Delft University of Technology

Proxy-model for Flow and Transport in Geothermal Reservoirs

Ahmad Mohammad Tahir

August 2019

Proxy-model for Flow and Transport in Geothermal Reservoirs

by

Ahmad Mohammad Tahir

to obtain the degree of **Master of Science** in **Petroleum Engineering and Geosciences** at the Delft University of Technology, defended publicly on Friday August 16, 2019 at 12:00 PM.

Department of Applied Earth Sciences

Student number:	4626737	
Project duration:	July 1, 2018 – August 16, 2019	
Supervisor:	Dr. Denis Voskov	
Committee:	Dr. Denis Voskov	TU Delft
	Prof. Dr. David F. Bruhn	TU Delft
	Dr. Maren Brehme	TU Delft
	Engr. Yang Wang	TU Delft

Abstract

The hot water produced from a geothermal doublet possesses energy, which once utilized, the water cools down and is re-introduced back into the same reservoir at a sufficient distance using an injector well. As cold water flows through the reservoir, it acquires thermal energy from surrounding in-situ rocks. This process recurs until a substantial drop in rock temperature occurs and the water is unable to be recharged adequately; as a result, cold water starts to "break-through" into the production well and the doublet soon needs to be abandoned.

This breakthrough time can be predicted using reservoir simulation. Significant work has been done in the past to determine the effect of different parameters on the breakthrough time and accuracy of models have been improved by incorporating real world physics. Recently, sensitivity of breakthrough time was investigated with changes in well doublet location. Just by relocating the well to a neighboring block, cold water experienced a completely different path attributed to large-scale geological heterogeneity, consequently resulting in a different breakthrough time [1]. Despite being capable to predict breakthrough time, accurate high-fidelity 3D models require significant time in uncertainty quantification and data assimilation analysis due to CPU demanding simulations.

In this project, a physics-based proxy model is developed to predict flow and heat transport in low-enthalpy reservoirs. Streamlines that describe flow in a system and mostly controlled by steady-state pressure distribution are traced using Pollock's method [2] and the reservoir is divided into streamtubes. Rock-heat depletion is modeled by semi-analytic model along streamlines. The objective is to predict geothermal doublet breakthrough time using only a limited number of streamtubes, thus minimizing simulation time and CPU resources.

Comparison with accurate high-fidelity model reveals that results for proxy model are optimistic; the error for pressure and temperature distributions, as well as the breakthrough curves is within the acceptable tolerance. The time required to simulate the proxy-model is less than the high-fidelity model. And as the number of streamtubes (to simulate the proxy model with) decrease, the time required for simulation further decreases but conversely the error between the breakthrough curves increases.

Ahmad Tahir
Delft, August 2019

Acknowledgements

Firstly, I would like to thank Dr. Denis Voskov who believed in me and gave me the opportunity to work on one of his key projects. It was due to his frequent guidance that I was able to complete this research with favorable results. Throughout the course of my thesis, he displayed great patience to clear my doubts and answer my queries, even the simplest ones. During a short period of time, I was also struggling personally but he always dealt with me leniently and I am grateful to him for bearing with me and showing his complete support.

I would also like to thank Engr. Yang Wang for always being available to help and spared me with his valuable time for long discussion; it would have been difficult to proceed without his help. I would like to thank Ms. Sanaz Saeid who helped me with the analytical solution of the heat equation. I would also like to thank Engr. Huzefa Ammiwala to help me understand the concepts of unstructured gridding that helped me implement the semi-analytical solution.

I am also grateful to my friends, who offered me help from time to time. It was due to their company and the occasional get-togethers that made the duration of my masters lively. It will be these moments that will be remembered the most, down the road.

I would like to thank my parents, who not only helped me financially (during the 3rd year of my Masters) but continuously encouraged me and gave me the moral support I needed to complete my study.

And lastly, but most importantly, I would like to thank the God Almighty who has brought me so far. If it was not for His will, I would never have made it to a reputable institution like TU Delft in the first place. Unlike most of my class-fellows, I do not belong to a Petroleum Engineering or Geosciences background. In addition, my residence permit approval for Netherlands was delayed by a month due to which I had suffered significantly at the very beginning of my Masters; it had a sort of "Domino effect" on my subsequent study. But it was God Almighty Who answered my prayers and helped me get up whenever I fell down.

Table of Contents

List of Figures

List of Tables

Nomenclature

The following terminology is used throughout the report, unless specified.

Abbreviations/Acronyms

Abbreviation	Description
BHP	Bottom Hole Pressure
COMSOL	finite element analysis, solver and multi-physics simulation software
DARTS	Delft Advanced Research Terra Simulator
FD	Finite Difference
GEA	Geothermal Energy Association
GW	Giga- (10^9) Watts
IPCC	Intergovernmental Panel on Climate Change
MATLAB	Matrix Laboratory (programming language developed by MathWorks)
N/G	Net-to-Gross
NPV	Net Present Value
OBL	Operator Based Linearization
PDE	Partial Differential Equation
TOF	Time of Flight
TW	Tera- (10^{12}) Watts
WNB	West Netherlands Basin

Symbols

General

Symbol	Description	Dimensions
A	area	L^2
D	depth (backward oriented)	L^1
K	permeability tensor	L^2
L	length	L^1
Ø	azimuth	*dimensionless*
Q	flow rate (e.g. heat or volumetric)	*not applicable*
T	temperature	Θ^1
c	specific heat capacity	$L^2\ T^{-2}\ \Theta^{-1}$
∂	partial differential operator	*not applicable*
$f(x)$	function of variable ‘x’	*not applicable*
g	acceleration due to gravity (9.8065)	$L^1\ T^{-2}$
h	specific enthalpy	$L^2\ T^{-2}$
i	specific internal energy	$L^2\ T^{-2}$
k_r	relative permeability	*dimensionless*
n	number of	*not applicable*
p	pressure	$M^1\ L^{-1}\ T^{-2}$
q	flux	*not applicable*

$\tilde{q}$	sink/source rate	T^{-1}
r	radius	L^{1}
s	saturation	*dimensionless*
t	time	T^{1}
u	Darcy velocity	L^{1} T^{-1}
v	velocity/interstitial velocity	L^{1} T^{-1}
x	particle position in x-dimension	L^{1}
y	particle position in y-dimension	L^{1}
z	particle position in z-dimension	L^{1}
Γ	transmissibility	
Ψ	velocity gradient	T^{-1}
γ	specific weight	M^{1} L^{-2} T^{-2}
$\dot{\varepsilon}$	energy sink/source	M^{1} L^{2} T^{-2}
ϵ	error	*not applicable*
ζ	distance along streamline	M^{1}
∇	gradient operator	M^{-1}
λ	thermal conductivity	M^{1} L^{1} T^{-3} Θ^{-1}
μ	viscosity	M^{1} L^{-1} T^{-1}
ξ	any parameter (e.g. Pressure, Temperature)	*not applicable*
ρ	density	M^{1} L^{-3}
$\dot{\rho}$	mass sink/source	M^{1} L^{-3} T^{-1}
τ	time of flight	T^{1}
ϕ	porosity	*dimensionless*
χ	mole fraction	*dimensionless*

Operator Based Linearization

Symbol	**Description**
J	Jacobian matrix
k	nonlinear iteration
r	residual vector
ω	state variables (on current time-step)
ξ	spatial coordinate variable
a	block a
b	block b
ω_n	state variables on previous time-step
v	well control variables
n_c	number of components
α_c	state-dependent operator for component c
β_c	state-dependent operator for component c
α_e	state-dependent operator
β_e	state-dependent operator
γ_e	state-dependent operator

Superscripts

Superscript	**Description**
$'$	derivative

n	current time-step
$n-1$	previous time-step
$n+1$	next time-step
ν	current non-linear iteration
$\nu+1$	next non-linear iteration

Subscripts

Subscript	**Description**
c	component
h	heat
i	cell index in x-dimension
j	cell index in y-dimension
k	cell index in z-dimension
m	mass
out	exit of particle from cell
p	phase
pt	particle
r	rock
v	volumetric
w	water component
x	x-dimension
y	y-dimension
z	z-dimension

1

Introduction

With the widely gaining acceptance of geothermal energy and the public awareness of harms of hydrocarbon energy utilization, geothermal energy development is on the rise. With this development, numerical simulation of geothermal reservoirs is becoming a very important tool [3]. It is not only useful for financial but also for operation management, such as determining the quantity of recoverable energy, strategies of field exploitation including the rate at which mass and energy may be extracted [4], injection and production well location and re-injection schemes [3].

Geothermal reservoir simulation, simply refers to reproducing and then predicting the behavior of geothermal reservoir using a numerical model. The very initial models of geothermal simulation appeared in the 1970’s but it was not until 1980 that these models were generally accepted for reservoir management [5].

The mathematical models which are defined as a set of equations that describe the physical processes present in the geothermal reservoir (e.g. energy and mass conservation equations) and the solution to these equations that are subjected to boundary and initial conditions. The construction of these models must be corroborated with detailed knowledge of spatial distribution of reservoir properties. The accuracy in collection of this data is fundamental for construction of an effective conceptual model [3]. Due to the complexity of underlying physical processes and considerable uncertainties in geological structure of reservoirs, there is a persistent demand for accurate and efficient models.

In order to increase the accuracy of a model, one can apply a finer computational grid in space or time, or use a more detailed description of the fluids such as in thermal-compositional model. However, an improvement in accuracy of models is usually counterbalanced by an increase in the simulation time. Moreover, the modern industry is challenged to run a large ensemble of numerical models for uncertainty analysis that requires thousands of simulations, therefore simulation performance becomes a critical issue [6].

Generating a model in which the natural distribution of parameters (e.g. temperature, pressure and fluid composition) is reproduced by simulator is known as “Initial-state modeling” [7]. But reservoir modeling and simulation should not be limited to initial-state model, and rather should be a continuous activity throughout the field life-time. As more geological information becomes available, the simulation model should be updated in order to yield a more accurate analysis of the reservoir [8]. History matching may be performed in which the impact of production and injection is reproduced by the numerical model and the initial-state model is re-run in order to ensure compatibility. This can also help in forecasting reservoir behavior under variety of possible future exploitation schemes [7].

1.1 Geothermal Systems

Geothermal energy is thermal energy that is generated and stored in the subsurface. This energy originates from two source: (1) residual heat from planet formation, mostly as gravitational and collisional energy, and (2) radiogenic heat, i.e. the decay of radioactive isotopes (e.g. Uranium-238, Potassium-40 and Thorium-232) [9].

Most of the energy, generated during Earth formation was released to the universe, estimated to be 15 – $35 \cdot 10^{30}$ J. The remaining energy from residual heat and radioactive decay amounts to be 12 – $24 \cdot 10^{30}$ J. Only 0.1% of the energy stored in Earth's crust (~ 10^{26} J) can satisfy the world energy consumption for next 10,000 years [9].

Around 22% of Earth's heat flow occurs due to conduction inside the inner solid core. While convection is responsible for almost half of the Earth's internal heat flow to the surface. And the remaining 24% of heat flow occurs in the crust primarily due to conduction, although some heat flow in crust also occurs by convection in areas of circulating fluids and by advection due to the rising of magma fluid below active volcanoes [10]. Most of the geothermal power stations operate on such convection-dominated systems. [11]

In order to harvest the geothermal energy, three things are required: (1) heat source, (2) fluid, (3) fluid pathways and (4) cap rock. The source of heat is usually a magma chamber that is a few kilometers below the surface. The fluid pathways in the subsurface can be in the form of pores, fractures or even Karsts. When the meteoric fluid (e.g. rain water) infiltrates the ground to a depth of few kilometers, it can return towards the surface after being heated, as steam or hot water. If this fluid is trapped under an impermeable layer of rock, i.e. cap rock, it can form a geothermal reservoir. But if there is sufficient pathway to the surface of earth, the hydrothermal systems can manifest as hot springs, geysers, fumaroles and mud pots. [9] [12]

Power generation from geothermal energy is cost-effective, reliable, sustainable and environmentally friendly, but it is not evenly distributed geographically. It tends to concentrate in environments with an ample volcanic activity and be controlled by plate tectonic processes [13]. Although recently advances in technology have expanded the range of viable resources (e.g. home heating), industrial applications need the higher temperatures from deeper resources [14]. Outside seasonal variations, the average temperature gradient through the crust is around 30˚C/km in most of the world [9]. The capacity factor of geothermal power is quite large (up to 96%) because it does not rely on variable energy sources (e.g. wind or solar) [15]. Geothermal wells may release greenhouse gases trapped inside the earth, but based on per energy unit, these emissions are much lower than those of fossil fuels. Most of the costs that are to incur with geothermal power are capital costs, while operational costs associated with geothermal reservoir are low and mainly due to the re-injection of the brine into the sub-surface.

Geothermal energy has been used in the form of water from hot springs for bathing since Paleolithic times and for space heating since ancient Roman times. Now, it is better known for power generation [16]. As of the end of year 2018, geothermal power is generated in 27 countries with total installed capacity of 14.6 GW [17]. The GEA estimates that only few percent of total global potential has been tapped so far. On the other hand, IPCC reported that geothermal power has a potential up to 2 TW [18].

Based on temperature ranges, the geothermal systems can be classified as either one of two types:

1) Low enthalpy systems (below 120˚C)
2) High enthalpy systems (above 120˚C)

In low enthalpy systems, the fluid exists in a liquid state and is best suited for direct application and household heating. Whereas in high enthalpy systems, the fluid may exist as steam as is suited for industrial application, such as power generation [19].

A geothermal reservoir is exploited with the help of well doublets. A "doublet" refers to a pair of wells where one serves as the producer (from which hot fluid is extracted out from the reservoir) and the other serves as injector (from which cooled fluid is injected back into the reservoir). When the hot fluid is extracted from reservoir, it is used for various applications. As the fluid is used, the temperature of the fluid, along with the exergy available, drops. This water is treated to remove salts and chemicals in order to comply with subsurface injection regulations. Next the cool water is re-injected into the reservoir. This re-injected water is used to provide pressure support in order to offset surface subsidence and prevent the induced seismicity due to pressure decline from mass extraction [9]. This re-injection also supplements the natural recharge, which is often limited but essential to sustain a geothermal system. While the sub-surface rocks are recharged by conduction, the fluid carrying the heat energy moves by advection. And during injection, this cold water front propagates through the reservoir towards the producer well. With the passage of time, the surrounding rock is unable to recharge the rocks in contact with the injected fluid, and hence unable to thermally recharge the fluid. Gradually the front of the cold water injected reaches the production wells and the temperature of the produced water begins to decline. Once the temperature of the produced water starts to decline, this is known as the "Breakthrough Time". But the breakthrough does not mean the immediate end of doublet lifetime, because it may take several years for the temperature of the produced fluid to drop by another degree, however breakthrough does serve as an indicator for nearing the end of doublet lifetime [9].

Simulation of geothermal processes have many complexities, many of which are not encountered in petroleum reservoirs. It involves solution of highly non-linear, coupled equations, that describe mass and energy transport in a heterogeneous media is a complex problem [5].

Convection cells arising from local differences in heat flux are encountered in the pre-exploited state, and both, energy and mass are in a state of dynamic equilibrium. In addition to the more conventional issues of reservoir structure, fault locations, permeability structure, reservoir boundaries are typically not sealed and can impact initializing a geothermal model. Conceptual models must capture heat flux from a variable heat source from below and heat loss to cap-rock or atmosphere (e.g., via fumaroles, steaming ground, etc.), It must also capture fluid recharge and discharge locations and magnitude [5]. Moreover the primary component of geothermal reservoirs is water that can exist in a vapor, liquid, or critical state. Phase behavior is further complicated by vapor pressure lowering and by the presence of non-condensable gases (e.g., CO_2) and salts. Phase changes (condensation and vaporization) occur in native state heat pipes and also because of injection/production operations. In addition, minerals may also precipitate or dissolve in response to phase change, affecting permeability and porosity in near-well regions [5]. These challenges can hinder the development of reliable models of these reservoirs.

The following equation shows the mass conservation equation:

$$\frac{\partial}{\partial t}\left(\phi\sum_{p=1}^{n_p}\chi_{c,p}\rho_p s_p\right)-\operatorname{div}\sum_{p=1}^{n_p}\chi_{c,p}\rho_p\left(K\frac{k_{r,p}}{\mu_p}\left(\nabla p_p-\gamma_p\nabla D\right)\right)+\sum_{p=1}^{n_p}\chi_{c,p}\rho_p\tilde{q}_p=0, \qquad c=1,\dots,n_c, \tag{1.1}$$

while energy conservation is given by the following equation [6]:

$$\frac{\partial}{\partial t}\left(\phi\sum_{p=1}^{n_p}\rho_p s_p u_{e,p}+(1-\phi)u_{e,r}\right)-\operatorname{div}\sum_{p=1}^{n_p}h_p\rho_p\left(K\frac{k_{r,p}}{\mu_p}\left(\nabla p_p-\gamma_p\nabla D\right)\right)+\operatorname{div}(\lambda\nabla T)+\sum_{p=1}^{n_p}h_p\rho_p\tilde{q}_p=0. \tag{1.2}$$

Note: Description of these variables can be found in the nomenclature

1.2 Previous Work:

Significant work has been done in the past to improve models in order to incorporate the real world physics and increase the accuracy in predicting geothermal reservoir breakthrough.

In [20], S. Saeid had introduced a new prototype design model for deep low-enthalpy hydrothermal systems. This model empirically predicted hydrothermal system lifetime as a function of porosity, discharge rate, well spacing, average initial reservoir temperature and injection temperature. In order to conduct a parametric analysis for wide range of physical parameters, finite element package COMSOL was utilized. The model was capable of predicting lifetime of low-enthalpy geothermal system with less than 10% error, but this was only valid for a limited range of parameters that had been assessed in the analysis. Based on parametric analysis, it was concluded that the discharge had the most significant effect on project lifetime and that seasonal variation of injection temperature had not a significant difference on average heat flow in production well. In addition, while implementing a computationally efficient finite element model for transient heat and fluid flow in deep low-enthalpy geothermal system utilizing COMSOL, the sensitivity of geothermal system to hydrological parameters was displayed, namely viscosity and porosity. It showed that dependency of viscosity on temperature can affect life of the system: with higher viscosity, propagation of cold front slows down, therefore increasing the expected geothermal system lifetime. Another example showed that variation of porosity can also affect the expected lifetime of system.

In [21], Cees Willems had improved the regional model of Lower Cretaceous, fluvial Nieuwerkerk Formation in order to obtain a better understanding of the lateral continuity of sandstone-rich successions and thickness variations of these successions throughout the WNB. His study showed that impermeable bodies can have a positive effect on heat exploitation as they provide thermal recharge to the cold water plume. This increases the doublet life time as the speed of temperature reduction after thermal breakthrough decreases. But at the same time, impermeable bodies, such as claystone, can reduce doublet life by separating permeable sandstone bodies into isolated clusters, consequently reducing the net aquifer volume and causing an early thermal breakthrough. He also showed that taking into account, the thermal recharge from under-burden and over-burden also increased life-time. He also showed that reduction in well spacing can still result in sufficient doublet life-time while the NPV of the project can be improved.

Cees Willems' study also showed that doublet interference has the most significant impact on the life-time. Relations indicate that required doublet distance is a function of the doublet configuration, minimal required production temperature and production rate variations between adjacent doublets. In tramline configuration (injector and producer wells are aligned), negative interference is still recognizable therefore the required doublet distance is large, as compared to checkboard configuration (injector and producer wells are alternated). The latter configuration can result in positive life time interference.

Later in [1], Sannidhi investigated reasons for large spread in breakthrough curves at different well locations of geothermal doublet. For this study, the same regional geological model of WNB, studied by C. Willems, was used. Two models were compared: coarse-scale and fine-scale that had the same N/G ratio. Despite the fine scale model having a better connectivity along sand bodies, the spread still persisted. Next the sand bodies were made completely homogenous, and there was still significant variation. Finally looking into cross sections of the model, along with the streamlines produced, it was obvious that the cause of significant difference in thermal breakthrough was due to the heterogeneity in the way channels were deposited. It was observed that simply by relocating the well to nearest grid block, cold water encountered an entirely different pathway, which consequently effects the way in which the cold front is thermally recharged along reservoir, having a completely different breakthrough time compared to that of its neighboring doublet location. Placing the pair of wells from west of the reference position to the north of reference, that was calculated to be around 117 meters where each cell is 90 m × 75 m, the lifetime of the doublet had increased from 34 years to 62 years.

1.3 Streamlines

An important step in understanding the flow and its consequences is the ability to visualize it, either experimentally, analytically or conceptually. Three of the most common lines in a flow which the particles of fluid can trace out as it proceeds are:

- **Streamlines** – Imaginary line drawn in the flow field such that the tangent drawn at any point on this line represents the direction of velocity vector of the fluid particles at that point. Only the direction of the fluid velocity is important: not its magnitude. There is no requirement that fluids be incompressible or the velocity be steady state. Although, the streamlines are based on instantaneous velocity, therefore if the velocity is time-varying, then streamlines are traced from a snapshot of the velocity at a time of interest. For unsteady-state problems, streamlines are a representation of the instantaneous velocity, not the physical trajectory of the particle. Streamlines can never intersect each other, i.e. there can be no flow normal to streamline. [22] [23]

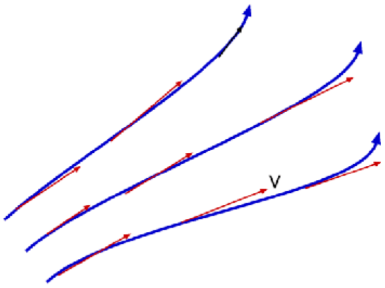


Figure 1.1 *– Streamlines - integrated curves locally tangential to velocity direction* [24]

- **Streamtubes** – tubular region of fluid that is surrounded by streamlines. Streamtubes are "impermeable" since the walls of the tube are made up of streamlines and there can be no flow normal to streamline. From a mass conservation view-point, in a steady, one-dimensional flow, mass-flow rate is constant along a streamtube [23].

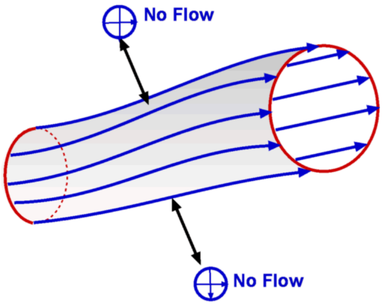


Figure 1.2 *– Streamtubes - region of fluid surrounded by streamlines* [24]

- **Pathlines** – line traced by a given particle as it flows from one point to another. It is the physical trajectory of a particle. Pathlines can be generated by injecting a dye into fluid and following its path by photography. They depend upon both the magnitude and direction of the velocity. For a steady velocity, the pathline and the streamline describe the same path. By construction, streamlines can never cross but pathlines may often cross [22] [24].

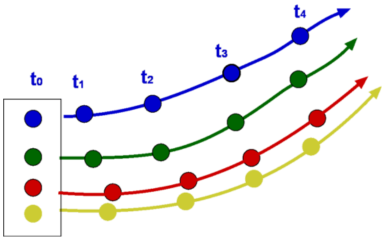


Figure 1.3 *– Pathline - physical trajectory of a particle* [24]

- **Streaklines** – focuses on fluid particles that have passed continuously through a particular spatial point in past. At some time, a line is drawn through the instantaneous position of all these particles. [24]

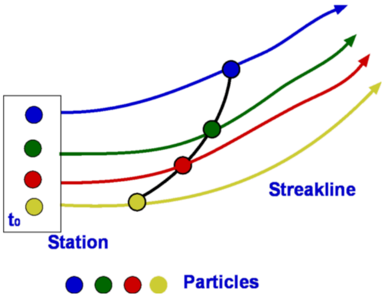


Figure 1.4 – Streaklines [24]

- **Timeline** – generated by drawing a line through adjacent particles in flow at any instant of time [24].

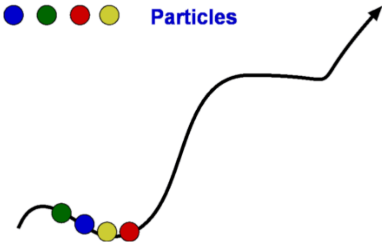


Figure 1.5 – Timeline - line through adjacent particles in flow [24]

For the purpose of this project, only streamlines will be used to visualize the flow and generate regions (or streamtubes) for proxy model. And the velocity used will be the total interstitial velocity, which is the total multiphase Darcy velocity divided by the porosity [22].

In the petroleum industry, the application of the streamline method began in the 1930s. Over several decades of development the streamline tracing method has made great progress. It has evolved from the streamtube method, which can only trace simple two-dimensional cross-section models, to streamline which can trace the real three-dimensional reservoir; from the study of single-phase one-dimensional flow to two-phase three-dimensional flow, and the analytical and numerical solution methods along streamline are developed [25].

Tracing streamlines from injectors to producers is based on the analytical description of a streamline path within a grid-block as outlined by Pollock. The underlying assumption is that the velocity field in each coordinate direction varies linearly and is independent of the velocities in the other directions. This method is attractive because it is analytical and consistent with the governing material balance equation [2].

1.4 Grid

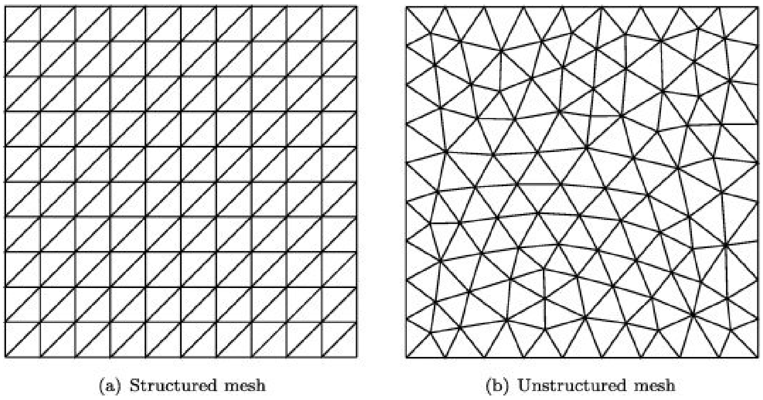
Gridding is the process of subdividing a region to be modeled into a set of small control volumes. With each control volume, one or more values of dependent flow variables are associated (e.g. velocity, pressure, temperature, etc.). These represent some sort of locally averaged values. Numerical solutions that represent approximations of the conservations laws of mass, momentum and energy are then used to compute these variables in each control volume [26].

Grid (also known as mesh) mainly comprises of two properties [27]:

- geometry (position of x's, y's and z's of the nodes)
- topology (how nodes are connected)

Grids can be primarily of two main types:

- structured
- unstructured


(a) Structured mesh

(b) Unstructured mesh

Figure 1.6 – Example of a structured triangular mesh and an unstructured triangular mesh in a unit box [28].

1.4.1 Structured Grid

Structured grids are ordered-type by indices such as (i, j, k) so that any value for these integers correspond to a spatial location and its topology is easily associated to next indices. But the geometry may not be constant, for instance the grid can be deformed from place to place as to match a specific boundary (as shown in Figure 1.7). Therefore, to be more precise, a grid is said to be structured if its topology is regular i.e. node connectivities have a fixed pattern [27]. The most common choices of volume element shapes for structured grid are quadrilateral in 2D and hexahedron in 3D [29].

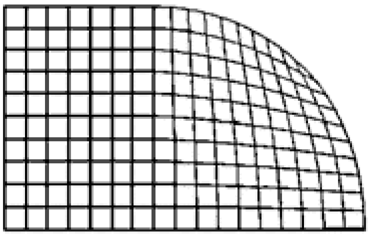


Figure 1.7 – Deformed mesh [26].

The advantages for structured grid is that it is simple to implement and nodal neighbor connectivity can easily be determined. The points of an elemental cell can be easily addressed by double indices (i, j) in 2D and triple indices (i, j, k) in 3D [30]. Unlike unstructured grid, structured grid does not require the storage of any connectivity table, as a result, the CFD computation time is reduced due to reduced cell count. Moreover, with a higher degree of control, the grid can be produced precisely as required. Lastly, the alignment in a structured grid is achieved almost implicitly because grid lines and flow follow the geometry contours, therefore leading to more accurate results and a better convergence in CFD solvers [31].

Despite having several advantages, structured grids have some drawbacks. Structured grid cannot be used for highly complex geometry (i.e. more complex than flow over cylinder). In order to achieve accuracy, it might be required to have concentration of point in one region, consequently unnecessary small spacing in other parts of solution domain is produced, which is waste of resources [29]. For more complex geometries, there is an increase in grid non-orthogonality or skewness. This distortion of elements away from purely rectangular shape can have several consequences. Firstly, distortion may reduce accuracy since numerical approximations are no longer centered about the centroid of the volume element. Although this drawback can be offset by an increase in local grid resolution afforded by distortion. Another consequence of distortion is that numerical approximations become increasingly complex; not only must the forces and fluxes be converted to normal and tangential components at element faces, but it is generally necessary to include data from all surrounding elements as well [26].

1.4.2 Unstructured grid

Unstructured grids have no ordering and the volume elements can be joined in any manner (as illustrated in Figure 1.6b). A node is associated to a single integer index and topology is known by associating connections of node with neighboring nodes therefore special lists comprising of the indices of neighboring cells have to be stored for each cell. The most common choices of volume element shapes for unstructured grid involve triangles in 2D and tetrahedron in 3D [29].

Unstructured grids are best adapted for complex geometry discretization particularly for domains having high curvature boundaries [30]. It is the most flexible type of grid that can be used with any discretization scheme (e.g. Finite Difference, Finite Volume and Finite Element) and can fit any type of boundary [29]. It also allows the user to change the resolution over domain, so the model can be applied over a large domain and the resolution can be refined where fine scales are important, without increasing too much of the total computation time.

Like structured grid, unstructured grids have drawbacks too. Since special lists comprising of the indices of neighboring cells have to be stored for each cell, there is a cost of having to store this connectivity. In addition, there is a large computational overhead owing to large number of operations per node [29]. Triangular or tetrahedral cells, in contrast with quadrilateral and hexahedral cells, are usually ineffective to resolve wall boundary layers. This is due to the fact that unstructured grid ordinarily yields very long thin rectangular (or tetrahedral) cells adjacent to the wall boundaries, therefore creating major problems in approximation of diffusive fluxes [30]. Moreover, since the unstructured grid follows the mass distribution in a domain, this is great as long as hydrodynamics are concerned but it can be a problem if radiation based physics need to be implemented because light travels in straight lines which doesn't occur in unstructured grid. This will cause the radiation field to become more diffusive than it should [32]. Also, owing to the random way elements may be assembled to fill the computational space, unstructured mesh generation techniques are inherently more automatic than traditional structured grids. As a result of this automation, engineers have to sacrifice control over grid generation. Another problem is largely of designing these automatic algorithms that are robust and yield suitable element shapes for the flow solver [33].

1.4.3 How expensive is Unstructured Gridding?

[34] discusses the study that Shaw undertook to model inviscid flow over a wing-foreplane-fuselage and compared a grid that was solely block-structured with a hybrid grid with unstructured region containing the foreplane. Although this study cannot be viewed as totally rigorous, but the findings revealed that in order to achieve a similar accuracy in flow simulation, the surface mesh density of unstructured grid had to be nearly an order of magnitude denser than structured grid. Also, in the viscous region of the flow domain, the point density in an unstructured grid will be approximately ten times greater than in structured grid. It was also found that the rate at which time marching can be performed on unstructured grid is about one-half that of structured grid of same point density. Further, the amount of work done per time step will be greater and there will be increased processing time due to amount of indirect addressing that needs to be done. Lastly, another finding by Shaw indicates that the storage requirement per point for flow solution on unstructured grid is about four times that of structured grid.

1.4.4 Conclusion

If programming time is a significant constraint, structured meshing is the way to go [32]. Also from the findings of Shaw in [34], there are clear incentives to use structured grids whenever possible. For hybrid grids, the proposition is that extent of unstructured grid used should be minimal as possible.

However, the type of gridding used in DARTS, as well as the energy equation solver developed on MATLAB in chapter 1 use unstructured grid. Appendix B illustrates how to develop a connectivity table for such a grid. Moreover, a governing partial differential equation can be comprised of more than one component that illustrate physical phenomena (e.g. the energy equation can have conduction, convection, accumulation, energy sinks/sources, etc.). Appendix B, also describes how the Jacobian matrix and residual vector can be constructed by forming a loop over different components of the governing equation rather than having to calculate everything at once; therefore providing more control to the user.

1.5 Operator Based Linearization (OBL)

Numerical geothermal reservoir simulation requires discretization of governing PDEs, which describe mass and energy transport in a reservoir. An implicit nature of the time approximation increases the nonlinearity of the governing equations [6]. In order to increase the accuracy of a model, one can apply a finer computational grid in space or time, or use a more detailed description of the fluids such as in thermal-compositional model [35]. For models (e.g. geothermal) that use both gas and liquid phases, complex multiphase behavior and the assumption of thermodynamic equilibrium further amplify nonlinearity [6]. Due to these complexities of the underlying physical processes and considerable uncertainties in the geological structure of reservoirs, there is a persistent demand for accurate and efficient models. However, the improvement in the accuracy of models is usually counterbalanced by reduction in the turnaround time of simulation [35]. Large ensembles of reservoir models are run to perform sensitivity analysis and reduce uncertainties in parameters estimation. Therefore efficient reservoir simulation performance is essential for geothermal industry: any noticeable improvement could positively affect production workflow.

The solution process of PDEs requires the linearization of strongly nonlinear governing equation. The flexibility of linearization reflects the simplicity of changing the nonlinear formulation in an existing simulation framework. During linearization stage, all properties and their derivatives need to be determined with respect to nonlinear unknowns. The linearization stage defines the accuracy and robustness of nonlinear solution, dictates the data layout of a linear system and therefore has a great impact on the reservoir simulation performance. In reservoir simulation, Newton-Raphson‘s method has become a standard solution for solving the nonlinear system of equations by linearizing it. It requires the assembly of Jacobian and residual for fully couple system of equations and consumes a significant portion of simulation time, especially for complex physical processes [6].

$$J\,(\omega^{\nu})(\omega^{\nu+1} - \omega^{\nu}) \;+\; r\,(\omega^{\nu}) \;=\; 0. \tag{1.3}$$

Commonly, three method of linearization are used: (1) Numerical derivatives that are flexible to implement but are short of robustness and may cause stalled behavior in some cases, (2) Straight-forward analytical derivations, but are limited in flexibility, (3) Automatic Differentiation, acquire computational expenses thus decreasing performance [6].

Recently, a new linearization approach for non-isothermal flow and transport was proposed and tested, named “Operator-Based Linearization”. The key idea is to transform governing equations that have been discretized in space and time, into operator form where each term is specified as product of two operators. The first operator consists of physical properties of rock and fluids – that depend on current physical state of grid block (e.g. density or viscosity) and are fully defined by values of nonlinear unknowns. The value of these operators can be determined for parametrized set of points, which are known as base nodes or nodal values. The second operator includes all terms that were not included in the first operator, and depends on both the state and spatial position of a control volume (e.g. permeability) [6].

After applying the OBL, the discretized mass conservation and energy conservation equations read as follows:

$$a(\xi,\omega)\big(\alpha_c(\omega) - \alpha_c(\omega_n)\big) + \sum_l b(\xi,\omega)\beta_c(\omega) + \theta_c(\xi,\omega,\upsilon) = 0, \qquad c = 1,\dots,n_c, \tag{1.4}$$

$$a_e(\xi,\omega)\big(\alpha_e(\omega) - \alpha_e(\omega_n)\big) + \sum_l b_e(\xi,\omega)\beta_e(\omega) + \sum_l c_e(\xi,\omega)\gamma_e(\omega) + \theta_e(\xi,\omega,\upsilon) = 0. \tag{1.5}$$

α and β here are state-dependent operators hence are only functions of pressure and overall composition in the grid block. At the pre-processing stage, or adaptively, the operators α_c, β_c, α_e, β_e and γ_e can be evaluated at every point in the discrete parameter space and store them in (n_c+2)-dimensional tables A_e and Γ_e and (n_c+1)-dimensional tables A_c, B_c, B_e. As a result, the expensive phase behavior calculations are done once and limited to the pre-processing stage. During the course of simulation, both the values and the partial derivatives of first type of operators (i.e. state-dependent) are interpolated using multi-linear interpolation in parameter space.

This makes the simulation computationally efficient with marginal loss in accuracy and also provides a continuous description based on the interpolation operator, whose accuracy is controlled by the resolution of discretization in parameter space. The larger the number of base-node points, the higher is the accuracy of the interpolation of the operators and lower is the error in solution. The second type of operator is processed based on conventional approach [35].

To improve the performance of OBL approach for a case with large number of species, an adaptive extension has been proposed by Khait and Voskov [35]. This is a minor extension of OBL in which the tables are generated during the course of the simulation. The grids are uniform but the values of the operators at the nodes are not pre-processed but are calculated during the course of simulation as they are required. Adaptive OBL is useful for simulating a multicomponent system where only a few values of the compositional space define the complete transport. Therefore, the phase behavior is resolved only at those nodes which are used for interpolation sparing the calculations at the unphysical nodal values. The following table summarizes the efficiency of the adaptive OBL method with changing resolution of the basal nodes and comparing it with the conventional linearization approach.

Resolution	**Newton Iter.**	E_{pres}	E_{Temp}	E_{comp}	**Linearization cost per Newton Iter.**
[nodes]	[#]	[%]	[%]	[%]	[-]
Std.	7616	Reference			1
64	2715	0.023	0.092	0.711	0.027
32	2629	0.071	0.356	2.257	0.027
16	2489	0.096	0.496	3.443	0.026
8	2118	0.070	0.615	3.748	0.026
4	2113	0.088	1.104	3.990	0.027
2	1901	0.108	4.279	3.672	0.026

***Table 1.1** – Efficiency of adaptive OBL method with changing resolution of basal nodes and comparison with conventional linearization approach* [6]

Application of this approach was investigated on geothermal processes, particularly low-enthalpy and high-enthalpy geothermal doublet models with hydrocarbon co-production. Performance and robustness of this new method were tested against conventional approach; it showed significant improvement on simulation performance, while errors that were introduced by coarsening in physics remained controlled [35].

Primary benefits of OBL are simplicity of application, while simultaneously improving nonlinear solver performance considerably due to simplification of residual and Jacobian assembly. Moreover, since the operators are piece-wise multi-linear functions, OBL avoids stalled behavior and builds a Jacobian with a high level of accuracy. The cost of OBL does not grow significantly with an increase in resolution, while using a coarser representation further reduces number of nonlinear iterations. The adaptive parametrization in physical space reduces memory utilization while increases efficiency of interpolation.

1.6 Thesis Outline

- **Chapter 1**, gives a general introduction about the geothermal energy and aspects of the geothermal reservoir simulation. It also introduces the mass and energy conservation equations, unstructured gridding and the Operator Based Linearization technique based on which the DARTS is built upon. The chapter also discusses the previous work that has been performed on geothermal simulation in order to predict temperature breakthrough.
- **Chapter 2**, a semi-analytical method is devised based on the reduction and discretization of the energy conservation equation. The results of this method are compared with the results obtained with DARTS.
- **Chapter 3**, introduces the streamlines and discusses the streamline tracing based on Pollock's method [2] in detail. It also provides results obtained from the streamline tracing algorithm developed in MATLAB.
- **Chapter 4**, gives the results obtained from the full-scale model and proxy model and compares them revealing how effectively the proxy model can be used in order to predict the breakthrough curves. It also provides a sensitivity analysis with respect to some of the parameters that can augment the error between the two models.
- **Chapter 5**, provides a summary of the work and the feasibility of the proxy model, based on the results. It also gives recommendations based on the difficulties encountered during the course of this project that can be incorporated into future work.

2

Semi-Analytical Heat Solution

2.1 Conservation of Mass

$$\frac{\partial}{\partial t}\left[\varphi\sum_{c/p=1}^{n_{c/p}} x_{c/p}\cdot\rho_p\cdot s_p\right]+\nabla\cdot\left[\varphi\sum_{p=1}^{n_p} x_{c/p}\cdot\rho_p\cdot u_p\right]+\dot{\rho}=0, \tag{2.1}$$

Equation (2.1) is the general form of conservation of mass [1]. Since, in our system there is only one component (e.g. water), and single phase (i.e. liquid), this equation can be further simplified to:

$$\frac{\partial}{\partial t}[\varphi\cdot\rho]+\nabla\cdot[\varphi\cdot\rho\cdot u]+\dot{\rho}=0. \tag{2.2}$$

2.2 Conservation of Momentum

Darcy's Law can be used as simplified form of conservation of momentum [1],

$$u=-\frac{K\cdot k_{rp}}{\mu_p}\left[\nabla P_p+\gamma_p\nabla D\right], \tag{2.3}$$

The bracketed term in Darcy's equation is the potential. The negative sign of Darcy's law arises due to potential gradient since the flow is from a higher potential to a lower potential. In horizontal x- and y-planes, the second term in the potential may be neglected as there exists no gradient due to height. Equations (2.2) and (2.3) can be coupled to solve for the velocity of particle at each location. The resulting equation is non-linear due to the fact that density and viscosity are pressure and temperature dependent.

2.3 Conduction

In a geothermal application, heat is transferred through the surrounding rocks and to the fluid passing through by the phenomena of conduction. This can be modelled as a tube (i.e. 1D stream passing through the rocks) and the heat energy being transferred from the perpendicular direction:

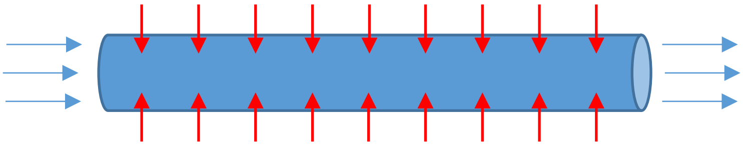


Figure 2.1 – *Modelling fluid passing (blue arrows) through rock with conduction from surrounding (red arrows)*

The energy transferred to the fluid can be modelled using Fourier's Law of conduction given by the expression below:

$$\dot{\varepsilon}_{conduction} = \nabla\cdot(\lambda\nabla T), \tag{2.4}$$

Further derivation of analytical solution for temperature distribution profile in a cylindrical solid can be seen in Appendix section A.3. The following final expression is obtained:

$$T = T_1 - \frac{\ln(r/r_1)}{\ln(r_2/r_1)}\cdot(T_1 - T_2). \tag{2.5}$$

2.4 Convection

The heat that is transferred from the rock to the fluid, is then carried by convection. The phenomena of convection is further composed of two concepts: advection (that is governed by fluid velocity) and diffusion (governed by gradient of concentration). In our system, the fluid velocity is significantly high therefore the diffusion term can be neglected.

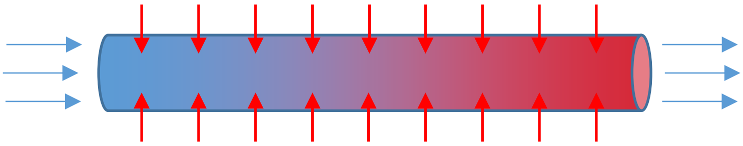


Figure 2.2 – *Modelling advection - fluid carrying thermal energy*

2.5 Conservation of Energy

The two modes of energy transfer: thermal 'conduction' and 'convection' can be combined and written in a conservation of energy equation given by the equation below. Since there is no radiation within a geothermal reservoir, it is not included in the conservation of energy equation otherwise it should have been considered as well.

$$\frac{\partial}{\partial t}\left[(1-\varphi)\cdot\rho_r\cdot u_{e,r}+\varphi\sum_{c/p=1}^{n_{c/p}}\rho_p\cdot u_{e,p}\cdot s_p\right]+\nabla\cdot\left[\sum_{c/p=1}^{n_{c/p}}h_p\cdot\rho_p\cdot u_p\right]+\nabla \\ \cdot(\lambda\nabla T)+\dot{\varepsilon}=0, \tag{2.6}$$

Equation (2.6) is the general form of conservation of energy. Since, in our system there is only one component (e.g. water), and single phase (i.e. liquid), and moreover the flow is in 1D (i.e. x-direction), this equation can be further simplified to

$$\frac{\partial}{\partial t}\left[\left((1-\varphi)\cdot\rho_r\cdot u_{e,r}\right)+\left(\varphi\cdot\rho_w\cdot u_{e,w}\right)\right]+\nabla\cdot[h_w\cdot\rho_w\cdot u_w]+\nabla\cdot\left(\lambda\frac{dT}{dr}\right)+\dot{\varepsilon}$$
$$=0. \tag{2.7}$$

Further derivation of energy conservation equation can be found in Appendix section A.1. The implicit solver that was formulated based on the energy equation was tested using the following parameters:

#	Parameter	Symbol	Value	Units
1	Number of grid cells	N	30 × 30	-
2	Thermal conductivity (matrix)	λ	2.0	W/(m·K)
3	Density (matrix)	ρ_{matrix}	2650	kg/m^3
4	Specific heat capacity (matrix)	c_{matrix}	800	J/(kg·K)
5	Time step	dt	10^5	sec
6	Simulation end time	t_{end}	10^9	sec
7	Initial reservoir temperature	T_{ini}	200	°C

***Table 2.1** – Parameters used for the Implicit Solver*

2.6 Comparison between Cases (MATLAB)

Next we study two types of simulation where full solution was decoupled in two set of solutions including frontal convection and conduction solved in 2D vs. conduction applied in 1D, orthogonal to the direction of convection. Simulation was run primarily for two cases:

Case 1: Convection propagation in the y-direction and conduction in both x- and y- directions.

Case 2: Convection propagation in the y-direction and conduction in x-direction.

For case 1, the terms for the conduction (in 2 dimensions) and convection in the general equation were being solved simultaneously. Whereas for case 2, the term for convection was being solved initially, and then another loop was run in which the conduction in x-direction was solved. This was achieved by generating a connection list that was only in the x-direction, whereas for conduction in the two dimensions was achieved by generating a connection list in both the direction.

As can be seen from both the plots in Figure 2.3, the reservoir is at an initial temperature of 200°C. The reservoir at the left and right boundaries is held at constant temperature of T = 200°C. Once cold water at t_{inj} = 35°C is injected into the reservoir (at length = 5 m), there is a bend in the temperature surface plot fabric. This bend illustrates a local temperature decrease caused due to the propagation (advection) of fluid in the y-direction.

The simulations were run using MATLAB 2017b on HP OMEN 15-ce041nd laptop consisting of Intel® Core™ i7-7700HQ CPU @ 2.80 GHz with 15.9 GB usable RAM. Below are the temperature profiles at t = 10^7 seconds for cases 1 and 2:

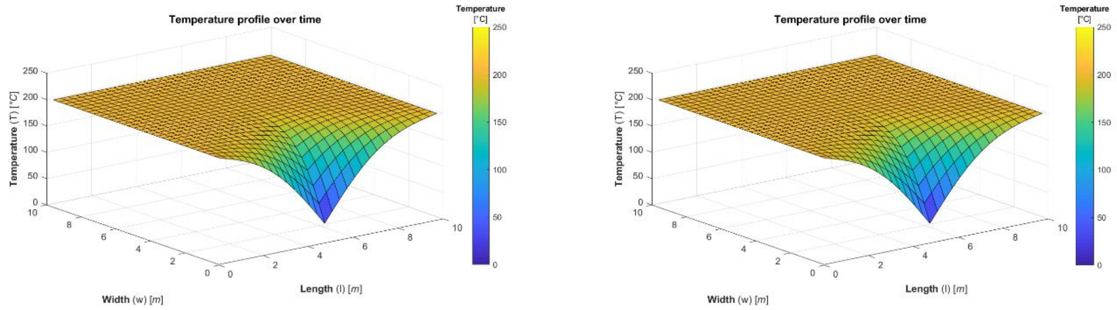
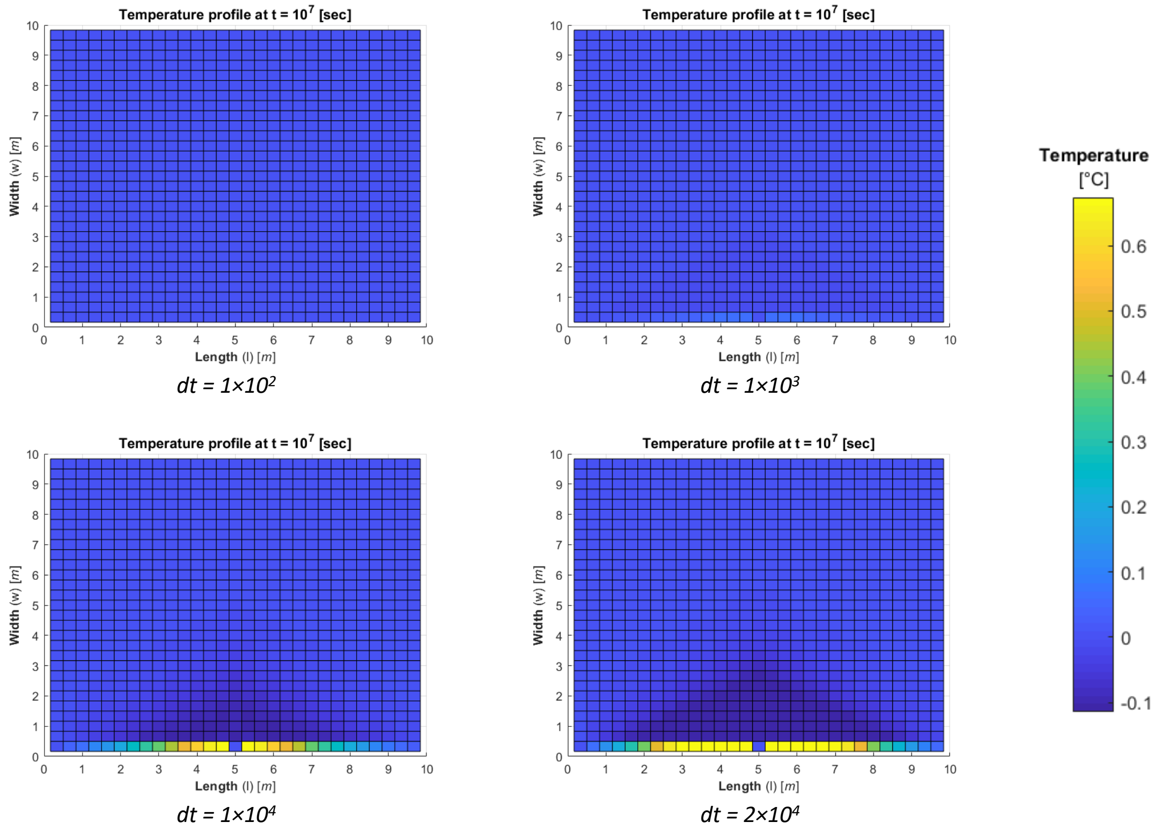


__Figure 2.3__ – Temperature profiles for Case 1 (left) and Case 2 (right) (at t = 10^7 sec)

Generally, the temperature profiles for the two cases look similar, until a closer look is taken using error analysis where the temperature profile for case 2 is subtracted from case 1. The error analysis shows that there is slight difference in temperature profiles for the two cases. Furthermore, below is the error analysis when the simulation is run at different time steps:

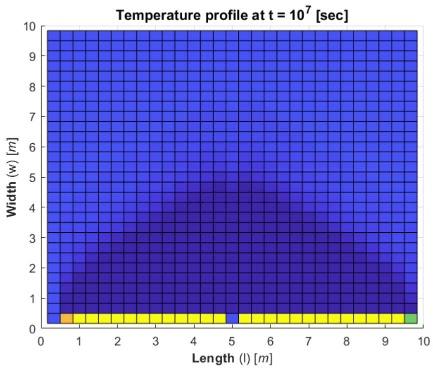


dt = 1×10^2

dt = 1×10^3

dt = 1×10^4

dt = 2×10^4



dt = 2×10⁵

***Figure 2.4** – Error Analysis for different simulation time-steps*

#	Time Step	Simulation Time	Temperature Difference		Iterations per Time Step		Total Iterations
			Minimum	Maximum	Minimum	Maximum	
	[sec]	[sec]	[°C]	[°C]	[#]	[#]	[#]
1	1×10^2	5886	-1.14×10^{-3}	6.92×10^{-3}	7	8	1.24×10^6
2	1×10^3	804	-1.14×10^{-2}	6.90×10^{-2}	8	10	1.37×10^5
3	1×10^4	105	-1.14×10^{-1}	6.74×10^{-1}	12	14	1.99×10^4
4	2×10^4	61	-2.81×10^{-1}	1.31×10^0	15	16	1.21×10^4
5	2×10^5	14	-6.40×10^0	1.07×10^1	48	54	4.20×10^3

***Table 2.2** – Simulation statistics for varying time-steps*

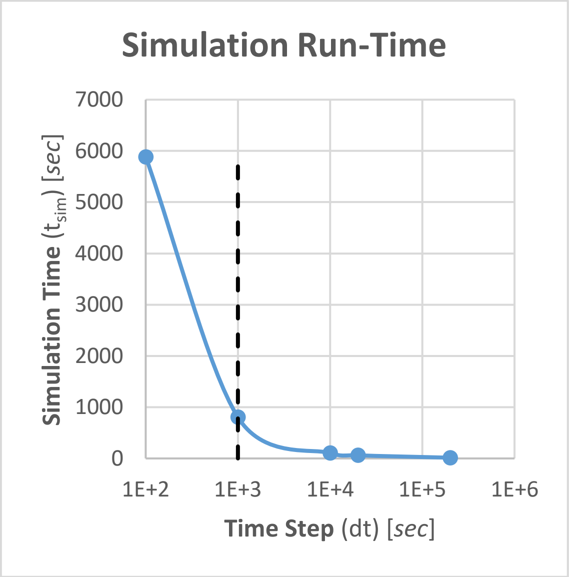


***Figure 2.5** – Variation of Simulation Run-time (t_{sim}) with Time Step (dt)*

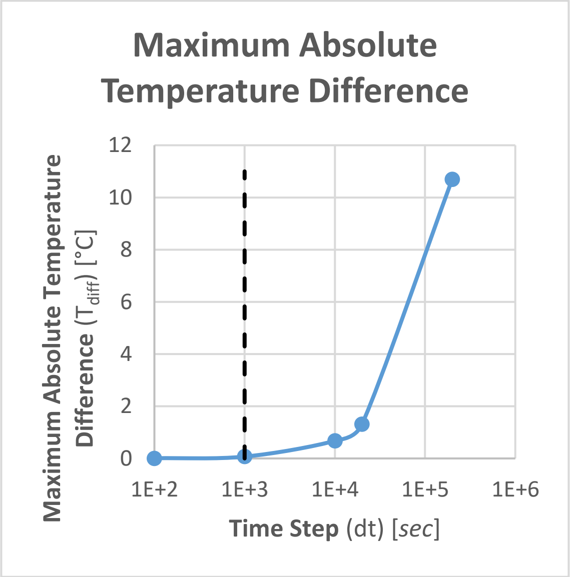


***Figure 2.6** – Variation of Maximum Absolute Temperature Difference (T_{diff}) with Time Step (dt)*

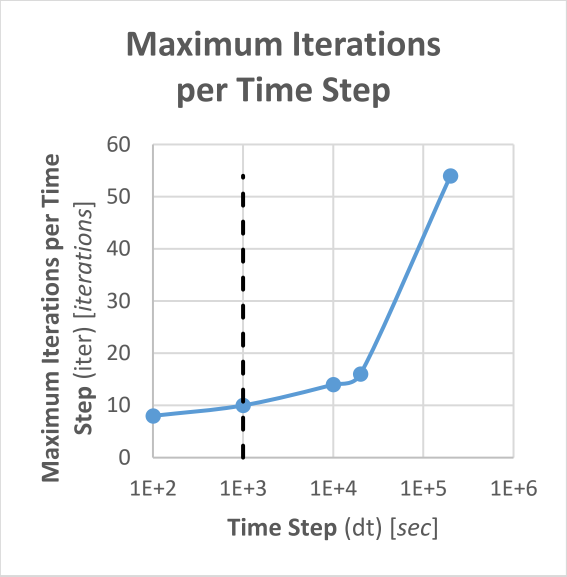


Figure 2.7 *– Variation of Maximum Iterations per Time Step (iter) with Time Step (dt)*

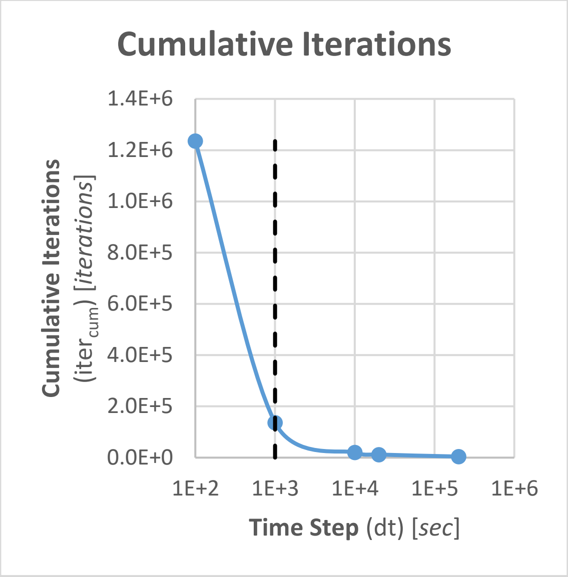


Figure 2.8 *– Variations of Cumulative Iteration (iter$_{cum}$) with Time Step (dt)*

As can be observed from the error maps Figure 2.4, the difference in temperature profiles for Case 1 and Case 2 increases for increasing time step. From Table 2.2 and charts (Figure 2.5 - Figure 2.8), the simulation statistics can be seen. With an increase in time step, the simulation run-time and cumulative Newtonian iterations required to converge, reduce drastically, initially (i.e. from 10^2 to 10^3 secs) and then the decline rate decreases. As for the number of Newtonian iterations required in each time-step and the maximum absolute temperature difference between the two cases increase gradually with increasing time-step and then the rate increase drastically.

The objective is to select a time-step that minimizes all the simulation statistical parameters. As seen from the plots the optimum time-step that decreases the simulation run-time and cumulative Newtonian iterations, while keeping the temperature difference to a minimum is 10^3 secs.

2.7 Comparison between MATLAB and DARTS Results

In order to validate the MATLAB implementation, the results were compared with DARTS. Most of the parameters in MATLAB and DARTS were kept constant to obtain an “apple-to-apple” comparison.

Parameters	**MATLAB**		**DARTS**	
	Value	**Units**	**Value**	**Units**
Grid	60 × 40	cells	60 x 40	cells
Thermal Conductivity	23.27	W/(m·K)	23.27 × 86.4	kJ/(day·m·K)
Specific Heat Capacity	950	J/(kg·K)	950 × 2.65	kJ/(m^3·K)
Injection Temperature	308.15	K	308.15	K
Time Step	1×10^6 ≈ 11.57	secs days	10	days
Simulation Run-Time	1×10^{10}	secs	1×10^{10} ≈ 1.157×10^5	secs days

Table 2.3 *– Common parameters between MATLAB Implicit Solver and DARTS*

Since the MATLAB code implementation was only based on the energy equation, and no flow solver therefore some additional parameters for flow equation need to be defined in DARTS:

Parameters		DARTS	
		Value	**Units**
Porosity	**Low**	0.008	-
	High	0.31851	-
Permeability	**Low**	0.008	m^2·m
	High	5000	m^2·m
Cell Volume		2000	m^3

Table 2.4 *– Parameters assumed for DARTS*

The full physics solution in DARTS was performed for incompressible assumptions, so the velocity in the simulation stays constant. To match the results, the velocity in MATLAB code was defined as u = 1.4 m/s.

Below are the temperature profile results obtained from simulations run on MATLAB and DARTS. The third figure illustrates the error analysis (i.e. the difference between the temperature profiles obtained from DARTS and MATLAB):

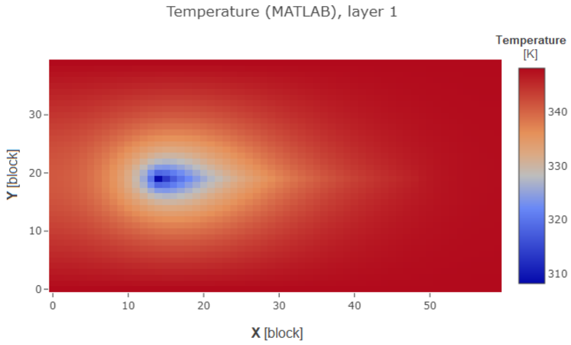


Figure 2.9 – Temperature profile obtained from MATLAB simulation

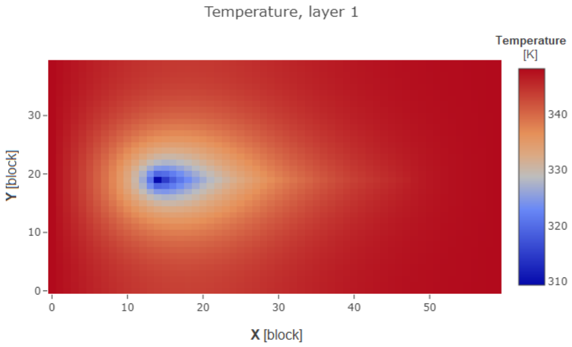


Figure 2.10 – Temperature profile obtained from DARTS simulation

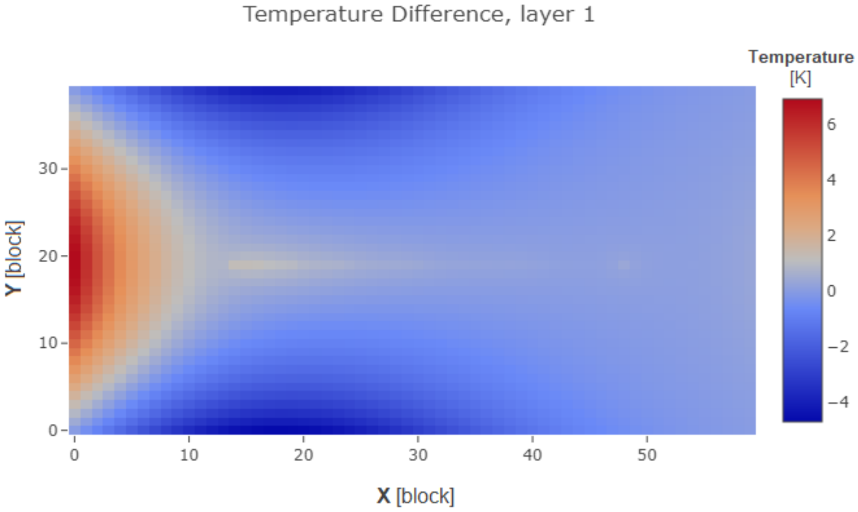


***Figure 2.11** – Error analysis between temperature profiles from DARTS and MATLAB*

2.8 Comparison between Cases (DARTS)

To model conductive heat distribution along one streamline, the following simulation study was performed:

Case 1: Advection in y-direction between the wells and conduction in the x- and y-directions.

Case 2: Advection in y-direction between the wells and conduction in x-direction only.

For case 1, the terms for the conduction (in 2 dimensions) and convection in the general equation were being solved simultaneously. Whereas for case 2, the term for convection was being solved initially, and then another loop was run in which the conduction in x-direction was applied. This was achieved by generating a connection list that was only in the x-direction, whereas for conduction in the two dimensions was achieved by generating a connection list in both direction.

Advection was imposed only between the wells; this was achieved by creating a highly permeable channel and therefore setting a high transmissibility for the cells between the wells. On the other hand, conduction in Case 1 was achieved by setting a non-zero value for heat equation transmissibility in both the directions. In Case 2, a zero value was set for heat equation transmissibility in y-direction.

The reservoir is at an initial temperature of 75°C. In addition, the reservoir at the left and right boundaries is held at constant temperature of T = 75°C. Once cold water at T_{inj} = 35°C is injected into the reservoir (at the injector well), the temperature along the streamline between the two wells starts gaining energy from the reservoir, while depleting the reservoir energy around.

Table 2.5 shows the parameters settings for simulation in DARTS. Figure 2.12 show the temperature profiles at t=100 days for Case 1 and Case 2, respectively. Figure 2.13 shows the difference between the temperature profiles for Case 1 and Case 2, for varying time-steps.

The simulations were run using MATLAB 2017b on HP OMEN 15-ce041nd laptop consisting of Intel® Core™ i7-7700HQ CPU @ 2.80 GHz with 15.9 GB usable RAM.

Parameters		**Value**	**Units**
Grid		60 x 40	cells
Porosity		0.1	-
Permeability	**Low**	0.008	m^2·m
	High	5000	m^2·m
Cell Volume	**Internal**	2×10^3	m^3
	Boundaries	8×10^9	m^3
Thermal Conductivity		23.27 × 86.4	kJ/(day·m·K)
Specific Heat Capacity		950 × 2.65	kJ/(m^3·K)
Reservoir Temperature		348.15	K
Injection Temperature		308.15	K
Injection Rate		200	m^3/day
Producer BHP		199.9	bar
Simulation Run-Time		100	days

Table 2.5 – Parameter settings for simulation in DARTS

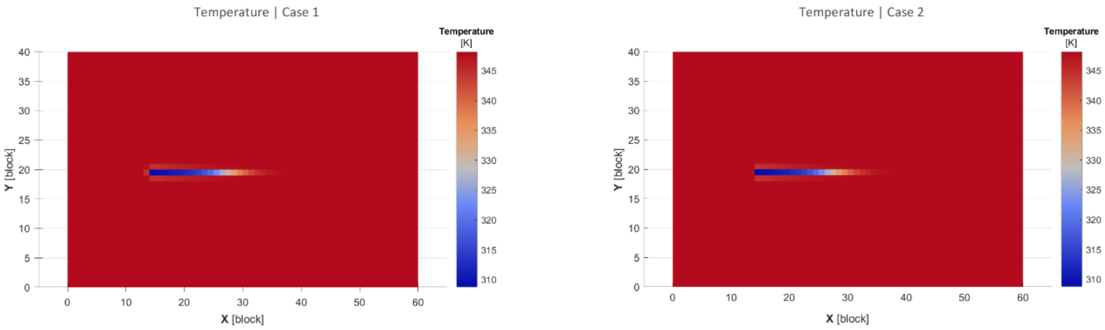
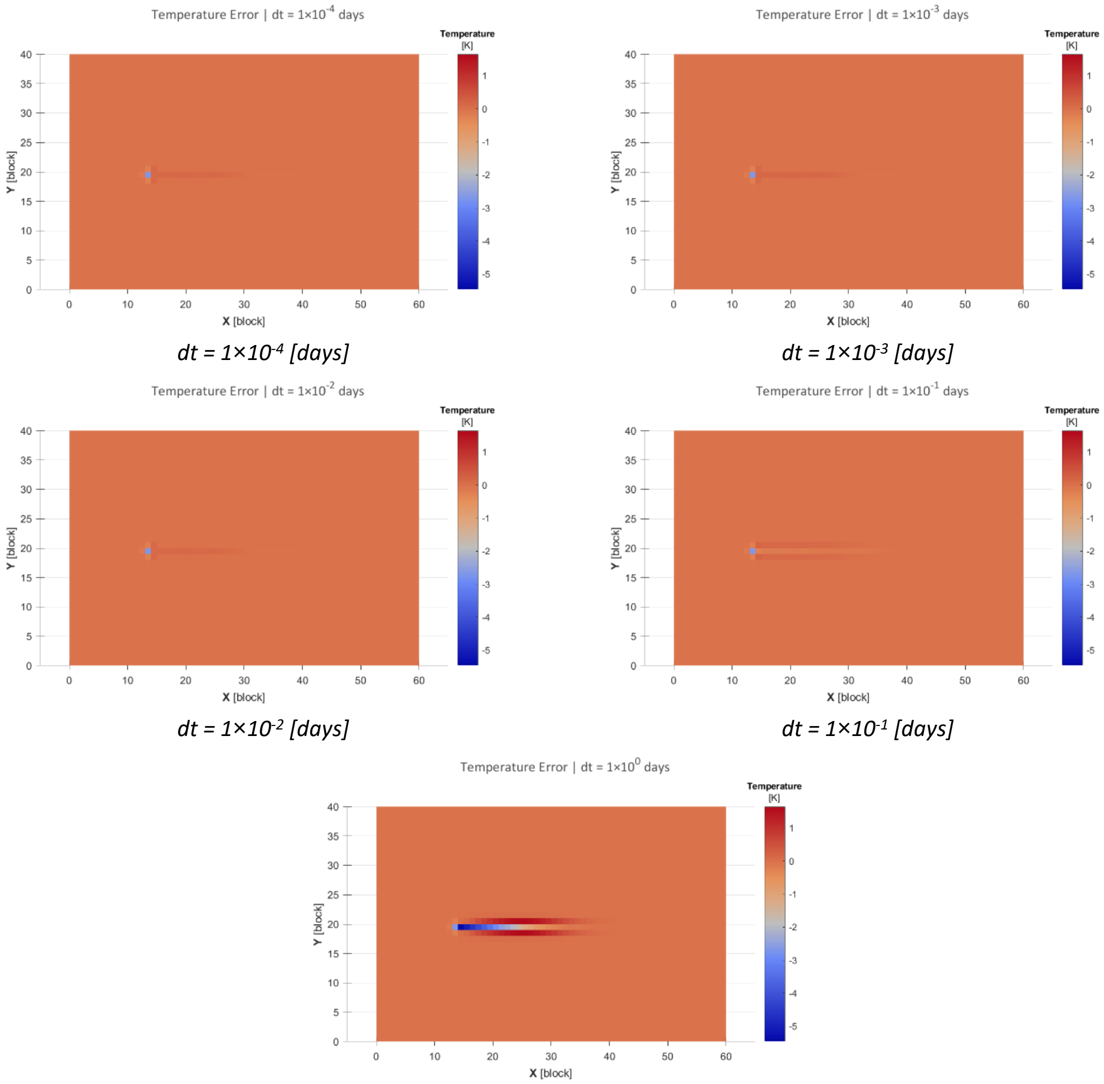


Figure 2.12 – Temperature profile for Case 1 (left) and Case 2 (right) at t = 100 days

Generally, the temperature profiles for the two cases look similar, until a closer look is taken using error analysis where the temperature profile for case 2 is subtracted from case 1. The error analysis shows that there is slight difference in temperature profiles for the two cases but below time-step dt = 1×10^{-1} [*days*], there is hardly any difference. Below is the error analysis when the simulation is run at different time steps:



dt = 1×10^{-4} [days]

dt = 1×10^{-3} [days]

dt = 1×10^{-2} [days]

dt = 1×10^{-1} [days]

dt = 1×10^{0} [days]

Figure 2.13 *– Error Analysis for different simulation time-steps*

The table below shows the summary for of statistics when the simulation is run at different time steps:

Time Step		**[days]**	**10^{-4}**	**10^{-3}**	**10^{-2}**	**10^{-1}**	**10^{0}**
Max. Absolute Temp Difference		[°K]	2.6889	2.6887	2.6889	2.6902	5.4499
Mean Temperature Difference		[°K]	2.71×10^{-3}	2.70×10^{-3}	2.71×10^{-3}	4.40×10^{-3}	3.77×10^{-2}
Std. Dev. Temperature Difference		[°K]	5.66×10^{-2}	5.70×10^{-2}	5.66×10^{-2}	5.85×10^{-2}	2.93×10^{-1}
Cumulative Newtonian Iterations	**Case 1**	[#]	1.00×10^{6}	1.00×10^{5}	1.01×10^{4}	1.56×10^{3}	2.17×10^{2}
	Case 2	[#]	1.99×10^{6}	9.95×10^{4}	1.01×10^{4}	1.50×10^{3}	2.21×10^{2}
Cumulative Linear Iterations	**Case 1**	[#]	1.00×10^{6}	1.00×10^{5}	1.05×10^{4}	3.58×10^{3}	6.61×10^{2}
	Case 2	[#]	1.99×10^{6}	2.03×10^{5}	2.25×10^{4}	4.74×10^{3}	8.66×10^{2}
Min Newtonian Iter. per Time Step	**Case 1**	[#]	1	1	1	0	2
	Case 2	[#]	0	0	0	0	2
Max Newtonian Iter. per Time Step	**Case 1**	[#]	3	4	4	5	5
	Case 2	[#]	3	4	4	5	5
Simulation Run-Time	**Case 1**	[sec]	6202	894	117	22	5
	Case 2	[sec]	12404	1787	234	43	9

Table 2.6 *– Simulation statistics for varying time-steps*

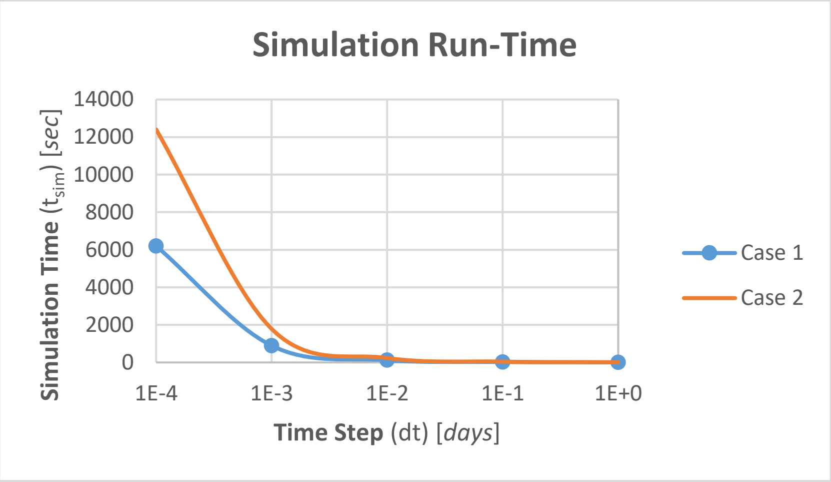


***Figure 2.14** – Variation of Simulation Run-time (t_{sim}) with Time Step (dt)*

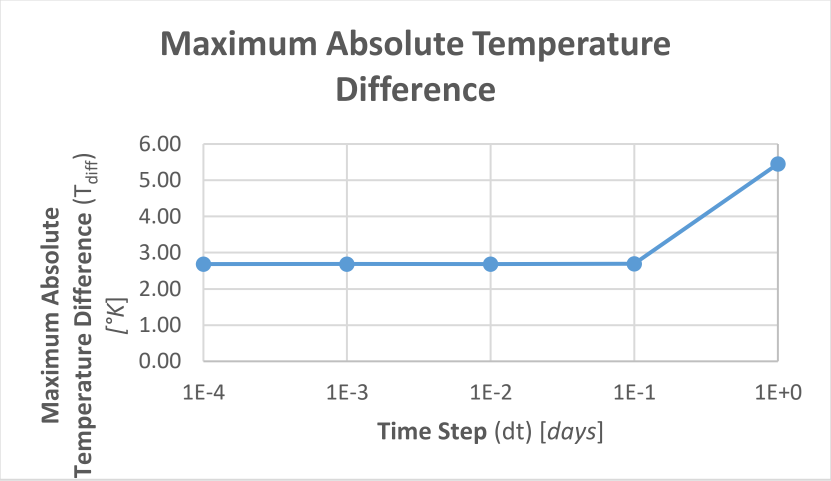


***Figure 2.15** – Variation of Maximum Absolute Temperature Difference (T_{diff}) with Time Step (dt)*

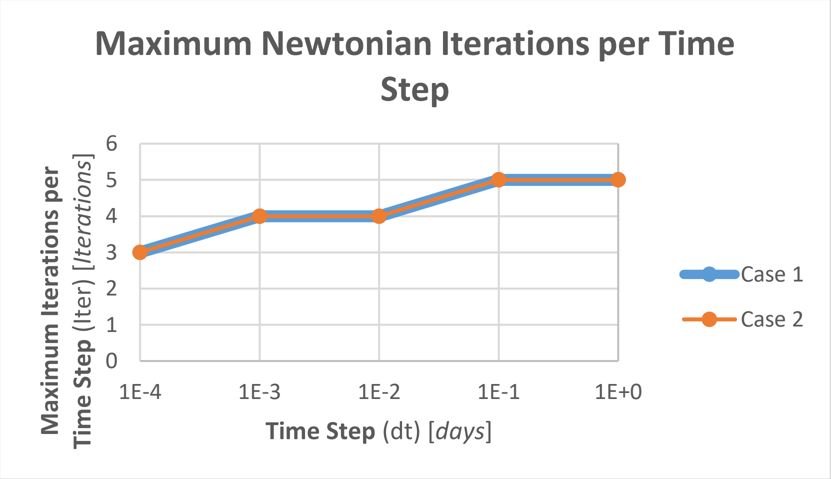


***Figure 2.16** – Variation of Maximum Iterations per Time Step (iter) with Time Step (dt)*

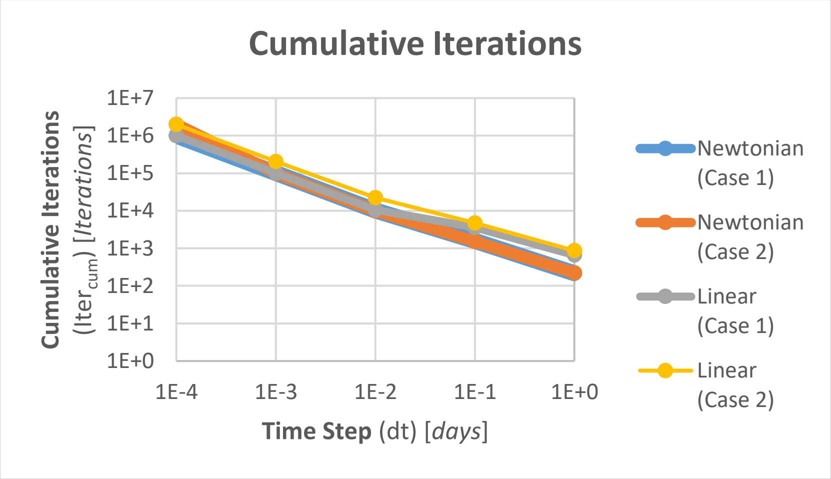


***Figure 2.17** – Variations of Cumulative Iteration ($iter_{cum}$) with Time Step (dt)*

As can be observed from the error analysis plots above (Figure 2.14 – Figure 2.17), simulation run-time and cumulative Newtonian iterations to converge reduce with an increase in time-step; both the performance parameters decrease drastically initially (i.e. 10^{-4} – 10^{-3} secs) but then the decline rate decreases. On the other hand, the number of Newtonian iterations required in each time-step and maximum absolute temperature difference (between the two cases) gradually increase with increasing time-step.

This study helps to validate that the convective and conductive solution of energy equation can be separated. However, this decoupling introduces a significant error which increases when the simulation time-step is increasing. These observations inspire a strategy for building of the proxy model described in the upcoming chapter.

3

Streamlines

This chapter will focus on tracing streamlines and utilizing them as a no-flow boundaries in order to define the regions (i.e. streamtubes) for the proxy model.

Once the short-term simulation for the full-scale model has run and the pseudo steady-state pressure distribution acquired, the velocities at the cell interfaces can be calculated using Darcy's Law. Next, the streamlines are traced using the algorithm for particle tracking developed by Pollock [2], as outlined in Appendix C. The following figure shows an example of the streamlines that were drawn for the results:

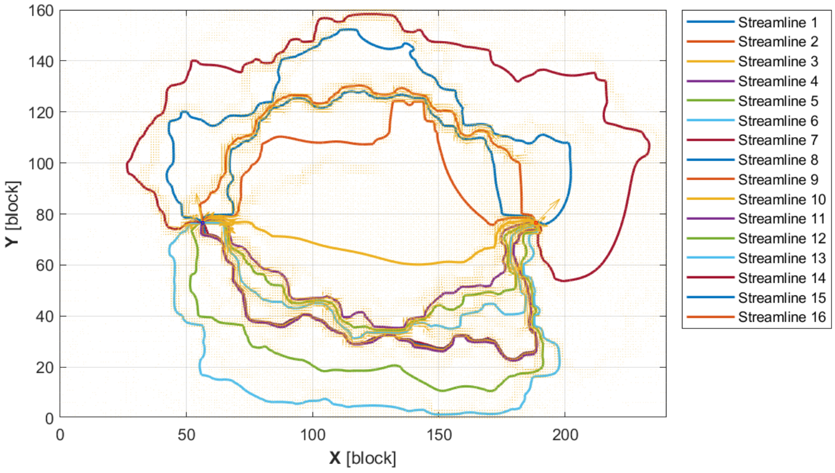


Figure 3.1 *– Streamlines traced for a Heterogeneous distribution*

In order to verify the accuracy of streamlines, they are traced over a permeability map. The regions with higher permeability (i.e. better connectivity) should have more number of streamlines passing through it, or near it. The following figure shows streamlines over permeability map:

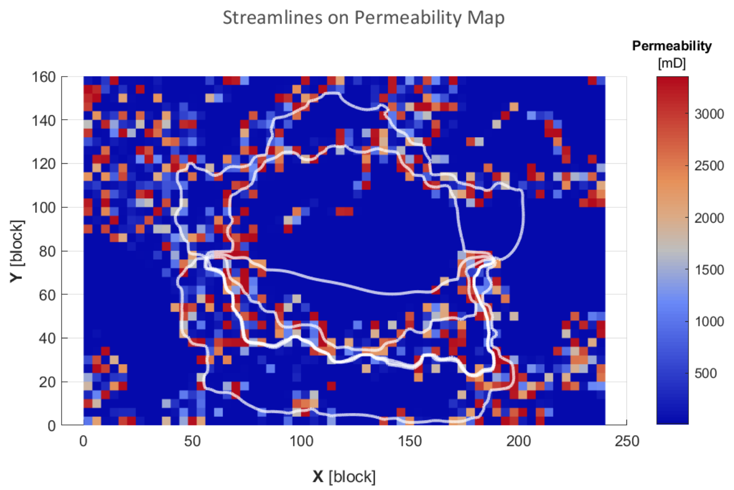


Figure 3.2 – Streamlines on Permeability Map

As can be seen from the figure above, for the heterogeneous case, the streamline geometry and density reflect the underlying permeability distribution. Streamlines tend to cluster in regions of high flow and are sparsely distributed in lower-permeability regions.

3.1 Boundary Conditions

Using Darcy's law, the velocities can only be obtained at the internal interface between two cells. Therefore if the grid has $m \times n$ dimensions, the velocity can only be obtained for $(m-1) \times n$ interfaces in the x-direction and $m \times (n-1)$ interfaces in the y-direction.

Whereas, the number of interfaces in the x-direction are $(m+1) \times n$ and $m \times (n+1)$ interfaces in the y-direction. Therefore the x-velocity at the left and right reservoir boundaries, and similarly, the y-velocity at the top and bottom reservoir boundaries were not calculated.

In this work, it is already assumed that the boundaries have no-flow condition (i.e. no flow occurs across the boundaries), therefore the missing velocities at the boundaries are taken to be 'zero'.

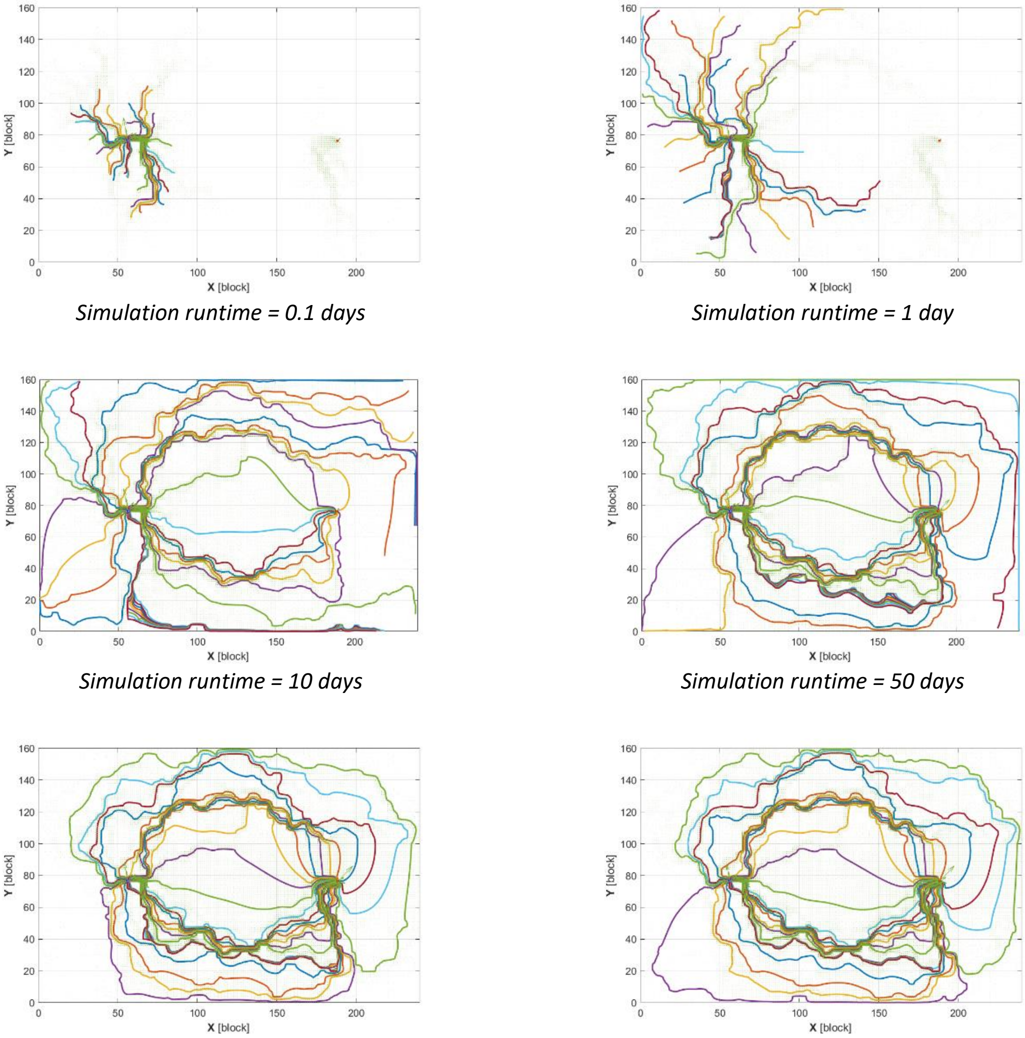
3.2 Fluctuating Streamlines

During the initial transient and semi-steady state, not only the overall reservoir pressure distribution is changing due to the compressibility but the relative pressure distribution (with respect to neighboring cells) is changing that causes the streamlines to alter. The velocities are obtained using Darcy's law that has the following expression:

$$u_x = \frac{Q_{v,x}}{A_x} = -\frac{k_x}{\mu \cdot \Delta x} \cdot (P_i - P_{i+1}). \tag{3.1}$$

As can be seen from the expression (3.1), the velocity depends on pressure difference between the cell in question and the neighboring cell.

Once the pseudo-steady state has been achieved, the overall pressure distribution may change, but the relative pressure distribution becomes more stable, therefore the streamlines do not change considerably. The following figures shows the variation and growth of the streamlines as simulation proceeds:



Simulation runtime = 0.1 days

Simulation runtime = 1 day

Simulation runtime = 10 days

Simulation runtime = 50 days

Simulation runtime = 1000 days

Simulation runtime = 20000 days

Table 3.1 *– Variation and growth of Streamlines as simulation progresses*

3.3 Time of Flight

Time of Flight is the time it takes for a non-reactive tracer particle that is introduced at the injection well or at an influx boundary of domain, to arrive at the production well. Alternately, it may also be defined as the distance along the streamline divided by particle velocity.

For this to be the actual transit time, the tracer particle must move at the interstitial velocity (v), not the Darcy velocity (u). Since the velocity can change along a streamline, time of flight can be represented as the integral of reciprocal of interstitial velocity along the distance of a streamline:

$$\tau = \int_{0} \frac{\phi}{|\vec{u}|} d\zeta, \tag{3.2}$$

where the ratio $|\vec{u}|/\phi$ is the interstitial velocity the particle moves at, and ζ is the spatial distance along the streamline.

The objective of this work is to be able to predict the breakthrough time of a geothermal producer well using a certain number of (fastest) regions. Therefore, the significance of the time of flight will become apparent in the upcoming chapter where the fastest regions are selected based on the average TOF of the streamlines enclosing a region (i.e. streamtube).

The time of flight can also be used as a spatial coordinate where the distance from inlet of a system is measured by this time, not by Euclidean distance. This concept of using TOF as spatial coordinate is fundamental to streamline simulation, but further details are out of the scope of this work.

3.4 Center-Lines

A center-line is simply an imaginary line traced between two streamlines. Initially, the objective of this thesis project was to trace center-lines that act as no-flow boundaries, and the regions were to be defined based on these center-lines, rather than streamlines. However, during course of the project, it was realized that the center-lines were difficult to trace and a proper procedure was needed. Moreover, since streamlines themselves are no-flow boundaries, they were used as an alternative to center-lines.

Despite not having used center-lines as no-flow boundaries to separate the regions, it is worthy to discuss the different methods that had been used to draw the center-lines. Appendix E elaborates on the different methods used to draw these center-lines and the short-comings of each method.

3.5 Active Cell, Volume and Transmissibility Multiplier Lists

Once the streamlines have been traced and the regions (i.e. streamtubes) have been defined as the area enclosed between two streamlines, it is important to identify the cells that are active in each region that will be used to initialize the models in the DARTS.

From the perspective of the grid with streamlines traced, there are two kinds of cells: (1) cells that completely lie in a region, (2) cells with streamlines passing through them and lie in more than one region. Even if a very small fraction of a cell lies a certain region, that cell needs to be marked active for that region. In order to initialize the region, the volume of the cell also needs to be expressed. The volume of the cell that lies in that region can be calculated using the method devised in Appendix D.

Moreover, if a pair of cells has a streamline crossing it, the transmissibility between them needs to be corrected in order to account for the flow between them, otherwise the flow for such cells is exaggerated, yielding incorrect results. The original transmissibility that is calculated by DARTS during the initializing of a model needs to be corrected by multiplying it with a fraction. Further discussion can be found in Appendix F.

These lists need to be exported into DARTS in order to initialize regions for the proxy model and yield correct results.

4

Results

This section contains the comparison between the results obtained from the full-scale model simulation and the proxy model simulation. Firstly, full-scale model simulation is performed using the parameters summarized in the table below.

Parameter	Value	Units
Reservoir Dimensions	1800 × 1200 × 105	m
Original Grid Size	60 × 40 × 42	cells
Downscaled Grid Size	240 × 160 × 42	cells
Layers under observation	only 2nd Layer (i.e. Layer ‘1’ in Python)	
Initial Reservoir Pressure	200	bar
Initial Reservoir Temperature	75 (348.15)	°C (K)
Injection Temperature	35 (308.15)	°C (K)
Injector Well Control Type	BHP	
Injector Well Control Value	210	bar
Producer Well Control Type	BHP	
Producer Well Control Value	190	bar
Time step (max)	10	days
Boundary Conditions	Closed boundary	
Heat Capacity	2200	kJ/day/(m·K)
Conductivity	100	kJ/(m^3·K)
Runtime	1000	days
Full-scale Model Well Index	10	m^3/day/bar
Proxy Model Well Index	10/8	m^3/day/bar
No. of Streamlines/Regions	8	

Table 4.1 *– Parameters for Simulation*

Once the full-scale model simulation is performed, the pressure distribution of the layer under observation is exported from the DARTS. Next this pressure distribution is imported into the Streamline tracer built in MATLAB. The ‘regions’ are identified as the area enclosed between two streamlines. Once the regions have been defined, the active cell list, volume for each cell, and the transmissibility between each pair of neighboring cell, for each region is generated.

Lists for each region are exported from MATLAB into DARTS. Then each region is simulated using the same parameters that were used to simulate the full-scale model (Table 4.1) prior to tracing of the streamlines. Next the results for each region are merged into one. If a cell lies in more than one region, the properties are weight-averaged based on their volume.

Finally, the results for both the properties, temperature and pressure, are compared for the full-scale model and proxy model and the error is determined by:

$$\epsilon_{i,j} = \xi_{i,j\,(full)} - \xi_{i,j\,(proxy)}, \qquad (4.1)$$

where the variable ξ means any property (e.g. Temperature or Pressure)

4.1 Homogenous Reservoir

Firstly, the case for a Homogenous reservoir is considered. The full-scale model simulation is run with most of the parameters being used from Table 4.1 – except for permeability and porosity. The following table summarizes these constant distribution of the reservoir:

Parameter	Value	Units
Permeability (x)	374.9389	mD
Permeability (y)	374.9389	mD
Permeability (z)	37.4939	mD
Permeability Anisotropy (k_{ver}/k_{hor})	0.1	-
Porosity	0.1578	-

***Table 4.2** – Additional Parameters for Homogenous Reservoir Simulation*

The full-scale model simulation is run. The following results illustrate the p-T distribution over the reservoir:

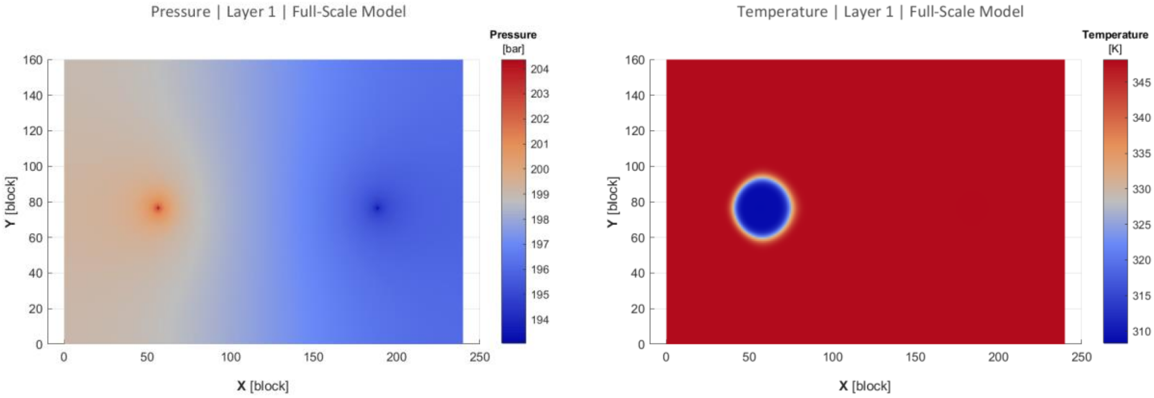


***Figure 4.1** – Pressure and Temperature Distribution of Homogenous Reservoir Full-Scale Model at t=1000 [days]*

Then, the streamlines for this pressure distribution are traced. The objectives of this thesis project includes generating streamlines for steady-state pressure distribution. The time-data for the full-scale model simulation reveals that the steady-state was achieved within few days, whereas the simulation was run for t = 200,000 days. This means, the pressure distribution at t = 200,000 days – based on which the streamlines are traced – was at a time much later and steady-state had already been achieved way before. The following figure shows the streamlines traced for this pressure distribution:

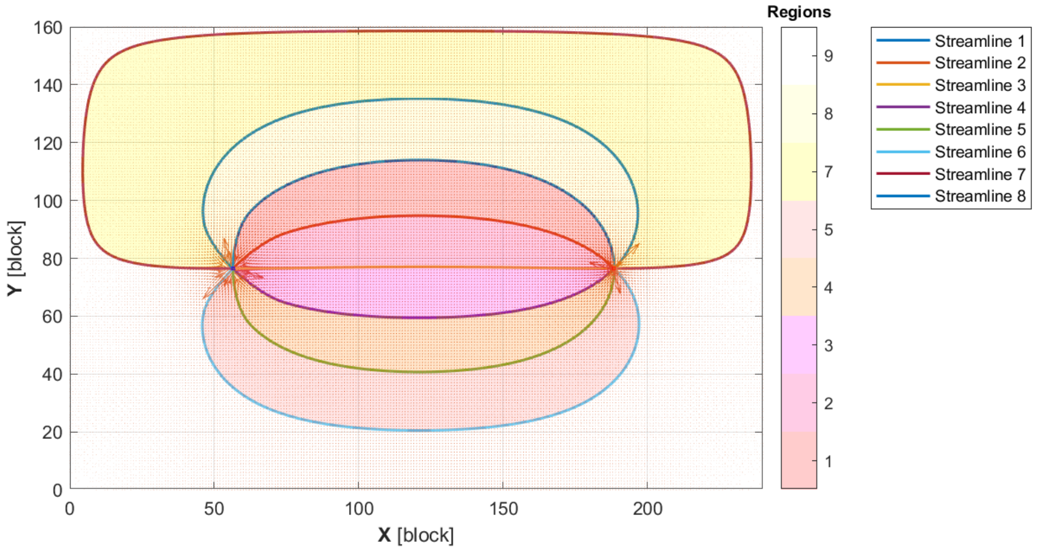


Figure 4.2 – Streamlines drawn for Homogenous Reservoir pressure distribution at t=1000 days

Once the streamlines have been traced, the regions are identified. In the figure above, each area with a different color is a separate region. For this example, the reservoir has been divided into 8 regions. The following table shows few of the entries from the active cell and cell volume list as an example of the data that is exported from the MATLAB Streamline tracer into the DARTS:

Cell Index	Active Cell	Volume
56696	1	17.578
56827	1	0.005
56828	1	17.749
56936	1	69.339
56937	1	72.366
56938	1	0.112
57066	1	0.487
57067	1	75.639
57068	1	68.886
57176	1	65.114
57177	1	140.625

Table 4.3 – Example of Active Cell / Cell Volume List

Next, the transmissibility multiplier list is also exported for each region in order to account for the flow between pair of cells that have streamlines crossing them.

Once, all the lists have been imported into DARTS, they are used to initialize a separate model for each of the regions that is defined by these streamlines. The simulation is re-run for each of the regions and the results are merged based on a weighted average of the cell volume. The figure below shows an example of the merged temperature distribution of the proxy model for homogenous reservoir.

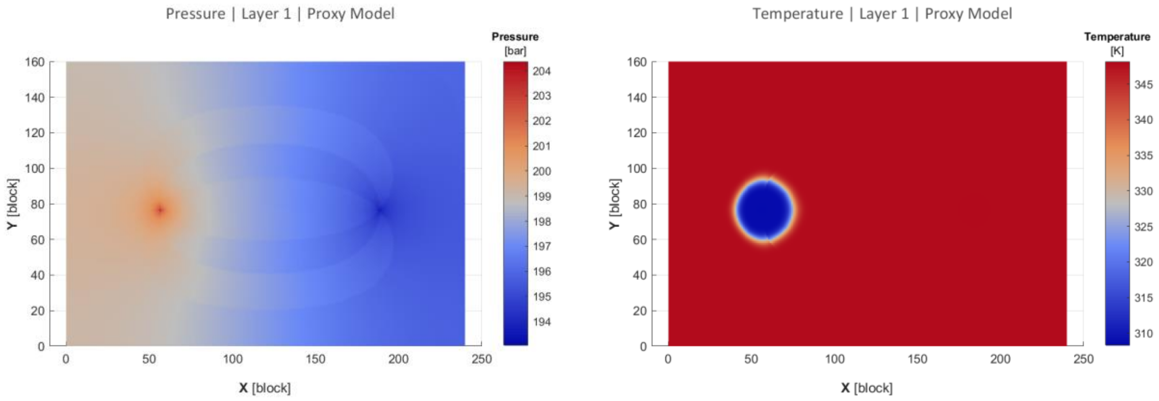


***Figure 4.3** – Pressure and Temperature Distribution of Homogenous Reservoir Proxy Model at t=1000 [days]*

At this stage, it is important to compare these p-T distributions from proxy model, with the original full-scale model simulation to determine the error between them. This can be done by generating an error map that shows the difference between the proxy model and full-scale model:

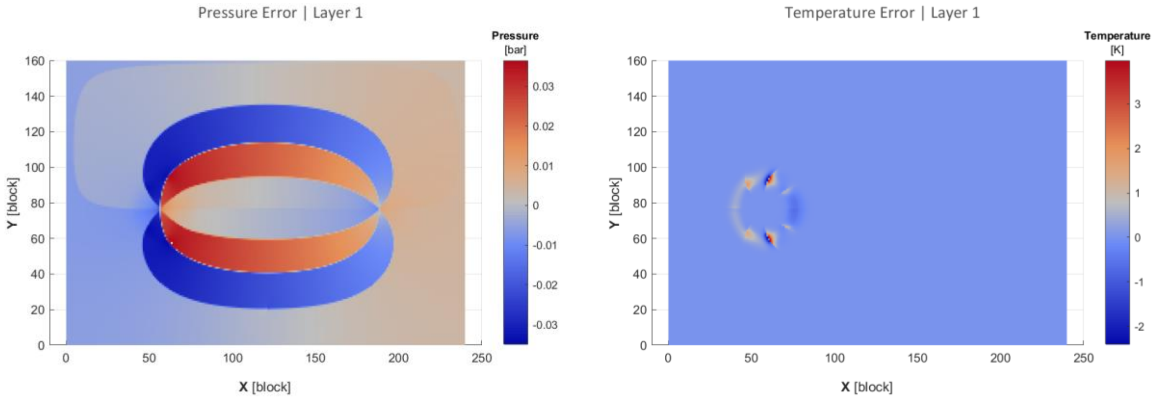


***Figure 4.4** – Pressure and Temperature Error Maps for Homogenous Reservoir at t=1000 [days]*

From the error maps, it can be determined that the error for the pressure distribution is quite small, while the error for the temperature distribution is larger but still within the acceptable tolerance. The following table represents the error statistics:

	Pressure	**Temperature**
	[bar]	[K]
Maximum Absolute Error:	3.65×10^{-2}	3.98×10^{0}
Average Error:	-1.29×10^{-3}	4.92×10^{-3}
Standard Deviation:	1.28×10^{-2}	1.08×10^{-1}

***Table 4.4** – Error Statistics for Homogenous Reservoir at t=1000 [days]*

The simulation results above are for a runtime of 1000 days (~ 2.74 years). But geothermal reservoirs do not have a life-time of mere 2.8 years, rather their economic lifetime is generally between 30 to 100 years. Therefore the full-scale model and proxy model simulations are re-run for a very large run-time, i.e. 20,000 days (~ 55 years). This will give a clearer picture how pressure and temperature error is distributed as time elapses. It will also determine how well the proxy model is able to predict the breakthrough temperature over the reservoir life-time. The following temperature and error maps are for time 20,000 days.

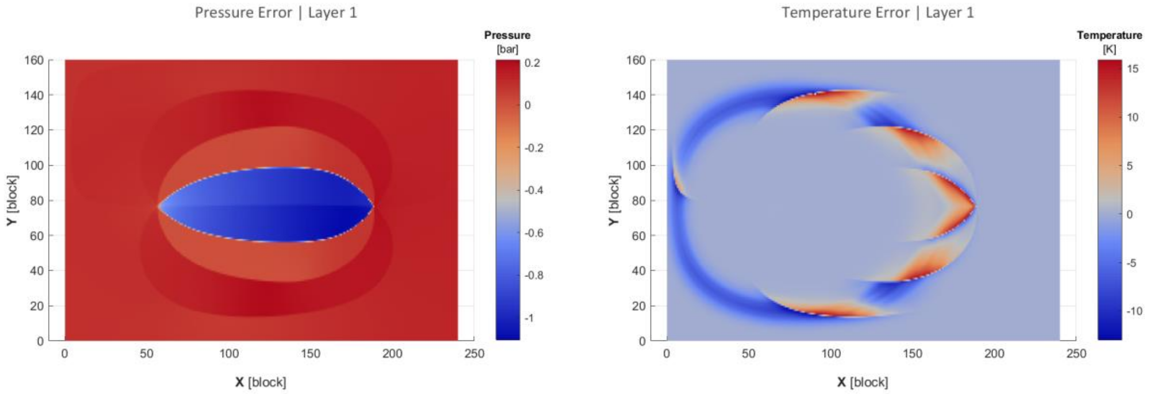


Figure 4.5 – Pressure and Temperature Error Maps for Homogenous Reservoir at t=20,000 [days]

From these error maps generated at t = 20,000 days, it can be determined that the error for the pressure distribution is quite small, while the error for temperature distribution is again larger but still within the acceptable tolerance. The increase in error with increase in simulation time is discussed in subsequent section. The following table gives the error statistics:

	Pressure	**Temperature**
	[bar]	[K]
Maximum Absolute Error:	1.11×10^{0}	1.59×10^{1}
Average Error:	5.80×10^{-4}	-1.44×10^{-1}
Standard Deviation:	3.33×10^{-1}	2.41×10^{0}

Table 4.5 – Error Statistics for Homogenous Reservoir at t=20,000 [days]

The following ‘Temperature Breakthrough’ curve shows how temperature at the producer well varies with time, for both full-scale model and proxy model:

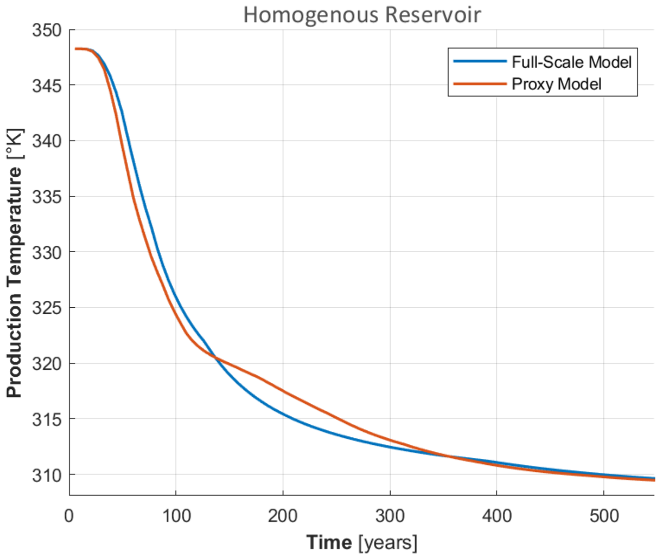
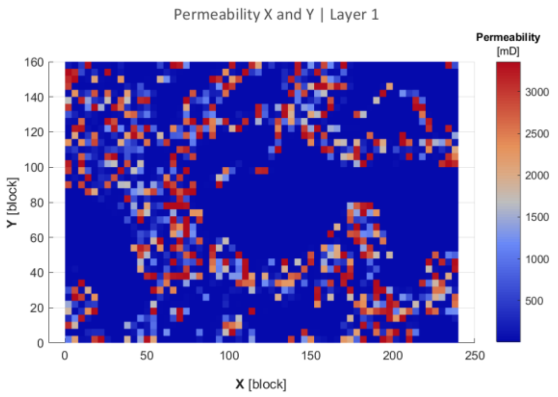


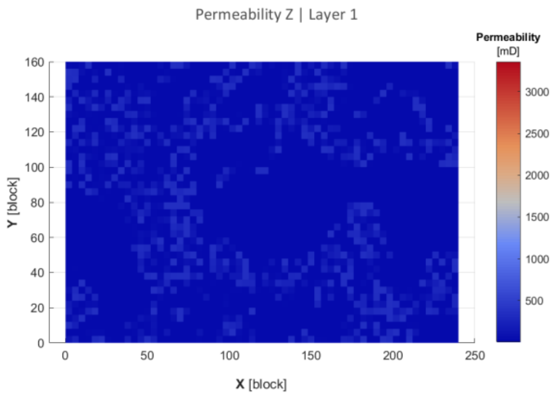
Figure 4.6 – Temperature Breakthrough curve for Homogenous Reservoir

The curve for the proxy model closely follows the full-scale model curve, therefore the breakthrough time can be predicted with acceptable approximation. In the graph above, the error increases between years 140 to 340, this may be due to the fact that the relative pressure distribution is varying due to which the streamlines change. If the streamlines are updated frequently during this interval, the error must reduce. As mentioned earlier, the steady-state distribution is achieved after few days, for which streamlines need to be drawn, but this does not mean that the streamlines do not change anymore. As time lapses, relative distribution of pressure due to compressibility, causes the streamlines to vary marginally.

4.2 Heterogeneous Reservoir

Next, the case for heterogeneous reservoir is considered. The simulations are run with parameters being used from Table 4.1 – except for permeability and porosity. The following maps illustrate the permeability and porosity distributions of the reservoir:





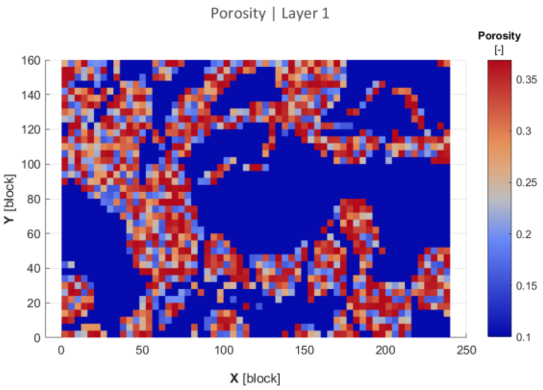


Figure 4.7 – Permeability Distributions in X, Y and Z directions and Porosity Distribution for Heterogenous Reservoir

The full-scale model simulation with heterogeneous distribution is run for 1000 days. The following results illustrate the p-T results over the reservoir:

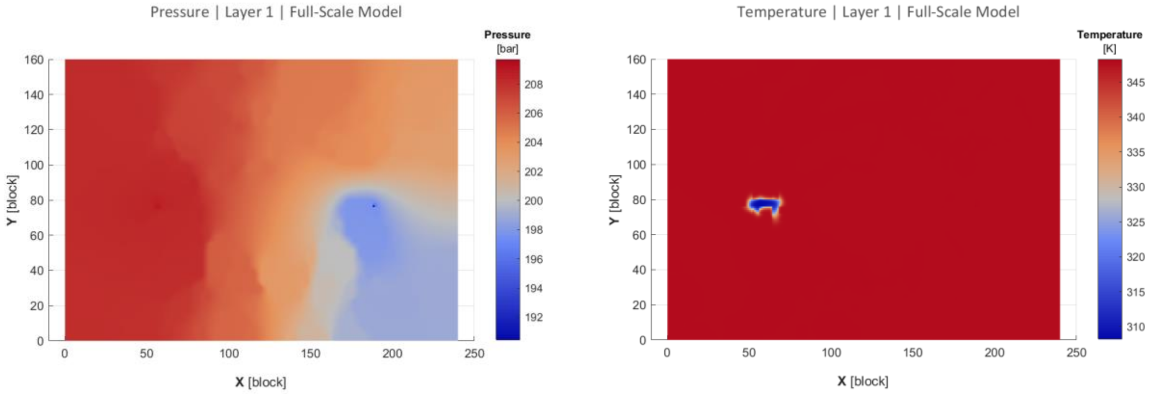


Figure 4.8 – Pressure and Temperature Distribution of Heterogeneous Reservoir Full-Scale Model at t=1000 [days]

Tracing the streamlines for this pressure distribution gives the following result:

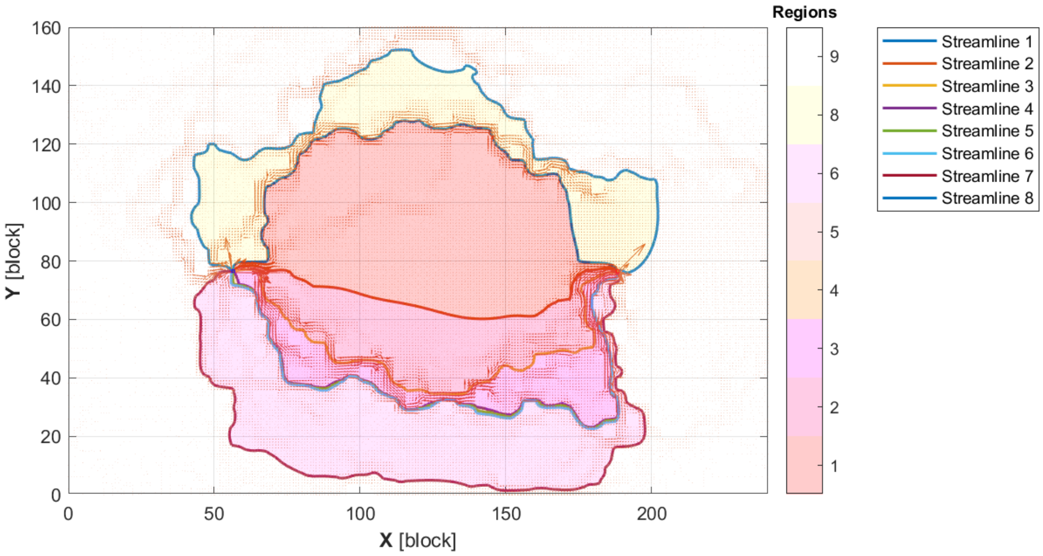


***Figure 4.9** – Streamlines traced for Heterogeneous Reservoir pressure distribution at t=1000 days*

Once the regions have been determined and the active cell and volume list and transmissibility multiplier list has been exported into DARTS, the proxy model is run with the following results:

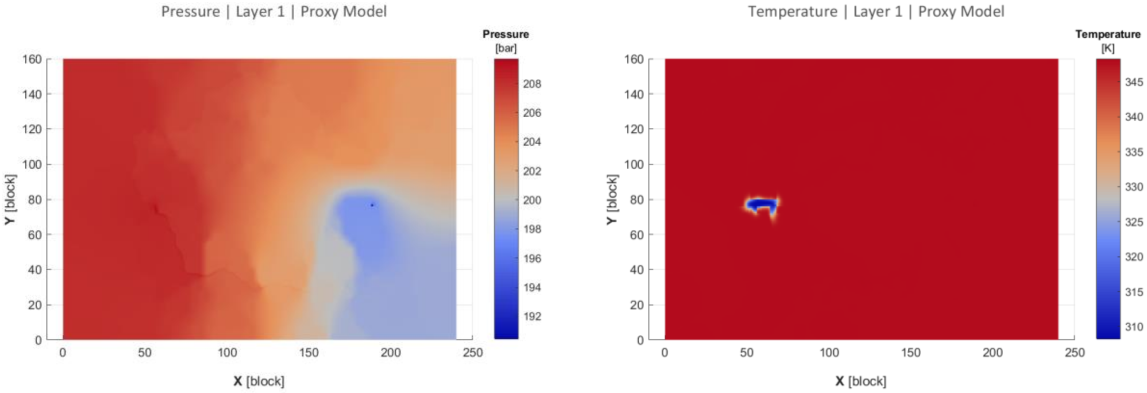


***Figure 4.10** – Pressure and Temperature Distribution of Heterogeneous Reservoir Proxy Model at t=1000 [days]*

Comparing the proxy model p-T distributions with the original full-scale model, the following error maps are generated:

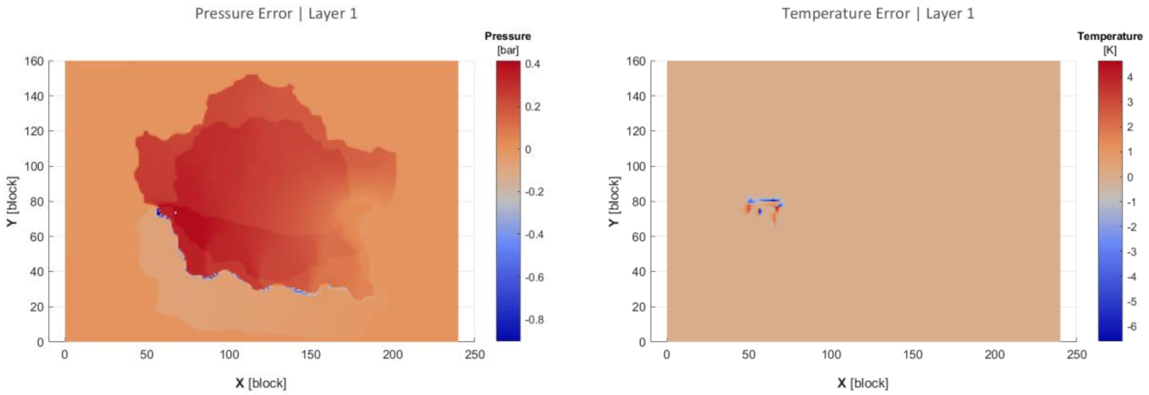


Figure 4.11 – *Pressure and Temperature Error Maps for Heterogeneous Reservoir at t=1000 [days]*

Re-running the simulations at 20,000 days and comparing the pressure and temperature distributions between full scale model and proxy model yield the following error maps:

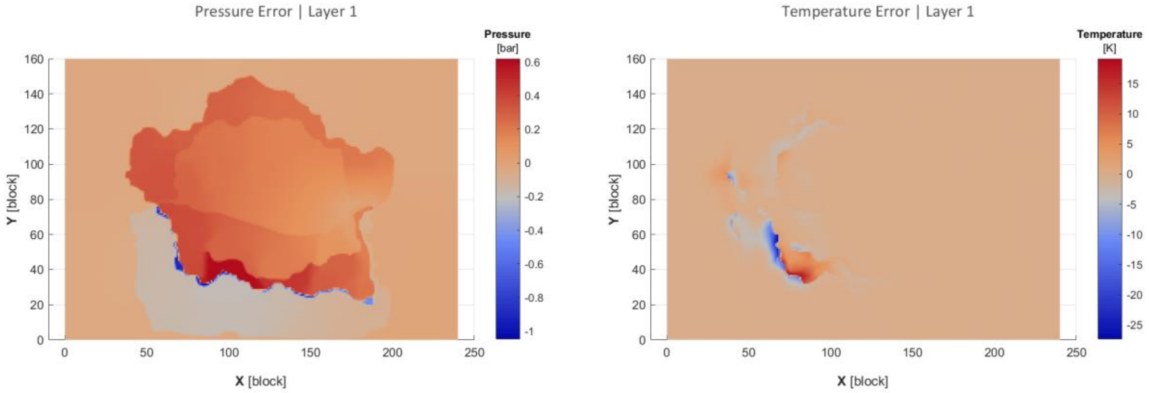


Figure 4.12 – *Pressure and Temperature Error Maps for Heterogeneous Reservoir at t=20,000 [days]*

Determining the error statistics for the difference between the full-scale and proxy model at 20,000 days gives the following:

	Pressure	**Temperature**
	[bar]	[K]
Maximum Absolute Error:	1.05×10^{0}	2.74×10^{1}
Average Error:	4.24×10^{-2}	-5.35×10^{-2}
Standard Deviation:	1.62×10^{-1}	1.31×10^{0}

Table 4.6 – *Error Statistics for Heterogeneous Reservoir at t=20,000 [days]*

With the increase in simulation time and heterogeneity in the reservoir, the error in pressure and temperature distributions is increasing. Despite this increasing error, it is within the acceptable tolerance, particularly the pressure error.

Next, drawing the temperature breakthrough curves for the full-scale model and the proxy model gives the following result:

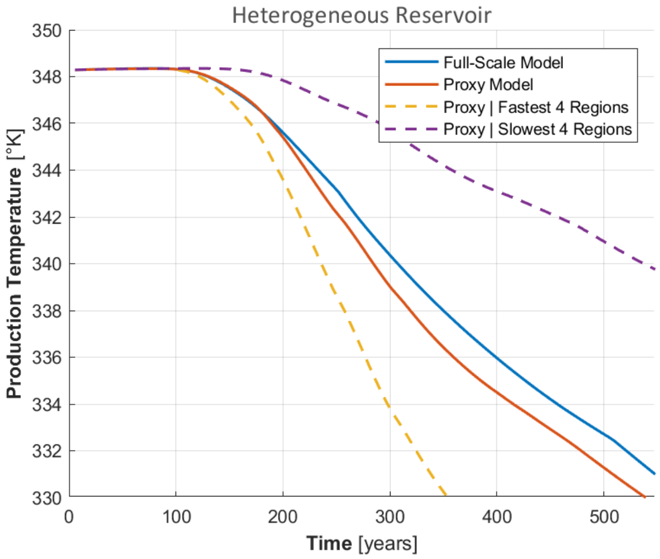


***Figure 4.13** – Temperature Breakthrough curve for Heterogeneous Reservoir*

The primary objective of this research project is to be able to accurately predict the breakthrough temperature with minimum number of regions. An increase in the number of regions to simulate the proxy model with, translates to increase in computer resources and time required for simulation. Logically, as the number of regions increase, the breakthrough curve approaches the proxy model curve, and the error decreases to that of the proxy model.

The approach is to use specific number of fastest regions. Since the regions are enclosed by streamlines, and each streamline possesses a time-of-flight. As defined earlier in section 3.3, time-of-flight is the time required for a particle to move from the injector well to the producer well following a specific path. The time-of-flight of a region can be approximated as the average of time-of-flight of streamlines it is enclosed by. This calculation is performed in MATLAB, and the following table gives an example of average time-of-flight for each region for the heterogeneous reservoir:

Region	**Time of Flight**
[-]	[×10^5 sec]
1	5.65
2	5.32
3	4.11
4	6.04
5	5.98
6	111.97
7	6.27
8	110.08

***Table 4.7** – Example of average time-of-flight for each region for Heterogeneous Reservoir*

During analysis, the regions are sorted based on the time-of-flight, with the fastest region being one with the shortest time-of-flight. The following curves prove the afore-mentioned hypothesis that increasing the number of regions to simulate the proxy model can increase the accuracy of the break-through prediction:

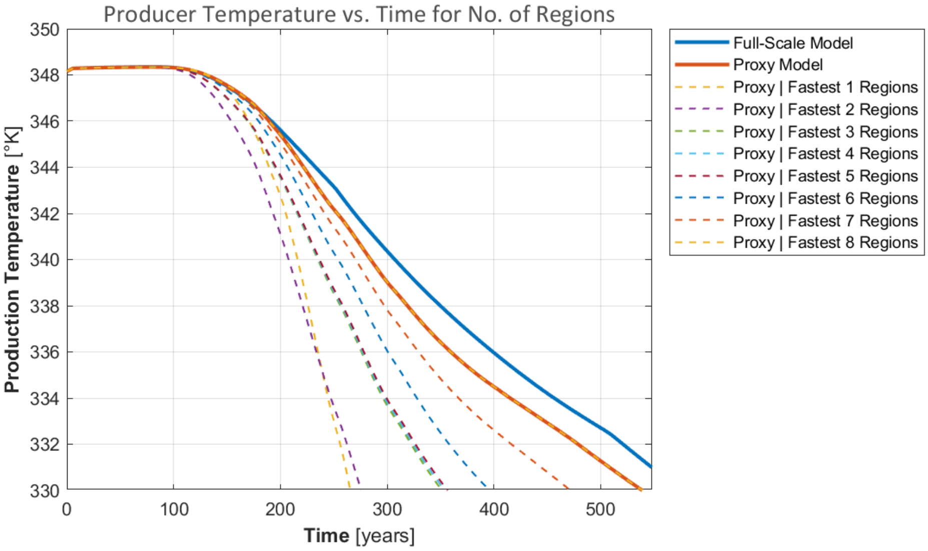


Figure 4.14 *– Increasing accuracy of Breakthrough curves with increase in number of regions to simulate with*

For the case of Heterogeneous reservoir layer divided into 8 regions, the following table shows the time required to simulate each region as compared to the time for full scale model:

Model	**Region**	**Time required to simulate** [sec]
Proxy Model	Region 1	14
	Region 2	7
	Region 3	5
	Region 4	1
	Region 5	1
	Region 6	29
	Region 7	11
	Region 8	26
Complete Proxy Model		94
Full-Scale Model		122

Table 4.8 *– Time required to simulate each region of Proxy Model compared to Full-scale Model*

It is interesting to note that the cumulative time required to simulate the proxy model is still less than the full-scale model. This is due to the fact that during the course of full-scale model simulation, a large number of cells need to be handled simultaneously, which increases the size of the Jacobian matrix and residual vector, thus requiring more computer resources and simultaneously increasing simulation time.

The following graph illustrates how increasing the number of regions to simulate the proxy model with, decreases the error but simultaneously increases the time required for simulation:

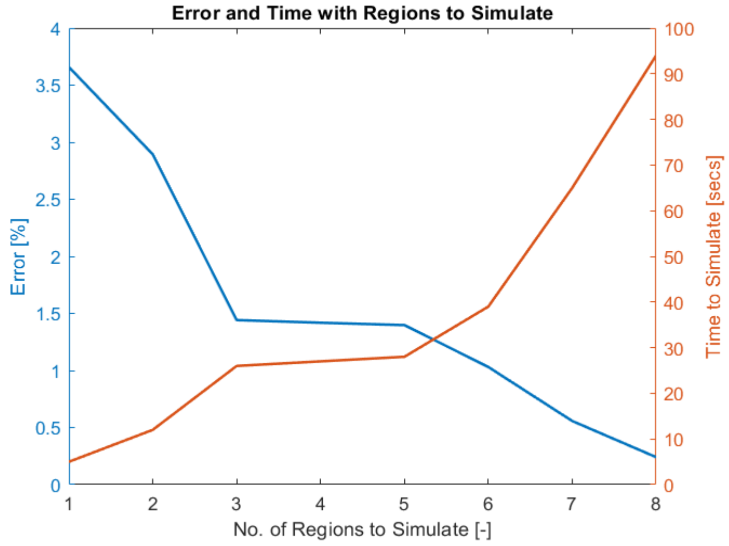


***Figure 4.15** – Variation of Error and Time with No. of Regions to simulate with*

4.3 Sensitivity Analysis of Temperature Breakthrough Curves

This section comprises of the sensitivity analysis of the breakthrough curves. Certain parameters, such as the well BHP's, simulation runtime and number of streamlines will be varied and their effect on the breakthrough curves will be observed. For the purpose of this section, all the parameters that are being used are from Table 4.1, except for the parameter that is being studied. In addition, heterogeneous reservoir permeability and porosity distributions (Figure 4.7) are being used.

4.3.1 Simulation Run-time

Varying the simulation run-time from 0 days to 200,000 days (i.e. 550 years), yields the following 'breakthrough temperature' error curve:

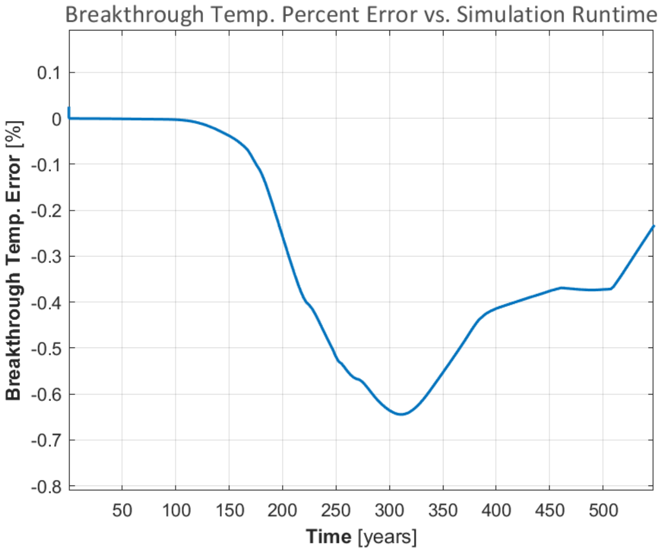


***Figure 4.16** – Variation of Producer Temperature Error with Simulation Run-time*

This curve shows the producer temperature error increases as the simulation run-time increases. As it has been established previously that dividing the reservoir into regions (i.e. proxy-model) already introduces some error in the results as compared to the full-scale model. Moreover as time lapses, the relative pressure distribution changes and so do the streamlines. Whereas the simulation performed for this analysis was performed at fixed streamlines (traced at t = 200,000 days), rather than updating the streamlines at each time-step. Therefore both these facts increase the error as time lapses. But the error seems to reduce after 1.15 × 10^5 days (~ 274 years). This may be due to the fact that streamlines' distribution do not change much. Moreover, since the streamlines were traced at 200,000 days, therefore as the runtime approaches that time, the streamlines distribution must be approaching increasingly similar to that distribution traced, hence reducing the error.

4.3.2 Number of Streamlines

Varying the number of streamlines from 2 to 8, yields the following 'breakthrough temperature' error curve:

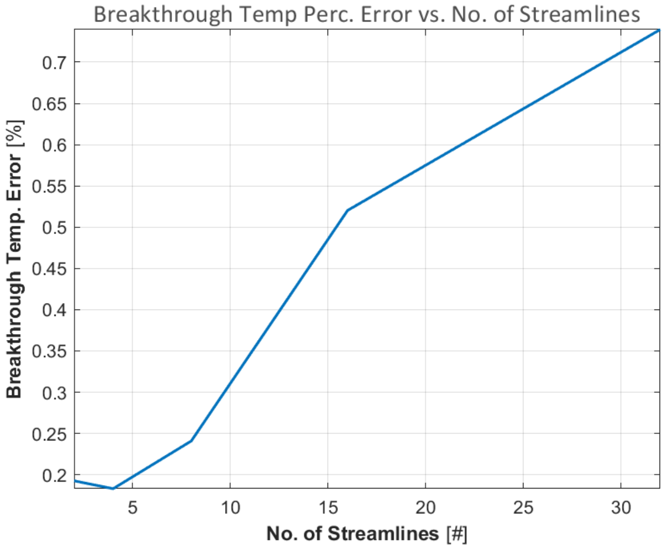


***Figure 4.17** – Variation of Breakthrough curves' Average Error with Number of Streamlines*

This graph shows that the error between full-scale model and proxy model curves increases as the number of streamlines the reservoir layer is divided into increases. Again, as it has been established previously that dividing the reservoir into regions (i.e. proxy-model) already introduces some error. Therefore, more the number of regions the reservoir layer is divided into, the further an increase in error.

4.3.3 Producer BHP

Varying the producer bottom-hole pressure from 10 bars to 190 bars, yields the following 'breakthrough temperature' error curve:

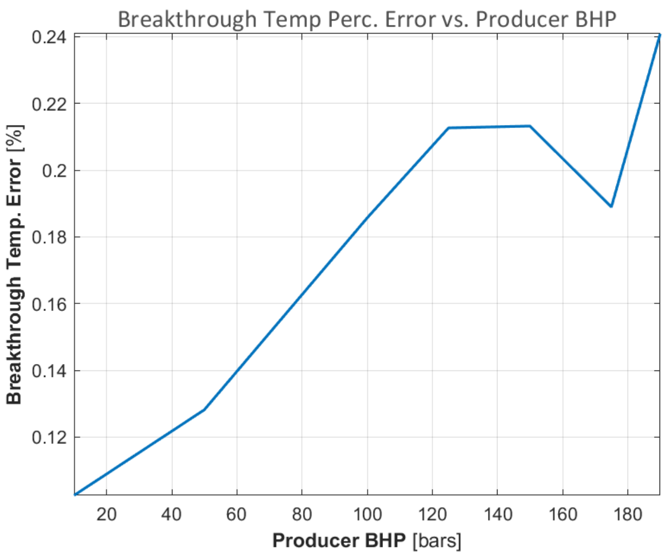


Figure 4.18 – Variation of Breakthrough curves' Average Error with Producer BHP

This graph shows that the error between full-scale model and proxy model breakthrough curves increases as the producer BHP increases. This shows that as the producer BHP is decreased, the streamlines' distribution stabilizes and does not vary significantly. For the purpose of this analysis, the injector BHP was kept at a constant 210 bar. Another perspective to view these results is that as the BHP range decreases, the error increases.

4.3.4 Injector BHP

Varying the injector bottom-hole pressure from 210 bars to 400 bars, yields the following 'breakthrough temperature' error curve:

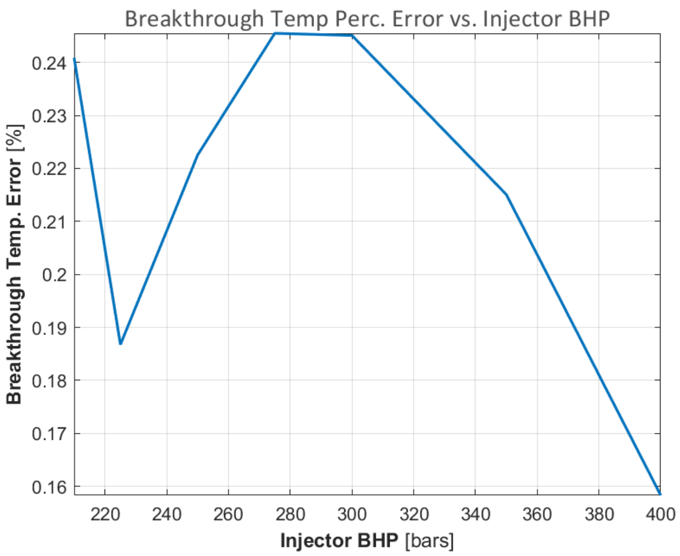


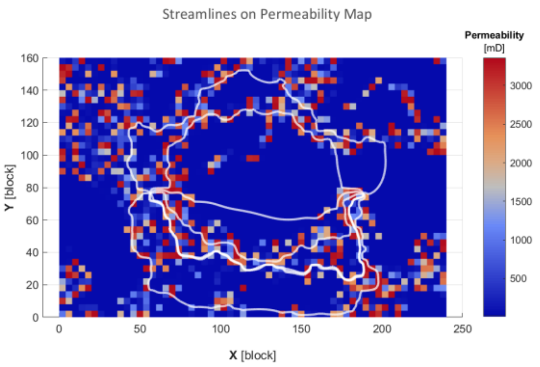
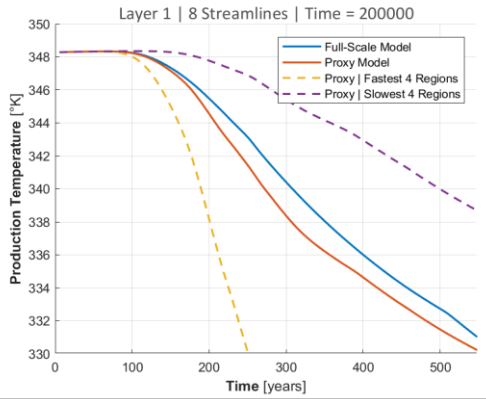
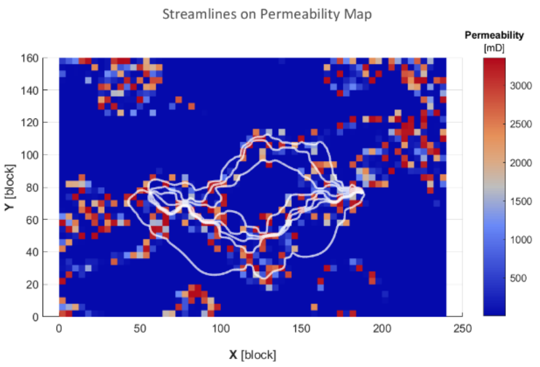
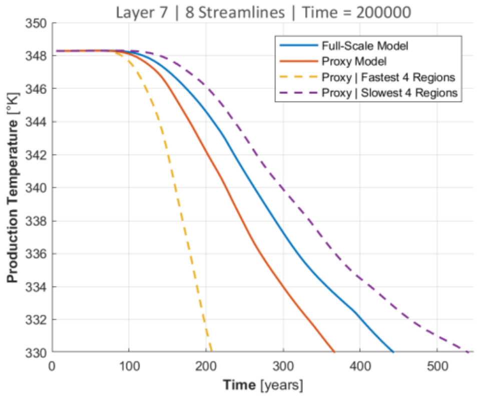
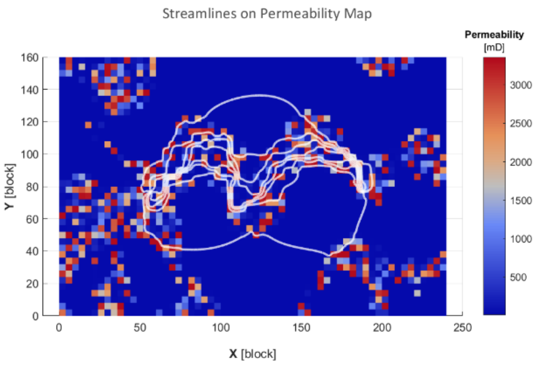
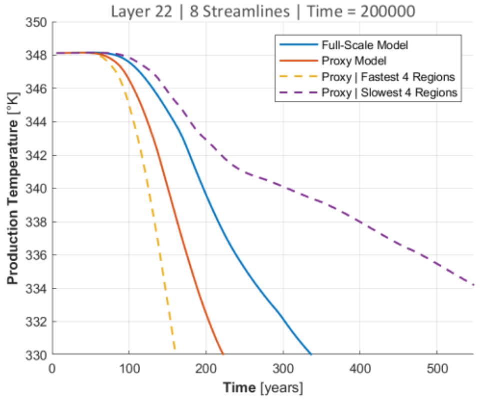
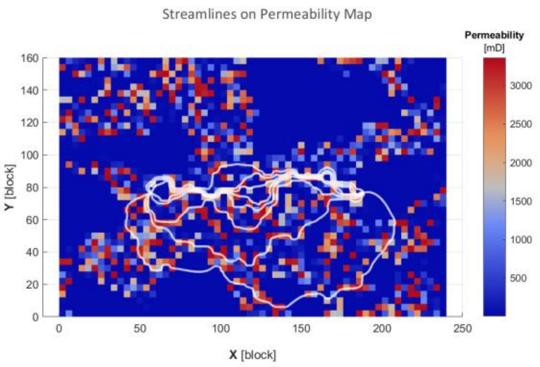
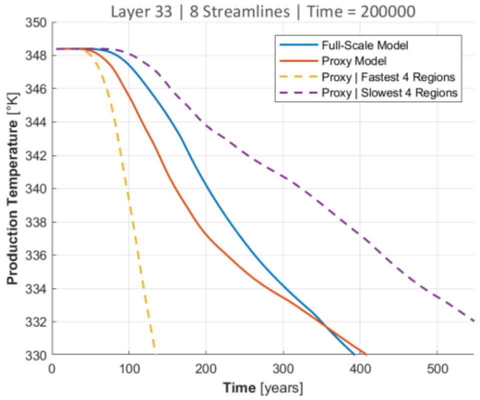
Figure 4.19 – Variation of Breakthrough curves' Average Error with Injector BHP

This graph shows that the error between full-scale model and proxy model breakthrough curves decreases as the injector BHP increases. This reveals that as the producer BHP is decreased, the streamlines' distribution stabilizes and does not vary significantly.

4.3.5 Layers

Until now, the layer that was under observation was the 2nd layer of 3D model [1]. But now a sensitivity analysis is performed with respect to the heterogeneity related to different layers in the model. Six different layers were selected based on the criteria that they have a reasonably fast break-through. Each layer has its own permeability and porosity distribution.

The following table shows the streamlines' distribution and the full-scale and proxy model breakthrough curves for each layer:

#	Streamlines	Temperature Breakthrough Curve
Layer 1	 Streamlines on Permeability Map Permeability [mD] 3000 2500 2000 1500 1000 500 Y [block] X [block] 0 20 40 60 80 100 120 140 160 50 150 200 250	
Layer 7	 Streamlines on Permeability Map Permeability [mD] 3000 2500 2000 1500 1000 500 Y [block] X [block] 0 20 40 60 80 100 120 140 160 50 150 200 250	
Layer 22	 Streamlines on Permeability Map Permeability [mD] 3000 2500 2000 1500 1000 500 Y [block] X [block] 0 20 40 60 80 100 120 140 160 50 150 200 250	
Layer 33	 Streamlines on Permeability Map Permeability [mD] 3000 2500 2000 1500 1000 500 Y [block] X [block] 0 20 40 60 80 100 120 140 160 50 150 200 250	

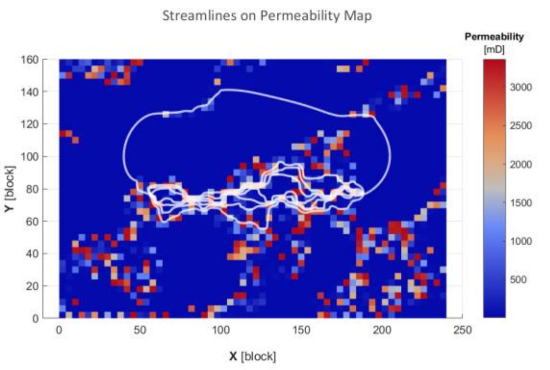
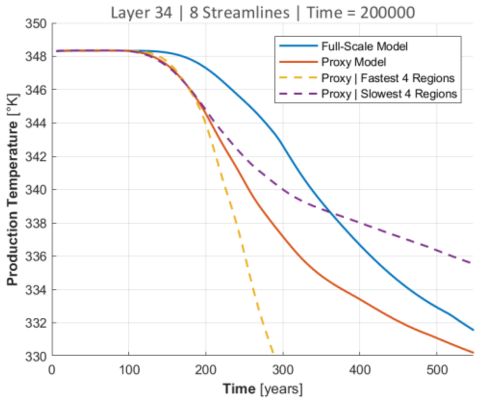
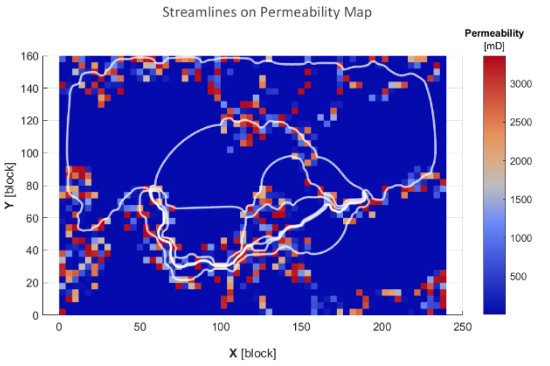
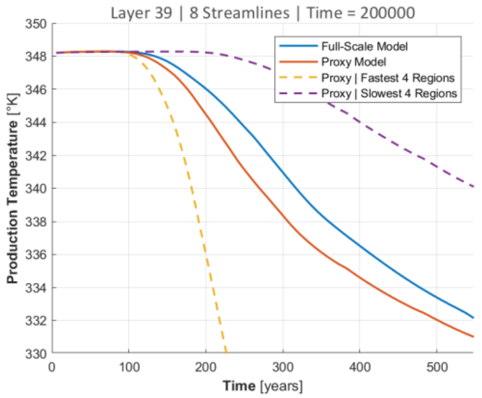
Layer 34		
Layer 39		

***Table 4.9** – Streamlines’ distribution and Breakthrough Curves' comparison for different layers*

As seen from the table of results above, the two layers with a high breakthrough curve mean error, are 22 and 34. This is due to the fact the streamlines are concentrated through a small region, rather than being spread as in layers 1 and 39. As it has already been established that dividing a layer into regions introduces error, and when several streamlines are passing through a cell, this further increases the error in the p-T distributions and consequently the temperature breakthrough curves.

5

Conclusion and Recommendations

Comparison between the full-scale model and proxy model reveal that the results for proxy model are quite optimistic. The error for the pressure and temperature distributions, as well as the breakthrough curves is within the acceptable tolerance. It has also been established that the time required to simulate the proxy-model is lower than the time for full-scale model. And as the number of fastest regions to simulate the proxy model decreases, the time required for simulation decreases but simultaneously the error between the breakthrough curves increases.

There are several factors that can increase the error in the proxy model results:

- Increase in Heterogeneity of the Reservoir
- Increase in Simulation Runtime
- Increase in Number of Streamlines
- Increase in the Producer BHP
- Decrease in the Injector BHP

While keeping in view, the factors contributing to the error, the proxy model can be used to obtain a good approximation of the pressure and temperature distributions, and predict the temperature breakthrough of the reservoir.

During the course of this project, several difficulties were encountered and several parameters were determined that decrease the accuracy of the proxy model. Based on these arguments, there are some recommendations for future work:

- The streamlines used in this work are traced only once, when the reservoir layer has reached steady-state. Future work can include periodic updating of streamlines (particularly during the transient state) for accurate dynamic modelling.

- The initial objective of this project was to define the regions using center-lines between streamlines, rather than the streamlines themselves. Several different methods of drawing center-lines had been used but every method had its disadvantages and fell short in some areas. If center-lines are to be used, a method needs to be devised to draw the center-line accurately.

- Pollock's method to trace streamlines is not very accurate since there are some assumptions and approximations that influence the results. After Pollock, few other methods have been devised to trace streamlines accurately and can be used.

- Pollock's equations assume orthogonal grid blocks, but very few real reservoirs models use such a strict Cartesian framework anymore. Transform corner-point geometry grids (CPG) into unit cubes, apply Pollock's method, and then transform exit coordinate back to physical space. Details are given by Prevost et al. (2001) [36].

- One of the features of existing streamline simulation is that the streamlines are truly 3D, rather than 2D as assumed several decades ago, in the 'streamtube' method. Streamlines appear as being 2D since they are generally depicted from a birds-eye perspective, but streamlines now correctly account for previously missing vertical component of flow description that is fundamental to current success of technology. Therefore, the approach used in this M.Sc. thesis project of 2D model can work more accurately with 3D models. Rather than simulating one layer, the entire reservoir can be simulated and streamlines can be traced in 3 dimensions, consequently giving rise to 3D streamtubes.

- In simulations involving the wells being flow-rate controlled, rather than BHP controlled, the flow-rate that was used for the wells in full-scale model cannot simply be divided equally into the number of regions in the proxy model. The flow-rate needs to be calculated for each region and may depend on more than one factor (e.g. average time-of-flight of region, volume of region). These factors need to be determined that govern the regional flowrate.

- In reality, geothermal reservoir not only produce hot water, but co-produce non-condensable gases and hydrocarbons (particularly gas). Geothermal reservoir simulations should be performed along with hydrocarbon co-production in order to determine its effects on the accuracy of the proxy model.

Appendix

A

Discretization of Energy Equations

A.1 Discretization of Energy Conservation equation:

This section illustrates the discretization of the energy conservation equation. The general form of this equation is given by the following expression [6]:

$$\frac{\partial}{\partial t}\left(\phi\sum_{p=1}^{n_p}\rho_p s_p u_{e,p}+(1-\phi)u_{e,r}\right)-\operatorname{div}\sum_{p=1}^{n_p}h_p\rho_p\left(K\frac{k_{r,p}}{\mu_p}(\nabla p_p-\gamma_p\nabla D)\right)+\operatorname{div}(\lambda\nabla T)+\sum_{p=1}^{n_p}h_p\rho_p\tilde{q}_p=0, \tag{A.1}$$

<u>Note:</u> Description of these variables can be found in the nomenclature

The first term (highlighted in orange) is the 'accumulation' term that determines the thermal energy in a system due to the fluid and the rock. The second term (highlighted in red) is the 'convection' term, the third term (highlighted in blue) is the 'conduction' term, while the fourth term (highlighted in green) is 'sink/source' term.

The following assumptions are taken into consideration for the above general equation:

- There is no porosity (ϕ = 0), therefore there are no fluids in the system.
- Since there is no fluid involved, 'fake' convection is being considered. The propagation of the front is only being simulated.
- For solids, c_p (specific heat capacity at constant pressure) is same as c_v (specific heat capacity at constant volume), therefore the enthalpy is same as internal energy.
- Specific heat capacity (c_p) is constant, and not a function of temperature.

After these assumptions, the energy equation simplifies to:

$$\frac{\partial u_{e,r}}{\partial t}-\operatorname{div}(h\rho\vec{u})+\operatorname{div}(\lambda\nabla T)+\dot{\varepsilon}=0. \tag{A.2}$$

This equation can be re-written as:

$$\rho c \frac{\partial T}{\partial t} - \rho c \cdot \nabla \cdot (\vec{u}T) + \nabla \cdot (\lambda \nabla T) + \dot{\varepsilon} = 0. \tag{A.3}$$

It is assumed that the surrounding rock thermally recharges the rock next to the fluid, by conduction. The thermal energy provided by conduction and sink/source (if there exists one) is stored by the rock or absorbed by the fluid and propagated away. Equation (A.3) can be re-arranged as following:

$$\nabla \cdot (\lambda \nabla T) + \dot{\varepsilon} = \rho c \frac{\partial T}{\partial t} + \rho c \cdot \nabla \cdot (\vec{u}T). \tag{A.4}$$

The expression is expanded in 3-dimension to yield the following:

$$\begin{aligned} \rightarrow \frac{\partial}{\partial x}\left(\lambda_x \frac{dT}{dx}\right) + \frac{\partial}{\partial y}\left(\lambda_y \frac{dT}{dy}\right) + \frac{\partial}{\partial z}\left(\lambda_z \frac{dT}{dz}\right) + \dot{\varepsilon} \\ = \rho c \frac{\partial T}{\partial t} + \rho c \left[u_x \frac{dT}{dx} + u_y \frac{dT}{dy} + u_z \frac{dT}{dz}\right]. \end{aligned} \tag{A.5}$$

The equation is discretized using:

- Cartesian grid
- ‘Finite Difference’ scheme
- Spatially using backward difference
- Temporally using Euler backward (i.e. implicit time scheme)

Following assumptions are further taken into consideration:

- Heat Diffusion (conduction) exists in all 3 dimensions
- Convection exists in all 3 dimensions
- Transient state (not only steady-state)
- Thermal conductivity is:
    - anisotropic (changes over dimension),
    - spatially variable (changes over location), but
    - constant temporally (does not change with time)

The value at the neighboring nodes can be approximated using Taylor Series:

$$\boldsymbol{T_i} = T_i, \tag{A.6}$$

$$\boldsymbol{T_{i-1}} = T_i - \left[\frac{(\Delta x)}{1!} \cdot \frac{\partial T}{\partial x}\right] + \left[\frac{(\Delta x)^2}{2!} \cdot \frac{\partial^2 T}{\partial x^2}\right] + O[(\Delta x)^3], \tag{A.7}$$

$$\boldsymbol{T_{i+1}} = T_i + \left[\frac{(\Delta x)}{1!} \cdot \frac{\partial T}{\partial x}\right] + \left[\frac{(\Delta x)^2}{2!} \cdot \frac{\partial^2 T}{\partial x^2}\right] + O[(\Delta x)^3]. \tag{A.8}$$

Subtracting equation (A.7) from equation (A.6) yields:

$$\frac{\boldsymbol{\partial T}}{\boldsymbol{\partial x}} = \frac{1}{\Delta x}(T_i - T_{i-1}) + O(\Delta x). \tag{A.9}$$

As can be observed from the result, expression (A.9) is 1st order accurate.

Individually discretizing the terms in equation (A.5):

$$\frac{\boldsymbol{\partial}}{\boldsymbol{\partial x}}\left(\lambda_x \frac{\boldsymbol{dT}}{\boldsymbol{dx}}\right) = \frac{1}{(\Delta x)^2}\left[\lambda_{x+0.5}(T_{i+1} - T_i) - \lambda_{x-0.5}(T_i - T_{i-1})\right], \tag{A.10}$$

$$\frac{\boldsymbol{\partial}}{\boldsymbol{\partial y}}\left(\lambda_y \frac{\boldsymbol{dT}}{\boldsymbol{dy}}\right) = \frac{1}{(\Delta y)^2}\left[\lambda_{y+0.5}(T_{j+1} - T_j) - \lambda_{y-0.5}(T_j - T_{j-1})\right], \tag{A.11}$$

$$\frac{\boldsymbol{\partial}}{\boldsymbol{\partial z}}\left(\lambda_z \frac{\boldsymbol{dT}}{\boldsymbol{dz}}\right) = \frac{1}{(\Delta z)^2}\left[\lambda_{z+0.5}(T_{k+1} - T_k) - \lambda_{z-0.5}(T_k - T_{k-1})\right], \tag{A.12}$$

$$\frac{\boldsymbol{\partial T}}{\boldsymbol{\partial x}} = \frac{1}{\Delta x}(T_i - T_{i-1}), \tag{A.13}$$

$$\frac{\boldsymbol{\partial T}}{\boldsymbol{\partial t}} = \frac{1}{\Delta t}(T^{n+1} - T^n). \tag{A.14}$$

Substituting equations (A.10) to (A.14) into equation (A.5) gives:

$$\begin{aligned}
\rightarrow\ &+ \frac{1}{(\Delta x)^2}\left[\lambda_{x+½}\left(T^{n+1}_{i+1,j,k} - T^{n+1}_{i,j,k}\right) - \lambda_{x-½}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i-1,j,k}\right)\right] \\
&+ \frac{1}{(\Delta y)^2}\left[\lambda_{y+½}\left(T^{n+1}_{i,j+1,k} - T^{n+1}_{i,j,k}\right) - \lambda_{y-½}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i,j-1,k}\right)\right] \\
&+ \frac{1}{(\Delta z)^2}\left[\lambda_{z+½}\left(T^{n+1}_{i,j,k+1} - T^{n+1}_{i,j,k}\right) - \lambda_{z-½}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i,j,k-1}\right)\right] \\
&+ \dot{\varepsilon} = \rho c \cdot \frac{1}{\Delta t}\left(T^{n+1}_{i,j,k} - T^{n}_{i,j,k}\right) \\
&+ \rho c \cdot u_{x-½} \cdot \frac{1}{\Delta x}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i-1,j,k}\right] \\
&+ \rho c \cdot u_{y-½} \cdot \frac{1}{\Delta y}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i,j-1,k}\right] \\
&+ \rho c \cdot u_{z-½} \cdot \frac{1}{\Delta z}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i,j,k-1}\right].
\end{aligned} \tag{A.15}$$

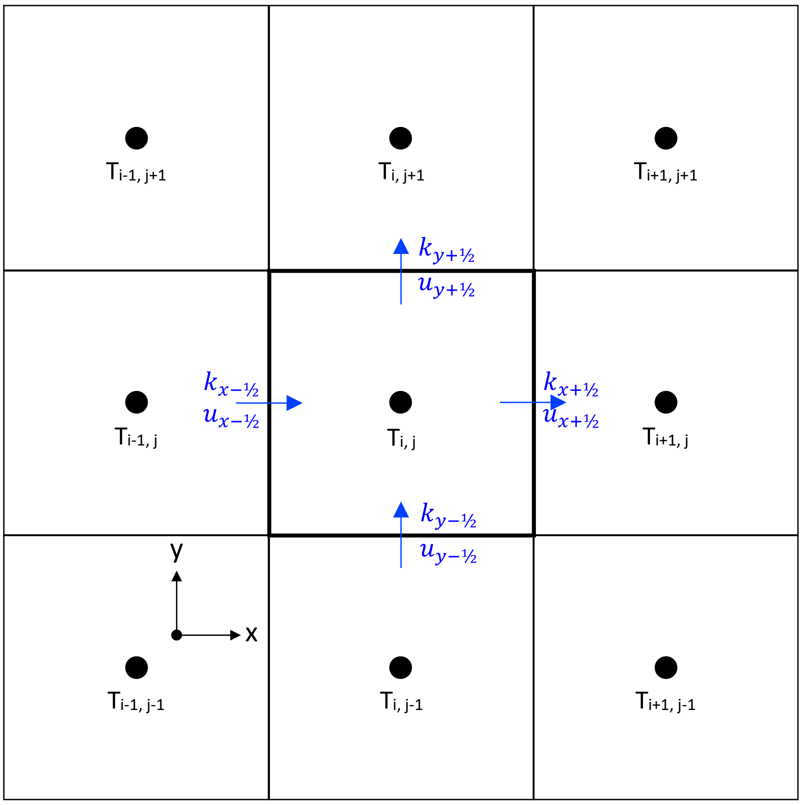


***Figure A.1** – Schematic of 2D grid cell*

For simplification purposes, let:

$$\alpha = \rho c, \qquad (A.16)$$

$$\beta_x = \frac{\Delta t}{(\Delta x)^2} \cdot \lambda_x, \quad (A.17) \qquad \beta_y = \frac{\Delta t}{(\Delta y)^2} \cdot \lambda_y, \quad (A.18) \qquad \beta_z = \frac{\Delta t}{(\Delta z)^2} \cdot \lambda_z, \quad (A.19)$$

$$\theta_x = \frac{\Delta t}{\Delta x} \cdot \rho c \cdot u_x, \quad (A.20) \qquad \theta_y = \frac{\Delta t}{\Delta y} \cdot \rho c \cdot u_y, \quad (A.21) \qquad \theta_z = \frac{\Delta t}{\Delta z} \cdot \rho c \cdot u_z. \quad (A.22)$$

Substituting these simplifications $(A.16)$ to $(A.22)$ into the derived expression $(A.15)$:

$$\begin{aligned}
\rightarrow\ & + \beta_{x+}\left(T^{n+1}_{i+1,j,k} - T^{n+1}_{i,j,k}\right) - \beta_{x-}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i-1,j,k}\right) \\
& + \beta_{y+}\left(T^{n+1}_{i,j+1,k} - T^{n+1}_{i,j,k}\right) - \beta_{y-}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i,j-1,k}\right) \\
& + \beta_{z+}\left(T^{n+1}_{i,j,k+1} - T^{n+1}_{i,j,k}\right) - \beta_{z-}\left(T^{n+1}_{i,j,k} - T^{n+1}_{i,j,k-1}\right) \\
& + \Delta t \cdot \dot{\varepsilon} = \alpha\left(T^{n+1}_{i,j,k} - T^{n}_{i,j,k}\right) \qquad (A.23) \\
& + \theta_{x-}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i-1,j,k}\right] \\
& + \theta_{y-}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i,j-1,k}\right] \\
& + \theta_{z-}\left[T^{n+1}_{i,j,k} - T^{n+1}_{i,j,k-1}\right].
\end{aligned}$$

In equation $(\mathrm{A}.23)$, the subscripts '+' and '-' refer to the values at the previous interface and the next interface respectively, e.g. β_{x+} refers to $\left(\frac{\Delta t}{(\Delta x)^2} \cdot \lambda_{x+½}\right)$ and θ_{x-} refers to $\left(\frac{\Delta t}{\Delta x} \cdot \rho c \cdot u_{x-½}\right)$.

$\boldsymbol{T^n}$ is the temperature at the current time-step, while $\boldsymbol{T^{n+1}}$ is the temperature at the next time-step. In order to determine the temperature at next time-step, Newton-Raphson method is used and may involve more than one iteration. Therefore the temperature at the current iteration can be denoted by $\boldsymbol{T^\nu}$ and temperature at next iteration denoted by $\boldsymbol{T^{\nu+1}}$. Once the temperature for all the grid cells falls within the specified tolerance, the solution at $\boldsymbol{T^{\nu+1}}$ becomes the solution for future time-step $\boldsymbol{T^{n+1}}$. Therefore, for derivation purposes, $\boldsymbol{T^{n+1}}$ can be re-written as $\boldsymbol{T^{\nu+1}}$. Furthermore, adding/subtracting all temperature terms by $\boldsymbol{T^\nu}$:

$$\begin{aligned}
\rightarrow\ & + \beta_{x+} \cdot [(T_{i+1}^{\nu+1} - T_{i+1}^{\nu} + T_{i+1}^{\nu}) - (T_i^{\nu+1} - T_i^{\nu} + T_i^{\nu})]_{j,k} \\
& - \beta_{x-} \cdot [(T_i^{\nu+1} - T_i^{\nu} + T_i^{\nu}) - (T_{i-1}^{\nu+1} - T_{i-1}^{\nu} + T_{i-1}^{\nu})]_{j,k} \\
& + \beta_{y+} \cdot \left[\left(T_{j+1}^{\nu+1} - T_{j+1}^{\nu} + T_{j+1}^{\nu}\right) - \left(T_j^{\nu+1} - T_j^{\nu} + T_j^{\nu}\right)\right]_{i,k} \\
& - \beta_{y-} \cdot \left[\left(T_j^{\nu+1} - T_j^{\nu} + T_j^{\nu}\right) - \left(T_{j-1}^{\nu+1} - T_{j-1}^{\nu} + T_{j-1}^{\nu}\right)\right]_{i,k} \\
& + \beta_{z+} \cdot [(T_{k+1}^{\nu+1} - T_{k+1}^{\nu} + T_{k+1}^{\nu}) - (T_k^{\nu+1} - T_k^{\nu} + T_k^{\nu})]_{i,j} \\
& - \beta_{z-} \cdot [(T_k^{\nu+1} - T_k^{\nu} + T_k^{\nu}) - (T_{k-1}^{\nu+1} - T_{k-1}^{\nu} + T_{k-1}^{\nu})]_{i,j} \\
& + \Delta t \cdot \dot{\varepsilon} = \alpha\left(T_{i,j,k}^{\nu+1} - T_{i,j,k}^{\nu} + T_{i,j,k}^{\nu} - T_{i,j,k}^{n}\right) \\
& + \theta_{x-}[(T_i^{\nu+1} - T_i^{\nu} + T_i^{\nu}) - (T_{i-1}^{\nu+1} - T_{i-1}^{\nu} + T_{i-1}^{\nu})]_{j,k} \\
& + \theta_{y-}\left[\left(T_j^{\nu+1} - T_j^{\nu} + T_j^{\nu}\right) - \left(T_{j-1}^{\nu+1} - T_{j-1}^{\nu} + T_{j-1}^{\nu}\right)\right]_{i,k} \\
& + \theta_{z-}[(T_k^{\nu+1} - T_k^{\nu} + T_k^{\nu}) - (T_{k-1}^{\nu+1} - T_{k-1}^{\nu} + T_{k-1}^{\nu})]_{i,j}.
\end{aligned} \tag{A.24}$$

Since,

$$\delta T^{\nu+1} = T^{\nu+1} - T^{\nu}, \tag{A.25}$$

where $\delta T^{\nu+1}$ is the difference in Temperature between next iteration and current iteration. Substituting this relation $(\mathrm{A}.25)$ into the derived expression $(\mathrm{A}.24)$ yields:

$$\begin{aligned}
\rightarrow\ & + \beta_{x+} \cdot [(\delta T_{i+1}^{\nu+1} + T_{i+1}^{\nu}) - (\delta T_i^{\nu+1} + T_i^{\nu})]_{j,k} \\
& - \beta_{x-} \cdot [(\delta T_i^{\nu+1} + T_i^{\nu}) - (\delta T_{i-1}^{\nu+1} + T_{i-1}^{\nu})]_{j,k} \\
& + \beta_{y+} \cdot \left[\left(\delta T_{j+1}^{\nu+1} + T_{j+1}^{\nu}\right) - \left(\delta T_j^{\nu+1} + T_j^{\nu}\right)\right]_{i,k} \\
& - \beta_{y-} \cdot \left[\left(\delta T_j^{\nu+1} + T_j^{\nu}\right) - \left(\delta T_{j-1}^{\nu+1} + T_{j-1}^{\nu}\right)\right]_{i,k} \\
& + \beta_{z+} \cdot [(\delta T_{k+1}^{\nu+1} + T_{k+1}^{\nu}) - (\delta T_k^{\nu+1} + T_k^{\nu})]_{i,j} \\
& - \beta_{z-} \cdot [(\delta T_k^{\nu+1} + T_k^{\nu}) - (\delta T_{k-1}^{\nu+1} + T_{k-1}^{\nu})]_{i,j} \\
& + \Delta t \cdot \dot{\varepsilon} = \alpha\left[\left(\delta T_{i,j,k}^{\nu+1} + T_{i,j,k}^{\nu}\right) - T_{i,j,k}^{n}\right] \\
& + \theta_{x-} \cdot [(\delta T_i^{\nu+1} + T_i^{\nu}) - (\delta T_{i-1}^{\nu+1} + T_{i-1}^{\nu})]_{j,k} \\
& + \theta_{y-} \cdot \left[\left(\delta T_j^{\nu+1} + T_j^{\nu}\right) - \left(\delta T_{j-1}^{\nu+1} + T_{j-1}^{\nu}\right)\right]_{i,k} \\
& + \theta_{z-} \cdot [(\delta T_k^{\nu+1} + T_k^{\nu}) - (\delta T_{k-1}^{\nu+1} + T_{k-1}^{\nu})]_{i,j}.
\end{aligned} \tag{A.26}$$

In the expression derived above, there are primarily two kinds of terms: $\boldsymbol{\delta T}$ and $\boldsymbol{T}$. The terms with $\boldsymbol{\delta T}$ form the Jacobian matrix, while the terms with $\boldsymbol{T}$ form the residual vector. In vector calculus, the Jacobian matrix is the matrix of all first-order partial derivatives of a vector-valued function. While the residual (also known as out-of-balance load vector), is the amount the system is out of equilibrium. Therefore, separating the variables into Jacobian and residual yields the following expressions:

$$\begin{aligned} Jacobian = & + \boldsymbol{\delta T^{v+1}_{i+1,j,k}}(\beta_{x+}) + \boldsymbol{\delta T^{v+1}_{i-1,j,k}}(\beta_{x-} + \theta_{x-}) \\ & + \boldsymbol{\delta T^{v+1}_{i,j+1,k}}(\beta_{y+}) + \boldsymbol{\delta T^{v+1}_{i,j-1,k}}(\beta_{y-} + \theta_{y-}) \\ & + \boldsymbol{\delta T^{v+1}_{i,j,k+1}}(\beta_{z+}) + \boldsymbol{\delta T^{v+1}_{i,j,k-1}}(\beta_{z-} + \theta_{z-}) \\ & + \boldsymbol{\delta T^{v+1}_{i,j,k}}[-\alpha - \beta_{x+} - \beta_{x-} - \beta_{y+} - \beta_{y-} - \beta_{z+} - \beta_{z-} - \theta_{x-} - \theta_{y-} - \theta_{z-}]. \end{aligned} \tag{A.27}$$

$$\begin{aligned} Residual = & + \boldsymbol{T^{v}_{i+1,j,k}}(-\beta_{x+}) + \boldsymbol{T^{v}_{i-1,j,k}}(-\beta_{x-} - \theta_{x-}) \\ & + \boldsymbol{T^{v}_{i,j+1,k}}(-\beta_{y+}) + \boldsymbol{T^{v}_{i,j-1,k}}(-\beta_{y-} - \theta_{y-}) \\ & + \boldsymbol{T^{v}_{i,j,k+1}}(-\beta_{z+}) + \boldsymbol{T^{v}_{i,j,k-1}}(-\beta_{z-} - \theta_{z-}) \\ & + \boldsymbol{T^{v}_{i,j,k}}[\alpha + \beta_{x+} + \beta_{x-} + \beta_{y+} + \beta_{y-} + \beta_{z+} + \beta_{z-} + \theta_{x-} + \theta_{y-} + \theta_{z-}] \\ & + \boldsymbol{T^{n}_{i,j,k}}(-\alpha) + (\dot{\varepsilon} \cdot \Delta t). \end{aligned} \tag{A.28}$$

A.2 Discretization of Heat Diffusion Equation (in Radial Coordinates)

Heat transfer occurs by conduction within the reservoir rock. Next, it is transferred from the rock to the fluid by conduction as well. In order to model the conduction phenomena, the following governing heat diffusivity equation (in cylindrical coordinates) is used [37]:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\lambda\frac{\partial T}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \emptyset}\left(\lambda\frac{\partial T}{\partial \emptyset}\right) + \frac{\partial}{\partial z}\left(\lambda\frac{\partial T}{\partial z}\right) + \dot{\varepsilon} = \rho c\frac{\partial T}{\partial t}, \tag{A.29}$$

The first term (highlighted in red) is heat flux in r-coordinate. Similarly, the second term (highlighted in green) is in the azimuthal coordinates while the third term (highlighted in blue) is for vertical coordinates. The fourth term (highlighted in orange) is the heat source/sink rate. The term on the right-hand side of the equation is the heat accumulation term. This shows that any heat flux coming into the control volume (from any of the coordinate directions), along with any heat generation or heat loss in the system, is accumulated within the rock.

In our case, the heat is assumed to diffuse in the radial direction only, and there is no heat generation or heat loss. These assumptions reduce the governing equation down to:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\lambda\frac{\partial T}{\partial r}\right) = \rho c\frac{\partial T}{\partial t}. \tag{A.30}$$

Further, in our case, the thermal conductivity is a function of radius $'r'$, i.e. it is not constant and may change with the radius. This is done to incorporate heterogeneity in our reservoir rock and make the code general, although initially for simulation purposes it will be kept constant.

The following figure shows the grid of a reservoir rock in radial direction:

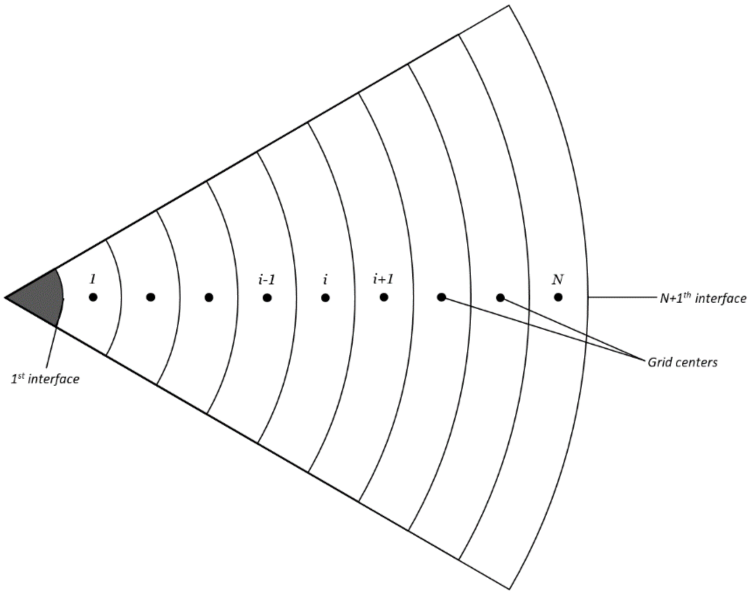


Figure A.2 *– Grid of a reservoir rock in radial direction*

The partial differential equations will be discretized using Finite Difference method. The spatial partial derivative will be approximated using the Taylor series:

$$f(x) = f(x), \tag{A.31}$$

$$f(x+\Delta x) = f(x) \quad + \quad \Delta x \cdot f'(x) \quad + \quad \frac{\Delta x^2}{2!} \cdot f''(x) \quad + \quad O(\Delta x^3), \tag{A.32}$$

$$f(x-\Delta x) = f(x) \quad - \quad \Delta x \cdot f'(x) \quad + \quad \frac{\Delta x^2}{2!} \cdot f''(x) \quad - \quad O(\Delta x^3). \tag{A.33}$$

Subtracting the equation (A.33) from (A.32) yields the following:

$$f'(x) = \frac{1}{2 \cdot \Delta x}[f(x+\Delta x) - f(x-\Delta x)]. \tag{A.34}$$

This is the central finite difference method. It shows the discretization in spatial coordinates. In our case, the grid spacing will be the spacing between two radii, and the temperature is a function of radius that is known at the nodes. Therefore the FD equation will be modified to:

$$\frac{\partial T}{\partial r} = \frac{1}{2 \cdot \Delta r}[T_{i+1} - T_{i-1}]. \tag{A.35}$$

While the temporal partial derivative will be approximated using the following expression:

$$\frac{\partial T}{\partial t} = \frac{1}{\Delta t}[T_i^{n+1} - T_i^{n}]. \tag{A.36}$$

And since the Euler backward method (Implicit) numerical scheme is being used to determine the temperature at the future time step, substituting expressions $(\mathrm{A}.35)$ and $(\mathrm{A}.36)$ into expression $(\mathrm{A}.30)$ yield the following:

$$\frac{1}{(\rho c r)_i} \cdot \frac{\Delta t}{(\Delta r)^2} \cdot \left[\left(r_{i+\frac{1}{2}} \cdot \lambda_{i+\frac{1}{2}} \cdot (T_{i+1}^{n+1} - T_i^{n+1}) \right) - \left(r_{i-\frac{1}{2}} \cdot \lambda_{i-\frac{1}{2}} \cdot (T_i^{n+1} - T_{i-1}^{n+1}) \right) \right] = T_i^{n+1} - T_i^n. \tag{A.37}$$

For simplification of this expression, the following parameters are introduced:

$$\alpha = r_{i+\frac{1}{2}} \cdot \lambda_{i+\frac{1}{2}}, \tag{A.38}$$

$$\beta = r_{i-\frac{1}{2}} \cdot \lambda_{i-\frac{1}{2}}, \tag{A.39}$$

$$\gamma = \frac{1}{(\rho c r)_i} \cdot \frac{\Delta t}{(\Delta r)^2}. \tag{A.40}$$

Introducing these parameters into expression $(\mathrm{A}.37)$ yields the following:

$$\gamma[\alpha(T_{i+1}^{n+1} - T_i^{n+1}) - \beta(T_i^{n+1} - T_{i-1}^{n+1})] = T_i^{n+1} - T_i^n. \tag{A.41}$$

A.3 Analytical solution for Temperature Distribution profile in a Cylindrical Wall

In a geothermal application, heat is transferred through the surrounding rocks and to the fluid passing through, by the phenomena of conduction. This can be modelled as a tube (i.e. 1D stream passing through the rocks) and the heat energy being transferred from the perpendicular direction as illustrated in Figure 2.1. The energy transferred to the fluid can be modelled using Fourier's Law of conduction given by the expression below:

$$\dot{\varepsilon}_{conduction} = \nabla \cdot (\lambda \nabla T), \tag{A.42}$$

Since our system is modelled as a tube, it would be more convenient to use the radial coordinate system for the heat conduction phenomena rather than the Cartesian coordinate system. Here, the heat is being transferred in the r-coordinate direction.

In order to model the heat flow through pipe wall, it is convenient to represent the phenomena using the cross-section of tube, as shown below:

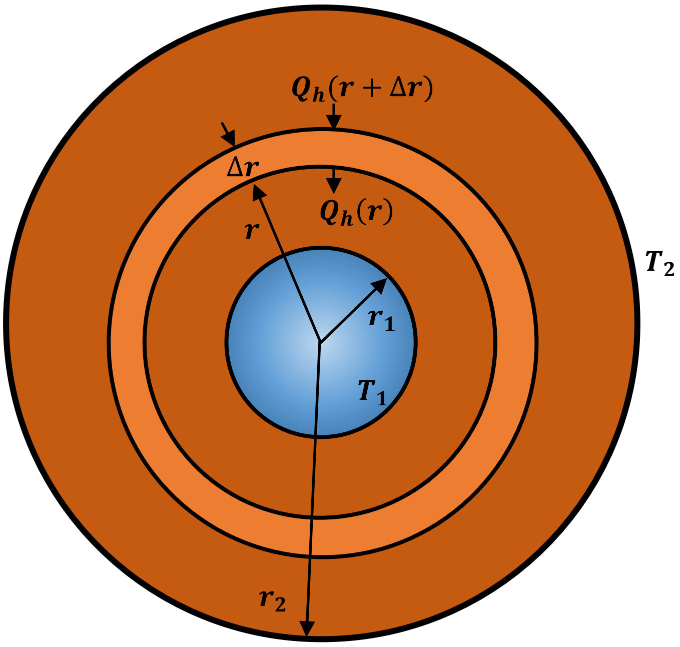


Figure A.3 *– Cross-section of pore (modelled as tube)* [38]

A cylindrical shell (*colored light orange in Figure A.3*) is considered with a radius (r) and outer radius $(r + \Delta r)$. The length (L) of this shell extends entire length of the pipe. $Q_h(r)$ is the heat flow rate within pipe wall. As shown, the heat flow rate into the cylindrical shell is $Q_h(r)$, while the heat flow rate out of the shell is $Q_h(r + \Delta r)$. At steady state, the heat flux being transferred through the surrounding rock and to the fluid stream is constant, i.e.:

$$Q_h(r) = Q_h(r + \Delta r). \tag{A.43}$$

Rearranging this result, along with division by Δr and taking the limit to zero, yields:

$$\lim_{\Delta r \to 0} \frac{Q_h(r + \Delta r) - Q_h(r)}{\Delta r} = \frac{dQ_h}{dr} = 0. \tag{A.44}$$

Integration leads to the result:

$$Q_h = constant. \tag{A.45}$$

Since the heat flow rate can also be written as:

$$Q_h = q_{h,rad} \cdot A, \tag{A.46}$$

where q is the heat flux crossing the surface, the subscript rad denotes the heat flux in radial direction, and A is the area of cylindrical surface that is normal to the r-direction:

$$A = 2\pi r L. \tag{A.47}$$

Deriving from Fourier's Law (equation (2.4)), it can be determined that heat flux in the radial direction is:

$$q_{h,rad} = -\lambda \frac{dT}{dr}. \tag{A.48}$$

Substituting this expression $(\mathrm{A}.48)$ and area equation $(\mathrm{A}.47)$ into equation $(\mathrm{A}.45)$ yields:

$$-2\pi rL \cdot \lambda \frac{dT}{dr} = constant. \tag{A.49}$$

The $2\pi L\lambda$ is a constant and can be merged to the R.H.S. of the equation. Next, rearranging the equation results:

$$dT = constant \cdot \frac{dr}{r}. \tag{A.50}$$

Integrating both the sides yield:

$$\int dT = C_1 \int \frac{1}{r} dr$$

$$T + C_{2a} = C_1 \cdot \ln(r) + C_{2b}$$

$$T = C_1 \cdot \ln(r) + C_2 \tag{A.51}$$

The constants of integration can be determined by substituting the two boundary conditions into the following expressions:

$$C_1 = \frac{T_1 - T_2}{\ln(r_1/r_2)}, \tag{A.52}$$

$$C_2 = T_1 - \frac{T_1 - T_2}{\ln(r_1/r_2)} \cdot \ln r_1. \tag{A.53}$$

Substituting these expressions into the third-last equation yields the expression for temperature distribution profile in a cylindrical shaped wall [38]:

$$T = T_1 - \frac{\ln(r/r_1)}{\ln(r_2/r_1)} \cdot (T_1 - T_2). \tag{A.54}$$

B

Connection List for Unstructured Grid

A governing partial differential equation can be comprised of more than one component that define a physical phenomenon. For example, the energy equation that is expressed by equation (1.2). It comprises of the four physical phenomena, i.e. heat storage, convection, conduction and energy sink/source. The solution is found by the linearization of these partial differential equation, of which, Newton-Raphson method is the most common method. This section describes how the Jacobian matrix and residual vector can be constructed by forming a loop over each of the four physical phenomenon, rather than having to calculate them at once. This procedure will help in the understanding of connection list and transmissibility in unstructured gridding.

Equations $(A.27)$ and $(A.28)$ are the Jacobian and residual expressions for the discretized form of Energy equation in 3-dimensional form. Reducing these equations to 1-dimension yield the following expressions:

$$\because Jacobian = \boldsymbol{\partial T_{i-1}^{\nu+1}}(\beta_{i-½}\Delta t + \theta_{i-½}\Delta t) + \boldsymbol{\partial T_{i}^{\nu+1}}(-\alpha_i - \beta_{i-½}\Delta t - \beta_{i+½}\Delta t - \theta_{i-½}\Delta t) + \boldsymbol{\partial T_{i+1}^{\nu+1}}(\beta_{i+½}\Delta t) \tag{B.1}$$

$$\because Residual = \boldsymbol{T_{i-1}^{\nu}}(-\beta_{i-½}\Delta t - \theta_{i-½}\Delta t) + \boldsymbol{T_{i}^{\nu}}(\alpha_i + \beta_{i-½}\Delta t + \beta_{i+½}\Delta t + \theta_{i-½}\Delta t) + \boldsymbol{T_{i+1}^{\nu}}(-\beta_{i+½}\Delta t) + \boldsymbol{T_{i}^{n}}(-\alpha_i) + (-\dot{\varepsilon}_i\Delta t) \tag{B.2}$$

Where the terms α, β and θ are defined by the expressions $(A.16)$, $(A.17)$ and $(A.20)$. For a brief overview, the term highlighted in orange is the 'accumulation' term that determines the thermal energy in a system due to the rock (and fluid). The term highlighted in red is due to 'convection', the term highlighted in blue is due to 'conduction', while the remaining term highlighted in green is due to 'sink/source'. The terms β_x and θ_x are calculated at the cell interfaces, rather than cell centers, therefore their subscripts include a ½ at the end (e.g. $i-½$ and $i+½$). The term β_x is the transmissibility due to conduction between two adjacent cells, and similarly θ_x is transmissibility due to convection. Lastly, the assumptions discussed in Appendix 'A', hold valid for this section as well.

The following figure shows the schematic for a 1-D grid:

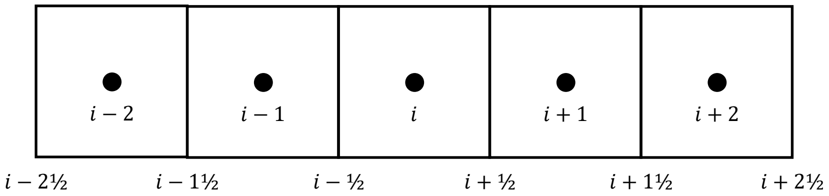


***Figure B.1** – Schematic for 1-D grid*

Once the entries for the Jacobian matrix and residual vector are filled, they appear as shown below. The unknowns can be determined by taking the inverse of the Jacobian matrix and multiplying it to the R.H.S. of the equation.

$$\underbrace{\begin{bmatrix} J_{i-2,i-2} & J_{i-2,i-1} & & & \\ J_{i-1,i-2} & J_{i-1,i-1} & J_{i-1,i} & & \\ & J_{i,i-1} & J_{i,i} & J_{i,i+1} & \\ & & J_{i+1,i} & J_{i+1,i+1} & J_{i+1,i+2} \\ & & & J_{i+2,i+1} & J_{i+2,i+2} \end{bmatrix}}_{Jacobian\ Matrix} \underbrace{\begin{bmatrix} \boldsymbol{\partial T^{\nu+1}_{i-2}} \\ \boldsymbol{\partial T^{\nu+1}_{i-1}} \\ \boldsymbol{\partial T^{\nu+1}_{i}} \\ \boldsymbol{\partial T^{\nu+1}_{i+1}} \\ \boldsymbol{\partial T^{\nu+1}_{i+2}} \end{bmatrix}}_{Unknowns} = \underbrace{\begin{bmatrix} R_{i-2} \\ R_{i-1} \\ R_{i} \\ R_{i+1} \\ R_{i+2} \end{bmatrix}}_{\substack{Residual \\ vector}} \qquad \text{(B.3)}$$

Each non-zero element of the Jacobian matrix has two entries in the subscript that are separated by a comma. The first number corresponds to the index of the cell the Jacobian is being constructed for, while the second number corresponds to the index of the neighboring cell. For example, for entry $J_{i,i+1}$, the Jacobian is being written for cell i and $i+1$ is the neighboring cell.

The following expression shows how the Jacobian matrix is the sum of matrices due to different physical phenomena:

$$\begin{bmatrix} J_{i-2,i-2} & J_{i-2,i-1} & & & \\ J_{i-1,i-2} & J_{i-1,i-1} & J_{i-1,i} & & \\ & J_{i,i-1} & J_{i,i} & J_{i,i+1} & \\ & & J_{i+1,i} & J_{i+1,i+1} & J_{i+1,i+2} \\ & & & J_{i+2,i+1} & J_{i+2,i+2} \end{bmatrix}$$
$$= \begin{bmatrix} -\alpha_{-2,-2} & & & & \\ & -\alpha_{-1,-1} & & & \\ & & -\alpha_{i,i} & & \\ & & & -\alpha_{+1,+1} & \\ & & & & -\alpha_{+2,+2} \end{bmatrix} + \Delta t \begin{bmatrix} \theta_{-2\frac{1}{2}} & & & & \\ \theta_{-1\frac{1}{2}} & -\theta_{-1\frac{1}{2}} & & & \\ & \theta_{-\frac{1}{2}} & -\theta_{-\frac{1}{2}} & & \\ & & \theta_{+\frac{1}{2}} & -\theta_{+\frac{1}{2}} & \\ & & & \theta_{+1\frac{1}{2}} & -\theta_{+1\frac{1}{2}} \end{bmatrix} \qquad \text{(B.4)}$$
$$+ \Delta t \begin{bmatrix} -(\beta_{-2\frac{1}{2}} + \beta_{-1\frac{1}{2}}) & \beta_{-1\frac{1}{2}} & & & \\ \beta_{-1\frac{1}{2}} & -(\beta_{-1\frac{1}{2}} + \beta_{-\frac{1}{2}}) & \beta_{-\frac{1}{2}} & & \\ & \beta_{-\frac{1}{2}} & -(\beta_{-\frac{1}{2}} + \beta_{+\frac{1}{2}}) & \beta_{+\frac{1}{2}} & \\ & & \beta_{+\frac{1}{2}} & -(\beta_{+\frac{1}{2}} + \beta_{+1\frac{1}{2}}) & \beta_{+1\frac{1}{2}} \\ & & & \beta_{+1\frac{1}{2}} & -(\beta_{+1\frac{1}{2}} + \beta_{+2\frac{1}{2}}) \end{bmatrix}$$

For the 1-D grid illustrated above, the following connection lists are generated:

Cell Index	Neighbor Cell Index	Transmissibility	
		Conduction	Convection
$i-2$	$i-1$	$\beta_{i-1½}$	$\theta_{i-1½}$
$i-1$	i	$\beta_{i-½}$	$\theta_{i-½}$
i	$i+1$	$\beta_{i+½}$	$\theta_{i-½}$
$i+1$	$i+2$	$\beta_{i+1½}$	$\theta_{i-½}$

Table B.1 – Example of 'Conduction' and 'convection' connection lists for 1-D grid

These connection lists can be calculated and stored prior, and then used during the course of simulation, unless any of the parameters is changing for which the connection lists need to be updated at every time-step. For example, density (ρ), thermal conductivity (λ) and specific heat capacity (c) can be a function of temperature, therefore β and θ may need to be updated on every time-step. Similarly, if the pressure distribution is not steady-state and the velocity is varying, therefore θ may need to be updated at every time-step.

From the 1-D grid above, two adjacent cells are selected. The Jacobian and residual for cells i and $i+1$ are written. Next, the entries for energy transfer between cells i and $i+1$ remain, while the remaining entries (energy transfer between other neighboring cells) are omitted by crossing-out. The objective of this activity is to illustrate the transmissibility between two cells and consequently create a connection list between them.

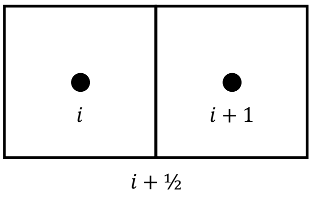


Figure B.2 – Schematic for 2 adjacent cells with energy transfer between them via the interface

Jacobian and Residual for cell i:

$$\rightarrow Jacobian(i) = \cancel{\partial T_{i-1}^{\nu+1}(\beta_{i-½}\Delta t + \theta_{i-½}\Delta t)} + \partial T_{i}^{\nu+1}(-\alpha_i \cancel{- \beta_{i-½}\Delta t} - \beta_{i+½}\Delta t \cancel{- \theta_{i-½}\Delta t}) + \partial T_{i+1}^{\nu+1}(\beta_{i+½}\Delta t)$$

$$\therefore Jacobian(i) = \partial T_{i}^{\nu+1}(-\alpha_i - \beta_{i+½}\Delta t) + \partial T_{i+1}^{\nu+1}(\beta_{i+½}\Delta t) \qquad \text{(B.5)}$$

$$\rightarrow Residual(i) = \cancel{T_{i-1}^{\nu}(-\beta_{i-½}\Delta t - \theta_{i-½}\Delta t)} + T_{i}^{\nu}(\alpha_i \cancel{+ \beta_{i-½}\Delta t} + \beta_{i+½}\Delta t \cancel{+ \theta_{i-½}\Delta t}) + T_{i+1}^{\nu}(-\beta_{i+½}\Delta t) + T_{i}^{n}(-\alpha_i) + (-\dot{\varepsilon}_i\Delta t)$$

$$\therefore Residual(i) = T_{i}^{\nu}(\alpha_i + \beta_{i+½}\Delta t) + T_{i+1}^{\nu}(-\beta_{i+½}\Delta t) + T_{i}^{n}(-\alpha_i) + (-\dot{\varepsilon}_i\Delta t) \qquad \text{(B.6)}$$

Jacobian and Residual for cell $i+1$:

$$\rightarrow Jacobian(i+1)$$
$$= \boldsymbol{\partial T_i^{\nu+1}}(\beta_{i+½}\Delta t + \theta_{i+½}\Delta t) + \boldsymbol{\partial T_{i+1}^{\nu+1}}(-\alpha_{i+1} - \beta_{i+½}\Delta t \cancel{- \beta_{i+1½}\Delta t} - \theta_{i+½}\Delta t)$$
$$\cancel{+ \boldsymbol{\partial T_{i+2}^{\nu+1}}(\beta_{i+1½}\Delta t)}$$

$$\rightarrow Jacobian(i+1) = \boldsymbol{\partial T_i^{\nu+1}}(\beta_{i+½}\Delta t + \theta_{i+½}\Delta t) + \boldsymbol{\partial T_{i+1}^{\nu+1}}(-\alpha_{i+1} - \beta_{i+½}\Delta t - \theta_{i+½}\Delta t) \qquad (B.7)$$

$$\rightarrow Residual(i+1)$$
$$= \boldsymbol{T_i^{\nu}}(-\beta_{i+½}\Delta t - \theta_{i+½}\Delta t) + \boldsymbol{T_{i+1}^{\nu}}(\alpha_{i+1} + \beta_{i+½}\Delta t \cancel{+ \beta_{i+1½}\Delta t} + \theta_{i+½}\Delta t)$$
$$\cancel{+ \boldsymbol{T_{i+2}^{\nu}}(-\beta_{i+1½}\Delta t)} + \boldsymbol{T_{i+1}^{n}}(-\alpha_{i+1}) + (-\dot{\varepsilon}_{i+1}\Delta t)$$

$$\therefore Residual(i+1)$$
$$= \boldsymbol{T_i^{\nu}}(-\beta_{i+½}\Delta t - \theta_{i+½}\Delta t) + \boldsymbol{T_{i+1}^{\nu}}(\alpha_{i+1} + \beta_{i+½}\Delta t + \theta_{i+½}\Delta t) + \boldsymbol{T_{i+1}^{n}}(-\alpha_{i+1}) \qquad (B.8)$$
$$+ (-\dot{\varepsilon}_{i+1}\Delta t)$$

It is necessary to re-emphasize that the four expressions $(B.5)$, $(B.6)$, $(B.7)$ and $(B.8)$ formulated above are the Jacobian and residual entries for energy transfer only between cells i and $i+1$, while the energy transfers with other neighboring cells are omitted. Also that these expressions will calculate the entries at once. Next these entries are broken down into their respective physical phenomenon (conduction, convection, accumulation and source), in order to be added in a loop.

In order to do this, the entries of Jacobian matrix and residual vector are initialized to zero. Alternately, this can be done by generating a 'J' matrix, filled with zeros, with dimensions $N \times N$ and 'Res' vector with dimensions $N \times 1$ (where N is the total number of cells in the grid). The following commands can be used in MATLAB in order to obtain null matrices:

```
J = zeros(N,N)

Res = zeros(N,1)
```

Next the coefficient of each of the physical phenomenon (mentioned above) is added to these existing entries of Jacobian matrix and residual vector, in a loop. These coefficients may constitute of the transmissibility, such as β and θ in conduction and convection loops, respectively.

Loop over Accumulation term:

$$J(i,i) \qquad = J(i,i) \qquad - \alpha_i \qquad (B.9a)$$

$$J(i+1,i+1) = J(i+1,i+1) - \alpha_{i+1} \qquad (B.9b)$$

$$Res(i) \qquad = Res(i) \qquad + \alpha_i(T_i^{\nu} - T_i^{n}) \qquad (B.10a)$$

$$Res(i+1) \qquad = Res(i+1) \qquad + \alpha_{i+1}(T_{i+1}^{\nu} - T_{i+1}^{n}) \qquad (B.10b)$$

Loop over Sink/Source terms:

$$Res(i) \qquad = Res(i) \qquad - \dot{\varepsilon}_i\Delta t \qquad (B.11a)$$

$$Res(i+1) \qquad = Res(i+1) - \dot{\varepsilon}_{i+1}\Delta t \qquad (B.11b)$$

Loop over Conduction (Diffusion) terms:

$$J(i,i) = J(i,i) - \beta_{i+½}\Delta t \tag{B.12a}$$

$$J(i,i+1) = J(i,i+1) + \beta_{i+½}\Delta t \tag{B.12b}$$

$$J(i+1,i) = J(i+1,i) + \beta_{i+½}\Delta t \tag{B.12c}$$

$$J(i+1,i+1) = J(i+1,i+1) - \beta_{i+½}\Delta t \tag{B.12d}$$

$$Res(i) = Res(i) + \beta_{i+½}\Delta t(T_i^\nu - T_{i+1}^\nu) \tag{B.13a}$$

$$Res(i+1) = Res(i+1) + \beta_{i+½}\Delta t(-T_i^\nu + T_{i+1}^\nu) \tag{B.13b}$$

Loop over Convection terms:

$$J(i+1,i) = J(i+1,i) + \theta_{i+½}\Delta t \tag{B.14a}$$

$$J(i+1,i+1) = J(i+1,i+1) - \theta_{i+½}\Delta t \tag{B.14b}$$

$$Res(i+1) = Res(i+1) + \theta_{i+½}\Delta t(-T_i^\nu + T_{i+1}^\nu) \tag{B.15}$$

C

Tracing Streamlines

This section of Appendix focuses on the methodology, assumptions and derivation of equations required for tracing the streamlines, utilizing a semi-analytical particle tracking method developed by David Pollock [2]. This method was developed for use with velocities generated from block centered finite-difference ground-water flow models. This streamline tracing method is semi-analytical because each streamline is made up of a series of streamline segments in single grid, and the streamline segment of each grid is determined by the analytical method.

For analytical solutions, it is simple since the velocity vector field can be computed at every point in field. But for finite-difference models, it becomes necessary to establish an interpolation scheme so that the velocity vector can be calculated at every point in flow field. A variety of interpolation schemes are available but the three simple approaches are:

1) **Step function interpolation** - assumes velocity components are constant between nodes but suddenly change value at the node.
2) **Simple linear interpolation** - assumes principal velocity components vary linearly within grid cell only with respect to their own coordinate direction, and independent on other directions.
3) **Multi-linear interpolation** - assumes principal velocity components vary linearly within grid cell with respect to all coordinate directions.

For this work, simple linear interpolation is used to generate velocity vector field. The three principal velocity component functions can be integrated directly within each individual grid cell to obtain an analytical expression for path line segments.

Once the pressure distribution across the reservoir is obtained using a Pressure solver, the velocity at the interface of each cell in the x-direction can be calculated using Darcy's Law:

$$v_x = \frac{Q_{v,x}}{A_x} = -\frac{k_x}{\mu \cdot \Delta x} \cdot (P_i - P_{i+1}). \tag{C.1a}$$

Similar expression can be obtained for y and z directions.

$$v_y = \frac{Q_{v,y}}{A_y} = -\frac{k_y}{\mu \cdot \Delta y} \cdot \left(P_j - P_{j+1}\right), \tag{C.1b}$$

$$v_z = \frac{Q_{v,z}}{A_z} = -\frac{k_z}{\mu \cdot \Delta z} \cdot (P_k - P_{k+1}), \tag{C.1c}$$

Next, some sort of piece-wise continuous interpolation function can be used to calculate the velocity components within a cell based on values at the interface. If simple linear interpolation is being used, principal velocity components can be expressed by the following expressions:

$$v_x = v_{x1} + \Psi_x \cdot (x - x_1), \tag{C.2a}$$

$$v_y = v_{y1} + \Psi_y \cdot (y - y_1), \tag{C.2b}$$

$$v_z = v_{z1} + \Psi_z \cdot (z - z_1), \tag{C.2c}$$

where Ψ_x, Ψ_y and Ψ_z are the velocity gradients within the cell and are defined as:

$$\Psi_x = (v_{x2} - v_{x1})/\Delta x, \tag{C.3a}$$

$$\Psi_y = (v_{y2} - v_{y1})/\Delta y, \tag{C.3b}$$

$$\Psi_z = (v_{z2} - v_{z1})/\Delta z. \tag{C.3c}$$

Linear interpolation helps producing a continuous velocity field within each individual grid cell that also satisfies the differential mass conservation equation, given that any internal sinks or sources are thought of as being uniformly distributed within cell.

Subsequent expressions will be derived for x-direction. Similar expressions can be derived for other dimensions simply by replacing the subscript x with y and z.

Taking into account the movement of a particle through a finite difference cell. As the particle moves through the cell, the rate of change of velocity (i.e. acceleration) in x-direction is given by the expression below. Using chain rule, this acceleration can be expressed as the product of velocity gradient and particle velocity:

$$\left(\frac{dv_x}{dt}\right)_{pt} = \left(\frac{dv_x}{dx}\right) \cdot \left(\frac{dx}{dt}\right)_{pt}, \tag{C.4}$$

where subscript '*pt*' refers to the particle. The second term on the right-hand-side of the equation $(dx/dt)_{pt}$ is the velocity (time rate of change) in x-direction of particle:

$$v_{x,pt} = \left(\frac{dx}{dt}\right)_{pt}. \tag{C.5}$$

Differentiating equation (C.2a) with respect to $'x'$ yields an additional relation:

$$\left(\frac{dv_x}{dx}\right) = \Psi_x. \tag{C.6}$$

Substituting expressions (C.5) and (C.6) into equation (C.4) yields the following expression

$$\left(\frac{dv_x}{dt}\right)_{pt} = \Psi_x \cdot v_{x,pt}. \tag{C.7}$$

This expression can be rearranged in the form of integral to give:

$$\frac{1}{v_{x,pt}} \cdot dv_x = \Psi_x \cdot dt. \tag{C.8}$$

The expression can be integrated further between times t_1 and t, where $\Delta t = t - t_1$:

$$\int_{v_1}^{v} \left(\frac{1}{v_{x,pt}}\right) dv_x = \int_{t_1}^{t} (\Psi_x) dt, \tag{C.9}$$

$$\ln\left(\frac{v_{x,pt}(t)}{v_{x,pt}(t_1)}\right) = \Psi_x \cdot \Delta t. \tag{C.10}$$

By taking exponential of each side of equation, and substituting it into equation (C.2a), the following relation is obtained:

$$x_{pt}(t) = x_1 + \frac{1}{\Psi_x}\left[v_{x,pt}(t_1) \cdot e^{\Psi_x \cdot \Delta t} - v_{x1}\right]. \tag{C.11a}$$

Similar procedure from expressions (C.4) to (C.11a) can be performed for the other two principal directions 'y' and 'z' to obtain the following expression:

$$y_{pt}(t) = y_1 + \frac{1}{\Psi_y}\left[v_{y,pt}(t_1) \cdot e^{\Psi_y \cdot \Delta t} - v_{y1}\right], \tag{C.11b}$$

$$z_{pt}(t) = z_1 + \frac{1}{\Psi_z}\left[v_{z,pt}(t_1) \cdot e^{\Psi_z \cdot \Delta t} - v_{z1}\right]. \tag{C.11c}$$

Coordinates at any future time (t) can be computed directly from these equations, where $v_{x,pt}(t_1)$, $v_{y,pt}(t_1)$ and $v_{z,pt}(t_1)$ are particle velocities as functions of particle coordinates at time t_1, with known values.

The equations derived above can be used for a steady-state flow where the particle's exit point from a cell can be determined directly, given any known starting location within cell.

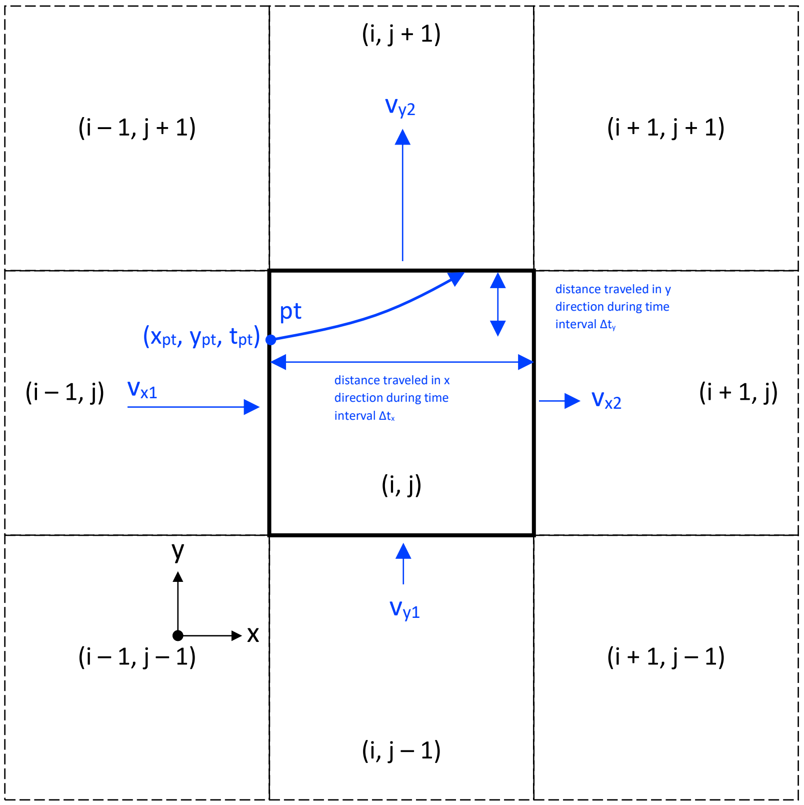


***Figure C.1** – Schematic of particle path through 2D grid cell*

Considering the two-dimensional example shown in Figure C.1 above, a particle 'pt' lies in the x-y plane and at time t_{pt} and is located at (x_{pt}, y_{pt}), i.e. the left interface of the cell (i, j). For this example it is assumed that the velocity at the left interface v_{x1} is greater than the right interface velocity v_{x2}, and similarly velocity at the bottom interface v_{y1} is greater than the top interface velocity v_{y2}.

The first step is to determine the cell face across which the particle 'pt' will exit cell (i, j). For this, first the particle velocity needs to be calculated at the existing point (x_{pt}, y_{pt}), in each of the principal directions using equations (C.2a), (C.2b) and (C.2c). The velocity at each cell interface is also known from equations (C.1a), (C.1b) and (C.1c). Equation (C.10) can be re-arranged in order to determine the time that would be required for the particle to reach face x_2:

$$\Delta t_x = \frac{1}{\Psi_x} \cdot \ln\left(\frac{v_{x2}}{v_{x,pt}}\right), \tag{C.12}$$

where v_{x2} is the v_x at face x_2. Similar calculations are made to determine time required for particle 'pt' to reach faces y_1 and y_2. The time required for particle 'pt' to travel from point (x_{pt}, y_{pt}) to a boundary face of cell (i, j) is taken to be smaller positive value of Δt_x and Δt_y, and is denoted as Δt_{out}.

Note, in this example, as previously mentioned that the velocity v_{y2} is greater than v_{y1}, therefore the particle velocity in y-direction $v_{y,pt}$ will be a value between v_{y1} and v_{y2}. In order to travel to face y_1, the ratio in equation (C.12) $v_{y1}/v_{y,pt}$ will be less than unity yielding in a negative value for time. Similarly, if the velocity at the interface y_1 had been negative and the particle velocity in y-direction $v_{y,pt}$ had been positive, the ratio in equation (C.12) $v_{y1}/v_{y,pt}$ would have been negative yielding an error value for time. Therefore, in both the cases the particle cannot travel to an interface that yields negative or erroneous value.

In order to determine the exit coordinates (x_{out}, y_{out}), this value of Δt_{out} is substituted into equation (C.11a), (C.11b) and (C.11c) in order to yield the following equations:

$$x_{out} = x_1 + \frac{1}{\Psi_x}\left[v_{x,pt}\left(t_{pt}\right) \cdot e^{\Psi_x \cdot \Delta t_{out}} - v_{x1}\right], \tag{C.13a}$$

$$y_{out} = y_1 + \frac{1}{\Psi_y}\left[v_{y,pt}\left(t_{pt}\right) \cdot e^{\Psi_y \cdot \Delta t_{out}} - v_{y1}\right]. \tag{C.13b}$$

The time at which particle was at $\left(x_{pt}, y_{pt}\right)$ is given as t_{pt} and the time at which particle exits the cell is given as t_{out}, and the relation between the two is:

$$t_{out} = t_{pt} + \Delta t_{out}. \tag{C.14}$$

The following calculations are repeated until the particle either reaches a discharge point or exits the reservoir boundary. The following flow chart summarizes the particle tracking algorithm:

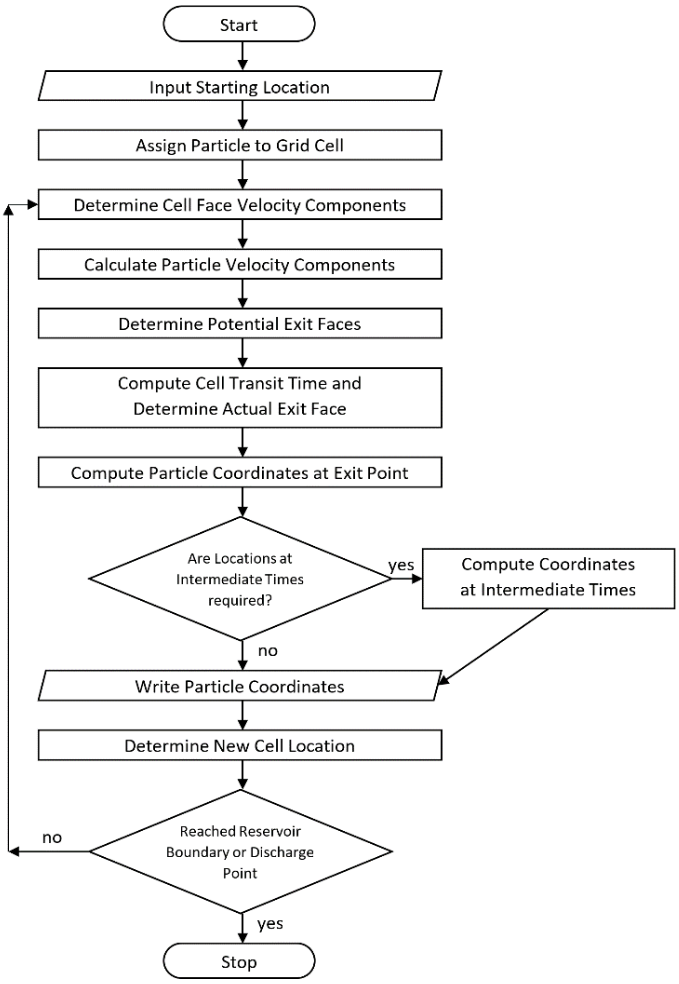

Start
Input Starting Location
Assign Particle to Grid Cell
Determine Cell Face Velocity Components
Calculate Particle Velocity Components
Determine Potential Exit Faces
Compute Cell Transit Time and Determine Actual Exit Face
Compute Particle Coordinates at Exit Point
Are Locations at Intermediate Times required?
yes
Compute Coordinates at Intermediate Times
no
Write Particle Coordinates
Determine New Cell Location
no
Reached Reservoir Boundary or Discharge Point
yes
Stop

Figure C.2 *– Flowchart for particle tracking algorithm for steady-state systems*

In case of transient problems, the velocity components at the cell interfaces change at each finite-difference time step, therefore the particle coordinates must also be computed at each of these time-steps.

C.1 Special Cases

C.1.1 Velocity components of opposite cell faces towards the cell

Sometimes, the velocity components of opposite cell faces are non-zero but are in the opposite directions; with the flow into the cell through both the opposite faces (Figure C.3). Therefore, once the particle enters the cell, it cannot leave, in that dimension. It can either leave through one of the cell faces in other dimensions, but if this situation prevails in all coordinate directions, this indicates that there is a strong sink present within the cell, hence no outflow can occur.

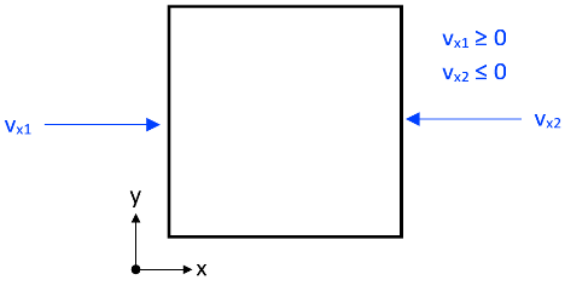


Figure C.3 *– Velocity components of opposite cell faces towards the cell (e.g. Producer well)*

C.1.2 Velocity components of opposite cell faces away from the cell

Similarly, if the velocity components of the opposite cell faces are non-zero, in the opposite direction, with the flow out of the cell faces, this implies that a local flow divide exists in that dimension within the cell (Figure C.4). Conversely, if this situation prevails in all coordinate directions, this indicates the presence of a strong source in the cell. For this situation, the potential exit face is determined by checking the sign of the v_{pt}. If v_{pt} is less than zero, the particle has the potential to leave the cell across left face (the face with negative velocity vector), while if v_{pt} is more than zero, the particle has potential to leave the cell across the right face. Once the potential exit face has been determined, the transit time can be found using equation (C.12).

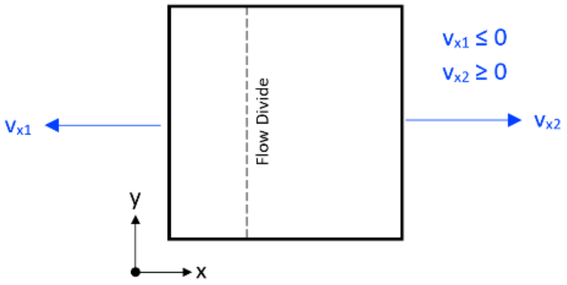


Figure C.4 *– Velocity components of opposite cell faces away from the cell (e.g. Injector well)*

C.1.3 Equal Velocities

In case, the velocities at face v_1 and v_2 are non-zero and equation, equation $(\text{C}.12)$ results in $[(1/0) \cdot \ln 1]$ that gives a mathematical error as it cannot be computed. In this case, the simple definition of velocity can be used:

$$\Delta t_x = (x_2 - x_{pt})/v_{x1} \qquad (v_{x1} > 0), \tag{C.15a}$$

$$\Delta t_x = (x_{pt} - x_1)/v_{x1} \qquad (v_{x1} < 0). \tag{C.15b}$$

Similar relations can be derived for other dimensions.

C.2 Treatment of Wells

In reservoir simulation, if a grid has well in it; whether producer well or injector well, it will be treated as point source or point sink. For a grid cell that contains a source or sink, the velocity field inside the cell cannot be considered as piecewise linear, so the streamlines emit from the surface of the injection well cell and conversely enter into the surface of the production well cell [25].

For the cells containing injector/producer well, the velocity vectors are calculated at the cell surface using Darcy's law with the neighboring cells. For the injector cell, the particles (starting point of streamlines) are equally distributed around the perimeter of the grid cell, and then the streamlines are traced forward towards the producer. The point at injection well cell surface is where the transit time equals zero, while the point at production well cell surface is the last point of streamline that records the transit time.

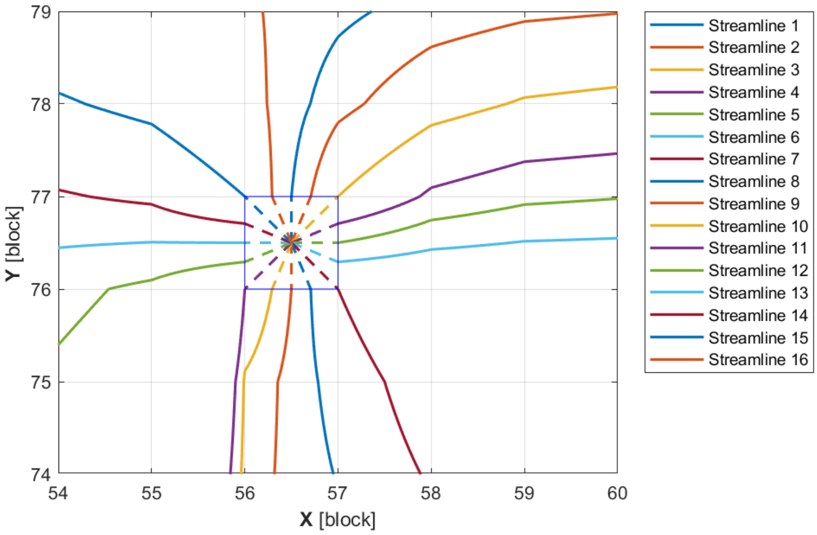


Figure C.5 *– Particle Distribution around Injector Well*

Figure C.5 gives an example of streamlines (solid lines) originating from the cell surface of an injector well. The streamlines are later manually extended to the center of the cell (dotted lines) only for the purpose to form a region.

D

Calculate Area of Irregular Polygon

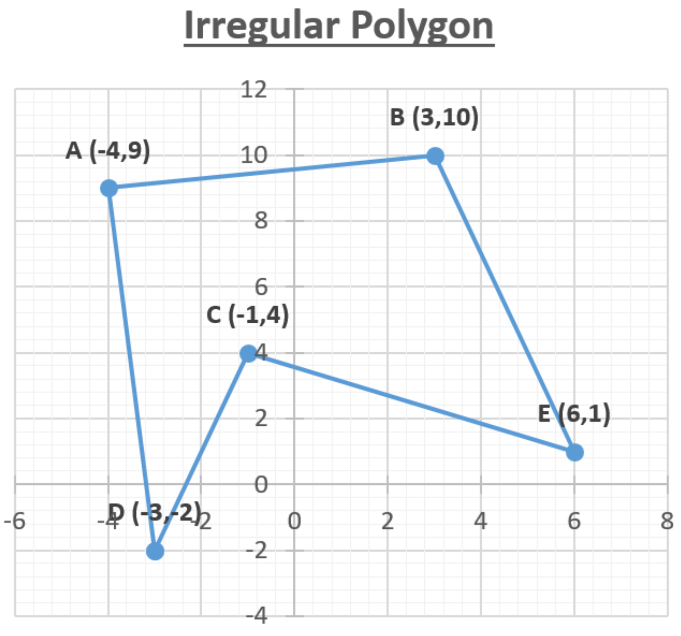


Figure D.1 – Irregular Polygon whose area needs to be determined

The area of an irregular polygon can be found if the coordinates of vertices are provided [39].

In order to determine the area of a polygon, firstly the coordinates of the vertices need to be written down.

Vertex	Coordinates
A	(-4,9)
B	(3,10)
C	(-1,4)
D	(-3,-2)
E	(6,1)

Table D.1 – Coordinates of Vertices

Next, the vertices of the polygon need to be sorted in a counterclockwise order. Then the x- and y-coordinates of each vertex need to be listed. And the coordinates of the first point need to be repeated at the bottom of the list.

Vertex	X Coordinate	Y Coordinate
D	-3	-2
C	-1	4
E	6	1
B	3	10
A	-4	9
D	-3	-2

Table D.2 *– Polygon vertices sorted counterclockwise and x- and y- coordinates listed separately*

Next, multiply the x-coordinate of each vertex by the y-coordinate of the next vertex and add these results.

$$Sum_1 = (-3 \times 4) + (-1 \times 1) + (6 \times 10) + (3 \times 9) + (-4 \times -2) = 82$$

Vertex	X Coordinate	Y Coordinate
D	-3	-2
C	-1	4
E	6	1
B	3	10
A	-4	9
D	-3	-2

Table D.3 *– Cross-multiplication of x- and y-coordinates*

Similarly, multiply the y-coordinate of each vertex by the x-coordinate of the next vertex and add these results.

$$Sum_2 = (-2 \times -1) + (4 \times 6) + (1 \times 3) + (10 \times -4) + (9 \times -3) = -38$$

Then subtract the sum of the second products from the sum of the first products:

$$Sum_1 - Sum_2 = 82 - (-38) = 120$$

Finally, divide this difference by 2 to get the area of the polygon.

$$\boldsymbol{Area} = 120/2 = 60$$

D.1 Calculate Volume of Cell in a Region

A cell with streamlines crossing it, lies in more than one region. If it has 'n' streamlines crossing it, then it lies in 'n+1' regions. The total volume of the cell is shared among these regions. This section devises a method to determine how much does this cell contribute to each region.

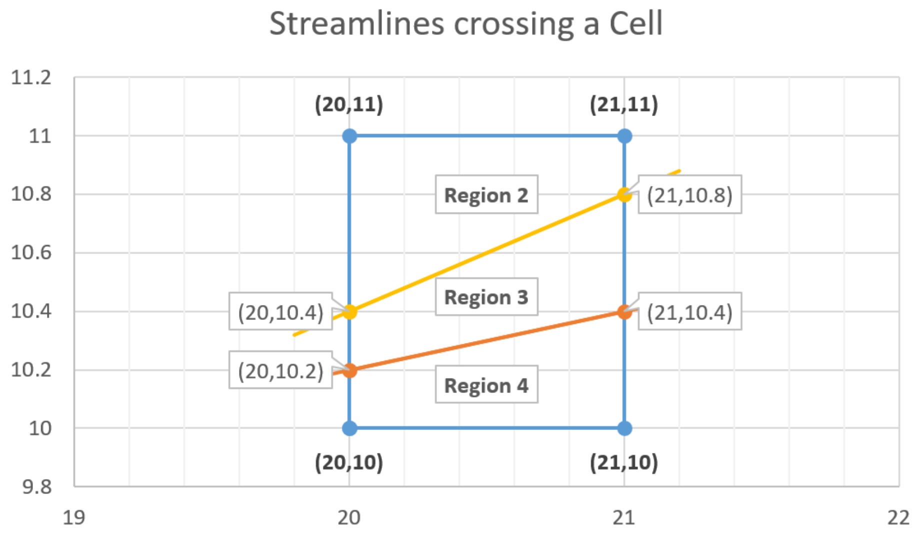


***Figure D.2** – Schematic of Streamlines Crossing a Cell*

The figure above shows a cell with streamlines crossing it. Firstly the coordinates of all the points where streamline intersects the cell face and all the cell vertices are listed down. The following table shows an example:

Point	Coordinates	Type
1	(20,10)	Cell Vertex
2	(20,10.2)	Streamline
3	(20,10.4)	Streamline
4	(20,11)	Cell Vertex
5	(21,10)	Cell Vertex
6	(21,10.4)	Streamline
7	(21,10.8)	Streamline
8	(21,11)	Cell Vertex

***Table D.4** – List of coordinates of all cell vertices and points where streamlines intersect cell faces*

Next, for each point, the region it lies in, is determined. If a point lies at the boundary of two regions, it will counted in both the regions (e.g. point 2 with coordinates (20,10.2) lies in regions 3 and 4). In MATLAB, this can be determined using the `INPOLYGON` function with the following syntax:

```
BOOLEAN  =  INPOLYGON(  point_x_coordinate  ,  point_y_coordinate  ,
region_x_coordinates , region_y_coordinates )
```

Each point is looped over each region to determine if it lies in the region. If the INPOLYGON function returns TRUE, it means that point lies in that region. The following table gives an example of summary of each point and the region it lies in:

Point	Coordinates	Region
1	(20,10)	4
2	(20,10.2)	3
2	(20,10.2)	4
3	(20,10.4)	2
3	(20,10.4)	3
4	(20,11)	2
5	(21,10)	4
6	(21,10.4)	3
6	(21,10.4)	4
7	(21,10.8)	2
7	(21,10.8)	3
8	(21,11)	2

***Table D.5** – Example summary of each point and the region it lies in*

Once the region(s) that each point lies in has been determined, the points are grouped according to region. The following tables show the points grouped by region:

Region 2	
Point	Coordinate
3	(20,10.4)
7	(21,10.8)
8	(21,11)
4	(20,11)

Region 3	
Point	Coordinates
2	(20,10.2)
6	(21,10.4)
7	(21,10.8)
3	(20,10.4)

Region 4	
Point	Coordinates
1	(20,10)
5	(21,10)
6	(21,10.4)
2	(20,10.2)

***Table D.6** – Points are grouped according to region and sorted counter-clockwise*

Next, for each region, the set of points are sorted counter-clockwise and the area is determined by the method mentioned in Appendix D.

D.1.1 Area of cell in Region 2:

$$A = \frac{1}{2} \times \Big[\big((20 \times 10.8) + (21 \times 11) + (21 \times 11) + (20 \times 10.4)\big) - \big((10.4 \times 21) + (10.8 \times 21) + (11 \times 20) + (11 \times 20)\big)\Big] = 0.4\, unit^2$$

D.1.2 Area of cell in Region 3:

$$A = \frac{1}{2} \times \Big[\big((20 \times 10.4) + (21 \times 10.8) + (21 \times 10.4) + (20 \times 10.2)\big) - \big((10.2 \times 21) + (10.4 \times 21) + (10.8 \times 20) + (10.4 \times 20)\big)\Big] = 0.3\, unit^2$$

D.1.3 Area of cell in Region 4:

$$A = \frac{1}{2} \times \Big[\big((20 \times 10) + (21 \times 10.4) + (21 \times 10.2) + (20 \times 10)\big) \\ - \big((10 \times 21) + (10 \times 21) + (10.4 \times 20) + (10.2 \times 20)\big)\Big] = 0.3\, unit^2$$

The depth is assumed to be ‘unity’, therefore the value of the volume is same as of the area. Next, the regions a cell lies in and the volume it contributes to each of these regions, is listed. The following table gives an example:

Cell Coordinates	Region	Volume
(21,11)	2	0.4
(21,11)	3	0.3
(21,11)	4	0.3

Table D.7 *– Example of all the regions a cell lies in and the volume it contributes to each*

Then, a similar ‘regions and volumes’ list is generated for each cell in the grid. Finally, this list is re-grouped by regions, rather than the existing group that is by cell number. For each region, the cells that lie in it and the volume each cell contributes to this region, are determined. The cell coordinates are converted into cell indexes for single value referencing. And lastly, for any positive value of volume, the cell is considered to be active in that region. The following table gives an example of this *actnum_volume* list:

Cell Index	Active Cell	Volume
56696	1	0.12500
56827	1	0.00004
56828	1	0.12622
56936	1	0.49308
56937	1	0.51460
56938	1	0.00080
57066	1	0.00346
57067	1	0.53788
57068	1	0.48986
57176	1	0.46303
57177	1	1.00000

Table D.8 *– Example of Active Cell / Cell Volume List*

E

Center-line Methods and Drawbacks

Drawing center-lines between two arbitrary lines, as easy as it sounds, was the most difficult part of this project. Several methods were used but no method was entirely accurate and had its own downsides. Five of the several methods to draw center-lines are listed as follows:

E.1 Center-line by average at x- (or y-)

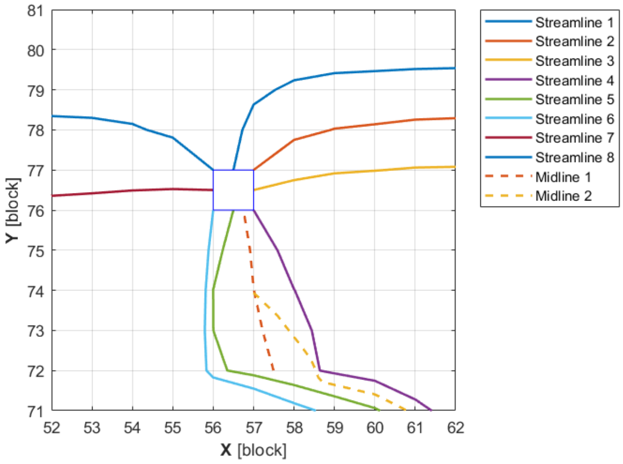


Figure E.1 *– Center-line by average at x-coordinate or y-coordinate*

The red dotted lines is the center-line that is drawn by averaging the x-coordinates of streamlines 5 and 6 at common y-coordinates. Similarly, the yellow dotted line is the center-line that is drawn by averaging the y-coordinates of streamlines 5 and 6 at a common x-coordinates. But as can be seen from the figure, between $x = 56$ and $x = 57$, there exists only streamline 6 while streamline 5 is non-existent, therefore the center-line cannot be drawn for this interval.

Although, an algorithm can be made such that it draws center-line by averaging at common x-coordinates where x-coordinate exists for both streamlines, and then draw a center-line in the remaining intervals by averaging at common y-coordinates. But as can be seen from the figure that these two center-lines are not continuous, there is a discontinuity between the two center-lines (e.g. center-line based on common 'x' terminates at $(57,74)$ while the center-line based on common 'y' terminates at $(57.5,72)$). Therefore these two methods do not yield accurate results and cannot be used to draw center-lines.

E.2 Center-line by drawing normal

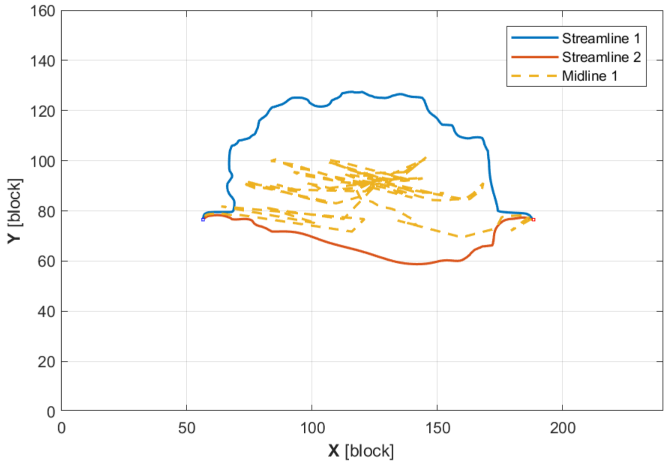


__Figure E.2__ – Center-line by drawing line normal at the point and averaging the x- and y- coordinates

For this method, the longer streamline is selected (i.e. streamline 1 in this case). Next for each point on the streamline (i.e. point 1), a line normal to that point is drawn and extended such that it intersects the other streamline (e.g. streamline 2 in this case). This point of intersection can be called point 2. The x- and y- coordinates of the two points (point 1 and point 2) are averaged and this point is known as the center-point. For each point on streamline 1, the center-point is determined and a line is drawn through them. The result is the yellow dotted line in Figure E.2, which is completely inaccurate. The reason for this is that streamlines are not smooth, and there are several bumps on streamlines. On each bump, when going up, the normal line may have a negative gradient, while coming down the bump the line normal to the point can have a positive gradient, therefore the gradient changes rapidly and this can occur very frequently. This yields a center-line with a bizarre pattern (as can be seen in the figure), therefore this method cannot be used to determine the center-line.

E.3 Center-line by minimum distance between two lines

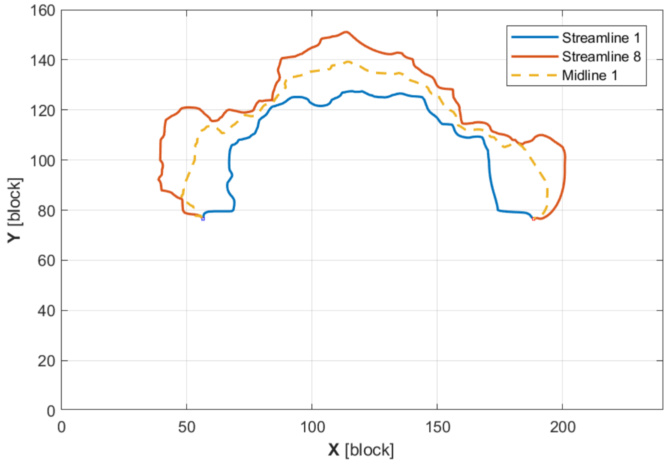


__Figure E.3__ – Center-line by minimum distance between the two streamlines

For this method, the longer streamline is selected (i.e. streamline 8 in this case). Next for each point on the streamline (i.e. point 1), a line is drawn such that has the shortest distance to the other streamline (i.e. streamline 1 in this case). The point at which this shortest line intersects is point 2. The x-coordinates of the two points (point 1 and point 2) are averaged, and similarly the y-coordinates of the two points are averaged and this point is known as the center-point. The center-point is determined for each point on streamline 1 and a line is drawn through them. The result is the yellow dotted line in Figure E.3, which is reasonably quite accurate. But there may be cases where the shortest distance to the next streamline does not correspond to the appropriate point on the streamline (e.g. at $x = 60$ or $x = 170$) and yields an erroneous point on the center-line thus making this method inaccurate as well.

E.4 Center-line by distance travelled

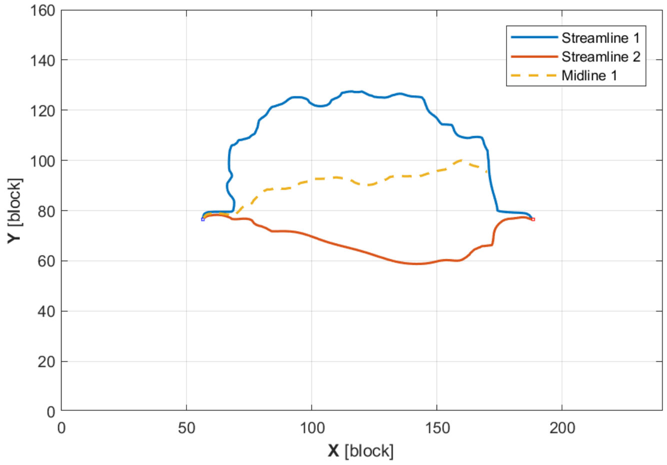


Figure E.4 *– Center-line by distance travelled by the two streamlines*

This method uses the distance travelled by each particle (e.g. 1 m displacement on streamline 1 and same displacement on streamline 2), and then the x- and y- coordinates of the two points on the two streamlines are averaged. This step is repeated after every fixed interval and finally a line is drawn through all these center-points. But the draw-back for this algorithm is that it works only until the length of the shorter streamline. For example, if streamline 1 has a total length of 80 m and streamline 2 has a total length of 100 m, this algorithm will only work until 80 m. For the remaining 20 m of the longer streamline, the algorithm does not have point on streamline 1 to average it with. Therefore, this method for drawing center-line is inaccurate as well.

E.5 Center-line by percent of path travelled

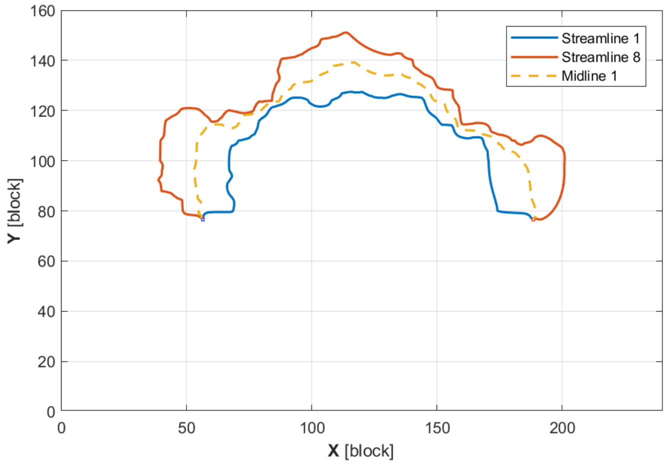


Figure E.5 *– Center-line by percent of path travelled*

This method uses ratio (or percent) of path travelled by each particle and then it averages the x-and y-coordinates of the two points on the two streamlines (e.g. if streamline 1 has a total length of 80 m and streamline 2 has a total length of 100 m, then 1% of path travelled for particle 1 is 0.8 m and particle 2 is 1 m respectively.) The x- and y-coordinates at these two particle are recorded. Then the x-coordinates of the two particles are averaged, and similarly the y-coordinates of the two particles are averaged and the center-point is obtained. This is repeated for each percent and a line is drawn through all these center-points, consequently obtaining a center-line. This is by far the best method to determine the center-line. Although this method also has its draw-backs as well because a streamline (particularly the longer streamline) can have bumps on it (e.g. bumps at $(50{,}120)$ and $(110{,}190)$ on streamline 8). These bumps not only cause the streamline to become lengthier but also cause the particle to lag behind compared to the particle on the shorter streamline (i.e. streamline 1). Therefore the point on one streamline may not correspond to the appropriate point on other streamline yielding an erroneous point. For example, 20% of path on streamline 1 roughly corresponds to coordinates (75,110) while the same 20% on streamline 8 approximates to coordinates $(50{,}120)$ that clearly shows will yield an erroneous point. Despite the slight weakness of this method, it has given best results so far to trace most of the center-lines.

F

Transmissibility Correction

Once the streamlines have been traced and the active cell and cell volume lists for each region have been exported into DARTS, the simulation is run for each of the regions. Next, the pressure and temperature distribution results are merged based on the weighted average of the volume that region contributes to the volume of the entire cell. The figure below shows the comparison between the temperature distribution of the full-scale model and proxy model. Most of the parameters used in simulation are from Table 4.1 and the permeability/porosity distribution used is that of homogeneous reservoir (as in Table 4.2):

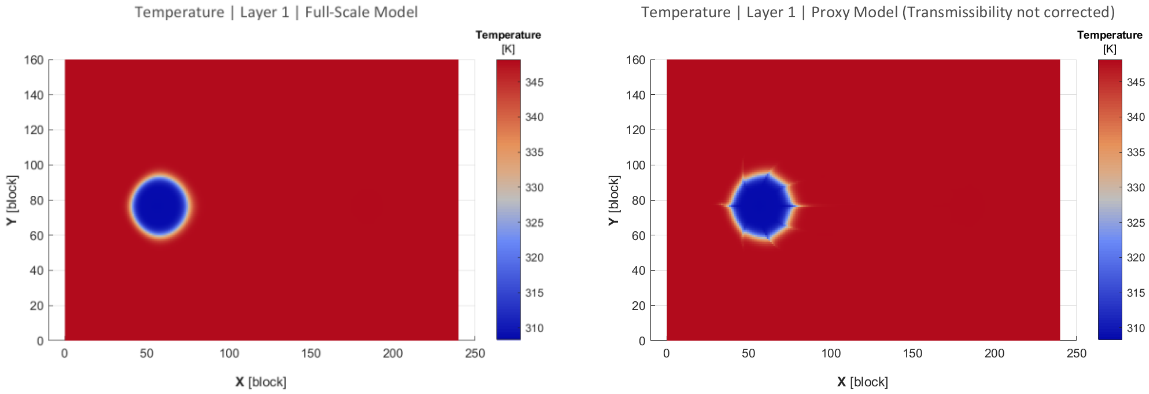


***Figure F.1** – Comparison between Temperature distributions of Full-Scale Model (left) and merged Proxy Model (transmissibility not corrected) (right)*

The temperature distribution of the proxy model does not exactly match the distribution of the full-scale model (as shown in Figure F.1). The temperature distribution has 'spikes', particularly near the interface of the regions (i.e. streamlines).

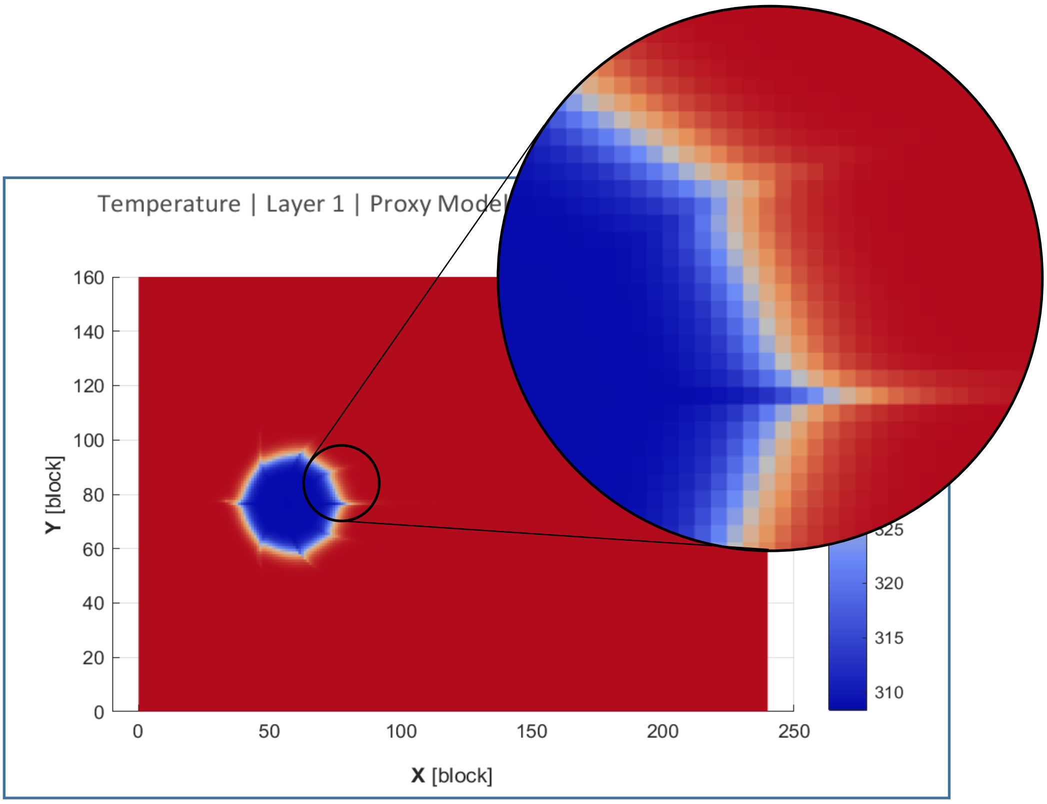


Figure F.2 – Error in Proxy Model due to incorrect transmissibilities

The lens in Figure F.2 zooms on region 1 (between streamline 1 and 2), near the injector well. Investigating into the reason of the 'spikes' reveals that these exist due to the fact that transmissibility is not corrected for cells that have streamlines crossing it.

Before correcting the transmissibility for such cells, it is crucial to understand why it is important to do such. As shown in Figure F.3 below, if a pair of cells has a streamline passing through them, then these cells lie in two regions. For a pair of cells, they have a certain value of transmissibility between them. If this pair is active in both regions, they will have the same transmissibility in the connection lists of both the regions. Whereas it can be viewed in Figure 4, that the streamline is cutting the interface to a fraction ('half' in this case). If the transmissibility is not being corrected, this means that the flow is occurring twice as much as it should be, between the two cells due to the following expression:

$$Q_v = \Gamma(p_i - p_{i+1}). \qquad \text{(F.1)}$$

Therefore, this is the reason why in Figure F.1, the temperature front in much advance in cells that have streamlines crossing them. Therefore the transmissibility in both the regions needs to be corrected by multiplying it with a fraction.

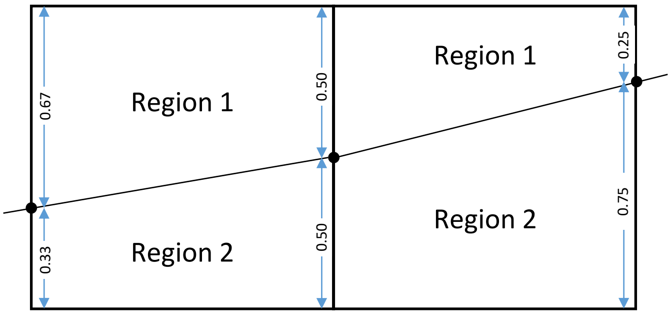


Figure F.3 *– Schematic of a streamline crossing a pair of cells and its effect on transmissibility*

This 'transmissibility multiplier' is also calculated in MATLAB, and exported into DARTS along with 'active cell and volume' list. The following table shows a few entries from the transmissibility list of 'region 1' as an example of the data exported.

Cell	Neighboring Cell	Transmissibility Multiplier
0	3	0.500
1	2	0.004
1	7	0.017
2	1	0.004
2	8	0.500
3	0	0.500
3	4	1.000
3	9	0.481
4	3	1.000
4	5	60.029
5	4	0.029

Table F.1 *– Example of Transmissibility multiplier list*

Once the transmissibility is corrected, the simulation for the proxy model is re-run and the following temperature distributions are obtained.

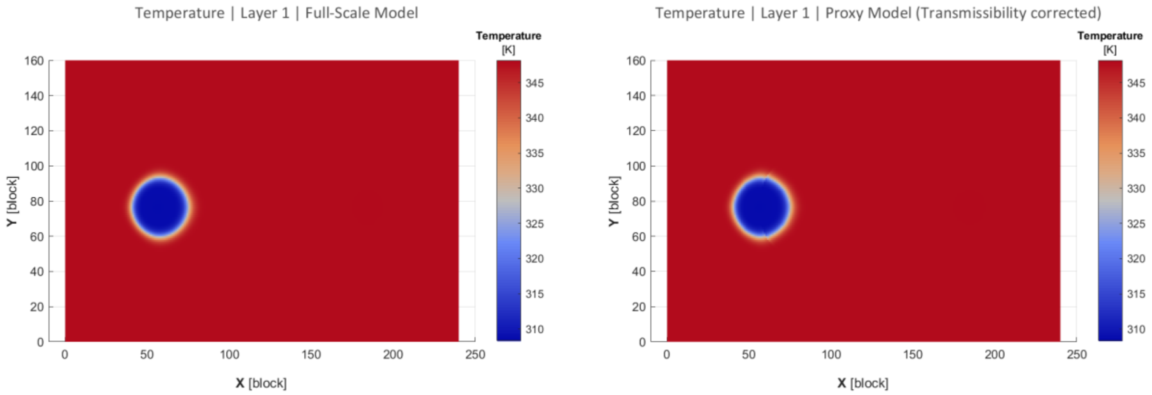


Figure F.4 *– Comparison between Temperature distributions of Full-Scale Model (left) and merged Proxy Model (transmissibility corrected) (right)*

Comparing Figure F.4 to Figure F.1, the temperature distribution obtained from the proxy-model after correcting the transmissibility is considerably similar to that of the full-scale model (as compared temperature distribution of proxy model before correcting transmissibility).

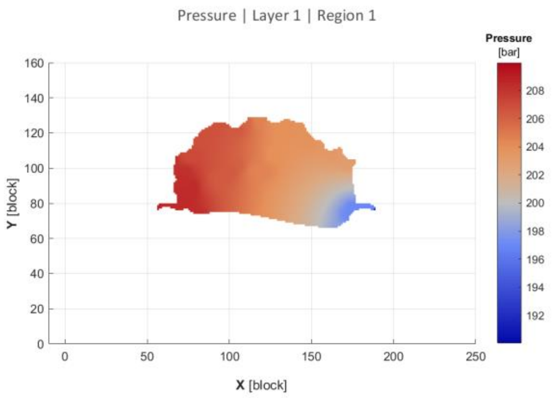
G

Results for each Region of Proxy Model

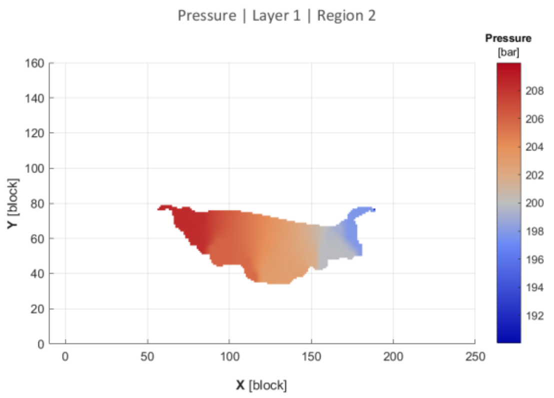
This section of the Appendix displays the Pressure and Temperature distribution of each region of the proxy model and a comparison of the merged proxy model results with the full-scale model results. Most of the parameters used to run the full-scale model and proxy model simulations are from Table 4.1, and the permeability and porosity distribution used is from the heterogeneous reservoir as illustrated in Figure 4.7.

G.1 Pressure Distribution for each region of Proxy Model

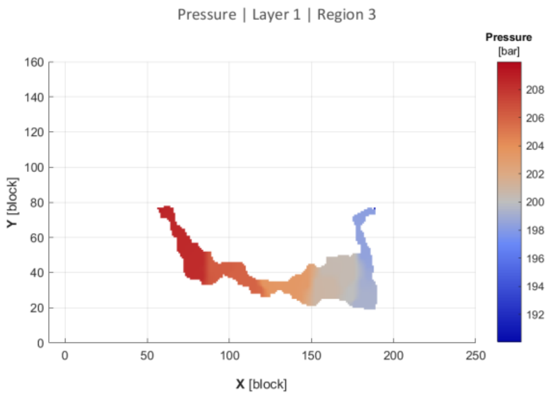
The following figures summarize the pressure distribution for each region of the proxy model:



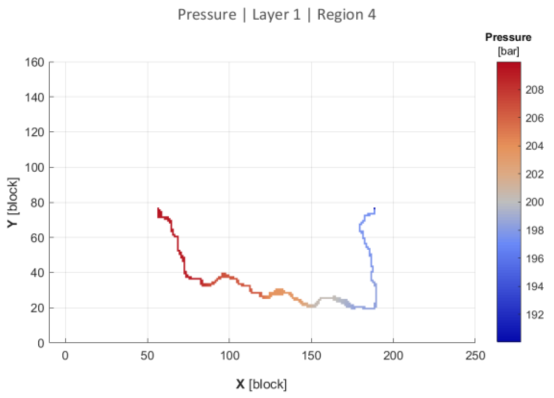
Region 1



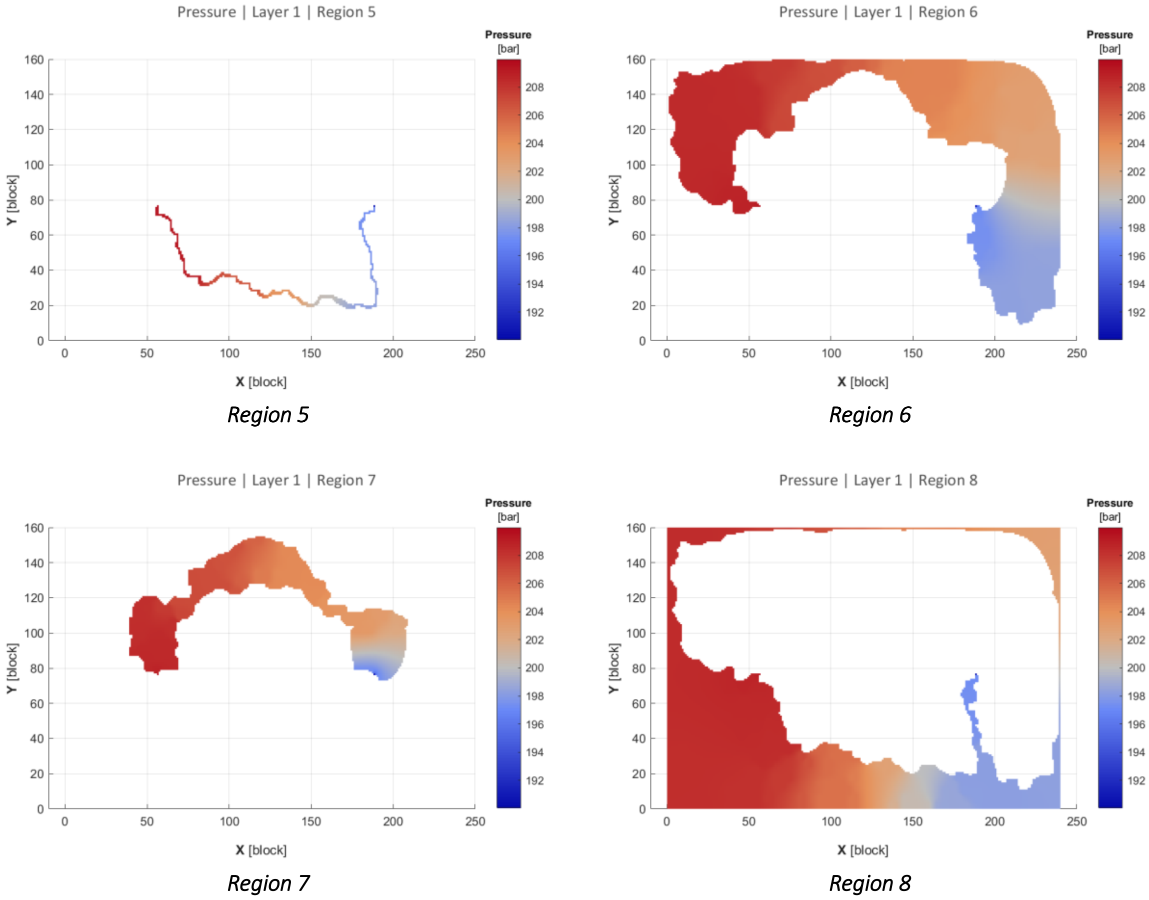
Region 2



Region 3



Region 4



Region 5

Region 6

Region 7

Region 8

Figure G.1 *– Pressure Distribution for each region of the Proxy Model of Heterogeneous Reservoir Layer 1 run to t = 200,000 days*

G.2 Comparison between Pressure distributions

The following figures compare the pressure distributions of full-scale model and (merged) proxy model:

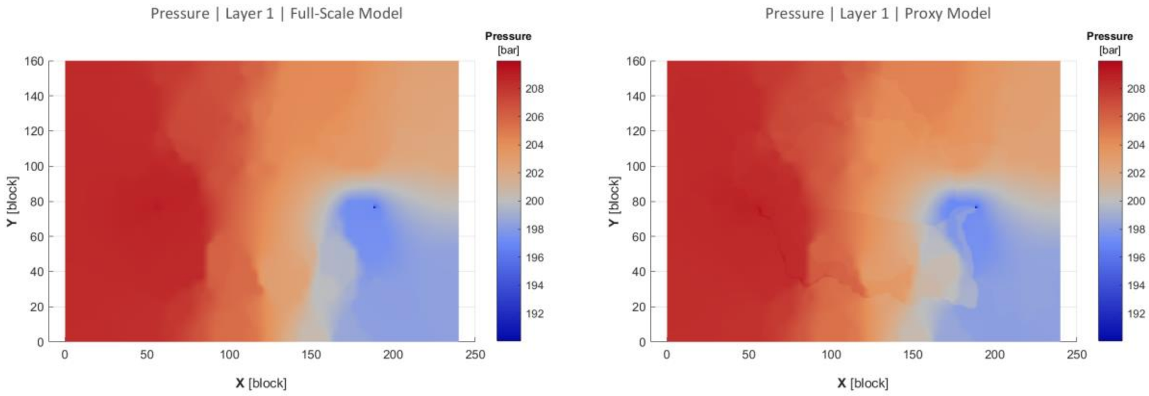
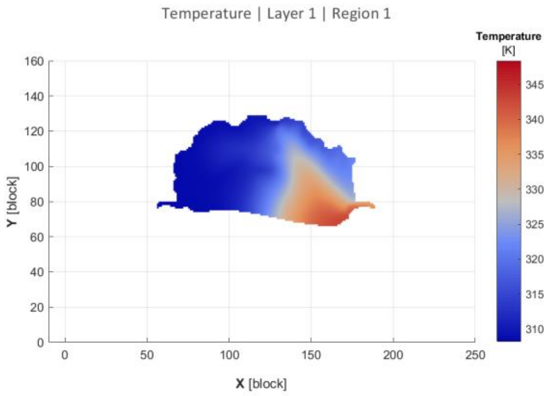


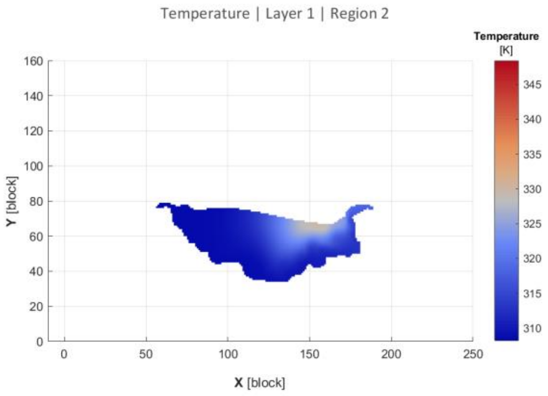
Figure G.2 *– Comparison between Pressure distributions of Full-Scale Model and Proxy Model*

G.3 Temperature Distribution for each region of Proxy Model

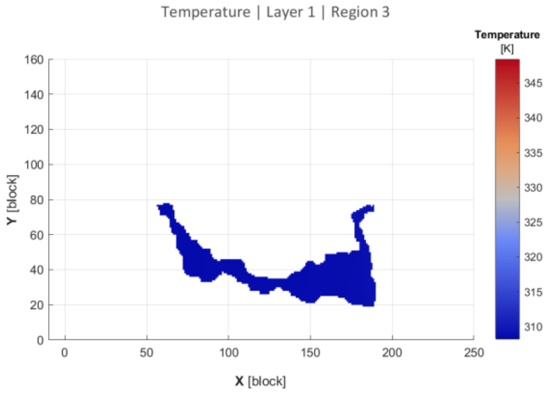
The following figures summarize the temperature distribution for each region of the proxy model:



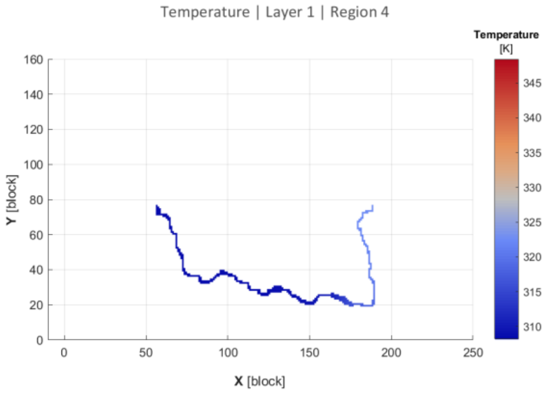
Region 1



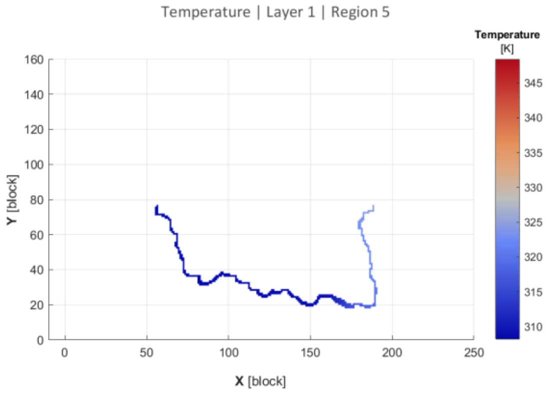
Region 2



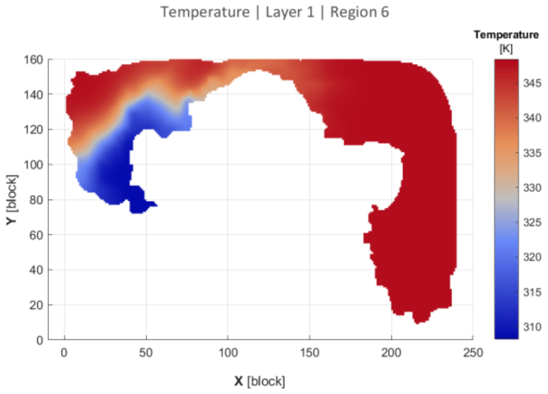
Region 3



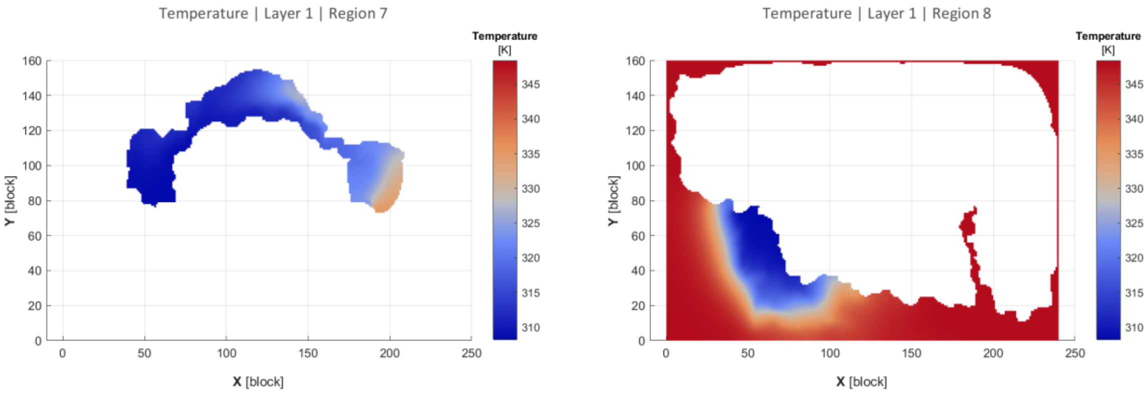
Region 4



Region 5



Region 6



Region 7 *Region 8*

Figure G.3 *– Temperature Distribution for each region of the Proxy Model of Heterogeneous Reservoir Layer 1 run to t = 200,000 days*

G.4 Comparison between Temperature distributions

The following figures compare the temperature distributions of full-scale model and (merged) proxy model:

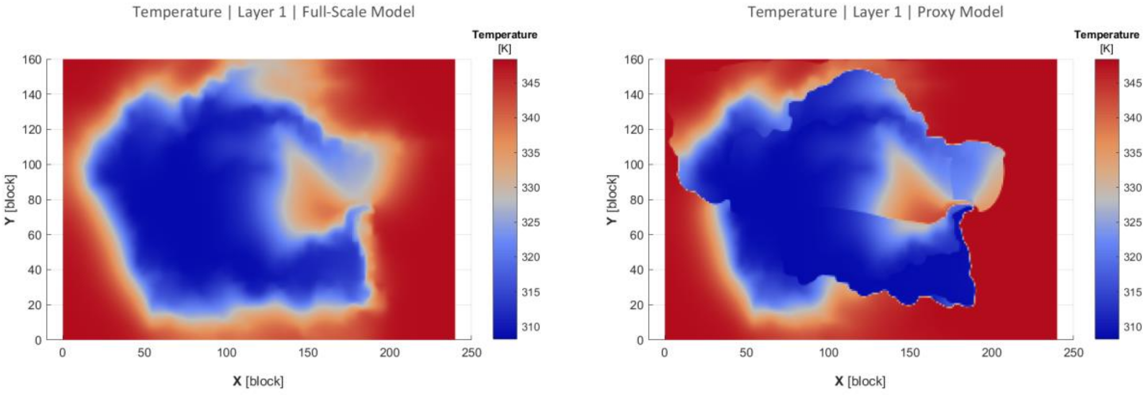


Figure G.4 *– Comparison between Temperature distributions of Full-Scale Model and Proxy Model*

Bibliography

[1] S. N. Shetty, D. V. Voskov and D. F. Bruhn, "Numerical Strategy for Uncertainty Quantification in Low Enthalpy Geothermal Projects," M.Sc. Thesis, Department of Applied Earth Sciences, T.U. Delft, Delft, 2017. [Accessed 8 August 2019]. [Online]. Available: http://resolver.tudelft.nl/uuid:10fe0b97-744f-4b3c-a1c9-f619456109a0

[2] D. W. Pollock, "Semianalytical computation of path lines for finite-difference models," *Groundwater,* vol. 26, no. 6, pp. 743-750, November 1988.

[3] V. K. Koech, "Numerical Geothermal Reservoir Modelling and Infield Reinjection Design, Constrained by Tracer Test Data: Case Study for the Olkaria Geothermal Field in Kenya," Reykjavík, Iceland, 2014.

[4] J. W. J. Mercer, C. Faust and G. F. Pinder, "Geothermal Reservoir Simulation," in *Conference on Research for the Development of Geothermal Energy Resources*, Pasadena, California, September 23-25, 1974.

[5] PetroWiki, "Modeling geothermal reservoirs," 16 July 2015. [Online]. Available: https://petrowiki.org/Modeling_geothermal_reservoirs. [Accessed 8 August 2019].

[6] M. Khait and D. V. Voskov, "Operator-based linearization for efficient modeling of geothermal processes," *Geothermics,* vol. 74, pp. 7-18, 13 February 2018.

[7] Schlumberger, "Geothermal Services," [Online]. Available: https://www.slb.com/services/additional/geothermal/technical_consulting/reservoir_modeling_simulation.aspx. [Accessed 8 August 2019].

[8] S. Garg, *Geothermal Reservoir Engineering [PowerPoint slides],* 2013.

[9] H. M. Nick, *Geothermal Reservoir Engineering [PowerPoint slides],* Delft, 2018.

[10] D. R. Boden, Geologic Fundamentals of Geothermal Energy, 1st ed., CRC Press, 2016.

[11] M. A. Grant and P. F. Bixley, Geothermal Reservoir Engineering, 2nd ed., Academic Press, 2011.

[12] R. DiPippo, Geothermal Power Generation: Developments and Innovation, Woodhead Publishing, 2016.

[13] W. E. Glassley, Geothermal Energy: Renewable Energy and the Environment, 1st ed., Boca Raton, Florida: CRC Press, June 22, 2010.

[14] J. W. Lund, L. Bjelm, G. Bloomquist and A. K. Mortensen, "Characteristics, development and utilization of geothermal resources - a Nordic perspective," *Episodes,* vol. 31, no. 1, pp. 140-147, March 2008.

[15] L. Rybach, "Geothermal Sustainability," *Geo-Heat Centre Quarterly Bulletin,* vol. 28, pp. 2-7, September 2007.

[16] R. Cataldi, "Review of historiographic aspects of geothermal energy in the Mediterranean and Mesoamerican areas prior to the Modern Age," *Geo-Heat Center Quarterly Bulletin,* vol. 15, pp. 13-16, 1 August 1993.

[17] International Geothermal Association, "Beyond Electricity - Geothermal Energy Development Is Picking Up," 21 February 2019. [Online]. Available: https://www.geothermal-energy.org/beyond-electricity-geothermal-energy-development-is-picking-up/. [Accessed 8 August 2019].

[18] I. B. Fridleifsson, R. Bertani, E. Huenges, J. W. Lund, A. Ragnarsson, L. Rybach and others, "The possible role and contribution of geothermal energy to the mitigation of climate change," in *IPCC Scoping Meeting on Renewable Energy Sources*, Lübeck, Germany, January 20-25, 2008.

[19] E. Okandan, Geothermal reservoir engineering, 1st ed., vol. 150, Ankara: Kluwer Academic Publishers, 1988.

[20] S. Saeid and M. A. Hicks, "Experimental and Numerical Study of Heat Flow under Low-Enthalpy Hydrothermal Conditions," Ph.D. Dissertation, Department of Geo-Engineering, T.U. Delft, Delft, 2015. [Accessed 8 August 2019]. [Online]. Available: https://doi.org/10.4233/uuid:63aea8ee-420c-4cc0-9ec7-4ffe089352f5

[21] C. J. Willems, "Doublet deployment strategies for geothermal Hot Sedimentary Aquifer exploitation: Application to the Lower Cretaceous Nieuwerkerk Formation in the West Netherlands Basin," Ph.D. Dissertation, Department of Applied Earth Sciences, T.U. Delft, Delft, 2017. [Accessed 8 August 2019]. Available: https://doi.org/10.4233/uuid:2149da75-ca29-4804-8672-549efb004048

[22] A. Datta-Gupta and M. J. King, Streamline Simulation: Theory and Practice, vol. 11, Richardson, Texas: Society of Petroleum Engineers, 2007.

[23] Princeton University, "Streamlines and Streamtubes," [Online]. Available: https://www.princeton.edu/~asmits/Bicycle_web/streamline.html. [Accessed 07 August 2019].

[24] Aerospace, Mechanical & Mechatronic Engg., University of Sydney, "Flow Description, Streamline, Pathline, Streakline and Timeline," 2005. [Online]. Available: http://www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/cvanalysis/node8.html. [Accessed 08 August 2019].

[25] Y. Jun and W. Minglu, "Streamline Numerical Simulation Theory and Method," in *Streamline Numerical Well Test Interpretation*, Gulf Professional Publishing, 2011, pp. 17-42.

[26] Flow Science Inc., "Grid Systems | CFD-101 | What Are the Advantages and Disadvantages?," [Online]. Available: https://www.flow3d.com/resources/cfd-101/general-cfd/grid-systems/. [Accessed 1 September 2019].

[27] "Structured and unstructured mesh," 23 February 2011. [Online]. Available: https://imechanica.org/node/9864. [Accessed 1 September 2019].

[28] H. R. Hiester, M. D. Piggott, P. E. Farrell and P. A. Allison, "Assessment of spurious mixing in adaptive mesh simulations of the two-dimensional lock-exchange," *Ocean Modelling,* vol. 73, pp. 30-44, 2014.

[29] "What is the difference between structured and unstructured mesh types in a FVM software?," 16 August 2017. [Online]. Available: https://www.researchgate.net/post/What_is_the_difference_between_structured_and_unstructured_mesh_types_in_a_FVM_software. [Accessed 1 September 2019].

[30] J. Tu, G.-H. Yeoh and C. Liu, Computational Fluid Dynamics - A Practical Approach, 3rd ed., Cambridge: Butterworth-Heinemann (Elsevier), 2019.

[31] Envenio, "Top 6 Reasons to Choose Structured Grids in CFD," 15 November 2005. [Online]. Available: https://www.design-engineering.com/cfd-automeshing-1004028397-1004028397/. [Accessed 1 September 2019].

[32] "When should we use a Structured grid or an unstructured grid in model simulation?," 13 December 2014. [Online]. Available: https://www.researchgate.net/post/When_should_we_use_a_Structured_grid_or_an_unstructured_grid_in_model_simulation. [Accessed 1 September 2019].

[33] D. J. Mavriplis, "Unstructured Grid Techniques," *Annual Review of Fluid Mechanics,* vol. 29, no. 1, pp. 473-514, 1 January 1997.

[34] J. F. Thompson, B. K. Soni and N. P. Weatherill, Handbook of Grid Generation, 1st ed., Boca Raton, Florida: CRC Press, 1998.

[35] M. Khait and D. V. Voskov, "Operator-based linearization for general purpose reservoir simulation," *Journal of Petroleum Science and Engineering,* vol. 157, pp. 990-998, 5 August 2017.

[36] M. R. Thiele, "Streamline Simulation," in *6th International Forum on Reservoir Simulation*, Schloss Fuschl, Austria, September 3rd-7th, 2001.

[37] F. P. Incropera, T. L. Bergman, D. P. DeWitt and A. S. Lavine, Fundamentals of Heat and Mass Transfer, 7th ed., Jefferson City, MO: Wiley, 2011.

[38] R. S. Subramanian, "Conduction in the Cylindrical Geometry," [Online]. Available: https://web2.clarkson.edu/projects/subramanian/ch330/notes/Conduction%20in%20the%20Cylindrical%20Geometry.pdf. [Accessed 8 August 2019].

[39] Math Open Reference, "Area of a polygon (Coordinate Geometry)," [Online]. Available: https://www.mathopenref.com/coordpolygonarea.html. [Accessed 08 August 2019].

[40] IRENA, "Geothermal Power: Technology Brief," International Renewable Energy Agency, Abu Dhabi, September, 2017.