Lattice Modelling of Moisture Transport in Uncracked and Cracked Concrete

Anmol Singla

TU Delft

Lattice Modelling of Moisture Transport in Uncracked and Cracked Concrete

by

Anmol Singla

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday November 29, 2019 at 01:30 PM.

Student number:	4751205	
Project duration:	January 1, 2019 – November 29, 2019	
Thesis committee:	Prof. dr. ir. L. J. Sluys,	TU Delft, chairman
	Dr. B. Šavija,	TU Delft, mentor
	C. R. Rodriguez,	TU Delft, mentor
	Ir. L. J. M. Houben,	TU Delft, supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Acknowledgements

I would like to express my gratitude to the people who supported me during my thesis.

I would like to thank Bert Sluijs, the chair of my committee, for giving me the opportunity to work on this topic. His astute observations and critical scrutiny of my work steered me in the right direction during the crucial stages of my thesis and inspired a rigorous academic approach.

I would like to thank my supervisor, Branko Šavija for his insightful suggestions and constructive feedback. My thesis was made easier by the quality of his work in this field which served as the foundation on which I based my research. His open door policy, willingness and great consideration to solve my problems were really helpful throughout my thesis.

I would like to give my sincere thanks to my daily supervisor, Claudia Romero Rodriguez, without whose supervision, it would have been really difficult to complete the work that I did. I really exploited her open door policy and often went to her for help. She helped me at every stage of my work, motivated me to think critically about my topic, had brainstorming sessions with me and gave critical feedback of my work, all of which were invaluable for my research. I could not have asked for a better advisor and mentor than her and I wish her good luck for her PhD.

I would like to thank my family for their immense love and continuous encouragement. I would never have had the opportunity to come to Netherlands to pursue higher education without their unfailing support. I would like to thank my mother, Sarita Singla, my uncles, Krishan Goyal and Neeraj Goyal, my brothers, Gautam Singla and Karan Singla, my aunts Kavita Goyal and Bhawna Goyal and all my cousins, who were always there for me and I will be forever grateful to them.

I would like to thank my friends Suman, Enxhi, David, Vanshaj, Palash, Maneesh and many others for their support and encouragement. They were there to pick me up at my lows and also to celebrate my success with me.

Finally, I would like to thank Sanya, for her unconditional support and bearing with my talks about my research while not even being of a civil engineering background. I am grateful for her appreciation of my work and her encouragement to do better.

Thanks to everyone who helped me to complete my MSc at TU Delft.

A. Singla
Delft, November 2019

Abstract

The service life of concrete structures depend largely on the durability of concrete. The durability of concrete is influenced by mass transport mechanisms that can have severe deteriorating effects. Transport of water in concrete is of paramount importance as water can act as a carrier of ions such as chlorides which can corrode the reinforcement and reduce the service life of concrete structures. The main objective of this thesis is to study moisture transport through capillary absorption in concrete. Numerical simulation of moisture transport is performed through lattice elements with an irregular mesh configuration.

New computational tools are developed and compared with the existing tools in terms of effectiveness to simulate moisture transport in homogeneous uncracked concrete. The existing numerical model uses approximate volume of transport elements to determine the volumetric capacity of elements. During discretization of the governing equation, the variation of diffusivity of the elements is approximated as a uniform mean distribution and an explicit time stepping scheme is implemented which has consequences on the flow equilibrium at each time step. In the proposed numerical model, exact volume of transport elements is considered and during the discretization of governing equations, exponential variation of diffusivity is considered along with an implicit time stepping scheme. Moisture transport is accurately simulated in uncracked homogeneous concrete even by using the existing model as validated through experimental results. Both the models are compared through error analysis by varying mesh size and time step. Moisture flow through different diffusivity coefficients is simulated using both the models and it is observed that the existing model suffers from oscillations in the saturation level during initial stages of flow due to inability of the existing model to maintain flow equilibrium at each time step. The proposed model on the other hand shows no such oscillations due to the flow equilibrium being maintained at each time step. Mesh size, magnitude of time step and diffusivity coefficient are shown to be the parameters limiting the effectiveness of either of the models.

A single discrete crack is considered within the domain and its influence on moisture transport is observed. The nature of the crack is considered through two approaches. In the first approach, crack is considered as an additional porous phase in cement matrix and the entire volume occupied by the crack is considered to be a void. In the second approach, in addition to considering the porous nature of crack, moisture surface interaction between the water surface and crack walls is also considered. Horizontal moisture distribution around a planar crack is simulated using both the approaches which show similar results as in the experiment. The vertical moisture distribution is simulated within a wedge shaped crack and the results are compared with experimental observations. The first approach shows a slower rate of saturation of the crack as compared to the experiment. The second approach shows the vertical saturation of the crack similar to the experiment. A comparison between the Delaunay and Voronoi modelling techniques of moisture simulation in cracked domain is discussed. It is shown that the presence of a crack accelerates the moisture transport in concrete as it exposes additional surface of concrete from where water can penetrate in the material and also increases the diffusivity of the concrete material lying in the vicinity of the crack.

Moisture transport is simulated in concrete considering its multi-phase nature. Concrete is considered to be composed of three phases i.e. cement paste, aggregates and ITZ around aggregates. Aggregates are considered to be impervious which do not allow the flow of moisture through them which slows down the flow in concrete. The ITZ around the aggregates have higher transport properties and accelerate the flow. A numerical framework is formulated in which spherical aggregates are projected on lattice elements which are assigned transport properties pertaining to the phase of concrete they represent. Moisture transport is simulated and compared with experimental results where an increase of volume fraction of aggregates leads to a decrease of cumulative saturation level. The influence of ITZ is observed on a local and global scale by varying the ITZ diffusivity coefficient. The moisture simulation is also modelled on a finer mesh to see the effect of mesh refinement. Finally, moisture transport is simulated in a numerical model that combines a planar crack along with heterogeneities and flow is observed at different time stages.

Contents

List of Figures

List of Tables

1 Introduction

1.1. General Overview

Concrete is the most widely used construction material in the world. A huge portion of the total infrastructure budget is spent on repair and maintenance of existing structures which are largely related to the lack of durability of concrete structures. Mass transport in concrete is a significant factor in determining the durability and service life of concrete structures. The transport of fluids containing agents of deterioration can cause degradation of cementitious material and corrode the steel reinforcement. Mass transport mechanisms may influence durability of concrete structures indirectly by controlling penetration rate of aggressive agents [48]. Corrosion of steel bars due to presence of chloride ions is considered to be one of the main deterioration mechanisms of concrete [48] which decreases it's service life. Water acts as the major fluid that is transported within concrete and it can act as a carrier of chloride ions, sulphates, carbonates, etc. that have severe deteriorating effects on concrete. Hence, it is important to know the mass transport phenomena in concrete and the various factors affecting these phenomena.

Concrete is found in nature in an unsaturated and cracked state. Mechanical loading can induce cracks having macroscopic crack widths. Other type of loading such as drying shrinkage leads to microcracks distributed in concrete and cannot be related to a macroscopic crack width [13, 51]. Cracking accelerates the penetration of fluids and aggressive agents within the concrete which subsequently leads to faster corrosion of steel and reduces the service life of the structure. Hence, to study the coupling of cracking and transport phenomena is important to predict the durability and the service life of structures. If concrete is treated as uncracked and saturated material, its service life cannot be realistically evaluated and hence, mass transport mechanisms should be considered in cracked and unsaturated concrete.

Modelling of flow in cracked concrete within the finite element framework can be done through lattice models which are shown to be capable of describing complex fracture patterns on the mesoscale of concrete [18, 42]. Mass transport can be described by a lattice of conduit elements, which can be linked to the structural lattice to couple fracture and transport processes [6].

The inherent heterogeneity of concrete complicates the transport processes. The inclusion of aggregates reduces concrete permeability by decreasing the volume fraction of porous matrix (dilution effect) and further lengthening the flow path (tortuosity effect) [1, 19, 58]. Also, the presence of aggregates leads to formation of interfacial transition zones (ITZ) which are highly porous and accelerate transport [43]. Numerical analysis of mass transport considering concrete as a three-phase composite consisting of coarse aggregates, hardened cement mortar and ITZ enhances the understanding of non-homogeneous concrete and can be used to represent the influence of material composition on transport properties [48] which further affects the durability and service life of structures.

This thesis focuses on investigating moisture transport through capillary absorption in uncracked and cracked concrete under unsaturated conditions. Computational tools are developed to understand the nature of flow of moisture in concrete based on moisture penetration and moisture distribution profiles.

1.2. Research Objectives

The aim of this research is to study moisture transport through capillary absorption in both homogeneous and heterogeneous unsaturated concrete under uncracked and cracked conditions. This thesis seeks to build a numerical model that is properly able to simulate moisture transport in concrete. The following research questions are formulated to achieve the objective of this research:

Research Questions

1. What are the effective modelling techniques to simulate moisture transport in concrete?
2. How does a discrete crack affect moisture transport in concrete?
3. How is moisture transport affected due to the multi-phase nature of concrete?

1.3. Research Methodology

Firstly, an extensive literature review is undertaken to lay the theoretical foundation of the mass transport mechanisms in concrete. The literature study highlighted the physics behind the transport mechanisms, the derivation of governing equations and their discretization in one-dimensional lattice networks. A numerical model in MATLAB, using finite elements, is already provided that could simulate moisture transport in homogeneous, unsaturated and uncracked concrete in a 3-D domain discretized through Delaunay elements. This research seeks to modify the existing numerical model to incorporate heterogeneities and discrete cracks in the 3-D concrete domain discretized through Voronoi elements. The properties of the numerical solver of the given numerical model are also varied to determine its influence on the numerical analysis. The numerical models are validated by comparing numerical results with experimental results and parametric studies on the validated numerical models are carried out to understand the nature of flow.

1.4. Thesis Overview

The thesis is structured in a systematic order in which each chapter pertains to a specific research topic while displaying a coherent transition between topics. Chapter 2 presents a critical review of literature related to the underlying transport mechanisms of concrete. In this chapter, the physics behind the transport mechanisms is studied along with the mathematical formulation of the phenomena followed by the basic discretization procedure of the domain. Chapter 3 focusses on moisture transport in uncracked concrete. In this chapter, numerical tools are developed to simulate moisture transport in uncracked concrete and the results are validated with experimental results. The numerical model formulated in this chapter serves as a basis to simulate moisture transport when a discrete crack and heterogeneities are included in concrete as presented in the subsequent chapters. In Chapter 4, moisture transport in cracked concrete with a discrete crack is simulated considering two approaches of treating the cracked domain. Results from both the approaches are compared with experimental results and to each other in order to evaluate the feasibility of the approaches. In Chapter 5, moisture transport in concrete is simulated considering it as a multi-phase material. In this chapter, the effect of aggregates on the penetration depth of moisture in concrete is observed. This is followed by the effect of ITZ diffusivity on the transport mechanism and a mesh refinement study ending with a simulation of moisture transport in a domain combining a discrete crack and heterogeneous nature of concrete. Finally, Chapter 6 concludes the thesis by presenting the main conclusions of the results followed by recommendations for further research.

2

Theory and Literature Review

2.1. Mass Transport in Concrete

Depending on various physical and material parameters, there are different mass transport mechanisms in concrete. Cracks provide additional routes for water along with chlorides to ingress within the material [8, 47]. The concrete pore structure, extent of saturation, presence of heterogeneities and environmental conditions are some of the factors that also govern the transport mechanism within concrete [27]. In this section, first the transport behaviour in sound concrete is elucidated followed by the effect of having cracks and heterogeneities in the concrete on the transport mechanism.

2.1.1. Transport Mechanisms in Sound Concrete

Permeable nature of concrete allow fluids and ions to travel within it. The underlying kinetics of mass transport in the body of concrete depends on the mechanism it follows to travel within the material. Considering concrete to be isotropic, homogeneous and having uniform porosity, transport mechanisms in concrete can be categorized as:

Capillary absorption

Concrete is a porous material and in the unsaturated state, fluids such as moisture or gases ingress the pores of concrete. Due to surface interaction between the entering fluid and the walls of the pores, there is a development of capillary pressure. The interaction between the fluid and the pore walls depends on the surface tension, viscosity, contact angle of the fluid with the pore wall, size of the pore etc [39]. The capillary pressure in the pores is given by Young-Laplace equation:

$$P_c = -\frac{2\gamma cos\theta}{r} \tag{2.1}$$

where P_c is the capillary pressure developed in the pore, γ is the surface tension between the liquid and the vapour phase, θ is the contact angle and the r is the pore radius [39]. Capillary pores are assumed to be spherical cavities that are connected to each other through cylindrical capillary tubes. The gradient of the capillary pressure across the material body becomes the driving force for the fluids to travel within the material. For unsaturated concrete, this is the dominant mechanism through which mass transfer occurs.

Diffusion

Diffusion is caused by a gradient in the concentration of the ionic species in concrete which causes the ions to travel from a region of higher concentration to a region of lower concentration. Diffusion is considered the principal mechanism through which ionic species such as chlorides travel within the concrete.

Degree of saturation plays an important role in the diffusion process. For ionic diffusion, greater saturation of the concrete will escalate the diffusion process and it is most effective in fully-saturated concrete. If the concrete is saturated, it's pores are filled with water which acts a medium for chloride

ions to diffuse within the concrete. Usually, the diffusion of chloride ions in concrete is described by Fick's second law [39]:

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial C}{\partial x}\right) \tag{2.2}$$

where C is the chloride ion concentration (mol/m^3), D is the diffusivity (m^2/s), x is the spatial dimension (m) and t is the time (s).

To solve the above mentioned differential equation, diffusivity and chloride surface concentration (C_s) is considered unvarying in space and time. The following initial and boundary conditions are considered [39]:

$$C = 0 \;\; x > 0 \; t = 0 \tag{2.3}$$

$$C = C_s \;\; x = 0 \; t > 0 \tag{2.4}$$

Conforming with the initial and the boundary conditions and assuming concrete to be a semi-infinite medium, the chloride ion concentration can be evaluated using the mathematical error function erf [48]:

$$C(x,t) = C_0 + (C_s - C_0)\left[1 - erf\left(\frac{x}{2\sqrt{Dt}}\right)\right] \tag{2.5}$$

where C_0 is the initial chloride concentration. The diffusion coefficient D in the above expression is usually replaced by D_{eff} which implicitly considers the physical characteristics and connectivity of the pore structure along with the effect of binding of the ionic species [39].

Electromigration

A presence of an electric field across the body of the concrete causes the ionic species present in concrete to travel along the electric field. The negative ions (such as Cl^-) migrate towards the anode and the positive ions migrate towards the cathode. Saturated concrete promotes the transport of the ions under an electric field. The porosity and the connectivity of the pore network also affect the magnitude of the electromigration. The diffusion coefficient for electromigration is given as [39]:

$$D_i = \frac{RTu_i}{z_i F} \tag{2.6}$$

where D is the diffusion coefficient, R is the gas constant, T is the temperature, u_i is the mobility of the ion, z_i is the valency of the ion i and F is the Faraday constant.

Permeation

A hydraulic pressure gradient present across the concrete causes the permeates to travel under the pressure head. Permeability of concrete is an important factor that determine the rate of permeation. Darcy's law can be applied to express the process of permeation:

$$\frac{\partial q}{\partial t}\frac{1}{A} = k_p \frac{\partial h}{\partial x} \tag{2.7}$$

where $\partial q/\partial t$ is the rate of flow per unit area, A is the area, k_p is the permeability coefficient and $\partial h/\partial x$ is the gradient of pressure. The permeability coefficient of concrete depends upon the connectivity of the pore network, porosity and viscosity [39].

In the domain of the work presented in this thesis, the transport of water through capillary absorption is the only transport phenomenon that is considered while neglecting the rest of the mechanisms. But it is important to note that the transport of mass within concrete takes place with one or more combinations of the above mentioned phenomena.

2.1.2. Effect of Cracks on Moisture Transport

Studying the transport phenomena in uncracked concrete is important but not enough. Almost all of the concrete structures that are practically used are subjected to mechanical loading and shrinkage deformations because of which cracks and microcracks form in the body of the concrete. Cracks formed because of mechanical loading expose additional concrete surfaces to the atmosphere and also increases the diffusivity of the matrix in the vicinity of the crack. Microcracks are formed due to

shrinkage deformations and they are not visible to the naked eye but all concrete structures have them. They are really fine cracks present in the concrete matrix and if these cracks connect, they can form a continuous network of high permeability within the matrix [39]. This will provide additional passages for water to travel within concrete.

Crack as an additional porous phase in mortar
Cracks can be considered as an additional phase of concrete as the transport properties in and near the crack changes. The porosity of the space that comprises the crack within the domain of the material can be considered as 1 because the cracked volume does not contain any cementitious material and the entire cracked volume can be considered as a void. This results in high permeability within the crack and because of this, the transport of water in and around the crack escalates considerably. Subsequently, the transport of ions within the cracked concrete also increases and this can accelerate the deterioration of concrete by corroding the reinforcement and exfoliating the concrete cover [12, 50]. Usually cracking and mass transport phenomena constitute a two-way coupling mechanism. Cracking escalates the transport in the concrete which subsequently accelerates the deterioration of the concrete which causes the present crack to propagate further [37]. The scope of the work presented in this thesis is limited to a one-way coupling mechanism in which the effect of cracking is considered on the water transport in concrete and not vice-versa. This is a reasonable simplification because the effect of water transport on cracking is a comparatively slower process and does not significantly affects the durability of concrete.

Capillary suction in cracks
Capillary suction is a process in which a liquid rises up in a thin capillary tube because of the interfacial interactions between the surfaces of the fluids and the solid. If the crack is narrow and the crack surface is exposed to moisture, then the crack can act as a capillary tube (or a combination of such tubes) and the water can rise up in the crack and spread to the adjacent domain through the new exposed surfaces. Capillary suction in the crack occurs when moisture surface interactions between the crack walls and moisture present at the boundary is also accounted for, in addition to the porous nature of the crack.

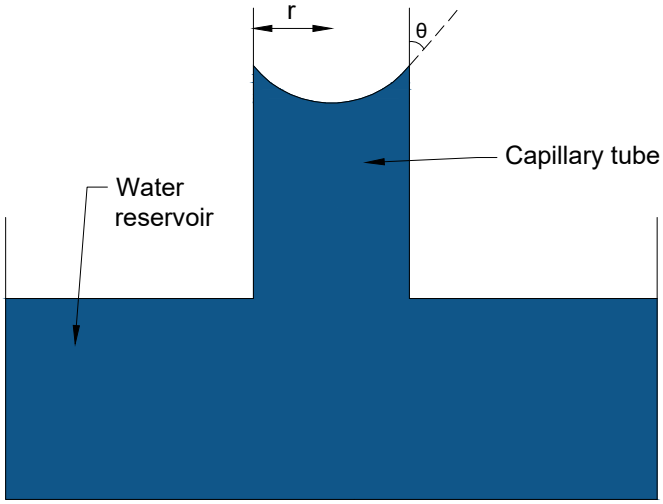


Figure 2.1: Capillary rise in a tube

In a capillary tube submerged in water, a capillary pressure is developed resulting from the surface tension of the fluids with the wall of the capillary tube. This capillary pressure makes the liquid rise up in the tube. The capillary pressure in the tube can be obtained through the well known Young-Laplace equation as:

$$p_c = \frac{2\sigma cos(\theta)}{r} \tag{2.8}$$

where σ is the surface tension, θ is the contact angle and r is the radius of capillary as shown in Figure2.1.

2.1.3. Effect of Aggregates on Moisture Transport

The assumption that concrete is homogeneous does not entail a numerical model that represents the actual physical transport behaviour of concrete. Concrete can be considered as a three-phase composite at the meso-scale as shown in Figure2.2. The first phase is the cement matrix, the second phase is the aggregates within the matrix and the third phase is the interfacial transition zone (ITZ) between the aggregates and the matrix. All three phases have different structural and transport properties. Thus, the mass transport mechanism is affected by the presence of aggregates in the matrix.

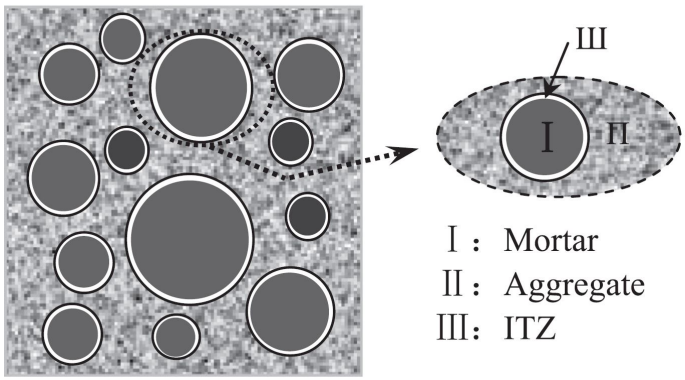


Figure 2.2: Mesoscale structure of concrete [48]

When large aggregates are surrounded by small cement particles in concrete, there is a *wall effect* at the boundary of aggregates. The cement particles pack efficiently together when present in the bulk of the material but this packing is comparatively loose near a wall-like aggregate boundary. As a result of this, the cement particle density near the aggregate boundary is less than that in the bulk material [44]. Also, during the hydration process, cement particles expand in all directions but the ones present near the boundary of the aggregate cannot expand in the direction of the aggregate [10]. These two factors cause the formation of a coarse micro-structure around the aggregate surface leading to a region of high porosity and diffusivity around the aggregate. This region comprised of increased transport properties forms the interfacial transition zone (ITZ) for the transport mechanism. The thickness of the ITZ region depends upon the ratio of the size of the aggregate to the cement particles but usually it has been seen to lie between 15-50μm [44].

Considering the aggregates themselves, the permeability of the aggregates compared to that of the cement matrix is negligible and it is assumed that there is no net flow through the aggregates. Only when high-porosity artificial produced aggregates are included in the matrix, the flow through the aggregate has to be considered.

2.2. Governing Equations of Mass Transport

Mathematical formulation of the transport phenomena is important as these equations describe the physics of flow and will be discretized in the numerical model to simulate moisture transport. In this section, governing equations for moisture transport through capillary absorption in uncracked and cracked concrete will be presented.

2.2.1. Capillary absorption in sound concrete

The flow theory used to describe water uptake in soil can also be used to explain capillary absorption in unsaturated concrete [22]. Linear Darcy's law is used to describe water flow in perfectly saturated concrete. But as shown in Figure 2.3, there are various zones of saturation in concrete. The outermost saturated zone has pores fully filled with water, the innermost dry zone has pores that are completely dry and the unsaturated zone in between has pores that are partially filled with water. This represents

the actual state of concrete and therefore it is desirable to have a unified description of saturated and unsaturated flow.

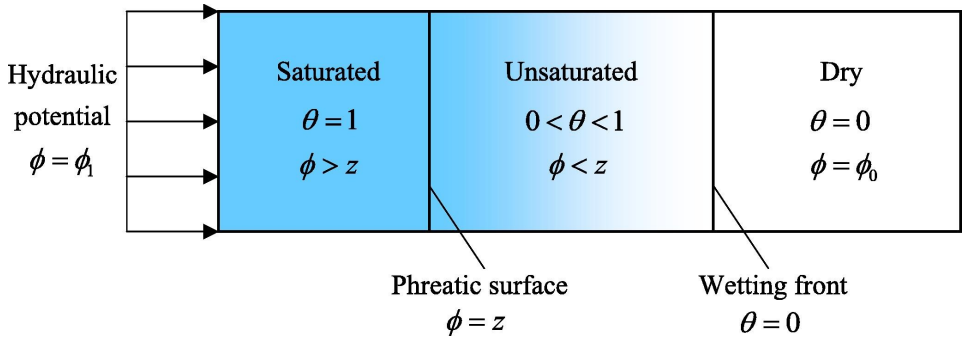


Figure 2.3: Unsaturated flow of water through concrete under a hydraulic potential [20]

Under the assumption that concrete is considered as a semi-infinite porous medium, flow in concrete only involves a liquid phase, the system of concrete is isothermal and vapour adsorption and evaporation processes on the surface are neglected, the extended Darcy's law can be used to describe the unsaturated flow in concrete [16] which is given as:

$$u = K(\theta)\nabla p_c \tag{2.9}$$

where u is the flow rate of water, p_c is capillary pressure, $K(\theta)$ is hydraulic conductivity function and θ is defined as relative water content which is given by:

$$\theta = \frac{\Theta - \Theta_i}{\Theta_s - \Theta_i} \tag{2.10}$$

where Θ_i and Θ_s is the initial and saturated volumetric water content. Hence, θ is an indicator of relative saturation level in the concrete pores. Considering the concrete skeleton to be rigid and water present in the concrete to be incompressible, the mass conservation equation can be written as [57]:

$$\frac{\partial \theta}{\partial t} + div\ u = 0 \tag{2.11}$$

Combining the mass balance equation and extended Darcy's law due to capillary pressure gradient, and assuming unsaturated and homogeneous state, leads to Richards equation used to describe unsaturated flow of water in porous media which is given as [16, 22, 48]:

$$\frac{\partial \theta}{\partial t} = \nabla(D(\theta) \cdot \nabla\theta) \tag{2.12}$$

where, $D(\theta)$ is the hydraulic diffusivity (L^2T^{-1}) of the material. Eq.(2.12) is the saturation-form of the Richard's equation. The hydraulic diffusivity ($D(\theta)$) is closely related to unsaturated permeability ($K(\theta)$) as [48]:

$$D(\theta) = -K(\theta)\left(\frac{\partial p_c}{\partial \theta}\right) \tag{2.13}$$

Hydraulic diffusivity is considered as the fundamental indicator to describe moisture transport in unsaturated concrete. It is known to be dependent on the material type, saturation level and temperature [20]. Under isothermal conditions, for the case of unsaturated concrete, hydraulic diffusivity can be approximated as [33]:

$$D(\theta) = D_0 e^{n\theta} \tag{2.14}$$

where n is an empirical constant and has been proved to lay between 6-8 [16] and D_0 is the diffusivity of dry cement. D_0 can be estimated with good results through sorptivity experiments as proposed in [22]:

$$D_0 = \frac{n^2 S_0^2}{(\Theta_s - \Theta_i)^2[e^n(2n-1) - n + 1]} \tag{2.15}$$

where S_0 is the sorptivity of the material which is used for characterizing tendency of porous material to absorb and transmit water through capillary absorption [20]. Figure 2.4 shows a concrete specimen where one face of the sample is submerged in water and sorptivity test is conducted.

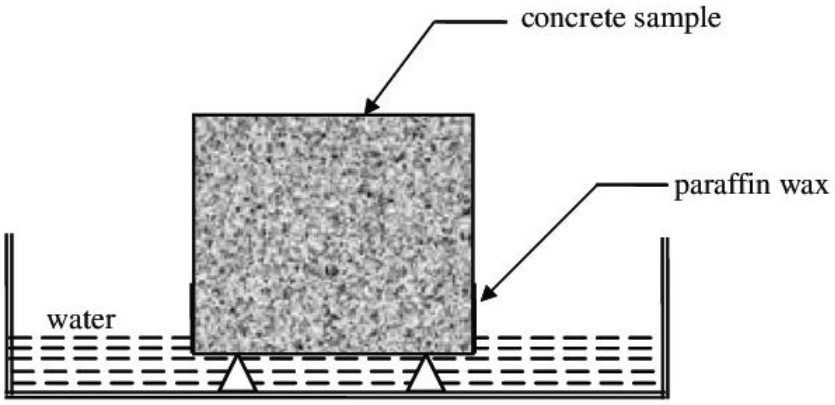


Figure 2.4: A concrete specimen under sorptivity test [15]

In the sorptivity test, the variation of cumulative volume of absorbed water per unit area (i) with the square root of time ($t^{1/2}$) is observed which follows:

$$i = S_0 t^{1/2} \tag{2.16}$$

where sorptivity (S_0) is obtained from the slope of the best-fit line of the curve $i - t^{1/2}$ obtained through the experiment [20].

2.2.2. Transport Equations for Cracking and Heterogeneities

The basic transport mechanism in homogeneous and heterogeneous concrete is the same. In both cases, moisture transport is considered through capillary absorption and the governing equations remain the same. The only difference in heterogeneous concrete is that the aggregates present in the matrix can slow down the moisture transport and the ITZ formed around the aggregates accelerates transport. Aggregates are considered impervious so no flow is considered through them. Flow in ITZ also occurs through the process of capillary absorption but with enhanced diffusivity and hence, the governing equations remain the same.

Crack as an additional phase in mortar

As mentioned earlier, cracking causes an increase in the transport properties of the matrix in the vicinity of the crack. The diffusivity of the elements in the crack is given as a function of permeability and the pressure head [48]:

$$D_{cr}(\theta) = -K_{cr}\frac{dp_c}{d\theta} \tag{2.17}$$

where p_c is the capillary pressure, θ is the saturation level and K_{cr} is the unsaturated permeability of cracked concrete. In case of uncracked concrete, the function of unsaturated permeability is given as [46]:

$$K(\theta) = \frac{k_s k_r(\theta)}{\mu} \tag{2.18}$$

where μ is the viscous coefficient of water, k_s is the permeability of uncracked concrete under saturated condition with dimensions L^2 and $k_r(\theta)$ is the normalized permeability given as [46]:

$$k_r(\theta) = \sqrt{\theta}[1 - (1 - \theta^{1/m})^m]^2 \tag{2.19}$$

where $m = 0.4396$ is the Van-Genuchten coefficient successfully used to describe soil-water retention curve [45]. Eq(2.18) is used to describe permeability in uncracked concrete under unsaturated conditions. To get the function for permeability in cracked concrete, k_s in eq(2.18) is replaced by k_s^{cr} which

is the permeability of cracked concrete under saturated conditions. In this research, the expression for k_s^{cr} is obtained as:

$$k_s^{cr} = \left(\sum_{i=1}^{n} w_i^3 l_i\right)\frac{1}{12A} \tag{2.20}$$

where w_i and l_i are the equivalent crack widths and crack lengths respectively of a Voronoi element (a-b) passing through a Delaunay triangle (1-2-3) as shown in Figure 2.5 and A is the area of the Delaunay triangle (1-2-3). This can be better explained once the domain is discretized and hence, this term is explained during the construction of numerical model for moisture transport in cracked concrete given in Chapter 4.

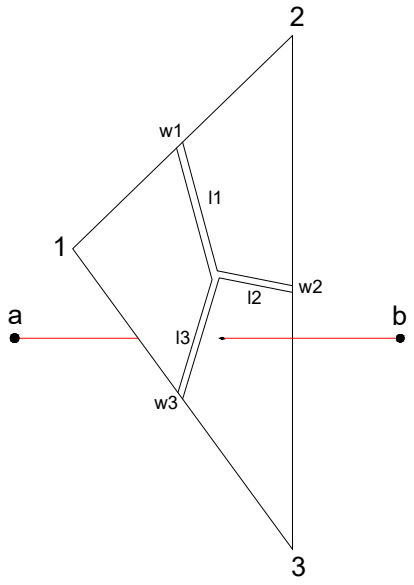


Figure 2.5: Equivalent crack properties of a transport element (a-b)

Water retention curve is assumed to follow Mualem [30] distribution and based on that, a relation between saturation level θ and capillary pressure p_c is proposed in [28] which is given as:

$$\theta(p_c) = [1 + (p_c/p_r)^{1/(1-m)}]^{-m} \tag{2.21}$$

where p_r is the reference pressure and experimentally determined to be $18.6237\ N/mm^2$. Combining eq(2.18)-eq(2.21), the cubic law for the diffusivity of the cracked domain is given as:

$$D_w^{cr}(\theta) = \left(\sum_{i=1}^{n} w_i^3 l_i\right)\frac{\xi p_r(1-m)}{12\mu Am}\sqrt{\theta}[1-(1-\theta^{1/m})^m]^2\theta^{-1/m-1}(\theta^{-1/m}-1)^{-m} \tag{2.22}$$

where ξ is the tortuosity of the crack. All the elements in the domain representing cracked concrete as assigned diffusivity based on eq(2.22).

Capillary suction

Capillary absorption can be described by Lucas-Washburn equation derived by equating capillary pressure difference across the interface between two immiscible fluids to the pressure loss due to internal friction [11]. The Lucas-Washburn equation takes into account the surface interactions between a capillary tube and the liquid surface and describes the dynamic flow of liquid in a capillary tube by balancing driving capillary force and retarding viscous force [36]. Initially, Lucas [24] and Washburn [49] considered a flow regime where the effect of inertia and gravity was neglected and the capillary force was

balanced with viscous force giving the equation:

$$\frac{2\sigma cos(\theta)}{r} = \frac{8\mu z\dot{z}}{r^2} \tag{2.23}$$

where σ is the surface tension, r is the radius of capillary, μ is dynamic viscosity, θ is the solid/liquid contact angle and z is the capillary rise height. But equation2.23 had some limitations. For the small time stages just at the beginning of the flow, the fluid velocity tends to go to infinity which is not realistic. To account for that, an inertial term [5, 23, 36] can be added into the equation which gives:

$$\frac{2\sigma cos(\theta)}{r} = \frac{8\mu z\dot{z}}{r^2} + \rho\frac{d(z\dot{z})}{dt} \tag{2.24}$$

where ρ is the density of the fluid. Equation2.24 still has a limitation as when the flow is considered to occur in a vertical capillary, there is no limit to the maximum height that can be attained in the capillary. This can be accounted for by considering the effect of gravity as:

$$\frac{2\sigma cos(\theta)}{r} = \rho g z sin(\phi) + \frac{8\mu z\dot{z}}{r^2} + \rho\frac{d(z\dot{z})}{dt} \tag{2.25}$$

where ϕ is tube inclination angle, ρ is the density of liquid, g is the gravitational acceleration and t is the capillary rise time. The parameters used in the equation can be seen in Figure2.6. Equation2.25 considers the effect of gravity and inertia completing the momentum balance of the original Lucas-Washburn equation and is known as amended Lucas-Washburn equation.

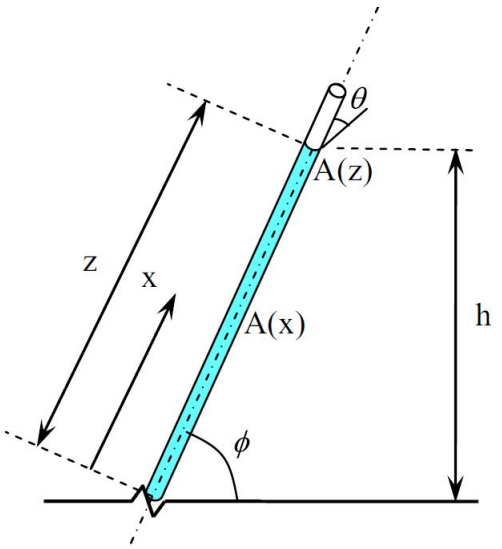


Figure 2.6: Parameters used to model water flow in a uniform capillary [11]

As it was mentioned earlier that the inertial term is relevant only for the initial time stages of flow and it is shown in [9] that the difference of capillary rise height is insignificant when the inertial term is neglected and when it is considered. Hence, the inertial term is disregarded for the time scales considered in this research which reduces the equation to:

$$\frac{2\sigma cos(\theta)}{r} = \rho g z sin(\phi) + \frac{8\mu z\dot{z}}{r^2} \tag{2.26}$$

The experiments conducted in [11] found that the above equation was not able to reproduce the capillary rise behaviour in an actual crack and some corrections have to be applied to eq(2.26):

1. Stick-slip behavior at meniscus: Raindrops stuck on a window can trickle down or stop at random times as the air-water contact line is pinned by the random potential on glass surface [41]. When the weight of the drop is more than the pinning force, the drop moves down and when the pinning force is greater than the weight of the drop, the drop stops falling. Similar effect may be observed

in a glass capillary. For a vertical capillary of radius r, the equilibrium rise height of the water column is given by:

$$h_{eq} = \frac{2\sigma cos(\theta)}{r\rho g} \tag{2.27}$$

If the pinning effects are considered in the capillary, capillary rise would stop at a height less than the equilibrium rise height (h_s). As a result, there will be a reduction in capillary pressure in the capillary tube which is given as [41]:

$$p_s = \rho g(h_{eq} - h_s) \tag{2.28}$$

The final capillary pressure is given as:

$$p_{ceff} = p_c - p_s = p_c(1 - \beta_s) \tag{2.29}$$

where

$$p_c = \frac{2\sigma cos(\theta)}{r} \tag{2.30}$$

$$\beta_s = 1 - \frac{h_s}{h_{eq}} \tag{2.31}$$

2. Dynamic contact angle: As presented in [17], there are frictional effects associated with a rising liquid front in a capillary that are responsible for a dynamic contact angle. The effect of the dynamic contact angle is proposed in the form of a retardation constant allowing for frictional dissipation at a moving front given as [3]:

$$\beta_m = \frac{k_b T v}{2\pi\lambda^3 \kappa_0^s \hbar}\eta \tag{2.32}$$

where k_b is the Boltzmann constant, T is the temperature, v is the molecular volume of the liquid, $lambda$ is the distance between adsorption sites on the solid surface, k_0^s is the surface rate constant for molecular displacement, $2\pi\hbar$ is the Planck constant and η is the viscosity of the liquid. The correction is included in the Lucas-Washburn equation by replacing the term of static contact angle with the term of dynamic contact angle as [17]:

$$\sigma cos(\theta(t)) = \sigma cos(\theta_0) - \beta_m \dot{z} \tag{2.33}$$

3. Wall slip: The correction for wall-slip affects the Hagen-Poiseuille (H-P) equation. The correction is incorporated as [29]:

$$\dot{z} = \left(\frac{r\beta_w}{2} + \frac{r^2}{8\mu}\right)\left(\frac{dp}{dx}\right) \tag{2.34}$$

where, p is the mean position x along the tube. β_w is Mooney's constant with units m^3/Ns.

Including all the three corrections in the Lucas-Washburn equation and neglecting the inertial term leads to the equation:

$$\left(\frac{2\beta_m}{r} + \frac{z}{\frac{r\beta_w}{2} + \frac{r^2}{8\mu}}\right)\dot{z} = p_{c0}(1 - \beta_s) - \rho g z sin(\phi) \tag{2.35}$$

2.3. Numerical Modelling of Mass Transport

In order to simulate flow through the governing transport equations, the domain has to be discretized using finite elements and the discretized form of the transport equations have to be formulated. In this section, the properties of the numerical model will be described including the type of elements used to generate the mesh along with the procedure to generate the dual mesh. In the last subsection, different approaches to model aggregates in concrete are presented.

2.3.1. Lattice Models

Lattice models are an assembly of discrete one dimensional lattice elements representing a structural solid. Such models allow simplicity in the modelling of complex structures with lighter computational cost. Phenomena such as localized failure and cracking is not easy to model in solid elements but can be efficiently and easily modelled through lattice elements [32]. Solutions of complex governing equations can be simply solved on 1-D lattice elements which is easier to solve than on 2-D or 3-D elements. Due to these reasons, lattice elements are chosen to represent the structural and transport properties of the domain and the governing equations are discretized on the lattice elements composing the lattice model.

To represent the structural behaviour of concrete, lattice beam elements are chosen. Lattice beam elements having 2 nodes and 6 degrees of freedom (three displacement and three rotational) at each node are used in the numerical model (Figure2.7) as the structural behaviour like localized damage and cracking can be simulated easily on them.

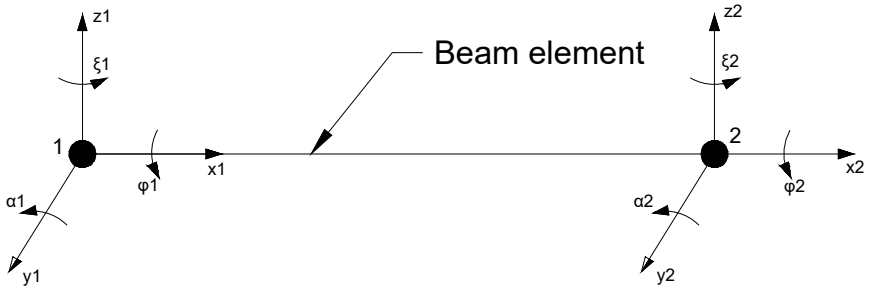


Figure 2.7: Structural lattice element showing corresponding nodal degrees of freedom

To represent the transport behaviour of concrete, two noded lattice elements are chosen with saturation level as the degree of freedom at each node varying linearly with the element length (Figure2.8). The governing equations for the moisture transport through capillary suction has to be discretized on 1-D lattice elements to simulate moisture transport in the domain.

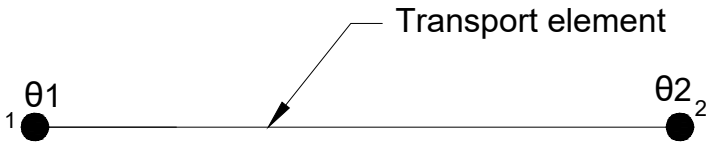


Figure 2.8: Transport lattice element showing corresponding nodal degrees of freedom

2.3.2. Mesh Generation

A semi-random mesh is used to discretize the domain using lattice elements. Delaunay and Voronoi meshes are used which are the geometric dual of each other. The Delaunay mesh is used to represent the structural properties of the domain and the Voronoi mesh is used to represent the transport properties of the domain. The approach introduced in [54], is adopted for mesh generation.

Generation of Delaunay mesh

The entire domain is divided into cubic voxels, as shown in Figure2.9, with the length of each side equal to L. Each cubic voxel houses a Delaunay node within a definite region present inside the body of the voxel that depends on the randomness of the mesh.

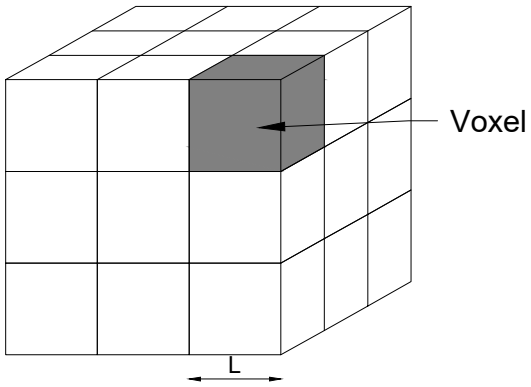
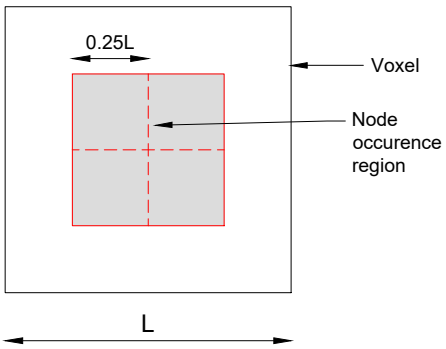


Figure 2.9: Division of domain into voxels

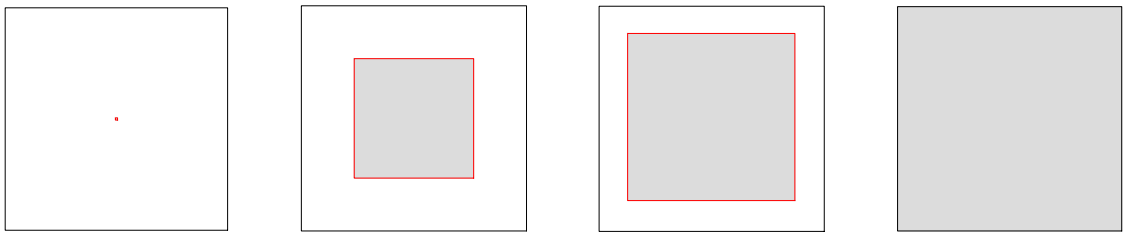
The procedure to determine the position of a node within a voxel is shown in Figure2.10a. A uniform randomness value that varies between 0-0.5 is assigned to each voxel in the mesh. The position of the Delaunay node depends on the randomness value. Within each voxel, there is a smaller cube (sharing the same centre as the voxel), with the length of each side equal to:

$$l = 2rL \tag{2.36}$$

where r is the randomness, L is the length of each side of the voxel and l is the length of each side of the inner cube within the voxel. This inner cube comprises the boundary of the region where the Delaunay node can occur. The position of the node within the inner cube is totally random and can occur anywhere within it.



(a) Region of occurrence within a voxel with a randomness of 0.25



(b) Randomness=0 (c) Randomness=0.25 (d) Randomness=0.4 (e) Randomness=0.5

Figure 2.10: Region of occurrence in a voxel for different randomness

The volume of the inner cube changes with the change in randomness as can be seen in Figure2.10b-Figure2.10e. When randomness is zero, the inner cube reduces down to a point and the Delaunay node can only occur at the centre of the voxel. As randomness increases, the volume of the inner cube also increases, thus increasing the boundary of the region where the node can occur. When the randomness is increased to 0.5, the inner cube coincides with the voxel itself and the Delaunay node can lie anywhere within the voxel in that case. This scenario is not preferred because the Delaunay node can lie at the boundary of the voxel and that can cause problems in constructing the Delaunay tetrahedra from the Delaunay nodes. In the numerical model adopted in the research, a randomness of 0.2 is considered for all discretized meshes. Once the Delaunay nodes are generated within the voxels, they are connected to form the Delaunay elements as shown in Figure2.11.

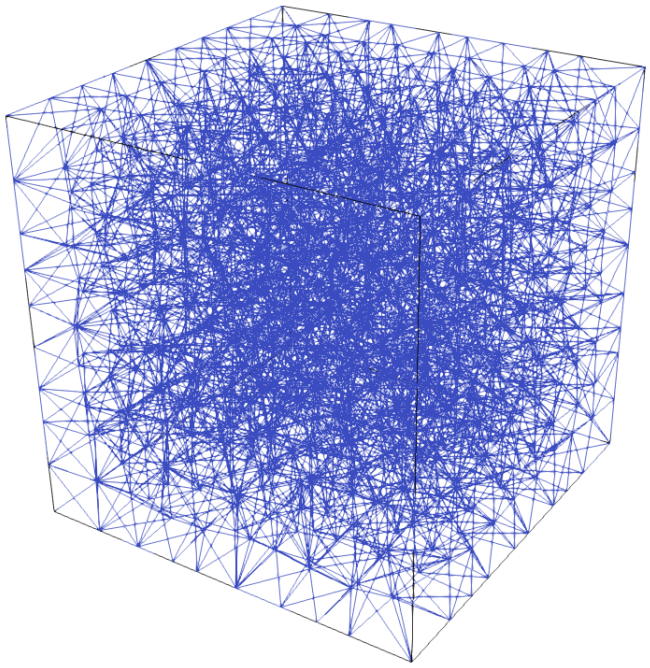


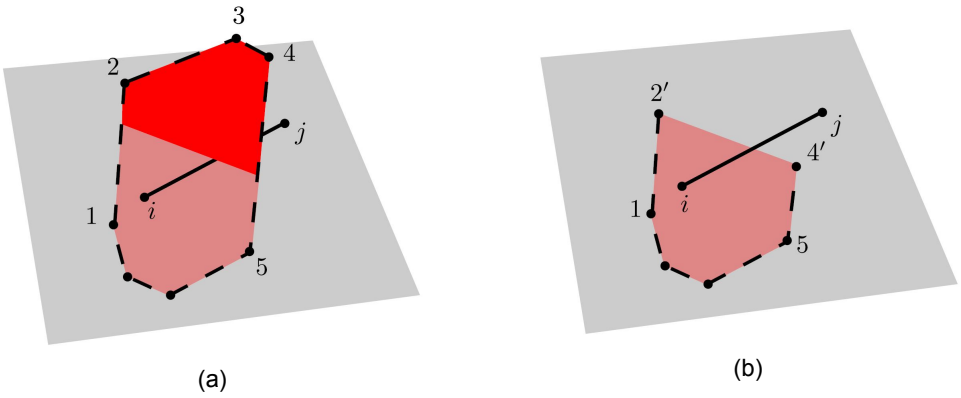
Figure 2.11: Delaunay mesh

In constructing the Delaunay mesh, in order to have a well defined boundary of the domain, the Delaunay nodes lying on the corner and edge voxels have to be generated in a different manner. The Delaunay nodes that form the eight corners of the domain have a predetermined position and there is no randomness in their position. The nodes lying on either of the 12 edges are generated in such a way that they can occur on the edge but their position on the edge is not known and that it generated randomly. All the nodes lying either of the 6 faces are generated in such a manner that they are restricted to lie only at the corresponding face but their position within the face is generated randomly. In such way, a Delaunay mesh with well defined boundaries is generated as shown in Figure2.11.

Generation of Voronoi mesh

The Voronoi mesh is the geometric dual of the Delaunay mesh. Each Voronoi node is generated in such a way that it is the centre of the circumsphere of a tetrahedron composed of four Delaunay nodes [34]. In this way, another semi random mesh is generated through the Delaunay mesh. But there is a problem in generating the Voronoi nodes corresponding to the Delaunay nodes lying at the boundary of the cube. At the boundary face of the cube, the four Delaunay nodes lie on a plane which implies that the radius of the circumsphere will extend till infinity and the Voronoi node will lie at infinity. Due to this, there won't be any definite boundary of the Voronoi elements. To counter this problem, a solution is proposed in [54], in which auxiliary Delaunay nodes are introduced in the domain. At each boundary

face, auxiliary nodes are generated which are the mirror images of the nodes in the domain (considering the boundary plane to be the plane of the mirror). Voronoi nodes are generated corresponding to both Delaunay nodes and auxiliary nodes and the Voronoi cells crossing the domain boundary are truncated at the position of the boundary plane.



(a) (b)

Figure 2.12: Discretization of domain boundaries: (a) Voronoi facet of Delaunay edge i–j located on the surface of the domain after initial tessellation; and (b) modified arrangement used for definition of transport nodes and elements [31].

Figure2.12a shows a Voronoi face 1-2-3-4-5 that serves as the cross section of a Delaunay element i-j. The Voronoi face is extending beyond the domain boundary and it is truncated as shown in Figure2.12b. The Voronoi face 1-2'-4'-5 is retained in the mesh and lie completely within the domain boundary. In this way, a Voronoi mesh is constructed which is the geometric dual of the Delaunay mesh and lies completely within the domain as shown in Figure2.13.

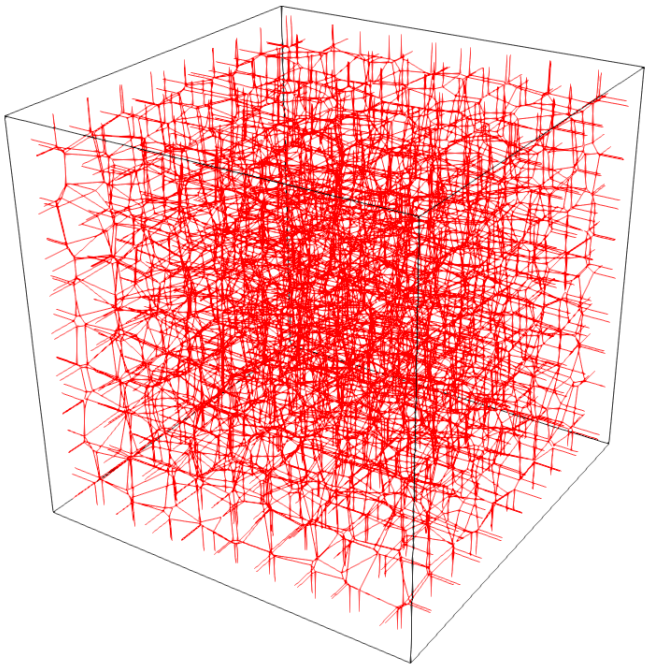


Figure 2.13: Voronoi mesh

2.3.3. Modelling Heterogeneities in Concrete

To model the transport behaviour of heterogeneous concrete, aggregates have to be incorporated in the numerical model. The aggregates are included in the discretized mesh such that a part of the elements represent the aggregate body. There are different methods to project the aggregates on the discretized domain.

Projecting the aggregates with their original shape

In this case, the aggregates are included in the discretized mesh according to their original shape as shown in Figure2.14. The boundary of the aggregates is retained in this case.

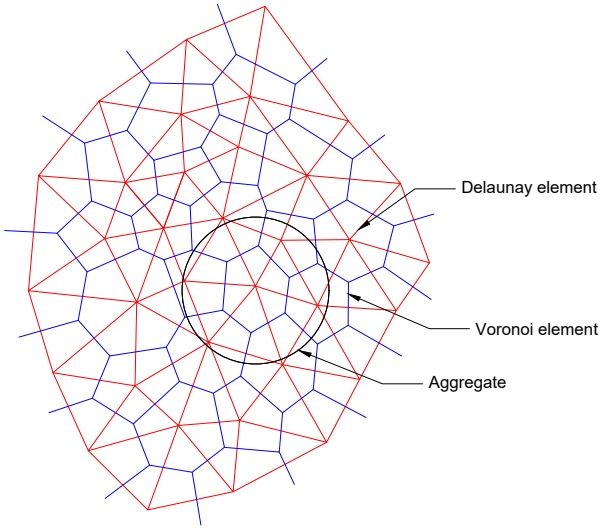


Figure 2.14: Spherical aggregate

In order to model the structural and transport properties of aggregates in this case, it will be comparatively complex. The Delaunay elements represent the structural elements and Voronoi elements represent the transport elements. As it can be seen from the figure that both Delaunay and Voronoi elements pass through the boundary of the aggregates which implies that when the respective structural and transport properties will be modelled on these elements, there will be discontinuity in the properties at the location of the boundary. Embedded discontinuities will have to be included in the shape functions for both Delaunay and Voronoi boundary elements.

Projecting the aggregates on Delaunay elements

In this case, the shape of the aggregates is projected on Delaunay elements as shown in Figure2.15. It can be seen that once the aggregate is projected on the Delaunay element, the original shape of the aggregate is modified.

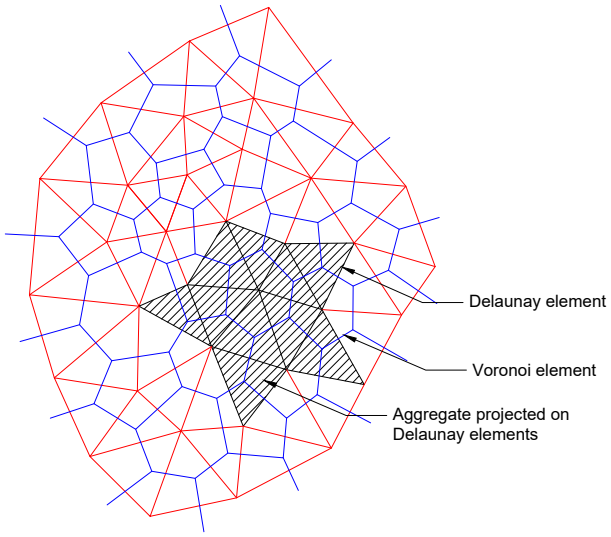


Figure 2.15: Aggregate projected on Delaunay elements

Modelling the structural properties of the aggregates will be comparatively easier in this case as Delaunay elements lie at the boundary of the aggregate. There won't be discontinuities in structural properties within an element. But in this model, as the boundary of the aggregate is projected on Delaunay elements, the Voronoi elements pass perpendicularly through the boundary Delaunay elements. ITZ around the aggregates accelerates the transport along the direction of the aggregate boundary [26] but Voronoi elements pass perpendicularly through the boundary which can undermine the ITZ effect around the aggregates. Also, there will be an abrupt change in the transport properties of the boundary Voronoi elements at the location where the boundary Delaunay element passes through it. the transport behaviour of the aggregates is not effectively modelled in this approach. This approach also requires a fine mesh so that when the aggregates are projected on the Delaunay elements, there is not a significant change in the shape and volume of the original aggregate.

Projecting the aggregates on Voronoi elements
In this case, the shape of the aggregates is projected on Voronoi elements as shown in Figure2.16. In this case also, the shape of the aggregates is modified once they are projected on Voronoi elements.

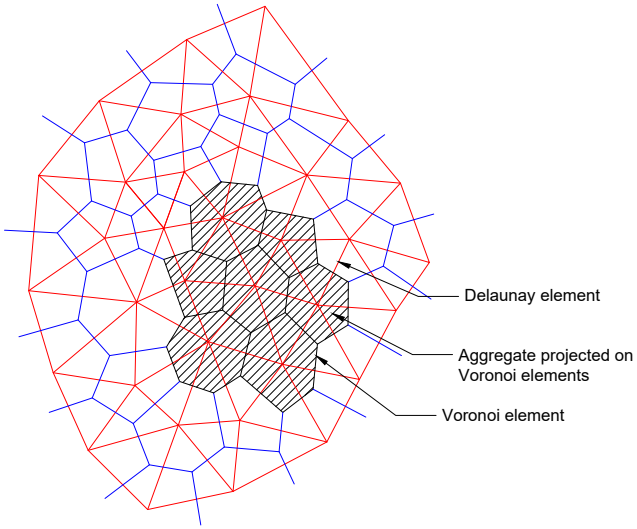


Figure 2.16: Aggregate projected on Voronoi elements

Voronoi elements are present in such a way that either a Voronoi element represent the body of the aggregate, boundary of the aggregate or the cement paste. It is not the case that a single Voronoi element passes through more than one phase of concrete and hence there won't be any discontinuity in the transport properties within any single Voronoi element. There are Voronoi elements present along the boundary of the aggregate which will effectively be able to simulate the ITZ effect of accelerating the moisture transport around the aggregates. It can also be seen that, now Delaunay elements pass perpendicularly through the boundary Voronoi elements which implies that there will be discontinuity in structural properties of the boundary Delaunay element at the location where the boundary Voronoi elements pass through. In this case also, the mesh has to be refined enough so that once the aggregate is projected on Voronoi elements, there is not a significant change in the shape and volume of the original aggregate.

The approaches described above can be used to project the aggregates in the discretized mesh. As in this research, the main focus is on the transport behaviour of concrete, projecting the aggregates on Voronoi elements will be the most effective way. In the numerical modelling of moisture in heterogeneous concrete, spherical aggregates will be randomly generated and projected on Voronoi elements.

3

Moisture Transport in Uncracked Concrete

Modelling moisture transport in uncracked homogeneous concrete under unsaturated conditions serves as a basis to study the transport behaviour of concrete. This chapter mainly focuses on developing the numerical model that is able to simulate moisture transport in uncracked homogeneous concrete under unsaturated conditions. The numerical model serves as the foundation to implement complex features in concrete such as cracks and heterogeneities and simulate moisture transport in them which is presented in further chapters. In this section, the procedure implemented to discretize the domain in the numerical model along with details of the numerical solver used to solve the discretized equation are presented. In order to validate the numerical model, the results of the numerical model are compared with analytical and experimental results. An error analysis between the approximate and the exact form of the numerical solver is presented followed by the conclusions based on the results of the analyses.

3.1. Model Discretization

The first step in simulating moisture transport in concrete is to discretize the model using finite elements and using them as transport elements. In order to implement the governing equation for capillary absorption in the numerical model, it has to be discretized in space and time. This section presents the discretized form of the transport equation and the approximations adopted in the numerical model to solve the discrete equation.

3.1.1. Discretized Equation

As already derived in Chapter 2, the strong form of the equation describing the phenomenon of capillary absorption is described through Richard's equation as given in eq.(2.12) as:

$$\frac{\partial \theta}{\partial t} = \nabla(D(\theta) \cdot \nabla\theta) \tag{3.1}$$

D is the diffusivity of the material (units L^2T^{-1}) and θ is the saturation level. Considering the boundary and initial conditions as given in eqs.(3.2a)-(3.2c), the strong form of the equation can be transformed into it's weak form using the Galerkin weighted formulation.

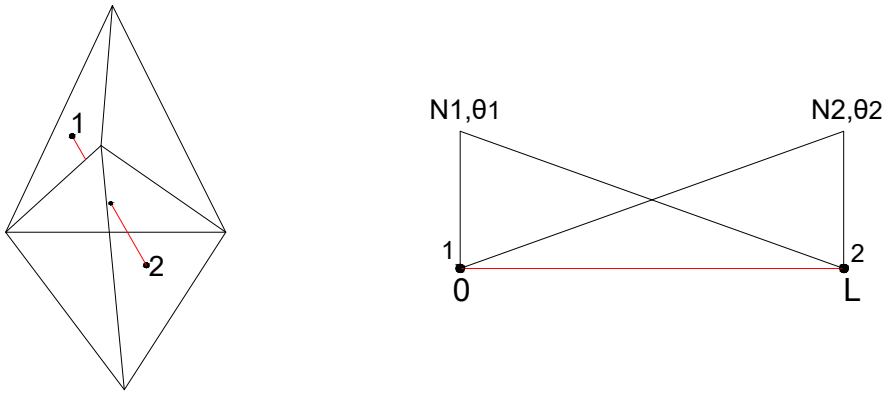
$$\theta = 1 \;\; on \;\; \Gamma_1 \tag{3.2a}$$

$$\frac{\partial \theta}{\partial n} = 0 \;\; on \;\; \Gamma_2 \tag{3.2b}$$

$$\theta(t=0) = \theta_0 \;\; in \;\; \Omega \tag{3.2c}$$

Weak Form

The strong form of the equation is multiplied with virtual saturation $\delta\theta_e$ and integrated over the entire domain. Applying the initial and the boundary conditions and solving the integral using integration by



(a) Voronoi element (1-2) (b) Shape functions

Figure 3.1: Spatial discretization using shape functions

parts, the weak form of the governing equation is obtained:

$$\int_{\Omega} \delta\theta_e \frac{\partial \theta_e}{\partial t} d\Omega = \delta\theta_e \cdot D \cdot \nabla(a)\Big|_{\Gamma_b} - \int_{\Omega} \nabla(\delta\theta_e) \cdot D \cdot \nabla(\theta_e) d\Omega \tag{3.3}$$

Spatial discretization

The continuous saturation field is discretized in space using two noded lattice elements in which the saturation is considered to vary linearly along the length of the element as shown in Figure3.1. Applying the shape functions, the saturation field is discretized as:

$$\theta_e = \mathbf{N}\boldsymbol{\theta} \tag{3.4a}$$

where:

$$\mathbf{N} = \begin{bmatrix} N_1 & N_2 & .. & .. & N_n \end{bmatrix} \quad and \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ .. \\ .. \\ \theta_n \end{bmatrix} \tag{3.4b}$$

Substituting the discretized form of the saturation field into the weak form and readjusting the terms leads to the semi-discrete equation:

$$\int_{\Omega} \mathbf{N}^{\mathbf{T}}\mathbf{N}\dot{\boldsymbol{\theta}} d\Omega + \int_{\Omega} \mathbf{B}^{\mathbf{T}} D \mathbf{B}\boldsymbol{\theta} d\Omega = \mathbf{N}^{\mathbf{T}} D \mathbf{B}\boldsymbol{\theta}\Big|_{\Gamma_b} \quad (where \quad \mathbf{B} = \nabla(\mathbf{N})) \tag{3.5}$$

Eq.(3.5) can also be written in a compact form of eq.(3.6a) where $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the diffusivity matrix and $\mathbf{f}$ is the external flux vector. A_e is the cross-sectional area of the Voronoi elements and l_e is the length of the Voronoi elements. This equation is discretized in space but it still needs to be discretized in time to get the fully discrete form of the governing equation.

$$\mathbf{M}\dot{\boldsymbol{\theta}} + \mathbf{K}\boldsymbol{\theta} = f \tag{3.6a}$$

where:

$$\mathbf{M} = \int_{\Omega} \mathbf{N}^{\mathbf{T}}\mathbf{N} d\Omega = \int_{L} A_e \mathbf{N}^{\mathbf{T}}\mathbf{N} dx \; = \frac{A_e l_e}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{3.6b}$$

$$\mathbf{K} = \int_{\Omega} \mathbf{B}^{\mathbf{T}} D \mathbf{B} d\Omega = \int_{L} D A_e \mathbf{B}^{\mathbf{T}}\mathbf{B} dx \; = \frac{A_e}{l_e^2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \int_{L} D dx \tag{3.6c}$$

$$f = \mathbf{N}^{\mathbf{T}} D \mathbf{B}\boldsymbol{\theta}\Big|_{\Gamma_b} \tag{3.6d}$$

Time discretization
Time is treated as a degree of freedom and is incremented in finite steps. Time discretization is done using the Crank-Nicolson procedure which is a semi-implicit scheme and considered to be unconditionally stable. At time t_{n+1}, eq.(3.7) holds for the semi-discrete form of the transport equation.

$$\mathbf{M}\dot{\boldsymbol{\theta}}_{\mathbf{n+1}} + \mathbf{K}_{\mathbf{n+1}}\boldsymbol{\theta}_{\mathbf{n+1}} = f_{n+1} \tag{3.7}$$

As per the Crank-Nicolson procedure, the moisture content at time t_{n+1} is given by:

$$\boldsymbol{\theta}_{\mathbf{n+1}} = \boldsymbol{\theta}_{\mathbf{n}} + \Delta t\left(\frac{1}{2}\dot{\boldsymbol{\theta}}_{\mathbf{n}} + \frac{1}{2}\dot{\boldsymbol{\theta}}_{\mathbf{n+1}}\right) \tag{3.8}$$

Applying eq.(3.8) to discretize time and substituting the time derivatives in eq.(3.7) leads to the fully discretized form of the governing equation:

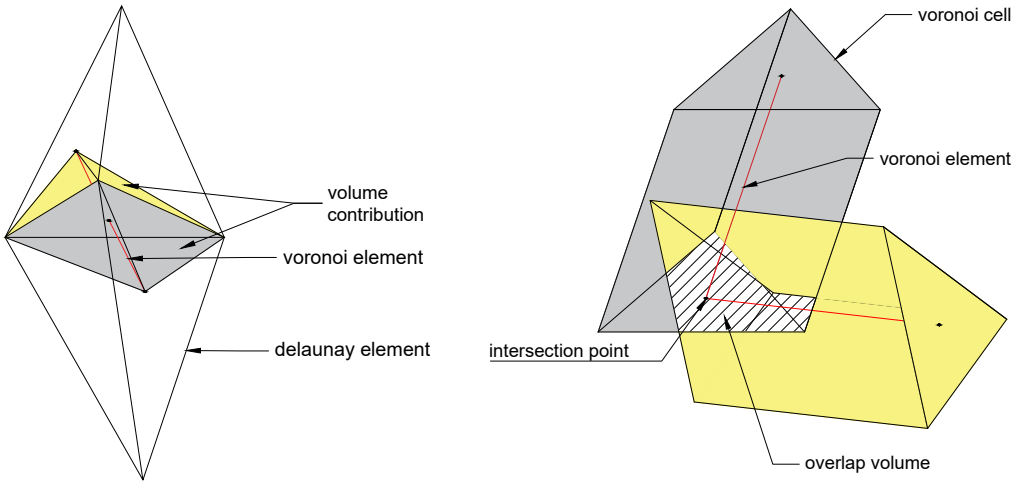
$$\left(\mathbf{M} + \frac{1}{2}\Delta t\mathbf{K}_{\mathbf{n+1}}\right)\boldsymbol{\theta}_{\mathbf{n+1}} = \left(\mathbf{M} - \frac{1}{2}\Delta t\mathbf{K}_{\mathbf{n}}\right)\boldsymbol{\theta}_{\mathbf{n}} + \Delta t f \tag{3.9}$$

Eq.(3.9) is the fully discretized form of the governing equation which is solved numerically in the model to simulate the moisture transport in concrete.

3.1.2. Approximations

After obtaining the discretized form of the governing equation, a numerical solver has to be constructed that can solve the discretized equation and simulate moisture transport in the discretized domain. In order to solve the discretized equation as given by eq.(3.9), some approximations are made in the numerical model which help to increase the computation efficiency of the solver, given that the error induced is within reasonable limits. This section only gives an introduction of the approximations used in the numerical model and the flow in uncracked concrete is simulated with the approximate model so that it can be verified through the experimental results. After doing that, the approximate model will be compared with the exact model with a detailed error analysis in a later section of this chapter.

Volume approximation
The domain is discretized using the dual mesh where each Voronoi element passes through a common Delaunay triangle and the corresponding Voronoi nodes lie within two adjacent delaunay tetrahedra as shown in Figure3.1a. Each Voronoi element is representative of a part of the volume of the Delaunay tetrahedra which is given by smaller tetrahedra inside the Delaunay tetrahedra as shown in Figure 3.2a. These smaller tetrahedra are formed by joining the Voronoi nodes with the three Delaunay nodes that form the common triangle.



(a) Exact volume (b) Approximated volume

Figure 3.2: Volume of the Voronoi elements

Instead of taking the exact volume of the Voronoi element as shown in Figure 3.2a, the volume is approximated by considering that the area of the common delaunay triangle extends throughout the length of the Voronoi element [31] as shown in Figure 3.2b. It can be seen from the figure that there is an overlap of volume of two corresponding Voronoi elements. To account for this, a correction factor ω is included in the mass matrix:

$$\mathbf{M} = \frac{A_e l_e}{6\omega}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{3.10a}$$

where:

$$\omega = \frac{\Sigma A_e l_e}{V} \tag{3.10b}$$

Using the exact volume is comparatively complex to include in the numerical model as the volume of each sub-tetrahedra will be required to be calculated. Instead, the approximated volume is used for the analysis in the numerical model.

Linearization of the time stepping procedure

From eq.(3.9), it can be observed that in the numerical model, to compute the saturation $\boldsymbol{\theta}_{\mathbf{n+1}}$ at the end of a single time increment Δt (within time interval $t_n - t_{n+1}$), the diffusivity matrix $\mathbf{K}_{\mathbf{n+1}}$ at the end of that time interval is also required i.e. the time stepping scheme is implicit. So, for each time increment, there are two unknowns that have to solved for i.e. $\boldsymbol{\theta}_{\mathbf{n+1}}$ and $\mathbf{K}_{\mathbf{n+1}}$. An iterative procedure can be set up till the left hand side of eq.(3.9) converges to the right hand side within a specified tolerance.

As is evident from the procedure described above, iterations will have to be carried out within each time step till the solution converges, which increases the computation time of the solver. Instead, a linear approximation can be used to solve eq.(3.9) which circumvents the need of the iterative procedure as done in [25]. Instead of using the diffusivity matrix $\mathbf{K}_{\mathbf{n+1}}$ at the end of the time interval, the diffusivity matrix $\mathbf{K}_{\mathbf{n}}$ at the beginning of the time interval can be used that leads to the discretized equation given by:

$$\left(\mathbf{M} + \frac{1}{2}\Delta t \mathbf{K}_{\mathbf{n}}\right)\boldsymbol{\theta}_{\mathbf{n+1}} = \left(\mathbf{M} - \frac{1}{2}\Delta t \mathbf{K}_{\mathbf{n}}\right)\boldsymbol{\theta}_{\mathbf{n}} + \Delta t f \tag{3.11}$$

Eq.(3.11) presents an explicit form of the time stepping scheme which does not depend on the information at time t_{n+1}. The saturation $\boldsymbol{\theta}_{\mathbf{n+1}}$ is dependent on the diffusivity $\mathbf{K}_{\mathbf{n}}$ because of which, it can be solved linearly.

Using an explicit time stepping scheme has consequences on the stability and the accuracy of the results which are analyzed later in the chapter. Crank-Nicolson procedure, used for time discretization of the semi-discrete form of the governing equation, is considered to be unconditionally stable. When

the linear approximation is done, to convert the implicit scheme to explicit, there is a loss of unconditional stability of the scheme. In each time step, an error is induced in the solution, the magnitude of which depends on the magnitude of the time step. This error can accumulate with the accession of the time steps and the solution can diverge rapidly from any realistic results causing instability. Hence, the magnitude of the time step should be sufficiently small so that the error induced does not cause instability.

Uniform mean diffusivity

The diffusivity matrix $\mathbf{K}$ requires the computation of the integral of diffusivity along the length of the element as given by:

$$\mathbf{K} = \frac{A_e}{l_e^2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\int_L D\,dx \tag{3.12}$$

The diffusivity varies exponentially with the saturation level [48]. Using the spatial discretization, the saturation level in the domain is discretized and varies linearly along the element length. Extending the discretization to diffusivity, it varies exponentially over the element length as:

$$D = D_c e^{n_c x} \tag{3.13a}$$

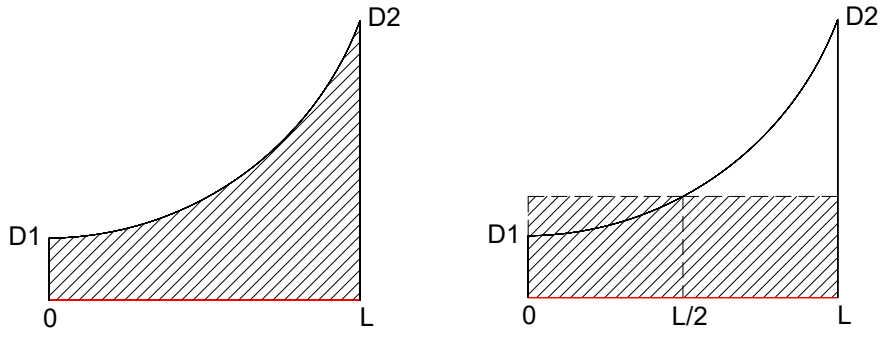
where:

$$D_c = D_0 e^{n\theta_1} \tag{3.13b}$$

$$n_c = \frac{n}{L}(\theta_2 - \theta_1) \tag{3.13c}$$

The integral of diffusivity over the length of the element is evaluated precisely if the total area under the curve is considered as in Figure 3.3a. To numerically calculate this area, four point Gauss-integration can be applied. Considering the exponential variation of diffusivity over the element length, the analytical expression for the diffusivity matrix is given by:

$$\mathbf{K} = \frac{A_e D_0 e^{n\theta_1}(e^{n(\theta_2-\theta_1)} - 1)}{n l_e(\theta_2 - \theta_1)}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{3.14}$$



(a) Exact diffusivity (b) Uniform mean diffusivity

Figure 3.3: Variation of diffusivity over the element

But, instead of using the exact distribution of the diffusivity to compute the integral, the diffusivity is considered to be uniform throughout the length of the element and equal to the diffusivity at $0.5L$ as shown in Figure3.3b. The integral is computed using the uniform mean diffusivity along the length of the element and the corresponding analytical expression for the diffusivity matrix is given by eq.(3.15). For the analysis carried out for uncracked concrete, the uniform distribution of the diffusivity is considered instead of the exact distribution to solve the discretized transport equation.

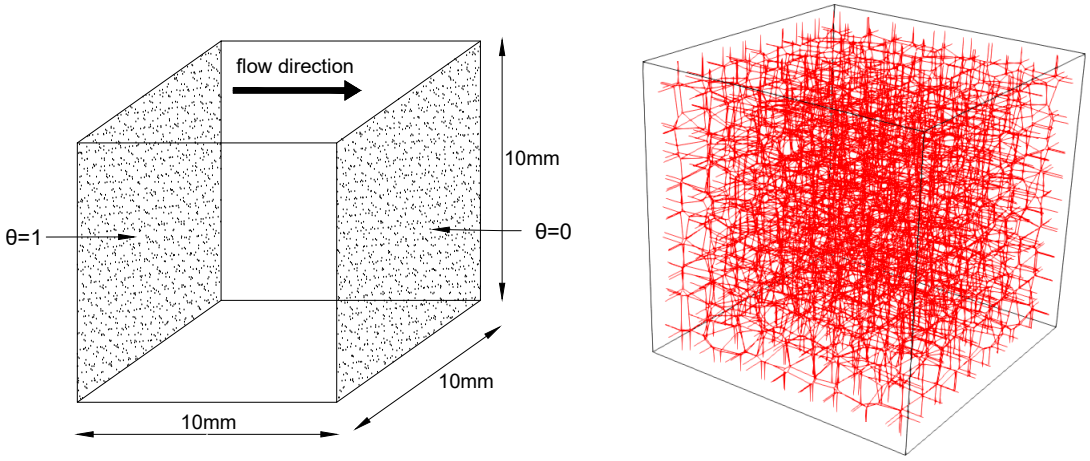
$$\mathbf{K} = \frac{A_e D_0 e^{0.5n(\theta_1+\theta_2)}}{l_e}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{3.15}$$

3.2. Constant Diffusivity

Once the transport equations are discretized, they can be numerically solved to obtain the moisture distribution in concrete. In the following section, the moisture transport in uncracked concrete for the two cases of steady-state and non-stationary flow will be modelled considering constant diffusivity. In reality, the diffusivity of the material for the process of capillary absorption is not constant but varies with the saturation level of the pores which changes with time [48]. Making the diffusivity constant is not representative of the actual physical phenomenon which is more accurately modelled through time-dependent diffusivity. Regardless, making the diffusivity constant serves to validate the numerical model by comparing it with the analytical results so that the model is further used to simulate non-stationary flow with variable diffusivity.

3.2.1. Steady-State flow

The steady-state is reached when the flow in the material is independent of change in time. Steady-state flow is simulated on a cubic domain of $10 \times 10 \times 10\ mm^3$. As shown in Figure 3.4a, all the nodes on the left hand side boundary are subjected to constant saturation of $\theta = 1$ while all the nodes on the right hand side boundary are subjected to constant saturation of $\theta = 0$ which directs the flow from left to right. The other four boundary faces are considered sealed without any interaction with the surrounding by considering their boundary flux to be zero. All the nodes lying within the domain have an initial saturation of 0 and the diffusivity remains constant with time and is assumed to be $D = 1\ mm^2/s$.



(a) Schematic of the flow conditions in the domain (b) Domain discretization using Voronoi elements

Figure 3.4: Cubic domain under steady-state flow

The cubic domain as shown in Figure 3.4a, is discretized using the Voronoi mesh in which the transport elements are modelled on the Voronoi lattice elements. Table 3.1 shows the Voronoi mesh properties used to discretize the domain.

Table 3.1: Mesh properties

Discretization	Number of nodes	Number of elements	Average element length (mm)
Voronoi	5857	10256	0.4534

As the steady-state flow is being analyzed, the time dependent terms in eq.(3.9) do not contribute to the transport of moisture and the discretized equation reduces down to:

$$\mathbf{M}\boldsymbol{\theta}_{\mathbf{n+1}} = \mathbf{M}\boldsymbol{\theta}_{\mathbf{n}} \tag{3.16}$$

Simulating the transport of moisture using eq.(3.16), the variation of saturation with the x-coordinate for Voronoi discretization can be seen in Figure 3.5. The plot shows that the discretization provides accurate steady-state results. The accuracy of the modelling approach is assessed by comparing the error norms to the exact solution.

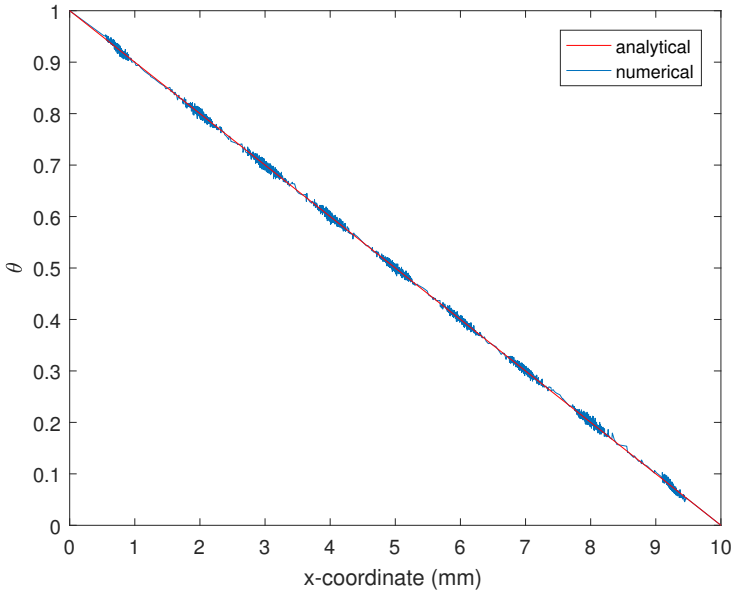


Figure 3.5: Steady-state analysis in Voronoi discretization

Two error norms are considered here:

$$||r||_\infty = max|r_n|, n = 1..N \tag{3.17a}$$

$$||r||_2 = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N}|r_n|^2\right)} \tag{3.17b}$$

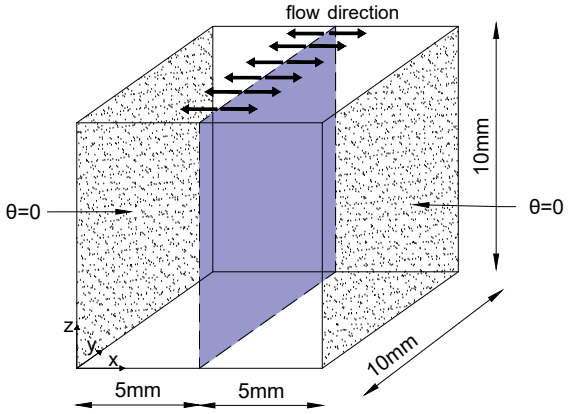
where r_n is the difference between theoretical and numerical moisture content. The error between the analytical and the numerical solution for the two error norms is presented in Table 3.2. The error computed for both the norms is within reasonable limit and it can be concluded that the numerical model simulates the steady-state flow successfully.

Table 3.2: Error

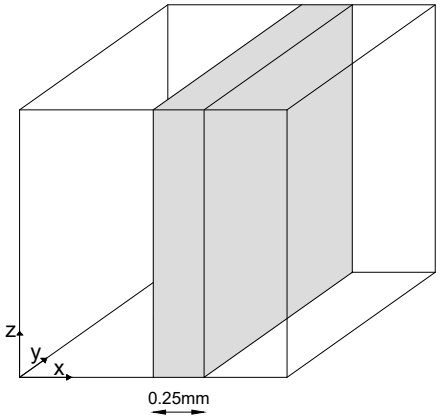
Discretization	$\|\|r\|\|_\infty$	$\|\|r\|\|_2$
Voronoi	0.0153	0.0043

3.2.2. Non-Stationary Flow

Now that the model is able to simulate moisture transport under steady-state conditions, the moisture transport under non-stationary conditions will be simulated in this subsection. In non-stationary flow analysis, the flow of moisture in concrete varies with time. A model, as presented in [31], is used to simulate non-stationary flow with constant diffusion. The discretized form of the governing equation as presented in eq.(3.11) is used to simulate the flow. The flow is simulated on a cubic domain of $10 \times 10 \times 10\ mm^3$. As shown in Figure 3.6a, the right and left faces of the domain are maintained at zero saturation and the rest of the faces of the cube are considered sealed with no interaction with the surroundings.



(a) Cubic domain flow conditions



(b) Division of the domain

Figure 3.6: Cubic domain under non-stationary flow

The initial saturation of the nodes within the domain is given as:

$$\theta_i = \theta_0 sin\left(\frac{\pi x}{L}\right) \tag{3.18}$$

The distribution of the initial saturation level in the domain is such that the saturation of the nodes lying in the centre of the domain is maximum and the saturation of the nodes lying at the faces of $x = 0\ mm$ and $x = 10\ mm$ is zero. Because of this, the flow of moisture is from the centre towards the boundary surfaces (Figure 3.6a). The results of the numerical model to simulate the non-stationary transport can be verified with the analytical solution given in [4] as:

$$\theta(x,t) = \theta_0 sin\left(\frac{\pi x}{L}\right) e^{\left(-\frac{\pi^2}{L^2}t\right)} \tag{3.19}$$

The cubic domain, as shown in Figure 3.6a, is discretized using the Voronoi mesh and Voronoi elements are used to simulate the moisture transport. Table 3.3 shows the mesh properties used to discretize the domain. The diffusivity of the material is kept constant at $D = 1\ mm^2/s$ and the non-stationary flow is simulated on the discretized mesh.

Table 3.3: Mesh properties

Discretization	Number of nodes	Number of elements	Average element size (mm)
Voronoi	5881	10298	0.4534

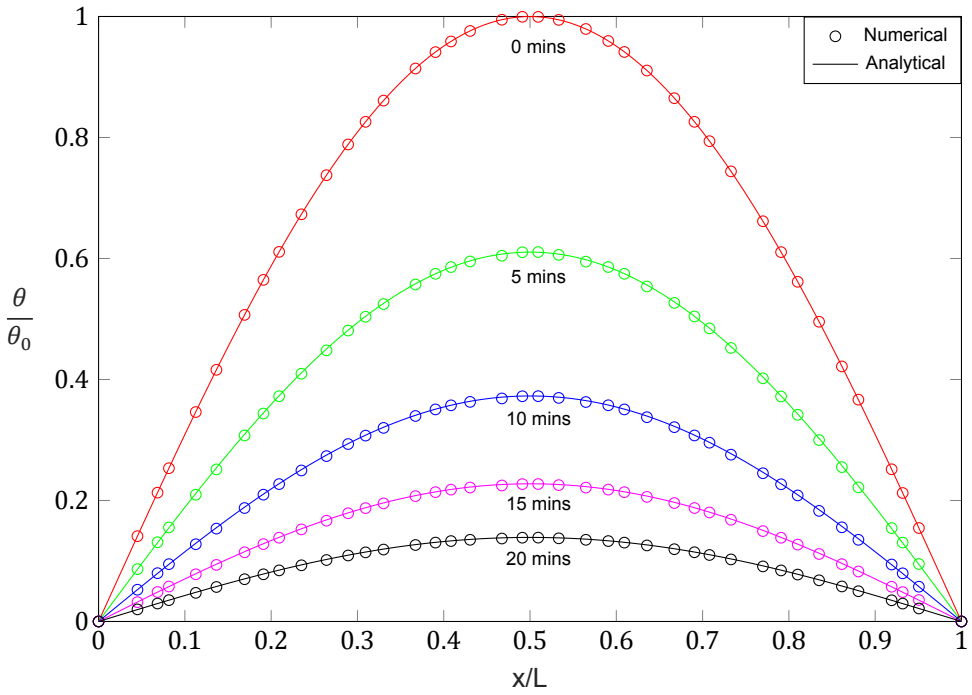


Figure 3.7: Non-stationary transport analysis in medium mesh

The total volume of the domain is divided into smaller representative volumes along the x-axis with dimensions $0.25 \times 10 \times 10\ mm^3$. One such representative volume is shown in Figure 3.6b. Each such volume is representative of the average saturation of all the nodes lying within that particular volume. Figure 3.7 presents the results of the non-stationary flow as simulated by the numerical model where the average saturation is plotted against the mean x-coordinate (for each representative volume).

It can be seen that the results from the numerical simulation conform well with the analytical solution. As the time progresses the saturation of the domain decreases as the flow is directed from the centre to the boundary faces which are kept at a constant zero saturation.

3.3. Time Dependent Diffusivity

In the previous section, the model has been verified for the case of constant diffusivity. In reality, the diffusivity of the material varies non-linearly with time as the diffusion coefficient depends on the saturation level. In this section, the numerical model is presented where the diffusivity is varying with time and the numerical results are compared with the experimental observations. The approximations, considered before, in constructing the numerical model will be validated by comparing their numerical results with those of the exact models.

3.3.1. Experiment

The experiment performed in [56] is used as a reference to numerically model moisture transport in sound concrete. In the experiment, moisture transport in an uncracked concrete block of dimensions $100 \times 50 \times 20\ mm^3$ was observed as shown in Figure3.8. A water-cement ratio of 0.6 and a concrete mix proportion of 1:3.3:0.6 (cement:sand:water) by volume was used [48].

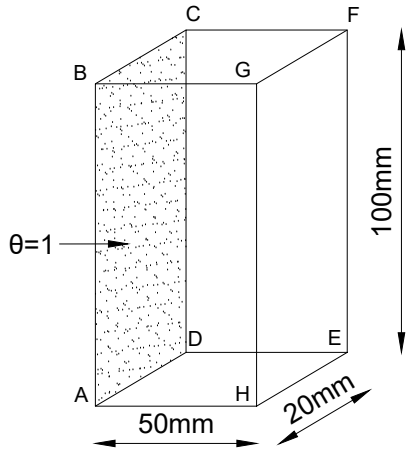


Figure 3.8: Schematic diagram of the uncracked concrete block

After curing the specimens for 14 days, they were dried by keeping inside an oven for 4 days at a temperature of $50°C$ until the specimens attained a constant weight. In order to get unidirectional flow of water, except for faces ABCD and EFGH in Figure3.8, all other faces were sealed by covering them with self-adhesive aluminium foils. Face ABCD was submerged in water and the moisture distribution in the sample over a period of 4 hours was observed using neutron radiography technology.
The diffusivity of the sound mortar is required to numerically model the moisture transport. According to [21], based on the model of the relationship between the capillary coefficient k and sorptivity S, the diffusivity of dry concrete D_0 can be approximated as:

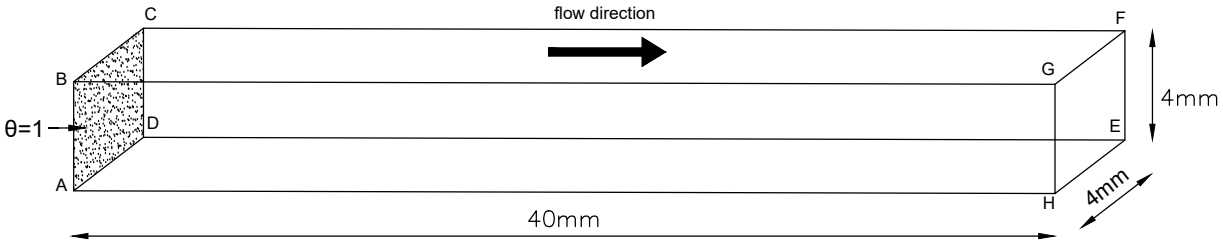
$$D_0 = \frac{(k/1.294)^2}{123.131} \tag{3.20}$$

For the first 4 hours, the capillary coefficient of the sound mortar mix as measured experimentally [56] is $k = 10.714\ mm/h^{1/2}$ which leads to the diffusivity of dry concrete as $D_0 = 1.54676 \times 10^{-4}\ mm^2/s$. The moisture distribution is analyzed in the concrete block over the height of $50\ mm$ at different time intervals.The numerical model uses the input parameters from the experiment.

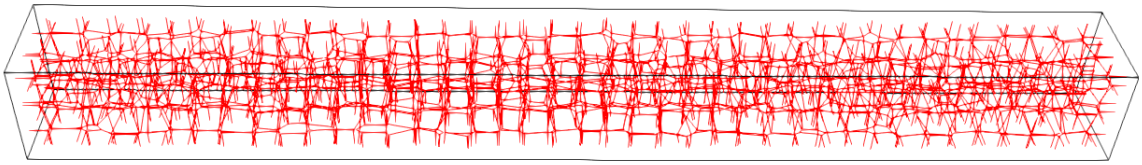
3.3.2. Numerical Analysis

A numerical model is used to simulate the moisture transport in the uncracked concrete. In the experiment, the moisture transport is observed along the length of $50mm$ and the flow from the rest of the faces is restricted. In the numerical model, a domain of $40 \times 4 \times 4\ mm^3$ is used to simulate the moisture transport as shown in Figure3.9 and the flow is observed along the length of $40mm$. Firstly, for the time intervals at which the flow is observed, it doesn't reach the length of $40mm$ and secondly, as the rest of the faces are considered sealed, the same flow in the experiment can be modelled numerically using a

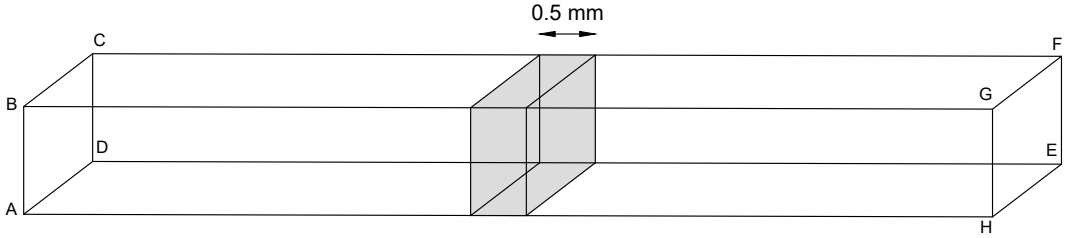
much smaller cross-section. Because of these reasons, the domain used in the numerical model can be used to represent the flow as performed in the experiment. Doing this, a finer mesh can be used to get accurate results and also the computation time is saved considerably.



(a) Schematic of the flow conditions in the domain



(b) Voronoi Discretization of the domain



(c) Schematic of a representative volume used to average the results

Figure 3.9: Domain used in the Numerical Model

As shown in Figure3.9a, the face ABCD represents the surface that is submerged in water and forms the boundary at which all the nodes have a constant saturation of 1. Flow is directed from face ABCD to EFGH along the Voronoi elements as shown in Figure3.9b. The initial saturation of all the nodes in the domain is considered as zero and concrete is assumed to be completely dry. This is not representative of the actual state of concrete in the experiment but as the results in the experiment are presented in the form of increment of the moisture content from the initial state, using this as the initial condition is justified. Applying this boundary condition and using the flow parameters as shown in Table3.4, the numerical analysis is carried out. The numerical model used approximate volume, explicit time stepping technique and uniform mean diffusivity of the the elements to simulate moisture transport in concrete.

Table 3.4: Flow parameters for moisture transport in sound concrete

Parameter	**Symbol**	**Value**
Diffusivity of dry concrete	D_0	$1.5467 \times 10^{-4}\ mm^2/s$
Empirical constant	n	6
Initial saturation	θ_0	0
Porosity	p	0.0462
Time step	Δ t	$6s$

Mesh sensitivity analysis
Using the flow parameters, boundary and initial conditions as mentioned above, the analyses is carried out using four different meshes, the details of which are given in Table3.5.

Table 3.5: Mesh properties

Mesh	Number of nodes	Number of elements	Average element length (mm)
Mesh1	62028	112878	0.1757
Mesh2	30859	54786	0.2231
Mesh3	3333	5208	0.4792
Mesh4	280	329	1.1306

Flow is simulated using the above 4 meshes and the results at 60 minutes is observed and compared with the experimental results as shown in Figure3.10. As it can be seen that the results are quite similar for meshes 1 to 3 but for mesh 4, the results deviate from the experiment results. That is simply due to the fact that mesh 4 is the most coarse mesh and the element size in mesh 4 is comparatively larger to accurately model the flow results. But for the rest of the meshes, the results agree well with the experimental results. Similar trend is observed if the flow is observed at different time intervals. Mesh 3 is used for further analysis because the number of nodes and elements are fewer than that of mesh 1 and mesh 2 which decreases the computation time of the analysis without compromising the accuracy of the results.

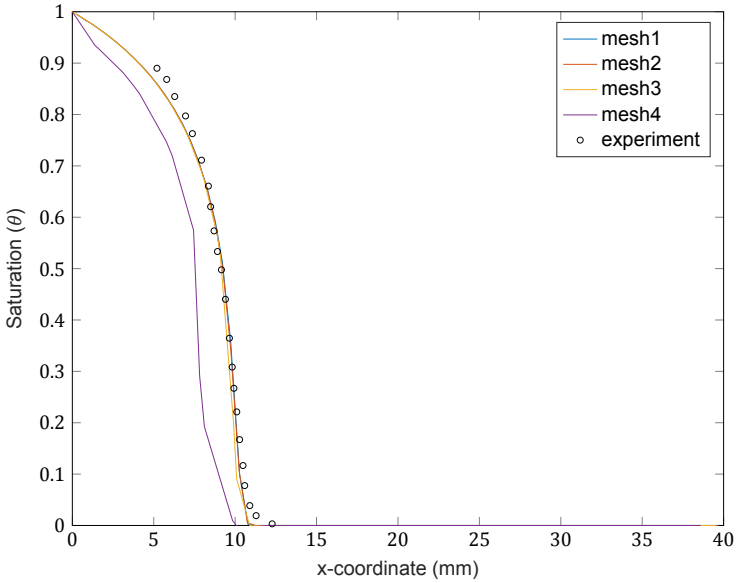


Figure 3.10: Flow results at 60 mins

The results of the numerical model for mesh 3 are shown in Figure3.11. The entire domain is divided into small representative volumes along the x-axis as shown in Figure3.9c and for each volume, the saturation and the x-coordinate are averaged.

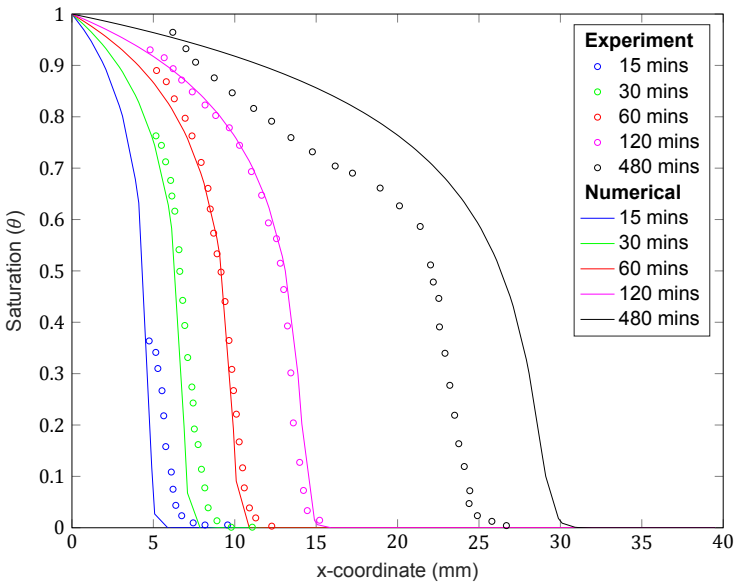
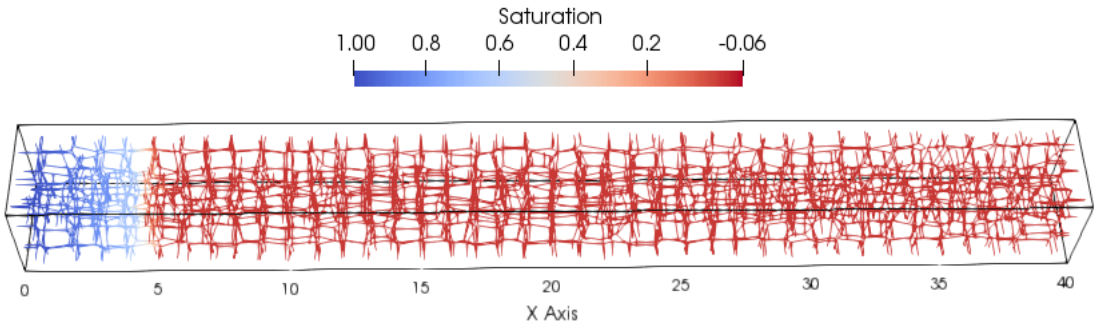


Figure 3.11: Moisture transport in uncracked concrete

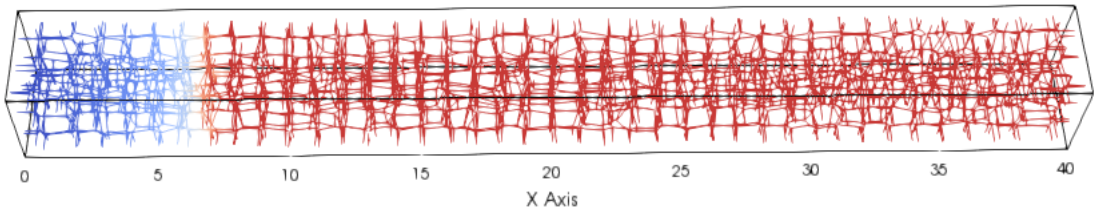
It can be seen in Figure3.11 that, at 15 minutes, the numerical results are under-predicting the flow compared to the experiment. The numerical results for 30, 60 and 120 minutes conform well with the experimental results. The results for 480 minutes are significantly over-predicted by the numerical model. This is because of the way the diffusivity of dry concrete is computed in the numerical model [48]. As explained in the previous subsection, the diffusivity of the dry concrete is based on the relationship between the capillary coefficient and sorptivity of concrete which is given as [53]:

$$S = k\phi \tag{3.21}$$

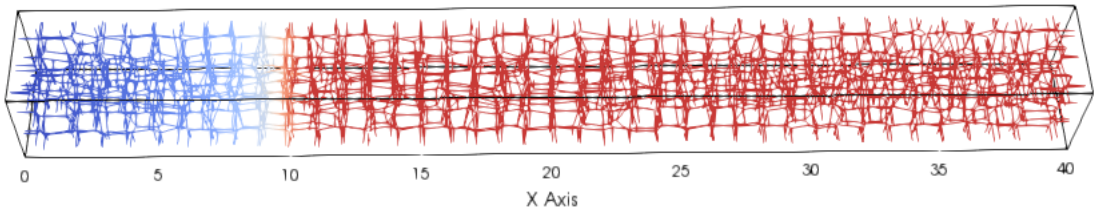
where S is sorptivity, k is capillary coefficient and ϕ is the porosity of concrete. The porosity of concrete is considered uniform throughout the domain in the numerical model even after considering the effect of aggregates. But in reality, concrete is heterogeneous in nature and the porosity of concrete is different in different parts of the domain. With the advancement of flow, more part of the domain is involved in the flow which changes the overall porosity of concrete with time. This change of porosity is not accounted for in the numerical model and instead, a constant uniform porosity throughout the domain is considered which causes a deviation of results between the experimental and the numerical model. Also, in the experiment, the shape of the curve at 480 minutes is different than the shape at the preceding time steps which is due to the aggregates included in the concrete mix [14]. In the numerical model, the concrete is assumed to be homogeneous and because of this, the shape of the curve for the numerical model at 480 minutes is similar to the shape at the preceding time intervals.



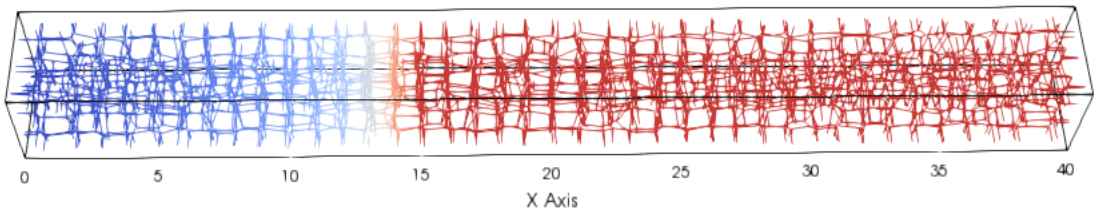
(a) Saturation of the domain at 15 minutes



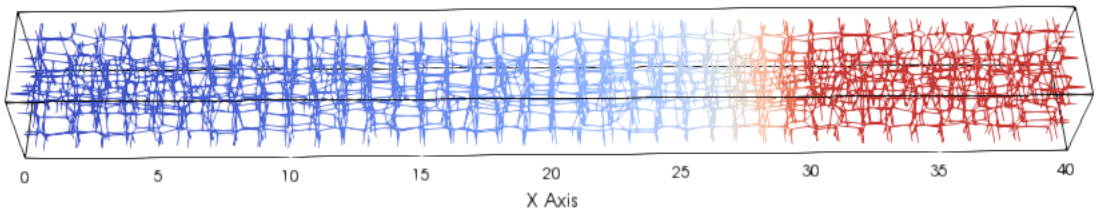
(b) Saturation of the domain at 30 minutes



(c) Saturation of the domain at 60 minutes



(d) Saturation of the domain at 120 minutes



(e) Saturation of the domain at 480 minutes

Figure 3.12: Transition of flow in the discretized domain

Figure 3.12 shows the saturation distribution for mesh 3 in the uncracked domain at different time stages. The flow originates from the right boundary at $x = 0$ and as the flow progresses in the x-direction, more and more number of Voronoi nodes become saturated.

3.3.3. Validity of Approximations

Now that the numerical model using approximate volume, explicit time stepping technique and uniform mean diffusivity of the elements is able to reproduce the experimental results, the underlying approximations, under which the numerical model was constructed, will be verified. The three approximations where the approximated volume instead of the exact volume of the Voronoi elements is used, explicit instead of implicit time stepping technique is used and the uniform mean diffusivity instead of the actual exponential variation of diffusivity over the element length is used will be validated by computing the error between the approximated and the exact models.

The reference numerical model is the one that uses all the above mentioned approximations. In the first part of this subsection, a numerical model with exact volume, explicit time stepping technique and uniform mean diffusivity is compared with a numerical model with approximate volume, explicit time stepping technique and uniform mean diffusivity. In the second part, a numerical model with approximate volume, implicit time stepping technique and exponential variation of diffusivity is compared with a numerical model with approximate volume, explicit time stepping technique and uniform mean variation of diffusivity.

Volume analysis

The volume approximation in which the mean correction factor is used to compute the volumetric capacity of the Voronoi elements as mentioned in section 3.1.2 is analyzed. Two numerical models with explicit time stepping technique and uniform mean variation of diffusivity are constructed, one considering exact volume of the Voronoi elements and the other with the approximated volumes of the Voronoi elements. Analyses are carried out using the exact volume (Figure 3.2a) and compared with the results obtained by using the approximated volume (Figure 3.2b) by calculating the error between the saturation levels for the two analyses at each time step using the L-2 error norm which is given as:

$$||r||_2 = \sqrt{\left(\frac{1}{N}\sum_{n=1}^{N}|r_n|^2\right)} \tag{3.22}$$

where, $||r||_2$ is the error at a particular time step, N is the total number of Voronoi nodes and r_n is the difference of saturation level of a particular node obtained from the exact and the approximate volume models. The analyses are carried out on 4 meshes given in Table 3.5 and the error vs time plot is shown in Figure3.13.

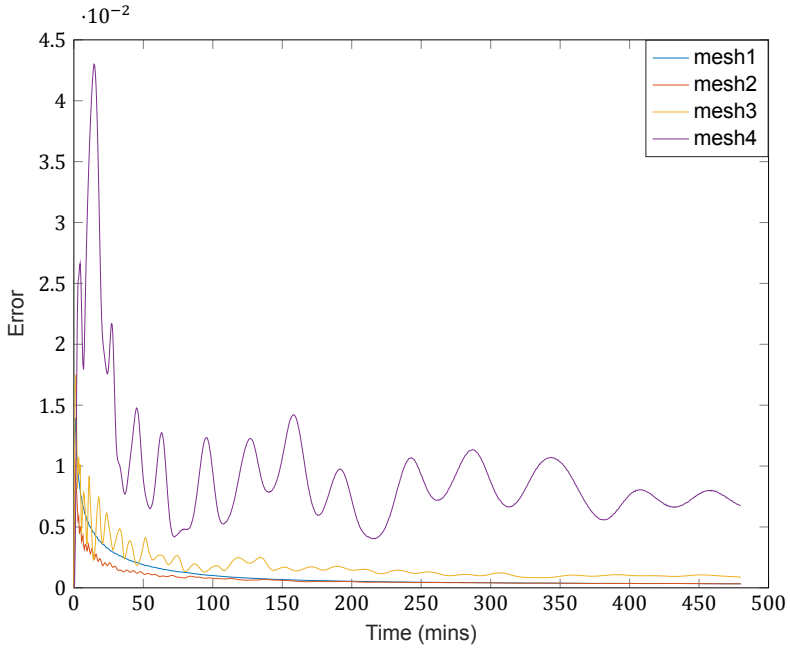


Figure 3.13: Error vs Time graph for Volume approximation

It can be seen in Figure3.13 that for all the meshes, the global behaviour is such that the error is decreasing with time showing stability of the volume approximation considered. Generally, at a given time stage, the magnitude of error is smaller for a finer mesh. But this trend does not hold for meshes 1 and 2. It is due to the fact that the number of elements in mesh 1 are more than the number of elements in mesh 2. When the volumetric capacity of the domain ($\mathbf{M}$ matrix) is computed for all the elements using the approximate (mean correction factor) and the exact approach, the error per element is less for mesh 1 as it is the finer mesh. But in the error-norm, the summation of error for all the nodes is done and on account of greater number of nodes present in mesh 1, the error in mesh 1 exceeds that of mesh 2 in spite of being the finer mesh. Hence, mesh refinement might not lead to more accuracy of results when the approximated model is used. Regardless, the error for the meshes 1,2 and 3 remain below 2 percent showing reasonable accuracy of the results.

The local and global behaviour of the error vs time graph (Figure3.13) can be explained by the transitional behaviour of the flow in the discretized domain. The global behaviour of the graph is such that the error is reducing with time. The approximate volume model uses the average correction factor (ω) on all the individual volumes of the elements, because of which, the volumetric capacity of the individual elements is not accurate but the total volumetric capacity of all the elements combined is the same as the exact volume of the domain. During the initial time stages, the predominant flow in the domain is limited to a comparatively small fraction of elements. When the correction factor is applied to compute the volumetric capacity of this small fraction of elements, it gives a higher error because the volume of a comparatively small fraction of elements is being computed using a correction factor which is computed based on a much bigger data set comprising of all the elements in the domain. As the flow continues, the water front travels in the direction of flow increasing the fraction of elements involved in the predominant flow. As this happens, the error in saturation level decreases as the volumetric capacity of the elements predicted by the approximate and the exact model come closer. Eventually, when the domain becomes completely saturated, all the elements in the domain will be involved and the volumetric capacity calculated from both the exact and approximate model will be the same.

It can be observed in Figure3.13 that the error does not decrease gradually but there are local peaks where the error first increases, attains a local maximum and then decreases. The frequency of these peaks becomes higher and the magnitude of the local maximum becomes lower as the mesh becomes finer. The behaviour of these local peaks is highly mesh-dependent.

Local Behaviour of the Error vs Time graph

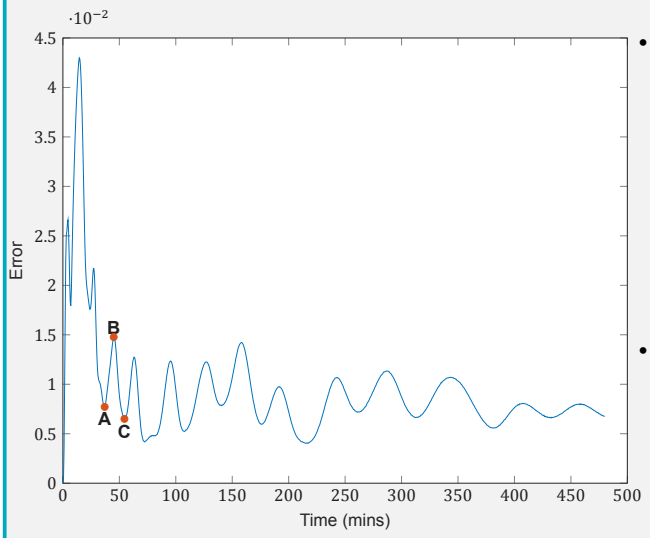


Figure 3.14: Error vs Time graph for Volume approximation of mesh 4

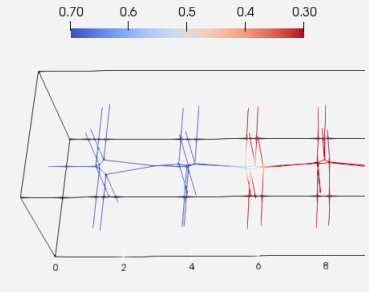


Figure 3.15: Saturation at point A

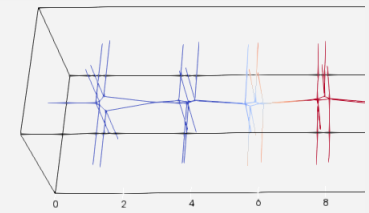


Figure 3.16: Saturation at point B

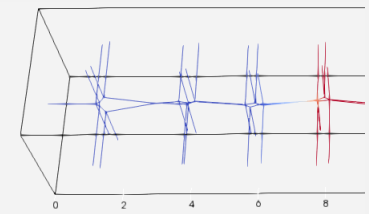


Figure 3.17: Saturation at point C

- The local behaviour of the error vs time graph is explained for mesh 4. In Figure3.14, points **A** (37 minutes), **B** (45 minutes) and **C** (54.5 minutes) are marked that represent the start of the peak, the local maximum of the peak and the end of the peak respectively.

- The flow at point **A** as shown in Figure 3.15 is at a stage when it is just about to involve the Voronoi element (4-6) (composed of node at 4 and node at 6) in the water front. Node 4 is at a saturation level closer to fully saturated condition and node 6 is at a saturation level nearer to the initial condition. After this point, the error between the saturation level of the exact and the approximate volume models increases due to an initial difference of the volumetric capacity causing subsequent difference in the diffusivity of the element.

- Figure3.16, which represents the flow at the time stage where point **B** occurs, the saturation level at the node 6 has increased since before but the Voronoi element (4-6) has not yet fully become a part of the saturated domain. At this stage, the error between the exact and the approximate volume method starts decreasing because of the decrease in the difference of the diffusivity calculated from both the models.

- After the decrease in the error, eventually flow reaches point **C**, where as shown in Figure3.17, both the nodes of the Voronoi element (4-6) are close to complete saturation and the entire element forms a part of the saturated domain. The flow now moves on to the next Voronoi element (6-8) and all the above mechanism is repeated for it as well.

The nature of flow in the domain is such that the moisture is not spread uniformly over the entire domain but it gets accumulated in the nodes that form the water front. After the accumulation of moisture, it then spreads over to the adjacent nodes in the direction of the flow. As the water front moves forward in the direction of the flow, it is continually involving more elements and leaving saturated elements behind. The elements that constitute the water front have one node whose saturation will be close to the fully-saturation condition and the second node will have saturation that will be closer to the initial saturation. Local peaks in the error vs time graph occur when new elements are being involved in the flow. The point at which the peak starts (when the error starts increasing) is the point at which one of the nodes of the water front elements is close to complete saturation and the second node is close to the initial saturation. As these elements undergo flow, the saturation of the second node will gradually increase. But due to a difference of the volumetric capacity of the elements computed from both the models, the saturation level of the second node will be different in both the models. According to the global behaviour, this error in the saturation of the nodes should have decreased. But what happens is that due to a difference of saturation of the second node in both the models, the variation of diffusivity over the element for both the models is different. During the initial stages when these elements start to be involved in the flow, the diffusivity of the element is quite high as one node is close to complete saturation and the other is close to initial saturation. Because of this, even if there is a small difference of the saturation of node 2, the diffusivity of the element for both the models is quite different. Due to this, on further flow, the difference of saturation levels of the second node increases further and the error increases. This is the reason for the local increase of error. At a certain point, as the second node becomes more and more saturated, the difference in the diffusivity decreases. Eventually, there comes a stage when the increase in error because of a difference in diffusivity is negated by the decrease in error that should have been caused by the volume approximation. This is the point of local maximum of the peak. After this point,the decrease in error because of volume approximation becomes the predominant factor. This accounts for the region of the peak when the error is decreasing. At the bottom point of the peak, the element has been completely involved in the flow and the water front goes on to the next element and this whole mechanism is repeated. The frequency of the peaks is higher in a fine mesh because there are more number of elements and smaller elements in the domain. Because of this, the water front involves more elements at a faster rate and there are more peaks. Also, the magnitude of the peaks is small because as the element size is smaller in the finer mesh, the magnitude of the difference of the volumetric capacity and the diffusivity variation will be less and the error in the saturation level will be lower.

Linearization of the time stepping technique and exact diffusivity

As it was mentioned before, the fully-discretized equation representing the flow in uncracked concrete is solved using approximations in which the iterative scheme was circumvented by a linear solver and also instead of the exact diffusivity, the uniform mean diffusivity was used. An implicit time stepping scheme is used to accurately solve the discretized equation. The details of the iterative mechanism involved in the implicit scheme is given in the box below. The tolerance of the scheme is set to 10^{-6} and the L-2 error norm is used to compute the error in the saturation levels of two corresponding time steps. The exact diffusivity is considered by taking the exponential variation of the diffusivity along the element length and the 4-point Gauss integration is used in the numerical integration of diffusivity over the length of the element as shown in Figure3.3a.

Iterative Mechanism used in the Implicit Time Stepping Model

The fully discretized form of the transport equation is given as:

$$\left(\mathbf{M}+\frac{1}{2}\Delta t\mathbf{K_{n+1}}\right)\boldsymbol{\theta}_{\mathbf{n+1}}=\left(\mathbf{M}-\frac{1}{2}\Delta t\mathbf{K_n}\right)\boldsymbol{\theta}_{\mathbf{n}}+\Delta tf \tag{3.23}$$

The external flux is considered to be zero so the last term disappears in the equation. For simplicity, the above equation is rewritten as:

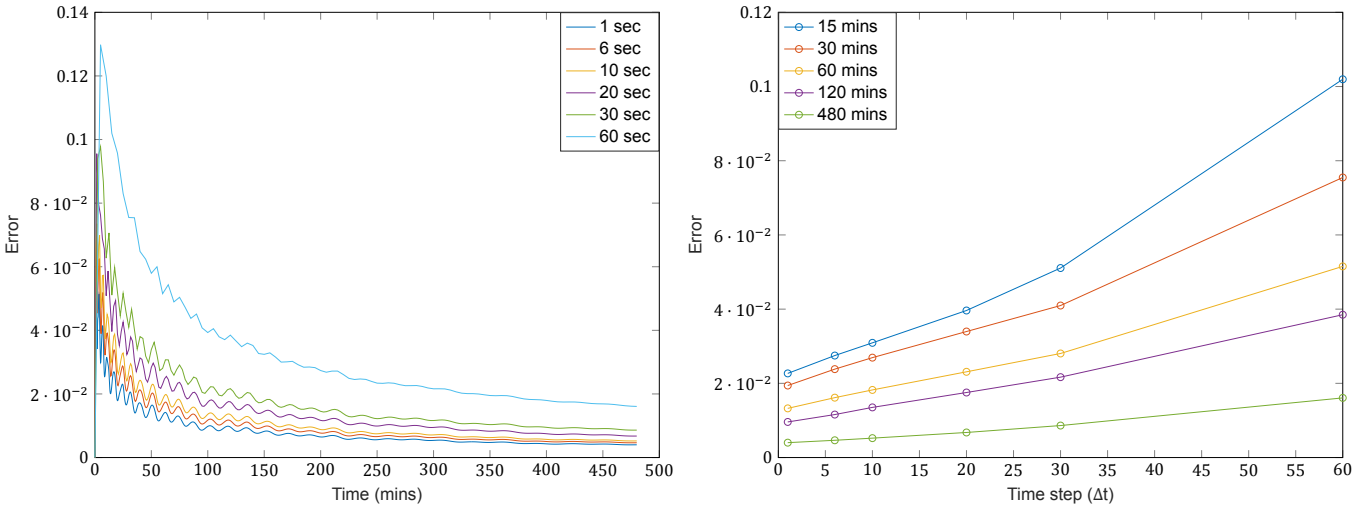
$$\mathbf{LHS_{n+1}}\boldsymbol{\theta}_{\mathbf{n+1}}=\mathbf{RHS_n}\boldsymbol{\theta}_{\mathbf{n}} \tag{3.24}$$

The subscript (n+1) refers to the quantities in the current time step and subscript (n) refers to the quantities in the previous time step. The moisture content at the previous time step ($\boldsymbol{\theta}_{\mathbf{n}}$) is known and this is considered as the starting point of the iterative mechanism as presented below.
For each time step:

1. Compute $\mathbf{RHS_n}$ from $\boldsymbol{\theta}_{\mathbf{n}}$.
2. Assume $\boldsymbol{\theta}_{\mathbf{n+1}}$=$\boldsymbol{\theta}_{\mathbf{n}}$.
3. Compute $\mathbf{LHS_{n+1}}$ from $\boldsymbol{\theta}_{\mathbf{n+1}}$.
4. Compute error using the L-2 error norm: $\mathbf{RHS_n}\boldsymbol{\theta}_{\mathbf{n}}-\mathbf{LHS_{n+1}}\boldsymbol{\theta}_{\mathbf{n+1}}$.
5. If error > tolerance (10^{-6}), go to the next step, Else go to the next time step starting from step 1.
6. Compute $\boldsymbol{\Delta\theta}=\dfrac{\mathbf{RHS_n}\boldsymbol{\theta}_{\mathbf{n}}-\mathbf{LHS_{n+1}}\boldsymbol{\theta}_{\mathbf{n+1}}}{\mathbf{LHS_{n+1}}}$.
7. Compute $\boldsymbol{\theta}_{\mathbf{n+1}}=\boldsymbol{\theta}_{\mathbf{n+1}}+\boldsymbol{\Delta\theta}$ and go to step 3.

The above exact model is used to simulate the moisture transport in uncracked concrete and the results are compared with the approximate model. Time step analysis is done for both the models for 6 different time steps of 1 sec, 6 secs, 10 secs, 20 secs, 30 secs and 60 secs. The error in saturation levels is computed using the L-2 error norm. Figure 3.18a shows the error vs time graph for all the 6 time steps. It is clear form the graph that the error is reducing as the magnitude of the time step is reducing. The analyses for all the time steps are carried out on mesh 3, the details of which are given Table 3.5. It can be seen that the behaviour of error vs time graph for all the time steps is quite similar and the major difference lies in the magnitude of error at different time stages.

The decrease of the error with the decrease in the magnitude of the time steps can be explained in the numerical model if the flow at each time interval is observed. During each time interval, a certain amount of flow occurs in the domain because of which the saturation of the nodes and the diffusivity of the element changes by the end of that time interval. The difference between the explicit and the implicit model is that the former uses the diffusivity of the elements at the beginning of the time interval and the latter uses the diffusivity of the elements at the end of the time interval. If the time interval is quite small i.e. a smaller time step is used, the flow that occurs in the domain in one time interval is quite small and there is no significant change in the nodal saturation level and the corresponding element diffusivity by the end of that time step. In this case, the error between the implicit and the explicit model is small as the diffusivity is not varying significantly within one time step. As the magnitude of the time step is increased, the difference in the diffusivity at the beginning and at the end of the time interval increases and due to this, the error increases.



(a) Error vs Time

(b) Error vs Time Step

Figure 3.18: Time step analysis

As it can be seen in Figure3.18a that for the time observed, the error (for all time steps) is decreasing with time showing that the explicit model is stable. But as it can be seen in Figure3.18b that a larger time step gives more error which lowers the accuracy of the explicit model. The time step of 6 seconds is used in the numerical model used to simulate the transport in uncracked concrete. For the time stages at which the flow is observed and compared with experimental results, the error is below 4 percent. It can be concluded that the explicit model is stable even for larger time steps but also accuracy of the model decreases with it.

The model using the exact diffusivity is also run on 4 different meshes as presented in Table 3.5. The results for the mesh sensitivity analysis are shown in Figure 3.19.

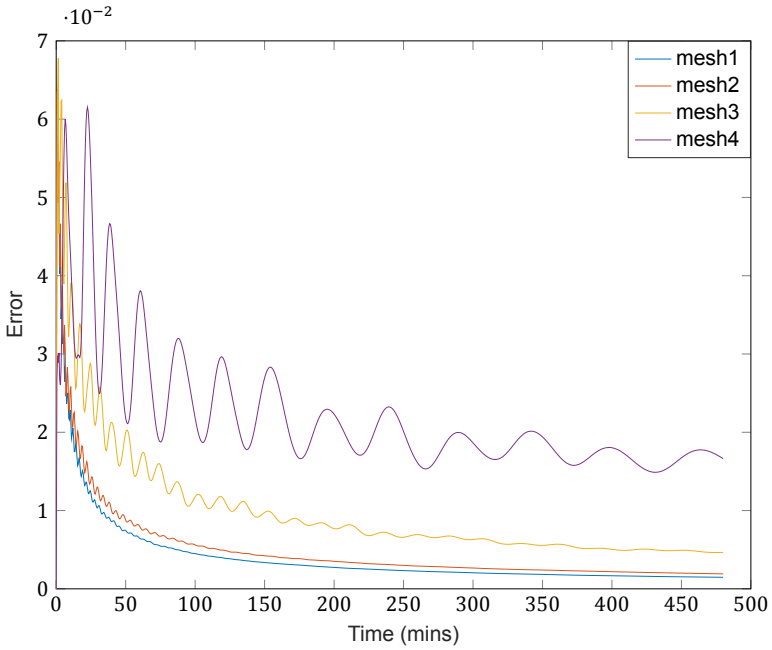


Figure 3.19: Moisture transport in uncracked concrete

It can be seen from Figure 3.19 that the finest mesh produces the most accurate and stable results.

As, the behaviour of the error vs time graph is similar to the one obtained in volume analysis, it can be explained in a similar fashion. Firstly, the global behaviour of the graph is that error is decreasing

with time. This is due to the fact that as time progresses and the water front moves forward in the direction of the flow, more and more number of nodes become completely saturated. In an element where both of its nodes are completely saturated, the diffusivity remains constant over the length of the element and the numerical integration of the diffusivity along the length gives the same value for both the uniform mean diffusivity and the 4-point Gauss integration. Because of this, as the flow progresses, more and more number of nodes become completely saturated and the results from the exact and the uniform mean method tend to become equal. At the final stages of the flow, when all the nodes in the domain will be completely saturated, the diffusivity of the entire domain for both the methods will be equal.

Similar to the volume analysis, there are local peaks in the error vs time graph. At the point when the error starts increasing, that is the point when new elements are included in the flow and one of the nodes is close to fully-saturated condition and the other is close to the initial saturated condition. Due to a significant difference in the diffusivity computed from both the models, the saturation level of the nodes in the next time step is also different and the error increases. This process carries on and the second node gets more and more saturated. Eventually a point occurs when the difference in the diffusivity calculated from both the models starts decreasing because as the second node gets more saturated, the diffusivity computed from the uniform mean and the 4 point Gauss integration comes closer. After that point, the error starts decreasing. This whole process repeats itself when new set of elements are involved in the flow.

As the mesh is refined, new elements are involved in the flow at a faster rate than in a coarser mesh. Because of this, the frequency of the peaks is more for the finer mesh and low for the coarser mesh. In the finer mesh, as the average element length is smaller, the difference of areas (computed from exact and approximate models) is also smaller and the magnitude of the local peaks in error is also smaller.

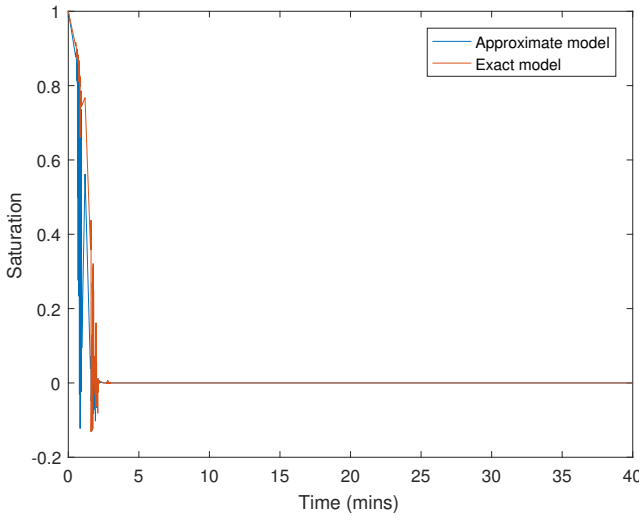
The analyses presented above lead to some limitations of the approximate model. As it can be seen that the mesh should be sufficiently fine so that the error induced because of the approximations is not high. The time step should be chosen in such a way that it gives accurate results but also does not significantly increase the computation time.

3.4. Results and Conclusions

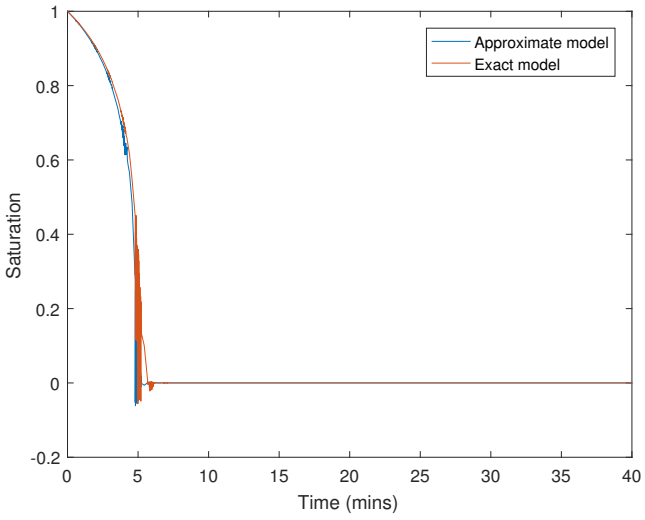
In this section, first a parametric study of varying the diffusivity coefficient is presented followed by the discussion of the results obtained from the numerical analyses and conclusions are drawn based on the numerical results.

3.4.1. Variation of Initial Diffusivity

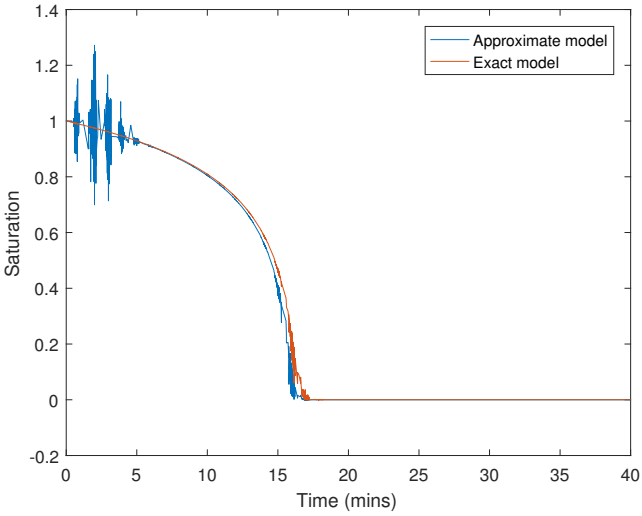
The diffusivity of dry concrete used in the experiment is $D_0 = 1.5467 \times 10^{-4}\ mm^2/s$. In this subsection, the initial diffusivity is varied and the flow is analyzed at 15 minutes for different initial diffusivities. The diffusivity values correspond to $0.1D_0$, D_0, $10D_0$ and $50D_0$ and the rest of the flow parameters are the same used to model the flow in sound concrete as given in Table3.4. Also, the analysis is carried out on mesh 2 (finer) and mesh 3 (reference) (Table3.5) for all the variations in the initial diffusivity. The analyses are carried out on the approximate model which uses approximate volume of Voronoi elements, explicit time stepping technique and uniform mean diffusivity of Voronoi elements. These results are compared with the exact numerical model which uses approximate volume of Voronoi elements, implicit time stepping technique and exponential variation of diffusivity of Voronoi elements.



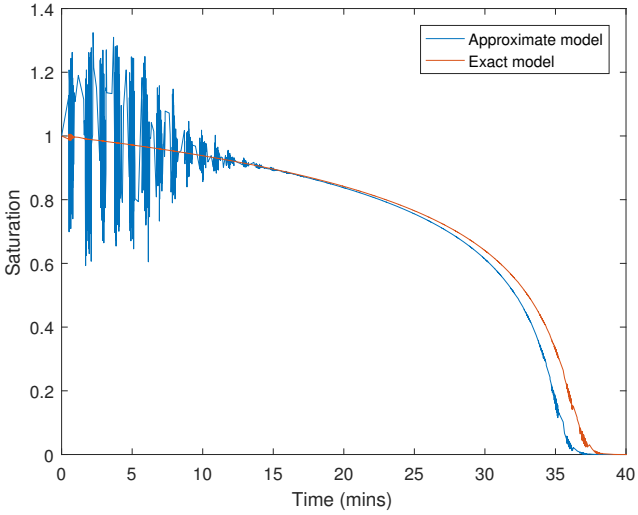
(a) $0.1D_0$



(b) D_0



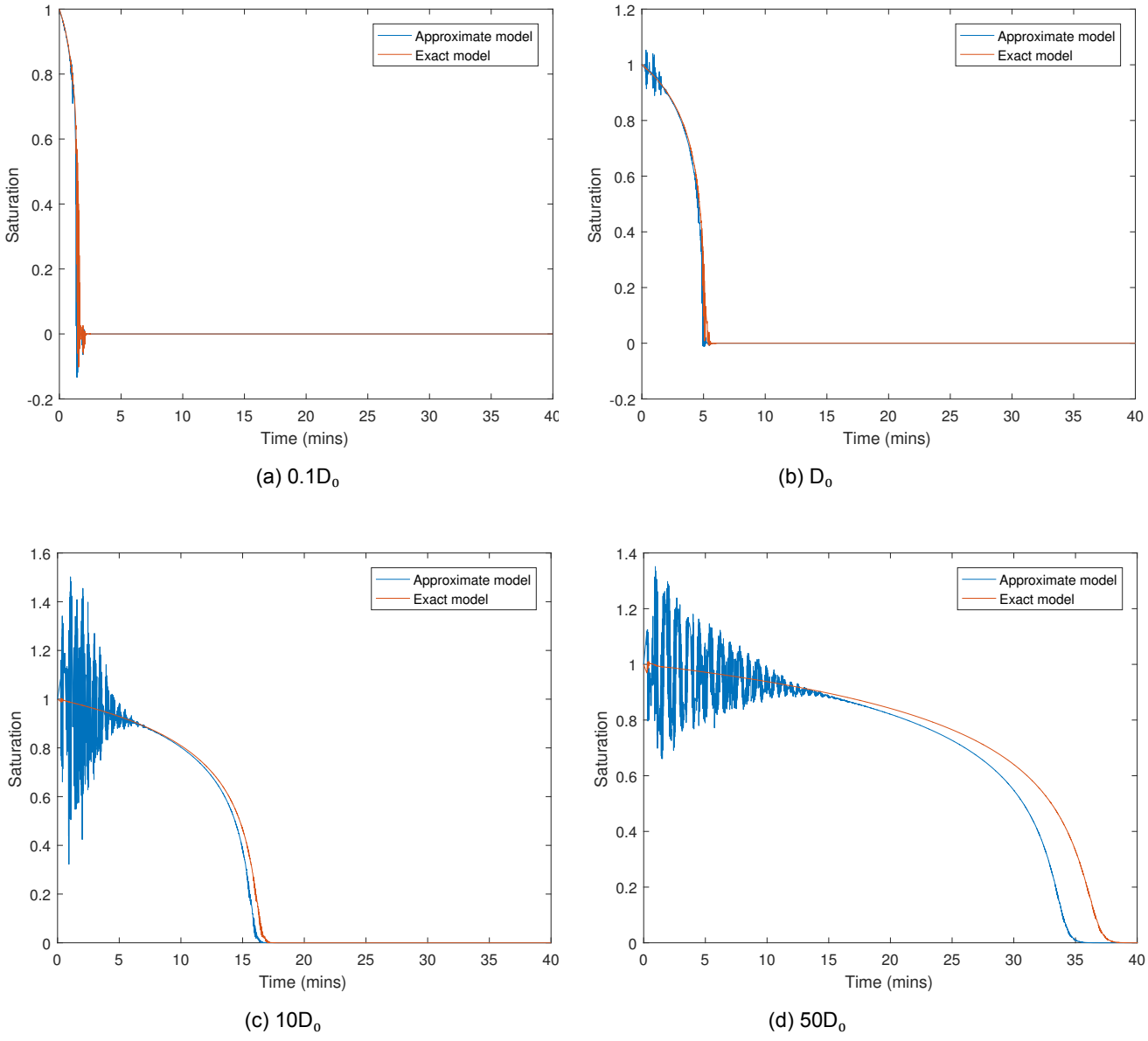
(c) $10D_0$



(d) $50D_0$

Figure 3.20: The results at 15 minutes for mesh 3

The results presented in Figure 3.20 for mesh 3 are not averaged by taking small representative volumes but the x-coordinate of the Voronoi nodes is plotted against the saturation level of that node. The results show that as the diffusivity is increased, the domain gets saturated at a faster rate which is also expected. The same analysis is done for a finer mesh corresponding to the mesh 2. The results are shown in Figure 3.21.



(a) $0.1D_0$ (b) D_0

(c) $10D_0$ (d) $50D_0$

Figure 3.21: The results at 15 minutes for mesh 2

There are some important remarks that are to be highlighted by observing the results. As it can be seen in Figure 3.20 that as the initial diffusivity is increased, there are oscillations in the saturation levels of the nodes that lie in the vicinity of the boundary surface from where the flow originates. As the diffusivity is increased, nodes at a greater distance from the flow boundary show oscillations. The affect of diffusivity can be considered to be the same as the affect of time step in the discretized equation because both of these terms occur in the same form in the fully discretized equation. The oscillations are only observed in the approximate model (approximate volume, explicit time stepping, uniform mean diffusivity) but not in the exact model (approximate volume, implicit time stepping, exponential diffusivity) model. This is an important limitation of the approximate model. In the approximate model, the

linearization of the time-stepping technique is done. In the exact model, there are iterations that are carried out to solve the discretized equation. The main purpose of the iterations is to maintain the mass and momentum balance of the model. It ensures that the total amount of moisture that leaves at time t_n is equal to the total amount of moisture that is entering at time t_{n+1} maintaining the flow equilibrium. When the diffusivity of the elements is low, the amount of moisture leaving the previous time step is very small even if the flow equilibrium is not maintained, there is insignificant difference between the saturation levels obtained from both the models. But when the diffusivity is high, using the approximate model causes oscillations. At the nodes in the vicinity of the boundary, there is a huge gradient of diffusivity in a single element. Because of this, the moisture travels to the adjacent nodes at a very high rate without maintaining the equilibrium between the time steps. Because of this, the moisture in one node from where the flow starts can decrease considerably and the moisture in the adjoining node which receives the flow can accumulate there in one time step. Because of this, there are oscillations in the flow. But as the flow progresses further in the domain, the gradient of diffusivity along the elements decreases and the oscillations decrease when the flow reaches further from the boundary of the origin of flow.

In Figure 3.21, the same analyses with the variation of diffusivity is considered but for a finer mesh. It can be seen that the global behaviour is still the same. But using the finer mesh has caused an increase of oscillations. In mesh 3, the oscillations for the diffusivity D_0 as shown in Figure 3.20b are almost negligible but in case of mesh 2 which is the finer mesh the oscillations at the same diffusivity D_0 (Figure3.21b) are observed. Also, when the magnitude of the initial diffusivity increases, the magnitude of the oscillations also increases with mesh refinement. This can also be explained with a similar line of reasoning that when the mesh is refined, the element size is small. Because of the small element length, the gradient of the exponential distribution of diffusivity over the length of the element is very high. Because of this, when the approximated model is used, the flow in these elements is not able to maintain the equilibrium as it could have been done in the exact model. Because of this, the flow in the elements near the boundary progresses in such a way that in one of the node, there is excessive withdrawal of moisture and in the adjacent node, there is accumulation of moisture. Hence, the oscillations are observed.

3.4.2. Discussion

In the case of constant diffusion in the domain, the numerical model is able to simulate the flow with high accuracy with respect to the analytical results for both steady-state and non-stationary flow as shown in Figure 3.5 and Figure3.7.

For the case of non-stationary transport with time-dependent diffusion, the results of the numerical model are compared to the experimental values. The numerical model is calibrated using the material parameters, initial and the boundary conditions as given in the experiment. Mesh sensitivity analysis on 4 different meshes concluded that mesh 3 gives results of reasonable accuracy with the minimum computation time. Using mesh 3, the flow is simulated using the numerical model and the results are compared to the experimental results in Figure3.11. The numerical model is under predicting the flow at 15 minutes by a small amount which is just the inherent property of the numerical model. At 30, 60 and 120 minutes, the results of the numerical model conform well with the experimental results. At the time stage of 480 minutes, the results from the numerical model are significantly different from the experimental values both in terms of the shape of the curve and the values of the saturation level. This is because of the heterogeneities present in the concrete used in the experiment which has an affect on the later stages of the flow [14] whereas the concrete is modelled as a homogeneous material in the numerical model.

The three approximations made in the numerical model help for easy implementation and reduces the computation time. The approximations are developed looking at the flow in the domain when it attains complete saturation. Under complete saturation, the flow in the approximated model and the exact model is exactly the same. Error within the approximate and the exact models is high in the beginning but as the domain gets more and more saturated, the error decreases. The results show that the approximated numerical model is stable for the observed time of 480 minutes.

Based on the results of the approximations, there are some limitations on the numerical model:

Mesh size
A too coarse mesh will give high errors in the saturation level if the volume and diffusivity approximations are used. If the mesh is too fine, first of all it will increase the computation time. Also in the volume approximation, it is found that a finer mesh can give higher error on account of having more number of elements (Figure3.13). Also, in a finer mesh there can be oscillations in the saturation levels in the nodes closer to the flow boundary (as observed in Figure3.21) when the approximate model is used. These oscillations increase in magnitude if the diffusivity is high. Hence, the mesh size should be chosen such that it is not too coarse and not too fine and does not gives oscillating results for that particular diffusivity of the domain.

Time Step
The time step should not be too small that the computation time increases significantly without significant increase of accuracy. The error in the saturation level, computed from the implicit and the explicit model, increases with the increase of the magnitude of the time step (Figure3.18). If the time step has to be chosen large, then the implicit time stepping scheme should be used. But doing that will increase the computation time on account of the iterations that have to be done in each time step to attain the flow equilibrium.

Diffusivity
As shown in Figure3.20 and Figure3.21 that for higher values of diffusivity, the approximated model shows oscillations in saturation level of the nodes lying closer to the flow boundary. Also, the error at a particular instance between the exact and the approximated model increases for higher diffusivity. If a material with a higher diffusivity has to be modelled, the approximate model won't give reliable results. Instead the implicit exact diffusivity model should be used. But there is a shortcoming with that too. The initial diffusivity (D_0) has the same affect on the discretized equation as the magnitude time step (Δt) as they occur in the same form in the diffusivity matrix. For a higher value of diffusivity, the implicit scheme will require a lot of iterations for convergence to occur and that can significantly increase the computation time. The magnitude of the time step can also be decreased to reduce the number of iterations but that can significantly limit the number of time steps that can be used (limiting the total time for which the flow is observed) on behalf of the computation time.

4

Moisture Transport in Cracked Concrete

In the previous section, the numerical model was shown to be able to simulate moisture transport in uncracked concrete and results were in conformation with the experimental results. In this section, moisture transport in cracked concrete will be simulated. A single discrete crack is modelled in the mesh and its affect on transport behaviour of the domain is analyzed. The first section presents the methodology to include the crack in the discretized domain along two approaches to simulate moisture transport in the crack. In the second and third section, horizontal and vertical moisture distribution in the crack using both approaches is modelled and compared with each other. Last section highlights the difference in the modelling technique when the transport elements are modelled on the Delaunay and on the Voronoi elements respectively followed by simulation of moisture transport when both the approaches to model moisture transport in a crack are combined into a single model. This is followed by the discussion of results of the analyses.

4.1. Model Discretization

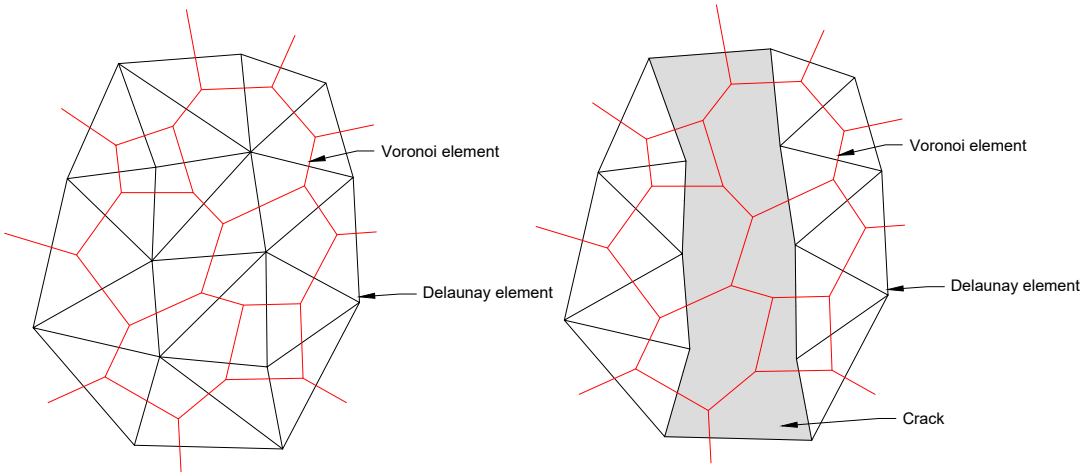
Firstly, the crack has to be modelled in the discretized domain and the transport properties of the Voronoi elements lying in the crack and the cement matrix have to be assigned accordingly. Effect of a single discrete crack on moisture transport is observed in the numerical model. To account for the effect of crack on moisture transport, two approaches are considered.

Approach 1: Crack as a Porous Phase in Mortar
In this approach, concrete is considered to be a two-phase composite consisting of the cement matrix and the crack. When moisture transport in uncracked concrete is simulated, concrete is considered as a network of porous pipes through which capillary absorption takes place [7]. In this approach, the crack is also considered to be a network of porous pipes and the underlying mechanism of transport through capillary absorption remains the same as in uncracked concrete but the difference lies in the magnitude of transport that takes place in the crack which is much higher than in the uncracked part of the domain. The porosity of the crack is considered to be 1 and the Voronoi elements used to simulate transport in the cracked domain are assigned diffusivity according to the cubic law.

Approach 2: Surface Interactions between Moisture and Crack Walls
In Approach 1, the surface interaction between the crack walls and the moisture surface is not taken into account and the physical material behaviour of the crack is incomplete. In this approach, the surface interactions between the crack walls and moisture is taken into account through capillary suction. The net effect of interfacial tension and inertial forces causes the water in the crack to rise up to a certain height known as the equilibrium rise height. The height till which the liquid rises in the crack remains completely submerged with moisture and acts as additional submerged boundaries. In the numerical model, the part of the crack walls that lie within the equilibrium rise height is modelled as just another boundary with constant saturation, while the part of the crack walls lying outside the equilibrium rise height are considered as a porous phase in mortar as done in Approach 1. Approach 2 takes into account the surface interactions of moisture and crack walls which is a better representation of the actual physical behaviour of the cracks.

4.1.1. Crack Generation

The structural properties of concrete are modelled on Delaunay elements in the dual mesh. Due to this, the discrete crack in the concrete is modelled using the Delaunay elements and the corresponding transport properties are modelled on the Voronoi elements. Figure 4.1 shows a segment of the dual mesh before and after cracking. The Delaunay elements are loaded under an external force (in finite loading steps) because of which the stress in the Delaunay elements starts increasing. When a Delaunay element reaches its tensile limit, it is removed from the mesh. The new mesh, after removing the Delaunay element, is further loaded till another Delaunay element reaches it's tensile limit and is subsequently removed. This process continues till the entire external load has been applied or till the structure cannot resist any more load or the desired deformation of the structure is attained. At the end of the loading procedure, the entire space that was previously occupied by the removed Delaunay elements, form the crack in the discretized domain as shown in Figure4.1b. Once the crack is generated, the Voronoi elements can be assigned the respective transport properties to simulate moisture transport in the domain.



(a) Uncracked dual mesh (b) Cracked dual mesh

Figure 4.1: A segment of the dual mesh

4.1.2. Approach 1: Crack as a porous phase in mortar

The discretized equation, as given in eq(4.1), used to simulate the moisture transport remains the same for both the cracked and uncracked concrete as the underlying phenomenon of transport is via capillary absorption. The difference is that the diffusivity matrix $\mathbf{K}$ is formulated with different diffusivities for the elements lying in the crack than the elements lying in the uncracked concrete.

$$\left(\mathbf{M} + \frac{1}{2}\Delta t \mathbf{K_{n+1}}\right)\boldsymbol{\theta}_{\mathbf{n+1}} = \left(\mathbf{M} - \frac{1}{2}\Delta t \mathbf{K_n}\right)\boldsymbol{\theta}_{\mathbf{n}} + \Delta t f \tag{4.1}$$

The Voronoi elements that are the geometric dual of the removed Delaunay elements lie in the cracked part of the domain. Their transport properties differ from the Voronoi elements lying in the uncracked part of the domain where the diffusivity is given as:

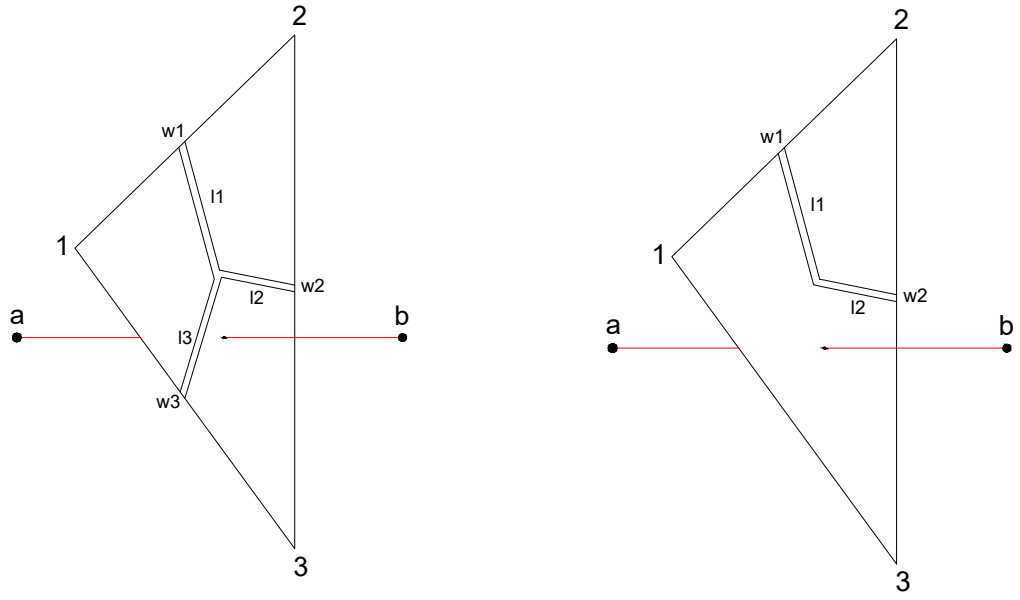
$$D = D_0 e^{n\theta} \tag{4.2}$$

The diffusivity of the Voronoi elements lying in the crack (D_w^{cr}) is given by eq(2.22) where the crack is considered to absorb moisture as a porous phase and the diffusivity is proposed to be:

$$D_w^{cr}(\theta) = \left(\sum_{i=1}^{n} w_i^3 l_i\right)\frac{\xi p_r (1-m)}{12\mu A m}\sqrt{\theta}[1-(1-\theta^{1/m})^m]^2\theta^{-1/m-1}(\theta^{-1/m}-1)^{-m} \tag{4.3}$$

where,
w_i is the equivalent crack width,
l_i is the equivalent crack length,
n is the number of cracked Delaunay elements
A is the cross-sectional area of the Voronoi element,
ξ is the tortuosity factor,
p_r is the reference pressure, experimentally determined as $18.6237\ N/mm^2$,
m is the Van-Genuchten parameter equal to 0.4396,
μ is the viscous coefficient of water.

In eq(4.3), it can be seen that in the first term, the affect of the crack geometry comes into account. The equivalent crack widths and the equivalent crack lengths can be computed for a Voronoi element as shown in Figure4.2a. The Voronoi element a-b is surrounded by three Delaunay elements 1-2, 2-3 and 3-1 that form the common triangle. When the Delaunay elements are cracked, the crack widths $w1$, $w2$ and $w3$ are assumed to be present in the middle of the corresponding elements. The crack lengths $l1$, $l2$ and $l3$ are obtained by measuring the distance from the centre of the corresponding Delaunay elements to the centroid of the common triangle 1-2-3. The cross-sectional area A of the Voronoi element is the area of the triangle 1-2-3. When cracking is simulated in three dimensions, the Delaunay elements lying in the cracked plane are considered to be cracked. Due to these elements lying in a plane, it is never the case that a Voronoi element passes through three cracked Delaunay elements as shown in Figure4.2a. Instead, the most number of cracked Delaunay elements that pass through a Voronoi element is two as shown in Figure4.2b. In such a case, the Voronoi element lies partly in the crack and partly in the uncracked region and it's diffusivity should be considered as the weighted average of the diffusivity of the cracked and the uncracked region. But the contribution of the uncracked region towards the diffusivity is considerably smaller as compared to that of the cracked region and only the cracked Delaunay elements (1-2 and 2-3 as shown in Figure4.2b) are used to assign the diffusivity of the Voronoi element.

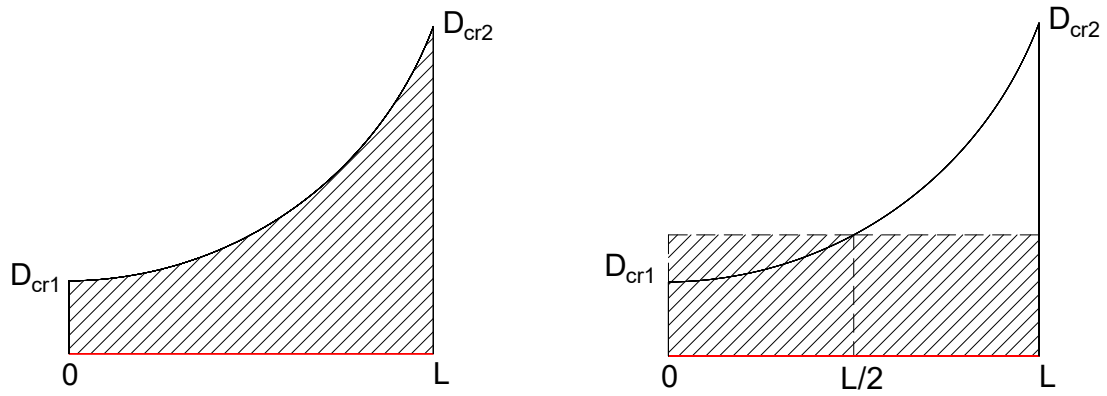


(a) Voronoi element surrounded by three cracked Delaunay elements

(b) Voronoi element surrounded by two crack Delaunay elements

Figure 4.2: Equivalent crack properties of the Voronoi element

Numerical Solver
It can be seen in eq(4.3) that the diffusivity of the Voronoi elements lying in the crack is varying with the saturation level. Figure4.3a shows the variation of the diffusivity over the length of the element. It should be noted that the variation is not exponential as in the case of uncracked concrete. As it was seen in the case of uncracked concrete, if the mesh is refined enough, the approximate volume of the Voronoi elements give stable results with reasonable accuracy. In the numerical model used here, the approximate volume of the Voronoi elements will be used. The diffusivity in the crack is of a much higher order of magnitude than of the uncracked cement matrix. As it was observed in the previous chapter that as the magnitude of the diffusivity increases, the explicit time stepping scheme coupled with mean uniform variation of diffusivity (Figure4.3b) gives oscillating results. But using the implicit time stepping scheme along with the actual variation of diffusivity (Figure4.3a) will drastically increase the computation time because of the high diffusivity of the crack. The number of iterations to attain convergence at each time step will be really high which will increase the computation cost. Hence, to simulate moisture transport in cracked concrete a numerical model using approximate volume of the Voronoi elements, explicit time stepping technique and the uniform mean diffusivity of the elements is adopted.



(a) Exact diffusivity (b) Uniform mean diffusivity

Figure 4.3: Variation of diffusivity for cracked element

4.1.3. Approach 2: Surface interactions between moisture and crack walls

In Approach 1, the real water absorption potential of the crack is not modelled as it does not take into account the surface interactions of the moisture and the crack walls. In this approach, the surface interactions of the moisture with the crack surface is described by the Lucas-Washburn equation[35]. The correction for stick-slip behaviour, dynamic contact angle and wall slip are applied which results in the modified Lucas-Washburn equation[11]. The equation is already described in Chapter 2 eq(2.35), which is given as:

$$\left(\frac{2\beta_m}{r} + \frac{z}{\frac{r\beta_w}{2} + \frac{r^2}{8\mu}}\right)\dot{z} = p_{c0}(1-\beta_s) - \rho g z sin(\phi) \tag{4.4}$$

where
β_m is the correction factor for dynamic contact angle,
r is the radius of the capillary,
z is the capillary rise,
β_w is the correction factor for the wall slip,
μ is the dynamic viscosity,
β_s is the correction factor for stick-slip behaviour,
ρ is the density of the liquid,
ϕ is the tube inclination angle,
g is the acceleration due to gravity,

p_{c0} is the capillary pressure and $p_{c0} = \frac{2\gamma cos(\theta_0)}{r}$ where γ is the surface tension.

For simplicity, eq(4.9) is written in the from of:

$$k_1 k_2 \dot{z} + z\dot{z} = k_2 k_3 - k_2 k_4 z \tag{4.5}$$

where

$$k_1 = \frac{2\beta_m}{r}$$

$$k_2 = \frac{r\beta_w}{2} + \frac{r^2}{8\mu}$$

$$k_3 = p_{c0}(1 - \beta_s)$$

$$k_4 = \rho g sin(\phi)$$

Eq(4.9) gives the expression for the rate of capillary rise in a capillary. This equation is discretized in time using the Crank-Nicolson procedure so that it can be implemented in the numerical model. According to the Crank-Nicolson procedure, the capillary rise at time t_{n+1} is given as:

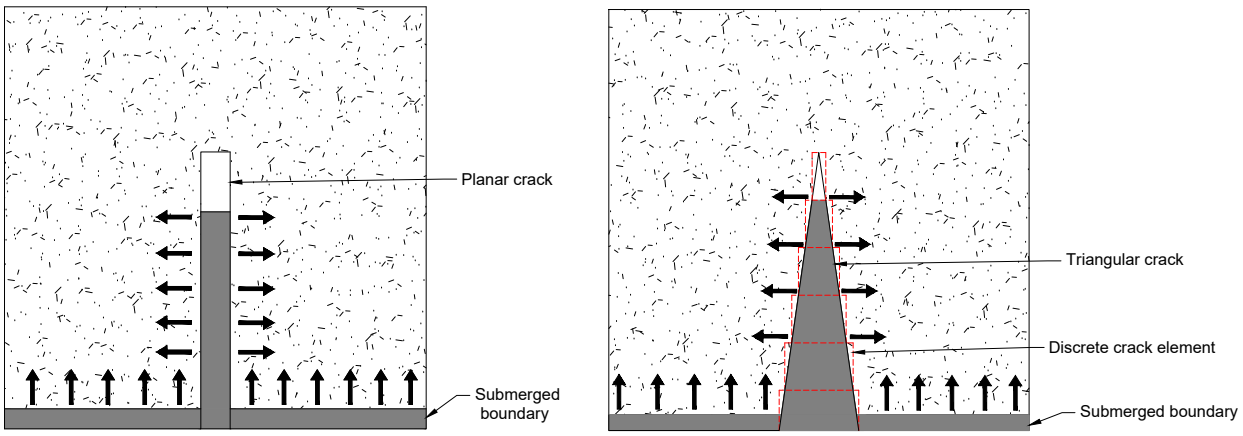
$$z_{n+1} = z_n + \frac{\Delta t}{2}\left(\dot{z}_n + \dot{z}_{n+1}\right) \tag{4.6}$$

Substituting the expression given in eq(4.6) into the modified Lucas-Washburn equation and eliminating the time derivatives, the following discrete from of the modified Lucas-Washburn equation is obtained:

$$\frac{2}{\Delta t} z_{n+1}^2 + \left(\frac{2}{\Delta t} k_1 k_2 - \frac{2}{\Delta t} z_n - \frac{k_2 k_3 - k_2 k_4 z_n}{k_1 k_2 + z_n} + k_2 k_4\right) z_{n+1} - \left(\frac{2}{\Delta t} k_1 k_2 z_n + \frac{k_1 k_2 (k_2 k_3 - k_2 k_4 z_n)}{k_1 k_2 + z_n} + k_2 k_3\right) = 0 \tag{4.7}$$

Coupling of transport in the crack and uncracked concrete

When the surface of concrete is in contact with water, the crack acts as a capillary tube. Due to surface interactions with the moisture present at the submerged boundary, the water rises up in the capillary under the capillary suction. The rise of the water comes to a rest once the capillary attains equilibrium height. The height up till which the crack gets filled up acts a boundary with constant saturation and the moisture starts spreading into the adjoining matrix through the crack walls as shown in Figure4.4. The capillary suction in the crack happens in mere seconds in which the equilibrium height is obtained, while the process of moisture transport in the adjoining matrix is comparatively a much slower process. Because of this, the coupling of both the phenomenon in one numerical model is quite complex. Due to this reason, first the capillary suction in the crack is modelled and then considering the crack walls as additional submerged boundaries, the transfer of moisture in the adjoining cement matrix is simulated using a separate numerical model.



(a) Planar crack

(b) Triangular crack

Figure 4.4: Capillary suction in the crack

For the case of a planar crack (Figure4.4a), the capillary suction can be simulated directly using the modified Lucas-Washburn equation (eq(4.9) as the crack width is uniform throughout the depth of the crack. Once the equilibrium height is attained, the moisture starts spreading into the adjoining matrix. In case the crack width is not uniform, but varies with the crack depth as shown in Figure4.4b, the crack is divided into a number of discrete elements. The crack width is assumed to be constant for each discrete element and the capillary suction is modelled using the discretized form of the Lucas-Washburn equation (eq(4.1)). Due to capillary suction, the water in the crack reaches an equilibrium height but if the crack depth is greater than the equilibrium height, a part of the crack is left which is not filled with water. This part is considered to act as a porous material in which the transport of water takes place according to the diffusivity obtained through the cubic law.

Numerical Solver

In the case when the crack depth is smaller than the equilibrium rise height of the capillary, the crack gets completely filled up. In such a case both the crack walls are just additional submerged boundaries in the numerical model. Simulating moisture transport in such a domain is no different than simulating moisture transport in uncracked concrete. The numerical model using approximate volume, explicit time stepping scheme and uniform mean variation of diffusivity is able to give results with reasonable accuracy. In case the crack is not completely filled up but a part of the crack is not submerged in water, then that part of the crack is considered as a porous phase in mortar and the numerical model adopted to simulate the moisture remains the same as is mentioned in Approach 1. Hence, in this case too, a numerical model with approximate volume of the Voronoi elements, explicit time stepping scheme and uniform mean diffusivity of the elements is used to simulate moisture transport.

4.2. Horizontal Moisture Distribution in Cracked Concrete

In this section, the horizontal moisture distribution in a planar crack is observed. A planar crack is modelled in the discretized domain with constant crack width and parallel surfaces. The moisture transport in the cracked domain is simulated using the two approaches of considering crack as a porous phase in mortar and moisture-surface interactions as mentioned before and the results of each are compared with experimental results.

4.2.1. Experiment

The experiment used in [38] is used as a reference to simulate horizontal moisture distribution in cracked concrete. A cylindrical specimen of $16\ mm$ diameter and $32\ mm$ height was used. There were two grooves of $2\ mm$ width and $2\ mm$ height placed diametrically opposite to each other as shown in Figure4.5. The water-cement ratio was kept at 0.45 and the cement to sand ratio was kept at 3.27. Aggregates of size ranging from $0.25 - 2\ mm$ were included in the mix and there particle distribution was measured using laser diffraction.

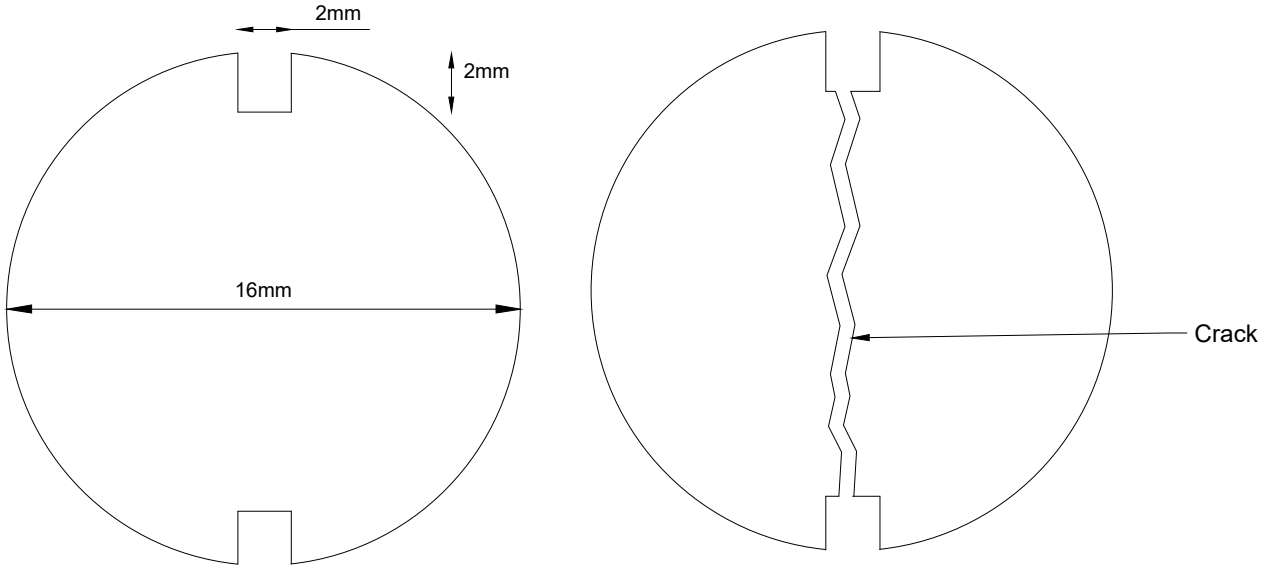


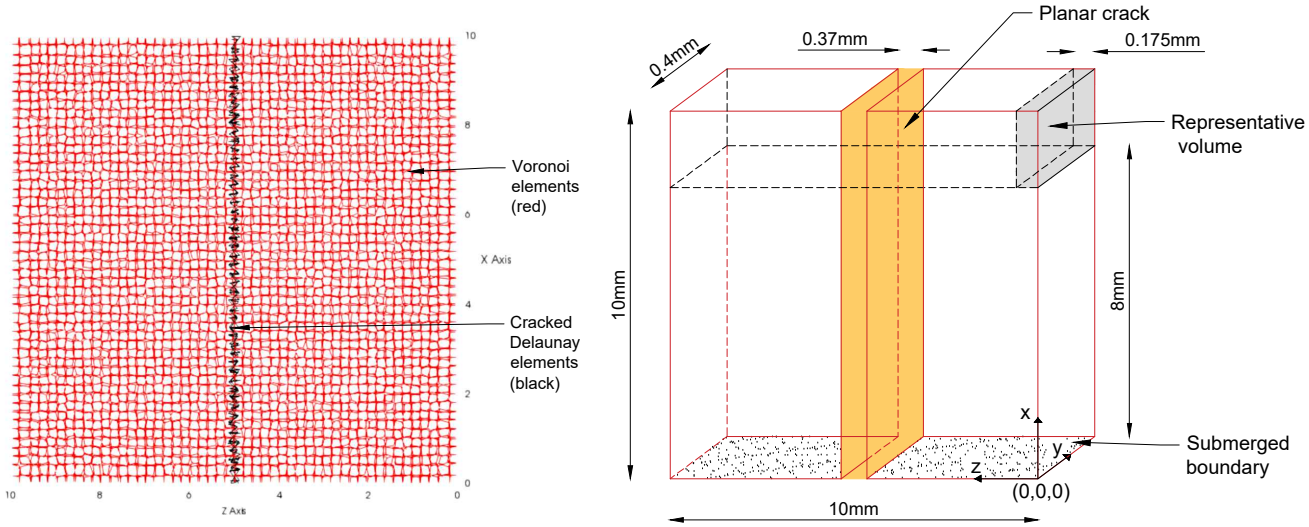
Figure 4.5: Schematic of the sample and the crack [38]

The sample was covered with a plastic foil and left to harden for 24 hours under laboratory conditions. After 24 hours, the sample was demoulded and stored in a fog room at about $20°C$ for 28 days. At 21 days, the sample was sawn into a smaller cylinder of $10\ mm$ height and returned to the fog room. At 28 days, the sample was put in an oven at $40°C$ until constant weight was achieved. All the surfaces were covered with a duct tape until the testing. A planar crack with a crack-width of about $0.37\ mm$ was formed going from one groove end to the other. X-Ray Micro Tomography was used to observe the moisture content at different time stages [38]. As it can seen in Figure4.5 that the crack is actually a natural crack, but since a 3D imaging technique is used, the data has been arranged to have the center of the crack for each depth and height in the same position. This results in a planar crack data and the crack width can be considered constant.

4.2.2. Numerical Analysis

For the numerical analysis, a cracked domain is generated that can simulate the moisture distribution as given in the experiment. A dual mesh for a domain size of $10 \times 0.4 \times 10\ mm^3$ is generated and a crack starting from $x = 0$ to $x = 10\ mm$ is modelled on the Delaunay elements as shown in Figure4.6a. The mesh properties are given in the Table 4.1. The average Delaunay element size is taken as close to the

crack width as possible so that when the crack is generated in the Delaunay mesh and the Delaunay elements are eliminated, the width of the crack generated in the mesh is representative of the actual crack width as used in the experiment.



(a) Cracked mesh (b) Crack Details

Figure 4.6: Discrete crack in the domain

Table 4.1: Mesh Properties

Parameter	Value
Number of nodes	26225
Number of elements	37456
Average element size	$0.1\ mm$
Crack width	$0.3719\ mm$
Crack depth	$10\ mm$

Table 4.2: Flow Parameters

Parameter	Symbol	Value
Diffusivity of dry concrete	D_0	$3.5 \times 10^{-5}\ mm^2/s$
Empirical constant	n	6
Initial saturation	θ_0	0.2119
Porosity	p	0.1471
Time step	Δ t	$6s$

The flow parameters used to simulate moisture transport in uncracked concrete are given in Table4.2. The flow is simulated in the numerical model and the results are compared with the experimental results. Figure4.6b shows the details of the cracked domain. The plane $x = 0\ mm$ is submerged and kept at a constant saturation level. The crack planes are considered parallel and smooth with a constant crack-width of $0.37\ mm$ and running from $x = 0$ to $x = 10\ mm$. In order to compare the horizontal distribution of moisture, only the volume of the domain lying between $x = 8\ mm$ and $x = 10\ mm$ is considered. This volume is further divided into small representative volumes (running along the z-axis) of dimensions $2 \times 0.4 \times 0.175\ mm^3$ (one such representative volume is shown in Figure4.6b). The saturation level and the z-coordinate of all the Voronoi nodes lying in a particular representative volume are averaged and plotted in order to compare the results with the experimental results.

Approach 1

The entire domain is divided into uncracked and cracked concrete and the flow parameters of both regions are defined separately. The moisture transport in uncracked concrete is based on the flow parameters as given in Table4.2. The moisture transport in the cracked domain is modelled by specifying different diffusivity of the Voronoi elements lying in the crack as given by:

$$D_w^{cr}(\theta) = \left(\sum_{i=1}^{3} w_i^3 l_i\right) \frac{\xi p_r (1-m)}{12 \mu A m} \sqrt{\theta} [1 - (1 - \theta^{1/m})^m]^2 \theta^{-1/m-1} (\theta^{-1/m} - 1)^{-m} \tag{4.8}$$

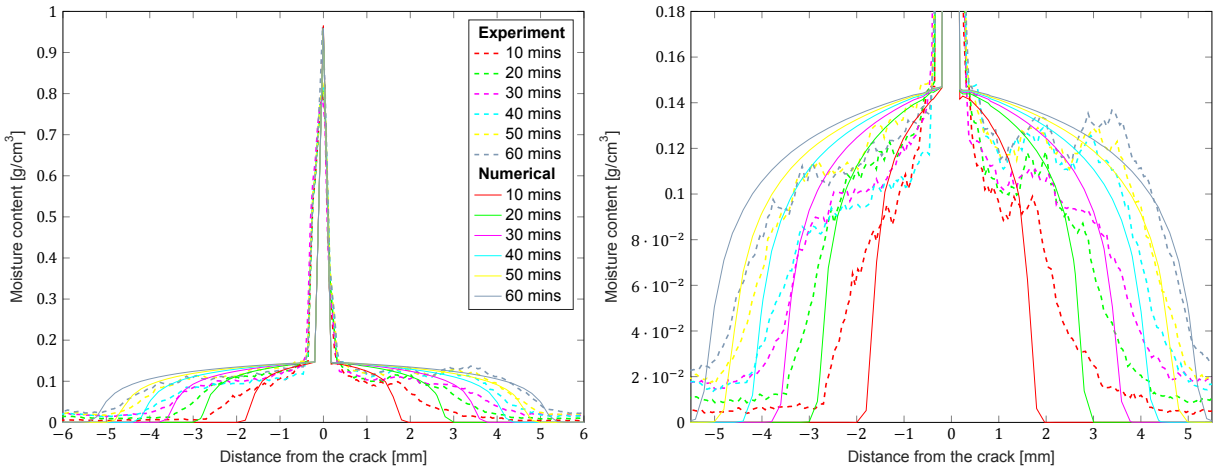


Figure 4.7: Horizontal moisture distribution considering crack as a porous phase in mortar

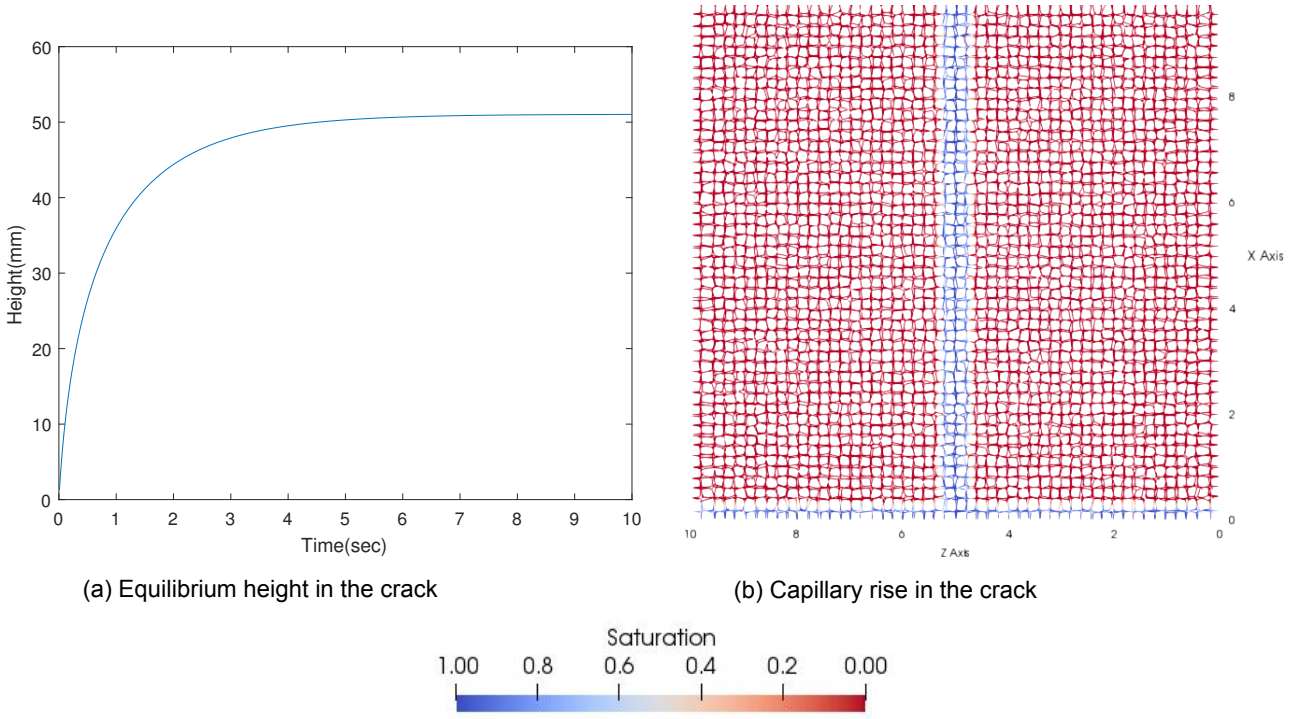
The results of the numerical analysis for a total time period of 60 minutes is given in Figure4.7. The left hand side figure shows the horizontal moisture distribution and the right hand side figure magnifies a part of the left hand side figure to better observe the moisture distribution in the vicinity of the crack. It can be seen that the effect of crack is successfully simulated in the numerical model. Voronoi nodes lying in the crack are at a completely saturated state and the saturation level of the nodes keep on decreasing with increasing distance from the crack. The penetration depth of the water around the crack for the different time stages obtained in the experiment and numerical model are closer for the later time stages than for the initial time stages. Also, the horizontal moisture distribution curves for the different time stages are not of the same shape in the numerical and the experimental model. These differences in the results is because, in the numerical model, the concrete is considered as a homogeneous material and the crack is considered to be planar with uniform crack width over the crack depth. In the experiment, aggregates are present in the material and the crack that is generated is not completely planar. Because of these approximations, the results in the numerical model show some deviations from the experimental results.

Approach 2

There is no need to discretize the crack in the numerical model as the crack is planar and the crack width is not changing with the crack depth. The modified Lucas-Washburn equation can be directly used in this case to get the equilibrium height:

$$\left(\frac{2\beta_m}{r} + \frac{z}{\frac{r\beta_w}{2} + \frac{r^2}{8\mu}}\right)\dot{z} = p_{c0}(1-\beta_s) - \rho g z sin(\phi) \tag{4.9}$$

Taking the crack width as 0.3719mm, the Lucas-Washburn equation gives the capillary equilibrium height of 51mm. As the equilibrium height is more than the crack depth, the crack is completely filled with water and the entire crack surfaces act as additional boundaries which are completely saturated as shown in Figure4.8. The moisture starts spreading into the adjoining matrix from these submerged boundaries.



(a) Equilibrium height in the crack

(b) Capillary rise in the crack

Figure 4.8: Capillary rise in the crack

The results of the analysis, for a total time period of 60 minutes, considering the capillary suction in the crack are presented in Figure4.9. It can be seen that the results are similar to the ones obtained when the crack is considered as a porous phase. The crack is completely saturated and the saturation level of the nodes decreases with increase of distance from the crack. The penetration of the water into the surrounding cement matrix around the crack is more for the case when capillary suction is considered compared to when the crack was considered as a porous phase of mortar. The reason behind this is that the process of capillary suction in the crack is very fast and the crack gets immediately filled up with moisture. The moisture starts penetrating the surrounding cement immediately after the crack gets filled up. When the crack is considered as a porous phase of mortar, the crack takes comparatively more time to completely saturate and as a result, the water penetration into the surrounding matrix starts at a later time stage. As a result of this, the penetration depth considering crack as another porous phase is less for the same time stage as the one when capillary suction is considered in the crack.

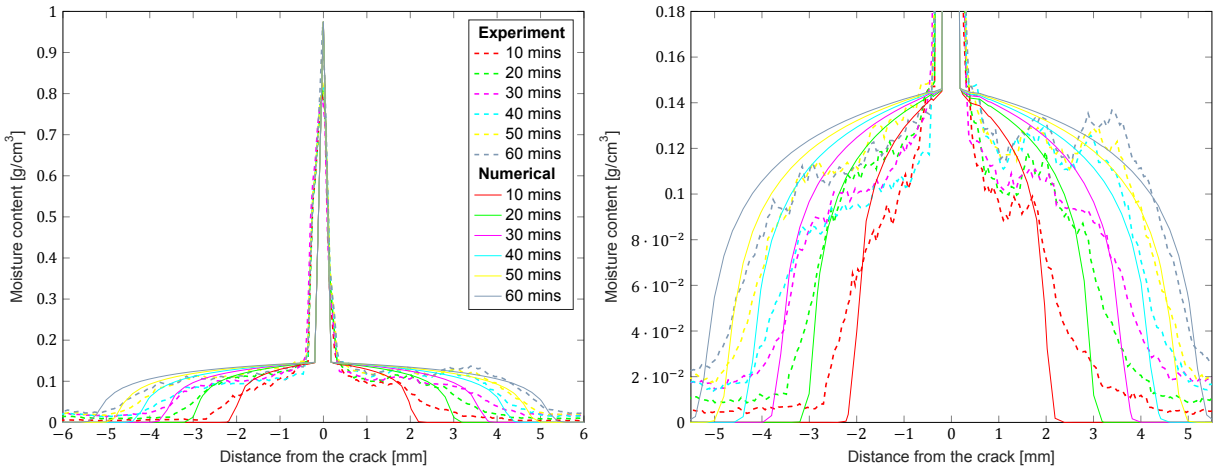


Figure 4.9: Analysis results considering capillary suction in the crack

4.3. Vertical Moisture Distribution in a Crack

In this section, vertical moisture distribution in a wedge crack is observed. The experiment is described entailing the procedure followed to get the cracked specimen and the results of moisture transport in the cracked domain. A numerical model is presented which will be used to simulate moisture in the cracked domain as given in the experiment. Instead of using a natural crack in the numerical model, a triangle shaped crack is used with decreasing crack width with increasing crack depth. The results of the numerical model obtained through the two approaches to simulate moisture transport in crack are compared with the experimental results.

4.3.1. Experiment

The experiment conducted in [55] is used as a reference to model vertical moisture distribution in the crack. Prismatic specimens of dimensions $100 \times 100 \times 300\ mm^3$ were cast with 6 steel bars of $8\ mm$ diameter acting as reinforcements as shown in Figure4.10. A water-cement ratio of 0.6 and concrete mix-proportion of 1:3.3:0.6 (cement:sand:water) by volume was used. After curing for 14 days, the prisms were sliced along the long axis and smaller samples of size $25 \times 100 \times 300\ mm^3$ were obtained as shown in Figure4.11. These smaller samples were subjected to three-point bending under controlled conditions to induce a single crack in the middle with maximum crack width of $0.35\ mm$. From the centre part of all the cracked specimens, slices of width $100\ mm$ were cut as shown in Figure4.11 to obtain the samples of dimension $25 \times 100 \times 100\ mm^3$ where the induced crack is situated at the centre of these slices. At the age of 28 days, these slices were dried for 4 days in a ventilated oven at $50°C$ till constant weight was achieved.

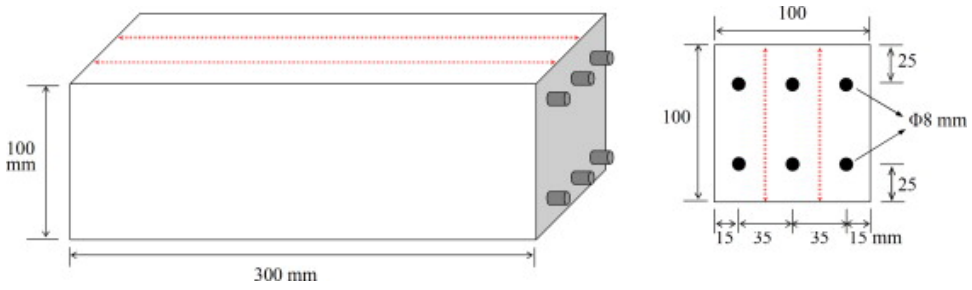


Figure 4.10: Positions of the steel reinforcements in the prismatic specimens and cutting lines [55]

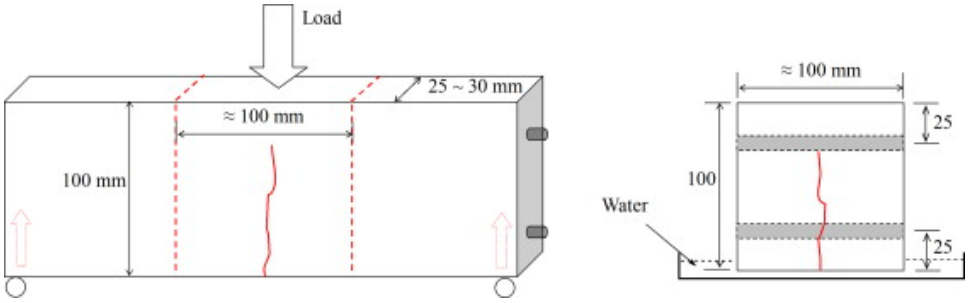


Figure 4.11: Formation of a centre crack under three point bending (left) and detached centre part in contact with water for observation of water penetration (right) [55]

All the surfaces of the sample except for the one where the crack occurs, were sealed with self-adhesive aluminium foils to impose unidirectional movement of moisture in the specimen. This specimen was then placed in a container and positioned according to the neutron beams. After taking the first image of the dry sample using the neutron beam, the container is filled with water so that it just touches the bottom surface of the block as shown in Figure4.11. Water starts penetrating into the crack and the surrounding porous material and the water distribution is monitored through the neutron images taken at different time intervals. As already mentioned in the case of uncracked concrete, the diffusivity of the sound mortar is taken as $D_0 = 1.54676 \times 10^{-4}\ mm^2/s$ which is obtained using the relationship between the capillary coefficient and sorptivity [56].

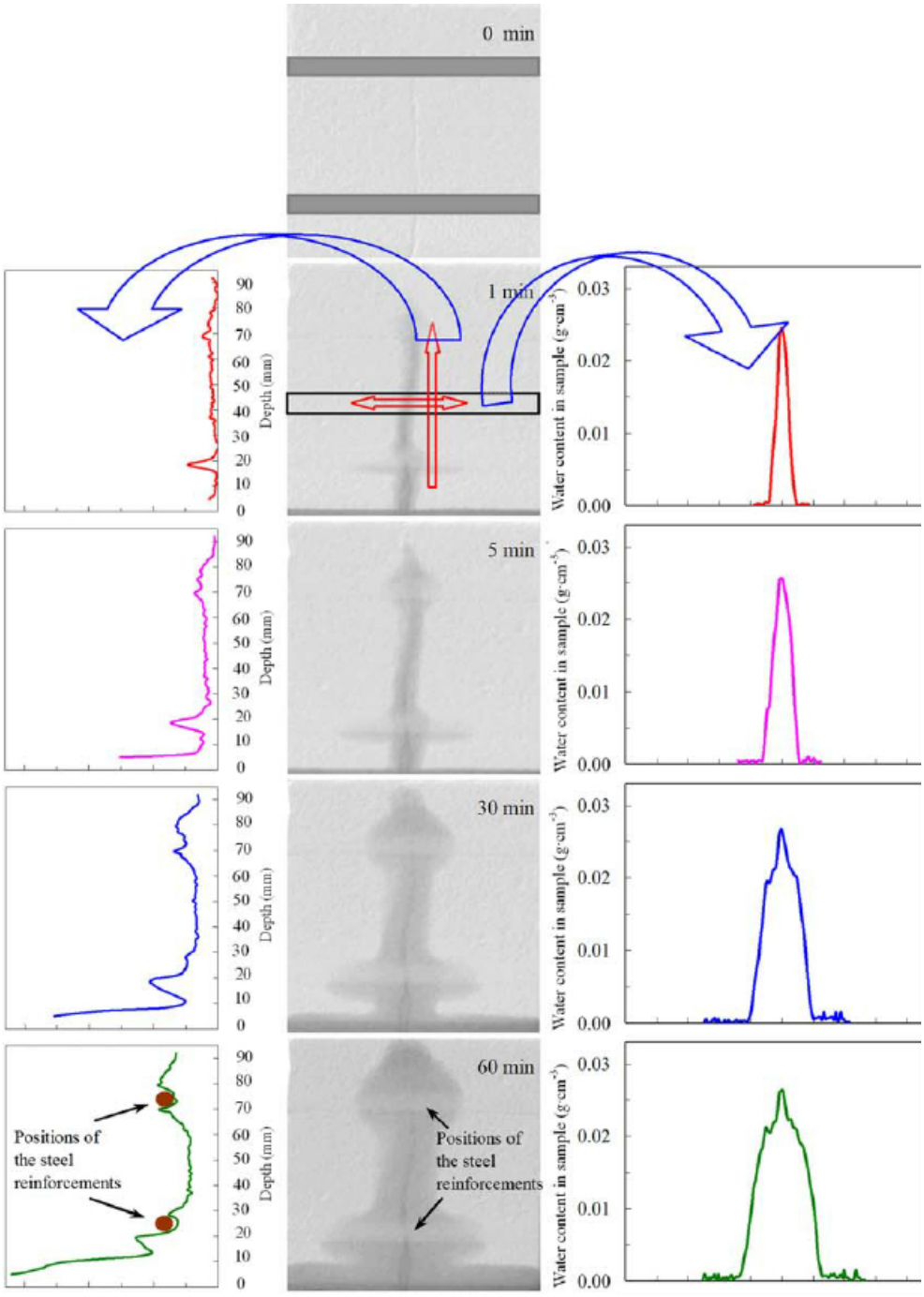


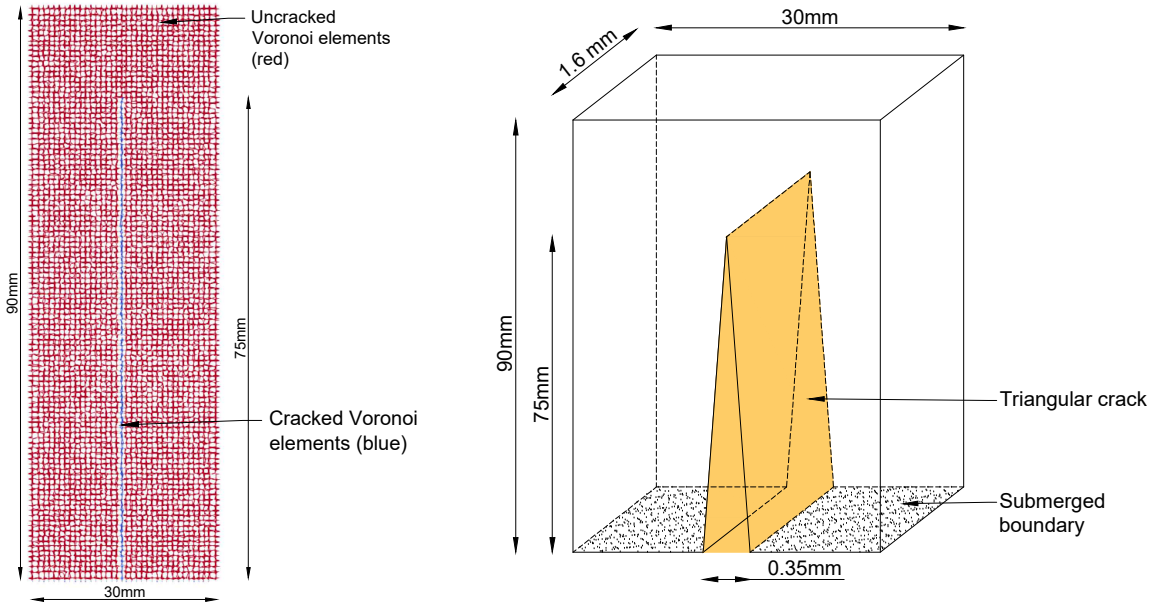
Figure 4.12: Neutron images of water penetration into cracked (crack width of 0.35mm) steel reinforced mortar (centre column) during contact with water for upto 60min, and the corresponding quantitative water profiles along a vertical axis of the sample (left) and along a horizontal axis within the rectangular area marked in the image (right) [55]

Moisture distribution in the cracked sample for time intervals of $1, 5, 30, 60$ minutes are shown in Figure4.12. The left hand side figures shows the vertical moisture distribution in the crack and the right hand side figures show the horizontal moisture distribution in the crack. The effect of reinforcement on the moisture distribution can also be seen clearly. At the place where the concrete is bonded to the reinforcement, the density of the cement particles is less as compared to that in the cement matrix. As a result, the permeability of the region around the aggregates is increased as the porosity is higher and consequently, there is higher moisture content in these regions as compared to the rest of the concrete.

Although, the experiment included reinforcements in the concrete sample, the moisture distribution in the numerical model is simulated in a domain without the steel reinforcements and the moisture profiles are only qualitatively compared to the ones as presented in Figure4.12.

4.3.2. Numerical Analysis

The cracked specimen used in the experiment was formed through three point bending and because of that the crack width is maximum at the bottom and decreases towards the top. In the numerical model, a triangular crack is implemented in the discretized domain so as to reproduce the nature of the crack used in the experiment as shown in Figure4.13b. The bottom surface is submerged in water and the rest of the faces are considered sealed through which no flow occurs. The domain of dimensions $90 \times 1.6 \times 30\ mm^3$ is generated with the maximum crack width of $0.35\ mm$ at the bottom and linearly decreasing till the crack depth of $75\ mm$. The crack is generated using the Delaunay elements and the moisture transport in the crack and the surrounding elements is simulated through the Voronoi elements. The mesh properties of the discretized domain are given in Table4.3 and the flow parameters used in the numerical model to simulate moisture transport are given in Table4.4.



(a) Cracked mesh (b) Crack Details

Figure 4.13: Cracked domain used in the numerical model

Table 4.3: Mesh Properties

Parameter	Value
Number of nodes	43536
Number of elements	62214
Average element size	$0.4039\ mm$
Max. crack width	$0.35\ mm$
Crack depth	$75\ mm$

Table 4.4: Flow Parameters

Parameter	Symbol	Value
Diffusivity of dry concrete	D_0	$1.547 \times 10^{-4}\ mm^2/s$
Empirical constant	n	6
Initial saturation	θ_0	0
Porosity	p	0.0462
Time step	Δ t	$6s$

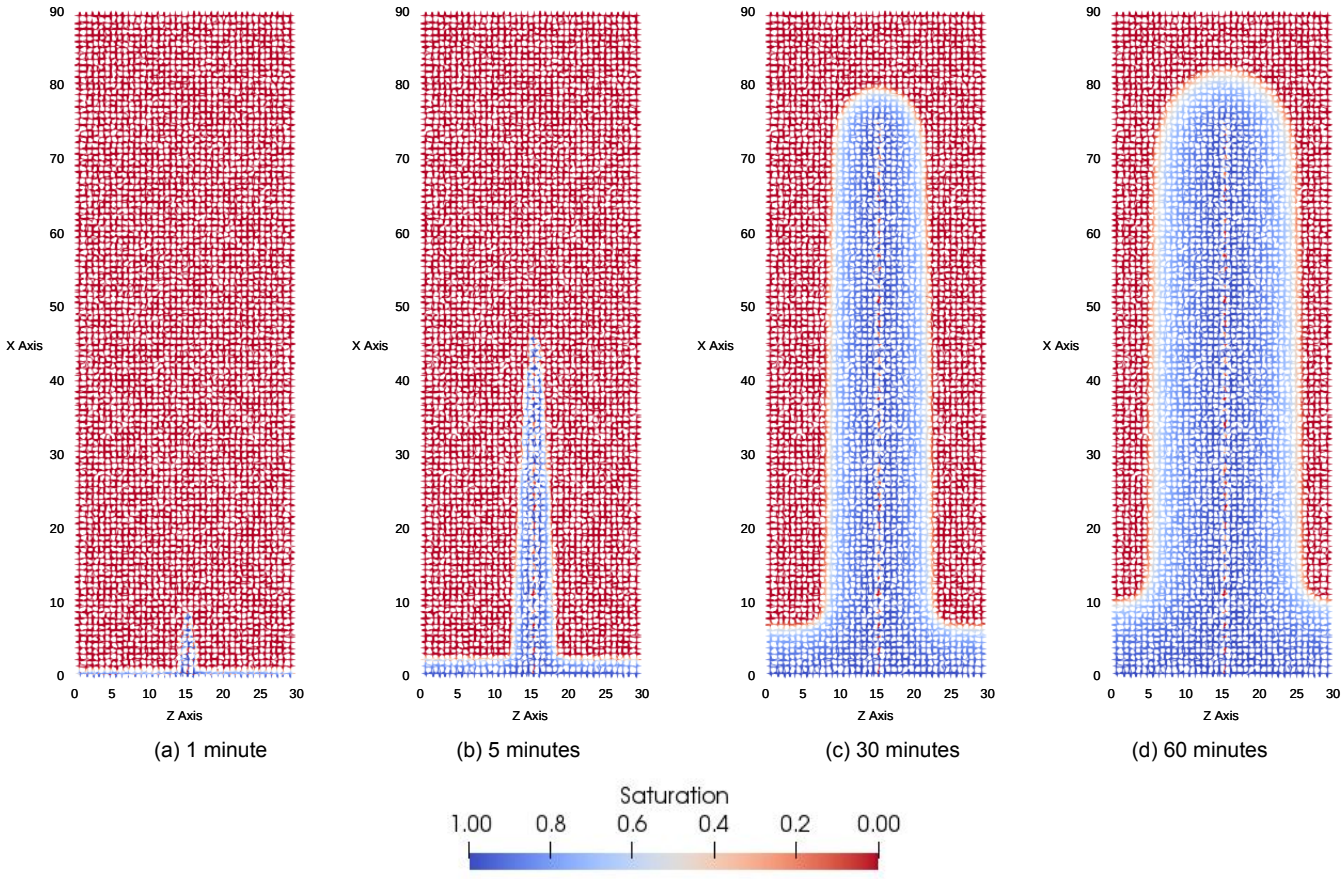
Numerical analysis considering crack as a porous phase in mortar

In the case when the crack is considered as a porous phase in mortar, the diffusivity of the Voronoi elements lying in the crack is defined in the same way as was done for the case of planar crack using

the equation:

$$D_w^{cr}(\theta) = \left(\sum_{i=1}^{3} w_i^3 l_i\right) \frac{\xi p_r (1-m)}{12\mu A m} \sqrt{\theta}[1-(1-\theta^{1/m})^m]^2 \theta^{-1/m-1}(\theta^{-1/m}-1)^{-m} \tag{4.10}$$

The difference with the planar crack is that in this case, the crack width is not uniform for the given crack depth but varying linearly with it. As the diffusivity of the cracked elements depend on the cube of the crack width, the diffusivity of the elements will decrease with increase of crack-depth (as the crack-width decreases with the crack-depth). The moisture transport in the uncracked concrete is based on the flow parameters as given in Table4.4.



(a) 1 minute (b) 5 minutes (c) 30 minutes (d) 60 minutes

Figure 4.14: Vertical moisture distribution in the crack considering crack as a porous phase in mortar

The results of the numerical analysis for the time stages of 1 , 5 , 30 and 60 minutes are shown in Figure4.14. It can be seen that the shape of the water front around the crack for later time stages of 30 and 60 minutes is similar to that obtained in the experiment but the shape of the water front around the crack for initial time stages of 1 and 5 minutes is different as the crack seems to be completely filled up in the experiment as contrary to the numerical results. The diffusivity of the cracked elements play a role in this. The diffusivity of the cracked elements is defined through the cubic law as given in eq(4.10). For the initial time stages, when the crack is not completely saturated, the diffusivity of the cracked elements depend on the crack width. As seen in Figure4.14a and Figure4.14b, the shape of moisture profiles for 1 and 5 minutes is such that the water front is wider at the bottom of the crack and it becomes narrower with increase of crack depth. This is due to higher diffusivity of the cracked elements (which depend on the cube of crack width) at the bottom than at the top. For the later time stages, the crack becomes completely saturated. At this point, the Voronoi elements lying in the crack have both

nodes which are completely saturated and because of this, there is no change in the saturation level within a cracked element. According to the cubic law, the diffusivity of the cracked elements depend on the gradient of the capillary pressure with respect to the saturation level given as:

$$D_w^{cr}(\theta) = -K_{cr}(\theta)\frac{dp_c}{d\theta} \tag{4.11}$$

where, K_{cr} is the permeability of the crack and p_c is the capillary pressure. As the moisture content remains constant in an element when it is completely saturated, the gradient of the capillary pressure with respect to the saturation level becomes zero and due to this, the diffusivity of the cracked element also becomes zero and the element remains in a completely saturated state for the rest of the flow. That is why the effect of the crack width on the diffusivity of the cracked elements is not observed for the later time stages as the crack is already completely saturated.

Also, it can be seen in Figure4.14, that for all time stages, there are nodes within the crack that seem to be in a completely unsaturated state (Voronoi nodes in red representing unsaturated state can be seen to lie in the crack). The saturation level of the nodes alternated between being completely saturated and being unsaturated. Ideally, all the nodes in the crack should always be in completely saturated state once the crack is filled up completely with moisture. But that does not appear to happen in the numerical model. The reason behind this is the approximation considered in defining the diffusivity of the cracked elements. As the uniform mean diffusivity is considered instead of the actual diffusivity and the transport equation is solved using an explicit time stepping technique, the mass and momentum balance of the transport equation does not hold for each time step and as a result the flow equilibrium is disrupted and moisture can accumulate in certain nodes and make other nodes completely unsaturated. This can be solved if an implicit time stepping technique is used with the actual variation of the diffusivity of the cracked elements instead of the uniform mean diffusivity. But the diffusivity in the cracked elements reach a considerably high magnitude and if the actual diffusivity is used coupled with an implicit time stepping procedure, the number of iterations required to attain equilibrium even for one time step is really high which considerably increases the computation time of the solver.

Numerical analysis considering crack and moisture surface interactions

The crack considered in this case does not has a uniform crack-width along it's depth but it is varying linearly. For this case, the capillary suction in the crack is modelled using the discretized form of the modified Lucas-Washburn equation:

$$\frac{2}{\Delta t}z_{n+1}^2+\left(\frac{2}{\Delta t}k_1k_2-\frac{2}{\Delta t}z_n-\frac{k_2k_3-k_2k_4z_n}{k_1k_2+z_n}+k_2k_4\right)z_{n+1}-\left(\frac{2}{\Delta t}k_1k_2z_n+\frac{k_1k_2(k_2k_3-k_2k_4z_n)}{k_1k_2+z_n}+k_2k_3\right)=0 \tag{4.12}$$

where

$$k_1 = \frac{2\beta_m}{r}$$

$$k_2 = \frac{r\beta_w}{2}+\frac{r^2}{8\mu}$$

$$k_3 = p_{c0}(1-\beta_s)$$

$$k_4 = \rho g sin(\phi)$$

The crack is divided into a number of discrete elements and the crack width for each discrete element is considered to be constant. In this case, a triangular crack is considered and due to capillary suction in the crack, it becomes completely saturated and this whole process takes less than 10 seconds. As a result, by the time the moisture starts spreading into the surrounding concrete, the crack is already completely saturated and the crack faces act as additional submerged boundaries.

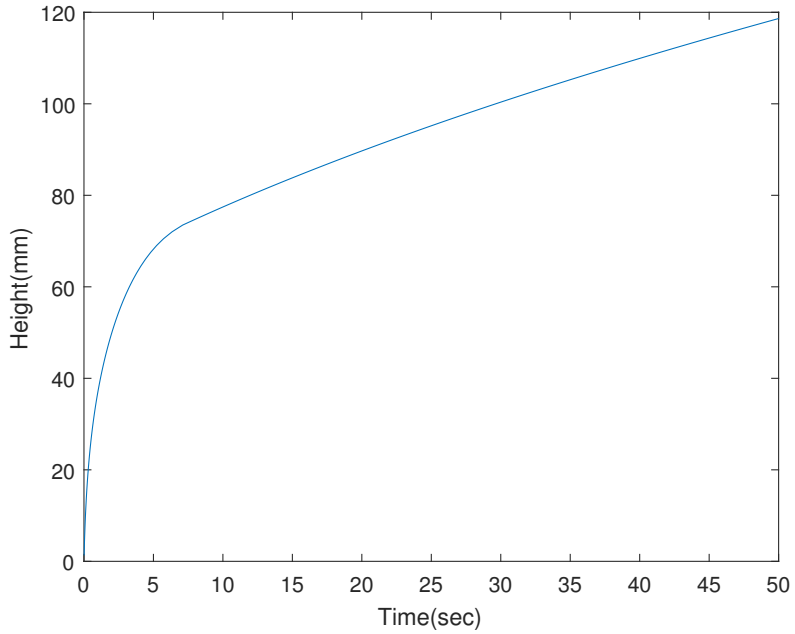
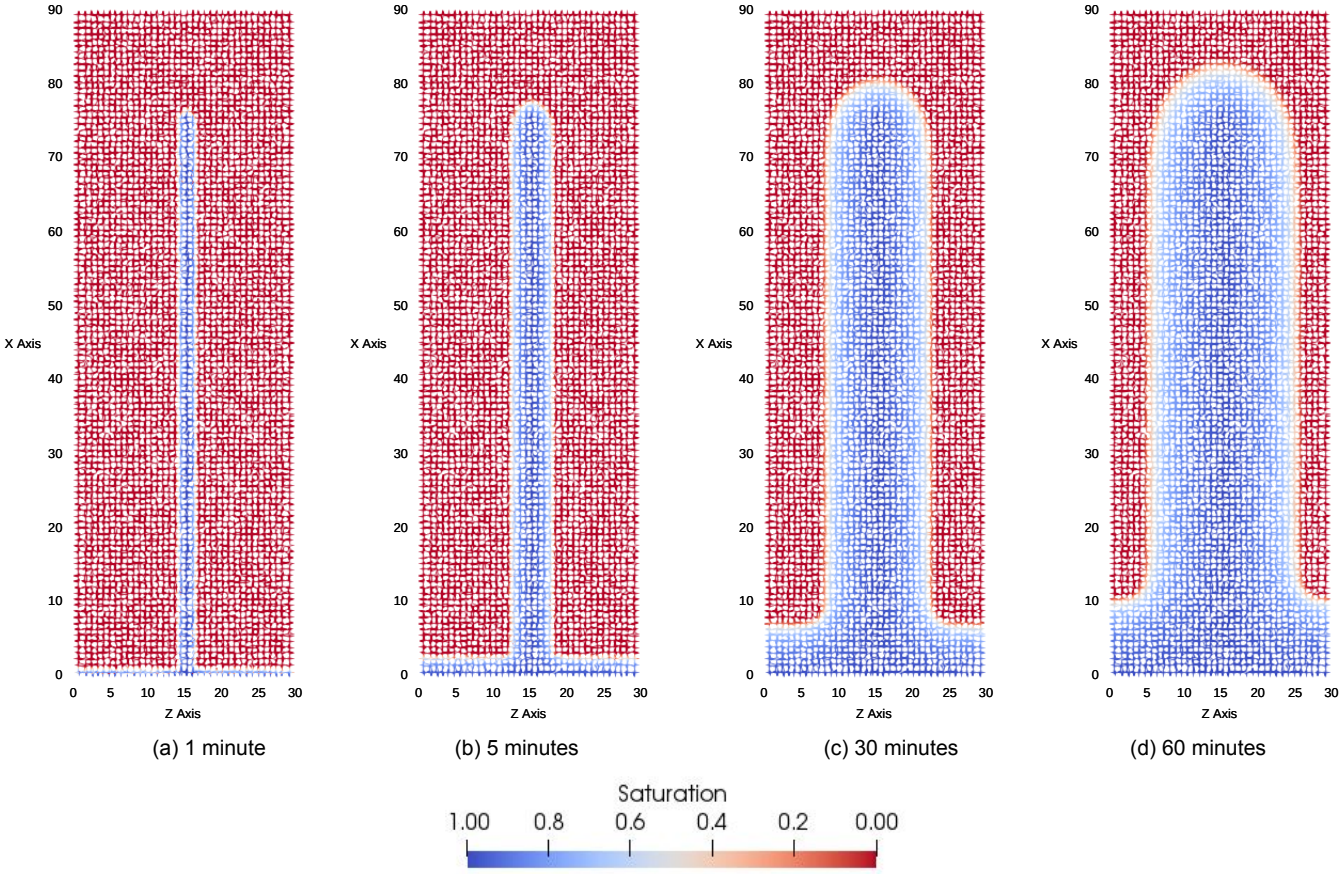


Figure 4.15: Capillary rise in the crack

Figure4.15 shows the rise of the capillary in the crack with respect to time. In case of the triangular crack, an equilibrium height is not attained where the capillary rise stops, but instead the capillary keeps on rising. The reason is that, the capillary rise attains an equilibrium when the interfacial forces are balanced by the inertial forces of the liquid but as the width of the capillary reduces, the interfacial forces also increases which further causes the rise in the capillary. When the capillary rise reaches the tip of the crack, the crack width is zero and when this is modelled through the discretized equation, the capillary keeps on rising to infinity. But as the crack has a depth of $75\ mm$, the crack gets completely filled up and the crack faces act as additional submerged boundaries.



(a) 1 minute (b) 5 minutes (c) 30 minutes (d) 60 minutes

Figure 4.16: Vertical moisture distribution in the crack considering crack and moisture surface interactions

The results of the numerical analysis for the time stages of 1 , 5 , 30 and 60 minutes are shown in Figure4.16. It can be seen that the moisture profile around the crack even for initial time stages of 1 and 5 minutes is similar to the one obtained in the experiment. For the later time stages of 30 and 60 minutes, the moisture profile is similar to the one obtained in the numerical model when the crack was considered as a porous phase and also in the experiment. Although the penetration of the moisture in the surrounding concrete is more for the case when capillary suction is considered as the crack is already completely saturated when the flow in the adjoining matrix starts. This method gives a better qualitative conformation with the experimental results. Also, as the flow in the crack is modelled separately using the discrete capillary elements, all the nodes in the crack are saturated as opposed to what was obtained when the crack was considered as a porous phase. In this particular approach, the explicit time stepping technique does not make any difference in the moisture transport occurring in the crack.

4.4. Results and Conclusions

In this section, firstly a comparison of the Delaunay and the Voronoi discretization is done. The difference in the modelling approach when the transport elements are modelled on the Delaunay and on the Voronoi elements is discussed. This is followed by a moisture transport simulation in a crack when the crack depth is greater than the equilibrium rise height of the capillary. A combined model, implementing the capillary suction in the crack and also treating it as a porous phase in mortar, is used to simulate the moisture transport. In the end, a discussion of the results is presented and the two approaches to model the moisture distribution in the crack are compared.

4.4.1. Comparison of Voronoi and Delaunay discretization

The structural properties of the concrete domain were modelled on the Delaunay elements and the transport properties of the domain were modelled on the Voronoi elements. When the moisture transport has to be modelled in uncracked concrete, the Delaunay elements can also serve to model the transport behaviour of the domain. The purpose of using the dual mesh is not highlighted in the uncracked concrete. But when the concrete cracks, the role of the dual mesh comes into play.

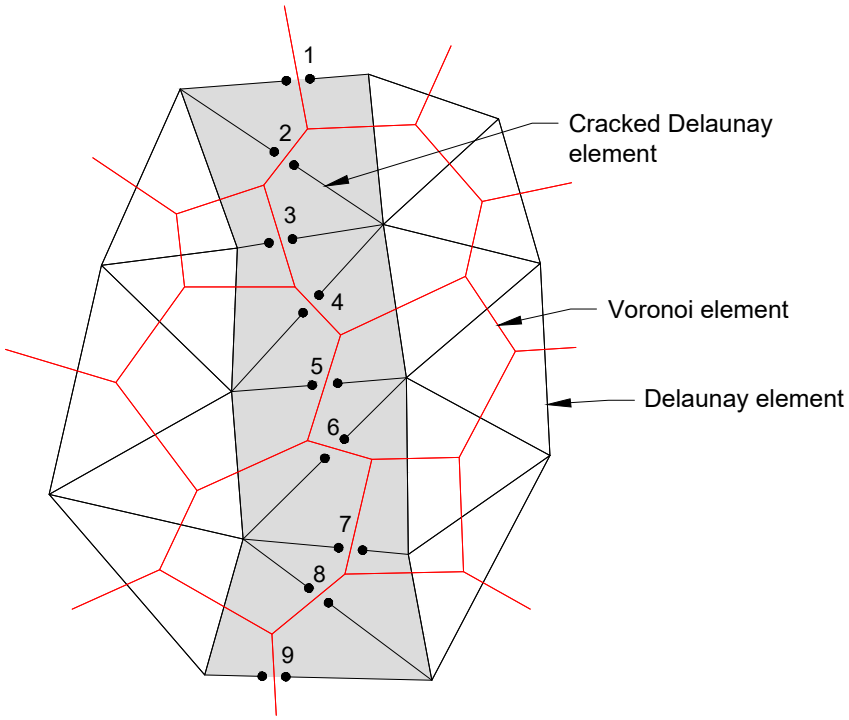


Figure 4.17: Delaunay elements used to model the transport behaviour of the cracked domain

Firstly, the case is considered when both the structural and the transport properties are modelled on the Delaunay elements as shown in Figure4.17. The cross-section of the Delaunay elements crack once the elements reach their tensile limit under the applied load. In the figure, the cracked Delaunay elements are numbered from 1-9. It can be seen that the direction of propagation of crack is perpendicular to the orientation of the Delaunay elements and crack width is along the direction of Delaunay elements. There is a change in transport properties of the cracked Delaunay elements at the location of the crack. The saturation level in an element will suddenly rise at the location of the crack. The transport properties of the cracked Delaunay elements do not remain uniform throughout the length of the element but changes at the crack location.

The change in the transport properties of the Delaunay elements can be included in the numerical model by incorporating discontinuities in the shape functions of the saturation level of the cracked ele-

ments. Discontinuities can be included at the position of the crack and the sudden rise of the saturation level within an element, at the position of the crack, can be modelled through that. Such a numerical model is presented in [2] where embedded discontinuities are incorporated in the lattice elements to model the diffusion of chloride ions. Incorporating the discontinuities for the cracked elements can increase the complexity to construct a numerical model to simulate moisture transport in cracked concrete. This can be circumvented by using the dual mesh in which the transport behaviour is modelled on the Voronoi elements rather than on the Delaunay elements.

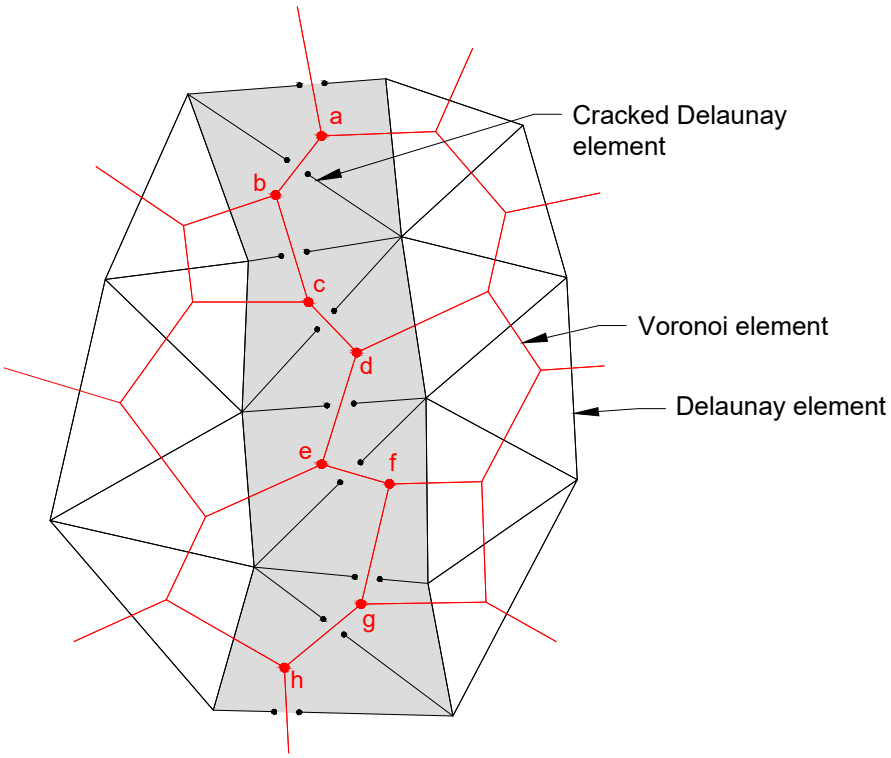


Figure 4.18: Voronoi elements used to model the transport behaviour of the cracked domain

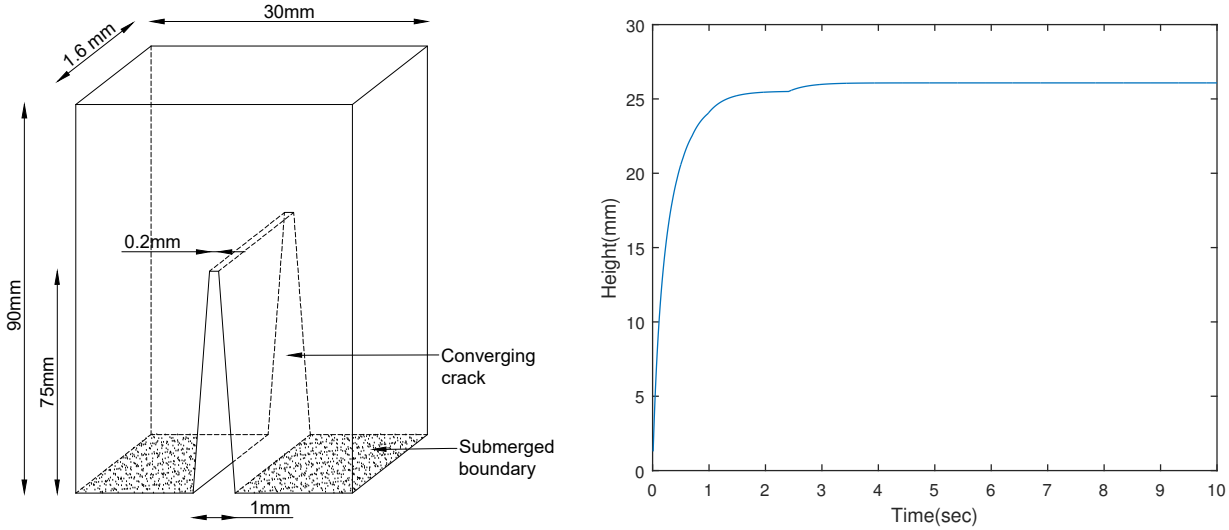
It can be seen in Figure4.18 that when the Delaunay elements are cracked, the direction of propagation of the crack is along the direction of orientation of the Voronoi elements and the crack width is aligned perpendicular to the Voronoi elements. The transport properties of a Voronoi element is determined by the Delaunay element it passes through. For the Voronoi elements passing through uncracked Delaunay elements, the transport properties represent the transport of the uncracked part of the domain. The transport in the cracked part of the domain is simulated through the Voronoi elements passing through the cracked Delaunay elements, marked from a-h in the figure. These Voronoi elements lie completely in the cracked part of the domain and there is no sudden change in the saturation level within the element. Unlike the Delaunay elements, each Voronoi element either represents the cracked or the uncracked part but never both. As a result, the discontinuities need not be modelled in this case. There can be an accumulation of moisture at the nodes which form a part of both the cracked and the uncracked Voronoi elements (nodes a-h in Figure4.18), but due to the mass and momentum balance of the transport equations, these nodes never exceed the fully saturated condition and rest of the moisture is transferred to the adjoining Voronoi nodes lying in the uncracked part of the domain.

An important thing to note is that to discretize a particular domain, the number of Delaunay elements required are less as compared to the number of Voronoi elements required to discretize the same domain. This reduces the computation time of the simulations in case the transport is simulated on the Delaunay elements in case the concrete is uncracked. In the uncracked state, there is no difference in the way transport is modelled using the Delaunay or the Voronoi elements and as a result Delaunay elements are preferred to model moisture transport in uncracked concrete. But in the cracked state, as

mentioned above, using the dual mesh and modelling the Delaunay elements as structural elements and Voronoi elements as transport elements account for a better modelling approach.

4.4.2. Crack-width Analysis

In the previous section, moisture transport in a triangle shaped crack was simulated but the crack width was small enough for the entire crack to get saturated when moisture-surface interactions were considered. But there can be a case when the crack width is large enough so that the entire crack do not get saturated and the crack depth is more than the equilibrium height of the capillary suction. In such a case capillary rise will be only limited to a fraction of the crack depth and rest of the crack surface cannot be considered as an additional submerged boundary. The moisture transport in the remaining part of the crack is simulated by considering it a porous phase in mortar and the diffusivity is obtained through the cubic law.



(a) Schematic of the cracked domain

(b) Capillary rise height in the crack

Figure 4.19: Schematic of the sample and the crack

The crack geometry as shown in Figure4.19a is used to simulate moisture transport. The crack width is decreasing from $1\ mm$ till $0.2\ mm$ over the crack depth giving it a trapezoidal shape. The rest of the mesh properties and the flow parameters are the same as given in Table4.3 and Table4.4 used for the case of vertical moisture distribution in Section4.3. The crack is divided into discrete elements and discretized form of the Lucas-Washburn equation gives the equilibrium height of the capillary rise as $26\ mm$ as shown in Figure4.19b. The crack surfaces only till a depth of $26\ mm$ will be completely submerged under capillary suction and act as additional submerged boundaries. The transport of moisture in the crack from $26 - 75\ mm$ depth will be considering the crack as a porous phase and the diffusivity will be defined through the cubic law.

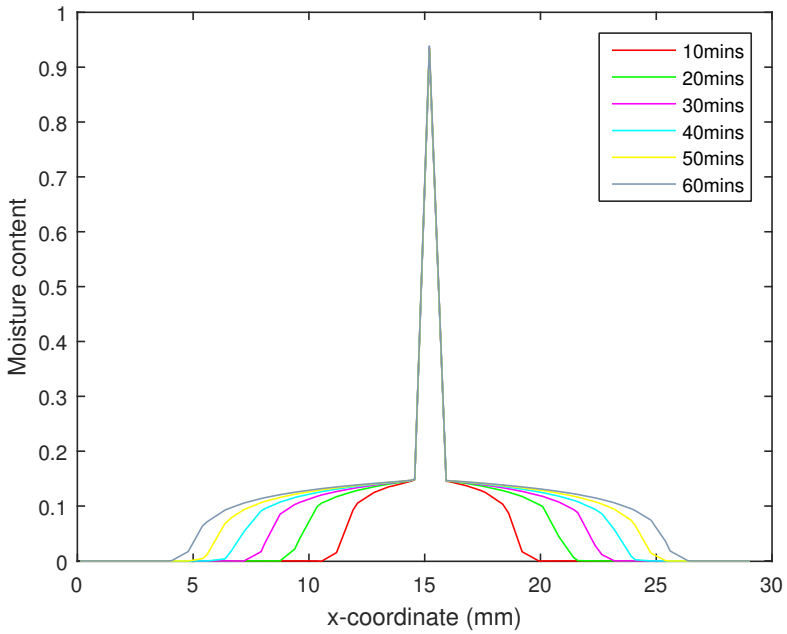


Figure 4.20: Horizontal moisture distribution in the crack

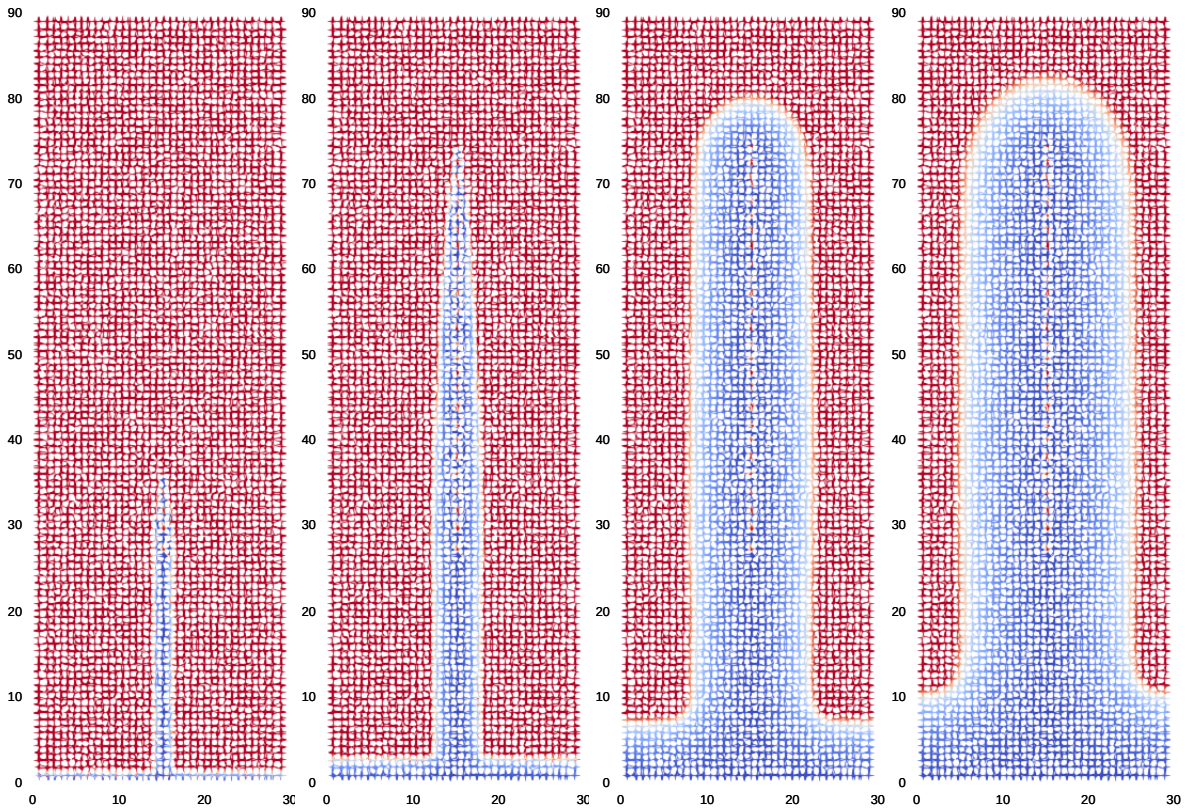


Figure 4.21: Vertical moisture distribution in the crack when the crack depth is more than equilibrium rise height of the capillary

The horizontal and the vertical moisture distribution of the crack is shown in Figure4.20 and Figure4.21 respectively. The crack is only partially saturated due to capillary suction and its effect is shown in the vertical moisture distribution for the initial time stages of 1 and 5 minutes (Figure4.21a,Figure4.21b). Till the crack depth of $26\ mm$, the water penetration into the surrounding matrix is uniform but for the depth between $26 - 75\ mm$, the water penetration is more where the crack width is more and reduces as the crack width decreases, showing the dependence of the diffusivity of the crack on the crack width. Once the crack becomes saturated, the penetration of water from the crack to the adjoining matrix is uniform. The horizontal moisture distribution and the vertical moisture distribution for the later time stages is similar to the ones obtained in the numerical models used to simulate the experiments.

4.4.3. Conclusions

When the crack is considered as a porous phase in mortar in the numerical model, the horizontal moisture distribution around the crack conforms well with the experimental results. The penetration depth of the moisture and the shape of the moisture distribution curves are close to the experimental observations although not exactly similar which is due to the homogeneous material and uniform crack depth assumed in the numerical model. Nevertheless, when the crack is considered as a porous phase, the vertical moisture distribution around the crack is similar to the experimental results for later time stages but different for the initial time stages. This is due to the diffusivity of the cracked elements which is dependent on the crack width. Also, in this case, it takes comparatively more time for the crack to get completely saturated than in the case of capillary suction.

When the moisture-surface interactions are considered, the horizontal moisture distribution is similar to the case when crack is considered as a porous phase. The penetration depth of water is more at any particular time stage because the crack immediately fills up and subsequently starts dispersing water to the adjacent cement matrix. The vertical moisture distribution shows good agreement with the experimental results for both the initial and the later time stages as the crack is completely saturated as observed in the experiment.

Including the capillary suction in the crack gives better results for the vertical moisture distribution during the initial stages of the flow. There is no significant difference in the horizontal moisture distribution especially in the later stages. Also, the approximation of the uniform mean diffusivity coupled with explicit time stepping technique, in case the crack is a porous phase in mortar, causes some of the nodes lying in the crack to be in an unsaturated state which is not representative of the physical condition of the crack. But when the capillary suction is considered in the crack and in the case when the crack depth is less than the equilibrium rise height, all the nodes in the crack are at a constant state of complete saturation.

5

Moisture Transport in Heterogeneous Concrete

Until now, the moisture transport in homogeneous concrete was observed but in reality, the concrete microstructure is highly heterogeneous and it needs to be analyzed as a multi-scale composite material where the transport in the microstructure is realistically simulated. The presence of aggregates in concrete affects the moisture distribution in the material. Concrete acts as a three-phase material composed of cement matrix, aggregates and the interfacial transition zone (ITZ) with different transport properties of each phase as shown in Figure5.1. In most cases, the coarse aggregates are considered to be virtually impermeable and no transport of moisture occurs through them. The transport of moisture through the cement matrix under capillary absorption is already studied in the previous chapters. The packing of the cement particles is not as dense at the aggregate boundaries as in the cement matrix and as a result, the permeability of the region around the aggregates is higher than in the cement matrix [26]. This increases the porosity of the region surrounding the aggregates higher than in the cement matrix. This causes the acceleration of flow in the ITZ and to simulate this in the numerical model, ITZ is assigned higher diffusivity than the cement matrix.

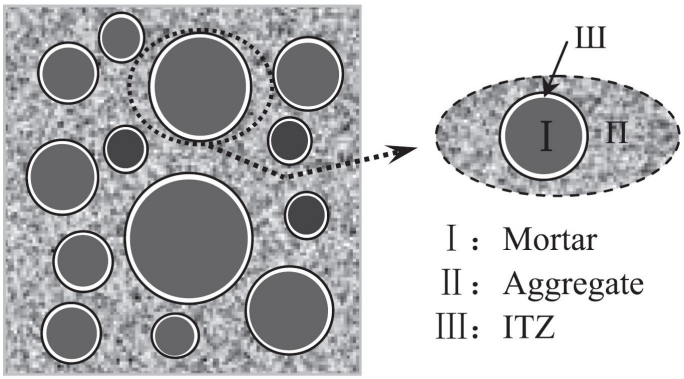


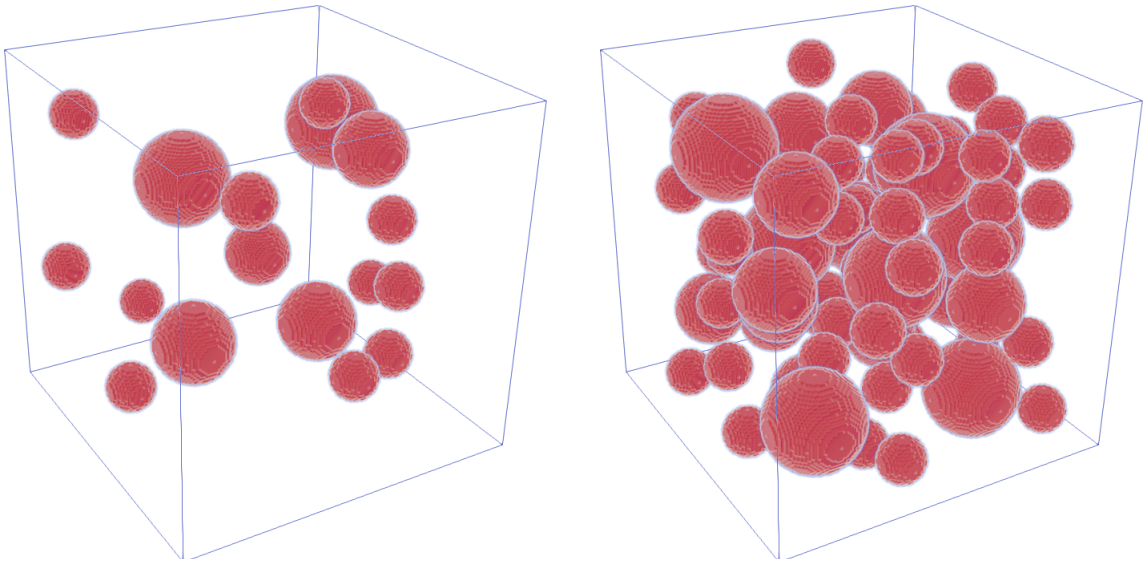
Figure 5.1: Mesoscale structure of concrete [48]

5.1. Model Discretization

In order to simulate the effect of aggregates on the transport of moisture, the aggregates have to be included in the discretized mesh. In the present numerical model, for a given volumetric fraction of the domain, the aggregates are generated according to Fuller's distribution [19]. All Voronoi elements in the domain are categorized to either represent the cement matrix, aggregates or the ITZ. The aggregate boundaries along with their interface zone are defined in the discretized domain and the transport properties of Voronoi elements lying in a particular phase are assigned accordingly.

5.1.1. Generation of Aggregates

To evaluate the transport behaviour of heterogeneous concrete at meso-scale, random aggregates have to be generated in which the size and distribution of aggregates resemble real concrete. For the sake of simplicity, the aggregates that are generated are spherical in shape with different diameters. Aggregate generation code as developed by [40], is used to generate spherical aggregates in this research. The generation of the random aggregates and their placement must satisfy the basic statistical characteristics of the real material. The main principle behind the generation and placement process is that samples of aggregate particles are chosen from a source which contains aggregates of different sizes following a certain grading curve and the chosen aggregates are placed one by one in the domain such that there is no overlap with the already placed particles [52]. This process is commonly known as take and place method and the Fuller's curve is used for grading the particles according to their size. Coarse aggregates are considered particles greater than 4.75 mm in diameter and they represent around 40-50 percent of the concrete volume [52]. Considering spherical shape, the coarse aggregates are generated using take and place method and their size distribution follows the Fuller's curve. Figure5.2 shows the generation of aggregates for a total aggregate volume fraction of 5 percent and 20 percent respectively using the take and place method.

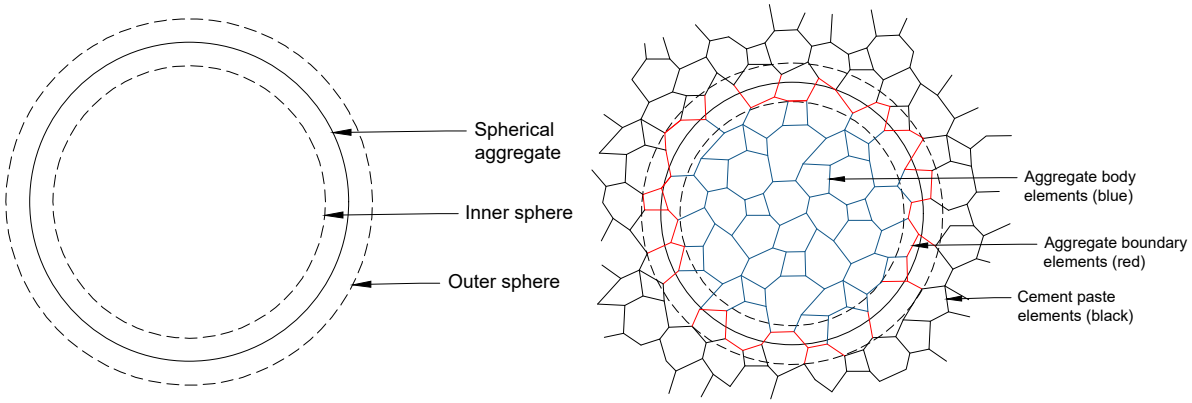


(a) 5 percent volume of aggregates (b) 20 percent volume of aggregates

Figure 5.2: Randomly generated aggregates

5.1.2. Projecting Aggregates on Voronoi Elements

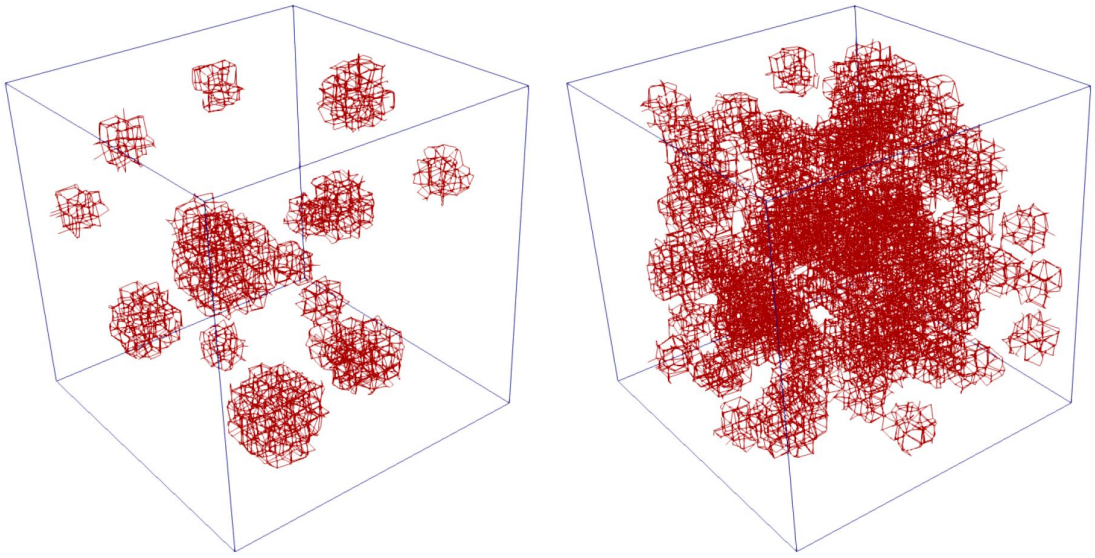
Once the size and position of the aggregates is determined, they have to be included in the discretized domain. The boundary of the aggregates is projected on Voronoi elements in such a way that the Voronoi elements lie along the direction of the boundary of the aggregates. The procedure adopted to project a single spherical aggregate is shown in Figure5.3. For any given spherical aggregate, two concentric spheres are constructed, one with a smaller radius (inner sphere) and the other with a larger radius (outer sphere) than that of the spherical aggregate as shown in Figure5.3a. The outer sphere is constructed with a radius obtained by adding the radius of the spherical aggregate and half of the mean length of all the Voronoi elements lying in the domain. Similarly, the inner sphere is constructed with a radius obtained by subtracting half of the mean length of all the Voronoi elements from the radius of the spherical aggregate.



(a) Spherical aggregate (b) Voronoi elements in the spherical aggregate

Figure 5.3: Projection of spherical aggregates on Voronoi elements

After generating the two spheres, Voronoi elements are projected on the spherical aggregate. All Voronoi elements having both of their nodes inside the inner sphere or one node inside the inner sphere and the other node outside the inner sphere, constitute the body of the aggregate (shown with blue colour in Figure5.3b). All Voronoi elements having both of their nodes outside the inner sphere and inside the outer sphere constitute the boundary elements of the aggregate (shown with red colour in Figure5.3b). Thickness between the inner and outer spheres is equal to the mean length of the Voronoi elements so that, as far as possible, the boundary layer does not contain more than one Voronoi element in the radial direction. This process is carried out for all the aggregates and in this way the spherical aggregates are projected on the Voronoi elements and become a part of the discretized domain. Figure5.4 shows spherical aggregates as projected on the Voronoi elements for a volume fraction of 5 and 20 percent respectively. It can be seen in the figure that, as the domain is not continuous but discretized through one dimension lattice elements, the spherical shape of the aggregates is not retained when they are projected on the Voronoi elements. Nevertheless, mapping aggregates on Voronoi elements in such a way causes the Voronoi elements lying at the boundary to realistically simulate the physical phenomenon of increased transport along the aggregate boundary.



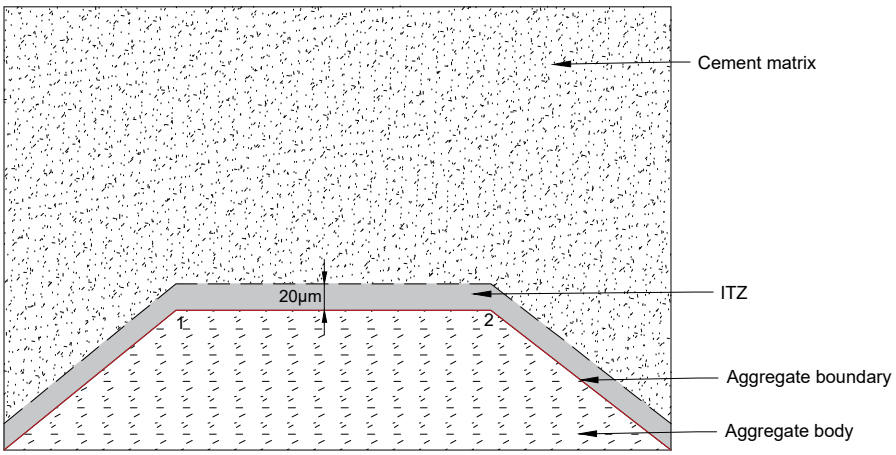
(a) 5 percent volume of aggregates (b) 20 percent volume of aggregates

Figure 5.4: Randomly generated aggregates

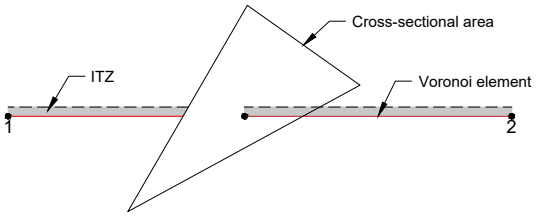
5.1.3. Assigning Transport Properties to Voronoi Elements

Once the spherical aggregates are generated in the domain and projected on the Voronoi elements, the transport properties of Voronoi elements have to be assigned according to the phase of concrete they represent. The Voronoi elements that lie completely within the aggregate boundary are assigned zero diffusivity as the aggregates are considered impermeable. The Voronoi elements lying within the cement paste are assigned the initial diffusivity of dry cement (D_0). The Voronoi elements that form the boundary of the aggregates and represent the ITZ are assigned 10 times the diffusivity of the cement paste i.e. $10D_0$. This is because the density of cement paste is lower at the boundary surface than in the matrix due to wall effect [26] and the permeability of the region around the aggregates increases.

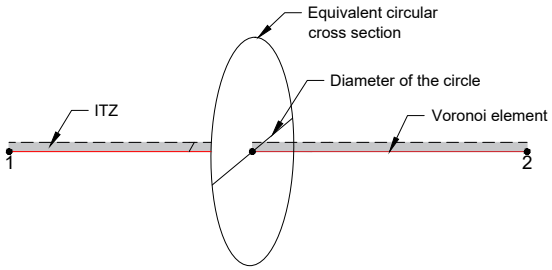
The thickness of the ITZ is considered to be $20\mu m$ in the numerical model [19] which means that the ITZ forms a region of $20\mu m$ around the aggregate boundary where the diffusivity is enhanced as shown in Figure5.5a. The Voronoi elements that form the boundary of the aggregates and represent the ITZ are so large that their cross sectional area extend beyond the thickness of the ITZ as shown in Figure5.5b. When such Voronoi elements are assigned higher diffusivity of the ITZ, the effect of higher diffusivity extend well beyond the thickness of the ITZ in the numerical model whereas in reality, the enhanced diffusivity should only be limited to the ITZ. This can cause a net acceleration of the transport around the aggregates which can be more than what is observed experimentally. The mesh has to be extremely refined so that the Voronoi elements that lie on the boundary of the aggregates are small enough to have cross sectional area that does not extend beyond the thickness of the ITZ. Having a refined mesh to such an extent will increase the computational cost drastically.



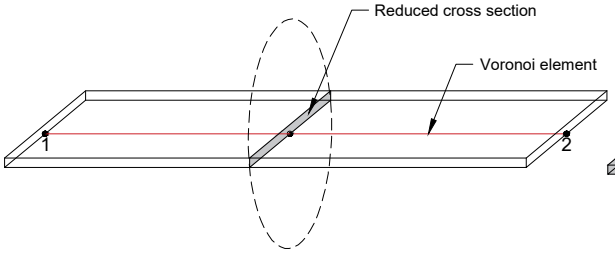
(a) Concrete as a three phase composite



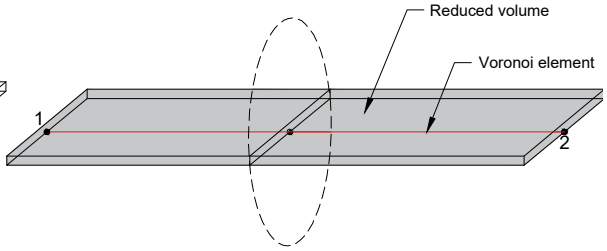
(b) Actual cross section of a boundary Voronoi element



(c) Equivalent circular cross section



(d) Reduced cross-sectional area



(e) Reduced volume

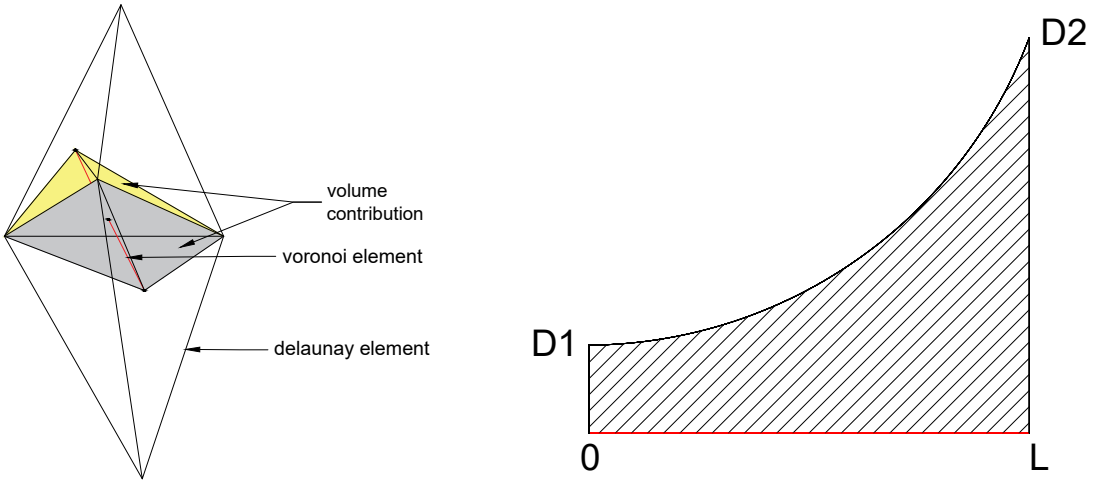
Figure 5.5: Transport properties of boundary Voronoi elements

Instead, the Voronoi elements lying on the boundary of the aggregates are manually assigned the cross sectional area. The thickness of ITZ is known but the width of the Voronoi element over which the ITZ effect acts is still unknown. This is given by the triangular cross section of the Voronoi elements. By knowing the orientation of the boundary Voronoi element, the part of the cross section lying outside the aggregate can be used to give the width over which ITZ effect will act. But all the boundary Voronoi elements are oriented differently with different cross sections and to find the affected width of the boundary elements can become complex. Instead, the approach adopted in this research is that, for every Voronoi element lying on the boundary, an equivalent circular cross section is formed which has the same area as the original cross section of the Voronoi element as shown in Figure5.5c. The diameter of this circular cross section is considered to be the width over which ITZ effect acts. Using the thickness of ITZ and the diameter of the equivalent circular cross section, the modified rectangular cross section of the boundary Voronoi element can be formed as shown in Figure5.5d. This reduced rectangular cross section acts over the entire length of the boundary Voronoi element which gives the reduced volume of the element as shown in Figure5.5e. In this way, the mesh only has to be

refined enough so that the shape of the aggregates is projected without significant error and the affect of higher diffusivity of the Voronoi elements lying on the boundary of the aggregates is limited to the physical thickness of the ITZ.

5.1.4. Properties of Numerical Solver

Now that the aggregates are included in the discretized domain and the corresponding transport properties are assigned to the Voronoi elements, the numerical solver used to solve the discretized equation has to be set up. In case of uncracked concrete, the model using approximate volume, explicit time stepping scheme and uniform mean diffusivity was able to accurately simulate the moisture transport in concrete. When aggregates are included in concrete, the transport properties of the domain are not homogeneous and a numerical model using exact volume, implicit time stepping scheme and exponential variation of diffusivity is used as shown in Figure5.6.



(a) Exact volume

(b) Exponential variation of diffusivity

Figure 5.6: Properties of the numerical solver

The Voronoi elements lying on the boundary of the aggregates, representing the ITZ, have higher diffusivity as compared to the elements lying in the cement matrix. The Voronoi elements lying within the aggregates have zero diffusivity as the aggregates are considered impermeable. A numerical solver using uniform mean diffusivity and explicit time stepping scheme won't be able to maintain the equilibrium of flow as the properties are varying and the flow is irregular in the domain. Using implicit time stepping scheme and exponential variation of diffusivity (Figure5.6b) will ensure that the mass and momentum balance of the flow is maintained at each time step regardless of the highly variable transport properties of the Voronoi elements.

The Voronoi elements lying on the boundary of the aggregates are assigned a much smaller cross-sectional area than their actual geometric cross sectional area. This drastically changes the volumetric capacity of these Voronoi elements which affects the volumetric capacity matrix $\mathbf{M}$ in the discretized equation. Using the approximate volume of the Voronoi elements can cause an inaccurate prediction of the volumetric capacity of the Voronoi elements lying on the boundary and the adjacent Voronoi elements lying in the cement matrix. To avoid this, the exact volume of the Voronoi elements is considered as shown in Figure5.6a. Hence, a numerical solver using the exact volume, implicit time stepping scheme and exponential variation of diffusivity is set up to simulate moisture transport in heterogeneous concrete.

5.2. Numerical Analysis of Moisture Transport in Heterogeneous Concrete

Till now, the aggregates are included in the discretized domain and Voronoi elements are assigned transport properties according to the phase of concrete they represent. A numerical model based on the exact volume of the Voronoi elements, implicit time stepping technique and exponential variation of diffusivity will be used to simulate moisture transport in the three phase composite. In this section, firstly the experiment in [19] is described which will be used to set up the numerical model followed by the results obtained by the numerical model and subsequent comparison of the numerical and experimental results.

5.2.1. Experiment

The experiment used in [19] is used as a reference to simulate moisture transport in heterogeneous concrete in the numerical model. The mortar mixture is made with Ordinary Portland cement (OPC, Grade P.O 42.5) with water-cement ratio of 0.45 and sand-cement ratio of 2. The density of the mortar is $2200\ kg/m^3$ and that of the coarse aggregates is $2630\ kg/m^3$. The coarse aggregates are of spheroidal shape with diameter ranging from $D_{min} = 5mm$ to $D_{max} = 40mm$. The aggregates are graded based on their size according to Fuller's curve. The concrete specimens used are of frustum-shaped, with base diameter of $185\ mm$, top diameter of $175\ mm$ and height of $150\ mm$ as shown in Figure5.7.

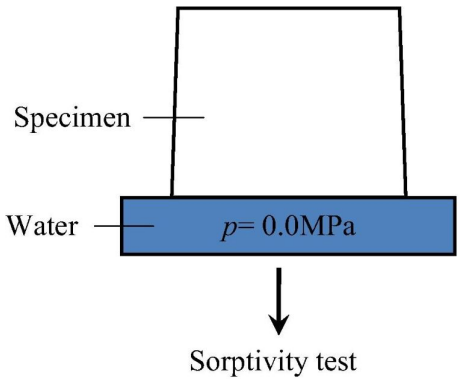


Figure 5.7: Specimen used in the experiment[19]

Sorptivity test is conducted on the specimens with different aggregate volume fraction of 0, 0.3, 0.4 and 0.5. In the experiment, concrete samples are also subjected to different hydrostatic pressures but only the results without considering the hydrostatic pressure and only the capillary absorption are used as a reference to simulate moisture transport in the numerical model. The experiment is conducted on a permeability test set-up using pure water. Uniform initial moisture content in the specimens is obtained by drying in an air oven at $105°C$ until they reach constant weight. The sides of the specimen are sealed and the bottom surface is submerged in water as shown in Figure5.7. To reflect the dynamic movement of water in concrete, cumulative volume of water penetration per unit area of the inflow surface is calculated at different time stages.

5.2.2. Numerical Analysis

Four discretized cubic domains of $150 \times 150 \times 150\ mm^3$ are generated with volume fraction of aggregates as 0, 0.3, 0.4 and 0.5 respectively. As there is a considerable difference of the transport properties across the different phases of concrete, to maintain flow equilibrium, numerical analysis is performed on the discretized domain using exact volumetric capacity, implicit time stepping scheme and exponential variation of diffusivity over the element length. The mesh properties and the flow parameters as given in Table(5.1) and Table(5.2) respectively are used to simulate moisture transport in the four discretized domains.

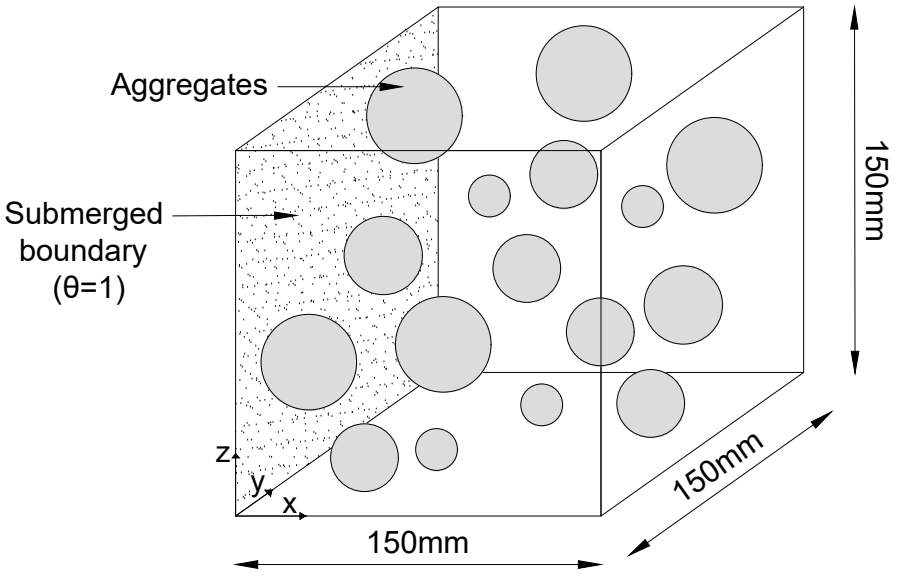


Figure 5.8: Schematic of the flow conditions in the heterogeneous domain

Table 5.1: Mesh Properties

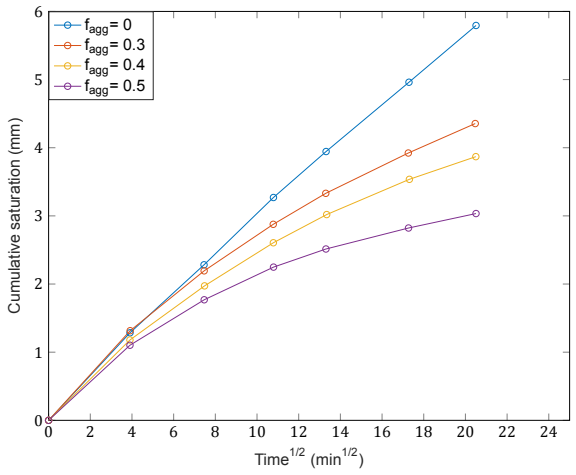
Parameter	Value
Number of nodes	50229
Number of elements	93960
Average element length	3.299 mm
Maximum aggregate diameter	40 mm
Minimum aggregate diameter	5 mm

Table 5.2: Flow Parameters

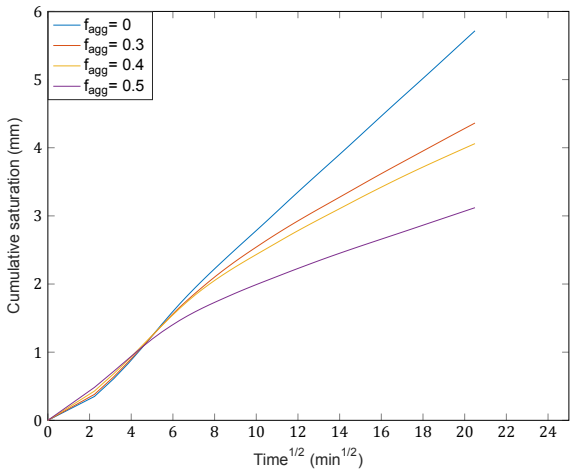
Parameter	Symbol	Value
Diffusivity of dry concrete [19]	D_0	$3.85 \times 10^{-4}\ mm^2/s$
Empirical constant	n	6
Initial saturation	θ_0	0
Time step	Δt	$60s$
Diffusivity of ITZ	D_{itz}	$3.85 \times 10^{-3}\ mm^2/s$
Diffusivity of aggregate	D_{agg}	0

Table 5.3: Volume Specification of Aggregates

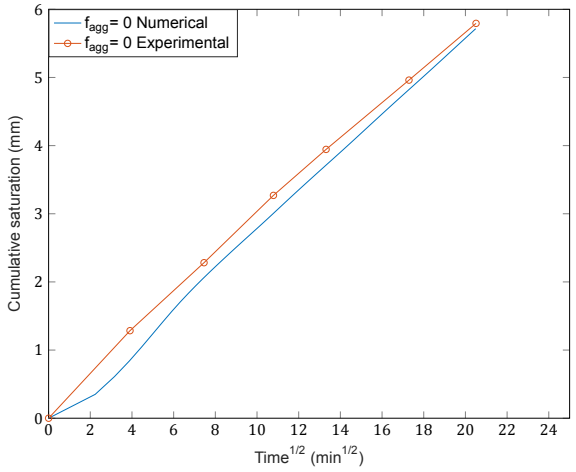
Volume fraction of aggregates	Number of aggregates	Spherical aggregates		Voronoi aggregates	
		Aggregate volume ($10^6\ mm^3$)	ITZ volume ($10^3\ mm^3$)	Aggregate volume ($10^6\ mm^3$)	ITZ volume ($10^3\ mm^3$)
0	0	0	0	0	0
0.3	143	1.018	4.584	1.025	3.852
0.4	454	1.354	7.196	1.342	5.854
0.5	1252	1.686	11.465	1.611	8.809



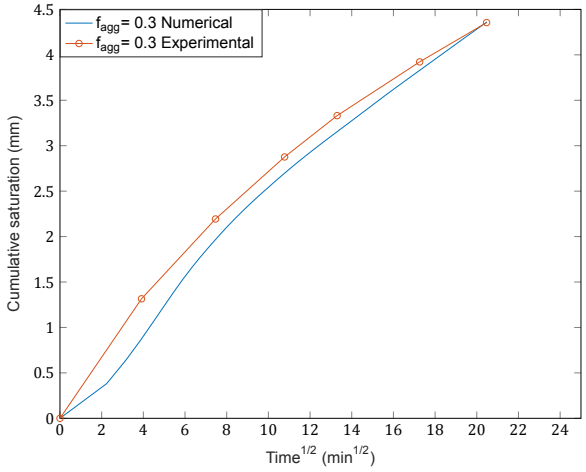
(a) Experimental Results



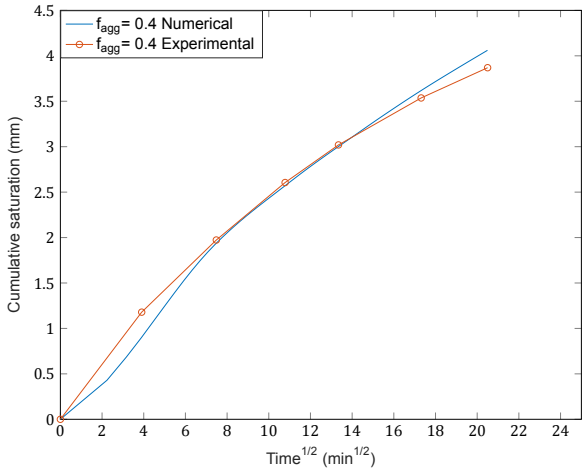
(b) Numerical Results



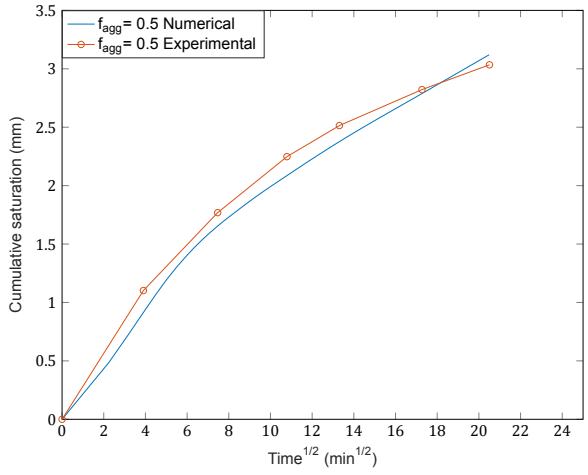
(c) $f_{agg} = 0$



(d) $f_{agg} = 0.3$



(e) $f_{agg} = 0.4$



(f) $f_{agg} = 0.5$

Figure 5.9: Variation of cumulative saturation with square root of time in concrete samples for a duration of 420 minutes

The results show the variation of cumulative saturation level with square root of time. At each time

stage, the saturation level of all the nodes is multiplied by the respective volume of the corresponding nodes which gives the saturated volume of each node. The saturated volumes of all the nodes is added to get the total saturated volume of the domain. The saturated volume of the domain is divided by the porosity of the material to get absolute volume under saturation of the domain. The absolute volume is divided by the cross-sectional area of the domain ($150 \times 150\ mm^2$) to get the cumulative saturation level which is presented in the figures. This quantity is also representative of the penetration depth of moisture in the domain. It can be seen in Figure5.9, that the numerical model is successfully able to simulate the behaviour of aggregates in concrete. In the numerical model, when the volume fraction of the aggregates is increased, the cumulative saturation level decreases (Figure5.9b) which is also observed in the experiment (Figure5.9a). Due to the presence of aggregates, there are two competing phenomena that affect the transport of moisture in concrete. The aggregates themselves are considered impervious and there is no diffusion of moisture through them. This slows down the transport of moisture in the matrix. Due to the presence of aggregates, there are ITZ formed around the aggregates which accelerates the transport of moisture. It can be seen from the results, that the effect of impermeability of the aggregates dominates over the ITZ acceleration effect and causes a net decrease of flow with increase of the volume fraction of the aggregates. In the numerical models considered here, there is no overlap of the aggregates in the matrix. If the volume fraction of the aggregates is increased to the extent that there is an overlap of the aggregates, ITZ can connect and increase the transport of moisture to a greater extent which can cause the ITZ effect to dominate and reverse the trend.

Although, the numerical model is able to reproduce the influence of aggregates in concrete, still there are some differences with the experimental results as seen in Figures5.9c-5.9f. The experiment is performed on a frustum shaped specimen in which the cross sectional area is changing with depth. In the numerical model, due to numerical constraints, a cubic domain of $150 \times 150 \times 150\ mm^3$ is considered with a slightly different volume than that of the experimental specimen.

The volume specifications for different aggregate fractions are shown in Table(5.3). It can be seen that for each volume fraction of aggregates, the total aggregate volume of the spherical aggregates (column 3) is quite close to the total aggregate volume of the Voronoi aggregates (column 5). This implies that the number of Voronoi elements that represent the body of the aggregates in the discretized domain will accurately be able to simulate the nature of the aggregate body in the domain. For each volume fraction of aggregates, the ITZ volume of the spherical aggregates (column 4) and ITZ volume of Voronoi aggregates (column 6) are a little different. This difference arises due to the modelling approach adopted to simulate the flow in the boundary elements. As explained before, the cross section of the boundary elements is reduced as per ITZ thickness and the width of the cross section is considered as the diameter of the equivalent circular cross section (Figure5.5). By considering this approximation, the total volume of ITZ in Voronoi aggregates is different than that of the spherical aggregates. To account for this difference in the ITZ volume, a correction to the volume of the boundary elements is applied which is given as:

$$\Delta V = \frac{V_{sph} - V_{vor}}{N} \tag{5.1}$$

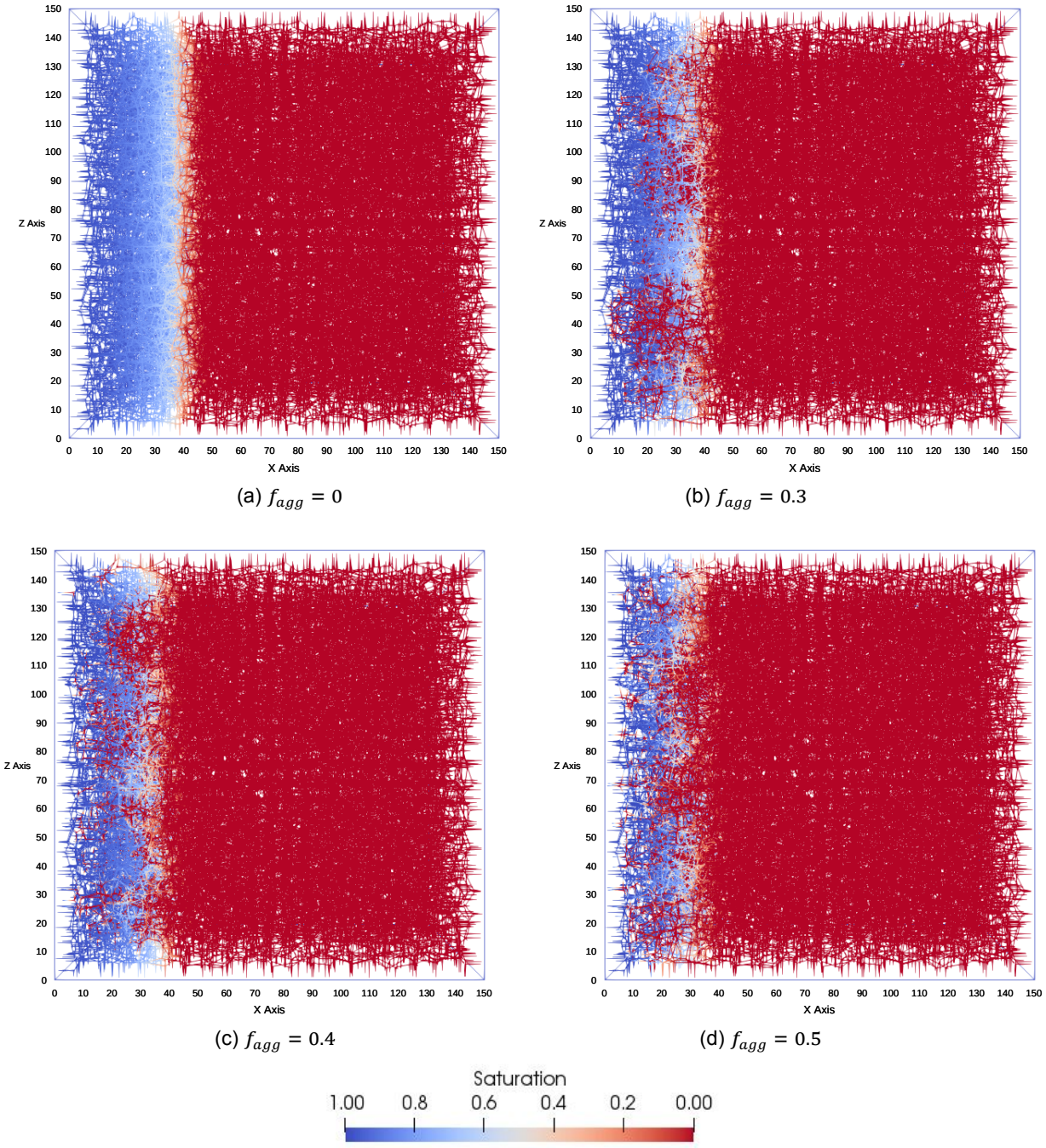
where,V_{sph} is the total ITZ volume of spherical aggregates, V_{vor} is the total ITZ volume of Voronoi aggregates, N is the total number of boundary elements and ΔV is the ITZ volume correction to be added to the volume of each boundary element. Subsequently, the cross sectional area of the boundary elements is also corrected to account for the difference of ITZ volume which is given as:

$$\Delta A_i = \frac{\Delta V}{L_i} \tag{5.2}$$

where, A_i is the correction in the cross sectional area of a boundary element i, ΔV is the ITZ volume correction and L_i is the length of the boundary element i. The correction for the cross sectional area is added to the cross section of the boundary elements. In this way, the corrections for the volume and cross sectional area accounts for the difference of ITZ volume in the spherical and Voronoi aggregates. Another way to account for the difference is to calculate the correction factors for each individual aggregate separately and apply the corrections to the Voronoi elements of only that particular aggregate. Doing this will give an even more accurate behaviour of the boundary elements in simulating the flow. But the increase in accuracy of this approach is not that significant as the increase in computation time

and increase of the modelling complexity of the compared to the first approach, because of which, the corrections using the first approach are considered.

Finally, the experiment only specifies the volume fraction of the aggregates in the different specimens but not the size distribution of the aggregates. The size distribution in the numerical model is randomly done according to Fuller's curve and it is not necessary that it might be the same as used in the experiment. This can change the transport behaviour because if the size distribution is such that there are more number of aggregates with smaller diameters, the ITZ effect of acceleration of transport will become more pronounced and cause an increased acceleration of the flow whereas on the other hand, if the size distribution is such that more number of aggregates with larger diameters are used, the effect of impermeability of aggregates will become more pronounced and cause the moisture transport to slow down even further.



(a) $f_{agg} = 0$ (b) $f_{agg} = 0.3$

(c) $f_{agg} = 0.4$ (d) $f_{agg} = 0.5$

Figure 5.10: The moisture distribution in concrete at 420 minutes.

Figure5.10 shows the moisture distribution in the discretized domain at 420 minutes. Figure5.10a shows the moisture distribution in homogeneous concrete. The moisture is evenly distributed in the submerged part of the domain and the water front forms a plane and progress towards the unsaturated part of the domain. Figures5.10b-Figure5.10d show the moisture distribution in heterogeneous concrete. The moisture distribution is not uniform and the effect of aggregates is clearly visible as the Voronoi nodes lying inside the aggregates are in an unsaturated state even if the water front has already passed over them. The water front is not completely planar and the water penetration is different in different parts of the domain due to variable magnitude of ITZ acceleration and aggregate impermeability effect in different parts of the domain. The net penetration depth at 420 minutes decreases with increase of the volume fraction of the aggregates as shown in the figures.

5.3. Results and Conclusions

Now that the numerical model is able to simulate the experimental results, it can be used to understand the nature of the flow occurring in heterogeneous concrete. There are two competing phenomena affecting the transport when aggregates are present in concrete. The ITZ tends to accelerate the moisture transport around the boundary of the aggregates and the aggregates themselves are considered impervious which slows down the moisture transport. But to what extent these phenomena play a role in moisture distribution is yet to be addressed which is done in the first subsection by varying the ITZ diffusion coefficient. In the second subsection, the effect of mesh refinement on the moisture penetration will be observed. In the third subsection, moisture transport is simulated in a combined model which incorporates a discrete planar crack in heterogeneous concrete. In the last subsection, the conclusions based on the results of the numerical model will be presented.

5.3.1. Effect of ITZ on Water Penetration Depth

To observe the effect of ITZ on moisture distribution in heterogeneous concrete, the diffusivity coefficient of the ITZ is varied and it's affect on the moisture penetration is observed. The numerical model of $f_{agg} = 0.4$ (Table(5.3)) is taken as the reference numerical model and all the results are compared with the reference model. In the reference mesh, the ITZ diffusivity is taken as $10D_0$. Three other numerical models are constructed with the same aggregate distribution as in the reference model with the ITZ diffusivity of $20D_0$, $5D_0$ and D_0. All the mesh properties and flow parameters are the same as given in Table(5.1) and Table(5.2) respectively. The numerical model is based on exact volume of the aggregates, implicit time stepping scheme and exponential variation of diffusivity and the moisture transport is simulated for a time period of 420 minutes and the results are shown in Figure5.11.

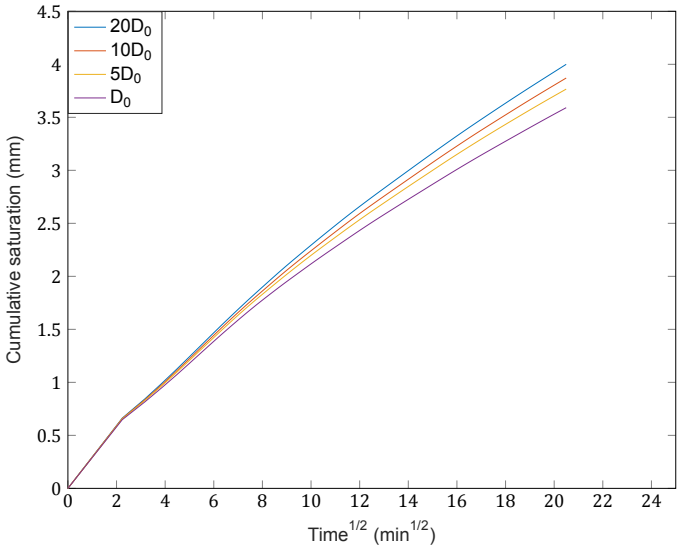


Figure 5.11: Variation of cumulative saturation with square root of time in concrete samples ($f_{agg} = 0.4$) for a duration of 420 minutes for different ITZ diffusivity coefficients

The effect of varying the ITZ diffusivity coefficient can be clearly seen in the figure. When the diffusivity coefficient is increased from $10D_0$ (reference model) to $20D_0$, the penetration depth increases at any given time. The ITZ volume around the aggregates accelerate the moisture transport and thus moisture is able to penetrate more in the concrete body. When the diffusivity coefficient is decreased to $5D_0$ and D_0, the acceleration of moisture transport around the aggregates is less than that in the reference model and the penetration depth decreases.

When considering the cumulative saturation of the entire domain, ITZ effect is observed on a global scale. In order to observe ITZ effect on a local scale, single node in the domain is observed. A node corresponding to the coordinates of (12.5997, 74.0331, 70.1872) is observed which represents the cement phase in the domain. For the time period observed, the nodes goes from an unsaturated state to a saturated state and the nature of this transition for different ITZ diffusivity coefficients is shown in

Figure5.12.

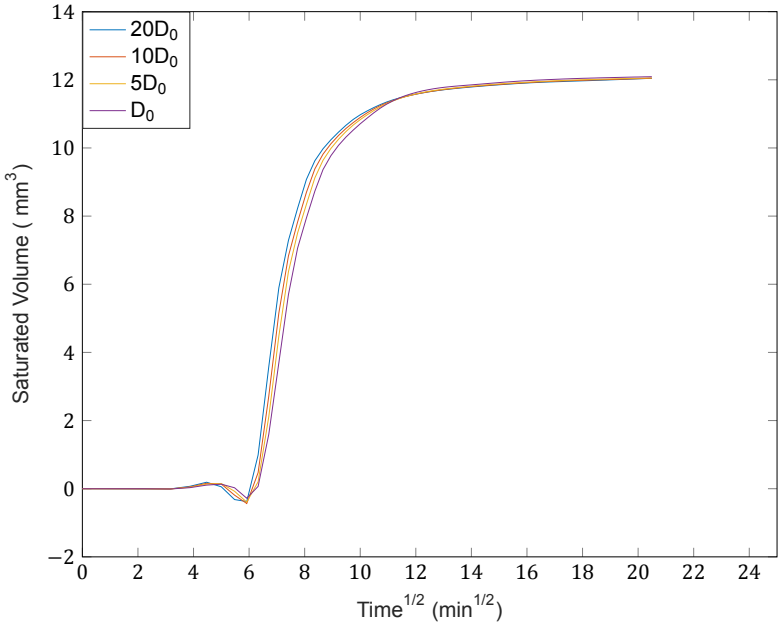


Figure 5.12: Variation of cumulative saturation with time in the observed volume

The variation of saturation volume of the nodes with square root of time can be seen in Figure5.12. The flow observed at the node can be divided into 4 different time intervals. In the initial stage, for a time interval from 0 to 16 minutes in Figure5.12, the water front has not yet reached the node. The saturated volume of the node is close to initial saturation for all ITZ diffusivity coefficients although it is higher for a higher diffusivity (although the difference is small). The next is the transition stage for a time interval from 16 minutes to 36 minutes as shown in Figure5.12. During this stage, the water front is just about to reach the node but has not yet reached it. The saturated volume of the node decreases beyond the initial saturation during this stage. It can be seen from the figure that during this time interval, the trend for diffusivity reverses and the saturation level increases with the decrease of ITZ diffusivity coefficient. Following this stage is the saturation stage for a time interval from 36 minutes to 121 minutes as shown in Figure5.12. During this stage, the water front reaches the node and the saturation of the node increases. By the end of this time interval, the node becomes completely saturated. It can be observed that again the effect of diffusivity reverses from the previous stage as the saturation level at any given time is higher for a higher ITZ diffusivity coefficient. The final stage is the post-saturation stage for a time interval of 121 minutes to 420 minutes. In this stage, the node is already completely saturated. The saturation level of the nodes decreases to a small extent because as the water front moves forward, new nodes are saturated which takes the moisture of the already saturated nodes. In this stage, again a reversal of trend is observed as the saturation level increases with the decrease of diffusivity. While observing the flow at a local level at a particular node, it is found that a higher diffusivity coefficient accelerates the process of saturation when the water front reaches a node but also increases the rate at which saturation level decreases just before the water front reaches the node and also after the node is completely saturated. Regardless of whether a particular node receives or transmits moisture, a higher ITZ diffusivity coefficient accelerates the flow in both ways.

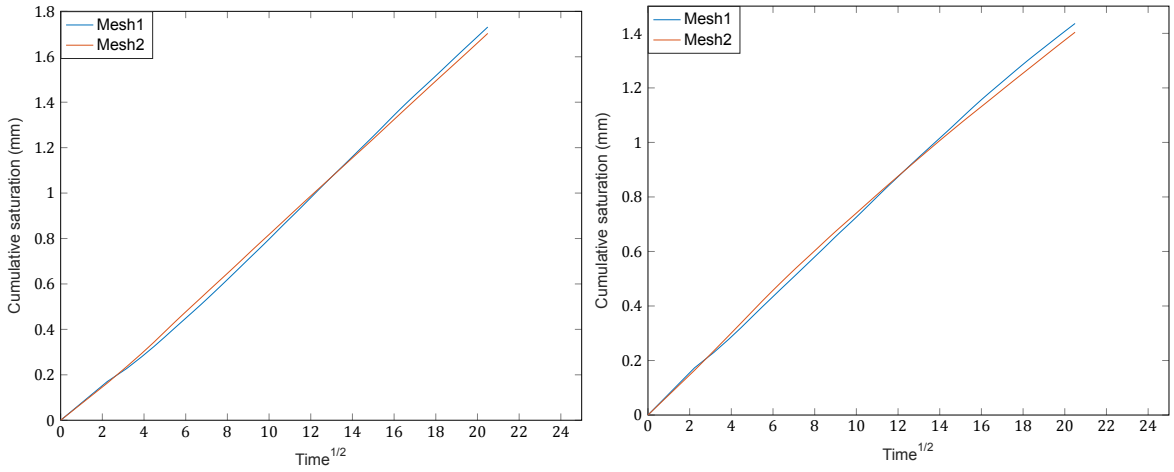
5.3.2. Effect of Mesh Refinement

The effect of mesh refinement on moisture penetration is observed in this subsection. To save computation time, instead of observing the effect of mesh refinement on the domain of $150 \times 150 \times 150\ mm^3$ as presented in the previous section, two new meshes are constructed that represent a domain of $90 \times 90 \times 90\ mm^3$. The mesh properties are given in Table(5.4). Mesh 1 can be considered to represent the reference mesh used in the previous section as the average element length of both the meshes is quite close. Mesh 2 is the finer mesh with more number of elements and less average element length.

Spherical aggregates are generated which corresponds to a volume fraction of 20 percent of the total volume of the domain. The total volume of the spherical aggregates and ITZ is shown in Table5.4. These spherical aggregates are projected on both the meshes and flow is simulated with the same flow parameters as given in Table5.2. The results of the moisture simulation for homogeneous domain and heterogeneous domain with $f_{agg} = 0.2$, for total time of 420 minutes, are presented in Figure5.13.

Table 5.4: Mesh properties

Mesh	Number of nodes	Number of elements	Average element length (mm)	Spherical aggregates		Voronoi aggregates	
				Aggregate volume ($10^5\ mm^3$)	ITZ volume (mm^3)	Aggregate volume ($10^5\ mm^3$)	ITZ volume (mm^3)
Mesh 1	10330	18482	3.375	1.476	611.376	1.479	400.311
Mesh 2	50021	93544	1.984	1.476	611.376	1.470	497.001



(a) $f_{agg} = 0$ (b) $f_{agg} = 0.2$

Figure 5.13: Effect of mesh refinement on moisture transport

It can be seen from the results that for both the homogeneous and heterogeneous case, variation of cumulative moisture distribution with square root of time is almost the same for both the meshes. The slight difference of results in case of heterogeneous domain is due to the small difference in aggregate and ITZ volume as projected on Voronoi elements for both the meshes (Table(5.4), columns 7 and 8). It should be noted here that although the results seem to be mesh independent but the numerical modelling of aggregates is dependent on the mesh. The procedure adopted to project spherical aggregates on the discretized domain was explained in Section 5.1. The procedure involved making two virtual spheres (inner and outer spheres) concentric to the spherical aggregate to determine the boundary of the aggregates in the discretized domain as shown in Figure5.14.

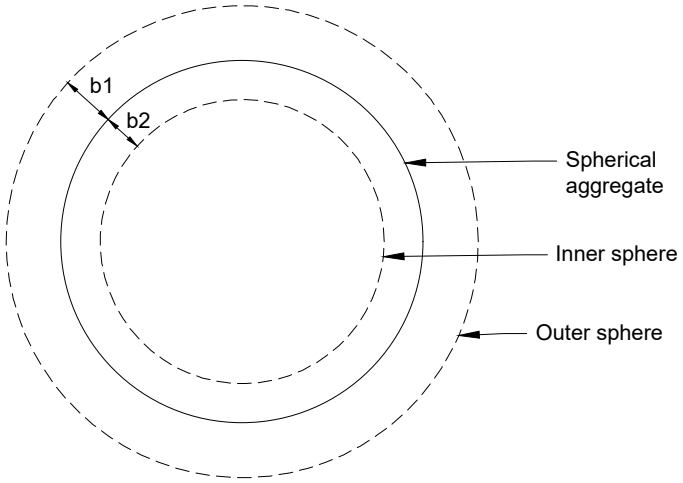


Figure 5.14: Virtual spheres around a spherical aggregate

As shown in Figure5.14, $b1$ and $b2$ dictate the radius of the outer and inner spheres respectively which consequently has an affect on the number of Voronoi elements that will be involved in defining the body and boundary of the aggregate. A global correction for the total ITZ volume is already being considered (as explained in Section 5.2), but no such correction for the volume of the body of the aggregates is being considered. As a result, it is essential that the volume of the body of the aggregates as projected on Voronoi elements is close to the actual volume of the spherical aggregates. The variables $b1$ and $b2$ are varied and iterations are carried out till the projected volume of the aggregate body comes closer to the actual volume of the spherical aggregates. For Mesh 1, $b1$ is 0.6 times the mean element length and $b2$ is 0.22 times the mean element length. For Mesh 2, $b1$ is 0.7 times the mean element length and $b2$ is 0.235 times the mean element length. Using these parameters, the volume of the aggregates as projected on Voronoi elements is obtained as presented in Table(5.4). For each different mesh, a different set of values of $b1$ and $b2$ will give Voronoi aggregates with the same volume as that of the spherical aggregates. Hence, modelling of the aggregates is dependent on the mesh but the numerical results seem to be mesh independent for the cases considered here.

5.3.3. Moisture Transport in Heterogeneous Concrete with a Planar Crack

In Chapter-3, tools were developed to construct a numerical model to simulate moisture transport in uncracked concrete. An approximate model using approximate volume of the Voronoi elements, explicit time stepping scheme and uniform mean diffusivity of the elements and an exact model using exact volume of the Voronoi elements, implicit time stepping scheme and exponential variation of diffusivity of the elements were compared and limitations on the approximate model were presented. In Chapter-4, horizontal moisture distribution in concrete around a planar crack and vertical moisture distribution in a wedge crack were successfully simulated. These simulations were based on two approaches, one where the crack is only considered a porous phase in mortar and the other where moisture-surface interactions are also considered. The latter approach seemed to give better results for vertical moisture distribution in the crack. In this chapter, moisture transport in concrete considering heterogeneities is successfully simulated based on the variation of cumulative saturation level with square root of time.

In this subsection, moisture transport in heterogeneous concrete having a planar crack will be simulated. Due to the highly variable nature of flow around the aggregates and in the vicinity of the crack, a numerical solver using exact volume of the Voronoi elements, implicit time stepping scheme and exponential variation of diffusivity over the element will be used to maintain mass and momentum equilibrium of the flow. In order to realistically simulate moisture transport in the planar crack, the approach considering moisture surface interactions will be implemented in the numerical model.

Table 5.5: Mesh Properties

Parameter	Value
Number of nodes	50229
Number of elements	93960
Average element size	3.299 mm
Crack width	0.2 mm
Crack depth	75 mm
Maximum aggregate radius	20 mm
Minimum aggregate radius	2.5 mm

Table 5.6: Flow Parameters

Parameter	Symbol	Value
Diffusivity of dry concrete	D_0	$3.85 \times 10^{-4}\ mm^2/s$
Empirical constant	n	6
Initial saturation	θ_0	0
Porosity	p	0.185
Time step	Δ t	$1s$
Diffusivity of ITZ	D_{itz}	$3.85 \times 10^{-3}\ mm^2/s$
Diffusivity of aggregate	D_{agg}	0

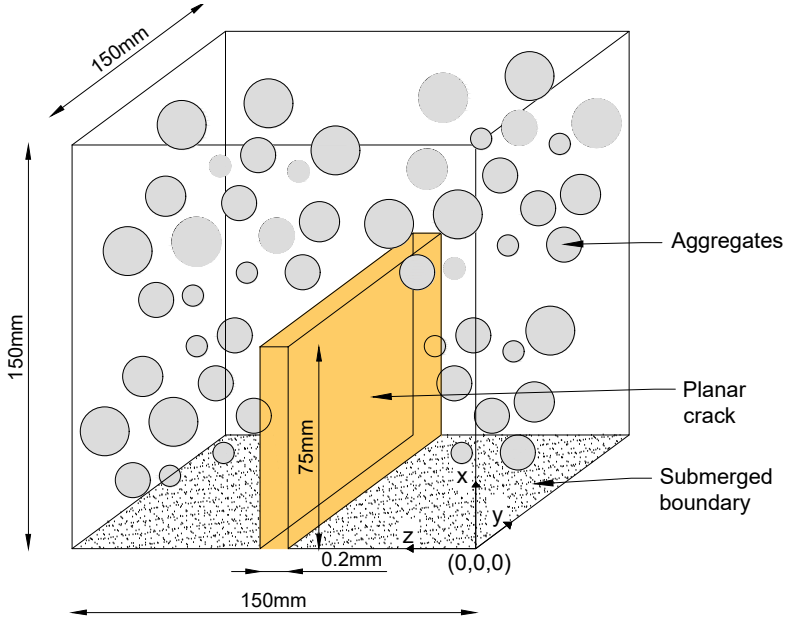
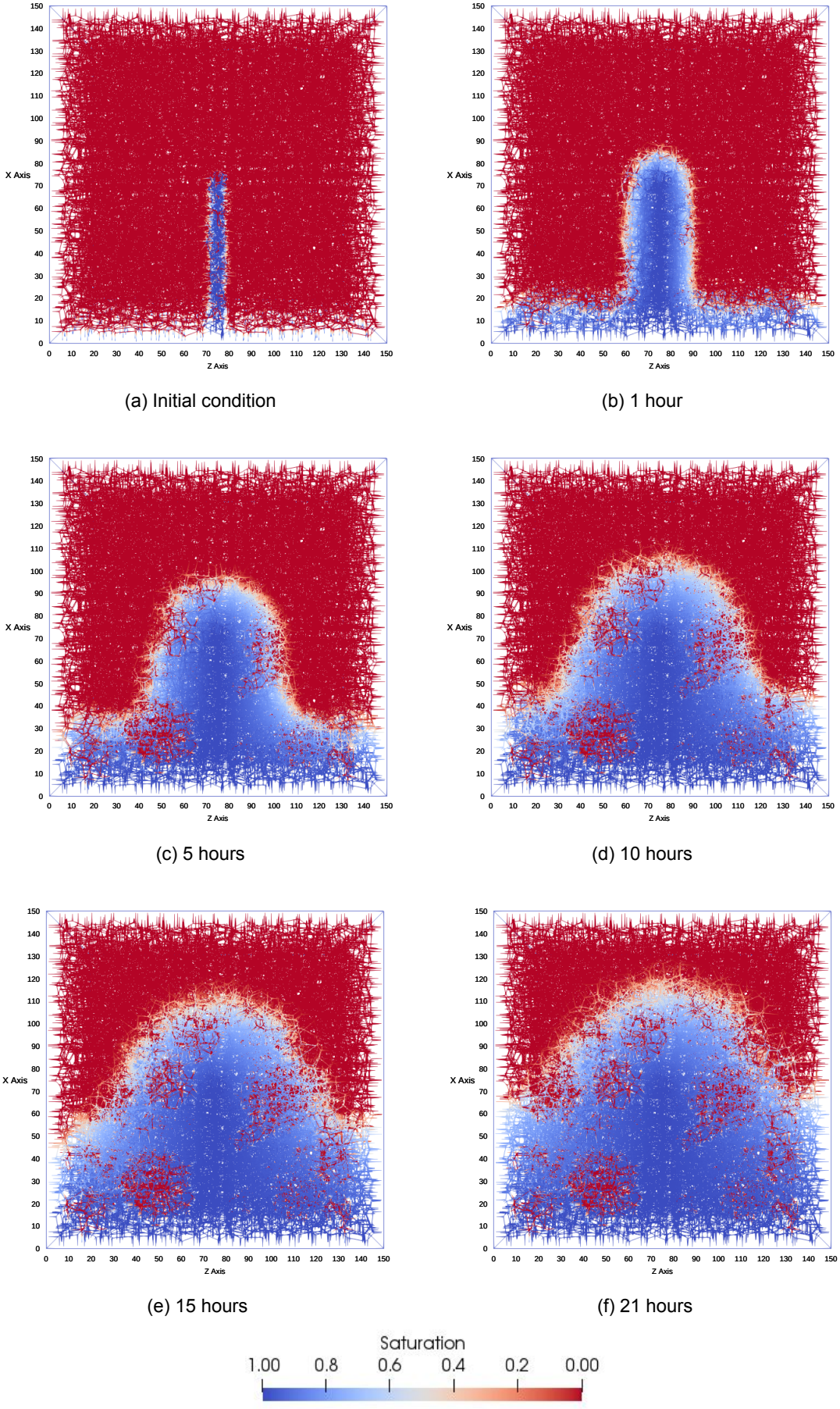


Figure 5.15: Schematic of the cracked heterogeneous domain

The mesh properties and flow parameters of the numerical model are given in Table(5.5) and Table(5.6) respectively. The details of heterogeneous cracked domain used in numerical model are shown in Figure5.15. A planar crack of uniform crack width of $2\ mm$ is generated in the domain and capillary suction in the crack, due to moisture-surface interactions of the moisture at the boundary and the crack walls, is simulated. For a uniform crack width of $2\ mm$, the equilibrium capillary rise height is of $90\ mm$. As the crack depth is shorter than the equilibrium rise height, both the crack faces will be completely submerged in water and act as additional submerged boundaries. The aggregates are generated in the cracked domain (using the take and place process) in such a way that the aggregates do not overlap with the crack. The total of 143 aggregates are generated which comprise 40 percent of the total volume of the domain (as projected on the Voronoi elements) with a mean aggregate diameter $10.7\ mm$. It can be seen in Table(5.6), that the time step used is 1s as opposed to 60s that was used to simulate moisture transport in uncracked heterogeneous concrete. By considering the crack in the domain, the variation of the transport properties across the Voronoi elements increases even more and the simulation using a time step of 60s took too many iterations in a single time step to attain convergence. The time step is reduced which reduces the number of iterations is each time step making the overall numerical simulation faster.



(a) Initial condition

(b) 1 hour

(c) 5 hours

(d) 10 hours

(e) 15 hours

(f) 21 hours

Figure 5.16: Moisture distribution in the domain with a planar crack for volume fraction of aggregates as $f_{agg} = 0.4$

The moisture distribution in the domain at time 0, 1 hour, 5 hours, 10 hours, 15 hours and 21 hours is shown in Figure5.16. The initial condition, as shown in Figure5.16a is the state of the domain at time t=0. Due to capillary suction, the crack is completely saturated before the flow in the domain starts which can be clearly seen in the figure. As time progresses, the moisture travels in the domain through the submerged boundary and the crack walls and more and more nodes become saturated. The effect of aggregates is clearly visible in Figure5.16c-Figure5.16f where the Voronoi nodes lying within the aggregates are in a permanent unsaturated condition.

5.3.4. Conclusions

The numerical model used to simulate moisture transport in heterogeneous concrete was able to closely simulate the experimental results based on variation of cumulative saturation with the square root of time. It was shown in Section(5.2), that as the volume fraction of the aggregates increases, flow in the domain slows down and penetration depth decreases. This is due to impervious nature of aggregates in the concrete which does not allow moisture to pass through. The effect of ITZ on penetration of moisture in heterogeneous concrete was analyzed in Section(5.3). It was found that as the diffusivity coefficient of ITZ is changed, the penetration depth is affected by it. For a higher diffusivity coefficient, the acceleration of moisture transport is more which increases the penetration depth. A lower diffusivity coefficient has a smaller impact on the acceleration of the transport due to which the penetration depth is comparatively less. When observing flow on a local level at a single node in the domain representing the cement phase of concrete, it was observed that a higher ITZ diffusivity coefficient accelerates both the processes of reception and transmission of moisture through the node.

While simulating moisture transport in heterogeneous concrete, due to a high variability of transport properties of the Voronoi elements across the domain, a numerical solver, using exact volume of the Voronoi elements, implicit time stepping scheme and exponential variation of diffusivity over the elements, is used. This is crucial in maintaining the mass and momentum balance of the flow in each time step. As shown in Section(5.3), such a model is also able to simulate moisture transport in heterogeneous concrete with a planar crack in the domain. The numerical model using approximate volume of the Voronoi elements, explicit time stepping scheme and uniform mean diffusivity of the elements won't be suitable to simulate moisture transport in heterogeneous concrete.

Mesh size plays a significant role while projecting the spherical aggregates on Voronoi elements. If a finer mesh is used, the volume of the aggregates as projected on the Voronoi elements will be closer to the actual volume of the spherical aggregates. Also, with a finer mesh the difference in the shape of the projected aggregates and the spherical aggregates will reduce, leading to more accuracy of results. The mesh refinement study showed that there is no significant changes of results even when a finer mesh is used because the modelling approach adopted to project the spherical aggregates over the Voronoi elements was able to generate discretized aggregates with volume almost equal to the volume of the spherical aggregates.

6

Conclusions and Recommendations

6.1. Conclusions

The main objective of this thesis was to understand the nature of flow through capillary absorption in concrete. Physical and material properties of concrete were varied to account for cracking and heterogeneities and moisture transport was analyzed in each model. Computational tools were developed and compared, to see their effectiveness in simulating moisture transport in concrete under different conditions. The results obtained from numerical analyses were compared with experimental results to validate the numerical models. The main conclusions of this thesis, pertaining to specific research questions, are described below.

1. ***What are the effective modelling techniques to simulate moisture transport in concrete?***

   Moisture transport through capillary absorption in uncracked concrete was simulated by solving the discretized form of Richard's equation using different numerical models using lattice elements. An existing numerical model used the approximate volume of Voronoi elements, explicit time stepping scheme and uniform mean diffusivity of elements. These approximations are related to volumetric capacity of Voronoi elements, diffusivity matrix at a particular time step and variation of diffusion over the element length. In addition, a new numerical model was proposed that featured a more accurate behaviour of the numerical and physical phenomenon by using the exact volume of Voronoi elements, implicit time stepping technique and exponential variation of diffusivity of the elements. Both of these numerical models were compared with each other for each approximation using mesh size and time step analysis. The approximations in the existing model were motivated by considering flow in a fully saturated domain when all the approximations actually simulated the exact flow on a global level as the proposed model. This also explained the observation that the error between the existing and the proposed numerical model reduced with the progression of flow as the existing model came closer to the proposed behaviour as more and more part of the domain got involved in the flow. Although both the existing and proposed numerical models were able to successfully simulate moisture transport in uncracked concrete, the results of which were validated through experimental findings, there were some limitations proposed on the use of the existing model. For higher diffusivity, the existing model suffered from oscillations in the saturation level of the nodes (in the part of the domain closer to the origin of flow) due to inadequacy of the existing model to maintain the flow equilibrium at each time step. These oscillations became more frequent for a finer mesh. The proposed model on the other hand was able to maintain the flow equilibrium and there were no oscillations for higher diffusivity or a finer mesh. Mesh size, time step and diffusivity coefficient were shown to be the limiting factors for using either of the models.

   Comparing the two modelling frameworks for uncracked concrete served as a basis to see the validity of the respective numerical models to simulate moisture transport in cracked and heterogeneous concrete. When considering a discrete crack in concrete and treating the crack as an additional porous phase in concrete, the diffusivity of the elements lying in the crack was found to

be extremely high where the proposed model was not found to be suitable owing to a large number of iterations to attain flow equilibrium. The existing model was used instead, which displayed oscillations of saturation level of the nodes lying in the crack but was successfully able to simulate moisture transport in the adjoining uncracked concrete as validated through experiments.

When heterogeneities were considered in concrete, there was a significant variation of transport properties across Voronoi elements. The existing model was not considered favourable in this case as the flow equilibrium at each time step would not have been maintained and instead the proposed model was found suitable to simulate moisture transport as the iterations required to attain flow equilibrium in the proposed model were within an acceptable limit.

2. ***How does a discrete crack affect moisture transport in concrete?***

   The nature of a discrete crack within concrete was envisaged through two approaches. In the first approach, the crack was considered as another porous phase in concrete and in the second approach, in addition to considering the porous nature of the crack, moisture-surface interactions between water present at the boundary and the crack walls was also considered. In both approaches, water transport in uncracked concrete was simulated by solving Richard's equation. In the first approach, water transport in the cracked domain was also simulated using Richard's equation only with a higher diffusivity of the cracked elements. In the second approach, the Lucas-Washburn equation was used to consider moisture-surface interactions between the water present at the boundary and the crack walls. The water transport in part of the cracked domain that was unaffected by the moisture-surface interactions was simulated through Richard's equation with a higher diffusivity than the cement paste. A numerical framework was constructed to incorporate a discrete planar and wedge crack in the discretized domain and Voronoi elements were distinguished to represent the cracked and uncracked domain and were assigned the respective transport properties. Both approaches were able to accurately simulate horizontal moisture distribution around the crack which was validated through experimental observations. In case of simulating vertical moisture distribution in the crack, the second approach showed better performance than the first approach for initial time stages. The rate of saturation of the crack during initial time stages was observed to be comparatively slower in the first approach than observed in the experiment. The second approach showed more realistic vertical moisture distribution in the crack for the initial time stages that were observed.

3. ***How is moisture transport affected due to the multi-phase nature of concrete?***

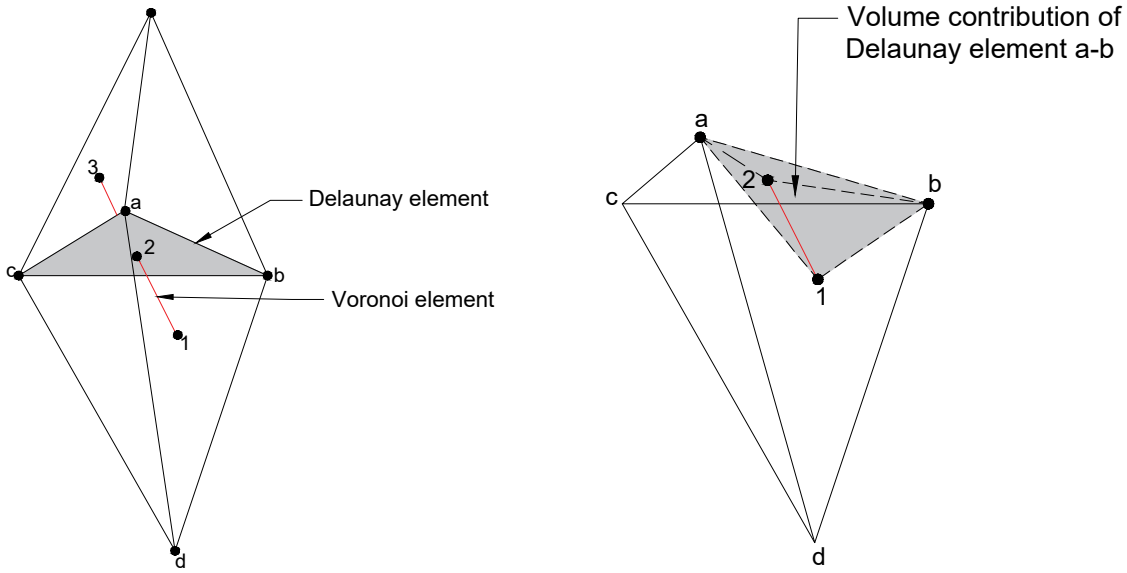
   A numerical framework was formulated to include spherical aggregates to account for the multi-phase nature of concrete. For a given volume fraction, spherical aggregates were randomly generated such that there was no overlap of ITZ of any two adjacent aggregates. A numerical procedure was developed to incorporate the spherical aggregates in the discretized domain which included projection of spherical aggregates on Voronoi elements and assigning transport properties to the elements. Voronoi elements were distinguished to either represent cement paste, aggregate body or ITZ and transport properties of the Voronoi elements were assigned according to the phase they represented. When the flow was simulated in heterogeneous concrete, there were two competing phenomena influencing the rate of moisture transport in concrete. Due to the impervious nature of aggregates, the flow was being slowed down and due to high diffusivity of ITZ, the flow was being accelerated. Simulations for different volume fraction of aggregates were carried out and it was found that the moisture penetration depth was being decreased with increase of volume fraction of aggregates. This behaviour was validated through experimental observations which established the efficiency of the numerical model. The extent to which ITZ affects the transport was analyzed by running numerical simulations with different ITZ diffusivity which showed the expected behaviour of increased transport for higher ITZ diffusivity and vice-versa at both global and local level. Mesh refinement did not seem to affect the results for the cases that were analyzed although the modelling of aggregates in the numerical model was shown to be mesh dependent. It can be concluded that based on the validation of the numerical model through experimental results, the effect of multi-phase nature of concrete on the transport of moisture on the global scale was accurately simulated.

6.2. Recommendations

This section presents recommendations for future research and development related to the work in this thesis.

Exact Porosity

Every Voronoi node present in the discretized domain represents a part of the physical volume of the domain. Figure 6.1a shows a Voronoi element (1-2-3) that passes through a Delaunay triangle (a-b-c). If Delaunay element a-b cracks, then in the numerical model presented in this thesis, the porosity of the entire volume represented by Voronoi node 1 and Voronoi node 3 is considered as 1. But in reality, only a part of the entire volume represented by the Voronoi node lies in the cracked domain and the rest of the volume represents the uncracked part of the domain.



(a) Voronoi and Delaunay element structure

(b) Volume contribution of a Delaunay element

Figure 6.1: Volume contribution for exact porosity

To account for this dual behaviour of the Voronoi node, volume contribution of each Delaunay element should be calculated. Figure 6.1b shows that for the cracked Delaunay element a-b, the volume of the sub-tetrahedra 1-2-a-b should be calculated which represents the volume contribution of Delaunay element a-b and only this part of the volume should be considered to lie in the cracked domain (for Voronoi node 1). Each Voronoi node lying inside the domain is surrounded by 6 Delaunay elements and thus the volume occupied by a Voronoi node can be divided into the volume of 6 sub-tetrahedra. If one or more Delaunay elements crack that surround a Voronoi node, the porosity of that Voronoi node should be calculated using the weighted average (according to volume) of porosity of each sub-tetrahedra. Not just in cracks but also in case of heterogeneities, it can be the case that the volume of a Voronoi element lies in multiple phases of concrete and the porosity of those Voronoi nodes should be calculated accordingly to get a more realistic moisture content of the nodes. Hence, it is recommended that the exact porosity of the Voronoi nodes should be used in the numerical models where there is a transition of material composition in the domain.

Local flow at a node

The variation of saturation level of a single node in a homogeneous and uncracked concrete domain is shown in Figure6.2. The node lies somewhere within the middle of the domain and can be used as a representative of any node in general present in the domain. The flow is simulated using exact volume of Voronoi elements, implicit time stepping scheme and exponential variation of diffusivity.

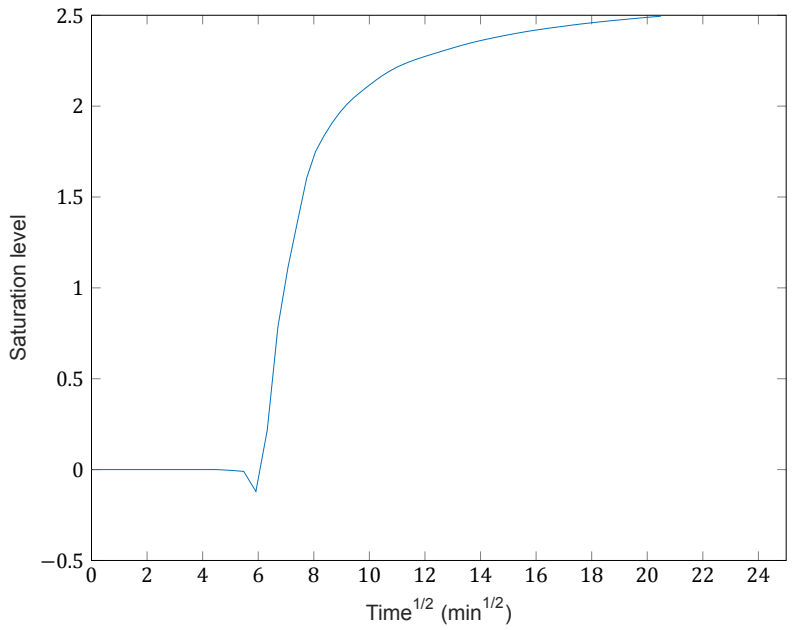


Figure 6.2: Variation of saturation level of a node with square root of time

It can be seen from the figure that initially when the water front is away from the node, the saturation level of the node is close to the initial saturation. At 36 minutes, the water front reaches the node and starts to increase the saturation level of the node. Just before the saturation level starts increasing at 36 minutes, there is a drop in the saturation level of the node below the initial saturation. Physically, there should be no drop in the saturation level. Even after using a comparatively exact behaviour of the numerical model, there is still a drop in the saturation level of the node. It is recommended that to get a realistic behaviour of the flow at a local level, numerical model should be refined to incorporate this local drop in the saturation level just before the water front reaches the node.

Moisture transport in heterogeneous concrete
In the numerical model used to simulate moisture transport in heterogeneous concrete, when assigning the ITZ transport properties to the Voronoi elements, a correction factor was used that takes into account the difference of the total ITZ volume of the spherical and discretized aggregates. It would be more accurate to correct the ITZ volume for each aggregate separately than to do it for all the aggregates combined. In the present numerical model, due to global correction, the local behaviour around the aggregates due to ITZ might not be correctly simulated as the numerical model is only validated for global behaviour. It is recommended that the present numerical model be validated for local behaviour as well and if the results are significantly different then local correction for ITZ volume pertaining to each individual aggregate should be applied. Finally, the presented numerical model can only simulate the behaviour of spherical aggregates. It is recommended that the model should be extended to incorporate non-spherical shapes of aggregates for a more realistic analysis.

Coupling of additional transport mechanisms
In this research, the basic numerical framework is developed to model moisture transport in concrete through capillary absorption. This numerical model can be extended to couple different transport mechanisms such as moisture transport through permeation, diffusion of chlorides etc. that can be used to predict the durability of concrete structures. As water is considered as a carrier of free ions in concrete, this model can be extended to simulate the transport of various ions such as sulphates, carbonates etc. The numerical framework is developed to simulate moisture transport through capillary absorption. It is not necessary that this numerical model pertains to only cementitious materials. The numerical framework can be adjusted to model a material which undergoes unsaturated capillary absorption with known diffusion parameters. The numerical model for moisture transport presented in this thesis can be coupled with mechanical model and even 2-way coupling where the effect of moisture transport on cracking can be simulated.

Bibliography

[1] S Dehghanpoor Abyaneh, HS Wong, and NR Buenfeld. Modelling the diffusivity of mortar and concrete using a three-dimensional mesostructure with several aggregate shapes. *Computational Materials Science*, 78:63–73, 2013.

[2] N Benkemoun, Mohammed Naji Hammood, and Ouali Amiri. Embedded finite element formulation for the modeling of chloride diffusion accounting for chloride binding in meso-scale concrete. *Finite Elements in Analysis and Design*, 130:12–26, 2017.

[3] Terry D Blake, A Clarke, and EH Stattersfield. An investigation of electrostatic assist in dynamic wetting. *Langmuir*, 16(6):2928–2935, 2000.

[4] John E Bolander Jr and Stefano Berton. Simulation of shrinkage induced cracking in cement composite overlays. *Cement and Concrete Composites*, 26(7):861–871, 2004.

[5] CH Bosanquet. Lv. on the flow of liquids into capillary tubes. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 45(267):525–531, 1923.

[6] George Chatzigeorgiou, Vincent Picandet, Abdelhafid Khelidj, and Gilles Pijaudier-Cabot. Coupling between progressive damage and permeability of concrete: analysis with a discrete model. *International journal for numerical and analytical methods in geomechanics*, 29(10):1005–1018, 2005.

[7] Sidney Diamond. The microstructure of cement paste and concrete—-a visual primer. *Cement and Concrete Composites*, 26(8):919–933, 2004.

[8] Assia Djerbi, Stéphanie Bonnet, Abdelhafid Khelidj, and Veronique Baroghel-Bouny. Influence of traversing crack on chloride diffusion into concrete. *Cement and Concrete Research*, 38(6): 877–883, 2008.

[9] N Fries and M Dreyer. The transition from inertial to viscous flow in capillary rise. *Journal of colloid and interface science*, 327(1):125–128, 2008.

[10] EJ Garboczi and DP Bentz. Computer simulation of the diffusivity of cement-based materials. *Journal of materials science*, 27(8):2083–2092, 1992.

[11] Diane Gardner, Anthony Jefferson, and Andrea Hoffman. Investigation of capillary flow in discrete cracks in cementitious materials. *Cement and Concrete Research*, 42(7):972–981, 2012.

[12] B Gérard, D Breysse, A Ammouche, O Houdusse, and O Didry. Cracking and permeability of concrete under tension. *Materials and Structures*, 29(3):141–151, 1996.

[13] Peter Grassl, Hong S Wong, and Nick R Buenfeld. Influence of aggregate size and volume fraction on shrinkage induced micro-cracking of concrete and mortar. *Cement and concrete research*, 40 (1):85–93, 2010.

[14] Peter Grassl, Caroline Fahy, Domenico Gallipoli, and John Bolander. A lattice model for liquid transport in cracked unsaturated heterogeneous porous materials. In *VIII International Conference on Fracture Mechanics of Concrete and Concrete Structures, Toledo, Spain*, 2013.

[15] Erhan Güneyisi, Mehmet Gesoğlu, Seda Karaoğlu, and Kasım Mermerdaş. Strength, permeability and shrinkage cracking of silica fume and metakaolin concretes. *Construction and Building Materials*, 34:120–130, 2012.

[16] C Hall. Water sorptivity of mortars and concretes: a review. *Magazine of concrete research*, 41 (147):51–61, 1989.

[17] Ahmed Hamraoui and Tommy Nylander. Analytical approach for the lucas–washburn equation. *Journal of colloid and interface science*, 250(2):415–421, 2002.

[18] Hans J Herrmann, Alex Hansen, and Stephane Roux. Fracture of disordered, elastic lattices in two dimensions. *Physical Review B*, 39(1):637, 1989.

[19] Xinxin Li, Shenghong Chen, Qing Xu, and Yi Xu. Modeling the three-dimensional unsaturated water transport in concrete at the mesoscale. *Computers & Structures*, 190:61–74, 2017.

[20] Xinxin Li, Shenghong Chen, Qing Xu, and Yi Xu. Modeling the three-dimensional unsaturated water transport in concrete at the mesoscale. *Computers & Structures*, 190:61–74, 2017.

[21] WANG Li-cheng. Analytical relationship between capillarity coefficient and sorptivity of building material [j]. *Journal of Hydraulic Engineering*, 9, 2009.

[22] D Lockington, J-Y Parlange, and P_ Dux. Sorptivity and the estimation of water penetration into unsaturated concrete. *Materials and Structures*, 32(5):342, 1999.

[23] Élise Lorenceau, David Quéré, Jean-Yves Ollitrault, and Christophe Clanet. Gravitational oscillations of a liquid column in a pipe. *Physics of Fluids*, 14(6):1985–1992, 2002.

[24] Richard Lucas. On the temporal law of the capillary rise of fluxes. *Colloid Polymer Science*, 23 (1):15–22, 1918.

[25] Mladena Luković, Branko Šavija, Erik Schlangen, Guang Ye, and Klaas van Breugel. A 3d lattice modelling study of drying shrinkage damage in concrete repair systems. *Materials*, 9(7):575, 2016.

[26] Koichi Maekawa, Tetsuya Ishida, and Toshiharu Kishi. *Multi-scale modeling of structural concrete*. Crc Press, 2008.

[27] Leif Mejlbro and Ervin Poulsen. *Diffusion of Chloride in Concrete: Theory and Application*. Taylor & Francis, 2005.

[28] Günther Meschke and Stefan Grasberger. Numerical modeling of coupled hygromechanical degradation of cementitious materials. *Journal of engineering mechanics*, 129(4):383–392, 2003.

[29] Melvin Mooney. Explicit formulas for slip and fluidity. *Journal of Rheology (1929-1932)*, 2(2):210, 2005.

[30] Yechezkel Mualem. A new model for predicting the hydraulic conductivity of unsaturated porous media. *Water resources research*, 12(3):513–522, 1976.

[31] H Nakamura, W Srisoros, R Yashiro, and M Kunieda. Three-dimensional network model for coupling of fracture and mass transport in quasi-brittle geomaterials. *Journal of Advance Concrete Technology*, 4:147–158, 2006.

[32] Mijo Nikolić, Emir Karavelić, Adnan Ibrahimbegovic, and Predrag Miščević. Lattice element models and their peculiarities. *Archives of Computational Methods in Engineering*, 25(3):753–784, 2018.

[33] MR Nokken. Water transport in brick, stone and concrete. *Cement, Concrete and Aggregates*, 25 (1):1–1, 2003.

[34] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons, 2009.

[35] David Quéré. Inertial capillarity. *EPL (Europhysics Letters)*, 39(5):533, 1997.

[36] David Quéré. Inertial capillarity. *EPL (Europhysics Letters)*, 39(5):533, 1997.

[37] Julien Réthoré, Rene De Borst, and Marie-Angèle Abellan. A two-scale model for fluid flow in an unsaturated porous medium with cohesive cracks. *Computational Mechanics*, 42(2):227–238, 2008.

[38] Claudia Romero Rodriguez, Maxim Deprez, Fernando F. de Mendonca Filho, Stefanie van Offenwert, Veerle Cnudde, Erik Schlangen, and Šavija Branko. X-ray micro tomography of water absorption by superabsorbent polymers in mortar. 2020.

[39] B Šavija. Experimental and numerical investigation of chloride ingress in cracked concrete. *TU Delft*, 2014.

[40] Branko Šavija, José Pacheco, and Erik Schlangen. Lattice modeling of chloride diffusion in sound and cracked concrete. *Cement and Concrete Composites*, 42:30–40, 2013.

[41] Erik Schäffer and Po-zen Wong. Dynamics of contact line pinning in capillary rise and fall. *Physical review letters*, 80(14):3069, 1998.

[42] E Schlangen and JGM Van Mier. Simple lattice model for numerical simulation of fracture of concrete materials and structures. *Materials and Structures*, 25(9):534–542, 1992.

[43] John D Shane, Thomas O Mason, Hamlin M Jennings, Edward J Garboczi, and Dale P Bentz. Effect of the interfacial transition zone on the conductivity of portland cement mortars. *Journal of the American Ceramic Society*, 83(5):1137–1144, 2000.

[44] Klaas Van Breugel. Simulation of hydration and formation of structure in hardening cement-based materials. 1993.

[45] M Th Van Genuchten. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils 1. *Soil science society of America journal*, 44(5):892–898, 1980.

[46] Licheng Wang and Tamon Ueda. Mesoscale modeling of water penetration into concrete by capillary absorption. *Ocean Engineering*, 38(4):519–528, 2011.

[47] Licheng Wang, Mitsutaka Soda, and Tamon Ueda. Simulation of chloride diffusivity for cracked concrete based on rbsm and truss network model. *Journal of Advanced Concrete Technology*, 6 (1):143–155, 2008.

[48] Licheng Wang, Jiuwen Bao, and Tamon Ueda. Prediction of mass transport in cracked-unsaturated concrete by mesoscale lattice model. *Ocean Engineering*, 127:144–157, 2016.

[49] Edward W Washburn. The dynamics of capillary flow, physical review. 1921.

[50] P Whiterspoon, J Wang, K Iwai, and J Gale. Validity of cubic law for fluid flow in a deformable rock fracture. *Water Resources Research*, 16(6):1016–1024, 1980.

[51] HS Wong, M Zobel, NR Buenfeld, and RW Zimmerman. Influence of the interfacial transition zone and microcracking on the diffusivity, permeability and sorptivity of cement-based materials after drying. *Mag. Concr. Res*, 61(8):571–589, 2009.

[52] Peter Wriggers and SO Moftah. Mesoscale models for concrete: Homogenisation and damage behaviour. *Finite elements in analysis and design*, 42(7):623–636, 2006.

[53] Lin Yang, Danying Gao, Yunsheng Zhang, Jiyu Tang, and Ying Li. Relationship between sorptivity and capillary coefficient for water absorption of cement-based materials: theory analysis and experiment. *Royal Society Open Science*, 6(6):190112, 2019.

[54] Mien Yip, Jon Mohle, and JE Bolander. Automated modeling of three-dimensional structural components using irregular lattices. *Computer-Aided Civil and Infrastructure Engineering*, 20(6):393–407, 2005.

[55] Peng Zhang, FH Wittmann, Tiejun Zhao, and EH Lehmann. Neutron imaging of water penetration into cracked steel reinforced concrete. *Physica B: Condensed Matter*, 405(7):1866–1871, 2010.

[56] Peng Zhang, Folker H Wittmann, Tie-jun Zhao, Eberhard H Lehmann, and Peter Vontobel. Neutron radiography, a powerful method to determine time-dependent moisture distributions in concrete. *Nuclear Engineering and Design*, 241(12):4758–4766, 2011.

[57] Chunsheng Zhou. Predicting water permeability and relative gas permeability of unsaturated cement-based material from hydraulic diffusivity. *Cement and Concrete Research*, 58:143–151, 2014.

[58] Chunsheng Zhou and Kefei Li. Numerical and statistical analysis of permeability of concrete as a random heterogeneous composite. *Computers & Concrete*, 7(5):469–482, 2010.

A

Computational Time Analysis

In Chapter 3, during the discretization of the governing equations, two alternate methods were proposed for time stepping technique. The explicit time stepping technique makes use of diffusivity matrix at the previous time step and the implicit time stepping technique uses the diffusivity matrix at the current time step to follow flow equilibrium. Moisture flow is simulated in a homogeneous and uncracked domain with mesh properties corresponding to mesh 3 in Table(3.5) and flow parameters as given in Table(3.4). The first numerical model is formulated using approximate volume of Voronoi elements, explicit time stepping scheme and uniform mean variation of diffusivity. Second numerical model uses approximate volume of Voronoi elements, implicit time stepping scheme and exponential variation of diffusivity. The moisture flow is simulated for 480 minutes and the time required to simulate the flow for different magnitudes of time step is shown in Figure A.1.

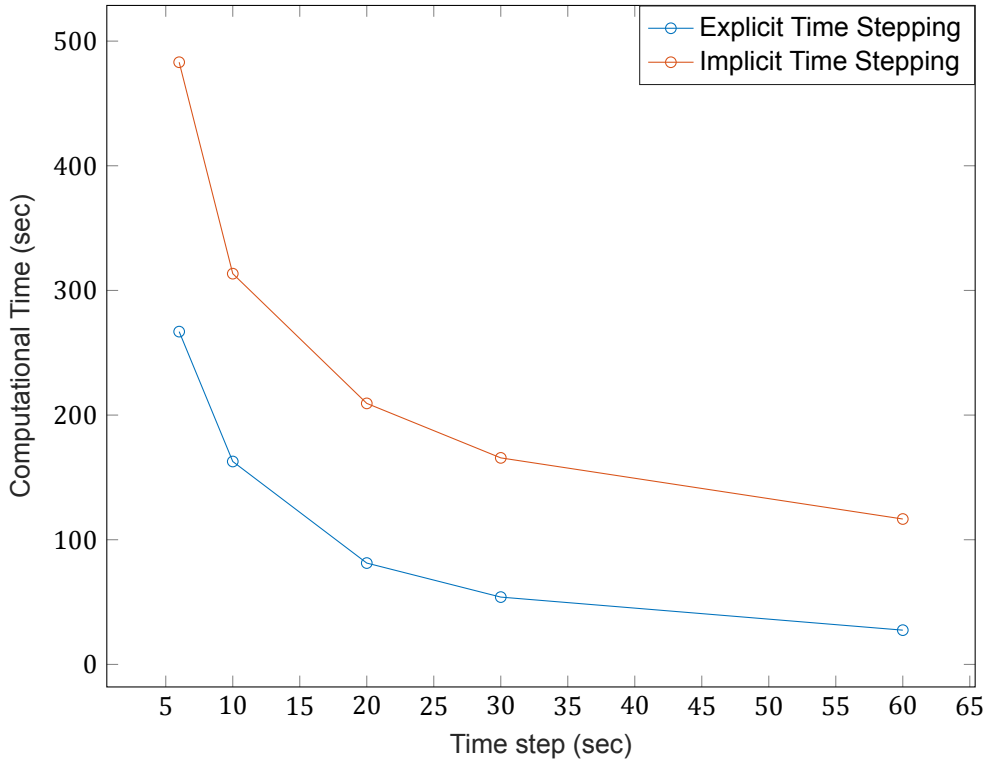


Figure A.1: Computational time comparison for implicit and explicit time stepping schemes

It can be seen that for the magnitudes of time steps observed, computational time for the second model is more than that of the first model. This is due to the iterations required to attain convergence

for the second model that increases the computational time. As the magnitude of time step increases, the computational time decreases for both the models.

In case of the first model using explicit time stepping scheme, the computation time is expected to decrease with the increase of the magnitude of time step. But the accuracy of the results for higher time steps will be less and the results from this model won't be reliable. For the second model which uses the implicit time stepping scheme, it is expected that after a critical time step, the computational time should start to increase. If the magnitude of time step is made larger, then there can come a point when the time required to attain convergence at each time step becomes more which increases the overall computation time. Going further, the computational time will increase with the increase of magnitude of time step.

B

MATLAB Scripts

Node Generation

This section presents the MATLAB script to generate Delaunay nodes in the domain in a semi-random way.

```
clear;
% Define the lattice size
% The domain is divided into number of cells in each of which a single
    node will
% be placed.
lsizeX=20; %number of cells in x-direction
lsizeY=2;
lsizeZ=2;
L=2; % Cell size

A=zeros(lsizeX*lsizeY*lsizeZ,3);%A is a zero matrix with size (no.of cells
     x 3)
p=0;
randomness=0.2; % here the lattice randomness is defined. 0.5-full
    randomness
for i=L:L:lsizeX*L%going through each cell in z-direction
    for j=L:L:lsizeY*L
        for k=L:L:lsizeZ*L
            p=p+1;
            pom1=rand(1);%matlab function to give a random number from 0
                to 1
            if pom1>=0.5
                m1=2;% m will decide the direction of the node from the
                    centre of the cell
            else
                m1=1;
            end
            pom2=rand();
            if pom2>=0.5
                m2=2;
            else
                m2=1;
            end
            pom3=rand(1);
            if pom3>=0.5
                m3=2;
```

```
32              else
33                  m3=1;
34              end
35              dx=rand(1)*randomness*L; %distance of the node from the centre
                    of the cell in x-direction
36              dy=rand(1)*randomness*L;% this distance cannot exceed 0.2L on
                   each side of the centre
37              dz=rand(1)*randomness*L;
38
39              %Assigning the corner cells a fixed coordinate of the nodes
                   for
40              %defining the domain boundary. 8 corner nodes are assigned 8
41              %fixed cordinates.
42              if(i==L) && (j==L) && (k==L)
43                  A(p,1:3)=[0 0 0];
44              end
45              if(i==L) && (j==L) && (k==lsizeZ*L)
46                  A(p,1:3)=[0 0 lsizeZ*L];
47              end
48              if(i==L) && (j==lsizeY*L) && (k==L)
49                  A(p,1:3)=[0 lsizeY*L 0];
50              end
51              if(i==L && j==lsizeY*L && k==lsizeZ*L)
52                  A(p,1:3)=[0 lsizeY*L lsizeZ*L];
53              end
54              if(i==lsizeX*L && j==L && k==L)
55                  A(p,1:3)=[lsizeX*L 0 0];
56              end
57              if(i==lsizeX*L && j==L && k==lsizeZ*L)
58                  A(p,1:3)=[lsizeX*L 0 lsizeZ*L];
59              end
60              if(i==lsizeX*L && j==lsizeY*L && k==L)
61                  A(p,1:3)=[lsizeX*L lsizeY*L 0];
62              end
63              if(i==lsizeX*L && j==lsizeY*L && k==lsizeZ*L)
64                  A(p,1:3)=[lsizeX*L lsizeY*L lsizeZ*L];
65              end
66
67              %now the nodes will be placed at the edges in a semi random
                   way.
68              %There are 12 edges and all the cells having these edges will
69              %have the node on the edge itself.
70
71              if(i==L && j==L && k~=L && k~=lsizeZ*L)
72                  A(p,1:3)=[0 0 k-L/2+(-1)^m3*dz];
73              end
74              if(i==L && j==lsizeY*L && k~=L && k~=lsizeZ*L)
75                  A(p,1:3)=[0 lsizeY*L k-L/2+(-1)^m3*dz];
76              end
77              if(i==lsizeX*L && j==L && k~=L && k~=lsizeZ*L)
78                  A(p,1:3)=[lsizeX*L 0 k-L/2+(-1)^m3*dz];
79              end
80              if(i==lsizeX*L && j==lsizeY*L && k~=L && k~=lsizeZ*L)
81                  A(p,1:3)=[lsizeX*L lsizeY*L k-L/2+(-1)^m3*dz];
82              end
83              if(i==L && k==L && j~=L && j~=lsizeY*L)
```

```
        A(p,1:3)=[0 j-L/2+(-1)^m2*dy 0];
    end
    if(i==L && k==lsizeZ*L && j~=L && j~=lsizeY*L)
        A(p,1:3)=[0 j-L/2+(-1)^m2*dy lsizeZ*L];
    end
    if(i==lsizeX*L && k==L && j~=L && j~=lsizeY*L)
        A(p,1:3)=[lsizeX*L j-L/2+(-1)^m2*dy 0];
    end
    if(i==lsizeX*L && k==lsizeZ*L && j~=L && j~=lsizeY*L)
        A(p,1:3)=[lsizeX*L j-L/2+(-1)^m2*dy lsizeZ*L];
    end
    if(j==L && k==L && i~=L && i~=lsizeX*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx 0 0];
    end
    if(j==L && k==lsizeZ*L && i~=L && i~=lsizeX*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx 0 lsizeZ*L];
    end
    if(j==lsizeY*L && k==L && i~=L && i~=lsizeX*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx lsizeY*L 0];
    end
    if(j==lsizeY*L && k==lsizeZ*L && i~=L && i~=lsizeX*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx lsizeY*L lsizeZ*L];
    end


    %Now the faces are determined. The nodes are placed on the
    %faces of the cell. There are 6 faces

    if(i==L && j~=L && j~=lsizeY*L && k~=L && k~=lsizeZ*L)
        A(p,1:3)=[0 j-L/2+(-1)^m2*dy k-L/2+(-1)^m3*dz];
    end
    if(i==lsizeX*L && j~=L && j~=lsizeY*L && k~=L && k~=lsizeZ*L)
        A(p,1:3)=[lsizeX*L j-L/2+(-1)^m2*dy k-L/2+(-1)^m3*dz];
    end
    if(j==L && i~=L && i~=lsizeX*L && k~=L && k~=lsizeZ*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx 0 k-L/2+(-1)^m3*dz];
    end
    if(j==lsizeY*L && i~=L && i~=lsizeX*L && k~=L && k~=lsizeZ*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx lsizeY*L k-L/2+(-1)^m3*dz];
    end
    if(k==L && i~=L && i~=lsizeX*L && j~=L && j~=lsizeY*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx j-L/2+(-1)^m2*dy 0];
    end
    if(k==lsizeZ*L && i~=L && i~=lsizeX*L && j~=L && j~=lsizeY*L)
        A(p,1:3)=[i-L/2+(-1)^m1*dx j-L/2+(-1)^m2*dy lsizeZ*L];
    end


    %Now the nodes will be assigned to all interior cells that are
        not on
    %any face, edge or corner

    if(i~=L && i~=lsizeX*L && j~=L && j~=lsizeY*L && k~=L && k~=
        lsizeZ*L)
    A(p,1:3)=[i-L/2+(-1)^m1*dx j-L/2+(-1)^m2*dy k-L/2+(-1)^m3*dz];
```

```
                %i-L/2+(-1)^m3*dz];
            %here the second term in each expression gives the direction
                of the node from the centre of the cell
            end
        end
    end
    disp(i/lsizeX*100);
end
%scatter3(A(:,1),A(:,2),A(:,3))

%
% for i1=(1+L/2):L:(lsizeZ-L/2)
%     for j1=(1+L/2):L:(lsizeY-L/2)
%         for k1=(1+L/2):L:(lsizeX-L/2)
%             p=p+1;
%             A(p,1:3)=[k1 j1 i1];
%         end
%     end
% end
%


%
% pom=1:200;
% A(pom,3)=0;
% % pom1=(50*50*50-50*50+1):(50*50*50);
% pom1=19601:2000;
% A(pom1,3)=50;
%
% for i=1:(lsizeX*lsizeY*lsizeZ)
%     if A(i,1)<1
%         A(i,1)=0.001;
%     elseif A(i,1)>29
%         A(i,1)=29.999;
%     end
%     if A(i,2)<1
%         A(i,2)=0.001;
%     elseif A(i,2)>29
%         A(i,2)=29.999;
%     end
%     if A(i,3)<1
%         A(i,3)=0.001;
%     elseif A(i,3)>29
%         A(i,3)=29.999;
%     end
%     disp(i/(lsizeX*lsizeY*lsizeZ));
% end
% for i=1:100
%     A(i,3)=0.5;
% end
%
% for i=901:1000
%     A(i,3)=9.5;
% end

fid=fopen('nodes3D.dat','w');%here we are saving the nodes in this file
    nodes3D.dat
```

```
% A=0.5*A;
for i1=1:p
    fprintf(fid,'%d %f %f %f',i1,A(i1,1),A(i1,2),A(i1,3));% this gives the
        output as node number, x,y,z cordinate of that node
    fprintf(fid,'\n');
 end

 fclose(fid);
```

Generation of Delaunay and Voronoi Elements

This section presents the MATLAB scripts and functions used to generate lattice elements once the nodes have been generated. The main script generated Delaunay and Voronoi elements and calculates their cross-sectional area and volume.

The main script is given below:

```

% Updated on 29-6-2017
tic
fileid=fopen('mesh.opt','r');
OPT=fscanf(fileid,'%g',5);
NoN=OPT(1); % Total number of nodes
a=OPT(2); % domain size in X direction
b=OPT(3); % domain size in Y direction
c=OPT(4); % domain size in Z direction
delta=OPT(5); % lattice cell size (cubic cells are assumed)
fclose(fileid);
clear('OPT');
fid=fopen('nodes3D.dat','r');
A=fscanf(fid,'%f',[4 NoN]); % this needs to be ajdusted if the overall
    number of nodes is changed, i.e. [4 NNODES]
A=A';
A=A(:,2:4);
% Create elements between connected nodes (points)
Tes=DelaunayTri(A(:,1),A(:,2),A(:,3));



conNodes1=[]; %initialize the matrix containing connected nodes, with
    repeated elements
conNodes1=uint32(conNodes1);
pomoc1=max(size(Tes));
AllElements=zeros(6*pomoc1,2);%initialize the matrix containing connected
    nodes, with repeated elements

for i=1:pomoc1
    disp('Delaunay Tesselation in %:');
    disp(i/max(size(Tes))*100);
    AllElements(i*6-5,1)=Tes(i,1);
    AllElements(i*6-5,2)=Tes(i,2);
    AllElements(i*6-4,1)=Tes(i,1);
    AllElements(i*6-4,2)=Tes(i,3);
    AllElements(i*6-3,1)=Tes(i,1);
    AllElements(i*6-3,2)=Tes(i,4);
    AllElements(i*6-2,1)=Tes(i,2);
    AllElements(i*6-2,2)=Tes(i,3);
    AllElements(i*6-1,1)=Tes(i,2);
```

```
39      AllElements(i*6-1,2)=Tes(i,4);
40      AllElements(i*6,1)=Tes(i,3);
41      AllElements(i*6,2)=Tes(i,4);
42  end
43
44  AllElements=AllElements';
45  tepm=sort(AllElements);
46  AllElementsF=tepm'; % Elements are sorted in such a way that node1<node2
47
48  clear('Tes','AllElements');
49
50
51  final=unique(AllElementsF,'rows');
52  clear('AllElementsF');
53
54
55  % Creating auxiliary points %
56
57  % Auxiliary points in X direction %
58
59
60  n=3; % Number of rows of auxiliary cells
61  n1=0;
62  n2=0;
63  n3=0;
64  n4=0;
65  n5=0;
66  n6=0;
67
68  % Initialize matrices of auxiliary cells %
69  X1=zeros(n,3);
70  X2=zeros(n,3);
71  Y1=zeros(n,3);
72  Y2=zeros(n,3);
73  Z1=zeros(n,3);
74  Z2=zeros(n,3);
75
76  for i=1:NoN
77
78      % Auxiliary points "under" the X boundary (n rows) %
79      if A(i,1)<=n*delta
80          n1=n1+1;
81          X1(n1,1)=-A(i,1);
82          X1(n1,2)=A(i,2);
83          X1(n1,3)=A(i,3);
84      end
85
86      % Auxiliary points "over" the X boundary (n rows) %
87      if A(i,1)>=(a-n*delta)
88          n2=n2+1;
89          X2(n2,1)=2*a-A(i,1);
90          X2(n2,2)=A(i,2);
91          X2(n2,3)=A(i,3);
92      end
93
94      % Auxiliary points "under" the Y boundary (n rows) %
```

```
95      if A(i,2)<=n*delta
96          n3=n3+1;
97          Y1(n3,1)=A(i,1);
98          Y1(n3,2)=-A(i,2);
99          Y1(n3,3)=A(i,3);
100     end
101     % Auxiliary points "over" the Y boundary (n rows) %
102     if A(i,2)>=(b-n*delta)
103         n4=n4+1;
104         Y2(n4,1)=A(i,1);
105         Y2(n4,2)=2*b-A(i,2);
106         Y2(n4,3)=A(i,3);
107     end
108
109     % Auxiliary points "under" the Z boundary (n rows) %
110     if A(i,3)<=n*delta
111         n5=n5+1;
112         Z1(n5,1)=A(i,1);
113         Z1(n5,2)=A(i,2);
114         Z1(n5,3)=-A(i,3);
115     end
116     % Auxiliary points "over" the Y boundary (n rows) %
117     if A(i,3)>=(c-n*delta)
118         n6=n6+1;
119         Z2(n6,1)=A(i,1);
120         Z2(n6,2)=A(i,2);
121         Z2(n6,3)=2*c-A(i,3);
122     end
123 end
124
125 % Form the real+auxiliary cell matrix %
126
127 TOT=[A;X1;X2;Y1;Y2;Z1;Z2];
128
129
130 Tes1=DelaunayTri(TOT);
131
132 [v,r]=voronoiDiagram(Tes1);
133
134 clear('Tes1');
135 clear('Tes');
136 r1=r(1:NoN);%Voronoi cells of "real" nodes only
137 clear('r');
138 qq=0;
139 pp=0;
140
141
142 %the code below bring the points that are lying outside the domain
        boundary
143 %to lie on the boundary (not required)
144 %for i=1:1:max(size(v))
145 %         if(v(i,1)<0)
146 %             v(i,1)=0;
147 %         end
148 %         if(v(i,2)<0)
149 %             v(i,2)=0;
```

```
150 %         end
151 %         if(v(i,3)<0)
152 %             v(i,3)=0;
153 %         end
154 %         if(v(i,1)>a)
155 %             v(i,1)=a;
156 %         end
157 %         if(v(i,2)>b).
158 %             v(i,2)=b;
159 %         end
160 %         if(v(i,3)>c)
161 %             v(i,3)=c;
162 %         end
163 %end
164
165
166 %the code below makes an idmat (id matrix) which stores the voronoi ids in
        the first col vs
167 %delaunay ids in the second column
168 ind1=1;
169 ind2=1;
170 for i=1:1:max(size(r1))
171     for j=1:1:max(size(r1{i}))
172         idmat(ind1,1)=r1{i}(1,ind2);
173         idmat(ind1,2)=i;
174         ind1=ind1+1;
175         ind2=ind2+1;
176     end
177     ind2=1;
178 end
179 idmat=sortrows(idmat,1);
180 %here the ids are not starting with 1 coz some ids belonged to auxilliary
181 %voronoi nodes which got removed
182
183 %the code below makes a mutidimensional array in which the index of the
184 %array represents the voronoi node ids and the set of points on that index
185 %represents the delaunay node ids that surround it
186 ind1=1;
187 for i=1:1:max(size(idmat))
188     indexmat{idmat(i,1)}(1,ind1)=idmat(i,2);
189     ind1=ind1+1;
190     if(i~=max(size(idmat)) && idmat(i,1)~=idmat(i+1,1))
191         ind1=1;
192     end
193 end
194
195
196
197 % Create vector cell array of Voronoi vertice coordinates for each cell %
198 %
199 for i1=1:NoN
200     pomoc=r1{i1};
201     pomoc=pomoc';
202     s1=v(pomoc,:);
203
204     for i=1:max(size(s1))
```

```
205         for j=1:max(size(s1))
206             if (i~=j && abs(s1(i,1)-s1(j,1))<0.000001 && abs(s1(i,2)-s1(j
                    ,2))<0.000001 && abs(s1(i,3)-s1(j,3))<0.000001)
207                 s1(j,:)=[-rand(1)*i*rand(1) -rand(1)*j*rand(1) -rand(1)*j
                        *(i*rand(1))];
208
209             end
210         end
211     end
212     E{i1}=s1;
213 end
214
215  count_out_1=0;
216         count_out_2=0;
217         count_out_3=0;
218         count_out_4=0;
219         count_out_5=0;
220         count_out_6=0;
221 voronoiz=[];
222 brojanje=0;
223 for i4=1:max(size(final))
224     disp('List correction and Voronoi triangulation in %:\n');
225     disp(i4/max(size(final))*100);
226     E1=E{1,final(i4,1)};
227     E2=E{1,final(i4,2)};
228     rr=r1{final(i4,1)};
229     vertice=[];
230     indexing=[];
231     %vertice=sparse(vertice);
232     n=0;
233
234     for p=1:length(E1)
235         for q=1:length(E2)
236             if (abs(E1(p,1)-E2(q,1))<0.000001 && abs(E1(p,2)-E2(q,2))
                    <0.000001 && abs(E1(p,3)-E2(q,3))<0.000001) % take into
                    account numerical error
237                 n=n+1;
238                 vertice(n,:)=E1(p,:);% vertice contains the nodal
                        coordinates of the common voronoi points between two
                        delaunay nodes forming a connection
239                 indexing(n)=rr(p);% indexing contains the node ids of the
                        common voronoi points between two delaunay nodes forming
                         a connection
240             end
241         end
242     end
243
244     pom(i4,1)=n;
245     %[vertice]=arrange(vertice1); % arrange vertice so that they are
            consistent
246
247        %code to see if there are common points lying outside the domain (
               of
248        %course they will)
249
250       % for i=1:1:max(size(vertice))
```

```
251             % if(vertice(i,1)<0)
252              %    count_out_1=count_out_1+1;
253              %end
254              %if(vertice(i,2)<0)
255               %   count_out_2=count_out_2+1;
256              %end
257              %if(vertice(i,3)<0)
258               %   count_out_3=count_out_3+1;
259              %end
260              %if(vertice(i,1)>a)
261               %   count_out_4=count_out_4+1;
262              %end
263              %if(vertice(i,2)>b)
264               %   count_out_5=count_out_5+1;
265              %end
266              %if(vertice(i,3)>c)
267               %   count_out_6=count_out_6+1;
268              %end
269           %end
270
271       if (pom(i4,1)>2) % exclude Delaunay elements whose Voronoi cells do
              not touch (share a facet)
272       %if (pom(i4,1)>2)
273           qq=qq+1;
274   %         kon_elementi(qq,1)=final(i4,1);
275   %         kon_elementi(qq,2)=final(i4,2);
276   %        original{qq}=vertice;
277
278
279           arranged=area_facet_new(vertice,a,b,c);
280   %          arranged{qq}=area_facet_new(vertice);
281         for kk=1:length(arranged)-1
282             brojanje=brojanje+1;
283             voronoiz(brojanje,1)=indexing(arranged(kk)); %voronoiz stores
                  the connection of voronoi elements. Each row has two node ids
                   that form a voronoi element (just as in final)
284             voronoiz(brojanje,2)=indexing(arranged(kk+1));
285         end
286
287       end
288
289       %code below will give the area of the delaunay elements
290
291       if (pom(i4,1)>2) % exclude Delaunay elements whose Voronoi cells do
              not touch (share a facet)
292       %if (pom(i4,1)>2)
293           pp=pp+1;
294           kon_elementi(pp,1)=final(i4,1);%node 1 which is a part of delaunay
                  element
295           kon_elementi(pp,2)=final(i4,2);%node 2 which is a part of delaunay
                  element
296           area_delaunay(pp,1)=area_facet(vertice,a,b,c);% the area of the
                 delaunay element
297
298       end
299   end
```

```
300
301 %code to check the number of nodes that are part of a connection which lie
302 %in the negative direction of the domain
303
304
305 neg=0;
306 for i=1:1:max(size(voronoiz))
307     for j=1:1:2
308         if(v(voronoiz(i,j),1)<0 || v(voronoiz(i,j),2)<0 || v(voronoiz(i,j)
                ,3)<0)
309            neg=neg+1;
310         end
311     end
312 end
313
314
315
316 %uptill now voronoiz has the pair of node ids forming voronoi elements and
        v has the coordinates of the voronoi nodes corresponding to their ids.
317
318
319 %code to bring the coordinates outside the domain to lie on the domain
320 %boundary
321 for i=1:1:max(size(voronoiz))
322     for j=1:1:2
323         if(v(voronoiz(i,j),1)<0)
324             v(voronoiz(i,j),1)=0;
325         end
326         if(v(voronoiz(i,j),2)<0)
327             v(voronoiz(i,j),2)=0;
328         end
329         if(v(voronoiz(i,j),3)<0)
330             v(voronoiz(i,j),3)=0;
331        end
332         if(v(voronoiz(i,j),1)>a)
333             v(voronoiz(i,j),1)=a;
334         end
335         if(v(voronoiz(i,j),2)>b)
336             v(voronoiz(i,j),2)=b;
337         end
338        if(v(voronoiz(i,j),3)>c)
339             v(voronoiz(i,j),3)=c;
340         end
341     end
342 end
343 voronoiz=unique(voronoiz,'rows');
344
345
346 %the code below is for calculating the area  of the voronoi elements using
        the delaunay nodes.An
347 %important assumption is that here, only the connections having atleast 3
348 %common delaunay nodes are assigned the area and the rest of them are not.
349 qq=0;count=0;
350 for i=1:1:max(size(voronoiz))
351     indexmat1=indexmat{voronoiz(i,1)}; %indexmat has the node ids of the
          delaunay nodes surrounding a voronoi node
```

```
352     indexmat2=indexmat{voronoiz(i,2)};
353
354     comn=0;
355     for p=1:1:max(size(indexmat1))
356         for q=1:1:max(size(indexmat2))
357             if(indexmat1(p)==indexmat2(q))
358                 comn=comn+1;
359                 commonid(comn,1)=indexmat1(p);
360                 commoncord(comn,:)=A(indexmat1(p),:);
361             end
362         end
363     end
364
365 rep(i,1)=comn;
366
367 if(rep(i,1)>3)
368     count=count+1;
369 end
370
371
372
373 if(rep(i,1)>2)
374     qq=qq+1;
375      kon_elementj(qq,1)=voronoiz(i,1);%kon_elementj is storing the
            connections that have atleast 3 common points
376      kon_elementj(qq,2)=voronoiz(i,2);
377      area_voronoi(qq,1)=area_facet(commoncord,a,b,c);
378      length_cracks(qq,:)=crack_length(commoncord);
379 end
380
381 commonid(:,1)=[];
382 commoncord(:,:)=[];
383 end
384
385         kon_elementi=unique(kon_elementi,'rows');
386         kon_elementj=unique(kon_elementj,'rows');
387
388        %the problem is that we need to eliminate the coordinates that lie
389        %outside the domain in v matrix and assign new nodal ids to the
390        %voronoi connections.
391
392         index_new11=unique(kon_elementj(:,1));
393         index_new12=unique(kon_elementj(:,2));
394         index_new13=unique(vertcat(index_new11,index_new12));
395         for i=1:max(size(index_new13))
396             v_new(i,1)=index_new13(i,1);
397             v_new(i,2:4)=v(index_new13(i,1),:);
398         end
399        %v_new contains the voronoi nodal ids in the first column and their
400        %coordinates in the subsequent columns
401         for i=1:max(size(v_new))
402             v_trans(v_new(i,1),1)=i;
403             v_trans(v_new(i,1),2:4)=v_new(i,2:4);
404         end
405        %in v_trans the index represents the old node id and the first
406        %column represents the new node id
```

```
407         for i=1:max(size(kon_elementj))
408             kon_elementj_new(i,1)=v_trans(kon_elementj(i,1),1);
409             kon_elementj_new(i,2)=v_trans(kon_elementj(i,2),1);
410         end
411         v=v_new(:,2:4);
412         kon_elementj=kon_elementj_new;
413
414         %the code below is to calculate the volume
415
416
417         %the code below stores the nodal ids of the delaunay nodes that
418         %surround a voronoi node
419         for i=1:max(size(v))
420             indexmat_new{i}=indexmat{v_new(i,1)};
421         end
422         %the code below the delaunay node coordinates surrounding a voronoi
423         %node
424         for i=1:max(size(indexmat_new))
425             for j=1:length(indexmat_new{i})
426                 coordmat{i}(j,1)=A(indexmat_new{i}(1,j),1);
427                 coordmat{i}(j,2)=A(indexmat_new{i}(1,j),2);
428                 coordmat{i}(j,3)=A(indexmat_new{i}(1,j),3);
429             end
430         end
431         %coordmat has only 4x3 or 3x3 entries
432
433         %the following code saves the delaunay node ids surrounding a
                voronoi
434         %node
435         for i=1:length(indexmat_new)
436             for j=1:length(indexmat_new{i})
437                 indexmat_open(i,j)=indexmat_new{i}(1,j);
438             end
439         end
440         %the following code checks whether if two voronoi nodes are
                surrounded
441         %by the same 4 delaunay nodes which turns out that it is not the case
                .
442         count_rep_nodes=0;
443             for i=1:length(indexmat_open)-1
444                 for j=i+1:length(indexmat_open)
445                     if(indexmat_open(i,:)==indexmat_open(j,:))
446                         count_rep_nodes=count_rep_nodes+1;
447                     end
448                 end
449             end
450
451
452             %this code calculates the volume of the tetrahedra that
                    surrounds
453             %a voronoi node
454         for i=1:max(size(coordmat))
455             if(length(coordmat{i})<4)
456                 volume(i,1)=0;
457             else
458                 P1=coordmat{i}(1,:);
```

```
459             P2=coordmat{i}(2,:);
460             P3=coordmat{i}(3,:);
461             P4=coordmat{i}(4,:);
462             volume(i,1)=volume_tetrahedra(P1,P2,P3,P4);
463         end
464     end
465
466
467     %{
468     %the following code tells us that there is more than one voronoi
469     %element sharing the same common delaunay face which in the end
        turns
470     %out that it is the same element but with nodal positions stored in
471     %the reverse order.
472     count_del=0;
473     count=1;
474     for i=1:(length(common_del_nodes)-1)
475         for j=(i+1):length(common_del_nodes)
476             if(common_del_nodes(i,1)==common_del_nodes(j,1))
477                 if(common_del_nodes(i,2)==common_del_nodes(j,2))
478                     if(common_del_nodes(i,3)==common_del_nodes(j,3))
479                         count_del=count_del+1;
480                         count_del_score(count,1)=i;
481                         count_del_score(count,2)=j;
482                         count=count+1;
483                     end
484                 end
485             end
486         end
487     end
488     %this code proves that all the elements sharing the common delaunay
489     %face are the same just the nodes written in the reverse order
490     for i=1:length(count_del_score)
491         repeat_elements(i,1:2)=kon_elementj(count_del_score(i,1),:);
492         repeat_elements(i,3:4)=kon_elementj(count_del_score(i,2),:);
493     end
494     %}
495
496     %hence there is no other element sharing the common delaunay face.
497     %Hence there is only one element passing through a delaunay face.(So
498     %why does it looks like in paraview that there is more than one
499     %voronoi element on the same delaunay face on the edges?????)
500
501     %the code below gives the element ids that have been repeated
502     count_rep_ele=1;
503     for i=1:length(kon_elementj)
504         for j=i+1:length(kon_elementj)
505             if(kon_elementj(i,1)==kon_elementj(j,2))
506                 if(kon_elementj(i,2)==kon_elementj(j,1))
507                     index_rep_ele(count_rep_ele,1)=j;
508                     count_rep_ele=count_rep_ele+1;
509                 end
510             end
511         end
512     end
513     index_rep_ele=unique(index_rep_ele);
```

```
514
515     %the code below replaces the value of the entries in the repeated
516     %voronoi element rows with zeros and then delete the rows containing
            zeros
517     count_rep_ele=1;
518     for i=1:length(kon_elementj)
519         if(i==index_rep_ele(count_rep_ele,1))
520             kon_elementj_rep_ele(i,:)=0;
521             count_rep_ele=count_rep_ele+1;
522         else
523             kon_elementj_rep_ele(i,:)=kon_elementj(i,:);
524         end
525     end
526     kon_elementj=[];
527     kon_elementj= kon_elementj_rep_ele(any(kon_elementj_rep_ele,2),:);
528
529
530     %the code below is deleting the repeated voronoi elements in the
            area matrix
531       count_rep_ele=1;
532     for i=1:length(area_voronoi)
533         if(i==index_rep_ele(count_rep_ele,1))
534             area_voronoi_rep_ele(i,:)=0;
535             count_rep_ele=count_rep_ele+1;
536         else
537             area_voronoi_rep_ele(i,:)=area_voronoi(i,:);
538         end
539     end
540     area_voronoi=[];
541     area_voronoi= area_voronoi_rep_ele(any(area_voronoi_rep_ele,2),:);
542
543
544
545     %the code below conatins the common delaunay node ids for two
            voronoi
546     %element forming nodes.
547     for i=1:max(size(kon_elementj))
548         comn=1;
549         for j=1:max(size(indexmat_new{kon_elementj(i,1)}))
550             for k=1:max(size(indexmat_new{kon_elementj(i,2)}))
551                 if(indexmat_new{kon_elementj(i,2)}(1,k)==indexmat_new{
                        kon_elementj(i,1)}(1,j))
552                     common_del_nodes(i,comn)=indexmat_new{kon_elementj(i
                            ,2)}(1,k);
553                     comn=comn+1;
554                 end
555             end
556         end
557     end
558
559
560     %code to get the volume contribution of each node in a voronoi
561     %element
562     for i=1:length(kon_elementj)
563         for j=1:2
564             P1=v(kon_elementj(i,j),:);
```

```
565             P2=A(common_del_nodes(i,1),:);
566             P3=A(common_del_nodes(i,2),:);
567             P4=A(common_del_nodes(i,3),:);
568             volume_node(i,j)=volume_tetrahedra(P1,P2,P3,P4);
569         end
570
571     end
572     count_vol_rep(1:length(v),1)=1;
573     for i=1:length(kon_elementj)
574         vol_rep(kon_elementj(i,1),count_vol_rep(kon_elementj(i,1),1))=
                volume_node(i,1);
575         vol_rep(kon_elementj(i,2),count_vol_rep(kon_elementj(i,2),1))=
                volume_node(i,2);
576         count_vol_rep(kon_elementj(i,1),1)=count_vol_rep(kon_elementj(i
                ,1),1)+1;
577         count_vol_rep(kon_elementj(i,2),1)=count_vol_rep(kon_elementj(i
                ,2),1)+1;
578     end
579     vol_rep(:,5)=vol_rep(:,1)+vol_rep(:,2)+vol_rep(:,3)+vol_rep(:,4);
580     vol_rep(:,6)=volume(:,1);
581     vol_rep(:,7:9)=v(:,1:3);
582     count_rep_ind_vol_rep=1;
583   for i=1:length(volume_node)
584       for j=1:2
585           if(volume_node(i,j)==0)
586               vol_node(i,j)=0;
587           else
588               vol_node(i,j)=volume_node(i,j)/vol_rep(kon_elementj(i,j)
                      ,5)*vol_rep(kon_elementj(i,j),6);
589           end
590       end
591   end
592    %{
593     %the code below tells if two voronoi elements share the same set
          of
594     %common delaunay nodes or not which turns out to be not
595     count_rep_common_del_nodes=1;
596     for i=1:length(common_del_nodes)
597         for j=i+1:length(common_del_nodes)
598             if(common_del_nodes(i,1)==common_del_nodes(j,1)||
                    common_del_nodes(i,1)==common_del_nodes(j,2)||
                    common_del_nodes(i,1)==common_del_nodes(j,3))
599                 if(common_del_nodes(i,2)==common_del_nodes(j,1)||
                        common_del_nodes(i,2)==common_del_nodes(j,2)||
                        common_del_nodes(i,2)==common_del_nodes(j,3))
600                     if(common_del_nodes(i,3)==common_del_nodes(j,1)||
                            common_del_nodes(i,3)==common_del_nodes(j,2)||
                            common_del_nodes(i,3)==common_del_nodes(j,3))
601                         index_rep_common_del_nodes(
                                count_rep_common_del_nodes,1)=j;
602                         count_rep_common_del_nodes=
                                count_rep_common_del_nodes+1;
603                     end
604                 end
605             end
606         end
```

```
607         end
608         index_rep_common_del_nodes=unique(index_rep_common_del_nodes);
609        %}
610
611 %code to check if there are nodes with a negative coordinate in a
612 %connection (which turns out that there arent!)
613 %count1=0;
614 %count2=0;
615 %for i=1:1:max(size(voronoiz))
616  %        if(v(voronoiz(i,1))<-0.00000000000001)
617   %           count1=count1+1;
618    %          id=i;
619     %         break;
620       %   end
621      %            if(v(voronoiz(i,2))<-0.00000000000001)
622        %      count2=count2+1;
623         %     id=i;
624          %    break;
625         %end
626 %end
627 %%%%%%% VORONOIZ-VORONOI FACETS; V-THEIR COORDINATES!!! %%%%%%%%%
628 %v(1,:)=[rand(1),rand(1)/2,rand(1)/3];
629 % for ii=1:qq
630 % abc=original{ii};
631 %      for j=1:max(size(oriqinal{ii}))
632 %          for k=1:length(v)
633 %              if abc(j,:)==v(k,:)
634 %                  numeracija
635
636 % for i=1:qq
637 %      for j=1:
638 save('v.mat','v'); %nodal coordinates of end points of facet vertices
639 save('elements_voronoi.mat','kon_elementj');% connectivity of those end
        points/voronoi elements
640 save('area_voronoi.mat','area_voronoi'); %area of the voronoi elements
641 save('elements_delaunay.mat','kon_elementi');%connectivity of the delaunay
       elements
642 save('area_delaunay.mat','area_delaunay');%area of the delaunay elements
643 save('length_cracks.mat','length_cracks');%lengths of cracks around a
        voronoi element
644 save('vol_node.mat','vol_node');%nodal contribution of volume in an
        element
645 fdi1=fopen('Number_of_delaunay_elements_generated.txt','w');
646 fprintf(fdi1,'%g',max(size(kon_elementi)));
647 fprintf(fdi1,'\n');
648 time=toc/3600;
649
650 fprintf(fdi1,'time=%f',time);
651 fclose(fdi1);
652
653 fdi2=fopen('Total execution time.txt','w');
654 % fprintf(fdi,'%g',max(size(kon_elementi)));
655  fprintf(fdi2,'\n');
656  time=toc/3600;
657 %
658 fprintf(fdi2,'time=%f',time);
```

```
659 fclose(fdi2);
```

The functions used in this script are given below:

```
1  function[area3]=area_facet(vertice,a,b,c)
2  % Calculate the area of a 3D convex hull using %
3  % a projection of a polygon onto a plane %
4  % Snyder & Barr 1987 %
5  % Internet page :
6  % http://softsurfer.com/Archive/algorithm_0101/algorithm_0101.htm#3D%20
       Polygons
7  % Input data:
8  % Vertice coordinates in 3D
9  % Output data:
10 % area3- Area of the convex facet in 3D
11 a;
12 b;
13 c;
14 vertice; % full vector of coordinates
15 % First 3 points are used to calculate the unit normal %
16 A=vertice(1,:);
17 B=vertice(2,:);
18 C=vertice(3,:);
19
20 v1=B-A; % vector 1
21 v2=C-A; % vector 2
22
23 normal=cross(v1,v2); % calculate the normal vector using the cross product
24
25 l=sqrt(normal(1)^2+normal(2)^2+normal(3)^2); % normal vector length;
26
27 unit_normal=normal/l; % unit normal
28
29 [a1 b1]=max(abs(unit_normal)); % find max abs value in the unit normal and
       its position
30
31 % a- abs max value in the unit vector %
32 % b- its position %
33 for i=1:1:max(size(vertice))
34     if(vertice(i,1)<0)
35         vertice(i,1)=0;
36     end
37     if(vertice(i,2)<0)
38         vertice(i,2)=0;
39     end
40     if(vertice(i,3)<0)
41         vertice(i,3)=0;
42     end
43     if(vertice(i,1)>a)
44         vertice(i,1)=a;
45     end
46     if(vertice(i,2)>b)
47         vertice(i,2)=b;
48     end
49     if(vertice(i,3)>c)
50         vertice(i,3)=c;
51     end
```

```
end

if b1==1
    [K,area]=convhull(vertice(:,2),vertice(:,3));
elseif b1==2
    [K,area]=convhull(vertice(:,1),vertice(:,3));
else
    [K,area]=convhull(vertice(:,1),vertice(:,2));
end

clear('K');
area3=area/a1;

% Get the area in 3D %
```

```
function[K]=area_facet_new(vertice,a,b,c)
% Calculate the area of a 3D convex hull using %
% a projection of a polygon onto a plane %
% Snyder & Barr 1987 %
% Internet page :
% http://softsurfer.com/Archive/algorithm_0101/algorithm_0101.htm#3D%20
    Polygons
% Input data:
% Vertice coordinates in 3D
% Output data:
% area3- Area of the convex facet in 3D
a;
b;
c;
vertice; % full vector of coordinates
% First 3 points are used to calculate the unit normal %
A=vertice(1,:);
B=vertice(2,:);
C=vertice(3,:);

v1=B-A; % vector 1
v2=C-A; % vector 2

normal=cross(v1,v2); % calculate the normal vector using the cross product

l=sqrt(normal(1)^2+normal(2)^2+normal(3)^2); % normal vector length;

unit_normal=normal/l; % unit normal

[a1 b1]=max(abs(unit_normal)); % find max abs value in the unit normal and
    its position

% a- abs max value in the unit vector %
% b- its position %


for i=1:1:max(size(vertice))
    if(vertice(i,1)<0)
        vertice(i,1)=0;
    end
    if(vertice(i,2)<0)
```

```
        vertice(i,2)=0;
    end
    if(vertice(i,3)<0)
        vertice(i,3)=0;
    end
    if(vertice(i,1)>a)
        vertice(i,1)=a;
    end
    if(vertice(i,2)>b)
        vertice(i,2)=b;
    end
    if(vertice(i,3)>c)
        vertice(i,3)=c;
    end
end



if b1==1
    [K,area]=convhull(vertice(:,2),vertice(:,3));
elseif b1==2
    [K,area]=convhull(vertice(:,1),vertice(:,3));
else
    [K,area]=convhull(vertice(:,1),vertice(:,2));
end

clear('area');


% Get the area in 3D %
```

```
function[area3]=area_facet(vertice)
% Calculate the area of a 3D convex hull using %
% a projection of a polygon onto a plane %
% Snyder & Barr 1987 %
% Internet page :
% http://softsurfer.com/Archive/algorithm_0101/algorithm_0101.htm#3D%20
    Polygons
% Input data:
% Vertice coordinates in 3D
% Output data:
% area3- Area of the convex facet in 3D

vertice; % full vector of coordinates
% First 3 points are used to calculate the unit normal %
A=vertice(1,:);
B=vertice(2,:);
C=vertice(3,:);

v1=B-A; % vector 1
v2=C-A; % vector 2

normal=cross(v1,v2); % calculate the normal vector using the cross product

l=sqrt(normal(1)^2+normal(2)^2+normal(3)^2); % normal vector length;

```

```
unit_normal=normal/l; % unit normal

[a b]=max(abs(unit_normal)); % find max abs value in the unit normal and
    its position

% a- abs max value in the unit vector %
% b- its position %

if b==1
    [K,area]=convhull(vertice(:,2),vertice(:,3));
elseif b==2
    [K,area]=convhull(vertice(:,1),vertice(:,3));
else
    [K,area]=convhull(vertice(:,1),vertice(:,2));
end

clear('K');

area3=area/a;

% Get the area in 3D %
```

```
function [ volume ] = volume_tetrahedra( P1,P2,P3,P4 )
%VOLUME_TETRAHEDRA Summary of this function goes here
%   Detailed explanation goes here

volume=1/6*abs(dot(cross(P2-P1,P3-P1),P4-P1));
end
```

Generation of an Artificial Discrete Crack

This section contains the script used generate an artificial crack in the discretized domain.

```
%in this code an artificial wedge type crack is generated so that it can
    be
%viewed in paraview. In the second part of the code, the crack width is
%assigned to the delaunay element and further to the voronoi elements

clear;

load elements_delaunay.mat % load the result from Delaunay's triangulation
load area_delaunay.mat; % the same
load v.mat;
load elements_voronoi.mat;
load area_voronoi.mat;
load length_cracks.mat;
load common_del_nodes.mat;
load common_del_ele.mat;
fileid=fopen('mesh.opt','r');
OPT=fscanf(fileid,'%g',5);
NoN=OPT(1); % Total number of nodes
fclose(fileid);
clear('OPT');
fdi1=fopen('Number_of_delaunay_elements_generated.txt','r');
NoE=fscanf(fdi1,'%g',1); % total number of elements in the triangulation
fclose(fdi1);

```

```
fid1=fopen('nodes3D.dat','r'); % load the initial node coordinates

A=fscanf(fid1,'%f',[4 NoN]); % this needs to be ajdusted if the overall
    number of nodes is changed, i.e. [4 NNODES]
A=A';
A=A(:,2:4); % the initial node coordinates
fclose(fid1);
%%
%this part of the code is for making the paraview file so that the cracked
%elements can be visualized
analys_step=1;
crack_depth=75;
crack_z_lower=70;
crack_z_mid=75;
crack_z_upper=80;
count_crack=1;
for i=1:length(kon_elementi)
    if(A(kon_elementi(i,1),3)>=crack_z_mid && A(kon_elementi(i,1),3)<=
        crack_z_upper && A(kon_elementi(i,1),1)<=crack_depth)
        if(A(kon_elementi(i,2),3)>=crack_z_lower && A(kon_elementi(i,2),3)
            <=crack_z_mid && A(kon_elementi(i,2),1)<=crack_depth)
            elements_cracked(count_crack,1)=kon_elementi(i,1);
            elements_cracked(count_crack,2)=kon_elementi(i,2);
            element_crack_id(count_crack,1)=i;
            count_crack=count_crack+1;
        end
    end
     if(A(kon_elementi(i,2),3)>=crack_z_mid && A(kon_elementi(i,2),3)<=
         crack_z_upper && A(kon_elementi(i,2),1)<=crack_depth)
        if(A(kon_elementi(i,1),3)>=crack_z_lower && A(kon_elementi(i,1),3)
            <=crack_z_mid && A(kon_elementi(i,1),1)<=crack_depth)
            elements_cracked(count_crack,1)=kon_elementi(i,1);
            elements_cracked(count_crack,2)=kon_elementi(i,2);
            element_crack_id(count_crack,1)=i;
            count_crack=count_crack+1;
        end
    end
end

%element_crack_id contains the id of the delaunay elements that is cracked
count_nodes=1;
for i=1:1:length(elements_cracked)
    nodes_cracked_temp(count_nodes,2:4)=A(elements_cracked(i,1),1:3);
    nodes_cracked_temp(count_nodes,1)=elements_cracked(i,1);
    count_nodes=count_nodes+1;
    nodes_cracked_temp(count_nodes,2:4)=A(elements_cracked(i,2),1:3);
    nodes_cracked_temp(count_nodes,1)=elements_cracked(i,2);
    count_nodes=count_nodes+1;

end

nodes_cracked_temp=unique(nodes_cracked_temp,'rows');
nodes_cracked=nodes_cracked_temp(:,2:4);
for i=1:length(nodes_cracked_temp)
node_change_temp(nodes_cracked_temp(i,1),1)=i;
end
```

```
for i=1:length(elements_cracked)
elements_cracked_temp(i,1)=node_change_temp(elements_cracked(i,1),1);
elements_cracked_temp(i,2)=node_change_temp(elements_cracked(i,2),1);
end
elements_cracked=[];
elements_cracked=elements_cracked_temp;
clear('elements_cracked_temp');
clear('node_change_temp');




%till here now we have two matrices, elements_cracked and nodes_cracked.
%elements_cracked contains the set of nodal ids that from the cracked
%element and nodes_cracked has the nodal coordinates of the cracked
%element.


fname1=sprintf('cracked_%d.ele',analys_step);
 file=fopen(fname1,'w');

p=0;

% L1=zeros(NoE,1);
% for i=1:NoE
%     disp(i/NoE*100);
%     L1(i)=sqrt((A(kon_elementi(i,1),1)-A(kon_elementi(i,2),1))^2+(A(
    kon_elementi(i,1),2)-A(kon_elementi(i,2),2))^2+(A(kon_elementi(i,1),3)-
    A(kon_elementi(i,2),3))^2);
% end

for i=1:length(elements_cracked)
    n1=elements_cracked(i,1);
    n2=elements_cracked(i,2);
r(i)=1;
%    if (AA(n1)==0 && AA(n2)==0)
%             fprintf(file,'%g %g %g %f %g %f %g %g %f %f %f %f %g %g %g %g
    %f %g %g',i,n1,n2,r(i),0,1.1,1,1,characteristics(1,2),characteristics
    (1,3),characteristics(1,4),characteristics(1,5),0,2,1,1,0.05,1,
    characteristics(1,1)); %1
             fprintf(file,'%g %g %g %f %g %f %g %g %f %f %f %f %g %g %g %f
                 %f %g %g',i,n1,n2,r(i)
                ,0,1.1,3,1,46,19.1667,0.0035,-0.035,0,2,1,0.75,0.75,1,1);
                %1
             fprintf(file,'\n');
%     elseif (AA(n1)==0 && AA(n2)==1) ||  (AA(n2)==1 && AA(n1)==0)
%             fprintf(file,'%g %g %g %f %g %f %g %g %f %f %f %f %g %g %g %g
    %f %g %g',i,n1,n2,0.25,0,1.1,1,1,15,6,0.001,-5,0,2,1,1,0.05,1,2);
%             fprintf(file,'\n');
%     else
%             fprintf(file,'%g %g %g %f %g %f %g %g %f %f %f %f %g %g %g %g
    %f %g %g',i,n1,n2,0.25,0,1.1,1,1,70,28,0.008,-5,0,2,1,1,0.05,1,3);
%             fprintf(file,'\n');
%     end
end

```

```
121 fclose(file);
122
123 fname2=sprintf('cracked_%d.nod',analys_step);
124 fid7=fopen(fname2,'w');
125 for i=1:length(nodes_cracked)
126     fprintf(fid7,'%g %f %f %f %g',i,nodes_cracked(i,1),nodes_cracked(i,2),
        nodes_cracked(i,3),1);
127     fprintf(fid7,'\n');
128 end
129
130 fclose(fid7);
131
132 %%
133 %this part of the code inputs element_crack_id from the previous section
    of
134 %the code which has the ids of the deleted elements. That is required to
    assign crack widths to the elements.
135
136 %the elements are assigned the crack width according to their x-coordinate
    .
137 %As the x-coordainte increases, the crack width decreases. The process
    used
138 %here is that a delaunay element is taken and the average of the
139 %x-coordinate of both of it's nodes is calculated. Then the crack width is
140 %assigned according to linear interpolation of the crack width.
141
142
143 max_crack_width=0.2;
144 min_crack_width=0.2;%using these values, the crack is a planar crack with
    constant width
145
146 for i=1:length(element_crack_id)
147     element_crack_width(i,1)=element_crack_id(i,1);
148     avg_element_depth=0.5*(A(kon_elementi(element_crack_id(i,1),1),1)+A(
        kon_elementi(element_crack_id(i,1),2),1));
149     element_crack_width(i,2)=max_crack_width-abs(max_crack_width-
        min_crack_width)*avg_element_depth/crack_depth;
150 end
151 %element_crack_width contains in the 1st column the delaunay element id
    and
152 %in the 2nd column the crack width
153
154
155 %here the crack lengths are calculated for all the voronoi elements
156 size_voronoi=max(size(kon_elementj));
157 voronoi_crack=zeros(size_voronoi,6);
158 for i=1:length(kon_elementj)
159     vertice(1,:)=A(common_del_nodes(i,1),:);
160     vertice(2,:)=A(common_del_nodes(i,2),:);
161     vertice(3,:)=A(common_del_nodes(i,3),:);
162
163     voronoi_crack(i,1:3)=crack_length(vertice);
164     vertice=[];
165 end
166
167
```

```
168
169 %here the crack widths are assigned to the voronoi elements according to
170 %the cracked delaunay elements.
171 for i=1:length(element_crack_width)
172     for j=1:length(kon_elementj)
173         if(element_crack_width(i,1)==common_del_ele(j,1))
174             voronoi_crack(j,4)=element_crack_width(i,2);
175         end
176         if(element_crack_width(i,1)==common_del_ele(j,2))
177             voronoi_crack(j,5)=element_crack_width(i,2);
178         end
179         if(element_crack_width(i,1)==common_del_ele(j,3))
180             voronoi_crack(j,6)=element_crack_width(i,2);
181         end
182     end
183 end
184
185 file2=fopen('Input_voronoi_cracks_art.cr','w');
186 for i=1:length(voronoi_crack)
187     fprintf(file2,'%f %f %f %f %f %f',voronoi_crack(i,1),voronoi_crack(i
        ,2),voronoi_crack(i,3),voronoi_crack(i,4),voronoi_crack(i,5),
        voronoi_crack(i,6));
188     fprintf(file2,'\n');
189 end
190 fclose(file2);
191
192 %the voronoi elements that are passing through a cracked delaunay element
193 %should have the moisture content of both it's nodes as 1 at all times. So
194 %a list of all the voronoi nodes is required that have to be maintained at
      the same moisture content throughout.
195 %this list of nodes will be in Bo_crack
196 count_bo_node=1;
197 count_crack_ele=0;
198 for i=1:length(voronoi_crack)
199     if(voronoi_crack(i,4)==0 && voronoi_crack(i,5)==0 && voronoi_crack(i
        ,6)==0)
200         filler=1;
201     else
202         Bo_crack(count_bo_node,1)=kon_elementj(i,1);
203         count_bo_node=count_bo_node+1;
204         Bo_crack(count_bo_node,1)=kon_elementj(i,2);
205         count_bo_node=count_bo_node+1;
206       count_crack_ele=count_crack_ele+1;
207     end
208 end
209 Bo_crack=unique(Bo_crack);
210
211 file3=fopen('Bo_crack_art.cr','w');
212 for i=1:length(Bo_crack)
213     fprintf(file3,'%g',Bo_crack(i,1));
214     fprintf(file3,'\n');
215 end
216 fclose(file3);
217
218 count_vor_crack_ele=1;
219 for i=1:length(voronoi_crack)
```

```
    if(voronoi_crack(i,4)==0 && voronoi_crack(i,5)==0 && voronoi_crack(i
        ,6)==0)
            filler=1;
    else
        vor_crack_ele(count_vor_crack_ele,1)=i;
        count_vor_crack_ele=count_vor_crack_ele+1;
    end
end

save('element_crack_id.mat','element_crack_id');
save('vor_crack_ele.mat','vor_crack_ele');
```

Projection of Aggregates on Voronoi Elements

This section contains the script used for the projection of spherical aggregates on Voronoi elements. The main script is given below:

```
clear all
clc
tic

load elements_delaunay.mat % load the result from Delaunay's triangulation
load area_delaunay.mat; % the same
load v.mat;
load elements_voronoi.mat;
load area_voronoi.mat;
load length_cracks.mat;
load common_del_nodes.mat;
load common_del_ele.mat;
load element_crack_id.mat
%load vol_node.mat
load Radius.mat;
load center.mat;
fileid=fopen('mesh.opt','r');
OPT=fscanf(fileid,'%g',5);
NoN=OPT(1); % Total number of nodes
voxel_size=OPT(5);
fclose(fileid);
clear('OPT');
fdi1=fopen('Number_of_delaunay_elements_generated.txt','r');
NoE=fscanf(fdi1,'%g',1); % total number of elements in the triangulation
fclose(fdi1);

fid1=fopen('nodes3D.dat','r'); % load the initial node coordinates

A=fscanf(fid1,'%f',[4 NoN]); % this needs to be ajdusted if the overall
    number of nodes is changed, i.e. [4 NNODES]
A=A';
A=A(:,2:4); % the initial node coordinates
fclose(fid1);

length_voronoi=length_element(v,kon_elementj);
boundary_thickness=0.5*mean(length_voronoi);

%x_cuboid=1:77;
%y_cuboid=0:150;
%z_cuboid=72:78;
```

```
40
41 %[x_cub,y_cub,z_cub]=meshgrid(x_cuboid,y_cuboid,z_cuboid);
42 %Pcub=[x_cub(:),y_cub(:),z_cub(:)];
43 %shp_cub = alphaShape(Pcub(:,1),Pcub(:,2),Pcub(:,3));
44 %figure
45 %lot(shp_cub)
46 %%
47 %the voronoi elements that completely lie inside the aggregate
48 %for i=1:length(center)
49 %    voronoi_aggregate{i,1}=[];
50 %    voronoi_interface{i,1}=[];
51 %    voronoi_boundary{i,1}=[];
52 %end
53
54 for i=1:length(center)
55     [x,y,z] = sphere(40);
56     x=x(:);
57     y=y(:);
58     z=z(:);
59
60     x1=x*(Radius(i,1)-boundary_thickness)+center(i,1);
61     y1=y*(Radius(i,1)-boundary_thickness)+center(i,2);
62     z1=z*(Radius(i,1)-boundary_thickness)+center(i,3);
63
64     x2=x*(Radius(i,1)+boundary_thickness)+center(i,1);
65     y2=y*(Radius(i,1)+boundary_thickness)+center(i,2);
66     z2=z*(Radius(i,1)+boundary_thickness)+center(i,3);
67
68
69     P = [x1 y1 z1];
70     P = unique(P,'rows');
71
72
73     P2 = [x2 y2 z2];
74     P2 = unique(P2,'rows');
75
76     shp = alphaShape(P(:,1),P(:,2),P(:,3),(Radius(i,1)-boundary_thickness)
           );
77     shp2= alphaShape(P2(:,1),P2(:,2),P2(:,3),(Radius(i,1)+
           boundary_thickness));
78
79    % figure
80     %plot(shp)
81
82     %figure
83     %plot(shp2)
84
85     count_aggregate=1;
86     count_interface=1;
87     count_boundary=1;
88     for j=1:length(kon_elementj)
89         %this if statement is for the elements within the aggregates
90         if(inShape(shp,v(kon_elementj(j,1),:))==1 && inShape(shp,v(
               kon_elementj(j,2),:))==1)
91             voronoi_aggregate{i,1}(count_aggregate,1)=j;
92             count_aggregate=count_aggregate+1;
```

```
93          end
94          %the following two if statements are for elements on the interface
95           if(inShape(shp,v(kon_elementj(j,1),:))==1 && inShape(shp,v(
                kon_elementj(j,2),:))~=1)
96              voronoi_interface{i,1}(count_interface,1)=j;
97              count_interface=count_interface+1;
98          end
99           if(inShape(shp,v(kon_elementj(j,1),:))~=1 && inShape(shp,v(
                kon_elementj(j,2),:))==1)
100             voronoi_interface{i,1}(count_interface,1)=j;
101             count_interface=count_interface+1;
102         end
103         %the following if statements are for the ones that lie within the
104         %boundary thickness
105          if(inShape(shp,v(kon_elementj(j,1),:))~=1 && inShape(shp2,v(
                kon_elementj(j,1),:))==1)
106              if(inShape(shp,v(kon_elementj(j,2),:))~=1 && inShape(shp2,v(
                    kon_elementj(j,2),:))==1)
107                 voronoi_boundary{i,1}(count_boundary,1)=j;
108                 count_boundary=count_boundary+1;
109             end
110         end
111     end
112
113
114     %agg_bound_temp=vertcat(voronoi_aggregate{i,1},voronoi_interface{i,1},
            voronoi_boundary{i,1});
115     %agg_bound_temp=unique(agg_bound_temp);
116     %for k=1:length(agg_bound_temp)
117     %     if(inShape(shp_cub,v(kon_elementj(agg_bound_temp(k,1),1),:))==1
            || inShape(shp_cub,v(kon_elementj(agg_bound_temp(k,1),2),:))==1)
118     %         voronoi_aggregate{i,1}=[];
119     %         voronoi_interface{i,1}=[];
120     %         voronoi_boundary{i,1}=[];
121     %     end
122     %end
123
124
125     clear('agg_bound_temp');
126     clear('x');
127     clear('y');
128     clear('z');
129     clear('shp');
130     clear('P');
131
132      clear('x2');
133     clear('y2');
134     clear('z2');
135     clear('shp2');
136     clear('P2');
137     i/length(center)*100
138 end
139 %%
140 %making the paraview file for elements within the aggregates
141
142 elements_aggregate=[];
```

```
143 for i=1:length(voronoi_aggregate)
144     if(length(voronoi_aggregate{i})~=0)
145         elements_boundary_temp=voronoi_aggregate{i,1}(:,1);
146         elements_aggregate=vertcat(elements_aggregate,
                elements_boundary_temp);
147         clear('elements_boundary_temp');
148     end
149 end
150 elements_aggregate=unique(elements_aggregate);
151 elements_aggregate=kon_elementj(elements_aggregate,:);
152
153 paraview_file(elements_aggregate,v,'aggregate.ele','aggregate.nod');%the
    element and the node data
154
155
156 %%
157 %paraview file for the interface elements
158 elements_interface=[];
159 for i=1:length(voronoi_interface)
160     if(length(voronoi_interface{i})~=0)
161         elements_boundary_temp=voronoi_interface{i,1}(:,1);
162         elements_interface=vertcat(elements_interface,
                elements_boundary_temp);
163         clear('elements_boundary_temp');
164     end
165 end
166 elements_interface=unique(elements_interface);
167 elements_interface=kon_elementj(elements_interface,:);
168
169 paraview_file(elements_interface,v,'interface.ele','interface.nod');%the
    eleme  nt and the node data
170 %%
171
172 %paraview file for boundary elements
173 elements_boundary=[];
174 for i=1:length(voronoi_boundary)
175     if(length(voronoi_boundary{i})~=0)
176         elements_boundary_temp=voronoi_boundary{i,1}(:,1);
177         elements_boundary=vertcat(elements_boundary,elements_boundary_temp
                );
178         clear('elements_boundary_temp');
179     end
180 end
181 elements_boundary=unique(elements_boundary);
182 elements_boundary=kon_elementj(elements_boundary,:);
183
184 paraview_file(elements_boundary,v,'boundary.ele','boundary.nod');%the
    element and the node data
185
186 time=toc;
187 %%
188 count_agg_bod=1;
189 for i=1:length(voronoi_aggregate)
190     for j=1:length(voronoi_aggregate{i,1})
191         aggregate_body(count_agg_bod,1)=voronoi_aggregate{i,1}(j,1);
192         count_agg_bod=count_agg_bod+1;
```

```
    end
end
for i=1:length(voronoi_interface)
    for j=1:length(voronoi_interface{i,1})
        aggregate_body(count_agg_bod,1)=voronoi_interface{i,1}(j,1);
        count_agg_bod=count_agg_bod+1;
    end
end
aggregate_body=unique(aggregate_body);

count_agg_bound=1;
for i=1:length(voronoi_boundary)
    for j=1:length(voronoi_boundary{i})
        aggregate_bound(count_agg_bound,1)=voronoi_boundary{i}(j,1);
        count_agg_bound=count_agg_bound+1;
    end
end
aggregate_bound=unique(aggregate_bound);
save('aggregate_body.mat','aggregate_body');
save('aggregate_bound.mat','aggregate_bound');
save('voronoi_aggregate.mat','voronoi_aggregate');
save('voronoi_boundary.mat','voronoi_boundary');
save('voronoi_interface.mat','voronoi_interface');

load volume_node.mat
for i=1:length(aggregate_body)
    volume_agg_bod(i,1)=volume_node(aggregate_body(i,1),1)+volume_node(
        aggregate_body(i,1),2);
end
for i=1:length(aggregate_bound)
    volume_agg_bound(i,1)=volume_node(aggregate_bound(i,1),1)+volume_node(
        aggregate_bound(i,1),2);
end
tot_vol_agg_bod=sum(volume_agg_bod);
tot_vol_agg_bound=sum(volume_agg_bound);
tot_vol_agg=tot_vol_agg_bod+tot_vol_agg_bound;

count_repeat=1;

a = aggregate_bound;
b = aggregate_body;
repeat=ismember(a,b);
repeat_sum=sum(repeat);
%repeat=unique(repeat);
```

The functions used in this script are given below:

```
function [ elements_cracked, nodes_cracked ] = paraview_file(
    elements_aggregate,v,notation1,notation2 )
%PARAVIEW_FILE Summary of this function goes here
%   Detailed explanation goes here

count_nodes=1;
for i=1:length(elements_aggregate)

    nodes_cracked_temp(count_nodes,2:4)=v(elements_aggregate(i,1),1:3);
    nodes_cracked_temp(count_nodes,1)=elements_aggregate(i,1);
```

```
    count_nodes=count_nodes+1;
    nodes_cracked_temp(count_nodes,2:4)=v(elements_aggregate(i,2),1:3);
    nodes_cracked_temp(count_nodes,1)=elements_aggregate(i,2);
    count_nodes=count_nodes+1;

end

nodes_cracked_temp=unique(nodes_cracked_temp,'rows');
nodes_cracked=nodes_cracked_temp(:,2:4);
for i=1:length(nodes_cracked_temp)
node_change_temp(nodes_cracked_temp(i,1),1)=i;
end
for i=1:length(elements_aggregate)
elements_cracked_temp(i,1)=node_change_temp(elements_aggregate(i,1),1);
elements_cracked_temp(i,2)=node_change_temp(elements_aggregate(i,2),1);
end
elements_cracked=[];
elements_cracked=elements_cracked_temp;
clear('elements_cracked_temp');
clear('node_change_temp');
clear('nodes_cracked_temp');



fname1=sprintf(notation1);
 file=fopen(fname1,'w');

p=0;



for i=1:length(elements_cracked)
    n1=elements_cracked(i,1);
    n2=elements_cracked(i,2);
r(i)=1;

            fprintf(file,'%g %g %g %f %g %f %g %g %f %f %f %f %g %g %g %f
                %f %g %g',i,n1,n2,r(i)
                ,0,1.1,3,1,46,19.1667,0.0035,-0.035,0,2,1,0.75,0.75,1,1);
            fprintf(file,'\n');

end

fclose(file);

fname2=sprintf(notation2);
fid7=fopen(fname2,'w');
for i=1:length(nodes_cracked)
    fprintf(fid7,'%g %f %f %f %g',i,nodes_cracked(i,1),nodes_cracked(i,2),
        nodes_cracked(i,3),1);
    fprintf(fid7,'\n');
end

fclose(fid7);

end
```

Flow Simulation in the Discretized Domain

This section presents the script used to simulate flow in discretized domain. The script presented here is the one that uses implicit time stepping scheme and exponential variation of diffusivity over the element.

The main script is given below:

```
%% About (last version 29/09/2017)
%saturation form of Richards equation w/o forcing vector
%for every new cementitious matrix and for each mesh size, the ratio
%between transport properties of itz and matrix should be validated
%for Xinxin Li et al (2017) this ratio seems to be between 1.3 and 2

%% Input data

clear;
tic
Read_input_PSM; % data from the input file
L=[]; %initialize the element length vector
L=single(L); % single precision

% calculate and store element lengths %
for i=1:NoE
    L(i)=sqrt((coord(elements(i,1),1)-coord(elements(i,2),1))^2+(coord(
        elements(i,1),2)-coord(elements(i,2),2))^2+(coord(elements(i,1),3)-
        coord(elements(i,2),3))^2);
end
L=L';

for i=1:length(elements)
    omega(i,1)=elements(i,3)*L(i)/(volume_agg(i,1)+volume_agg(i,2));
end
L_uni=unique(L);
for i=1:NoE
     D_c=elements_ini_diff(i,1)*exp(nDe(NeT1,2)*coord(elements(i,1),5));
        n_c=nDe(NeT1,2)/L(i)*(coord(elements(i,2),5)-coord(elements(i,1)
            ,5));

        diff=0.34785*(L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.861136))+L(
            i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*0.861136)))+0.652145*(L(i)
            /2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.339981))+L(i)/2*D_c*exp(
            n_c*(0.5*L(i)+0.5*L(i)*0.339981)));
        elements(i,4)=diff;
end
%omega=omega_cal(elements(:,3),L,sum(volume));
%{
for i=1:length(elements)
    omega(i)=omega_cal(elements(i,3),L(i),volume(i,1),volume(i,2));
end

for i=1:NoE
                % 4 point gauss integration
    D_c=D0e(NeT1,2)*exp(nDe(NeT1,2)*coord(elements(i,1),5));
    n_c=nDe(NeT1,2)/L(i)*abs(coord(elements(i,2),5)-coord(elements(i,1),5)
        );

    diff=0.34785*(L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.861136))+L(i)
        /2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*0.861136)))+0.652145*(L(i)/2*D_c*
```

```
        exp(n_c*(0.5*L(i)+0.5*L(i)*-0.339981))+L(i)/2*D_c*exp(n_c*(0.5*L(i)
        +0.5*L(i)*0.339981)));
    elements(i,4)=diff;
end
%}
%% Calculate elemental [m], initial [k] and assemble them in Sparse global
    matrices [M],initial [K]


I=zeros(1,4*NoE); % initialize indexing for sparse input rows
J=zeros(1,4*NoE); % initialize indexing for sparse input columns
Stiff=zeros(1,4*NoE); % initialize sparse values of K
Mass=zeros(1,4*NoE); % initialize sparse values of M (4 due to the
    different elements in the local elemental matrix)

count=0;

for i=1:NoE
    n1=coord(elements(i,1),4);
    n2=coord(elements(i,2),4);
    [m]=m_matrix(omega(i),elements(i,3),L(i)); % element m matrix
    [k]=k_matrix_quad(elements(i,4),elements(i,3),L(i)); % element k
        matrix is different from zero for unsaturated elements
    len=length(I);
    for irow=1:2
        for icol=1:2
            count=count+1;
            I(count)=elements(i,irow); %subscript i for global indexing
                coincident with the node of the element influencing
            J(count)=elements(i,icol); %subscript j for global indexing
                coincident with the node of the element being influenced
            Mass(count)=m(irow,icol);
            Stiff(count)=k(irow,icol);
        end
    end
end

K_orig=sparse(I,J,Stiff,NoN,NoN);
M_orig=sparse(I,J,Mass,NoN,NoN);


%% Boundary and Initial conditions
for time_step=1:length(deltaT)
LHS=M_orig+0.5*deltaT(time_step)*K_orig; % left hand side
RHS=M_orig-0.5*deltaT(time_step)*K_orig; %+deltaT*F; % right hand side

% Initial Condition (t=0)
teta=sparse(NoN,1); % initalize the potential phi vector
teta(1:NoN,1)=coord(:,5); % define potential phi at the start of the
    analysis (eq2)


% Apply the boundary conditions on the LHS

% Boundary conditions top boundary

```

```
92  LHS(Bo,:)=0; % together with the loop below constitutes a trick to make
        that all the elements departing from BN contribute to mantain the same
        phi in the BN from start to end
93  for i=1:size(Bo)
94      LHS(Bo(i),:)=0;
95      LHS(Bo(i),Bo(i))=1; % in order that phi is the same at every time step
            for Boundary nodes
96  end
97
98
99  % Boundary conditions lateral surfaces (if needed)
100
101 %% Print intial
102 fname=sprintf('Diffusivity_%d.dat',0);
103 fname1=sprintf('Saturation_%d.dat',0);
104 fid=fopen(fname,'w');
105 fid1=fopen(fname1,'w');
106 teta=full(teta);
107
108 for i=1:NoN
109     fprintf(fid1,'%g\n',round(teta(i),4));
110 end
111 for j=1:NoE
112     fprintf(fid,'%g\n',round(elements(j,4),4));
113 end
114 fclose(fid);
115 fclose(fid1);
116
117 teta=sparse(teta);
118
119 %% Start the time dependent analysis
120 %tol=zeros(NoN,1);
121 tol=0.000001;
122 loop=1;
123 iterations=zeros(totalT,1);
124 for t=1:totalT
125
126
127     % 1-Solves the main equation and outputs phi % OK
128     Right=RHS*teta;
129     Right(Bo)=Boundary; % calculate phi_b as phi for teta=1 in the gls
            input
130
131     while(loop>0)
132     Left=LHS*teta;
133
134     delta_theta=error_norm3(Left,Right);
135
136     if(delta_theta>tol)
137         iterations(t,1)=iterations(t,1)+1;
138         resi=Right-Left;
139         delta_teta=LHS\resi;
140         teta=teta+delta_teta;
141
142
143         coord(:,5)=teta;
```

```
        %for i=1:length(elements)
            %quad_moist_cont(i,1)=teta(elements(i,1),1);
            %quad_moist_cont(i,2)=teta(elements(i,2),1);
        %end
        %quad_diff{t}{iterations(t,1)}(:,1)=coord(:,5);
    for i=1:NoE %% for loop (for each element finds the average -AS PER
        NOW- of the nodal tetas and
                % calculates the new value of the diffusivity coefficient
                    for the
                % elements and stores them in elements(i,4)
         %{
        teta_ele=(coord(elements(i,1),5)+coord(elements(i,2),5))/2;
        NeT1=1;
        while NeT1<=NeT
            if elements(i,5)==NeT1
                elements(i,4)=D0e(NeT1,2)*exp(nDe(NeT1,2)*teta_ele);
                elements(i,6)=teta_ele;
                break
            else
                NeT1=NeT1+1;
            end
        end
        %}
        D_c=elements_ini_diff(i,1)*exp(nDe(NeT1,2)*coord(elements(i,1),5))
            ;
        n_c=nDe(NeT1,2)/L(i)*(coord(elements(i,2),5)-coord(elements(i,1)
            ,5));

        diff=0.34785*(L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.861136))+L(
            i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*0.861136)))+0.652145*(L(i)
            /2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.339981))+L(i)/2*D_c*exp(
            n_c*(0.5*L(i)+0.5*L(i)*0.339981)));
        elements(i,4)=diff;
        %quad_diff{t}{iterations(t,1)}(i,1)=diff;

                %{
            D_c=D0e(NeT1,2)*exp(nDe(NeT1,2)*coord(elements(i,1),5));
            n_c=nDe(NeT1,2)/L(i)*(coord(elements(i,2),5)-coord(elements(i
                ,1),5));

            diff=0.34785*(L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.861136)
                )+L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*0.861136)))
                +0.652145*(L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*-0.339981)
                )+L(i)/2*D_c*exp(n_c*(0.5*L(i)+0.5*L(i)*0.339981)));
            elements(i,4)=diff;
                %}
    end
    % quad_diff{t}{iterations(t,1)}(:,2:3)=quad_moist_cont;
     %quad_moist_cont=[];

     % 3-Updates the Stiffness matrix, LHS and RHS %  OK
     count=0;

     for i=1:NoE
         [k]=k_matrix_quad(elements(i,4),elements(i,3),L(i)); %element k
             matrix is different from zero for unsaturated elements
```

```
188         len=length(I);
189         for irow=1:2
190             for icol=1:2
191                 count=count+1;
192                 I(count)=elements(i,irow); %subscript i for global
                        indexing coincident with the node of the element
193                 J(count)=elements(i,icol); %subscript j for global
                        indexing coincident with the node of the element
194                 Stiff(count)=k(irow,icol);
195             end
196         end
197     end
198     K=sparse(I,J,Stiff,NoN,NoN);
199     LHS=M_orig+0.5*deltaT(time_step)*K; % left hand side
200     RHS=M_orig-0.5*deltaT(time_step)*K; % right hand side
201     SaN=find(teta(:,1)==1);
202     LHS(Bo,:)=0;
203     for i=1:size(Bo)
204         LHS(Bo(i),Bo(i))=1; % in order that phi is the same at every time
            step for Boundary nodes
205     end
206     else
207         break;
208     end
209     end
210
211
212
213     % 4-PLots t, phi, teta per each node every (plot_frequency) time steps
        % OK
214
215     pomoc1=rem(t,plot_frequency);
216      num=10^(numel(num2str(totalT)));%basically this is done so that
         results are stored in the ascending order of the time steps
217     xRound = ceil(totalT/num)*num;
218     num_id=xRound+t;
219     if pomoc1==0
220         fname=sprintf('Diffusivity_%d.dat',num_id);
221         fname1=sprintf('Saturation_%d.dat',num_id);
222         fid=fopen(fname,'w');
223         fid1=fopen(fname1,'w');
224         teta=full(teta);
225         for i=1:NoN
226            fprintf(fid1,'%g\n',round(teta(i),4));
227         end
228         for j=1:NoE
229            fprintf(fid,'%g\n',round(elements(j,4),4));
230         end
231         fclose(fid);
232         fclose(fid1);
233         teta=sparse(teta);
234     end
235
236    % 5-Shows the performed percentage of the analysis % OK
237         if SLVR==1
238             disp(t/totalT*100);
```

```
        end
    end
end
%%
%{
save('mean_error','mean_error');
for i=1:max(size(mean_error))
    x_plot(i,1)=i;
end
figure
plot(x_plot,mean_error(7,:))
%}
%%
time=toc;
beep;
```

The functions used in the main script are given below:

```
function [ k ] = k_matrix( D,A,l )
%k_MATRIX elemental Stiffness matrix: Calculates the element stiffness
    matrix
% input parameters:
% A–cross sectional area
% l–length of the element
% D–diffusion coefficient
% output:
% k–elemental stiffness matrix

k=[];
%k=single(k);


% Calculation of the element diffusion matrix
k(1,1)=D*A/l;
k(1,2)=–D*A/l;
k(2,1)=k(1,2);
k(2,2)=k(1,1);


end
```

```
function [ k ] = k_matrix_quad( D,A,l)
%k_MATRIX elemental Stiffness matrix: Calculates the element stiffness
    matrix
% input parameters:
% A–cross sectional area
% l–length of the element
% D–diffusion coefficient
% output:
% k–elemental stiffness matrix

k=[];
%k=single(k);


% Calculation of the element diffusion matrix
k(1,1)=D*A/(l*l);
```

```
k(1,2)=-D*A/(l*l);
k(2,1)=k(1,2);
k(2,2)=k(1,1);


end
```

```
function [m] = m_matrix( omega, A,l )
%m_PSM elemental Mass matrix: Calculates the element mass matrix
% input parameters:
% omega-correction factor(see input)
% A-cross sectional area
% l-length of the element
% output:
% m-elemental mass matrix

m=[]; % m=single(m);

% Calculation of the element mass matrix
m(1,1)=A*l*2/(6*omega);
m(1,2)=A*l/(6*omega);
m(2,1)=m(1,2);
m(2,2)=m(1,1);

end
```

```
function [ error ] = error_norm3( saturation_linear,saturation_iterative )

%In this error norm, the sum of the square of the absolte difference of
    the
%values is taken for all the nodes and then divided by the number of nodes
%and the the square root of that value is taken.


clear('error');
error=(abs(saturation_linear-saturation_iterative)).^2;
error=(sum(error))./length(error);
error=error.^(0.5);
%error=error';
clear('saturation_linear');
clear('saturation_iterative');
end
```

```
function [ D_cr ] = crack_diffusivity( elements_crack,area,theta)


rho=997*10^-9; %density (kg/mm3)
mu=8.9*10^-7;%dynamic viscosity (kg/mm/s)
xi=1;%crack tortuosity factor
m=0.4396; %Van_Genuchten parameter (pre determined and unitless)
pr=18.6237*10^-3; %reference pressure(N/mm2 but converted into kg/(mm.s2)
cube_law=0;
for i=1:3
    cube_law1=elements_crack(1,i)*elements_crack(1,i+3)^3;
    cube_law=cube_law+cube_law1;
```

```
13 end
14
15  exp1=xi*cube_law*pr*(1-m)/(12*mu*area*m);
16  exp2=(theta^0.5)*(1-(1-theta^(1/m))^m)^2*theta^(-1/(m-1))*(theta^(-1/m)
        -1)^(-m);
17  D_cr=abs(exp1*exp2);
18 end
```