LUNDS
UNIVERSITET

Machine Learning

Assignment 1

Submitted by:

Vahid Faraji

Master's Student in Machine Learning, Systems and Control
Lund University, Sweden

Submission Date: April 11, 2025

2 Hyperparameter-learning via K-fold cross-validation

In this section, you shall consider LASSO regression for the noisy data `t` in `A1_data`, consisting of $N = 50$ data points generated by

$$t(n) = f(n) + \sigma \cdot e(n), \tag{9}$$
$$f(n) = \Re\left(5e^{i2\pi(\frac{n}{20}+\frac{1}{3})} + 2e^{i2\pi(\frac{n}{5}-\frac{1}{4})}\right), \tag{10}$$
$$e(n) \sim \mathcal{N}(0,1) \tag{11}$$

for $n = 0, \ldots, N-1$, i.e., the real part of a sum of two complex-valued sinusoids with frequencies $1/20$ and $1/5$ revolutions per sample, respectively, plus an additive Gaussian noise with standard deviation σ. The regression matrix you'll be using is given by `X` in `A1_data`, which consists of 500 candidate sine and cosine pairs, i.e.,

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 & \ldots & \mathbf{X}_{500} \end{bmatrix} \tag{12}$$
$$\mathbf{X}_i = \begin{bmatrix} \mathbf{u}_i & \mathbf{v}_i \end{bmatrix} \tag{13}$$
$$\mathbf{u}_i = \begin{bmatrix} \sin(2\pi f_i 0) & \sin(2\pi f_i 1) & \ldots & \sin(2\pi f_i (N-1)) \end{bmatrix}^\top \tag{14}$$
$$\mathbf{v}_i = \begin{bmatrix} \cos(2\pi f_i 0) & \cos(2\pi f_i 1) & \ldots & \cos(2\pi f_i (N-1)) \end{bmatrix}^\top, \tag{15}$$

where f_i are the 500 frequency candidates, equally spaced on the interval $[0.02, 0.48]$.

---

1. Implement a (cyclic) coordinate descent solver for the coordinate-wise LASSO solution in (3), using data `t` and regression matrix `X` in `A1_data`. Do **not** use `Xinterp` for estimation, it is provided for visualisation only.

   Hint: To simplify the implementation, you may fill in and use the function `skeleton_lasso_ccd()`.

2.0.1 Solution

We chose a regularization parameter $\lambda = 0.1$. The first 50 estimated weights $\hat{\mathbf{w}}$ were printed using:

```
print(w_hat[:50])
```

Example output

```
[[ 0.13]
 [ 1.21]
 [ 0.00]
 [ 0.67]
 ...
 [ 0.00]]
```

As expected, the LASSO estimate is sparse, meaning many weights are exactly zero due to the ℓ_1-penalty. The non-zero weights correspond to the selected frequency components that explain the signal $\mathbf{t}$, while irrelevant ones are removed.

- **Larger** λ values yield sparser solutions (more weights are zero).
- **Smaller** λ values allow more non-zero weights (denser models).
- **Sign of weights:** Positive and negative values indicate the direction of influence of each feature on the target.
- **Magnitude of weights:** Larger absolute values suggest that the corresponding feature plays a more significant role in prediction.
- **Zero weights:** Features with $\hat{w}_i = 0$ are considered irrelevant by the model and are effectively removed, promoting sparsity.

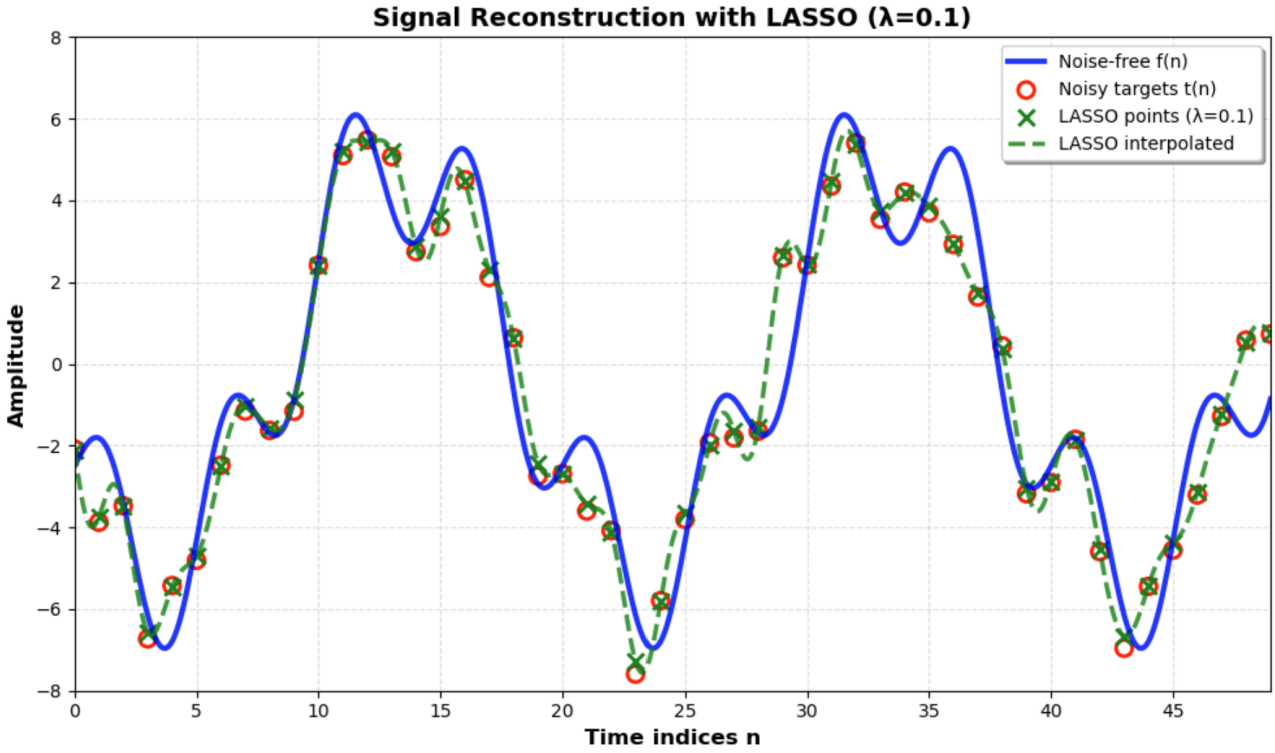
The presence of many zero weights confirms the sparsity property of LASSO, which selects only a subset of relevant features and shrinks others to zero. Also, the result is consistent with the expected behavior of LASSO: reducing overfitting by ignoring irrelevant variables while retaining the informative ones.

---

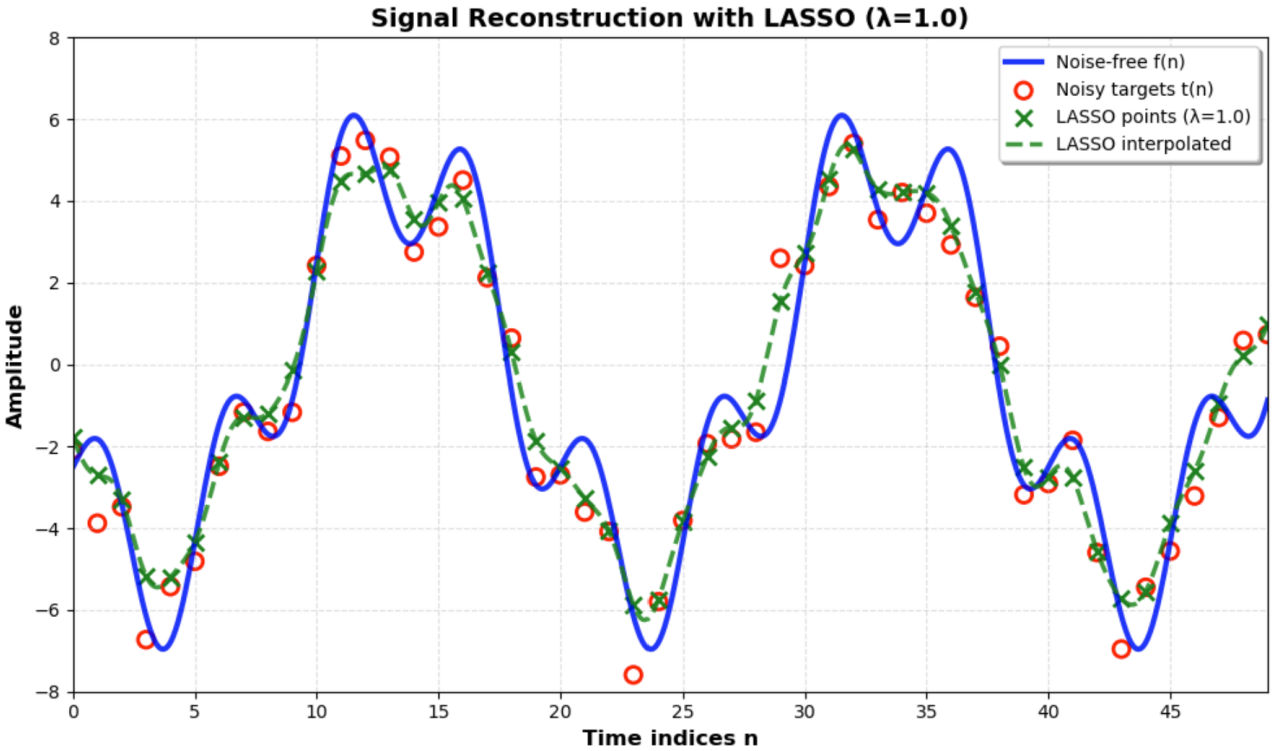
2. Produce reconstruction plots with the following formatting: Plot the original $N = 50$ data points with disconnected markers (no connecting lines) vs. the time axis given by `n`. Overlay these with $N = 50$ reconstructed data points, i.e., $\mathbf{y} = \mathbf{X}\hat{\mathbf{w}}(\lambda)$, also using disconnected markers. In addition, add a solid line without markers for an interpolated reconstruction of the data, vs. an interpolated time axis. The interpolatation can be calculated using a highly sampled regression matrix, given as `Xinterp`. The interpolated time axis is found as `ninterp`. Specify legends for the all three lines, label the x-axis, and adjust line widths, font sizes and so on to make the plot clearly interpretable. Make three variants of this plot, one for each of the following values of the hyperparameter:
   - $\lambda = 0.1$.
   - $\lambda = 10$.
   - Try other values of λ to see the effect of the hyperparameter, and let $\lambda = \lambda_{\text{user}}$ be the one among these you find most suitable. Consider the problems of over- and under-fitting.

2.0.2 Solution

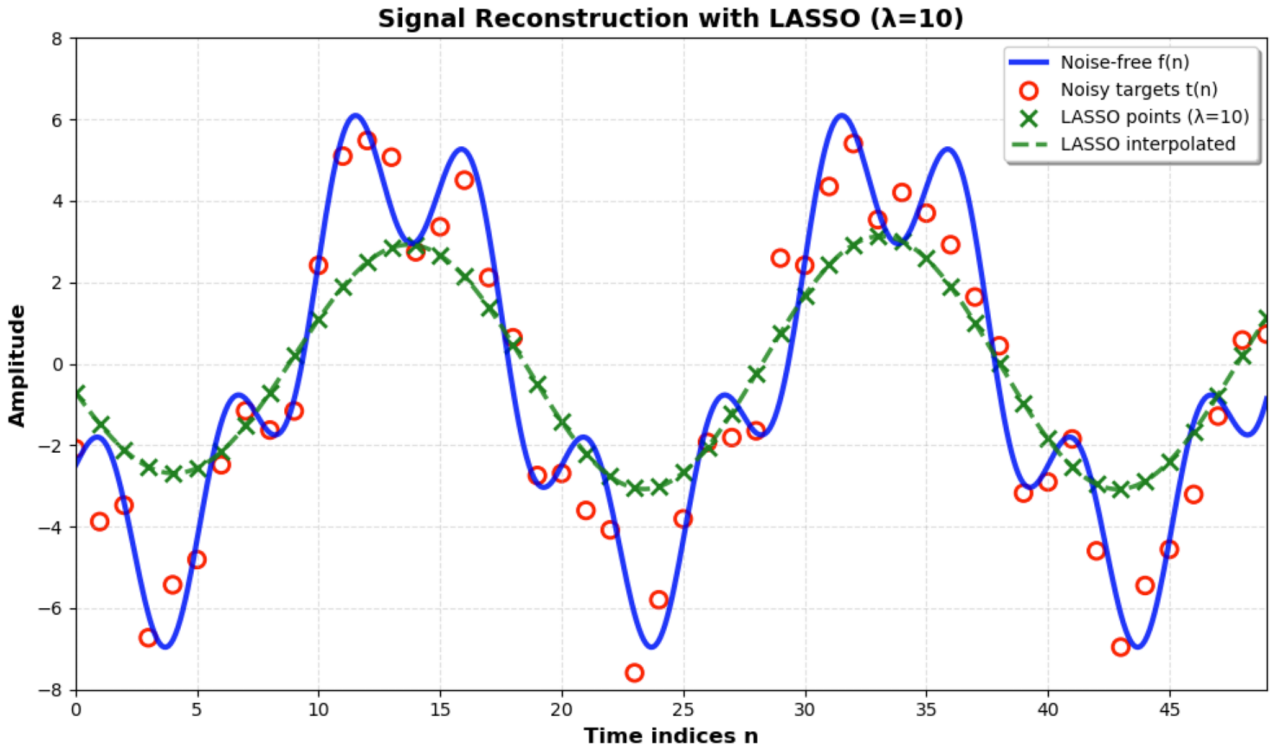
Three LASSO reconstructions were performed for $\lambda = 0.1$, 1.0, and 10.0. The results are summarized below:



- $\lambda = 0.1$ **(Low regularization):** The model closely follows the noisy data, capturing small variations and noise. Although the reconstruction aligns well with the signal $f(n)$, it risks **overfitting** due to low sparsity.



- $\lambda = 10$ **(High regularization):** The model becomes too simplistic. Most weights are zero, leading to a smooth and underfitted reconstruction. Important features of $f(n)$ are lost **(Underfitting)**.



- $\lambda = 1.0$ **(Balanced regularization):** The reconstruction balances fitting accuracy with sparsity. It closely follows $f(n)$ while ignoring some noise. This appears to be the most suitable λ among the three.

Conclusion: Choosing $\lambda = 1.0$ provides the best trade-off between overfitting and underfitting. It allows sufficient flexibility while enforcing sparsity, yielding a reliable reconstruction.

---

3. Count the number of non-zero coordinates for the values of the hyperparameter used above, i.e., $\lambda \in \{0.1, 10, \lambda_{\text{user}}\}$, and compare these to the actual number of nonzero coordinates needed to model the data given the true frequencies, which amounts to two pairs, or four coordinates.

   Hint: Look at Figure 1 at the end of this document and compare the signal with your reconstruction.

2.0.3 Solution

We computed the number of non-zero weights for different values of the regularization parameter λ:

- $\lambda = 0.1$: 213 non-zero weights
- $\lambda = 1.0$: 59 non-zero weights
- $\lambda = 10$: 7 non-zero weights

- For $\lambda = 0.1$, the number of non-zero weights is too high. While the reconstructed signal closely follows the true function, the model is clearly *overfitting*, capturing noise rather than the true underlying structure.

- For $\lambda = 10$, the number of non-zero weights (7) is closest to the true number (4). However, as seen in the reconstruction plot, the signal deviates significantly from the true function due to *underfitting* – the model is too sparse to capture the important features of the data.
- For $\lambda = 1.0$, the number of non-zero weights (59) represents a better compromise. The model is relatively sparse compared to $\lambda = 0.1$, and the reconstructed signal follows the true function quite well, striking a balance between model complexity and data fidelity.

When $\lambda \to 0$, optimization problem Equation (1) reduces to an ordinary least squares problem, which if underdetermined, i.e., $M \geq N$, becomes ill-posed, and/or tends to overfit the data. The LASSO's penalty term promotes sparsity in $\mathbf{w}$ by setting coordinates with small magnitude to zero, thereby reducing the degrees of freedom for the problem. The choice of the hyperparameter λ thus becomes critical; if set too low then too many coordinates become non-zero, thus overfitting the data. If λ is set too high, then too many coordinates will be set to zero, thus underfitting the data. Let $\boldsymbol{\lambda}$ be a grid of J potential values of the hyperparameter, i.e.,

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \dots & \lambda_J \end{bmatrix} \tag{16}$$

where typically λ_1 is selected very small, and λ_J is selected as $\lambda_{\max}$, where,

$$\lambda_{\max} = \max_i \left(\left| \mathbf{x}_i^\top \mathbf{t} \right| \right). \tag{17}$$

To select λ optimally among these candidates, one may for instance use a K-fold cross-validation scheme. In this approach, the training data is randomly split in K non-overlapping and equally sized folds (subsets), where (K-1) folds are used for estimation, and the remaining fold is used for validation. More specifically, let the set $\mathcal{I}_k \subset \{1, \dots, N\}$ denote the $N_{\text{val}} = |\mathcal{I}_k|$ data indices selected for validation in the k:th fold. Thus, the complement of the validation index set, $\mathcal{I}_k^c \subset \{1, \dots, N\}$ denotes the set of $N_{\text{est}} = |\mathcal{I}_k^c| = N - N_{\text{val}}$ data indices used for estimation, such that, by definition, $\mathcal{I}_k \cap \mathcal{I}_k^c = 0$. This partitioning is repeated K times for each potential value of λ, cycling through the non-overlapping validation folds and using it for validation, while using the rest for estimation. To measure the performance of the estimated model when used to predict on the validation data, the root mean squared error on validation ($RMSE_{\text{est}}$) term

$$RMSE_{\text{val}}(\lambda_j) = \sqrt{K^{-1} \sum_{k=1}^{K} N_{\text{val}}^{-1} \left\| \mathbf{t}(\mathcal{I}_k) - \mathbf{X}(\mathcal{I}_k) \hat{\mathbf{w}}^{[k]}(\lambda_j) \right\|_2^2} \tag{18}$$

is used, where $\hat{\mathbf{w}}^{[k]}(\lambda_j)$ denote the weight estimate (herein the LASSO estimate) for the k:th fold using hyperparameter λ_j. Note that $\mathbf{X}(\mathcal{I}_k)$ refers to the (concatenation of) rows of $\mathbf{X}$ indexed by $\mathcal{I}_k$. The hyperparameter is then selected as

$$\hat{\lambda} = \lambda_p, \quad p = \operatorname*{argmin}_j \, RMSE_{\text{val}}(\lambda_j). \tag{19}$$

Algorithm 1 K-fold cross-validation for LASSO

Select grid of hyperparameters $\lambda_j \geq 0$, $\forall j$, and $K > 1$.
Intitialize $SE_{\text{val}}(k, \lambda_j) = 0$, $\forall k, j$.
Randomize validation indices $\mathcal{I}_k$, $\forall k$.
for k = 1, ..., K **do**
  for j = 1, ..., J **do**
    Calculate LASSO estimate $\hat{\mathbf{w}}(\lambda_j)^{[k]}$ on the k:th fold's estimation data.
    Set $SE_{\text{val}}(k, \lambda_j) = N_{\text{val}}^{-1} \left\| \mathbf{t}(\mathcal{I}_k) - \mathbf{X}(\mathcal{I}_k)\hat{\mathbf{w}}^{[k]}(\lambda_j) \right\|_2^2$.
  end for
end for
Calculate $RMSE_{\text{val}}(\lambda_j) = \sqrt{K^{-1} \sum_{k=1}^{K} SE_{\text{val}}(k, \lambda_j)}$.
Select $\hat{\lambda}$ as the λ_j which minimizes $RMSE_{\text{val}}$.

The following algorithm outlines a suggested implementation of a K-fold cross-validation scheme for the LASSO estimator:
To display the generalization gap, the corresponding root mean squared error on estimation data ($RMSE_{\text{est}}$) becomes

$$RMSE_{\text{est}}(\lambda_j) = \sqrt{K^{-1} \sum_{k=1}^{K} N_{\text{est}}^{-1} \left\| \mathbf{t}(\mathcal{I}_k^c) - \mathbf{X}(\mathcal{I}_k^c)\hat{\mathbf{w}}^{[k]}(\lambda_j) \right\|_2^2} \tag{20}$$

Exercise 5 - 20 (6) points: Implement a K-fold cross-validation scheme for the LASSO solver implemented in Task 4. Illustrate your results using the following plots and make detailed comments for them:

1. A plot illustrating $RMSE_{\text{val}}(\lambda_j)$ vs. $\lambda_j, \forall j$, using a solid line with markers. Overlaid in the same plot, also plot the corresponding $RMSE_{\text{est}}(\lambda_j)$ using a solid line with different markers. Also add a dashed vertical line at the location of the selected $\hat{\lambda}$. Specify legends for the all three lines, label the x-axis, and adjust line widths, font sizes and so on to make the plot clearly interpretable. Depending on your choice of λ-values you may want to use a logarithmic x-axis to make the plot more visible.

2.0.4 Solution

The RMSE was computed for both the validation and estimation sets across a logarithmically spaced range of λ values from 0.01 to 100.

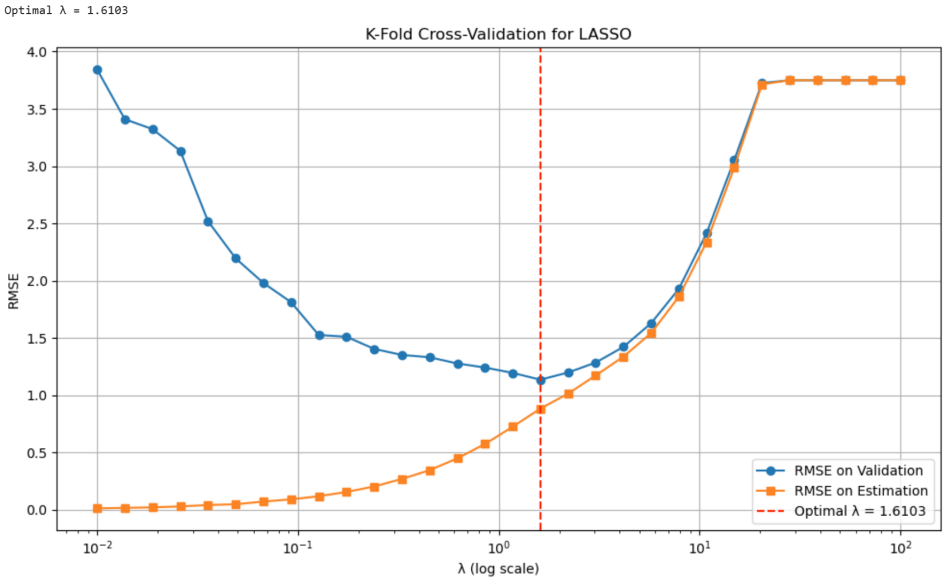


Figure 1: K-Fold Cross Validation

As shown in Figure 1, when λ is too small (e.g., $\lambda = 0.01$), the model tends to overfit the noisy data, resulting in a low estimation error but high validation error. Conversely, when λ is too large (e.g., $\lambda = 100$), the model underfits and performs poorly on both training and validation data.

The optimal value $\hat{\lambda} = 1.6103$ was selected by minimizing the validation RMSE. This value provides a good trade-off between bias and variance, resulting in a low validation error and moderate model complexity. This confirms the theoretical behavior of the LASSO penalty, promoting sparsity and avoiding overfitting.

---

2. A reconstruction plot for the selected $\hat{\lambda}$. Use the same layout and formatting as described for the reconstruction plots in Task 4.

   Hint: For simpler implementation, you may fill in and use the function sketch `lasso_cv()`. Also, for improved vizualization and efficiency, select your grid of candidate λ to be evenly spaced on a log-scale. I MATLAB you can use `lambda_grid = exp( linspace( log(lambda_min), log(lambda_max), N_lambda))` while equivalent methods exist in Python.

2.0.5 Solution

Signal Reconstruction Using Optimal $\hat{\lambda}$

Figure 2 shows the reconstructed signal using the LASSO estimate for the optimal hyperparameter value $\hat{\lambda} = 1.6103$, as determined via K-fold cross-validation. The use of the optimal λ achieves a good balance between model complexity and data fidelity. This confirms that the selected $\hat{\lambda}$ results in a sparse solution with strong generalization capability.

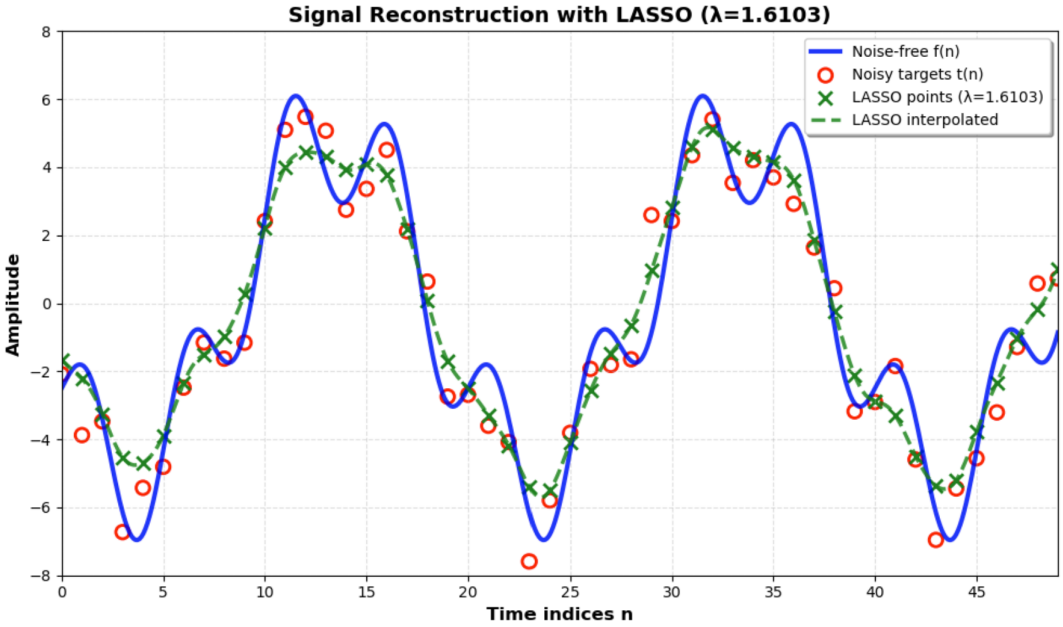


Figure 2: Signal reconstruction using LASSO for $\hat{\lambda} = 1.6103$