1. Assignment 2

Suppoprt vector machine is a machine learning technique that can be used for classification and for regression. In this section, we will focus on regression using least square support vector machines.

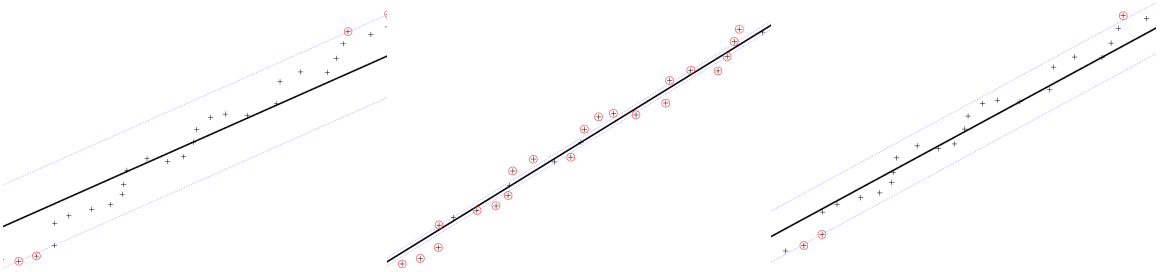
- • Construct a dataset where a linear kernel is better than any other kernel (around 20 data points). What is the influence of e (try small values such as 0.10, 0.25, 0.50, . . .) and of Bound (try larger increments such as 0.01, 0.10, 1, 10, 100). Where does the sparsity property come in?

We draw the datapoints where a linear kernel is better than any other kernel. Afterwards, we will try to investigate the effect of the ϵ and *bound* variables on the result.
ϵ variable controls the width of the of the ϵ-tube. Since, we know that it is impossible to have all the datapoints inside the ϵ-tube, we introduce extra slack variable which determines the degree to which the points outside of the ϵ-tube are tolerated. The equation for the minimization term is shown in 1. If we

$$\frac{1}{2}w^T w + c\sum_{k=1}^{N}(\xi_k + \xi_k^*)$$

Figura 1: Equation to minimize

choose the C value to be very large then we are focusing on minimize the error only as compared to the first term. If the C variable is 0 then changing the ϵ value will result in increment or decrement of ϵ-tube shape.. Sparsity refers to the fact that we can have regression line with few support vectors. ϵ value



(a) $C = 0{,}1$ & $\epsilon = 0{,}1$ (b) $C = 0{,}1$ & $\epsilon = 0{,}01$ (c) $C = 0{,}1$ & $\epsilon = 0{,}06$

Figura 2: Linear kernel & data

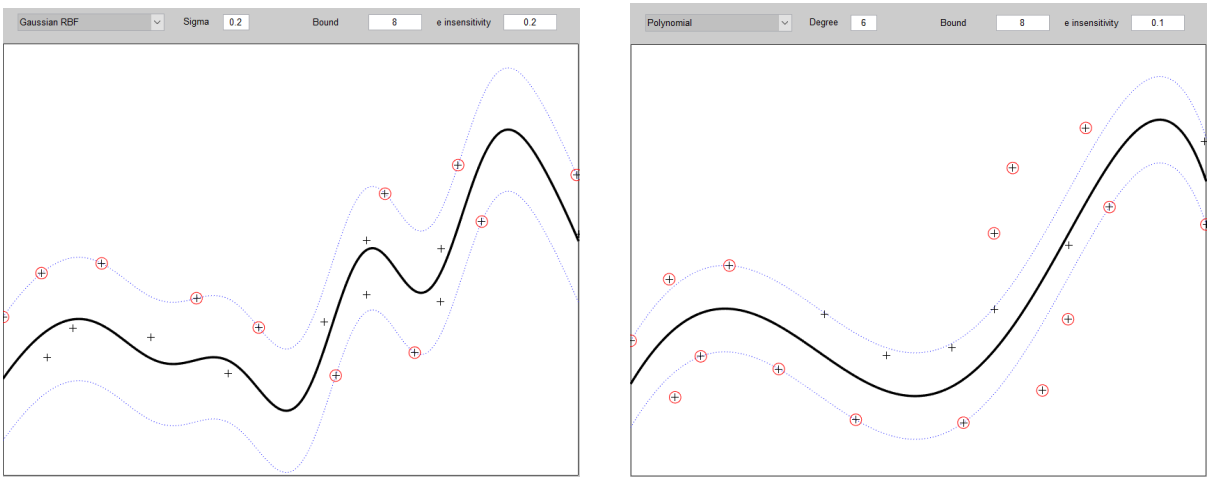
determines the sparsity of the model. The points that lie outside the ϵ-tube are the support vectors. From figure 2, it can be seen that those points which lie inside the ϵ-tube are not support vectors. Hence, the ϵ-value determines the sparsity of the model.

- • Construct a more challenging dataset (around 20 data points). Which kernel is best suited for your dataset? Motivate why.

In this section, we will construct a more complex dataset and try to fit the RBF and polynomial kernel. Figure 3 shows the rbf kernel and poly kernel used for regression on the same data. Increasing the degree of polynomial kernel gives smoother curve but RBF kernel can create complex boundary. We have also very few support vectors with rbf kernel as compared to polynomial kernel. In the next section, we will use least square support vector machine which differs from the standard svm because ls-svm tries to minimize the square of the errors which leads to simplified constraints equations. In standard svm, we have two different in-equality constraints but in ls-svm there is not in-equality constraints. As we use square of the errors, the in-equality constrains turn into equality constrainst.

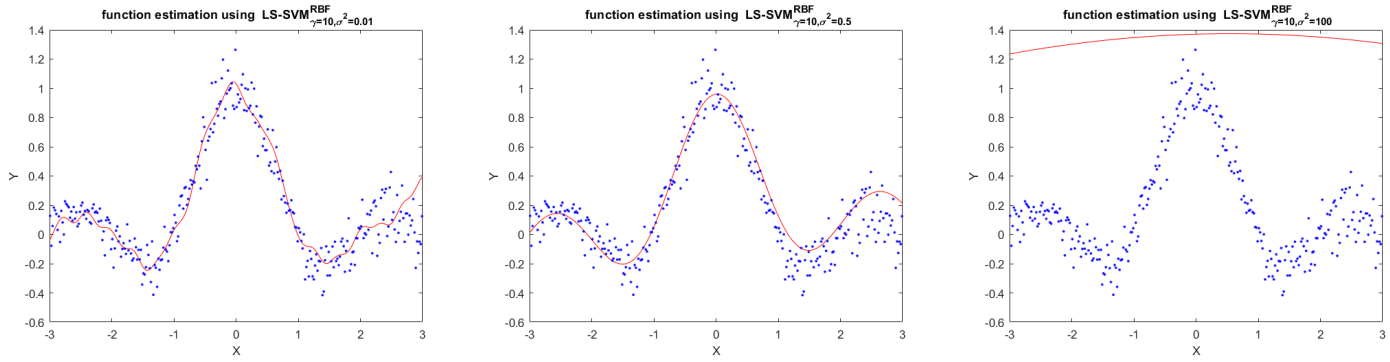
LS-SVM

We will use least square svm for function estimation. We will try different values of σ and γ using rbf kernel. Figure 4 shows the visualization of the different function estimations with different parameters of

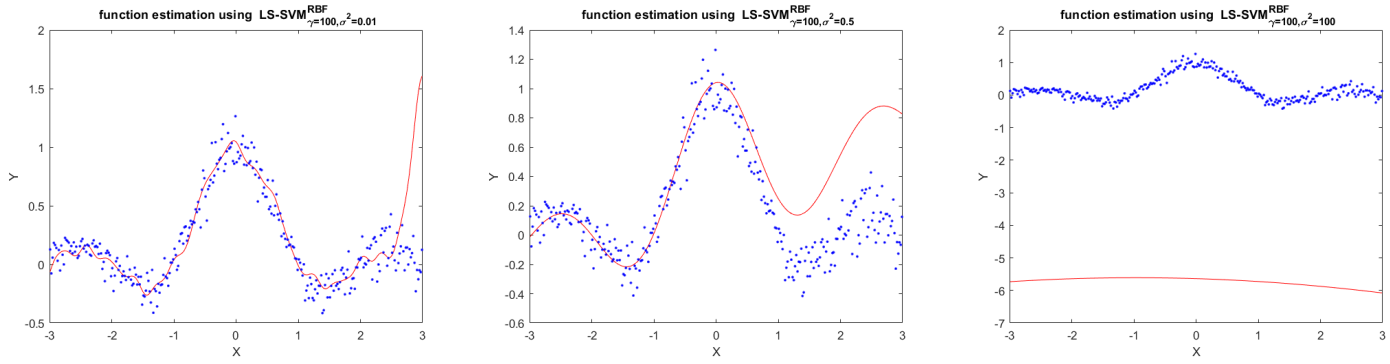


(a) RBF kernel (b) Polynomial kernel

Figura 3: RBF kernel & poly kernel



(a) RBF kerne: $\gamma = 10$ & $\sigma^2 = 0{,}01$ (b) RBF kerne: $\gamma = 10$ & $\sigma^2 = 0{,}5$ (c) RBF kerne: $\gamma = 10$ & $\sigma^2 = 10$

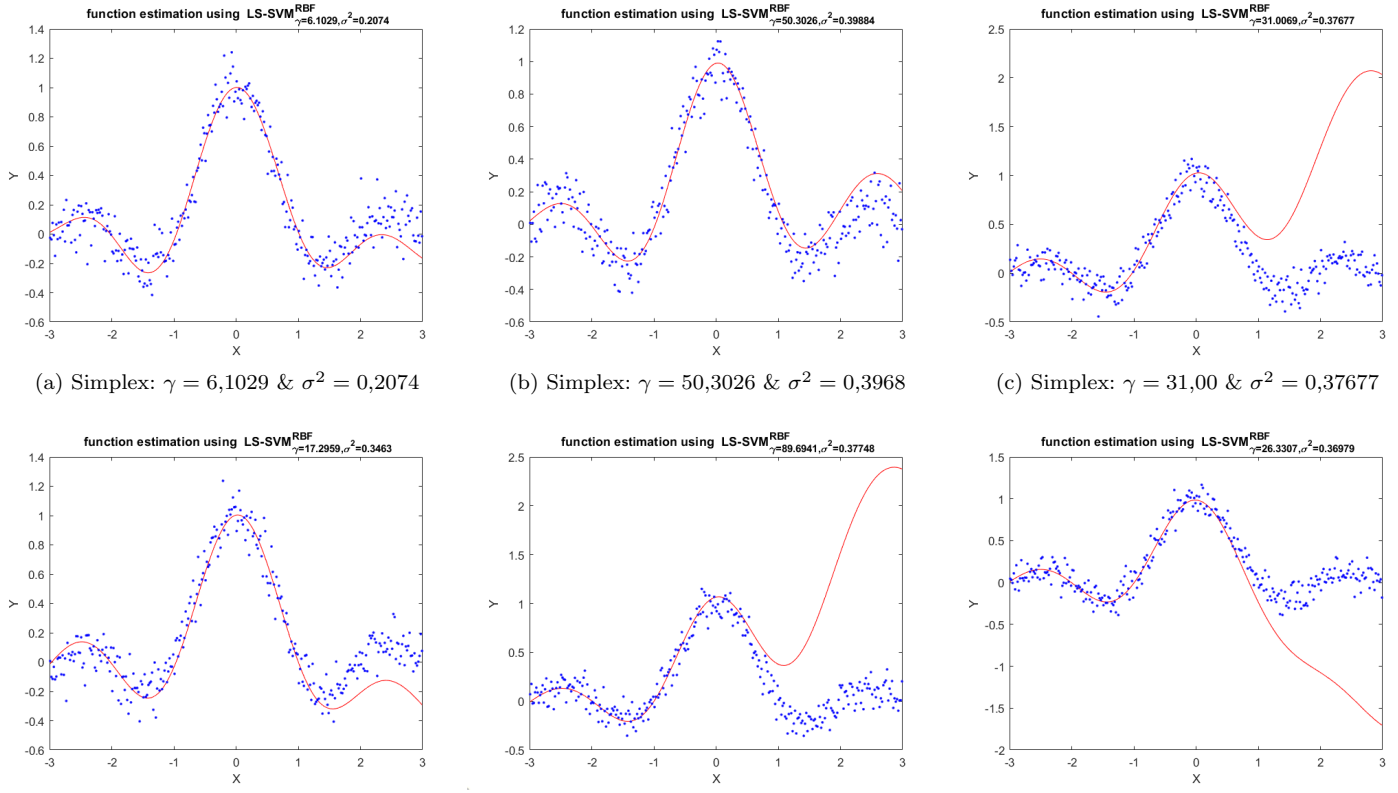


(d) RBF kerne: $\gamma = 100$ & $\sigma^2 = 0{,}01$ (e) RBF kerne: $\gamma = 100$ & $\sigma^2 = 0{,}5$ (f) RBF kerne: $\gamma = 100$ & $\sigma^2 = 100$

Figura 4: RBF kernel & function estimation

rbf kernel. There is probably no single best choice for parameters but we can define a range of variables in which we can expect good results. Next, we can try to find the optimal σ and γ using auto tuning. By tuning multiple times, we obtain each time a different value of σ and γ As mentioned eearlier, there is no single choice for parameters. When we minimize the objective function, we have two variables, σ and γ. If both of these values are low or high then we can obtain better result but sometime we minimize our objective function with respect to only one value and the second value remains large. In that case,

gamma/sigma	0.01	0.1	0.5	1	100
10	0.0124	0.0116	0.0130	0.0357	0.1363
10^2	0.0122	0.0111	0.0112	0.0161	0.1187
10^3	0.0122	0.0113	0.0107	0.0126	0.1168



(a) Simplex: $\gamma = 6{,}1029$ & $\sigma^2 = 0{,}2074$ (b) Simplex: $\gamma = 50{,}3026$ & $\sigma^2 = 0{,}3968$ (c) Simplex: $\gamma = 31{,}00$ & $\sigma^2 = 0{,}37677$

(d) GridSearch: $\gamma = 17{,}2959$ & $\sigma^2 = 0{,}3463$ (e) GridSearch: $\gamma = 89{,}6941$ & $\sigma^2 = 0{,}37748$ (f) GridSearch: $\gamma = 26{,}3707$& $\sigma^2 = 0{,}3697$

Figura 5: RBF kernel & function estimation using auto tuning with simplex search: Test set visualization

overfitting occurs.
We will now investigate the simplex search and gridsearch. Since tuning with simplex or gridsearch gives different result each time, we will compute average mean squared error.

Method	Avg_mse_10-Fold	γ 10-fold	σ 10-fold	Avg_mse_L-O-O	γ LOO-fold	σ LOO-fold
Gridsearch	0.0098	2.3509×10^{04}	0.5388	0.0103	461.0218	0.4492
Simplex	0.0090	4.6167×10^{03}	0.5995	0.0105	892.9561	0.5010

It can be seen from the table shown above that on average, mse with 10-fold crossvalidation and leaveoneout crossvalidation is almost same for both models. However, the gamma value indicates that the models with 10 crossvalidation are complex. These models have put more emphasis on the errors. Gridsearch seems to work better with leaveoneout strategy because both values(σ,γ) are small.

Bayesian framework

Crossvalidation is a regularization technique which is used to avoid overfitting. However, some portion of the data is lost because we have to use it for validation.
Another way of determining the parameters of the model are by using bayesian framework.

- Discuss in a schematic way how parameter tuning works using the Bayesian framework. Illustrate this scheme by interpreting the function calls denoted above.

Bayesian framework infers the parameters at three different levels. These three levels are:

$$p(w,b|D,u,C,H_0) = \frac{p(D|w,b,u,C,H_0)}{p(D|u,C,H_0)} * p(w,b|u,C,H_0) \tag{1.1}$$

At first level, parameter in primal space are infered. These give us probabilistic interpretation of the output. In matlab, we simply pass the number corresponding level for which we want to infer the parameters.
At second leve, we infer the hyper-parameters. These parameters define the bias and variance tradeoff in original equations.

$$p(u,c|D,H_0) = \frac{p(D|u,C,H_0)}{p(D|,H_0)} * p(u,C|H_0) \tag{1.2}$$

At third level, one infers the kernel parameters.

$$p(H_0|D) = \frac{p(D|H_0)}{p(D)} * p(H_0) \tag{1.3}$$

In matlab, this level corresponds to number 3. All of these level has same form which is:

$$Posterior = \frac{Likedlihood}{Evidence} * Prior \tag{1.4}$$

1.1. Automatic Relevance Determination

Automatic Relevance Determination is technique to find out which input variables have significant impact on results(better).

- Visualize the results in a simple figure.

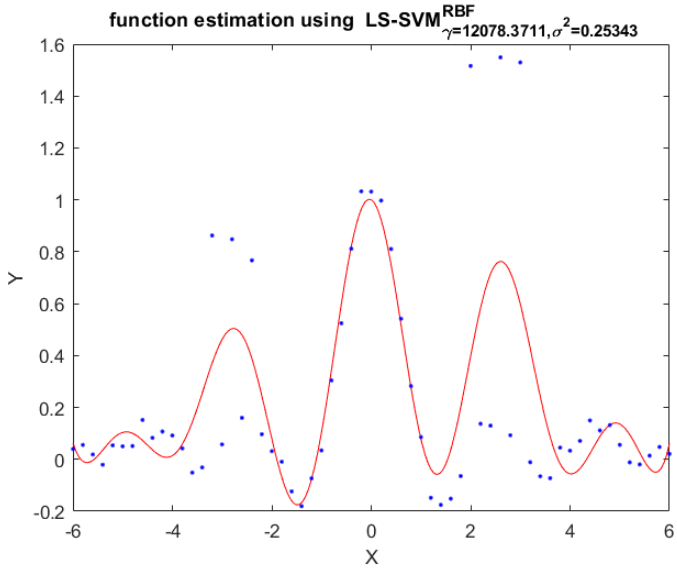
```
X = 6.* rand (100 , 3) - 3;
Y = sinc ( X (: ,1) ) + 0.1.* randn (100 ,1) ;
[ selected , ranking ] = bay_lssvmARD ({ X , Y , 'f', gam , sig2 }) ;
```

Robust Regression

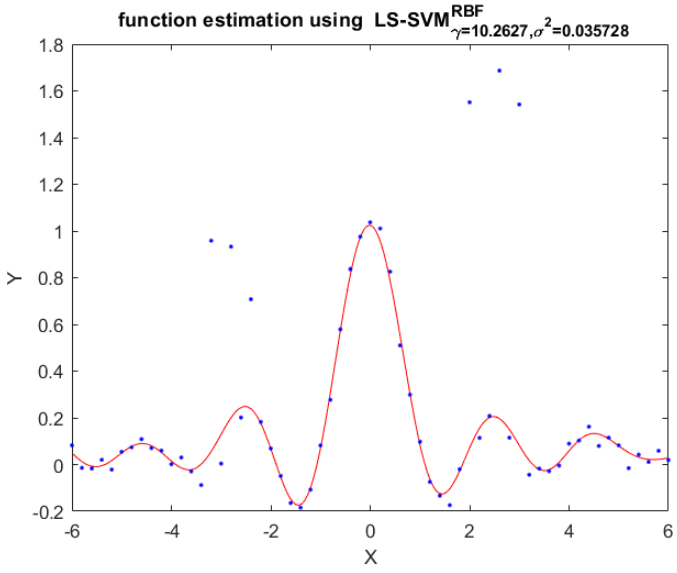
Sometimes we have outliers in our data and special attention must be given so that the model can be made rosbut against such outliers. Figure 1.1 shows the difference between a function which was estimated without handling noise while the second figure shows a function for which parameters were tuning using rcrossvalidatelssvm en whuber. Function estimated non-robustly has two peaks toward outliers while the function which is estimated by taking robustness into account, fits the data nicely. We tried different weight functions but there was not much difference in results except wmyriad weight function, which threw some warnings about matrix being bad scaled.

Logmap dataset

We will build a timeseries prediction model for Logmap dataset. In time series predition, we have to tune three different variables(σ,γ,order). In order to tune our model, we can put different order values in a list and then try to optimize gamma and sigma value with each of these order value. In the end, the model with the least mae error is used.



(a) Non-robust fucntion estimation



(b) Robust function estimation

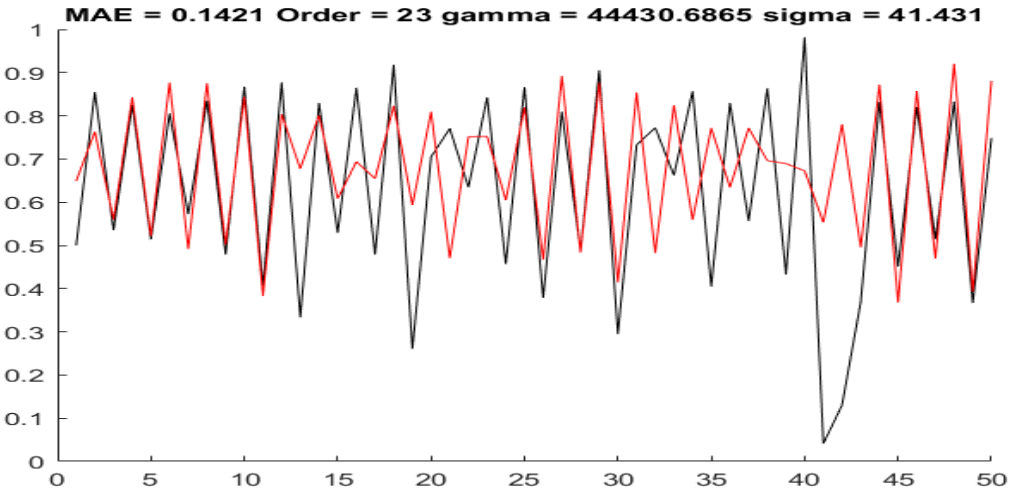
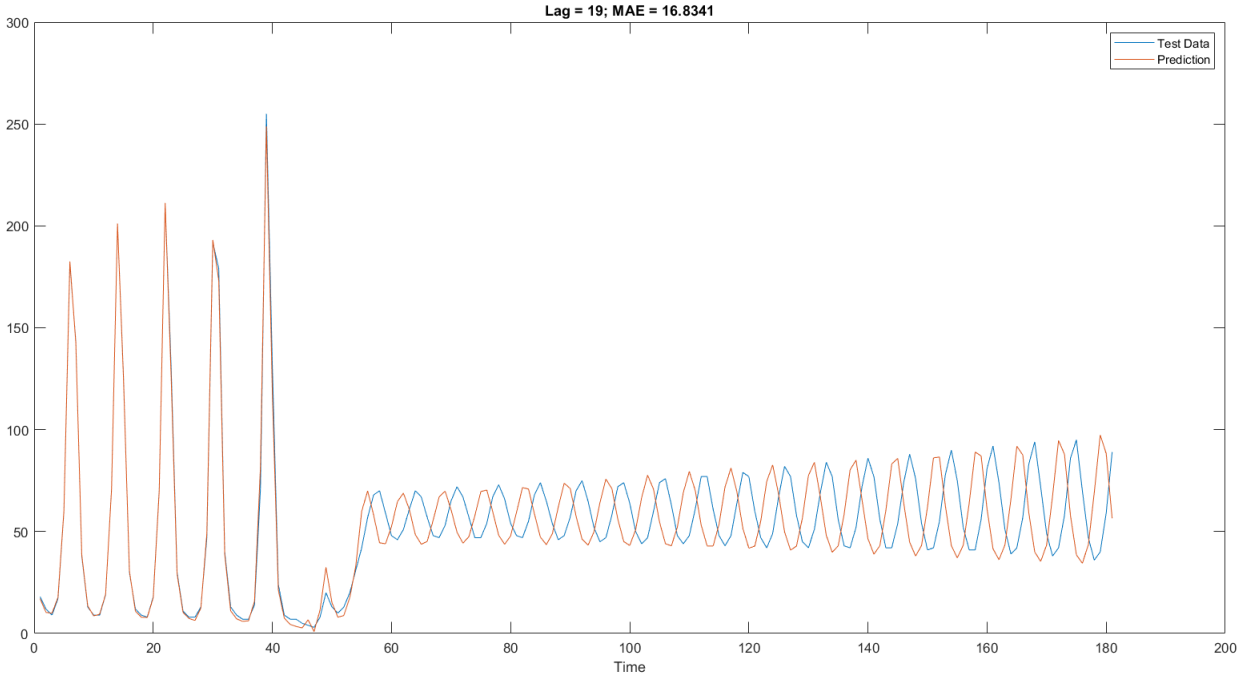


Figura 6: Optimal results after tuning

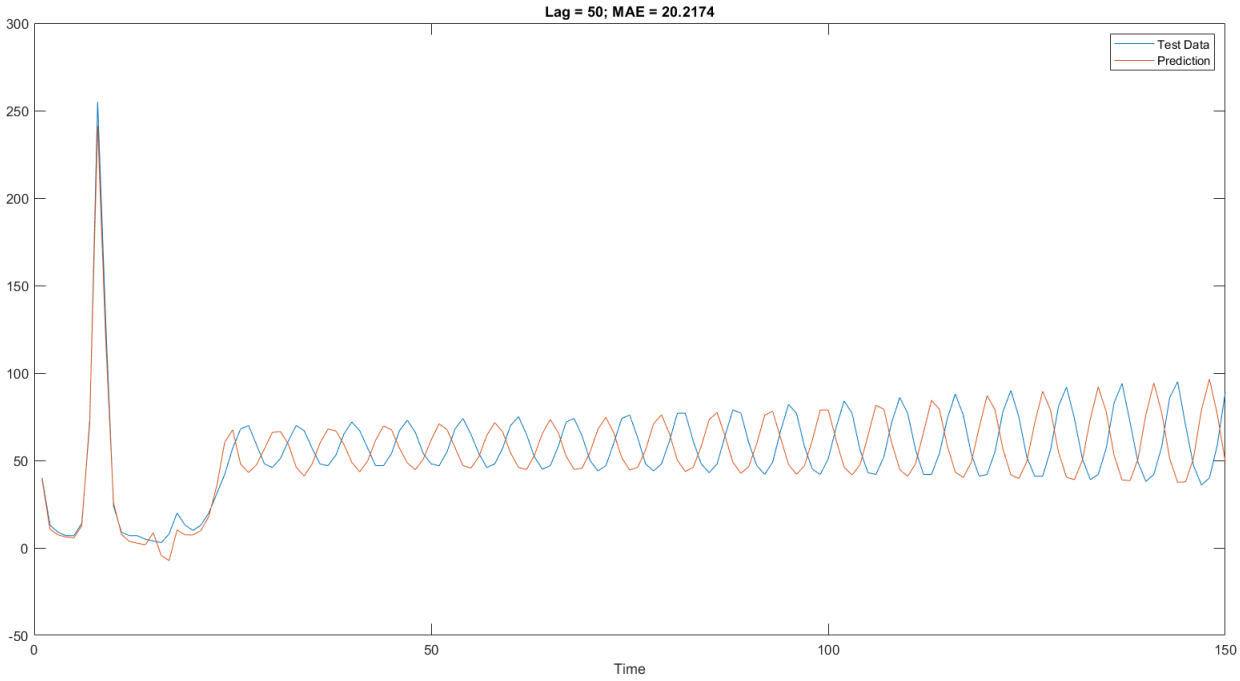
Time section prediction

In this section, we will experiment with Santa Fe dataset. The aim of this section is to train a model that can be used for timeseries prediction. There are three parameters(σ,γ,order) that need to be tuned. We can not chose a fixed value for order as it is an iterative process to find a better value for order. We will use different values of order and compute the error. We will use the order value for which the error is minimal.
We will tune the model to find better parameters. Initially, we insert the order values from 1 to 50 in a list then with each order value we try to find the best σ and γ value.
The best performance we found with lag= 19. From 7 and 8, we can see that lag =50 is a good choice but we can find better results with lag= 19.



(a) Time series prediction with lag = 19

Figura 7: Time series prediction



(a) Time Series prediction with lag = 50

Figura 8: T