Assignment 1 - Machine Learning

Theodora Gaiceanu

Task 1

From the problem formulation, we know that $\lambda \geq 0$. Therefore, we can minimize the expression $\frac{1}{2}\|r_i - x_i w_i\|_2^2 + \lambda|w_i|$ with respect to w_i considering two cases: $\lambda = 0$ and $\lambda > 0$. We also know that $w_i \neq 0$.

Case 1 $\lambda = 0$
In this case, the objective function becomes:

$$minimize_{w_i} \frac{1}{2}\|r_i - x_i w_i\|_2^2$$

So we take the first derivative of the objective function with respect to w_i and equalize it to 0.

$$0 = \frac{\partial}{\partial w_i}(\frac{1}{2}\|r_i - x_i w_i\|_2^2)$$

$$0 = \frac{\partial}{\partial w_i}(\frac{1}{2}(r_i^2 - 2x_i^T r_i w_i + w_i^2 x_i^T x_i))$$

$$0 = \frac{1}{2}(-2x_i^T r_i + 2x_i^T x_i w_i)$$

$$0 = -x_i^T r_i + x_i^T x_i w_i$$

$$0 = -x_i^T(r_i - x_i w_i)$$

$$w_i = \frac{x_i^T r_i}{x_i^T x_i}$$

Case 2 $\lambda > 0$
In this case, our objective function is:

$$minimize_{w_i} \frac{1}{2}\|r_i - x_i w_i\|_2^2 + \lambda|w_i|$$

We take again the first derivative of the objective function with respect to w_i and equalize it to 0.

$$0 = \frac{\partial}{\partial w_i}(\frac{1}{2}\|r_i - x_i w_i\|_2^2 + \lambda|w_i|)$$

$$0 = \frac{\partial}{\partial w_i}(\frac{1}{2}(r_i^2 - 2x_i^T r_i w_i + w_i^2 x_i^T x_i) + \lambda|w_i|)$$

We know from the problem formulation that $\frac{\partial|w_i|}{\partial w_i} = \frac{w_i}{|w_i|}$.

$$0 = -x_i^T r_i + x_i^T x_i w_i + \lambda\frac{w_i}{|w_i|}$$

$$-x_i^T r_i + \lambda\frac{w_i}{|w_i|} = -x_i^T x_i w_i$$

But we also know that $\frac{w_i}{|w_i|} = sgn(w_i)$.

$$-x_i^T r_i + \lambda sgn(w_i) = -x_i^T x_i w_i$$

$$w_i = \frac{x_i^T r_i - \lambda sgn(w_i)}{x_i^T x_i}$$

Since $\lambda > 0$ and $x_i^T x_i > 0$, we can say that:

$$sgn(w_i) = sgn(x_i^T r_i) = \frac{x_i^T r_i}{|x_i^T r_i|}$$

Therefore, one has that:

$$w_i = \frac{x_i^T r_i - \lambda sgn(x_i^T r_i)}{x_i^T x_i}$$

$$w_i = \frac{x_i^T r_i - \lambda \frac{x_i^T r_i}{|x_i^T r_i|}}{x_i^T x_i}$$

And this last equation leads to:

$$w_i = \frac{x_i^T r_i}{x_i^T x_i |x_i^T r_i|}(|x_i^T r_i| - \lambda)$$

This is what was needed to be proven.

Task 2

We have know that the regression matrix is an orthonomal basis. Therefore, it holds that $X^T X = I_N$.

We also know that $r_i^{(j-1)} = t - \sum_{l<i} x_l w_l^{(j)} - \sum_{l>i} x_l w_l^{(j-1)}$. Consequently, one can rewrite $x_i^T r_i^{(j-1)} = x_i^T(t - \sum_{l<i} x_l w_l^{(j)} - \sum_{l>i} x_l w_l^{(j-1)})$. As $X^T X = I_N$, one can say that $x_i^T x_l = 0, \forall l \neq i$. Therefore, it holds that $x_i^T r_i^{(j-1)} = x_i^T t$. Also, as the regression matrix is an orthonomal basis, it holds that $x_i^T x_i = 1$.

Case 1 $|x_i^T r_i^{(j-1)}| > \lambda$

From the problem formulation, we know that:

$$w_i^{(j)} = \frac{x_i^T r_i^{(j-1)}}{x_i^T x_i |x_i^T r_i^{(j-1)}|}(|x_i^T r_i^{(j-1)}| - \lambda)$$

Since $x_i^T x_i = 1$, it holds that:

$$w_i^{(j)} = \frac{x_i^T r_i^{(j-1)}}{|x_i^T r_i^{(j-1)}|}(|x_i^T r_i^{(j-1)}| - \lambda)$$

$$w_i^{(j)} = \frac{x_i^T r_i^{(j-1)} |x_i^T r_i^{(j-1)}|}{|x_i^T r_i^{(j-1)}|} - \lambda \frac{x_i^T r_i^{(j-1)}}{|x_i^T r_i^{(j-1)}|}$$

$$w_i^{(j)} = x_i^T r_i^{(j-1)} - \lambda sgn(x_i^T r_i^{(j-1)})$$

But it was showed before that $x_i^T r_i^{(j-1)} = x_i^T t$. Therefore, it holds that:

$$w_i^{(j)} = x_i^T t - \lambda sgn(x_i^T t)$$

From the last equation, one can notice that $w_i^{(j)}$ does not depend on previous estimates. Consequently, one has that:

$$w_2^{(2)} - w_2^{(1)} = x_i^T t - \lambda sgn(x_i^T t) - x_i^T t + \lambda sgn(x_i^T t) = 0$$

Case 2 $|x_i^T r_i^{(j-1)}| < \lambda$

From the problem formulation, it is known that for this case $w_i^{(j)} = 0$. Consequently, $w_2^{(2)} - w_2^{(1)} = 0 - 0 = 0$.

Task 3

When $\sigma \to 0$, $e \to 0$. We have that $t = x_i w_i^* + e$. Therefore, when $\sigma \to 0$, $\lim_{\sigma\to 0} x_i^T t = \lim_{\sigma\to 0} x_i^T(x_i w^*) = w^*$.

Moreover, from Task 2, we have that $\overline{w}_i^{(1)} = x_i^T t - \lambda sgn(x_i^T t)$.

Also, in Task 2, we have showed that $x_i^T r_i^{(j-1)} = x_i^T t$. Consequently, when $\sigma \to 0$, $x_i^T r_i^{(j-1)} \to w^*$.

Case 1 $x_i^T r_i^{(j-1)} > \lambda$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = \lim_{\sigma\to 0} E(x_i^T t - \lambda sgn(x_i^T t) - w_i^*)$$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = \lim_{\sigma\to 0} E(w_i^* - \lambda sgn(w_i^*) - w_i^*)$$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = -\lambda sgn(w_i^*) = -\lambda$$

Case 2 $x_i^T r_i^{(j-1)} < -\lambda$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = \lim_{\sigma\to 0} E(x_i^T t - \lambda sgn(x_i^T t) - w_i^*)$$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = \lim_{\sigma\to 0} E(w_i^* - \lambda sgn(w_i^*) - w_i^*)$$

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = -\lambda sgn(w_i^*) = \lambda$$

Case 3 $|x_i^T r_i^{(j-1)}| \leq \lambda$
From the problem, we know that $\overline{w}_i^{(j)} = 0, |x_i^T r_i^{(j-1)}| \leq \lambda$.
Therefore, it holds that:

$$\lim_{\sigma\to 0} E(\overline{w}_i^{(1)} - w_i^*) = \lim_{\sigma\to 0} E(0 - w_i^*) = -w_i^*, |w_i^*| \leq \lambda$$

In conclusion, one should notice that the bias of the LASSO estimate increases as λ increases.

Task 4

First, the *lasso_ccd* function, which solves the LASSO optimization problem using cyclic coordinate descent, should be completed. According to equation (6) from the home assignment description, the residual should be computed as:

$$r = t - Xw$$

Then, at lines 56 - 65 from the function code, the current regression vector is selected using $x = X(:, kind)$. In order to put impact of old $w(kind)$ back to the residual, one need to fill in $r = r + x * w(kind)$. Then one updates $w(kind)$ according to equation (3) from the assignment description. This means we have the following:

$$w(kind) = \frac{x^T r(|x^T r| - \lambda)}{x^T x|x^T r|}, |x^T r| > \lambda$$

and $w(kind) = 0$, if $|x^T r| \leq \lambda$.

First, the *lasso_ccd* function should be filled in. After, one needs to call it for every λ value. The considered λ values are: 0.1, 10 and 2. Then one can obtain in the same plot the interpolated reconstruction of data ($Xinterp * w$), the reconstructed data points ($X * w$) and the original data (t).

Figure 1 illustrates the LASSO solution for different values of λ. The solid line in all figures represents the interpolated reconstruction of the data $Xinterp * w$. The big colourful circles represent the reconstructed data points $X * w$, and the black small points represent the original data t. The x-axis represents the time and the y-axis represents the data points. Also, it is known that the LASSO's penalty term promotes sparsity in w by setting the coordinates with small magnitude to zero. Therefore, if λ is too low, we will end up with a lot of non-zero coordinates, and if λ is too high, we will have a lot of zero coordinates. As it can be seen in Figure 1a, for $\lambda = 0.1$, the points are overfitting the data. This is also observable in Table 1, where we have 266 non-zero coordinates. From Figure 1b, one can deduce that a value of 10 is too high for λ, as the points are clearly underfitting the data. The same conclusion can be drawn from Table 1, as we have only 4 non-zero coordinates for this value of λ. The best value for λ seems to be 2. Figure 1c seems to represent a decent LASSO solution that fits the data. Also, in Table 1 we can see that we have 35 non-zero coordinates, which is a reasonable value.

$\lambda = 0.1$	$\lambda = 10$	$\lambda = 2$
266	4	35

Table 1: Non-zero coordinates for different values of λ

Task 5

First of all, the *skeleton_lasso_cv* function needs to be completed. This function computes the LASSO solution and uses cross-validation to train the hyperparameters. For the K-fold algorithm, one needs to specify the validation set and the estimation set for every fold. This can be done by specifying that the validation set is $randomind(location + 1 : location + Nval)$. Then the estimation contains the rest of the samples that are not in the validation set (this can be computed by using the $setdiff$ function in MATLAB). Then, for every λ_j, one computes the LASSO estimate using the previously used function, *skeleton_lasso_ccd*. Then one computes the SE values for both validation and estimation according to the formula from the assignment description. This means that the SE value for validation is:

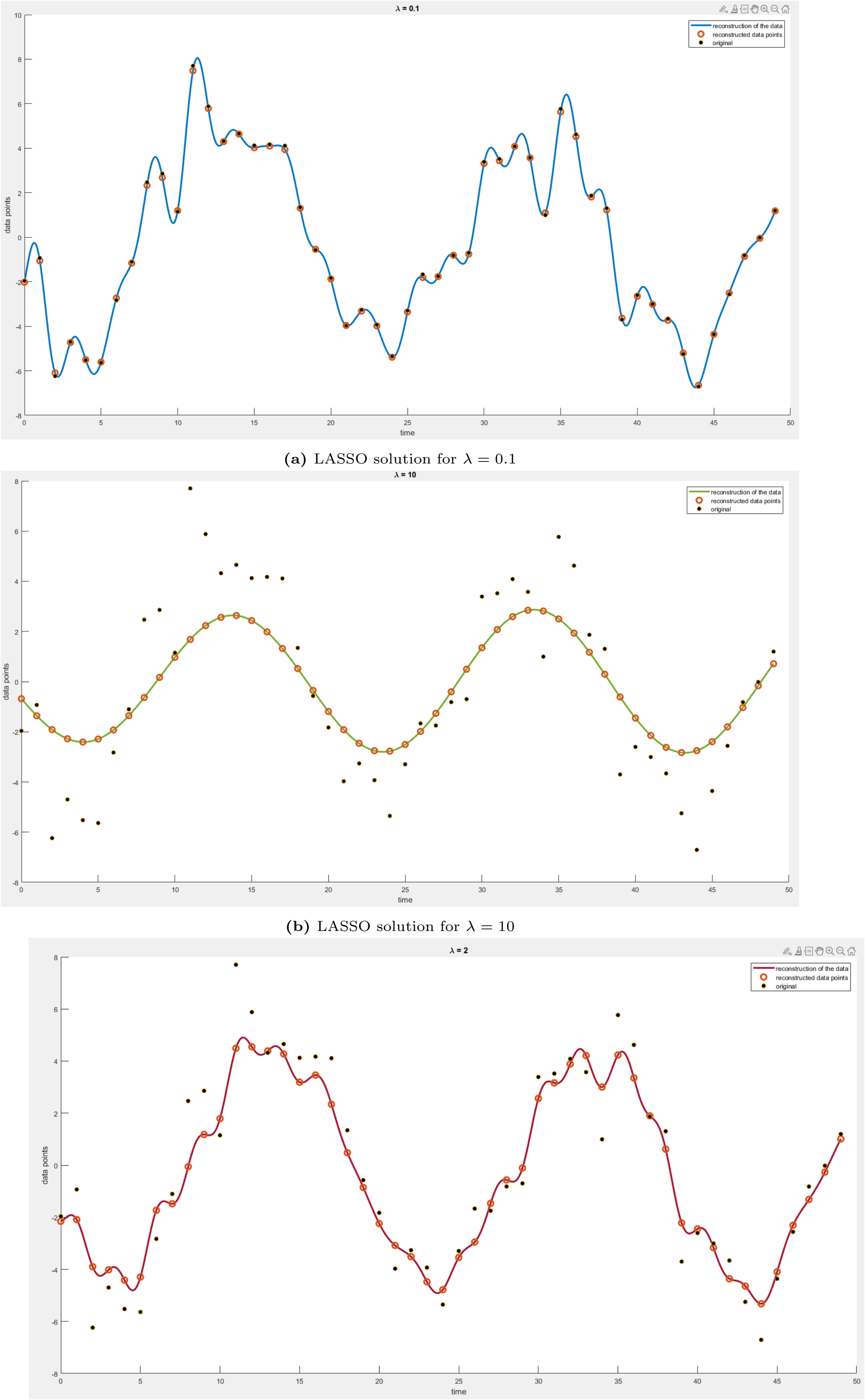
$$SE_{val}(kfold, klam) = Nval^{-1} \|t(valind) - X(valind, :) * what\|^2$$

Similarly, the SE value for estimation is:

$$SE_{est}(kfold, klam) = (N - Nval)^{-1} \|t(estind) - X(estind, :) * what\|^2,$$

where $kfold$ is the current fold, $klam$ is the current value of λ, $Nval$ is the number of samples per fold, $valind$ is the current validation index, $estind$ is the current estimation index, $what$ is the LASSO solution for the current λ, t is the original data, X is the regression matrix.

After completing the K-fold part, one needs to compute the mean of the validation/ estimation error over the folds. The index of the minimum value for the mean of the validation error is the index



(a) LASSO solution for $\lambda = 0.1$

(b) LASSO solution for $\lambda = 10$

(c) LASSO solution for $\lambda = 2$

Figure 1: LASSO solution for different values of λ

of the optimal λ. Then one computes the LASSO estimate w for the optimal λ using the function *skeleton_lasso_ccd*.

After completing the *lasso_cv* function, one needs to call it and make the two required plots. The interval for λ is [0.1, 10], as it needs to be the same as the interval in Task 4. The number of folds used is $K = 5$. After running the algorithm, the **optimal λ value found is 2.1544**. The results can be also visually analyzed in Figure 2.

In Figure 2a, it can be seen the evolution of the RMSE value for validation (blue line), the RMSE value for estimation (orange line), and the logarithm of the optimal λ value ($ln(\lambda_{optimal}) = 0.7675$). The x axis represents the logarithm of the λ values. Th estimation error is growing when λ is increasing. We have seen in Task 4 that for small values of λ, the model is overfitted, meaning that it is very biased. This means that the training (estimation) error is very small. For large values of λ, the model is not so biased, tending to be underfitted when $\lambda = 10$. Anyway, when choosing a model what is really important is the validation error, as this is the one that tells how well the model performs with new data. The validation error is large when λ is small, then it decreases, reaching the minimum at the optimal λ. After this, it increases again for larger values of λ. Figure 2b represents the interpolated reconstruction of the data (purple line) when using the optimal value for λ. There can be seen also the reconstructed data points (orange empty circles) and the original data points (black points). The x-axis represents the time. As it can be seen, the solution neither overfits the data, nor underfits it. The plot is very similar to the plot 1c in Task 4, where $\lambda = 2$.

Task 6

This task is very similar to Task 5, the only difference is that one needs to compute the LASSO solution for all the frames and then train the hyperparameters using cross-validation. Therefore, the *skeleton_multiframe_lasso_cv* needs to be completed. So, an additional for loop is put in order to loop over all the frames. Then inside this loop, the cross-validation is implemented. The validation and the estimation sets are defines, similar to Task 5. For every λ, one needs to compute the LASSO estimation at the current frame and fold. Then one needs to add the validation error at the current frame, fold and lambda to the validation error for this fold and lambda, summing the error over the frames. This is done with:

$$SEval(kfold, klam) = SEval(kfold, klam) + Nval^{-1} * \|t(valind) - X(valind, :) * what\|^2,$$

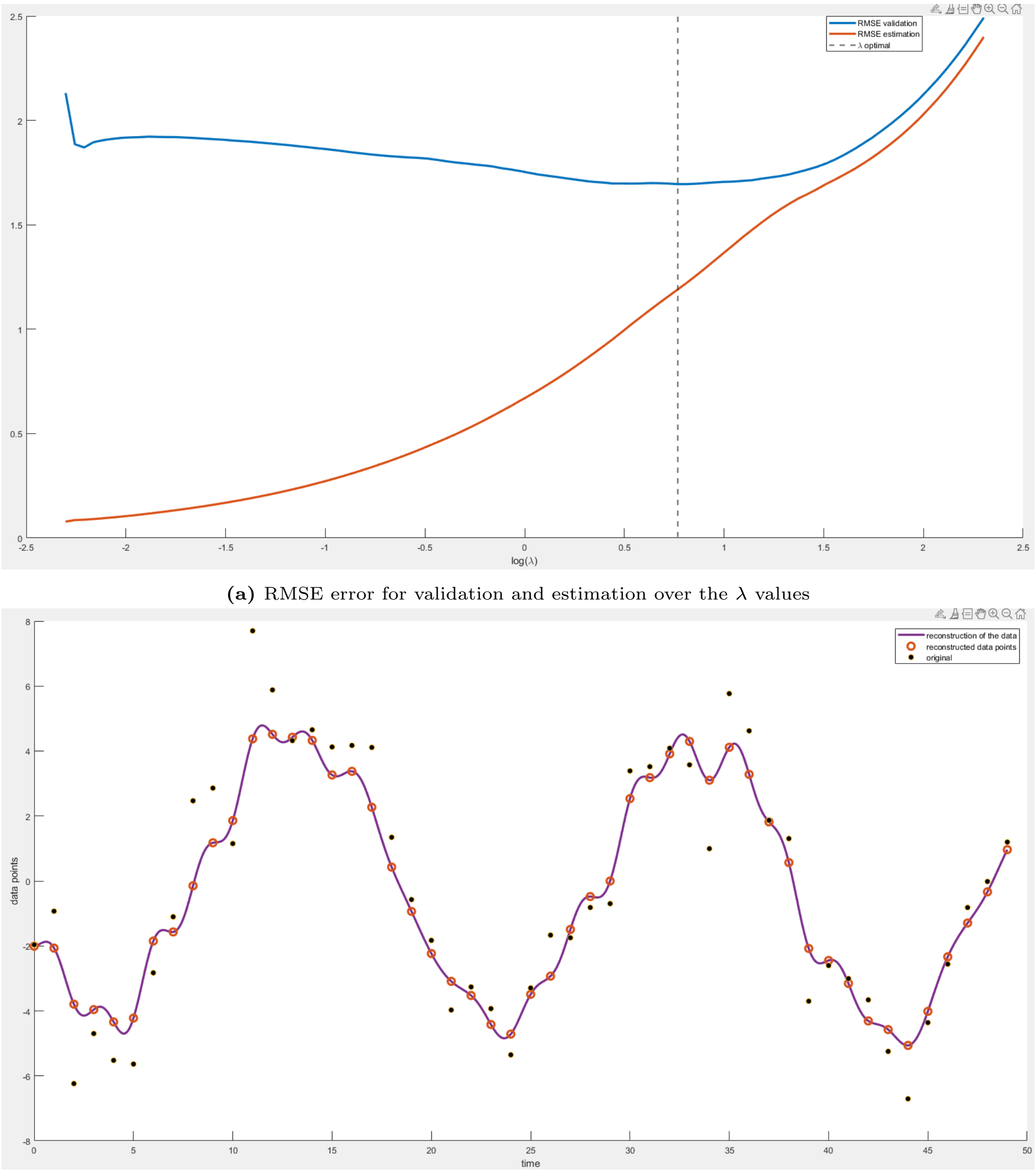
where $kfold$ is the current fold, $klam$ is the current λ, $Nval$ is the number of samples per fold, $SEval$ is the SE value for validation, t is the original data, X is the regression matrix, $what$ is the LASSO estimation.

Similarly, one needs o compute the SE value for estimation.

After completing the K-fold part, one needs to compute the mean of the validation/ estimation error over the folds. The index of the minimum value for the mean of the validation error is the index of the optimal λ. Then one moves through frames and computes the LASSO estimate w for the optimal λ using the function *skeleton_lasso_ccd*.

Then one needs to call the *skeleton_multiframe_lasso_cv* function in order to analyze the results. The considered interval for λ is [0.001, 0.04]. The number of folds used for cross-validation is 5. The results can be analyzed in Figure 3.

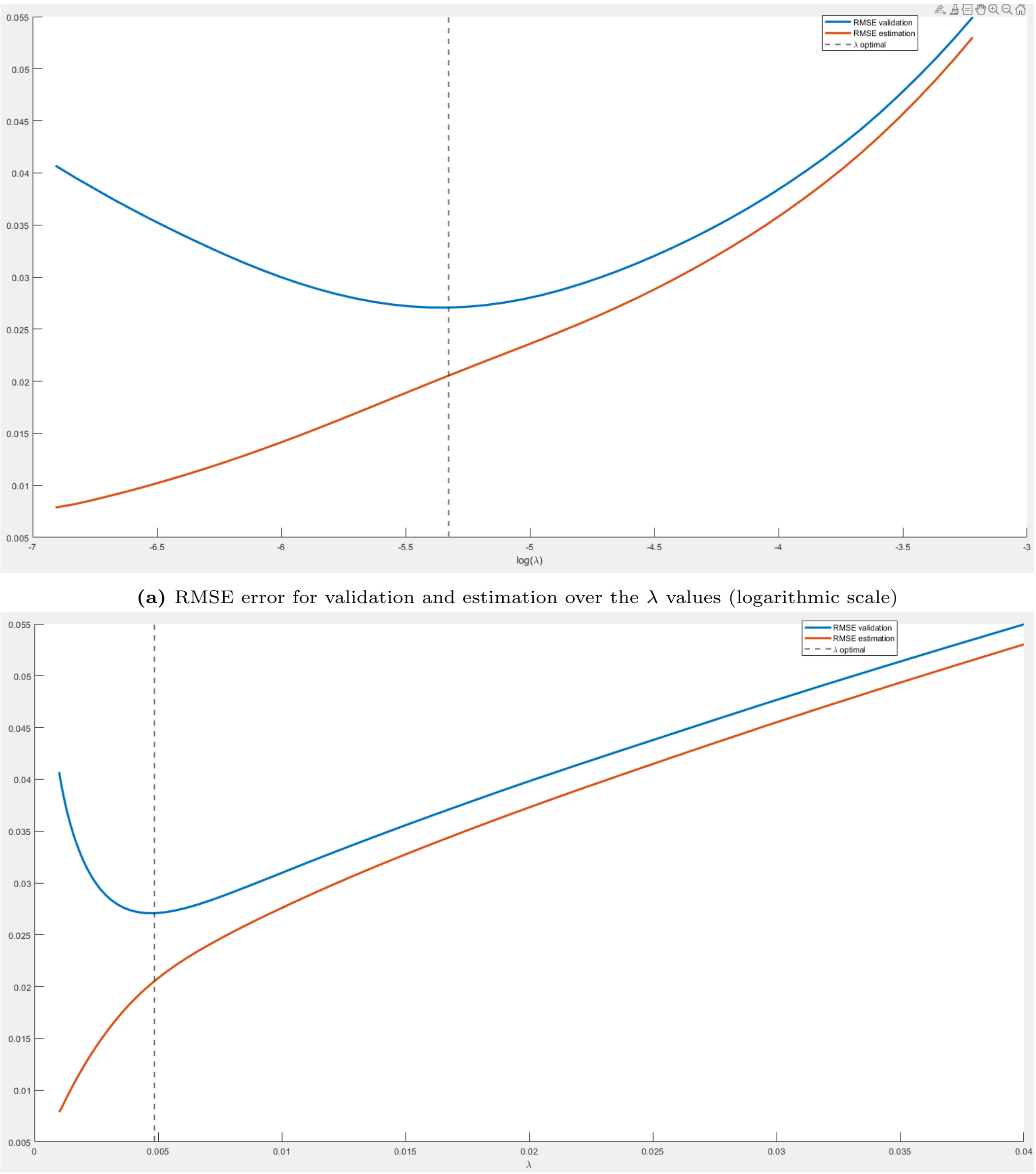
The optimal λ value for the multi-frame audio is 0.0049. Figure 3a represents the RMSE evolution over the logarithmic values of λ (x-axis represents the $log(\lambda)$). Similar to Task 5, the RMSE for estimation (orange line) increases when λ increases. The explanation is the same as in Task 5, for small values of λ, the data is very biased towards the estimation (training) set. The RMSE for validation (blue line) is high when λ is very low (as the data is very biased in that region). Then it



(a) RMSE error for validation and estimation over the λ values

(b) LASSO solution for the optimal λ

Figure 2: Results from the K-fold cross-validation scheme for the LASSO solver implemented in Task 4



(a) RMSE error for validation and estimation over the λ values (logarithmic scale)

(b) RMSE error for validation and estimation over the λ values (normal values)

Figure 3: Results of the K-fold cross-validation algorithm for the multi-frame audio

reaches a minimum at the optimal value of λ (see the grey vertical line). After this, the RMSE is increasing as λ is increasing. As it was expected, the same behaviour can be seen in Figure 3b, but here the x-axis represents exactly the λ values. It can be seen that the optimal λ is found at 0.0049 (grey line), exactly where the RMSE value for validation reaches a global minimum.

Task 7

The data can be denoised using the LASSO estimates for the optimal λ. This is done in function *lasso_denoise*. When using the optimal λ, the noise is reduced, but it can still be heard in the audio. The best value for λ seemed to be 0.01. When using this value, there is almost no noise in the audio. When I increased the values for λ, the piano sound faded away (for $\lambda = 0.05$ for example). When I used lower values for λ (for example, 0.002), there was more noise in the audio. For $\lambda = 0.0005$, there is more noise than piano sounds.