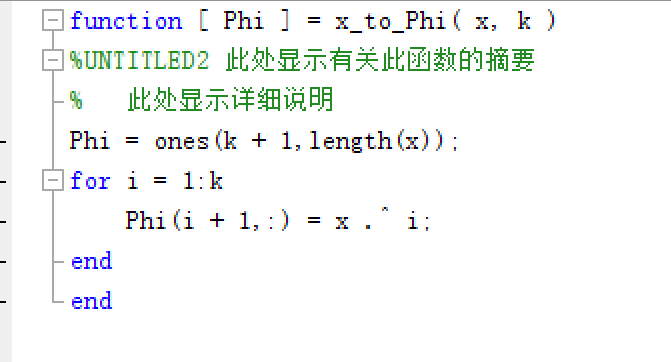
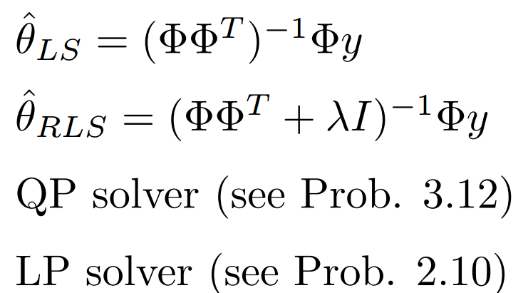
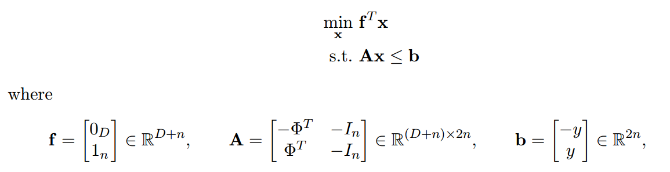
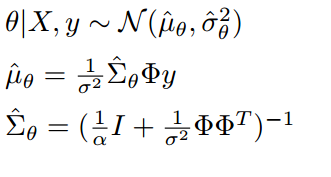
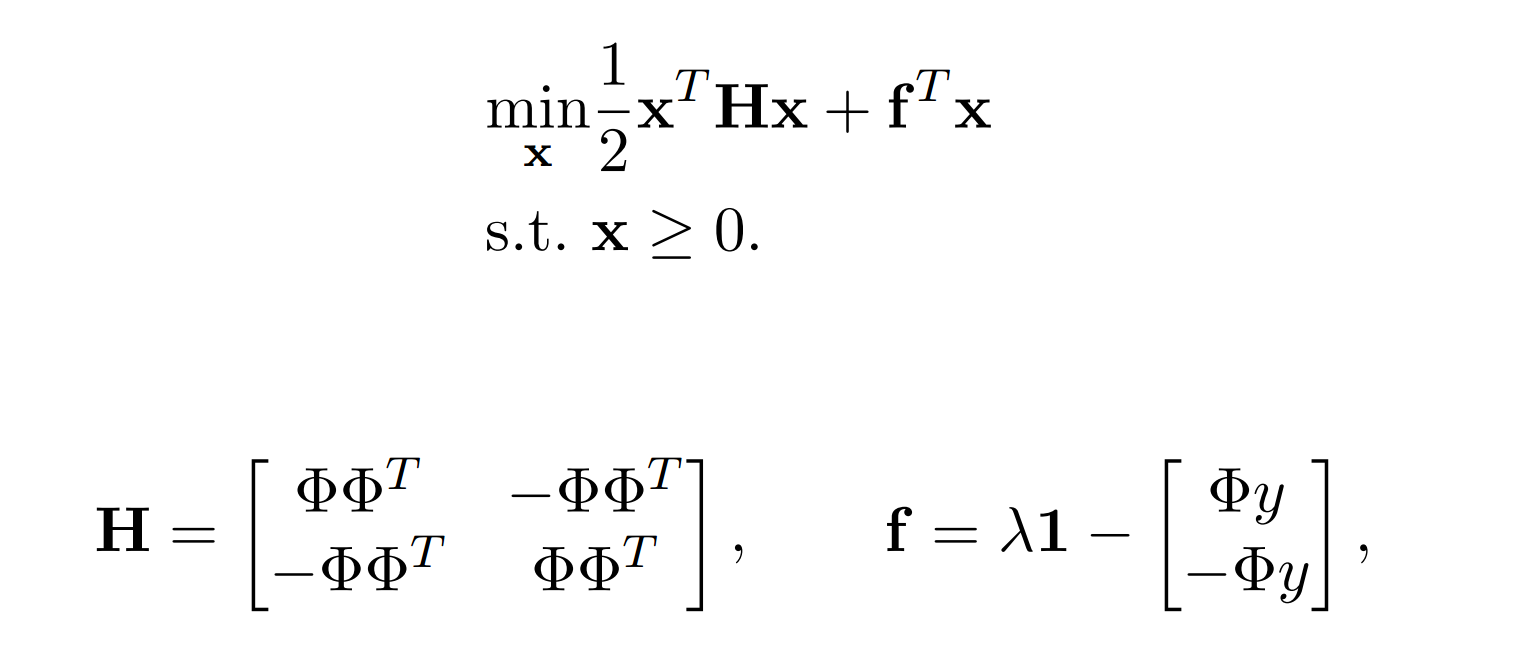
|  |  |
| --- | --- |
|  | 🞂Program Assignment-1  Regression |
|  |  |
|  | **翟冠勋** 🞂54345382 🞂 11/29/2015 |
| Part 1 – Polynomial Function | |
| Part II – Counting People | |

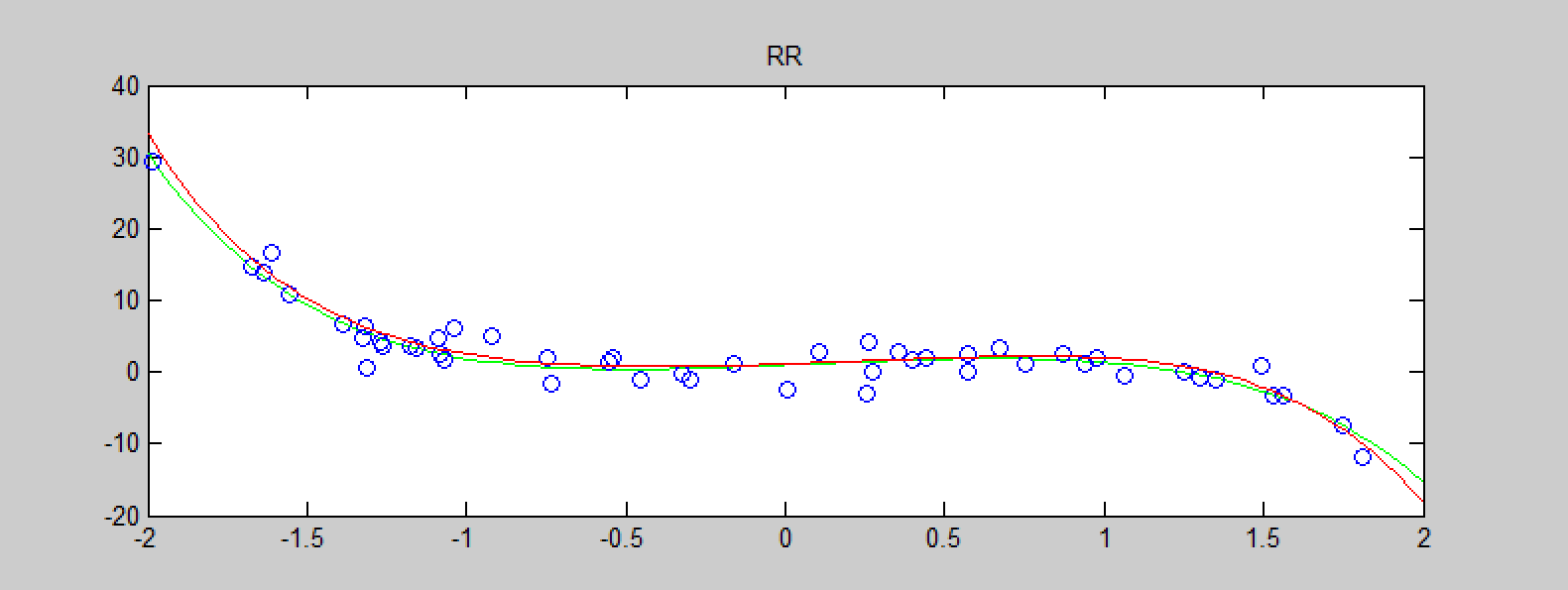
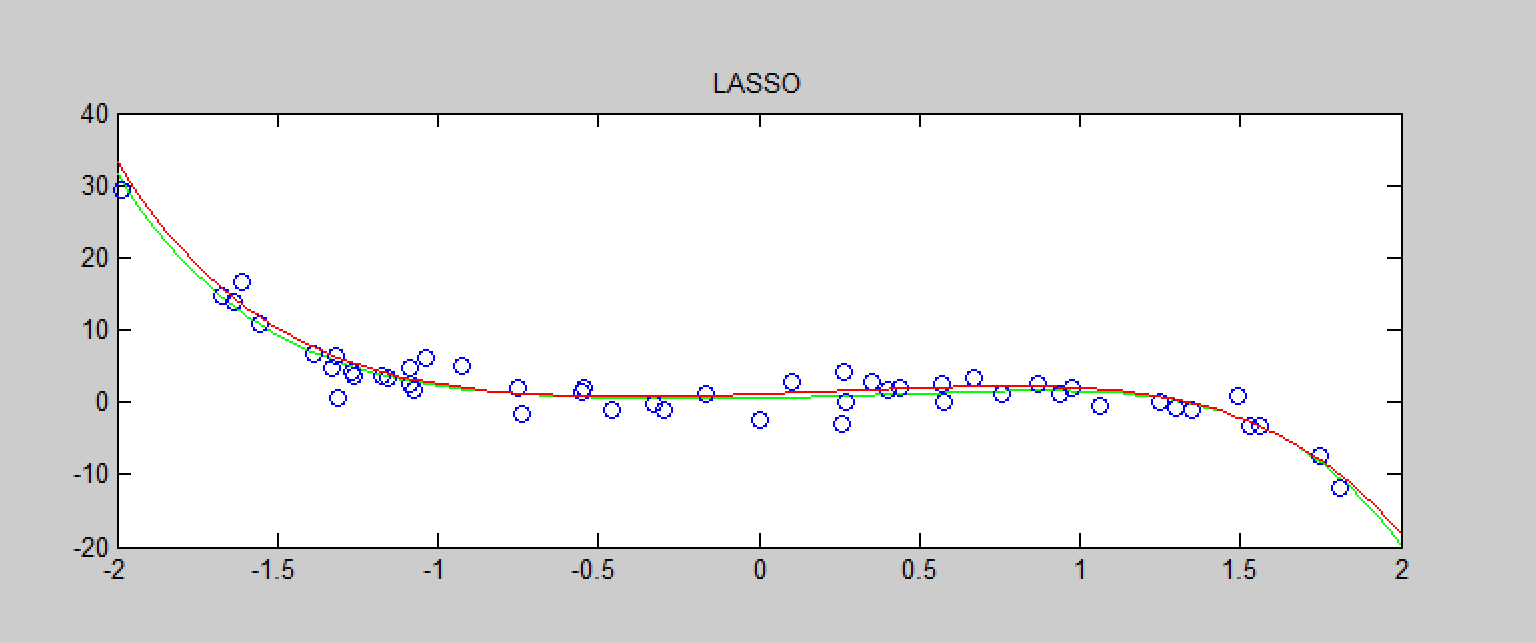
Program Assignment-1

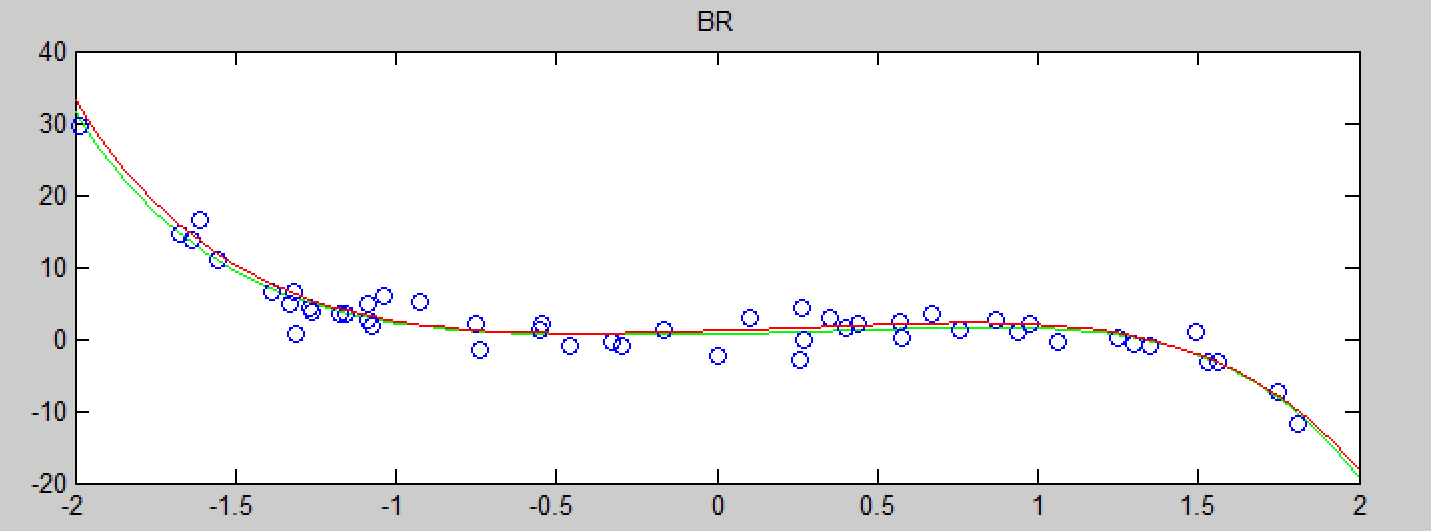
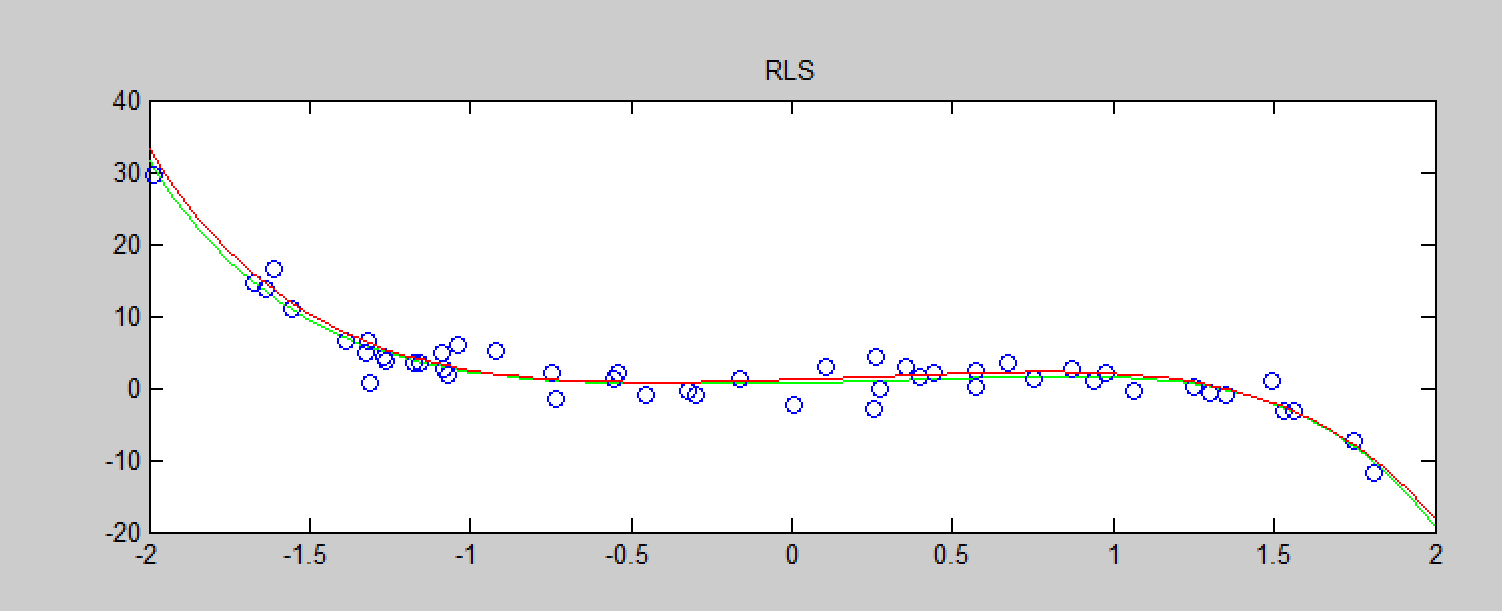
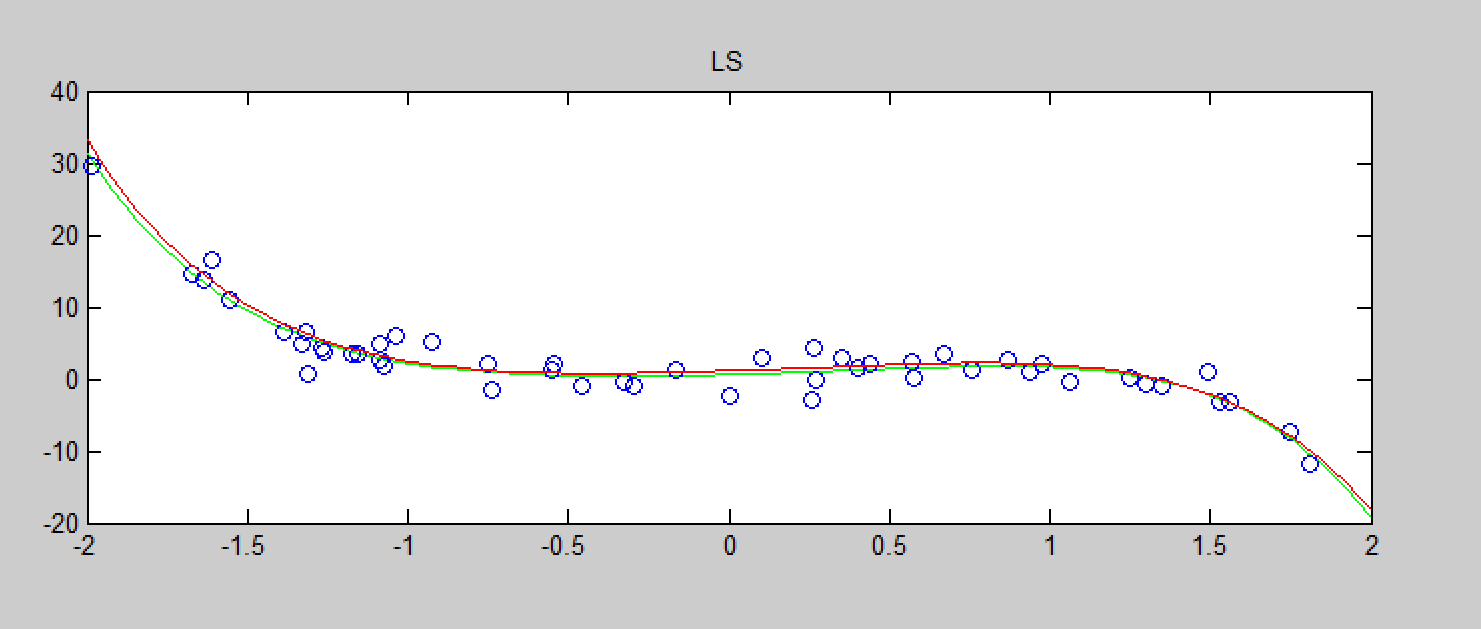
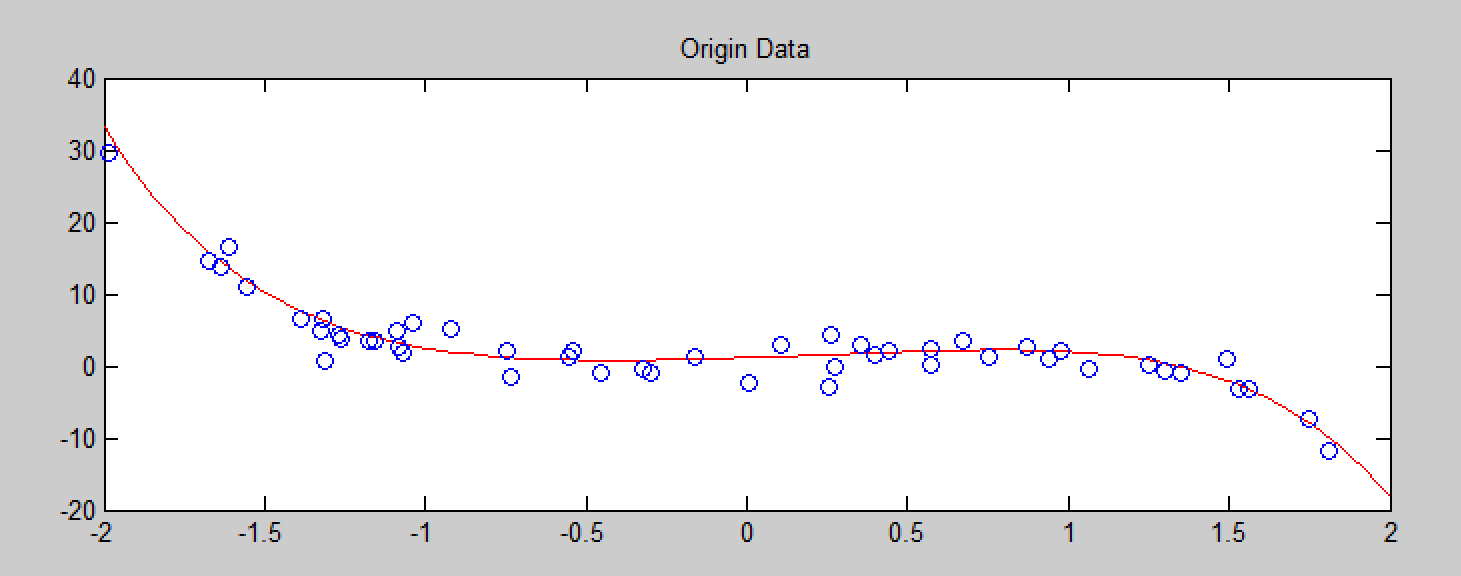
Regression

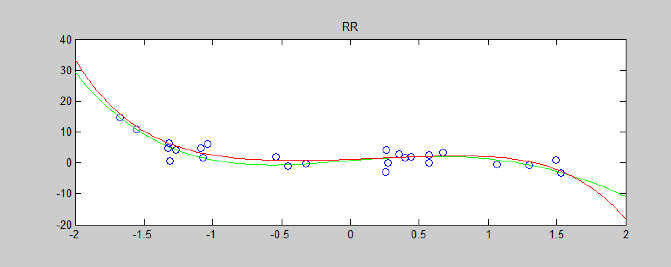
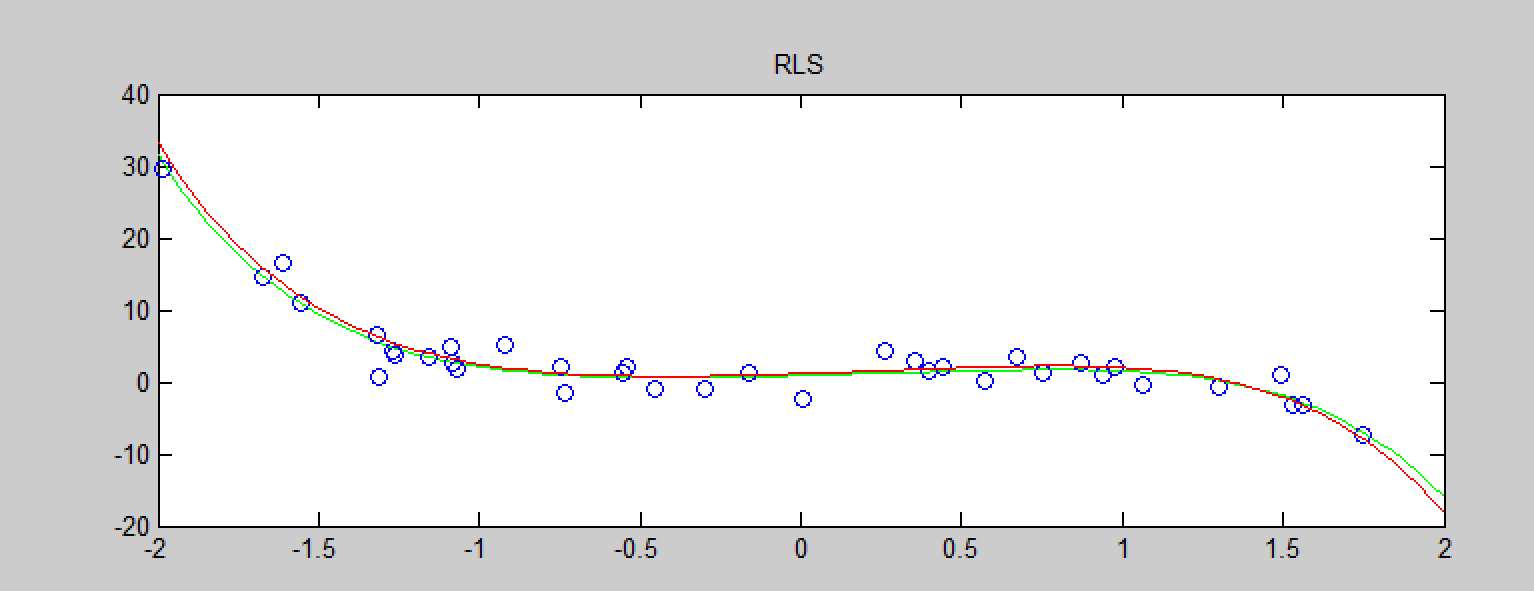
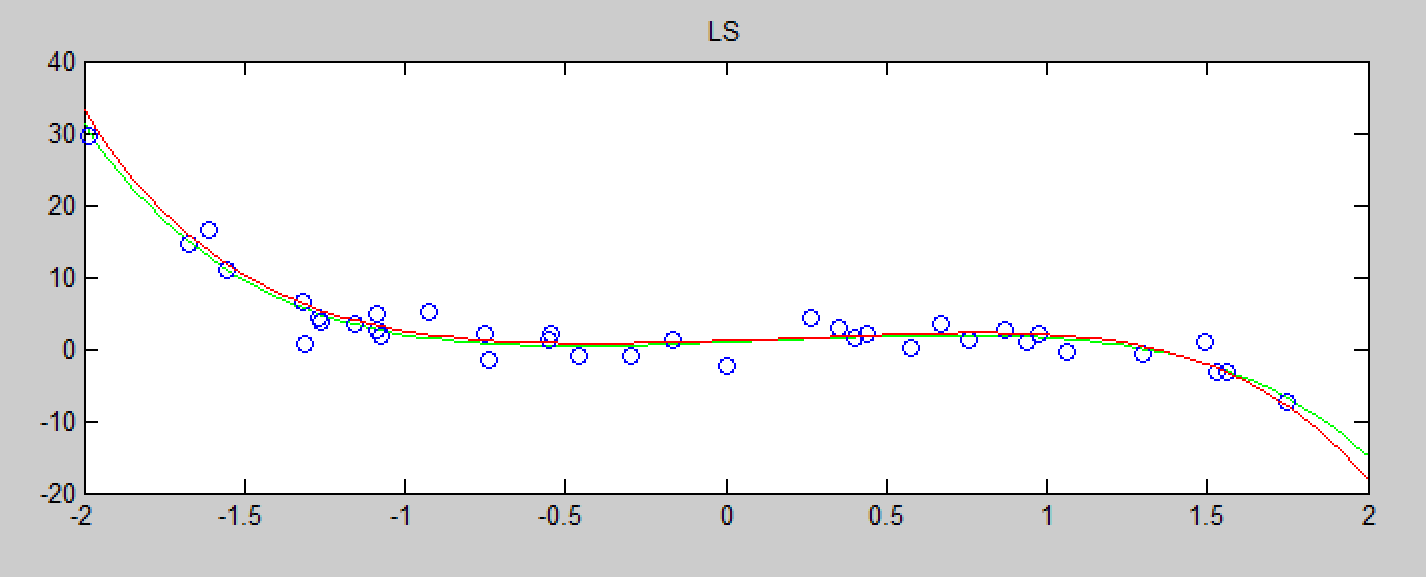
Part I Polynomial Function

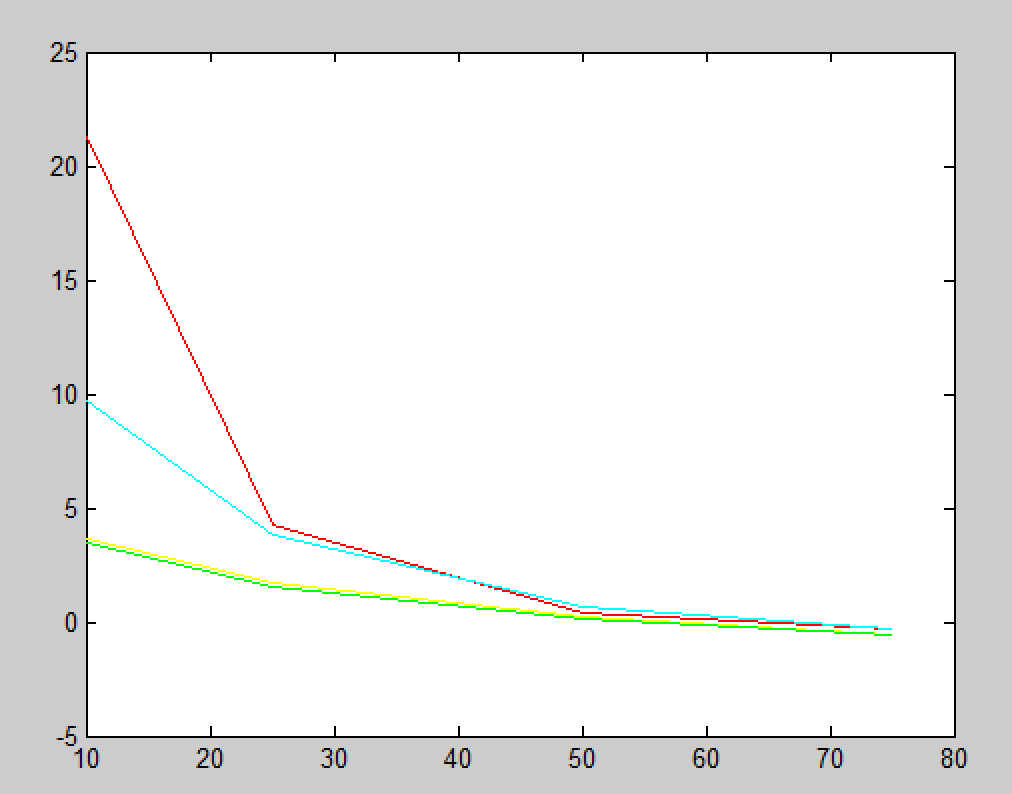
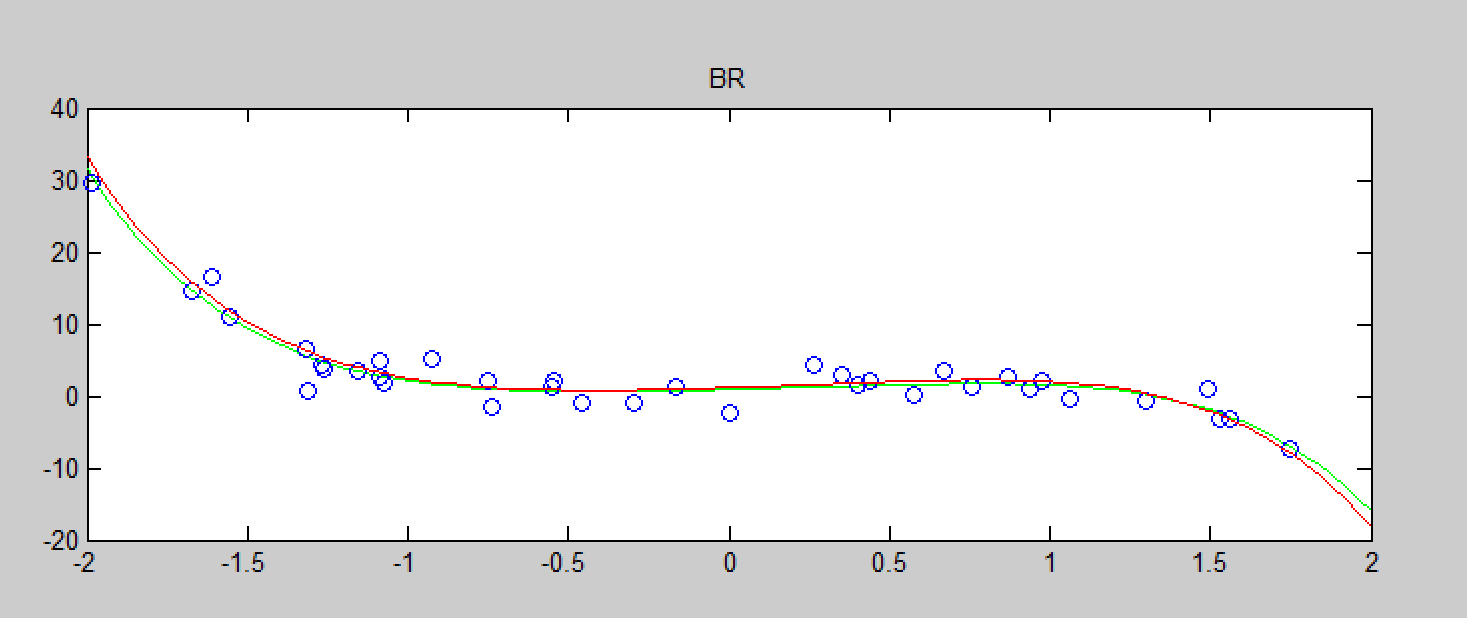
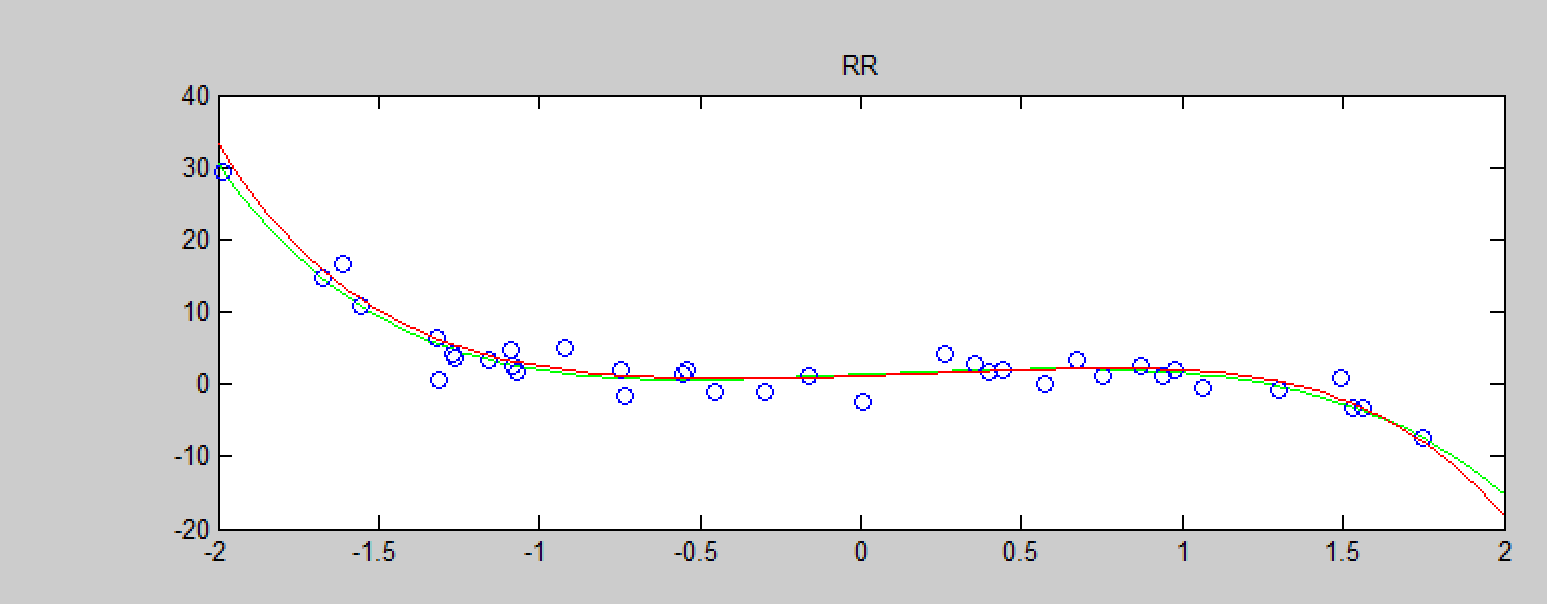
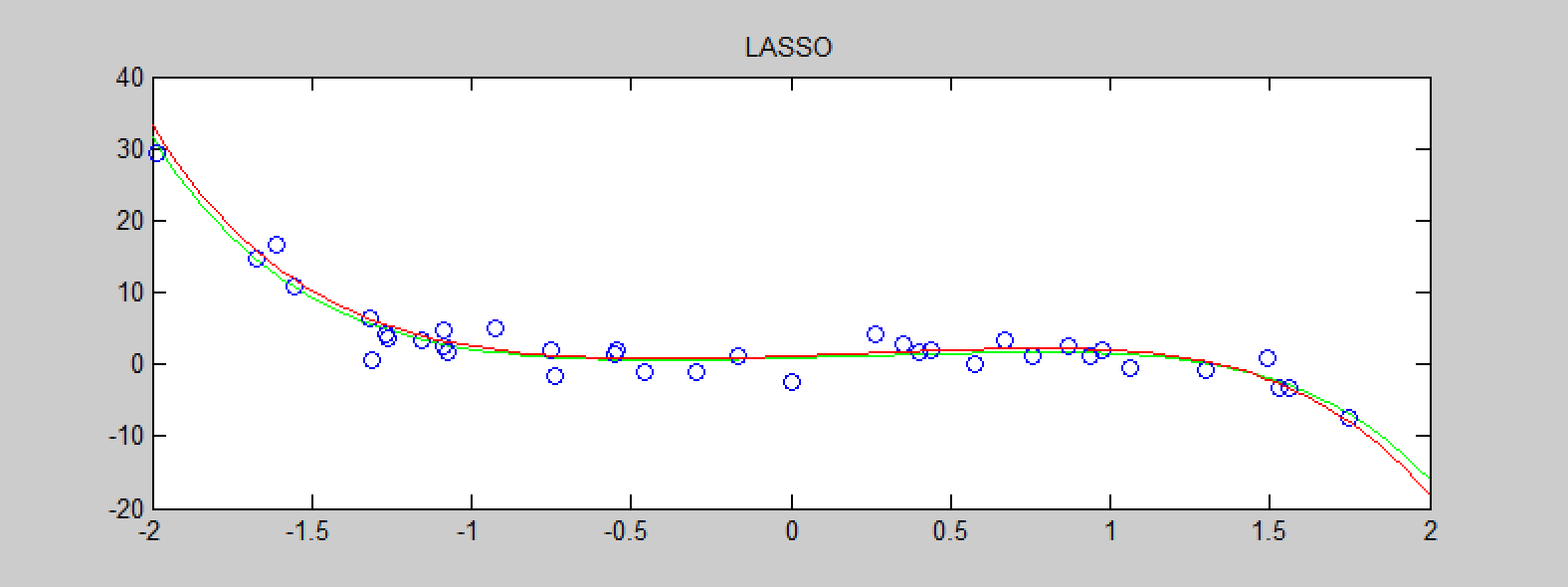
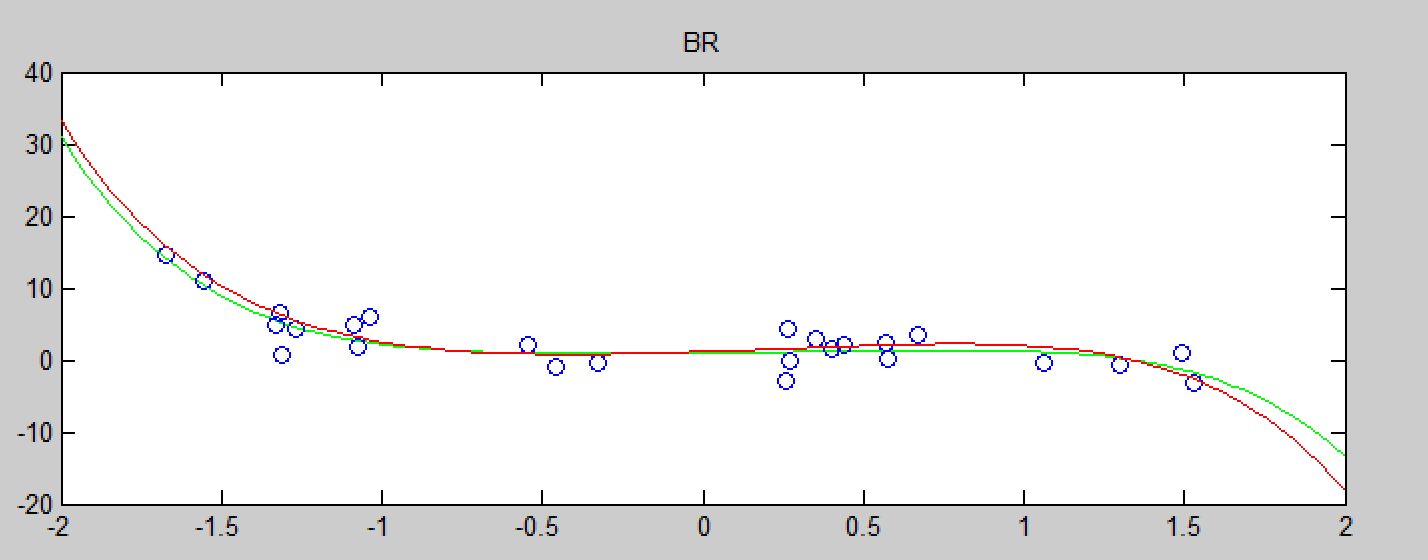
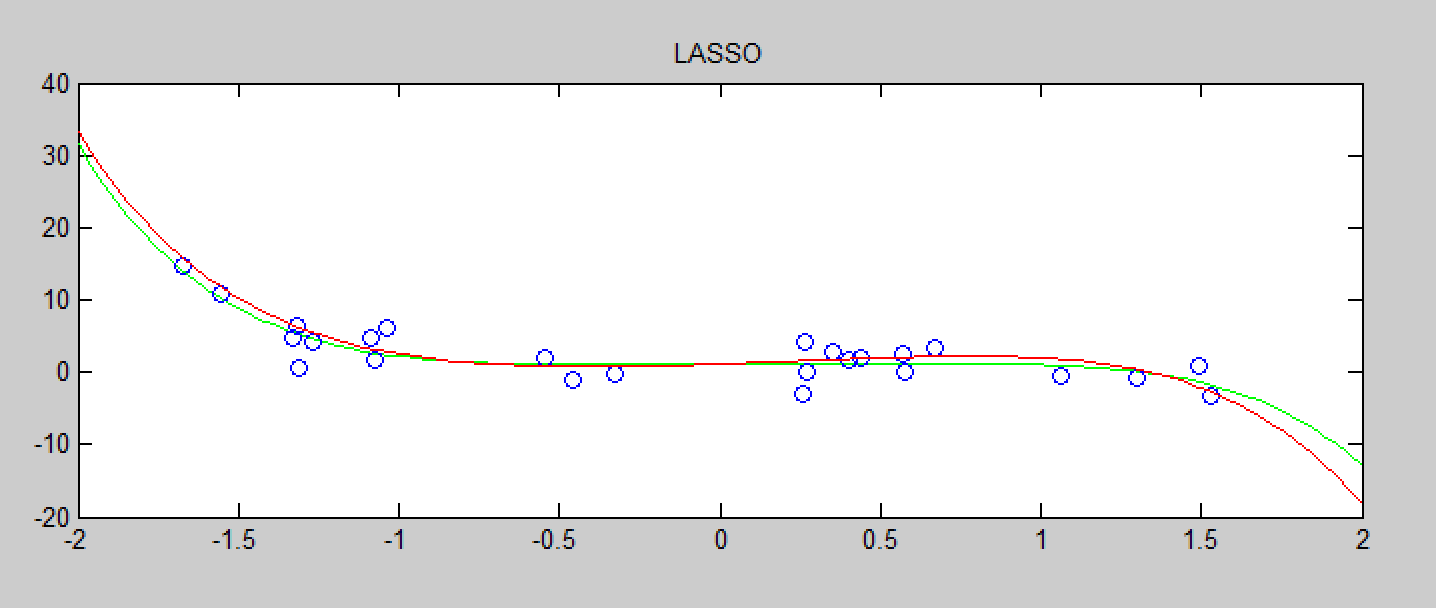
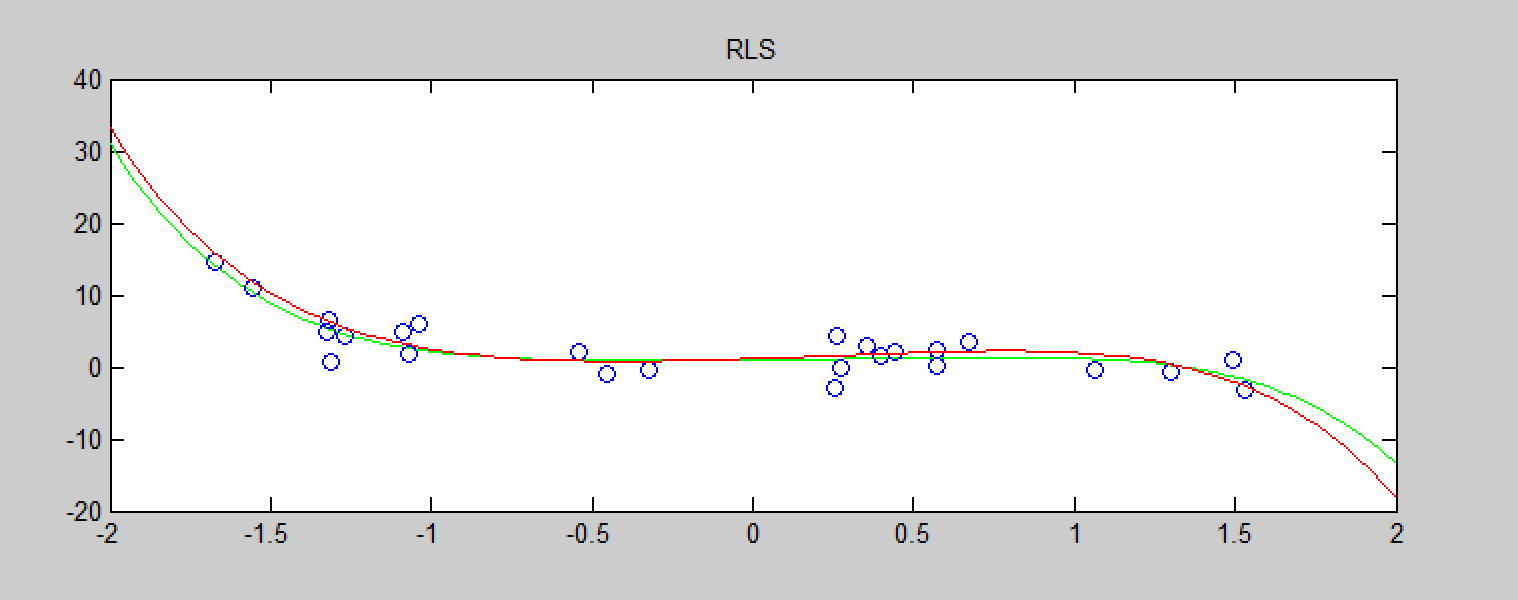
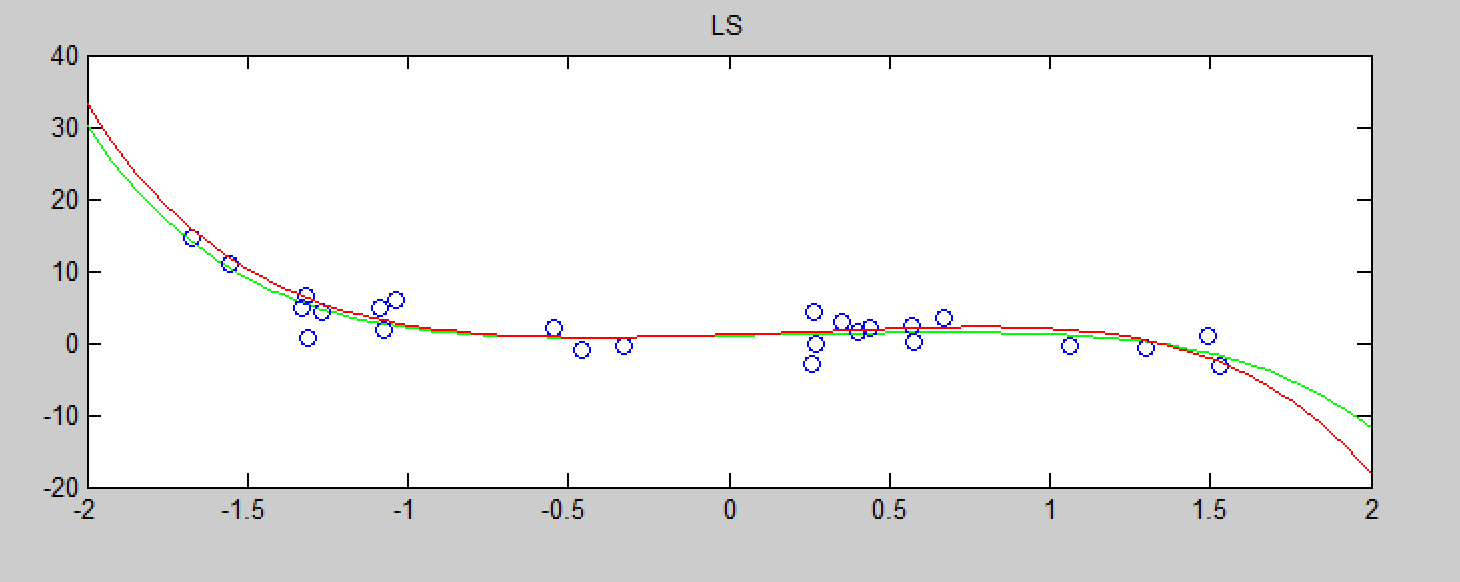
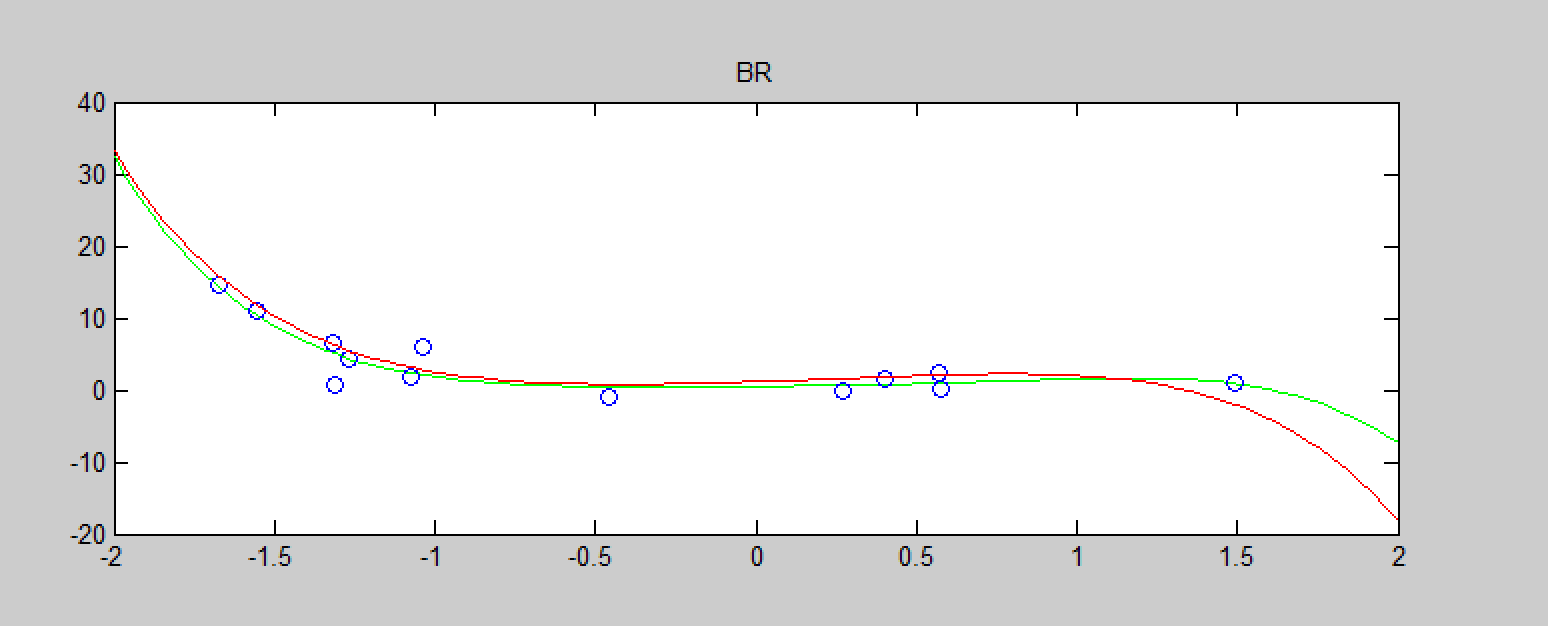
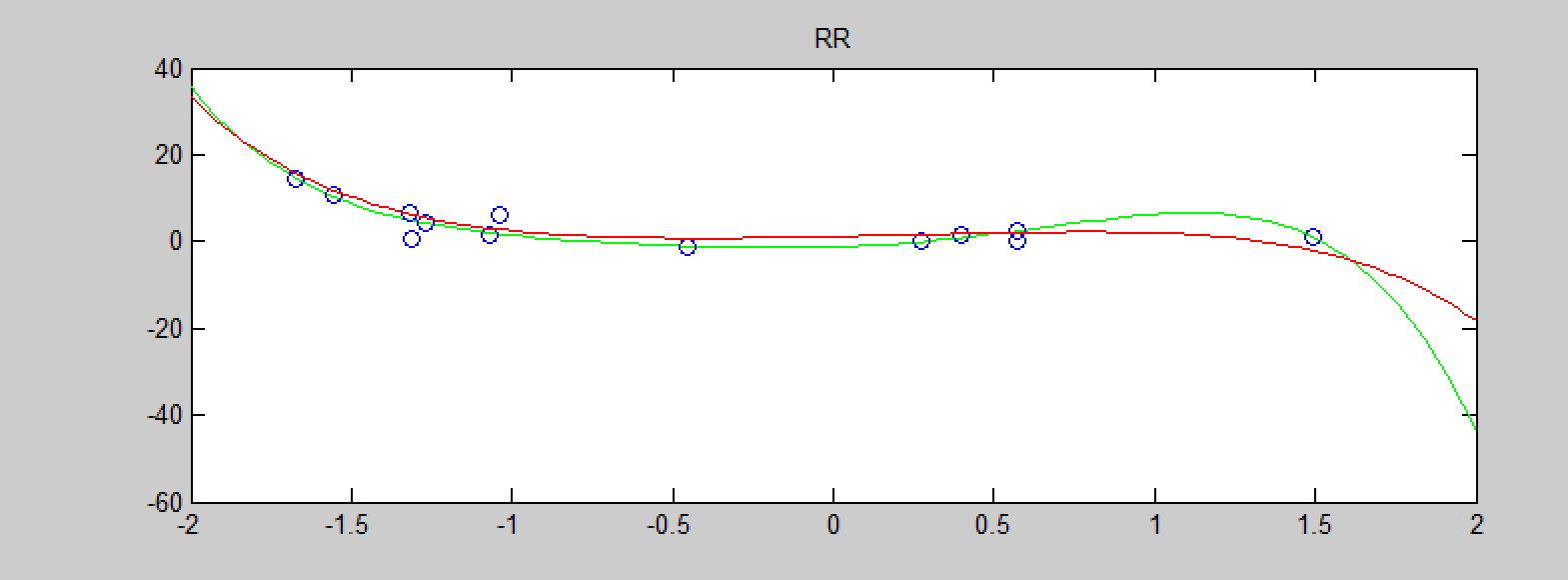
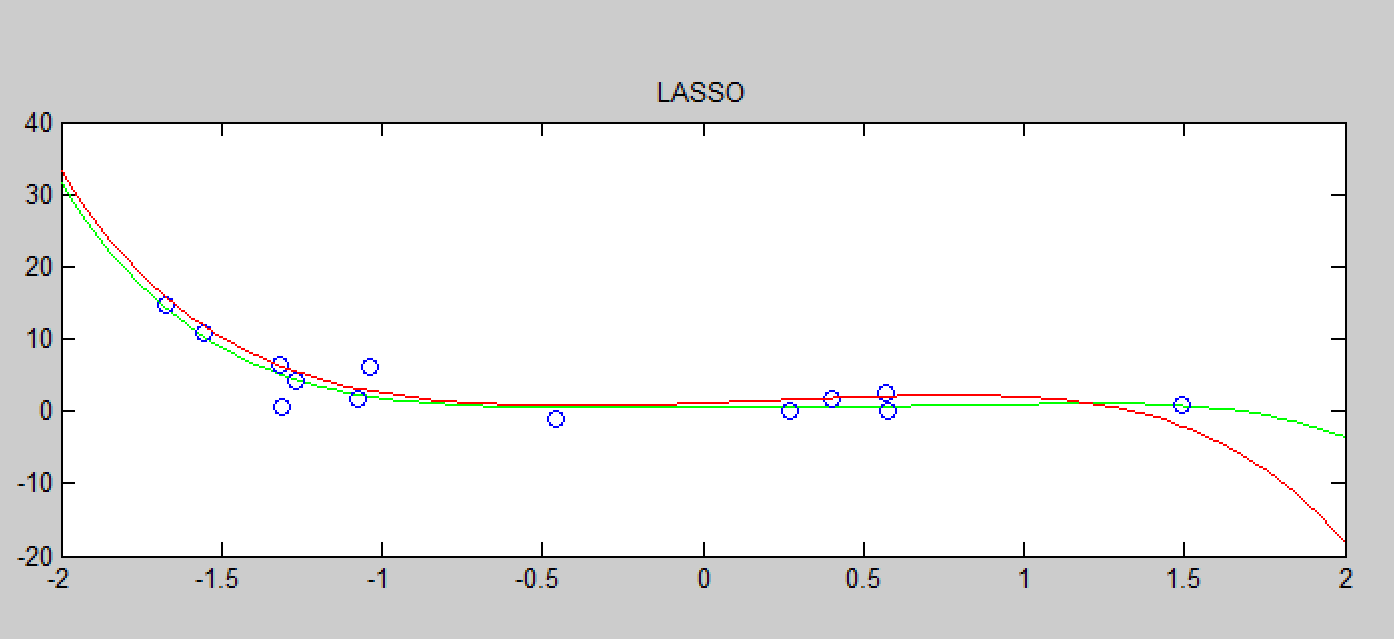
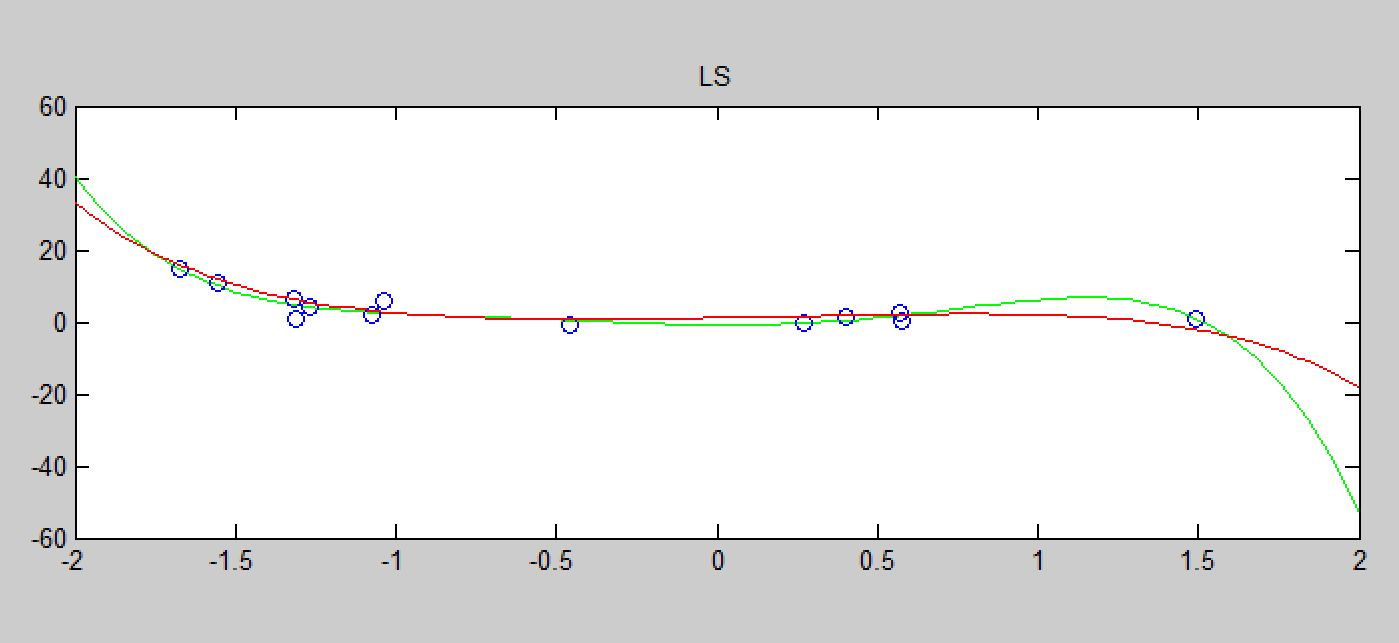
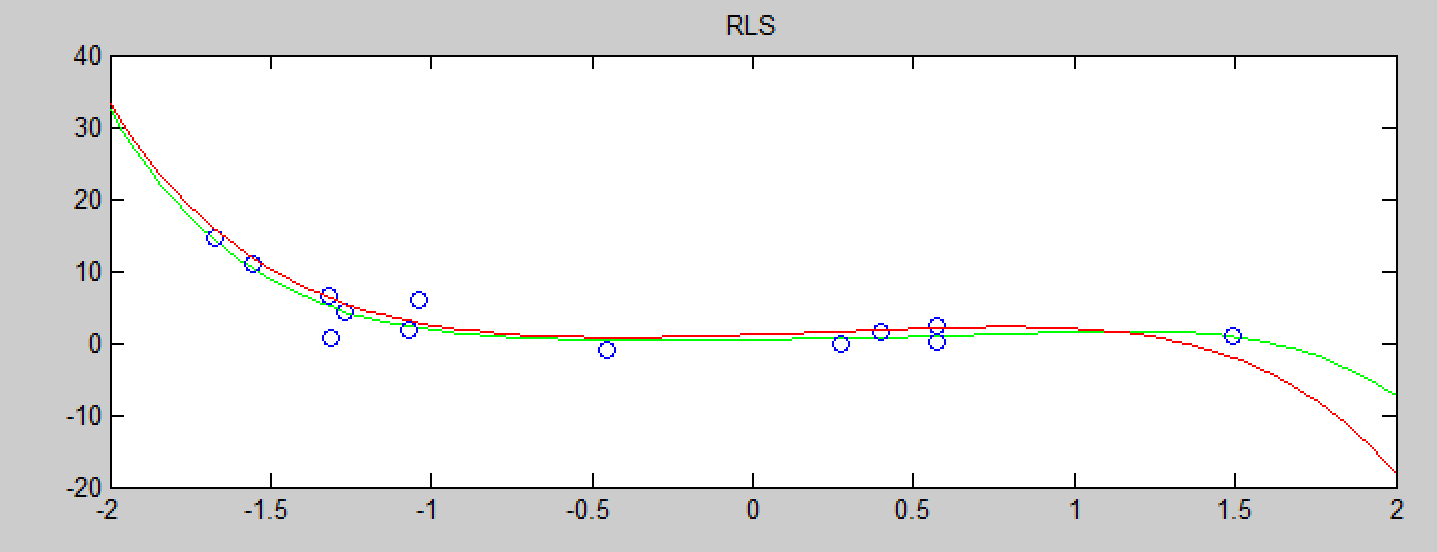
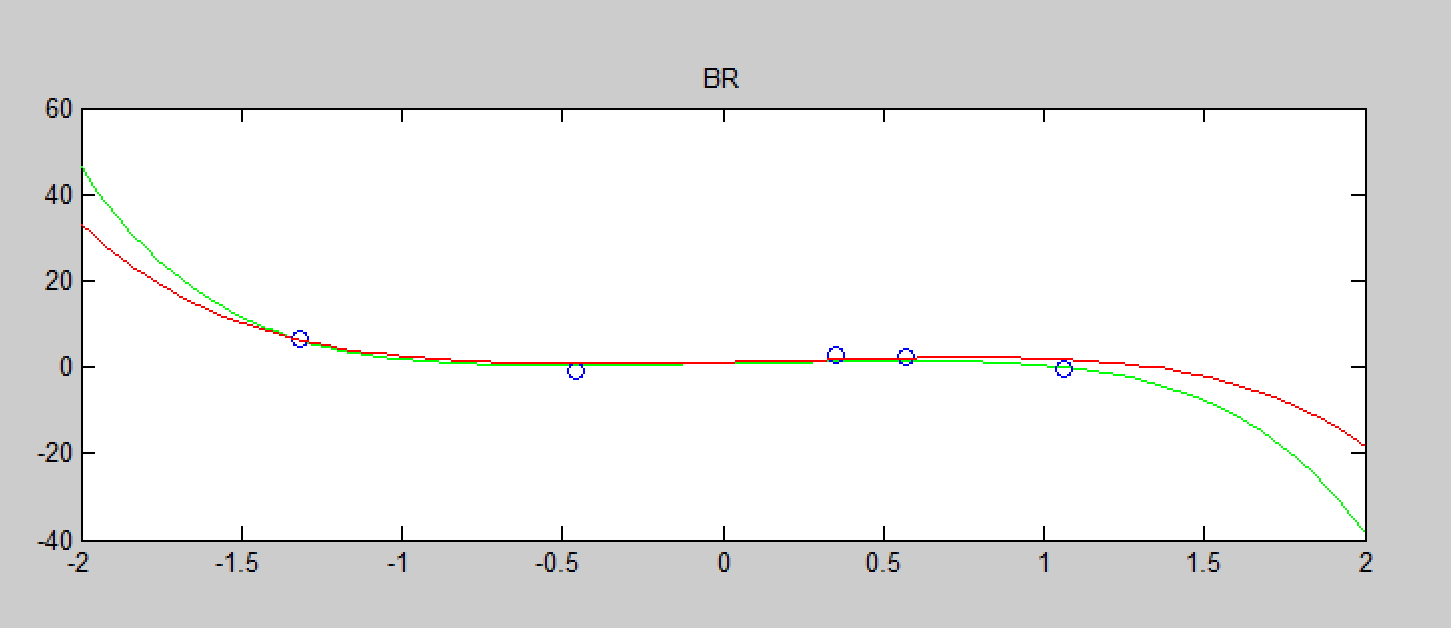
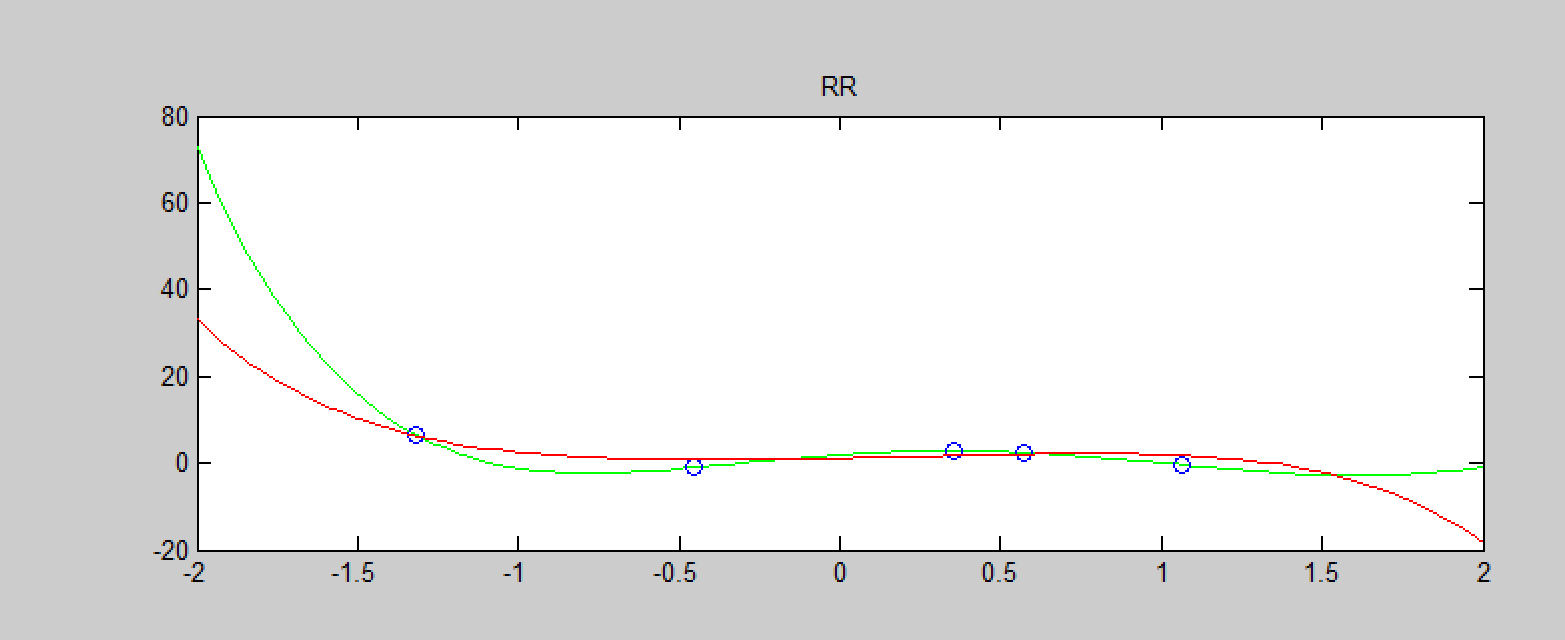
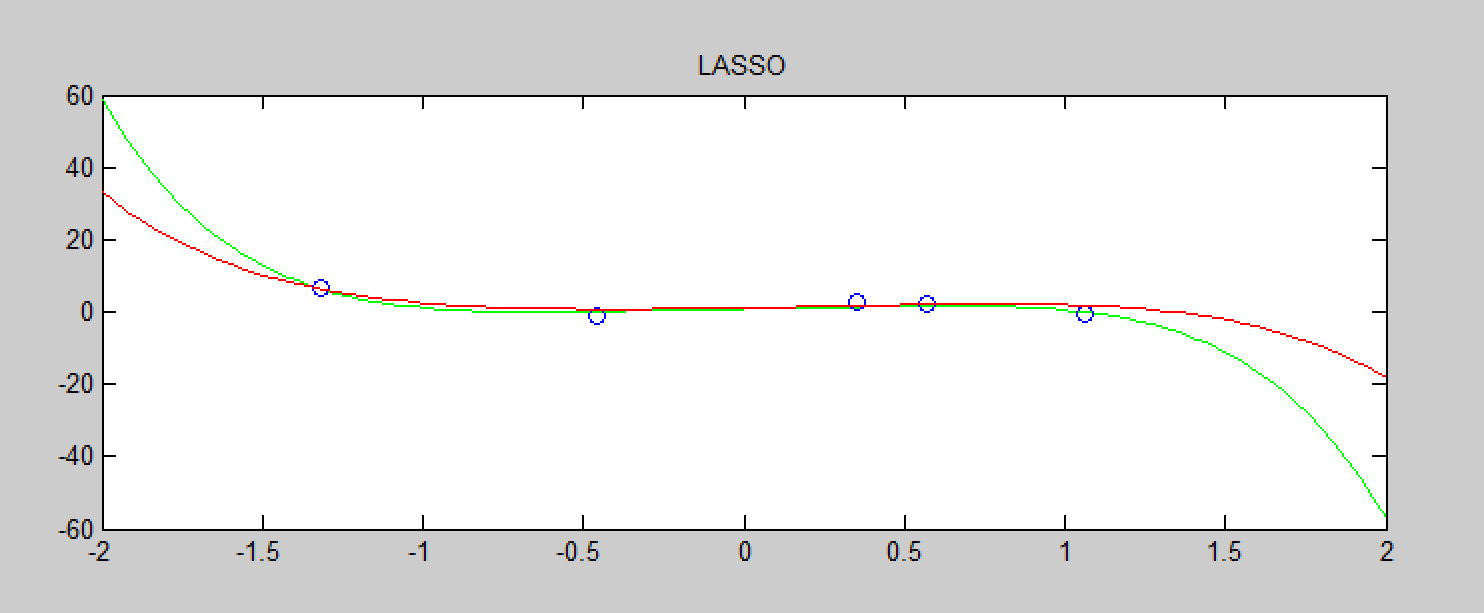
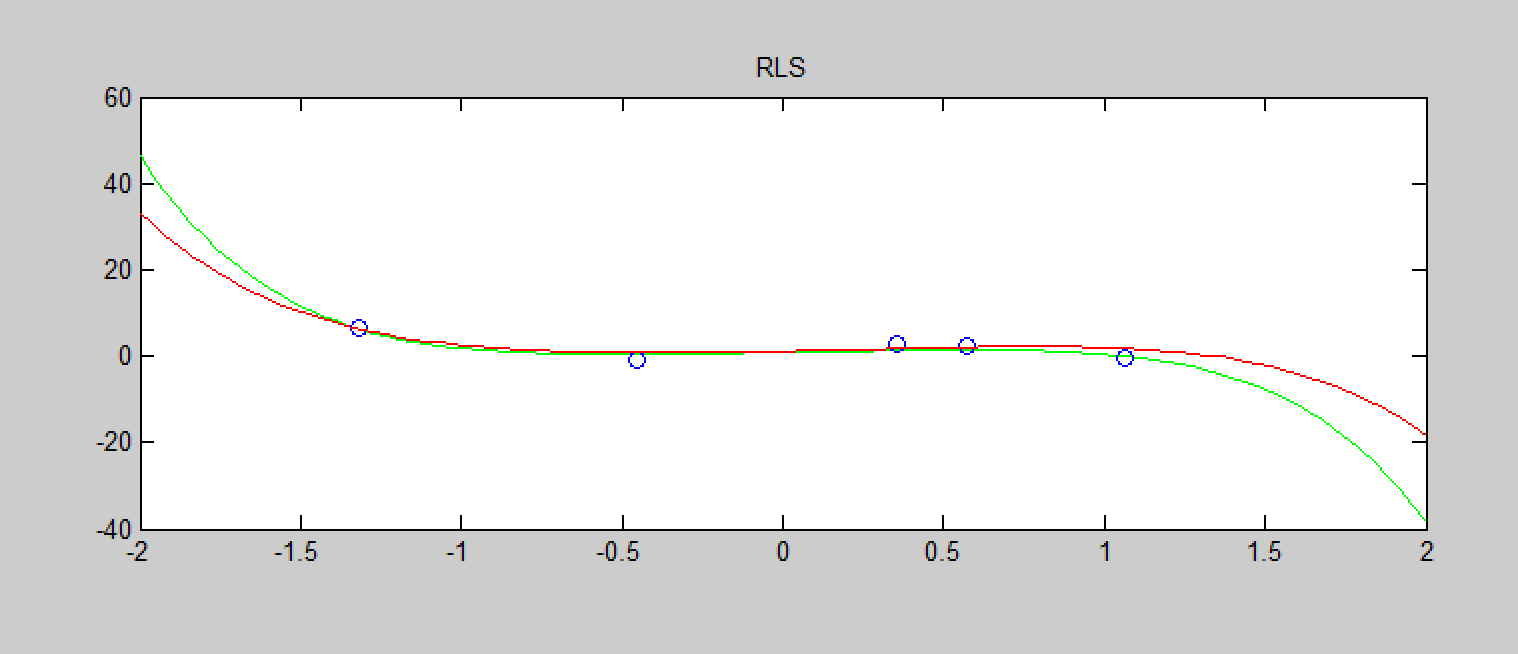
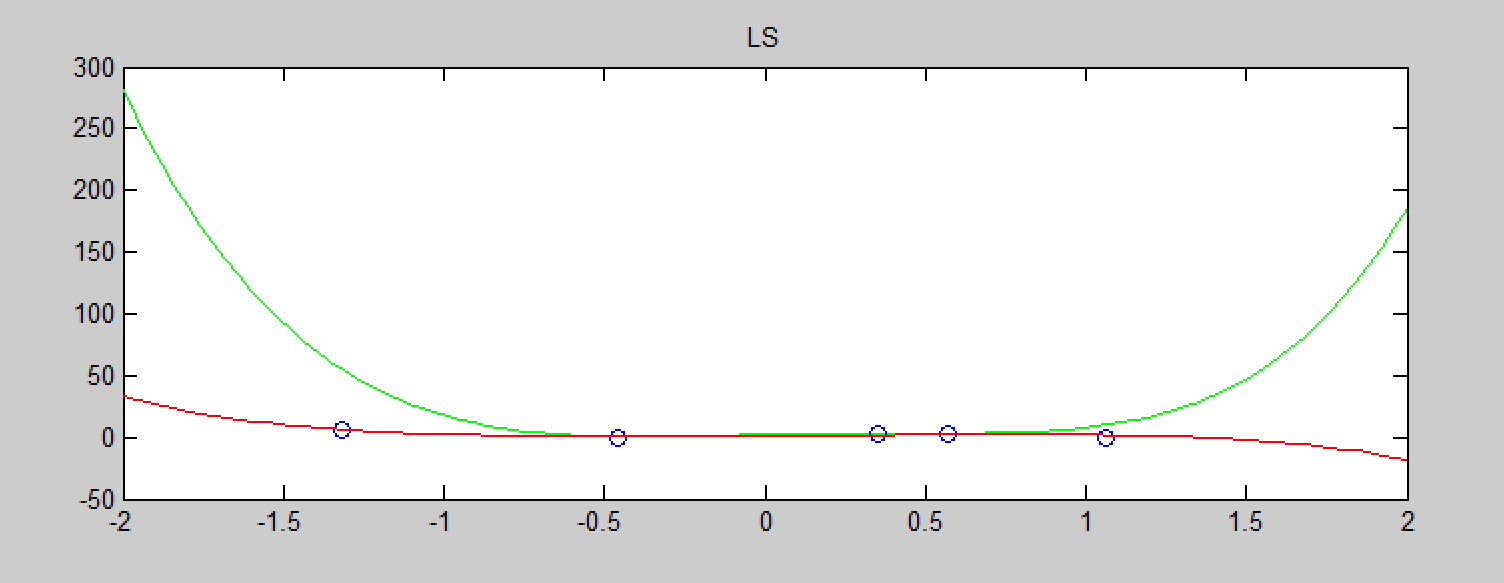
1. Firstly, before assign the algorithms, the feature needs to be transformed from a vector variable to a matrix according to the order K of the polynomial.   
   Here I write a function called ‘x\_phi’ to achieve this operation.  
   Then the calculation will be all about matrix or vectors.  
     
   The objective is to calculate the θ vector which should be multiplied by the matrix Ф to get the regression result of y.  
     
   Then the 5 kinds of regression algorithms should be implemented:  
   1) LS (Least Square)  
   LS method is easy to code, I just need to use the equation given as left picture shows.  
   2) RLS (Regularized Square)  
   RLS is similar to LS. The difference between them is that RLS has a square regularized term of θ. And when implement the algorithm. We need to insert a parameter λ, The value of it can be changed to get a better regression result.  
   3) LASSO (L1-regularized LS)  
   To implement LASSO, a function ***quadprog***  is needed according to one question in the problem sets, which is the QP solver.  
   And before using the function ***quadprog*** , we should use the matrix Ф and y to get H and f. And after that, the ***quadprog*** can be assigned, which is the algorithm of the equation below.  
   And besides, one important thing should be dealt with, the constraints of x ≥ 0. This constraints need to be achieved by linear algebra, not just simply set the value of x above 0.   
   The hyperparameter here is λ.  
   4) RR (Robust Regression)  
   RR is a 1-order regression method. It simply use the distance between observation and prediction.   
   To get the parameter estimate by RR, LP solver is needed, which is also included in the problem sets. The function is ***linprog***. The method is similar to ***quadprog*** . The difference is that linprog has more constraints.   
   There is no hyperparameter in RR method.  
     
   5) BR (Bayesian Regression)  
   BR method is different from the four methods before. Because the parameter estimate is a distribution here. In order to obtain a defined value. We need to get a mean value of the distribution to assign to the regression data.   
     
   Firstly the ∑ need to be calculated according to the equation. Then use the ∑ to get the mean value, μ, which will be used as the θin the regression steps.   
   The hyperparameter are α and σ.

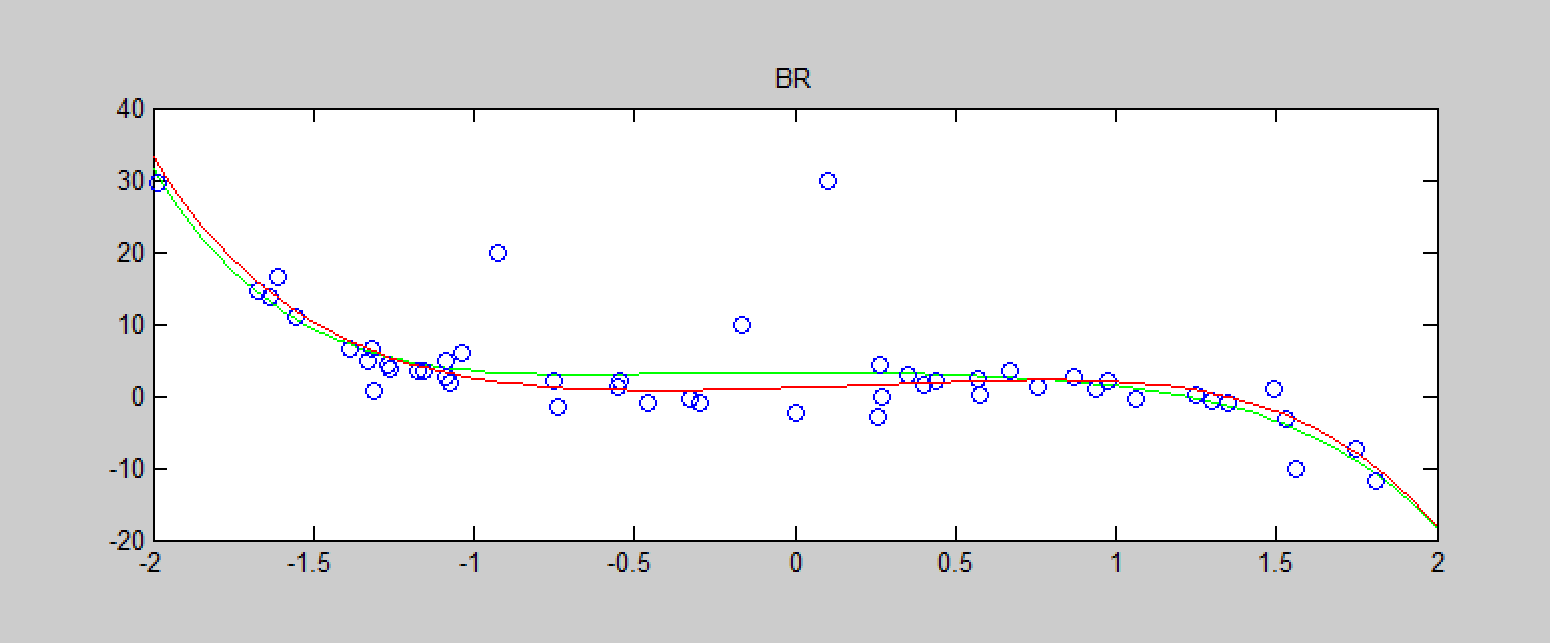
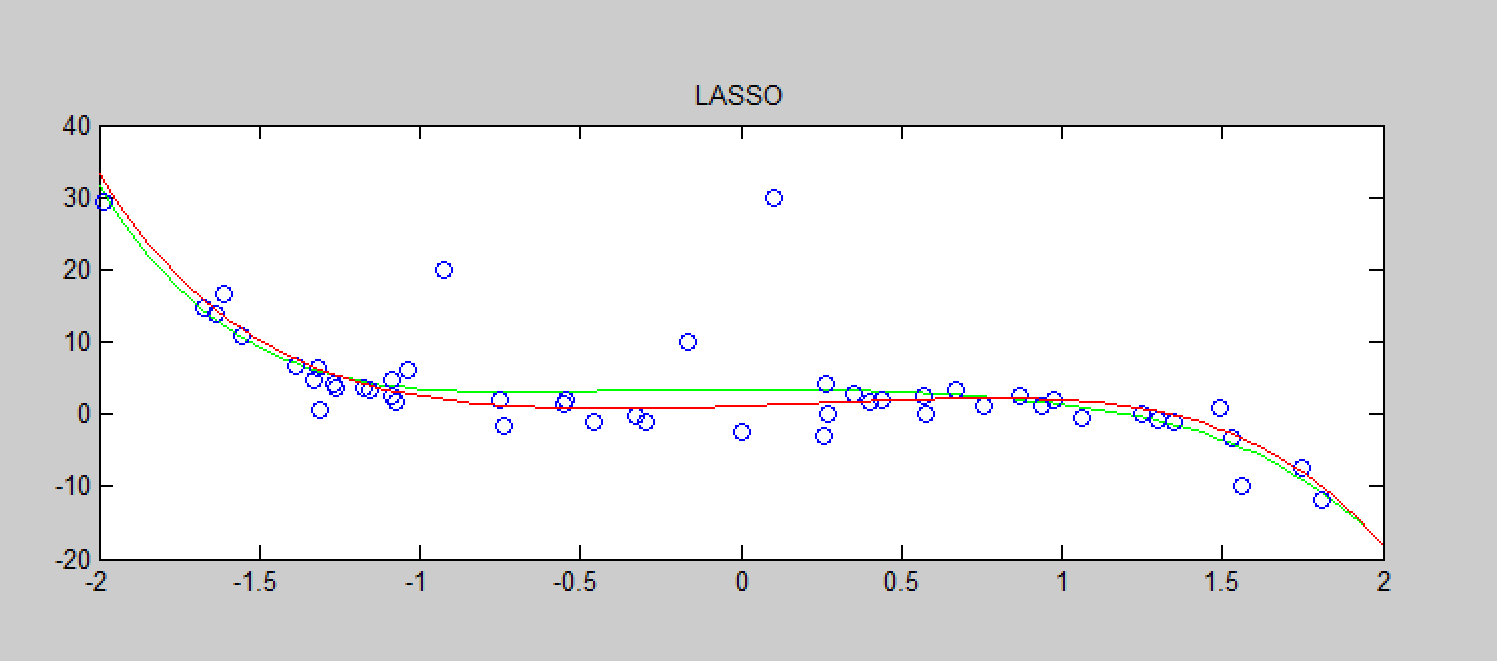


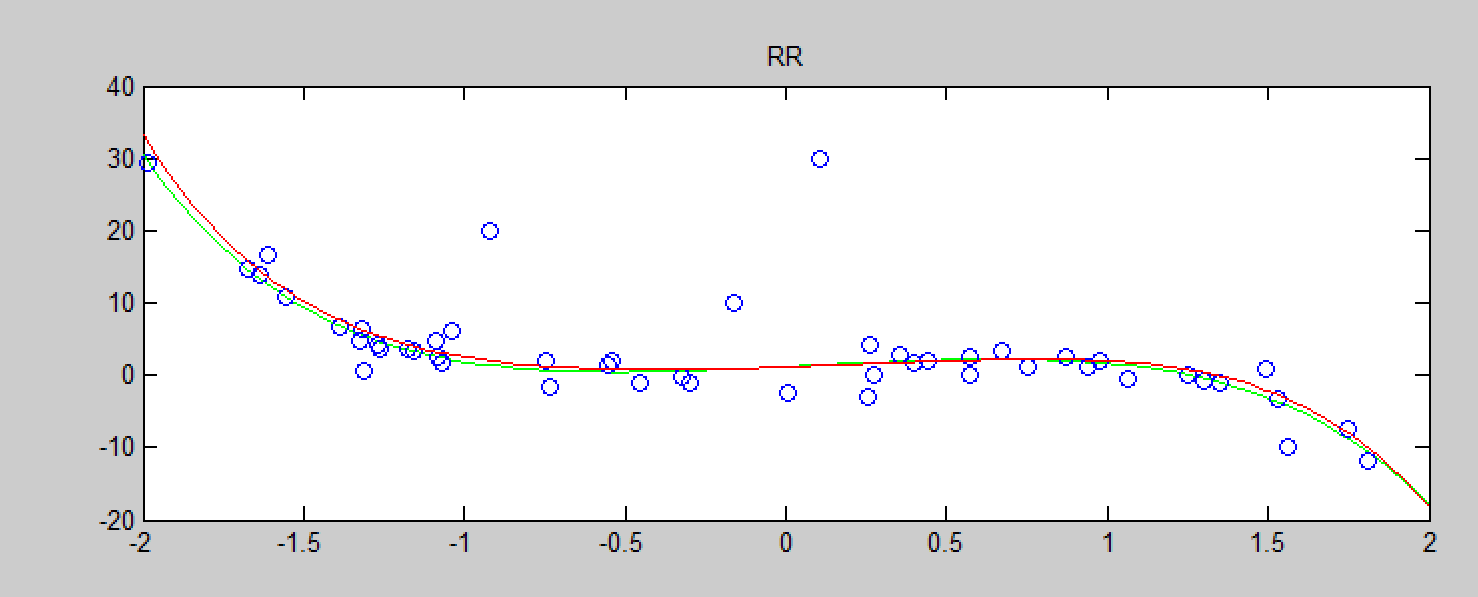
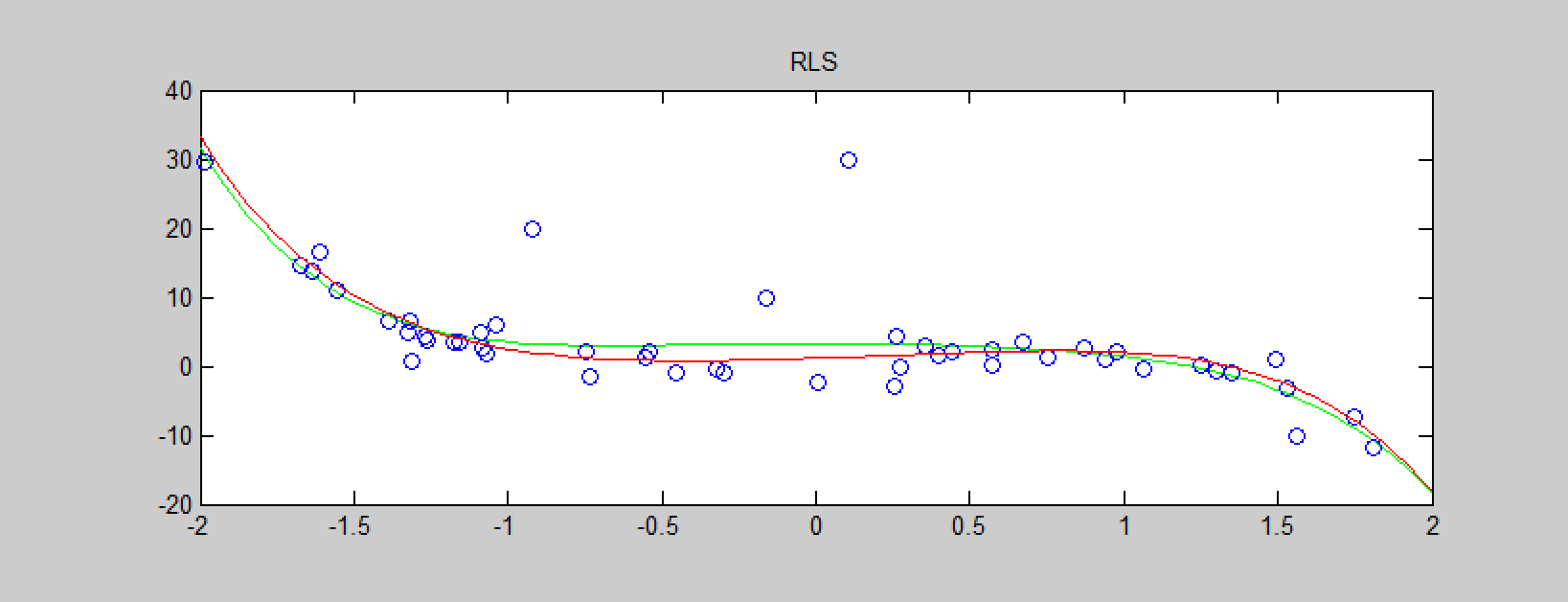
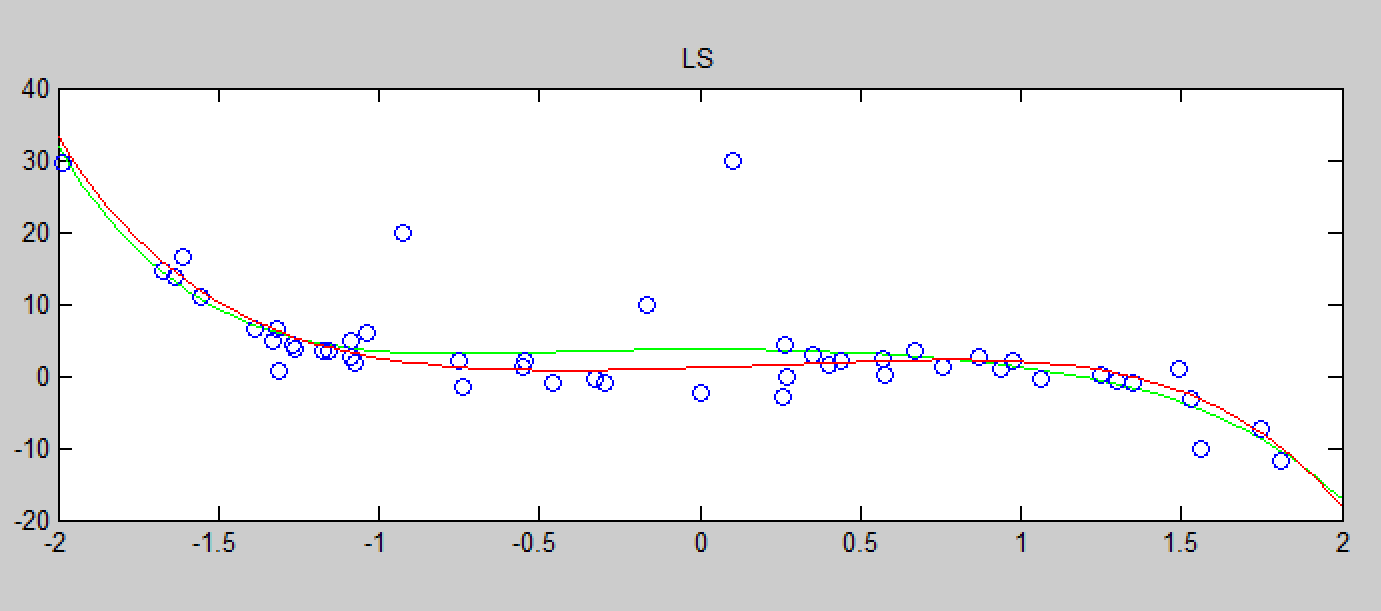
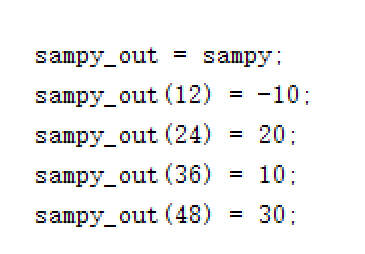
1. Regression of different methods using sample data is showed in the following picture.   
      
   As showed here, the order is 5th, the data set has 50 points. To simplify the results, the hyperparameters are set as λ = 1, σ=1, α = 1.   
     
   (here is for the BR standard deviation.)  
     
   The mean-squared errors between the learned function and the true function outputs are calculated as follows.  
   LS\_mse = 0.408643883570308  
   RLS\_mse = 0.408632570883915  
   LASSO\_mse = 0.475353854646079  
   RR\_mse = 0.768046316128372  
   BR\_mse = 0.408632570883692  
   It can be oberved that the LS, RLS, and BR are close in mean-squred error. That might be due to the fact that α/λ in BR method is the same as λin RLS method, whereas the two methods has different sensitivity to the data sets. This can be seen in the followin problems.   
     
   (Here is for different parameters.)

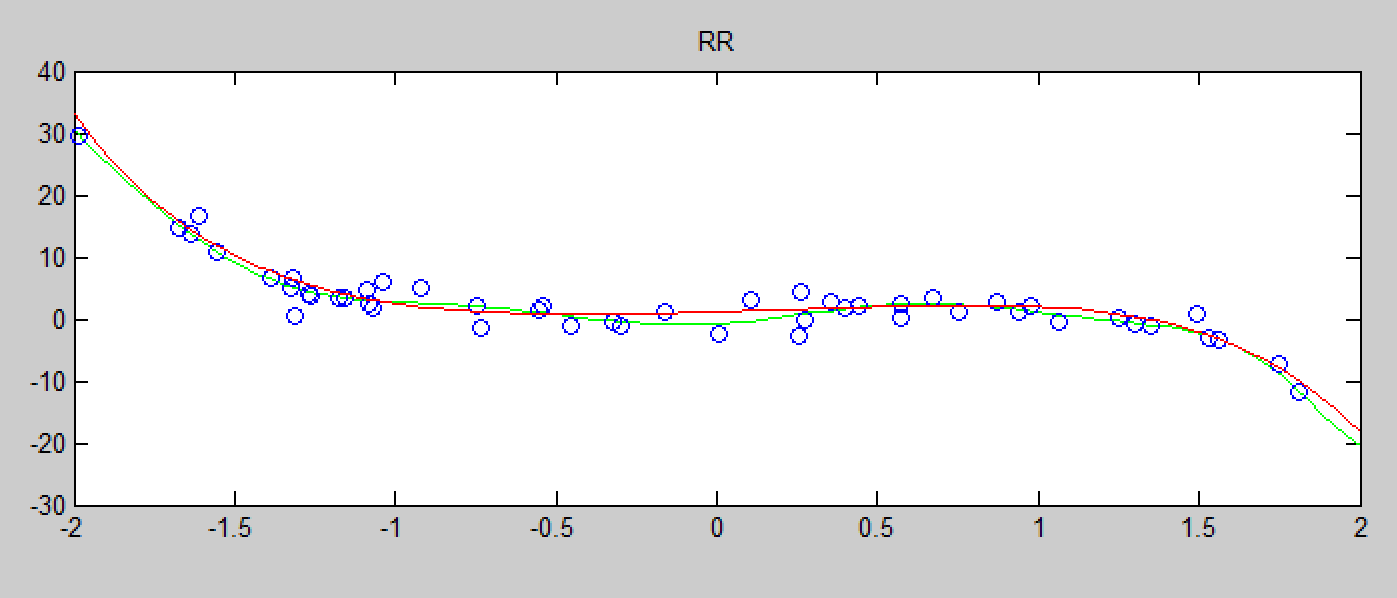


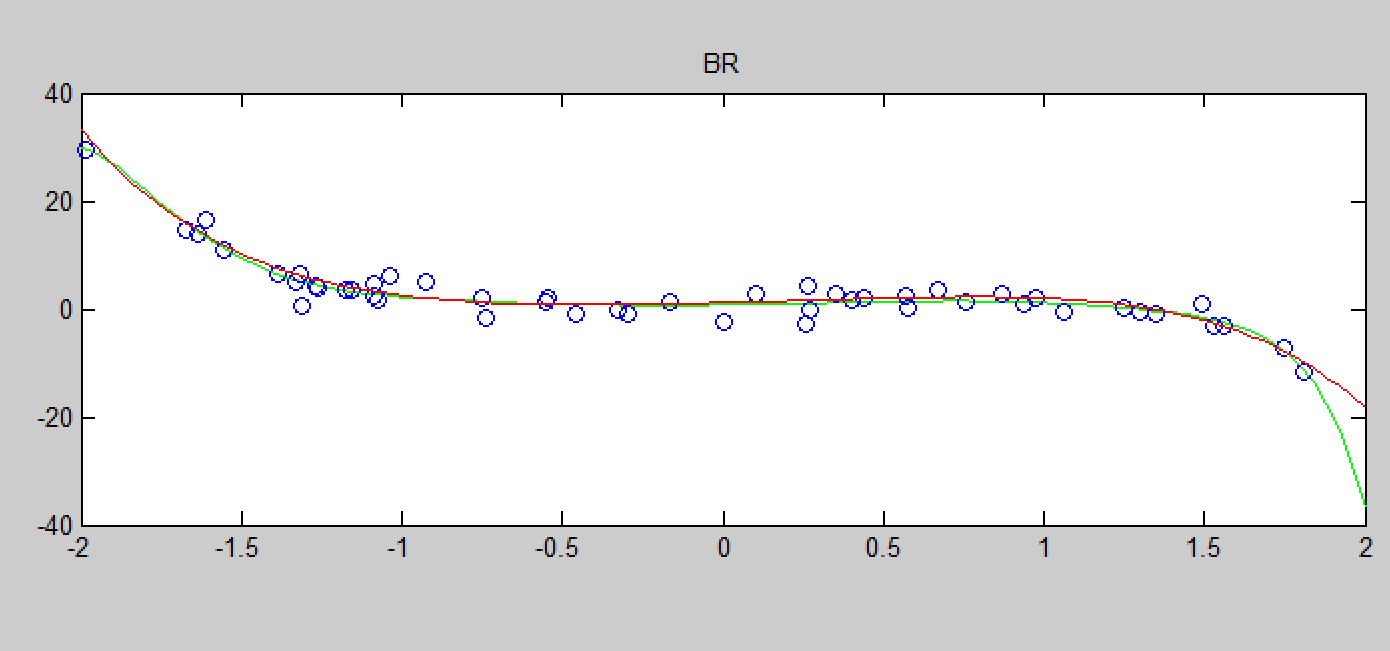
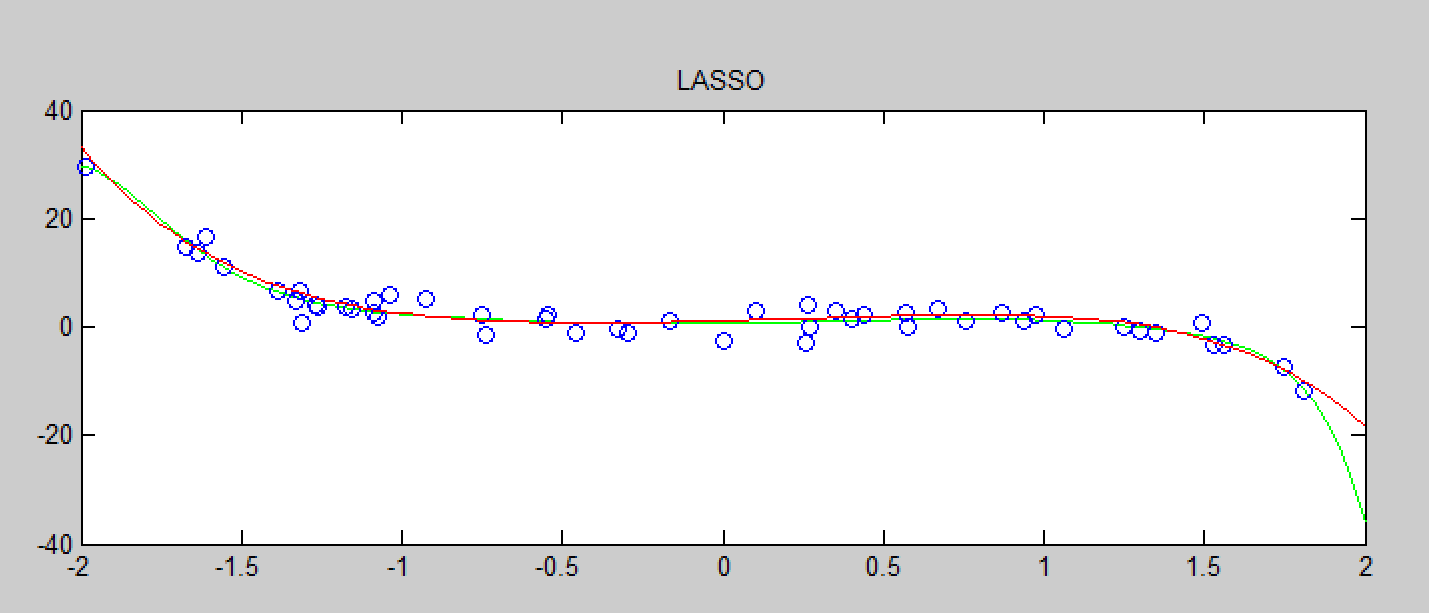
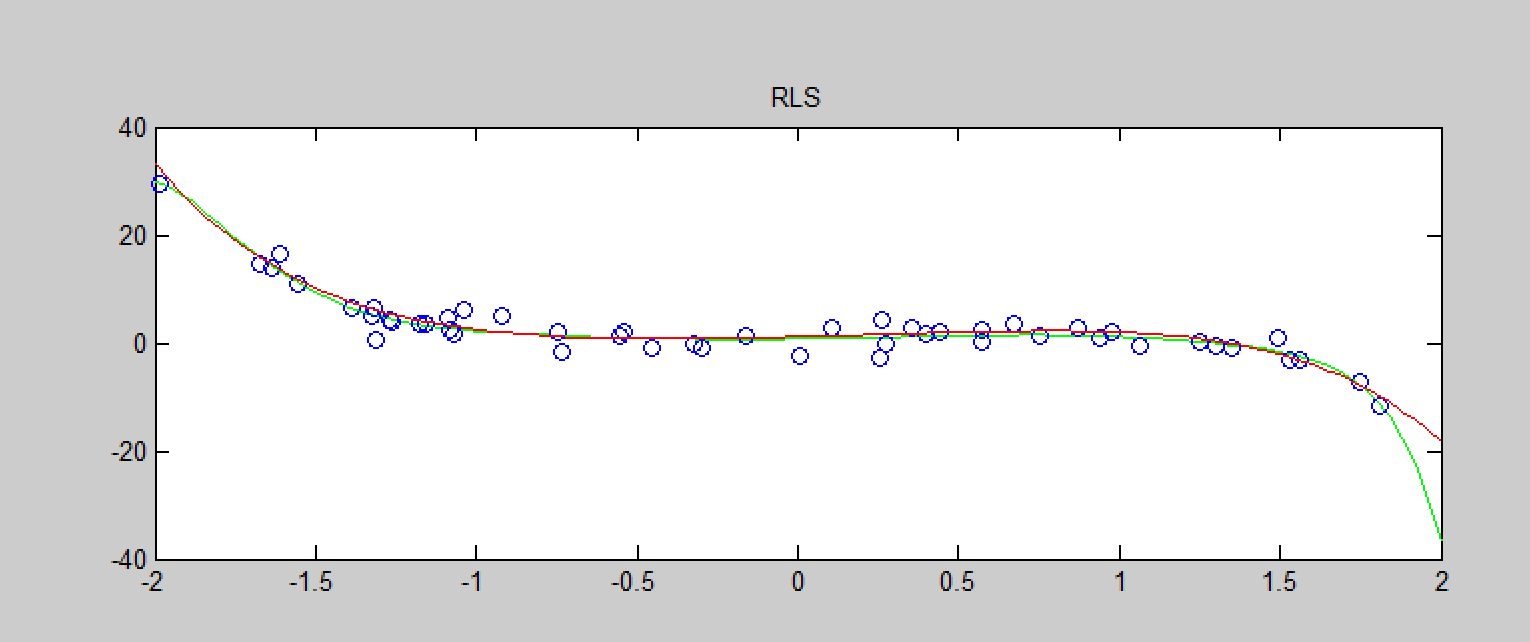
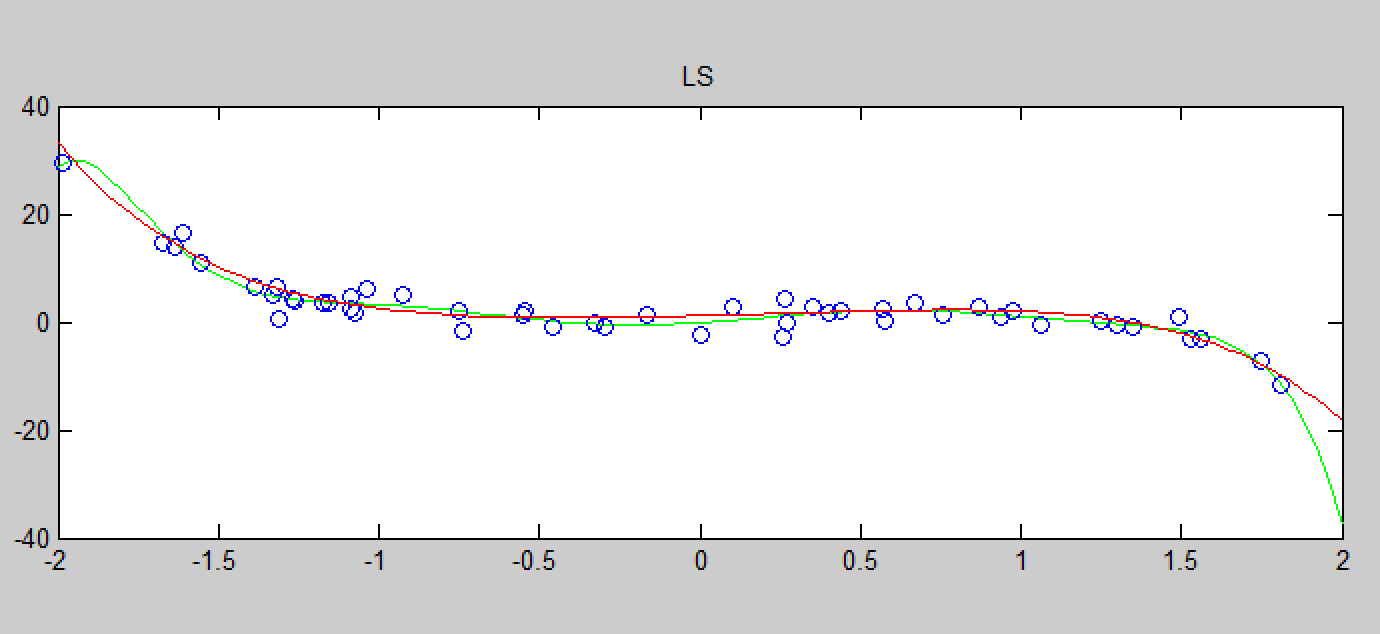
1. For 10% data, the estimated function is showed as follows.(initial point is the 1st point)  
   Here the data is generated into 10 different trials according to different initial sample point. And the mean squared error is the mean of the same method with different trials.  
   The mean-squared error:  
   mse\_LS = 1.797703885604259e+09  
   mse\_RLS = 30.657908739312518  
   mse\_LASSO = 39.069799496542416  
   mse\_RR = 1.605140020001430e+04  
   mse\_BR = 30.657908739312518  
   Obviously, RLS and BR methods tend to be more robust, while LS underfits quite a lot. Besides, RR also doesn’t have a good result when data is not much.  
     
   For 25% data, the estimated function is showed as follows.(intial from 1st point)  
     
   Here the data is generated into 4 different trials according to different initial sample point. And the mean squared error is the mean of the same method with different trials.  
   The mean-squared error:  
   mse\_LS = 70.153277000795870  
   mse\_RLS = 4.647013016673829  
   mse\_LASSO = 5.777384507341600  
   mse\_RR = 47.150214353632016  
   mse\_BR = 4.647013016674033  
   Obviously, the mean-squared error decreased dramatically when the data points is more. While still LS and RR have the worst regression result, whereas the RLS and BR have similar result, and LASSO has a tiny worse result than RLS and RR.   
     
   For 50% data, the estimated function is showed as follows.(intial from 1st point)  
     
   Here the data is generated into 2 different trials according to different initial sample point. And the mean squared error is the mean of the same method with different trials.  
   The mean-squared error:  
   mse\_LS = 1.478958888497502  
   mse\_RLS = 1.102741313794592  
   mse\_LASSO = 1.276431503084854  
   mse\_RR = 1.856753697118480  
   mse\_BR = 1.102741313794367  
   And this time the mean-squared errors are very close between different methods, which is due to the greatly increase in data size. And RR now has the worst result here. Where as the difference between them is actually very tiny.   
     
   For 75% data, the estimated function is showed as follows.(intial from 1st point)   
     
     
     
   Here the data is generated into 3 different trials according to different initial sample point. And the mean squared error is the mean of the same method with different trials.  
   The mean-squared error:  
   mse\_LS = 0.704311761067983  
   mse\_RLS = 0.560956677060883  
   mse\_LASSO = 0.567560841195825  
   mse\_RR = 0.727294660016577  
   mse\_BR = 0.560956677060892  
   When the size reach 75% of the data sets, the mean-squared errors have decreased below +1. Still RR > LS > LASSO > RLS > BR, while the difference is approximately 0.2.   
     
   And here is the error versus training size, where the plot of the mean-squared error is the natural logarithm (log) due to the size of the data.  
     
   red = LS;  
   cyan = RR;  
   blue = RLS;  
   yellow = LASSO;  
   green = BR;  
     
   Obviously, the LS method has the largest decrease magnitude among all the methods, followed by RR method, whereas the other 3 methods are close to each other.   
   And all of them will decease dramatically until when the data size is above 50%. After that, the regression result will be similar to each other.



1. The outliers is added as the following code shows:  
     
   And the regression result is showed as follows:  
     
   mse\_LS = 2.510650586343948  
   mse\_RLS = 2.015654141649858  
   mse\_LASSO = 2.243471687777492  
   mse\_RR = 0.588367365640390  
   mse\_BR = 2.015654141649540  
   The mean-squared error shows that RR method is much more robust than all other 4 methods. However, the difference between the four other methods is not so big, where LASSO is the least robust to the outliers.  
   This may be speculated according to the function of each methods. Firstly, the RR methods doesn’t use the squared difference between predicted value and true value, where the others are all based on the squared values. Besides, the regularized term for LASSO is also a squared value, thus for outliers, this method has more bias than RLS.

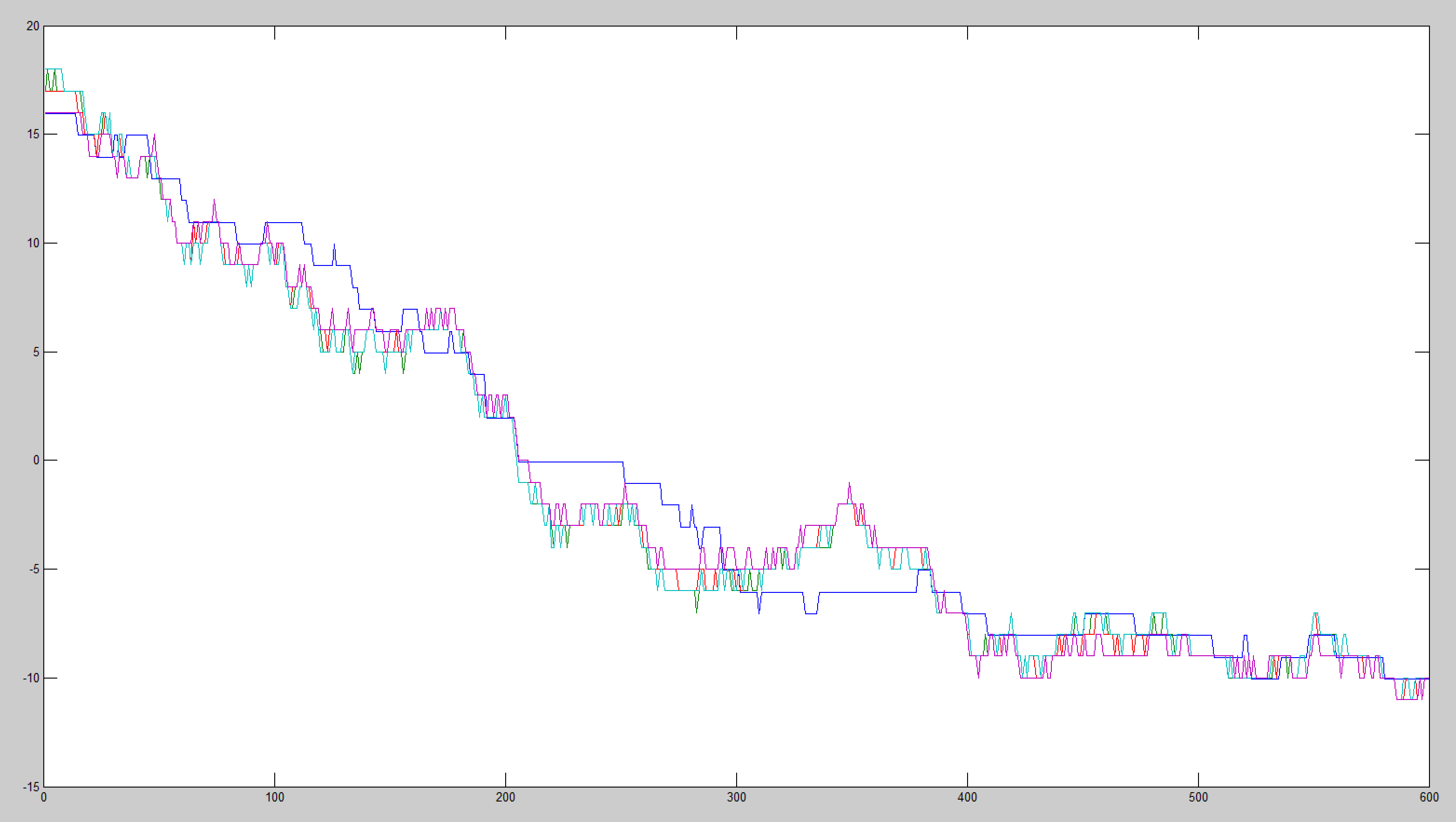
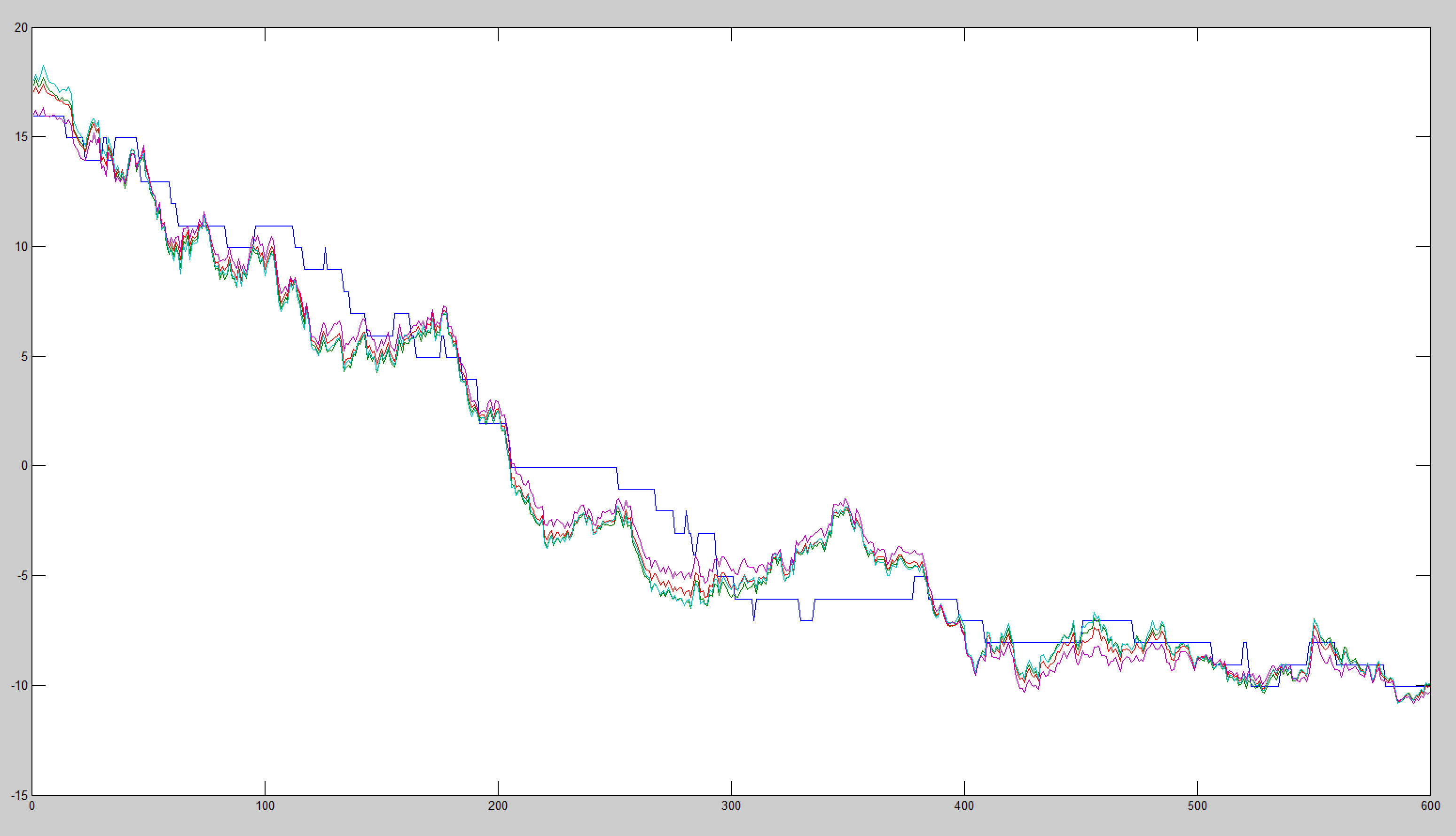


1. Here the data is generated in a 10th polynomial,   
     
   Here the mean-squared errors are:  
   mse\_LS = 7.983120896948182  
   mse\_RLS = 6.54809618840197  
   mse\_LASSO = 6.141728517181902  
   mse\_RR = 1.289856786131285  
   mse\_BR = 6.548096165217130  
   Obviously, the mean-squared error shows that RR method doesn’t change too much when the order of the polynomial is higher. And the other 4 methods tend to be overfitted.



Part II – Counting People

1. The code generated by the last question can still be used here. However, for the x to Ф step, since the feature is a 9-dim vector, the transform method is a little bit different. Even though, the idea is the same. For detail, please refer to the code.  
   Since the RLS and BR methods are very close in results, where BR is a little bit better. I omit RLS method in the regression. After implement the 4 methods, the regression result is showed as follows:  
     
   And the rounded version:  
     
   It seems that the result is not quite close to the true value of the count. And the mean-squared error and mean-absolute errors are:  
     
   mae\_LS = 1.358443521146632  
   mae\_LASSO = 1.300875110268075  
   mae\_RR = 1.364670844369748  
   mae\_BR = 1.282432855732872  
     
   mse\_LS = 3.102838014135252  
   mse\_LASSO = 2.782724080695631  
   mse\_RR = 3.119676745235948  
   mse\_BR = 2.618733922241896  
     
   It can be concluded that BR > LASSO > LS > RR in the regression result. Even though, the four methods all don’t have a good result which is close to the true value.



1. After implement the xN\_to\_PhiN, which is used to expand the origin feature to a 2nd order feature transformation.  
   The result is actually better as showed below:  
     
   And rounded version:  
     
   We can see the regression result is closer than the 1st order result. And the mean-squared error:  
   mse\_LS = 2.923594026741448  
   mse\_LASSO = 2.545355619640815  
   mse\_RR = 2.896194850746470  
   mse\_BR = 2.467773902568049  
   However, the mean-squared error didn’t change too much.  
   And I tried a higher order, which are 3\*9 = 27 version and 5 \* 9 = 45 version.  
   For 3rd order, the rounded regression results are:  
     
   For 5th order, the rounded regression results is:  
     
   Obviously, the count results are over fitted.  
   Thus we can conclude that the result in 2nd order is slightly better than the 1st order regression, however, when the order is higher than 2nd, the results will be over fitted, so the polynomial non-linear feature may not be so sufficient.

