KTH Computer Science
and Communication

# Prediction of peptide retention time based on Gaussian Processes

XUANBIN QIU

Master's Thesis at NADA
External Supervisor: Lukas Käll
Internal Supervisor: Patric Jensfelt
Examiner: Stefan Carlsson

TRITA 2015-3-19

# Abstract

Shotgun Proteomics is the leading technique for protein identification in complex mixtures. However, it produces a large amount of data which results in a extremely high computational cost for identifying the protein. Retention time (RT) is an important factor to be used to enhance the efficiency of protein identification. By predicting the retention time successfully, we could decrease the computational cost dramatically. This thesis uses a machine learning method, Gaussian Processes, to predict the retention time of a set of peptide in hand. We also implement a feature extraction method called Bag-of-Words to generate the features for training the model. In addition, we also investigate the effect of different types of optimization methods to the model's parameters. The results show comparable precision of the prediction and relatively low time cost when comparing with the state-of-art prediction model.

# Referat

Shotgun Proteomics ?r den ledande tekniken f?r proteinidentifiering i komplexa blandningar. Den ger dock en stor m?ngd data som resulterar i en extremt h?g ber?kningskostnad f?r att identifiera proteinet. Retentionstid (RT) ?r en viktig faktor som anv?nds f?r att ?ka effektiviteten vid proteinidentifiering. Genom att f?rutse retentionstiden framg?ngsrikt, skulle vi kunna minska ber?kningskostnad dramatiskt. Denna avhandling anv?nder en maskininl?rningsmetod, Gaussian Processes, f?r att f?ruts?ga retentionstiden av en upps?ttning av peptider. Vi implementerar ocks? en metod som kallas bag-of-words f?r att generera data f?r tr?ning av modellen. Dessutom unders?ker vi ocks? effekten av olika typer av optimeringsmetoder f?r modellens parametrar. Resultaten visar j?mf?rbar precision i prediktionen och relativt l?g ber?kningskostnad j?mf?rt med state-of-art modeller.

# Contents

# Chapter 1

# Introduction

## 1.1 Introduction

Proteins, the most essential components of all the cells and organisms, plays an irreplaceable role in people's daily lives including providing energy, strengthening the immune system and catalyzing metabolic reactions within the body. The functions of different biological organisms are determined by the proteins that are presented in them. For this reason, it is important and necessary for us to know what proteins are in different organisms. As a result, this leads us to the problem of large-scale identification of proteins in complex mixtures.

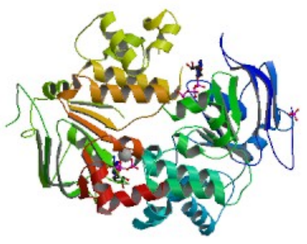


**Figure 1.1.** protein chain

Shotgun proteomics [1][2] is the most popular technique for identifying proteins in complex mixtures. The main part of this technique is high performance liquid chromatography (HPLC). HPLC has three main steps, firstly, the protein in the mixture of interest will be enzymatically digested into peptides. A peptide is a short sequence (chain) of amino acid as in Fig.1.2. Then the resulting peptides are separated according to their hydrophobicity on a thin tube called liquid chromatog-

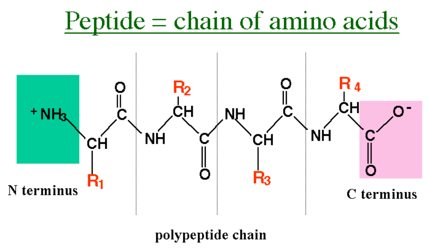


**Figure 1.2.** Peptide model

raphy (LC) column. The hydrophobicity is the physical property of a molecule that is seemingly repelled from a mass of water. To be more specific, a specific solvent will be pumped through the column with increasing concentration. Peptide with different hydrophobicity will gradually elute in the solvent due to their hydrophobicity and then wash out from the column.

In the end, those peptides will separated from the column and ionized by electrospray ionization and then injected into a mass spectrometer. The result is a set of spectra. Each spectrum corresponds to only one or at most a few specific peptides. The fig.1.3 shows the general process of HPLC method.

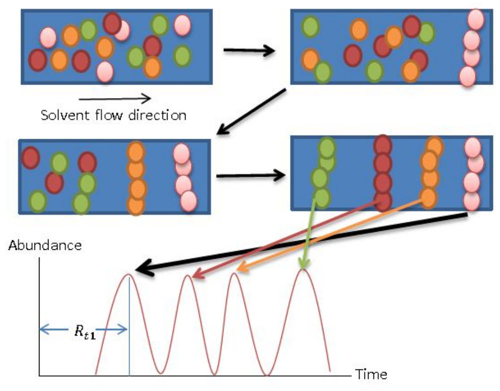


**Figure 1.3.** Demonstration of HPLC method. (circles with different colour represents different peptides and the blue column represents the liquid chromatography column, The $R_{t1}$ is the retention time of $peptidePink$)

Since we want to know which peptides were in the sample, we will compare them with a list of theoretical peptides from the database that we know should be in the sample. Currently, we have to compare every peptide with every spectrum to see if the peptide exists in the sample or not. However, when the number of peptides becomes increasingly large, the complexity of this method becomes extremely high which will result in a poor computational efficiency.

An alternative way to solve this problem is to utilize its retention time. We could reduce the number of peptides compared to a spectrum if we know which peptides are more likely to separated from the column at the time the spectrum is produced. This time is defined as the retention time of a peptide. So the more precise the predicted retention time we could get, the less peptides we need to match and the higher efficiency we could have to identify the peptide.

## 1.2 Problem Statement

Different models have been applied to the task of predicting the retention time of the peptides. For example, the Sequence Specific Retention Calculator (SSRC) which predicts retention times based on amino acid sequences is available on-line [18].

To train a regression model, we first need some proper features which can represent the peptide in a vector space. Since most of most of the available features are biochemical features, we are interested in BOW features that have shown to be successful for sequence analysis. Therefore, in this thesis will utilize these features for training model and compare their performance to the traditional biochemical features.

The main algorithm behind current RT prediction methods is mostly based on either Support Vector Regression (SVR) or Artificial Neural Network (ANN) and all of these methods have their own drawbacks that need to be improved.

In this thesis, we would like to apply the Gaussian Processes (GP) model to predict the retention time and investigate its performance by comparing it with the other existing methods.

GP provides a flexible framework for probabilistic regression and is widely used to solve high-dimensional, small-sample or nonlinear problems. Yet none of the existing publications has used this model for predicting retention times.

In addition, since GP model has some parameters need to be optimized which have a significant impact on the prediction accuracy, in this thesis, we will investigate different optimization methods towards these parameters and analyze their properties.

## 1.3 Outline

The structure of the thesis is as following: Chapter 2, introduces the related works. Chapter 3, provides the whole methodologies and is divided into 3 sections, section 1 provides the detailed description of Gaussian Processes and a detailed mathematics derivation and section 2 gives the idea of Bag of word method for features selection and section 3 will cover the two optimization methods that are used to optimize the parameters of our model. Chapter 4, the experimental results and evaluation will be analyzed and our model will be compared with some other popular predictors. Chapter 5, a discussion of the limitation of our model will be given. Chapter 6, discusses the future work and gives some perspectives for it.

# Chapter 2

# Related Work

Prediction of the retention time for a given peptide could help to improve the confidence of peptide identification. Therefore, it is important and necessary to develop efficient algorithms and establishing proper models to generate the prediction of peptide.

To design a successful predictor for our task, one would has the following questions. Firstly, what kind of model will have better performance of the prediction? Secondly, what factors will have a great impact on the retention time? In order to answer the questions, a number of models based on different features have been proposed to characterize the peptide and to predict its retention time.

## 2.1 Models / Methods

Different kinds of machine learning models have been implemented. One of the model is the multiple linear regression (MLR) model. In [4], Baczek T, Wiczling P, Marszall M *et al.* has implemented this model to predict the peptide retention time at different HPLC conditions and shows high precision. However, the linear regression model mainly depends on the chromatographic condition and thus prediction bias could be a serious problem when the model is used under different circumstances.

In [5], Petritis K *et al.* has proposed another model based on artificial neural network (ANN). The ANN model was based on the contributions of individual amino acids taken the neighborhood effect into account and a data set of 7000 was used for the training. The large amount of experimental data provided by this approach produced high accuracy of the retention time prediction and enhance the confidence of the peptide identification. In [6], Petritis K *et al.* have improved the ANN model by adding more descriptors, such as peptide length, sequence, hydrophobicity as well as nearest-neighbor amino acid. Each of the peptide is characterized by 1052 features instead of 20. The new model was trained by 345,000 peptide and tested using another 1303 confidently identified peptide which reported excellent performance.

Although a great deal of effort has been made into improving the models, there are still some remaining issues. One of the main weaknesses of ANN is its time consumption. In order to train ANN, a large amount of training data is required which will result in a high cost in terms of both time and resources. What's worse, its high computational cost also implies its weakness in terms of repeatability and comparability which are essential to laboratory research. It also limits its development for commercial purpose.

In order to solve this problem, another alternative machine-learning method, support vector machine(SVM) has been implemented. In [7], Klammer *et al.* proposed a predictor based on this method. Compared to ANN, SVM require a smaller training set which can reduce the computational cost dramatically. Besides, dynamic SVR (support vector machine for regression problem) model avoids the RT variation between different chromatographic conditions which could have better performance than the linear regression model in terms of adaptability. However, using small dataset with low complexity to train the model also leads to the poor performance in terms of accurate regression and gives lower quality prediction compared to ANN. If we simply use the same amount of training data as in ANN, the SVR model could have extremely high computational cost which might be even higher than ANN.

To enhance the capacity of the predictor and raise the predicted accuracy, Luminita Moruz*et al.* in [8] has developed a new predictor, ELUDE, which is based on support vector regression (SVR). The predictor firstly derives a retention time index for the condition at hand. Then by using those indices, the predictor creates 60 peptide features which are optimally combined during a second training procedure. After that, $\epsilon - SVR$ is used to train the model based on the features. The resulting retention model can be subsequently used to predict the retention time of other peptides of interest.

The performance of Elude is excellent and considered as one of the state-of-the-art retention time predictors. However, one of the shortcomings of the Elude algorithm is the rather limited use of positional information. The positional information here is mainly about the relative position of different amino acids in the peptide. The retention index used for computing features is calculated by only using the amino acid composition. Adding positional data is likely to capture more information, and thus lead to higher prediction accuracy [9].

However, even with the proper models, the prediction precision could be extremely low if the selected features are improper. So the features selection is also a essential part of an successful prediction.

## 2.2 Features

The effect of an individual amino acid was firstly taken into account when building the model because evidence has implied that the inclusion of this type of additional information could increase the confidence of peptide identification [4]. Since peptides are composed of amino acids, it can increase the performance the chromatographic behavior [10]. A set of retention time coefficients were generated from only different amino-acid composition by using iterative regression methods [11] [12] [13]. To be specific, the regression model was trained by amino-acid with different compositions and their corresponding retention time. As a result, the model gives a set of coefficient regarding the amino-acid composition and use them to represent the peptides.

However, Author Houghten RA *et al.* in [12] pointed out that the relative location of different amino acids in a peptide could affect the behavior of the peptide too. In[14], author Zhou NE *et al.* has also emphasized the relation between the chromatographic behavior of peptides and their amino acid composition. Retention coefficient of one amino acid need to be assigned different value according to its neighborhoods amino acid and the type of peptide. Besides, other biological characters, such as hydrophobicity [15], ion-pairing reagents [16], stationary phase [17] and even the length of the peptide could influence the peptide retention time.

In [18], O.V. Krokhin and R. Craig have developed an improved model, Sequence-

Specific Retention Calculator(SSRC), for prediction of retention times in ion pair reversed-phase based on a database of 346 peptides. The ability of this model for prediction can assist detailed peptide mapping significantly, thus increase confidence in peptide identification and the protein characterization. However, the model is not completed and has not included amino acid modification that may occur during the sample preparation. In addition, the prediction capacity becomes worse for an experimental condition which diverges from the setup.

After SSRC, several different kinds of factors have been used. One of them is the structural descriptors. Author Kaliszan R, Baczek T, Cimochowska A *et al.* in [19] has employed the Quantitative structure-retention relationships with the information of the peptide's chemical characters and its structure . In [4], Baczek T, Wiczling P, Marszall M *et al.* has also developed a predictor based on three types of chemical structural descriptors and both of them showed good results with the experimental data.

In general, most of the features that are widely used have strong biological meaning, such as amino acid composition, hydrophobicity of the peptide and some other chemical structural descriptors and the performances of these features could fulfill the requirement. However, few of the less-biological meaning features have been tried so far.

From the perspective of computer science, anything that can represent an object could be considered as a feature of this object. Therefore, it is worth to believe that some less-biological meaning features could give the same performance in this case. It might be a breakthrough to introduce some new features instead of just using the biochemical features. In this thesis, some features with less biological meaning will be utilized and compared to the traditional biochemical features. 2

# Chapter 3

# Methodology

To predict the peptide's retention time, there are mainly three things we need to cover: feature selection, model building and parameter optimization. The idea is that we first extract some representative features from the peptides, then we train GP regression model by using these features. At the meanwhile, we try to optimize the parameters of the model so that it can better fit the data. In this chapter, we will introduce each of these steps and explain them in details.

## 3.1 Feature selection

### 3.1.1 Bag of Words

We are describing the bag-of-words feature extraction method. This method has shown excellent performance in other aspects of computer vision, such as image analysis and image recognition.

As mentioned in Chapter 2, few of the new features which have excellent performance in other fields have been implemented to address this problem so far. Therefore, we would like to introduce the Bag-of-words method for feature selection.

The peptide could be considered as a string where each amino acid is a letter. Therefore, one alternative way is to extract the features from the peptide is to implement a method used for analysing strings. Bag-of-words model is one of the most popular methods for analysing strings. This method is widely used in computer science and data mining. Bag-of-words enables us to compare strings by embedding them into a vector space.[20].

To be more specific, the peptide sequence is characterized by a predefined set of sub-strings which is considered as the embedding language $L$ and each of the sub-strings in this set is a word $w$ of $L$. We use *n-grams* to define the embedding language $L$ and its words $w$.

The *n-grams* method simply slides a small window with fixed length $n$ along all of the peptide sequences to extract substrings with length $n$. The language $L$ then consist of all the possible substrings that have been extracted.

After establishing the embedding language $L$, we try to project each peptide $p_i$ into a vector space composed by the words of $L$ using a function $\phi$ defined as

$$\phi : p_i \rightarrow \phi_w(p_i), \quad w \in L \tag{3.1}$$

where the return of $\phi_w(p_i)$ is defined as frequency for the occurrences of $w$ in $p_i$.

In our case, we extract features based on *2-grams* to build our embedding language $L$ then each $w$ in $L$ is a 2-amino-acid group. Since we have 20 different amino acid in total, each of the peptide will be represented by the frequencies of occurrences of 400 words. A graphical flow of the bag-of-word method can be seen as Fig.3.1

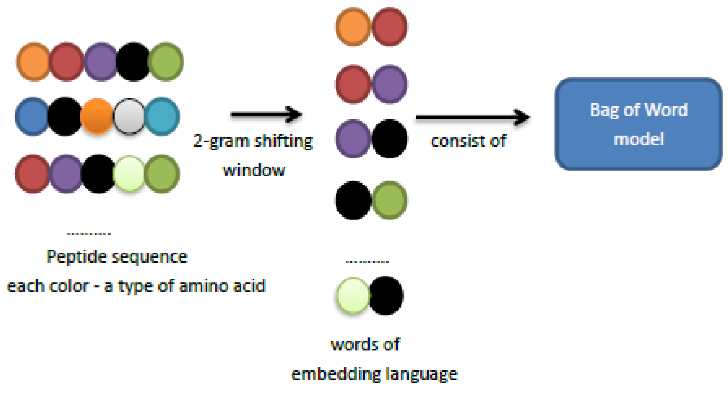
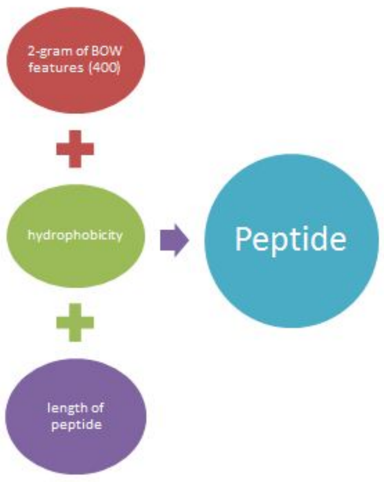


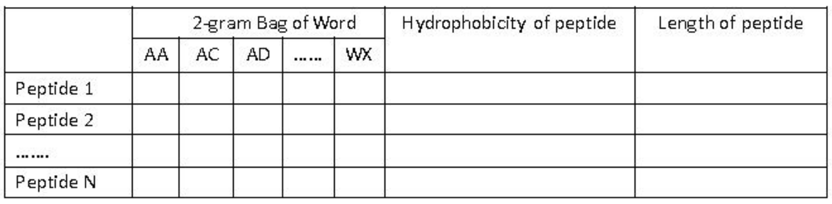
**Figure 3.1.** Bag of Word

To give more meaning to the features, we can include some biological features that are not captured by Bag-of-words. The first feature is the length of the peptide since Since the number of the words $w$ in $L$ is directly related to it and the second feature is the hydrophobicity because it is one of the determinative factors of the retention time. Adding these will result in a 402 dimensional feature as fig.3.2 showed.

Each peptide $p_i$ is then represented by 402 features as fig.3.2(a) and all of them together form the feature matrix as fig.3.2(b).



(a) Feature for each peptide



| | 2-gram Bag of Word | | | | | Hydrophobicity of peptide | Length of peptide |
|---|---|---|---|---|---|---|---|
| | AA | AC | AD | …… | WX | | |
| Peptide 1 | | | | | | | |
| Peptide 2 | | | | | | | |
| ……. | | | | | | | |
| Peptide N | | | | | | | |

(b) Feature matrix for all peptide

**Figure 3.2.** Feature matrix of peptide

The advantages of the Bag-of-words method are mainly discussed in two aspects: complexity and flexibility.

In terms of complexity, the most popular features that have been used so far are mainly biochemical features. However, in order to calculate these features, a large amount of time is required. What's worse, some particular facilities are needed to calculate some of them, for example, if we use the structure descriptor as a feature, we need to use some structure analyzer to produce them which is quite expensive and time consuming. Comparing with computing these features, the computational cost of our method is very low. According to [20], the run-time complexity of Bag-of-Words is only $O(|x| + |y|)$ when it is $O(|x| \cdot |y|)$ for another benchmark method where $x$ and $y$ are the peptide sequences while $|x|$ and $|y|$ are the maximal nonzero dimensions of $\phi_w(x)$ and $\phi_w(y)$.

As for the flexibility, since the embedding language $L$ is constituted by the words that are directly and simply extracted from the samples, this method could easily change the language set $L$ for particular context of the task. This property implies a wider applicability in other field of biological research.

## 3.2 Gaussian Processes

By implementing the Bag of Word model, we now have the features to present the peptides in hand. The next step is to establish a nonlinear regression model to find out the relation between the peptides and their retention time. In this thesis, we use GP regression model to make this prediction.

Gaussian Processes is a supervised machine learning method based on Bayesian theory and statistical learning. In order to understand the theory behind it and build an appropriate model, this chapter starts with brief introduction to the supervised learning method as well as the specification of Standard Bayesian linear Regression. Then we will specify the Gaussian Process in our case along with the detailed mathematical derivations. After that, we will specify the concept of multi-kernel functions in Gaussian Process and explain its properties. Finally, we will make a comparison between Gaussian Process and the other two methods, such as ANN and SVR.

### 3.2.1 Standard Bayesian linear Regression

Let's first consider a set of observations $\mathcal{D} = \{(x_i, y_i)|i = 1, 2...n\}$ along with some noise $\epsilon$. In the regression setting, the targets are real values with additive noise. We are interested in inferring the relationship between inputs and targets, i.e. the conditional distribution of the targets given the inputs.

In supervised learning, one way to decide this relationship is using non-parametric regression. It uses a set of $w$ as the best descriptors of the model to describe the relationship between the inputs and outputs. The problem now comes to how to evaluate if the trained model fits the observations.

There are two methods for performing this evaluation, loss function and maximum likelihood. The first one is achieved by optimizing the parameters $w$ that minimizing the predefined loss function, such as minimal squared error function (MSE). However, one of the obvious shortages of this method is the over-fitting. To minimize the error of the model, one would increase the complexity of the model that might easily over fit to the training set. This means the model's predictive capacity for unknown (testing) data could be really bad even though it has high predicted precision in terms of training data. In addition, making a simpler model by ignoring the noise could avoid the over-fitting problem but could also lead to lower accuracy for prediction.

An alternative way to identify the quality of model is Maximum Likelihood, also known as Bayesian regression. The idea of Bayesian regression is finding the optimal parameters $w$ that maximizes the likelihood function. The main difference between Bayesian regression and loss function is the fact that Bayesian regression considers a prior distribution for $w$, having this distribution prevents the model from over-fitting and forces it to be smooth.

The Bayesian analysis of the linear regression model can be written as :

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{w}, \quad y = f(\mathbf{x}) + \epsilon, \tag{3.2}$$

where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is the parameters of the model. The function value is represented by $f(\mathbf{x})$ and a bias (noise) is included. We assume that the $\epsilon$ is independent identically distributed Gaussian noise with zero mean and variance $\sigma_n^2$.

$$\epsilon \sim N(0, \sigma_n^2), \tag{3.3}$$

The likelihood function, the probability density of the observation given the parameters, will then have the following form:

$$p(\mathbf{y}|X, \mathbf{w}) = \prod_{i=1}^{n} p(y_i|\mathbf{x}_i, \mathbf{w}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} exp(-\frac{(y_i - \mathbf{x}_i^T\mathbf{w})^2}{2\sigma_n^2}) \tag{3.4}$$

The idea of Bayesian linear model is mainly based on the posterior distribution computed by Bayes' rule:

$$p(\mathbf{w}|\mathbf{y}, X) = \frac{p(y|X, \mathbf{w})p(\mathbf{w})}{p(y|X)} \tag{3.5}$$

To calculate the posterior, we also need a prior over the parameters. The prior is normally considered to represent our prior beliefs over the distribution of the parameters we expect to observe before seeing any data. Therefore, we put a zero mean Gaussian prior with covariance matrix $\Sigma$ on the weights:

$$\mathbf{w} \sim N(0, \Sigma) \tag{3.6}$$

Another essential part is the marginal likelihood function $p(\mathbf{y}|X)$. The marginal likelihood,(normalizing constant), gives the likelihood of the output by considering only one specific kind of input and is independent of the other factors. To be specific, the marginal likelihood of expected output here only consider the effect of the input $\mathbf{x}$ without taking the effect of the weights $\mathbf{w}$ into account. Therefore, it is given by

$$p(\mathbf{y}|X) = \int p(\mathbf{y}|X, \mathbf{w})p(\mathbf{w})dw \tag{3.7}$$

By combining the likelihood function in 3.4 with the prior in 3.6 and marginal likelihood function in 3.7, we could obtain the actual form of the posterior $p(\mathbf{w}|\mathbf{x}, y)$ in 3.5

$$p(\mathbf{w}|\mathbf{y}, X) = \frac{p(\mathbf{y}|X, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|X)} \sim exp(-\frac{1}{2}(\mathbf{w} - \bar{\mathbf{w}})^T(\frac{1}{\sigma_n^2}XX^T + \Sigma^{-1})(\mathbf{w} - \bar{\mathbf{w}})) \tag{3.8}$$

From 3.8, it can be seen that the posterior is actually another Gaussian distribution with the new mean and new covariance matrix.

$$\begin{aligned} \bar{\mathbf{w}} &= \sigma_n^{-2}(\frac{1}{\sigma_n^2}XX^T + \Sigma^{-1})^{-1}X\mathbf{y} \\ \Sigma_{new} &= \frac{1}{\sigma_n^2}XX^T + \Sigma^{-1} \end{aligned} \tag{3.9}$$

For Bayesian linear regression, the prediction is made by all possible parameters along with corresponding posterior probability. Thus, we could conclude the predictive jointly distribution of the output $f(x_{new})$ of a new data point $x_{new}$ based on the training dataset $X$ along with their weights $\mathbf{w}$ and their outputs $\mathbf{y}$ as follows :

$$\begin{aligned} p(f(x_{new})|x_{new}, \mathbf{y}, X) &= \int p(f(x_{new})|x_{new}, \mathbf{w})p(\mathbf{w}|X, \mathbf{y})dw \\ \sim N(\frac{1}{\sigma_n^2}x_{new}^T(\sigma_n^2XX^T + \Sigma^{-1})^{-1}X\mathbf{y}, &\, x_{new}^T(\sigma_n^2XX^T + \Sigma^{-1})^{-1}x_{new}) \end{aligned} \tag{3.10}$$

### 3.2.2 Gaussian Processes Regression

A Gaussian Process regression model inherits the Bayesian theorem and compute the posterior predictive distributions for new test inputs instead of identifying the "best-guess" predictions for them as SVR did.

A Gaussian Process, a distribution over a set of functions, is fully specified by its mean function and covariance function as 3.11

$$f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{3.11}$$

where the mean function $m(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}')$ are:

$$m(\mathbf{x}) = E[(f(\mathbf{x}))] \tag{3.12}$$

$$k(\mathbf{x}, \mathbf{x}') = E[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \tag{3.13}$$

Normally, for the sake of simplicity, we will take the mean function to be zero and only consider the effect of covariance function. In fact, the problem of learning in Gaussian Processes is the problem of finding suitable properties for the covariance functions[21].

Since the Bayesian linear regression model has natural disadvantages in terms of expressiveness and the relationship between our input (peptide sequence) and output (retention time) is barely linear, we take the advantage from Gaussian Processes in terms of dealing with nonlinear problem by utilizing the kernel function.

The principle of a kernel function is to project the original nonlinear input data into a higher dimensional space where they become linear separable so that the linear model could be implemented directly in that space. The Fig3.3 shows the idea of the kernel function

To be more specific, the Gaussian Process specifies the covariance function as the kernel function which takes the inner dot product between a pair of random variables. We assume the observations have independent identically distributed Gaussian noise, then the prior of the observations becomes:

$$cov(\mathbf{y}) = K(X, X) + \sigma_n^2 I \tag{3.14}$$

where the $K(X, X)$ is a general form of kernel function, $I$ is a identify matrix.

Then, we could generate the joint distribution of the observations $\mathbf{y}$ and the predicted value $f_{new}$ under the prior as

$$\begin{bmatrix} \mathbf{y} \\ f(x_{new}) \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_{new}) \\ K(X_{new}, X) & K(X_{new}, X_{new}) \end{bmatrix}\right) \tag{3.15}$$

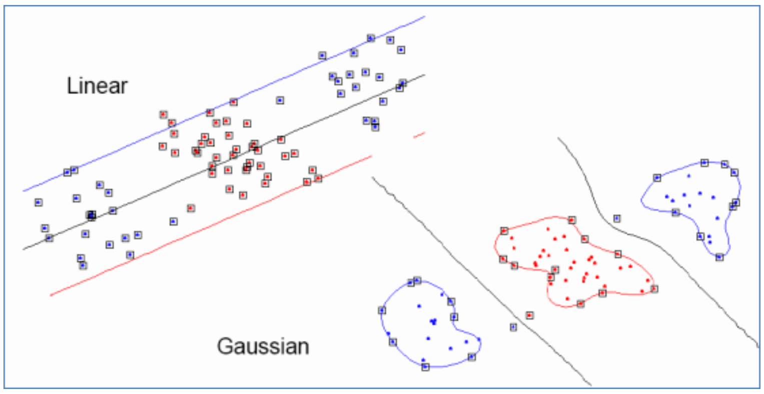


**Figure 3.3.** Projection of data from low-dimensional space into higher dimensional space by Radius Basis Function makes inseparable data separable

Deriving the conditional distribution, we finally arrive at the key predictive equations for Gaussian process regression

$$f_{new}|X, \mathbf{y}, X_{new} \sim N(\bar{f}(X_{new}), cov(f(X_{new}))), \quad where \tag{3.16}$$
$$\bar{f}(X_{new}) = K(X_{new}, X)[K(X, X) + \sigma_n^2 I]^{-1}\mathbf{y} \tag{3.17}$$
$$cov(f(X_{new})) = K(X_{new}, X_{new}) - k(X_{new}, X)[k(X, X) + \sigma_n^2 I]^{-1}k(X, X_{new}) \tag{3.18}$$

These equations give us the corresponding mean value and covariance value of the predictive point which is used to represent the predicted point. A graphical model of Gaussian Processes is can be seen in Fig3.4.

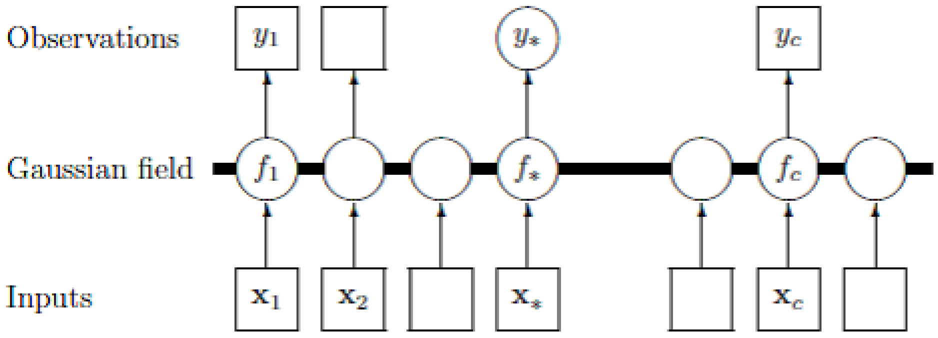


**Figure 3.4.** Graphical model for a GP for regression[21].

### 3.2.3 Kernel Function and Multi-Kernels

As we could seen from 3.16, the accuracy of the estimation of our retention time is highly dependent on the covariance function $k(x_a, x_b)$ . In Gaussian Process model, there are several popular covariance functions.

- Matern covariance:

$$k(x_a, x_b) = (1 + \frac{\sqrt{3}(x_a - x_b)}{l})exp(-\frac{\sqrt{3}(x_a - x_b)}{l}) \tag{3.19}$$

- Linear covariance:

$$k(x_a, x_b) = x_a * x_b \tag{3.20}$$

- Polynomial covariance:

$$k(x_a, x_b) = (x_a * x_b + 1)^k \tag{3.21}$$

- Rational quadratic covariance:

$$k(x_a, x_b) = (1 + \frac{|x_a - x_b|^2}{2\alpha l^2})^{-\alpha} \tag{3.22}$$

- RBF covariance:

$$k(x_a, x_b) = exp(-\frac{|x_a - x_b|^2)}{2l^2}) \tag{3.23}$$

Different covariance functions has different properties and generates different similarity measures between pairs of data points. The RBF covariance in 3.23 is the most popular kernel function. In Gaussian process, all these covariance function are weighted by a smoothness factor $\sigma_f$ in practical. This variable together with the $\sigma_n^2$ and other free parameters in each covariance function are known as the hyperparameters. These hyperparameters will need to be optimized to allow the model to have better performance. In this thesis, we simply set the weight $\sigma_f$ to be 1 and have not considered its effect. The optimization is achieved by maximizing the objective function (marginal log-likelihood function) of GP model. The detail of how to achieve it will be covered in the section of optimization.

Another interesting thing of these kernel functions is that they could be divided into two types: global kernel function and local kernel function. The RBF kernel function is considered as a typical local kernel while the Polynomial kernel is a global one[24].

Both types of kernel functions have their own strengths and limitations in different aspects. The local kernel function has strength in terms of approximation but the generalization ability is weak while the global one has stronger generalization ability but weaker approximation capacity.

Since we want the kernel function to have stronger capacity, we would like to overcome the weakness of the single kernel function. Therefore, one simple way to achieve this is combining those two types of kernel function together as a multi-kernel function as 3.24.

$$k_{mix} = m * k_{poly} + (1 - m) * k_{rbf} (0 \leq m \leq 1) \tag{3.24}$$

where $m$ is the weight between two kernel functions. We create a multi-kernel function by combining RBF kernel function with Polynomial kernel function linearly. We can see that if $m = 0$, the multi-kernel is actually a RBF kernel and it is a polynomial kernel when $m = 1$. By combining two kernels as a new one and find a proper weight between them, we could achieve our goal. As for how to find the proper weight factor $m$, we will also explain in the section of optimization. The performance of this multi-kernel function will also be evaluated and compared to the single kernel function later.

### 3.2.4 Properties of Gaussian Processes

As a supervised learning algorithm, the Gaussian Processes is also based on the idea that similar inputs result in similar outputs or output distributions. But it also have its unique strengths compare with ANN (Back-propagation (BP) algorithm as example) and SVR.

Since BP algorithm is based on the Empirical risk minimization, it is minimizing the expected error of the model by minimizing the differences between the output of training data and their corresponding true value. This will lead to a serious overfitting problem and poor generalization for new data, since it only optimizes the parameters based on training set without considering the test set. Secondly, for a particular BP model, the number of layers of hidden units and the number of units in each layers play an irreplaceable role in ANN algorithm but how to decide these value still remain as a series problem and need to solved. Besides, the BP model adjusted the weights by iteratively calculating the difference between the predicted value and the true ones and giving feedbacks to the system, this principle is in fact similar to the idea of gradient descending. As a result, it will have a low convergence rate and could not guarantee to find the global optima every time.

As for the SVR, it improves this situation by using structure risk minimization and introducing the penalty coefficient. The structure risk minimization describes the minimization of the sum of the empirical risk and VC confidence which is caused by the complexity of the function space. The VC confidence is related to the amount of training data and could be considered as a reflection of generalization ability of the model. The more training data it has, the smaller the VC confidence will be and the stronger generalization the model will have. These principles could avoid the overfitting problem efficiently and allow the model to have a better generalization capacity. However, the SVR still has many obvious problems such as the choice of the type of kernel function and its parameters and the proper value for the penalty coefficient. In addition, the output of SVR lacks probabilistic meaning.

The Gaussian Processes is based on the Gaussian distribution and Bayesian theory which implies a solid foundation of probabilistic and statistic. For example, the Gaussian Process not only gives the predicted value but also the uncertainty of the prediction. This property makes the prediction easier to interpret and understand. What's more, since Gaussian Processes make the prediction based on the distribution of all data and weaken the contribution of single outlier, it prevents the overfitting problem and gives better generalization capacity than SVR. In addition, in terms of the choice of hyperparameters, the Gaussian Processes is self-adaptive. In other words, the Gaussian Process choose the optimal hyperparameters during its process of training a model while SVR could only choose them by using cross-validation method or based on the pervious experiences. Besides, compared with the BP and SVR model, the number of parameters of Gaussian Processes is quite small which makes the optimization easier to implement.

## 3.3 Optimization

In the Gaussian Process model, the most important part is the covariance function of the model since the Gaussian Process consider it as the kernel function. As we mentioned before, the parameters in each covariance function are known as the hyperparameters. These hyperparameters directly determine the properties of the covariance function and further the GP model. Thus, we want to optimize these hyperparameters to get the best combination of these hyperparameters so that we could have a better model for predicting the retention time.

As for the definition of best combination, we are simply looking for the combination of the hyperparameters that maximizes the value of the objective function. In this case, the objective function is the negative log marginal likelihood of Gaussian Processes as 3.25

$$max f(\mathbf{x}) = -logp(\mathbf{y}|X,\theta) = \frac{1}{2}\mathbf{y}^T(C)^{-1}\mathbf{y} + \frac{1}{2}log|C| + \frac{n}{2}log(2\pi) \tag{3.25}$$

where $C = K(X,X) + \sigma_n^2 I_n$

Here, we will implement two different optimization methods based on different principles. The first one is the Conjugate Gradient method and the second one is the Particle Swarm Optimization method. The principle of each method will be explained in detail along with their own strengths and weaknesses towards this case.

### 3.3.1 The Conjugate Gradient

The conjugate gradient method (CG) is one of the most widely used methods and is developed based on the gradient descending method and Newton Method.

In CG theory, we have two important assumptions:

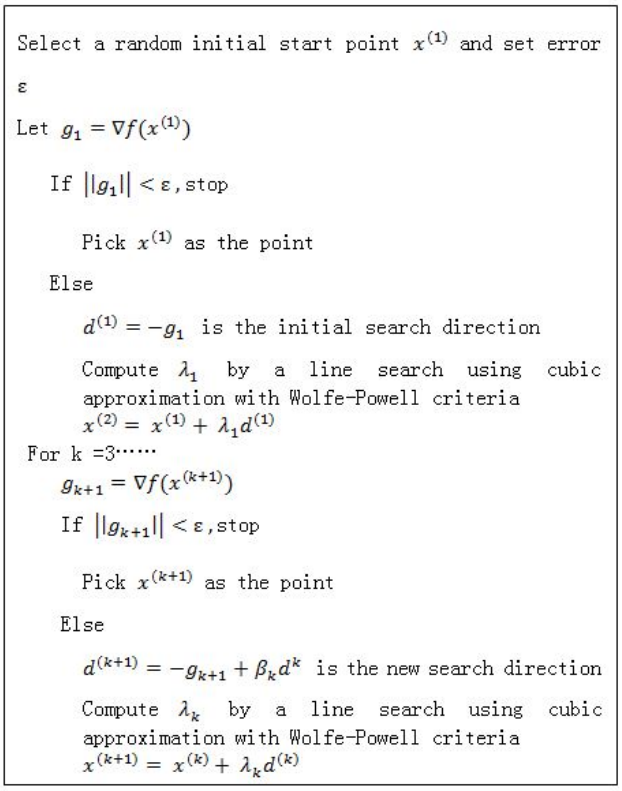
- Linearity:
  the $k_{th}$ search direction $d_k$ is a linear combination of the previous direction and the gradient .
- Conjugation:
  all the search direction are conjugated with some positive definitive matrix $A$.

The first assumption allow us to have the follow equation:

$$d^k = -g^k + \beta_{k-1} d^{k-1} \tag{3.26}$$

where$g^{k+1}$ is the gradient of the current point and $\beta_k$ is some coefficients that need to be calculated. The second assumption is actually used to allow us calculating the $\beta_k$

The process of Conjugate Gradient (CG) method could be described as Fig.3.5



```
Select a random initial start point $x^{(1)}$ and set error
$\varepsilon$
Let $g_1 = \nabla f(x^{(1)})$
   If $||g_1|| < \varepsilon$, stop
      Pick $x^{(1)}$ as the point
   Else
      $d^{(1)} = -g_1$ is the initial search direction
      Compute $\lambda_1$ by a line search using cubic
      approximation with Wolfe-Powell criteria
      $x^{(2)} = x^{(1)} + \lambda_1 d^{(1)}$
 For k =3……
    $g_{k+1} = \nabla f(x^{(k+1)})$
    If $||g_{k+1}|| < \varepsilon$, stop
      Pick $x^{(k+1)}$ as the point
    Else
      $d^{(k+1)} = -g_{k+1} + \beta_k d^k$ is the new search direction
      Compute $\lambda_k$ by a line search using cubic
      approximation with Wolfe-Powell criteria
      $x^{(k+1)} = x^{(k)} + \lambda_k d^{(k)}$
```

**Figure 3.5.** Conjugate Gradient algorithm

From the algorithm above, we could see the CG method firstly finds a descent direction along which the objective function 3.25 will be reduced and then computes a step size that determines how far should we move along the direction, aiming at finding a local minimum of the objective function.

Therefore, the problem could approximately be divided into two parts: Searching the descending direction and determining the step size iteratively. To be more specific, find the proper value for parameters $\beta_k$ and $\lambda_k$ respectively.

When it comes to the search direction, due to the second assumption of CG method, we could know that

$$\beta_j = \begin{cases} = 0 & j = 0....k-1 \\ \neq 0 & j = k \end{cases}$$

Therefore, we are able to compute the $\beta_k$ by some methods. In fact, we have many different ways to calculate the parameters $\beta_k$ and each of them based on different principles. Here, we implement the Polak-Ribiere method[22] that compute the parameter $\beta_k$ as 3.27.

$$\beta_k = \frac{g_{k+1}^T (g_{k+1} - g_k)}{g_k^T g_k} \tag{3.27}$$

After getting the new search direction $d^{(k+1)}$, the problem comes to define the step size $\lambda_k$. Here, we use cubic approximation as 3.28 to compute the new step size.

$$\begin{aligned} \lambda_k &= \lambda_{k-2} - \frac{(-d^{(k-2)}d^{(k-2)}) * (\lambda_{k-1} - \lambda_{k-2})^2}{(B + \sqrt{(B * B - A * (-d^{(k-2)}d^{(k-2)})) * (\lambda_{k-1} - \lambda_{k-2})})} \\ A &= 6 * (f(x_{k-2}) - f(x_{k-1})) - 3 * (-d^{(k-1)}d^{(k-1)} + (-d^{(k-2)}d^{(k-2)})) * (\lambda_{k-1} - \lambda_{k-2}) \\ B &= 3 * (f(x_{k-1}) - f(x_{k-2})) - (2 * (-d^{(k-2)}d^{(k-2)}) + (-d^{(k-1)}d^{(k-1)})) * (\lambda_{k-1} - \lambda_{k-2}) \end{aligned} \tag{3.28}$$

We will then calculate the objective function value $f(x_k + \lambda_k d_k)$ and new slope $g_{k+1}^T d_k$ based on this new step size $\lambda_k$ and judge it by using the Wolfe-Powell conditions as 3.29

$$\begin{aligned} f(x_k + \lambda_k d_k) &\leq f(x_1) + \lambda_1 \rho g_1^T d_1 \\ g_{k+1}^T d_k &\geq \sigma g_1^T d_1 \qquad \sigma \in (\rho, 1), \quad \rho \in (0, \frac{1}{2}) \end{aligned} \tag{3.29}$$

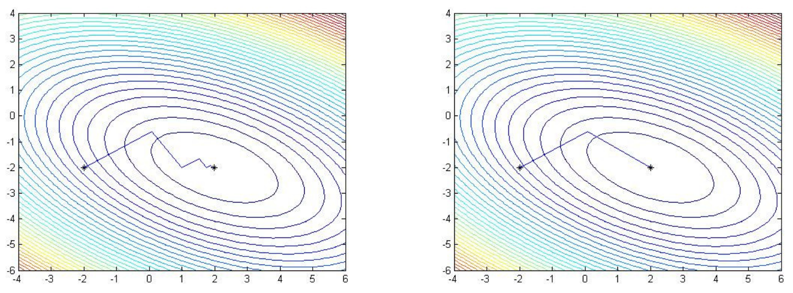
If the current point objective function value and slope fulfill the Wolfe-Powell conditions, we could claim we find the proper step size that is calculated by the cubic extrapolation and stop searching.

By doing these iteratively, we could minimize the objective function and eventually have the set of hyperparameters that gives the minimum value of the objective function or fulfills the required accuracy.

The CG method is based on the gradient descent and Newton method. It only uses the information provided by the first order derivative which means it does not need any further information and could be easily implemented in many aspects.

This property also implies that it does not need to compute and store the Hessian Matrix as well as its inverse. As a result, it could reduce the computational cost significantly compared with the Newton method.

What's more, since the traditional gradient descent method will slow down the searching rate dramatically when it is getting closer to the minimal value, the CG manage to overcome this drawback of the gradient descent by using the conjugation which could speed up the convergence rate. The following plots show the convergence rate of two different methods



(a) Gradient Descent (b) Conjugate Gradient

**Figure 3.6.** Comparison of convergence rate

### 3.3.2 The Particle Swarm Optimization

The Particle Swarm Optimization (PSO) method is one of the new iterative methods for problem optimization. It improves the solution by iteratively searching for the best result in current circumstance.

Initially, we have a branch of separated particles that represent the potential solution by their position, a group of the hyperparameters in our case. And for each parameter in the one particle, we assign different velocity for them. In each iteration, the particle updates its position simply by its velocity of each parameter. Then we use these parameter to calculate the model and evaluate the result by the objective function we mentioned before. After that, we record the best position for each particle(local optimal value) as well as the best position in the whole search space(global optimal value).

The smart part of this method is the update of the velocity of each particle. The individual velocity for each particle is influenced not only by its local best known position (local optimal value) but also the best known positions in the search-space (global optimal value), which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solution.

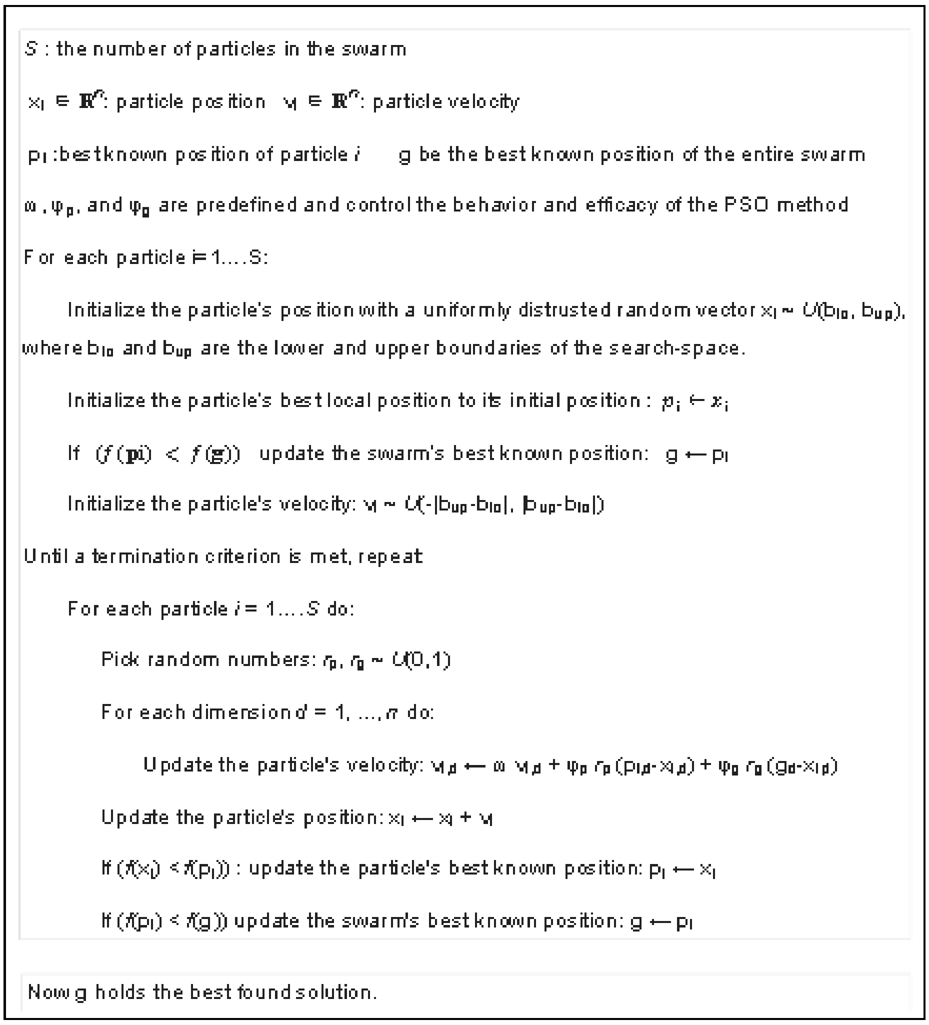
In the end, the iteration will be terminated by some criterion, such as the maximal number of iterations or a solution that gives the adequate loss function value. The following Fig. 3.7 shows the basic algorithm of Particle Swarm Optimization

PSO is a heuristic search algorithm based on swarm intelligence since it makes no assumptions about the problem being optimized and can search very large spaces. Therefore, PSO can also be used on optimization problems that are partially irregular, noisy, change over time, etc. What's more, since it uses swarm intelligence which means each of the particle is related to the others and the final result is depended on the contribution of all particles, the robustness of system is strong and will not be be dramatically influenced by some outliers. In addition, its strengths in terms of convergence, global search capacity and practicability also make it one of the most popular optimization algorithm in the world.

## 3.4 Summarize of the Chapter

After optimizing the hyperparameters, the GP regression model is ready to predict the peptide's retention time. The general process of this chapter could be illustrated as the fig.3.8 and the performance of the GP model will be showed in the next chapter.

## 3.4. SUMMARIZE OF THE CHAPTER



```
$S$ : the number of particles in the swarm
$x_i \in \mathbb{R}^n$: particle position   $v_i \in \mathbb{R}^n$: particle velocity
$p_i$ :best known position of particle $i$      g be the best known position of the entire swarm
$\omega$ , $\varphi_p$, and $\varphi_g$ are predefined and control the behavior and efficacy of the PSO method
For each particle i=1....S:
    Initialize the particle's position with a uniformly distrusted random vector $x_i \sim U(b_{lo}, b_{up})$,
where $b_{lo}$ and $b_{up}$ are the lower and upper boundaries of the search-space.
    Initialize the particle's best local position to its initial position : $p_i \leftarrow x_i$
    If  $(f(\mathbf{pi}) < f(\mathbf{g}))$  update the swarm's best known position:  $g \leftarrow p_i$
    Initialize the particle's velocity: $v_i \sim U(-|b_{up}-b_{lo}|, |b_{up}-b_{lo}|)$
Until a termination criterion is met, repeat
    For each particle $i$ = 1....$S$ do:
        Pick random numbers: $r_p$, $r_g \sim U(0,1)$
        For each dimension $d$ = 1, ..., $n$ do:
            Update the particle's velocity: $v_{i,d} \leftarrow \omega\ v_{i,d} + \varphi_p\ r_p\ (p_{i,d}-x_{i,d}) + \varphi_g\ r_g\ (g_d-x_{i,d})$
        Update the particle's position: $x_i \leftarrow x_i + v_i$
        If $(f(x_i) < f(p_i))$ : update the particle's best known position: $p_i \leftarrow x_i$
        If $(f(p_i) < f(g))$ update the swarm's best known position: $g \leftarrow p_i$
Now g holds the best found solution.
```

**Figure 3.7.** Particle Swarm Optimization

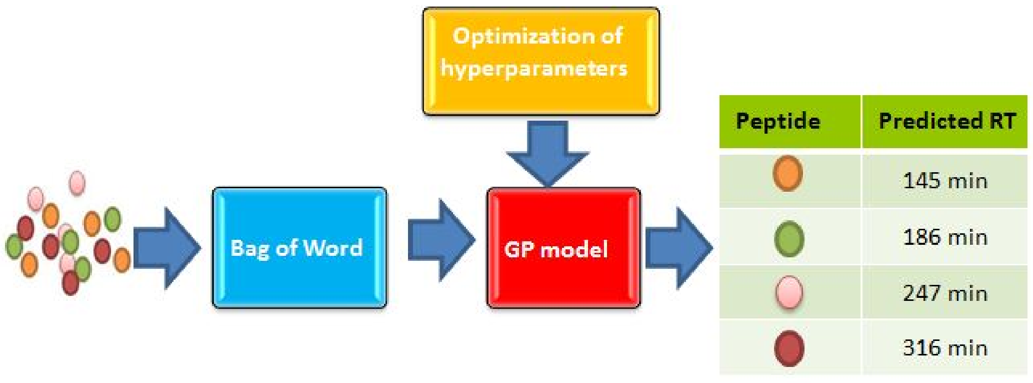


**Figure 3.8.** The general process of predicting peptide's retention time

Chapter 4

# Results and Evaluation

In this section, we report the experiments in order to compare the GP regression model with other prediction models and discuss their performance. We are interested in whether the GP model leads to a better predictive performance than the other models as well as its strengths and weakness compared with other models.

To get a better impression of the performance, we will firstly investigate the importance of kernel functions. To be specific, we will compare the GP model with single kernel to the one with multiple kernels. Further, we will investigate the importance of optimization. In another word, we will compare the results before and after optimization for GP model. After this comparison, we will evaluate the performance of different optimization methods, PSO and CG method specifically. All these models will be tested in two different set of features respectively so that we could make a comparison between different kind of features. The first kind of feature is the feature created by Bag of Word method: we simply call it $BOWfeature$, and the second one is the features used for prediction in the state-of-art model Elude: we call it $Eludefeature$.[8]

In the end of this part, we will make a comparison between different kinds of state-of-art retention time predictors including Elude, BioLCCC (an on-line retention time predictor)[27],SSRC and our GP model. These models will be tested by the peptide we use for the GP model and evaluated by the same criteria as show below.

Before introducing the experimental setup, we will first introduce some main concepts which we will use for the result evaluation later: Pearson Correlation Coefficient and minimal time window.

- Pearson Correlation Coefficient:
  In statistic theory, Pearson correlation coefficient is used to identify the linear correlation between two variables $X$ and $Y$. The formula is as follow:

  $$\rho = Cor(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}} \tag{4.1}$$

  The value of Pearson Correlation Coefficient is between -1 and 1. A value of 1 (or -1)implies that a linear equation describes the relationship between X and Y perfectly, the sign implies whether the relationship is positive or negative. A value of 0 implies that there is no linear correlation between the variables. The closer the value is to 1, the stronger the linear relation they have, otherwise, the weaker the relation they have.

- Minimal Time Window:
  The minimal time window is a time window including the deviations between observed and predicted retention times for 95% of the peptides ($\Delta t 95\%$). Given the time window, we could know the expected range of observing time in advance and thus reduce the number of peptide we need to compare and enhance matching accuracy of the peptides. In this case, the minimal time window will be showed as ratio of total time instead of a absolute value. The total time is defined as the time interval between the smallest observed retention time and largest one. The smaller the time window is, the more peptide we could generate at the same time.

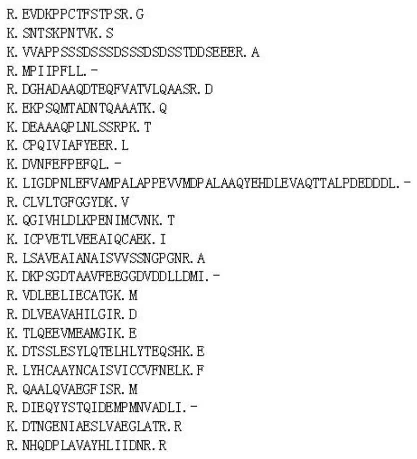
- Computational Cost:
  The computational cost is a parameter reflects the efficiency of a model. A model with too large computational cost should be considered as a bad model since it is hard to implement and reproduce. The computational cost in this thesis stands for the total running time from importing raw peptide data to getting the predicted retention time of the peptide.

In this thesis, the top priority is given to minimize the minimal time window since we want to reduce the number of peptide we need to match and improve the parallelized processing capacity.

## 4.1 Experimental Setup

The experiment is tested on a 64-bit Ubuntu computer and run by Matlab R2014a. To build the Gaussian Process model, we need to install the software package called GPML which is written by Carl Edward Rasmussen and Hannes Nickisch.[21] In addition, to benchmark the GP model, we also need the SVR software packages libsvm [23].

The experimental data we use in this thesis is a set of 15933 peptides generated from the laboratory as 4.1. We separate them into training set and testing set respectively. The training set contains 12747 peptides while the testing set has 3186 peptides.



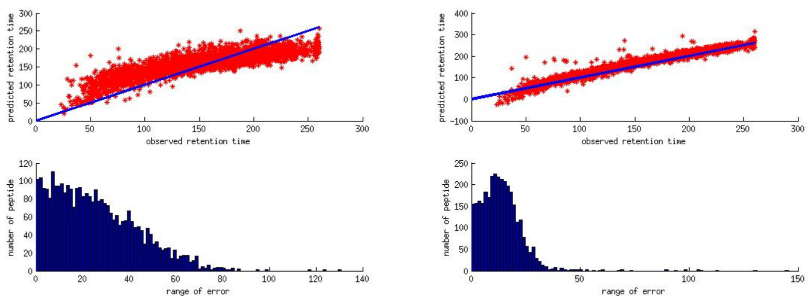
```
R.EVDKPPCTFSTPSR.G
K.SNTSKPNTVK.S
K.VVAPPSSSDSSSDSSSDSDSSTDDSEEER.A
R.MPIIPFLL.-
R.DGHADAAQDTEQFVATVLQAASR.D
K.EKPSQMTADNTQAAATK.Q
K.DEAAAQPLNLSSRPK.T
K.CPQIVIAFYEER.L
K.DVNFEFPEFQL.-
K.LIGDPNLEFVAMPALAPPEVVMDPALAAQYEHDLEVAQTTALPDEDDDL.-
R.CLVLTGFGGYDK.V
K.QGIVHLDLKPENIMCVNK.T
K.ICPVETLVEEAIQCAEK.I
R.LSAVEAIANAISVVSSNGPGNR.A
K.DKPSGDTAAVFEEGGDVDDLLDMI.-
R.VDLEELIECATGK.M
R.DLVEAVAHILGIR.D
K.TLQEEVMEAMGIK.E
K.DTSSLESYLQTELHLYTEQSHK.E
R.LYHCAAYNCAISVICCVFNELK.F
R.QAALQVAEGFISR.M
R.DIEQYYSTQIDEMPMNVADLI.-
K.DTNGENIAESLVAEGLATR.R
R.NHQDPLAVAYHLIIDNR.R
```

**Figure 4.1.** a subset of experimental peptide sequence

## 4.2 Experimental Results

The Gaussian Processes model is firstly built by $BOW feature$. The Fig.4.2 shows the performance of the GP model based on different kernel function. The left one is simply using RBF kernel while the right one is using a multiple kernel that consist of RBF and Poly kernel function. The blue line in the plot represents the ideal result of the correlation coefficient and the red one is the true results from the experiment.

For each subplot, the top plot shows the correlation between the predicted retention time and the real retention time and tells how linear they are related to each other.

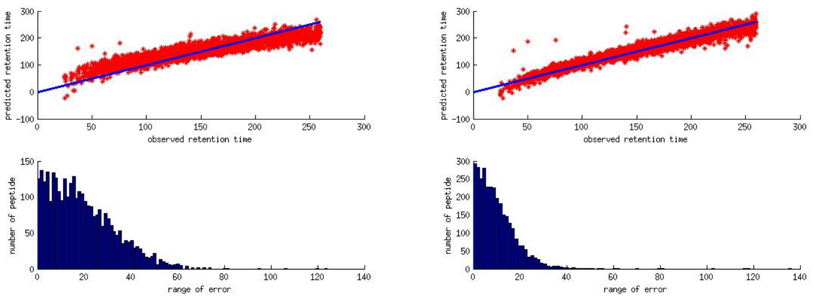


(a) Bag of Word feature with RBF kernel (b) Bag of Word feature with RBF+Poly kernel

**Figure 4.2.** Comparison of performance based on different kernels

The histogram in the bottom shows the the distribution of the deviation between the predicted retention time and the observed one.

As benchmark, we also test the model by using $Elude feature$. The fig.4.3 gives us the performance of the model based on this feature set.



(a) Elude feature with RBF kernel (b) Elude feature with RBF+Poly kernel

**Figure 4.3.** Comparison of performance based on different kernel with Luminita's feature

The Table.4.1 shows detail information in terms of the Pearson Correlation Coefficient and the minimal time window of them. The left number is the Person Coefficient and the right one is the corresponding minimal time window.

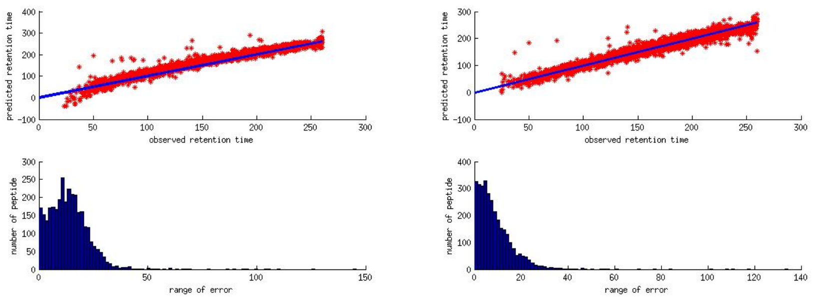
As we can see from the figures, the multi-kernel gives better performance than a single kernel in general. The multi-kernel function helps GP model to improve the correlation coefficient in both cases. The scatter plots in both cases become narrower and closer to the ideal blue line.

| | RBF kernel | | RBF + Poly kernel | |
|---|---|---|---|---|
| | $\rho$ | $\Delta_{t95\%}$ | $\rho$ | $\Delta_{t95\%}$ |
| BOW feature | 0.8506 | 0.5080 | 0.9674 | 0.2579 |
| Elude feature | 0.9213 | 0.3912 | 0.9737 | 0.2469 |

**Table 4.1.** Comparison between Single kernel and multi-kernel

More importantly, it also reduces the minimal time window dramatically, from 0.5080 to 0.2579 and 0.3912 to 0.2052 for $BOWfeature$ and $Eludefeature$ respectively. This also implies that the multi-kernel makes the prediction more precisely and narrow down the deviations between observed and predicted retention times.

After comparing the Gaussian Process with single kernel and multi-kernel, we will now investigate the effect of optimization. In this case, we only perform Particle Swarm Optimization method as an example. The following plots show the results



(a) BOW feature with GP model (b) Elude feature with GP model

**Figure 4.4.** Performance of the feature based on Gaussian Process model after PSO optimization

| | | Before Optimization | | After Optimization | |
|---|---|---|---|---|---|
| GP | BOW | 0.9674 | 0.2579 | 0.9740 | 0.2379 |
| | Elude | 0.9737 | 0.2469 | 0.9735 | 0.2065 |

**Table 4.2.** Comparison of performance before and after PSO for GP

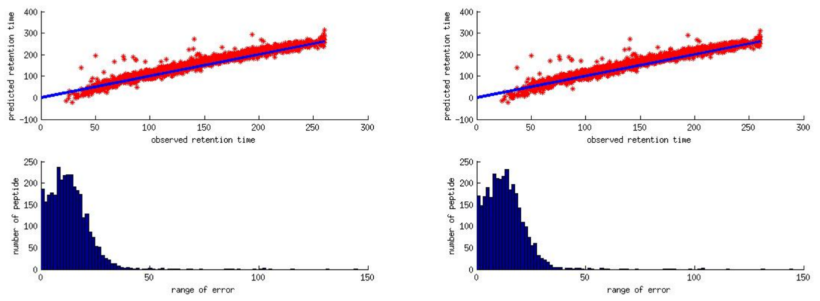
From Table.4.2, we could tell that the PSO optimization is useful and have improved the prediction result. Although the difference is only $0.04$, it is actually about 4 min of the deviations between observed and predicted retention times.

However, we could not deny that the optimization method in this case does not make a significant difference to the prediction results. The potential explanations for this phenomenon could be as follows:

First of all, the initial parameters for the model is too good and the better solution that the optimization method have found could not make a significant improvements. Secondly, the PSO optimization method itself is not appropriate and could not find some much better solution. Thirdly, it has not found the optimal combination for the hyperparameters this time due to the fact that the PSO method does not guarantee to find the optimal solution every time.

Among all these probabilities, the second and the third explanations are probably the reasons for this phenomenon. To investigate more about the reasons, we will now try to compare the PSO method with another optimization method, Conjugate Gradient method, to see if our explanations are correct.

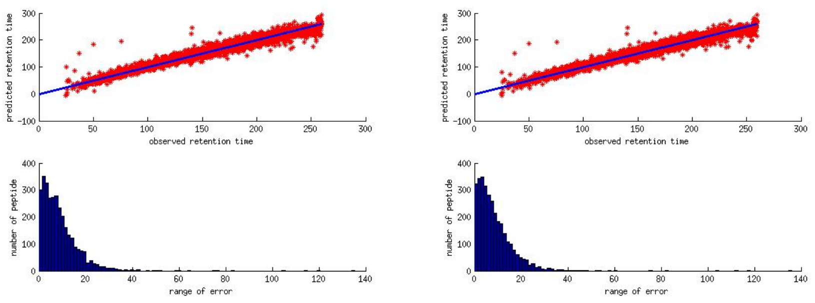
The following plots show the experimental results of different optimization methods.



(a) BOW feature with PSO (b) BOW feature with CG

**Figure 4.5.** Performance of the BOW feature based on different optimization method

As we can see from fig.4.5 and fig.4.6 , these two methods have similar performance in this case. However, when we look at the details of the comparison of these two optimization in Table.4.3, we could see that the CG method actually gives better performance than the PSO method. If we combine this with the Table.4.2, we could tell that the CG method is more appropriate than PSO method in this



(a) Elude feature with PSO (b) Elude feature with CG

**Figure 4.6.** Performance of the Elude feature based on different optimization method

case. It reduces the $\Delta_{t95\%}$ of $Eludefeature$ from $0.2469$ to $0.2011$ and the one of $BOWfeature$ from $0.2579$ to $0.2248$. Another interesting thing to see is that the result of PSO method in two tables are different and the result in Table.4.3 is better than table.4.2. This is exactly what we have guessed that the PSO does not guarantee to find the optimal solution every time.

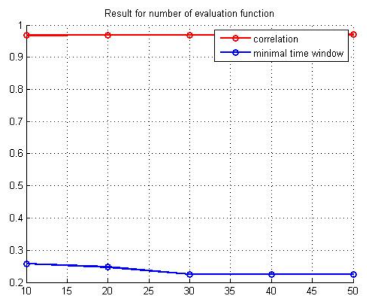
| | Particle Swarm Optimization | | Conjugate Gradient | |
|---|---|---|---|---|
| | $\rho$ | $\Delta_{t95\%}$ | $\rho$ | $\Delta_{t95\%}$ |
| BOW | 0.9740 | 0.2329 | 0.9760 | 0.2248 |
| Elude | 0.9741 | 0.2061 | 0.9762 | 0.2011 |

**Table 4.3.** Comparison of performance of GP based on PSO and CG optimization

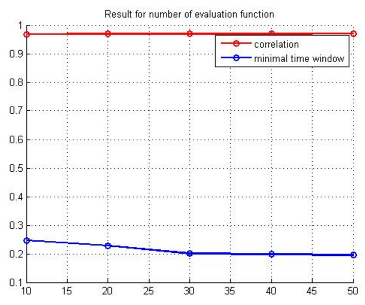
About the optimization methods, apart from having a impression of the general performance, we would also like to investigate the influence of some parameters for the prediction, such as the number of evaluating function, the number of generation, the number of particles and the changing trend of each particle during the evolution.

The fig.4.7 show the effect of different number of evaluating function (iteration) in CG method based on different features. As we can see, increasing the number of evaluating function results in better performance in terms of correlation coefficient and minimal time window. However, when the number of evaluating function reaches to the level of 30 and above, the difference becomes small and can be ignored. These plots implies that we could set the number of evaluating function around 30 and no need to worry about not getting much better prediction result.

Following this, we will now try to find out the influence of the number of genera-



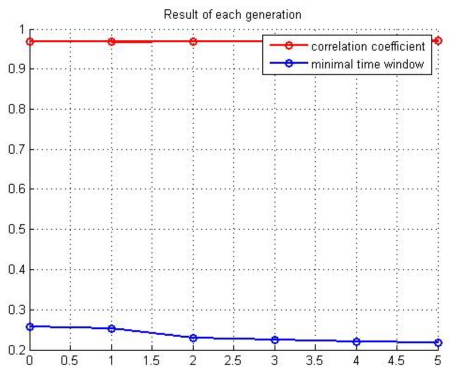
(a) BOW feature with CG



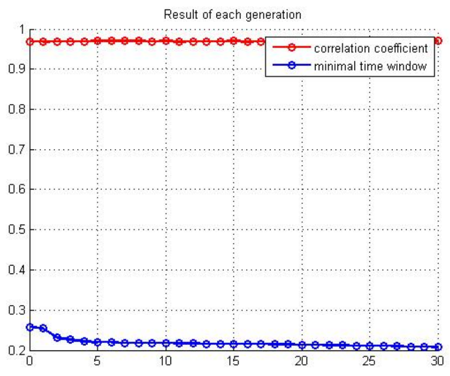
(b) Elude feature with CG

**Figure 4.7.** Comparison of different number of evaluating functions

tion in PSO methods.



(a) BOW feature with 5 generation



(b) BOW feature with 30 generation

**Figure 4.8.** Comparison of different number of generation

The fig.4.8 shows that the number of generation actually have slightly influence on the prediction. The top one traces the changing trend of correlation coefficient and minimal time window during 5 generation and in the bottom one, we increase the number of generation to 30 to make a comparison. Each dot in the plots represent the prediction result of that generation.

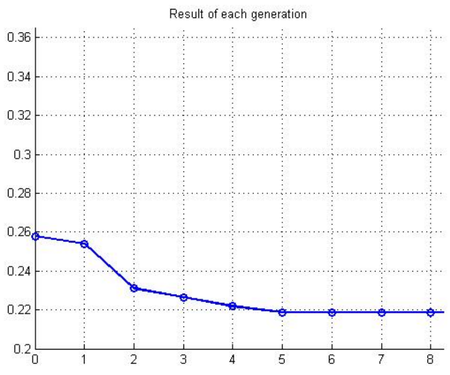


**Figure 4.9.** Detail changes of 30 generation

The detail could be seen from fig.4.9. The optimization has reduced the minimal time window from around $26\%$ to around $23\%$ in the second generation of evolution. Due to the fact that the prediction after the first few generation is already really good, the impact of increasing the number of generation seems to be ignorable. The Table.4.4 shows the detail information based on different number of generation.

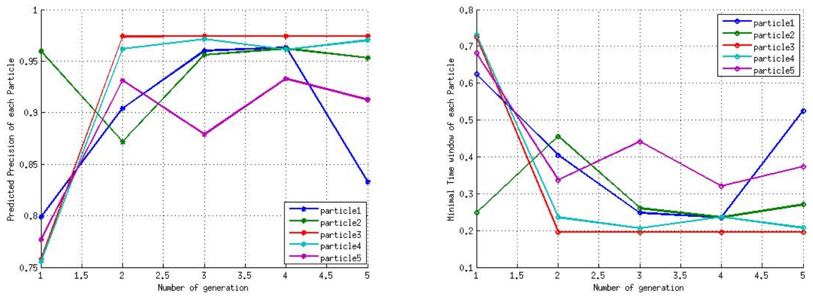
Another problem comes along with increasing the number of the generation is that the computational cost becomes increasingly large. For example, the computational cost of a GP model with $Eludefeature$ using 5 generation PSO method is $2687sec$ while the one using 30 generation PSO is $15273sec$ which is 5.7 times as much as before, but the improvement of $\Delta_{t95\%}$ is just slightly. The situation becomes even worse when the number of the features increase. Therefore, we could simply set the default number of generation around 5 if we want to limit the computation cost.

Now, we will take a look at the effect of different number of particles and each particle's evolution trend in terms of correlation coefficient and minimal time window during each generation.

As the Fig.4.10(a) and Fig.4.10(b) showed, the trend of all the particles in both

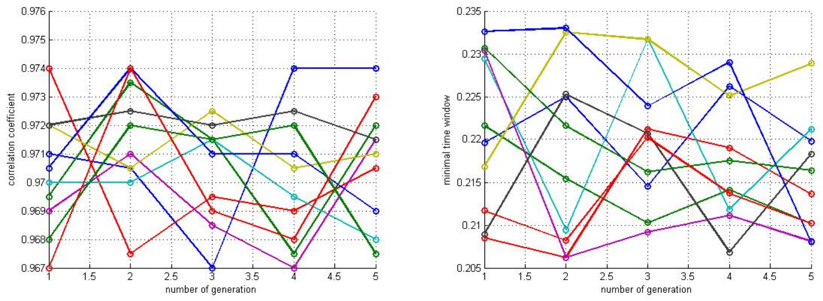
| | 5 generation | | 30 generation | |
|---|---|---|---|---|
| | $\rho$ | $\Delta_{t95\%}$ | $\rho$ | $\Delta_{t95\%}$ |
| BOW | 0.9740 | 0.2329 | 0.9760 | 0.2105 |
| Elude | 0.9741 | 0.2076 | 0.9743 | 0.1995 |

**Table 4.4.** Best performance of GP model based on PSO with different number of generation



(a) Trend of correlation coefficient of each particle (b) Trend of minimal time window of each particle

**Figure 4.10.** The evolution trend of each particle during 5 generations with 5 particles



(a) Trend of correlation coefficient of each particle (b) Trend of minimal time window of each particle

**Figure 4.11.** The evolution trend of each particle during 5 generations with 10 particles

cases are going towards the better situation and converging gradually. In addition, we could also see that all particles have influenced each other's trend which totally follows the principle of PSO method. As comparison, we also display the results of using 10 particles instead of 5 particles. These plots also show the same properties as before when seeing the main trend from a bigger picture. Besides, it can be seen from Table.4.5 that increasing the number of particles do not contribute much to the improvement of both accuracy and minimal time window. One interesting thing is that we could also see the trend becomes more flat and stable after the third generation in 4.10. This is also support evidence for the conclusion we draw form Fig.4.9 about the proper number of generation.

| | 5 particles of PSO | | 10 particles of PSO | |
|---|---|---|---|---|
| | $\rho$ | $\Delta_{t95\%}$ | $\rho$ | $\Delta_{t95\%}$ |
| BOW | 0.9740 | 0.2329 | 0.9740 | 0.2283 |
| Elude | 0.9741 | 0.2076 | 0.9748 | 0.2060 |

**Table 4.5.** Comparison of performance based on different number of particles

In the end, as benchmark, we will summarize the performance of other popular retention time prediction models and compare them with our GP model in terms of correlation coefficient $\rho$ , minimal time window($\Delta_{t95\%}$) and computational cost. The Table.4.6 shows the result of different models.

| | GP Model | Elude Model | SSRC | BioLCCC |
|---|---|---|---|---|
| Pearson Correlation Coefficient($\rho$) | 0.9743 | 0.9707 | 0.8995 | 0.8849 |
| Minimal Time Window($\Delta_{t95\%}$) | 0.1950 | 0.2172 | 0.4205 | 0.5841 |
| Computational Cost | 3570 sec | 1 day + | | |

**Table 4.6.** Comparison of performance based on different number of particles

The last two columns are empty because we do not know how long it takes to train the model for prediction.

As we can seen form Table.4.6, our GP model gives much better performance than SSRC and BioLCCC. Our model successfully reduce the $\Delta_{t95\%}$ to a level of $0.1950$ while the BioLCCC only gives $0.5841$ and result of SSRC is $0.4205$. That means our model reduces almost 20 mins of the time window which which reduce the computational cost and enhance the identification accuracy dramatically. In addition, the $\rho$ of our GP model is also higher than the SSRC and BioLCCC which means our prediction is more precisely than the other two methods.

What's more, our GP model gives better performance than Elude which is considered as the most popular and best model so far. Although the difference between Elude and our model are not so significant in terms of correlation coefficient and minimal time window, the computational cost of Elude is much higher than our GP model. In big data analysis, such high computational cost could be a very serious problem and we should make a balance between the accuracy and the efficiency.

Considering that the computational cost of Elude is almost 24 times as our GP model's while the predicted results are similar, the practical ability of our model is much stronger than Elude.

# Chapter 5

# Discussion and Conclusion

## 5.1 Limitation and Discussion

As we discussed in the previous chapter, GP outperforms most other models. However, there are still some limitations on this model that needs to be improved. Solving these limitations can have a significant impact on the accuracy of our method.

### 5.1.1 Model limitation

One of the main limitation is the computational cost of Gaussian Process model. Although the computational cost of our GP model is better than Elude, it is still a main problem due to the fact that the Gaussian Process Regression model is a non-parametric model. For training part, the hyperparameters are obtained by optimizing the likelihood function. Therefore, we need to compute the inverse of covariance function $K_n + \sigma_n^2 I_n$ which makes the computational cost of the model to a level of $O(n^3 * p)$ where $p$ is the times of gradient computation. As for the prediction part, the computational cost for each data is $O(n^2)$. When dealing a large amount of data, this drawback could become more serious and do harm to the implementation.

Another limitation of the GP model is the assumption of the *Gaussian Noise* which means the noise needs to fulfill the Gaussian distribution. This assumption makes the GP model easier when calculating the matrix. However, the noise is actually not a *Gaussian Noise* in some cases and this assumption will therefore influence the prediction and leads to worse results.

### 5.1.2 Feature limitation

As for the features, the main limitation is the lack of information for our features. This means the *BOW features* do not have significant biological meaning. This limitation is reflected by the fact that the *Elude features* performs better than the *BOW features* in general.

However, on one hand, providing features with more information means the prediction are more likely to be precisely. On the other hand, compared to *BOW features*, the *Elude features* requires more time to be calculated. More importantly, the performance of these two features is actually not very different. This means that the features that are suitable for this task are still unknown and finding those features will require further investigation.

### 5.1.3 Optimization limitation

Like most evolutionary algorithms, PSO usually requires a large number of fitness evaluations to obtain a sufficiently good solution. This poses an obstacle for applying PSO to computationally expensive problems.

Besides, as all other heuristic algorithms, PSO do not guarantee an optimal solution is ever found. In fact, the optimized result could be slightly different from time to time but always has the same trend. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-newton methods. As a result, it might find the local optimal value instead of the optimal value in the whole search space and consider it as the "global optimal value" due to the its convergence.

To overcome the shortages of the PSO method, a number of different methods based on the original PSO have been carried out, such as Gray-PSO, Chaos-PSO. Each of these methods has combined PSO with some other methods so that it could overcome the problem of the original PSO and have their own advantages when dealing with some specific tasks.

As for the CG method, one of the most limitation of the CG method is its dependency on the initial point which would have a great influence on the final result.

Due to this limitation, the CG method could not ensure to converge to the minimum value in the whole search space.

To overcome this, there are many variations of CG. They could improve the performance of the original CG method when dealing with some specific tasks but they still have their drawbacks as well.

## 5.2 Conclusion

In this thesis, GP model has been proposed for the prediction of peptides' retention time and tested in different datasets. The importance of kernel functions has been investigated and comparison has been made between single kernel function and multi-kernel function. The multi-kernel function combines the strengths of two kernel functions and improves the accuracy of the prediction results significantly.

Secondly, a new method of feature selection has been tested in this case. The feasibility of these features has been proved and their performance is almost as good as the biochemical features. Although the performance of these less-biochemical features are still slightly worse than the biochemical features due to the fact of lacking information, the efficiency of generating them and the flexibility of these features still imply the unique and powerful advantages.

In addition, two optimization methods have been implemented and compared with each other. Although, we only utilize the original version of them, they still make a significant impact on the experimental results.

Finally, the GP model was also compared with the other state-of-art prediction methods. The GP model has much better performance than most of them in terms of the minimal time window ($\Delta_{t95\%}$) and correlation coefficient ($\rho$). Even compared with the best predicted model Elude, the predicted result of our model is still a bit better.

This implies a stronger and reliable processing capacity of our model when compared with the other models. More importantly, the computational efficiency of our model is much higher than Elude model which is a significant quality since a lot of biochemical experiments contain large amount of data that need to be processed.

## 5.3 Future Work

As we have analyzed before, one of the most important limitations of GP model is the computational cost. It is quite hard to overcome this problem since the Gaussian Process is non-parametric model. In the last 20 years, many new methods have been proposed to solve this problem. Tresp in [25] proposes a Bayesian committee machine(BCM) method and Snelson in [26] suggest a method based on sparse Gaussian Process using pseudo-input to enhance the computational efficiency. Therefore, one potential work in the future is to replace the original Gaussian Process with these new methods to investigate their performance. What's more, one could even implement some new machine learning methods, such as deep learning to address this problem and test their performance.

Another potential direction to go on is the feature selection method. Currently, we only test two different kinds of features separately and both of them give similar performance. So a alternative way to continue is try to combine different types of feature together as new multi-features and investigate their performance.

One more interesting thing is about finding some more proper optimization method to overcome the shortages of current ones. Since we know that the original CG method has the weakness in terms of its dependency of the initial value and the Particle Swarm Optimization could not ensure the convergence to the minimal value of the entire search space, we could try to investigate some new optimization methods which could overcome these drawbacks.

However, the theory of Gaussian Process is not complete and there are still a lot of aspects need to be fixed and improved. It has been proved to be a powerful and useful tool but the application of it in biotechnological field is still not as common as it is in the others. Thus, we strongly believe that with the further understanding of Bayesian theory and statistical learning theory and the rapidly development of computer techniques, more sophisticated and practical Gaussian Process Regression models will be created and they will continually broaden their application fields in biotechnology.

Further, we should also see that the prediction of peptide's retention time is just another example of implementation of machine learning method in biochemical problem. From a long-term perspective, with the help of computer science and machine learning theories, more and more advanced machine learning methods will be utilized to solve biochemical problems and with the development of biotechnology, more and more biochemical problems will arise which will also motivate the development of machine learning methods. The connection between machine learning techniques and biotechnology will become increasing strong in the further.

# Bibliography

[1] Mathias Wilhelm, Judith Schlegl, Hannes Hahne, Amin Moghaddas Gholami,Marcus Lieberenz, Mikhail M Savitski, Emanuel Ziegler, Lars Butzmann,Siegfried Gessulat, Harald Marx, Toby Mathieson, Simone Lemeer, Karsten Schnatbaum, Ulf Reimer, Holger Wenschuh, Martin Mollenhauer, Julia Slotta-Huspenina, Joos-Hendrik Boese, Marcus Bantscheff, Anja Gerstmair, Franz Faerber, and Bernhard Kuster. Mass-spectrometry-based draft of the human proteome. Nature, 509(7502):582-587, April 2015.

[2] Christine C Wu and Michael J MacCoss. Shotgun proteomics: tools for the analysis of complex biological systems. Curr Opin Mol Ther, 4(3):242-250,2002.

[3] Sparkman, O. David (2000). Mass spectrometry desk reference. Pittsburgh: Global View Pub. ISBN 0-9660813-2-3.

[4] Baczek T, Wiczling P, Marszall M, et al. (2005). "Prediction of peptide retention at different HPLC conditions from multiple linear regression models." Journal of Proteome Research, 4(2):555¨C63

[5] Petritis K, Kangas LJ, Ferguson PL, et al. (2003). "Use of artificial neural networks for the accurate prediction of peptide liquid chromatography elution times in proteome analyses." Analytical Chemistry, 75(5):1039¨C48

[6] Petritis K, Kangas LJ, Yan B, et al. (2006). "Improved peptide elution time prediction for reversed-phase liquid chromatography-MS by incorporating peptide sequence information." Analytical Chemistry, 78(14):5026¨C39.

[7] Klammer AA, Yi X, MacCoss MJ, et al. (2007). "Improving tandem mass spectrum identification using peptide retention time prediction across diverse chromatography conditions." Analytical Chemistry, 79(16):6111¨C8

[8] Luminita Moruz, Daniela Tomazela, Lukas Kall,(2010). "Training, Selection, and Robust Calibration of Retention Time Models for Targeted Proteomics"

[9] Luminita Moruz,(2013). "Chromatographic retention time prediction and its applications in mass spectrometry-based proteomics".

[10] Mant, C. T., and Hodges, R. S. (2002). "Analytical HPLC of peptides, in HPLC of Biological Macromolecules" (Gooding, K. M., and Regnier, F. E., eds) pp. 433¨C511, Marcel Dekker, New York.

[11] Meek JL (1980). "Prediction of peptide retention times in high-pressure liquid chromatography on the basis of amino acid composition." Proceedings of the National Academy of Sciences of the United States of America, 77(3):1632¨C6.

[12] Houghten RA and DeGraw ST (1987). "Effect of positional environmental domains on the variation of high-performance liquid chromatographic peptide retention coefficients." Journal of Chromatography, 386:223¨C8.

[13] Browne CA, Bennett HP, and Solomon S (1982). "The isolation of peptides by high-performance liquid chromatography using predicted elution positions." Analytical Biochemistry, 124(1):201¨C8.

[14] Zhou NE, Mant CT, and Hodges RS (1990). "Effect of preferred binding domains on peptide retention behavior in reversed-phase chromatography: amphipathic alpha-helices." Peptide Research, 3(1):8¨C20.

[15] Mant CT and Hodges RS (2006). "Context-dependent effects on the hydrophilicity/hydrophobicity of side-chains during reversed-phase high-performance liquid chromatography: implications for prediction of peptide retention behaviour." Journal of Chromatography A, 1125(2):211¨C9.

[16] Guo DC, Mant CT, and Hodges RS (1987). "Effects of ion-pairing reagents on the prediction of peptide retention in reversed-phase high-performance liquid chromatography." Journal of Chromatography,386:205¨C22.

[17] Gilar M, Xie H, and Jaworski A (2010). "Utility of retention prediction model for investigation of peptide separation selectivity in reversed-phase liquid chromatography: impact of concentration of trifluoroacetic acid, column temperature, gradient slope and type of stationary phase." Analytical Chemistry, 82(1):265¨C75.

[18] Krokhin OV, Craig R, Spicer V, et al. (2004). "An improved model for prediction of retention times of tryptic peptides in ion pair reversed-phase HPLC:its application to protein peptide mapping by off-line HPLC-MALDI MS." Molecular Cellular Proteomics, 3(9):908¨C19.

[19] Kaliszan R, Baczek T, Cimochowska A, et al. (2005). "Prediction of high-performance liquid chromatography retention of peptides with the use of quantitative structure-retention relationships." Proteomics, 5(2):409¨C15.

[20] Konrad Rieck.(2011)"Similarity measures for Sequential data"

[21] C. E. Rasmussen C. K. I. Williams,(2006) "Gaussian Processes for Machine Learning", the MIT Press.

[22] B. T. Polyak. (1969) "The conjugate gradient method in extreme problems." USSR Comp Math Math Phys, 9(4): 94-112.

[23] Chang CC and Lin CJ (2011). LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1¨C27:27. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[24] S Zhang, J liu, J.W Tian(2004) "An SVM-based small target segmentation and clustering approach [C]." In: Proceedings of the Third International Conference on Machine Learning and Cvbernetics.Shanghai:IEEE,2004,6(8):3318-3323.

[25] Tresp V. A Bayesian committee machine[J]. Neural Computation, 2000, 12: 2719-2741.

[26] Snelson E, Ghahramani Z. Sparse Gaussian processes using pseudo-inputs[C]. Proc of the NIPS 18. Vancouver, 2006: 1257-1264

[27] Gorshkov, Alexander V.; Tarasova, Irina A.; Evreinov, Victor V.; Savitski, Mikhail M.; Nielsen, Michael Lund; Zubarev, Roman A.; Gorshkov, Mikhail V. "Liquid chromatography at critical conditions: Comprehensive approach to sequence-dependent retention time prediction. " Analytical Chemistry, Vol. 78, No. 22, 2006, p. 7770-7777

.