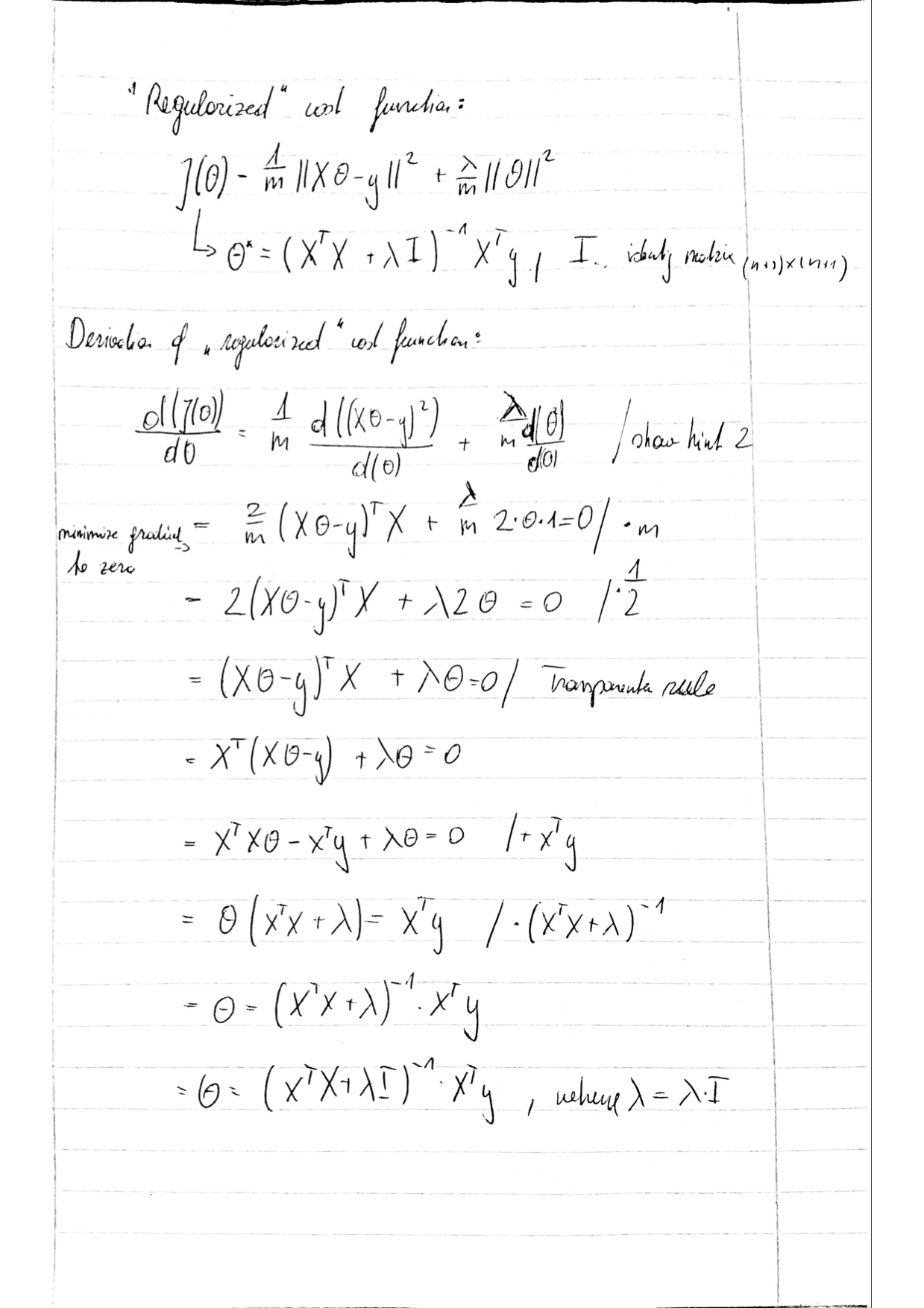
**Assignment 1**

# Computational Intelligence SEW, SS2017

|  |  |  |
| --- | --- | --- |
| **Team Members** | | |
| Last name | First name | Matriculation Number |
| Guggi | Simon |  |
| Papst | Stefan |  |
| Perkonigg | Michelle | 1430153 |

# Derivation of Regularized Linear Regression

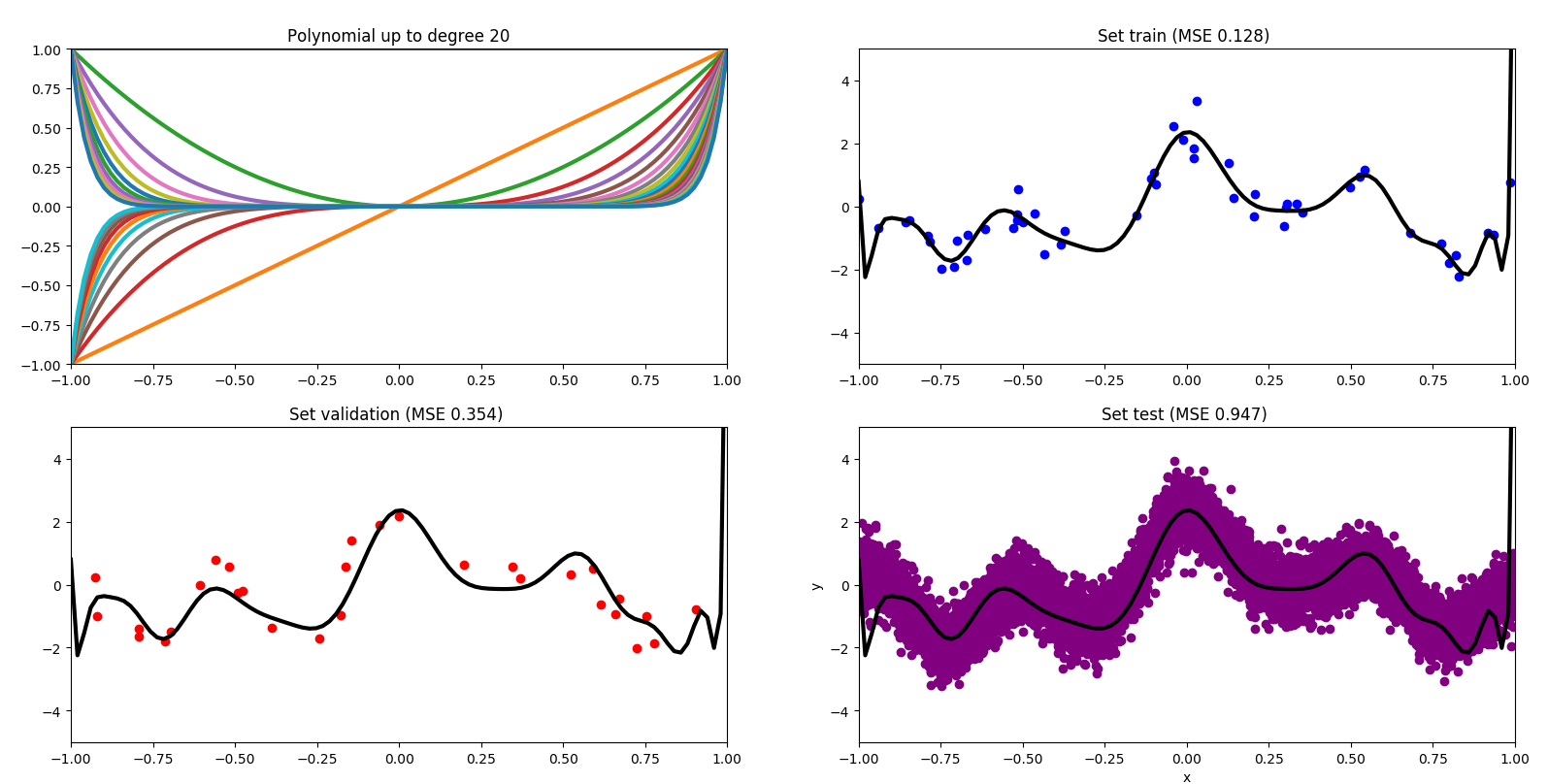


# Plot for each of the following polynomial degrees: 1, 2, 5, 20

### Polynomial degree 1

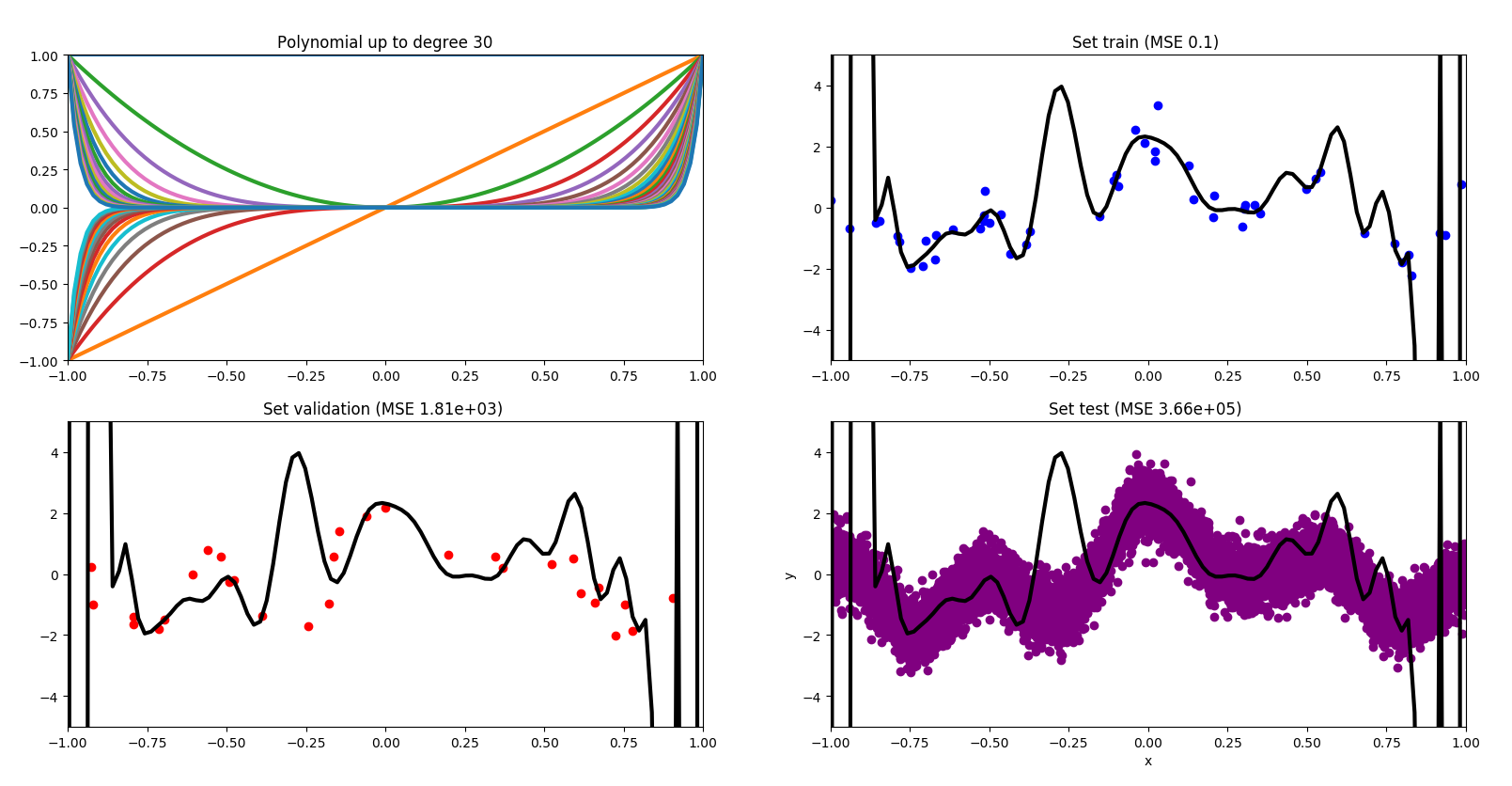
### Polynomial degree 2

### Polynomial degree 5

Polynomial degree 20

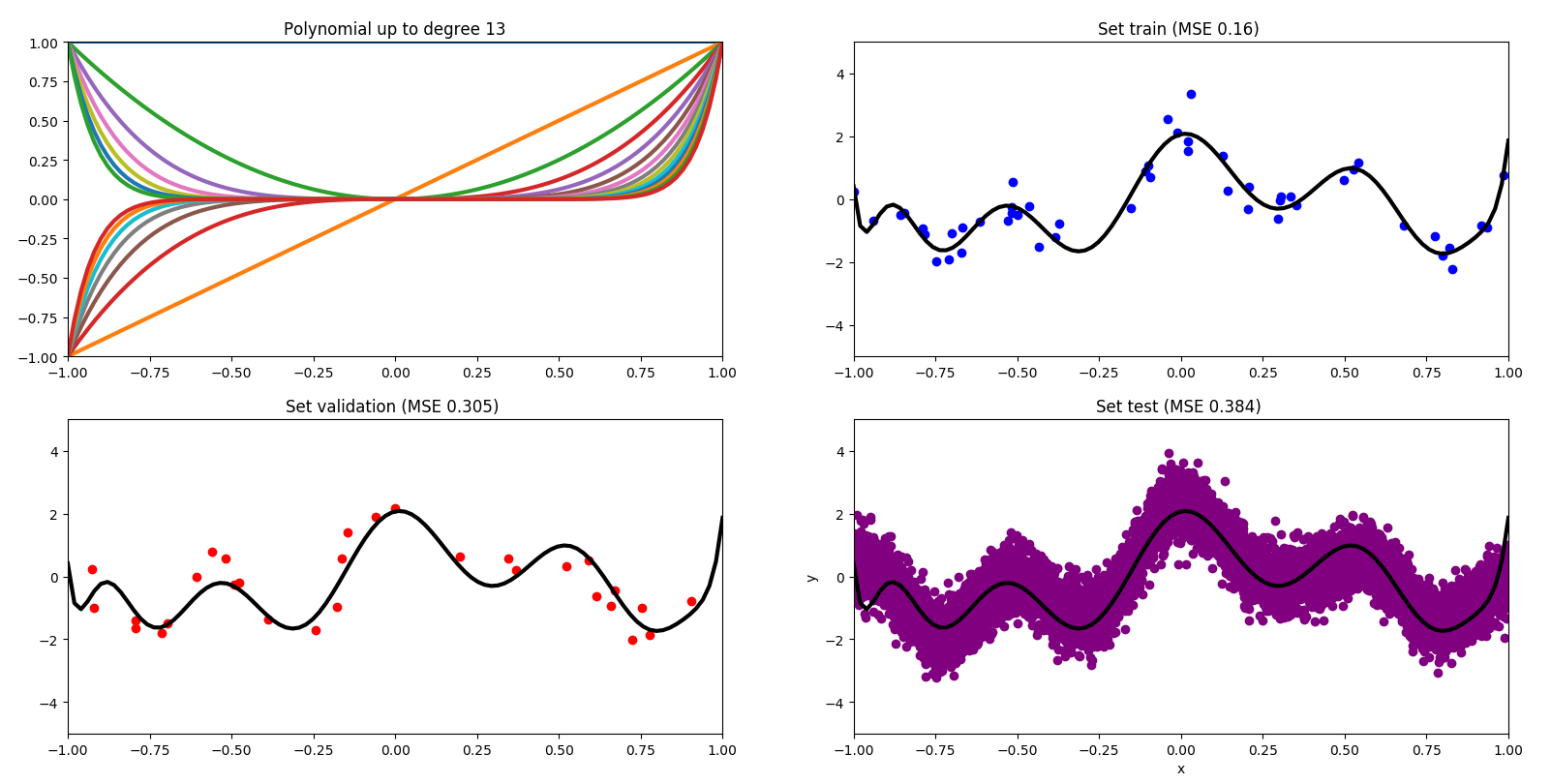
# Plot of polynomial degree with lowest training error

The polynomial degree 30 has the lowest training error.   
Training error: 0.10026711145129742  
Test error: 366141.23594091967

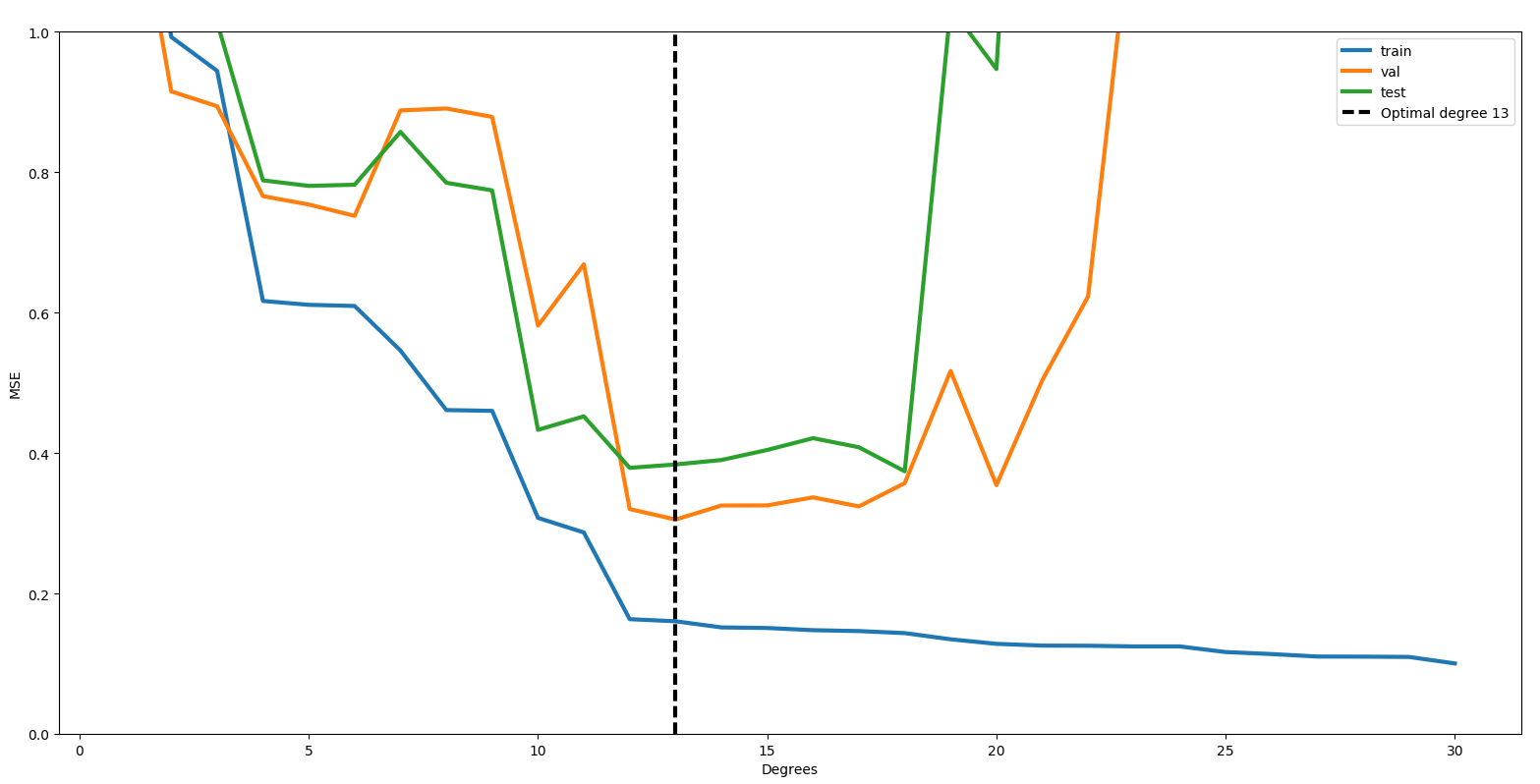


# Plot of polynomial degree with lowest validation error

The polynomial degree 13 has the lowest validation error.   
Validation error: 0.305228632706  
Testing error: 0.3837016046508086



# Plot of training, validation and testing errors as a function



# Discuss your findings in your own words using the concept of over-fitting. Why is it important to use a validation set?

*Overfitting:* When a model is overfitted it is too complex, because of too many parameters. Overfitting leads to a low training error but a very high testing error.

|  |  |  |
| --- | --- | --- |
|  | polynomial degree 13 | polynomial degree 30 |
| training error | 0.1603029754981785 | 0.1002671114512974 |
| validation error | 0.3052286327055114 | 1805.9699422675815 |
| testing error | 0.3837016046508086 | 366141.23594091967 |

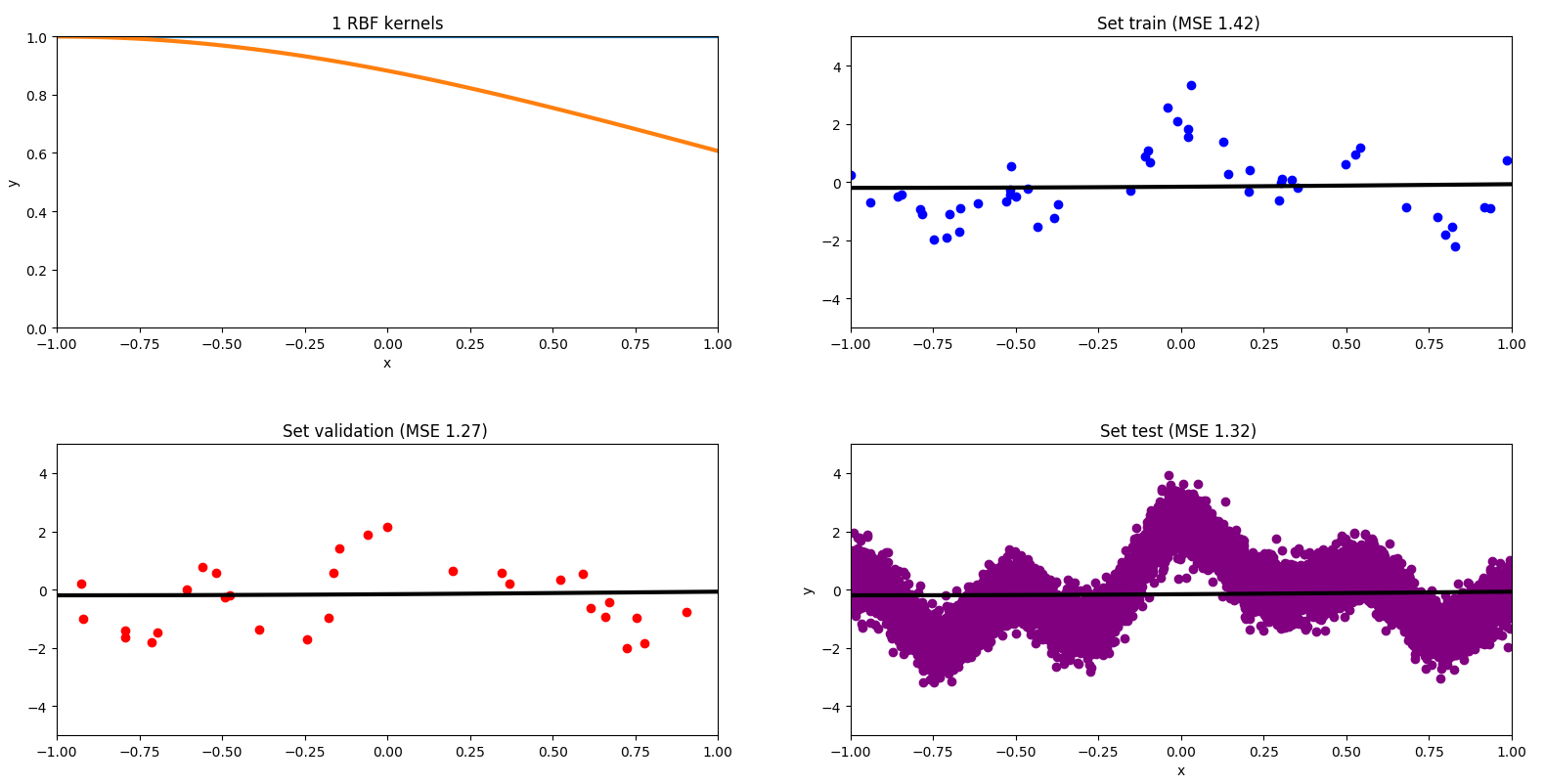
The table shows a comparison between a model with the lowest training error and a model with the lowest validation error. As you can see the results differ significantly. While the training error of the polynomial degree 30 is very small, the validation and testing errors are extremely high. This type of model is called as overfitted. The polynomial degree 13 shows a moderate training error, but a very small validation error. The validation set is used to avoid overfitting. The right fitted model is the one with the smallest testing error.  
   
The plot of the training, validation and testing errors as a function shows that the blue line, the training errors have its lowest point at the polynomial degree 30. The orange line, the validation error has its lowest point at the polynomial degree 13 and the green line, the testing errors have its lowest point at the polynomial degree 18.

# Derivation of Gradient

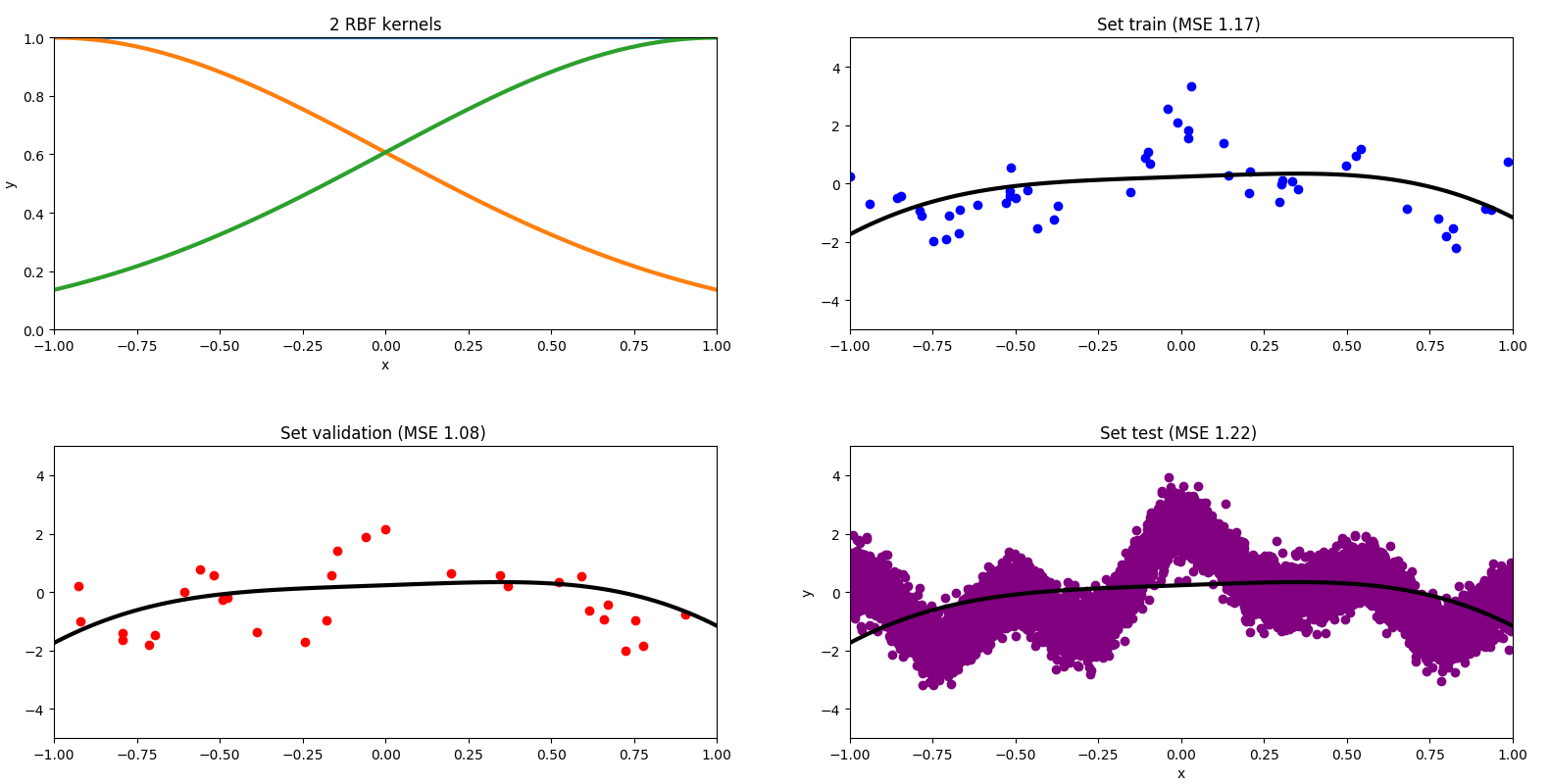
# 2.png

# Plot for each of the following degrees: 1, 2, 5, 20

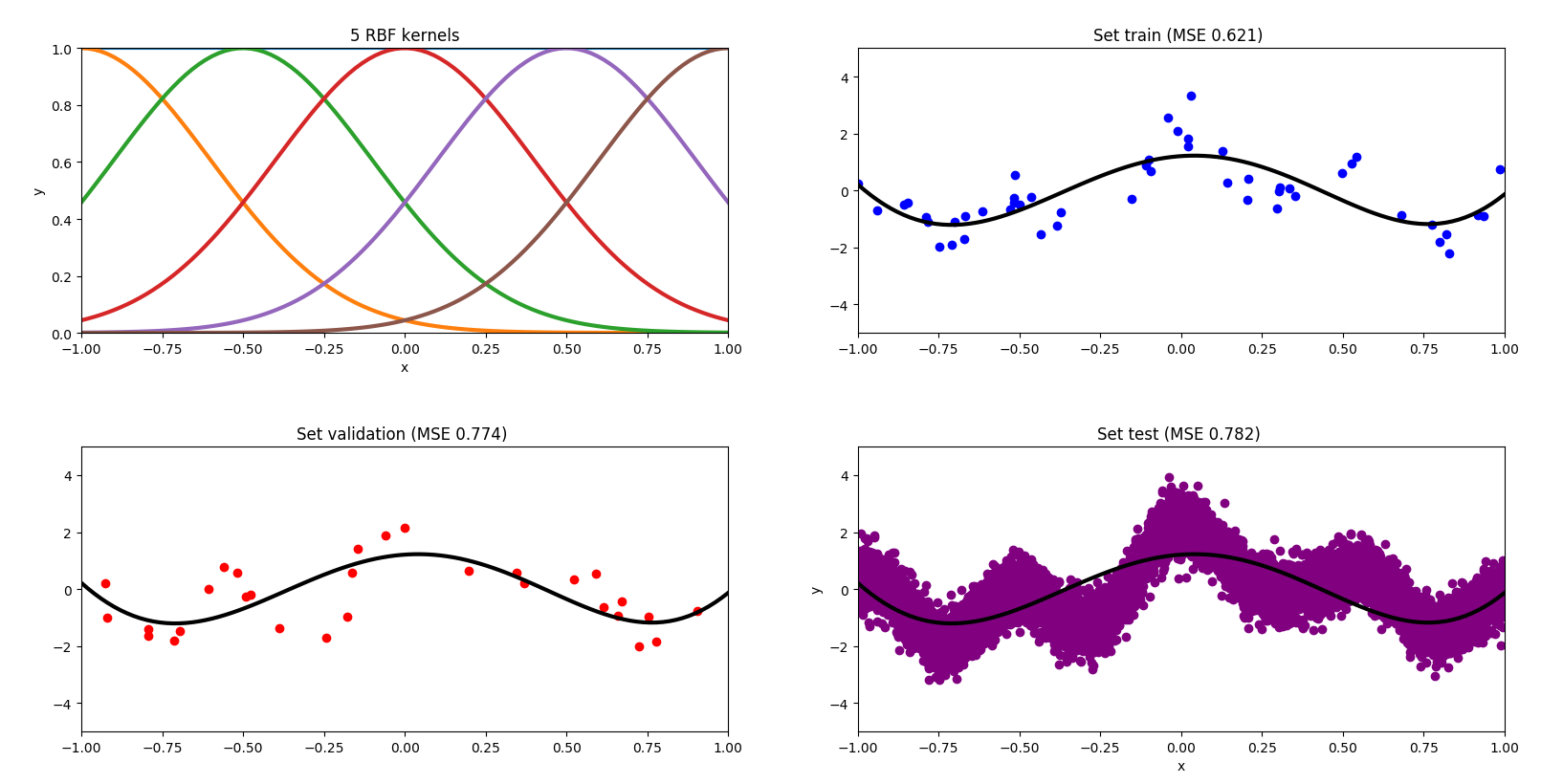
### degree 1



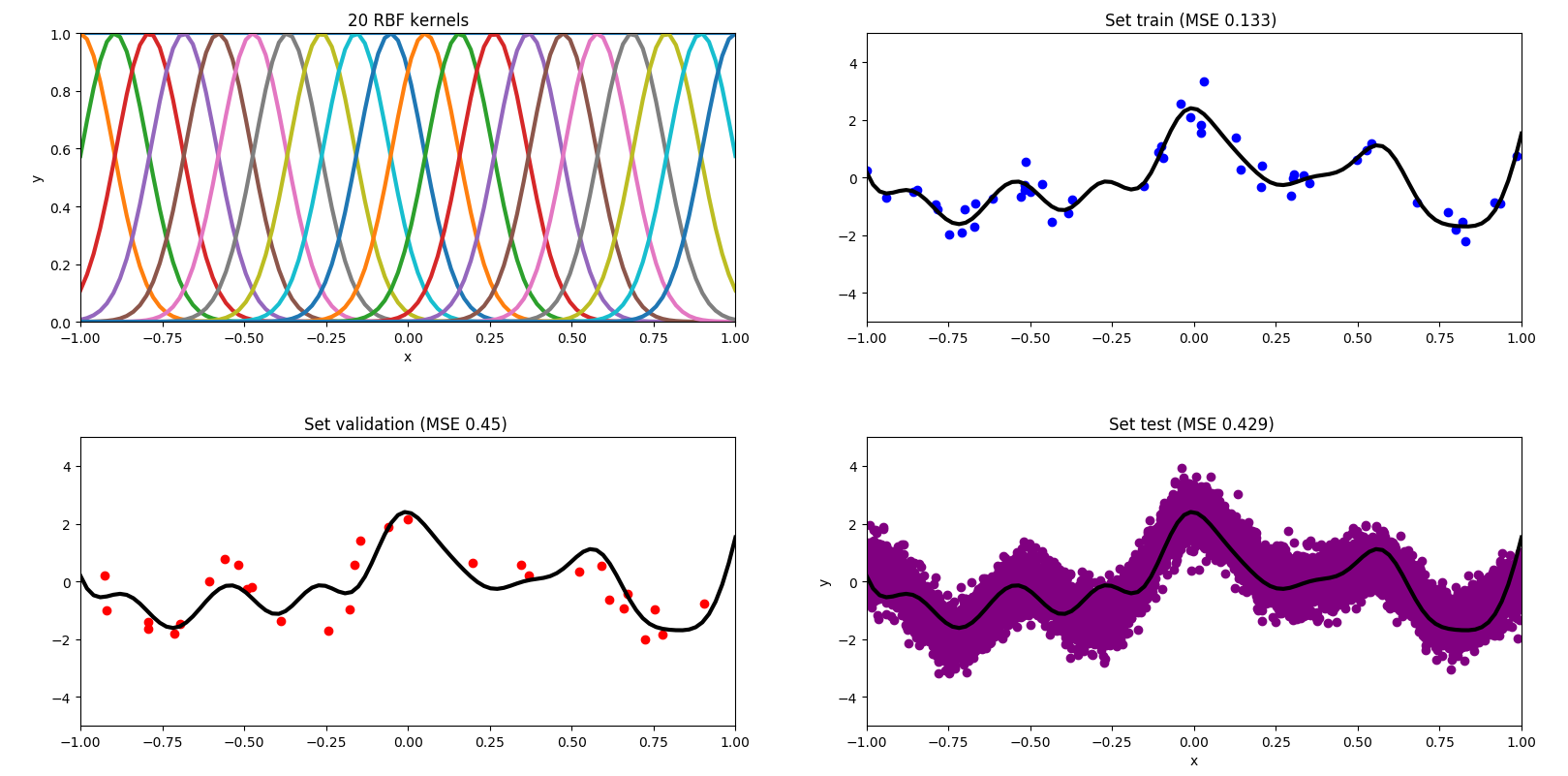
### degree 2



### degree 5

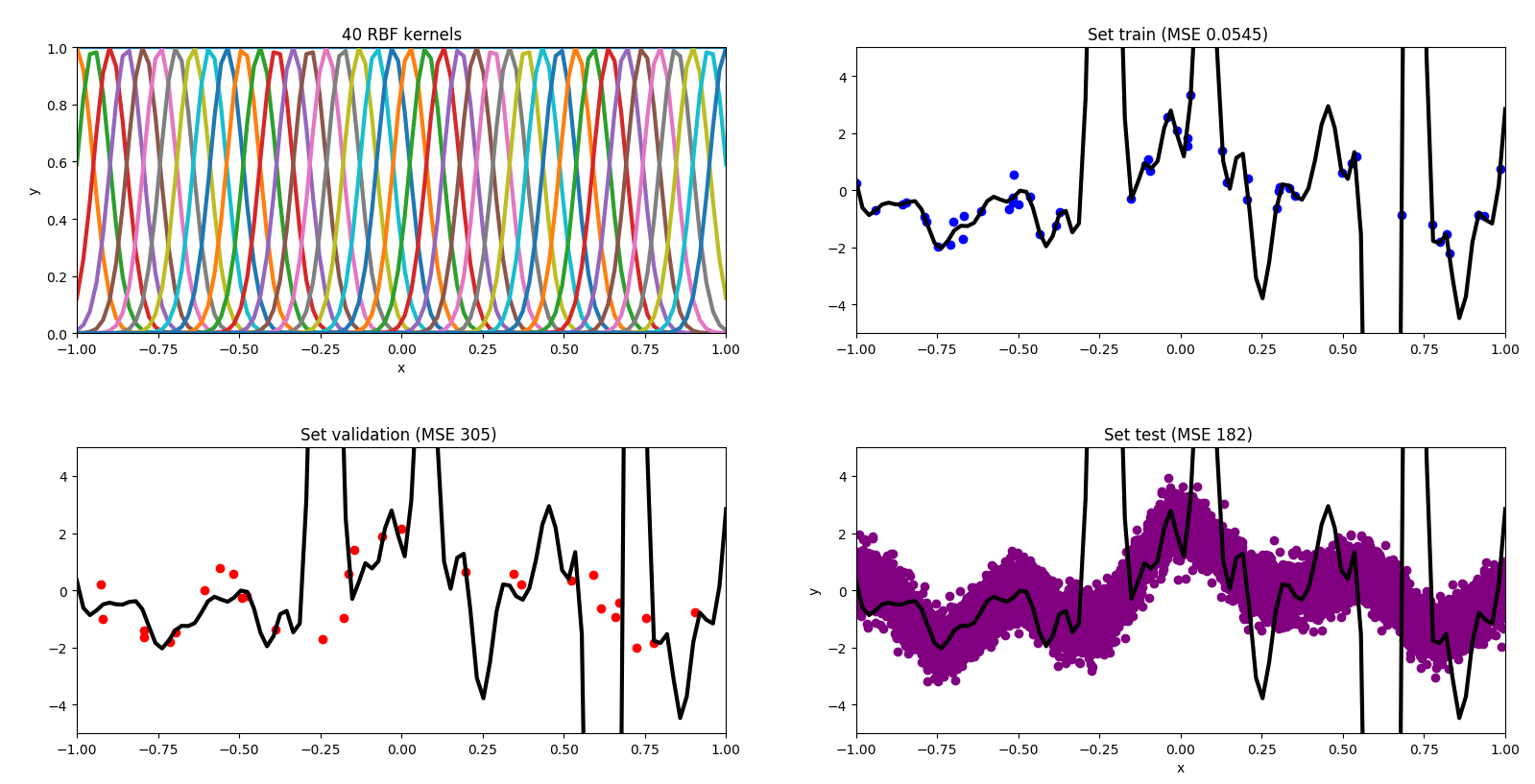


### degree 20



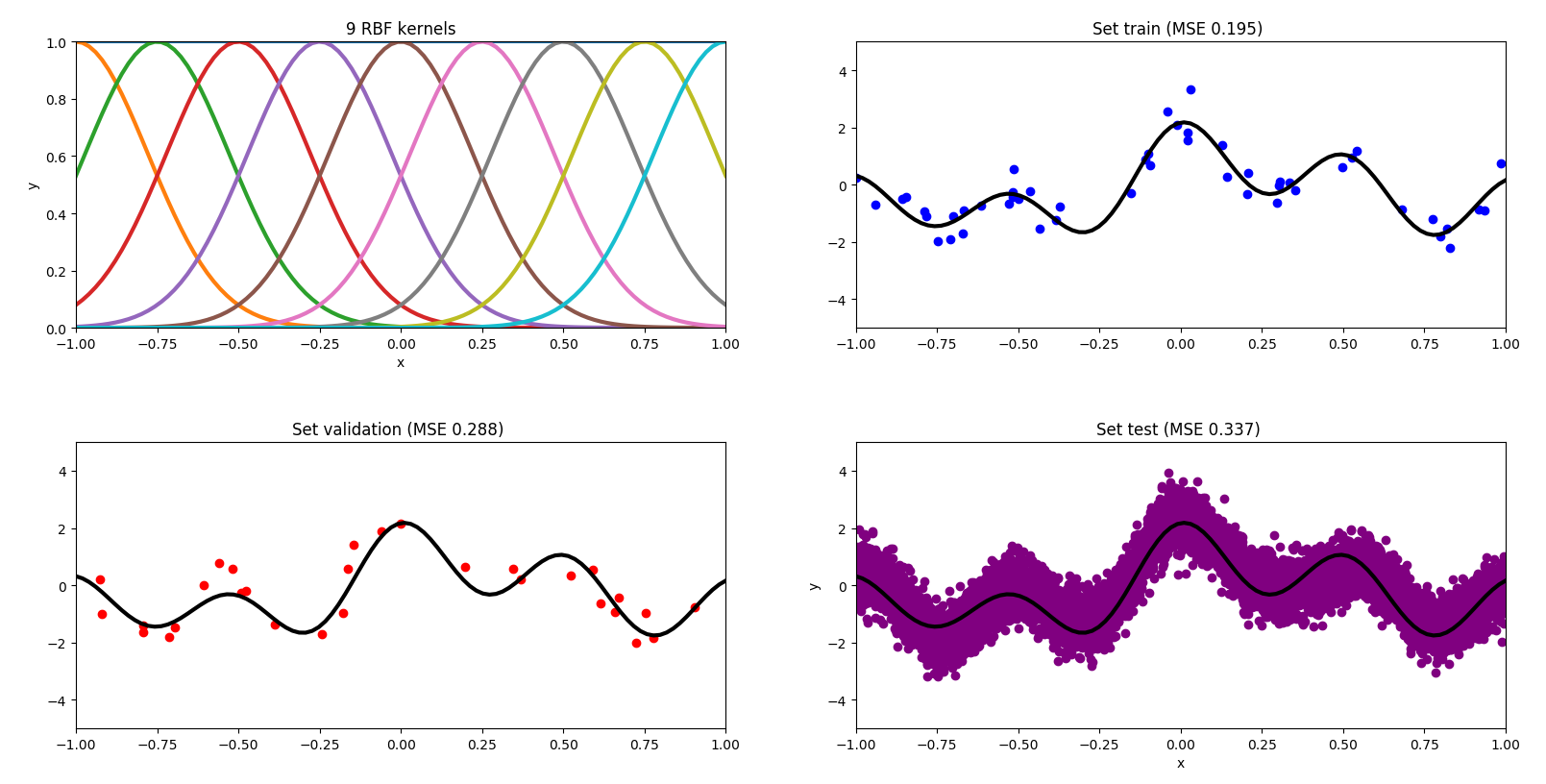
# Plot of RBF with lowest training error

The RBF 40 has the lowest training error.   
Training error: 0.054452916301051674  
Testing error: 181.67182614369543

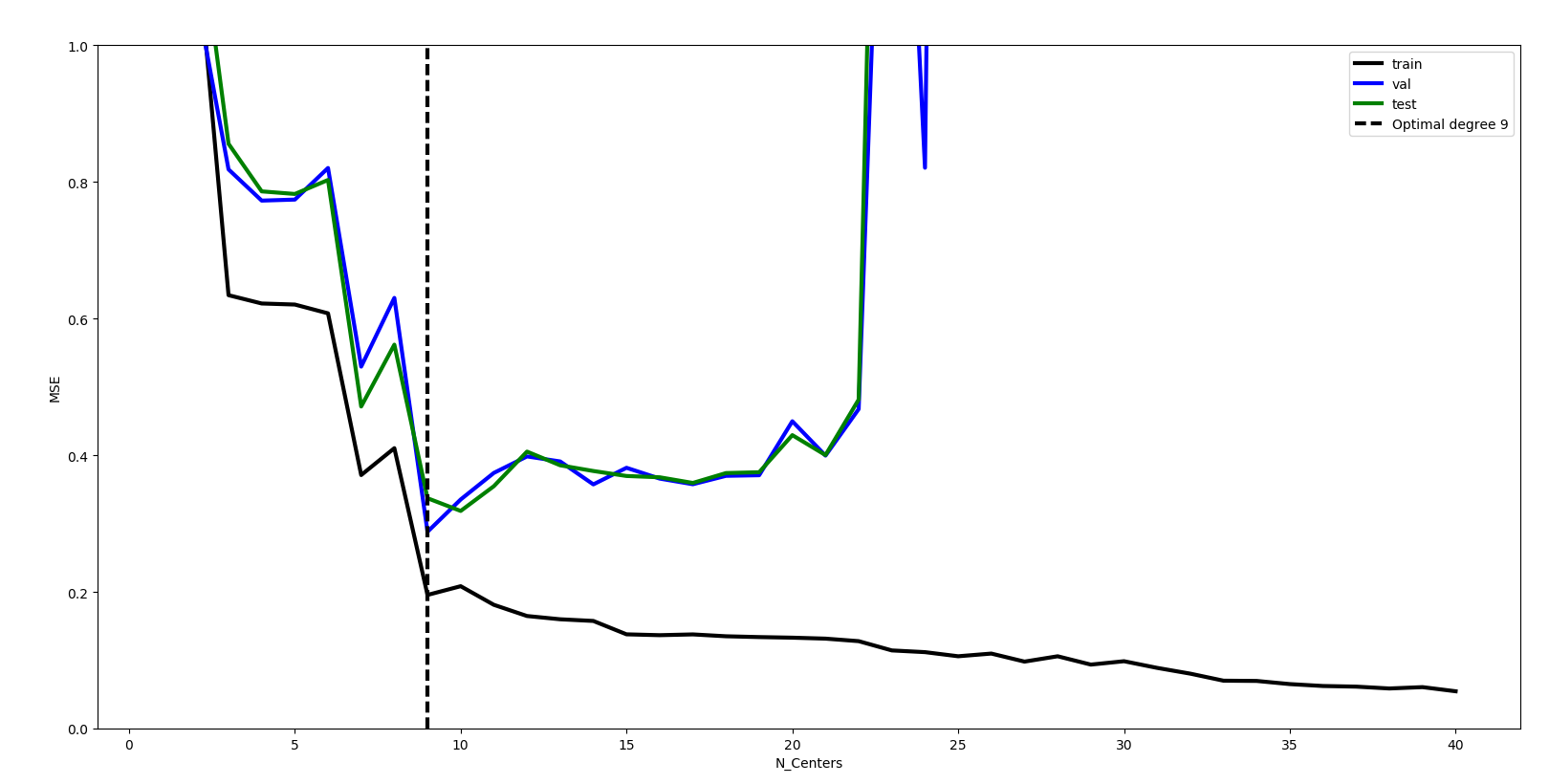


# Plot of RBF with lowest validation error

The RBF 9 has the lowest validation error.   
Validation error: 0.2879598418721303  
Testing error: 0.3370130082408977



# Plot of training, validation and testing errors as a function



# Briefly describe and discuss your findings in your own words. Is the polynomial or the RBF model better?

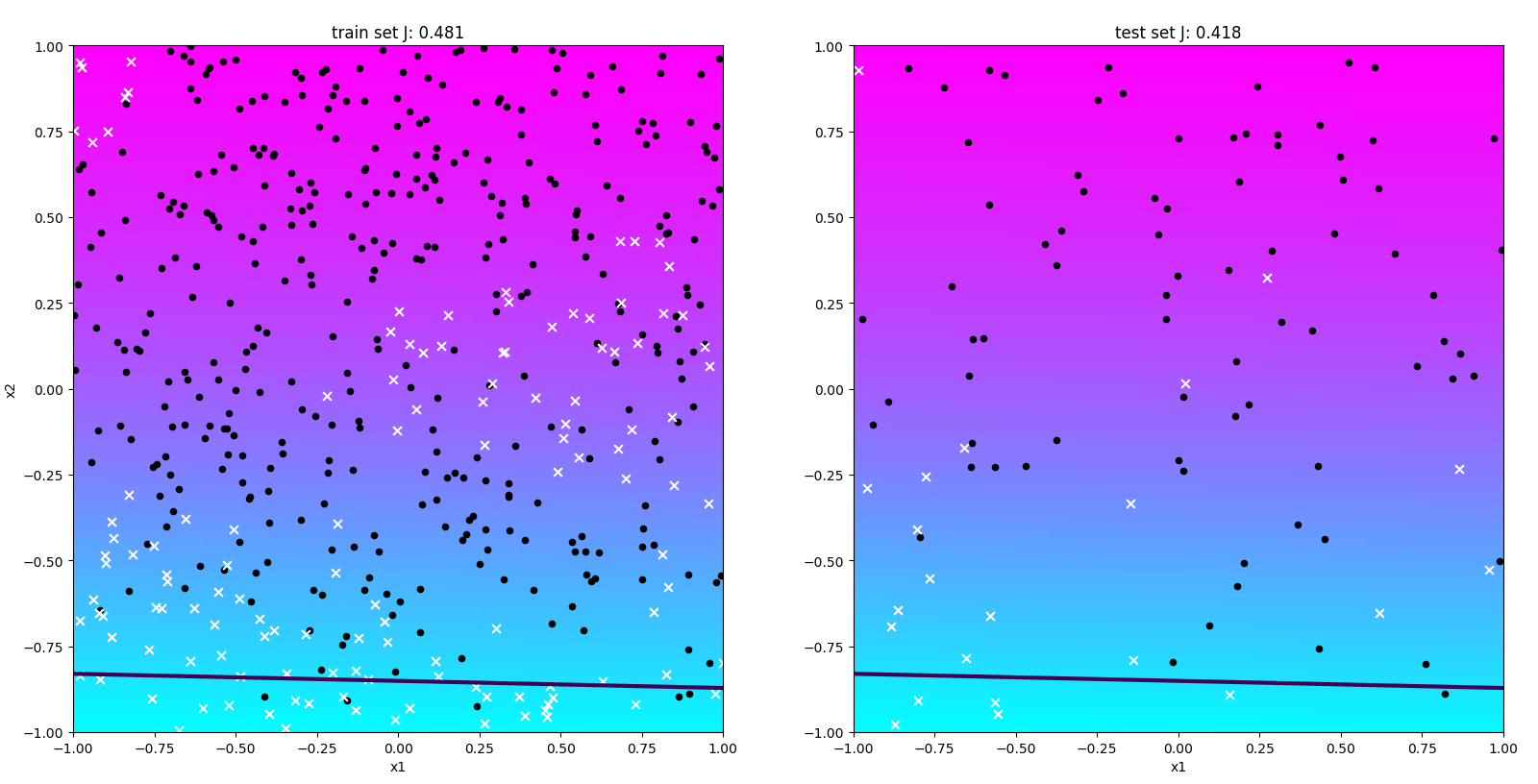
The RBF with the lowest training error is, as with the polynomial model, the highest degree tested (degree 40). The testing error is again very high and therefore the RBF is overfitted, because of the high number of parameters. The lowest validation error achieved the RBF with degree 9. This one has a significant lower testing error than the RBF with degree 40. In this case the RBF model with the lowest validation error has the same degree as the RBF model with the lowest testing error. Both have degree 9, as you can see in the plot of training, validation and testing errors as a function. The black line represents the training errors, the blue line the validation errors and the green line the testing errors.

The RBF model is better than the polynomial model because the validation and testing errors deliver both the RBF model with degree 9 as its best. Therefore, the validation process in model selection has found the best hypothesis for the testing process with the testing set.

# The function check\_gradient in toolbox.py is here to test if your gradient is well computed. Explain what it is doing.

# For degree l = 1 run GD for 20 and 2000 iterations (learning rate \_ = 1, all three parameters initialized at zero). Report training and test errors for each iteration number and plot the decision boundaries. Comment on the results and explain why the number of iterations should be neither too low nor too high.

degree: 1  
eta: 1  
max\_iter: 20



degree: 1  
eta: 1  
max\_iter: 2000

