NTNU

Norwegian University of
Science and Technology

Machine-learning algorithms for the computation of upscaled permeabilities

Roman Bohne

Master of Science in Physics and Mathematics
Submission date: May 2018
Supervisor: Knut Andreas Lie, IMF
Co-supervisor: Xavier Raynaud, SINTEF IKT

Norwegian University of Science and Technology
Department of Mathematical Sciences

Abstract

In full-scale reservoir simulation models, the characteristics of the rock are not fully resolved. Upscaled permeabilities are used to account for the effective behaviour of a composite region. By averaging the fine properties of the flow, they provide a simple linear correlation between the flux and the pressure drop. The goal of this master project is to assess the performance of machine-learning algorithms for the computation of upscaled permeabilities. The methodology is to use high resolution simulations on a randomly generated set of fine scale models. This data set will be used as training set for a machine learning algorithm. Ordinary least squares and Kernel Ridge regression algorithms will be the two different machine learning algorithms that will be tested and compared. We will also investigate the robustness of the algorithms with respect to the choice of the statistical distributions used for the generation of the fine scale models. Our results show that both the ordinary least squares and Kernel Ridge is capable to capture the upscaled behaviour of the flow and encode it into a single coefficient, namely the upscaled permeability. Hence they will capture the underlying physics of the problem. However, Ordinary least squares does so with a high error that is likely to limit the usefulness of that particular algorithm.

Sammendrag

I fullskala reservoarsimuleringsmodeller kan man ikke bevare alle karakteristiske egenskaper av et berg fult ut. Oppskalere permeabiliteter brukes til å regne ut effektiv oppførsel av en sammensatt region. Ved å beregne flyt får man en lineær sammenheng mellom flux og trykkfall i en slik region. Målet med dette masterprosjektet er å vurdere ytelsen til maskinlæringsalgoritmer for beregning av oppskalere permeabiliteter. Metodikk er å bruke høyoppløselige simuleringer på et tilfeldig generert sett av fine skala modeller. Dette datasettet vil deretter bli brukt som treningssett for en maskinlæringsalgoritme. De to regresjonsalgoritmene Ordinary least squares og Kernel Ridge vil bli testet og sammenlignet. Vi vil også undersøke robustheten av algoritmene med hensyn til valg av sannsynlighetsfordeling som brukes til generering av finskala modeller. Våre resultater viser at både Ordinary least squares og Kernel Ridge klarer å fange opp den oppskalerte oppførselen av flyt og lagre dette i en koefficient, den oppscalerte permeabiliteten. Med andre ord, så klarer de å fange opp fysikken. Desverre så gjør Ordinary least squares dette med en rimelig stor feilmargin noe som sansynligvis vil begrense nytten av den algoritmen.

Preface

This document is my thesis for the Master's degree program Industrial Mathematics at the Norwegian University of Science and Technology (NTNU), in the field of numerical mathematics. It is a joint project with NTNU and the independent research organization SINTEF, carried out in the period of November 2017 - May 2018.

First and foremost I would like to thank my supervisors at SINTEF, Xavier Raynaud and Knut-Andreas Lie for all the helpful meetings, close supervision and a general understanding of my problems. Without you I would never have pulled through with the thesis work.

I would also like to thank Sølvi Bente Sønvisen from the IME faculty administration.

Lastly, a big thank you to my family and friends for supporting me through my several years of study.

Roman Bohne
Oslo
18th May 2018

Contents

Chapter 1

Introduction

1.1 History and Importance of Petroleum

From the early cradle of civilization in the Indus Valley of the ancient times known as the bronze age (3300-1300 BCE) the green-brown-black flammable viscous liquid known as Petroleum was used as an adhesive with good waterproofing properties in construction [14]. Since then, early uses has been a variety of other purposes like early medicine and tarring of ship ropes. However the true value of the liquid was concealed until 1848 when James Young was called to find alternative use of a petroleum seepage in the Riddings colliery at Alfreton, Derbyshire England. With his knowledge of early industrial era chemistry he figured out that he could distil the liquid into light thin oil suitable as lamp oil and at the same time obtain a much thicker oil suitable as machine grease. The invention was a huge success and he ended up creating Young's Paraffin Light and Mineral Oil Company witch grew and expanded its operations, selling paraffin lamps all over the world. This is however only the start of the rise to importance of petroleum. Later inventions like the internal combustion engine, the rise in commercial aviation and the hundreds of different uses of petroleum in organic chemistry, particularly the synthesis of plastics, fertilizers, solvents, adhesives and pesticides sky-rocketed a whole new industry.

Today the petroleum industry, including the processes of exploration, extraction, refining, transporting and marketing of petroleum products, taken as a whole represents the world's largest industry in terms of dollar value [8] [18]. Petroleum is vital to many industries, and is of importance to the maintenance of industrial civilization in its current configuration, and thus is a critical concern for many nations.

Petroleum also accounts for a large percentage of the worlds energy consumption. Since the early 1970's Norway have been extracting petroleum from the continental shelf in their sector in the north sea and has been a major source of the country's wealth [12].

1.2 Petroleum Extraction

Petroleum is derived from ancient organic materials that were covered by stagnant water or sediments faster than they could decompose aerobically. Then through geological processes over millions of years being pushed deep underground. Down there the pressure and temperature is high enough that over said millions of years the organic matter cooks into petroleum. The petroleum created through this process have a lower density compared to other fluids and will slowly rise towards the earth's surface where it dissipates. Sometimes however it meets a geological formation where it gets trapped. Such a formation, called a oil reservoir, consists of a layer of porous rock where the petroleum can reside, that is topped by a denser rock that it cannot pass through as illustrated in Figure 1.1.

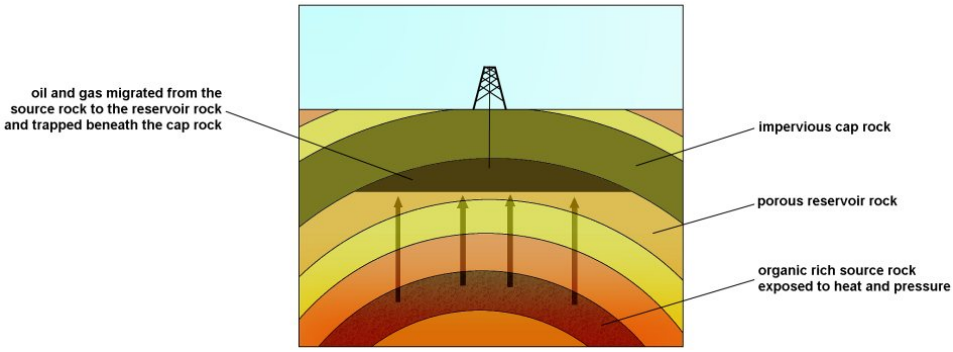

Figure 1.1 – Illustration of Oil getting trapped below impervious rock. Figure taken from [19]

One of the keystones in profitable petroleum extraction is proper estimation of reserves, making decisions regarding the development of a field, predicting future production, placing additional wells and evaluating alternative management scenarios. The main tool used in this estimation is reservoir modelling involving the construction of a computer model of a petroleum reservoir. These models are huge in scale as a reservoir itself can span several square kilometre in the horizontal plane. Hence, it is often not feasible to do micro and macro simulations on the same model. The purpose of the presented work in this thesis is looking at the performance of machine-learning algorithms for the computation of **upscaled** petrophysical properties. **Upscaling** refers to the process of propagating proper-

ties and parameters from a model of high spatial resolution to a model of lower spatial resolution as illustrated in Figure 1.2.

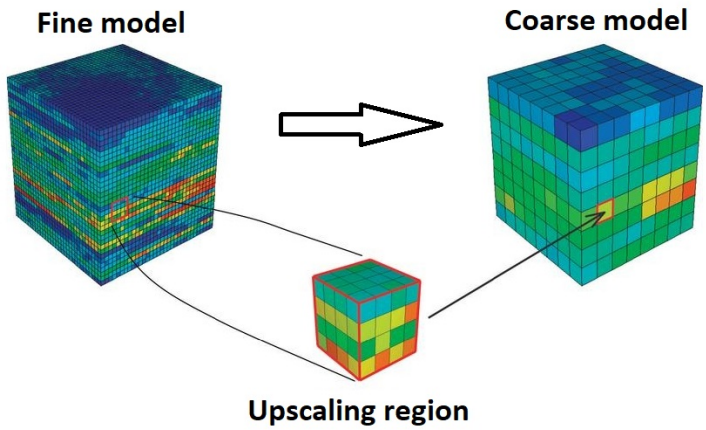

Figure 1.2 – Upscaling of reservoir models. Figure taken from [17]

1.3 Motivation for Machine Learning

The methodology is to use high resolution simulations on a randomly generated set of fine scale models. Calculating upscaled properties using the Matlab Reservoir Simulation Toolbox (MRST) framework will turn these models into a data training set for machine learning algorithms [13]. The goal of this master project is then to assess the performance of these machine-learning algorithms for the computation of upscaled permeabilities. Different machine learning algorithms will be tested and compared.

There are two purposes for using Machine learning technology in this thesis. The first goal is of theoretical nature. We want to study the capabilities of machine learning algorithms to reproduce physical processes. In this case, given a permeability distribution, is machine learning capable to capture the upscaled behaviour of the flow and encode it in a single coefficient, namely the upscaled permeability. In this case, we will use regression algorithms. At the practical level, computation of upscaled permeability using machine learning are expected to be much faster than direct computations. Teaching of the machine may take some time, but once the machine has learnt, evaluation is very fast. This is a core property of machine learning: learning may be costly, but once it is done, then it is very efficient. Direct computations of upscaled permeabilities require to assemble and solve a large linear system that can be costly and has to be done again and again.

There has been very little work done in this field trying to replace direct numerical computations with a machine learned algorithm. However, the recent resurgence in the popularity of machine learning image recognition, classification and analyses has spiked the interest. There are already several attempts being made at teaching a machine to learn the intricate patterns of a the porous media in a reservoir in order to do construction of geological models based on limited samples. In [15] L. Mosser et al. made a generative adversarial neural network capture the statistical and physical behaviour of three different porous media; Ketton limestone, Berea Sandstone and Beadpack. In their work they trained the neural network on 128^3 voxel samples and then let the learned machine create new synthetic ones. These where then evaluated through how close they resemble the test samples in terms of porosity, specific surface area, single-phase effective permeability and Euler characteristic. The latter being a number that describes a topological space's shape or structure regardless of the way it is bent. The method shows great promise and show a good agreement between synthetic and test samples. In [2] by S. Chan and A.H. Elsheikh. the same algorithm was applied on 50^2 conceptional images of meandering patterns and semi-straight channelled structures that appear in these types of porous media. In their evaluation they where comparing permeability, with the conclusion that the flow physics induced by the generated realizations were close to reference. Further work was done by E. Dupont et al. in [5] where among other things trained the algorithm on real world fluvial patterns from a meandering underground river in Kuskokwim, Alaska. In conclusion they state that they have developing the method into a powerful and flexible framework that is superior to existing geological modelling tools on several aspects.

The presented work in this thesis aims to look into further applications of machine learning in reservoir modelling. As described we are interested in seeing if machine learned algorithms also can replace direct computations for upscaling permeability. If our approach works, then this could be the start of a methodology in witch all the parts that constitute a reservoir model are done by one or several learned machines.

Chapter 2

Modelling Reservoir Rocks

The main goal of this chapter is to outline how reservoir rocks, rocks that contain hydrocarbons or aquifer systems, are modelled.

2.1 Properties of Aquifers

Aquifer is per definition a body of permeable rock or soil which can contain or transmit water. To achieve this aquifer rocks and soils consist of a certain percentage of solid and a certain percent of void space.The fraction of water/other contents in aquifers is called the **porosity**, ϕ, which is defined by the relation

$$\phi = \frac{\text{volume of contents that are not rock}}{\text{total volume}}. \tag{2.1}$$

While a high porosity means the rock may contain a lot of fluid it does not describes the fluvial patterns or lack thereof. Sandstone is a good example of that. While sandstone consists of thin layers of silt and clay dividing it into distinct layers with large porosity the specific layout of the pore networks spread through sandstone make it often act as a caprock. A layer of hard impervious rock overlying and often sealing in a deposit of water or petroleum. In reservoir modelling the full pore network in a rock is not feasible given their small size. Instead we introduce a quantifiable measure called permeability describing the ease with which fluid can flow through a rock on a macro scale. Finally there are aquifers that can act as a caprock despite having pores for oil to flow through (high porosity) and a good structure to facilitate flow through those pores (a generally high permeability). This is due to different types of rock attracting different types of fluids. Some

rocks attract water, while others will attract oil. Since oil does not flow through water by itself, a rock that is filled with water will have difficulties to facilitate flow of oil. The fluid with the strongest attraction is therefore commonly called the wetting fluid. An aquifer tends to get filled with its wetting fluid and often that results in a reduced relative permeability of other non-wetting fluids through the same rock. From now on in this work we will refer to relative permeability as just permeability.

2.2 Representative Elementary Volumes

Aquifer modelling is characterized by vast differences in scale. Our discussion of the concepts of porosity, permeability and wetting fluids was based on considerations of how fluid flows through the pore networks in a rock. A small slice where all the details of a specific rocks pore network is correctly represented is typically in the micrometer scale and larger pores in-between pebbles are in the millimetre scale. However, the size of the aquifer in a full reservoir we seek to model are often on a kilometre scale in the horizontal plane. To cope with this we can apply conservation laws on Representative Elementary Volumes (REV) of the rock. A REV is often defined to be the smallest volume over which measurements will yield values that are representative of the whole. E.g., lab measurements of the porosity of a rock would oscillate strongly with small samples. These oscillations would dampen out as the sample size became larger and larger, and a representative elementary volume could be defined when the measurements gave consistent readings. I.e there is a certain continuum within a REV.

2.3 Conservation laws

For mathematical purposes a REV can be looked at as an isolated domain that is governed by physical laws that apply to certain conserved measurable quantities $Q \in R^m$. These laws are called conservation laws. The conservation principle states that the rate of change in a conserved quantity within the domain of a isolated system Ω, is equal to the sum of inflow and production minus outflow and removal. Mathematically the general conservation law for Q on $\Omega \in R^m$ in integral form can be stated as

$$\begin{aligned} \frac{\partial}{\partial t}\int_\Omega Q(x_1, x_2, .., x_m, t)dx + \int_{\partial\Omega} F(x_1, x_2, .., x_m, t, Q)\cdot n\, d(\partial s) \\ = \int_\Omega S(x_1, x_2, .., x_m, t, Q)dx, \end{aligned} \tag{2.2}$$

for $t \geq 0$. This equation describes the rate of change of the variable Q inside the domain Ω after a certain time $t = 0$. The rate of change is dependent on the normal component of the flux function F, defined on the domain boundary $\partial\Omega$ with outer normal n, and the function $S \in \Omega$ representing the sources and sinks within the domain. The functions and variables might be vector valued functions, meaning that $Q, F, S : R^m \rightarrow R^m$. Also the source term can depend on other variables than what is stated in (2.2).

Using the divergence theorem and requiring the integrand to be identically equal zero, the partial differential equation (PDE) form can be obtained from equation (2.2),

$$Q_t(x_1, x_2, .., x_m, t) + \nabla \cdot F(x_1, x_2, .., x_m, t, Q) = S(x_1, x_2, .., x_m, t, Q). \tag{2.3}$$

Now in the case of a REV being the domain Ω and the particular conservation law being the law of mass conservation we have that change of mass in the REV being $Q_t(x_1, x_2, .., x_m, t) = \frac{\partial}{\partial t}(\rho\phi)$ equal to the flow in and out $\nabla \cdot F(x_1, x_2, .., x_m, t, Q) = \nabla \cdot (\rho \boldsymbol{v})$ plus any sources or sinks $S(x_1, x_2, .., x_m, t, Q) = q$. This gives us the following mass conservation equation

$$\frac{\partial}{\partial t}(\rho\phi) + \nabla \cdot (\rho \boldsymbol{v}) = q. \tag{2.4}$$

ρ is the density of the substance in the flow model, ϕ is the porosity and $\boldsymbol{v}$ is the Darcy velocity of the substance, further derived in (2.6). For an incompressible fluid the density does not change and hence the first term in the equation is zero. In that case, the law simplifies to just

$$\nabla \cdot \boldsymbol{v} = q. \tag{2.5}$$

2.4 Darcys Law

In the flux term in (2.4) the $\boldsymbol{v}$ is a relationship between the REVs rock permeability $\mathbf{k}$, μ the viscosity of the fluid and p pressure. This relationship,

$$\boldsymbol{v} = -\frac{\mathbf{k}}{\mu}\nabla p, \tag{2.6}$$

was first a law discovered through experimental observations by the French hydrologist Henry Darcy. Today it can also be derived from the Navier-Stokes equations. The rock permeability $\mathbf{k}$ in this equation is a tensor as pressure can be applied

in every direction. For a two dimensional case we will have a two dimensional permeability represented as a 2×2 matrix.

Darcys Law is a mathematical statement witch neatly summarizes several properties of flow through an aquifer, including:

- If there is no pressure gradient over a distance, no flow occurs.
- If there is a pressure gradient, flow will occur from high pressure towards low pressure.
- The greater the pressure gradient through the same material and formation, the greater the discharge rate.
- The discharge rate will be different through different material and formation, even if the pressure gradient is the same.

Chapter 3

Upscaling for reservoir simulation

3.1 Multiscale Modelling

For full sized reservoirs simulation the traditional approach has been to model geological structures with a geological model, and fluid flow with a coarser simulation model [13]. Geological models are produced to represent the heterogeneity of the reservoir and possibly incorporate a measure of inherent uncertainty. Pore, core, and bed models are mainly designed to give input to the geological characterization and to derive flow parameters for simulation models. The process of making a geological model is generally strongly under-determined. It is therefore customary, in particular on the reservoir scale, to use geostatistical methods to generate plausible distributions of petrophysical properties.

A starting point for the work done in this thesis is the creation of a python script that generate several simple simulation models derived through a Gaussian distribution of permeabilities. Given more time using some of the results from [5], [15] or [2] would be interesting as it would strengthen the idea that one could use Machine Learning through the whole modelling process. However using standard geostatistical methods does a better job at testing if replacing parts of the process of reservoir simulation integrate with the current methods.

3.2 Averaging Techniques

Heterogeneity and correlations in petrophysical properties depend strongly on the patterns in the sedimentary deposits and flow patterns tend to be strongly affected by details in the structural architecture of the reservoir. There is therefore a general trend to build complex, high-resolution models for geological characterization to represent small-scale geological structures. Likewise several models that are equally probable to be true are generated to systematically quantify model uncertainty. While high-resolution models can describe a wide variety of geological structures, there are also many structures on a finer scale than the resolution of the geological model that are thought to be important to understand the reservoir. Therefore, it is common to develop hierarchies of models that cover a wide range of physical scales to systematically propagate the effects of small-scale geological variations observed in core samples up to the reservoir scale. Upscaling refers to the process of propagating properties and parameters from a model of high spatial resolution to a model of lower spatial resolution as illustrated in Figure 1.2. In this process, heterogeneous regions in a reservoir model are replaced by homogeneous regions to make up a coarser model of the same reservoir.

In other words we are looking for the average of a petrophysical property. However simply applying the arithmetic mean witch is what most people think of as the average of something can be disastrous as the new homogeneous regions are unlikely that they preserve the effects of small-scale variations. A good example of where the arithmetic mean is disastrous as it returns a high value of permeability when there is none, is if we assume within our domain we have a huge permeable area that cover most of the domain. However this area is enclosed within non-permeable rock resulting in no flow through the area at all. The opposite is also quite likely to happen where there is a semi-straight "pipe" going through the aquifer with high permeability, but a large portion of the domain is non-permeable rock. Despite this, the arithmetic mean still gets used, alongside with other trivial averaging methods such as median, mode, geometric mean or harmonic mean. In certain special cases these methods combined with good geological knowledge of the aquifer is the preferred technique [17]. Still a more complex averaging algorithm that is more likely to preserve the effects of small-scale variations is needed in other cases. How this averaging should be performed depends on the type of property to be upscaled. One distinguishes between **additive properties** that can be upscaled using the trivial averaging methods and **nonadditive properties**, for which correct averaging or simple methods only exist in special cases and in most cases the best one can hope for is to compute accurate approximations. First off, when a change of scale happens it does not automatically follow that the same flow equations govern the model. Secondly is how to honour heterogeneities at the subgrid level.

This far there are only rigorous mathematical theory for asymptotic analysis of pe-

riodic structures, such as periodic and stratified media [13]. However since natural rocks very seldom are periodic these explicit analytical methods can only be used in special cases. Most upscaling techniques rely on some kind of local averaging procedure in which effective properties in each grid block are calculated solely from properties within the grid block. As such, these averaging procedures do not consider coupling beyond the local domain, which in turn implies that the upscaling methods fail to account for the effect of long-range correlation and large-scale flow patterns in the reservoir unless these can be represented correctly by the forces that drive flow inside the local domain. Hence it is generally acknowledged that global effects must also be taken into consideration to obtain robust coarse-scale simulation models. Since these obstacles are not the main focus of this thesis we will use a simplified grid, that will be outlined later, as well as making the assumption that said grid always accounts for global effects.

The literature on upscaling techniques is extensive, ranging from trivial averaging techniques [10], to more comprehensive global and localglobal methods [1] [6]. There are comprehensive review papers written on this topic, like [21]. Some attempts have been made to analyse the upscaling process itself, like [22], but so far there is generally no theory or framework for assessing the quality of an upscaling technique. In fact, upscaling techniques are seldom rigorously quantified with mathematical error estimates. Instead, the quality of upscaling techniques is usually assessed by comparing unscaled production characteristics with those obtained from a reference solution computed on an underlying fine grid.

3.3 Upscaling additive properties

Porosity is the simplest example of an additive property and can be upscaled through a simple volumetric average. If Ω denotes the region we want to average over, the averaged porosity value is simply given as

$$\phi* = \frac{1}{\Omega} \int_{\Omega} \phi(\boldsymbol{x}) d\boldsymbol{x}. \tag{3.1}$$

This is nearly trivial averaging and shows why upscaleing of additive properties is not a focus of the work in this thesis. However permeability being a non-additive property things are about to get much harder.

3.4 Upscaling permeability

Given a subdomain Ω, we want to compute a single upscaled permeability that can be used to compute an approximation of the flow in Ω. Our starting point is the fine scale problem for the permeability tensor distribution $K(x)$. The equations for the fine scale problem is the mass conservation equation,

$$\nabla \cdot (K(x)\nabla p) = 0 \tag{3.2}$$

with boundary conditions on $\partial\Omega$. The upscaled permeability tensor $\bar{K}$ is a constant tensor such that, for the same boundary conditions, the flow obtained by solving the problem

$$\nabla \cdot (\bar{K}\nabla p) = 0 \tag{3.3}$$

will provide a good approximation of the fine scale problem. By a good approximation, we do not mean to match point-wise quantities between the two computed flows, but only averaged quantities, such as for example the integrated outflux on one side, see below.

How accurate such an approximation can be depends on the coarse grid, the specific upscaling method, the purpose for witch the upscaled values are to be used, and the complexity of the fine-scale permeability distribution. Most techniques for upscaling absolute permeability seek an averaged tensor $\bar{K}$ that reproduces the same total flow through each homogeneous region as one would obtain if the single-pressure equation 3.2 was solved with the full fine-scale heterogeneity.

This reflects that $\bar{K}$ depends on the flow through Ω, which in turn is determined by the boundary conditions that are specified on $\partial\Omega$. The better we know the boundary conditions the homogenized region will be subject to in subsequent simulations, the more accurate estimates we can compute for the upscaled tensor $\bar{K}$. In fact, if we know these boundary conditions exactly, we can compute the true effective permeability. In general, we will not know these boundary conditions unless we have already solved our problem. This followed by the problem that the permeability tensor K needs to be symmetric and positive definite and there is no guarantee that this is the case. This makes it so that the single-phase upscaling problem is fundamentally ill-posed.

In the process of upscaling, we loose the fine description of the flow. The hope is that, with only few coefficients, we retain the main features of the flow. We follow the methodology described in [7]. While the general problem of upscaling permeabilities is not necessarily solvable, a case that is useful in many scenarios is to the reduced problem in two dimensions with a known boundary. For simplicity, we will also only consider diagonal permeability tensors, that is $K_{1,2}(x, y) = K_{2,1}(x, y) = 0$.

3.5 Computation of upscaled permeabilities in two dimensions

We consider a two dimensional square grid $\Omega \in [0, L] \times [0, L]$, as illustrated by Figure 3.1.

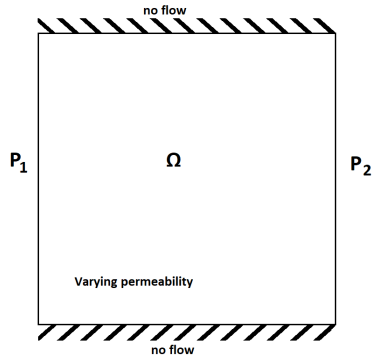

Figure 3.1 – Illustration of our two dimensional grid with known boundary

The mass conservation equation gives us

$$\nabla \cdot \boldsymbol{v} = 0, \tag{3.4}$$

see (2.5) for $q = 0$. We are going to compute the constant permeability tensor $\bar{K}$ which matches $K(x)$ for a specific set of boundary conditions. This consists of two problems, one for each Cartesian direction (horizontal or vertical). For the horizontal case, the boundary condition is given as follows. At the top and bottom boundary, we impose a no-flow boundary condition

$$\boldsymbol{v} \cdot \boldsymbol{n} = 0, \tag{3.5a}$$

On the vertical sides, we impose constant pressure. We have

$$p(0, y) = p_1 \text{ and } p(L, y) = p_2, \tag{3.5b}$$

for $y \in [0, L]$. The constant p_2 and p_1 are given. We want to find the upscaled permeability tensor $\bar{K}$ such that the integrated outflux matches exactly in this case with the one obtained by solving the fine scale problem.

Let us first assume that the permeability tensor is constant in Ω, that is $K(x, y) = \bar{K}$, for some diagonal matrix $\bar{K} \in \mathbb{R}^{2\times 2}$. The Darcy flux is given by

$$\bar{\boldsymbol{v}} = -\bar{K}\nabla\bar{p}. \tag{3.6}$$

We claim that the pressure given by

$$\bar{p}(x, y) = \frac{\Delta p}{L} x + p_1$$

is a solution to the problem as it satisfies (3.4), (3.5a) and (3.5b). Where $\Delta p = p_2 - p_1$.

Here we have

$$\nabla \bar{p} = \begin{bmatrix} \Delta p/L \\ 0 \end{bmatrix}$$

Hence, in vector notation equation (3.6) becomes

$$\bar{\boldsymbol{v}} = - \begin{bmatrix} \bar{K}_{11} & 0 \\ 0 & \bar{K}_{22} \end{bmatrix} \begin{bmatrix} \Delta p/L \\ 0 \end{bmatrix} = -\bar{K}_{11} \frac{\Delta p}{L} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

For easier notation we use E to denote the east boundary, and by extension of that $|E|$ denotes the area on the east boundary. Using law of mass conservation once again we have the relation that flux through the east boundary should be

$$\bar{\boldsymbol{v}}_E = \int_E \bar{\boldsymbol{v}} \cdot \boldsymbol{n} \, ds = - \int_E \bar{K}_{11} \frac{\Delta p_x}{L} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \, ds = - |E| \, \frac{\Delta p}{L} \bar{K}_{11}. \tag{3.7}$$

After running a fine scale computation, using the boundary conditions as illustrated in Figure 3.2 (3.5), we obtain a solution $p(x, y)$. To solve the flow problem, we use the standard two-point flux-approximation method (TPFA). For a two-dimensional Cartesian grid and with homogeneous permeability, the TPFA method is equivalent to a classical five-point finite-difference scheme for Poisson's equation. First, we compute the transmissibilities at each face, then we assemble and solve the corresponding discrete system. This is done through invoking the functions `computeTrans()` and `incompTPFA()` functions in the incompressible flow module from MRST, as explained in the manual [13]. We use this solution to compute the integrated flux,

$$\boldsymbol{v}_E = - \int_E K \nabla p \cdot \boldsymbol{n} \, ds.$$

Our requirement is $\boldsymbol{v}_E$ matches exactly the result obtained from the upscaled problem, that is we require $\bar{\boldsymbol{v}}_E = \boldsymbol{v}_E$. Hence, from (3.7), we obtain an expression for $\bar{K}_{1,1}$ that we use as our upscaled permeability, given by

$$\bar{K}_{11} = - \frac{\boldsymbol{v}_E}{|E|} \frac{L}{\Delta p}. \tag{3.8}$$

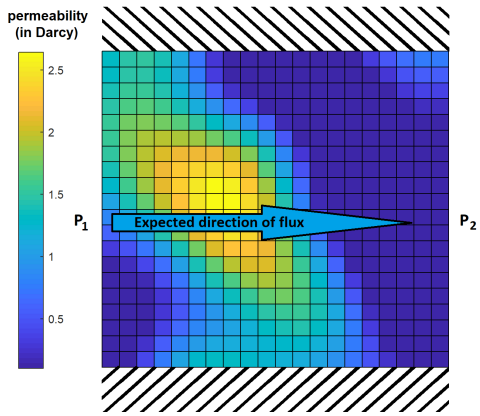

Figure 3.2 – Figure showing a grid containing a permeability field. The striped area represents no-flow on the northern and southern boundaries, while P_1 and P_2 are the constant pressures boundary conditions at the western and eastern boundaries with $P_1 >> P_2$. An expected direction of the fluid flux is added on top for clarity.

Chapter 4

Generation of permeability field

The methodology of the presented work is to use high resolution simulations on a randomly generated set of finer scale models. This data set will be used as training set for a machine learning algorithm. To generate the permeability field, we will use multivariate Gaussian fields.

4.1 Multivariate Gaussian

When creating our data set we create a a two dimensional square grid $\Omega \in [0, L] \times [0, L]$, as illustrated by Figure 3.1 from previous chapter. This domain is decomposed into a 20×20 grid constituting a total of $N = 400$ cells. To each cell corresponds a given value of permeability. Thus, we have to determine a vector $\boldsymbol{k} \in \mathbb{R}^N$ which, for each cell indexed i, gives the corresponding permeability value k_i at this location. We denote by $p_i \in \mathbb{R}^2$ the coordinates of the centroid of the cell indexed i. We want to generate this vector randomly but we also want to include the property that the permeability values for cells in the same neighbourhood are correlated. That is the correlation between k_i and k_j is function of the distance $\|p_i - p_j\|$. Multivariate Gaussian is a tool that has been used in such purpose and kriging methods have been used extensively in the field of geosciences, [3] [4]. Let us denote a generic multivariate variable by $\boldsymbol{x}$ and its probability distribution by $\rho(\boldsymbol{x})$. That is a function that describes all the possible values that $\boldsymbol{x}$ can take within its range. Taking the integral of this function over an interval will return the probability that $\boldsymbol{x}$ will lie within that interval. As a trivial follow up to this

taking the integral over the whole range of $\boldsymbol{x}$ will result in a probability of 1. A multivariate Gaussian random variable is a random variable vector whose probability distribution is given by

$$\rho(x_1, \ldots, x_k) = \frac{1}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\right) \tag{4.1}$$

where $\mu \in \mathbb{R}^N$ is a vector which gives the average of $\boldsymbol{x}$ and $\Sigma \in \mathbb{R}^{N \times N}$ is a matrix which is called the covariance matrix. We can check that

$$E((x_i - \mu_i)(x_j - \mu_j)) = \int_{\mathbb{R}^N} (x_i - \mu_i)(x_j - \mu_j)\rho(\boldsymbol{x})\, d\boldsymbol{x} = \Sigma_{i,j},$$

hence the name of covariance matrix. Because the covariance of the i-th random variable with the j-th one is the same thing as the covariance of the j-th random variable with the i-th one, every covariance matrix is symmetric. In addition, every covariance matrix is positive semi-definite.

We want to use a correlation matrix that takes into account the proximity between points as our covariance matrix. It can be done by introducing a covariance *function*, which we denote $k : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ and use in our matrix

$$\Sigma = \begin{pmatrix} k(\boldsymbol{p}_1, \boldsymbol{p}_1) & k(\boldsymbol{p}_1, \boldsymbol{p}_2) & \ldots & k(\boldsymbol{p}_1, \boldsymbol{p}_N) \\ k(\boldsymbol{p}_2, \boldsymbol{p}_1) & k(\boldsymbol{p}_2, \boldsymbol{p}_2) & & \\ \vdots & & \ddots & \vdots \\ k(\boldsymbol{p}_N, \boldsymbol{p}_1) & & \ldots & k(\boldsymbol{p}_N, \boldsymbol{p}_N) \end{pmatrix} \tag{4.2}$$

The covariance function must be such that the resulting matrix Σ is a proper covariance matrix, meaning that it must be symmetric positive. There exist several choices of covariance function. We choose the most standard one which is the radial basis function given by

$$k(\boldsymbol{p}, \bar{\boldsymbol{p}}) = \exp(-\kappa\, |\boldsymbol{p} - \bar{\boldsymbol{p}}|^2). \tag{4.3}$$

$k(\boldsymbol{p}, \bar{\boldsymbol{p}})$ return the same result if we swap $\boldsymbol{p}$ and $\bar{\boldsymbol{p}}$, while the range of the exponential function are only positive numbers. Hence, the matrix Σ becomes symmetric positive semi-definite. This covariance function also achieve our goal perfectly, creating a high dependency between cells in the domain that are close to each other and a rapidly reducing dependency with increased distance.

4.2 Implementation

The implementation of the generation of the permeability field is done in Python. In fact, all of the coding in this thesis is done in Python, with the exception of the

high resolution computations of the upscaled permeabilities which is done using MRST toolbox for MATLAB as explained in Chapter 3.5. All of the Python code we have written relies heavily on the `NumPy` library for all the mathematical and vector array functionality. In the generation of the permeability field, an important choice is to set the value of the covariance parameter κ, see (4.3). Large values of κ lead to less spatial dependences and therefore a more oscillating permeability field. We tested different values of κ ranging from 10^{-2} to 10^{2}, see Figure 4.1.

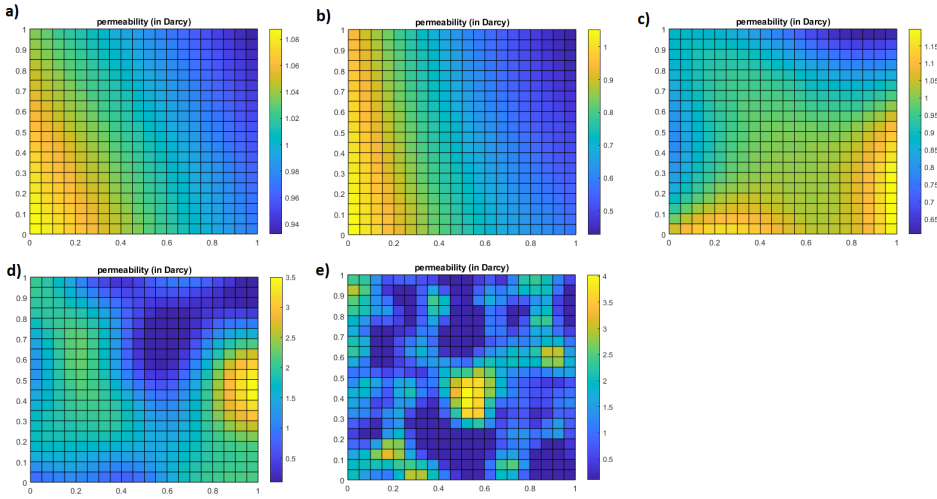

Figure 4.1 – Plot of randomly generated permeability fields with different values of κ: a) $\kappa = 10^{-2}$, b) $\kappa = 10^{-1}$, c) $\kappa = 1$, d) $\kappa = 10$, e) $\kappa = 100$. When comparing the plots, it is important to look at the color scales. For example case a) and b) are significantly different even if they look similar at first sight.

We compute a permeability field through random draw in accordance to the multivariate Gaussian probability distribution (4.1). The first step in the program is to create coordinate matrices out of the coordinate vectors through the function `numpy.meshgrid` corresponding to the grid. Using these coordinate matrices the next step is to compute the covariance matrix in accordance to Equation (4.2) and (4.3). Given this covariance matrix, we use the function `numpy.random.multivariate_normal` to generate a random multivariate vector $\boldsymbol{x} \in \mathbb{R}^N$ of zero mean. Then, we define the vector $\boldsymbol{K} \in \mathbb{R}^N$ giving the permeability at each cell, as

$$K_i = K_{\text{mean}} + \sigma_K x_i,$$

where K_{mean} and σ_K are given constants. We use $K_{\text{mean}} = \sigma_K = 1$ Darcy. Finally, we set the few negative values that are obtained to a given small value.

We created 100 permeability fields for each of our chosen values of κ: $\kappa = 10^{-2}$, $\kappa = 10^{-1}$, $\kappa = 1$, $\kappa = 10$ and $\kappa = 100$. For each of those, we obtain the corresponding upscaled permeability by running the high resolution computations. These results

give us the dataset on witch we train our machine learning algorithms. Figure 4.2 shows a sample of the generated permeability fields with $\kappa = 10$ we created. Figure 4.3 provides an example of how the permeability field effects the flow, as explained in Chapter 3.5.

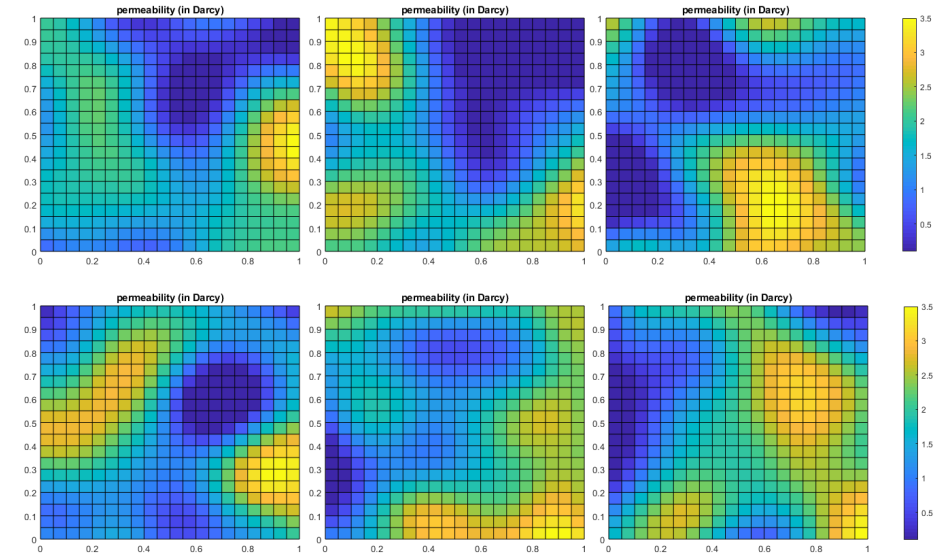

Figure 4.2 – Plot of six randomly generated permeability fields with $\kappa = 10$ taken from our dataset.

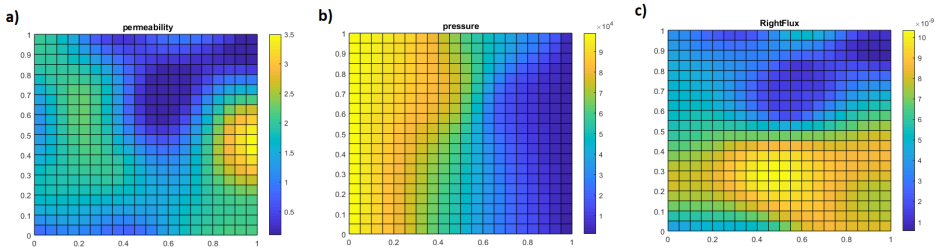

Figure 4.3 – Illustration of the flow pattern obtained from one of the permeability fields of our dataset (upper left permeability field in Figure 4.2. a) permeability field, b) pressure field obtained by running the high-resolution simulation, solving Equations (3.4) and (3.5), c) visualization of the flux in the horizontal direction.

Chapter 5

Machine Learning

As stated before, the whole purpose of the work in this thesis is to see if a machine learning algorithm can do a good job at solving the problem that upscaling absolute permeability presents. However first this chapter will introduce some basic concepts for machine learning. First off we can divide machine learning into three categories.

5.1 Reinforcement learning

The first category is an example of machine learning where the machine is trained to take specific decisions based on the business requirement expressed as an utility function. This utility function is often a expression that it needs to maximize or minimize, resulting in maximizing of efficiency, performance or profit in said business or minimizing accidents or expenses. The idea involved in reinforcement learning is that the machine trains itself on a continual basis based on the environment it is exposed to, and applies its enriched knowledge to in the future solve problems better in agreement with its utility function. While there is a whole ensemble of different machine learning algorithms in this category most of them build on the idea of making decisions, witch is not the problem we are trying to solve. However algorithms concerning dimensionality reduction are a integral part of more complex regression algorithms, so in a sence we are doing some reinforcement learning in one of our methods. They have both in common that they undergo a decision process in order to reduce the number of random variables under consideration.

5.2 Unsupervised learning / Descriptive models

Unsupervised learning is used to train descriptive models where no target is set and no single feature is more important than the other. The case of unsupervised learning can be: When a retailer wishes to find out what are the combination of products, customers tends to buy more frequently. Neural network algorithms, like the one used in [5], [15] and [2] belong to this category and have shown great success in learning the intricate patterns in aquifers. However as their end goal was descriptive, our goal is a little different. They tough the machine to learn what the patterns there were given no prior knowledge of them. We seek to do more than just describe the patterns and reproduce them in synthetic samples. We want to teach the machine to predict an upscaled permeability from the samples. Hence we should first look at a different category of machine learning algorithms. If we had more time we could perhaps revisit the descriptive approach to our problem.

5.3 Supervised Learning / Predictive models

Predictive model as the name suggests is used to predict the future outcome based on the historical data. Predictive models are normally given clear instructions right from the beginning as in what needs to be learnt and how it needs to be learnt. These class of learning algorithms are termed as Supervised Learning. As our upscaling problem is used in a predictive manner to predict upscaled permeabilities, the chosen algorithms are in this category. In fact, we will look at algorithms from a subcategory of predictive models; Regression algorithms.

Supervised machine learning is best understood as approximating a target function f that maps input variables X to an output variable Y

$$Y = f(X). \tag{5.1}$$

Once the function has been trained, we want to compute the output $\boldsymbol{Y}_*$ for an input data $\boldsymbol{X}_*$ which does not belong to the training set.

The characterization (5.1) describes the range of prediction problems and the machine algorithms that can be used to address them. If we narrow it down to regression algorithms the target function is defined in the model as we search for functions within a set of functions. If we use linear regression, we are looking for a linear target function. Quadratic regression looks for a quadratic equation, and so on. Model capacity is tied to the ability a algorithm has to match the training data. A curved line will be able to fit through more points than a straight line. While that is the strength of a higher capacity algorithm there are also weaknesses to choosing a too high capacity model.

5.4 Underfitting vs. Overfitting

An important consideration in learning the target function from the training data is how well the model generalizes to new data. Generalization is important because the data we collect is only a sample, it is incomplete and noisy. Overfitting occurs when our algorithm has learned too much from our data, up to the point of mapping curve shapes and rules that do not exist. It is picking up on not just the general characteristics of the data, but also the random noise, errors and very specific characteristics. This is often a symptom of choosing a to high capacity model or giving a machine algorithm to many similar training sets. While fitting the training data excellently any slight change in the procedure or in the training data produces erratic predictions and large generalization errors. Underfitting refers to a model that can neither model the training data nor generalize to new data. An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

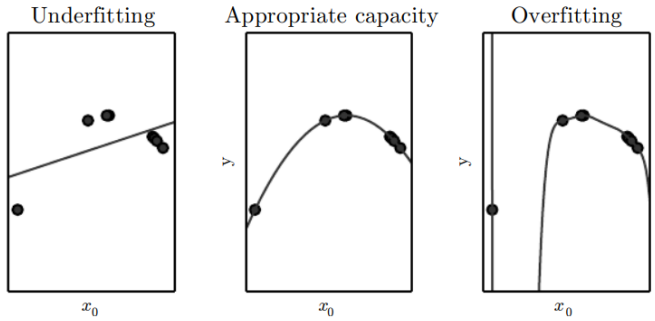

Figure 5.1 – Example figure showing a regression algorithm with increasing model complexity on the same dataset. First underfitting, then fit appropriatly and finally overfitting when the model complexity becomes to high. Illustration from [9].

General behaviour is that with increasing model capacity the error on the training set itself will decrease, but the model is loosing its ability to generalize to new data and hence increasing the overall error as shown in Figure 5.2.

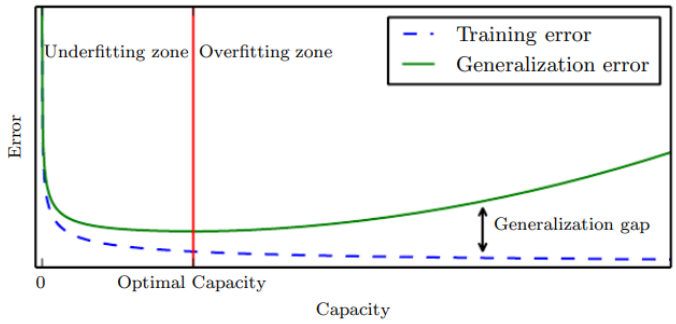

Figure 5.2 – Figure illustrating that after a certain capacity a model will have increasing generalization error. Illustration from [9].

5.5 Ordinary Least Squares

The method of least squares is a standard approach in regression analysis to the approximate solution of overdetermined systems. We have a set of N pairs of observations (x_i, y_i) that are used to find a function relating the value of the dependent variable $\boldsymbol{y}$ to the values of the independent variable $\boldsymbol{x}$. The algorithm seeks to find the linear mapping $f : \mathbb{R}^d \to \mathbb{R}$ which minimizes the error

$$\sum_{i=1}^{N} |y_i - f(x_i)|^2 . \tag{5.2}$$

More graphically, starting from a set of data points this corresponds to solving the problem of finding a line that fits this set of data points the best, as illustrated in Figure 5.3. Then the metric by witch the algorithm measures the error ie. the distance between the measurements and the model is sum of euclidean distances squared. This is also how the method derives it's name.

Since f is linear, $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + w_0$, then the w_0 parameter specifies the intercept of the line. Given that data can be preprocessed and centred,

$$x_i^{centred} = x_i - \frac{1}{N}\sum_{i=1}^{i=N} x_i \qquad \text{and} \qquad y_i^{centred} = y_i - \frac{1}{N}\sum_{i=1}^{i=N} y_i \tag{5.3}$$

the intercept parameter can be set to zero $w_0 = 0$, and in futher computations we will assume that this is always the case. The problem we seek to minimize is

$$\{\boldsymbol{w}^*\} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{N} |y_i - f(\boldsymbol{x}_i)|^2 \tag{5.4}$$

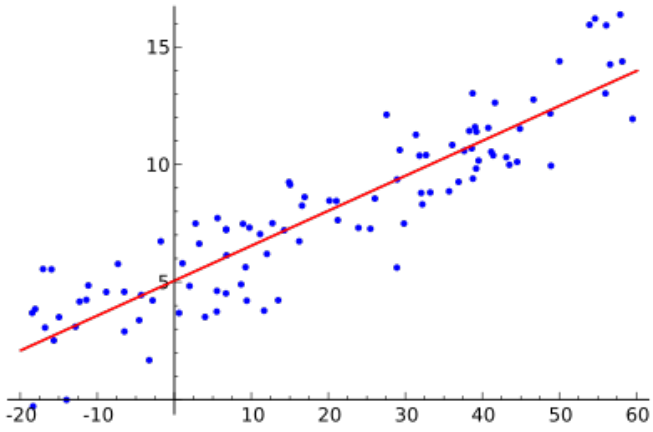

Figure 5.3 – Example figure conceptualizing Ordinary least squares.

We introduce the notation

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}, \qquad \boldsymbol{X} = \begin{pmatrix} x_{1,1} & \dots & x_{1,d} \\ & \vdots & \\ x_{N,1} & \dots & x_{N,d} \end{pmatrix} \tag{5.5}$$

In our implementation the response variable $\boldsymbol{y}$ is the upscaled permeability, a vector with the length equal to the amount of training samples. Correspondent our dataset of independent variables $\boldsymbol{X}$ is a matrix the size of all the permeabilities in each grid cell times the amount of training samples.

We denote the argument on the right-hand side in (5.4) by $e(\boldsymbol{w})$. We have

$$e(\boldsymbol{w}) = \sum_{i=1}^{M} |y_i - f(\boldsymbol{x}_i)|^2 = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 . \tag{5.6}$$

We can expand the expression above and get

$$e(\boldsymbol{w}) = \boldsymbol{y}^\top \boldsymbol{y} - \boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} + 2\boldsymbol{y}^\top \boldsymbol{X}\boldsymbol{w}$$

Convex problem, the solution satifies $\nabla e(\boldsymbol{w}^*) = 0$, i.e.

$$\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w}^* = \boldsymbol{X}^\top \boldsymbol{y}. \tag{5.7}$$

The solution to (5.7) is not well-defined (it is not unique), because $\boldsymbol{X}^\top \boldsymbol{X}$ can be ill-conditioned as a result of the problem being ill-posed. In these cases, the least squares estimate amplifies the measurement noise and may be grossly inaccurate.

5.6 Ridge regression

Ridge Regression is a complement to ordinary least squares that seeks to solve the shortcomings of the algorithm. By adding the small term we get the inner product to move away from possible singular matrices making the algorithm useful for solveing a ill-posed problem.

There are three ways to explain ridge regression. From a machine learning perspective ridge regression is about introducing weights that perform a decision process, often found in reinforced learning algorithms, in order to reduce the number of random variables under consideration. This is done by imposing a penalty on the size of coefficients through imposing a shrinkage parameter $\alpha > 0$ in the error term in the mineralization problem. We replace $e(\boldsymbol{w})$ defined in (5.6) with

$$e(\boldsymbol{w}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \alpha \|\boldsymbol{w}\|_2^2 \tag{5.8}$$

From a mathematical point of view we are introducing weights that seek to improve the conditioning of the underlying problem, enabling a direct numerical solution aka. de-singularization of the matrix. When computing the solution to the minimization problem, that is, get a formula like (5.7) we end up with

$$(\boldsymbol{X}^\top \boldsymbol{X} + \alpha \boldsymbol{I})\boldsymbol{w}^* = \boldsymbol{X}^\top \boldsymbol{y}.$$

The difference this time is that this can be seen as adding diagonal elements to the $\boldsymbol{X}^\top \boldsymbol{X}$ matrix witch gives it a full rank. This means that $\boldsymbol{X}^\top \boldsymbol{X} + \alpha \boldsymbol{I}$ is invertible for $\alpha > 0$ and therefore well-defined (unique).

From a qualitative viewpoint we can look at the behaviour of prediction models through the two qualitative subcomponents of error, bias and variance [11] as illustrated in Figure 5.4. From this point of view ridge regression is said to increase bias in the model and get a lower variance. This is due to the higher variance parameters being shrinked.

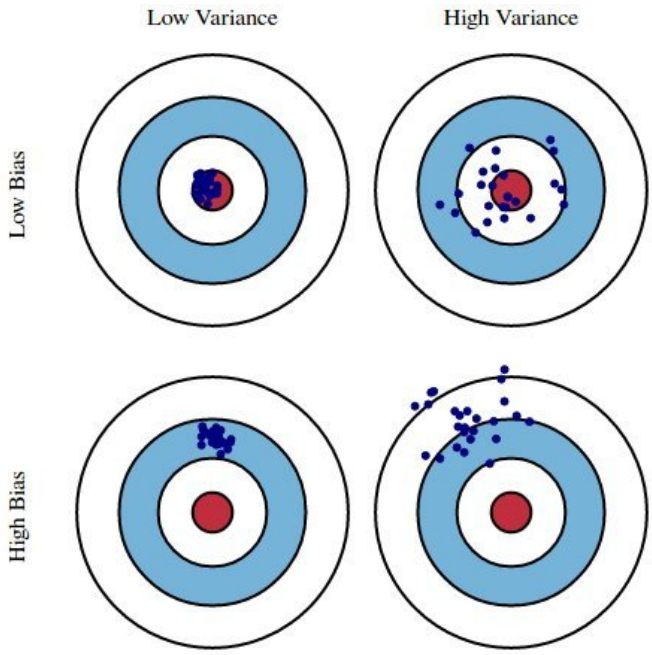

Figure 5.4 – Figure showing a conceptional image explaining bias vs. variance in statistical models.

5.7 The kernel trick

Another limitation of the Ordinary least squares is the low capacity of the model as it is by default a linear method. There are many ways to extend Ordinary least squares to a search for a solution function in higher degree polynomials or some other common functions, but this is often a a tedious way to do it and the generalization to higher dimensions is not always trivial. Instead introducing a map to an higher dimensional space to remove non-linearities and make linear methods efficient is the preferred approach. As an example, we can represent the space of polynomial of a given degree q by using the feature map $\boldsymbol{\phi}$ given by

$$\boldsymbol{\phi}(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^q \end{bmatrix}$$

The space of polynomial of degree q can then be represented by a vector $\boldsymbol{w} \in \mathbb{R}^q$ as

$$f(x) = \boldsymbol{w}^\top \boldsymbol{\phi}(x) = \sum_{i=1}^{q} w_i x^{i-1}.$$

As in the previous chapter we consider a set of data points $\boldsymbol{x}_i$, $i = 1, \ldots, N$. We move to an higher dimensional space, the **feature** space, by adding features. We introduce a feature map which has the form

$$\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^f.$$

The mapping $\boldsymbol{\phi}$ is typically non-linear and the dimension of the image is very large, that is $f \gg d$.

Then, we look for a function f using *standard linear* regression but in the feature space. The function takes thus the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \phi_i(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{w}.$$

If the feature map is the identity, $\boldsymbol{\phi} = \boldsymbol{I}$ with $\mathbb{R}^d = \mathbb{R}^f$, then we are back to standard linear regression. In addition to the notations given un (5.5), we introduce the matrix given by $\boldsymbol{\psi}(\boldsymbol{X}) \in \mathbb{R}^{f \times N}$

$$\boldsymbol{\psi} = \begin{pmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1) & \boldsymbol{\phi}(\boldsymbol{x}_2) & \ldots & \boldsymbol{\phi}(\boldsymbol{x}_N) \end{pmatrix} \tag{5.9}$$

The regression problem on the feature space consists of finding the minimum of

$$e(\boldsymbol{w}) = \left\| \boldsymbol{y} - \boldsymbol{\psi}^\top \boldsymbol{w} \right\|_2^2 + \alpha \left\| \boldsymbol{w} \right\|_2^2 \tag{5.10}$$

In this case, using the same techniques applied in Ridge regression we obtain that $\boldsymbol{w}$ must satisfy

$$(\boldsymbol{\psi}\boldsymbol{\psi}^\top + \alpha \boldsymbol{I})\boldsymbol{w} = \boldsymbol{\psi}\boldsymbol{y}. \tag{5.11}$$

Note that, typically, because the feature space is very large ($f \gg 1$), the matrix $\boldsymbol{\psi}^\top\boldsymbol{\psi}$ is very big and it may be prohibitively expensive to compute the solution $\boldsymbol{w}$ of this equation. In fact, as we are going to see, it is possible to avoid completely the computation of the solution $\boldsymbol{w}$. This essential computational gain constitutes the kernel trick. From (5.11), we obtain

$$\boldsymbol{w} = (\boldsymbol{\psi}\boldsymbol{\psi}^\top + \alpha \boldsymbol{I})^{-1}\boldsymbol{\psi}\boldsymbol{y}$$

Let us assume that the machine has been trained, so that f is now given, we want to evaluate $f(\boldsymbol{x}_*)$ for a given point $\boldsymbol{x}_*$. We have

$$f(\boldsymbol{x}_*) = \boldsymbol{\phi}(\boldsymbol{x}_*)^\top(\boldsymbol{\psi}\boldsymbol{\psi}^\top + \alpha \boldsymbol{I})^{-1}\boldsymbol{\psi}\boldsymbol{y} \tag{5.12}$$

This expression still contains the inverse of a very large matrix but this can be changed by using the following identity,

$$(\boldsymbol{\psi}\boldsymbol{\psi}^\top + \alpha \boldsymbol{I})^{-1}\boldsymbol{\psi} = \boldsymbol{\psi}(\boldsymbol{\psi}^\top\boldsymbol{\psi} + \alpha \boldsymbol{I})^{-1}. \tag{5.13}$$

To show that this identity holds, we rewrite it as

$$\boldsymbol{\psi}(\boldsymbol{\psi}^\top\boldsymbol{\psi} + \alpha \boldsymbol{I}) = (\boldsymbol{\psi}\boldsymbol{\psi}^\top + \alpha \boldsymbol{I})\boldsymbol{\psi},$$

which trivially holds. The expression (5.12) for $f(\boldsymbol{x}_*)$ then becomes

$$f(\boldsymbol{x}_*) = \boldsymbol{\phi}(\boldsymbol{x}_*)^\top\boldsymbol{\psi}(\boldsymbol{\psi}^\top\boldsymbol{\psi} + \alpha \boldsymbol{I})^{-1}\boldsymbol{y}. \tag{5.14}$$

The fundamental difference between (5.12) and (5.14) is that the dimension of the matrix to invert is significantly reduced from $\mathbb{R}^{f\times f}$ to $\mathbb{R}^{N\times N}$. Moreover, we observe that the values of the features $\boldsymbol{\phi}(\boldsymbol{x})$ only enter the expression (5.14) as scalar products, that is $\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{\phi}(\bar{\boldsymbol{x}})$. Let us now introduce the function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined as

$$k(\boldsymbol{x}, \bar{\boldsymbol{x}}) = \boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{\phi}(\bar{\boldsymbol{x}}).$$

The function k is called the kernel. We can rewrite the vector and matrices contained in (5.14) using only the kernel k. Let us introduce the vector $\boldsymbol{v} \in \mathbb{R}^N$ and the matrix $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ as

$$\boldsymbol{v} = \boldsymbol{\phi}(\boldsymbol{x}_*)^\top\boldsymbol{\psi} \quad \text{and} \quad \boldsymbol{K} = \boldsymbol{\psi}^\top\boldsymbol{\psi}.$$

We have

$$\boldsymbol{v}_i = k(\boldsymbol{x}_*, \boldsymbol{x}_i) \quad \text{and} \quad \boldsymbol{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

and the expression (5.14) can be rewritten as

$$f(\boldsymbol{x}_*) = \boldsymbol{v}^\top(\boldsymbol{K} + \alpha \boldsymbol{I})^{-1}\boldsymbol{y}. \tag{5.15}$$

In conclusion, it means that the features vector $\boldsymbol{\phi}(\boldsymbol{x})$, which can be very large and therefore very expansive to compute, may not be needed at all, as all the relevant information is encoded in the kernel function. Such simplification when it can be done is referred in the literature as the *kernel trick* [20]. It is typical that simple kernel functions correspond in fact to very large feature spaces.

5.8 Kernel Ridge regression

The kernel ridge regression method simply refers to applying the kernel trick as described above to Ridge regression. As in the Ridge regression method, the parameter α has a regularization effect. Indeed it penalizes large values of $\boldsymbol{w}$, see (5.10). In this thesis we have used the kernel

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma \left|\boldsymbol{x}_i - \boldsymbol{x}_j\right|^2). \tag{5.16}$$

witch is a popular kernel to choose in data science [16]. The parameters α (from Ridge) and γ (from kernel) are called hyper-parameters as the algorithm itself cannot learn them. Instead we use a grid-search to find the optimal hyper-parameters as will be described in Chapter 6.3.

In the implementation we use the `SciPy` python library for input and output handling between MATLAB where the data sets are created and python. We use the `sklearn` python library for calling the algorithms themselves witch is a library filled with tools for data mining and analysis. In particular the functions `sklearn.linear_model.LinearRegression()`

and

`sklearn.kernel_ridge.KernelRidge(kernel='rbf',` α`,` γ`)` where useful.

Chapter 6

Results

6.1 Evaluation Strategy

As explained in Chapter 4.2, we created a dataset of $N = 100$ samples with a permeability field generated through the Gaussian probability distribution and a up-scaled permeability calculated using a high-resolution finite volume method. This dataset would then be sorted ascending by upscaled permeability value, $k(1) \leq k(2) \leq \cdots \leq k(N)$. The reasoning for this are twofold. First reason is that the computed values can be graphed as a continuous line, making it easy to create a scatter plot of the predicted values on top of it. The second reason is that it also makes it easier to visually identify any patterns in the relative error, if there is any. This sorting should have no impact on the end result as the equations themselves are unchanged. We do a naive implementation of a k-fold cross-validation where we split our dataset into N equal subsets called a fold. Loop over each fold for $i = 1$ to $i = N$. Then keep the fold number i as a validation set and the remaining $N - 1$ folds in the cross-validation set. Then train the machine learning model using the cross-validation training set and calculate the accuracy of our model by validating the predicted results against the validation set. Estimate the accuracy of the machine learning model by averaging the accuracies derived in all the k cases of cross-validation.

When evaluating the trained machine we would do k-fold validation, however with a variant where we treat each sample in the dataset of 100 samples as a fold. Hence there would be one validation sample, $k_{validation}$ and 99 training samples, $\mathbf{k}_{train}$. This would go in a loop, starting with the first data being the test sample, then having the second data being the test sample, etc. In each iteration we would

predict the test sample, k_{ML} based on what the machine has learned from the other 99 samples and then measuring the relative error,

$$e_i = \frac{|k_{ML}(i) - k_{validation}(i)|}{|k_{validation}(i)|} \tag{6.1}$$

for sample number i and plotting it. This is in addition to calculating the average error

$$\bar{e} = \frac{1}{N} \sum_{i=1}^{i=N} e_i \tag{6.2}$$

as our main quantitative measure to evaluate the algorithms. This is done for a value of $\kappa = 10$ as it is the value that produced highly varying permeability fields, but without being sporadic oscillating witch often happens with higher values as seen in Figure 4.1.

6.2 Ordinary Least Squares

The results obtained for the ordinary least squares method with a $\kappa = 10$, as presented in Section 5.5, are given in Figure 6.6. The relative error can go as high as 80% for some samples. However the average error is $\bar{e} = 0.20$ ie. 20%. Another interesting observation is that the relative error seems to be shrinking with increasing sample number. Since we know the samples are sorted by ascending upscaled permeability value it means the error in the model is not depending on the actual size of the target variable.

We believe the high relative error is due to shortcomings in the algorithm that are very likely to create a underfitting problem. It assumes an underlying linear relationship between the independent and dependent variables. This is a very low model capacity and it is reasonable to believe that a higher capacity model is needed in order to match the training data and generalize to perform well on the test data. Another possible problem with the algorithm is that ordinary least squares is designed to perform well on well posed problems. However with us only using 100 samples, but having 400 cell grids that each have a corresponding coefficient in the target function the problem could in theory be underdetermined depending on how many of them are linearly independent of each other corresponding to a degree of freedom. To check if this was the case we ran the algorithm on a data set of 400 values, creating a determined or overdetermined system. The results where atrocious, as seen in Figure 6.2, showcasing that the results did not benefit from having these high number of samples with a $\bar{e} = 1.93$. The conclusion we draw from this is that, as expected, the error seen when using 100 samples is due to to low model/kernel complexity, as a more capable model would be expected to learn the test data well resulting in a very low training error typical in overfitting, when increasing the sample size.

When investigating the robustness of the algorithm with respect to the statistical distribution used for generation of the fine scale models we have to look at the performance of the algorithm with different values of κ. The results are presented in Figure 6.3, 6.4, 6.5, 6.6 and 6.7. The histogram of the errors are compared side by side in Figure 6.1. When looking at this comparison we can see that the error is increasing with increasing values of κ. This is as expected given that the oscillation and general complexity of the permeability field is higher the higher the kappa.

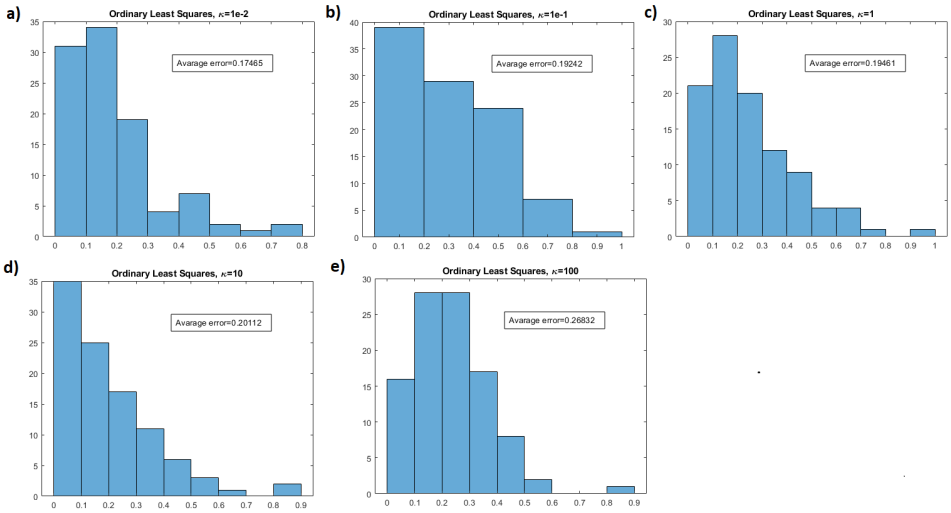

Figure 6.1 – Histogram plot of the relative error and also the computed average error for Ordinary Least Squares and different κ. a) $\kappa = 10^{-2}$ with $\bar{e} = 0.17465$. b) $\kappa = 10^{-1}$ with $\bar{e} = 0.19242$. c) $\kappa = 1$ with $\bar{e} = 0.19461$. d) $\kappa = 10$ with $\bar{e} = 0.20114$. e) $\kappa = 100$. with $\bar{e} = 0.26832$.

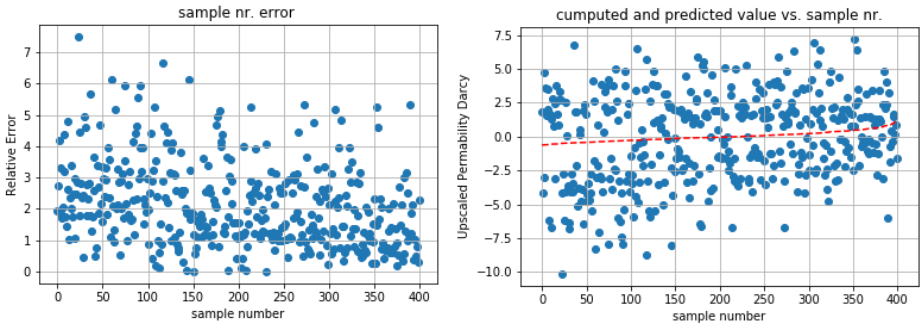

Figure 6.2 – Right plot showing the relative error in each sample for ordinary least squares with 400 samples. Left plot showing predicted value of a sample together with it's actual calculated value for ordinary least squares with 400 samples.

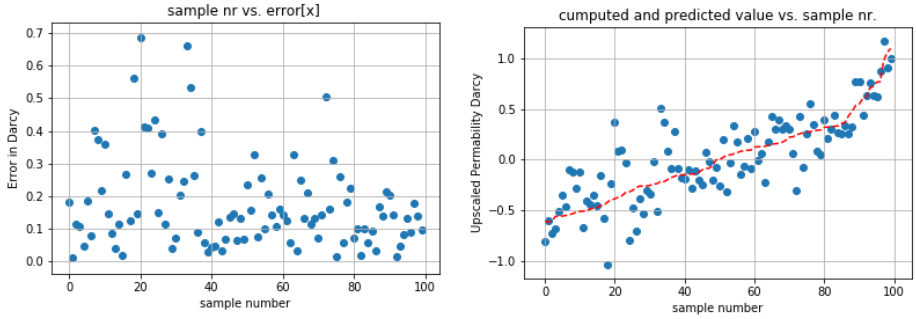

Figure 6.3 – Plot to the left showing the relative error in each sample for ordinary least squares and $\kappa = 10^{-2}$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

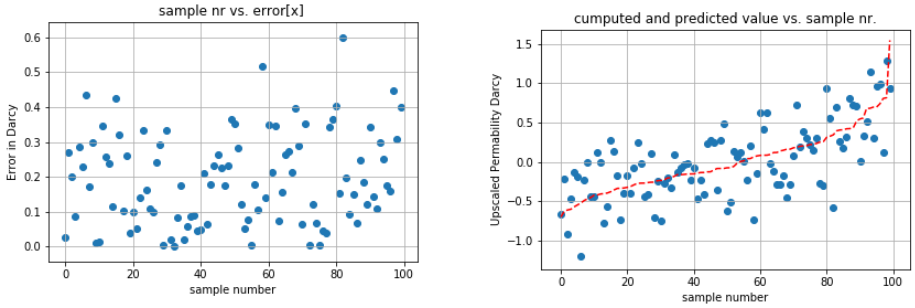

Figure 6.4 – Plot to the left showing the relative error in each sample for ordinary least squares and $\kappa = 10^{-1}$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

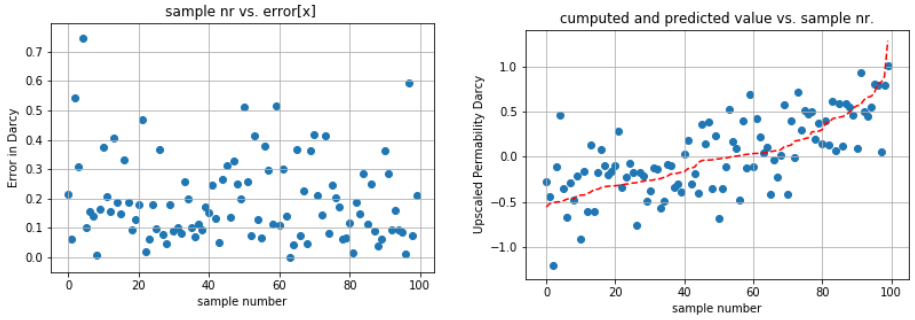

Figure 6.5 – Plot to the left showing the relative error in each sample for ordinary least squares and $\kappa = 1$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

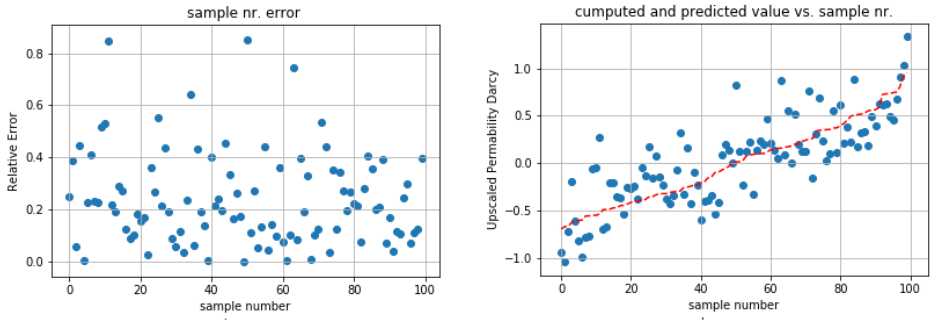

Figure 6.6 – Plot to the left showing the relative error in each sample for ordinary least squares and $\kappa = 10$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

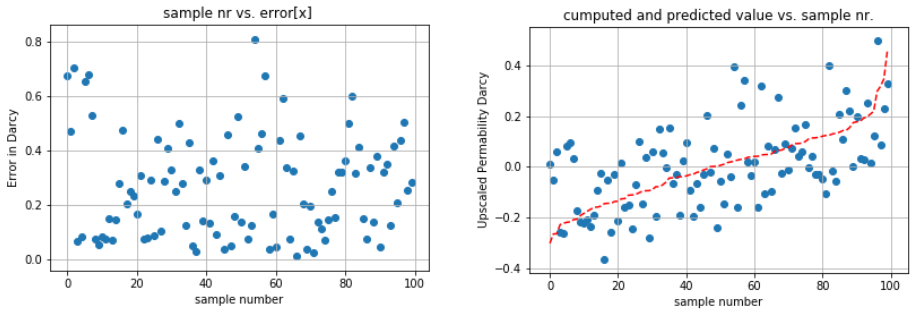

Figure 6.7 – Plot to the left showing the relative error in each sample for ordinary least squares and $\kappa = 100$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

6.3 Kernel Ridge regression

Kernel Ridge with a squared exponential kernel performed significantly better. With an average error of only $\bar{e} = 0.085$ it performs significantly better than ordinary least squares on the same cross-validating dataset with $\kappa = 10$. The machine has learned to upscale permeability and even in the worst scenario it is 31% off as seen in Figure 6.12. In order to achieve this accuracy, we optimized the value of the hyper-parameter in the algorithm. We found that trying exponentially growing sequences of α and γ was a practical method to identify good parameters. First, we did a very rough search to identify where the algorithm was performing well and finally did the full grid-search shown in Table 6.1. The parameter pair of $(\alpha, \gamma) = (2^{-7}, 2^{-11})$ gave us the highest cross-validation accuracy.

When investigating the robustness of the algorithm with respect to the statistical distribution we again have to look at the performance of the algorithm with different values of κ. The results are presented in Figure 6.9, 6.10, 6.11, 6.12 and 6.13. The histogram of the errors are compared side by side in Figure 6.8. When looking at this comparison we can see that the error is increasing with increasing values of κ. This is as expected given that the oscillation and general complexity of the permeability field is higher the higher the κ. However the way the kernel ridge regression has tough itself is different from the Ordinary least squares. For low values of κ, i.e a very smooth permeability field, as shown in Figure 6.9 the algorithm will predict the upscaled permeability with a very high accuracy, while Ordinary least squares still had a significant error for the same dataset Figure 6.3. Also for the most oscillating permeability fields created with $\kappa = 100$ as shown in Figure 6.13 we can clearly see that the algorithm has tough itself to ignore the extreme values and hence the method is bad at capturing the the extremes in such cases. A value of $\kappa = 100$ represents a extremely oscillating permeability field, so much so that it should not represent a typical real world permeability field.

$\alpha \mid \gamma$	2^{-14}	2^{-13}	2^{-12}	2^{-11}	2^{-10}	2^{-9}	2^{-8}
2^{-10}	$0,1$	$0,09$	$0,087$	$0,087$	$0,086$	$0,091$	$0,106$
2^{-9}	$0,105$	$0,093$	$0,087$	$0,086$	$0,086$	$0,091$	$0,106$
2^{-8}	$0,111$	$0,097$	$0,089$	$0,086$	$0,086$	$0,091$	$0,106$
2^{-7}	$0,116$	$0,103$	$0,091$	$\mathbf{0,085}$	$0,086$	$0,091$	$0,107$
2^{-6}	$0,119$	$0,109$	$0,095$	$0,087$	$0,086$	$0,091$	$0,107$

Table 6.1 – Table showing average error $\bar{e}$ for the purpose of trying to identify the best parameters in the kernel Ridge through Grid-search. We found that the pair $(\alpha, \gamma) = (2^{-7}, 2^{-11})$ gives the highest cross validation accuracy.

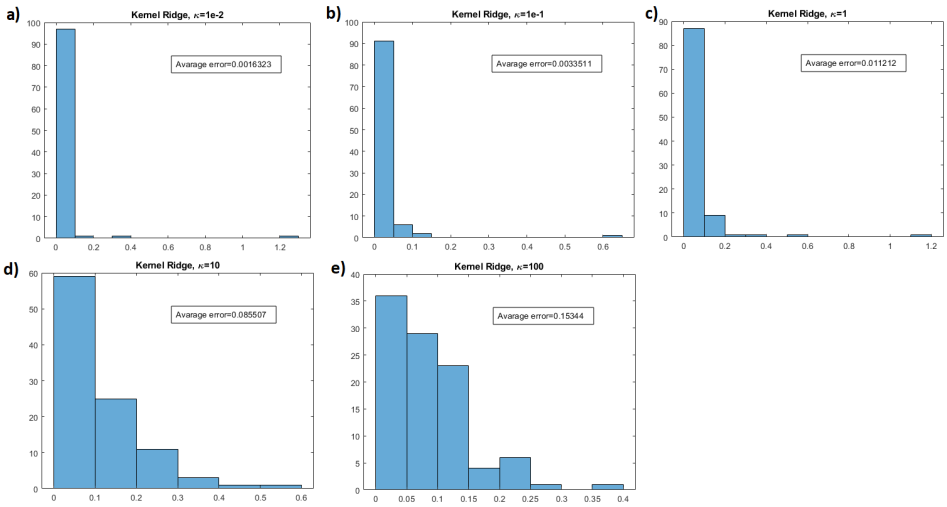

Figure 6.8 – Histogram plot of the relative error and also the computed average error for kernel Ridge and different κ. a) $\kappa = 10^{-2}$ with $\bar{e} = 0.0016323$. b) $\kappa = 10^{-1}$ with $\bar{e} = 0.0033511$. c) $\kappa = 1$ with $\bar{e} = 0.011212$. d) $\kappa = 10$ with $\bar{e} = 0.085507$. e) $\kappa = 100$. with $\bar{e} = 0.15344$.

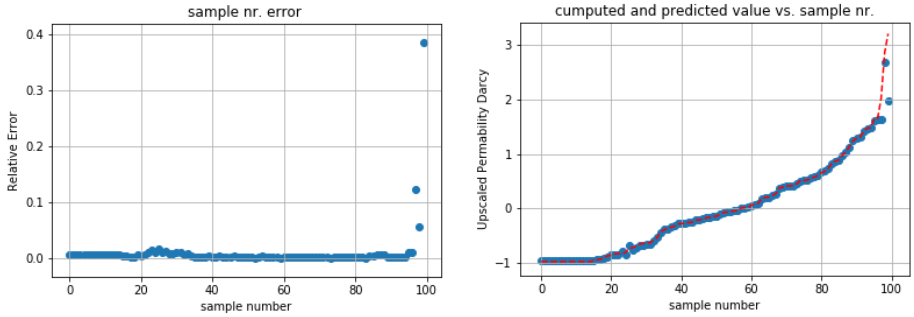

Figure 6.9 – Plot to the left showing the relative error in each sample for kernel Ridge regression and $\kappa = 10^{-2}$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

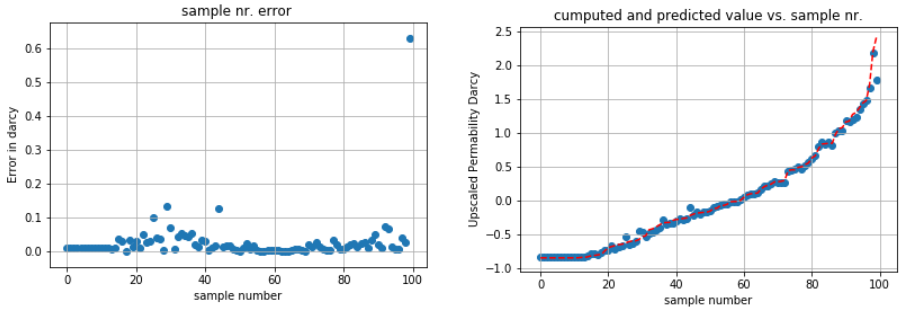

Figure 6.10 – Plot to the left showing the relative error in each sample for kernel Ridge regression and $\kappa = 10^{-1}$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

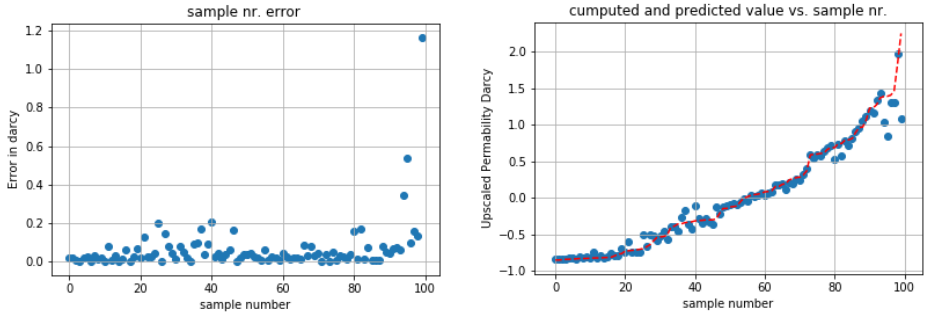

Figure 6.11 – Plot to the left showing the relative error in each sample for kernel Ridge regression and $\kappa = 1$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

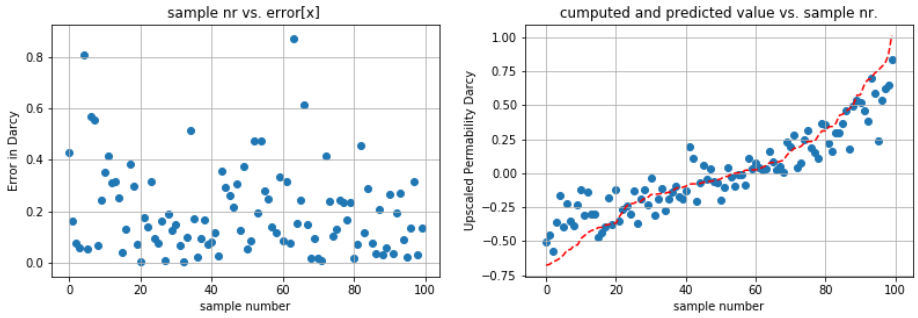

Figure 6.12 – Plot to the left showing the relative error in each sample for kernel Ridge regression and $\kappa = 10$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

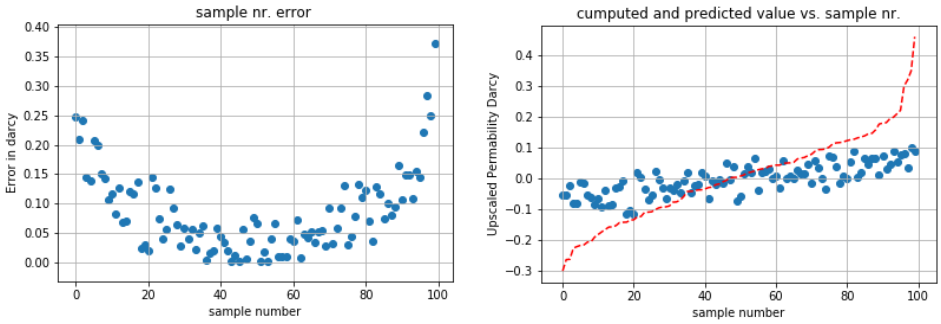

Figure 6.13 – Plot to the left showing the relative error in each sample for kernel Ridge regression and $\kappa = 100$. The plot to the right shows the same samples with predicted values (blue) together with calculated validation values (red).

Chapter 7

Conclusion

We have proposed that a machine learning algorithm, given a permeability distribution, is capable to capture the upscaled behaviour of the flow and encode it into a single coefficient, namely the upscaled permeability. Our results show that this is in fact the case and that both the ordinary least squares and kernel ridge regression algorithms have captured the underlying physics of the problem and taught themselves to upscale flow. This process however is quite error prone in the case of ordinary least squares, with a high average error of $\bar{e} = 0.24$. This leads us to the second goal of the work: the viability of replacing direct computations of upscaled permeabilities with learned machines. Here, a Ordinary least squares would not be useful due to the high error. In the case of the kernel ridge regression however, the algorithm shows some promising results with an error of $\bar{e} = 0.085$. This is however several orders of magnitude higher than the numerical error in a high-resolution finite volume simulation. To conclude whether or not the potential gains in faster computations can justify the increase in error, we need further research.

In future work we would like to try other types of machine learning algorithms and compare their performance to the ones we already have. We would also like to test the current machines on different permeability fields, in particular those with meandering patterns and/or semi-straight channelled structures, like the ones created in [2]. Especially as the kernel Ridge algorithm has shown some inability to capture the extremes in our artificial permeability fields with very high variance. There is also merit in trying it out on real world fluvial patterns from underground rivers or petroleum reservoirs.

Chapter 8

Bibliography

[1] SH Begg, RR Carter, P Dranfield, et al. Assigning effective values to simulator gridblock parameters for heterogeneous reservoirs. *SPE reservoir engineering*, 4(04), 1989.

[2] Shing Chan and Ahmed H Elsheikh. Parametrization and generation of geological models with generative adversarial networks. *arXiv preprint arXiv:1708.01810*, 2017.

[3] Noel Cressie. The origins of kriging. *Mathematical geology*, 22(3), 1990.

[4] Clayton V Deutsch and Libing Wang. Hierarchical object-based stochastic modeling of fluvial reservoirs. *Mathematical Geology*, 28(7), 1996.

[5] Emilien Dupont, Tuanfeng Zhang, Peter Tilke, Lin Liang, and William Bailey. Generating realistic geology conditioned on physical measurements with generative adversarial networks. *arXiv preprint arXiv:1802.03065*, 2018.

[6] Louis J Durlofsky. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water resources research*, 27(5), 1991.

[7] Louis J Durlofsky. Upscaling and gridding of fine scale geological models for flow simulation. In *8th International Forum on Reservoir Simulation Iles Borromees, Stresa, Italy*, volume 2024, 2005.

[8] A Forbes. The worlds 25 biggest oil companies, 2012.

[9] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[10] AG Journel, C Deutsch, AJ Desbarats, et al. Power averaging for block effective permeability. In *SPE California Regional Meeting*. Society of Petroleum Engineers, 1986.

[11] Derek Kane. *Data Science - Part XII - Ridge Regression, LASSO, and Elastic Nets*. 2015. https://www.slideshare.net/DerekKane/presentations [Online; accessed 1-May-2018].

[12] Torbjørn Kindingstad and Fredrik Hagemann. *Norges oljehistorie*. Wigestrand, 2002.

[13] Knut-Andreas Lie. An introduction to reservoir simulation using MATLAB: user guide for the Matlab Reservoir Simulation Toolbox (MRST)., 2016. www.mrst.no.

[14] Jane McIntosh. *The Ancient Indus Valley: New Perspectives*. ABC-CLIO, 2008.

[15] Lukas Mosser, Olivier Dubrule, and Martin J Blunt. Reconstruction of three-dimensional porous media using generative adversarial neural networks. *Physical Review E*, 96(4), 2017.

[16] Mohamad T Musavi, Wahid Ahmed, Khue Hiang Chan, Kathleen B Faris, and Donald M Hummels. On the training of radial basis function classifiers. *Neural networks*, 5(4), 1992.

[17] Petroleum-Geology-Forums. *How to Upscale Permeability*. 2018. http://www.epgeology.com/static-modeling-f39/how-upscale-permeability-t6045.html [Online; accessed 1-May-2018].

[18] Robert L Pirog. The role of national oil companies in the international oil market. Congressional Research Service, Library of Congress, 2007.

[19] Roy Shepherd. *What are Fossil Fuels?* 2018. http://www.discoveringfossils.co.uk/fossilfuels.htm [Online; accessed 1-May-2018].

[20] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Academic Press, 2008.

[21] Xian-Huan Wen and J Jaime Gómez-Hernández. Upscaling hydraulic conductivities in heterogeneous media: An overview. *Journal of Hydrology*, 183(1-2), 1996.

[22] Xiao-Hui Wu, Y Efendiev, and Thomas Y Hou. Analysis of upscaling absolute permeability. *Discrete and Continuous Dynamical Systems Series B*, 2(2), 2002.