

# SVM : assignment 2

Florentijn Degroote

May 2019

## Contents

<b>1</b>	<b>Exercises</b>	<b>2</b>
1.1	Support vector machine for function estimation . . . . .	2
1.1.1	Focus on linear kernel . . . . .	2
1.1.2	Kernel analysis on a more complex data set . . . . .	2
1.2	A simple example: the sinc function . . . . .	4
1.2.1	Regression of the sinc function . . . . .	4
1.2.2	Application of the Bayesian framework . . . . .	5
1.3	Automatic Relevance Determination . . . . .	6
1.4	Robust regression . . . . .	7
<b>2</b>	<b>Homework problems</b>	<b>9</b>
2.1	Logmap dataset . . . . .	9
2.2	Santa Fe dataset . . . . .	10

# 1 Exercises

## 1.1 Support vector machine for function estimation

### 1.1.1 Focus on linear kernel

To understand what effect the parameters *bound* and *e* have, we first take a look at the math behind the general optimisation problem for regression in SVM's:

$$\begin{aligned}
 \text{minimize} \quad & \frac{1}{2}w^T w + \text{bound} \sum_{k=1}^N (\xi_k + \xi_k^*) \\
 \text{s.t.} \quad & y_k - w^T \phi(x_k) - b \leq e + \xi_k \\
 & w^T \phi(x_k) + b - y_k \leq e + \xi_k^* \\
 & \xi_k, \xi_k^* \geq 0
 \end{aligned} \tag{1}$$

Which can be rewritten in dual form, where  $K(x_k, x_l) = \phi(x_k)^T \phi(x_l)$ :

$$\begin{aligned}
 \text{maximize}_{\alpha, \alpha^*} \quad & -\frac{1}{2} \sum_{k,l=1}^N (\alpha_k - \alpha_k^*)(\alpha_l - \alpha_l^*) K(x_k, x_l) - e \sum_{k=1}^N (\alpha_k + \alpha_k^*) + \sum_{k=1}^N y_k (\alpha_k - \alpha_k^*) \\
 \text{s.t.} \quad & \sum_{k=1}^N (\alpha_k - \alpha_k^*) = 0 \\
 & \alpha_k, \alpha_k^* \in [0, \text{bound}]
 \end{aligned} \tag{2}$$

The *bound* parameter acts as a regularizing parameter; when chosen zero, only the norm  $\|w\|$  will be minimized. When *bound* is chosen higher and higher, more importance will go out to points that are not within the *bounds* of the function approximation. When the bound parameter is infinity, only the error will be taken into account, meaning the problem is a pure **classical least squares fit**; regression SVM's are thus an extension of the classical least squares fit. This parameter's value is thus a trade-off. The actual bounds wherein a point is considered erroneous are determined by the parameter *e*. When *e* is chosen zero, every data point will have effect on the result of the optimization problem, as no point exists between the (non-existing) bounds of the function approximation. As *e* grows larger, points close to the function approximation will not have any effect on the optimization function. The absence of effect of some points on the outcome of the regression SVM can be described to the **sparsity property**.

For regression using a linear kernel, the data set that is shown in figure 1a is used. On this figure, the black line represents the function estimation for a value of 0 for both *e* and *bound*. It is completely flat because the norm of *w* was the only parameters that was affecting the optimization problem. In figure 1b, the bound parameter was altered to 0.001, resulting in a slightly better, yet underfitted function estimation as the SVM is not able to incorporate the data's structure well due to the model being restrained too much. In figure 1c, this seems to be better. Now that a preliminary value of *bound* is determined, the value of *e* can be altered! In figure 2a, the bounds are visible sparsity effect is already noticeable (red circles around data points denote support vectors). After some tweaking, a good result was found for 0.1 as value for *bound* and 0.004 for *e*. Only 4 out of the original 24 points are now support vectors, shown in figure 2b. Making the *bound* parameter bigger does effect on the regression line, whereas increasing *e* leads to the undesirable function estimation shown in figure 2c, where no points lie in between the bounds.

### 1.1.2 Kernel analysis on a more complex data set

In a next analysis, a more complex data set was erected with the goal of finding the best kernel to represent the best function approximation. The best possible fit of a linear kernel, quadratic kernel, cubic kernel, quartic kernel, exponential RBF kernel and gaussian RBF kernel are shown in figures 3a, 3b, 3c, 3d, and 3e, 3f respectively. For all kernels, *bound* was set to 10, and *e* was 0.01.

The dataset seems to be best depicted by a regression SVM with a cubic, quartic or RBF kernel. Also quintic kernels and higher level polynomials do well, but are more complex. Occam's razor states that one should always reduce the complexity of the model as much as possible while

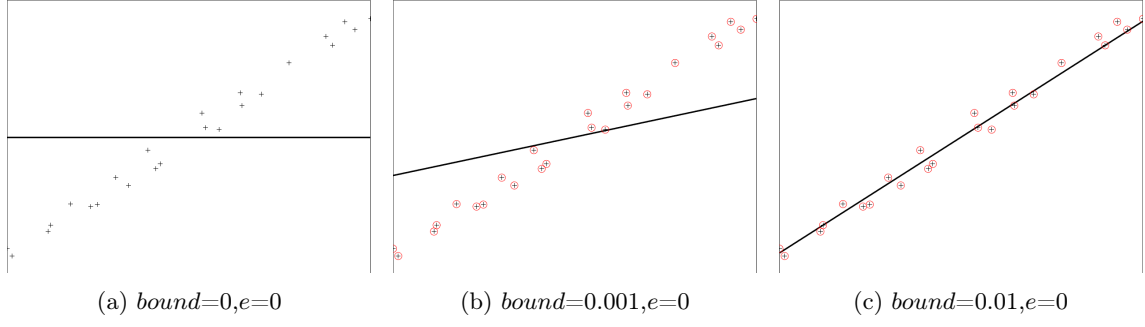


Figure 1: Effect of changing  $bound$  parameter.

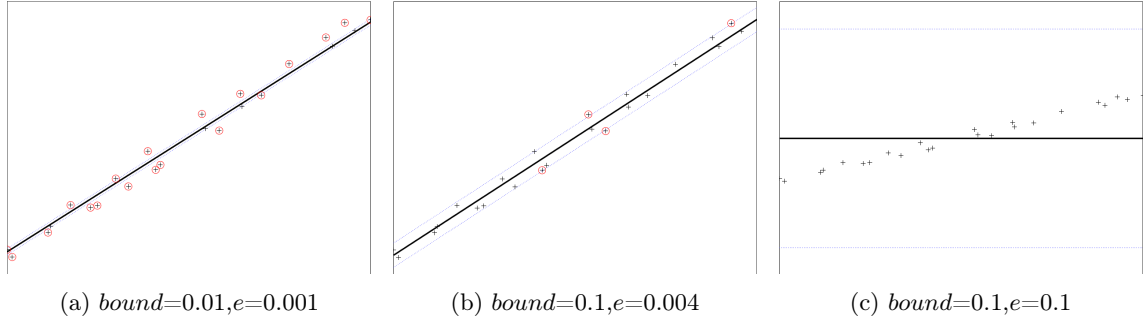


Figure 2: Taking into account  $e$ .

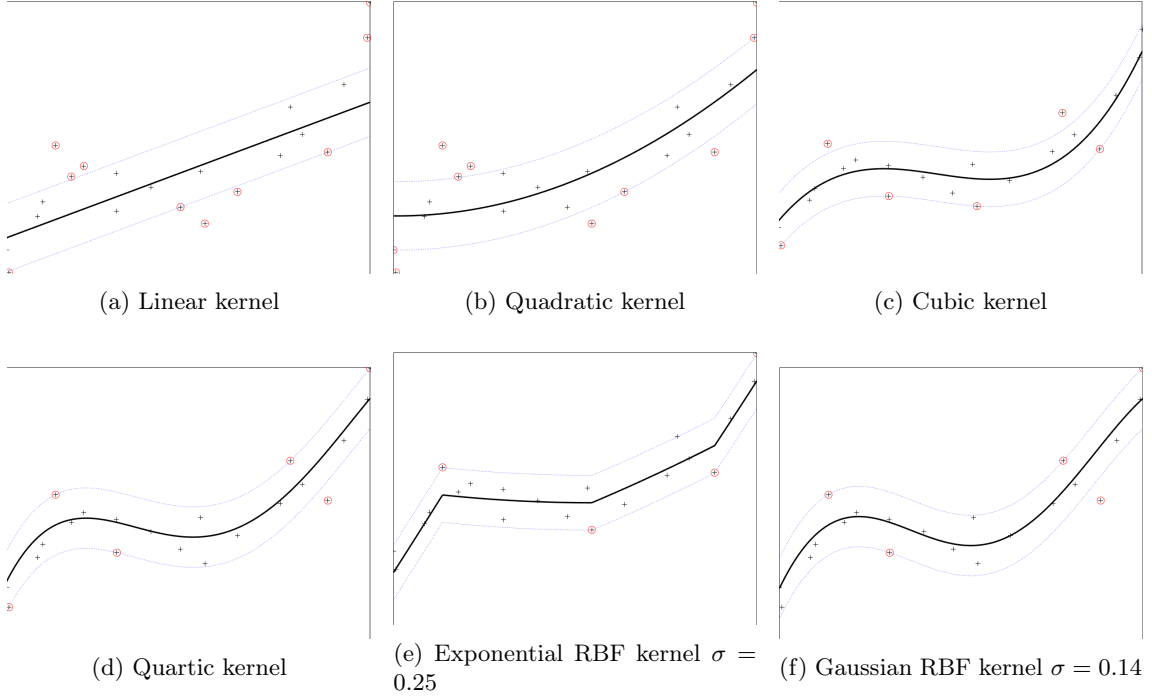


Figure 3: Regression of non-linear data set with different kernels.

being able reproduce the general scheme of the data. Applied to this data set, the cubic kernel is preferred over the quartic. Both the gaussian and exponential kernel have 4 support vectors, whereas the cubic and quartic have 6. Because the function estimation of the Gaussian kernel seems to be smoother and because it has less support vectors than the cubic kernel, I would think that this kernel is best suited for this data set.

## 1.2 A simple example: the sinc function

### 1.2.1 Regression of the sinc function

Using LS-SVM's with RBF kernels, two parameters need to be decided, namely  $\gamma$  and  $\sigma^2$ . The former acts as a regularizing parameter, just like *bound* in the previous subsection. Having this value very low results in the SVM not being able to generalise the underlying data distribution well. On the other hand, picking this parameter's value too high, the kernel would try to minimise the fitness error as much as possible. Knowing that in the training data is noisy, the approximation could very well result in overfitting. The latter represents the width of the zone of influence of each data point in the training set. This parameter could be interpreted as how big the area is to which a certain datapoint has influence on the result of the function estimation. Having this parameter too low will result in data points having a big say in how the function estimation looks like within their close proximity, resulting in overfitting. Having it too low will lead to the data points' influence being very wide so that they interfere with each other. This could result in bad function approximations.

Let's now conduct a graphical analysis of function estimations with a RBF kernel on a noisy sinc function. Figure 4a and 4d show the resulting function approximation and training set for an arbitrarily chosen  $\sigma^2 = 1$  and  $\gamma = 0.01$ . Increasing the value of  $\gamma$  allows the kernel to better estimate the function, shown in figures 4c and 4e. For  $\gamma = 10000$ , the function estimation seems satisfactory (see figures 4b and 4f). If we then decrease the  $\sigma^2$  parameter, the function estimation gets a lot worse as we also conferred in the previous paragraph (figures 5a and 5d). Increasing  $\sigma^2$  leads to an overfitted solution (figures 5b and 5e). It must be noted that the situation in figures 4b and 4f not the only configuration that works out. In figures 5c and 5f, for  $\gamma = 100$  and  $\sigma^2 = 0.4$ , the function estimation is also solid, and is actually preferred over the first, as the gamma parameter is lower (possibly averting overfitting and making the kernel more robust). Keeping in mind the parameter space and the trade-off that one makes when choosing the two parameters makes me believe that there is no optimal pair of hyperparameters. Instead, lots of pairs do work out well, while others are for sure not suitable.

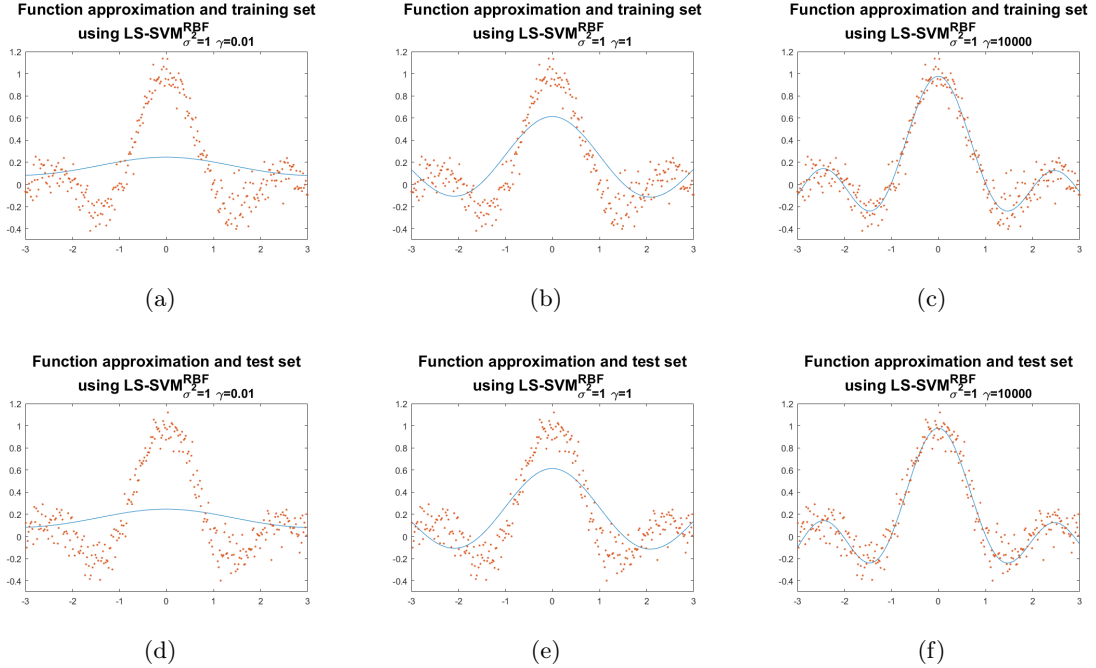


Figure 4: Effect of parameter  $\gamma$ .

The RMSE of the conducted experiments are shown in table 1 and 2.

Instead of tuning the parameters manually, we can also make use of the built-in automatic hyper-parameter tuning algorithms. Two major algorithms will be looked into, namely the grid search algorithm (which scans a grid of parameters and picks the best configuration, based on a measure) and the Nelder-Mead method (gradient-based optimization, which is faster than brute force grid search). In both cases, 10 fold cross-validation is applied.

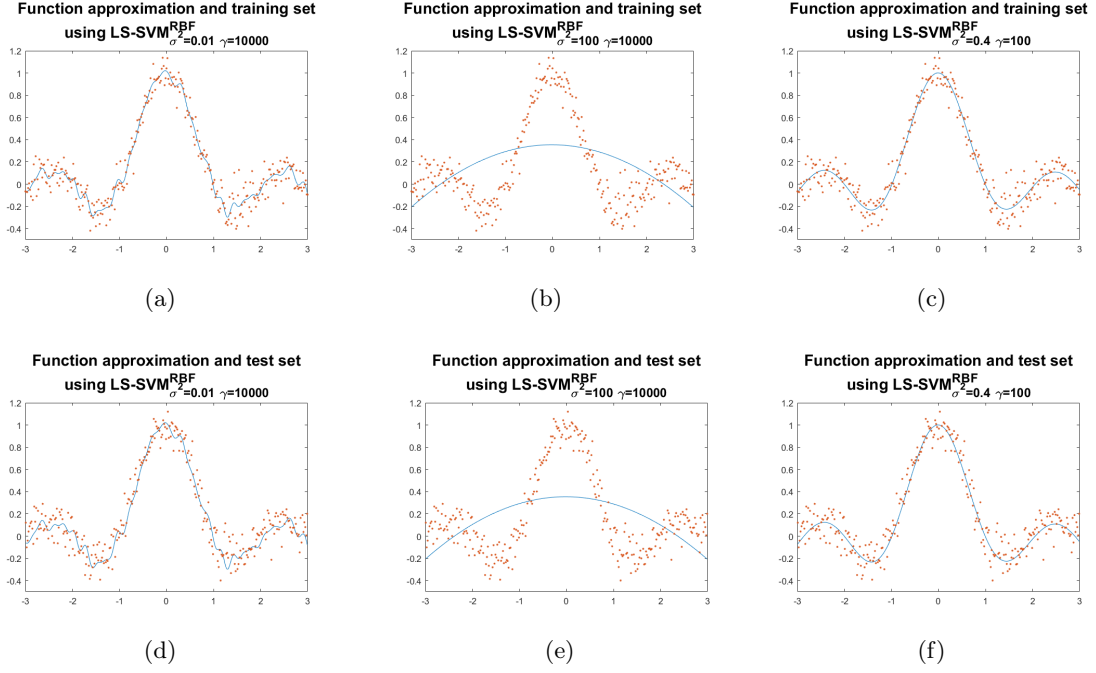


Figure 5: Further tweaking.

$\gamma$	0.01	0.1	1	10	100	1000	10000
RMSE	0.3438	0.291	0.2258	0.1779	0.1218	0.1124	0.1061

Table 1: RMSE for different values of  $\gamma$  when  $\sigma^2 = 1$ .

$\sigma^2$	0.01	0.1	1	10	100
RMSE	0.1111	0.1071	0.1061	0.2557	0.3374

Table 2: RMSE for different values of  $\sigma^2$  when  $\gamma = 10000$ .

Results (in terms of the mean  $\mu$  and variance  $\sigma^2$ ) of 30 runs for computation times and RMSE's of the solution of both techniques are shown in table 3. The Nelder-Mead method is significantly

	$\mu_t$	$\sigma_t^2$	$\mu_{RMSE}$	$\sigma_{RMSE}^2$	$\mu_\gamma$	$\sigma_\gamma^2$	$\mu_{\sigma^2}$	$\sigma_{\sigma^2}^2$
Nelder-Mead method	0.7318	0.0026	0.0990	7.96e-05	131.1567	2.18e+07	0.4407	0.0110
Gridsearch	1.1678	0.0011	0.0992	1.10 e-4	352.4359	2.62e+05	0.4530	0.0053

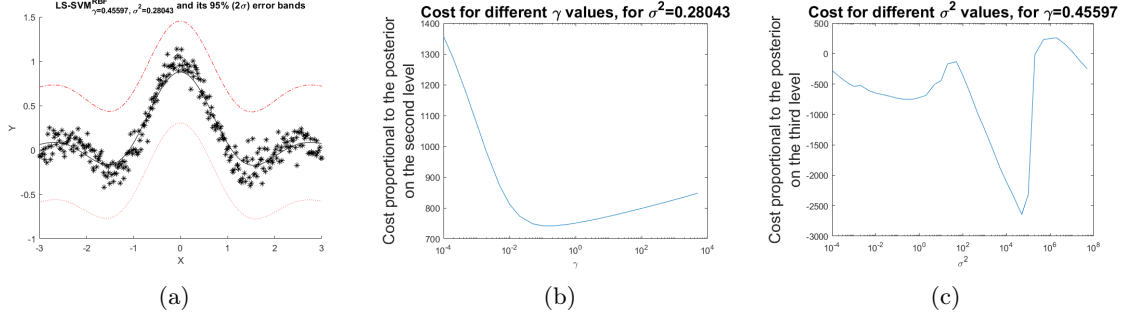
Table 3: Statistics of 30 runs for the two optimization procedures.

faster and RMSE values are also slightly better. For both methods, there is quite some variation in the solution of  $\gamma$  ( $> 10^5$ ) and less so for  $\sigma^2$ .

### 1.2.2 Application of the Bayesian framework

In the bayesian framework, parameters of the model can be evaluated on a measure called the likelihood, representing how likely it is that a given set of parameters make up for an SVM that is able to classify the data well (in case of classification) or represent the data well (in form of a function estimation, in regression). In total, three different interwoven levels make it possible to infer information with respect to specific parameter(s). The first level handles the parameter  $w$  and  $b$ , that are results from the SVM optimisation itself. The second level handles the regularization parameter  $\gamma$  and the third (optional) level infers information about any kernel parameter, in our case  $\sigma^2$ , as we are working with RBF kernels. Next to calculating the cost (negative logarithm of the posterior), the LS-SVM toolbox also permits the user to let a function decide the best model parameters ( $\gamma$  and  $\sigma^2$ ) with the function `bay_optimize()`. The framework needs to be initialized

with an initial "guess" of the parameters, aka prior, and in our case  $\gamma = 10$  and  $\sigma^2 = 0.4$  were wielded. The resulting regression is and 95 % error bars are shown in figure 6a. Figures 6b and 6c show the cost function for different parameter values according with level 2 and 3, respectively, each time calculated with the result of the *bay\_optimize()* function. As one can see, the calculated



value for  $\gamma$  is the actual minimum in the log-likelihood related cost function. For  $\sigma^2$ , 0.28043 is a local minimum rather than a global optimum.

### 1.3 Automatic Relevance Determination

In ARD, the bayesian framwork and its three levels of inference to calculate the posterior can be used to determine the most fitting features of a given data set that accord most with the (in this case regression) SVM, defined by the parameters  $\gamma$  and  $\sigma^2$ . The ARD first starts by considering all input dimensions important, and prunes them one by one. An input dimension is pruned if there is a  $\sigma^2$  of the subset without the considered input dimension which involves a higher likelihood than the best likelihood for the total set. For the provide code, ARD detects that only the first input is relevant. Figure 7 shows that indeed, this is the case as input dimension two and three are random noise.

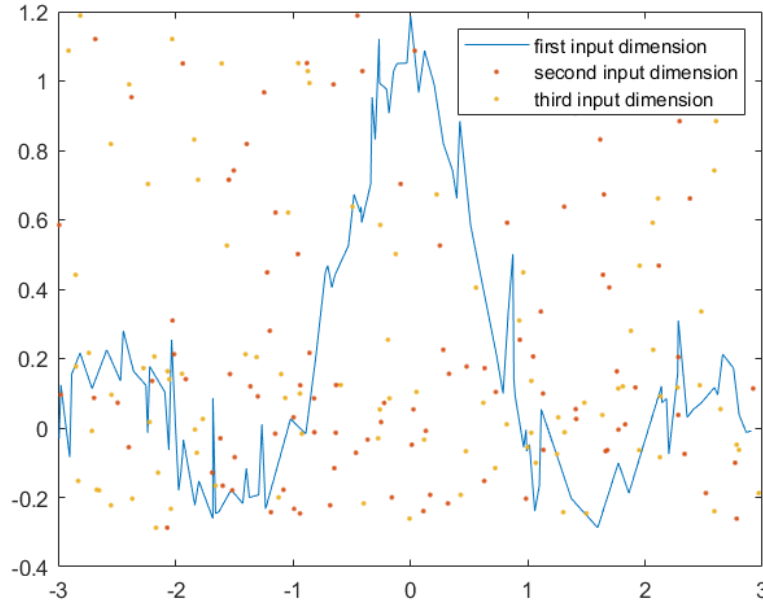


Figure 7

Instead of the cost function wielded by the ARD, which is based on the log likelihood of the third level of the bayesian framework, other measures can be taken into consideration to determine whether an input dimension is relevant or not. One of those measures is the mean-squared-error (MSE) of the data set on the function estimation with the given parameters  $\gamma$  and  $\sigma^2$ . In order to get a truthful estimation of the SME, cross-validation can be used. The reported statistic is then

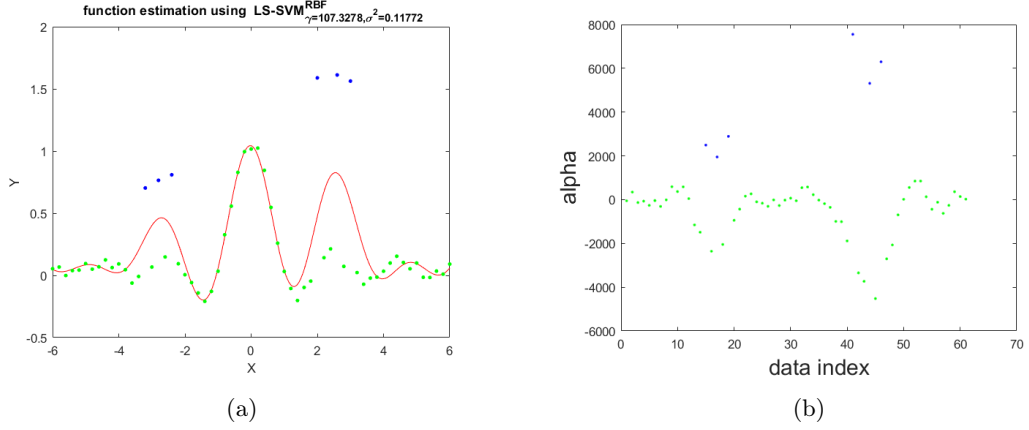
the mean MSE. These are listed in table 4 for the 7 different configurations. These results lead to the same conclusion as the ARD.

Configuration	1	2	3	{1,2}	{1,3}	{2,3}	{1,2,3}
Mean MSE	0.0270	0.1744	0.1676	0.0541	0.0650	0.1836	0.0962

Table 4: Results of cross validation for the different configurations of the input dimensions.

## 1.4 Robust regression

For regression with outliers, extra measures were taken and added to the LS-SVM toolbox to detect outliers and not let the function estimation be influenced by these data points. In figure 8a, the function estimation is depicted for parameters obtained by the *tunelssvm()* function for the data points shown in green and blue (of which the blue points denote the outliers). The function estimation in the region of these outliers seems to not follow the sinc function, due to the function trying to model the outliers as well. Figure 8b shows the alpha values (weights) of each data point. As we can see, the weight of the outliers and the points close to the outliers in x-direction have a big (opposite) influence. SVM in its classical form is thus very sensible to outliers. In order to



deal with outliers in the data, statistical methods have been developed such as the Huber loss or Hampel loss, that involve using a weighted cost function. These cost function effectively penalizes the outliers' weight on the function estimation. Also logistic losses or myriad weighted myriad filters can be used. For the formulas, I'd like to refer to figure 9 [1].

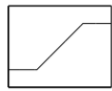
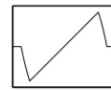
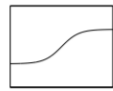
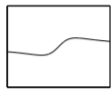
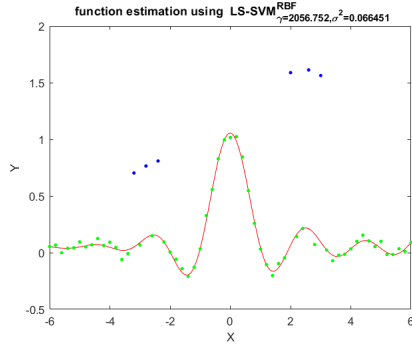
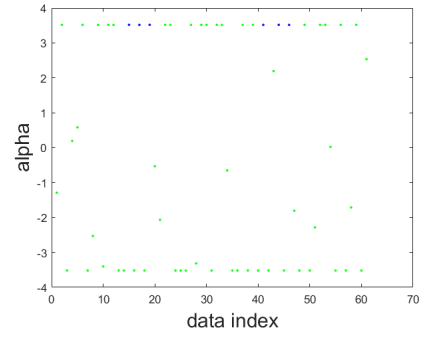
	Huber	Hampel	Logistic	Myriad
$V(r)$	$\begin{cases} 1, & \text{if }  r  < \beta; \\ \frac{\beta}{ r }, & \text{if }  r  \geq \beta. \end{cases}$	$\begin{cases} 1, & \text{if }  r  < b_1; \\ \frac{b_2 -  r }{b_2 - b_1}, & \text{if } b_1 \leq  r  \leq b_2; \\ 0, & \text{if }  r  > b_2. \end{cases}$	$\frac{\tanh(r)}{r}$	$\frac{\delta^2}{\delta^2 + r^2}$
$\psi(r)$				
$L(r)$	$\begin{cases} r^2, & \text{if }  r  < \beta; \\ \beta r  - \frac{1}{2}\beta^2, & \text{if }  r  \geq \beta. \end{cases}$	$\begin{cases} r^2, & \text{if }  r  < b_1; \\ \frac{b_2 r^2 -  r ^3}{b_2 - b_1}, & \text{if } b_1 \leq  r  \leq b_2; \\ 0, & \text{if }  r  > b_2. \end{cases}$	$r \tanh(r)$	$\log(\delta^2 + r^2)$

Figure 9: Weight functions  $V(r)$ , score functions  $\psi(r)$  and loss functions  $L(r)$  for the Huber, Hampel, Logistic and Myriad approaches [1].

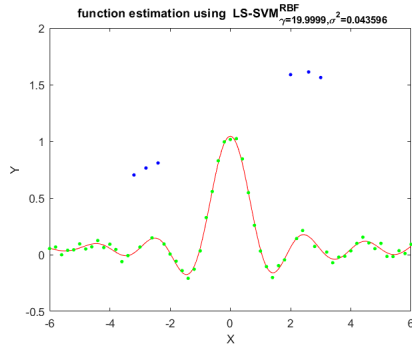
Figure 10 encapsulates the results for the four different techniques. It seems that for logistic and Huber, the weights have been bounded, so that the influence of the data points is minimized. Though, the best results are obtained for Hampel and Myriad, where the function estimation seems to be minimally affected by the outliers, and where the weights of the points are not among the biggest weights anymore.



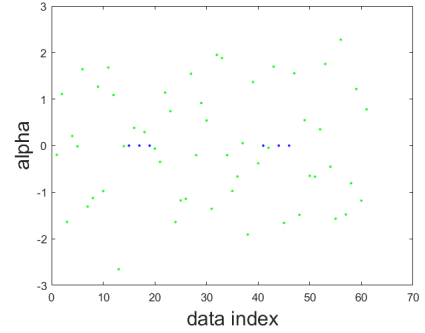
(a) Function estimation for Huber



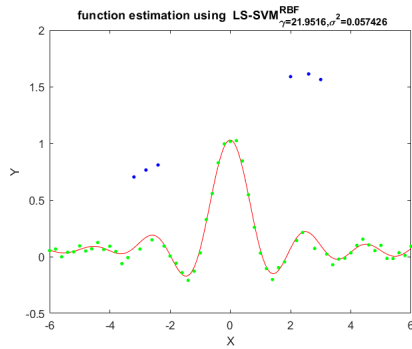
(b) alpha values for Huber



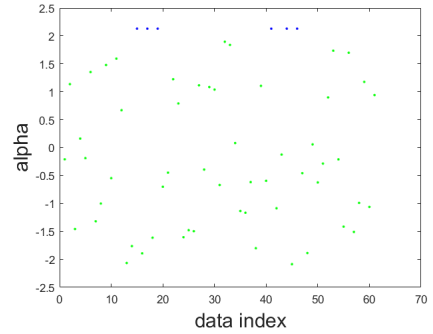
(c) Function estimation for Hampel



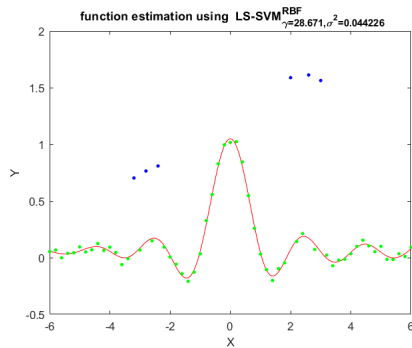
(d) alpha values for Hampel



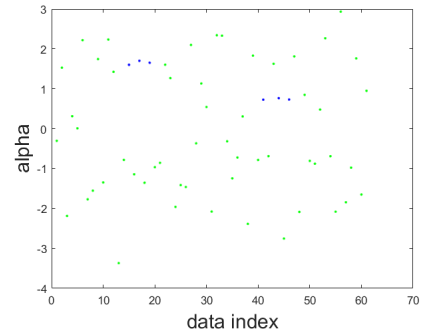
(e) Function estimation for logistic



(f) alpha values for logistic



(g) Function estimation for Myriad



(h) alpha values for Myriad

Figure 10: Robust regression techniques

Because of the square in mean squared error, large errors have relatively greater influence on MSE than smaller error do. Since MAE does not involve squaring the error, the measure is more



robust to outliers and is preferred in this setting.

## 2 Homework problems

### 2.1 Logmap dataset

For the logmap dataset, SVM regression is used in combination with the linear autoregressive model of the training and test data. The training set consists of 150 points, and the test set of 50 points. The example code of the assignment sheet generates a inferior model, not predicting the right tendencies of the data, as shown in figure 11.

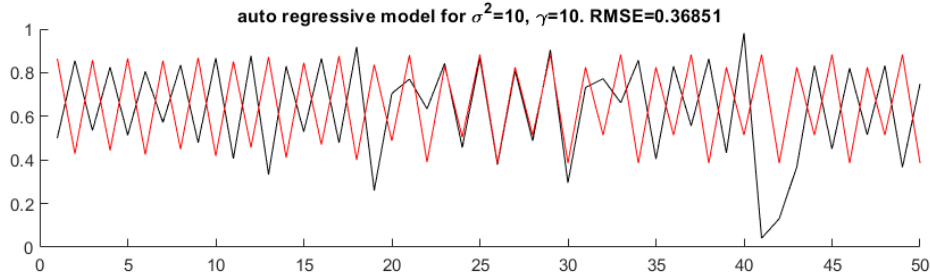
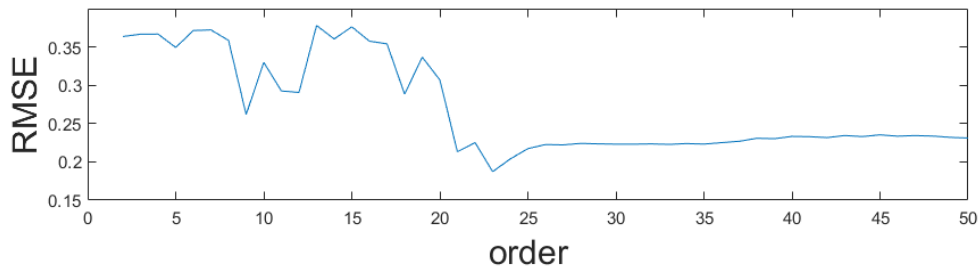
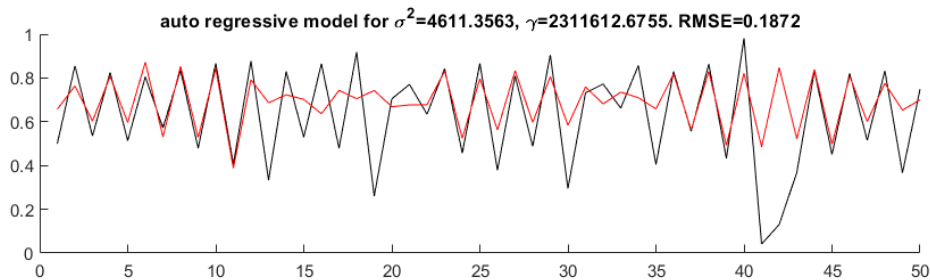


Figure 11

In order to correctly validate the model on overfitting, a validation set should be erected on which extra analysis should be conducted (e.g. leave-one-out, k-fold cross validation). In this exercise, 10-fold cross validation was used to determine the parameters  $\sigma^2$  and  $\gamma$ . This cross-validation splits the training set into a training and test set. Model orders of size 2 up until 50 are looked into. For each of these model orders, the *tunelssvm()* was wielded 10 times, and the best configuration of the parameters was being kept, on basis of the RMSE measure (note that RMSE was measured on the test set). In figure 12a, the best RMSE values for the different model orders are registered. The best one is registered for a model order of 23. Results of this particular configuration are shown in figure 12b. The results are far from perfect, still, but the prediction



(a)



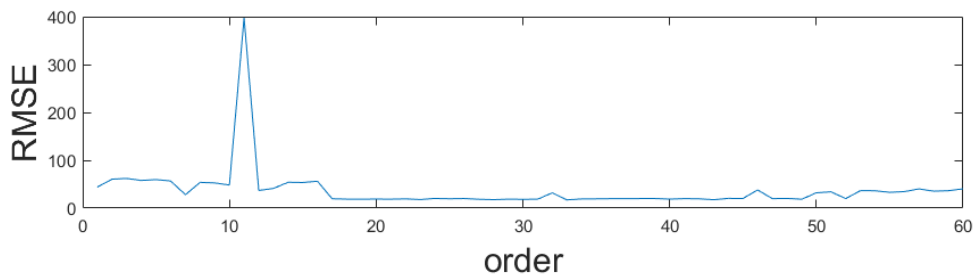
(b)

seems to be able to detect the correct frequency and in certain intervals, the right amplitude as well. Other measures could be used to choose the best model upon, in order to get different predictions (for example a weighted RMSE that puts a lot of weight on the first data points in comparison

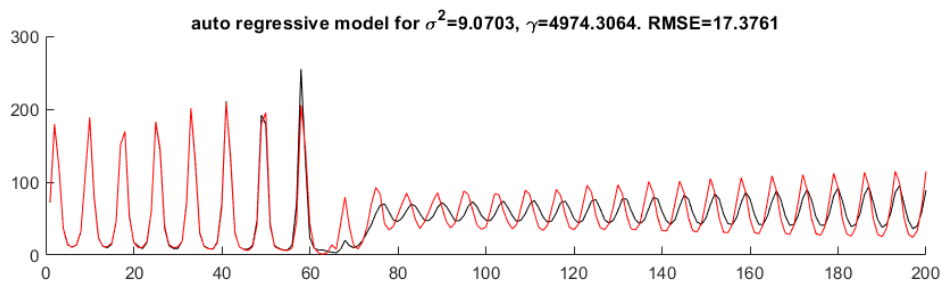
with the last could lead to a model that is able to fit the first 20 data points better than in this case). One could note that the values for  $\sigma^2$  and  $\gamma$  are quite large, especially compared to the results of a two dimensional problem, like in section 1. An important aspect of the explanation lies in the fact that the input is now 23-dimensional. The 150 data points are scattered in this 23-dimensional space and are very distant from each other. This permits the width of the RBF kernel ( $\sigma^2$ ) to be large. The big value of the regularization parameter indicates that the error of the data points is imperative for a good function estimation.

## 2.2 Santa Fe dataset

For the santa Fe dataset, the first 1000 laser measurements act as training set and the 200 consecutive measurements as test set. When visualising the data, I would think that 50 is not a bad choice for the model order, because the general tendency of the laser (frequency as well as collapses) can be depicted within a window of 50 data points. The question is however, if there are lower orders that perform as good as, or even better, than 50. It is sensible to use the performance of the recurrent prediction on a validation set to optimize hyperparameters and the model order, when the validation set is not the test set. It is possible that by tweaking the parameters more than necessary, so that the results on the validation set are outstanding, the results on the test set are disappointing. In that case, we have overfitted our solution to the validation set and are able to detect this with the test set. The validation is then, of course, not part of the training set (leave-one-out/k-fold cross-validation). The test set is thus used as a final judgment on the model. For this homework problem, the test set will, just like in the previous problem, be used to determine the best model order. As the test set is thus also used to tune a parameter with, the borderline between the test set not being a kind of validation set is vague. Analogous to the previous exercise, *tunelssvm()* is ran 10 times for each model order ranging from 2 to 60. The best RMSE values for each order are shown in figure 13a. The best prediction is shown in figure 13b.



(a)



(b)

## References

- [1] Kris De Brabanter, Kristiaan Pelckmans, Jos De Brabanter, Michiel Debruyne, Johan AK Suykens, Mia Hubert, and Bart De Moor. Robustness of kernel based regression: a comparison of iterative weighting schemes. In *International Conference on Artificial Neural Networks*, pages 100–110. Springer, 2009.