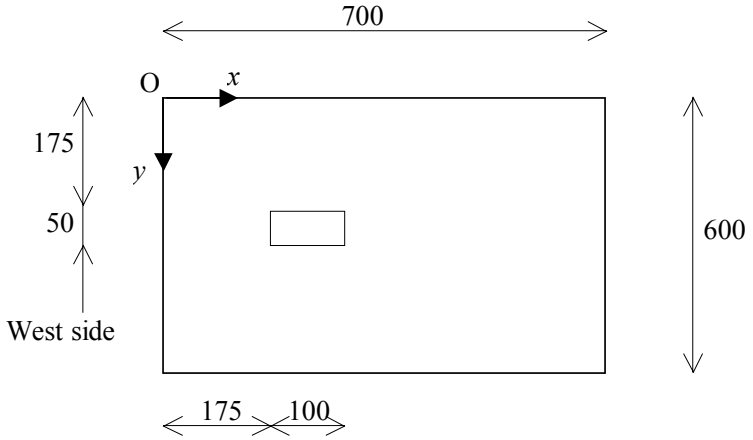


Figure 5.5: plan view of the domain. Dimensions are in m.

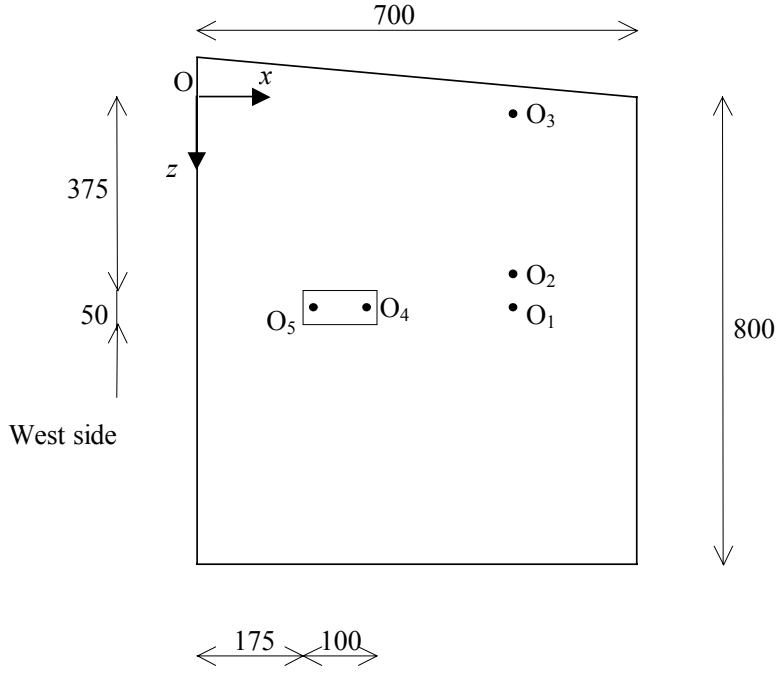


Figure 5.6: Case 1, side view of the domain. Dimensions are in m.

The boundary conditions on the external sides of the domain were hydraulic head equal to 35 m on the West side, zero on the East side and zero normal derivatives on all the other sides. As the pressure inside the room was considered to be equal to the atmospheric pressure, the boundary conditions on the walls of the room were hydraulic heads equal to the depth with negative sign, the depth being computed from the ground level. The hydraulic conductivity of the limestone was equal to 10^{-6} m/s and the effective porosity equal to 0.14.

The domain was subdivided into 2043 subdomains. The mesh can be seen in figure 5.7. Figures 5.8 to 5.10 show the results.

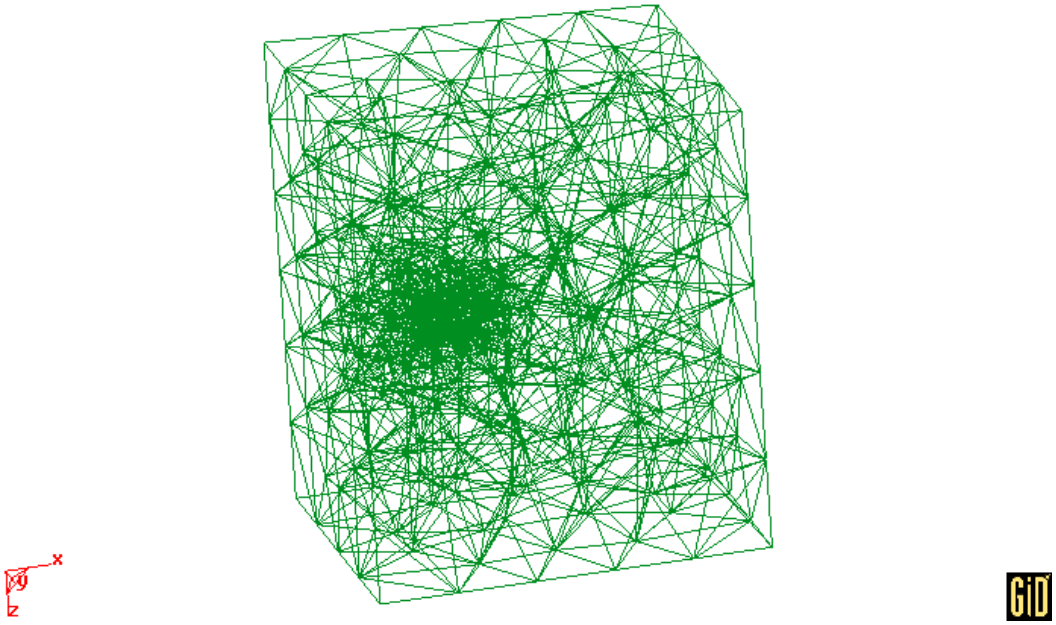


Figure 5.7: Case 1, mesh view.

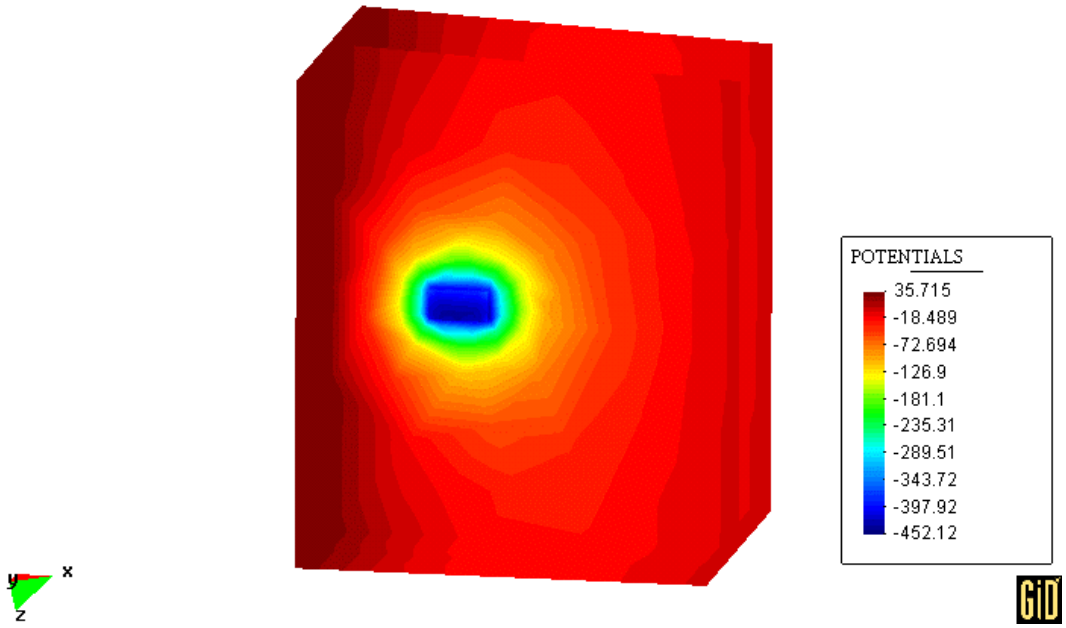


Figure 5.8: Case 1, hydraulic head contour fill [m].

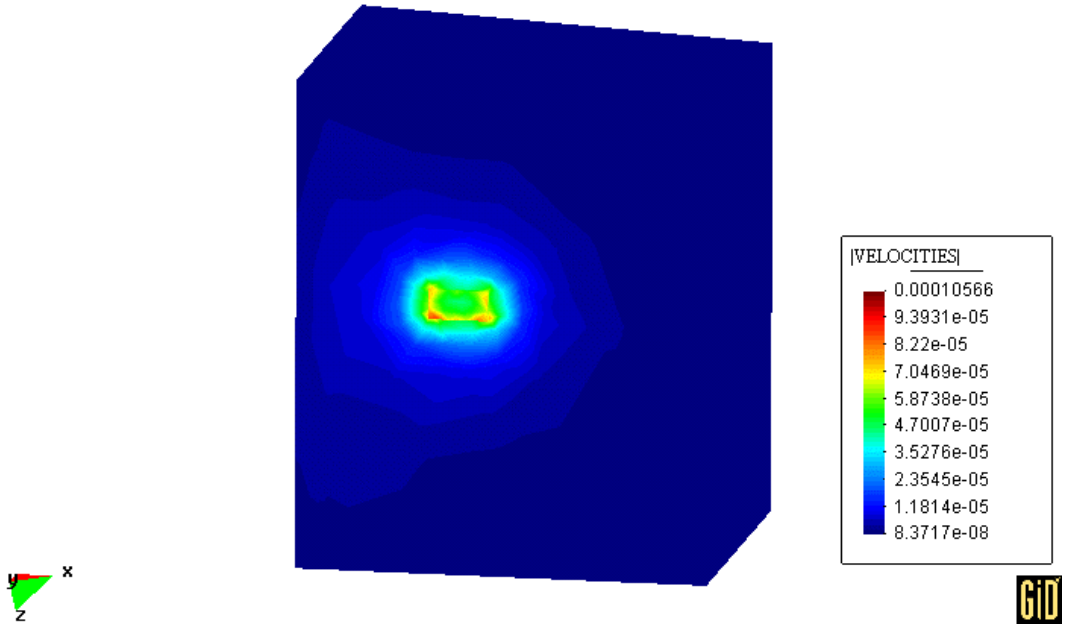


Figure 5.9: Case 1, module of velocities contour fill [m/s].

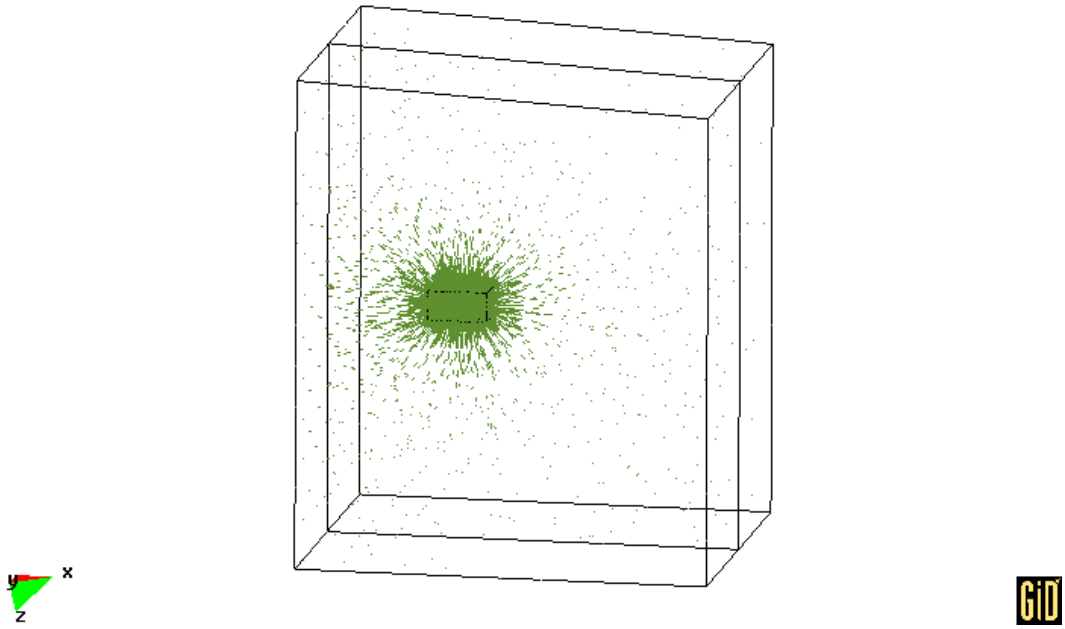


Figure 5.10: Case 1, velocity vectors.

The figures show how the water that surrounds the room is driven into it because of the lower pressure inside the room. Velocities around the room are of an order of magnitude of 5 m/d, while in the eastern part of the domain they are about 3 m/y. A more realistic model should have included a shaft, which would keep the

atmospheric pressure inside the room. The example is trivial but it shows the ability of the code to model steady flow, which is not surprising considering that the Laplace's equation governs the problem.

5.4 Flow and transport in limestone: case 2

The next case uses the same domain as the previous example. It is a simulation of transport of a pesticide stored inside the room.

The pesticide is Dichlorvos, an organophosphate insecticide with the chemical name 2,2-dichlorovinyl dimethyl phosphate. Common trade names are Astrobot, Atgard, Canogard, DDVP, and Vapona. As the half-life of Dichlorvos is very short compared to the time-scale considered for the safety assessment, it is the transport of its main decomposition component which is simulated: 2,2-dichloroacetic acid (DCA), which is at the same time the most toxic by-product of Dichlorvos.

The first scenario to be considered is that the room is filled with clay embedded in DCA at an initial concentration equal to 1, normalized concentration, and there is a steady flow field driven by the 5% slope of the terrain. The results will indicate the proportion of the initial concentration that can be expected to be found in the domain at certain simulation time. The values of the properties of the media that were adopted for this scenario are summarized in Table 5.2. It must be mentioned that the chapter is focused on the ability of the code to solve a possible case rather than assessing the real risk of burying pesticide in a repository surrounded by limestone.

Table 5.2: Case 2, adopted values of media properties.

	K [m/s]	R	D [m^2/s]	n_{eff}	n
Limestone	10^{-6}	1	10^{-6}	0.36	0.56
Clay	10^{-10}	1	10^{-10}	0.06	0.42

The flow model had the same boundary conditions on the external sides of the domain as the example presented in section 5.3. Boundary conditions on the room walls were not defined since the room itself is included in the domain. The domain

was subdivided into 2259 subdomains. The mesh, which is common to the transport model, can be seen in figure 5.11.

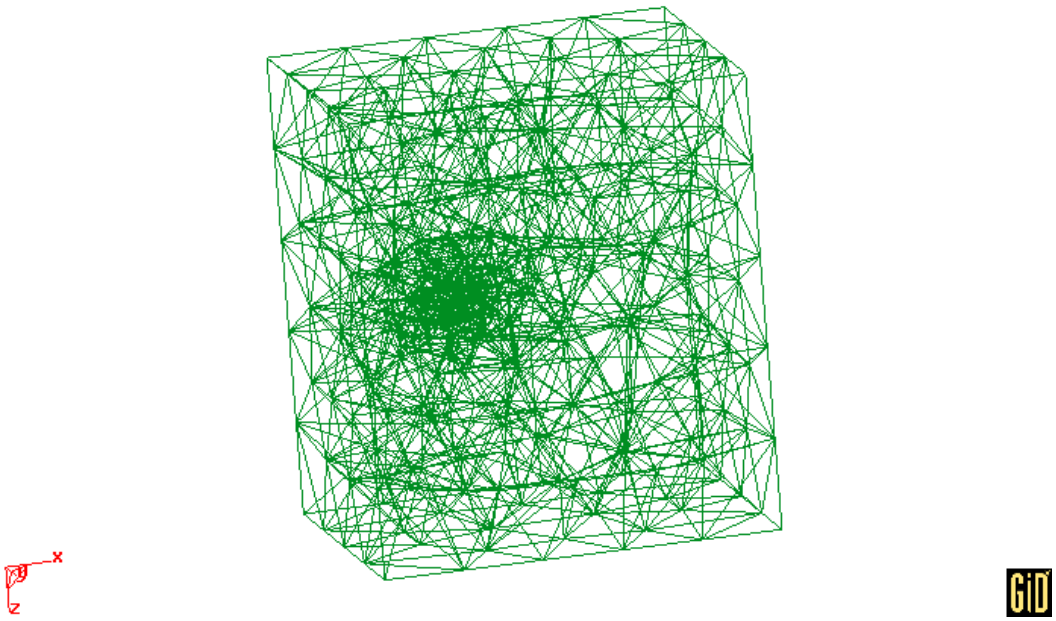


Figure 5.11: Case 2, mesh view.

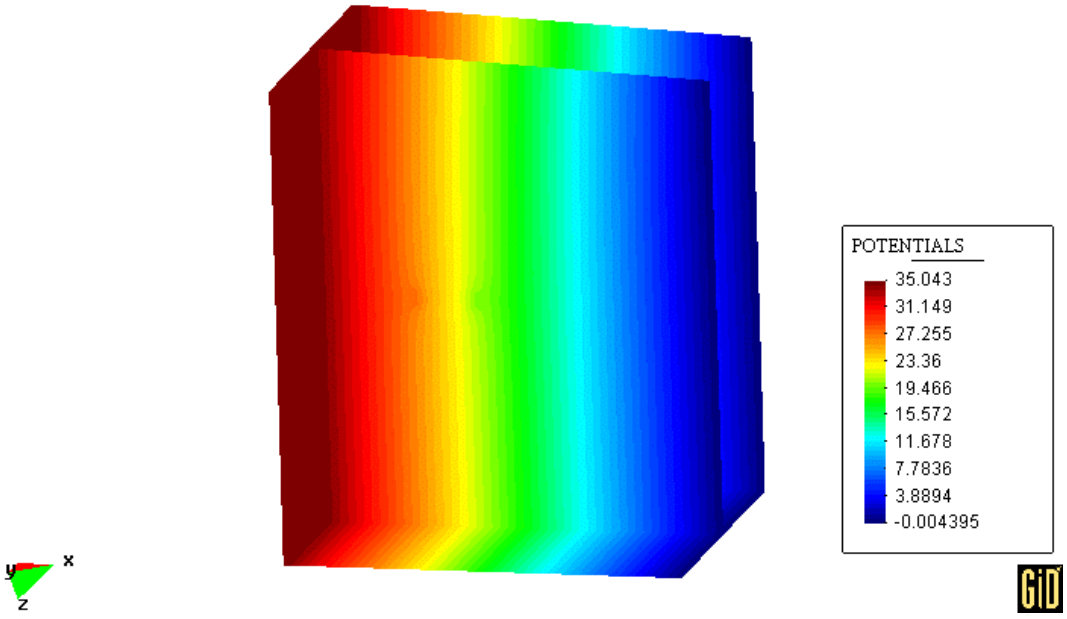


Figure 5.12: Case 2, hydraulic head contour fill [m].

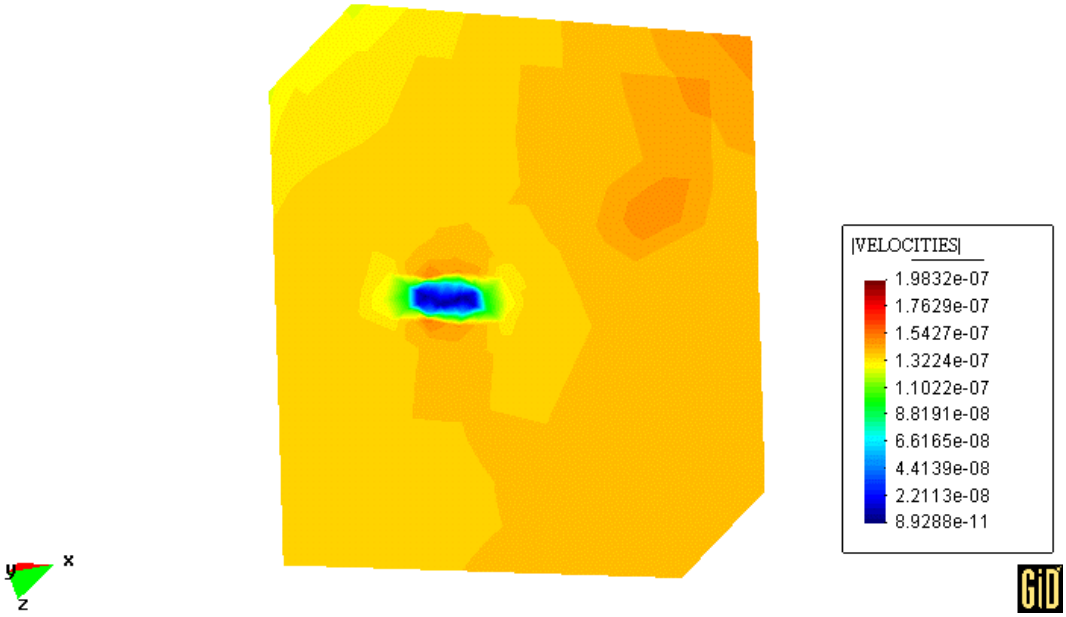


Figure 5.13: Case 2, module of velocities contour fill [m/s].

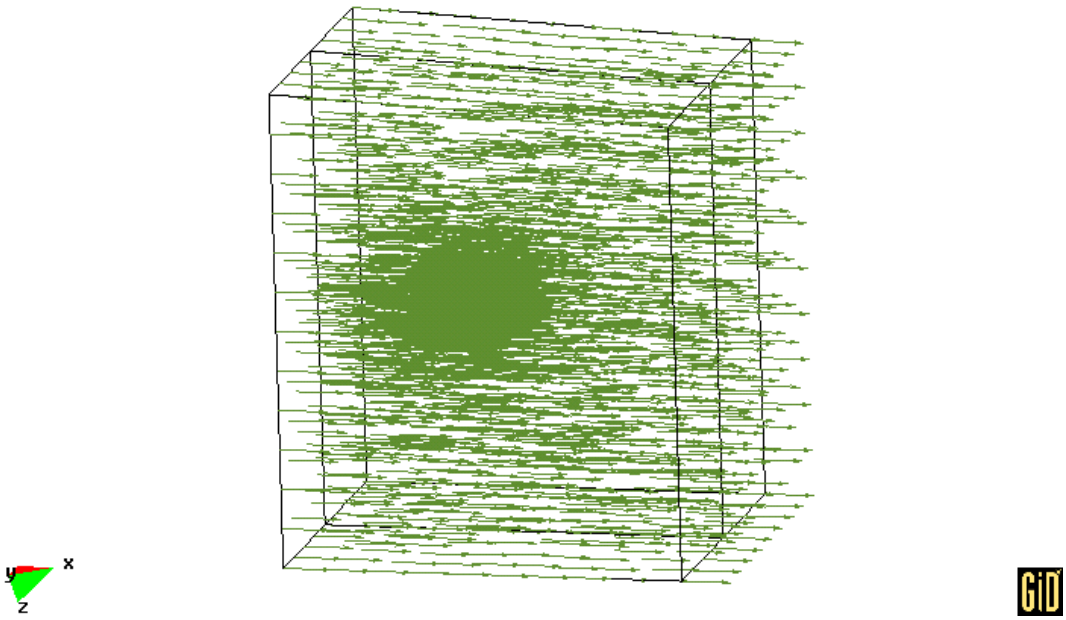


Figure. 5.14: Case 2, velocity vectors.

Figure 5.12 shows the hydraulic head field. There is an almost uniform gradient equal to 0.05 from West to East which is slightly perturbed around the room. Figure 5.13 displays the module of the velocities in m/s and Table 5.3 the velocities and their components at observation points; it can be seen that velocities are of an order

of magnitude of 4 m/y in the eastern part of the domain and 4 mm/y inside the room. Figure 5.14 shows the velocity vectors.

Table 5.3: Case 2, flow velocities at observation points.

	O_1	O_2	O_3	O_4	O_5
$\lvert\vec{V}\rvert$ [m/y]	4.4	4.4	4.5	4.6×10^{-3}	4.8×10^{-3}
V_x [m/y]	4.4	4.4	4.5	4.6×10^{-3}	4.8×10^{-3}
V_y [m/y]	1.5×10^{-4}	-6.3×10^{-4}	-8.9×10^{-4}	2.1×10^{-4}	9.3×10^{-5}
V_z [m/y]	2.8×10^{-2}	4.6×10^{-2}	1.9×10^{-1}	1.6×10^{-4}	5.0×10^{-5}

In order to test the mesh, scales and initial timesteps to be used, a 1D case was emulated by imposing the following boundary conditions on the domain: concentration 1 on the West side and zero normal derivatives on all the other sides. The initial condition was zero concentration inside the domain. A uniform steady velocity field parallel to the *x*-axis and equal to 1.4×10^{-7} m/s was imposed on the whole domain. The properties of the media corresponded to the ones of limestone displayed in Table 5.2. Figures 5.15 and 5.16 show results at 40 and 100 years. For every situation results are presented in natural and semi-logarithmic axes.

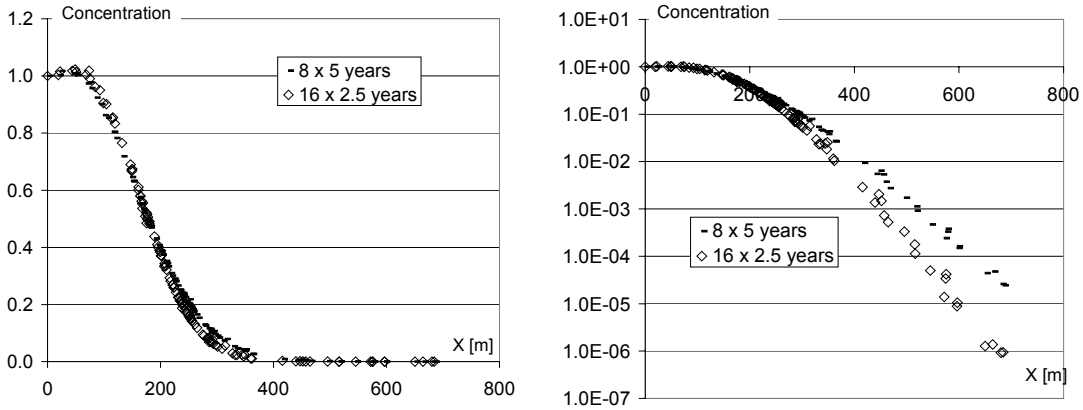


Figure 5.15: A 1D test case. Profile of concentration at t = 40 years using timesteps of 2.5 and 5 years. The velocities are equal to 1.4×10^{-7} m/s. The analytical model fails for this case.

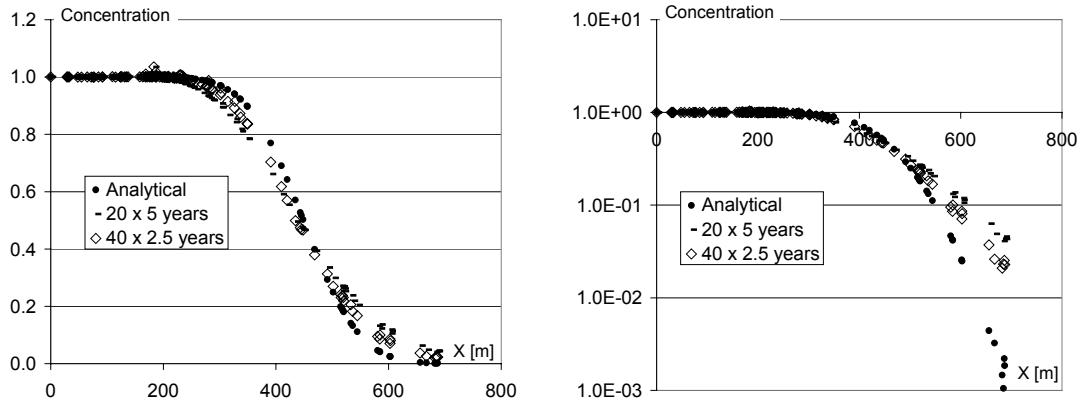
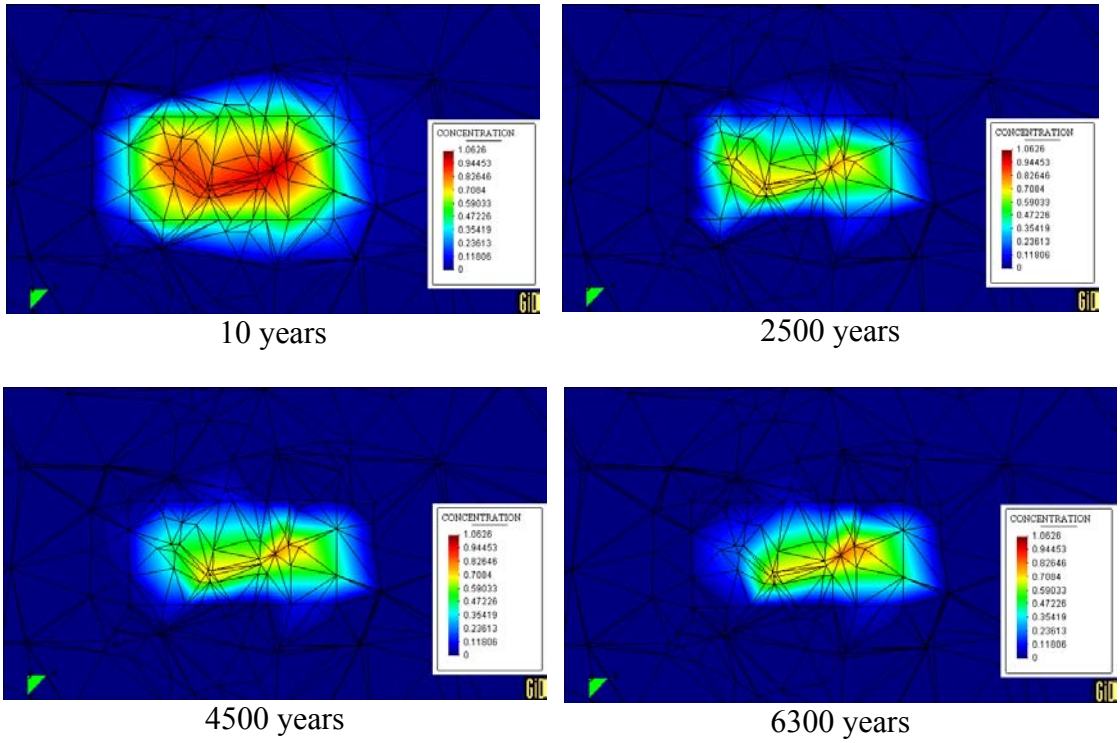


Figure 5.16: A 1D test case. Profile of concentration at t = 100 years using timesteps of 2.5 and 5 years. The velocities are equal to 1.4×10^{-7} m/s.

Boundary conditions for the transport model in case 2 were zero normal derivatives on all sides. The initial conditions were concentration 1 inside the room and concentration zero outside it. 6300 years of transport were simulated using the following sequence of iterations: 40 iterations of 2.5 years (40×2.5) plus 1240×5 years.

Figures 5.17 and 5.18 show concentration contour fills at different times. As concentrations inside and outside the room differ by several orders of magnitude, a single figure can not give a full picture of the evolution of concentration. The concentration change inside the room can be seen in figure 5.17. Figure 5.18, on the other hand, displays the concentration change outside the room.



10 years 2500 years

4500 years 6300 years

Figure 5.17: Concentration contour fills for case 2.

The evolution of the concentration at the observation points along 6300 years is displayed in figures 5.19 and 5.20. Figure 5.21 shows a more detailed story of concentrations during the first 300 years at the observation points 1, 2 and 3.

The contour plots show that the pollutant near the walls is carried away by the water during the first 100 years. In figure 5.18 it can be seen how this pollutant forms a cloud that is dragged towards the East by the water flow. Thereafter, the remaining pollutant migrates from the Western to the Eastern part of the room at the same rate as released into the host rock in a quasi-steady process.

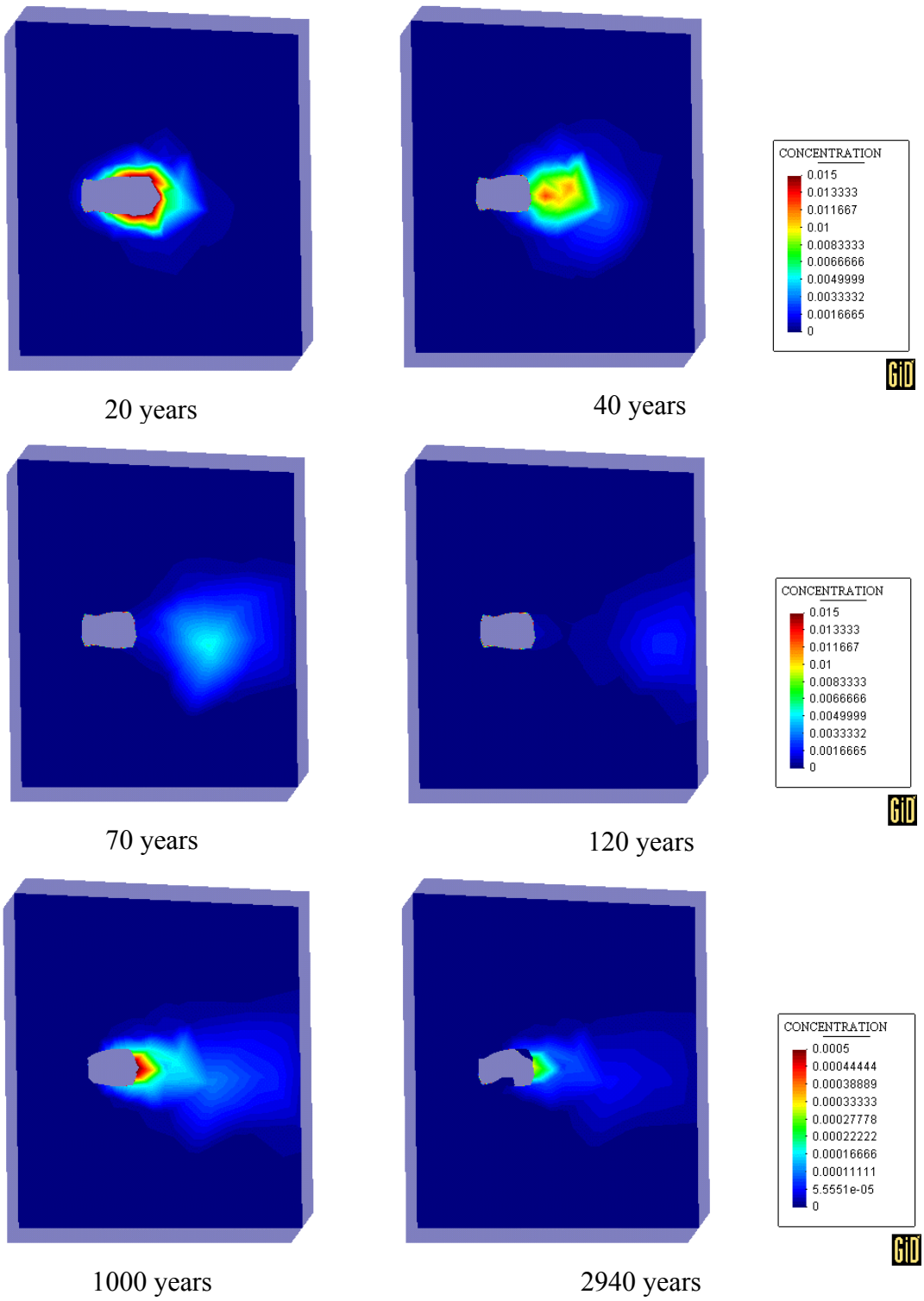


Figure 5.18: Concentration contour fills for case 2. Note that the colour scale of the two figures at the bottom is different than the one of the other four figures.

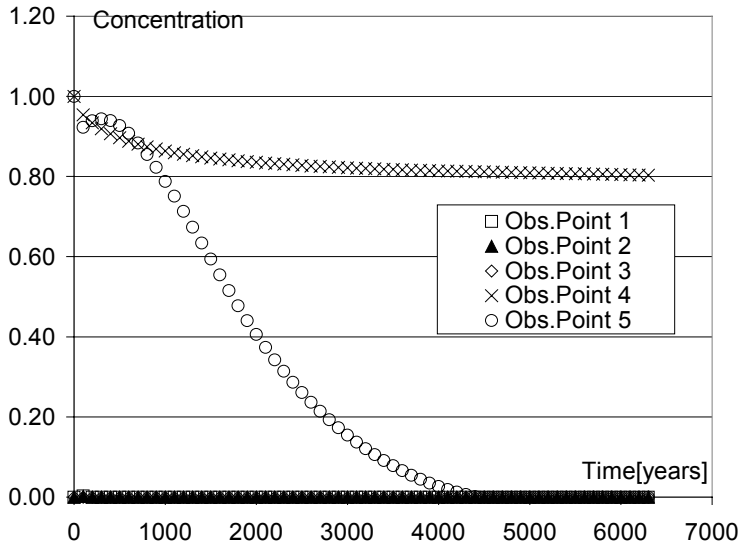


Figure 5.19: Case 2, concentration at observation points.

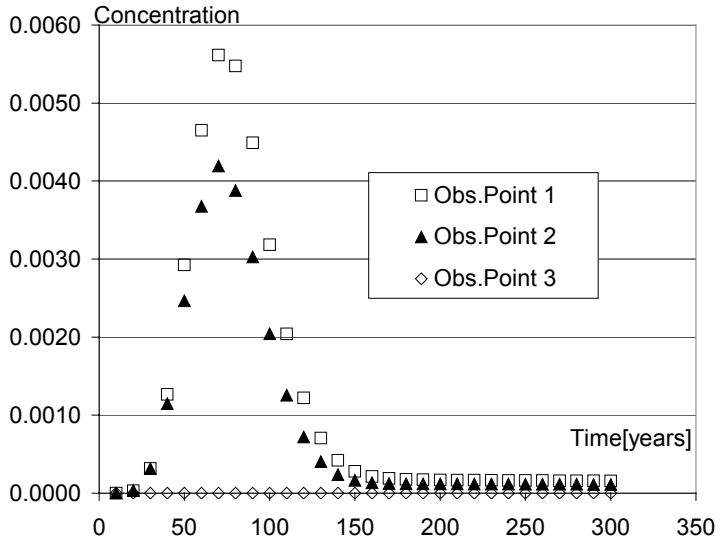


Figure 5.20: Case 2, concentration at observation points 1, 2 and 3 during the first 300 years.

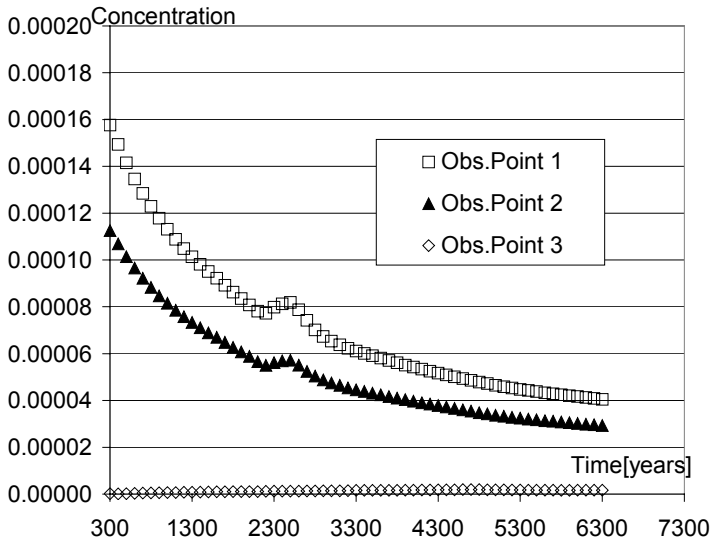


Figure 5.21: Case 2, concentrations at observation points.

The results presented in figures 5.19 to 5.21 quantify what has been observed in the contour plots. In the first 100 years appears a wave of pollutant caused by the DCA that is originally near the walls. Although the transport is mainly advective in both clay and limestone, at the beginning of the simulation it is predominantly diffusive near the walls of the room because of the high gradients of concentrations. Then, during the first years, the pollutant near the walls quickly reaches by diffusion the limestone, where flow velocities are three orders of magnitude higher than inside the room, and the contaminant is transported by the water flow. This first wave causes the highest peaks of concentration in O_1 and O_2 : 0.0056 and 0.0042 respectively in the year 70.

A more stable plume, with lower concentrations that slowly decrease with time, remains thereafter. After 1400 years the concentration in O_1 is below 10^{-4}. The concentration at observation point O_3 , which is near the ground level, has a peak of 1.9×10^{-6} in the year 4700, which means that it is virtually unaffected by the repository.

5.5 Transport in limestone: case 3

The case presented in this section is identical in every aspect to the previous case except by the fact that the pollutant is leaking during the first 1000 years instead of

being embedded in the clay inside the room since the year zero. This example is not equivalent to the previous case; it was proposed to see what would be the effect of releasing all the waste from the containers within 1000 years. The implementation of the leakage was made by adding a source term inside the room equal to 0.001/year during the first 1000 years. In this way, the total mass of pollutant appearing after the 1000 years is equal to the total mass of pollutant that is inside the room in case 2 at the beginning of the simulation. Figures 5.22 and 5.23 show concentrations at the observation points and figures 5.24 and 5.25 show contour fills of concentrations.

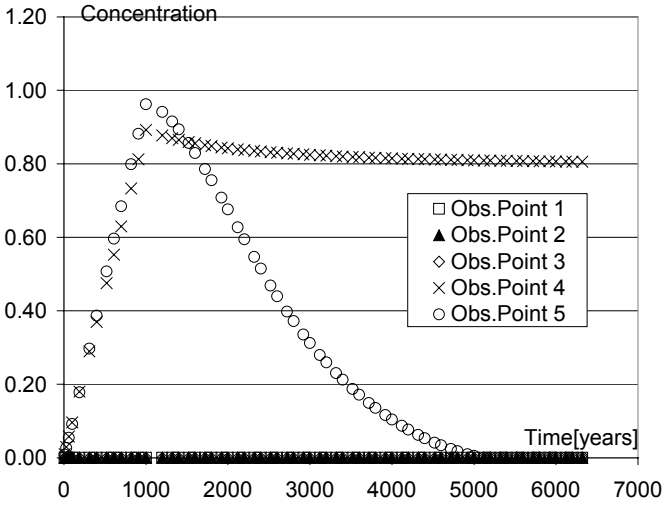


Figure 5.22: Case 3, concentrations at observation points.

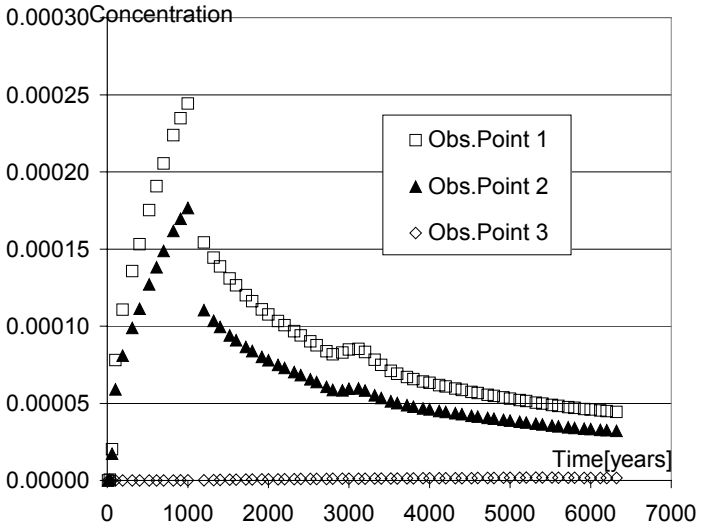


Figure 5.23: Case 3, concentrations at observation points.

Results show that inside the room the concentration of DCA increases during the first 1000 years due to the leakage. At the same time, the pollutant migrates from West to East, which causes the concentration at O_4 to decrease more slowly than at O_5. The wave of pollutant observed in case 2 does not appear in this case; as the leakage is progressive, the high gradients causing the migration of the pollutant near the wall is absent. Consequently, the peaks of concentration at observation points O_1 and O_2 are one order of magnitude smaller than in Case 2: 0.00024 and 0.00018 in the year 1000. Similarly to Case 2, O_3 has a peak of 1.8×10^{-6} but in the year 5220.

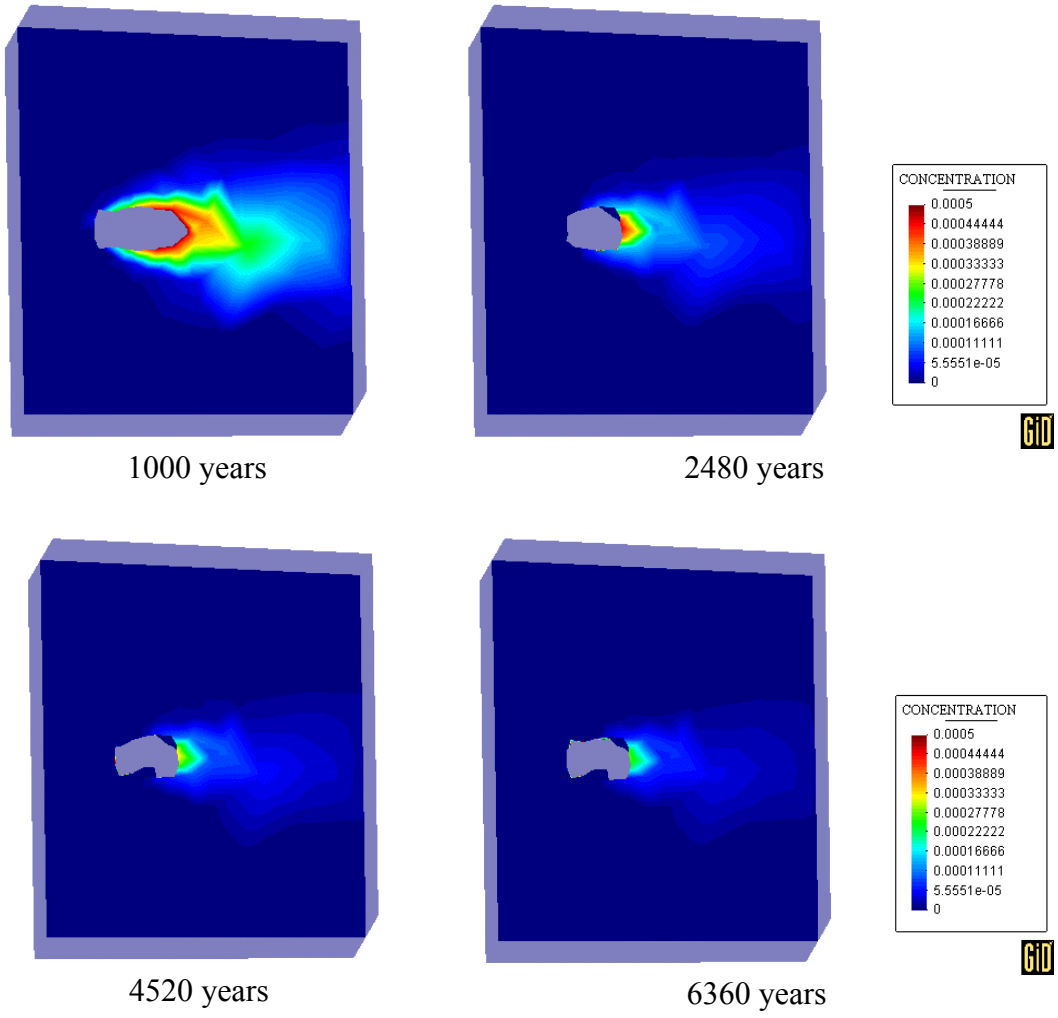


Figure 5.24: Case 3, concentration contour fills at different times.

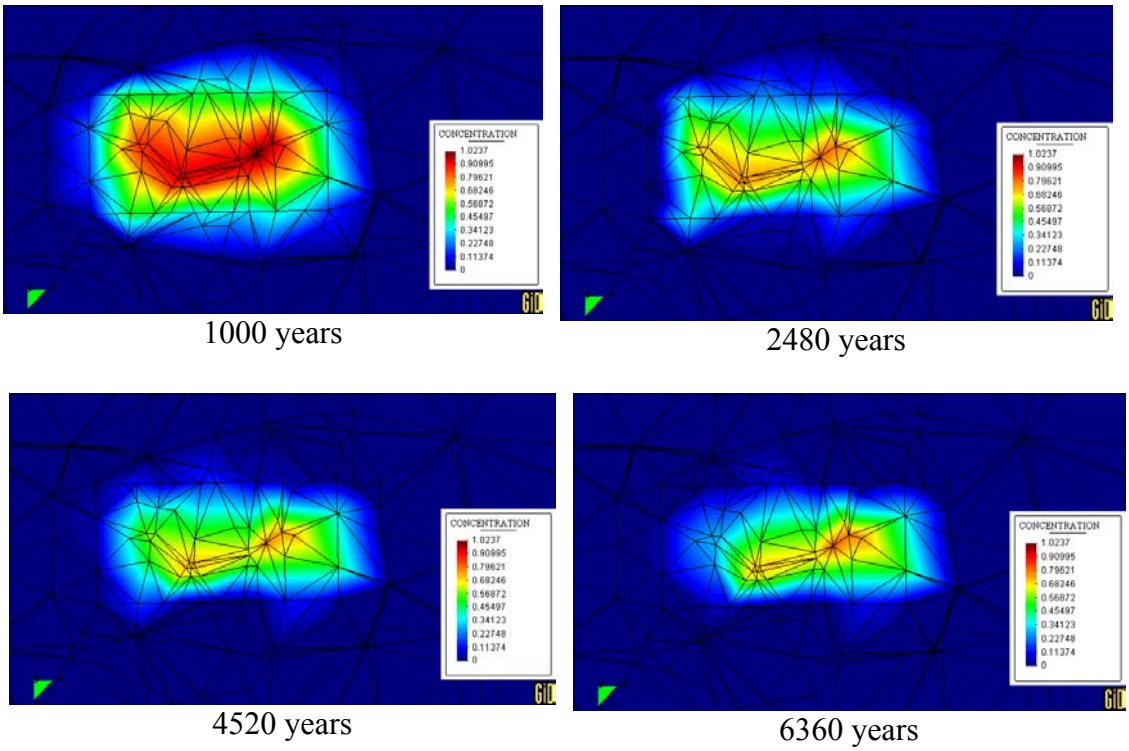


Figure 5.25: Case 3, concentration contour fills at different times.

5.6 Flow and transport in a heterogeneous domain: case 4

The case presented in this section incorporates two beds of a different kind of limestone. The domain, which can be seen in figures 5.26 and 5.27, is bigger in the *x*-direction than the previous ones, and the walls of the room are now not parallel to the vertical planes defined by the coordinate axes. The terrain, likewise the previous cases, has a 5% slope from West to East. Above the room, there is a 50 m thick bed of calcarenite, which is more permeable than the host rock. Underneath the room, there is a 50 m thick bed of dolomite, which is less permeable than the host rock. The beds have a noticeable slope from East to West and a much slighter slope from North to East. The position of the beds is displayed in figure 5.28. The properties of the media are shown in Table 5.3.

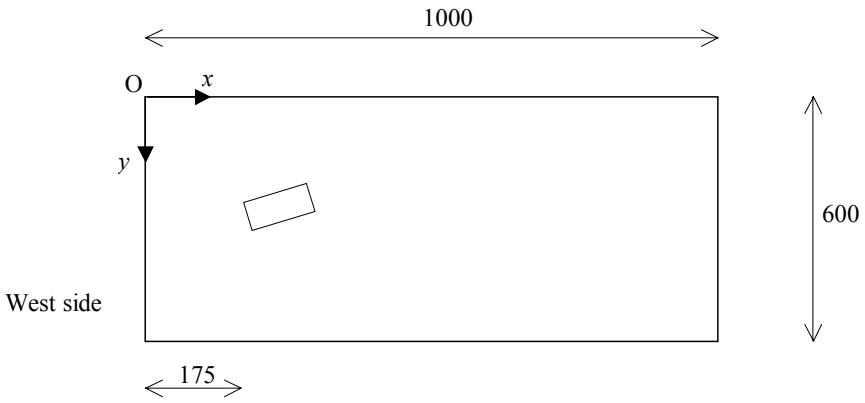


Figure 5.26: Case 4, plan view of the domain. Dimensions are in m.

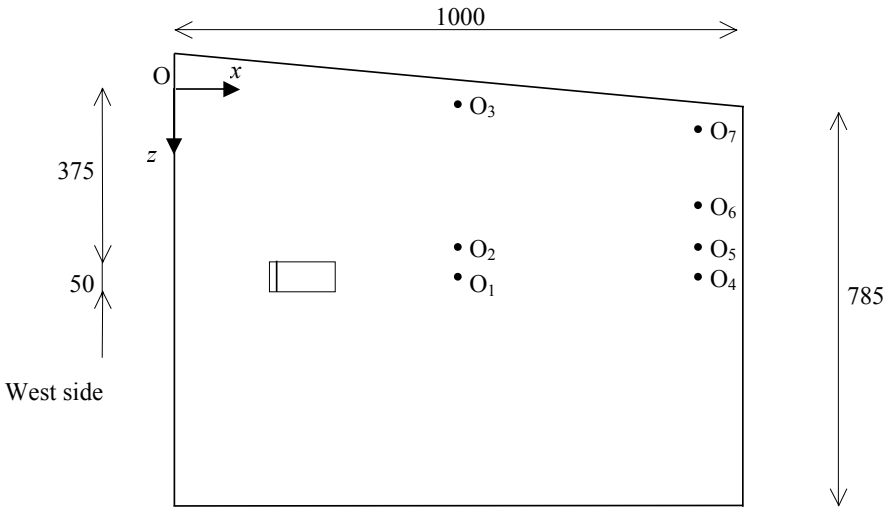


Figure 5.27: Case 4, side view of the domain. Dimensions are in m.

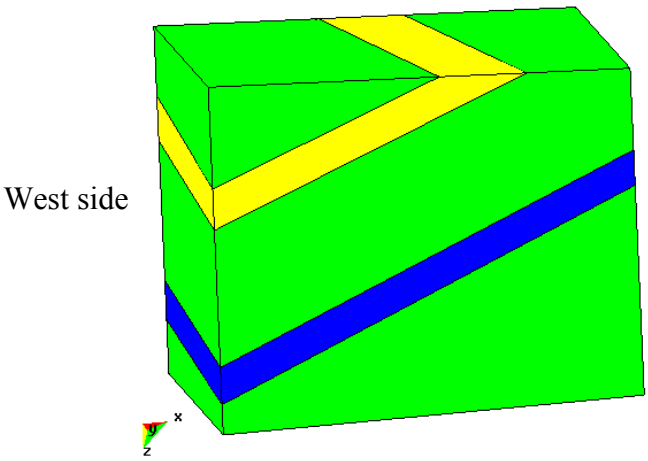


Figure 5.28: Case 4, view of the beds. The yellow bed is of calcarenite and the blue one is of dolomite.

Table 5.3: Case 4, adopted values of media properties.

	K [m/s]	R	D [m^2/s]	n_{eff}	n
Host limestone	10^{-6}	1	10^{-6}	0.36	0.56
Calcarenite	10^{-5}	1	10^{-5}	0.36	0.56
Dolomite	10^{-7}	1	10^{-7}	0.36	0.56
Clay	10^{-10}	1	10^{-10}	0.06	0.42

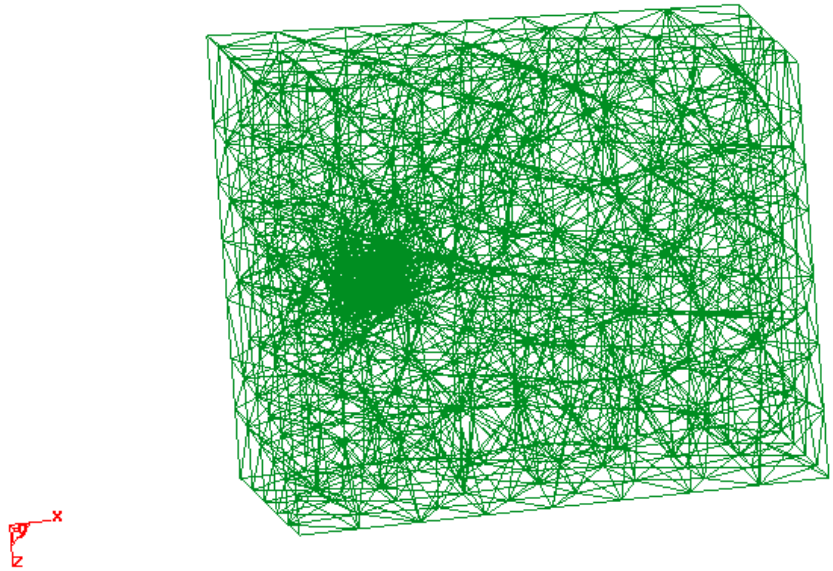


Figure 5.29: Case 4, View of the mesh.

The boundary conditions for the flow model were hydraulic head equal to 35 m on the West side, -15 m on the East side, a linear distribution on the upper side in agreement with the hydraulic head on the West and East sides, and zero normal derivatives on all the other sides. The domain was subdivided in 3157 subdomains. The mesh, which is common to the transport model, can be seen in figure 5.29.

Figures 5.30 to 5.32 show the results for water flow. It can be seen that the flow speeds up in the calcarenite bed and slows down in the dolomite bed. The flow is clearly driven towards ground level in the calcarenite bed. Tipically, velocities are 1.3×10^{-9} m/s (0.04 m/y) inside the room, 7×10^{-7} m/s (22 m/y) in the calcarenite bed, 7×10^{-8} m/s (2.2 m/y) in the dolomite bed and 1.4×10^{-7} m/s (4.4 m/y) in the host rock.

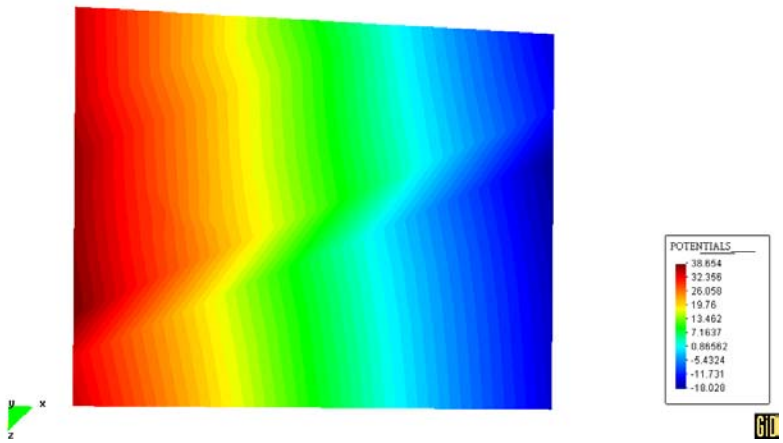


Figure 5.30: Case 4, hydraulic head contour plot in a vertical plane parallel to the xz-plane that intersects the room. The scale is in m.

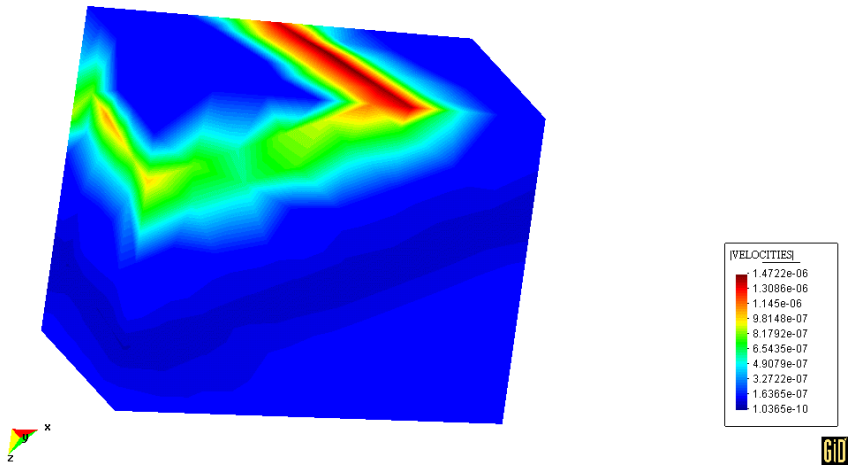


Figure 5.31: Case 4, contour plot of the velocity module. The scale is in m/s. The irregularities in the figure are due to the postprocessor (see section 5.3).

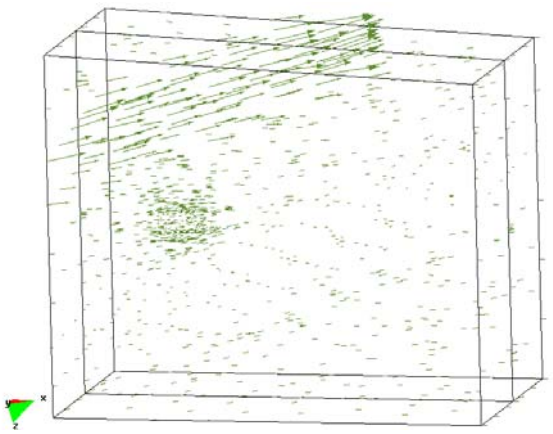


Figure 5.32: Case 4, velocity vectors.

Boundary conditions for the transport model were zero normal derivatives on all sides. The initial condition was zero concentration throughout the domain. The pollutant is supposed to leak during the first 500 years of simulation. Likewise in case 3, the implementation of the leakage was made by adding a source term inside the room. The source term was equal to 0.002/year and acted only during the first 500 years. In this way a linear increment of concentration from 0 at the beginning of the simulation to 1 in the year 500 was simulated.

Seven observation points, which are shown in figure 5.27, were defined in the domain. Their coordinates are shown in Table 5.4. The first three observation points are roughly analogous to the ones of cases 2 and 3; the other four ones are approximately 600 m downstream of the room. 2900 years of transport were simulated using 2 years timesteps, except for the first 100 years, where the timesteps were of 1 year.

Figure 5.33 displays concentrations at observation points. In figure 5.34 two cuts, a horizontal cut at z = 400 m and a vertical cut at y = 290 m, of the plume can be seen.

Table 5.4: Case 4, coordinates of the observation points.

	O_1	O_2	O_3	O_4	O_5	O_6	O_7
x [m]	523.0	528.5	523.1	851.2	850.3	847.5	852.6
y [m]	286.1	283.8	290.4	320.2	264.3	312.4	277.2
z [m]	419.7	333.3	20.2	391.1	374.8	192.0	25.5

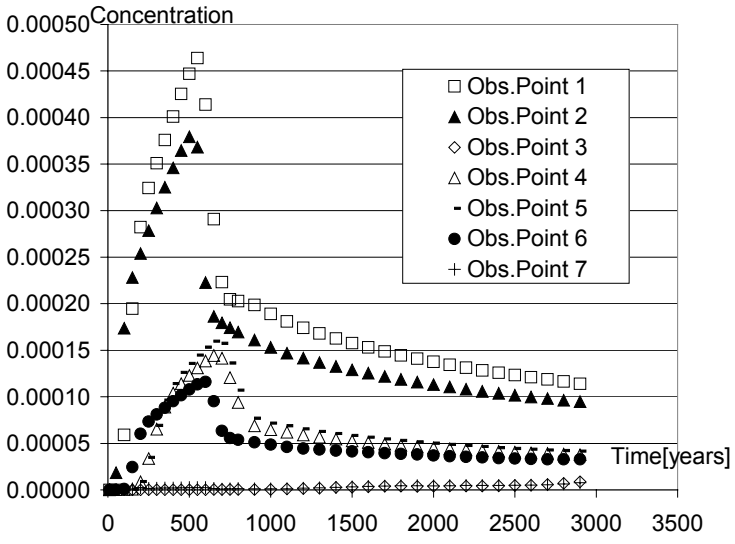


Figure 5.33: Case 4, concentration at observation points.

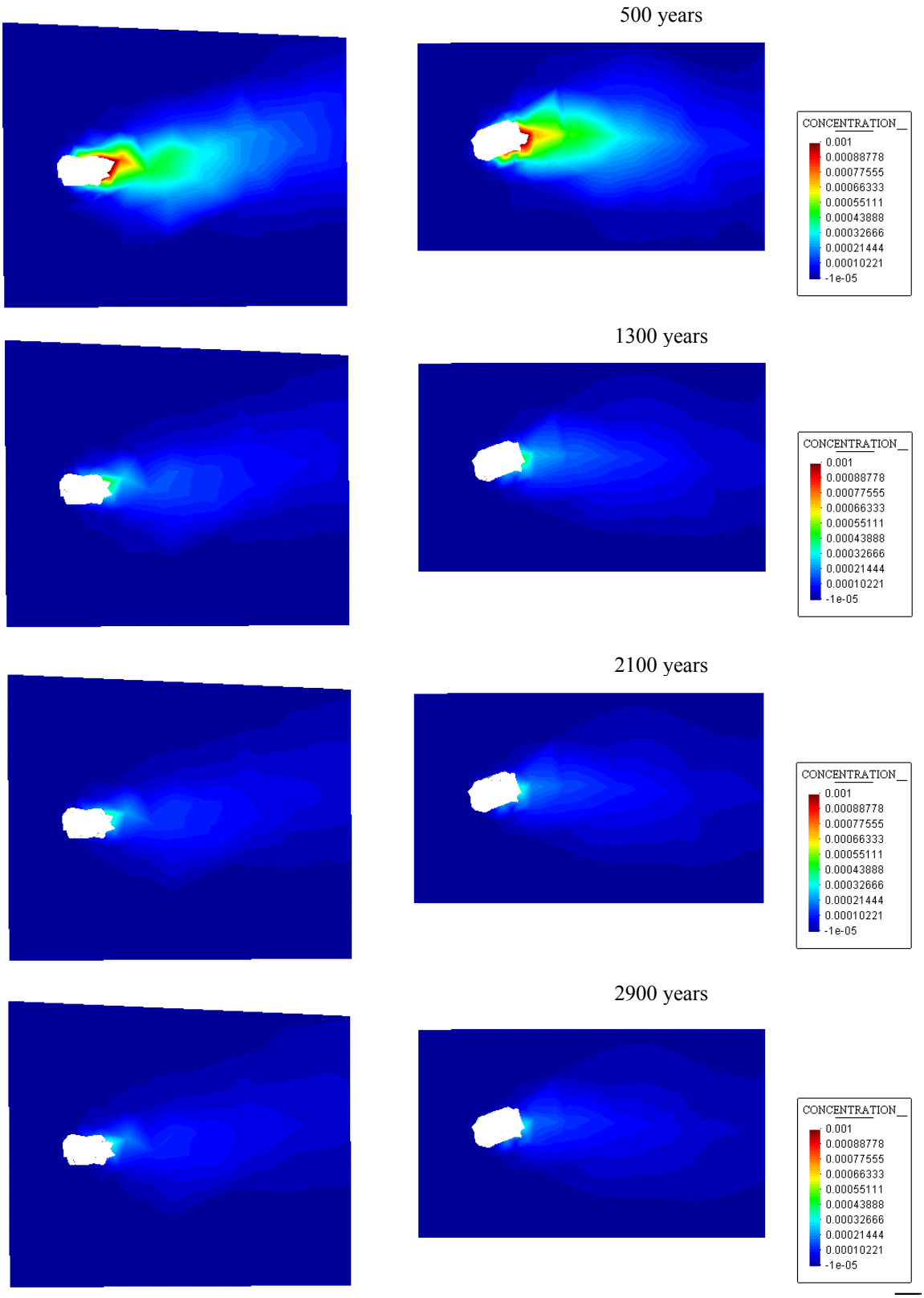


Figure 5.34: Case 4, contour plot of cuts of the plume. Left images correspond to a vertical cut at y = 290 m; right images to a horizontal cut at z = 400 m.

The most noticeable difference in regard with the previous cases is that the plume of DCA goes slightly upward. In terms of concentrations, the order of magnitude of what can be expected at observation points outside the domain is the same as in the previous cases: peaks of order 10^{-4} a few decades after the 500^{th} year that later drop around 10^{-5}. Table 5.5 presents the peaks of concentrations and the year of occurrence.

Table 5.5: Case 4, peak of concentration at observation points outside the room.

	O_1	O_2	O_3	O_4	O_5	O_6	O_7
Peak	4.6×10^{-4}	3.8×10^{-4}	1.0×10^{-5} still increasing at the end of the simulation	1.4×10^{-4}	1.6×10^{-4}	1.2×10^{-4}	8.7×10^{-6} still increasing at the end of the simulation
Time [years]	560	530		670	670	590	

5.7 Conclusions of the chapter

In this chapter, one case of flow and three cases of flow and transport in limestone have been presented. The cases were solved with a code using a DRM-MD algorithm, discontinuous elements, ATPS as DRM approximation function, the same scaling criterion as the one used in section 4.7 and no DRM nodes. All the examples were solved by the code in a common Pentium 4 PC even though a case such as case 4 required solving 1500 times a system of equation of 75768 degrees of freedom.

A linear finite difference approximation has been used, which is the most widely used in this advection-diffussion modelling using DRM, to evaluate the time derivative of the governing equation. The code showed to be sensitive to the size of timesteps. A rule of thumb which was calibrated in a problem with known semi-analytical solution was given in section 5.2 to adopt a time step. But, as practical problem have non homogeneous domains and meshes, every case must be controlled while running for instabilities appearing in the results. They can be identified in the contour plots as sudden outbursts of concentration, usually at very low levels. The instabilities can be eliminated by reducing the timesteps. For instance, preliminary tests showed that when using timesteps of 5 years or bigger in case 4, the solution

was spoiled by instabilities, which were eliminated when a timestep of 2 years was used.

Cases 2 to 4 analysed the transport of 2,2-dichloroacetic acid (DCA) in an underground repository built in a host rock of limestone. The domain was supposed to be saturated with water that was flowing from West to East driven by a 5% slope of the terrain. Several different scenarios were considered. In cases 2 and 3 the host rock was homogeneous, while in the case 4 two beds with different properties crossed the domain. In case 2 all the pollutant was embedded uniformly in the clay inside the room from the beginning of the simulation, while in cases 3 and 4 a source term simulates leakage along 1000 years, in case 3, and 500 years in case 4. Broadly speaking, the results showed that it can be expected that the DCA is slowly released into the host rock and that the plume of pollutant follows the flow direction. Concentrations reach peaks of an order of magnitude equal to 10^{-4} of the initial concentration inside the room in points located about 250 m directly downstream of the room. In another words, if the initial concentration is 10000 ppm of DCA, then, in points located about 250 m directly downstream of the room it can be expected to have maximum concentrations of order 1. Higher peaks of concentrations can be expected if the pollutant is uniformly embedding the clay from the beginning, because the pollutant which is near the walls quickly reaches the limestone by diffusion and is dragged by the water in the first years of the simulation. However, this is the most unrealistic situation since no repository is built in this way but with engineering barriers, which prevent the occurrence of this event.

The numerical model presented here showed to be a versatile tool for the analysis of this kind of problems. This have been widely confirmed by the work of Peratta and Popov [54, 55], who modelled massive problems of transport in fracture porous media using this scheme for the host rock. Comparisons with other numerical schemes in 2D problems can be found in ref. [21]. Many more examples could have been proposed regarding a repository in limestone. For instance, more attention should be paid to the engineering barriers. However, as assessing the safety of a repository requires precise information on field geological values, chemical processes, etc., which makes this a task for a team of researchers, the application of

geological science methods must always be considered in order to improve the understanding of computational 2D - 3D simulations, because most hazardous chemical reservoirs are being settled into different earth geosphere environments.

Chapter 6

Flow in unsaturated media

6.1 Introduction

In this part of the work a feasibility study was performed on the use of 3D DRM-MD for solution of the problem of wetting of clay in underground repositories. The study consisted of assuming that the air pressure is kept constant and equal to the atmospheric pressure, and then solving the equation for water. In this way it is tested whether the 3D DRM-MD can handle the type of non-linearities that this system of equations contains.

6.2 Governing equations

In this section a quick derivation of the governing equations is presented. It is considered that each phase occupies part of the domain and follows its own set of tortuous paths. A detailed treatment of the theory of flow in unsaturated media is given by Bear & Verruijt [49] and Helmig [56].

6.2.1 Equation for the water phase

The equation for mass balance is given as

$$\frac{\partial\left(n\rho_w S_w\right)}{\partial t} + \vec{\nabla}\cdot\rho_w\vec{q}_w = 0 \tag{6.1}$$

where:

n is the porosity

ρ_w is the water density

S_w is the water saturation

S_w is defined as the relation of the volume of water in a representative elementary volume (REV) and the volume of voids in the REV. S_w ranges from zero to one.
The specific discharge, for a Darcy flow, is defined as

$$\vec{q}_w = -\frac{k_w}{\mu_w}\left(\vec{\nabla} p_w - \rho_w g \vec{\nabla} z\right) \tag{6.2}$$

where:

k_w is the effective permeability for water (a function of S_w)

μ_w is the dynamic viscosity of water

p_w is the water pressure

z is the elevation

Note that (6.2) is valid provided the z-axis is pointing downward.
By substituting (6.2) in (6.1) the following equation is obtained

$$\frac{\partial\left(n\rho_w S_w\right)}{\partial t} + \vec{\nabla}\cdot\rho_w\left(-\frac{k_w}{\mu_w}\left(\vec{\nabla} p_w - \rho_w g\vec{\nabla} z\right)\right) = 0 \tag{6.3}$$

Onwards, n, ρ_w will be considered constants, then

$$n\rho_w \frac{\partial S_w}{\partial t} + \rho_w\vec{\nabla}\cdot\left(-\frac{k_w}{\mu_w}\vec{\nabla} p_w + \frac{k_w}{\mu_w}\rho_w g\vec{\nabla} z\right) = 0 \tag{6.4}$$

Developing the second term

$$n\rho_w \frac{\partial S_w}{\partial t} + \rho_w\left(-\frac{1}{\mu_w}\vec{\nabla} k_w\cdot\vec{\nabla} p_w - \frac{k_w}{\mu_w}\nabla^2 p_w + \frac{\rho_w g}{\mu_w}\vec{\nabla} k_w\cdot\vec{\nabla} z + \frac{k_w}{\mu_w}\rho_w g\overbrace{\nabla^2 z}^{0}\right) = 0 \tag{6.5}$$

and rearranging yields

$$\nabla^2 p_w = \frac{\mu_w}{k_w} n \frac{\partial S_w}{\partial t} - \frac{1}{k_w} \vec{\nabla} k_w \cdot \vec{\nabla} p_w + \frac{\rho_w g}{k_w} \vec{\nabla} k_w \cdot \vec{\nabla} z \tag{6.6}$$

On the other hand, the water relative permeability is defined as

$$k_{rw}(S_w) = \frac{k_w(S_w)}{k_w(1)} \tag{6.7}$$

As

$$k_w(1) = \frac{\mu_w K_w}{\rho_w g} \tag{6.8}$$

where K_w is the hydraulic conductivity for water, then

$$k_w(S_w) = \frac{k_{rw}(S_w) K_w \mu_w}{\rho_w g} \tag{6.9}$$

or using a shorter nomenclature

$$k_w = \frac{k_{rw} K_w \mu_w}{\rho_w g} \tag{6.10}$$

By substituting (6.10) in (6.6), the equation for the water phase is obtained

$$\nabla^2 p_w = \frac{n \rho_w g}{k_{rw} K_w} \frac{\partial S_w}{\partial t} - \frac{1}{k_{rw}} \vec{\nabla} k_{rw} \cdot \vec{\nabla} p_w + \frac{\rho_w g}{k_{rw}} \vec{\nabla} k_{rw} \cdot \vec{\nabla} z \tag{6.11}$$

Note that it is possible to formulate the equation for the water phase as it is in eqn. (6.6), however eqn. (6.11) is preferable since k_{rw} is non-dimensional and ranges from 0 to 1. Conversely, k_w has dimensions, which makes its order of magnitude

dependant of the scale factors, and computing the term $\frac{1}{k_w}\vec{\nabla}k_w$ leads to errors much bigger than computing $\frac{1}{k_{rw}}\vec{\nabla}k_{rw}$.

6.2.2 Equation for the air phase

Again, the starting point is the mass balance equation

$$\frac{\partial(n\rho_a S_a)}{\partial t} + \vec{\nabla}\cdot\rho_a\vec{q}_a = 0 \tag{6.12}$$

and

$$\vec{q}_a = -\frac{k_a}{\mu_a}\left(\vec{\nabla}p_a - \rho_a g\vec{\nabla}z\right) \tag{6.13}$$

where the nomenclature is analogous to the one of eqns. (6.1) and (6.2), the sub-index '*a*' identifying air properties. Note that

$$S_w + S_a = 1 \tag{6.14}$$

By substituting (6.13) into (6.12)

$$\frac{\partial(n\rho_a S_a)}{\partial t} + \vec{\nabla}\cdot\rho_a\left(-\frac{k_a}{\mu_a}\left(\vec{\nabla}p_a - \rho_a g\vec{\nabla}z\right)\right) = 0 \tag{6.15}$$

As *n* is constant and the gravitational term, $\rho_a\, g\vec{\nabla}z$, is neglected in the air equation, and $\frac{k_a}{\mu_a} = \frac{k_{ra}K_a}{\rho_a g}$

$$n\frac{\partial(\rho_a S_a)}{\partial t} - \frac{K_a}{g}\vec{\nabla}\cdot\left(k_{ra}\vec{\nabla}p_a\right) = 0 \tag{6.16}$$

Developing both terms in (6.16) yields

$$nS_a \frac{\partial \rho_a}{\partial t} + n\rho_a \frac{\partial S_a}{\partial t} - \frac{K_a}{g}\left(\vec{\nabla} k_{ra} \cdot \vec{\nabla} p_a + k_{ra} \nabla^2 p_a\right) = 0 \tag{6.17}$$

By rearranging, the equation for the air phase the following equation is obtained

$$\nabla^2 p_a = \frac{g}{k_{ra} K_a}\left(nS_a \frac{\partial \rho_a}{\partial t} + n\rho_a \frac{\partial S_a}{\partial t} - \frac{K_a}{g}\left(\vec{\nabla} k_{ra} \cdot \vec{\nabla} p_a\right)\right) \tag{6.18}$$

Under the conditions of an underground repository, we consider the air to be an ideal gas, in which ρ_a is a linked to p_a through the equation of state

$$p_a = \rho_a R' T \tag{6.19}$$

where R'is the particular gas constant and T is the absolute temperature. For instance, R' for dry air is $4.6447 \frac{mmHg \cdot m^3}{Kg \cdot^o K} = 599.29 \frac{m^2}{s^2 \cdot^o K}$.

By replacing (6.19) into (6.18) and rearranging, (6.18) becomes

$$\nabla^2 p_a = \frac{g}{k_{ra} K_a}\left[\frac{n}{R' T}\left(S_a \frac{\partial p_a}{\partial t} + p_a \frac{\partial S_a}{\partial t}\right) - \frac{K_a}{g}\left(\vec{\nabla} k_{ra} \cdot \vec{\nabla} p_a\right)\right] \tag{6.20}$$

The derivative in time of the saturations appearing in (6.11) and (6.20) can be handled in the following way:

$$\frac{\partial S_w}{\partial t} = \frac{dS_w}{dp_c}\frac{\partial p_c}{\partial t} = \frac{dS_w}{dp_c}\frac{\partial p_c}{\partial p_\gamma}\frac{\partial p_\gamma}{\partial t} \tag{6.21}$$

where subscript γ stands for "*w*" or "*a*" depending on the equation that is solved. Taking into account that $p_c = p_a - p_w$, it is obvious that $\partial p_c/\partial p_\gamma = (1 \text{ or } -1)$.
If furthermore we use (6.14) to eliminate S_a, the final equations will be:

$$\nabla^2 p_w = -\frac{n\rho_w g}{k_{rw}K_w}\frac{\partial S_w}{\partial p_c}\frac{\partial p_w}{\partial t} - \frac{1}{k_{rw}}\vec{\nabla}k_{rw}\cdot\vec{\nabla}p_w + \frac{\rho_w g}{k_{rw}}\vec{\nabla}k_{rw}\cdot\vec{\nabla}z \tag{6.22}$$

and

$$\nabla^2 p_a = \frac{g}{k_{ra}K_a}\left[\frac{n}{R'T}\left(\left(1-S_w\right) - p_a\frac{\partial S_w}{\partial p_c}\right)\frac{\partial p_a}{\partial t} - \frac{K_a}{g}\left(\vec{\nabla}k_{ra}\cdot\vec{\nabla}p_a\right)\right] \tag{6.23}$$

Equations (6.22) and (6.23) are the equations to be solved by the code, being p_w and p_a the unknown potential fields. The constants needed in the model are g, ρ_w, n, K_w, K_a, R' and T while k_{rw}, S_w and k_{ra} are functions of p_w and p_a. The functions linking the potential fields and k_{rw}, S_w and k_{ra} variables are given by the soil water retention curves.

6.3 Soil water retention curves

The soil water retention curve describes the relation between the capillary pressure, $p_{c,}$ and S_w. The capillary pressure is defined as

$$p_c = p_a - p_w \tag{6.24}$$

There are several functions that have been proposed to describe the soil-water retention curve; among the most popular for the air-water system are those given by Leverett [57], Brooks and Corey [58] and van Genuchten [59].

Before going into details let us define the effective saturation

$$S_e = \frac{S_w - S_{w0}}{1 - S_{w0} - S_{a0}} \tag{6.25}$$

The soil-water characteristic curve according to van Genuchten is [59, 56]

$$S_w = \begin{cases} S_{w0} + \dfrac{1 - S_{w0} - S_{a0}}{\left(1 + \left(\mathrm{a} p_c\right)^{\mathrm{n}}\right)^{\mathrm{m}}} & \textit{when} \quad p_c > 0 \\ 1 & \textit{when} \quad p_c \le 0 \end{cases} \tag{6.26}$$

where S_{w0} is the irreducible water saturation, and S_{a0} is the residual air saturation. There are also three parameters involved: a, n and m. Usually, authors use expressions with simple relation between n and m, e.g., Burdine [60] (m = 1 - 2/n), Mualem [61] (m = 1 - 1/n).

Van Genuchten's function for the relative water permeability as a function of saturation is given as

$$k_{rw} = \sqrt{S_e}\left(1 - \left(1 - S_e^{1/\mathrm{m}}\right)^{\mathrm{m}}\right)^2 \tag{6.27}$$

and the relative air permeability as a function of saturation

$$k_{ra} = \left(1 - S_e\right)^{\frac{1}{3}}\left(1 - S_e^{\,1/\mathrm{m}}\right)^{2\mathrm{m}} \tag{6.28}$$

Recently, an enhanced version of eqns. (6.26)(6.27) and (6.28) has been proposed by Vogel et al. [62, 63].

$$S_w = \begin{cases} S_{w0} + \dfrac{S_m - S_{w0} - S_{a0}}{\left(1 + (\mathrm{a}p_c)^{\mathrm{n}}\right)^{\mathrm{m}}} & \text{when} \quad p_c > p_s \\ 1 & \text{when} \quad p_c \le p_s \end{cases} \tag{6.29}$$

where S_m is a fictitious extrapolated parameter; $S_m > 1$, and p_s is called the minimum capillary pressure. The modified van Genuchten's relative water permeabilities as a function of saturations are

$$k_{rw} = \sqrt{S_e}\left(\frac{1 - F(S_e)}{1 - F(1)}\right)^2 \tag{6.30}$$

and

$$k_{ra} = (1 - S_e)^{\frac{1}{2}}\left(\frac{F(S_e) - F(1)}{1 - F(1)}\right)^2 \tag{6.31}$$

where

$$F(S_e) = \left(1 - S_e^{*1/\mathrm{m}}\right)^{\mathrm{m}} \tag{6.32}$$

and

$$S_e^* = \frac{1 - S_{w0}}{S_m - S_{w0} - S_{a0}} S_e \tag{6.33}$$

Eqns. (6.26), (6.29) and (6.33) have been originally proposed in terms of water contents; here they have been modified in order to match with the definition (6.25), which takes into account S_{a0}, and the relative permeability of the air phase. Eqns. (6.26), (6.27) and (6.28) will be referred to as the original VGM and the eqns. (6.29),(6.30) and (6.31) will be referred to as the modified VGM. In other models, the modified VGM eliminates numerical instabilities appearing near saturation.

Another soil water model that has been extensively used is due to Brooks and Corey [58]. It is defined by

$$S_w = \begin{cases} S_{w0} + (1 - S_{w0} - S_{a0}) \left(\dfrac{p_b}{p_c} \right)^{\lambda} & when \quad p_c > p_b \\ 1 & when \quad p_c \le p_b \end{cases} \tag{6.34}$$

$$k_{rw} = S_e^{\frac{2+3\lambda}{\lambda}} \tag{6.35}$$

and

$$k_{aw} = (1 - S_e)^2 \left(1 - S_e^{\frac{2+\lambda}{\lambda}} \right) \tag{6.36}$$

where p_c is a parameter called the air entry pressure head, and λ is a constant exponent.

6.4 Solving the system

When considering the simultaneous flow of both the water and the air in the unsaturated zone, the system of equations is represented by (6.22) and (6.23) together with those representing the soil water model. In the feasibility study, only the equation for the water phase was considered. In order to do so, the entire air phase is considered to be stationary at atmospheric pressure and only eqn. (6.22) is solved; thus the case becomes equivalent to solving the Richard´s equation. All the examples presented in this chapter relate to Richard's equation.

As in the case presented in the previous chapter, the code obtains solutions at different time-steps by using a linear time finite difference approximation. As the equations are non-linear, in each time-step an iterative procedure is required. The code starts by calculating p_c from the initial conditions of the problem, then S_w and k_{rw} are calculated and the eqn (6.22) is solved; in the next iteration, with the value of p_w that has been obtained, a new value for p_c is calculated, then S_w and k_{rw} are recalculated and the eqn. (6.22) is solved again. The process is repeated until convergence is reached and thereafter a new timestep is started.

The derivative of S_w in respect to p_c appearing in the equations will depend on the soil water retention model, which is used. Equations (6.37), (6.38) and (6.39) are the $\frac{\partial S_w}{\partial p_c}$ for original VGM, modified VGM and Brooks and Corey model respectively.

$$\frac{\partial S_w}{\partial p_c} = \begin{cases} -\dfrac{\mathrm{nma}\left(1 - S_{w0} - S_{a0}\right)\left(\mathrm{a}p_c\right)^{\mathrm{n}-1}}{\left(1 + \left(\mathrm{a}p_c\right)^{\mathrm{n}}\right)^{\mathrm{m}+1}} & when \quad p_c > 0 \\ 0 & when \quad p_c \le 0 \end{cases} \tag{6.37}$$

$$\frac{\partial S_w}{\partial p_c} = \begin{cases} -\dfrac{\mathrm{nma}\left(S_m - S_{w0} - S_{a0}\right)\left(\mathrm{a}p_c\right)^{\mathrm{n}-1}}{\left(1 + \left(\mathrm{a}p_c\right)^{\mathrm{n}}\right)^{\mathrm{m}+1}} & when \quad p_c > p_s \\ 0 & when \quad p_c \le p_s \end{cases} \tag{6.38}$$

$$\frac{\partial S_w}{\partial p_c} = \begin{cases} -\dfrac{\lambda\left(1 - S_{w0} - S_{a0}\right)}{p_c}\left(\dfrac{p_b}{p_c}\right)^{\lambda} & when \quad p_c > p_b \\ 0 & when \quad p_c \le p_b \end{cases} \tag{6.39}$$

6.5 DRM/MD implementation for the water phase

The DRM integral formulation for p_w is obtained by replacing the right-hand-side of eqn (6.22) into (2.14)

$$\sum_{j=1}^{n} h_{ij}\, p_{w_j} - \sum_{j=1}^{n} g_{ij}\, q_{w_j} = \sum_{j=1}^{n} s_{ij} \left[\begin{array}{l} -\dfrac{n\rho_w g}{\widetilde{k}_{rw_j} K_w} \dfrac{\partial \widetilde{S}_{w_j}}{\partial p_c} \dfrac{\partial p_{w_j}}{\partial t} - \\ \dfrac{1}{\widetilde{k}_{w_j}} \left(\dfrac{\partial \widetilde{k}_{rw_j}}{\partial x} \dfrac{\partial p_{w_j}}{\partial x} + \dfrac{\partial \widetilde{k}_{rw_j}}{\partial y} \dfrac{\partial p_{w_j}}{\partial y} + \dfrac{\partial \widetilde{k}_{rw_j}}{\partial z} \dfrac{\partial p_{w_j}}{\partial z} \right) \end{array} \right] + \sum_{j=1}^{n} s_{ij} \dfrac{\rho_w g}{\widetilde{k}_{rw_j}} \dfrac{\partial \widetilde{k}_{rw_j}}{\partial z} \tag{6.40}$$

where

$\widetilde{k}_{rw_j}$ and $\dfrac{\partial \widetilde{S}_{w_j}}{\partial p_c}$ are calculated using values of $p_a = p_{atm}$ and p_w coming from the previous iteration, which will be denoted by $\widetilde{p}_w$ from here on.

$\dfrac{\partial p_{w_j}}{\partial x}$ and all the others partial derivatives are obtained applying (3.9), which in index notation is

$$\frac{\partial p_{w_j}}{\partial x} = \sum_{k=1}^{n} \sum_{l=1}^{n} \frac{\partial f_{jl}}{\partial x} f_{lk}^{-1} p_{w_k} \tag{6.41}$$

Let us introduce the time finite difference approximation, similarly as it was done in section 6.1 for the transient advection-diffusion equation.

$$p_w = \left(1 - \theta_{p_w}\right) {p_w}^{\mathrm{m}} + \theta_{p_w} {p_w}^{\mathrm{m+1}} \tag{6.42}$$

$$q_w = \left(1 - \theta_{q_w}\right) {q_w}^{\mathrm{m}} + \theta_{q_w} {q_w}^{\mathrm{m+1}} \tag{6.43}$$

$$\frac{\partial p_w}{\partial t} = \frac{1}{\Delta t} \left({p_w}^{\mathrm{m+1}} - {p_w}^{\mathrm{m}} \right) \tag{6.44}$$

Note that while for the advection-diffusion equation the application of the time marching scheme represented by the last three equations has been based on previous work, in this case there is no previous work on which to base the time marching scheme, because it is the first time that the DRM is applied to the solution of unsaturated flow.

Let us replace the time finite difference approximation into eqn. (6.40) term by term and express them in matrix notation.

For the left-hand side

$$\sum_{j=1}^{n} h_{ij}\, p_{w_j} - \sum_{j=1}^{n} g_{ij}\, q_{w_j} = \mathbf{H}\left[\left(1-\theta_{p_w}\right)\boldsymbol{p}_w{}^{\mathrm{m}} + \theta_{p_w}\boldsymbol{p}_w{}^{\mathrm{m+1}}\right] - \mathbf{G}\left[\left(1-\theta_{q_w}\right)\boldsymbol{q}_w{}^{\mathrm{m}} + \theta_{q_w}\boldsymbol{q}_w{}^{\mathrm{m+1}}\right] \tag{6.45}$$

For the first member of the right-hand side

$$\sum_{j=1}^{n} s_{ij}\left[-\frac{n\rho_w g}{\widetilde{k}_{rw_j} K_w}\frac{\partial \widetilde{S}_{w_j}}{\partial p_c}\frac{\partial p_{w_j}}{\partial t}\right] = -\widetilde{\mathbf{R}}_w\left(\frac{1}{\Delta t}\left(\boldsymbol{p}_w{}^{\mathrm{m+1}} - \boldsymbol{p}_w{}^{\mathrm{m}}\right)\right) \tag{6.46}$$

where $\widetilde{\mathbf{R}}_w$ is a matrix of components

$$\widetilde{r}_{w_{ij}} = s_{ij}\frac{n\rho_w g}{\widetilde{k}_{rw_j} K_w}\frac{\partial \widetilde{S}_{w_j}}{\partial p_c} \tag{6.47}$$

For the second member of the right-hand-side

$$\sum_{j=1}^{n} s_{ij}\left[\frac{1}{\widetilde{k}_{w_j}}\left(\frac{\partial\widetilde{k}_{rw_j}}{\partial x}\frac{\partial p_{w_j}}{\partial x}+\frac{\partial\widetilde{k}_{rw_j}}{\partial y}\frac{\partial p_{w_j}}{\partial y}+\frac{\partial\widetilde{k}_{rw_j}}{\partial z}\frac{\partial p_{w_j}}{\partial z}\right)\right]=$$

$$\sum_{j=1}^{n} s_{ij}\left[\frac{1}{\widetilde{k}_{rw_j}}\left(\frac{\partial\widetilde{k}_{rw_j}}{\partial x}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial x}f_{lk}^{-1}+\frac{\partial\widetilde{k}_{rw_j}}{\partial y}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial y}f_{lk}^{-1}+\frac{\partial\widetilde{k}_{rw_j}}{\partial z}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial z}f_{lk}^{-1}\right)p_{w_k}\right]$$

(6.48)

Let us define the matrix $\widetilde{\mathbf{D}}_x$, with components

$$d_{ij}=s_{ij}\frac{1}{\widetilde{k}_{rw_j}}\frac{\partial\widetilde{k}_{rw_j}}{\partial x} \tag{6.49}$$

and analogous $\widetilde{\mathbf{D}}_y$ and $\widetilde{\mathbf{D}}_z$ matrices, then

$$\sum_{j=1}^{n} s_{ij}\left[\frac{1}{\widetilde{k}_{rw_j}}\left(\frac{\partial\widetilde{k}_{rw_j}}{\partial x}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial x}f_{lk}^{-1}+\frac{\partial\widetilde{k}_{rw_j}}{\partial y}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial y}f_{lk}^{-1}+\frac{\partial\widetilde{k}_{rw_j}}{\partial z}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial f_{jl}}{\partial z}f_{lk}^{-1}\right)p_{w_k}\right]$$
$$=\left(\widetilde{\mathbf{D}}_x\frac{\partial\mathbf{F}}{\partial x}\mathbf{F}^{-1}+\widetilde{\mathbf{D}}_y\frac{\partial\mathbf{F}}{\partial y}\mathbf{F}^{-1}+\widetilde{\mathbf{D}}_z\frac{\partial\mathbf{F}}{\partial z}\mathbf{F}^{-1}\right)\left[\left(1-\theta_{p_w}\right)\boldsymbol{p}_w{}^{\mathrm{m}}+\theta_{p_w}\boldsymbol{p}_w{}^{\mathrm{m+1}}\right]$$

(6.50)

If a matrix $\widetilde{\mathbf{T}}\mathbf{w}$ is defined, being

$$\widetilde{\mathbf{T}}\mathbf{w}-\left(\widetilde{\mathbf{D}}_x\frac{\partial\mathbf{F}}{\partial x}\mathbf{F}^{-1}+\widetilde{\mathbf{D}}_y\frac{\partial\mathbf{F}}{\partial y}\mathbf{F}^{-1}+\widetilde{\mathbf{D}}_z\frac{\partial\mathbf{F}}{\partial z}\mathbf{F}^{-1}\right) \tag{6.51}$$

then

$$\sum_{j=1}^{n} s_{ij}\left[\frac{1}{\widetilde{k}_{rw_j}}\left(\frac{\partial\widetilde{k}_{rw_j}}{\partial x}\frac{\partial p_{w_j}}{\partial x}+\frac{\partial\widetilde{k}_{rw_j}}{\partial y}\frac{\partial p_{w_j}}{\partial y}+\frac{\partial\widetilde{k}_{rw_j}}{\partial z}\frac{\partial p_{w_j}}{\partial z}\right)\right]=\widetilde{\mathbf{T}}\mathbf{w}\left[\left(1-\theta_{p_w}\right)\boldsymbol{p}_w{}^{\mathrm{m}}+\theta_{p_w}\boldsymbol{p}_w{}^{\mathrm{m+1}}\right]$$

(6.52)

Let us define the vector $\widetilde{\mathbf{v}}$, which components are

$$\widetilde{v}_j = \frac{1}{\widetilde{k}_{rw_j}} \frac{\partial \widetilde{k}_{rw_j}}{\partial z} \tag{6.53}$$

then

$$\sum_{j=1}^{n} s_{ij} \frac{\rho_w g}{\widetilde{k}_{rw_j}} \frac{\partial \widetilde{k}_{rw_j}}{\partial z} = \rho_w g \mathbf{S} \widetilde{\mathbf{v}} \tag{6.54}$$

By substituting (6.45), (6.46), (6.52) and (6.54) into (6.40)

$$\begin{aligned} &\mathbf{H}\left[\left(1-\theta_{p_w}\right)\boldsymbol{p}_w{}^{\mathrm{m}} + \theta_{p_w}\boldsymbol{p}_w{}^{\mathrm{m+1}}\right] - \mathbf{G}\left[\left(1-\theta_{q_w}\right)\boldsymbol{q}_w{}^{\mathrm{m}} + \theta_{q_w}\boldsymbol{q}_w{}^{\mathrm{m+1}}\right] = \\ &-\widetilde{\mathbf{R}}_w\left(\frac{1}{\Delta t}\left(\boldsymbol{p}_w{}^{\mathrm{m+1}} - \boldsymbol{p}_w{}^{\mathrm{m}}\right)\right) - \widetilde{\mathbf{T}}\mathbf{w}\left[\left(1-\theta_{p_w}\right)\boldsymbol{p}_w{}^{\mathrm{m}} + \theta_{p_w}\boldsymbol{p}_w{}^{\mathrm{m+1}}\right] + \rho_w\, g\, \mathbf{S}\, \widetilde{\mathbf{v}} \end{aligned} \tag{6.55}$$

By rearranging (6.55), the following system of equations is obtained for the water phase in the m+1$^{\mathrm{th}}$ time iteration

$$\begin{aligned} \left(\theta_{p_w}\mathbf{H} + \frac{\widetilde{\mathbf{R}}_w}{\Delta t} + \theta_{p_w}\widetilde{\mathbf{T}}\mathbf{w}\right)\boldsymbol{p}_w{}^{\mathrm{m+1}} - \theta_{q_w}\mathbf{G}\,\boldsymbol{q}_w{}^{\mathrm{m+1}} = &\left[\left(\theta_{p_w}-1\right)\left(\mathbf{H}+\widetilde{\mathbf{T}}\mathbf{w}\right) + \frac{\widetilde{\mathbf{R}}_w}{\Delta t}\right]\boldsymbol{p}_w{}^{\mathrm{m}} \\ &+\left(1-\theta_{q_w}\right)\mathbf{G}\,\boldsymbol{q}_w{}^{\mathrm{m}} + \rho_w g\,\mathbf{S}\,\widetilde{\mathbf{v}} \end{aligned} \tag{6.56}$$

Next, the matching conditions require some attention. The matching condition for pressure is trivial: p_w must be the same at the interface between subdomains, while the matching condition for flux requires some further analysis.

Lets us consider a node on the interface between two subdomains Ω_1 and Ω_2. The flux of water trough the interface, per unit surface and unit time, at that node will be

$$Q_1 = \vec{q}_{w_1} \cdot \vec{\mathrm{n}}_1 \tag{6.57}$$

and

$$Q_2 = \vec{q}_{w_2} \cdot \vec{\mathrm{n}}_2 \tag{6.58}$$

where $\vec{n}_1$ and $\vec{n}_2$ are the normal-to-the-interface versors calculated from Ω_1 and Ω_2 respectively. As $\vec{q}_{w_1} = \vec{q}_{w_2}$ and $\vec{n}_1 = -\vec{n}_2$

$$Q_1 = -Q_2 \tag{6.59}$$

Substituting (6.2) into (6.57) produces

$$Q_1 = -\frac{k_{w_1}}{\mu_w}\left(\vec{\nabla} p_{w_1} - \rho_w g \vec{\nabla} z\right) \cdot \vec{n}_1 \tag{6.60}$$

Further (6.60) can be recast as

$$Q_1 = -\frac{k_{w_1}}{\mu_w}\frac{\partial p_{w_1}}{\partial n} + \frac{k_{w_1}}{\mu_w}\rho_w g \vec{\nabla} z \cdot \vec{n}_1 \tag{6.61}$$

or

$$Q_1 = -\frac{k_{w_1}}{\mu_w} q_{w_1} + \frac{k_{w_1}}{\mu_w}\rho_w g \, n_{z_1} \tag{6.62}$$

where n_{z_1} is the z-component of $\vec{n}_1$.

Analogously,

$$Q_2 = -\frac{k_{w_2}}{\mu_w} q_{w_2} + \frac{k_{w_2}}{\mu_w}\rho_w g \, n_{z_2} \tag{6.63}$$

As $Q_1 = -Q_2$ and $n_{z_1} = -n_{z_2}$

$$-\frac{k_{w_1}}{\mu_w} q_{w_1} + \frac{k_{w_1}}{\mu_w}\rho_w g \, n_{z_1} = \frac{k_{w_2}}{\mu_w} q_{w_2} + \frac{k_{w_2}}{\mu_w}\rho_w g \, n_{z_1} \tag{6.64}$$

Reordering yields

$$q_{w_2} = \frac{-\dfrac{k_{w_1}}{\mu_w}}{\dfrac{k_{w_2}}{\mu_w}} q_{w_1} - \frac{\left(-\dfrac{k_{w_1}}{\mu_w} + \dfrac{k_{w_2}}{\mu_w}\right)}{\dfrac{k_{w_2}}{\mu_w}} \rho_w g \, \mathrm{n}_{z_1} \tag{6.65}$$

Finally, the matching condition for flux is obtained

$$q_{w_2} = -\frac{k_{rw_1} K_{w_1}}{k_{rw_2} K_{w_2}} q_{w_1} - \frac{k_{rw_1} K_{w_1} - k_{rw_2} K_{w_2}}{k_{rw_2} K_{w_2}} \rho_w g \, \mathrm{n}_{z_2} \tag{6.66}$$

or

$$q_{w_2} = A q_{w_1} - B \tag{6.67}$$

6.6 Numerical examples

All the examples presented here used discontinuous elements combined with ATPS as approximation function. No internal DRM nodes or length scaling have been used. A Conjugate Gradient-Normal Residual (CGNR) iterative solver was employed. Note that in the following examples, when convergence is discussed it refers to the convergence of the iterative scheme used to deal with the non-linearity and does not refer to the convergence of the solver, which always converged.

The examples were previously solved by Vogel et al. [63] using the HYDRUS 5.0 one-dimensional Galerkin FEM program documented by Vogel et al. [64], which uses a time stepping algorithm. In their calculations they used nodal spacing of 0.5 and 1.0 cm depending upon the non-linearity of the problem.

6.6.1 CASE 1: Upward infiltration in clay using the modified van Genuchten soil-water retention curve

The first case simulates infiltration in a 1 m long clay column that initially is assumed to be in equilibrium with an imposed water pressure, p_w, of zero Pa at the bottom of the column (z = 1). The boundary conditions were 98060 Pa of water

pressure (atmospheric pressure) at the bottom of the column (z = 1) combined with zero flux at the top (z = 0), leading to upward infiltration against gravity.
The soil-water retention curve is the modified van Genuchten model (6.29) and (6.30). The parameters of the codes were set up identically to the Vogel's example:

Porosity, n	0.38		
Hydraulic conductivity of water, K_w	5.56E-07 m/s	or	4.8 cm/day
Irreducible water saturation, S_{w0}	0.17895		
Van Genuchten's a parameter	0.8 1/m	or	0.008 1/cm
Van Genuchten's n parameter	1.09		

Conversely to Vogel's code, which used an adaptive time stepping algorithm, this code used a fixed timestep of 0.1 day. The 3D domain was a prismatic column of 0.2 × 0.2 × 1 m^3. The mesh had 410 subdomains being finer in the bottom. Figure 6.1 presents a view of the mesh. Preliminary tests failed when a uniform mesh of 173 subdomains (the most frequently used in the examples of advection-diffusion in chapter 4) was tried. The characteristic length of the subdomains in that case was 10 cm, that is, one order of magnitude bigger than the Vogel´s 1D elements. With refinement in the bottom the characteristic length was reduced to 6.6 cm. In order to produce results equivalent to the 1D case, a zero flux boundary condition was imposed on the sides of the domain. In order to start the iterations of the non-linear loop in the first timestep, the code requires an initial guess for $\widetilde{p}_w$ different from zero within the domain. For the results presented in figure 6.4, the initial guess was that $\widetilde{p}_w$ was equal to the initial conditions when $z \le 0.9$ and from there the water pressure increased linearly up to the atmospheric pressure at $z = 1$, where it matched the boundary condition. Initial guess and initial condition are displayed in figure 6.2.

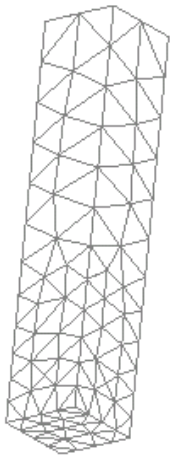


Figure 6.1: mesh of 410 subdomains.

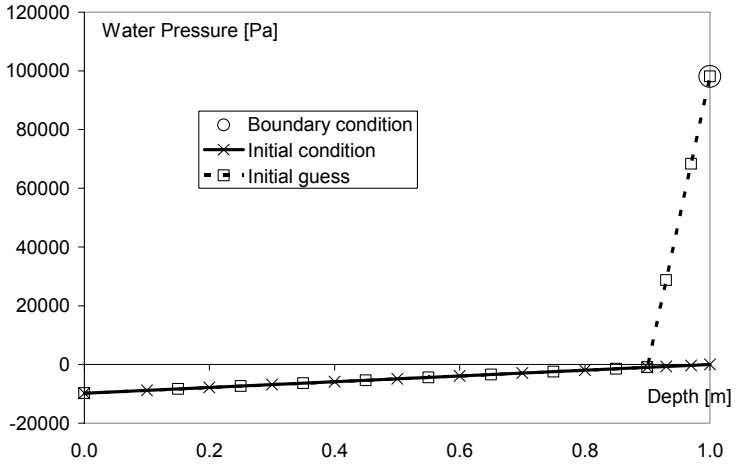


Figure 6.2: Initial guess and initial condition for case 1.

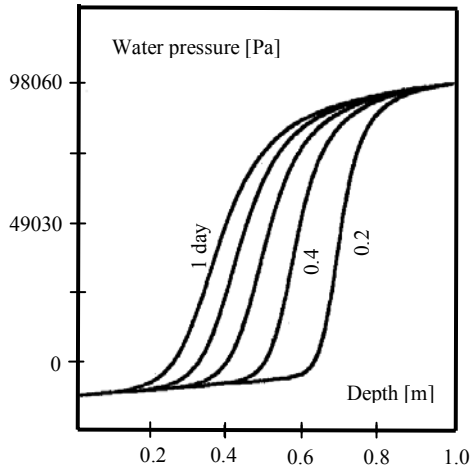


Figure 6.3: Vogel's [63] results of case 1.

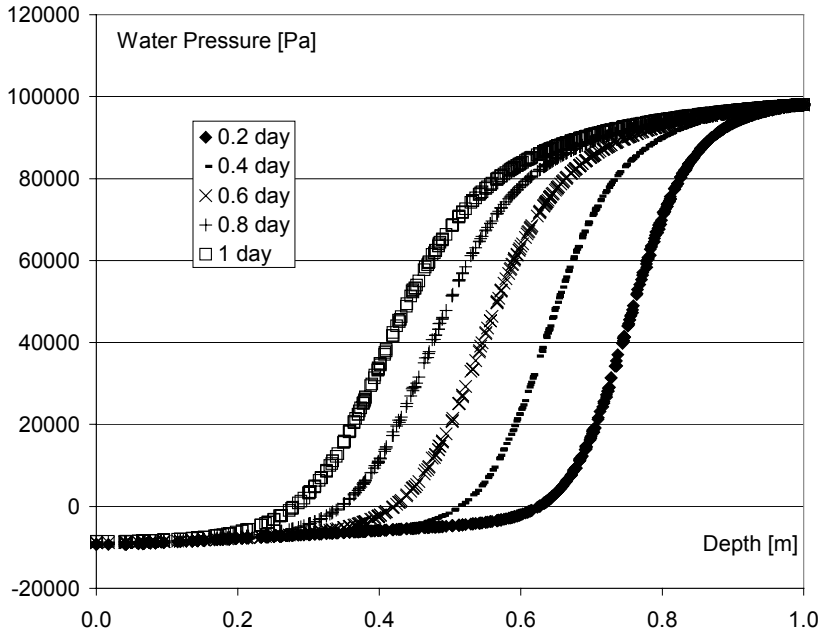


Figure 6.4: results of case 1.

The agreement of the results displayed in figure 6.4 and Vogel's, which are in figure 6.3, is close. Table 1 presents the number of iterations used to deal with the non-linear term in every timestep until convergence is achieved. Note how the convergence is easier as the pressure distribution becomes smoother.

Table 6.1: number of iterations due to the non-linear term in every timestep for case 1.

Timestep	1	2	3	4	5	6	7	8	9	10
No. of iterations	19	18	15	12	11	9	8	8	6	6

6.6.2 CASE 2: Downward infiltration in clay using the modified van Genuchten soil-water retention curve

This example is the simulation of infiltration in a 1 m long clay column that, again, initially was assumed to be in equilibrium with an imposed water pressure, p_w, of zero Pa at the bottom of the column. The boundary conditions were 98060 Pa of water pressure (atmospheric pressure) at the top of the column combined with zero flux at the bottom, leading to downward infiltration. Three meshes were used. The

first one was identical to the one displayed in figure 6.1, though the domain was inverted in order to have the finer part of the mesh in the top. Views of the other two meshes, which had 1062 and a 2233 subdomains, are in figure 6.5. The initial guess was a linear function of the z-coordinate that started with $\tilde{p}_w$ = atmospheric pressure at $z = 0$ (top of the domain) and decreases up to meet the initial condition curve at $z = 0.1$, from there the initial guess is identical to the initial condition. Figure 6.6 shows the initial guess and the initial condition. The remaining part of the set up was identical to case 1.

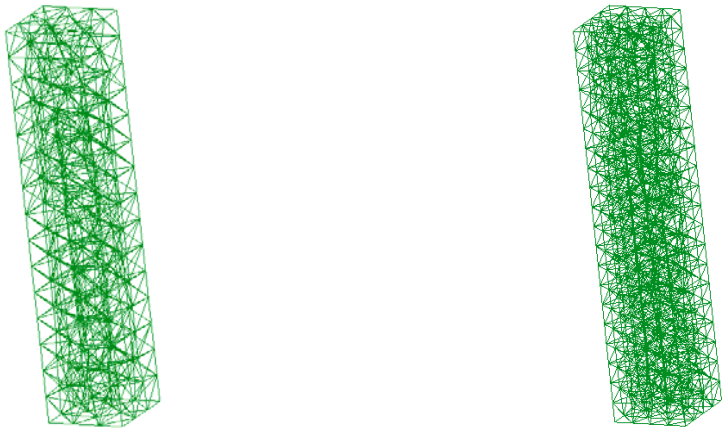


Figure 6.5: Case 2, view of the 1062 and 2233 subdomains meshes.

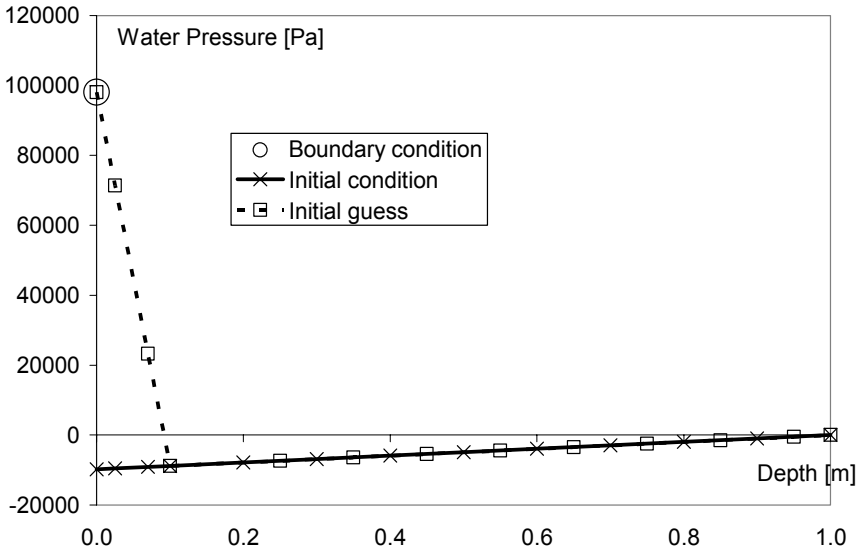


Figure 6.6: Initial guess and initial condition for case 2.

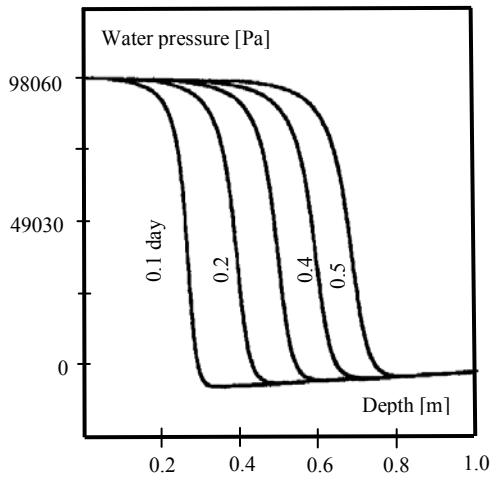


Figure 6.7: Vogel's [63] results of case 2.

Figures 6.8 to 6.10 show results using the three meshes. In every figure, only results of the timesteps that reached convergence are displayed. In Table 6.2 the number of iterations in every timestep is shown.

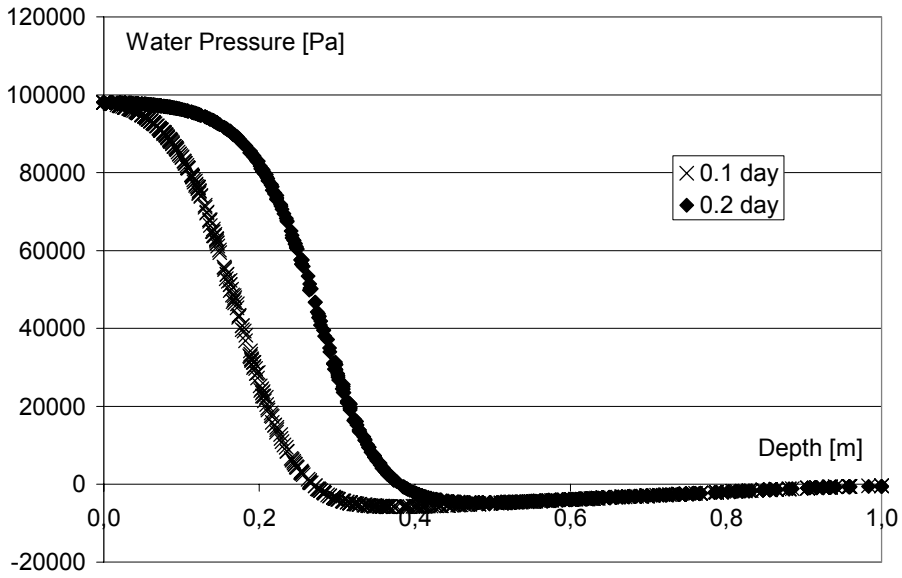


Figure 6.8: results of case 2 using the 410 subdomains mesh.

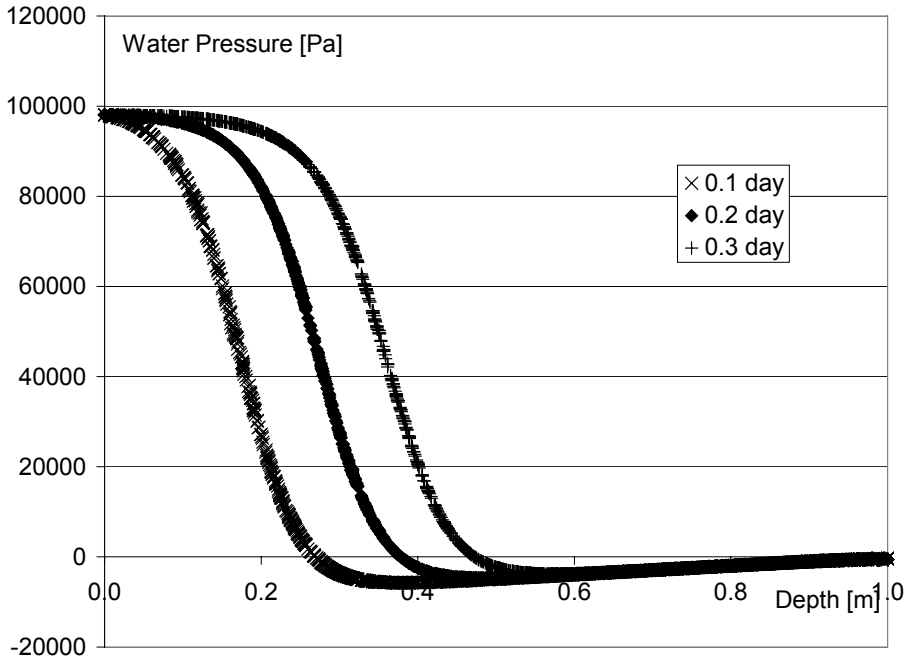


Figure 6.9: results of case 2 using the 1062 subdomains mesh.

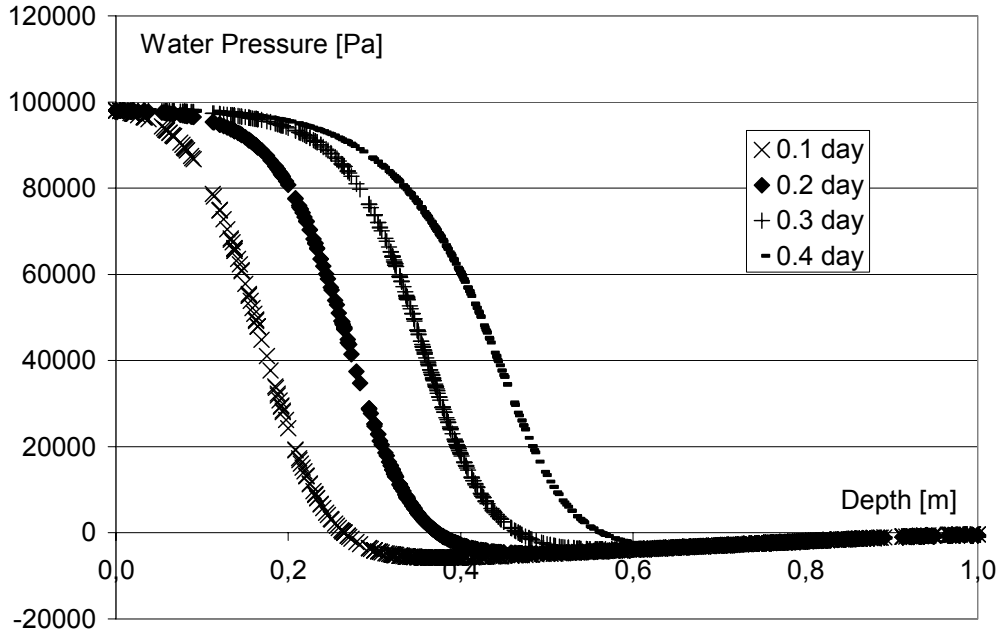


Figure 6.10: results of case 2 using the 2233 subdomains mesh.

Solving the downward infiltration case is more difficult than the upward infiltration case because the water goes down in a sharp front that resembles a step function. The

severe non-linearity of the soil-water curve near saturation makes difficult computing the term $\frac{1}{k_{rw}}\vec{\nabla}k_{rw}$ of eqn. (6.22), because in some points near saturation there are high values of $\vec{\nabla}k_{rw}$ combined with very small values of k_{rw}. In regard with case 1, when using the same mesh of 410 subdomains, the code needs more iterations to converge in the first 2 timesteps than in case 1 (see Table 6.2) and it is unable to converge in the third iteration. The situation improves with mesh refinement, as it can be seen in Table 6.2. Note that the characteristic length of subdomains is 6.6 cm and 5 cm for meshes of 1062 and 2233 subdomains, respectively. Compared with the 0.5-1 cm long 1D element of Vogel it is a noticeably coarser mesh. Mesh refinement improves only the convergence; the quality of the results is similar with different meshes, provided convergence has been reached. In regard with Vogel's results (Figure 6.7), although they are similar, the agreement is not as close as in case 1.

Table 6.2: number of iterations due to the non-linear term in every timestep for case 2.

Timestep		1	2	3	4	5
No. of iterations	410 subs	28	56	150 (truncated)	150 (truncated)	150 (truncated)
	1062 subs	29	46	117	150 (truncated)	It diverges
	2233 subs	24	37	75	300 (truncated)	300 (truncated)

6.6.3 CASE 3: Upward infiltration in clay using the Bruce and Corey soil-water retention curve

This case is identical to case 1 except that the soil water retention function is the one of Bruce and Corey (eqns. (6.35) and (6.35)).

In order to obtain the parameters of the Bruce and Corey function we follow van Genuchten [59]. For large values of the water pressure the van Genuchten expresion of the effective saturation tend to be

$$S_e = \frac{1}{\left(1+(\mathrm{a}p_c)^{\mathrm{n}}\right)^{\mathrm{m}}} \qquad \rightarrow \qquad \frac{1}{(\mathrm{a}p_c)^{\mathrm{nm}}} \tag{6.68}$$

The Bruce and Corey expression for S_e is

$$S_e = \left(\frac{p_b}{p_c}\right)^{\lambda} \tag{6.69}$$

For the Mualem theory one has $\mathrm{m} = 1 - 1/\mathrm{n}$, hence $\lambda = \mathrm{n} - 1$. The parameter 'a' is inversely related to the bubbling pressure p_b. Consequently, for this case λ was equal to 0.09 and p_b=1.25 m.

Results are displayed in figure 6.11 and Table 6.3.

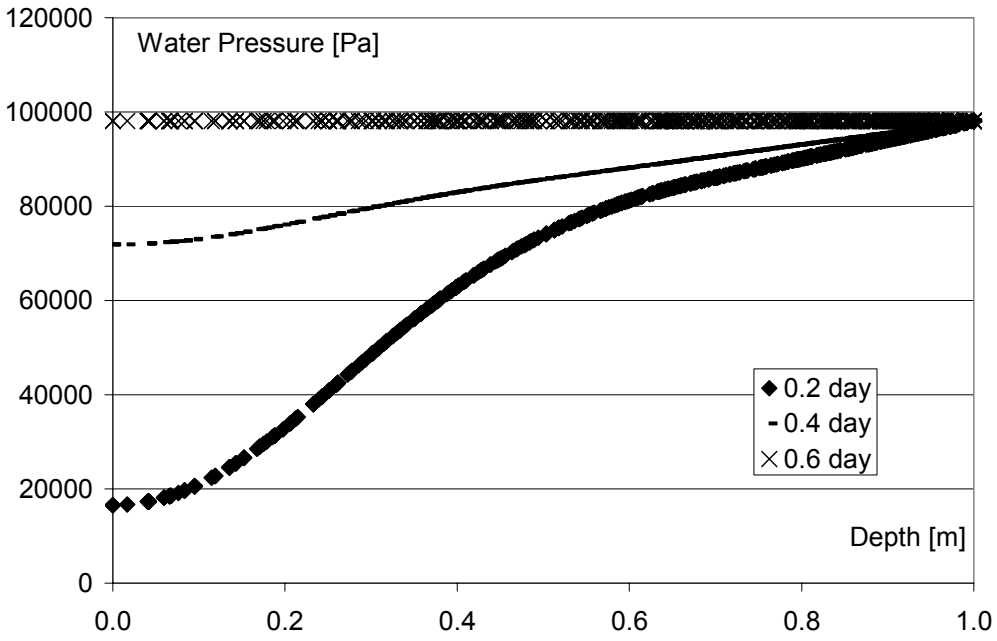


Figure 6.11: results of case 3.

Table 6.3: number of iterations due to the non-linear term in every timestep for case 3.

Timestep	1	2	3	4	5	6
No. of iterations	17	13	9	6	6	6

Although Bruce & Corey soil-water function performs satisfactorily in other numerical models, it does not perform so in this one. The code converges at a faster rate than the case when modified van Genuchten's function is used but the results are far from the ones of Vogel, which are in figure 6.3. Moreover, further test showed that, when using Bruce & Corey function, results are highly sensitive to the value of the timesteps.

6.6.4 CASE 4: Downward infiltration in clay using the Bruce and Corey soil-water retention curve

This case is identical to case 2 but with the Bruce and Corey soil-water retention curve. The results are shown in figure 6.12 and Table 6.4.

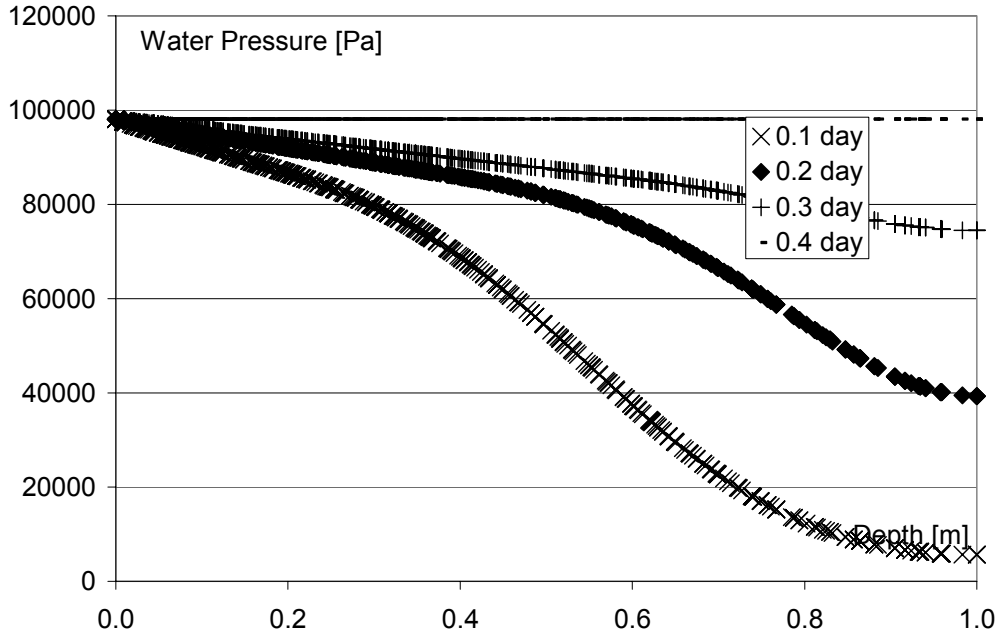


Figure 6.12: results for case 4.

Table 6.4: number of iterations due to the non-linear term in every timestep for case 4.

Timestep	1	2	3	4
No. of iterations	29	30	15	3

Cases 4 confirms the trends that have already been observed in case 3, that is, that the code is more efficient when using Bruce and Corey soil-water function though the results are far from the ones of figure 6.7.

6.6.5 CASE 5: Upward infiltration in clay using the original van Genuchten soil-water retention curve

This case is identical to case 1 but using the Original van Genuchten model (eqns. (6.26) and (6.27)).

The code diverges from the very beginning. Figure 6.14 shows the water pressure distribution after the first iteration of the first timestep. If the results in figure 6.14 are compared with figure 6.15, which shows the water pressure distribution after the first iteration of the first timestep for case 1, it becomes clear why the code diverges.

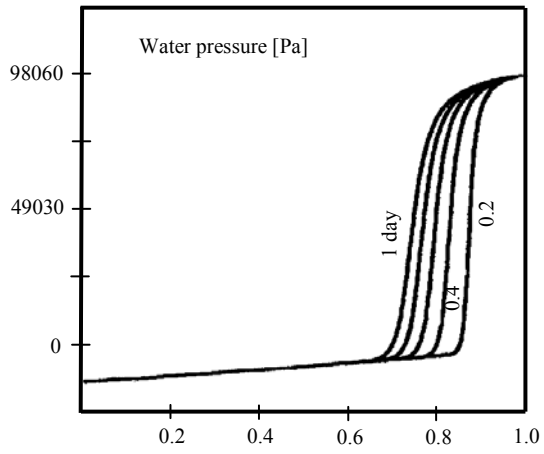


Figure 6.13: Vogel's [63] results of case 5.

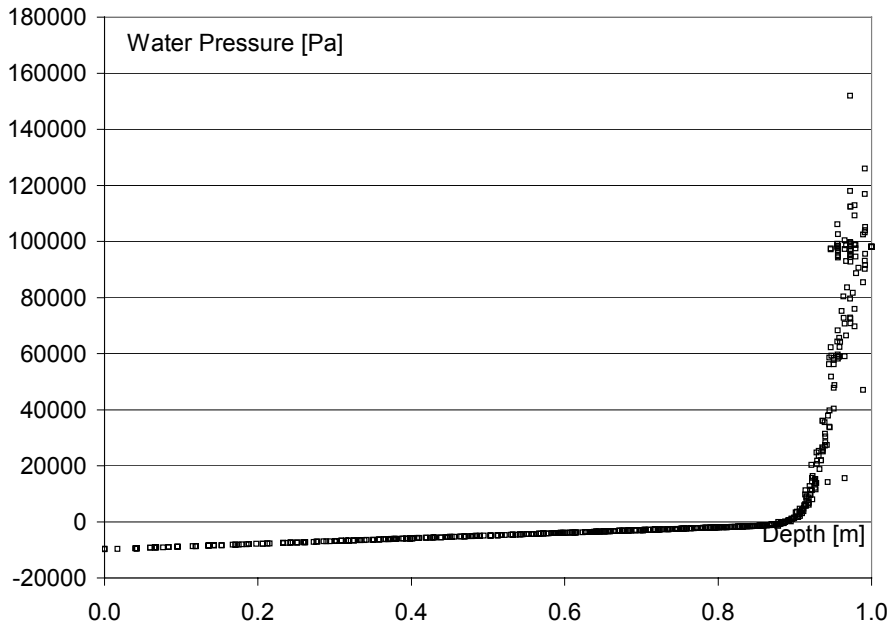


Figure 6.14: water pressure distribution after the first iteration of the first timestep for case 5.

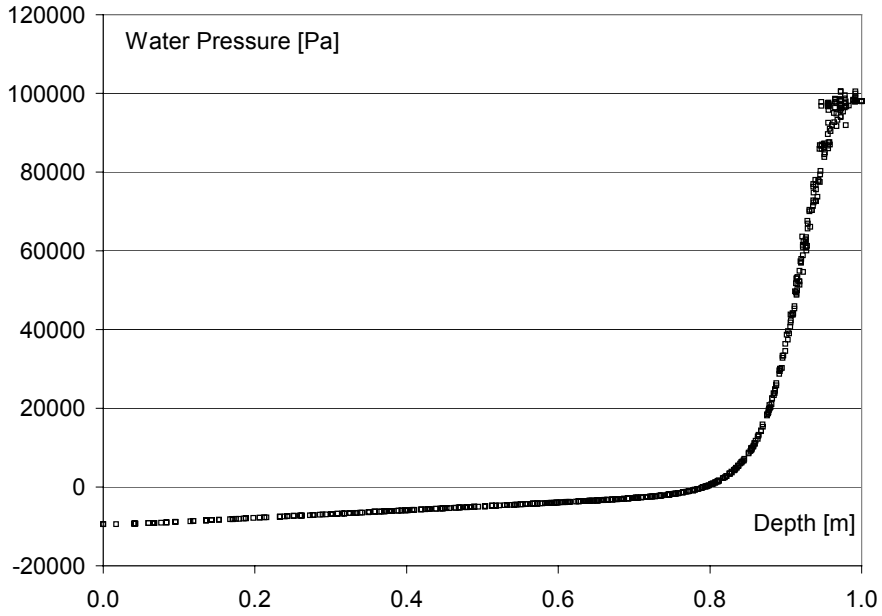


Figure 6.15: water pressure distribution after the first iteration of the first timestep for case 1.

Again, the failure in this case seems to be originated in the terms containing $\frac{1}{k_{rw}}\vec{\nabla}k_{rw}$ in eqn. (6.22). Vogel's results, which are in figure 6.13, show that the slope

of the pressure distribution is higher when using the Original van Genuchten model than when using the Modified van Genuchten model (case 1); which implies again that in some points near saturation it will be high values of $\vec{\nabla} k_{rw}$ combined with very small values of k_{rw}. Vogel et al. reported that the Original van Genuchten model can cause serious problems of convergence, which was confirmed in this study.

Next, results of two other cases which failed to converge will be presented. The problem causing the failure being the same as in the previous cases.

6.6.6 CASE 6: Upward infiltration in sandy clay loam using the modified van Genuchten soil-water retention curve

This case is identical to case 1 but for sandy clay loam. The following parameters were changed to:

Porosity	0.39		
Hydraulic conductivity of water, K_w	3.6389E-06 m/s	or	31.44 cm/day
Irreducible water saturation, S_{w0}	0.17895		
Van Genuchten's a parameter	5.9 1/m	or	0.059 1/cm
Van Genuchten's n parameter	1.48		
Timestep	3600 s	or	1 h

Likewise the previous case, the code diverges. Figure 6.16 presents the water pressure distribution after the first iteration of the first timestep.

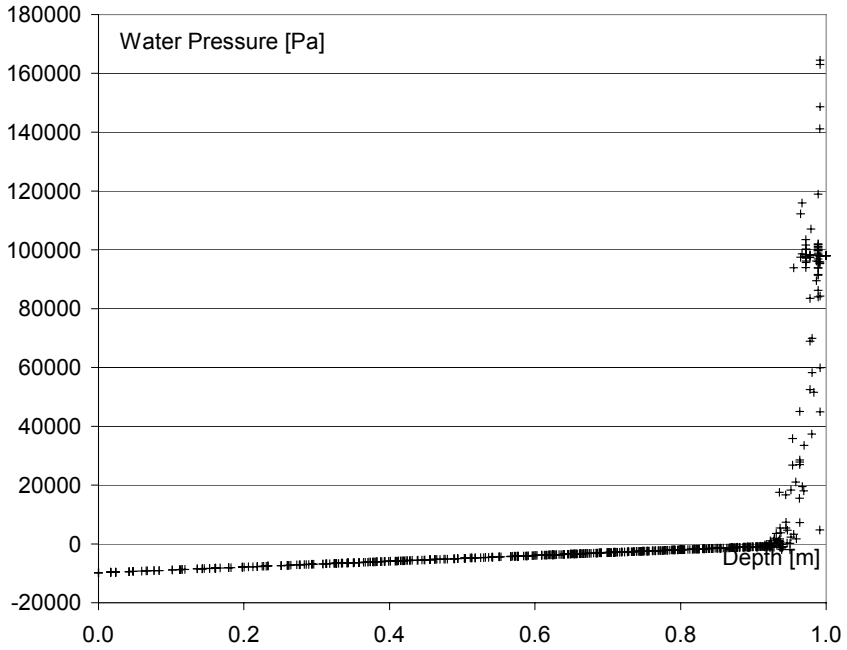


Figure 6.16: water pressure distribution after the first iteration of the first timestep for case 6.

Note that the soil in this case is characterized by van Genuchten's parameter a = 5.9 1/m, while for clay a is equal to 0.8. This difference makes the slope of the soil water curve much steeper in the sandy clay loam case. The effect of varying the 'a' parameter can be observed in ref. [65].

6.6.7 CASE 7: Upward infiltration in sand using the modified van Genuchten soil-water retention curve

This case is identical to case 1 but for sand. The following parameters were changed to:

Porosity	0.43		
Hydraulic conductivity of water, K_w	8.25E-05 m/s	or	712.8 cm/day
Irreducible water saturation, S_{w0}	0.00456		
Van Genuchten's 'a' parameter	14.5 1/m	or	0.145 1/cm
Van Genuchten's 'n' parameter	2.68		
Timestep	3600 s	or	1 h

Likewise the two previous cases, the code diverges. Figure 6.17 shows the water pressure distribution after the first iteration of the first timestep. Again, note that van Genuchten's parameter '*a*' is much bigger than the one in case1.

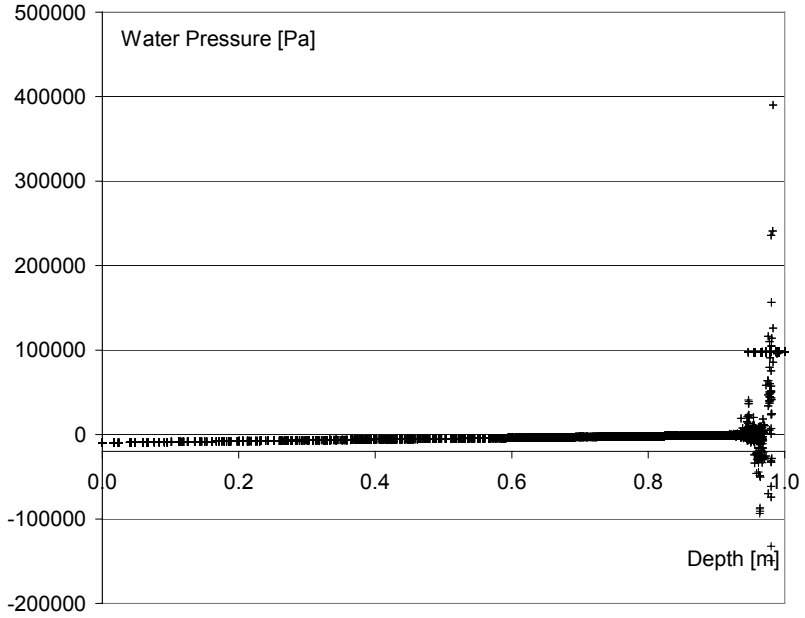


Figure 6.17: water pressure distribution after the first iteration of the first timestep for case 7.

6.7 Conlusions of the chapter

Flow in unsaturated media has been approached in this chapter with a DRM-MD model. The final aim of the model is to estimate the saturation of clay in underground repositories, which requires solving two coupled, non-linear partial differential equations; one for the air and one for the water phase.

Taking into account that the complexity of the model makes setting the code in many different ways possible, it was judged convenient to start studying the basic behaviour of the formulation under a simpler scheme before approaching the two phases model and real cases. Consequently, the starting point was testing examples in which the air phase was assumed to be at atmospheric pressure; a situation that is equivalent to solving the Richard's equation. Presently, only results of few cases of

this simpler scheme are available since the whole project is in an early stage; those are the results that have been presented in this chapter.

Not many attempts were reported of modelling groundwater flow in unsaturated media by using the BEM. The first reported application of the BEM without either linearizing the flow equation or solving the problem with special cases of soil constitutives relations was accomplished by Taigbenu & Onyejekwe [15], who used a multidomain approach. The problems that have been used to test the code have been previously solved by Vogel et al. [63] using a 1D FEM code and a quite similar one appears in Taigbenu's work, which was solved with a 1D GEM code.

The code reported in this thesis showed that it is able to accurately solve problems of infiltration in clay, though it had problem of convergence with other soils. For instance, cases 1 and 2 showed an excellent performance in regard with the one-dimensional results of Vogel's, even though the 3D mesh used was between 5 to 13 times coarser than the Vogel's 1D one, and that the used time marching scheme is the simplest one. But even for clay, convergence was not reached after the fourth time-step for downward infiltration. Case 2 results show that the problem is alleviated using mesh refinement. All the results point consistently out to the terms of the governing equations containing $\frac{1}{k_{rw}}\vec{\nabla}k_{rw}$ as the source of the lack of convergence when it occurs. The severe non-linearity of the soil-water curve near saturation makes difficult the task of computing the term $\frac{1}{k_{rw}}\vec{\nabla}k_{rw}$ of eqn. (6.22), because in some points near saturation there are high values of $\vec{\nabla}k_{rw}$ combined with very small values of k_{rw}. The code loses accuracy when it has to calculate $\vec{\nabla}k_{rw}$ and the water advances in a sharp front that resembles a step function, as it happens in cases 2, 5 and 6; or when the soil-water function resembles a step function, as it happens in cases 7 and 8. Convergence should be reached by using mesh refinement, which couldn't be accomplished due to lack of suitable hardware.

A key factor in the model is which soil-water retention curve is used. In this case three different soil-water models were implemented. Among them, only the modified

van Genuchten's model performed satisfactorily. However, Sillers et al. [65] reported eight different models that have not been tested in this work. All of them should be tested since it may happen that some of them perform better than the Modified van Genuchten's model.

It remains for the future work to test how the code will perform with different soil-water retention curves, finer meshes, internal DRM nodes, different scales factors, and specially with different times marching schemes.

Chapter 7

Conclusions

7.1 Conclusions

The DRM-MD had been applied in the past to a variety of problems that are governed by linear and non-linear PDEs. However, they were mainly limited to 2D problems using structured meshes, which restricts the applicability of the approach.
Here the range of applications was increased in regard with what was available in the literature and several aspects regarding the implementation of computer codes using DRM-MD algorithms using the fundamental solution of the Laplace's equation, with emphasize on the 3D implementation, have been studied.
Let us recall that the following kinds of problems were solved here in 3D domains:

- Poisson´s problems with constant source term
- Steady state advection diffusion problems
- Transient advection-diffusion problems with steady flow velocity field
- Flow in unsaturated media (Richard´s equation)

Among them, the transient advection-diffusion case was applied to the simulation of the transport of a pesticide in an underground repository under different scenarios, which were noticeably larger than the typical academic examples with known solution. It has been seen that all the examples that were tested were solved satisfactorily when considering both CPU time, on Pentium III and IV PCs, and accuracy. As for the last case (flow in unsaturated media), the example that was considered has been reported for a 1D case only when BEM solution was applied without either linearizing the flow equation or solving the problem with special cases of soil constitutives relations.

As any other numerical procedure the DRM-MD admits different configurations that can largely affect the performance of the codes. In order to approach 3D problems, in this work the domain of the problems was discretized using tetrahedrons with quadratic shape functions to represent the geometry of every subdomain, which are very flexible when used to discretize complicated geometries. Each side of the tetrahedrons is a quadratic, triangular boundary element.

In Chapter 4 the choice of the DRM approximation function was reported. The tests were done on Poisson´s and steady state advection-diffusion problems. Ten different radial basis functions, five globally and five compactly supported RBFs, were tested as DRM approximation functions. This was the first time that CS-RBFs has been used in DRM-MD codes. Chapter 4 provides guidelines to set the approximation function as well as continuity of the elements, scaling and number of internal nodes. Cases using continuous elements gave interesting information about the method, especially because they differ significantly from each other, i.e., the CEC produces an overdetermined system of equations that is solved in a least square sense, while the DEC produces a regular system of equations. The DEC has potential and normal derivatives as unknowns while the CEC has potential, normal derivatives and partial derivatives as unknowns. When using similar meshes the system of equations produced by the CEC has less than a half the number of degree of freedom than the one of the DEC. The DEC showed to be more accurate but slower than the CEC. Presently the DEC has the decisive advantage of been able to deal with non-homogeneous domains. The best strategy would be to take full advantage of the family of elements proposed by Do Rêgo Silva [36] and have a choice to use both types of elements in a single mesh, which would allow to optimize the use of CPU and memory resources.

Accordingly, provided quadratic tetrahedral subdomains are used, a robust recipe to apply the DRM-MD to 3D problems in a first approach is: use discontinuous elements, ATPS as approximation function, no DRM nodes, sparse format storage, an iterative solver using a Conjugate Gradient-Normal Residual (CGNR) algorithm and scaling, the scaling criterion depending on the nature of the problem.

Producing and handling a domain mesh is one of the bêtes noirs of the DRM-MD. This may have been true in the past, but nowadays, with the variety of pre and postprocessors that is available on the market, it cannot be considered to be a serious objection. In this work, GID™, a pre and postprocessor, was used with minor customizations.

On the other hand, it has been argued that the assembly of the system of equations is a tiresome business. In chapter 2 a general scheme for the implementation of DRM-MD codes was proposed. That scheme is general for 2D and 3D codes no matter what kind of elements are used. The main advantage of the proposed scheme is that the assembly is split up into two parts. The first part is common to all codes regardless of the equations that are being solved. It will not be denied here that the assembly is a tiresome business, but the proposed scheme requires that this is performed only once. If one code has been written to solve a given problem, adapting the code for different kinds of problems is straightforward. This has been explained in chapter 4, where details on the application of this scheme to Poisson´s and steady state advection-diffusion equations were given.

Throughout the development of this thesis a few improvements arose regarding the implementation of the DRM-MD. Among them, a new way to represent partial derivatives has been presented in chapter 3. The new formulation removes the singularities introduced by the partial derivatives of the approximation function. When using it combined with $1 + R$ as approximation function, reductions of one and two orders of magnitude in the error of advection-diffusion problems were observed. The new formulation showed to be the best option for representing second order partial derivatives in all the cases that were tested. Results in chapter 3 correspond to 2D linear and non-linear problems.

Other findings that can be mentioned is that iterative solvers are the best option for large problems. Here iterative solvers were not used for first time. The performance of iterative solvers in 2D DRM-MD codes has been exhaustively tried by Portapila and Power [38, 39]. Their work leads to the use of the solvers mentioned in this thesis. Iterative solvers have been also used in large 3D DRM-MD problems by

Peratta and Popov [22, 54, 55]. What is being pointed out is that the experience accumulated after using iterative solvers confirms the previous expectations.
A number of papers have been produced as a result of this thesis. They are referred as ref. [67, 68, 69, 70, 71, 72].

7.2 Further research

Based on the study presented in this thesis, some suggestions for further research can be given:

- The feasibility study on flow in unsaturated media presented in Chapter 6 showed that the DRM-MD was able to solve accurately problems of infiltration in clay governed by the Richard's equation which is a particularly difficult problem to solve. Further research on the optimal way of solving these types of problems, including tests of time marching schemes, is necessary.

- Once a better understanding of the previously mentioned problem is reached, the whole DRM-MD scheme proposed in Chapter 6 for flow in unsaturated media should be extended to include the non-wetting phase equation.

- The DRM-MD has been successfully applied to the solution of 2D problems governed by the Navier-Stokes equation [23] at low Reynolds numbers. The procedure could be extended to 3D problem.

- The 3D codes presented in this thesis used iterative solvers, which performed satisfactorily. The selection of the solvers was based on an extensive research performed in 2D problems. It seems advisable to test exhaustively iterative solvers for 3D problems in order to improve the performance of the DRM-MD.

References

[1] Cruse, T.A., Boundary integral equation-a personal view. *Engineering Analysis with Boundary Elements*, **25**, pp. 709-712, 2001.
[2] Partridge, P.W., Brebbia, C.A. & Wrobel, L.C., *The Dual Reciprocity Boundary Element Method*. Computational Mechanics Publications: Southampton and Boston, 1992.
[3] Nardini, D. & Brebbia, C.A., A new approach to free vibration analysis using boundary elements. *Applied Mathematical Modelling*, **7**, pp. 157-62, 1983.
[4] Golberg, M.A., Chen, C.S. & Schaback, R.A., Recent developments in the dual reciprocity method using compactly supported radial basis functions (Chapter 8). *Transformation of Domain Effects to the Boundary*, eds. Y.F. Rashed & C.A. Brebbia, WIT Press: Southampton and Boston, pp. 183-226, 2003.
[5] Golberg, M.A. & Chen, C.S., The theory of radial basis functions applied to the BEM for inhomogeneous partial differential equations. *Boundary Elements Communications*, **5**, pp. 57-61, 1994.
[6] Golberg, M.A. & Chen, C.S., A bibliography on radial basis functions approximation. *Boundary Elements Communications*, **7**, pp. 155-63, 1996.
[7] Schclar, N.A., *Application of the Boundary Element Method to the Structural Analysis of Three Dimensional Anisotropic Material*, Ph.D. Thesis, University of Portsmouth, 1993.
[8] Rodríguez, J.J. & Power, H., An adaptative dual reciprocity scheme for the numerical solution of the Poisson equation. *Engineering Analysis with Boundary Elements*, **26**, pp. 283-300, 2002.
[9] Schaback, R., Creating surfaces from scattered data using radial basis functions. *Mathematical Methods for Curves and Surfaces*, eds. T. Lyche, M. Dæhlen & L. Schumaker, Vanderbilt University Press: Nashville, pp. 477-496, 1997.
[10] Wendland, H., Piecewise polynomial, positive definite and compactly supported radial function of minimal degree. *Advances in Computational Mathematics*, **4**, pp. 389-396, 1995.
[11] Wu, Z., Multivariate compactly supported positive definite radial functions. *Advances in Computational Mathematics*, **4**, pp. 283-292, 1995.
[12] Buhmann, M.D., A new class of radial basis functions with compact support. *Mathematics of Computations*, **70**, No. 233, pp. 307-318, 2000.
[13] Taigbenu, A.E., *The Green Element Method*. Kluwer Academic Publishers: Boston, 1999.
[14] Taigbenu, A.E., The Green element method. *International Journal for Numerical methods in Engineering*, **38**, pp. 2241-2263, 1995.
[15] Taigbenu, A.E. & Onyejekwe O.O., Green element simulation of the transient nonlinear unsaturated flow equation. *Applied Mathematical Modelling*, **19**, 675-684, 1995.
[16] Žagar, I., Škerget, L. & Alujević A., Diffusion-convection problems using boundary-domain integral formulation for non-uniform flows. *Boundary Element Methods in Fluid Dynamics II*, eds. H. Power, C.A. Brebbia & D.B.

Ingham, Computational Mechanics Publications: Southampton and Boston, pp. 75-82, 1994.
[17] Škerget, L. & Rek, Z., Boundary-domain integral method using a velocity-vorticity formulation. *Engineering Analysis with Boundary Elements*, **15**, pp. 359-370, 1995.
[18] Popov, V. & Power H., DRM-MD approach for the numerical solution of gas flow in porous media with application to landfill. *Engineering Analysis with Boundary Elements*, **23**, pp. 175-188, 1999.
[19] Popov, V., Power, H. & Baldasano, J.M., BEM solution of design of trenches in a multi-layered landfill. *Journal of Environmental Engineering-ASCE*, **124(1)**, pp. 59-66, 1998.
[20] Popov, V. & Power, H., Numerical analysis of the efficiency of landfill venting trenches. *Journal of Environmental Engineering-ASCE*, **126(1)**, pp. 32-38, 2000.
[21] Popov, V. & Power, H., The DRM-MD integral equation method: an efficient approach for the numerical solution of domain dominant problems. *International Journal for Numerical Methods in Engineering*, **44**, pp. 327-353, 1999.
[22] Peratta, A. & Popov, V., Modelling flow and solute transport in fractured porous media. *Proc. of Boundary Elements XXV*, eds. C.A. Brebbia, D. Poljak & V. Roje, WIT Press: Southampton and Boston, pp. 63-72, 2003.
[23] Florez, W.F. & Power, H., DRM multidomain mass conservative interpolation approach for the BEM solution of the two-dimensional Navier-Stokes equations. *Computers & Mathematics with Applications*, **43(3-5)**, pp. 457-472, 2002.
[24] Florez, W.F. & Power, H., Multi-domain mass conservative dual reciprocity method for the solution of the non-Newtonian Stokes equations. *Applied Mathematical Modelling*, **26(3)**, pp. 397-419, 2002.
[25] Florez, W.F., *Nonlinear Flow Using Dual Reciprocity*. WIT Press: Southampton and Boston, 2001.
[26] Power, H. & Wrobel, L.C., *Boundary Integral Methods in Fluid Mechanics*. Computational Mechanics Publications: Southampton and Boston, 1995.
[27] Brebbia, C.A, *The Boundary Element Method for Engineers*. Pentech Press: London, Halstead Press: New York, 1978.
[28] Brebbia, C.A. & Dominguez, A., *Boundary Elements. An Introductory Course*. Computational Mechanics Publications, Southampton: 1992, second edition.
[29] Banerjee, P.K. & Butterfield, R., Boundary elements method in geomechanics (Chapter 16). *Finite Elements Method in Geomechanics*, ed. G. Gudehus, John Wiley and Sons: London, 1976.
[30] Zhu, S. & Zhang, Y., Improvement on dual reciprocity boundary element method for equations with convective terms. *Communications in Numerical Methods in Engineering*, **10**, pp. 361-71, 1994.
[31] Chen, K., Improving the accuracy of DRBEM for convective partial differential equations. *Engineering Analysis with Boundary Elements*, **23**, pp. 634-44, 1999.

[32] Golberg, M.A. & Chen, C.S., *Discrete Projection Methods for Integral Equations*. Computational Mechanics Publications: Southampton and Boston, 1997.
[33] Hardy, R.L., Multicuadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, **76**, pp. 1905-1915, 1971.
[34] Franke, R., Scattered data interpolation: test of some methods. *Mathematics of Computations*, **38**, pp. 181-200, 1982.
[35] Popov, V. & Power, H., *Landfill Emission of Gases into the Atmosphere*. Computational Mechanics Publications: Southampton and Boston, 1999.
[36] Do Rêgo Silva, J.J., *Acoustic and Elastic Wave Scattering Using Boundary Elements*. Computational Mechanics Publications: Southampton and Boston, 1994.
[37] Telles, J.C.F., A self-adaptative co-ordinate transformation for efficient numerical evaluation of general boundary element integrals. *International Journal for Numerical methods in Engineering*, **24**, pp. 959-973, 1987
[38] Portapila, M.I. & Power, H., Iterative schemes for the solution of system of equations arising from the DRM in multi domain approach, and a comparative analysis of the performance of two different radial basis functions used in the interpolation. *Engineering Analysis with Boundary Elements*, **29(2)**, pp. 107-125, 2005.
[39] Portapila, M.I. & Power, H., Efficiency and accuracy of an iterative solution for the DRM- MD. *Boundary Elements Communications*, **12(1)**, pp. 26-37, 2001.
[40] Saad, Y., *Iterative Methods for Sparse Linear Systems*. PWS Publishing: New York, 1996.
[41] Paige, C.C. & Saunders, M.A., Algorithm 583: LSQR: sparse linear equations and least squares problems. *ACM Transactions on Mathematical Software*, **8(2)**, pp. 195-209, 1982.
[42] Powell, M.J.D., The theory of radial basis functions approximation (Chapter 3). *Advances in Numerical Analysis*, ed. W. Light, Oxford University Press: Oxford, 1992.
[43] Micchelli, C.A., Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, **2**, pp. 11-22, 1986.
[44] Floater, M.S. & Iske, Multistep scattered data interpolation using compactly supported radial basis functions. *Journal of Computational and Applied Mathematics*, **73**, pp. 65-78, 1996.
[45] Schmidt, G. & Strese, H., BEM for Poisson equation. *Engineering Analysis with Boundary Elements*, **10**, pp. 119-123, 1992.
[46] Levesley, J., Pointwise estimates for multivariate interpolation using conditionally positive functions, approximation theory. *Wavelets and Applications*, ed. S.P. Singh, Kluwer Academic Publishers: Netherlands, pp. 381-401, 1997.
[47] Golberg, M.A., The numerical evaluation of particular solutions in the BEM-a review. *Boundary Elements Communications*, **6**, pp. 99-106, 1995.
[48] Duchon, J., Splines minimizing rotation invariant seminorms in Sovolev spaces. *Constructive Theory of Functions of Several Variables, Lectures Notes*

in Mathematics 571, eds. W. Schempp & K. Zeller, Springer-Verlag: Berlin, pp. 85-110, 1976.
[49] Bear, J. & Verruijt, A., *Modeling Groundwater Flow and Polution*. Reidel Publishing Company: Dordrecht, 1987.
[50] Wrobel, L.C., Brebbia, C.A. & Nardini, D., The dual reciprocity boundary element formulation for transient heat conduction. *Proc. of the 6th International Conference on Finite Elements in Water Resources*, eds. A. Sá da Costa, A. Melo Baptista, W. G. Gray, C.A. Brebbia & G.F. Pinder, Computational Mechanics Publications, Southampton and Springer-Verlag, Berlin and New York, 1986.
[51] WIT, IMF, ITEMA, Numerical simulation of gas flow in porous media with application to landfill, European Commission Programme, Contract number CI*_CT94-0077, Twelve Monthly Technical Report 2, January 1997.
[52] Popov, V. & Power. H., BEM solution for the problem of flux of a multicomponent mixture of gases out of a multilayered landfill. *International Journal for Numerical Methods in Fluids*, **23**, 503-524, 1996.
[53] Van Genuchten, M.T. & Alves, W.J., Analytical solutions of the one-dimensional convective-dispersive solute transport equation. *Technical Bulletin No. 1661*, US Department of Agriculture, 1982.
[54] Peratta, A. & Popov, V., Numerical modeling of contaminant dispersion from underground mine repositories. *Waste Management and the Environment II*, eds. C.A. Brebbia, S. Kungolos, V. Popov, & H. Itoh, WIT Press: Southampton, 2004.
[55] Low Risk Deposition Technology. European Commission FP5 Programme, contract number EVG1-CT-2000-00020, January 2001-December 2003.
[56] Helmig, R., *Multiphase Flow and Transport Processes in the Subsurface*. Springer-Verlag: Berlin and New York, 1997.
[57] Leverett, M.C., Capillary behavior in porous solids. *Transactions of the AIME*, **142**, pp. 152-169, 1941.
[58] Brooks, R.H. & Corey, A.T., Hydraulic properties of porous media. *Hydrology Papers Vol.3*, Colorado State University: Fort Collins, 1964.
[59] Van Genuchten, M.T., A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Science Society of America Journal*, **44**, pp. 892-898, 1980.
[60] Burdine, N.T., Relative permeability calculations from pore-size distribution data. *Journal of Petroleum Technology*, **5**, pp. 71-78, 1953.
[61] Mualem, Y., A new model for predicting the hydraulic conductivity of unsaturated porous media. *Water Resources Research*, **12**, pp. 513-522, 1976.
[62] Vogel, T. & Cislerova, M., Reliability of unsaturated hydraulic conductivity calculated from the moisture retention curve. *Transport Porous Media*, **3**, pp. 1-15, 1988.
[63] Vogel, T., Van Genuchten, M.T. & Cislerova, M., Effect of the shape of the soil hydraulic functions near saturation on variably-saturated flow predictions. *Advances in Water Resurces*, **24**, pp. 133-144, 2001.
[64] Vogel, T., Huang, K., Zhang, R. & van Genuchten, M.T., The HYDRUS code for simulating one-dimensional water flow, solute transport, and heat

movement in variably saturated media, version 5.0, Research Report 140, Riverside, California: US Salinity Laboratory, 1996, p.131.

[65] Sillers, W.S., Fredlund, D.G. & Zakerzadeh, N., Mathematical attributes of some soil-water characteristic curve models. *Geotechnical and Geological Engineering*, **19**, pp. 243-283, 2001.

[66]

[67] Natalini, B. & Popov, V., Tests of radial basis functions in the 3D DRM-MD, *Communications for Numerical Methods in Engineering*, , in print, available online 11 July 2005.

[68] Natalini, B. & Popov, V., An alternative approach for calculation of the first and higher order derivatives in the DRM-MD. *Engineering Analysis with Boundary Elements*, **28,** pp. 61-78, 2004.

[69] Natalini, B. & Popov, V., Treatment of Convective Terms in DRM-MD. *Boundary Elements Communications*, **12**, pp. 13-20, 2001.

[70] Natalini, B. & Popov, V., Radial basis function selection in 3D DRM-MD for potential problems. *Boundary Elements XXVI*, ed. C. A. Brebbia, WIT Press: Southampton, pp. 101-110, 2004.

[71] Natalini, B. & Popov, V., Improved calculation of higher order partial derivatives in the DRM, *Boundary Elements XXV*, eds. C.A. Brebbia, D. Poljak & V. Roje, WIT Press: Southampton, pp. 41-50, 2003.

[72] Natalini, B. & Popov, V., An Improved DRM Representation of partial derivatives, *Boundary Elements XXIV Incorporating Meshless Solutions*, eds. C.A. Brebbia, A. Tadeu & V. Popov, WIT Press: Southampton, pp. 395-407, 2002.