Exam project for course «Statistical machine learning»

Nikolai But

University of Geneva, Department of Mathematics

January, 2024

Goal of the project

We are given an artificially generated dataset of the form $[y, X]$ where first column is response and next 50 columns is input, sample size n = 2000. The goal of the project is to analyse it by firstly fitting it into a linear model and then using artificial neural network to identify existence of some non-linear dependence.
For the former we will use LASSO linear model and SURE estimator to determine optimal regularisation parameter λ. We will ensure that we do not have overfitting in this model by comparing it with 5-fold cross-validation.
For the second part we will compare models with different number of deep layers and varying number of neurons in them with one another and with the optimal linear model using cross-validation as well. This will determine presence of strong non-linear component. Moreover we will be able to chose optimal number of neurons for each number of layers and prevent overfitting.

Linear model

- Model: $\mathrm{y} = \beta_0 + \mathrm{X}\beta + \textbf{random noise}$

Here y is the output vector of length n, X is a matrix of size $n \times p$ which is in our case 2000×50, β_0 is a number and β is a vector of length p

- LASSO: $\widehat{\beta}_\lambda^{\text{lasso}} = \arg\min_{\beta_0,\beta} \frac{1}{n}\|\mathrm{y} - \beta_0 - \mathrm{X}\beta\|_2^2 + \lambda\|\beta\|_1$

We also need an unbiased estimator for σ of random noise since we don't know it in advance. For this we use the least square (LS) solution of the linear regression

- $\hat{\sigma}^2 = \frac{\textbf{RSS}(\hat{\beta}_0,\hat{\beta})}{n-p}$

Here $\hat{\beta}_0, \hat{\beta}$ are given by the least square solution and

$$\textbf{RSS}(\beta_0, \beta) = \|\mathrm{y} - \beta_0 - \mathrm{X}\beta\|_2^2$$

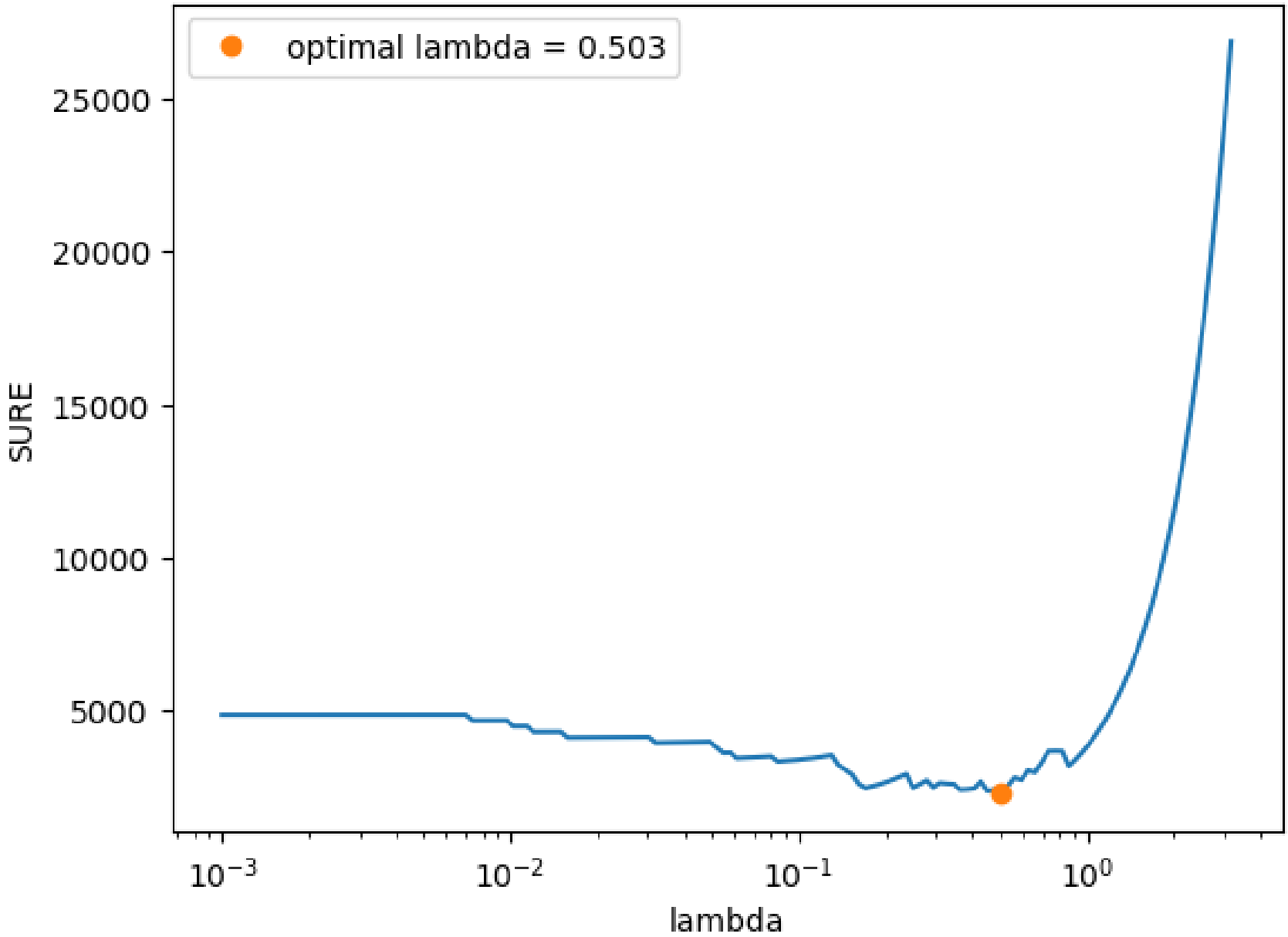
SURE

To determine the optimal λ we use Stein's unbiased risk estimate or SURE. The construction is more general but in our case the optimal value of the parameter will be arg min with respect to λ of the following expression:

$$\mathbf{SURE}(\lambda) = \mathbf{RSS}(\lambda) + 2\hat{\boldsymbol{\sigma}}^2 \mathrm{p}_\lambda - \mathrm{n}\hat{\boldsymbol{\sigma}}^2$$

Here p_λ is number of non zero entries in $\hat{\boldsymbol{\beta}}_0, \hat{\boldsymbol{\beta}}$ for given λ
This is an unbiased estimator of mean-squared error (MSE) which will allow us to make a good choice of the parameter
The outcome of the numpy program calculating SURE and finding optimal λ is presented by a graph on the next slide

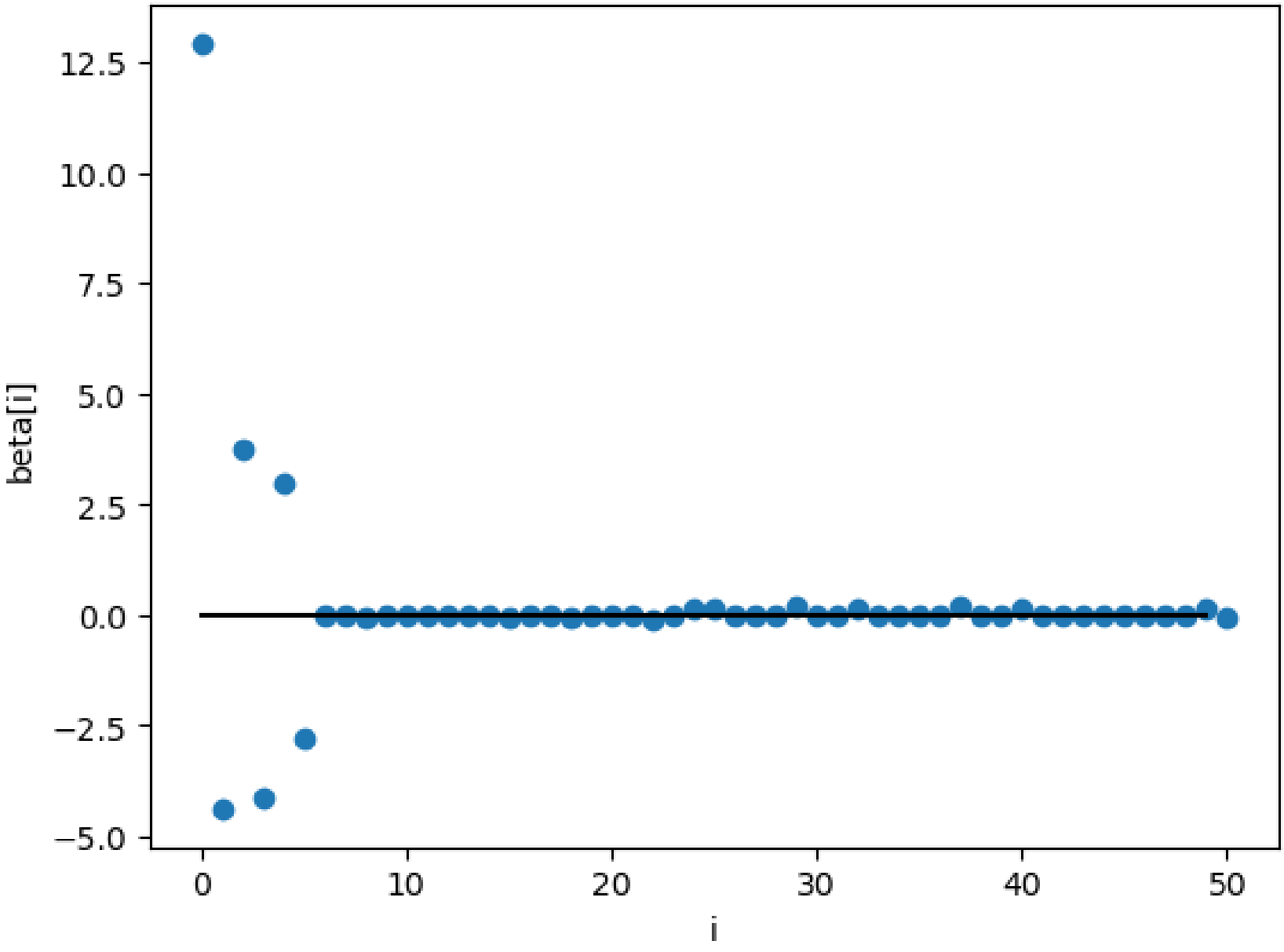
SURE


optimal lambda = 0.503
25000
20000
15000
10000
5000
SURE
10^{-3}
10^{-2}
10^{-1}
10^{0}
lambda

Sparseness of LASSO

One of the important features of LASSO is the fact that after a certain lambda $\beta_0 = (\text{mean of y}) = 12.97$ and $\beta = 0$
In our case it happens for $\lambda \geq 10$ and for λ between 1 and 10 SURE increases rapidly. This justifies the range of λ chosen for the graph above which is between 10^{-3} and $10^{0.5}$
The other consequence of this is sparseness of LASSO which means that for general λ in the middle of the range most of the coefficients of β_0, β vanish. This is demonstrated on the next slide by a graph depicting these coefficients for the optimal $\lambda = 0.503$
On this graph we clearly can see that only first six coefficients are different from zero which means that only five first variables in X are significant for this linear model

Sparseness of LASSO



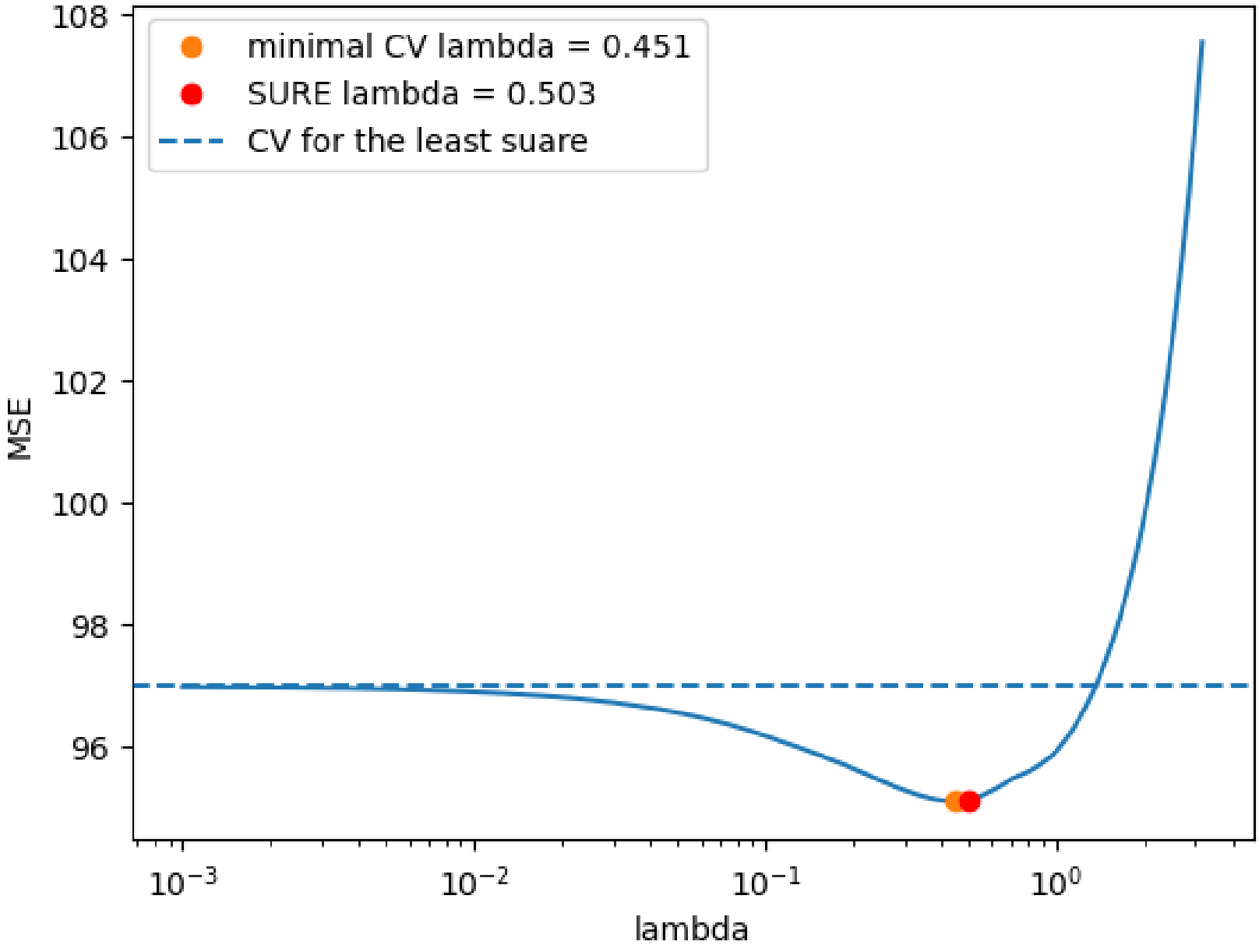
Cross-validation (CV)

We want to ensure that our choice of lambda does not lead to overfitting. For this we will use technique of 5-fold cross validation which is done as follows: we divide data into five pieces, train the model on four of them and calculate MSE on the remaining fifth. We repeat this five times with each piece of data as validation set. Then we take the mean of all errors which is our final value of cross-validation

On the graph bellow one can see it being done for our data-set. We can clearly observe that SURE lambda has low CV value which is not far from minimum value of CV

Also the graph has puncture line representing CV value for the least square. This value is above optimal since LS is prone to overfitting which can be observed for small lambdas. However, for big lambdas we have high bias so CV graph skyrockets. Overall we can conclude that previously obtained linear model has good balance of bias and variance

Cross-validation (CV)



Neural network analysis

In this part we use artificial neural network (ANN) to analyse the dataset remaining after subtracting our linear model:

- **remainder** $= \mathrm{y} - \beta_0 - \mathrm{X}\boldsymbol{\beta}$

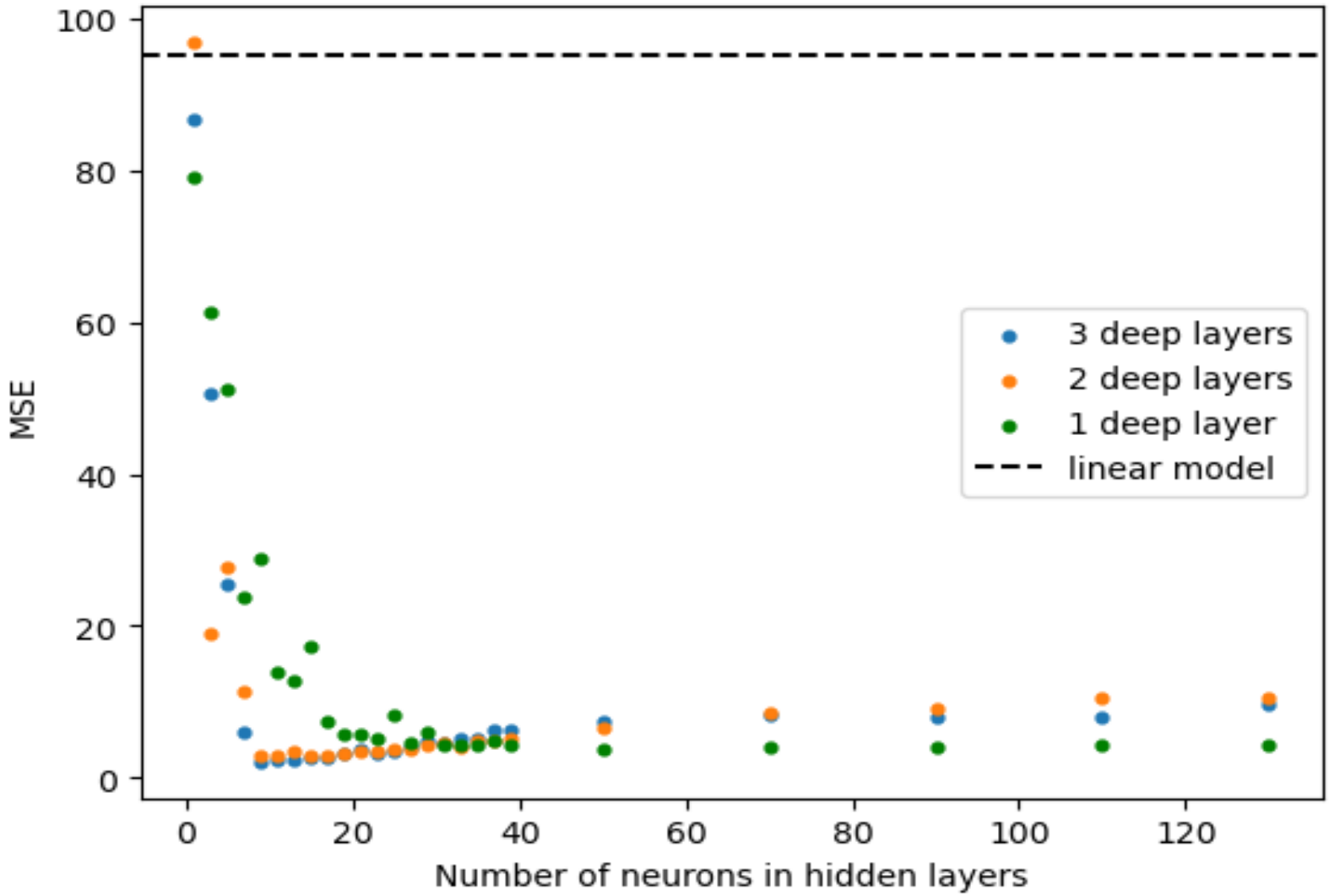
We will use ANN with ReLU activation function: $\sigma(x) = \max(x, 0)$, and MSE as cost function. Here is a training process of ANN with three hidden layers, 20 neurons each, executed by pytorch program:

Epoch [100/500], Loss: 75.26
Epoch [200/500], Loss: 3.09
Epoch [300/500], Loss: 1.37
Epoch [400/500], Loss: 0.99
Epoch [500/500], Loss: 0.80

For this project we will test ANNs with one, two and three hidden layers of the same size which we will vary to determine the optimal size for each number of hidden layers
NB: we use ANN to solve a regression problem, not a classification problem as it is often happens

CV for ANNs

We will apply the same 5-fold cross-validation technique to ANNs as we did for LASSO. CV values for different numbers of hidden layers and neurons in them are depicted on a graph below



CV for ANNs

From the graph above we can draw three main conclusions:

- By adding ANN to analysis of dataset we get much better CV values then we do for just a linear model which suggests existence of strong non-linear dependence
- For low numbers of neurons error is relatively high but it plummets quickly
- After the optimal value of neurons in hidden layers the error slowly grows which is a sign of overfitting

As could be expected overfitting is less prominent for one hidden layer. However for the optimal number of neurons ANNs with three and two hidden layers do better. Their respective behaviour is quite similar

Optimal size for two layers is 17 and for three layers is 9. Since ANN with three layers, 9 neurons each, has less weights to determine and also has a little lower value of CV we can use it as a possible model for our dataset

Conclusion

- For given dataset we have found a linear model using LASSO and SURE for regularisation of the least square solution. We also determined that only five first features of the input are significant for this dependence
- We have found a significant non-linear component which can be fairly well approximated by ANN with 3 hidden layers each consisting of 9 neurons. This approximation is validated by 5-fold cross-validation. It also can be noted that because the source of data is unknown it is difficult to do much more