Norwegian University
of Life Sciences

Master's Thesis 2019 30 ECTS
Faculty of Science and Technology

Metamodelling of the Hodgkin-Huxley Model and the Pinsky-Rinzel Model Using Local Multivariate Regression and Deep Learning

Lars Erik Ødegaard
Data Science (M. Sc)

Acknowledgement

The work presented in this thesis was performed as a part of the Master program in Data Science at the Faculty of Science and Technology (REALTEK) at the Norwegian University of Life Sciences (NMBU) the spring of 2019.

I would like to thank Professor Kristin Tøndel, my main supervisor, who has given excellent guidance and support. Further, I would like to thank my co-supervisor Dr. Klas Pettersen, whose enthusiasm and positivity has kept me motivated throughout the spring. I would also like to thank my co-supervisor Professor Gaute Einevoll, whose inspiring words piqued my interest for the field of neuroscience.

Lastly, big thanks to my friends and family for their support throughout the past six years.

May the 12th 2019, Ås
Lars Erik Ødegaard

Abstract

Biological processes, such as the electrical activity in neurons, are often modelled using complex, non-linear and high dimensional differential systems. Such models are usually associated with a high computational cost. Statistical tools are often needed in order to get a comprehensive overview of the behaviour of such systems. Using statistical emulators (metamodels) have been shown useful for providing insight into model behaviour, as well as reducing the computational demand. In this thesis, the two metamodelling techniques, Hierarchical Cluster-based Partial Least Squares Regression (HCPLSR) and deep learning were explored and compared. This was done by metamodelling the simpler Hodgkin-Huxley model and the more complex Pinsky-Rinzel model. The input parameters were varied in an Latin Hypercube Sampling (LHS) design, and the somatic membrane potentials were generated using the single neuron activity models. Further, the metamodelling techniques were used to find input-output and output-input relationships in the two models. The results indicate that deep learning metamodelling is a more efficient emulator of complex non-linear models, while the HCPLSR metamodelling allows for a more detailed interpretation of the model behaviour. These findings emphasize the need for using subspace analysis in order to accurately describe complex models with a wide range of behaviours, suggesting that subspace analysis in combination with deep learning emulation can further improve the understanding of model behaviour.

Sammendrag

Biologiske prosesser, som for eksempel den elektriske aktiviteten i nevroner, blir ofte modellert ved bruk av komplekse, ulineære og høydimensjonale differensialligninger. Slike modeller er ofte forbundet med et høyt krav til beregningskraft. For å få oversikt over oppførselen til modeller med høy kompleksitet er det derfor ofte nødvendig å bruke statistiske verktøy. Statistiske emulatorer (metamodeller) har vist seg å kunne redusere kravet til beregningskraft, samt å gi innblikk i adferden til uoversiktlige, adrubt ulineære modeller. I denne oppgaven er to metamodelleringsteknikker utforsket og sammenlignet: Hierarchical Cluster-based Partial Least Squares Regression (HCPLSR) og dyp læring. Dette er gjort ved å metamodellere den mindre komplekse Hodgkin-Huxley-modellen og den mer komplekse Pinsky-Rinzel-modellen. Inngangsvariablene er variert gjennom LHS-variabeldesign og somatiske spenninger er simulert ved bruk av de to nevnte nevronmodellene. Videre er metamodelleringsteknikkene brukt for å finne sammenhenger mellom inngang- og utgangsvariablene. Resultatene indikerer at dyp læring er en effektiv emulator for komplekse ikke-lineære modeller, men HCPLSR-metamodelleringen tilbyr en mer detaljert tolkning av modellenes oppførsel. Funnene i denne oppgaven understreker behovet for å analysere individuelle underrom i parameterrommet for å nøyaktig kunne tolke oppførselen til komplekse modeller med varierende, ulineær adferd. Funnene tyder på at en kombinasjon av dyp læring og underromsanalyse vil kunne forbedre vår forståelse av komplekse modellers oppførsel ytterligere.

Contents

List of Figures

List of Tables

Abbreviations

ANN	Artificial Neural Network
AP	Action Potential
CNN	Convolutional Neural Network
FCM	Fuzzy C-means
FFN	Feed Forward Network
HCPLSR	Hierarchical Cluster-based Partial Least Squares Regression
LHS	Latin Hypercube Sampling
PC	Principal component
PLS	Partial Least Squares
PLSR	Partial Least Squares Regression
RMSE	Root Mean Square Error
SVM	Support Vector Machine

1 Introduction

Mathematicians and physicists have for a long time tried to explain biological behaviour using mathematical models. In order to model biological processes, one usually has to use complex, non-linear and high dimensional differential equation systems. This is especially true in the field of computational neuroscience. The human brain is composed of roughly 100 billion neurons with 100 trillion axonal connections between them (Squire, 2008). The electrical activity of each neuron is determined by a wide range of different ion concentrations, ion gates, and neuronal morphology, among other factors. This leads to a large number of parameters needed to model these behaviours (Sterratt et al., 2011). The large complexity of these models leads to a high computational cost (Tøndel et al., 2014), especially when short time steps are needed in order to get accurate predictions. It is also very challenging to obtain a comprehensive overview of the behaviour of such a model across the high-dimensional input parameter space (Tøndel et al., 2012).

Most of the existing techniques for sensitivity analysis and parameter estimation are only suitable for relatively low dimensional outputs and typically focus on one output variable at the time (Saltelli et al., 2008). Statistical methods have gained acceptance as a method for analysis of complex models. Statistical emulations of dynamic models (metamodels), have shown the ability to reduce the computational cost and may serve as a basis for sensitivity analysis. (Tøndel et al., 2012).

The Partial Least Squares Regression (PLSR) extension Hierarchical Cluster-based Partial Least Squares Regression (HCPLSR) (Tøndel et al., 2013) has been shown to be an especially useful tool for multivariate metamodelling of dynamic models, which often generate spatiotemporal data. It can also be used to improve the analytical insight into the model behaviour by identification of subsets of the data with different behaviours (Tøndel et al., 2013). Metamodelling with neural networks has also been shown to be a robust emulator with high accuracy and a significant computational speedup (Fonseca et al., 2003), but one of the challenges with neural networks is getting insight into how the neural network predicts in order to interpret the model. The latter is due to that neural networks are built up of a large number of nonlinear functions connected by weights that are adjusted to fit the data at hand, but the combination of functions does not necessarily reveal the underlying patterns behind the input-output relationships that they predict.

In this project, we apply HCPLSR and deep learning in metamodelling of membrane potentials generated by two single neuron models; the Hodgkin-Huxley model(Hodgkin and Huxley, 1952), and the more complex Pinsky-Rinzel model(Pinsky and Rinzel, 2001). The goal is to investigate how accurately these metamodelling methods can emulate the models, and what insight the methods can give into their behaviour.

2 Theory

2.1 Metamodelling

A metamodel (also referred to as a surrogate model or an emulation model) is a model of a model. The metamodel substitutes a more complex mathematical model, where the latter is often characterized by a high computational cost and high complexity. The metamodel is calibrated from input-output data from simulations with the original model. The simulations are designed by varying the input parameters of the model in an experimental design or a sampling. The goal of metamodelling is to create a replacement model that is as accurate as possible. They are often used for visualization, sensitivity analysis and exploring the parameter space and output space (Gorissen et al., 2009). It also has the potential to discover "hidden" patterns of co-variation in the model (Martens et al., 2013). Metamodelling has been used in a wide variety of fields, including biology and physiology (Tøndel et al., 2012), finance(Yu et al., 2009) and risk assessment (Lai et al., 2006). Metamodelling can be divided into multiple subcategories. In this paper, we will explore two types of metamodelling; *classical metamodelling* and *inverse metamodelling.* In classical metamodelling, the outputs of the mathematical model are predicted from the input parameters. (outputs = f(inputs)). In inverse metamodelling, the input parameters of the mathematical model are predicted from the model outputs (inputs = f(outputs)). The difference between classical and inverse metamodelling is illustrated in Figure 2.1.

The different methods can be used alone or in combination in order to describe the behaviour of the original model (Martens et al., 2013) A wide range of different methods supervised and unsupervised learning has been applied for metamodelling. These include various versions of Partial Least Squares Regression (PLSR)(Martens et al., 2013)(Tøndel et al., 2013),neural networks(Yu et al., 2009)(Kilmer et al., 1997), and other machine learning techniques such as Support Vector Machine (SVM)(Lai et al., 2006). In this

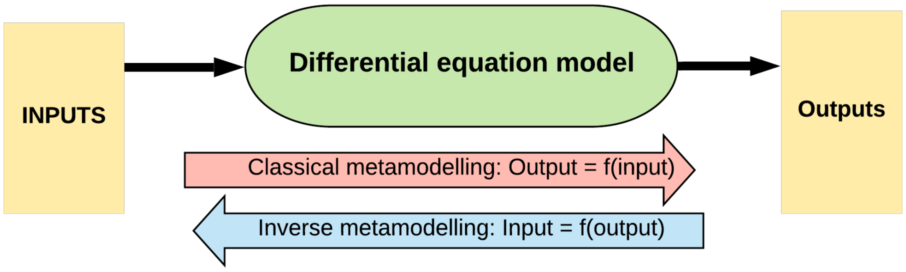

INPUTS
Differential equation model
Outputs
Classical metamodelling: Output = f(input)
Inverse metamodelling: Input = f(output)

Figure 2.1: Illustration of classical and inverse metamodelling.

thesis we will explore the use of the PLSR extension Hierarchical Cluster-based Partial Least Squares Regression (HCPLSR) and deep learning.

2.2 PLSR

PLSR is a statistical approach for modelling complex multivariate relationships based on decomposing both the input matrix and the output matrix into independent features of covariance (Tøndel et al., 2014). PLSR was pioneered by H. Wold and H. Martens in the late seventies (Wold et al., 1983). It has gained huge importance in the field of chemometrics (Geladi and Kowalski, 1986) and a wide range of other data-driven modelling fields (Haenlein and Kaplan, 2004). The goal of PLSR is to regress the input matrix X onto the output matrix Y.

The matrices X and Y are decomposed into scores and loadings as follows:

$$X = TP^\top + E = \sum t_h p_h^\top + E \tag{1}$$

$$Y = UQ^\top + F = \sum u_h q_h^\top + F \tag{2}$$

Where T and U are the scores matrices and P,Q are the loadings matrices of X and Y, respectively. E and F represent the unmodelled residuals (considered as noise) of X and Y, respectively. t_h, p_h, u_h and q_h represent column vector h of matrices T, P, U and Q. An inner relationship $u^h = b_h t_h$ and a mixed relationship $Y = TBQ^\top + F$

allows for the regression of Y given X. By using an iterative algorithm (e.g. NIPALS or SIMPLS) the inner relation is strengthened, and F is minimized (Haenlein and Kaplan, 2004). The resulting score and loading matrices consist of orthogonal columns, referred to as Principal component (PC)s. The PCs of the output matrix are sorted such that the first PC describes the largest part of the variation in the output matrix.

2.3 HCPLSR

HCPLSR is an extension of the PLSR-method and was proposed by Tøndel et.al (Tøndel et al., 2013). The HCPLSR pipeline splits the parameter space into multiple subspaces and trains a separate PLSR model for each sub-space. This allows for detecting different behaviours of the model for individual subspaces. HCPLSR has been shown able to improve prediction accuracy and provide insight into the relationships between inputs and outputs of a model. The method has been found especially useful for metamodelling of complex non-linear models. It has e.g. been used for revealing input-output patterns of biological models such as a dynamic model of a mouse heart (Tøndel et al., 2013), and for exploration of a dynamic model of the mammalian circadian clock (Tøndel et al., 2012).

2.4 Artificial neural networks

Artificial neural networks, mostly referred to as neural networks, computes the outputs by propagating the input data through layers of functions with trainable weights to the output layer neuron(s). They were first proposed and developed in the fifties and sixties soon after the invention of computers. Neural networks were designed to simulate the human nervous system, by connecting computational units (neurons) to one another through weights, inspired by how synapses connect the human neurons. In theory, the neural networks are capable of emulating any mathematical function, if not limited by computational cost or training data. The primary strength of neural networks, as for all machine learning methods, is to generalize from seen training data to unseen examples.

The following subsections are based on *Neural Networks and Deep Learning* by Charu C. Aggarwal (2018) if no other sources are directly referred to. For more detailed information regarding neural networks, it is recommended to consult this book.

2.4.1 Feed Forward Networks

Feed Forward Network (FFN) models are models in which artificial neurons are stacked into hierarchical layers, the first layer being the input vector x and the last layer being the output vector y. The intermediate layers are by convention referred to as "hidden layers". The depth of a network refers to the number of hidden layers in the network architecture. The number of nodes in a layer is commonly referred to as the width of the layer.

The input layer $x = [x_1 \dots x_d]$ contains d neurons which are the raw input data. The d inputs are passed to neuron j in the next layer. As the d inputs are passed they are multiplied with weight $w = [w_1 \dots w_d]$, and a bias is added. The receiving neuron, j, receives the linear function $z = w \cdot x + b$, and applies an activation function σ: $a = \sigma(z)$, which then is passed to the next layer. The process is repeated until the values arrive at the output layer $\hat{y} = [\hat{y}_1 \dots \hat{y}_m]$ containing m predictions. It is common, in FFN models, that all neurons in one layer are connected to all neurons in the following layer. Such a layer is often referred to as a fully connected, or dense layer. Different activation functions can be used in the neurons, and the most common ones are:

$$\text{sigmoid}: \qquad \sigma(z) = \frac{1}{1 + exp(-z)} \tag{3}$$

$$\text{tanh}: \qquad \sigma(z) = \frac{exp(2z) - 1}{exp(2z) + 1} \tag{4}$$

$$\text{ReLU}: \qquad \sigma(z) = \max(0, z) \tag{5}$$

$$\text{Linear}: \qquad \sigma z = z \tag{6}$$

The activation functions have different attributes that can contribute to the learning of the network. A key role of the activation functions is to allow for nonlinear learning. If all activation functions used in the network were linear, the network would only be able to learn linear dependencies. The sigmoid, tanh, and ReLU activation functions all allows for nonlinear learning but differs in the range of their activation. The tanh activation function allows for an activation in the range between -1 and 1, while the sigmoid and ReLU functions limit the activation to positive values. The sigmoid and ReLU activation functions differ in the sense that the ReLU is a piecewise linear function, with no upper limit and all negative values set to zero, while the sigmoid function fits arbitrary values into the [0, 1] interval. The sigmoid and tanh activation functions have traditionally been the most popular choices in the hidden layers, but ReLU activation has been increasingly popular in later years due to advances when it comes to avoiding issues known as vanishing and exploding gradient problems. In short, both these problems leads to non-optimal updates of the weights in the network. In the vanishing gradient problem, the partial derivative of the activation function is neglectable, and in the exploding gradient problem, the partial derivative is to high. The choice of activation function used in the output layer depends on the problem at hand. In a regression problem, it is common to use a linear activation function, which has an unlimited range.

The weights and biases in the network are usually updated through an iterative procedure called gradient descend. The error of the network is evaluated using an objective function commonly referred to as the loss function J. A example of such a function is the mean squared difference between the true output value(s) y and the predicted output value(s) $\hat{y}$; $J(\theta) = \frac{1}{N}\sum_{n=0}^{N}(y - \hat{y}_n)^2$, where n is iterated from the first sample (0) to sample N. The gradient descend algorithm updates the weights and biases through an algorithm in which the partial derivatives of the loss function are computed with respect to the parameters in the network. The weights and biases are updated with small "nudges" in the opposite direction of the gradient. Ideally, the gradient descend would update the parameters based on the cost function evaluated for all samples. However, doing so is often computationally expensive. Instead what is usually done is to evaluate the cost function for a subset of the samples at a time, and these subsets are also called *batches*.

The gradient descend method, is not always sufficient in order to get to the optimal weights, as it can get stuck in saddle points as the gradient get smaller. Optimizers have been developed as improvements of the gradient descend, an example of such an optimizer is adam (Kingma and Ba, 2014). By keeping track of the earlier partial derivatives and adding a momentum, increasing the ”nudges” if successive gradients points in the same direction, the adam allows for a faster training time, and the possibility to overcome saddle points.

Fitting a model to a particular training data set does not guarantee a good prediction performance for unseen test data. The lack of generalization ability of a model is referred to as overfitting and is a well known issue within machine learning. Overfitting can be avoided by reducing the number of parameters in the model, or by using regularization techniques such as dropout(Srivastava et al., 2014) layers. Dropout is a regularization technique where a predetermined number of nodes in a layer is randomly dropped, leading all incoming and outgoing connections from that node to be dropped as well. When implementing a dropout layer, it is common to exclude the nodes with a probability between $20 - 50\%$. Dropout effectively incorporates noise into both the input data and the hidden representations and forces a certain level of redundancy between the features learned at different hidden units. Such a redundancy leads to increased robustness.

2.4.2 CNN

An issue with fully connected layers is the huge number of parameters, causing long training time, and a danger of overfitting. Another type of neural networks, Convolutional Neural Network (CNN)s, have the potential to reduce the number of parameters. CNNs are designed to work with input data having strong spatial dependencies in local regions.

An obvious example is grid-structured 2D image data, but time series data is another type of data known to have various types of relationships between adjacent time points. The CNN and FFN function similar in many ways, except that the operations in the layers found in the CNN are spatially organized with sparse connections between the layers. There are three types of layers commonly present in a CNN: Convolutions, pooling and ReLU (equation 5), extracting spatial features from the data. In addition, a final set of dense layers, often fully connected, is used to map the spatial features onto a set of output nodes.

In 1D convolution layers, filters are slid across the input. The scalar product is calculated for each step at each filter. The filters are vectors of trainable parameters which are adjusted through gradient descend. The width of each filter, as well as the strides (i.e. step length for each slide), is predetermined. An additional trainable bias is added for each filter. The dimensions of the output data from a convolution layer is controlled by two factors: strides, and padding. In 1D convolution layers, padding consists of adding an appropriate number of columns to the beginning and the end of the input data. The padding makes it possible to center the convolution filters at every data point. There are two common types of padding configurations: same and valid padding. The *same* padding, pads in such a way that the output data have the same dimension as the input data. The *valid* padding, which means no padding, leads to a reduced output dimension compared to the input dimension. The outputs of a convolution layer are by convention referred to as a *feature map*. Each filter produces its own feature map, the filter width determines the range of spacial dependencies extracted and the number of spatial features extracted from the data is determined by the number of features. Multiple convolution layers can be stacked on top of each other, generating higher level spatial features.

The most common pooling layers encountered in the CNN are *max pooling* and *average pooling*. In max pooling, a window is slid over the feature maps and the algorithm outputs the maximum value within the window in each step, resulting in a reduced number of features. The width and strides of the windows are predetermined. The average pooling is similar, but instead of outputting the maximum value, it calculates and outputs the average of all values within each window. The activation function works in the same manner as for the FFN model. The most commonly activation function used in the convolution layers are the ReLU activation function (equation 5).

It should be noted that the CNN could also be considered as a feed-forward network, as the input parameters are propagated forward through the layers. To keep the two types of networks used in this thesis separated, only feed forward networks without convolutions layers will be referred to as FFN models.

2.5 Electrical activity in the neuron

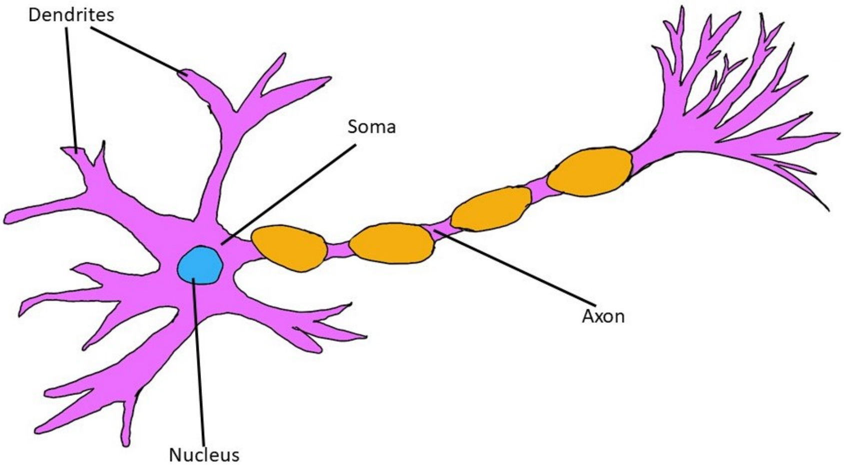


Figure 2.2: Illustration of a neuron.

Neuron cells are the fundamental building blocks of the nervous systems. The neuron cells can vary in shape and sizes, but the majority has three distinct parts: the soma, the dendrites and the axon. The cell body (soma) , which contains the nucleus and the majority of cytoplasmic organelles acts as the ”heart” of the neuron, generating electrical impulses that are transmitted through the axons. The dendrites extend from the soma, providing a large area for receiving synaptic input from other neurons. Most neurons have one axon, which in most cases extends much farther from the soma than the dendrites, and contacts other neurons (Squire, 2008). An illustration of the neuron can be seen in Figure 2.2.

The neuron, like most cells, is surrounded by a lipid bilayer, approximately 5 nm thick. The lipid bilayer has a hydrophobic part pointing both inwards to the intracellular space (i.e. the inside of the neuron), and the extracellular space (i.e. outside of the neuron), making the cell impermeable for the most water-solvable molecules (Galizia and Lledo, 2013). The separation of ions creates membrane potential, which allows for the electrical properties of the neuron. Some of the key ions playing a role for the membrane potential are positively charged cations such as sodium (Na^+), potassium (K^+), calcium (Ca^{2+}), magnesium (Mg^{2+}) and the negatively charged anion chloride (Cl^-) (Sterratt et al., 2011).

Ion pumps and ion channels are embedded in the lipid bilayer and allow for the transportation of ions through the membrane. Most ion channels and pumps are ion-specific, only allowing for certain ions to flow through the membrane. There are many types of ion channels, each of which has a particular permeability for each ion, that may change depending on different factors. The ion channels can be categorized into four types, depending on their behaviour: voltage gated ion channels, ligand gated ion channels, voltage- and ligand-gated ion channels and passive ion channels. The voltage gated ion channels mainly depend on the membrane potential. The flow of ions through a voltage gated ion channels activates gating currents, which changes the property of the ion channel, activating or inactivating a gate which controls the flow through the channel. The ligand gated ion channels can be activated by the binding of intracellular ligands, such as calcium and cyclic AMP (cAMP), and is mainly dependent upon the concentration of intracellular ligands. The voltage- and ligand gated ion channels can be activated by both the membrane potential and by binding of intracellular ligands. The passive ion channels permeability is constant (Galizia and Lledo, 2013).

As first described by A.L Hodgkin and A.F.Huxley, the activation of ion channels can lead to an Action Potential (AP). The flow of cations out of the neuron leads to increased polarization, meaning that the intracellular potential becomes more negative compared to the extracellular potential. The flow of cations into the cell leads to a depolarization (an increase of the membrane potential). The AP is characterized by a rapid depolarization followed by a less sharp repolarization period. Following the AP is an absolute refraction period and a relative refraction period. During the absolute refraction period, it is impossible to generate new APs. During the following relative refraction period, it is harder to generate new APs and APs generated in this period tends to have a lower peak voltage (i.e. amplitude) than the initial AP (Sterratt et al., 2011).

2.6 The Hodgkin-Huxley model

Alan Lloyd Hodgkin and Andrew Fielding Huxley began at the beginning of the 1950s to model the mechanics of a giant squid axon (Hodgkin and Huxley, 1952). They measured the membrane potential while controlling the flows of ions through the ion channels. By making some simplifications and representing the model as a one compartment RC-circuit (i.e. an electronic circuit composed of resistances and capacitors), they were able to describe how the membrane voltage would behave under different conditions. They arrived at a set of equations that collectively are called the Hodgkin-Huxley model. As a result of their work, they achieved a Nobel Prize in 1963. Even if more complex and precise models for describing the AP have been developed, the Hodgkin-Huxley model is still widely used in the field of computational neuroscience because of its low complexity and fast computation time (Sterratt et al., 2011).

2.6.1 The Model

$$C_m \frac{\partial V}{\partial t} = -\bar{g}_L(V - E_L) - \bar{g}_{Na} m^3 h (V - E_{Na}) - \bar{g}_K n^4 (V - E_K) + I$$

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \alpha_m(1-m) - \beta_m m \qquad \frac{\mathrm{d}h}{\mathrm{d}t} = \alpha_h(1-h) - \beta_h h \qquad \frac{\mathrm{d}n}{\mathrm{d}t} = \alpha_n(1-n) - \beta_n n$$

$$\alpha_m = 0.1 \frac{V - 40}{1 - e^{-\frac{(V+30}{10}}} \qquad \beta_h = \frac{1}{e^{-\frac{V+35}{10}} + 1}$$

$$\alpha_n = 0.01 \frac{V + 55}{1 - e^{-\frac{V+55}{10}}} \qquad \beta_n = 0.125 e^{-\frac{V+65}{80}}$$

$$V_{t=0} = V_{rest}$$

The Hodgkin-Huxley model tries to capture how the membrane potential is affected by the flow of ions through the cell membrane with a membrane capacitance of C_m. The key ions included are the sodium ion Na^+ and potassium K^+. Other ions are categorized as leak ions, which mostly consist of chloride ions Cl^-. The ions pass the membrane through ion specific ion channels creating ion currents that are expressed with the quasi-ohmic expression $I_x = \bar{g}_x \cdot (V - E_x)$ with a conductance of $\bar{g}_x$ and a reversal potential of E_x, x representing the ion. The probability of the gates being open is controlled by the gating variables m and h for the sodium and n for the potassium ion channel. The sodium and potassium ion channels are voltage gated, and the gating parameters are controlled by an activation rate (increased probability of being open), and an inactivation rate (decreased probability of being open), denoted as α_y and β_y for gating particle y. I represents the applied current.(Sterratt et al., 2011).

Analyses done with the resting reversal potential of sodium and potassium held constant have shown that the rapid rise leading to the AP is almost entirely due to the sodium conductance. After the AP peak, the potassium conductance takes a progressively more significant share of the importance, and the sodium conductance declines (Hodgkin and Huxley, 1952). through the quasi-ohmic expression $I_x = \bar{g}_x \cdot (V - E_x)$, it can be seen that varying the reversal potential will affect the flow of ions as well. Hence it is expected that the change of the sodium reversal potential will affect the sodium ion flow leading to the polarization of the membrane potential. Similarly, the reversal potential of potassium is likely to affect the depolarization following the AP.

2.7 The Pinsky-Rinzel model

The Pinsky-Rinzel model is a two-compartment model representing the soma and a dendrite and was developed by Paul F. Pinsky and J. Rinzel in 1994 (Pinsky and Rinzel, 2001). Despite the small number of compartments, the model can reproduce a variety of realistic activity patterns in response to somatic current injection or dendritic synaptic input. The model is a reduction from a 19-compartment model developed by Traub et al. (1991), but manage to mirror its behaviour. The reduction in size leads to a reduced computational cost, which is especially important when carrying out network simulations.

2.7.1 The Model

$$
\begin{aligned}
C_m \frac{\mathrm{d}V_s}{\mathrm{d}t} &= -\bar{g}_L(V_s - E_L) - g_{Na}(V_s - E_{Na}) - g_{DR}(V_s - E_K) + \frac{g_c}{p}(V_d - V_s) + \frac{I_s}{p}\\
C_m \frac{\mathrm{d}V_d}{\mathrm{d}t} &= -\bar{g}_L(V_d - E_L) - g_{Ca}(V_d - E_{Ca}) - g_{AHP}(V_d - E_K) - g_C(V_d - E_K) + \frac{g_c}{p}(V_s - V_d)\\
&\frac{g_c}{1-p}(V_s - V_d).
\end{aligned}
$$

$$
\begin{array}{lll}
g_{Na} = \bar{g}_{Na} m_{\infty}^{2} h & \alpha_m = -\dfrac{0.32V_1}{\exp\{-V_1/4\} - 1} & \beta_m = \dfrac{0.28V_2}{\exp\{V_2/5\} - 1} \\
 & \alpha_h = 0.128 \exp\left\{\dfrac{-43 - V}{18}\right\} & \beta_m = \dfrac{4}{1 + \exp\{-V_5/5\}} \\
g_{Dr} = \bar{g}_{Dr} n & \alpha_n = -\dfrac{0.016V_3}{\exp\{-V_3/5\} - 1} & \beta_m = 0.25 \exp\left\{\dfrac{-V_4}{40}\right\} \\
g_{Ca} = \bar{g}_{Ca} s^2 & \alpha_s = \dfrac{1.6}{1 + \exp\{-0.072(V - 5\}} & \beta_m = \dfrac{0.02V_6}{\exp\{V_6/5\} - 1} \\
g_C = \bar{g}_C c \chi([Ca^{2+}]) & \alpha_C = 0.0527(\exp\left\{\dfrac{V_8}{11} - \exp\left\{\dfrac{V_7}{27}\right\}\right\}) & \text{for } V \leq -10 \\
 & \alpha_C = 2\exp\{-V_7/27\} & \text{for } V > -10 \\
 & \beta_C = 2\exp\{-V_7/27\} - \alpha_C & \text{for } V \leq -10 \\
 & \beta_C = 0 & \text{for } V > -10 \\
 & \chi([Ca^{2+}]) = min([Ca^{2+}]/250, 1) & \\
g_{AHP} = \bar{g}_{AHP} q & q = min(0.0000[Ca^{2+}], 0.01 & \beta_q = 0.001
\end{array}
$$

$$
\frac{\mathrm{d}[Ca^{2+}}{\mathrm{d}t}] = -0.13 I_{Ca} - 0.075[Ca^{2+}]
$$

$$
\begin{gathered}
V1 = V + 46.9, V2 = V + 19.9, V3 = V + 24.9, V4 = V + 40, \\
V5 = V + 20, V6 = V + 8.9, V7 = V + 53.5, V8 = V + 50
\end{gathered}
$$

The Pinsky-Rinzel model tries to describe how the membrane potentials of a connected soma and a dendrite are affected by the flow of ions through their cell membranes, which both have a membrane capacitance of C_m. The percentage of the total area taken up by the soma is denoted by the parameter p. The connection between the soma and dendrite allows for an inter-compartment current also described as coupling current flowing between the soma and the dendrite. The coupling current is stated as the quasi-ohmic relationship $g_c(V_s - V_d)$, where the g_c represents the coupling conductance, and V_s and V_d represents the membrane potential of the soma and the dendrite, respectively. The soma is built up of the same elements as in the Hodgkin-Huxley model, with two voltage gated ion channels for transportation of sodium g_{Na} and potassium g_{DR} and a passive ion channel

$\bar{g}_L$ for transportation other ions referred to as leak ions. The ions affecting the dendritic membrane potential are potassium and calcium Ca^{2+}, in addition to leak ions.

Embedded in the dendritic membrane is a ligand gated ion channel allowing the flow of potassium g_{AHP} and a voltage gated ion channel allowing the flow of calcium g_{Ca}. There is also a voltage- and ligand gated potassium ion channel g_C, that is activated depending on the intracellular calcium concentration and the dendritic membrane potential. All of the ion channels have gating variables representing the probability of them being open. m, h represents the gating variables affecting the somatic sodium ion channel, and n,s,c and q represent the gating variables of the somatic voltage gated potassium ion channel, the dendritic voltage gated calcium ion channel, the dendritic voltage- and ligand gated potassium ion channel and dendritic ligand gated sodium ion channels, respectively. All gating parameters of voltage gated ion channels are controlled by an activation rate and an inactivation rate denoted as α_y and β_y for gating particle y. I_s represents the induced current in the soma. (Sterratt et al., 2011).

The initial burst sequence (sequence of APs) is initiated by a somatic sodium spike. The depolarization of the soma creates a potential difference between the two compartments, creating a coupling current depolarizing the dendritic membrane. During the repolarization phase of the soma (caused by the potassium current), the coupling current is shifted, which might initiate a second spike in the soma. A second spike may allow the dendrite to depolarize enough to initiate a calcium current, which further activates the ligand gated potassium channels leading to a rapid hyperpolarization of dendritic membrane. Which through the coupling current can lead to further depolarization of the soma (Pinsky and Rinzel, 2001).

2.8 Feature Importance

Feature importance measurement, also known as feature ranking, is a method used for feature selection in the data science community. It is closely linked to sensitivity analysis, that is the study of how uncertainty in the output of a model can be apportioned to different sources of uncertainty in the model input. One goal of sensitivity analysis is to find the relationship between input factors and output. Sensitivity techniques can be used for ranking the importance of various input parameters in terms of their influence on the variation of Y (Saltelli et al., 2008).

The PLSR regression coefficients are measures of the model sensitivity to the different input parameters, by providing direct measures of the influence of the input parameters on the output variables, and has been shown to be a robust feature importance measure. (Palermo, 2009)

Drop out feature importance is a technique commonly used in statistics and machine learning for ranking of feature importance. It is the ranking system that commonly drives the backwards/forward feature selection. The drop out feature scheme is illustrated in Figure 2.3. First, the model is trained with all available features, and the baseline error e_b is evaluated on the test set. The first feature is dropped from the feature set, a new model is trained, and a new error e_1 is evaluated on the test set. The first feature is then put back into the feature space, and the second feature is dropped, the model is retrained and the new error e_2 reevaluated. This iterative process continues until all features $f_1 \ldots f_n$ have been dropped once and the errors $e_1 \ldots e_n$ have been calculated. The feature importance is calculated as the error's subtracted from the baseline error $FI_i = e_b - e_i$. The problem with this technique is that it is computationally expensive. The time needed to calculate the drop out feature importance grows fast with the number of features since the model has to be retrained and optimized each time.

The permuted importance technique does not suffer from this problem. The technique was first proposed by Breiman in 2001 as an improvement of the random forest feature importance (Breiman, 2001). Based on this idea, Fisher, Rudin, and Dominic created a model-agnostic version of the feature importance (Fisher et al., 2018). Model-agnostic meaning it can be used with any machine learning model. It is important to keep in mind

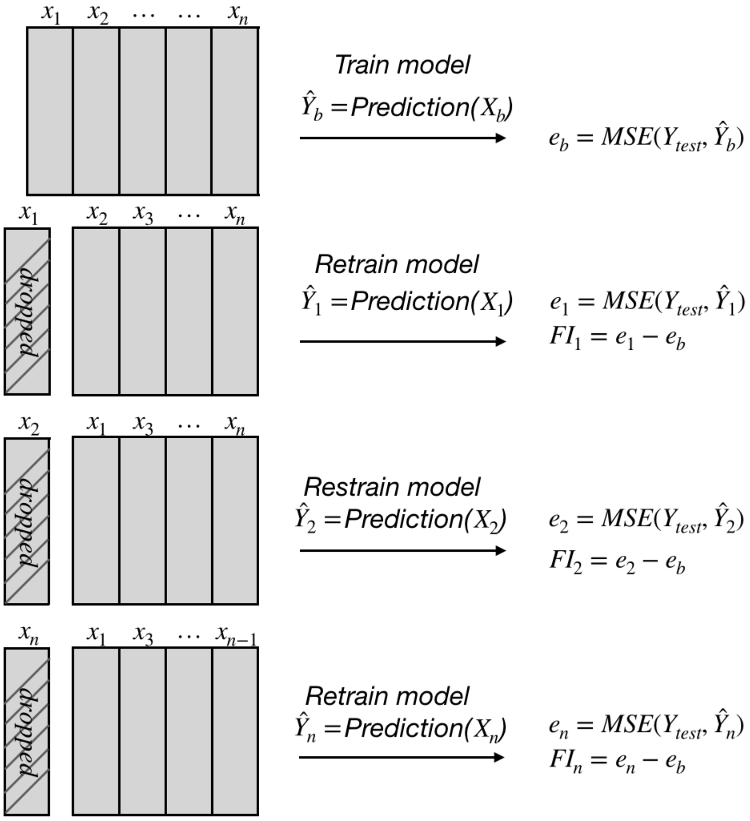

x_1 x_2 … … x_n
Train model
$\hat{Y}_b$ =Prediction(X_b)
$e_b = MSE(Y_{test}, \hat{Y}_b)$
x_1 dropped
x_2 x_3 … x_n
Retrain model
$\hat{Y}_1$ =Prediction(X_1)
$e_1 = MSE(Y_{test}, \hat{Y}_1)$
$FI_1 = e_1 - e_b$
x_2 dropped
x_1 x_3 … x_n
Restrain model
$\hat{Y}_2$ =Prediction(X_2)
$e_2 = MSE(Y_{test}, \hat{Y}_2)$
$FI_2 = e_2 - e_b$
x_n dropped
x_1 x_3 … x_{n-1}
Retrain model
$\hat{Y}_n$ =Prediction(X_n)
$e_n = MSE(Y_{test}, \hat{Y}_n)$
$FI_n = e_n - e_b$

Figure 2.3: Illustration of the drop out feature importance algorithm

that the permuted importance technique suffers from a bias towards input parameters with high correlation between them. Two input parameters explaining the same behaviour runs the risk of one of them having reduced feature importance.

The permuted importance technique scheme is illustrated in Figure 2.4. The model is first trained and optimized with the training set, using the entire feature space. A baseline e_b error is then calculated on the test set. Then, instead of dropping the first feature as in the drop out technique, the feature column of the test set is permuted and the prediction error e_1 is calculated using the same model. The first feature is then reordered and the second permuted. The process continues until the prediction errors $e_1 \ldots e_n$ have been calculated. The feature importance is calculated as the error's subtracted from the baseline error $FI_i = e_b - e_i$. The permuted importance technique does not suffer from the need to train multiple models, i.e. it has a less computational cost than the drop out model.

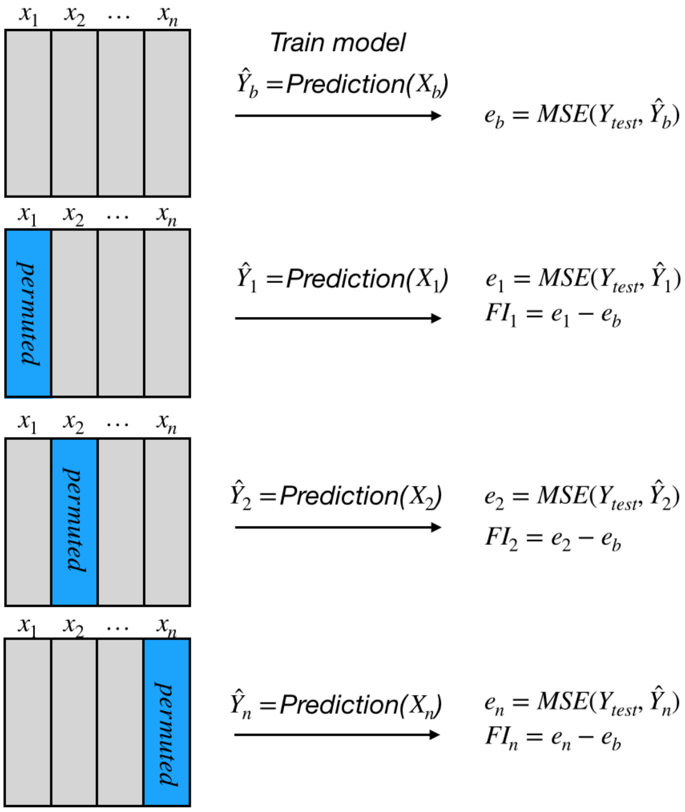

x_1 x_2 … x_n
Train model
$\hat{Y}_b$ =Prediction(X_b)
$e_b = MSE(Y_{test}, \hat{Y}_b)$
permuted
$\hat{Y}_1$ =Prediction(X_1)
$e_1 = MSE(Y_{test}, \hat{Y}_1)$
$FI_1 = e_1 - e_b$
$\hat{Y}_2$ =Prediction(X_2)
$e_2 = MSE(Y_{test}, \hat{Y}_2)$
$FI_2 = e_2 - e_b$
$\hat{Y}_n$ =Prediction(X_n)
$e_n = MSE(Y_{test}, \hat{Y}_n)$
$FI_n = e_n - e_b$

Figure 2.4: Illustration of the Permuted Feature Importance algorithm

3 Methods

The code created in this thesis is available at github:
https://github.com/odegaardmaster/Masterthesis

3.1 Data generation

The two sets of membrane potentials analyzed in this paper were simulated with the Hodgkin-Huxley model and the Pinsky-Rinzel model. The Hodgkin-Huxley model was implemented in python using the *odeint* module from the *scypy* package. The Pinsky-Rinzel model (Pinsky and Rinzel, 2001) used in this study was obtained from ModelDB (`http://modeldb.yale.edu` accession number 35358) and implemented and carried out using python and the *neuron* package with time adaptive time-steps. Interpolation was used to find the missing time steps. In addition to the simulated time series, statistics such as time to first AP, the aggregated phenotypes AP width and amplitude was calculated. The threshold defining an AP was chosen to $-20mV$. The width of the AP was calculated as half of the APs prominence height. An example of the calculations is shown in Figure 3.1.

3.1.1 The Hodgkin-Huxley model

In the simulations of the Hodgkin-Huxley model, the Resting potential V_{rest}, was held constant, while 7 variables were varied using Latin Hypercube Sampling (LHS). The LHS is a semi-random sampling procedure that is especially suitable for use on high-dimensional data, since it separates the data into several hypercubes and samples randomly within each hypercube. More details of LHS can be found in the original paper by Mckay et al. (2000). The range within which the parameters were varied was chosen to be ±20% of the default values shown in Table 1. The range of each input variable was stratified into three intervals with equal probability, and one observation was drawn from each interval,

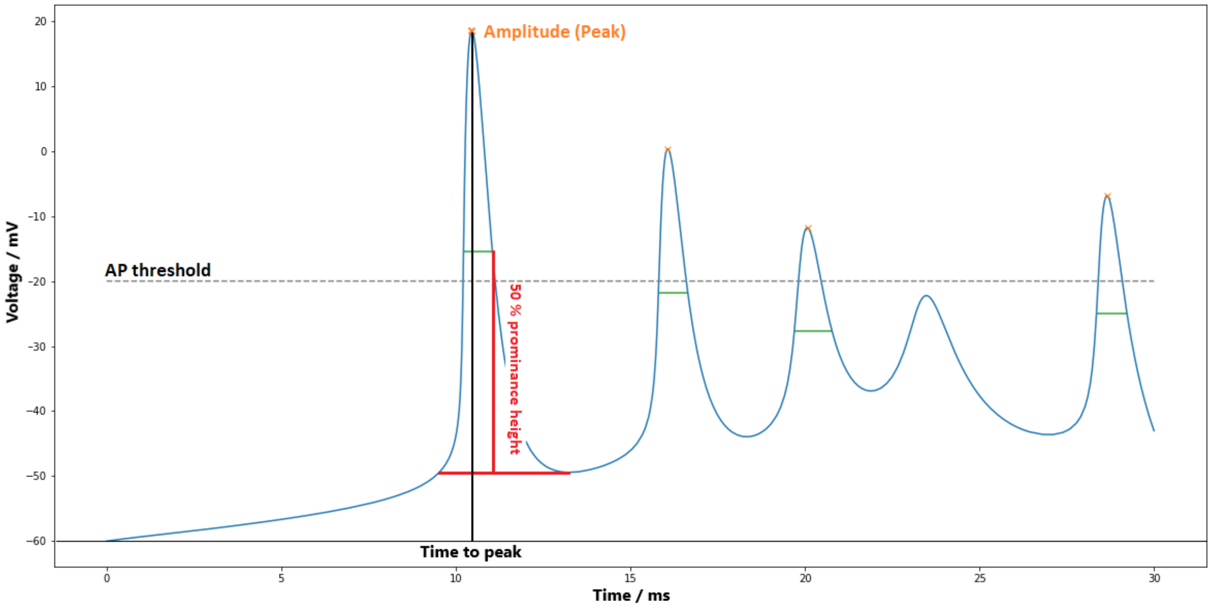


Figure 3.1: Calculation of aggregated phenotypes. The AP threshold is set to $-20mV$, the *width* (green line) is calculated at 50% prominence height. *Time to peak* and *amplitude* is calculated at the time of the AP peak.

resulting in 2187 parameter combinations. The model was exposed to a continuous external stimulus of 140 $\mu A/cm^2$ starting at $t = 0$. The membrane potentials were simulated for $30ms$ with a resolution of 0.025 ms, resulting in 1201 time points. As of the parameter combinations resulted generated at least one AP during the $30ms$, the aggregated phenotypes were extracted from the first AP, in all 2817 samples.

Table 1: Default values for simulations with the Hodgkin Huxley model

Parameter	Value	Description
V_{rest}	$-65\ mV$	Resting Potential (not varied)
C_m	$1\ \mu Fcm^{-2}$	Membrane capacitance
$\overline{g}_{Na}$	$120\ mScm^{-2}$	Sodium conductance
$\overline{g}_K$	$36\ mScm^{-2}$	Potassium conductance
$\overline{g}_L$	$0.3\ mScm^{-2}$	Leak ions conductance
E_{Na}	$50\ mV$	Sodium reversal potential
E_K	$-77\ mV$	Potassium reversal potential
E_L	$-54.4\ mV$	Leak ions reversal potential

3.1.2 The Pinsky-Rinzel model

In the Pinsky-Rinzel model, 13 parameters were varied using LHS. The range within which the parameters were varied was as for the Hodgkin-Huxley model chosen to be ±20% of the default values shown in Table 2. The range of each input variable was stratified into three intervals with equal probability, and one observation was drawn from each interval, resulting in 1594323 parameter combinations. The simulations were run over a time span from 0 to 30 ms with a resolution of 0.025 ms. The soma membrane potential was recorded during the simulation, resulting in 1201 time steps per somatic membrane potential. The large number of samples resulted in a huge training time for the metamodels especially the clustering method used in the HCPLSR model. The number of samples was therefore reduced by randomly sampling of 100000 samples, resulting in an input matrix of dimension 100000 X 13, and an output matrix of dimension 100000 X 1201. Not all parameter combinations resulted in APs during the first 30 ms. The aggregated phenotypes of the first occurring AP were calculated for the 81250 parameter combinations which achieved an AP within the first 30 ms.

Table 2: Default values for simulation of Pinsky Rinzel model

Parameter	Default Value	Description
C_m	$3.0\ \mu Fcm^{-2}$	Membrane capacitance
$\overline{g}_L$	$0.1\ mScm^{-2}$	Leak ion conductance
$\overline{g}_{Na}$	$30\ mScm^{-2}$	Somatic sodium ion conductance
E_L	$-60\ mV$	Leak ion reversal potential
E_{Na}	$50\ mV$	Sodium reversal potential
E_K	$-75\ mV$	Potassium reversal potential
$\overline{g}_c$	$2.1\ mScm^{-2}$	Coupling conductance
$\overline{g}_{kdr}$	$1.0\ mScm^{-2}$	Dendritic potassium conductance (kdr)
$\overline{g}_{AHP}$	$0.8\ mScm^{-2}$	Dendritic potassium conductance (AHP)
$\overline{g}_C$	$15\ mScm^{-2}$	Dendritic potassium conductance (C)
$\overline{g}_{Ca}$	$10\ mScm^{-2}$	Dendritic calcium conductance
E_{Ca}	$80\ mV$	Calcium reversal potential
p	0.5	Proportion of the area taken up by the soma

3.2 Metamodelling using HCPLSR

The implementation of HCPLSR used in this thesis was based on the original paper by Tøndel et al. (2013). The entire data set was split into a test set (30%) and a training set (70%). The training data, both the input matrix X and output matrix Y, was standardized by subtracting the mean, and dividing by the standard deviation for each column $X_i = \frac{X_i - \mu_i}{\sigma_i}$. A PLSR model, referred to as the global PLSR model, was calibrated using the training samples. The global PLSR model was made using 10-fold cross-validation (i.e using the mean of ten cross-validations). The optimal number of global PLSR PCs was chosen with the requirement that each global PLSR PC should account for at least 1% of the total cross-validated Y-variance, leaving the rest of the components as noise. The global PLSR model was used as a basis for the clustering operation. Fuzzy C-means (FCM)(Bezdek et al., 1984) clustering based on either the X- or Y- scores were used to separate the data into a preset number of subsets (clusters). The fuzziness parameter m of the FCM was chosen to 2. The subsets were then re-standardized by calculating a separate mean and standard deviation for each subset in order to gain an increased interpretation ability. For each subset, a new PLSR model, referred to as local PLSR models, was computed, using the same techniques and requirements as in the global model. The test set was standardized using the mean and standard deviations calculated for the training set. The input matrix of the test data X_{test} was then projected into the global PLSR model, resulting in the decomposed X- or Y-scores. Based on the score matrices, each test set sample was then classified into the most probable cluster using FCM. A final prediction of the output $\hat{Y}_{test}$ was predicted using the local PLSR model trained on the most probable cluster. An exception was that if the euclidean distance from an observation to the most probable cluster was more than 1.5 times the maximum distance observed in the test set. Then the global PLSR model was used to create the prediction.

3.2.1 Metamodelling of the Hodgkin-Huxley model and the Pinsky-Rinzel model

Multiple HCPLSR metamodels with different setups were trained for the classical metamodelling of the Hodgkin-Huxley model. Both the global X and Y-scores were tested as clustering basis for the FCM clustering. The input data was varied between using only the main effects of the input parameters, including cross- and interaction terms of the main effects, including sinus and cosinus terms, and including the combination of both cross- and interaction terms and the sinus and cosinus terms. The purpose of adding cross- and interaction terms serves to identify interactions between the main effects while using sinus and cosinus terms might improve the metamodelling of abrupt non-linearities in the data.

For each combination of clustering basis and input data, the number of clusters used in the FCM clustering method was varied from two to ten clusters. Resulting in a total of 72 HCPLSR metamodels trained for the classical metamodelling of the Hodgkin-Huxley model. The first time point ($0ms$) was taken out of the output matrix, as the PLSR model does not function when there is no variation in the data. The same metamodelling setups were used for the classical metamodelling of the Pinsky-Rinzel model, with the exception of using the combination of both cross- and interaction terms and sinus and cosinus terms. Resulting in a total of 54 HCPLSR metamodels trained.

The different setups of the HCPLSR metamodels used for the inverse metamodelling were varied between using the global X and Y-scores as clustering basis and for each clustering basis, the number of clusters was varied from two to ten. This resulted in 20 metamodels trained for the inverse metamodelling of both the Hodgkin-Huxley model and the Pinsky-Rinzel model.

The classical metamodelling of the aggregated phenotypes was conducted using the same HCPLSR setups as for the classical metamodelling of the Pinsky-Rinzel model, This resulted in a total of 54 HCPLSR metamodels trained for each model.

3.3 Metamodelling using deep learning

The classical metamodelling of the time series data and the aggregated phenotypes was conducted using FFN models, and the inverse metamodelling was conducted using CNN models. A wide range of different topologies (i.e the way the network is connected) and activation functions was explored before landing on a final architecture. The entire data set was randomly test set (30%) and a training data (70%). Further, the training data was divided into a training set (80% of training data) and a validation set (20% of the training data) when exploring for the best suitable topology, an optimal number of epochs, and the batch size used in training the models. By using a validation set, instead of the test set, when exploring the optimal setups, the test set is kept unseen for the model, avoiding information leaks (Aggarwal, 2018).

The deep learning metamodelling was done with Keras on Google Colaboratory (Pessoa et al., 2018).

3.3.1 Metamodelling of the Hodgkin-Huxley model and the Pinsky-Rinzel model

The architectures of the FFN and CNN models used for metamodelling of the two single neuron activity models are described in Appendix A. The ReLU function (equation 5) was used as an activation function in all hidden layers in the FFN models, and in all convolutions layers in the CNN models. In the fully-connected layers following the convolution layers in the CNN models, tanh (equation 4) was used. All metamodels were treated as regression problems, therefore a linear activation function (equation 6) was used in the output layer. Adam was used as the optimizer in all deep learning models used in this thesis.

The FFN model used in the classical metamodelling of the Hodgkin-Huxley model (architecture shown in Figure A.1) was a simple one, consisting of two hidden fully connected layers, both with a width of 28 neurons. In classical metamodelling of the Pinsky-Rinzel model an FFN model with four hidden layers, all fully connected was used. The architecture is shown in Figure A.4).

The architecture of the CNN model used for inverse metamodelling of the Hodgkin-Huxley model is shown in Figure A.2. Four convolution layers, a max pooling layer and a global average pooling are used in this model. The convolution layers are *same* padded(leading to no reduction of data size). The *same* padding was used since much of the information is expected to be in the first couple of ms, which could be lost if other padding methods were used. To connect the feature map with the output layer a fully connected layer with seven neurons is used. A similar architecture is used in the inverse metamodelling of the Pinsky-Rinzel model (figure A.5). However, the filters used in the first two convolution layers are wider, allowing for wider spatial dependencies. Moreover, *valid* padding is used in the convolution layers. In order to connect the feature map to the output, three fully connected layers were used.

The metamodelling of the aggregated features extracted from the Hodgkin-Huxley model (shown in figure A.3) consists of one fully connected layer with nine neurons. The model stated for the metamodelling of the aggregated features extracted from the Pinsky-Rinzel model had two fully connected layers, both with a width of 20 neurons.

3.4 Permuted Feature Importance

The permuted feature importance was calculated with multiple permutations each time in order to adjust for the random effects, and the mean and standard deviation were calculated for each permuted feature importance.

In an effort to describe the behaviour surrounding the first occurring AP, a permuted feature importance pipeline using aligned APs was developed. Parameter combinations which generated its first AP in the time span between 5ms and 25ms were selected from the samples. The predicted membrane potential from the unpermuted data and the true membrane potential were aligned such that the AP peak occurred at $t = 0$ (illustrated in Figure 3.2. The baseline error calculated for the unpermuted data was reduced to a 10 ms window, with the time of the predicted AP peak centralized at 0 ms, the centralization is illustrated in Figure 3.2. For each permutation, the predicted APs was also centralized at 0 ms, as illustrated in Figure 3.3, and the feature importances were calculated for a time window of 10 ms.

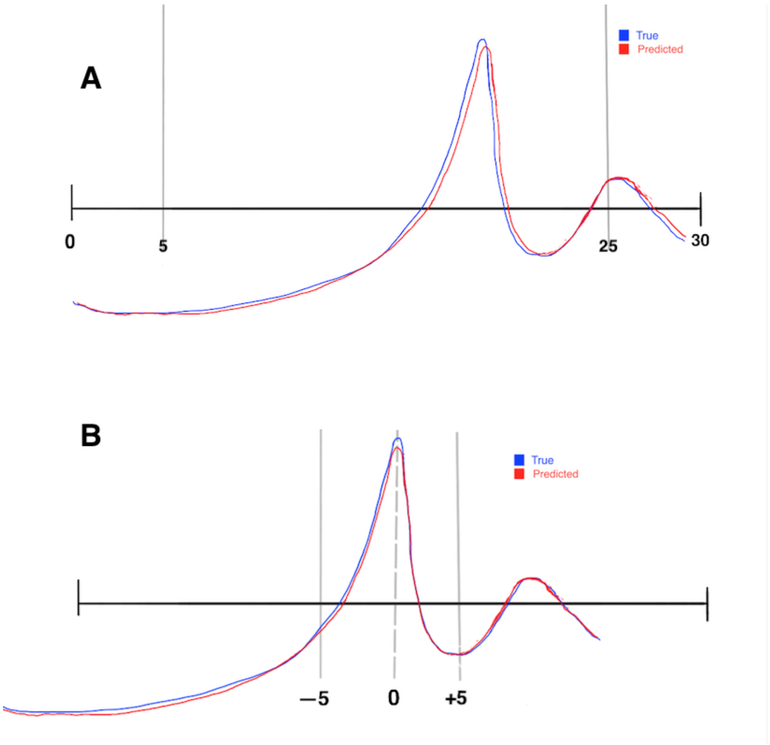


Figure 3.2: Centering of predicted membrane potentials.

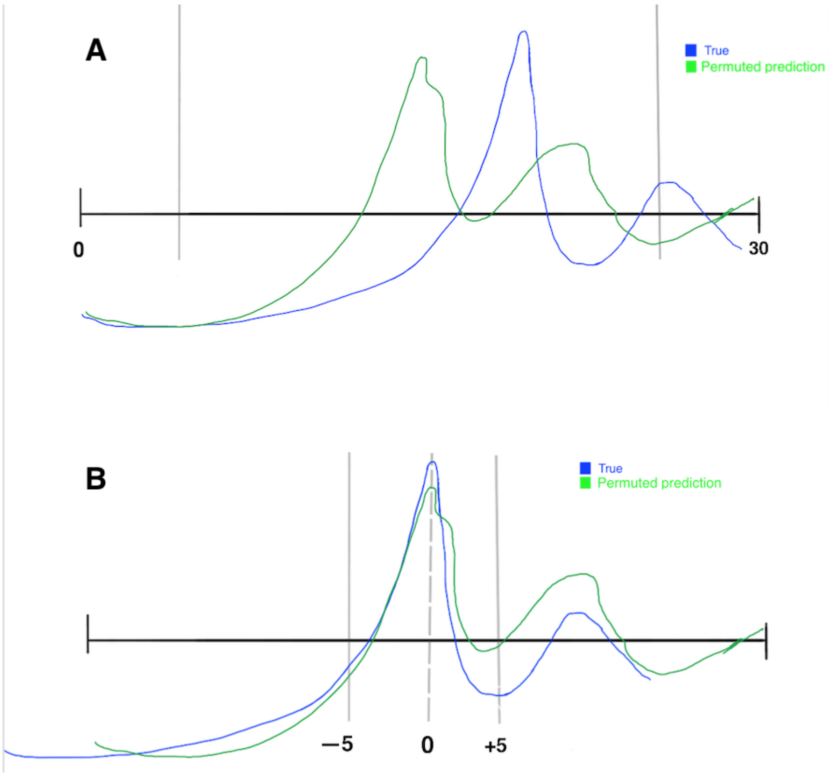


Figure 3.3: Centering of predicted membrane potentials after permuting of input column.

4 Results

4.1 Metamodelling of the Hodgkin-Huxley model

4.1.1 Data Generation

The 2817 membrane potentials that were obtained from simulations with the Hodgkin-Huxley model, are shown in Figure 4.1 A. For a better overview of the differences in behaviour, the mean of the 2187 simulations together with the standard deviation is shown in Figure 4.1 B. All parameter combinations triggered the first AP in roughly the same time ($0.0625 ms$). Not all parameter combinations triggered multiple APs. Therefore most of the variance between the simulations occurs after the first AP. The input parameters used in the simulations had no significant correlation between them. The correlation matrix for the input parameters is shown in Appendix C Figure C.1.

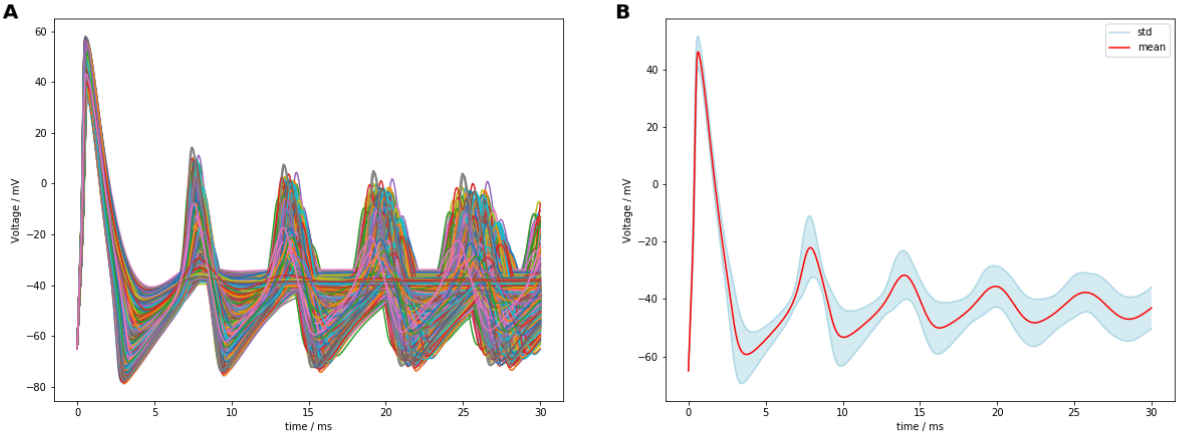


Figure 4.1: Plot of membrane potentials generated with the Hodgkin-Huxley model. A) The 2817 simulated membrane potentials with the Hodgkin-Huxley model. **B)** Mean and standard deviation of the 2817 membrane potentials simulated with the Hodgkin-Huxley model. red: mean, blue: Standard deviation.

4.1.2 Classical metamodelling of the Hodgkin-Huxley model

A classical metamodelling with HCPLSR and FFN was carried out in order to analyze the complex input-output relationship of the Hodgkin-Huxley model. Multiple HCPLSR models were trained, using both Y- and X-scores matrices as clustering basis in the FCM clustering. The number of clusters used in the FCM clustering was varied from two to ten. Additional sinus and cosinus terms and/or interaction and second-order terms were added to the parameters to capture possible interactions and abrupt non-linearities. Using the global X-scores matrix as the clustering basis with added second-order and interaction terms was found optimal for prediction of the membrane potentials. The prediction accuracy did not increase significantly when using more than four clusters in the metamodelling. Therefore four clusters were chosen for further exploration of the Hodgkin-Huxley model. The test set prediction accuracies achieved are plotted in Figure 4.2.

The variance of the output matrix explained by each PC is declining. Therefore the first few components are expected to be of most relevance. The first three PCs in the global PLSR model explained 84.5% of the total cross-validated variance present in the output matrix. The fourth PC did not increase the explained variance of the first three PCs with more than one per cent, hence three PC was chosen as the optimal model rank, used as a clustering basis. The X-scores for the first three PCs of the global PLSR model are plotted in Figure 4.3 A, and the four clusters later used for calibrating the local PLSR models are colourized. The percentage of the cross-validated variance explained by each PC is indicated. The mean and standard deviation for the membrane potentials picked out of cluster $1-4$ are plotted in Figure 4.3 B-E, respectively. The main effect regression coefficients for the global PLSR and the four local PLSR models are plotted in Figure 4.4A and 4.4B-E, respectively. The 15 most significant cross- and second order terms regression coefficients for the global PLSR and the four local PLSR models are plotted in Figure 4.4A and 4.4B-E, respectively.

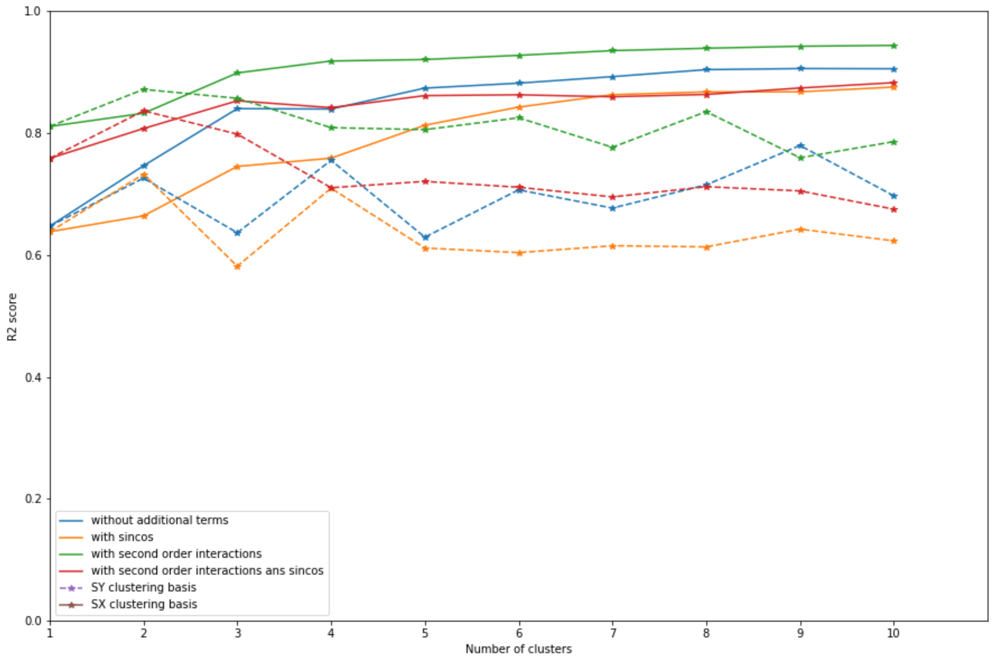


Figure 4.2: Optimizing the number of clusters and cluster matrix using HCPLSR in classical metamodelling of the Hodgkin-Huxley model. The average prediction accuracy (R^2) of the test data, given the number of clusters, clustering matrix and added terms. The average prediction accuracy achieved by the global PLSR model is noted as one cluster. Blue: Only input parameters. Orange: Input parameters with additional sinus and cosinus terms. Green: Input parameters with interaction- and second-order terms. Red: Input parameters with interaction- and second-order terms, with additional sinus and cosinus terms. The dashed line and the solid line represents Y- and X-scores, respectively, used as cluster matrix in the FCM clustering.

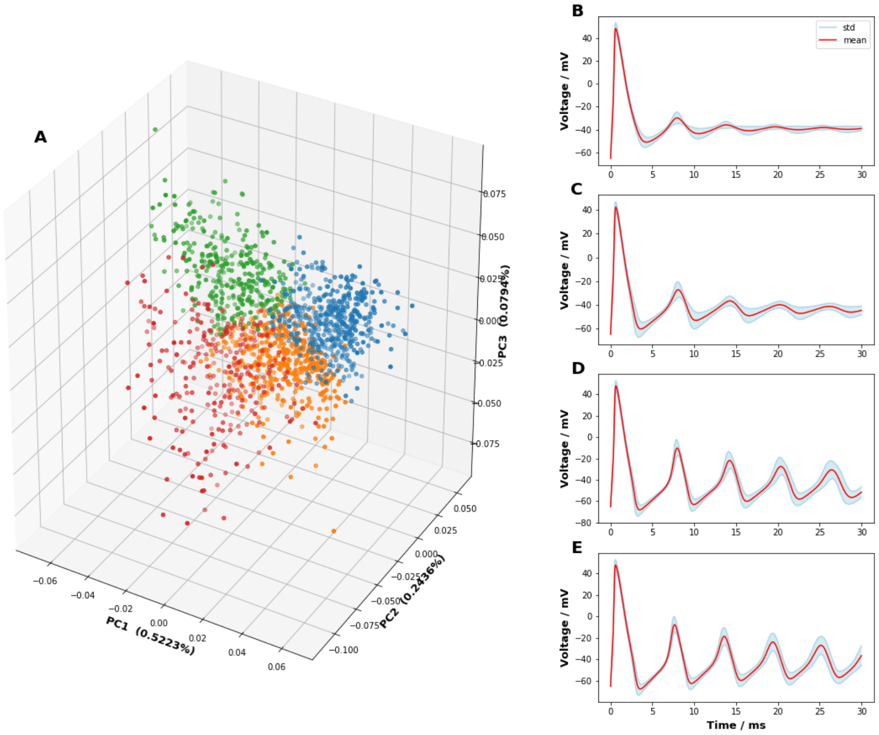


Figure 4.3: Clustering results from HCPLSR metamodelling of the Hodgkin-Huxley model using four clusters. A) Scatter plot of the X-scores from the global metamodel The samples are coloured according to cluster membership. Cluster1=blue, cluster2=orange,cluster3=green, cluster4=red. **B-E)** Mean (red) and standard deviation (blue) of the simulated membrane potentials for cluster 1-4, respectively.

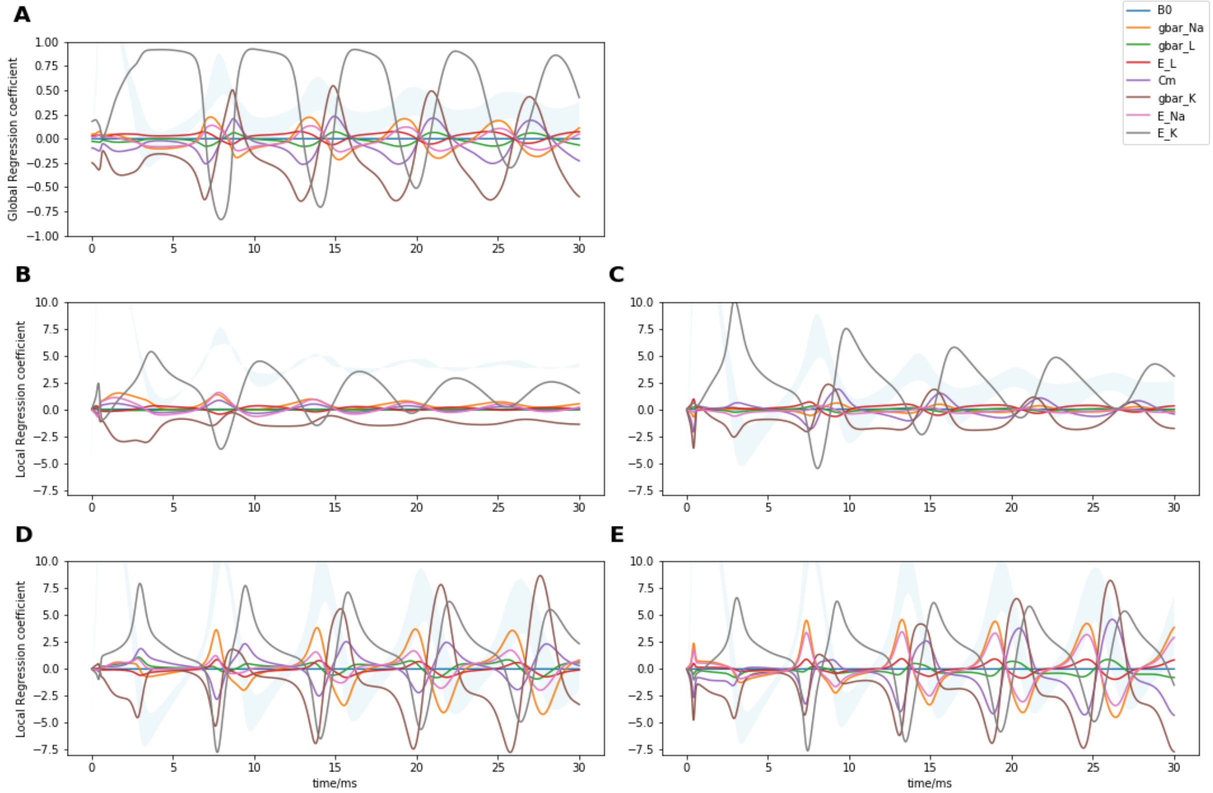


Figure 4.4: Global and local regression coefficients for the input parameters of the Hodgkin-Huxley model. The regression coefficients show the sensitivity of the input parameters over time. **A)** shows the global regression coefficients. **B-E)** shows the local regression coefficients for cluster 1-4, respectively. The standard deviation of the membrane potentials belonging to each individual clusters is plotted off scale in the background for comparison.

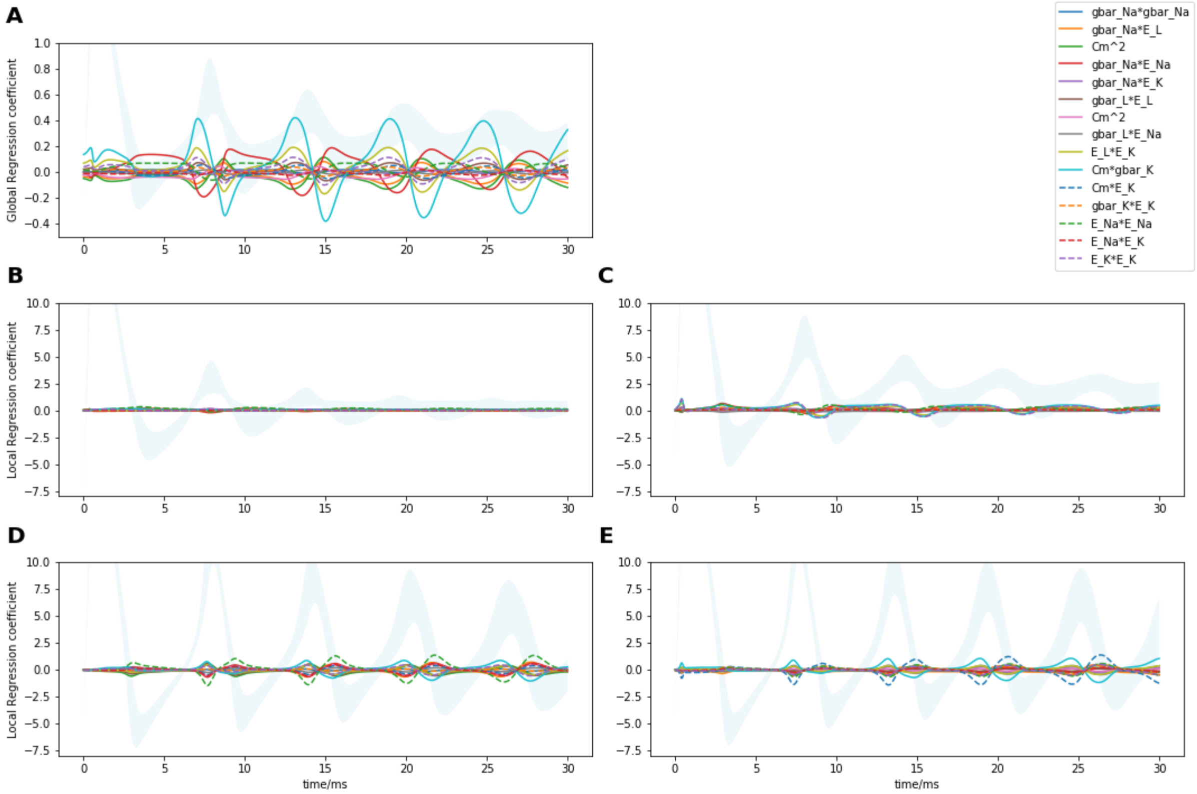


Figure 4.5: Global and local regression coefficients for the cross- and second order terms of the input parameters of the Hodgkin-Huxley model. The regression coefficients shows the sensitivity of the 15 most significant cross- and second order terms parameter over time. **A)** show the global regression coefficients. **B-E)** shows the local regression coefficients for cluster 1-4, respectively. The standard deviation of the membrane potentials belonging to each individual cluster is plotted off scale in the background for comparison.

The training plots used for hyperparameter tuning of the FFN model stated for the classical metamodelling are plotted in Appendix B Figure B.3. The final FFN model was trained for 1500 epochs and achieved a test set prediction of 0.993.

The feature importance ranking of the input variables of the Hodgkin-Huxley model was calculated with five permutations for each variable, using the permuted feature importance procedure (described in Section 2.8). This is illustrated in Figure 4.6. For an exploration of the importance of the parameters regarding the first occurring AP, the feature importance ranking for the first 5 *ms* is plotted in Figure 4.7.

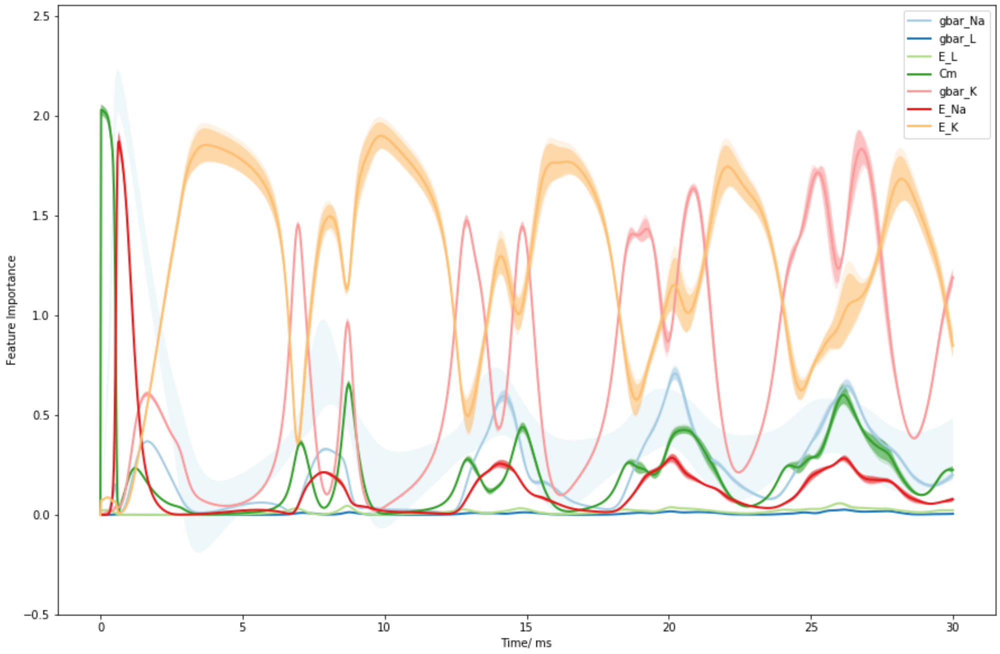


Figure 4.6: Feature importance for the input parameters of the Hodgkin-Huxley model. The feature importance is calculated with five permutations of each input variable. The solid line represents the mean with the standard deviation are plotted as a transparent extension. The standard deviation of the membrane potentials simulated is plotted off scale in the background for comparison.

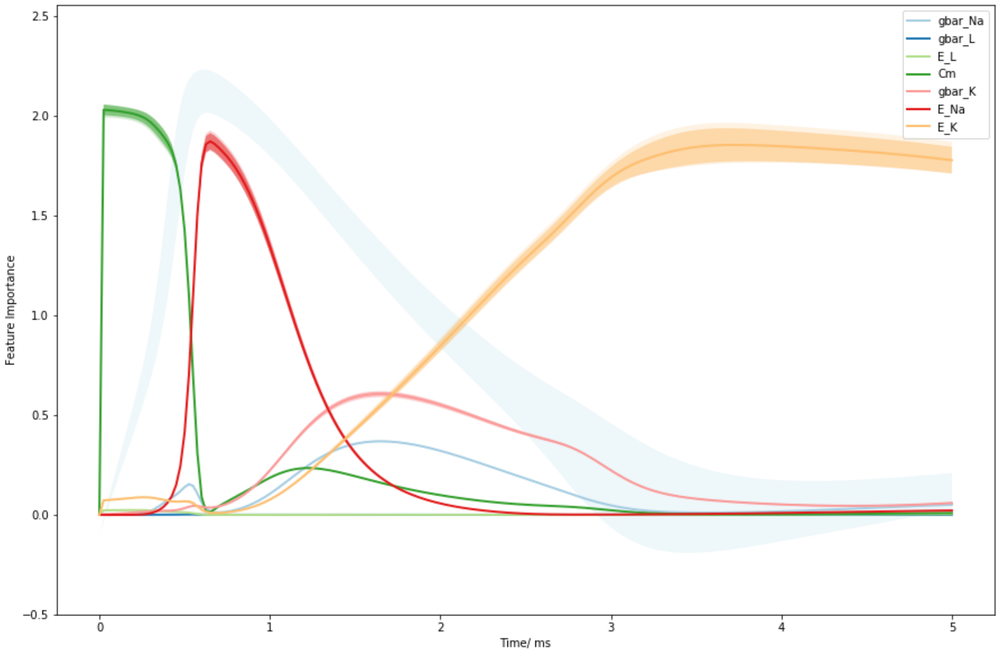


Figure 4.7: Feature importance for the input parameters of the Hodgkin-Huxley model for the first 5 ms. The time range of Figure 4.6 is narrowed down to 5 ms in order to increase the exploratory ability of the behaviour surrounding the first AP. The first AP occurs at roughly 0.625 ms. The standard deviation of the membrane potentials simulated is plotted off scale in the background for comparison.

4.1.3 Inverse metamodelling of the Hodgkin-Huxley model

The inverse metamodelling of the Hodgkin-Huxley model was carried out with HCPLSR and CNN models, in order to predict the output-input relationships of the model.

As in the classical metamodelling, both the Y scores matrix and the X scores matrix were explored as clustering basis for the HCPLSR model. Figure 4.8 shows the test set prediction accuracies of the input parameters of the Hodgkin-Huxley model, for number of clusters varied from two to ten. The best average prediction accuracy was achieved using five clusters and is listed in Table 3.

The training plots used for hyperparameter tuning of the CNN model stated for inverse metamodelling of the Hodgkin-Huxley model can be seen in Figure B.2. The final CNN model was trained for 1250 epochs, and the prediction accuracies of the test data are listed in Table 3.

Table 3: Prediction accuracies (R^2) for inverse metamodelling of Hodgkin-Huxley model using CNN and HCPLSR.

	CNN	HCPLSR
$\bar{g}_{Na}$	0.8834	0.8497
$\bar{g}_L$	0.8199	0.0045
E_L	0.0432	0.006
C_m	0.9152	0.4025
$\bar{g}_K$	0.8752	0.7256
E_{Na}	0.9103	0.9307
E_K	0.9146	0.9689

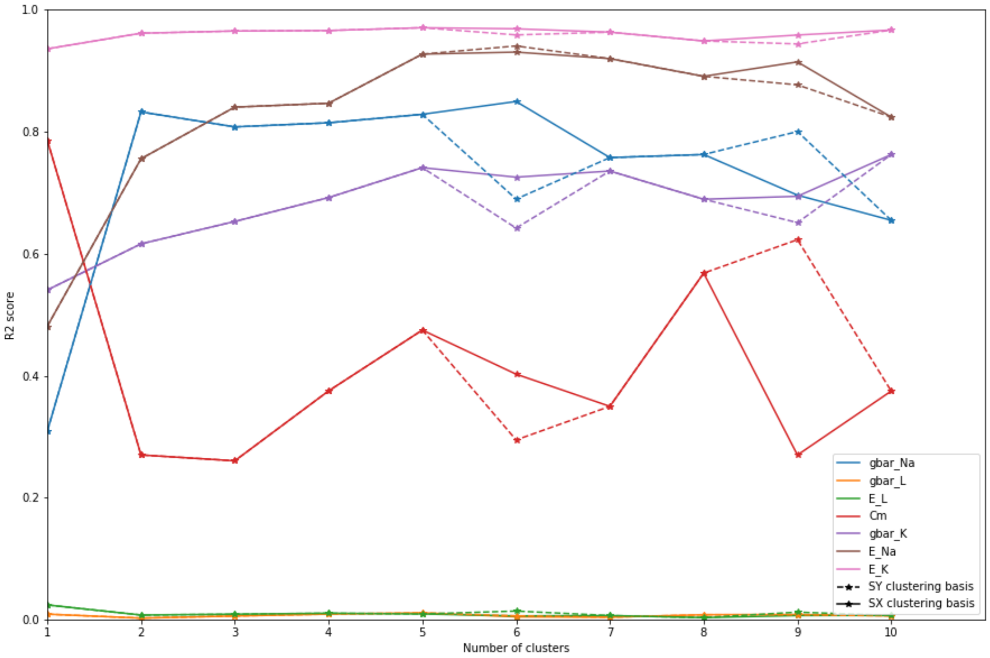


Figure 4.8: Optimizing the number of clusters and cluster matrix using HCPLSR in inverse metamodelling of the Hodgkin-Huxley model. The prediction accuracy (R^2) of the input parameters in the test data, given the number of clusters and clustering matrix. The prediction accuracy achieved by the global PLSR model is noted as one cluster. The dashed line and the solid line represent Y- and X-scores, respectively, used as cluster matrix in the FCM clustering.

4.1.4 Classical metamodelling of the aggregated phenotypes

The classical metamodelling of aggregated phenotypes extracted from the Hodgkin-Huxley model was carried out using FFN and HCPLSR. The aggregated phenotypes extracted from the Hodgkin-Huxley model cover ”time to peak”, ”width of first AP” and ”amplitude of first AP”. As all simulations generated at least one AP, all samples were used in the metamodelling.

Both Y- and X-scores were explored as clustering basis in the FCM clustering. The number of clusters used in the FCM clustering was varied from two to ten clusters. The test set prediction accuracies achieved using the HCPLSR metamodelling are plotted in Figure 4.9. The best average prediction accuracy was achieved using X-scores as clustering basis with six clusters and additional cross-terms and second-order terms. The individual prediction accuracy for each of the aggregated phenotypes when using these clusters are listed in Table 4. The main effect regression coefficients for the six local PLSR models are plotted in Figure 4.10, and the regression coefficients for the 15 most significant interaction- and second order terms are shown in Figure 4.11.

The training plot used for hyperparameter tuning of the FFN model stated for the classical metamodelling of the aggregated phenotypes is plotted in Appendix B Figure B.3. The final FFN model was trained for 1500 epochs, and the prediction accuracies of the test data are listed in Table 4.

The feature importance of the input variables was calculated as the mean of five permutations for each variable using the permuted feature importance procedure and is plotted in Figure 4.12.

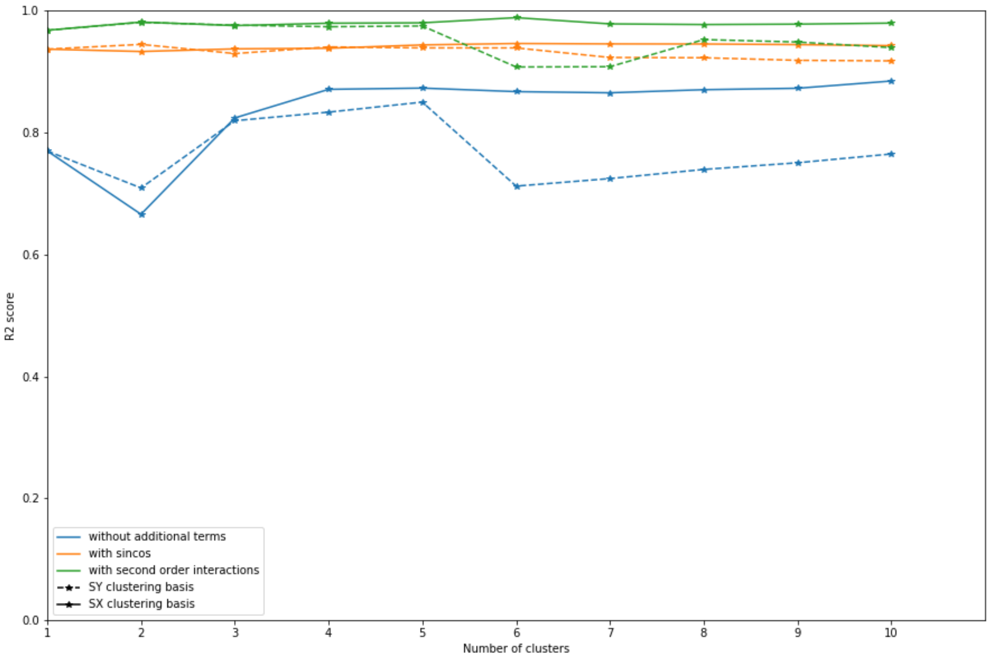


Figure 4.9: Optimizing the number of clusters and cluster matrix using HCPLSR in metamodelling of the aggregated phenotypes of the Hodgkin-Huxley model. The prediction accuracy (R^2) of the aggregated phenotypes in the test set, given the number of clusters and clustering matrix. The prediction accuracy achieved by the global PLSR model is noted as one cluster. The dashed line and the solid line represents Y- and X-scores, respectively, used as cluster matrix in the FCM clustering.

Table 4: Prediction accuracies (R^2) for metamodelling of aggregated phenotypes extracted from the Hodgkin-Huxley model.

	FNN	HCPLSR
time to peak	0.9765	0.9858
width of first AP	0.9996	0.9980
amplitude of first AP	0.9975	0.9826

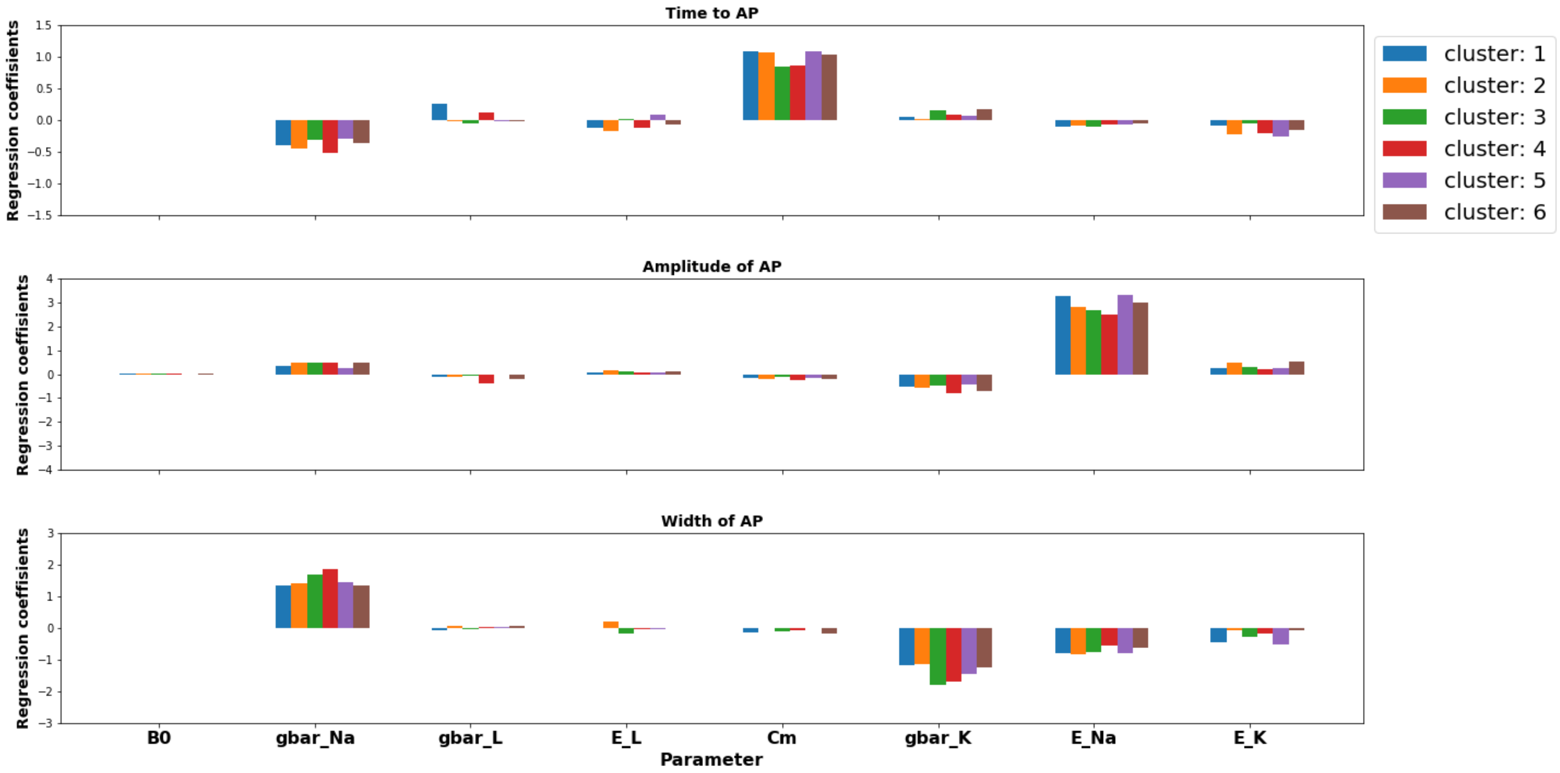


Figure 4.10: Local regression coefficients for the main effects of the aggregated phenotypes extracted from the Hodgkin-Huxley model. The regression coefficients show the sensitivity of the input parameters. **The top figure** shows the regression coefficients for the time taken for the membrane potential to reach the first AP. **The middle figure** shows the regression coefficients for the amplitude reached by the first AP. **The bottom figure** shows the regression coefficients for the width of the first AP. The regression coefficients are coloured according to their respective clusters.

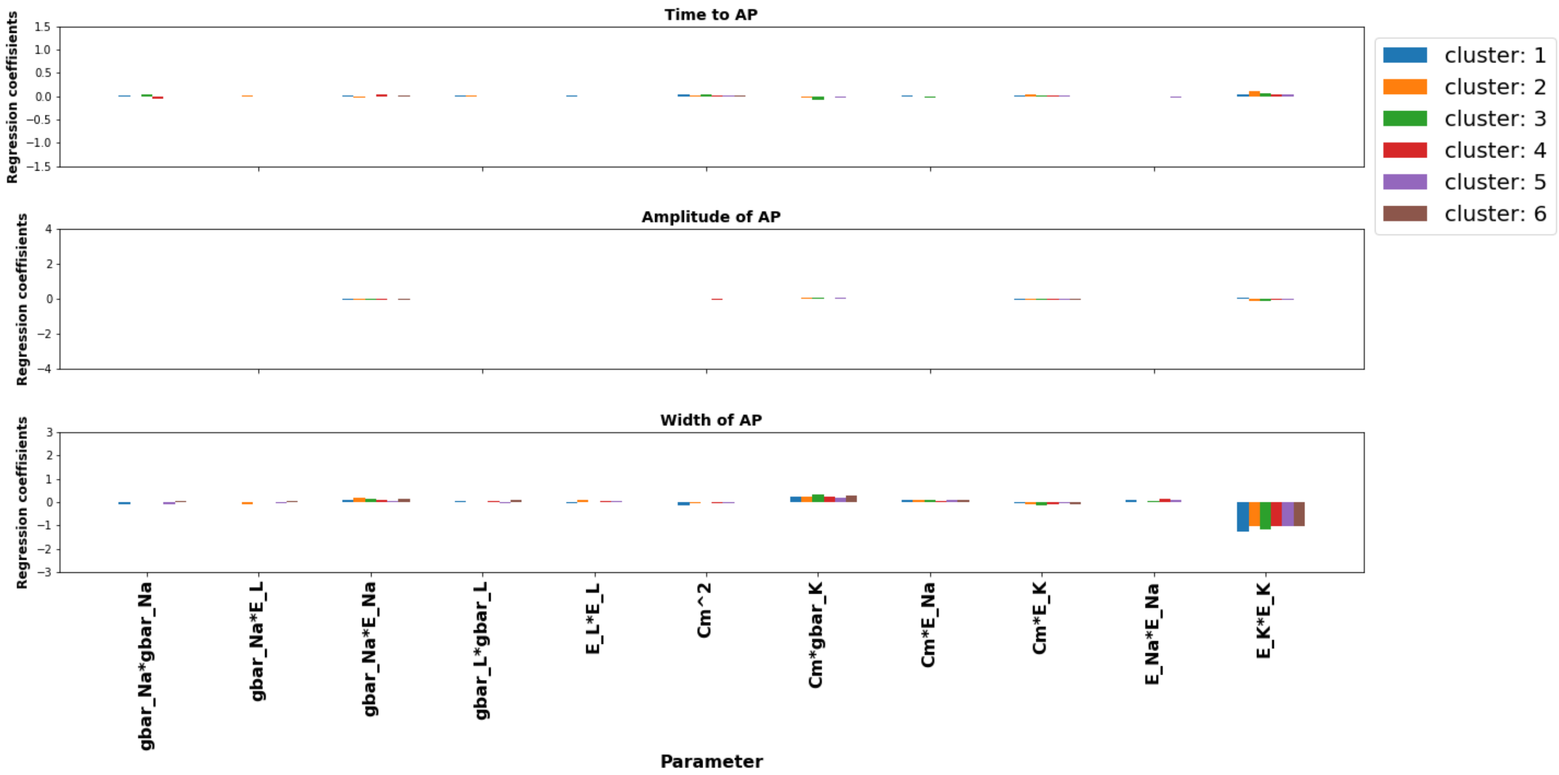


Figure 4.11: Local regression coefficients for the cross- and second order terms of the aggregated phenotypes extracted from the Hodgkin-Huxley model. The regression coefficients show the sensitivity of the interaction- and cross terms of the input parameters. **The top figure** shows the regression coefficients for the time taken for the membrane potential to reach the first AP. **The middle figure** shows the regression coefficients for the amplitude reached by the first AP. **The bottom figure** shows the regression coefficients for the width of the first AP. The regression coefficients are coloured according to their respective clusters.

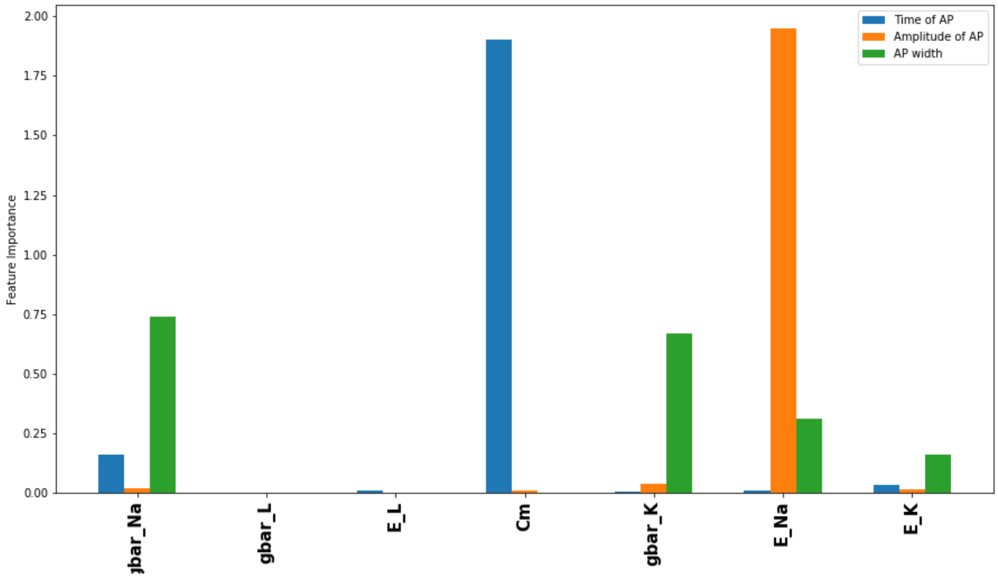


Figure 4.12: Feature importance for the input parameters of the aggregated phenotypes extracted from the first AP in the Hodgkin-Huxley model. The feature importance is calculated as the mean of five permutations for each input variable.

4.2 Metamodelling of the Pinsky-Rinzel model

4.2.1 Data Generation

159434 simulations were carried out with the Pinsky-Rinzel model. The membrane potentials of the 100000 randomly sampled simulations used in classical and inverse metamodelling of the Pinsky-Rinzel model are plotted in Figure 4.13A. The mean and standard deviation of the 100000 randomly sampled simulations are shown in Figure 4.13B. The parameter combinations sampled created a wide range of differently behaving membrane potentials, with the majority of the variation starting after $5ms$ of simulation.

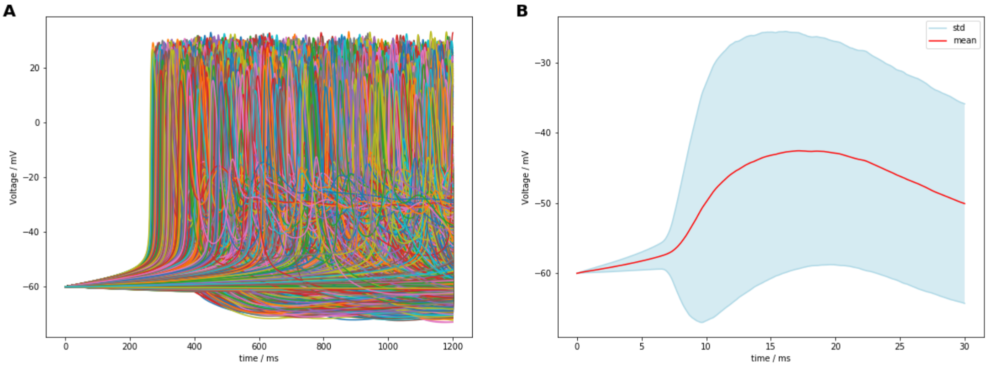


Figure 4.13: Plot of membrane potentials generated with the Pinsky-Rinzel model. A) Plots the 100 000 sampled membrane potentials simulated with the Pinsky-Rinzel model. **B)** shows the mean and standard deviation of the 2817 membrane potentials simulated by the Pinsky-Rinzel model. red: Mean, blue: Standard deviation.

4.2.2 Classical metamodelling of the Pinsky-Rinzel model

Classical metamodelling with HCPLSR and FFN was carried out in order to analyze the complex input-output relationships of the Pinsky-Rinzel model. The optimizing search of cluster basis, number of clusters and additional terms were done in the same manner as the classical metamodelling of the Hodgkin-Huxley model. Also here using the X-scores as the clustering basis for the FCM clustering with additional second-order terms and interactions achieved the best results. The test set prediction accuracy converges after more than three clusters are used in the FCM. Therefore three clusters were chosen for further exploration of the Pinsky-Rinzel model. The test set prediction accuracies achieved are plotted in Figure 4.14.

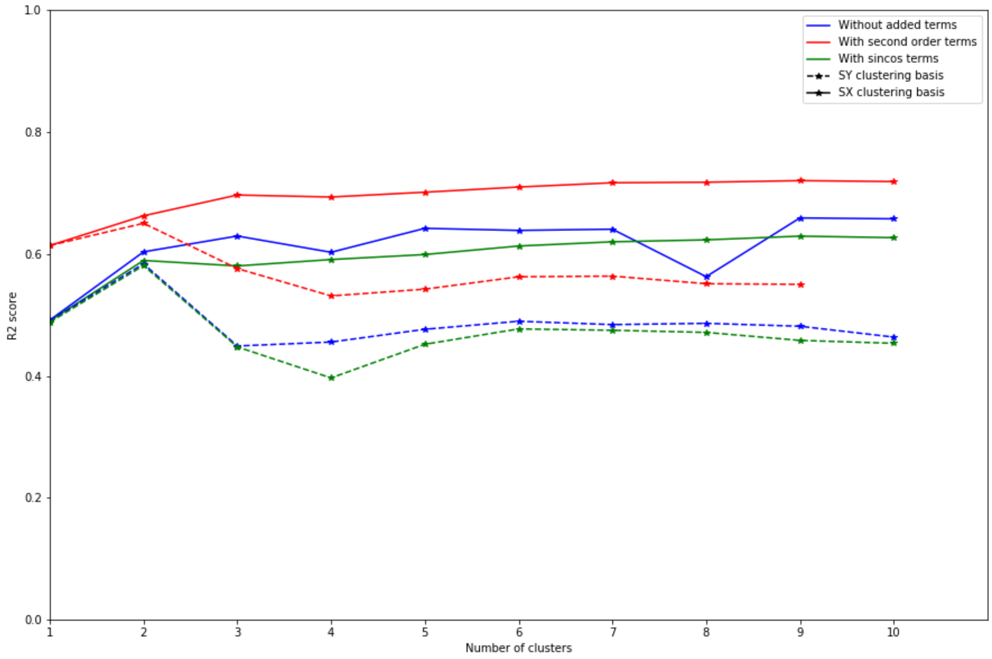


Figure 4.14: Optimizing the number of clusters and cluster matrix using HCPLSR in classical metamodelling of the Pinsky-Rinzel model. The average prediction accuracy (R^2) of the test data, given the number of clusters, clustering matrix and added terms. The average prediction accuracy achieved by the global PLSR model is noted as one cluster. Blue: Only input parameters. Orange: Input parameters with additional sinus and cosinus terms. Green: Input parameters with interaction- and second-order terms. Red: Input parameters with interaction- and second-order terms, and additional sinus and cosinus terms. The dashed line and the solid line represent Y- and X-scores, respectively, used as cluster matrix in the FCM clustering.

The first three PCs in the global PLSR model explained 0.601% of the total cross-validated variance present in the output matrix and were used as a clustering basis. The X-scores for the three PCs are plotted in Figure 4.15A. The mean and standard deviations of the membrane potentials belonging to cluster 1-3 are plotted in Figure 4.15B-D, respectively. The main effect regression coefficients for the global PLSR and the three local PLSR models are plotted in Figure 4.16. The 15 most significant interaction- and second order term regression coefficients for the global PLSR and the three local PLSR models are plotted in Figure 4.17.

The training plots used for hyperparameter tuning of the stated FFN model used in the classical metamodelling of the Pinsky-Rinzel model are given in Appendix B figureB.4. The final FFN model was trained for 2000 epochs, and test set prediction accuracy of 0.961.

The feature importance ranking of the input variables of the Pinsky-Rinzel model was calculated using the permuted feature importance procedure and is plotted in Figure 4.18. For an exploration of the importances of the input parameters surrounding the first occurring AP in each membrane potential, feature importances using a centralized permuted feature importance procedure (Described in section 3.4) were also calculated, where the time of the first occurring AP for each sample is aligned. These feature importances are plotted in Figure 4.19.f

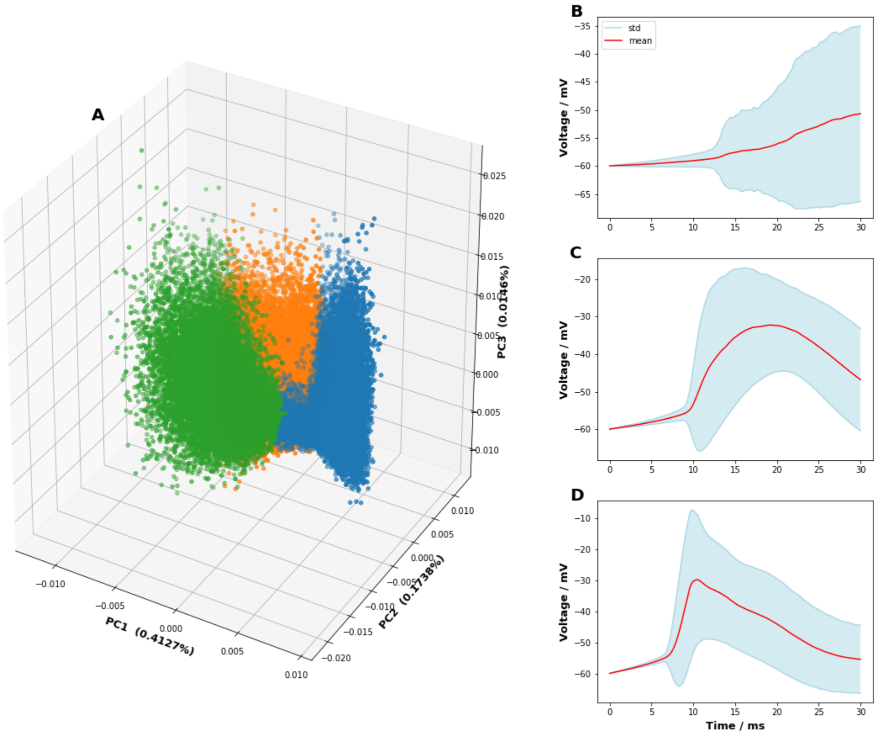


Figure 4.15: Clustering results from HCPLSR metamodelling of the Pinsky-Rinzel model using three clusters. A) Scatter plot of the X-scores from the global metamodel. The samples are coloured according to cluster membership. Cluster1=blue, cluster2=orange, cluster3=green. **B-D)** Mean (red) and standard deviation (blue) of the simulated membrane potentials for cluster 1-3, respectively.

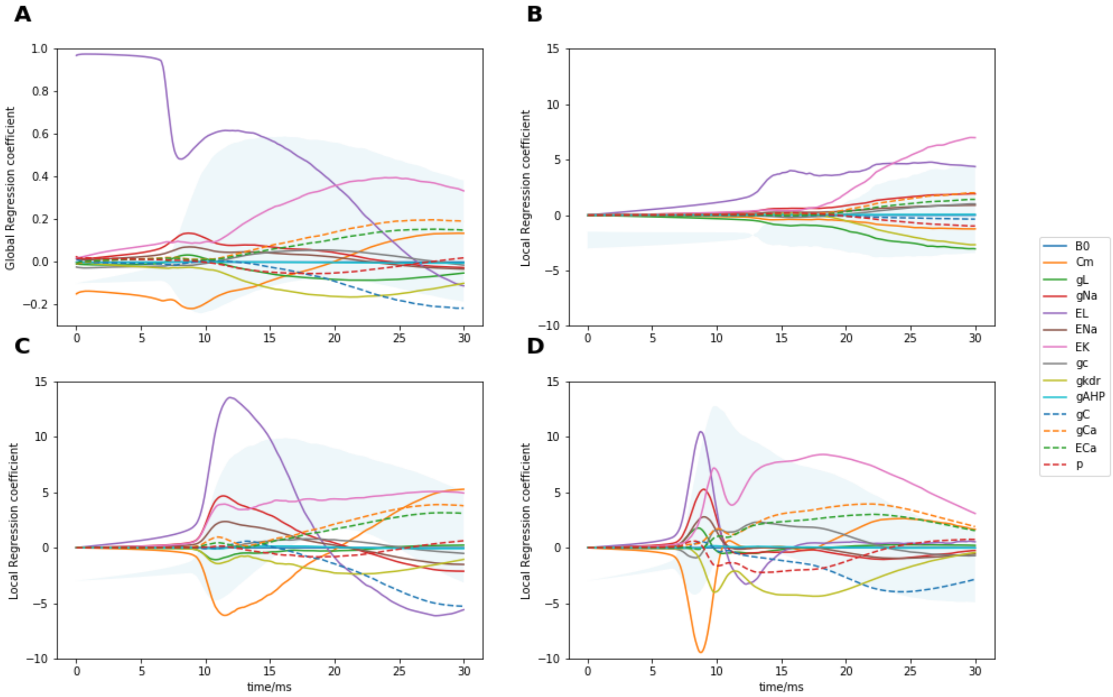


Figure 4.16: Global and local regression coefficients for the input parameters of the Pinsky-Rinzel model. The regression coefficients show the sensitivity of the input parameter over time. **A)** the global regression coefficients. **B-D)** show the local regression coefficients for cluster 1-3, respectively. The standard deviation of the membrane potentials belonging to each cluster is plotted off scale in the background for comparison.

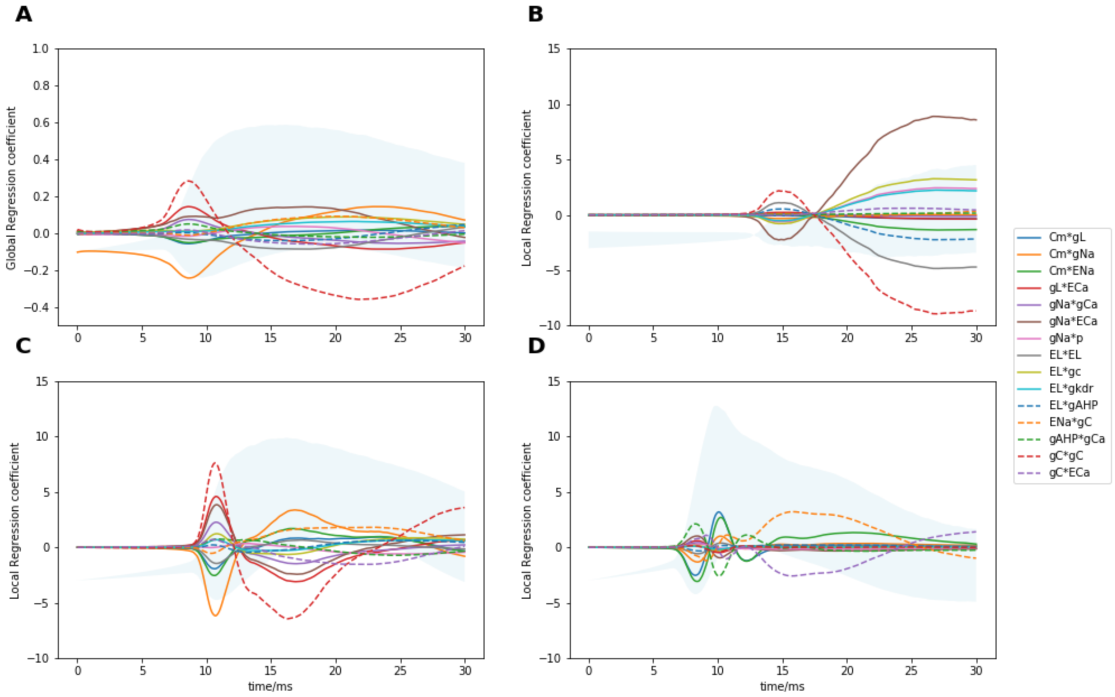


Figure 4.17: Global and local regression coefficients for the cross- and second order terms of the input parameters of the Pinsky-Rinzel model. The regression coefficients show the sensitivity of the 15 most significant cross- and second order terms parameter over time. **A)** the global regression coefficients. **B-D)** show the local regression coefficients for cluster 1-3, respectively. The standard deviation of the membrane potentials belonging to each cluster is plotted off scale in the background for comparison.

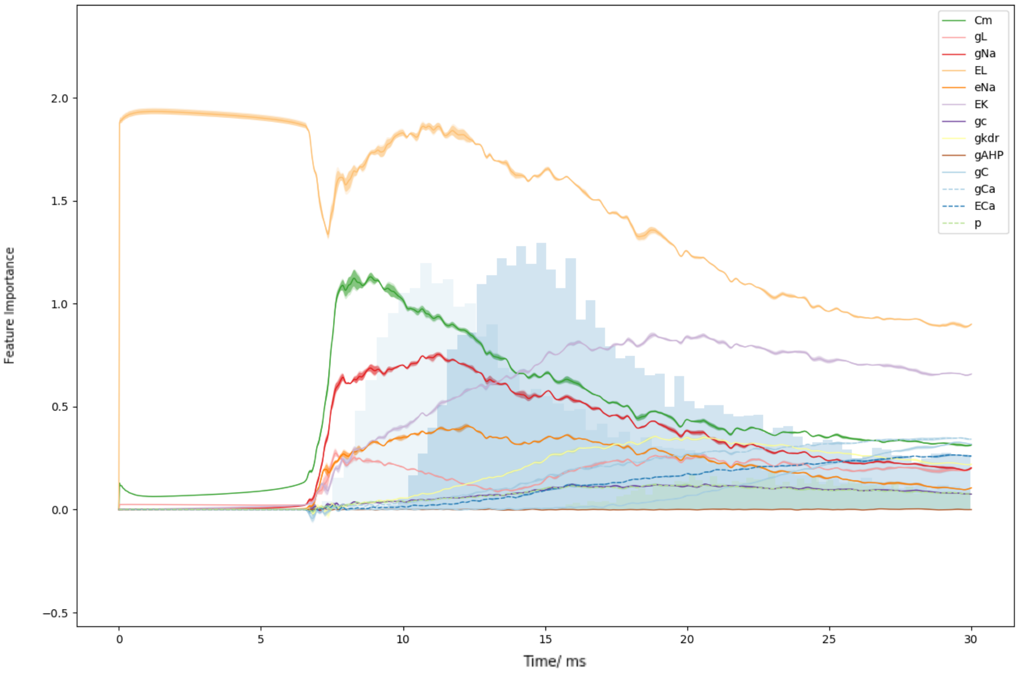


Figure 4.18: Feature importance for the input parameters of the Pinsky-Rinzel model. The feature importance is calculated with four permutations of each input variable. The histograms in the background show the distribution for the time of the first, second and third AP for each AP.

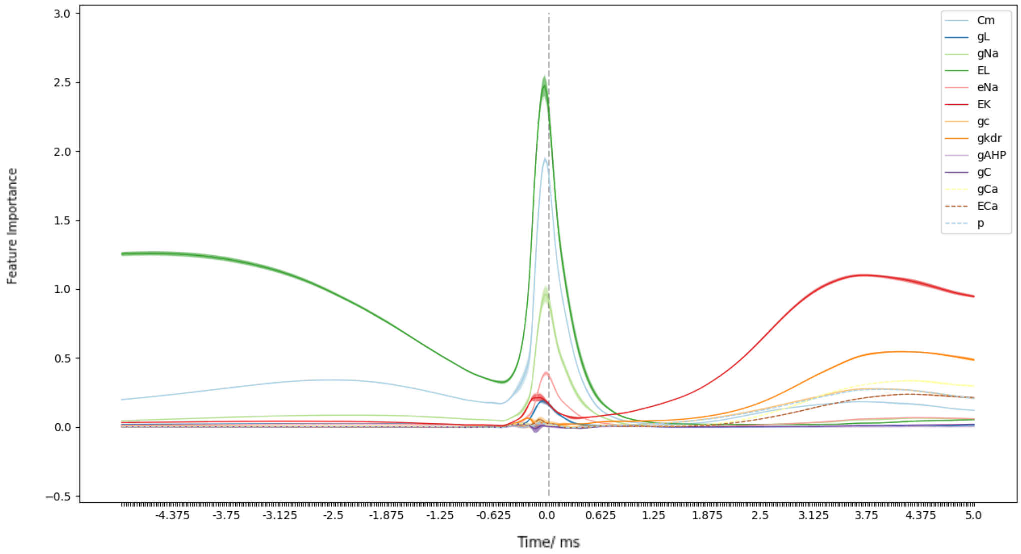


Figure 4.19: Feature importance for the input parameters of the Pinsky-Rinzel model centred around the first AP. The feature importances are calculated using centred feature importance and are calculated using four permutations of each input variable. The APs are centred at 0 ms. The x-axis represents the time relative to the AP peak.

4.2.3 Inverse metamodelling of the Pinsky-Rinzel model

The inverse metamodelling of the Pinsky-Rinzel model was carried out with HCPLSR and CNN, in order to predict the output-input relationships of the model. The test set prediction results used X-scores and Y-scores, respectively, as clustering basis can be found in Figure 4.20. The FCM clustering algorithm did not converge in less than 1000 iterations when the global Y-scores were used as a clustering basis, so the results are therefore not represented in the figure. The best average prediction accuracy was achieved using X-scores as a clustering basis for the FCM clustering with ten clusters. These are listed in Table 5.

The training plots used for hyperparameter tuning of the CNN model stated for inverse metamodelling of the Pinsky-Rinzel model are shown in Appendix B Figure B.6. The final CNN model was trained for 1750 epochs. Table 5 shows the prediction accuracies of the test set data.

4.2.4 Classical metamodelling of the aggregated phenotypes

The classical metamodelling of aggregated phenotypes extracted from the Pinsky-Rinzel model was carried out using FFN and HCPLSR. 81250 of the parameter combinations from the data set generated at least one AP and were included in the metamodelling.

Both Y- and X-scores were explored as clustering matrices for the HCPLSR model, and adding additional cross-and second order terms was tested in order to find the optimal model. The number of clusters used in the FCM clustering was varied from two to ten clusters. The test set prediction accuracies achieved using the HCPLSR metamodelling are plotted in Figure 4.21. The best performing HCPLSR metamodel was achieved using X-scores as a clustering basis for the FCM clustering with ten clusters, and additional cross- and second order terms added to the main effects of the input parameters. However, the local modelling did not manage to outperform the global PLSR model with the same added terms. This indicates that the nonlinearities present in the input-output relationships are

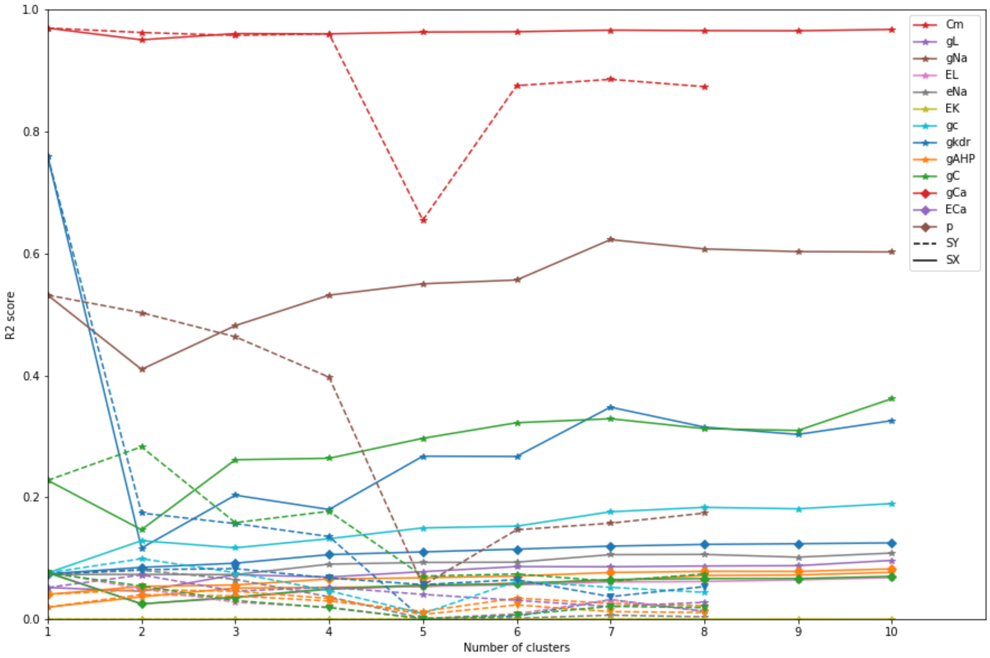


Figure 4.20: Optimizing the number of clusters in the inverse metamodelling of the Pinsky-Rinzel model using HCPLSR. The test set prediction accuracy (R^2) of the input parameters given the number of clusters. The prediction accuracy achieved by the global PLSR model is noted as one cluster. The colour represents the parameter estimated. The dotted line represents the prediction result achieved using the global Y-scores as clustering matrix. For the global X-scores as clustering basis, a solid line is used.

Table 5: Prediction accuracies (R^2) for inverse metamodelling of the Pinsky-Rinzel model using HCPLSR and CNN.

Parameter	CNN	HCPLSR
C_m	0.9130	0.3257
$\overline{g}_L$	0.6357	0.0768
$\overline{g}_{Na}$	0.7907	0.3617
E_L	0.9183	0.9679
E_{Na}	0.7410	0.0964
E_K	0.8464	0.6026
$\overline{g}_c$	0.4730	0.0681
$\overline{g}_{kdr}$	0.7331	0.1084
$\overline{g}_{AHP}$	-0.0018	0.0001
$\overline{g}_C$	0.5107	0.1896
$\overline{g}_{Ca}$	0.6426	0.1251
E_{Ca}	0.5146	0.0824
p	0.4760	0.0702

mainly non-abrupt, and hence a simple linear combination with cross- and second order terms is sufficient in this case. The individual prediction accuracies for the aggregated phenotypes achieved by the HCPLSR and global PLSR model are listed in Table 6. The global PLSR model is used for further exploration of the parameter importances. The main effect regression coefficients for the global PLSR model are plotted in Figure 4.22. The most significant cross- and second order effects are plotted in Figure 4.23.

The training plots used for hyperparameter tuning of the FFN model stated for the classical metamodelling of the aggregated phenotypes are plotted in Appendix B Figure B.5. The final model was trained for 600 epochs. Table 6 shows the prediction accuracies

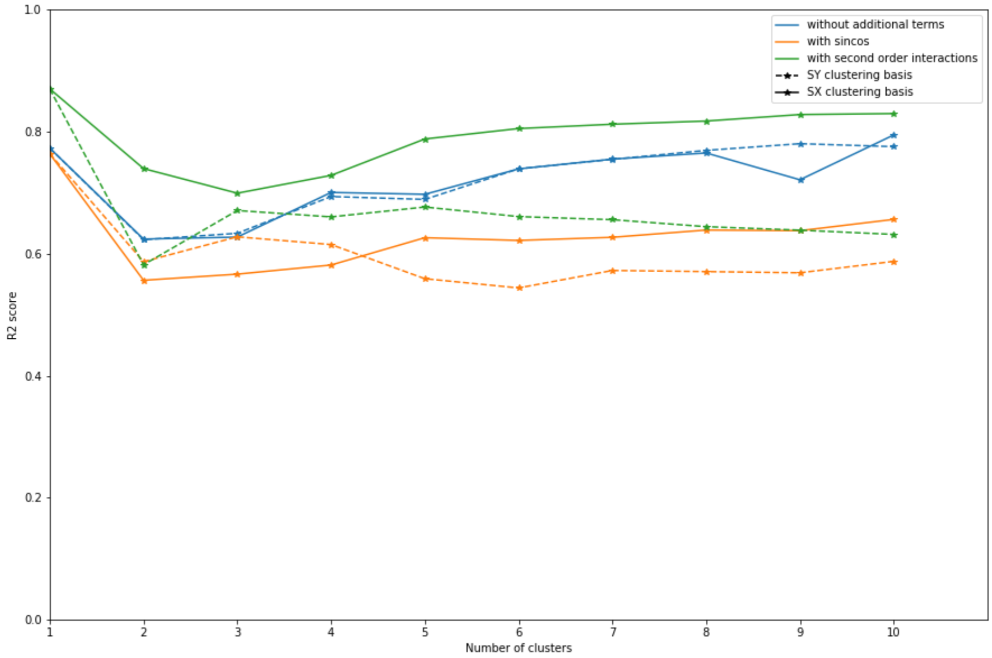


Figure 4.21: Optimizing the number of clusters and cluster matrix in metamodelling of the aggregated phenotypes of the Pinsky-Rinzel model using HCPLSR. The prediction accuracy (R^2) of the aggregated phenotypes in the test set, given the number of clusters and clustering matrix. The prediction accuracy achieved by the global PLSR model is noted as one cluster. The dashed line and the solid line represent Y- and X-scores, respectively, used as cluster matrix in the FCM clustering.

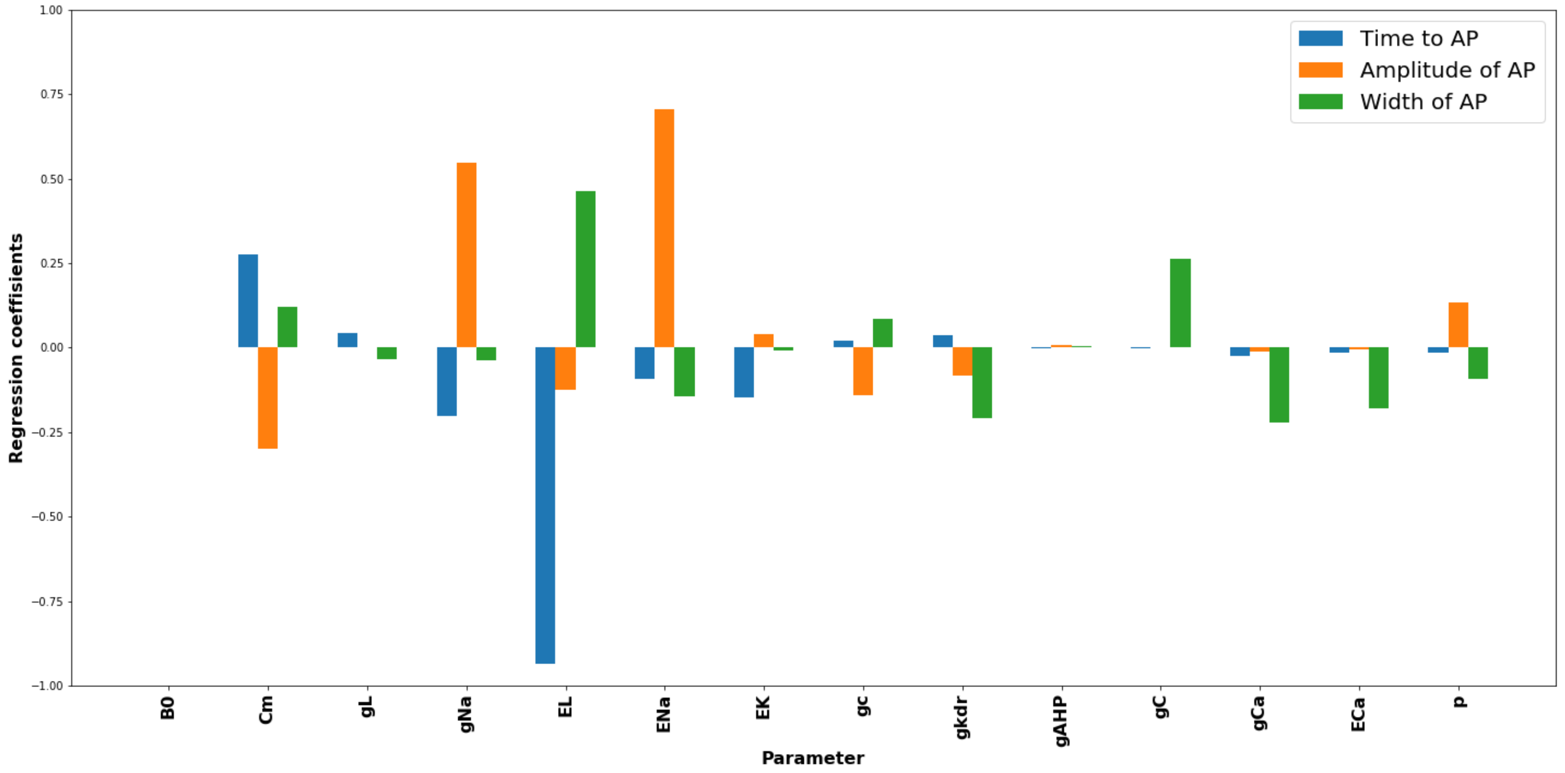


Figure 4.22: Global regression coefficients for the main effects of the aggregated phenotypes extracted from the Pinsky-Rinzel model. The regression coefficients are coloured by aggregated feature: Blue: *time* to first occurring AP, orange: *amplitude* of first occurring AP, green: *width* of first occurring AP.

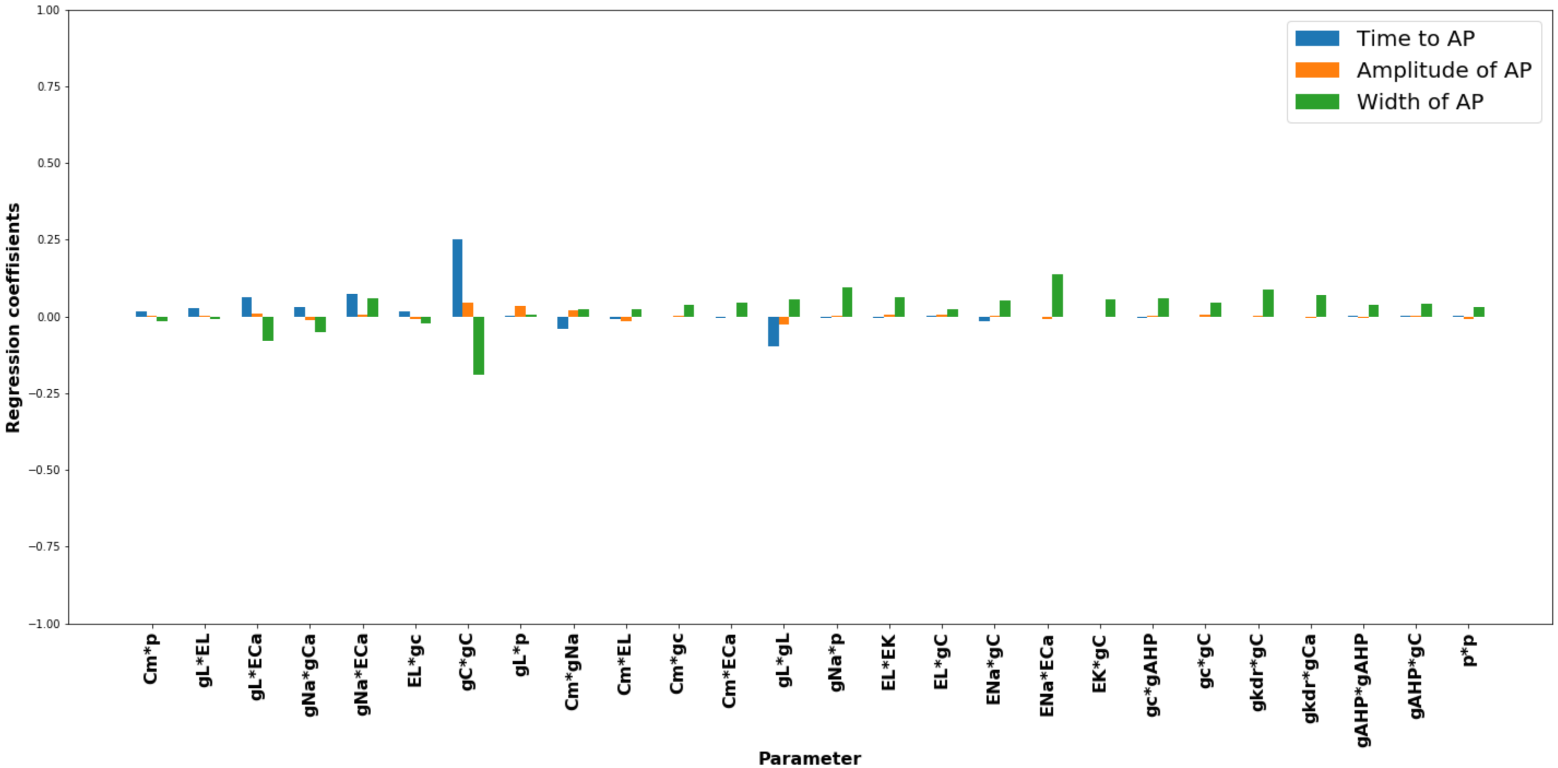


Figure 4.23: Global regression coefficients for the cross- and second order terms of aggregated phenotypes extracted from the Pinsky-Rinzel model. The regression coefficients are coloured by aggregated feature: Blue: *time* to first occurring AP, orange: *amplitude* of first occurring AP, green: *width* of first occurring AP.

of the test set data, achieved when the stated FFN model was trained on the full train set.

The feature importance of the input variables was calculated as the mean of five permutations for each variable using the permuted feature importance procedure and is plotted in Figure 4.24.

Table 6: Prediction results (R^2) for metamodelling of the aggregated phenotypes extracted from the Pinsky-Rinzel model using FFN, HCPLSR and PLSR.

	FFN	HCPLSR	Global PLSR
time to peak	0.9981	0.9927	0.9622
width of first AP	0.9982	0.8952	0.9919
amplitude of first AP	0.9251	0.6010	0.6571

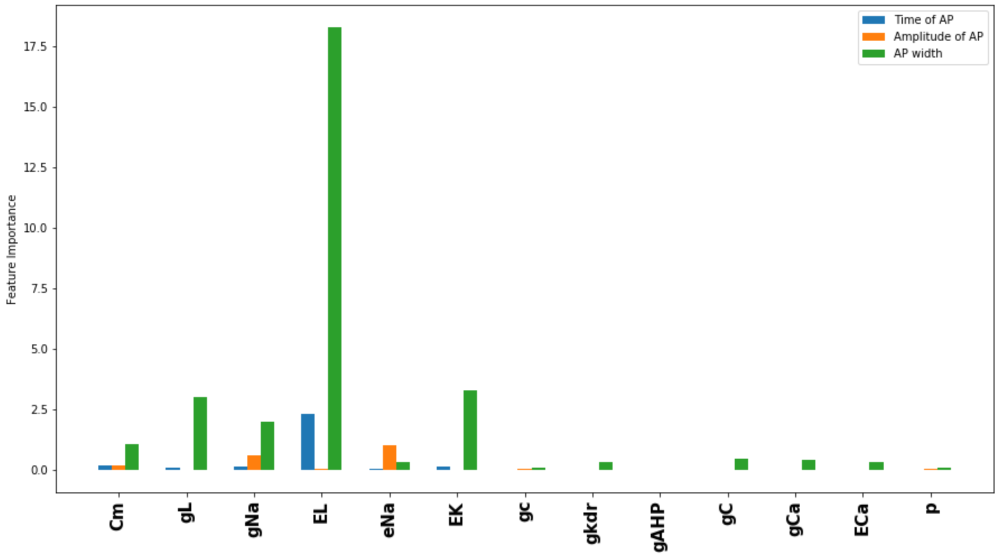


Figure 4.24: Feature importance for the input parameters of the aggregated phenotypes extracted from the first AP in the Pinsky-Rinzel model. The feature importance is calculated as the mean of five permutations for each input variable.

5 Discussion

5.1 Hodgkin-Huxley model

5.1.1 Classical metamodelling

The high prediction accuracy achieved with the global PLSR model indicates that the Hodgkin-Huxley model mainly shows soft nonlinear behaviour. However, the increased prediction ability achieved by using multiple clusters in the HCPLSR indicates that there are some more abrupt nonlinearities to consider, in order to better emulate the model outputs. While the HCPLSR model with added cross- and second order terms using ten clusters was able to achieve an average prediction score of 0.944, the simple FFN model consisting of two hidden layers was able to achieve a higher average prediction score of 0.993. This indicates that the FFN is more efficient at emulating the Hodgkin-Huxley model.

The global regression coefficients (Figure 4.4A), the local regression coefficients (Figure 4.4B-D) and the permuted feature importance (Figure 4.6) somewhat support what is already known about the model mechanisms. They all highlight the importance of the potassium reversal potential in the repolarization period following the AP. By comparing the local regression coefficients with the global regression coefficients, it is clear that the local regression coefficients are better at illustrating the decline in the importance of potassium in the period in between the APs. The sodium reversal potential is known to be important in the building of the first AP, a behaviour the global and two of the local PLSR models fail to show. This is, however, highlighted by the permuted feature importance. The low regression coefficients of the sodium reversal potential might be due to low variation in the membrane potential during the first 0.05 ms. This might also explain why the membrane capacitance is estimated to have greater importance by the permuted feature importance than by the local regression coefficients.

Even though adding cross- and second order terms increased the prediction accuracy, the regression coefficients (Figure 4.5B-D) show that none of these terms have a substantial individual contribution to the membrane potential. It might be that the combination of many small cross- and second order terms adds an explanation to the behaviour surrounding the AP. Another possibility is that the added terms lead to a better clustering, as some of the global cross- and second order term regression coefficients (Figure 4.5A) seem to be important.

5.1.2 Inverse metamodelling

The inverse metamodelling using HCPLSR had difficulties predicting the conductance and reversal potential of the leak ions. When examining the regression coefficients for the classical metamodelling (Figure 4.4), the results indicate that none of these two input parameters had a large influence on the membrane potential, something that may explain the low prediction ability. The stated CNN model manages to achieve a greater prediction accuracy for the conductance of the leak ions, indicating that there are spatial dependencies in the data lost to the HCPLSR model. A more surprising result is reduced prediction ability of the membrane capacitance achieved when using local metamodels within clusters, compared to the global metamodel (Figure 4.8). The most likely explanation is that it has something to do with the PLSR decomposition. The PLSR decomposition, is maximizing the covariance between the input matrix and the output matrix. It might be that the X-score matrix, which was used as a clustering basis, was not clustered in a way that favored the variance in the membrane capacitance.

The prediction accuracies of the reversal potentials of the sodium and potassium achieved by using the CNN model were worse than the one achieved using HCPLSR (Table 3), although it is likely possible to further optimize the CNN model, increasing its prediction ability.

5.1.3 Classical metamodelling of the aggregated phenotypes

The metamodelling of the aggregated phenotypes using HCPLSR achieved a greater prediction accuracy for the aggregated phenotype *Time to peak* than the FFN model, although both models achieved a relatively high prediction accuracy for all phenotypes (Table 4).

When looking at the feature importance for amplitude, both the permuted feature importance (Figure 4.12) and the local regression coefficients (Figure 4.10) estimated the membrane capacitance to be the most important variable for *Time to peak*. The reversal potential of sodium is estimated as the most important predictor for the *amplitude* of the first AP, which makes sense given that the importance for the reversal potential of sodium also kicks in at the same time points as the AP. The *width* of the AP seems to be driven by multiple parameters; the reversal potential and conductance of both sodium and potassium. As seen in the classical metamodelling, the parameters concerning the potassium start influencing at the repolarization stage of the AP.

5.2 Pinsky-Rinzel model

5.2.1 Classical metamodelling

The test set accuracy score obtained with the global PLSR indicates that the Pinsky-Rinzel model show more complex nonlinear behaviour than the Hodgkin-Huxley model. This is an expected result as the Pinsky-Rinzel model describes a two-compartment model, with membrane potential changes in both compartments. Moreover, only the somatic membrane potential is measured. Also here the best HCPLSR model, achieving an average test set prediction accuracy of 0.72, was outperformed by the FFN achieving an average test set prediction accuracy of 0.96.

By using FCM clustering, the HCPLSR was able to differentiate between different behaviours of the Pinsky-Rinzel model. Figure 4.15B-D shows how the clustering have separated the parameter space into subspaces that causes earlier (**D**), mid-range (**C**) and late (**B**) APs. This allows for an increased ability to connect the importance of the input parameters to the different behaviours of the model.

It is known that the initial AP is caused by a somatic sodium current, followed by a potassium current repolarizing the somatic membrane. These mechanics are confirmed by the local regression coefficients of the input parameters in Figure 4.16C-D. These indicate that the influence of the somatic sodium current, controlled by the reversal potential E_{Na} and the somatic sodium conductance g_{Na}, are increasing with the membrane potential. The influence of sodium is followed by a rise of importance for the somatic potassium current which is controlled by the reversal potential E_K and the somatic potassium conductance g_{kdr}. In the case were no early APs are generated (Figure 4.16B), the influence of the somatic sodium current is absent. Following the initial somatic AP, it can be seen that the influence of the coupling current (p and g_c) is increasing as the somatic membrane potential declines. The calcium current (E_{Ca} and g_{Ca}) is needed to explain the behaviour of the somatic membrane potential during the final stages. It is known that a difference in the potential between the soma membrane and the dendritic potential generates a coupling current flowing from the soma into the dendrite, that causes a repolarization of the soma and an increased membrane potential in the dendrite. The depolarization of the dendrite can in some cases activate a calcium current, generating a dendritic spike, which redirects the coupling current, allowing for new APs to be generated in the soma. The leak ion reversal potential has a greater influence on the membrane potential in the Pinsky-Rinzel model than the Hodgkin-Huxley model. It turns out that the leak ion reversal potential is the most important input variable needed in order to explain the time taken until the first AP(Figure 4.24). The influence of the membrane capacitance kicks in right before the majority of the APs in cluster three and four, but it is more or less absent in cluster two, where the majority do not generate APs. The membrane capacitance is expected to affect the speed of the potential change, something that might explain this result. The coupling conductance between the soma and dendrite seems to influence whether multiple APs is achieved, something that is more represented in cluster 4 than in cluster 3. When inspecting the regression coefficients of the cross- and second order terms (Figure 4.17), it is obvious that multiple interactions are important in order to explain the full behaviour of the model. The feature importances calculated with the permuted feature importance (Figure 4.18) are not able to give a detailed overview of the model behaviours, as it is based on the entire data set. The more generalized overview achieved by the permuted feature importances is similar to the global regression coefficients (Figure 4.17). It can be interpreted from the feature importances that the reversal potential for the leak ions is the most important parameter explaining the behaviour leading up to the initial AP. These results also indicate that the sodium current and membrane capacitance are the main parameters influencing the generation of the initial AP, followed by an increased importance of the potassium current.

The centralized permuted feature importance (Figure 4.19) is developed to get a more in-depth insight into the parameter importances surrounding the first action potential. It shows as the HCPLSR model suggested, that the reversal potential of the leak ions and the membrane capacitance are the most important parameters leading up to the first generated AP, followed by the reversal potential and conductance of potassium as the leading causes for the behaviour following the first AP. It also shows how the coupling current and calcium current are activated after the initial AP.

5.2.2 Inverse metamodelling

Neither the HCPLSR model or the CNN model used in inverse metamodelling of the Pinsky-Rinzel model managed to predict the dendritic ligand-gated potassium ion channel g_{AHP} successfully. Inspection of the regression coefficients (Figure 4.16) and the permuted feature importances (4.18) shows that the g_{AHP}, which is calcium dependent, was not found important for explaining the somatic membrane potential during the first $30ms$.

This indicates that there is not enough information in the data to predict this parameter. The fact that this parameter is calcium dependent means that it needs a sufficient calcium concentration inside the dendrite to activate the potassium current. As the calcium current is activated soon after the initial AP, there might not be sufficient time to build up the calcium concentration enough to activate the ligand gated potassium current. The stated CNN model achieved a greater prediction accuracy than the HCPLSR model, with the exception of the reversal potential for the leak ions. This indicates that there are spatial dependencies and/or abrupt non-linear output-input relationships. Figure 4.20 shows that the global PLSR model did, for some of the input parameters, achieve a higher prediction accuracy than the HCPLSR model. Most noticeable is the somatic potassium conductance g_{kdr}, whose prediction ability is nearly halved when using the HCPLSR metamodel with seven clusters compared to the global PLSR model. It should also be noted that this type of dynamic models generally show a large degree of sloppiness, meaning that a large number of different parameter value combinations can generate vary similar outputs. This makes inverse metamodelling more challenging than the classical metamodelling.

5.2.3 Classical metamodelling of the aggregated phenotypes

Although the global PLSR model achieved a higher average prediction accuracy than the best HCPLSR model for the aggregated phenotypes extracted from the Pinsky-Rinzel model, the HCPLSR model had a higher success in prediction of the *time to peak*-phenotype (Table 6). However, none of the PLSR-based models were able to achieve a good prediction accuracy for the *amplitude* of the first occurring AP. The global regression coefficients (Figure 4.22 and 4.23) and the permuted feature importances (Figure 4.24) show that the phenotypes are dependent upon a more complex system than the phenotypes extracted from the Hodgkin-Huxley model. This is expected, since the Pinsky-Rinzel model is a much more complex model, containing two compartments. Both the permuted feature importances and the global regression coefficients recognize the leak ion reversal potential as the most important parameter for predicting the time until the first AP, followed by the membrane capacitance, the reversal potential of potassium and the somatic conductance of sodium. The two feature importance measurements differ in the importance of the coupling conductance, as the global regression coefficients show a significant importance of the second order term $g_C g_C$. This results is not surprising, given that a coupling current would flow from the soma into the dendrite when the soma membrane potential exceeds the dendritic membrane potential. The amplitude of the initial AP is, as expected, mainly explained by the sodium current and the membrane capacitance. The width of the initial AP is affected by multiple parameters, the most important being the reversal potential of the leak ions. As seen in the classical metamodelling of the Pinsky-Rinzel model, the calcium current and coupling current, which are initiated after the first AP, contribute to the shape of the width of the first AP.

5.3 Comparing HCPLSR and Deep learning

When comparing the two metamodelling methods, it is obvious that the emulation capacities of using deep learning (both FFN and CNN) far exceed the HCPLSR. Especially when emulating more complex models such as the Pinsky-Rinzel model, or when spatial relationships are needed in inverse metamodelling. It might, of course, be possible to increase the number of clusters in the HCPLSR model, in order to obtain a greater prediction ability. However, an increased number of clusters leads to a longer computational time for the FCM clustering algorithm, which is already severe when a large number of samples is being clustered. Also, the number of samples needed in order to fully explain the behaviour of the models grows exponentially with the number of input parameters of the mathematical model. Another consequence of increasing the number of clusters is an reduced ability to interpret the model behaviours. It should be noted that the deep learning models used in this thesis could be further optimized. Exploration of different architectures, activation functions, optimizers and batch-sizes could further enhance the prediction ability and the computational cost associated with training of the network.

The strength of HCPLSR metamodelling lies in its ability to improve the analytic insight into the model being emulated, without any need for prior knowledge of the model behaviour. As has been shown by metamodelling of the Pinsky-Rinzel model, the HCPLSR model was able to differentiate different behaviours of the somatic membrane potential, and allows for interpretation that is not possible for a model generalizing the entire data set. In order to achieve similar insight into the behaviour surrounding the initial AP, using a FFN model trained on the entire parameter space, we had to take special measures that required prior knowledge of what we wanted to explain. An inconvenience of using the HCPLSR metamodelling is that the PLSR model needs variance in the output data in order to train. As seen in the classical metamodelling of the Hodgkin-Huxley model, the regression coefficients indicated a low contribution of most of the parameters leading up to the first AP, most significant was the lack of contribution from the membrane capacitance, when compared with the feature importances calculated with the FFN model. When comparing the time consumption needed for the analysis, the HCPLSR has an advantage. The calculation of the feature importances using the permuted feature importance algorithm, requires the permutation and analysis of one parameter at a time. This leads to an increasingly computationally demanding operation, as the number of input parameters being analyzed increased, where the PLSR is predicting the importance of all input parameters simultaneously.

The insight into the behaviour gained by using HCPLSR metamodelling might be restricted by its ability to emulate. If the HCPLSR models emulating capabilities are weakened due to increased complexity of the mathematical model being emulated, it raises the question about what amount of information is explained by the HCPLSR model. Or rather, what information is not explained.

5.4 The coefficient of determination: R^2

The coefficient of determination (R^2), which is used as a measure of prediction accuracy in this thesis, can be calculated in multiple ways.

One way of calculating the coefficient of determination is by squaring the correlation (r) between the predicted output ($\hat{y}$) and the true output (y):

$$R^2 = r^2_{\hat{y},y} \tag{7}$$

It follows from this equation that the R^2 is always in the interval [0,1], as the correlation is always in the interval [-1,1]. This definition of the coefficient of determination is used for calculating prediction accuracy of the HCPLSR metamodelling in this thesis.

For the deep learning metamodelling, the coefficient of determination is calculated using the more general term:

$$R^2 = 1 - \text{SSE/SST} = 1 - \frac{\Sigma(y - \hat{y})^2}{\Sigma(y - \bar{y})^2} \tag{8}$$

Where $\hat{y}$ and y are the predicted and true output value, respectively. $\bar{y}$ is the mean of the true output value. This calculation of the prediction accuracy allows the R^2 value to be negative (Can be seen Table 5).

The different implementations for calculating the prediction accuracy raises the question whether the prediction accuracies calculated with the two metamodelling techniques can be compared. The prediction results achieved using deep learning in inverse metamodelling of the Hodgkin Huxley model are calculated with both equation 7 and 8, and is listed in Table 7. In this case, the correlation based calculation of the coefficient of determination (equation 7) gives a higher score for the prediction accuracy than the more general term (equation 8). If this is the case for all prediction accuracies that have been calculated, then the prediction abilities of the deep learning metamodelling might have been underestimated.

Table 7: Comparison of two methods for calculating the coefficients of determination(R^2). Both R^2-values is calculated for the inverse metamodelling of the Hodgkin-Huxley model using deep learning. Equation 7: Correlation based, used in the HCPLSR implementation in this thesis. 8: General term, used in the deep learning implementation in this thesis.

	Equation 7	Equation 8
gbar_Na	0.9073	0.8828
gbar_L	0.8452	0.7838
E_L	0.0259	0.0120
C_m	0.9499	0.9163
gbar_K	0.8951	0.8758
E_Na	0.9415	0.9083
E_K	0.9530	0.9151

6 Conclusion

The first aim of this project has been to investigate how accurately HCPLSR and deep learning can emulate the simple Hodgkin-Huxley model and the more complex Pinsky-Rinzel model. The results in this thesis indicate that the use of deep learning metamodels is more effective than the HCPLSR metamodel at emulating non-linear mathematical models, especially for non-linear models with a high level of complexity.

The second aim has been to investigate what insight these two metamodelling techniques can give into the behaviour of the two single neuron activity models. The findings in this work show that the HCPLSR model is an efficient tool for analysis of non-linear mathematical models. By dividing the parameter space into subspaces, the HCPLSR allows for the investigation of the input parameters influences on the model outputs without requiring prior knowledge of the model outputs. In order to gain similar insight using permuted feature importances from a FFN model, generalizing the entire parameter space, the APs had to be aligned which require prior knowledge of the model output.

6.1 Further works

As the FCM clustering method used in this implementation of the HCPLSR model is computationally demanding for a large number of samples, exploration of an alternative, less demanding clustering method might make it feasible to increase the number of clusters. Thus, allowing for metamodelling of more complex models.

The insights possible to achieve using deep learning is limited by it generalizing the entire parameter space. This could thus be prevented by using clustering methods in combination with deep learning, for an increased interpretation ability of the model behaviour. Deep embedding (Xie et al., 2016) might be tested for subspace analysis purposes.

References

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning - A Textbook*. Springer.

Bezdek, J. C., Ehrlich, R., and Full, W. (1984). Fcm: The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 10(2):191 – 203.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Fisher, A., Rudin, C., and Dominici, F. (2018). Model class reliance: Variable importance measures for any machine learning model class, from the ”rashomon” perspective.

Fonseca, D., Navaresse, D., and Moynihan, G. (2003). Simulation metamodeling through artificial neural networks. *Engineering Applications of Artificial Intelligence*, 16(3):177 – 183.

Galizia, C. G. and Lledo, P.-M. (2013). Neurosciences - from molecule to behavior: A university textbook. *Neuroforum*, 19:27–27.

Geladi, P. and Kowalski, B. R. (1986). Partial least-squares regression: a tutorial. *Analytica Chimica Acta*, 185:1 – 17.

Gorissen, D., Crombecq, K., Couckuyt, I., and Dhaene, T. (2009). *Automatic Approximation of Expensive Functions with Active Learning*, pages 35–62. Springer Berlin Heidelberg, Berlin, Heidelberg.

Haenlein, M. and Kaplan, A. M. (2004). A beginner's guide to partial least squares analysis. *Understanding statistics*, 3(4):283–297.

Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544.

Kilmer, R. A., Smith, A. E., and Shuman, L. J. (1997). An emergency department simulation and a neural network metamodel. *Journal of the society for health systems*, 5(3):63–79.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lai, K. K., Yu, L., Huang, W., and Wang, S. (2006). A novel support vector machine metamodel for business risk identification. In Yang, Q. and Webb, G., editors, *PRICAI 2006: Trends in Artificial Intelligence*, pages 980–984, Berlin, Heidelberg. Springer Berlin Heidelberg.

Martens, H., Tøndel, K., Tafintseva, V., Kohler, A., Plahte, E., Vik, J. O., B. Gjuvsland, A., and W. Omholt, S. (2013). Pls-based multivariate metamodeling of dynamic systems. *Springer Proceedings in Mathematics and Statistics.*, 56.

Mckay, M., Beckman, R., and Conover, W. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code in wsc '05: proceedings of the 37th conference on winter simulation. *Technometrics*, 42:202–208.

Palermo, G. (2009). Performance of pls regression coefficients in selecting variables for each response of a multivariate pls for omics-type data. *Advances and Applications in Bioinformatics and Chemistry*, page 57.

Pessoa, T., Medeiros, R., Nepomuceno, T., Bian, G.-B., Albuquerque, V., and Filho, P. P. (2018). Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, PP:1–1.

Pinsky, P. and Rinzel, J. (2001). Intrinsic and network rhythmogenesis in a reduced traub model for ca3 neurons. *Journal of Computational Neuroscience - JCNS*, 1.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008). *Global Sensitivity Analysis. The Primer*, volume 304.

Squire, L. (2008). *Fundamental neuroscience.* Elsevier/Academic Press, Amsterdam Boston.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Sterratt, D., Graham, B., Gillies, A., and Willshaw, D. (2011). *Principles of Computational Modelling in Neuroscience.* Cambridge University Press, Cambridge.

Tøndel, K., Indahl, U. G., Gjuvsland, A. B., Omholt, S. W., and Martens, H. (2012). Multi-way metamodelling facilitates insight into the complex input-output maps of nonlinear dynamic models. *BMC Systems Biology*, 6(1):88.

Tøndel, K., Niederer, S. A., Land, S., and Smith, N. P. (2014). Insight into model mechanisms through automatic parameter fitting: a new methodological framework for model development. *BMC systems biology*, 8:59–59.

Tøndel, K., Vik, J. O., Martens, H., Indahl, U. G., Smith, N., and Omholt, S. W. (2013). Hierarchical multivariate regression-based sensitivity analysis reveals complex parameter interaction patterns in dynamic models. *Chemometrics and Intelligent Laboratory Systems*, 120:25–41.

Traub, R., Wong, R., Miles, R., and Michelson, H. (1991). A model of ca3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances. *Journal of neurophysiology*, 66:635–50.

Wold, S., Martens, H., and Wold, H. (1983). The multivariate calibration problem in chemistry solved by the pls method. *Lect Notes Math*, 973.

Xie, J., Girshick, R., and Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487.

Yu, L., Wang, S., and Lai, K. K. (2009). A neural-network-based nonlinear metamodeling approach to financial time series forecasting. *Applied Soft Computing*, 9(2):563 – 574.

A Architectures

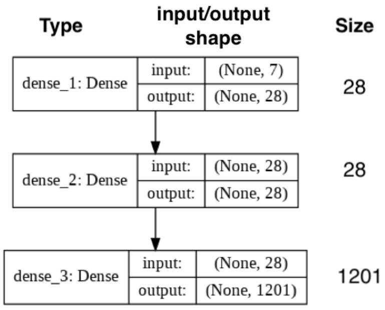


Figure A.1: Architecture of the FFN model used in classical metamodelling of the Hodgkin-Huxley model. Left box: Type of layer. **middle:** shape of input/output data. **Right:** Number of neurons in each layer.

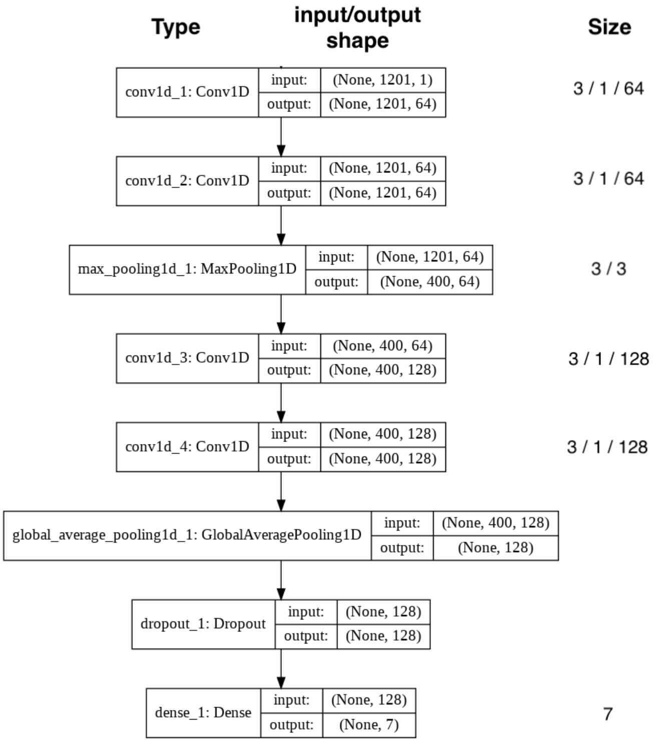


Figure A.2: Architecture of the CNN model used in inverse metamodelling of the Hodgkin-Huxley model. Left box: Type of layer. **middle:** shape of input/output data. **Right:** For dense layers the size represent the number of neurons in the dense layer. For convolution layers it represents *width / stride / number of filters*. For max pooling layers it represents *width / stride*. All convolution layers uses *same* padding.

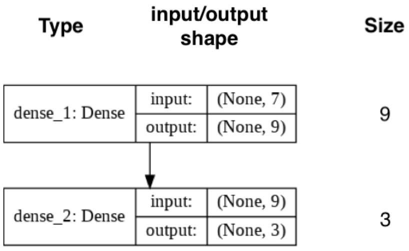


Figure A.3: Architecture of the FFN model used in classical metamodelling of aggregated phenotypes extracted from the Hodgkin-Huxley model. Left box: Type of layer. **middle:** shape of input/output data. **Right:** Number of neurons in each layer.

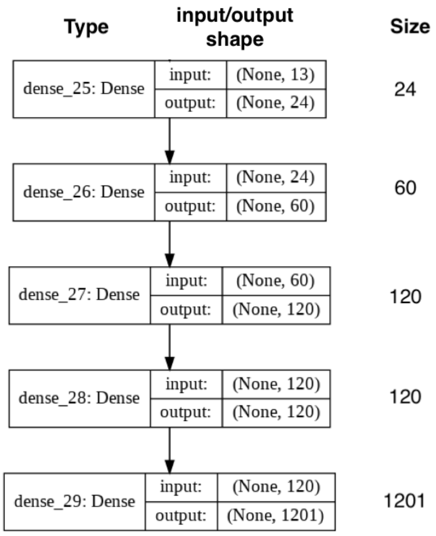


Figure A.4: Architecture of the FFN network used in classical metamodelling of the Pinsky-Rinzel model. Left box: Type of layer. **middle:** shape of input/output data. **Right:** Number of neurons in each layer.

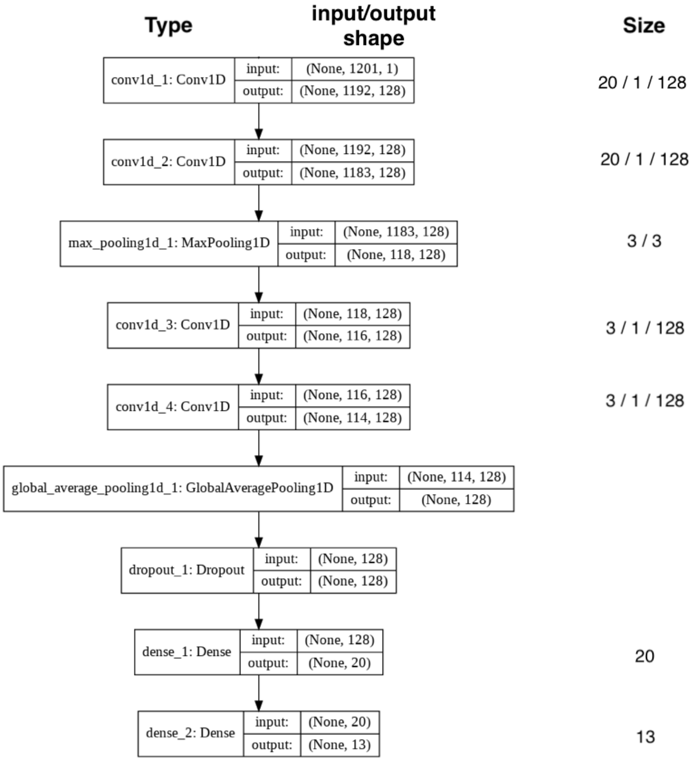


Figure A.5: Architecture of the CNN model used in inverse metamodelling of the Pinksy-Rinzel model Left box: Type of layer. **middle:** shape of input/output data. **Right:** For dense layers the size represent the number of neurons in the dense layer. For convolution layers it represents *width / stride / number of filters*. For max pooling layers it represents *width / stride* All convolution layers uses *valid* padding.

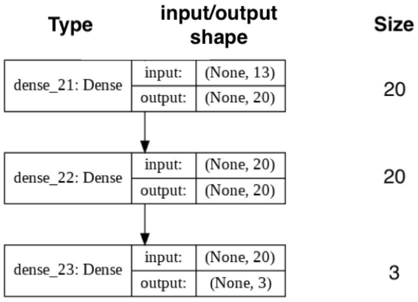


Figure A.6: Architecture of the FFN model used in classical metamodelling of aggregated phenotypes extracted from the Pinsky-Rinzel model. Left box: Type of layer. **middle:** shape of input/output data. **Right:** Number of neurons in each layer.

B Training Plots

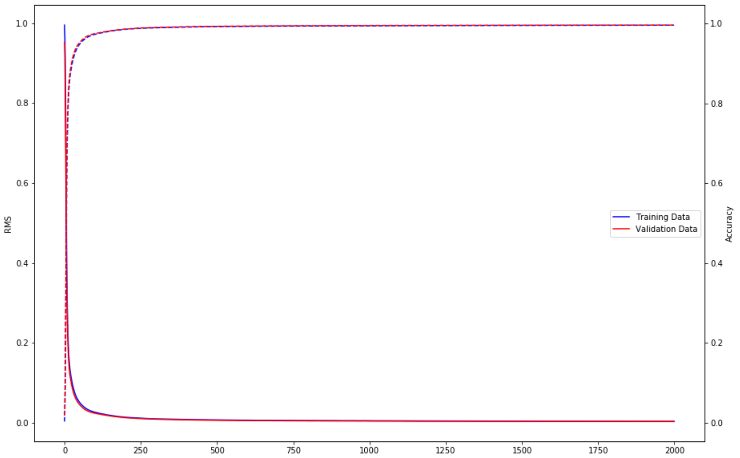


Figure B.1: Training plots from the training of the FFN used in the classical metamodelling of the Hodgkin-Huxley model.

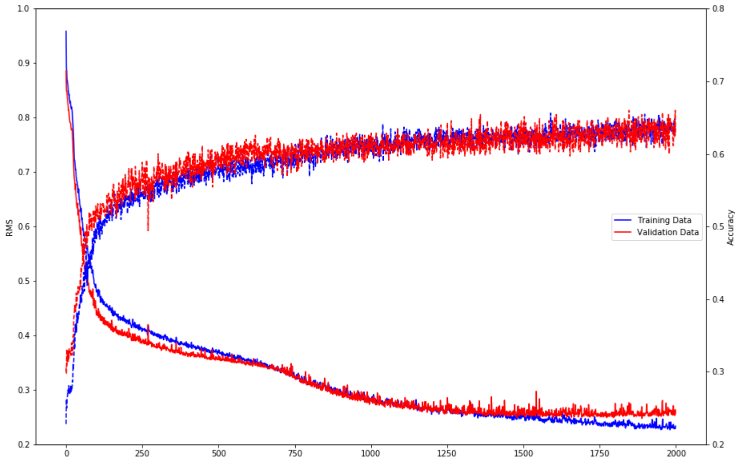


Figure B.2: Training plots from the training of the CNN used in the inverse metamodelling the Hodgkin-Huxley model.

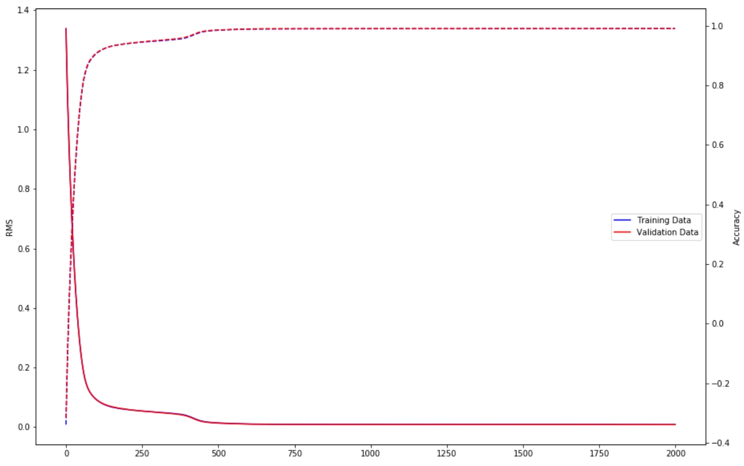


Figure B.3: Training plots from the training of the FFN used in the classical metamodelling of aggregated phenotypes extracted from the Hodgkin-Huxley model.

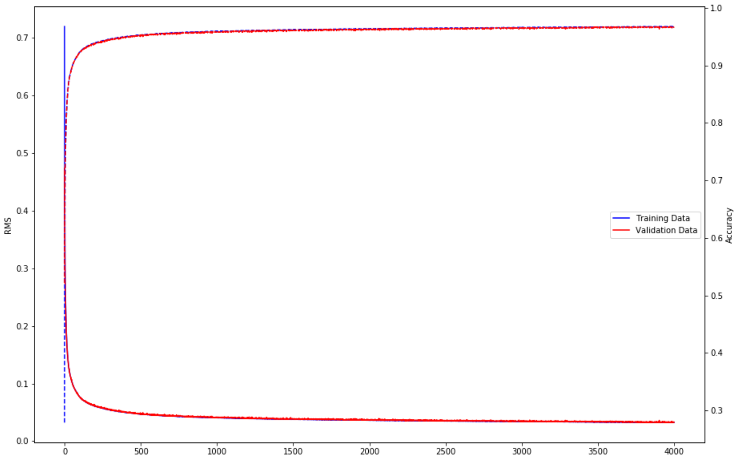


Figure B.4: Training plots from the training of the FFN used in the classical metamodelling of the Pinsky-Rinzel model.

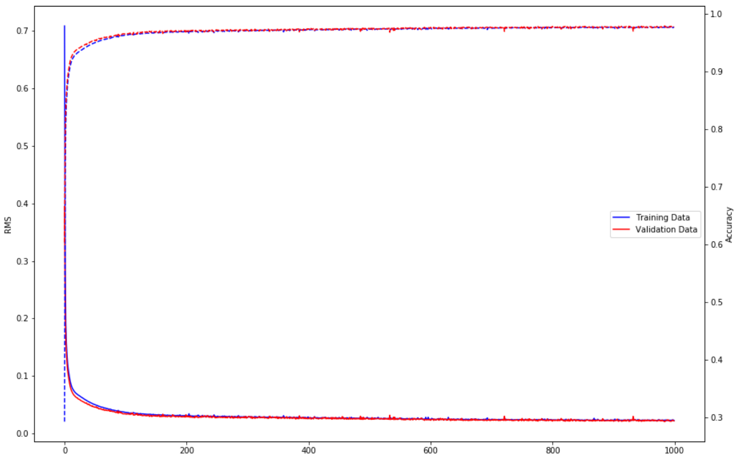


Figure B.5: Training plots from the training of the FFN used in the classical metamodelling of the Pinsky-Rinzel model.

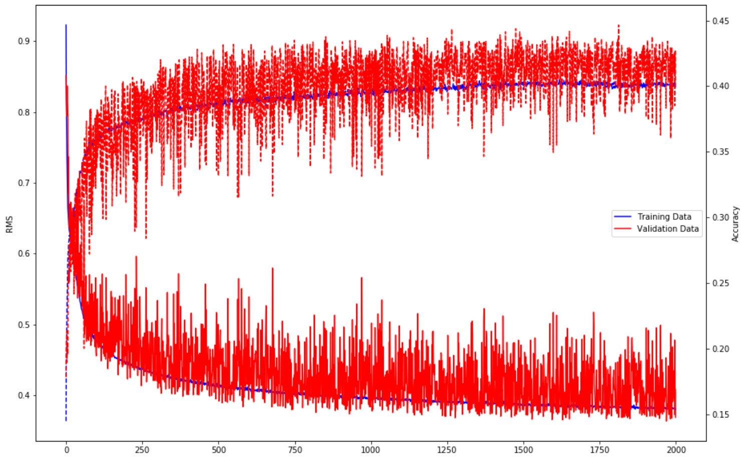
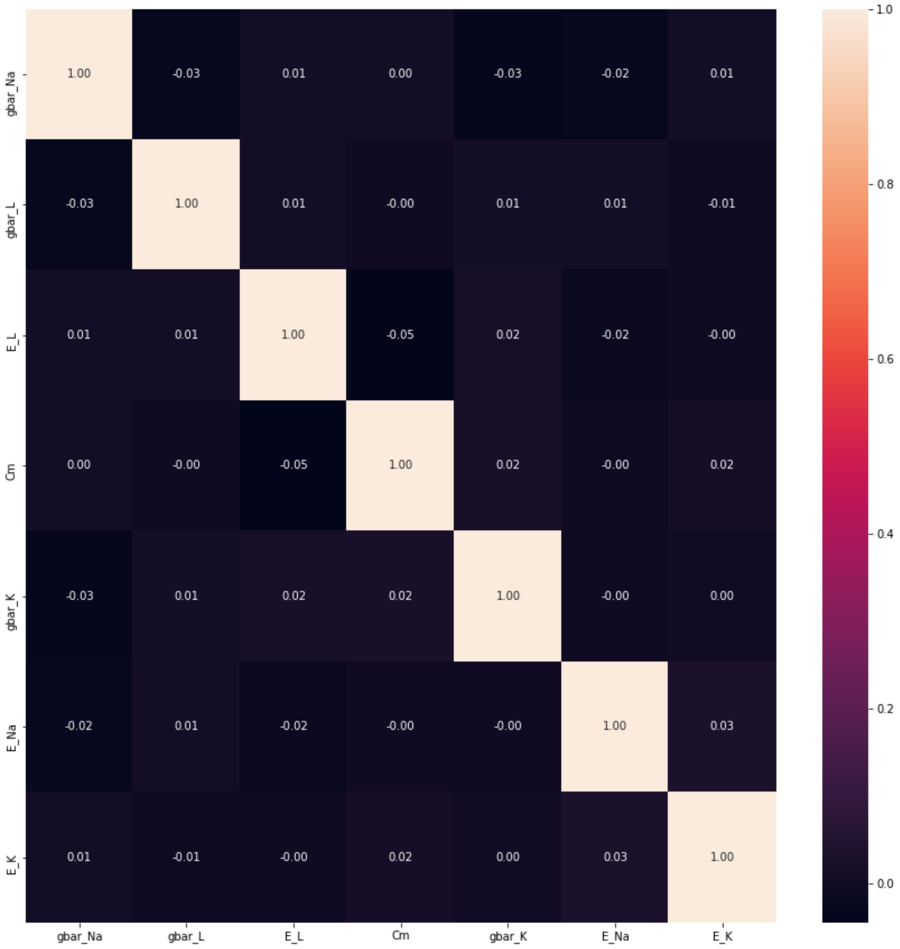


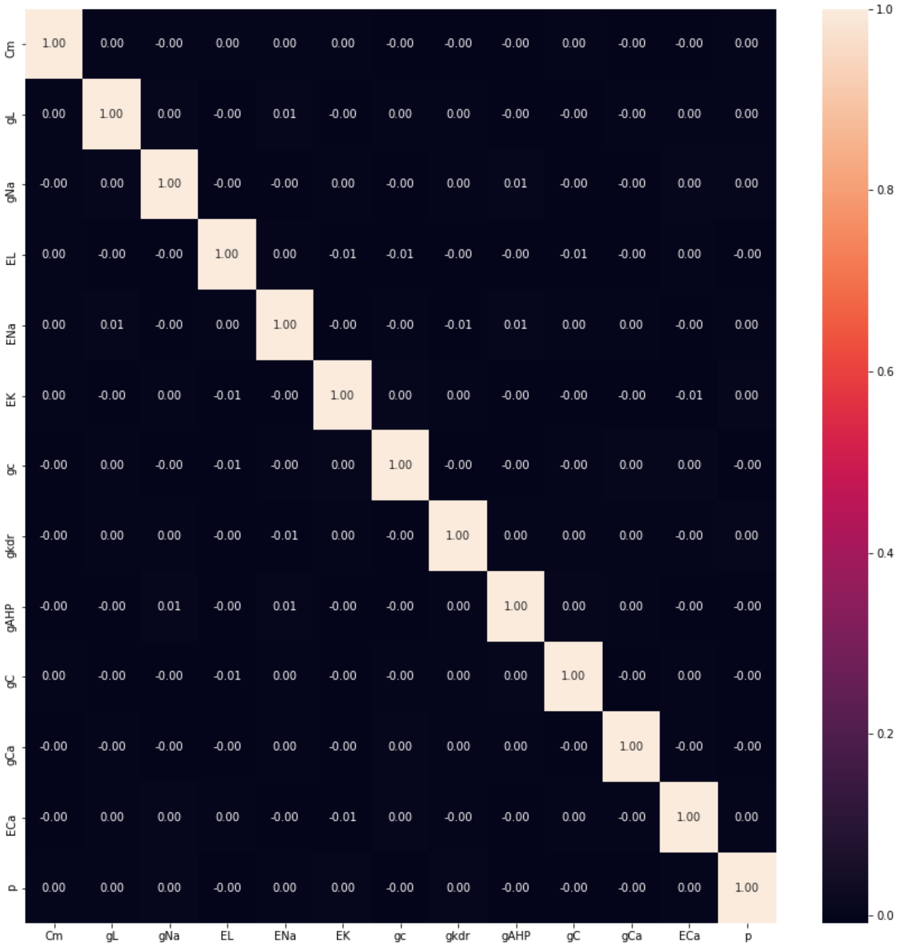
Figure B.6: Training plots from the training of the CNN used in the inverse metamodelling the Pinsky-Rinzel model.

C Correlations



	gbar_Na	gbar_L	E_L	Cm	gbar_K	E_Na	E_K
gbar_Na	1.00	-0.03	0.01	0.00	-0.03	-0.02	0.01
gbar_L	-0.03	1.00	0.01	-0.00	0.01	0.01	-0.01
E_L	0.01	0.01	1.00	-0.05	0.02	-0.02	-0.00
Cm	0.00	-0.00	-0.05	1.00	0.02	-0.00	0.02
gbar_K	-0.03	0.01	0.02	0.02	1.00	-0.00	0.00
E_Na	-0.02	0.01	-0.02	-0.00	-0.00	1.00	0.03
E_K	0.01	-0.01	-0.00	0.02	0.00	0.03	1.00

Figure C.1: Correlation matrix for input parameters used in the classical metamodelling of the Hodgkin-Huxley model.



	Cm	gL	gNa	EL	ENa	EK	gc	gkdr	gAHP	gC	gCa	ECa	p
Cm	1.00	0.00	-0.00	0.00	0.00	0.00	-0.00	-0.00	-0.00	0.00	-0.00	-0.00	0.00
gL	0.00	1.00	0.00	-0.00	0.01	-0.00	0.00	0.00	-0.00	-0.00	-0.00	0.00	0.00
gNa	-0.00	0.00	1.00	-0.00	-0.00	0.00	-0.00	0.00	0.01	-0.00	-0.00	0.00	0.00
EL	0.00	-0.00	-0.00	1.00	0.00	-0.01	-0.01	-0.00	-0.00	-0.01	-0.00	0.00	-0.00
ENa	0.00	0.01	-0.00	0.00	1.00	-0.00	-0.00	-0.01	0.01	0.00	0.00	-0.00	0.00
EK	0.00	-0.00	0.00	-0.01	-0.00	1.00	0.00	0.00	-0.00	-0.00	-0.00	-0.01	0.00
gc	-0.00	0.00	-0.00	-0.01	-0.00	0.00	1.00	-0.00	-0.00	-0.00	0.00	0.00	-0.00
gkdr	-0.00	0.00	0.00	-0.00	-0.01	0.00	-0.00	1.00	0.00	0.00	0.00	-0.00	0.00
gAHP	-0.00	-0.00	0.01	-0.00	0.01	-0.00	-0.00	0.00	1.00	0.00	0.00	-0.00	-0.00
gC	0.00	-0.00	-0.00	-0.01	0.00	-0.00	-0.00	0.00	0.00	1.00	-0.00	0.00	-0.00
gCa	-0.00	-0.00	-0.00	-0.00	0.00	-0.00	0.00	0.00	0.00	-0.00	1.00	-0.00	-0.00
ECa	-0.00	0.00	0.00	0.00	-0.00	-0.01	0.00	-0.00	-0.00	0.00	-0.00	1.00	0.00
p	0.00	0.00	0.00	-0.00	0.00	0.00	-0.00	0.00	-0.00	-0.00	-0.00	0.00	1.00

Figure C.2: Correlation matrix for input parameters used in the classical metamodelling of the Pinsky-Rinzel model.

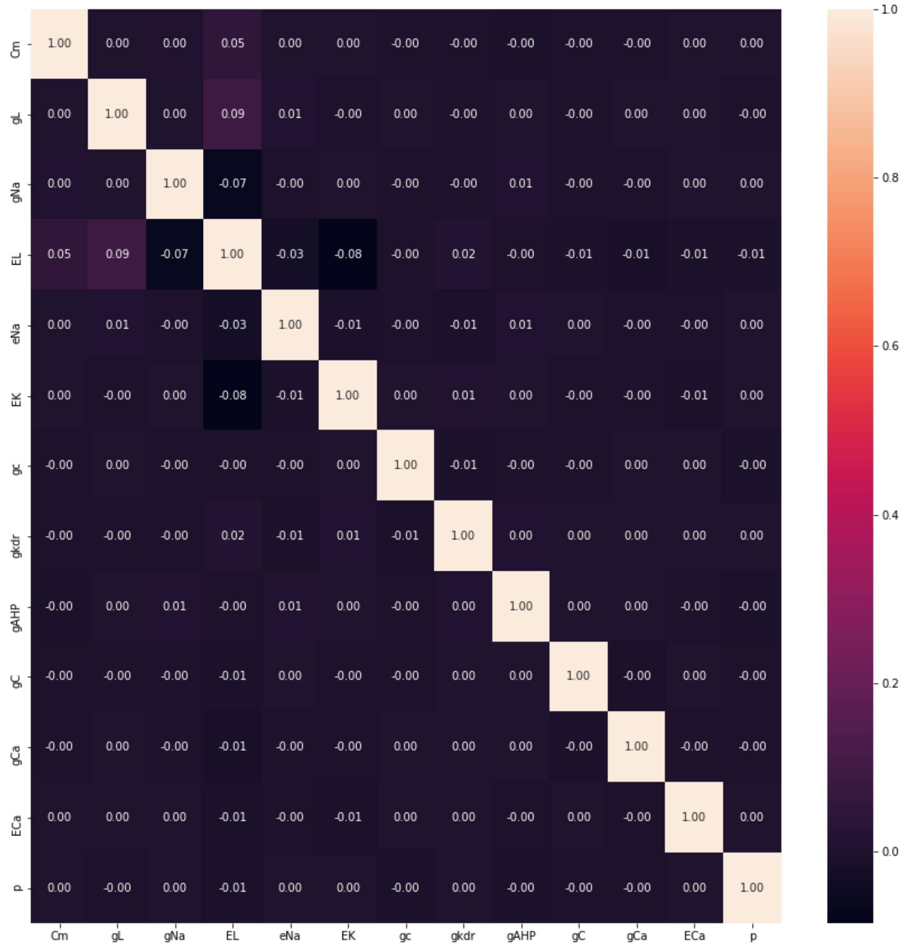


Figure C.3: Correlation matrix for input parameters used in classical metamodelling of aggregated phenotypes extracted from the Pinsky-Rinzel model.

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway