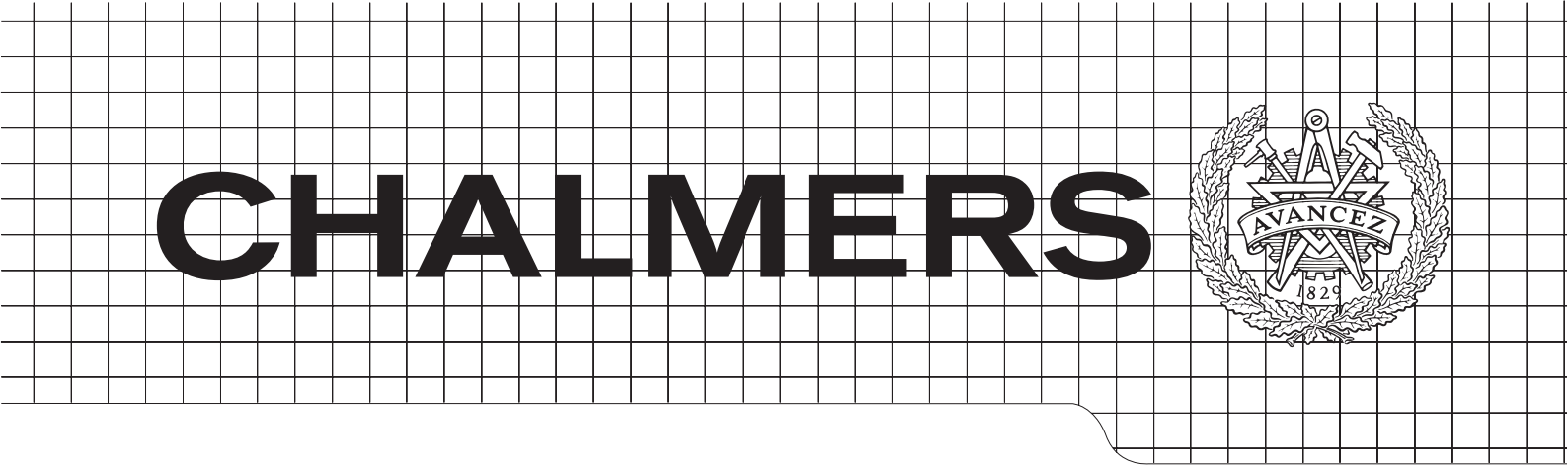


Permeability prediction using Support vector machines

Master's thesis at Chalmers University of Technology and University of Gothenburg

DAMIANO OGNISSANTI

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2013

Abstract

This thesis explores the possibility of calculating the permeability of materials with regression based of classification software instead of using famous physics formulae, like the Kozeny-Carman equation. The reason for this is that regarding permeability, no universal and exact formula has been discovered to date; the existing formulae depend on the constitution of the materials. The chosen model is based on classification from support vector machines. This classification algorithm was chosen because support vector machines have a history of showing accuracy comparable to those of other methods in recognizing various data and because they have a rigorous mathematical base. The thesis consists of a theoretical part and an applied one, where the first describe the basis on which the results rely and the second explains how certain parameters are calculated and used for the classification algorithm to perform well. It is shown that the classification algorithm surpasses the famous Kozeny-Carman equation in terms of accuracy of the calculations for fibre structures. It is also shown that it suffices to extract parameters from two dimensional images of the three dimensional structures to gain equal precision as if the whole three dimensional structure is taken into account. This raises hope that microscopy images can be used to calculate the permeability of materials. Finally it is shown that the content of the training set is more important than its size for the support vector machine to perform well.

Acknowledgements

בעזרת השם

I would like to thank my supervisor Tobias Gebäck and Alexei Heintz at the department of Mathematical Sciences at Chalmers University of Technology and the University of Gothenburg for their advices, suggestions and supervision. I would also like to thank Katarina Logg at the Swedish Institute for Food and Biotechnology for letting us use their software and data for calculations. Finally I would like to express my love for my family who has always given me their deepest support and guidance.

Damiano Ognissanti, Gothenburg May 4, 2014

Contents

Chapter 1

Introduction

Permeability is a measure of the ability of a material to allow fluids to pass through it. The notion of permeability was developed much thanks to the works of Henry Darcy who, with empirical studies, put forth the law named after him:

$$u = -\frac{\kappa \Delta p}{\mu L}$$

where u is the fluid velocity[1], κ is the permeability, μ the viscosity[2] and Δp is the pressure difference.

Permeability depends only on the properties of the material and not of the fluid and a formula commonly used for calculating the permeability of a material is the Kozeny-Carman equation. The equation was first put forward by Josef Kozeny in 1927 and then two times modified by Philip C. Carman, once in 1937 and later in 1956 [7] and in one of its simpler forms it looks like the following:

$$\kappa = \psi \frac{\phi^3}{T^2 S^2}$$

where κ is permeability, ϕ is the porosity of the bed, S is the specific surface, T is the Tortuosity of the material and ψ is a constant acquired by empirical studies of the given material in a laboratory or by curve fitting the formula with a set of measured permeabilities. It is a cumbersome process to acquire the constant and results in a formula which may not be as precise as required, e.g. it is approximately valid for sands but not for clays [21].

This report will study the possibilities of predicting the permeability of a material by using a Support vector machine (SVM).

A support vector machine is a computer program with a learning algorithm used for classification and regression analysis of given data.

[1] The discharge per area.

[2] A measure of the fluids resistance to gradual deformation.

The original algorithm was developed by Vladimir N. Vapnik and became famous in the '90s when it showed accuracy comparable to other methods in recognizing handwritten text [5]. The today standard version of the algorithm (called soft margin SVM) was derived by Vladimir N. Vapnik. and Corinna Cortes in 1995 [22].

Some reasons for using SVM when dealing with classifying data is that training the program is easy and that it scales well to high dimensional data. The trade-off between error and classifier complexity can also be controlled exactly.
A weakness is that we need to find a good kernel function [14] but a good first step is to choose the radial basis kernel and use cross-validation to find certain parameters [9].

1.1 Restrictions

There are several equations for calculating permeability, as seen in Chapter 3, but only the Kozeny-Carman equation was used to compare accuracy with the Support vector machine. There are also several versions of the Kozeny-Carman equation but only the simple version presented above was used. These choices were made since the report was focusing on the accuracy of the Support vector machine and not of the permeability equations (although at least one comparison had to be made to have a reference point on how accurate we can hope to be).

There are several libraries for performing Support vector machine calculations and libsvm by Chih-Chung Chang and Chih-Jen Lin was chosen [8]. This since the library is available to around 20 programming languages, it is well documented and has a great tutorial [9] on how to start working. The library itself also has a multitude of options and is able to perform five types of Support vector machine calculations with four built in kernel functions.

MATLAB was chosen as programming language since it is widely used [12] and has great toolboxes for optimization and curve fitting. No comparison was made between the calculation speed of the libsvm package implemented in different languages, since the average calculations took less than half a minute which was considered reasonable.

1.2 Material

The workload was distributed between studying literature and applying the theory on given data together with analysis of the results.

The literature studied for the theoretical part of the thesis was Pattern

Recognition and Machine Learning by C.M. Bishop [4], The Elements of Statistical Learning by Trevor Hastie et.al. [13] among others [11], [18].

The data for the fiber structures were acquired from Katarina Logg at SIK (Swedish Institute for Food and Biotechnology) and were created with a software called Geodict [23]. The data portrayed three dimensional fibre structures and from these structures two dimensional analogs were extracted.

First the SVM was trained with information computed from the three dimensional structures and was refined until satisfaction was achieved. Afterwards corresponding information was calculated from the two dimensional equivalents. Henceforth when required to distinguish the two SVMs the SVM with three dimensional data will be known as the 3D SVM and the two dimensional alternative as the 2D SVM.

1.2.1 3D Structures

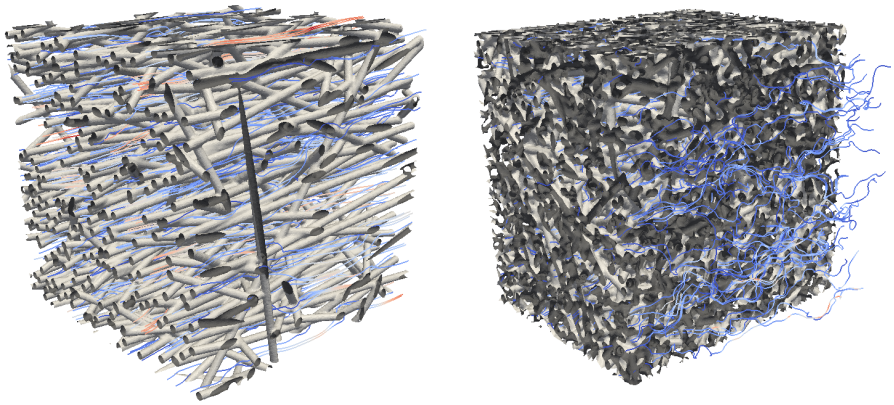
The structures were created with different degrees of anisotropy. Anisotropic means that the fibres have a greater probability of being aligned into one direction than the rest. The opposite of anisotropy is isotropy which means that the fibres are distributed evenly in all directions (so that it looks almost the same from every direction). The measure of anisotropy is called DT in this article and it consists of a vector of three elements: the probability that the fibres are distributed in respective direction. In the structures acquired from Geodict the probability of two of the directions were chosen to be the same, so that only one parameter was required to represent anisotropy. Thus a value of $\frac{1}{3}$ meant that the structure was isotropic and a different value meant it was anisotropic (in one direction or the other) (Fig. 1.1).

From these structures a number of parameters were extracted. First two parameters which could be used with the the Kozeny-Carman equation were obtained, namely the SVP value of the structure: a measure on the fraction of total volume consisting of solid material (the complement of the void ratio) and also the specific surface area (the area divided by the volume). Second Geodict performed simulations to find out the diameters of the biggest spheres which could penetrate the material from different starting points, and also the length of the paths these spheres took. Finally the program found out the pore-size distribution, calculated from taking the volume of the biggest spheres which could fit in the void of the material at different positions. The data was in some case multidimensional. For an overview of the data acquired and information on how it was treated see Chapter 4 and especially Table 4.1 for reference.

The permeabilities of the structures were calculated using the Lattice

Boltzmann method with a software called Gesualdo - a program package for modeling flow and diffusion through heterogeneous biomaterials. The program was written by Tobias Gebäck and Alexei Heintz at the department of Mathematical Sciences at Chalmers University of Technology and the University of Gothenburg.

The library used for performing the SVM calculations in the computer was libsvm [8] and it was implemented with MATLAB.



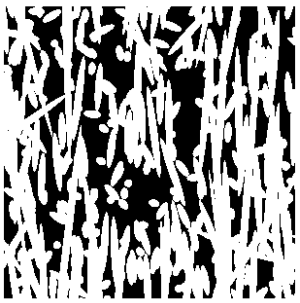
(a) An anisotropic fibre structure with 15% SVP.

(b) An isotropic fibre structure with 60% SVP.

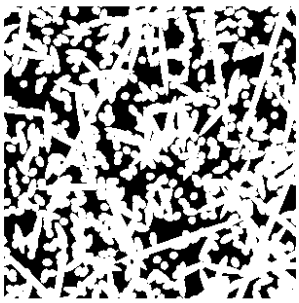
Figure 1.1: One can easily see the difference between the anisotropic and the isotropic structures in this case. The blue strings are streamlines; they represent the path of a liquid passing through the material.

1.2.2 2D Structures

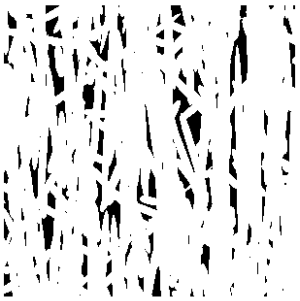
Thirty images were extracted for each 3D structure, 10 for each direction (Fig. 1.2). The white pixels (pixels with value 1) represent fibre and black pixels (pixels with value 0) represent pores. To make it realistic four sets of images were extracted. The first set contained images of depth 1, that is, perfect 2D slices of the 3D material (Fig. 1.2(a) and 1.2(b)). The other sets contained the same images but with depth 3, 7 and 10 respectively. Depth was created by just performing a logical OR operation between the pixels of consecutive images so that the final image would contain every fibre pixel from each image. As seen in Fig. 1.2(c) and 1.2(d) the depth of the images will greatly affect the DT and SVP values. From these images some of the parameters mentioned above were approximated (for reference see Chapter 4.2).



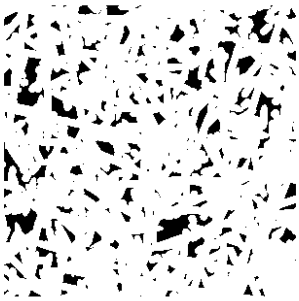
(a) An anisotropic fibre structure with 60% SVP with depth 1.



(b) An isotropic fibre structure with 60% SVP with depth 1.



(c) An anisotropic fibre structure with 60% SVP with depth 10.



(d) An isotropic fibre structure with 60% SVP with depth 10.

Figure 1.2: In the first row of images one can easily see that one image is anisotropic and the other isotropic and that they have different SVP values. In the second row it is harder distinguish the degree of anisotropy from the first image since it is almost all white. The SVP value will also be miscalculated.

Chapter 2

Support vector machines

When dealing with mathematical theory it is often useful to first study simple applications of the theory and gradually making the applications more general until we reach the theoretical level. This way confusion is greatly avoided since the material is allowed to get absorbed by the reader in a pleasant pace. Because of this the following pages will describe a simple problem solved by a SVM and gradually the exact definitions and theorems will appear.

2.1 Introductory problem

Consider the following problem: a set of three dimensional data points is given where the first dimension is in binary form and the other two are real numbers and the task is to search for a correlation between the real data and the binary one. A way of representing this data is by drawing it in a two dimensional graph and to let the binary data be represented by the shape of the point and the real to represent the position (Fig. 2.1(a)).

When the data is represented in this way it is very tempting to draw a line which splits the plane in two and to say that the binary value of the data is determined by whether the data is positioned to the right or left to the line. We begin with splitting our set of data into a smaller set of training data; this is done so that when we have drawn our line we can study the lines behaviour with respect to the whole set and see if it divides the sets well (Fig. 2.1(b)). We now define our problem:

Find a, b, c such that $ax + by \geqslant c$ for red dots, and $ax + by \leqslant c$ for blue dots.

Our line can now be used to predict the binary value of the data given the the position of the point. However, without any specific rules two or more

lines could work well with dividing the given data, so which line should be chosen (Fig. 2.2)?

We want the line to be as far away from both sets of data as possible, so let d_1 define the minimal distance[1] between our line and the nearest red dot[2] and d_2 between the line and the nearest blue dot. Let $d_1 + d_2 = \varepsilon$, then the line which optimally cuts the plane in half can now be defined as the line with $d_1 = \frac{\varepsilon}{2}$, $d_2 = \frac{\varepsilon}{2}$ (Fig. 2.3(a)).

After the line is in place we will assume any new dot appearing to the right of the line to be blue and to the left red (Fig. 2.3(b)). We have to remember though that there are some limitations with this model. Since it is not always possible to find a line which splits the two sets perfectly we sometimes have to make sacrifices and miss-classify some dots to make the method work. In Fig. 2.4(a) one red dot was misclassified since it would be impossible to split the sets perfectly. We could assume that the dot is there due to some calculation or measurement error, but we have to be more careful if they are too many strange dots. We also have to choose training data with caution. Choosing a bad training set may result in drawing a line which does not split the two sets optimally at all, as seen in Fig. 2.4(b). We just predict 273 of our 300 points right and we can make the accuracy even lower by choosing the sets carefully.

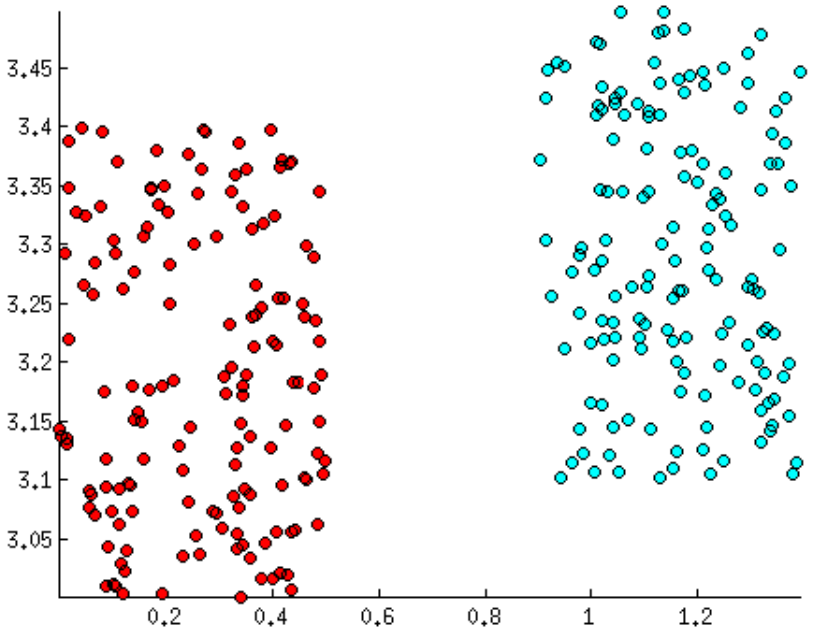
If we want to extend our capabilities to even handle data which cannot be split by a line we can just perform a transformation which makes our data linearly divisible. Take the following example: we have data which seems to be placed in a circle, then we just perform a transformation from Cartesian to polar coordinates after which we can find our splitting line easily (Fig. 2.5). It is not always easy to find a transformation which makes the data linearly divisible, but more on that later.

We can also use support vector machines for linear regression (Fig. 2.6). Assume we have data we want to approximate by a line. We define the perfect line to be a line such that if we create a tube with our line in the middle and of width 2ε, we have minimized the distances between the tube and the data points which lie outside the tube. The data points which lie inside the tube are not considered.

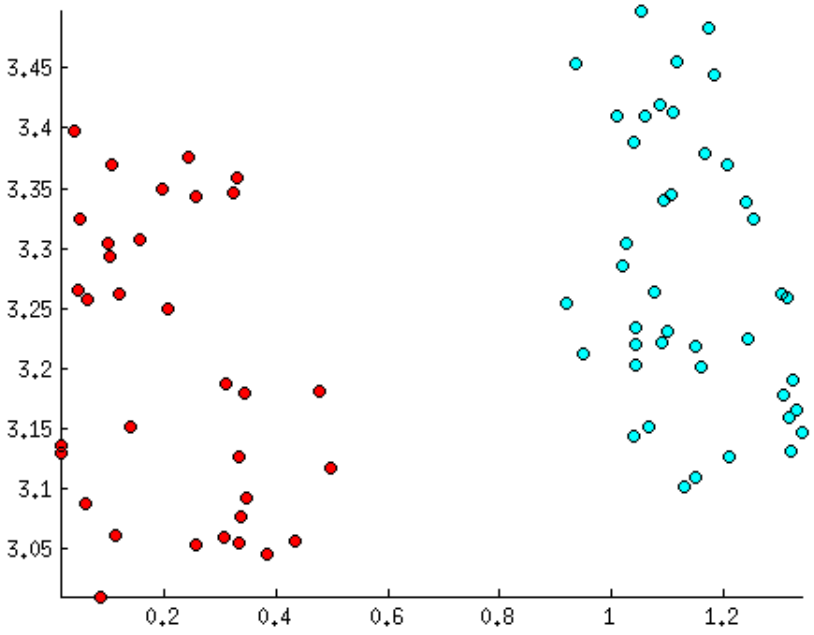
In the figure the red dots represent data which lie outside the tube. The smaller the distance between the tube and the red dots, the better we say our approximation is.

[1] With distance we obviously mean euclidean distance.

[2] Sometimes we have to study the distance between the line and several close dots. More on that later.



(a) Our original set.



(b) Our training set.

Figure 2.1: We start with splitting our set.

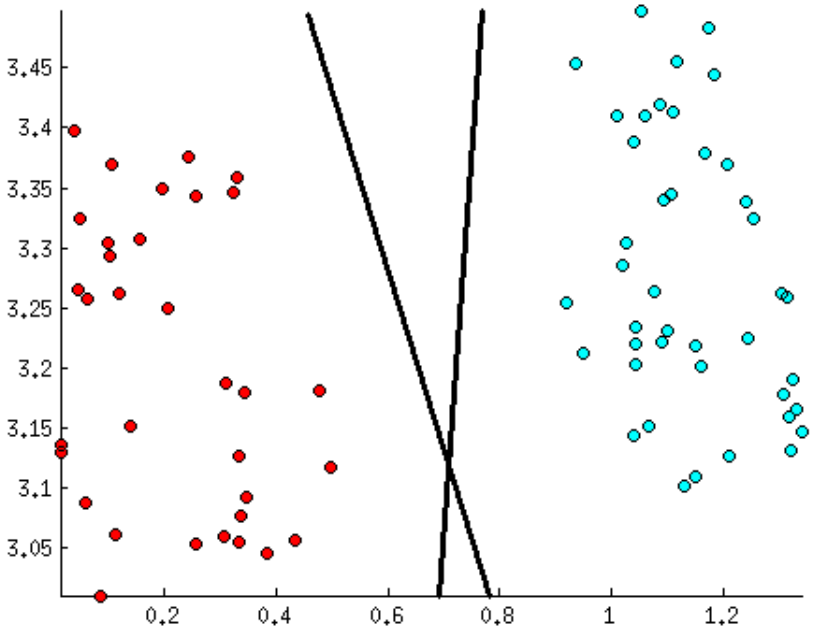
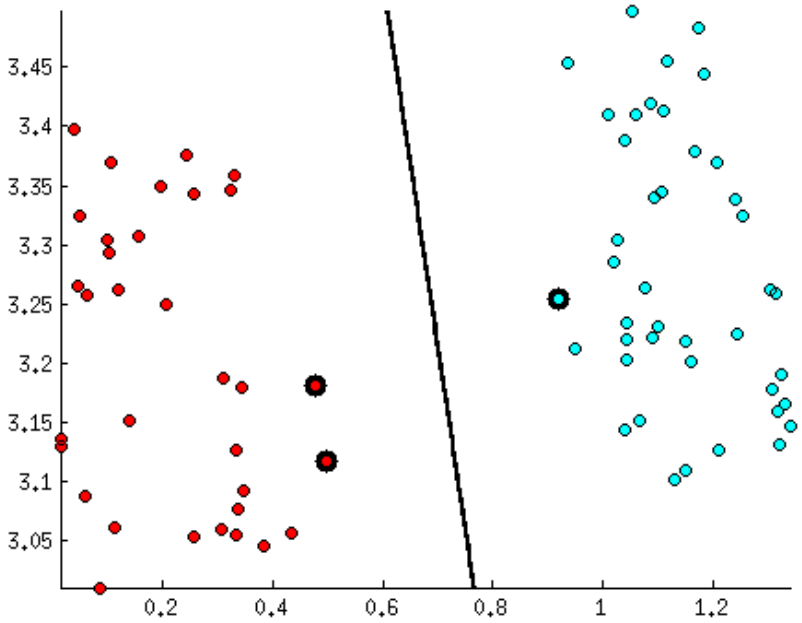
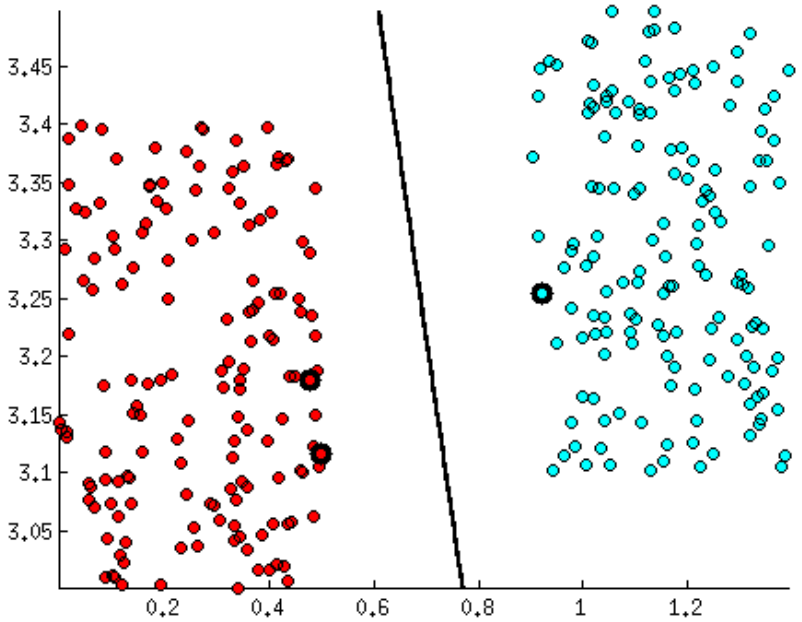


Figure 2.2: Both lines divide the plane well.

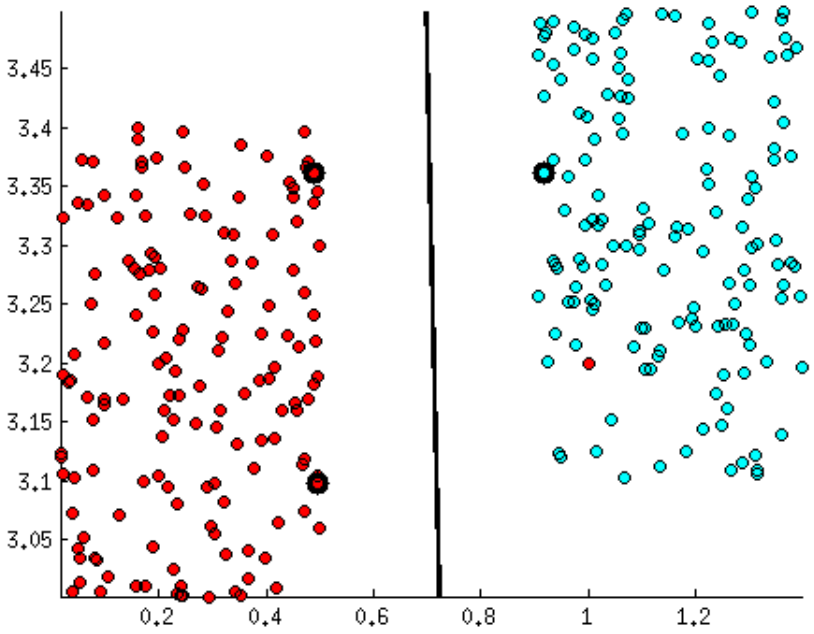


(a) The optimal line splitting the training set.

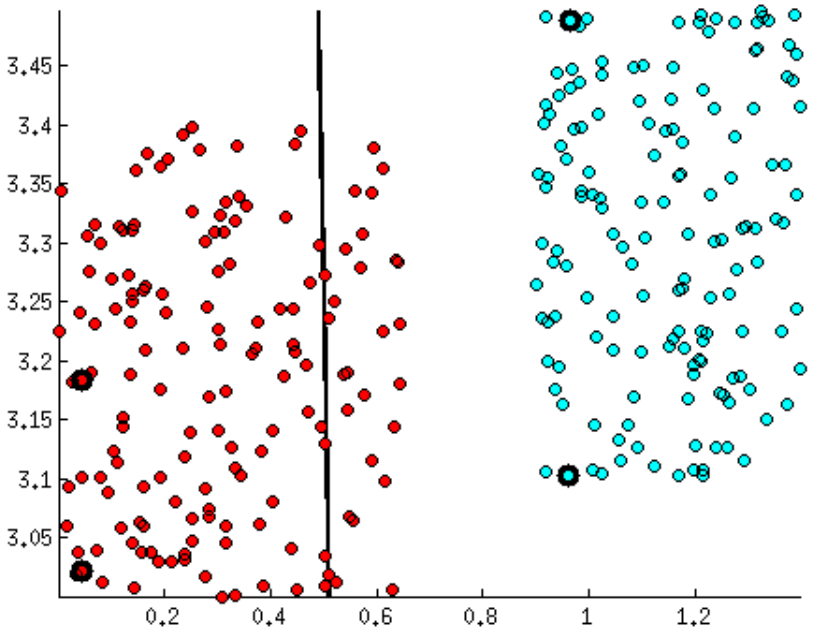


(b) The optimal line splitting the whole set.

Figure 2.3: Our solution. The encircled dots are the ones from the training set closest to our line.



(a) One misclassified dot.



(b) Many misclassified dots.

Figure 2.4: Problems which may occur.

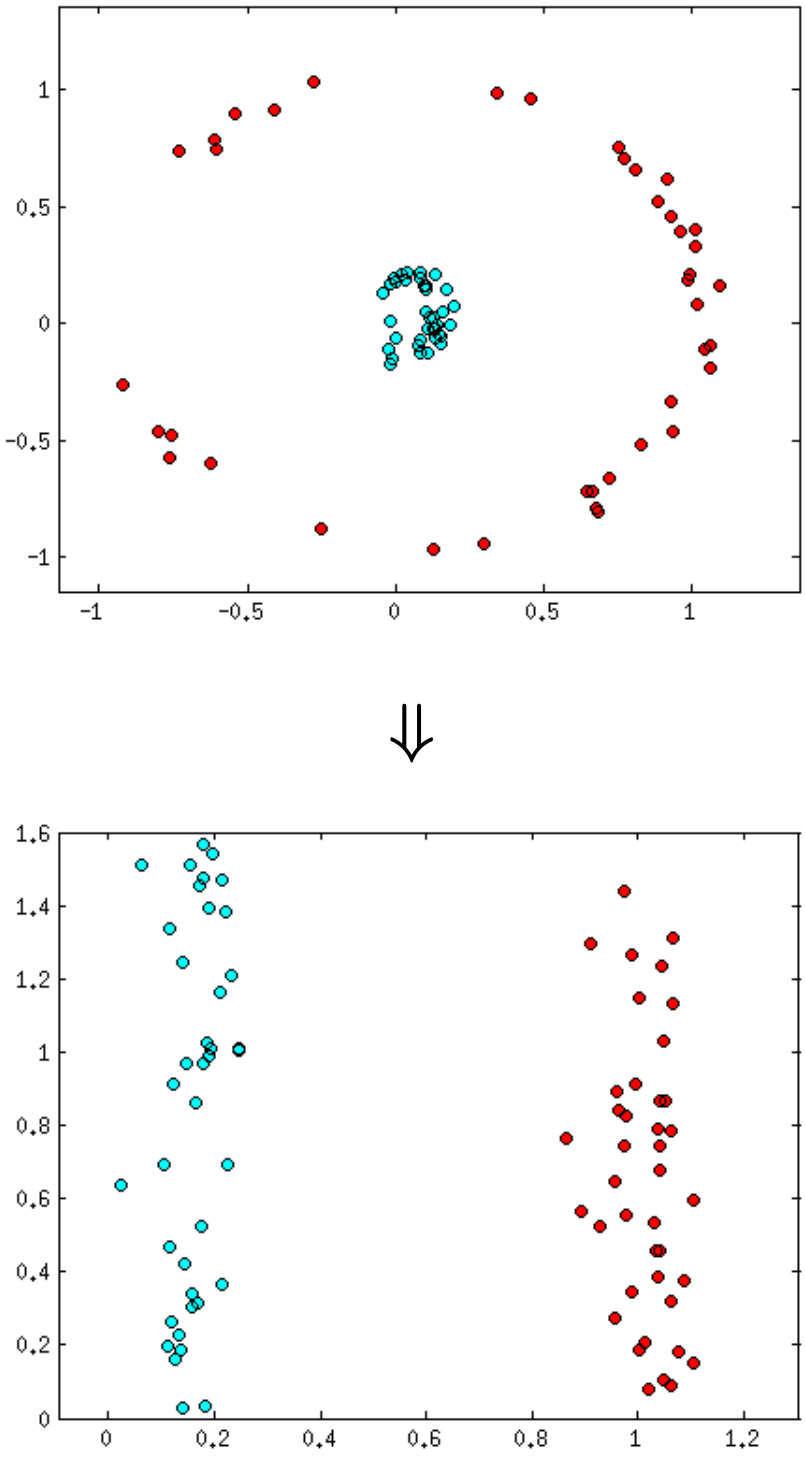


Figure 2.5: We transform data to make it simpler to divide.

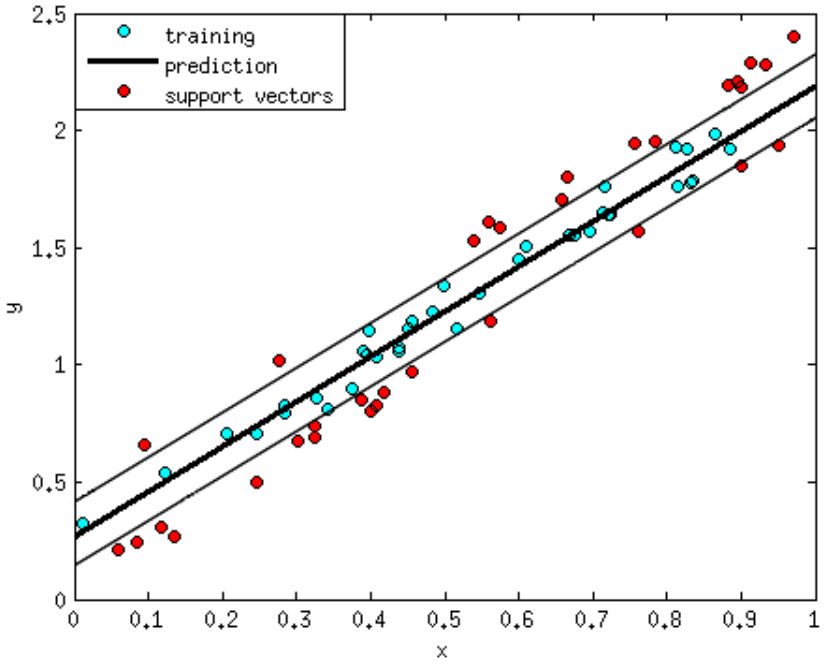


Figure 2.6: The red data points lie outside the tube and are thus considered, and the blue ones are inside the tube and therefore neglected.

2.2 Optimal Hyperplanes

The Support vector algorithm introduced by Vapnik in 1995 was divided into two parts, one part concerning finding the optimal hyperplane and one introducing a notion of soft margins, so the same approach will be dealt here [22].
We begin by defining the set of training patterns (the subset of red and blue dots in Chapter 2.1)

$$T = \{(y_k, \boldsymbol{x_k}) : k = 1, ..., l;\ y_k \in \{-1, 1\}\ \forall k;\ x_k \in \mathbb{R}^n\}$$

Let us also define subsets of T containing just data with the same value of y_k (sets with dots of the same colour).

$$T_+ = \{(y_k, \boldsymbol{x_k}) \in T : y_k = 1\}$$

$$T_- = \{(y_k, \boldsymbol{x_k}) \in T : y_k = -1\}$$

T is said to be linearly separable if there exists a vector $\boldsymbol{w}$ and a scalar b such that the following inequality is valid for all elements in T:

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geqslant 1, \quad i = 1, ..., l$$

The optimal hyperplane

$$\boldsymbol{w_0} \cdot \boldsymbol{x} + b_0 = 0$$

will be the uniquely defined plane which separates the elements of T with respect to their y_i value with a maximal margin. This hyperplane will determine the direction $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$ where the distance between the closest points from T_+ and T_- is maximal. The optimal distance will be denoted $\delta(\boldsymbol{w}, b)$ and is given by:

$$\delta(\boldsymbol{w}, b) = \min_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_+\}} \frac{(\boldsymbol{x} - b) \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} - \max_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_-\}} \frac{(\boldsymbol{x} - b) \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} =$$

$$\cancel{-\frac{b\boldsymbol{w}}{\|\boldsymbol{w}\|}} + \min_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_+\}} \frac{\boldsymbol{x} \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} + \cancel{\frac{b\boldsymbol{w}}{\|\boldsymbol{w}\|}} - \max_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_-\}} \frac{\boldsymbol{x} \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} =$$

$$\min_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_+\}} \frac{\boldsymbol{x} \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} - \max_{\{\boldsymbol{x}:(y,\boldsymbol{x})\in T_-\}} \frac{\boldsymbol{x} \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|}$$

The hyperplane does therefore not depend on b but solely on $\boldsymbol{w}$. The optimal hyperplane has distance:

$$\delta(\boldsymbol{w_0}, b_0) = \frac{1 - (-1)}{\|w_0\|} = \frac{2}{\sqrt{\boldsymbol{w_0}^T \boldsymbol{w_0}}} = \frac{2}{\|\boldsymbol{w_0}\|}$$

This can be written as the following optimization problem (Fig. 2.3):

$$\begin{aligned} &\underset{w,b}{\text{minimize}} && \|w\| \\ &\text{subject to} && y_i(w \cdot x_i + b) \geqslant 1, \quad i = 1, ..., l \end{aligned}$$

This optimization problem is hard to solve, since it involves a square root (in the norm), but fortunately the square root is a monotonically increasing function so it is possible to substitute $\|w\|$ with $\frac{1}{2}\|w\|^2$ without affecting the optimal solution[3].

The vectors on which the data points with the smallest distance to the optimal hyperplane lie is called the support vectors and are shown as encircled dots in Fig. 2.3(a).

2.3 Soft Margins

Assume there is no hyperplane to split our data perfectly with $y_i = 1$ on one side and $y_i = -1$ on the other, then we want a hyperplane which splits our data *as well as possible* while still maximizing the distance to the nearest well split example. So we need to define what we mean with *as well as possible*.

Let us start with defining a slack variable ξ_i which is a measure on the degree of misclassification of our data. Let our objective function be increased by a function which penalizes non-zero ξ_i so that the optimization will balance between a large margin and a small error penalty. Graphically ξ_i represents the distances between point x_i and the support vector with the same y value[4].

In Fig. 2.4(a) this would be the smallest distance between the red dot on the right and the line between the two encircled dots on the left. If our penalty function is linear we get the following (Fig. 2.4):

$$\begin{aligned} &\underset{w,b,\xi}{\text{minimize}} && \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \xi_i \\ &\text{subject to} && y_i(w \cdot x_i + b) \geqslant 1 - \xi_i, \quad i = 1, ..., l \\ & && \xi_i \geqslant 0, \quad i = 1, ..., l \end{aligned} \tag{2.1}$$

where C is a constant. If the data is classified well by the above methods we say that it is linearly classifiable.

[3]The constant $\frac{1}{2}$ is just there to simplify calculations and does not affect the result.

[4]Notice that if $\xi_i > 1$ the data point is misclassified.

2.4 Nonlinear classification

Assume the set S contains data that cannot be split well with hyperplanes, then we say that the data in S is nonlinearly classifiable. Since our method with optimal hyperplanes only splits linearly classifiable data well we must transform the elements in S to become just that. So we want to find a bijective mapping ϕ which maps all elements in S to the set S^* where the elements of S^* are linearly classifiable. Often it is too hard to find the explicit mapping ϕ and in these cases we perform the famous *kernel trick* [17] which is based on *Mercer's Theorem*:

Def. Let Λ be a compact, non empty subset of $\mathbb{R}^n$. A function $K : \Lambda \times \Lambda \to \mathbb{R}$ is said to be a *kernel function* if it is continuous, symmetric and positive semi-definite.

Thm. *Any kernel function* $K(\mathbf{x}, \mathbf{y})$ *can be expressed as an inner product* $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ *for some mapping* ϕ *in a high-dimensional space.*

Therefore we could either know the explicit mapping ϕ and take the dot product to perform the transformation or we can take a kernel and use it right away without knowing what ϕ looks like. The most used Kernel functions are shown in Table 2.1.

Name	Function
Homogeneous polynomial	$K(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j})^d$
Inhomogeneous polynomial	$K(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$
Gaussian radial basis function	$K(\mathbf{x_i}, \mathbf{x_j}) = e^{-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2}$
Hyperbolic tangent	$K(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa \mathbf{x_i} \cdot \mathbf{x_j} + c)$

Table 2.1: Common kernel functions.

To add non-linearity to problem 2.1 we just exchange $\mathbf{x_i}$ with $\phi(\mathbf{x_i})$ where ϕ is a mapping as discussed in this Section.

We will now use orthogonal projection to decompose $\boldsymbol{w}$ into a sum of two functions $\boldsymbol{w} = \mathbf{u} + \boldsymbol{v}$, where $\mathbf{u} \in Span\{\phi(\mathbf{x}_1), ..., \phi(\mathbf{x}_l)\}$ and $\boldsymbol{v}$ in its orthogonal complement (so $\mathbf{u} \cdot \boldsymbol{v} = 0$). That is we can express the solution to problem 2.1 as:

$$\boldsymbol{w} = \sum_{i=1}^{l} \beta_i \phi(\mathbf{x}_i) + v$$

With this fact along with Mercer's theorem we have

$$\boldsymbol{w} \cdot \phi(\boldsymbol{x}) = \sum_{i=1}^{l} \beta_i \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}) = \sum_{i=1}^{l} \beta_i K(\boldsymbol{x}_i, \boldsymbol{x})$$

So we do not need to know what ϕ looks like as long as we know K. Introducing this to our minimization problem 2.1 we get the following:

$$\begin{aligned} \underset{\boldsymbol{w},b,\boldsymbol{\beta},\boldsymbol{\xi}}{\text{minimize}} \quad & \sum_{i=1}^{l}\sum_{j=1}^{l} \beta_i \beta_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + C \sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & y_i(\sum_{i=1}^{l} \beta_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b) \geqslant 1 - \xi_i, \quad i = 1, ..., l \\ & \xi_i \geqslant 0, \quad i = 1, ..., l \end{aligned}$$

As mentioned before, a commonly used kernel for SVM is the radial basis kernel. This is due to that the radial basis function maps the data into an infinite dimensional Hilbert space, which allows us to even classify data which is far from linear. The proof of this is simple:

$$e^{(-\gamma||\boldsymbol{x}-\boldsymbol{y}||^2)} = e^{(-\gamma(\boldsymbol{x}-\boldsymbol{y})^2)} = e^{(-\gamma(\boldsymbol{x}^2 - 2\boldsymbol{x}\cdot\boldsymbol{y} + \boldsymbol{y}^2))} =$$

$$= e^{-\gamma \boldsymbol{x}^2} e^{-\gamma \boldsymbol{y}^2} \sum_{i=1}^{\infty} \frac{\gamma^i 2^i \boldsymbol{x}^i \boldsymbol{y}^i}{i!} = e^{-\gamma \boldsymbol{x}^2} [1, \sqrt{2\gamma}\boldsymbol{x}, ...] \cdot e^{-\gamma \boldsymbol{y}^2} [1, \sqrt{2\gamma}\boldsymbol{y}, ...] =$$

$$= \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle .$$

It is easy to see that we have mapped the data into an infinite dimensional Hilbert space.

2.5 Regression

First let $\mathbf{I} \subset \mathbb{R}^n$ denote the space of input patterns. With Support vector regression the goal is to find an as flat as possible function $f(\boldsymbol{x})$ for the training points $(y_i, \boldsymbol{x}_i)$, $\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$, that has at most ϵ deviation from the actual measured targets y_i for all training data. In the linear case we have $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} + b$, $\boldsymbol{w} \in \mathbf{I}$, $b \in \mathbb{R}$ and flatness here means a small (euclidean) norm on $\boldsymbol{w}$. Formally this is written:

$$\begin{aligned} \underset{w,b,\epsilon}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & (w \cdot x_i + b) - y_i \leqslant \epsilon, \quad i = 1, ..., l \\ & y_i - (w \cdot x_i + b) \leqslant \epsilon, \quad i = 1, ..., l \\ & \epsilon \geqslant 0 \end{aligned}$$

In the same manner as with problem 2.1 we introduce slack variables ξ_i, ξ_i^* to make the solution feasible. Similarly they represent the smallest distance between a data point and its support vector. Mathematically this is written (Fig. 2.6):

$$\begin{aligned} \underset{w,b,\xi,\xi^*,\epsilon}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{subject to} \quad & (w \cdot x_i + b) - y_i \leqslant \epsilon + \xi_i, \quad i = 1, ..., l \\ & y_i - (w \cdot x_i + b) \leqslant \epsilon + \xi_i^*, \quad i = 1, ..., l \\ & \xi_i, \xi_i^* \geqslant 0, \quad i = 1, ..., l, \epsilon \geqslant 0 \end{aligned}$$

Where $C > 0$ determines the trade off between flatness of the function f and the amount to which deviations larger than ϵ are tolerated. If the training point is at most at distance ϵ from $f(x)$ (inside the tube in Fig. 2.6) then ξ_i or ξ_i^* equals zero .

From [20] we can also introduce an additional parameter $\nu \in (0, 1]$ which is an upper bound on the fraction of training errors allowed and a lower bound on the fraction of support vectors. The formal definition of the problem in this case becomes:

$$\begin{aligned} \underset{w,b,\xi,\xi^*,\epsilon}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 + C(\nu\epsilon + \frac{1}{l}\sum_{i=1}^{l}(\xi_i + \xi_i^*)) \\ \text{subject to} \quad & (w \cdot x_i + b) - y_i \leqslant \epsilon + \xi_i, \quad i = 1, ..., l \\ & y_i - (w \cdot x_i + b) \leqslant \epsilon + \xi_i^*, \quad i = 1, ..., l \\ & \xi_i, \xi_i^* \geqslant 0, \quad i = 1, ..., l, \epsilon \geqslant 0 \end{aligned} \tag{2.2}$$

and finally, in case of non linearity, we may also introduce a kernel function to the problem, by changing x_i to $\phi(x_i)$, and performing the trick from the previous section. Minimization problem (2.2), also known as ν-SVM, with the help of the radial basis kernel function is what was used predicting permeabilities in this report.

Chapter 3

Permeability

In south of France, in the city of Dijon, Henri Darcy investigated the flow of water through homogeneous sand beds. This led to the discovery of his empirical law. The apparatus used by Darcy to deduce his law is pictured in Fig. 3.1.

Darcy concluded that:

$$Q = KA\frac{h^{(1)} - h^{(2)}}{L} \tag{3.1}$$

where:

Q is the volumetric flow rate of the liquid (volume of liquid passing per unit of time).
A is the cross-sectional area to the flow.
$h^{(1)} - h^{(2)}$ is the difference in water level elevations, at the inflow and outflow reservoirs.
L is the length of the sand bed (the sand bed was a cylinder filled with sand and water was pouring through) [3, p. 110].

The constant K is a coefficient of proportionality, called hydraulic conductivity. In an isotropic material one can define this constant as expressing the ease with which a fluid flows through the tortuous (curved) void space, the coefficient therefore depends therefore both on the material and fluid.

The hydraulic conductivity K can be expressed $K = \kappa\frac{\rho g}{\mu}$ where g is the gravity acceleration, ρ is the density of the fluid (measured in (kg/m^3)), and μ is the viscosity of the fluid, (a measure of a fluids resistance to gradual deformation by shear or tensile stress, measured in (m^2/s)). As we can see ρ and μ are properties of the fluid, but clearly the constant K depends on the material as well, which leaves us to study the constant κ called the permeability.

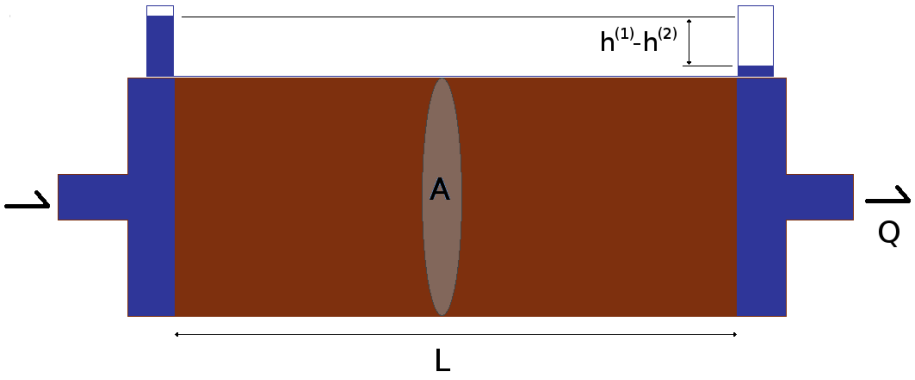


Figure 3.1: Darcy's apparatus. Water is injected to the left and flows through the sand filled tube. The liquid's pressure drop is measured by calculating the height difference in water level elevation at inflow and outflow. It is empirically shown that the volumetric flow rate of the liquid, Q, is proportional to the cross-sectional area A, the pressure and the length of the sand bed L.

By expressing K as above, along with letting $u = \frac{Q}{A}$ (average fluid velocity) this gives rise to a common version of Darcy's law, namely:

$$u = -\frac{\kappa \Delta p}{\mu L} \tag{3.2}$$

where $-\Delta p$ is the pressure difference ($-\Delta p = \rho g(h^{(1)} - h^{(2)})$).

Relevant solid properties are: pore-size distribution, shape of pores, tortuosity of passages T, specific surface S and porosity ϕ. These features all affect the permeability κ. Permeability depends thus only on the configuration of the void space and not on the properties of the fluid.

Permeability is measured in the SI unit m^2 but has also a frequently used unit named Darcy where $1\ D = 0.9869233\ (\mu m)^2$. Often the approximation $1\ D \approx 1\ (\mu m)^2$ is used.

Many formulae are developed to calculate permeability of porous materials, since no formula has been found which calculates permeability for structures of any constitutions. Different formulae works for different materials, so it is important to specify which material the formula is meant to work with.

An early empirical formula for calculating the permeability of a sand

bed is:

$$\kappa = Cd^2$$

where d is a measure on the diameter of the grains forming the material and C is a dimensionless constant. In 1943, Krumbein and Monk suggested $C = 6.17 \cdot 10^{-4}$ [2, p. 31].

Another formula for sand beds is the Fair and Hatch formula developed from dimensional considerations, and verified experimentally

$$\kappa = \frac{1}{\beta}\left[\left(\frac{(1-\phi)^2}{\phi^3}\frac{\alpha}{100}\sum_m \frac{P_m}{d_m}\right)^2\right]^{-1}$$

where β is a constant found by experiment to be 5, α is a grain shape factor, P_m is the weight fraction of sand held between adjacent sieves and d_m is the geometric mean diameter of the adjacent sieves [2, p. 31].

3.1 The Kozeny-Carman equation

A widely used equation though, is the famous Kozeny-Carman equation.

Imagine that we have a material where every pore is cylinder shaped. Assume now that we have an area cross-sectional to the flow A and a pore cross-sectional area A_p, then

$$A_p = A\phi$$

where ϕ is the porosity.

One can express Qdt [1], the volume of liquid passing through the pore in time dt, as:

$$Qdt = Ads\phi$$

where ds is the line element crossed in time dt.

Now if v_p is the average velocity of the liquid through the pore and u the average fluid velocity in the material[2] the following relationship can be established:

$$v_p A_p = uA = Q$$

From this, with $v_p = \frac{ds}{dt}$, we can write:

$$u = \frac{A_p}{A}v_p = \phi v_p \tag{3.3}$$

[1] Where Q is the volume per unit time from Equation 3.1 and dt the time step.

[2] $u = \frac{Q}{A}$ As in Equation 3.2

By solving the Navier-Stokes equation for the Poiseuille flow in a cylinder it can be deduced that:

$$v_p = -\frac{\Delta p \psi A_p^2}{L \mu C_p^2} \tag{3.4}$$

where C_p is the circumference of the pore. By putting together Equations 3.3 and 3.4 the following relationship can be established:

$$u = \frac{A_p}{A} v_p = -\frac{\Delta p \psi A_p^3}{L A \mu C_p^2}$$

With our assumption that the pore has cylindrical shape, the surface of the pore is $C_p L$ where L is the length of the tube. We also have:

$$\frac{A_p}{C_p} = \frac{A_p}{C_p}\frac{LA}{LA} = \frac{A_p}{A}\frac{AL}{C_p L} = \frac{\phi}{S}$$

where S is the specific surface of the tube (area over volume). We can thus write Equation 3.4 as:

$$u = -\frac{\Delta p \psi \phi^3}{L \mu S^2}$$

And by Equation 3.2:

$$\kappa = \psi \frac{\phi^3}{S^2} \tag{3.5}$$

This equation is called the Kozeny-Carman equation and the constant ψ was originally, by empirical studies, chosen to be around 0.5. Nowadays the constant is usually numerically adjusted to fit a training set of known permeabilities for a certain structure since it may vary a lot for different constitutions. The equation can be extended by introducing the Tortuosity which yields the equation[3]:

$$\kappa = \psi \frac{\phi^3}{T^2 S^2} \tag{3.6}$$

This equation has some limitations: it has geometrical assumptions, e.g. the pores are viewed as cylinders. Therefore it is not suitable for certain materials, as clayey soil for example. The formula does also not take into account anisotropy, which gives problem if the permeabilities of one direction is far greater than the other (since the equation would give the same value for each direction).

Finally Darcy's law is used. This law holds for silts, sands and gravelly sands, but as the size grows and the velocity increases and inertial effects must be taken into account.[19, p. 100–104][3, p.119][6].

[3]Note again that permeability only depends on the properties of the material, not fluid.

Chapter 4

Methods

This chapter is split into three sections: the first section is dealing with whole 3D models of the structures, the second one with the 2D analogs and the third with how stress tests were performed on the 2D SVM. The goal of the first part was to get the SVM perform at least as good as the Kozeny-Carman and the goal of the second was trying to make the simpler 2D SVM model perform at least as good as its 3D counterpart. The stress tests were performed to see if and when the 2D SVM would break, to gain more understanding on how it works.

4.1 3D Structures

The feature data for the SVM for the 3D structures was acquired from Geodict, as stated earlier, and is shown in Table 4.1. To begin with, since some features were represented by hundreds of numbers, instead of inputing all data directly into the SVM model (and thus letting the SVM treat each number as a separate feature) first some calculations were made to approximate these features with just a couple of numbers. The first obvious change was to take the path length data and combine it with the maximum particle diameter and define this as the maximum path volume (the path is viewed as a long cylinder). The mean and standard deviation was chosen to represent the path volume curve, thus we could approximate those features with just two parameters. One could ask if calculating the path volume has an impact on the SVM; if the result would be better if path length and maximum particle diameter were treated by itself. According to our empirical studies this is not the case, so it was chosen that they would be merged, to create a smaller, more comprehensible model.

The Pore size distribution was described with just its approximated mean and standard deviation. The surface area, SVP and DT values were

Data	Size	Description
Permeability [m^2]	1	A measure of the ability of a fluid to be transmitted through the structure.
DT [%]	1	A measure on isotropy
SVP [%]	1	Percent of total volume consisting of solid material
Specific surface	1	The specific surface of the structure.
Max. Particle Diameter [μm]	500	The diameter of the biggest sphere which can penetrate the material from different starting points
Path Length [μm]	500	The length of the path created when letting the sphere penetrate the material.
Pore size distribution [%]	20-70	How many percent of the void the largest spheres in the void constitutes.

Table 4.1: The original data acquired from Geodict.

already one dimensional, so they needed not be trimmed. See Table 4.2 for reference.

The SVM treated the mean and standard deviation as separate features, thus creating a model of 8 dimensions (one for the output i.e. Permeability and 7 for the input). Last but not least all input data was scaled to the compact interval $[0, 1]$ since the SVM works best with normalized data [8]. Our data was stored in a matrix where every row held our compressed data for each structure. Now we randomly permuted the rows and selected the last 15 rows to become our test data while the above 75 rows became our training data.

The C, γ and ν parameters in (2.2) were found with an algorithm based on MATLAB's built-in function *fminsearch* on the test set of permeabilities. We calculated one set of parameters for each direction and when the optimal parameters were chosen the support vector machine was put to the test on all data on this specific direction.

After the above process was tested to be accurate the process was re-

Data	Size
Permeability	1
DT	1
SVP	1
Specific surface	1
Maximum Path Volume	2
Pore size distribution	2

Table 4.2: Our data transformed.

peated but this time we removed columns of our matrix to find out which data we could remove while still preserving accuracy.

The Kozeny-Carman equation (3.6) contains a constant ψ and a value T called Tortuosity and since Tortuosity is somewhat messy to calculate we let $\hat{\psi} = \frac{\psi}{T^2}$ and used *fminsearch* to find the $\hat{\psi}$ which minimized the relative error[1]. We thus used the following version of Kozeny-Carman when calculating the permeability:

$$k = \hat{\psi}\frac{\phi^3}{S^2}.$$

We first tried to calculate just a single value for $\hat{\psi}$ for each direction (a total of 3 different $\hat{\psi}$); this can be seen as omitting Tortuosity from the Kozeny-Carman equation as in (3.5). As seen in Fig. 5.3, this gave such a bad approximation of the permeabilities we had to reconsider (the calculated permeabilities are off by up to 100-150% from their measured values, with a mean of 30% error). We had to make a choice in how we would approximate the Tortuosity well enough and so we decided that if a structure had the same DT values we viewed it as the "same structure" when we performed the curve fitting, so we got 6 different values for $\hat{\psi}$ for each direction (Fig. 5.4). We could not use the line fitting method to calculate one value of Tortuosity per structure since obviously it would lead to an unrealistic situation. We really tried to make the Kozeny-Carman equation as good as possible though. For example the formula does not take into account the anisotropy, as stated in Chapter 3, and therefore the value of $\hat{\psi}$ was recalculated for different DT values, so that it would optimize the equation to each direction and get a better result.

[1]The relative error $\eta = |1 - \frac{\kappa_{approx}}{\kappa}|$ where κ is the permeability. The relative error was used so that the optimization would not neglect errors in the small permeabilities in favour of the large ones.

4.2 2D Structures

By studying the results of the SVM with data from the 3D structures it was decided that the following values were to be approximated: DT, SVP, Specific surface and Pore size distribution.

Recall that the images were binary with a pixel being one if the pixel represented a fibre unit and zero if it represented a pore unit. For every structure, the values in the list mentioned above were calculated for each direction.

By definition the SVP value is just the fraction of solid material in the structure, so the quotient of the sum of the pixels and the size of the image was taken.

The Specific surface was simply approximated by taking the circumference of the connected regions of pores divided by the area of the image.

The DT value was approximated by first calculating the lineal-path function described in [15] with a Monte Carlo method. The lineal-path function describes the probabilities that line segments of different lengths which start in the fibre-part of the structure will fit entirely in that region and not traverse any pore-parts.

The algorithm starts by randomly choosing starting points in the fibre-part of the material. If the pixel n steps to the right of a starting point is still a fibre then a line of length n is counted as a success. After counting the number of successes for all starting points and repeating for every $n = 1, ..., R$ where R is a chosen maximum radius, the number of successes is divided by the number of starting points to give the horizontal lineal-path function. In the same manner the vertical lineal-path function is calculated (Fig. 4.1).

For each image the two functions were extracted. The functions were found to be exponential functions with y-intercept in 1[2], but with different slopes for each orientation of the line. Therefore a curve fitting of the type e^{-ax} was made and the DT value was chosen to be the quotient of the curves' respective constants a.

The Pore size distribution was approximated with an algorithm reverse engineered from the one used in Geodict but adapted for two dimensions instead of three (Fig. 4.2). The algorithm is as follows:

[2]Since we always start in the fibre part of the material, lines without length will always fit entirely in the fibre part

1. Load an image and perform a distance transform d on it.[3].
2. Initialize an array S as big as the image. Let S=0.
3. Split the interval $I = [0, \max_{x \in A}(d(x))]$ in N subintervals of equal width.
4. Create an array H of length N.
5. For each subinterval I_j with start in N and stepping down to 1:
   (a) Find every pixel with $d(x) \in I_j$.
   (b) Let $S(y) = j$ for $\|x - y\| \leqslant d(x)$ if S=0.
   (c) Count the number of pixels with $S(x) = j$ and save it in $H(j)$.
6. Normalize H so that $\sum_{j=1}^{N} H_j = 1$.

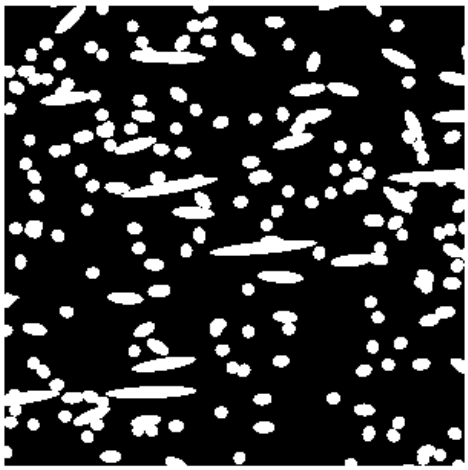
Finally the mean of the calculated values was taken for each image in all three directions to generate one of each values per structure.

To calculate all data for the 2D Structures with 10 images per direction for 90 structures (so a total of 2700 images) took around 30 minutes in MATLAB, where a mere 2 minutes were for calculating everything except the pore-size distribution.

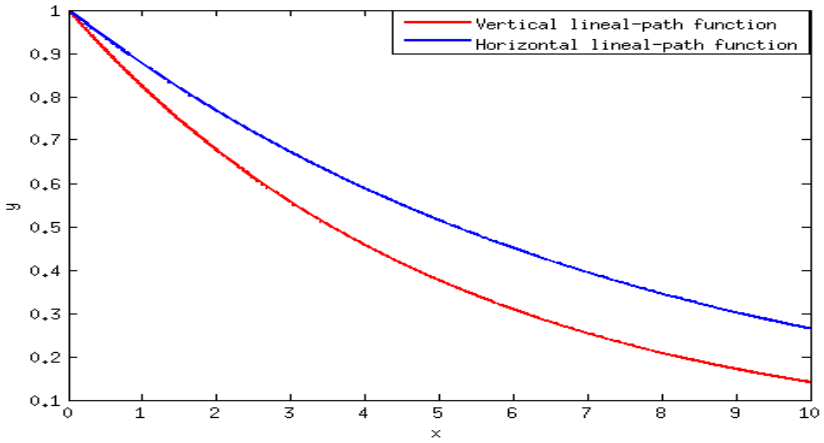
A couple of tests were performed with the 2D SVM. The first was obviously to compare it with the 3D SVM when every image had depth 1. Afterwards the error growth when images gained depth was checked. Finally the error growth when decreasing the amount of images per structures and directions was checked. To make it easier to follow every error was divided by the lowest error to capture the growth of the errors.

The constants approximated were also compared with the ones bundled with the 3D structures.

[3]This is further explained in [10]. The distance transform for a binary image calculates for each pixel the distance between that pixel and the nearest nonzero pixel of the image.



(a) The loaded image.



(b) The lineal-path functions

Figure 4.1: The calculated horizontal and vertical lineal-path functions for one image. As seen in the second plot the probability is slightly higher that a horizontal line will fit inside the fibre-part of the material rather than a vertical one, agreeing with the image we perform the calculations on.

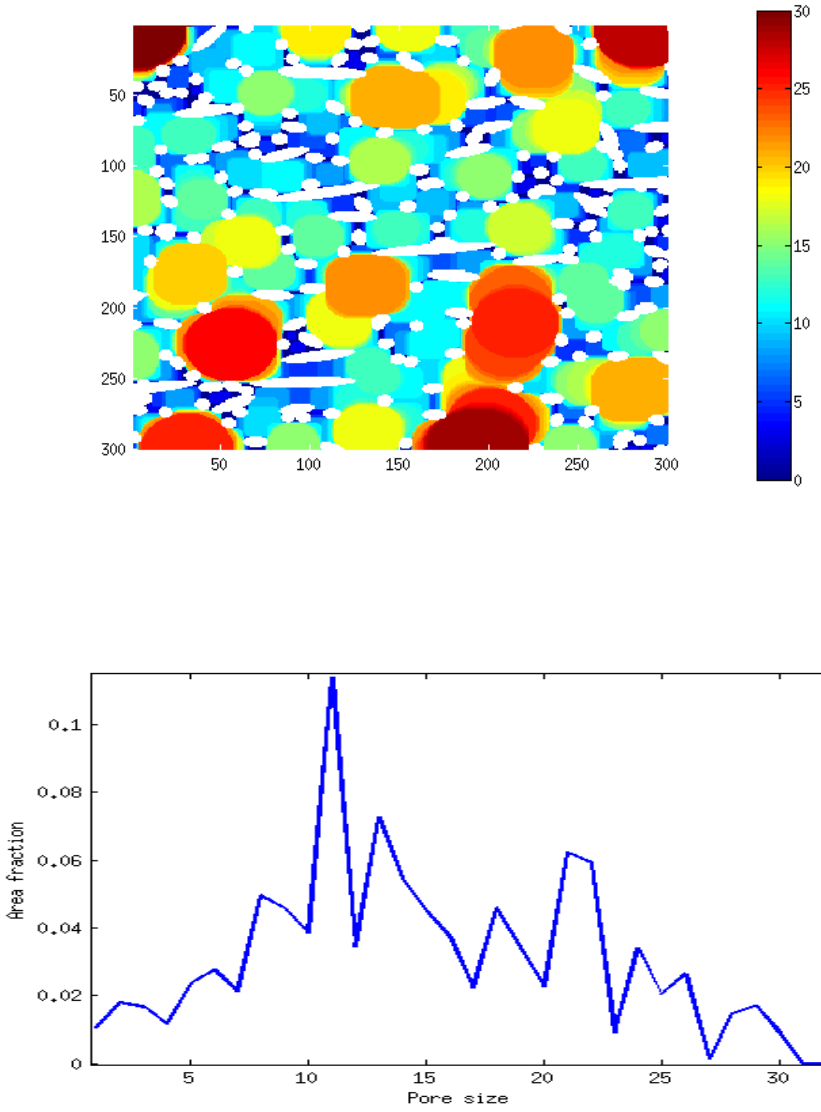


Figure 4.2: The top image shows the biggest spheres in the void in colour and the solid material in white. The bar to the right shows a measure on how much each sphere contributes to the permeability of the material. The bottom image shows the calculated pore-size distribution of the image.

4.3 Stress tests

Three stress tests were performed to see if and when the 2D SVM would break and become useless.

The first was to see if the SVM could handle to estimate the permeability for fibre structures with SVP values different than the ones in the training set. Each new structure had an SVP value of 20 or 40%. This was done to give some insight in how important the content of the training set is.

The second test was to see what would happen to the SVM if the training set was extremely small (just 20% of the total amount of structures) but contained structures with all different SVP values. This would further explore the importance of the contents of the training set.

The third and final test was to see how well the SVM estimated the permeability of a different kind of structures than fibre structures, given a training set of fibre structures. The new set of structures had permeabilities many magnitudes larger than the ones in the training set. This was to see how our approach would work with new notably different structures.

For these tests new structures were created. First a set of fibre structures with SVP values between the ones already existing. This set was used for the first two stress tests.

A set of structures made up of spheres instead of fibres was also created. They were created using the reaction limited cluster aggregation method [16] (Fig. 4.3). They had permeabilities up to a thousand times as big as the fibre structures. This set was used in the third stress test.

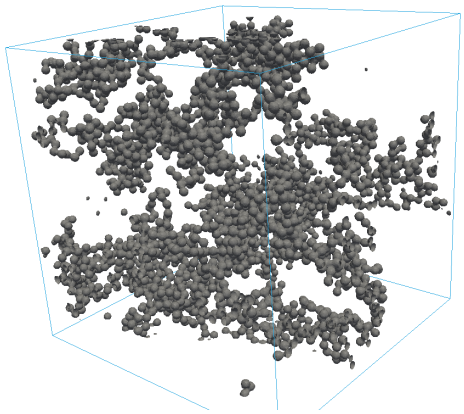


Figure 4.3: A sphere structure created with the reaction limited cluster aggregation method. The structures in the set were created with SVP values in the range of 0.02 to 0.08 resulting in extremely high permeabilities.

Chapter 5

Result

The training set was the same for all tests performed. It was created by randomly permuting the structures and choosing the first N of them (where N is a number between 1 and the number of structures obviously). Later the model created by the SVM was used to predict the permeability for all structures. The first test performed was to see how big training set was needed for accurate prediction. In Fig. 5.1 the relative error[1] of predicting the permeability for the 2D SVM given different sizes of the training set is shown. As seen in the figure the training set must be above 60% of the total amount of structures for the maximal relative error to be under 0.1 and the mean to be under 0.02. For the succeeding tests a training set of the reassuring size 70% of the total amount of structures was chosen. This can seem like a high number but it is needed since there were so many different types of structures present.

The following Section of this Chapter will compare the 3D SVM with the Kozeny-Carman equations (3.5) and (3.6), to examine if the 3D SVM is comparable. The next Section will compare the 3D SVM with its 2D counterpart, to inspect whether or not the same precision can be achieved with less information.

5.1 3D Structures

As seen in Fig. 5.2, the mean relative error for the SVM is around 0.01 if all feature data is present, which means a percentage error of 1%. The Kozeny-Carman equation on the other hand has a mean relative error of 0.2 i.e. 20%, which is around twice the maximum error of the SVM and also over 20 times bigger than the SVM's mean (Fig. 5.4).

[1] As stated before the relative error $\eta = |1 - \frac{\kappa_{approx}}{\kappa}|$ where κ is the permeability.

While excluding features from the input data the result was approximately the same in all directions, so the figures will just show the prediction in the X direction. If any data except DT was removed the error was almost not affected at all (Fig. 5.5(a)). If DT was removed though the mean error became 7% and the maximum 30%, so the maximum error got a threefold increase (Fig. 5.5(b)). If everything except DT was removed the program obviously became useless as well, but surprisingly by just adding SVP, Specific surface or Pore Size distribution the program worked with a mean error of just 2% (Fig. 5.5(c)).

Note that the values of C and γ are different for every plots, so removing data resulted in different optimal values for our constants.

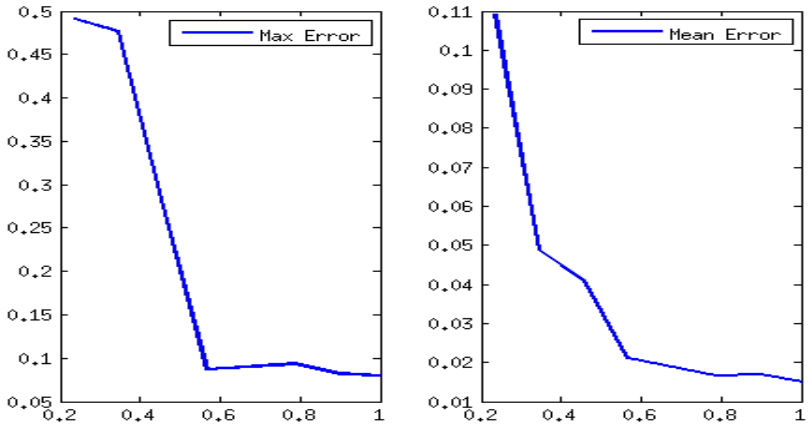


Figure 5.1: The bigger the set the better the prediction, as expected. The error is measured from predicting the permeability in the X direction with the 2D SVM. The x-axis shows the percentage of the total amount of structures used in the training set.

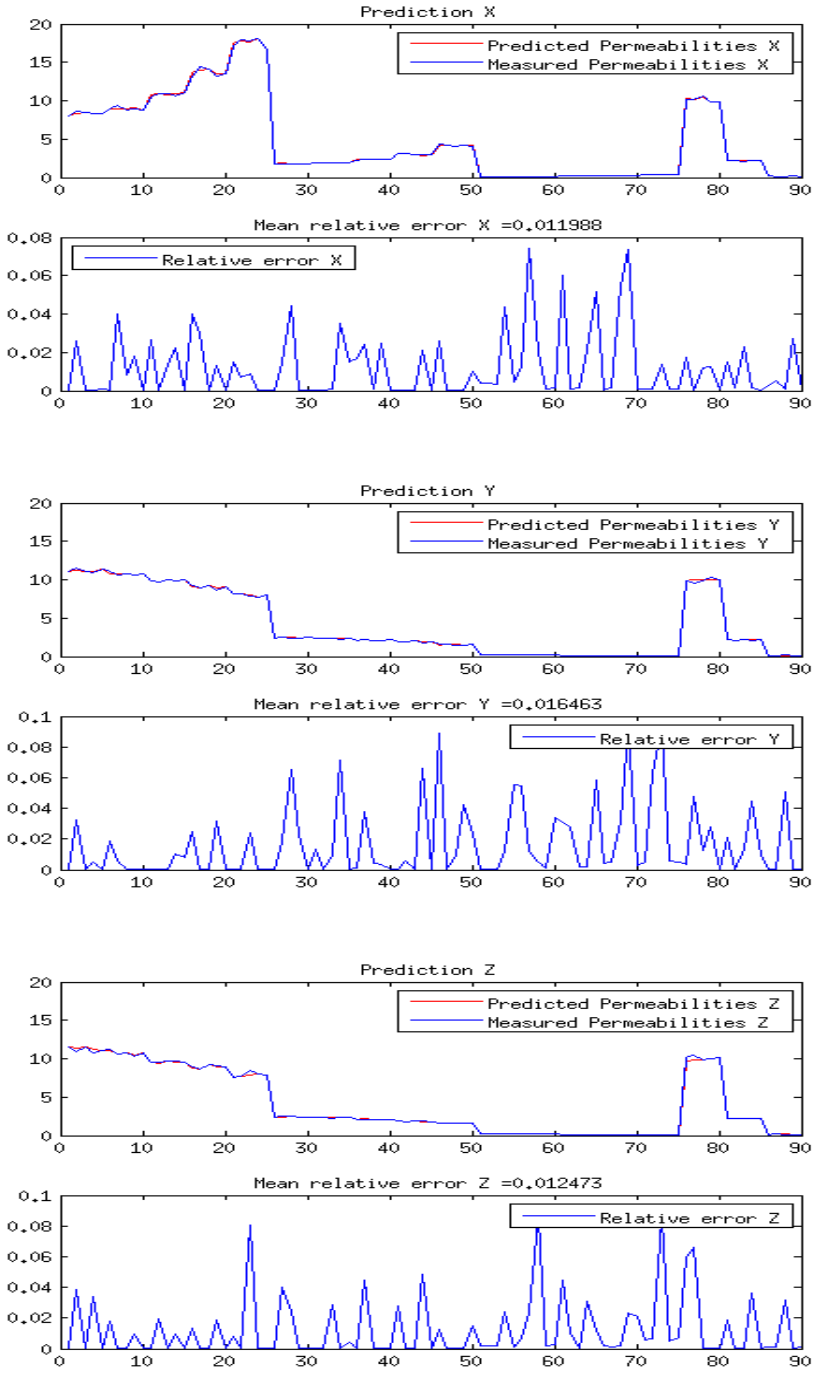


Figure 5.2: The upper plots show the predicted (red) permeabilities vs the measured (blue) and the lower plot shows the relative error of the methods in X,Y and Z direction respectively.

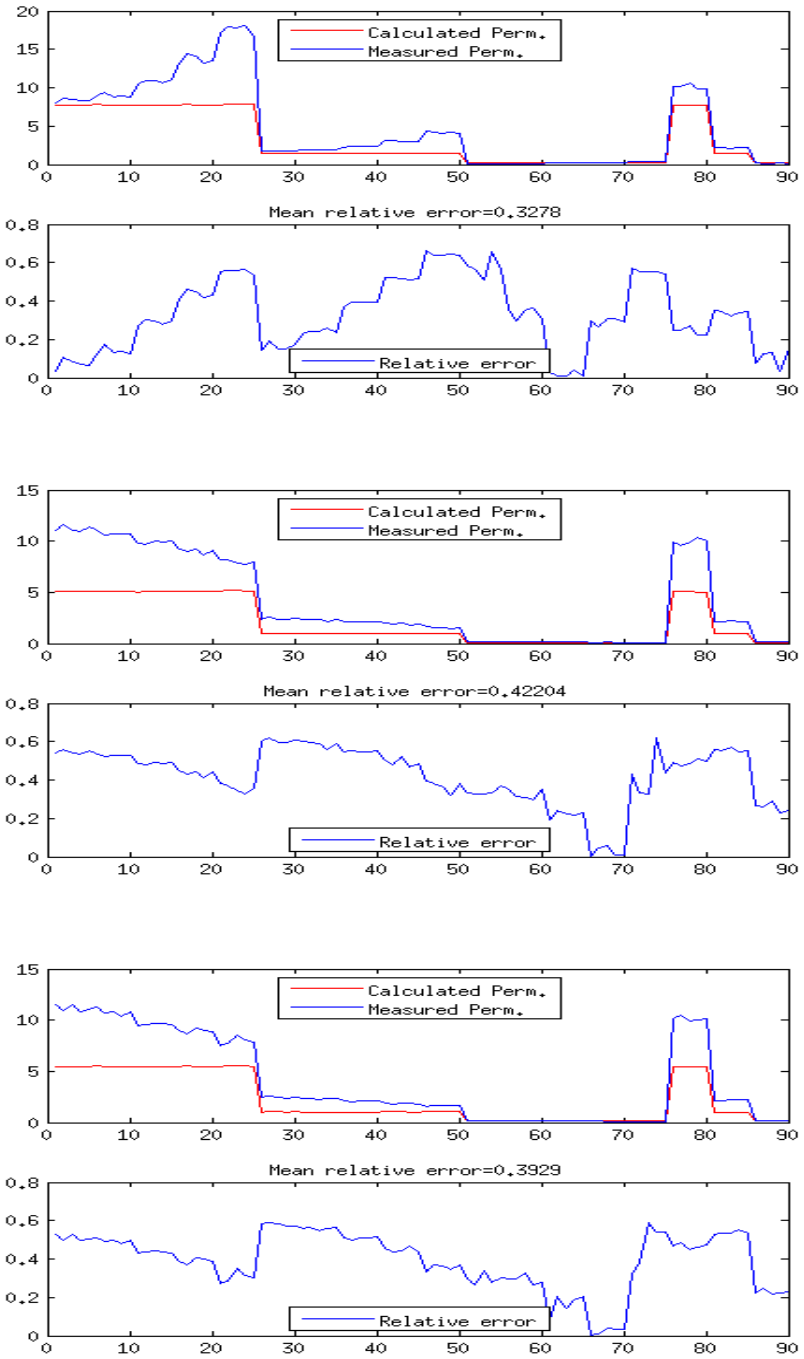


Figure 5.3: Permeabilities calculated with the Kozeny-Carman equation with one value for $\hat{\psi}$ for each direction. $\hat{\psi}$ was chosen to minimize the relative error, as explained in Section 4.

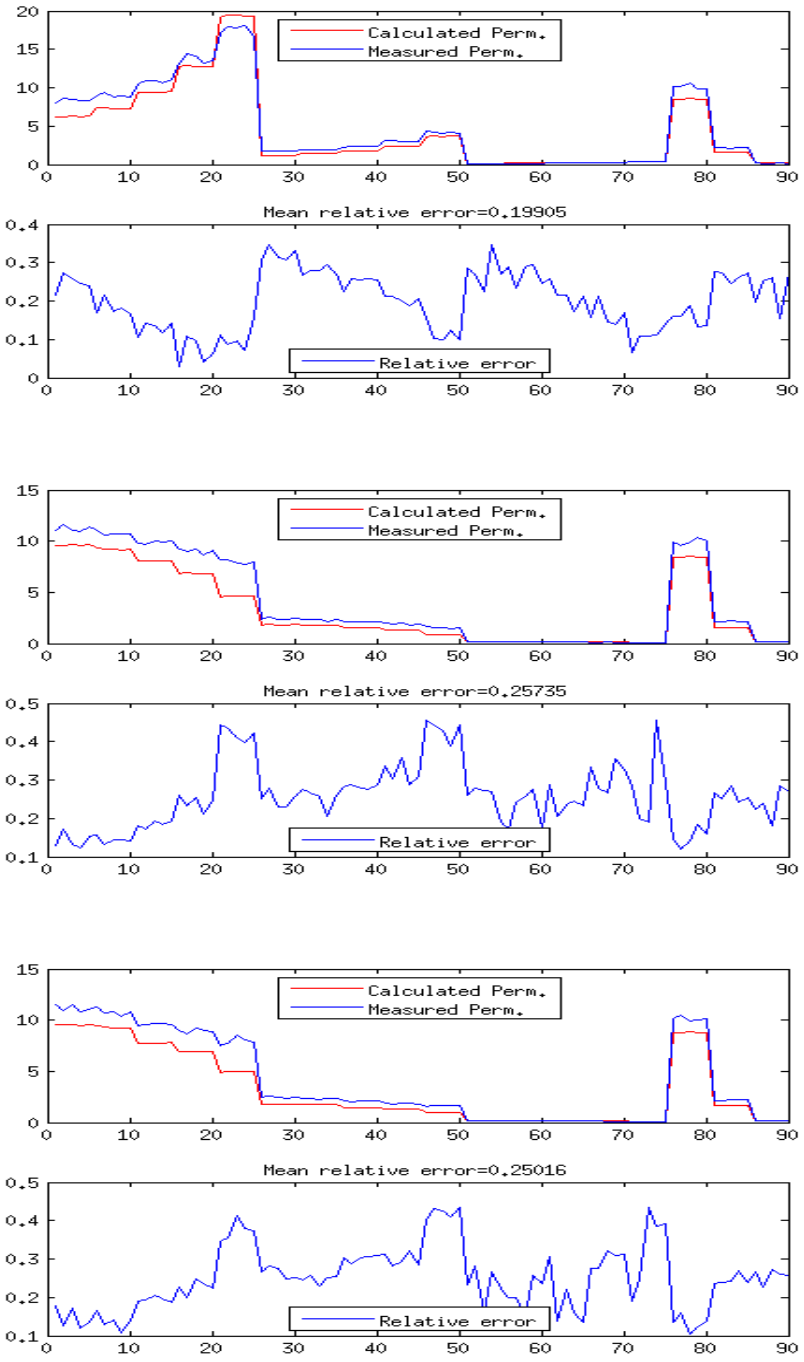
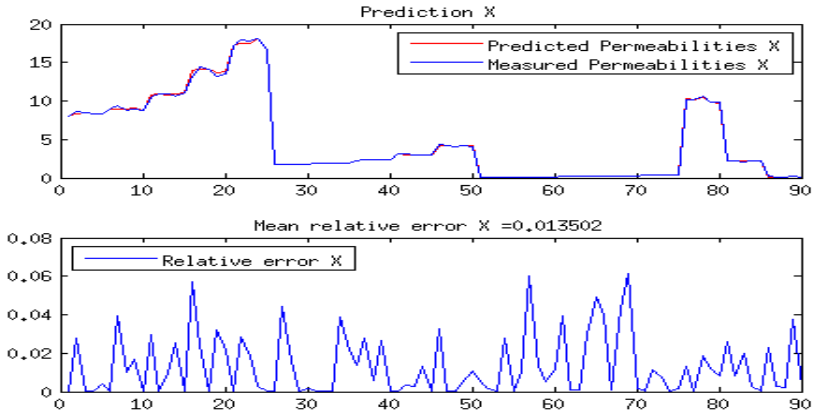
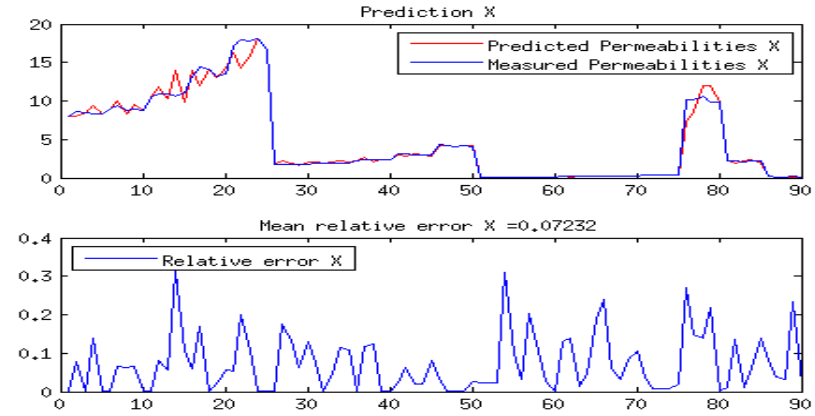


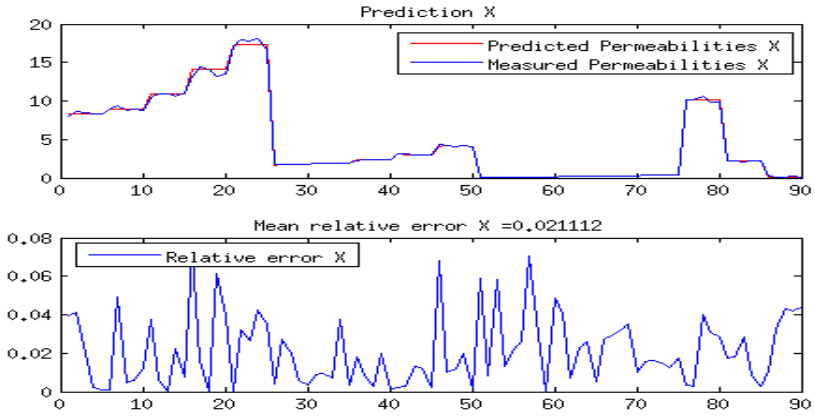
Figure 5.4: Permeabilities calculated with the Kozeny-Carman equation with 6 values for $\hat{\psi}$ for each direction. $\hat{\psi}$ was chosen to minimize the relative error, as explained in Section 4. This makes it look like the approximation is bad for structures with a small SVP value while the relative error remains fairly constant over all the structures.



(a) SVM without SVP in the X direction.



(b) SVM without DT in the X direction.



(c) SVM with just DT and Specific surface in the X direction.

5.2 2D Structures

As mentioned before, it is the shape of the curve rather than the exact value of the curve which is important for the SVM. The shapes of the curve were rather well captured from the 2D images as seen in Fig. 5.6.

As seen in Fig. 5.5 the 2D SVM is comparable with the 3D SVM in accuracy of its calculations.

The error when adding depth (Table 5.1) was small. The mean error with images of depth 10 was just 1.10 times the error when having perfect images of depth 1.

The error grew faster when the amount of images used to calculate the parameters were removed (Table 5.2). We have to remember though that the largest maximum error was still just 16%, which is still under 0.8 of the mean error of the Kozeny-Carman equation and below 0.2 of the maximum of the Kozeny-Carman equation. A test was also made using 50 images instead of 10, but no difference in error was found.

Image depth	Max Error	Mean Error
1	1.00	1.00
3	1.09	1.02
7	1.36	1.09
10	1.47	1.10

Table 5.1: Error growth when adding depth to the images.

Amount of images	Max Error	Mean Error
10	1.00	1.00
7	1.49	1.47
3	1.51	1.46
1	1.83	1.56

Table 5.2: Error growth when removing the number of images per direction used to calculate the parameters.

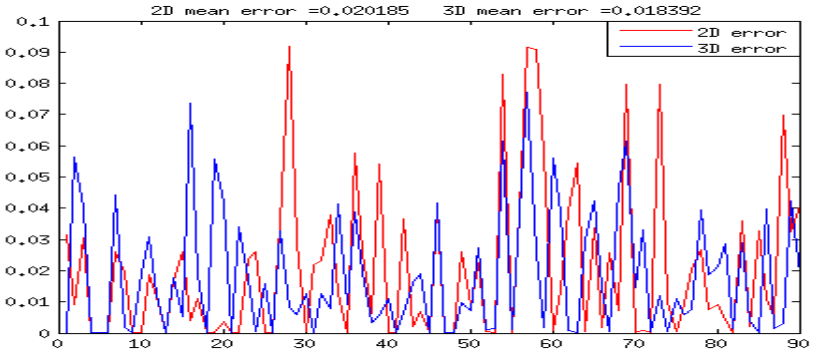
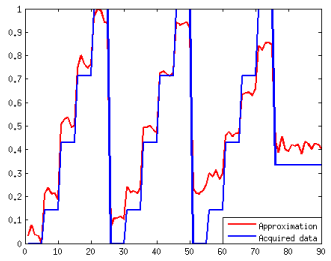
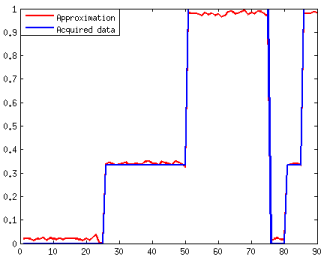


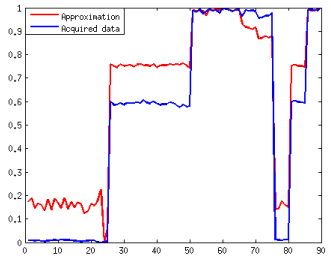
Figure 5.5: Comparison between the error of the 2D and 3D SVM with the same type of feature data.



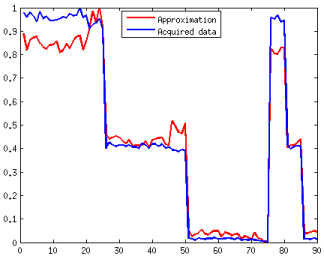
(a) The DT curve approximated.



(b) The SVP curve approximated.



(c) The Specific surface curve approximated.



(d) The Pore size distribution mean curve approximated.

Figure 5.6: The approximated parameters with ten images of depth one for each direction. All curves are normalized so that focus is drawn towards the shape of the curves.

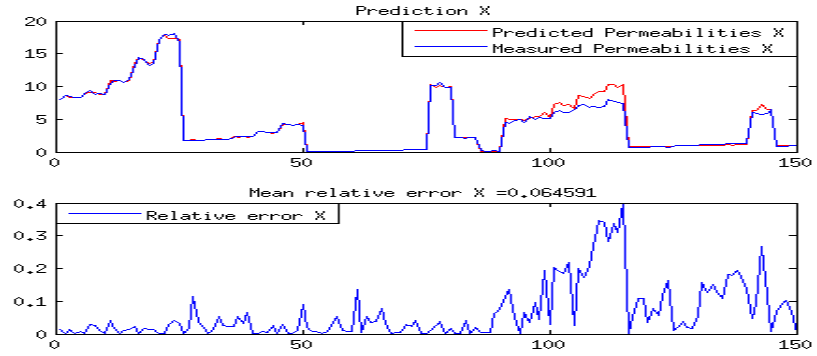
5.3 Stress tests

When calculating the permeability of a set of fibre structures with a different SVP value than the ones in the training set the mean relative error is raised to 6% and the maximum relative error is raised to 40%. From Fig. 5.7(a) it is easy to see that it is almost only the new structures which contribute to the error.

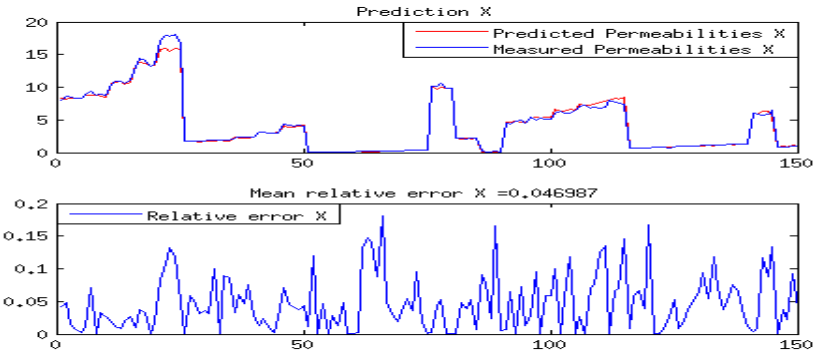
As seen in Fig. 5.7(b), the 2D SVM works well with a small training set if the data inside it is relevant. The structures in the testing set had SVP values in the same range as the ones in the training set, so the errors from the first stress test were not present. The relative error is evenly distributed between the training and testing set indicating that the little shape difference between the solid part of these structures does not matter much.

When predicting the permeability of a set of sphere structures the SVM greatly underestimates the permeability of the new structures (they have a permeability much larger than the ones in the training set). As seen in Fig. 5.7(c), since the training set contains small permeabilities, the SVM will continue to assume the predicted permeabilities to be in the same range. The SVM also greatly mis-classifies the permeabilities in the training set.

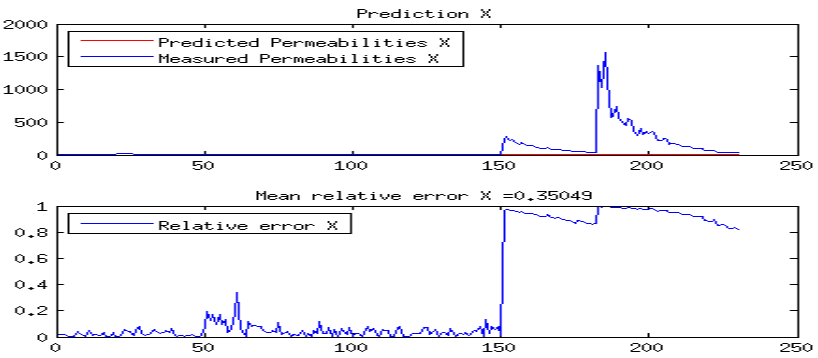
The Kozeny-Carman equation was used to calculate the permeabilities of new sphere structures as well. This time the Kozeny-Carman equation performed rather well. The largest relative error was around 20%.



(a) Prediction of the permeability of fibre structures with a different SVP value than what is in the training set.



(b) Prediction of the permeability of fibre structures with a small training set containing every different kind of SVP values.



(c) Prediction of the permeability of sphere structures with a permeability many magnitudes larger than what is in the training set.

Chapter 6

Discussion

It seems SVM is a good method for approximating permeability. Not only does it outperform the Kozeny-Carman equation with all input features present, it still works better when almost all of them are excluded. It must be noted though, that the Kozeny-Carman equation has a low error for big permeabilities and a high for small ones. The mean error for permeabilities over 1 is approximately 7%, so it is comparable with the SVM for high permeabilities but not for low.

With just the DT and either SVP, Specific surface or Pore Size Distribution present the SVM got a mean error of under 4%, so with a good way of approximating the DT value one could have an SVM calculate permeability better than Kozeny-Carman with less data. Acquiring data for permeability calculation is time consuming, so we were always trying to minimize the amount of data needed for a precise estimation.

By studying the feature data it seems the four parameters mentioned above are the key for our program to work. One could argue that it is cheating comparing Pore Size Distribution with the others since it adds two features instead of one to the SVM thus facilitating classification. Indeed with just the standard deviation the program is unusable and with just the mean it is just slightly better. If the classification gets better by adding dimensions to the feature set or that both mean and standard deviation are needed to get a grip on the distribution is hard to tell. To defend the use of both mean and standard deviation of the distribution one may compare it with adding one of the above features along with another feature which is not related (for example the mean of maximum path volume). This does not improve classification at all, so it seems the relevance of the features is much more important than how many they are, which is logical.

The maximum path volume can be regarded as superfluous, since it does not contribute much to lowering the error. Note that representing the

maximum particle diameter and path length separately does not help at all, it still gives the same result as when the features are fused.

One could ask why it is relevant to create a 2D equivalent of the 3D SVM. The obvious answer is that it is more feasible to extract 2D images of a structure via microscopy than gain full 3D information of the complete structure. Another aspect is computational workload. One 3D structure is a cube containing 300^3 pixels with a binary value in each pixel which means they occupy 27 Megabyte of space on a computer [1]. This is not a lot but imagine having one thousand structures with 1000^3 pixels per structure and it requires 100 Gigabyte of space. The 2D images on the other hand have no depth which means that if, for example, 30 images are chosen to represent the structure (10 images per direction), then the required space for a 300^3 cube decreases ten-fold and for a 1000^3 it is reduces 33-fold. It is thus easier to work with 2D images and the time to compute parameters as pore-size distribution is greatly reduced.

One could perhaps expect higher error rate in the 2D images with depth than actually attained. The strength of the SVM is that it just considers the relationship between the features of each structure and not their absolute value. Thus if there is a measurement error in one feature but this error is scaled so that the relationship is preserved, then the SVM will work just as well. To make the SVM worse the error must be in such way that it makes the relationship between the features change. This makes it more resistant to errors in some cases.

As seen in the stress tests one must train the SVM on data similar to the one it will be used for. All data must for example share SVP value for the SVM to work well. There are certainly more parameters than this which affects the preciseness of calculations. This is a drawback if one has not yet discovered what makes the data similar. One solution to this problem is to let a large random subset of the data be the training set, which makes it probable the training set contains enough relevant data of same types of structures as the ones in the testing set. But as seen in the first stress test size is not enough if the contents are bad. One could therefore first create multiple small randomized training sets for the SVM and explore the contents of the ones which give the best result. One cannot draw the conclusion that shape difference between structures affect the result of the SVM since the structures compared had SVP values so different that the errors noticed in the first stress tests were present. The structures also only had small shape differences in two dimensions. The 2D fibre images almost look like images of circles and makes it hard to distinguish them from the 2D

[1] Even though the images are binary most programming languages store logical values in 1 byte (8 bits) instead of 1 bit due to speed losses when calculating with single bits.

sphere images. To verify if the shape of the solid part of the material is of importance, new test must be performed with structures which look different in two dimensions but with SVP values in the same range. The exactness of the Kozeny-Carman equation when calculating the permeability of the new structures also indicates that their shape is not very different from the shape of the fibre structures. The new structures satisfy the assumptions of the equation well.

Studying the stress tests it can be concluded that if all the structures have the similar shape and have permeabilities in the same range and both the training set and testing set contain structures with similar SVP values, then the 2D SVM will perform well.

6.1 Future studies

The following subjects were partially or not answered in this thesis and are subject to future research:

1. Study how different type of noise in the images will affect the results.
2. Find effective ways of removing noise without compromising the result.
3. Discover new ways to obtain the best parameters C, γ, ν for the SVM and guarantee that a global minimum is found and not just a local.
4. Investigate how well the SVM calculates permeability for materials which in 2D look different than the ones in the training set.
5. Further explore how the contents of the training set will affect the exactness of the predicted permeabilities.
6. Run the 2D SVM on real microscopy data to verify if it works as well as with simulated data.
7. Procure more or alternative relevant features for calculating permeability with the SVM.

Chapter 7

Conclusions

In this thesis the following conclusions are drawn from the results acquired:

1. The support vector machine studied in this thesis performs better than the Kozeny-Carman equation in calculating the permeability. The reason for this is that the support vector machine takes more geometrical differences into account than the Kozeny-Carman equation.

2. The two most important features when calculating permeability for the structures studied in this thesis were shown to be the DT and SVP values. The DT value is a measure of the anisotropy of the structure and the SVP is the fraction of volume of solid material in the structure.

3. It is enough to study two dimensional images of three dimensional structures to get as good precision with the support vector machine. This makes it plausible that two dimensional microscopy data suffices to estimate permeability.

4. It seems as if both the training and testing set have structures with permeabilities and SVP in the same range, then the support vector machine will predict the permeability with small error. If either the permeabilities or the SVP values are very different in the two sets, the support vector machine will not work well.

5. A large training set is important to get good results (this way the problems mentioned in the previous paragraph are often avoided). Actually the contents of the set is more important than its size, but populating the training set with numerous structures is a good start to make the support vector machine work well.

Bibliography

[1] D. Basak; S. Pal; D.C. Patranabis, "Support Vector Regression". Neural Information Processing, Vol. 11, No. 10, 203-224, 2007.

[2] J. Bear; A. Verruijt, "Modeling Groundwater Flow and Pollution". Springer, 1987.

[3] J. Bear; H. Alexander, D. Cheng "Modeling Groundwater Flow and Contaminant Transport". Springer, 2010.

[4] C.M. Bishop, "Pattern Recognition and Machine Learning". Springer, 2006.

[5] L. Bottou; C. Cortes; J.S. Denker; H. Drucker; I. Guyon; L.D. Jackel; Y. LeCun; U.A. Muller; E. Sackinger; P. Simard; V. Vapnik, "Comparison of classifier methods: a case study in handwritten digit recognition". Pattern Recognition, 1994. Vol. 2 - Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on , Vol.2, No., pp.77,82 Vol.2, 9-13 Oct 1994.

[6] W. Carrier, "Goodbye, Hazen; Hello, Kozeny-Carman". F.ASCE, Journal of Geotechnical and Geoenvironmental Engineering, Vol. 129, No. 11, pp. 1054-1056, November 2003.

[7] R. P. Chapuis; M. Aubertin, "Predicting the coefficient of permeability of soils using the Kozeny-Carman equation". École Polytechnique de Montréal, 2003, Montréal.

[8] C. Chih-Chung; L. Chih-Jen, "LIBSVM : a library for support vector machines". ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm .

[9] H. Chih-Wei; C. Chih-Chung; L. Chih-Jen, "A Practical Guide to Support Vector Classification". Department of Computer Science and Information Engineering, National Taiwan University, Taiwan.

[10] P. F. Felzenszwalb; D. P. Huttenlocher, "Distance Transforms of Sampled Functions". Theory of computing, Vol. 8, pp. 415–428, 2012.

[11] R. Gholami; A. R. Shahraki; M. J. Paghaleh, "Prediction of Hydrocarbon Reservoirs Permeability Using Support Vector Machine". Mathematical Problems in Engineering, Vol. 2012, Hindawi Publishing Corporation, 2012.

[12] R. Goering, "Matlab edges closer to electronic design automation world". EE Times, 2004-04-10, Santa Cruz, California, http://www.eetimes.com/document.asp?doc_id=1151422, 2014-01-10 .

[13] T. Hastie; R. Tibshirani; J. Friedman, "The Elements of Statistical Learning". Stanford University, 2008, Stanford.

[14] M. Law, "A Simple Introduction to Support Vector Machines". 2011, Lecture slides for CSE 802, Department of Computer Science and Engineering, Michigan State University, Michigan.

[15] B. Lu; S. Torquato, "Lineal-path function for random heterogeneous materials". Physical Review A, Vol. 45, No. 2, 1992.

[16] W.C.K. Poon; M.D. Haw, "Mesoscopic structure formation in colloidal aggregation and gelation". Advances in Colloid and Interface Science 73, 71–126, 1997.

[17] M.R. Rwebangira, "Techniques for Exploiting Unlabeled Data". Carnegie Mellon University, 2008.

[18] S. Saffarzadeh; S.R. Shadizadeh, "Reservoir rock permeability prediction using support vector regression in an Iranian oil field". Petroleum University of Technology, 2011, Abadan.

[19] A. Scheidegger, "The physics of flow through porous media". University of Toronto Press, 1974.

[20] B. Schölkopf; A. Smola; R. C. Williamson; P. L. Bartlett, "New support vector algorithms". Neural Computation, 12:1207–1245, 2000.

[21] D.W. Taylor, "Fundamentals of soil mechanics". J. Wiley, 1948, New York.

[22] V. Vapnik; C. Cortes, "Support-Vector Networks". Machine Learning, 20, 273-297, 1995.

[23] A. Wiegmann; J. Becker; E. Glatt; S. Rief; L. Cheng, "GeoDict 2012, Basic Volume". 2012, Software available at http://www.geodict.de/