Lab Course SS2022:
Data Science in Astrophysics
Session 5: Robust Regression

Florian List[*]

April 7, 2022

[*]florian.list@univie.ac.at

1 Very brief recap of Lecture 4

1. Probabilistic model for the data: $y_n = f(\mathbf{x}_n) + \varepsilon_n$, where $\varepsilon_n \sim \mathcal{N}(0, \sigma_n^2)$.

   Equivalently: $p(y_n \mid \mathbf{x}_n) = \mathcal{N}(f(\mathbf{x}_n), \sigma_n^2)$, where $f(\mathbf{x}_n) = \boldsymbol{\phi}^\top(\mathbf{x}_n)\boldsymbol{\theta}$ with the feature vector $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^K$, $\mathbf{x}_n \mapsto \boldsymbol{\phi}(\mathbf{x}_n)$. Note that f is linear in the parameters $\boldsymbol{\theta} \in \mathbb{R}^K$.

2. Negative log-likelihood: $-\log p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^N \log p(y_n \mid \mathbf{x}_n, \boldsymbol{\theta})$, where

$$\log p\left(y_n \mid \mathbf{x}_n, \boldsymbol{\theta}\right) = -\frac{1}{2\sigma_n^2}\left(y_n - \boldsymbol{\phi}^\top(\mathbf{x}_n)\boldsymbol{\theta}\right)^2 - \frac{\log(\sigma_n^2)}{2} - \frac{\log(2\pi)}{2}. \tag{1}$$

3. Likelihood maximization ($\rightarrow$ negative log-likelihood minimization):

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta})^\top \mathbf{C}^{-1}(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}) \quad \rightarrow \quad \min, \tag{2}$$

   where $\boldsymbol{\Phi} \in \mathbb{R}^{N\times K}$ is the feature matrix, and $\mathbf{C}^{-1} \in \mathbb{R}^{N\times N}$ is the precision matrix.

4. Solution of the likelihood maximization:

$$\boldsymbol{\theta}^\star = \left(\boldsymbol{\Phi}^\top \mathbf{C}^{-1}\boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^\top \mathbf{C}^{-1}\mathbf{y}. \tag{3}$$

2 Overfitting and underfitting

In the last lecture, we have learned how to find the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^K$ that maximizes the likelihood $p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})$. We assumed that the family of functions $\{f_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \mathbb{R}^K\}$ was known and that maximizing the likelihood for the K parameters $\boldsymbol{\theta}$ led to the "optimal" fit. In this lecture, we will take a step back from this perspective and ask ourselves:

- How many parameters are needed / suited for a "good" description of the underlying random process?
- How to make sure our fit is **robust** to outliers / unphysical data?

We will start with the first question and consider **underfitting** and **overfitting**. For this purpose, let us consider an artificial dataset containing $D = 1$-dimensional input-output pairs (x_n, y_n), $n = 1, \ldots, N$, generated as $x_n \sim U(0, 1)$, and

$$y_n = \sin(2\pi x_n) + \varepsilon, \tag{4}$$

where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.1$. We will fit polynomials of different degrees to this data. As a criterion for how "good" our fit is, we will use the **root mean square error** (RMSE), defined as

$$\mathrm{RMSE}(\mathcal{X}, \mathcal{Y}, \boldsymbol{\theta}) = \sqrt{\frac{1}{N}\|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2} = \sqrt{\frac{1}{N}\sum_{n=1}^N \left(y_n - \boldsymbol{\phi}^\top\left(\mathbf{x}_n\right)\boldsymbol{\theta}\right)^2}. \tag{5}$$

The RMSE has the same units as the quantity y, so we can directly compare its magnitude to the magnitude of the data. Another advantage of the RMSE is that it is normalized by the number of samples N in the dataset, allowing us to compare the RMSEs between datasets of different size.

Figure 1 shows the least-square error for the training data $\mathcal{D}$ given by the blue points, for $N = 10$, for different polynomial degrees r as indicated above each panel (note that the number of free parameters is $K = r + 1$ because of the additional parameter for the intercept). Clearly, polynomials with $r \leq 2$ provide a very bad fit to the data. For $r \in [3, 6]$, the least-square polynomials represent a decent approximation of the underlying sine function. For $r \geq 7$, the least-square estimate starts to overfit the data. Notice that the least-square polynomial of degree $r = N - 1$ will pass through *all* the data points. If there is additional data available that has been generated by the same random process, but has *not* been used for fitting the parameters $\boldsymbol{\theta}$ (so-called **test data**), we can use this test data to assess the generalization properties of our fit. This is shown in Fig. 2 (*left*), which shows the RMSE for the training data $\mathcal{D}$ (based on which the $r+1$ polynomial coefficients are determined in each case) and $N_{\text{test}} = 100$ test data points, which were drawn from the same distribution as the training data. A polynomial order of $r = 5$ leads to the smallest test error (similar to that for $r = 3, 4$, and 6. For smaller and larger values of r, the polynomial clearly underfits and overfits, respectively, as indicated by the increasing test error. Notice that the training error continues decreasing as the degree of the polynomial increases. Also when training more complex machine learning methods such as neural networks, it is crucial to evaluate their performance on a separate dataset in order to detect overfitting and ensure the learned mapping from inputs to outputs generalizes to unseen data. In order not to "waste" too much data and to mitigate the impact of stochasticity, techniques such as leave-p-out cross validation or k-fold cross validation can be applied. The right panel in Fig. 2 shows how the magnitude of the best-fit parameters $\boldsymbol{\theta}^\star$ increases as the

3 Regularized regression

3.1 Maximum a-posteriori estimation

So far, we have maximized the likelihood function to find the best-fit parameters $\boldsymbol{\theta}^\star$, namely $p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})$. If we have prior knowledge about the parameters $\boldsymbol{\theta}$, we can include this information when performing a fit and determine a **maximum a-posteriori estimate** rather than a maximum likelihood estimate. Recall **Bayes' theorem**, which states that

$$p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y}) = \frac{p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{Y} \mid \mathcal{X})}, \tag{6}$$

where again $\mathcal{X}$ and $\mathcal{Y}$ are the inputs and observations together making up the training data $\mathcal{D}$. Here, $p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y})$ is the **posterior**, $p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})$ is the likelihood (our previous maximization objective), $p(\boldsymbol{\theta})$ is the **prior** for the parameters $\boldsymbol{\theta}$, and $p(\mathcal{Y} \mid \mathcal{X})$ is known as the **evidence**, which is constant with respect to $\boldsymbol{\theta}$ and can therefore be ignored

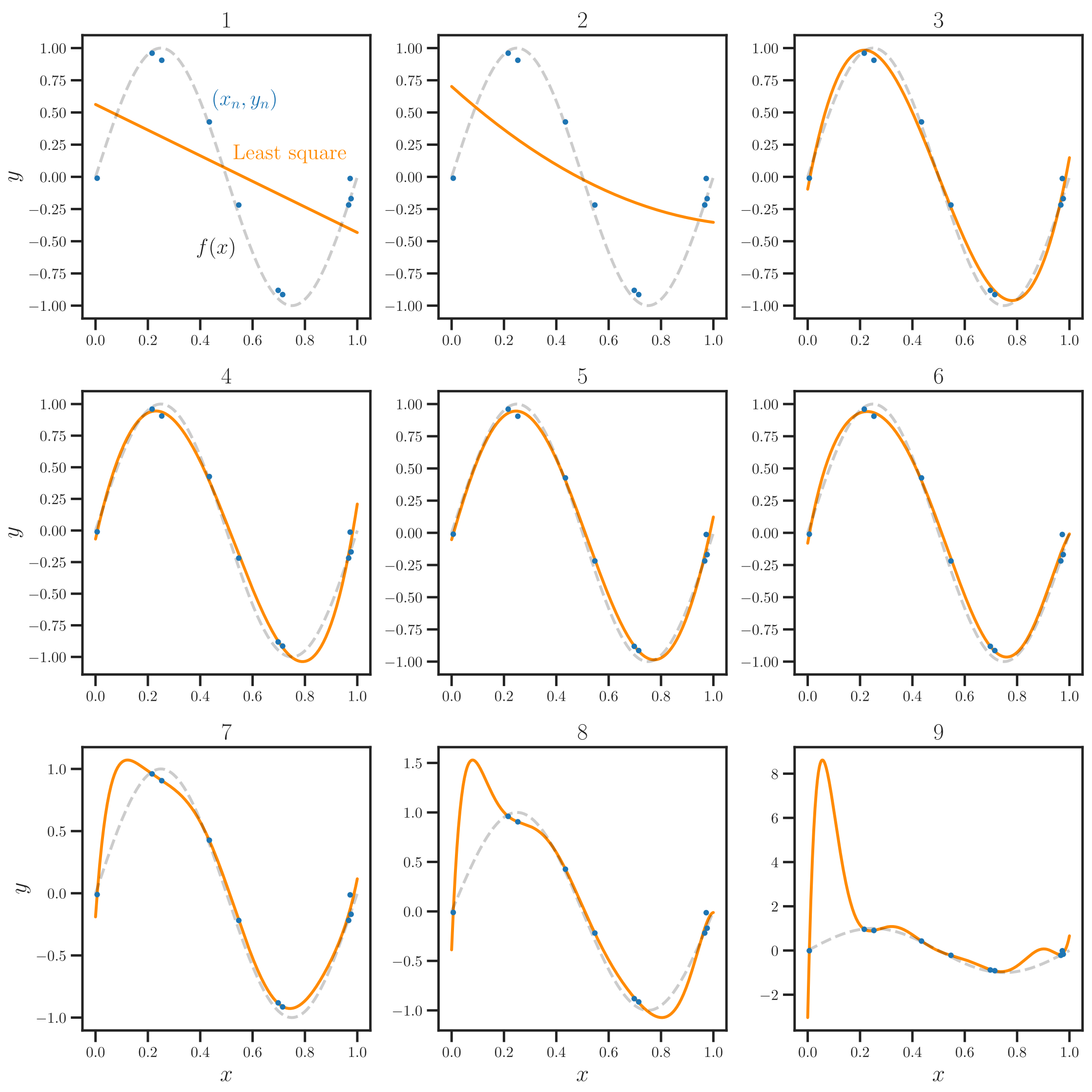

Figure 1: Polynomial least-square fits to the data $y_n = \sin(2\pi x_n) + \varepsilon$ for different polynomial degrees r as indicated above each panel. For low degrees, the polynomials are not expressive enough to approximate the underlying function. On the other hand, for high degrees the polynomials overfit the data, and generalization to unseen data points will be poor.

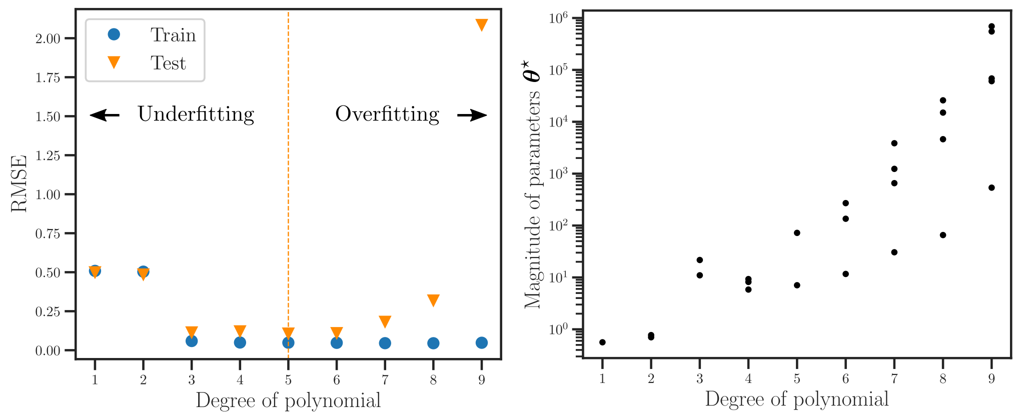

Figure 2: *Left*: Root mean square error (RMSE) for the fits in Fig. 1, computed on the training data (based on which the fit was done) and unseen test data. While the train error decreases monotonically as a function of the polynomial degree (and will be zero when $r = N - 1$), the test error starts increasing again for $r > 5$, indicating overfitting. *Right*: Magnitude of the least square fit parameters $\boldsymbol{\theta}^\star$ as a function of the polynomial degree r. The large magnitude of the parameters for large r is an indicator for overfitting.

in what follows. Our aim is now to determine the maximum a-posteriori estimate $\boldsymbol{\theta}^\star$ as

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left\{ p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})\, p(\boldsymbol{\theta}) \right\}. \tag{7}$$

Taking the logarithm of Eq. (6) yields

$$\log p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y}) = \log p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) + \text{const.} \tag{8}$$

Hence, when going from maximum likelihood estimation to maximum a-posteriori estimation, the prior provides an extra additive term that expresses our prior beliefs about the realistic range that the parameters $\boldsymbol{\theta}$ can be expected to lie in. Let us consider the case of Gaussian homoscedastic uncertainties $\sigma_n^2 = \sigma^2$ for $n = 1, \ldots, N$ and a Gaussian (i.e. conjugate) prior on $\boldsymbol{\theta}$ with variance b^2, i.e. $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, b^2\mathbf{I})$, where $\mathbf{I}$ is the identity matrix. Then, we have

$$-\log p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y}) = \frac{1}{2\sigma^2}(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta})^\top(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}) + \frac{1}{2b^2}\boldsymbol{\theta}^\top\boldsymbol{\theta} + \text{const.} \tag{9}$$

Taking the gradient with respect to $\boldsymbol{\theta}$ and setting it to zero yields

$$\mathbf{0}^\top = -\frac{\mathrm{d}\log p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y})}{\mathrm{d}\boldsymbol{\theta}} = \frac{1}{\sigma^2}\left(\boldsymbol{\theta}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi} - \mathbf{y}^\top\boldsymbol{\Phi}\right) + \frac{1}{b^2}\boldsymbol{\theta}^\top, \tag{10}$$

from which one finds the maximum a-posterior estimate $\boldsymbol{\theta}^\star$ to be

$$\boxed{\boldsymbol{\theta}^\star = \left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{\sigma^2}{b^2}\mathbf{I}\right)^{-1}\boldsymbol{\Phi}^\top\mathbf{y}.} \tag{11}$$

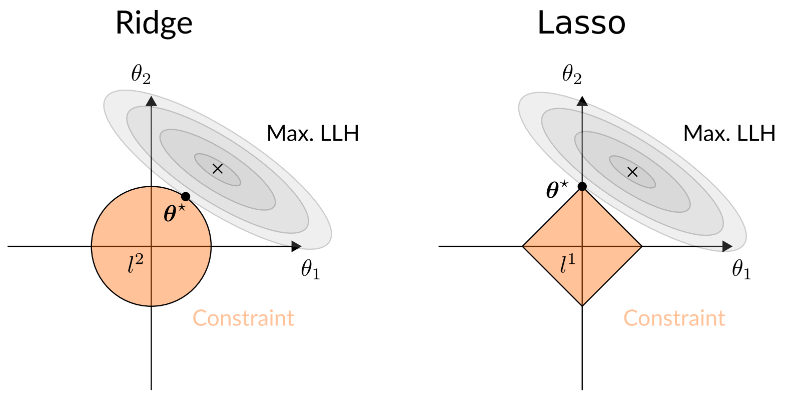

Figure 3: Comparison between ridge regression (Eq. (13)) and Lasso regression (Eq. (14)) for a regularized linear regression problem with $K = 2$ parameters $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top$. The gray contours show lines along which the log-likelihood term $\|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2$ is constant, and the solution of the unregularized minimization problem (i.e. $\lambda = 0$) is marked with a cross. Ridge and Lasso regression enforces the norm – l^2 and l^1, respectively – of $\boldsymbol{\theta}$ to be sufficiently small as indicated by the orange regions. The black dots show the resulting solutions of the regularized regression problems in each case. Notice that l^1-balls around the origin have "corners" where a component of $\boldsymbol{\theta}$ vanishes, making it more likely that the intersection between the gray and orange areas falls on one the corners and thus leading to sparsity. This is not the case for ridge regression due to the symmetric shape of l^2-balls, for which points with a vanishing component θ_k are not special in any way.

Comparing this result to the maximum likelihood estimate in the previous lecture, it becomes apparent that including the prior $p(\boldsymbol{\theta})$ produces an additional term $\frac{\sigma^2}{b^2}\mathbf{I}$, which vanishes as the prior becomes uninformative for $b^2 \to \infty$. Notice the interplay between the prior and the likelihood in Eq. (11): if the data uncertainties σ^2 are small in comparison to variance of the Gaussian prior b^2, the prior term will be small too, and the fit will be driven by the likelihood term $\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$. The prior term ensures that the matrix $\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{\sigma^2}{b^2}\mathbf{I}\right)$ is positive definite and therefore invertible.

3.2 Ridge regression and Lasso regression

Let us go back again to the negative log-posterior in Eq. (9), which we can rewrite as

$$-\log p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \frac{1}{2b^2}\|\boldsymbol{\theta}\|_2^2 + \text{const.} \tag{12}$$

Setting $\lambda := \frac{\sigma^2}{b^2}$, we find that minimizing the negative log-posterior is equivalent to minimizing the loss function

$$\mathcal{L}(\boldsymbol{\theta}) = \|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_2^2, \tag{13}$$

which is known as **ridge regression**, a special case of **regularized least square** regression. While the first term is the same as for standard least square estimation, the second term (which, as we have seen, can be interpreted in a Bayesian sense as the negative log-prior on $\boldsymbol{\theta}$) penalizes large weight magnitudes, where λ is the regularization parameter. Depending on the context, this form of regularization is also known as (a particular form of) **Tikhonov regularization**.

Another popular robust method for regression is obtained by replacing the l^2-norm in Eq. (13) by the l^1-norm, known as **Lasso regression** (where Lasso is an acronym for "least absolute shrinkage and selection operator"):

$$\mathcal{L}(\boldsymbol{\theta}) = \|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1. \tag{14}$$

From a Bayesian point of view, the l^1 regularizer can be interpreted as the log-prior of a Laplace distribution rather than a normal distribution. Minimizing the l^1-norm rather than the l^2-norm encourages **sparsity** of the parameter vector because of the geometry induced by the l^1-norm, as illustrated in Fig. 3.

Let us revisit the artificial dataset shown in Fig. 1 and apply ridge and Lasso regression. Specifically, we consider a polynomial of degree $r = 8$ for which the least-square estimate significantly overfitted the data, and we minimize the ridge and Lasso objectives in Eq. (13) and Eq. (14), respectively, with a regularization parameter of $\lambda = 10^{-3}$. Figure 4 compares the minimizers of the different loss functions. As expected, the regularized solutions are much smoother than the least-square solution. The right panel reveals that the regularization terms cause a vast reduction in the magnitude of the coefficients $\boldsymbol{\theta}^\star$, which drop from $> 10^4$ to < 10 due to the regularization. The Lasso and ridge fits are qualitatively very similar; however, three coefficients of the Lasso-regularized estimate $\boldsymbol{\theta}^\star$ vanish, which is not the case for the ridge-regularized solution, demonstrating that the l^1-norm encourages sparsity.

Finally, let us mention that Lasso regression and ridge regression can also be combined by including both an l^1 and an l^2 regularizer. This is known as **elastic net regularization**.

3.3 An application of Lasso regression: compressed sensing

The fact that the l^1-norm promotes sparsity finds usage in many different applications. We will consider **compressed sensing** as an example here and give a very brief overview of the topic – a much more in-depth introduction can be found in Ref. [2, Sec. 3.2]. Under certain conditions, compressed sensing allows one to reconstruct data such as audio signals, images, etc. from very few measurements. A fundamental theorem in signal processing is the **Nyquist–Shannon sampling theorem**, which states:

Theorem 1 (Nyquist–Shannon)**.** *If a function $x(t)$ contains no frequencies higher than f $[1/s]$, sampling its ordinates in a series of points spaced $\frac{1}{2f}$ $[s]$ completely determines the function $x(t)$.*

This theorem provides a *sufficient* condition to recover a band-limited function $x(t)$ through sampling. Figure 5 shows a scenario where the sampling rate is not as high as

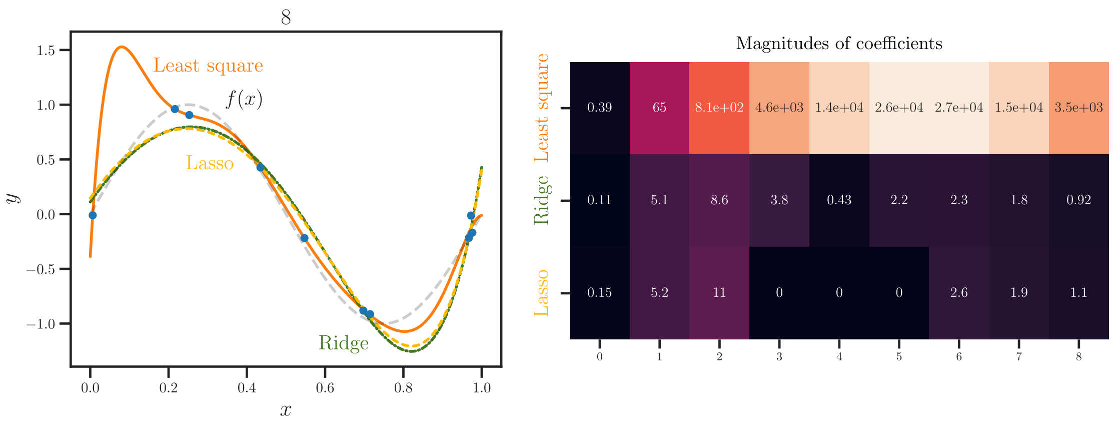

Figure 4: *Left*: Regressing the data defined by Eq. (4) with a polynomial of degree $r = 8$ by minimizing the least-square error, the ridge regression objective in Eq. (13), and the Lasso regression objective in Eq. (14). The minimizers of the regularized regression problems vary much slower than for the unconstrained least-square optimization. *Right*: Magnitudes of the coefficients $\boldsymbol{\theta}$ for each regression method. The regularization strongly reduces the parameter magnitudes, and Lasso regression causes three components to completely vanish.

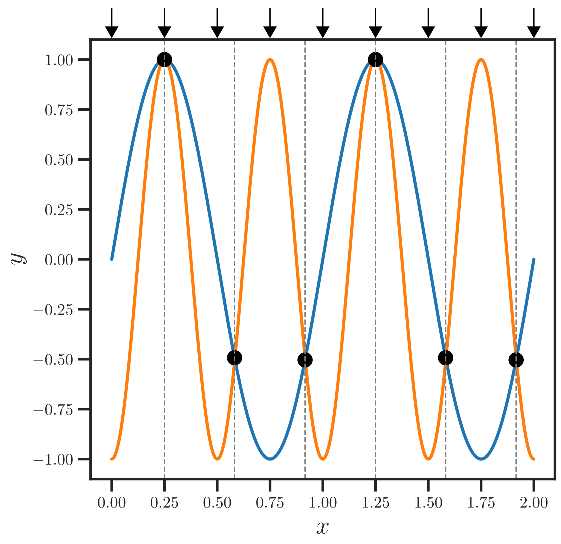

Figure 5: The two sine waves show an example for a situation where the conditions of the Nyquist–Shannon sampling theorem are not satisfied, and aliasing occurs. Specifically, based on sampling the signals at locations indicated by the black dots, it is not possible to distinguish between the orange and the blue signal. The black arrows above the plot correspond to uniformly spaced sampling at the Nyquist rate, which would be sufficient to prevent aliasing.

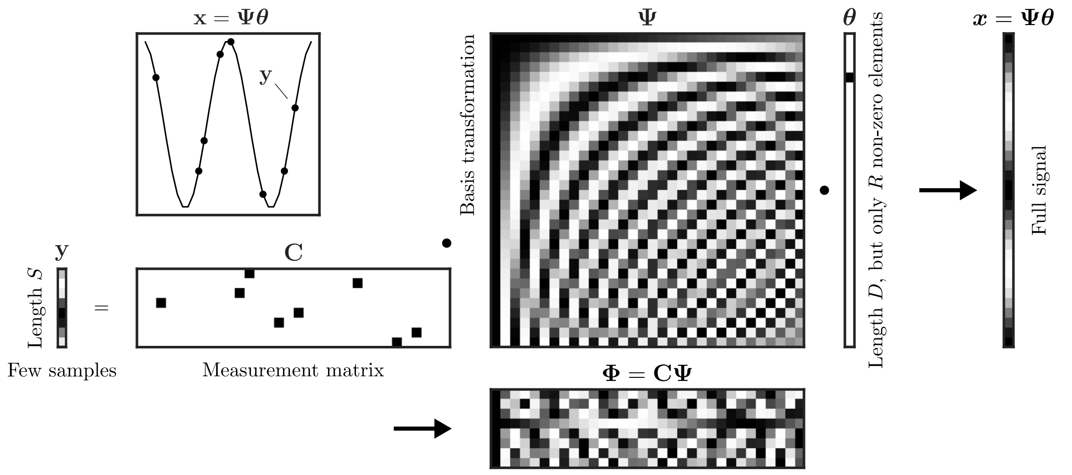

Figure 6: Matrices and vectors involved in the compressed sensing task. The full D-dimensional signal is given by $\mathbf{x}$, which is sparse in a basis represented by the transformation $\boldsymbol{\Psi}$ (e.g. Fourier basis or wavelet basis) such that $\mathbf{x} = \boldsymbol{\Psi\theta}$ with a coefficient vector $\boldsymbol{\theta}$ that only has $R < D$ non-zero elements (in the illustration $R = 1$). The measurement matrix $\mathbf{C}$ extracts $S \ll D$ samples from the full signal, which are gathered in the vector $\mathbf{y}$. Note that $\mathbf{C}$ could also have more than one non-zero element per row, which would correspond to taking linear combinations of values in $\mathbf{x}$. The compressed sensing task consists in recovering the full signal $\mathbf{x}$ (or equivalently the coefficients $\boldsymbol{\theta}$ in the sparse basis $\boldsymbol{\Psi}$). This is an underdetermined problem (the matrix $\boldsymbol{\Phi} := \mathbf{C}\boldsymbol{\Psi}$ that relates $\mathbf{y}$ and $\boldsymbol{\theta}$ as $\mathbf{y} = \boldsymbol{\Phi\theta}$ is wider than high, so generally infinitely many solutions exist), but one can impose the additional requirement that the solution be as *sparse* as possible. This leads to a Lasso optimization problem, see Eq. (20).

required by the Nyquist–Shannon sampling theorem, and **aliasing** occurs. That is, both continuous signals yield the same measurements at the sampling locations.

Although the requirements for the Nyquist–Shannon theorem are sharp for general functions, one can hope to recover the signal from measurements if the signal is **sparse** with respect to some basis $\boldsymbol{\Psi} \in \mathbb{R}^{D\times D}$, which could be given by e.g. the Fourier modes or wavelets (see Lecture 3). Recall from Lab 3 that in many situations, much of the information contained in the data can be recovered from only a few coefficients, which is the underlying principle of compression methods such as JPEG. If we know that a signal $\mathbf{x} \in \mathbb{R}^D$ is R-sparse in the basis $\boldsymbol{\Psi}$ (i.e., it can be represented by only R basis vectors), we can try to reconstruct $\mathbf{x}$ from a much sparser measurement vector $\mathbf{y} \in \mathbb{R}^S$, with $R \leq S \ll D$. We can relate the sparse measurements $\mathbf{y} \in \mathbb{R}^S$ to the full signal $\mathbf{x} \in \mathbb{R}^D$ via

$$\mathbf{y} = \mathbf{C}\mathbf{x}, \tag{15}$$

where $\mathbf{C} \in \mathbb{R}^{S\times D}$ takes S linear measurements on the state vector $\mathbf{x}$. Denoting the coefficients of $\mathbf{x} \in \mathbb{R}^D$ in the basis $\boldsymbol{\Psi} \in \mathbb{R}^{D\times D}$ as $\boldsymbol{\theta} \in \mathbb{R}^D$, we can try to find the sparsest vector $\boldsymbol{\theta}$ that explains the measurements $\mathbf{y} \in \mathbb{R}^S$, i.e.

$$\mathbf{y} = \mathbf{C}\boldsymbol{\Psi}\boldsymbol{\theta} =: \boldsymbol{\Phi}\boldsymbol{\theta}, \tag{16}$$

where we defined $\boldsymbol{\Phi} := \mathbf{C}\boldsymbol{\Psi}$. Since we only have $S \ll D$ measurements to determine the vector $\boldsymbol{\theta} \in \mathbb{R}^D$, this is an underdetermined system of equations with infinitely many solutions (see Fig. 6 for an illustration). However, the *sparsest* coefficient vector $\boldsymbol{\theta}$ can be found as the solution to the following optimization problem:

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \|\boldsymbol{\theta}\|_0 \quad \text{subject to} \quad \mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\theta}. \tag{17}$$

Here, $\|\cdot\|_0$ denotes the l^0-pseudo-norm[1] that counts the number of non-zero entries of a vector. Equation (17) represents a combinatorical problem that is NP-hard: assuming the number R of required non-zero coefficients $\theta_1, \ldots, \theta_D$ is known, all possible combinations for the R out of D non-zero components need to be considered, which makes it unfeasible even for relatively small problem sizes. If R is not known, the problem becomes even more difficult because different values of R must be tried out in addition. Fortunately, it can be shown that under certain conditions [4], one can replace the l^0-pseudo-norm by the l^1-norm and instead consider the problem

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \|\boldsymbol{\theta}\|_1 \quad \text{subject to} \quad \mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\theta} \tag{18}$$

to obtain the same unique solution $\boldsymbol{\theta}^\star$ as in Eq. (17). The great advantage of Eq. (18) is that it can be solved in *polynomial* time, in contrast to the *exponentially* growing search space in Eq. (17). We will summarize the essence of the two conditions required for the solution of Eq. (18) to converge to the sparsest solution in Eq. (17) in what follows. More mathematical background and rigorous statements of the necessary conditions can be found in [2] and the references therein.

1. The measurement matrix $\mathbf{C}$ must be *incoherent* (loosely speaking sufficiently *random*) with respect to the basis $\boldsymbol{\Psi}$. That is, the rows of $\mathbf{C}$ (each of which extracts one measurement from $\mathbf{x}$) should not be correlated with the columns of $\boldsymbol{\Psi}$.
2. The number of measurements S must be sufficiently large, approximately

   $$S \approx c\, R \log\left(\frac{D}{R}\right), \tag{19}$$

   where the constant $c > 0$ depends on the incoherence of $\mathbf{C}$ and $\boldsymbol{\Psi}$.

To provide some intuition why the l^1-norm may serve as a proxy for the l^0-norm, let us note that the l^1 is the "closest" convex set to the bounded l^0-ball, and we have already seen in the previous section that l^1 minimization leads to sparse solutions. We remark that the reconstruction of the signal via compressed sensing is not guaranteed, only

[1]By pseudo-norm, we mean that $\|\cdot\|_0$ is positive definite and satisfies the triangle inequality, but generally $|\alpha|\|\boldsymbol{\theta}\|_0 \neq \|\alpha\boldsymbol{x}\|_0$ for $\alpha \in \mathbb{R}$ and $\mathbf{x} \in \mathbb{R}^D$, violating absolute homogeneity.

possible *at high probability*, making it a statistical theory. In the setting of *regression*, where the measurements $\mathbf{y}$ are corrupted by noise, i.e. $\mathbf{y} = \mathbf{C}\mathbf{x} + \boldsymbol{\varepsilon}$, the equality in Eq. (18) can be relaxed, and we can instead consider the closely related problem

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1, \tag{20}$$

where $\lambda \geq 0$ weights the importance of sparsity with respect to the reconstruction quality expressed by the first term. Comparing this expression with Eq. (14) shows that $\boldsymbol{\theta}^\star$ then exactly becomes the solution to the Lasso regularized convex optimization problem we considered in the previous section.[2]

Example 1 (Reconstructing an audio signal). Let us consider an audio signal $x = x(t)$ made up of $R = 3$ different tones

$$x(t) = \cos(29 \times 2\pi t) + \cos(41 \times 2\pi t) + \cos(503 \times 2\pi t). \tag{21}$$

Thus, the highest frequency in the signal is $f = 503$ Hz and according to the Shannon–Nyquist theorem, the minimum (sufficient) spacing between sampling points to recover the signal is given by $\frac{1}{2f} = 1/1{,}006$ s. We consider the time interval $t \in [0, 1)$ s, leading to 1,006 required measurements. However, by exploiting the knowledge that the signal is sparse in a discrete cosine basis (as it is fully described by 3 cosine waves), we will see that much less *random* measurements suffice to reconstruct the signal x. We take the time resolution of the full signal to be $D = 4{,}096$ and randomly select $S = 128$ locations in time with uniform probability. This corresponds to a measurement matrix $\mathbf{C} \in \mathbb{R}^{128\times 4{,}096}$ whose rows are identically zero apart from a single one in a randomly selected column for each row. The sparse reconstruction found by the Lasso minimization closely resembles the true signal, see Fig. 7. Note, however, that the quality of the reconstruction may sensitively depend on the regularization parameter λ, making checks such as cross-validation or evaluation on a separate test dataset necessary.

The PYTHON code for this example looks as follows (using the Lasso minimizer from `scikit-learn`):

[2]Note that the dimensions of the involved quantities have changed, however: in Eq. (14), we were considering $\mathbf{y} \in \mathbb{R}^N$ and $\boldsymbol{\theta} \in \mathbb{R}^K$ with $K < N$, giving rise to an *overdetermined* system (so $\boldsymbol{\Phi} \in \mathbb{R}^{N\times K}$ is higher than wide). For the compressed sensing application, we have a vector of a few measurements $\mathbf{y} \in \mathbb{R}^S$ and a large coefficient vector $\boldsymbol{\theta} \in \mathbb{R}^D$ (which is however R-sparse), meaning that we have an *underdetermined* system with $R \leq S < D$ (so $\boldsymbol{\Phi} \in \mathbb{R}^{S\times D}$ is wider than high).

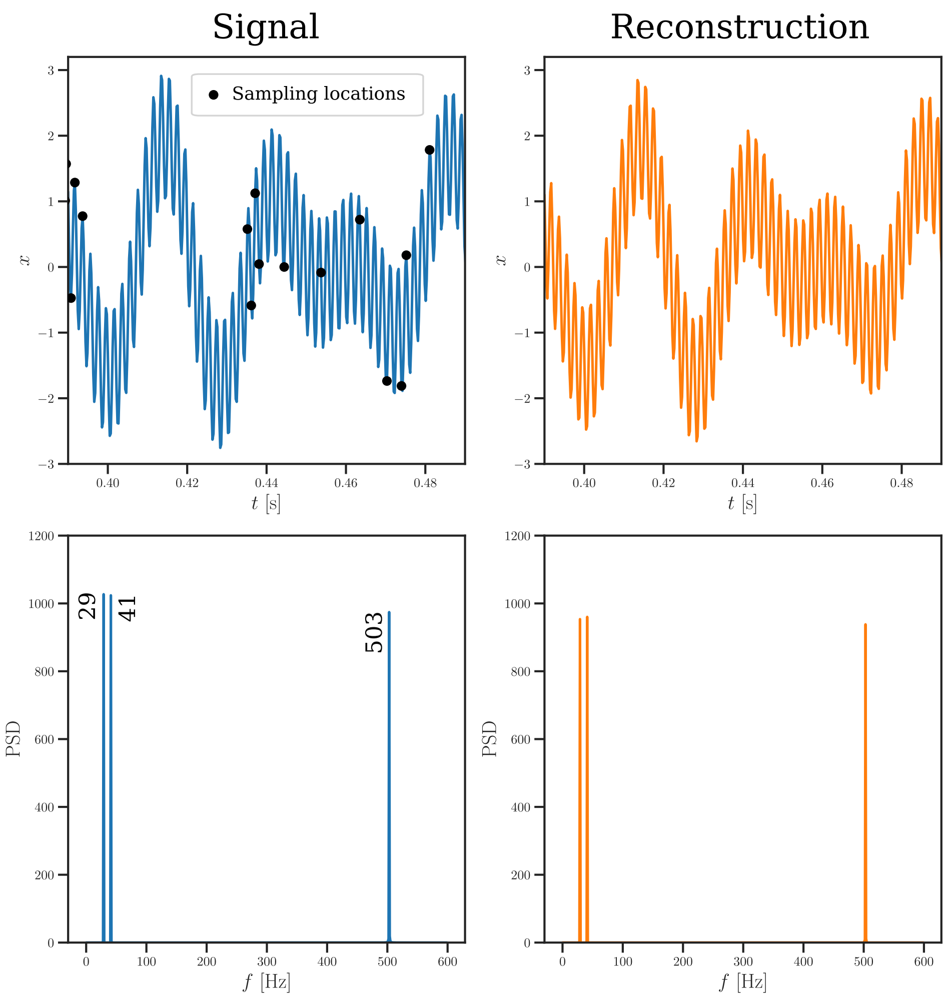

Figure 7: Audio signal composed of three tones (*left*) and compressed sensing reconstruction (*right*) based on sampling at the locations marked in black (128 locations in total). The top panels show $x(t)$ (zoomed into a small time interval), and the power spectral density (PSD) is plotted below. The three frequencies in the signal at $f = 29$, 41, and 503 Hz are recovered by the Lasso minimizer. The compressed sensing reconstruction works in this case (with far less measurements than needed to satisfy the sufficient condition for the recovery of all frequencies as stated by the Shannon–Nyquist sampling theorem) because 1) the signal is sparse in the discrete cosine basis, and 2) the sampling locations are sufficiently decoherent with respect to this basis as we draw them *randomly*.

```python
import numpy as np
from scipy.fftpack import dct, idct
from sklearn.linear_model import Lasso

# Define the high-resolution signal
D = 4096  # resolution
t_hr = np.linspace(0, 1, D)  # high resolution time grid
x_fct = lambda t: np.cos(29 * 2.0 * np.pi * t) \
                + np.cos(41 * 2.0 * np.pi * t) \
                + np.cos(503 * 2.0 * np.pi * t)
# Nyquist-Shannon: -> 2 * 503 samples / s necessary
x = x_fct(t_hr)  # high resolution signal
x_f = np.fft.fft(x)
psd = x_f * x_f.conj() / D  # power spectral density

# Get a random subsample of the signal
S = 128
sample_inds = np.random.choice(D, size=S, replace=False)
t_lr = t_hr[sample_inds]
y = x[sample_inds]

# Define DCT matrix
Psi = dct(np.eye(D))  # DCT matrix

# Take random measurements
Phi = Psi[sample_inds, :]  # measure rows of Psi

# Equivalently
# C = np.zeros((S, D))
# C[np.arange(S), sample_inds] = 1
# Phi = C @ Psi

# Define a Lasso optimizer and fit
lasso = Lasso(alpha=0.001)
lasso.fit(Phi, y)
theta_star = lasso.coef_  # Lasso best-fit parameters
x_recon = idct(theta_star)  # convert from DCT to time domain
x_recon_f = np.fft.fft(x_recon)
psd_recon = x_recon_f * x_recon_f.conj() / D
```

4 Bayesian linear regression

In Section 3.1, we placed a prior on the parameter vector $\boldsymbol{\theta}$ and used Bayes' theorem to determine the maximum a-posteriori estimate for $\boldsymbol{\theta}$. We found that incorporating prior knowledge for $\boldsymbol{\theta}$ gives rise to an additional regularization term as used in ridge regression and Lasso regression. However, we only derived *point estimates* for $\boldsymbol{\theta}$, namely the maximum likelihood estimate (without prior knowledge) and the maximum a-posteriori estimate (with prior knowledge).

Bayesian linear regression goes one step further and aims at determining the entire **posterior distribution** for $\boldsymbol{\theta}$. This means the probabilistic model is now given by

$$\text{prior}: \qquad p(\boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{m}_0, \boldsymbol{S}_0\right), \tag{22a}$$

$$\text{likelihood}: \qquad p(y \mid \mathbf{x}, \boldsymbol{\theta}) = \mathcal{N}\left(y \mid \boldsymbol{\phi}^\top(\mathbf{x})\boldsymbol{\theta}, \sigma^2\right), \tag{22b}$$

where $p(\boldsymbol{\theta}) \sim \mathcal{N}\left(\boldsymbol{m}_0, \boldsymbol{S}_0\right)$ is a Gaussian prior on the parameter vector $\boldsymbol{\theta}$, which is now treated as a random variable. Now, the probability to observe $\mathcal{Y}$ given the inputs $\mathcal{X}$ (known as the **evidence**) can be obtained by marginalizing over the possible parameter vectors $\boldsymbol{\theta}$, i.e.

$$p(\mathcal{Y} \mid \mathcal{X}) = \int p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} = \mathbb{E}_{\boldsymbol{\theta}}[p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\theta})]. \tag{23}$$

The posterior distribution for the parameters can then be shown to be

$$p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y}) = \mathcal{N}\left(\boldsymbol{\theta} \mid \boldsymbol{m}_N, \boldsymbol{S}_N\right), \tag{24}$$

where

$$\boldsymbol{S}_N = \left(\boldsymbol{S}_0^{-1} + \sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\right)^{-1}, \tag{25a}$$

$$\boldsymbol{m}_N = \boldsymbol{S}_N\left(\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \sigma^{-2}\boldsymbol{\Phi}^\top\mathbf{y}\right). \tag{25b}$$

Here, the subscript N stands for the number of training samples. Based on this expression, one can then predict the predictive distribution for a *test* observation y_* given a test input $\mathbf{x}_*$ by *marginalizing* over the parameter posterior, i.e.

$$p\left(y_* \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*\right) = \int p\left(y_* \mid \boldsymbol{x}_*, \boldsymbol{\theta}\right)\, p(\boldsymbol{\theta} \mid \mathcal{X}, \mathcal{Y})\, \mathrm{d}\boldsymbol{\theta}. \tag{26}$$

Bayesian (linear) regression is a rich topic with many applications and a solid probabilistic foundation. For more details (and the derivation of the parameter posterior distribution), take a look at e.g. Ref. [3, Sec. 9.3].

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning.* Information Science and Statistics. Springer New York, 2016.

[2] S. L. Brunton and J. N. Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control.* Cambridge University Press, 2022.

[3] M. P. Deisenroth, A. A. Faisal, and C. S. Ong. *Mathematics for Machine Learning.* Cambridge University Press, 2020.

[4] D. L. Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[5] Ž. Ivezić, A. J. Connolly, J. T. van der Plas, and A. Gray. *Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data, Updated Edition.* Princeton University Press, 2019.