UCLA College of Letters and Sciences
Department of Mathematics

MATH 170B PROJECT REPORT

Artem Pashchinskiy and Jianzhen Wang

June 4, 2018

In most treatments of Gaussian linear models, constant variance of error is assumed under the ordinary least squares approach. However, in many real world applications, noise variance tends to be correlated with the input variable(s). In this project, we try to estimate the parameters of data-generating process in 5 datasets with an unknown distribution of noise.

In the problem specification, we are told that the linear model is of the form

$$y_i = a * x_i + b + e_i$$

We started our work with creating scatter plots of the provided data. Based on the estimated distribution of variance, we concluded that heteroscedasticity is present in all 5 sets of data. That means that the variability of the response variable is unequal across the range of values of the independent variable. In order to find the best estimation for the parameters a and b in each data set, we formulated five different models, including Weighted Least Square Regression, Huber Regression, Ridge Regression, LASSO, and Generalized Least Square Regression to accommodate the volatile nature of the independent noise e_i.

From a technical realization standpoint, we used Python coding language along with functions from publicly available packages Pandas, NumPy (for data processing) and SciPy Sklearn (for regression models). Full implementation of the code can be accessed on the Git Hub repository.

Ordinary Least Squares Regression

The first and the most straight-forward approach was to perform Ordinary Least Squares Regression. It is known that although OLSE estimators remain unbiased in the cases of heteroskedastic data, those are no longer best estimators. Hence, we did not expect this model to have the most optimal performance. Instead, we used it in developing appropriate data transformations in more complicated examples of Weighted and Generalized Least Square regressions.

Huber Regression

Since we observed some potential outliers, we also implemented the Huber Regression on the given data. Huber Regression takes into consideration im-

pacts that the outliers make on the model, but it also makes sure that the loss function is not heavily influenced by the outliers. With Huber Regression, we optimized the squared loss and the absolute loss for the samples, in order to estimate a and b for the best-fitted line.

Ridge Regression

We utilized the given ridge cross validation function in Python to fit the data sets. We set the alpha values from -20 to 20 with 0.5 increments to try the model. With the best alpha value for the loss function, we estimated the parameters a and b for the fitted regression line.

LASSO

We applied the underlying LASSO cross validation function in Python on the response variable y_i with the 0.0001 length of path and 1000 alphas along the path. We then used the loss function to estimate the parameters a and b, trying to fit a regression line on each given data set.

Although all models presented above have their advantages over the Ordinary Least Squares regression, none of them is specifically targeted to address the heteroskedastic nature of the data. Hence, we believe that results of the next three models are more reliable for this specific scenario.

Weighted Least Squares

One of the ways to approach x-dependent distribution of noise is to scale the contribution of various data points to the values of estimated coefficients. One of the simple dependencies that we were able to come up with based on the visual observations of data is

$$var(e_i) = \sigma_i^2 = \sigma^2 x_i$$

If the error of the data-generating process is following this dependency, we can use

$$\sqrt{x_i}$$

as a weight coefficient of error, i.e. transform the data into a new set that has homoskedastic errors.

Although Python packages have built-in functions for an automatic computation of weighted least square regression, we decided to stay conservative and perform the data transformation manually and use built-in functions only for applying OLSE to the transformed homoskedastic model.

Generalized Least Squares - I

Although in many cases Weighted Lest Squares gives a good estimate of the true coefficients a and b, it is heavily dependent on how well the error dependency was estimated. Since we had limited amount of information about the error, we decided to also implement a more general framework of Generalized Least Squares regression.

According to the assumptions of Generalized Least Square Regression, we know that the distribution of the dependent variable y_i given x_i is normal, but the variance of the residual is not constant.

$$(y_i \mid x_i) \sim \mathcal{N}(a * x_i + b, var(e_i|x_i))$$

Advantage of the GLSE method lies on the fact that opposed to WLSE it allows to automatically deduce the coefficients of error dependency. In this case we specified our model as

$$var(e_i|x_i) = \alpha * x^{\beta}$$

Since squares of OLSE residuals σ_i can be used as estimates of true errors e_i, we can apply OLSE to the model

$$\ln(\sigma_i^2) = \alpha + \beta \ln(|x|)$$

to get the estimated values $\hat{\alpha}$ and $\hat{\beta}$. Using this estimates, we can predict

$$var(e_i|x_i) = \hat{\sigma_i^2} = \exp(\hat{\alpha} + \hat{\beta} \ln|x|).$$

Once we have the predicted values, we can weigh the errors similarly to what we did in WLSE method and get a homoskedastic model:

$$y_i^* = az_1 + bz_2 = a_1 * \frac{1}{\hat{\sigma}_i} + b * \frac{x_i}{\hat{\sigma}_i}$$

Finally, we can apply OLSE model to regress y^* against z_1 and z_2 and get the estimations $\hat{a}$ and $\hat{b}$.

Generalized Least Squares - II

Towards the end of project development we got additional input specifying that functional form

$$var(e_i|x_i) = c + dx$$

was used to create noise in the data-generating process. To accommodate this extra assumption, we repeated the of Generalized Least Squares method again using this new error form.

Since this last attempt incorporates both known methodology of dealing with heteroskedastic error and the the largest available set of reliable assumptions (both a known model for data generating process and a known model for noise-generating process), we concluded that parameter estimates produced by this model should be the most accurate out of all considered models.

Results and Discussion

After obtaining the point estimates for all the models above, we utilized the coefficient of determination (R^2) and the root mean square error ($RMSE$) to assessment the quality of the parameter estimates.

R^2 is a statistical measure of the percentage of the response variable variation that is explained by a linear model. For each set of data, we applied the different models and we calculated the corresponding R^2. The following table is our finding.

Model/Dataset	1	2	3	4	5
OLSE	0.16399	0.39809	0.10320	0.02760	0.28058
WLSE	0.16641	0.35762	0.03055	0.83423	0.43860
Huber	0.15938	0.39379	0.08308	0.00603	0.25819
Ridge	0.16386	0.39809	0.10320	0.02670	0.28052
LASSO	0.16397	0.39809	0.10320	0.02759	0.02806
GLSE - I	0.22744	0.68233	0.11953	0.97459	0.71159
GLSE - II	0.22744	0.68233	0.11953	0.97459	0.71159

Although R^2 can be easily applied to measure the goodness of fit of a linear model, it has some limitations. First, R^2 does not determine if the predictions are unbiased. Based on the residual plots of the given data, our sample is biased, since the independent noise varies with the input variable. Second,

R^2 does not indicate the accuracy of a regression model. In other words, high R^2 does not necessarily mean that the model fit the data well, and vice versa for the low R^2. Thus, we tried another metric, RMSE, to access the quality of our models. As its name suggests, the root mean square error is the square root of the average of squared differences between the predicted values and the sample values. The following table shows the RMSE value for each dataset with various regression models applied.

Model/Dataset	1	2	3	4	5
OLSE	5.29497	15.42378	17.51802	5.07327	11.10232
WLSE	5.84677	19.88008	19.01470	7.70563	14.80013
Huber	5.31586	15.47881	17.71345	5.12921	11.27377
Ridge	5.29536	15.42380	17.51803	5.07560	11.10278
LASSO	5.29501	15.42378	17.51806	5.07327	11.10233
GLSE - I	5.64948	23.97476	18.96527	8.51875	14.68550
GLSE - II	5.64948	23.97476	18.96527	8.51875	14.68550