Overview

Notation

Matrices are written in uppercase: A, vectors are written in lowercase: a. A_{ij} denotes the element of A in position (i, j), A_i denotes the ith column of A (it's a vector!) and a_i denotes the ith entry of a.

1 Sparsity

In the following sections we describe some applications of sparse models in signal and image processing, statistics and inverse problems. The goal is to provide a high-level picture of how to leverage sparsity assumptions in these domains.

1.1 Denoising via thresholding

In data processing, data are often described as a combination between a **signal** component, which contains the information that we are interested in, and a **noise** component, which accounts for perturbations that corrupt the signal. The problem of **denoising** data consists of teasing apart the signal and the noise, using prior information about their structure.

Let us consider an additive noise model

$$\text{data} = \text{signal} + \text{noise}. \tag{1}$$

If we expect the signal to be sparse, then an intuitive denoising method is to set to zero any entry that is below a certain value. This is known as **hard thresholding** (as opposed to soft thresholding, which we will learn about later in the course).

Definition 1.1 (Hard thresholding). *Let $x \in \mathbb{R}^n$. The hard-thresholding operator $\mathcal{H}_\eta : \mathbb{R}^n \to \mathbb{R}^n$ sets to zero any entries in x with magnitude smaller than a predefined real-valued threshold $\eta > 0$,*

$$\mathcal{H}_\eta (x)_i := \begin{cases} x_i & \text{if } |x_i| > \eta, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Figure 1 shows an example in which a sparse signal is denoised by applying hard thresholding.

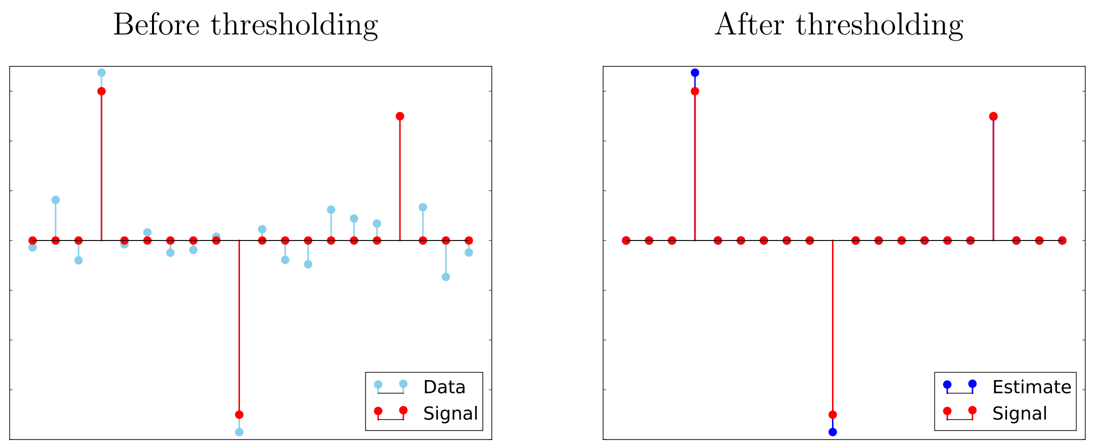

Figure 1: Denoising via hard thresholding.

1.2 Sparsity in a basis

Even though signals of interest in applications are often not sparse at all, they are often well modeled as a linear combination of a small number of predefined **atoms**. This can be conveniently expressed mathematically by grouping the atoms as columns of a certain matrix D, which is often known as a **dictionary**. Recall that for any matrix $D \in \mathbb{R}^{m \times n}$ and any vector $c \in \mathbb{R}^n$, the matrix-vector product

$$Dc = \sum_{i=1}^{n} D_i \, c_i, \tag{3}$$

is a linear combination of the columns of D weighted by the entries in c. If the signal $x = Dc$ can be represented by a combination of a few atoms in the dictionary, then its corresponding coefficient vector c will be sparse.

Sinusoids are an important example of atoms that allow to obtain sparse representations. Figure 2 shows an example of a signal that is sparse in such a dictionary. Another very popular sparsifying transform in signal and image processing is the **wavelet** transform. Figure 3 shows a natural image and its corresponding wavelet coefficients, most of which are extremely small.[1]

If the atoms form a basis of the ambient space, then the coefficients $c \in \mathbb{R}^n$ in the representation of x,

$$x = Bc, \tag{4}$$

[1]The numerical simulations in Figures 3 and 5 were implemented using Gabriel Peyré's Wavelet Toolbox which is available online at `http://www.ceremade.dauphine.fr/~peyre/matlab/wavelets/content.html`

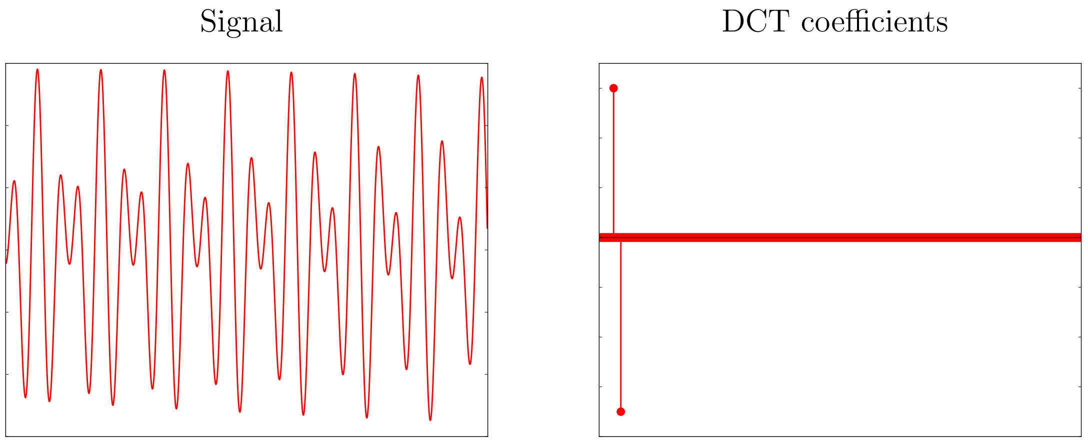

Figure 2: The signal on the left is sparse after applying a discrete cosine transform (DCT), which expresses the signal in terms of cosine atoms. The corresponding coefficients are shown on the right.

can be retrieved by applying the inverse of the dictionary B. This is the case for both examples in Figures 2 and 3.

If we consider data corrupted with additive noise,

$$y = Bc + z, \tag{5}$$

it is straightforward to exploit the sparsity assumption through thresholding. We only need to apply the inverse of the dictionary and then threshold. The estimated coefficients are of the form,

$$\hat{c} = \mathcal{H}_\eta \left(B^{-1} y \right), \tag{6}$$

which yields the signal estimate

$$\hat{y} = B\hat{c} = B\, \mathcal{H}_\eta \left(B^{-1} y \right). \tag{7}$$

Figures 4 and 5 show the results of applying this denoising method to the signals in Figures 2 and 3. Both signals are corrupted by additive Gaussian noise. In both cases, exploiting the sparse decomposition allows us to denoise the data very effectively.

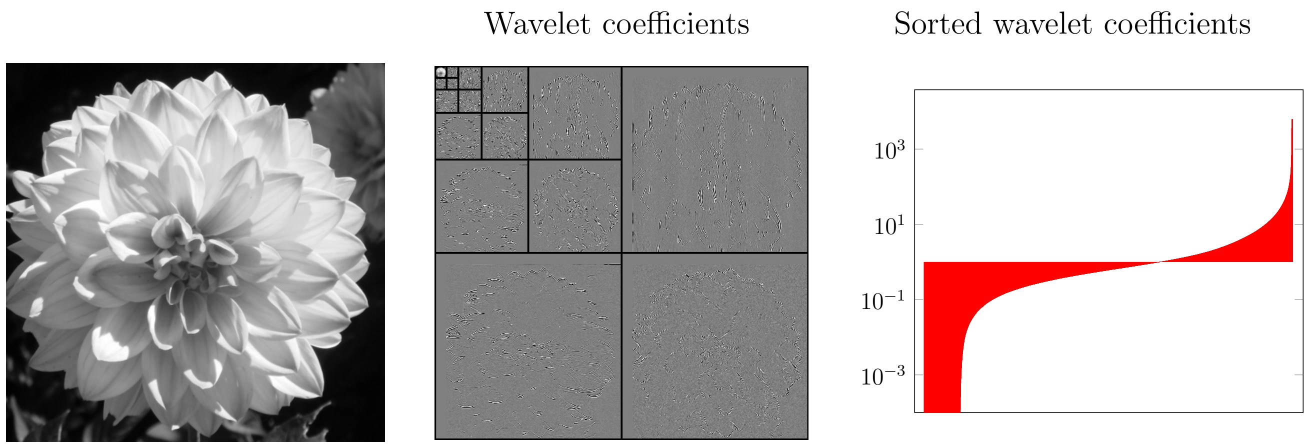

Figure 3: Coefficients in a biorthogonal wavelet basis (center) of a natural image (left). Plotting the ordered coefficients reveals that the image is highly compressible in the wavelet domain (right).

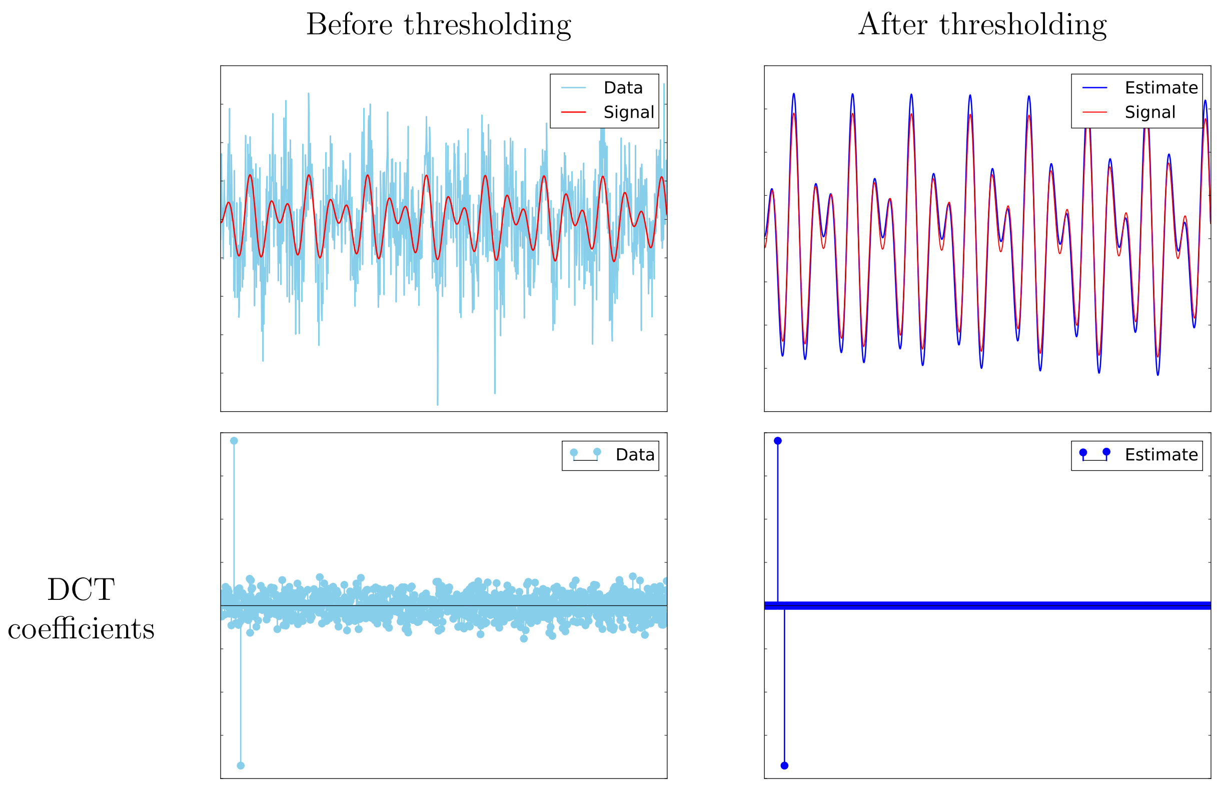

Figure 4: Denoising via hard thresholding in the DCT basis.

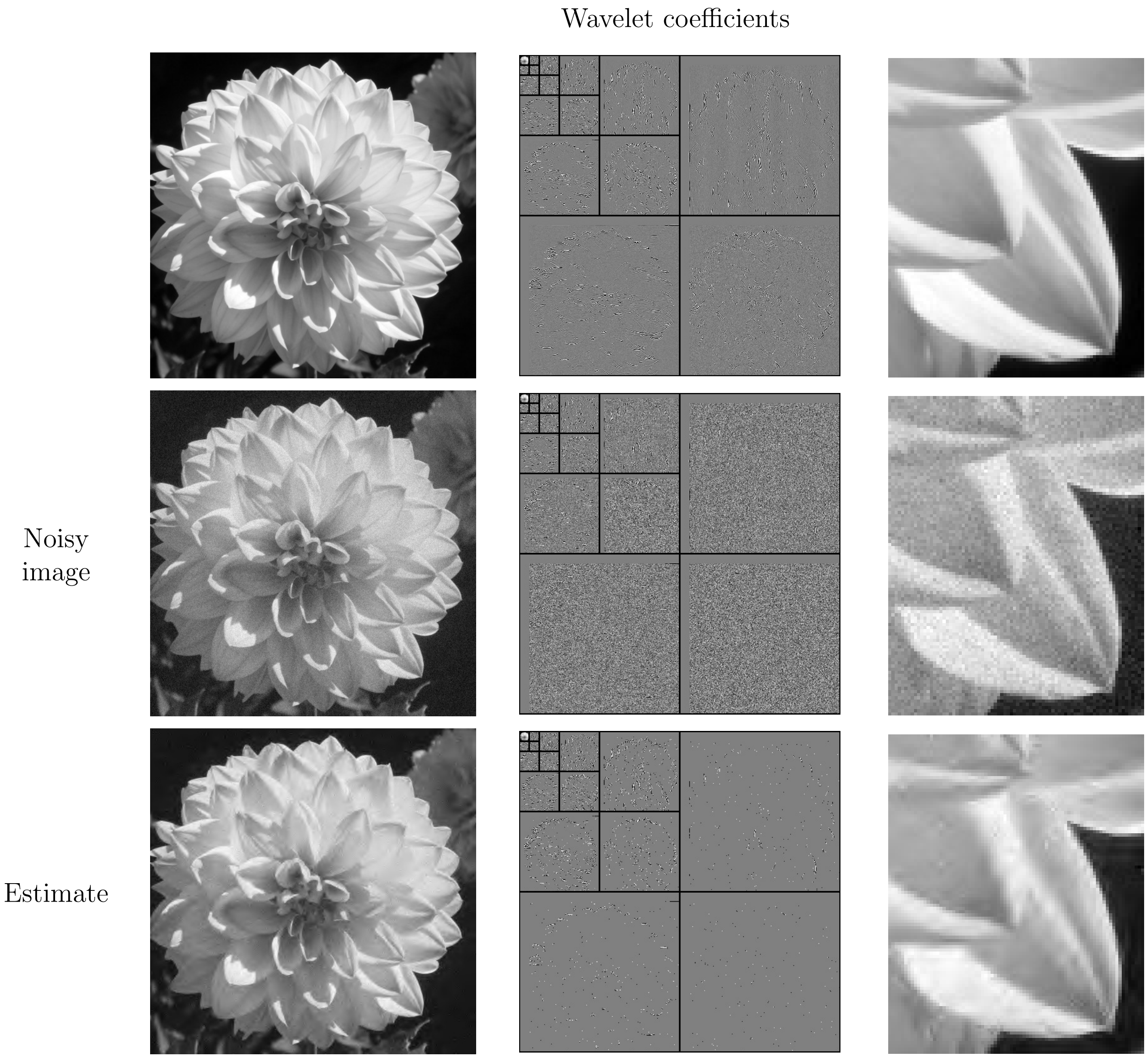

Figure 5: Denoising via hard thresholding in a biorthogonal wavelet basis.

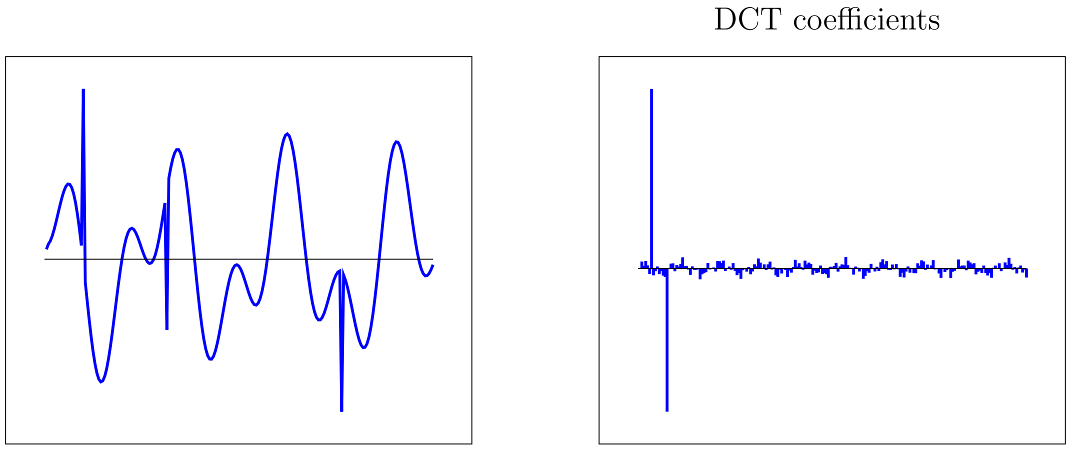

Figure 6: A signal consisting of spikes and sinusoids (left) and its DCT representation (right).

1.3 Sparsity in redundant representations

Figure 6 shows a signal that is not sparse either in a basis of spiky atoms or sinusoidal atoms. However, it is sparse in a dictionary that contains *both* sinusoids and spikes,

$$x = Dc = \begin{bmatrix} I & F \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = a + Fb, \tag{8}$$

where $I \in \mathbb{R}^{n \times n}$ is the identity matrix and $F \in \mathbb{R}^{n \times n}$ represents a basis of sinusoids (F stands for Fourier).

Clearly, the dictionary $D \in \mathbb{R}^{n \times 2n}$ is not invertible. There are many (in fact infinite!) ways of representing the signal in terms of the atoms of the dictionary. This makes it challenging to leverage the assumption that the signal has a sparse representation in the dictionary. Let us consider for a moment the problem of finding a sparse representation even if there is no noise. Ideally, we would like to solve the following optimization problem.

$$\min_{\tilde{c} \in \mathbb{R}^m} ||\tilde{c}||_0 \quad \text{such that } x = Dc, \tag{9}$$

where the ℓ_0 "norm" of a vector $x \in \mathbb{R}^n$ (which is not really a norm) is equal to the number of nonzero entries of x

$$||x||_0 := \operatorname{card}\left(\{i \mid x_i \neq 0\}\right). \tag{10}$$

Problem 9 is intractable even for signals of very moderate size. However, it turns out that there is a tractable optimization problem that often produces sparse representations; it is obtained by replacing the ℓ_0 "norm" with the ℓ_1 norm

$$\min_{\tilde{c} \in \mathbb{R}^m} ||\tilde{c}||_1 \quad \text{such that } x = Dc. \tag{11}$$

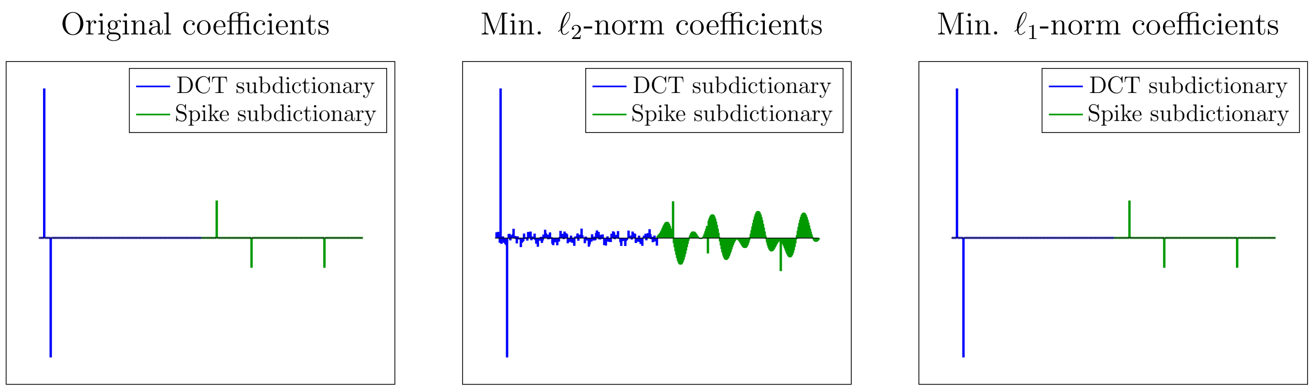

Figure 7: The original coefficients of the signal from Figure 6 in a dictionary containing spikes and sinusoids are shown on the left. Minimizing the ℓ_2 norm of the coefficients does not yield a sparse representation (center), but minimizing the ℓ_1 norm does.

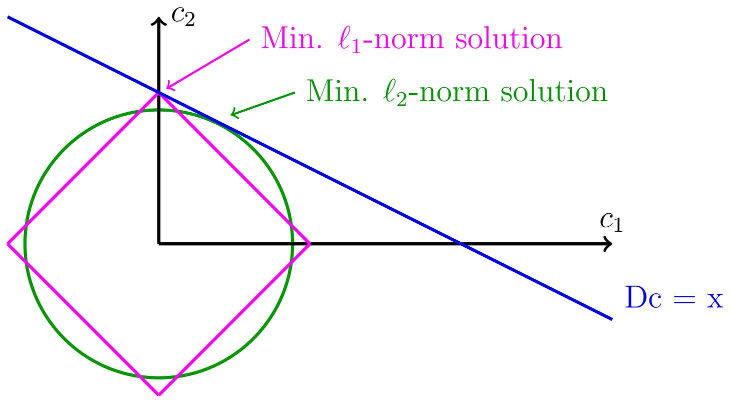

Figure 8: The minimum ℓ_1-norm solution is sparser than the minimum ℓ_2-norm solution because of the geometry of the ℓ_1-norm and ℓ_2-norm balls.

Figure 7 shows how ℓ_1-norm minimization allows to compute a sparse coefficient vector in the case of the signal in Figure 6. In contrast, minimizing the ℓ_2 norm of the coefficient vector produces a very dense solution.[2] This is due to the geometry of the ℓ_1 norm and the ℓ_2 norms. Figure 8 provides a picture for the case where the dictionary has two atoms. The ℓ_1-norm ball is more concentrated around the axes than the ℓ_2-norm ball. It is therefore more likely for the line representing the constraint $x = Dc$ to be tangent to the ball on an axis, where the solution has cardinality one instead of two. As a result, the minimum ℓ_1-norm solution is sparser than the minimum ℓ_2-norm solution.

In order to learn sparse representations when noise is present in the data, we can eliminate

[2]The numerical simulations in Figure 7 and several other figures in these notes were implemented using Stephen Boyd's and Michael Grant's CVX software which is available online at http://cvxr.com/cvx/

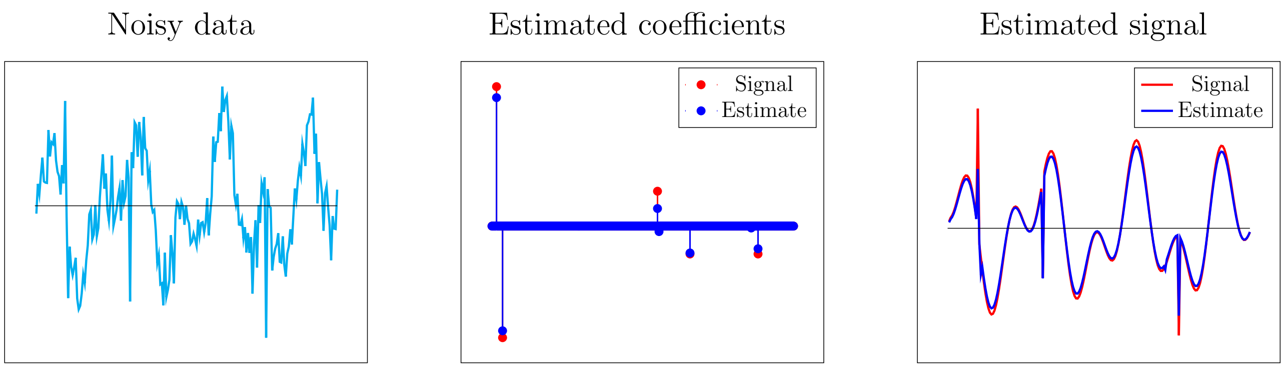

Figure 9: Denoising via ℓ_1-norm-regularized least squares.

the equality constraint in Problem 11 and add a data-fidelity term to the cost function. This is often known as **basis-pursuit denoising**. In more detail, we estimate the coefficients by solving

$$\hat{c} = \arg\min_{\tilde{c} \in \mathbb{R}^m} \left|\left|x - D\tilde{c}\right|\right|_2^2 + \lambda \left|\left|\tilde{c}\right|\right|_1 \tag{12}$$

where $\lambda > 0$ is a regularization parameter that determines the tradeoff between the term that promotes sparsity and the term that promotes data fidelity. Figure 9 shows the result of applying this denoising method to a signal that has a sparse representation in a dictionary of spikes and sinusoids.

1.4 Learning the dictionary

Dictionary-learning techniques allow to learn dictionaries directly from the data. This is very useful in situations where a dataset with a large number of signals cannot be compactly represented in any predefined dictionary. If the signals are grouped as columns of a matrix $X \in \mathbb{R}^{n\times k}$ (k is the number of signals in the dataset), the aim is to learn a dictionary $D \in \mathbb{R}^{n\times m}$ such that $X = DC$, where the matrix of coefficients $C \in \mathbb{R}^{m\times k}$ is very sparse. Following the heuristic that penalizing the ℓ_1 norm promotes sparse solutions, an option is to solve the following optimization program,

$$\min_{\widetilde{C} \in \mathbb{R}^{m\times k}} \left|\left|X - \widetilde{D}\widetilde{C}\right|\right|_{\mathrm{F}}^2 + \lambda \left|\left|\widetilde{C}\right|\right|_1 \quad \text{such that} \quad \left|\left|\widetilde{D}_i\right|\right|_2 = 1, \quad 1 \leq i \leq m, \tag{13}$$

where the atoms of the dictionary are constrained to have unit norm. $\left|\left|\cdot\right|\right|_{\mathrm{F}}$ denotes the Frobenius norm, which is equal to the ℓ_2 norm of the entries of the matrix interpreted as a vector.

As we will discuss later in the course, this optimization problem is much more challenging to solve than (12), where the dictionary is fixed. Without getting into too much detail,

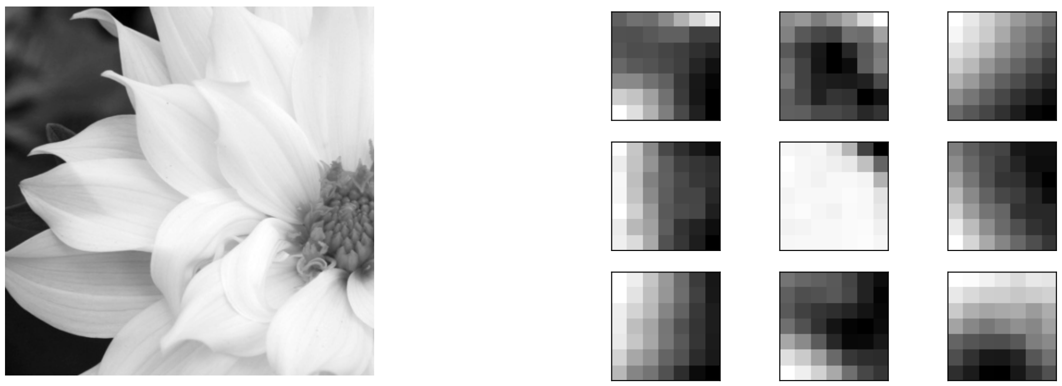

Figure 10: Atoms (right) learnt from patches extracted from a natural image (left).

the optimization problem is nonconvex, and hence may have many local minima. Figure 10 shows patches learnt from a natural image.[3] The corresponding dictionary can be used to denoise other images quite effectively, as shown in Figure 11.

1.5 Sparse regression

In statistics, the problem of **regression** is that of learning a function that allows to estimate a certain quantity of interest, the **response** or **dependent variable**, from other observed variables, known as **covariates** or **independent variables** . For example, we might be interested in estimating the price of a house from the number of square feet, the number of rooms and the location. In **linear regression**, we assume that the function is linear. In that case, the model is of the form

$$y_i \approx \sum_{j=1}^{p} \theta_j X_{ij}, \quad 1 \leq i \leq n, \tag{14}$$

where n is the number of data, $y \in \mathbb{R}^n$ is the response, $X_1, X_2, \ldots, X_p \in \mathbb{R}^n$ contain the covariates and $\theta_1, \theta_2, \ldots, \theta_p \in \mathbb{R}$ are the parameters of the linear model. In matrix form, $y \approx X\theta$. In order to calibrate the model, a common procedure is to fit the parameters so that the model approximates the response as closely as possible in ℓ_2 norm. This is achieved by solving the least-squares problem

$$\hat{\theta}_{\text{ls}} := \arg\min_{\tilde{\theta} \in \mathbb{R}^n} \left|\left| y - X\tilde{\theta} \right|\right|_2 . \tag{15}$$

[3]The numerical simulations in Figures 11 and other figures in these notes were implemented using scikit-learn, which is available online at http://scikit-learn.org.

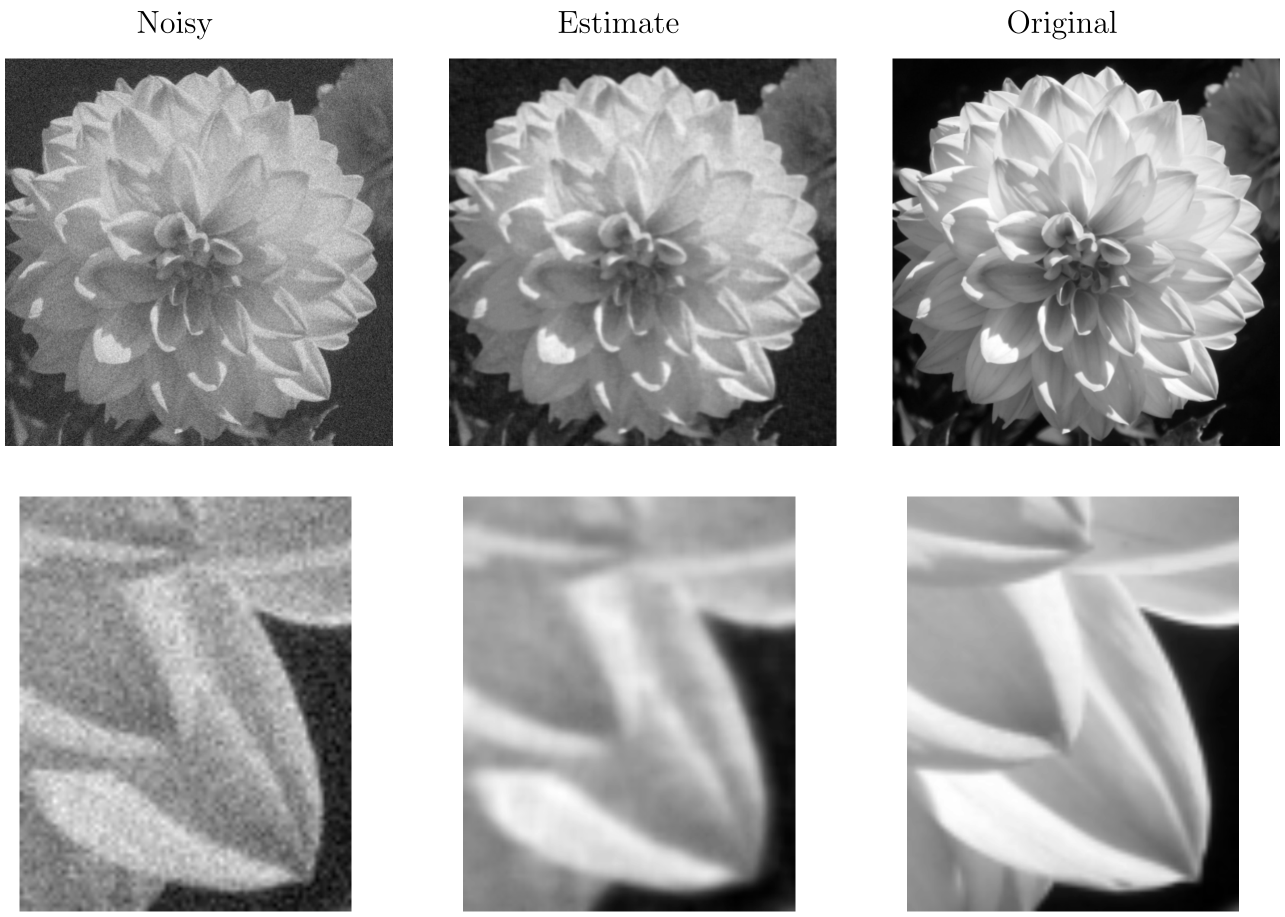

Figure 11: Denoising results using the dictionary learnt from the image shown in Figure 10.

In some applications, however, many of the covariates may actually be unrelated to the response. Imagine that we are trying to investigate the connection between the expression of several genes and a certain disease. Each covariate corresponds to a gene and the response quantifies a symptom of the disease. If most genes are unrelated to the disease, then we need to perform **model selection**, i.e. determine what genes we should incorporate in the regression function. Otherwise the model might try to explain the response using the irrelevant covariates. Although this could actually allow to obtain a better fit on the data that we are using to learn the model, it will hurt the ability of the model to **generalize** to new data. In machine learning and statistics this is known as **overfitting**. For this reason, if we have reason to suspect that only a few covariates are actually relevant, it makes sense to try to fit a sparse model to the data. As in the case of sparse representations in redundant dictionaries, penalizing the ℓ_1-norm often achieves this goal. In statistics, ℓ_1-norm regularized least squares is known as the **lasso**,

$$\hat{\theta}_{\text{lasso}} := \arg\min_{\tilde{\theta} \in \mathbb{R}^n} \left|\left| y - X\tilde{\theta} \right|\right|_2^2 + \lambda \left|\left| \tilde{\theta} \right|\right|_1, \tag{16}$$

$\lambda > 0$ is a regularization parameter that controls the level of regularization.

Let us illustrate all of this with a simple numerical simulation.

1. We generate a **training set** by computing the response as a linear combination of 3 covariates and add some noise to the data

$$y_{\text{train}} = \begin{bmatrix} X_1^{\text{train}} & X_2^{\text{train}} & X_3^{\text{train}} \end{bmatrix} \theta + z_{\text{train}}, \tag{17}$$

   where $n = 100$, i.e. the response, covariates and noise vectors all have dimension 100.

2. We fit the model using a covariate matrix that includes 47 extra covariates X_4, X_5, ..., X_{50} that are completely independent from the response,

$$X_{\text{train}} = \begin{bmatrix} X_1 & X_2 & X_3 & X_4 & \cdots & X_{50} \end{bmatrix}, \tag{18}$$

3. We test the model using a and a **test set** that consists of a response

$$y_{\text{test}} = \begin{bmatrix} X_1^{\text{test}} & X_2^{\text{test}} & X_3^{\text{test}} \end{bmatrix} \theta + z_{\text{test}}, \tag{19}$$

   and a matrix of covariates

$$X_{\text{test}} = \begin{bmatrix} X_1 & X_2 & X_3 & X_4 & \cdots & X_{50} \end{bmatrix}, \tag{20}$$

   where again $n = 100$, i.e. the response, covariates and noise vectors all have dimension 100. The coefficient vector θ is the same as in the training set.

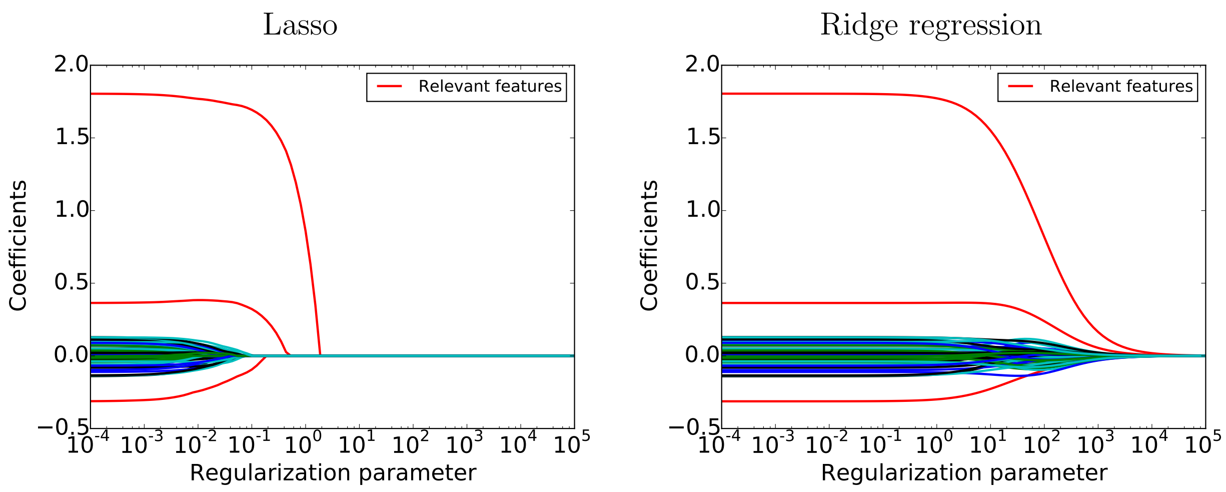

Figure 12: Value of the coefficients in the lasso (left) and ridge-regression models for different values of the regularization parameter λ.

4. We fit the model using least-squares regression, the lasso and also **ridge regression**, which penalizes the ℓ_2 norm of the parameter vector and is known as Tikhonov regularization in the applied-mathematics literature,

$$\hat{\theta}_{\text{ridge}} := \arg\min_{\tilde{\theta}\in\mathbb{R}^n} \left|\left|y - X\tilde{\theta}\right|\right|_2^2 + \lambda \left|\left|\tilde{\theta}\right|\right|_2^2. \tag{21}$$

Figure 12 shows the value of the fitted parameters for different values of the regularization parameter. When λ is very large, all of the coefficients are set to zero by the lasso and by ridge regression because the fitting error has no weight in the cost function. When λ is very small, both methods are equivalent to least-squares regression. In between these two regimes, the lasso model is sparse and for a certain range of values of λ it only includes the relevant features. In contrast, ridge regression allows to diminish the influence of the irrelevant covariates, but is not sparse. As discussed previously, this is due to the geometry of the ℓ_1-norm and ℓ_2-norm balls (see Figure 8).

Figure 13 plots the relative error achieved on the training and test sets for least-squares regression, the lasso and ridge regression. In the training set, least-squares regression achieves the smallest error by overfitting the data. This becomes obvious when we compare the error on the test set, where the fit of the sparse model obtained by the lasso is significantly better. Ridge regression also prevents overfitting to some extent, but not as effectively as the lasso.

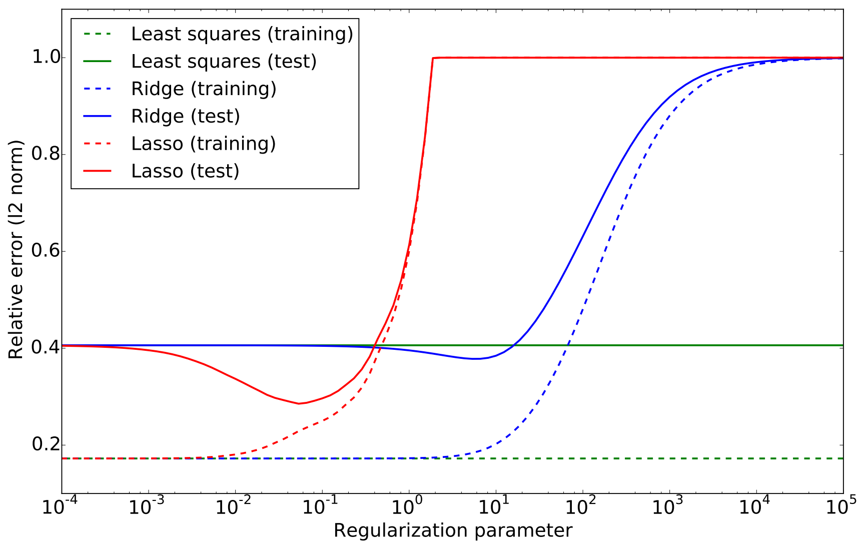

Figure 13: Relative ℓ_2 norm error in estimating the response achieved using least-squares regression, the lasso and ridge regression.

1.6 Inverse problems

This section describes two inverse problems that are ill posed unless we make some assumptions on the signal structure. In particular, leveraging sparsity-based models through the use of ℓ_1-norm regularization often allows to obtain estimates, even if the problem is underdetermined.

1.6.1 Super-resolution

Extracting fine-scale information from low-resolution data is a major challenge in many areas of the applied sciences. In microscopy, astronomy and any other application employing an optical device, spatial resolution is fundamentally limited by diffraction. An example with real data is shown in Figure 14. Figure 15 illustrates a model for the data-acquisition process in such cases: the object of interest is convolved with a point-spread function that blurs the fine-scale details, acting essentially as a low-pass filter. The problem of super-resolution is that of reconstructing the original signal from the blurred measurements.

Computing the convolution between two signals can be carried out by multiplying their spectra in the frequency domain. To simplify matters, let us assume that the signal of interest is a vector $x \in \mathbb{R}^n$ and that the point-spread function of the sensing mechanism is a perfect low-pass filter. In that case, the spectrum of the measurements, which we denote

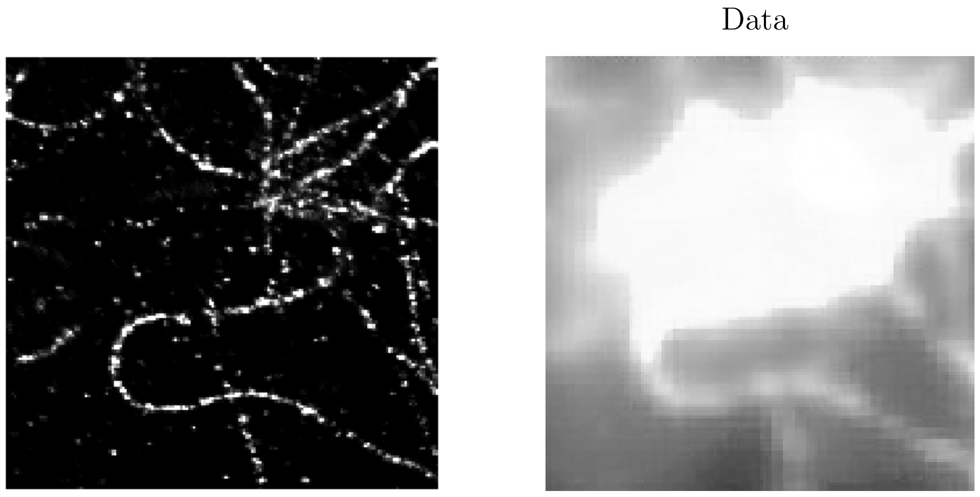

Figure 14: An example of deconvolution in fluorescence microscopy applied to data from the Moerner lab at Stanford by V. Morgenshtern. On the left, it is not possible to distinguish the individual fluorescent probes due to heavy aliasing. On the right, deconvolving the point sources in the individual frames and pooling the frames together reveals the fine-scale details of the object of interest.

by y, is equal to

$$y = \mathcal{F}_{\mathrm{L}}\, x, \tag{22}$$

where $x \in \mathbb{R}^m$ is the signal of interest and $\mathcal{F}_{\mathrm{L}}$ is a $n \times m$ submatrix of the discrete Fourier transform (DFT) matrix. Since some frequencies are suppressed, the number of measurements is greater than the dimension of the signal $m > n$, which implies that the system is underdetermined. If there is any solution to the system, there are infinite solutions, so we need to make further assumptions for the inverse problem not to be completely ill posed.

In many applications, a reasonable assumption is that the signal is well modeled as a superposition of point sources; examples include celestial bodies in astronomy, fluorescent probes in microscopy, or line spectra in signal processing. In such cases, we can enforce a sparsity prior in the same way by solving an ℓ_1-norm minimization problem, as in Section 1.3,

$$\min_{\tilde{x} \in \mathbb{R}^m} ||\tilde{x}||_1 \quad \text{such that } y = \mathcal{F}_{\mathrm{L}}\, \tilde{x}. \tag{23}$$

Figure 16 shows a simple example in which ℓ_1-norm minimization allows to recover a sparse signal exactly. In contrast, minimizing the ℓ_2 norm does not produces a sparse estimate, as illustrated in Figure 16. Later in the course, we will characterize under what conditions ℓ_1-norm minimization is guaranteed to achieve exact recovery.

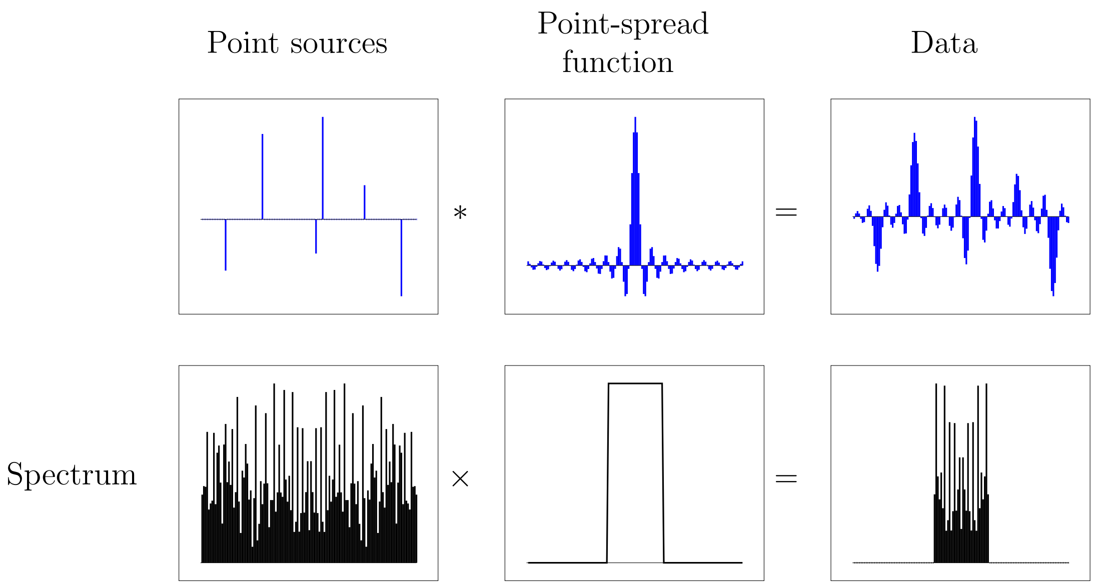

Figure 15: Convolution with a low-pass point-spread function (above) is equivalent to pointwise multiplication with a low-pass filter in the frequency domain (below).

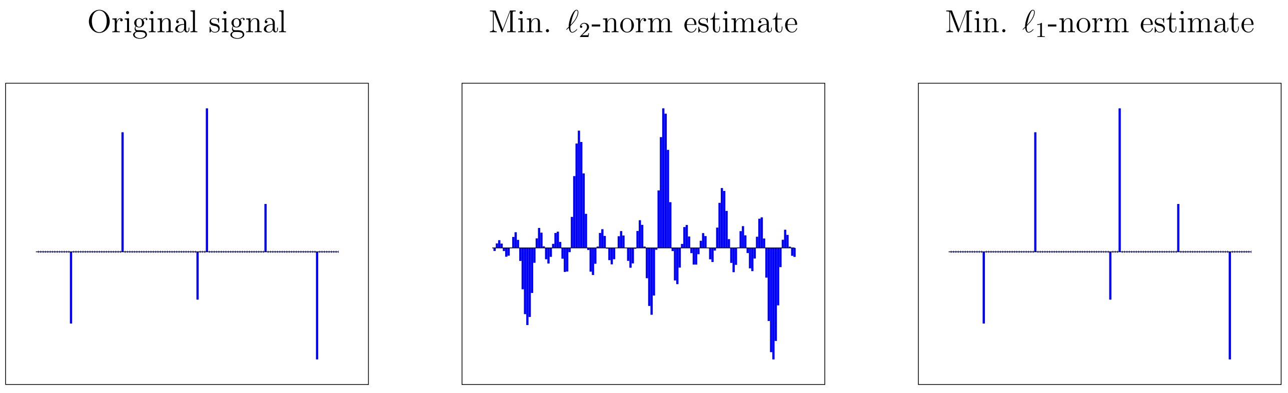

Figure 16: Minimizing the ℓ_1 norm of the estimate (right) allows to estimate the original signal (left) exactly, whereas minimizing the ℓ_2 norm does not produce a sparse estimate.

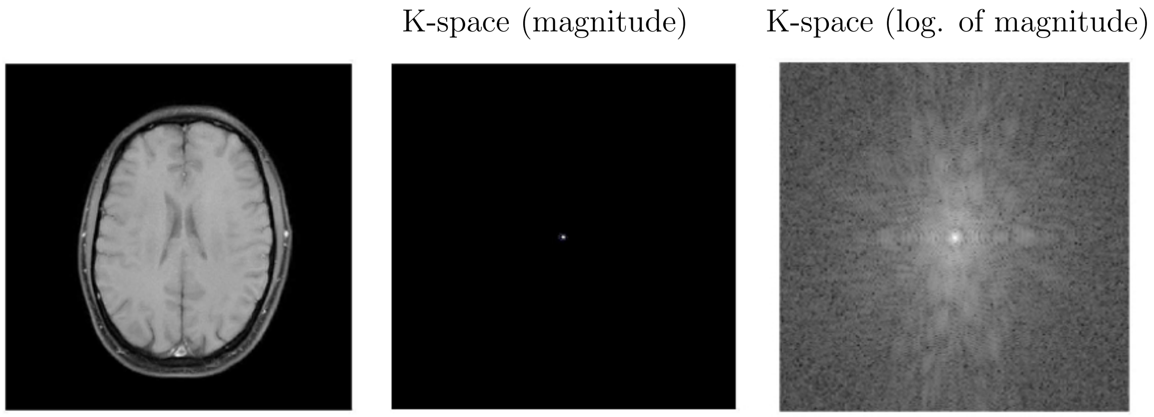

Figure 17: Image of a brain obtained by MRI, along with the magnitude of its 2D-Fourier or k-space representation and the logarithm of this magnitude.

1.6.2 Compressed sensing

Magnetic resonance imaging (MRI) is a popular medical imaging technique used in radiology. A simplified model of the data obtained via MRI is that they correspond to samples of the 2D or 3D Fourier transform of an image, known as **k space** in MRI jargon. An estimate of the image can be obtained by computing the inverse Fourier transform of the data, as shown in Figure 17.

An important challenge in MRI is to reduce measurement time. **Compressed sensing** achieves this by randomly undersampling the k-space representation of the image. Let us first consider a 1D version of the problem, where the signal is an m-dimensional vector x. The data y may be modeled as random samples of the DFT of the signal. More precisely,

$$y = \mathcal{F}_\Omega \, x, \tag{24}$$

where $x \in \mathbb{R}^m$ and the linear operator $\mathcal{F}_\Omega \in \mathbb{R}^{n \times m}$ corresponds to n random rows of the discrete Fourier transform (DFT) matrix. As in the case of super-resolution, $m > n$ so the system is underdetermined. Assuming that the signal is sparse, we can again solve ℓ_1-norm minimization problem to promote a sparse solution,

$$\min_{\tilde{x} \in \mathbb{R}^m} ||\tilde{x}||_1 \quad \text{such that } y = \mathcal{F}_\Omega \, \tilde{x}, \tag{25}$$

Figure 18 shows the results of applying compressed sensing to a sparse 1D signal.

In general, images such as the one in Figure 17 are not sparse. However they are often sparse in certain representations such as wavelet dictionaries, as discussed in Section 1.2. Minimizing the ℓ_1 norm of the coefficients in the sparsifying dictionary, allows to recover such images very accurately. An example is shown in Figure 19.

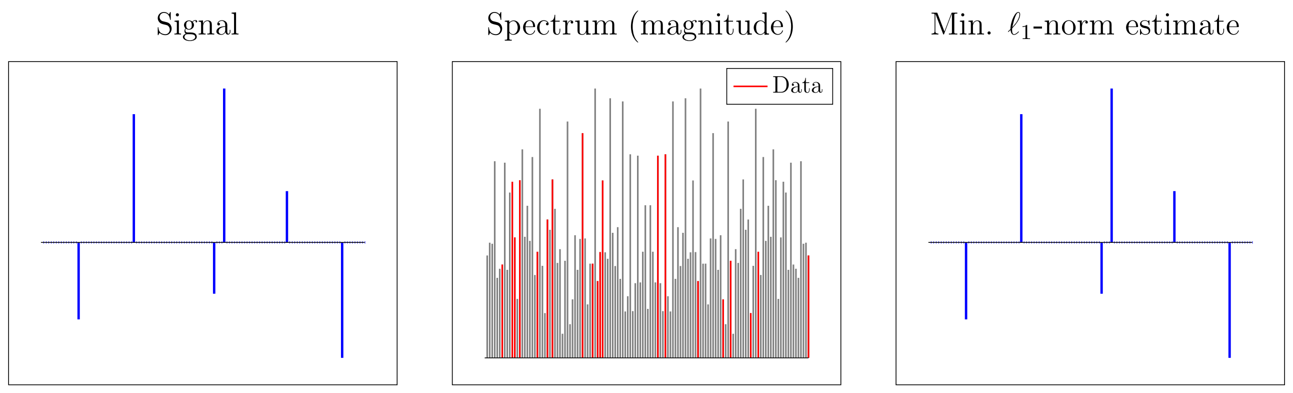

Figure 18: Minimizing the ℓ_1 norm of the estimate (right) allows to estimate the original signal (left) exactly from a small number of random samples of its spectrum (center).

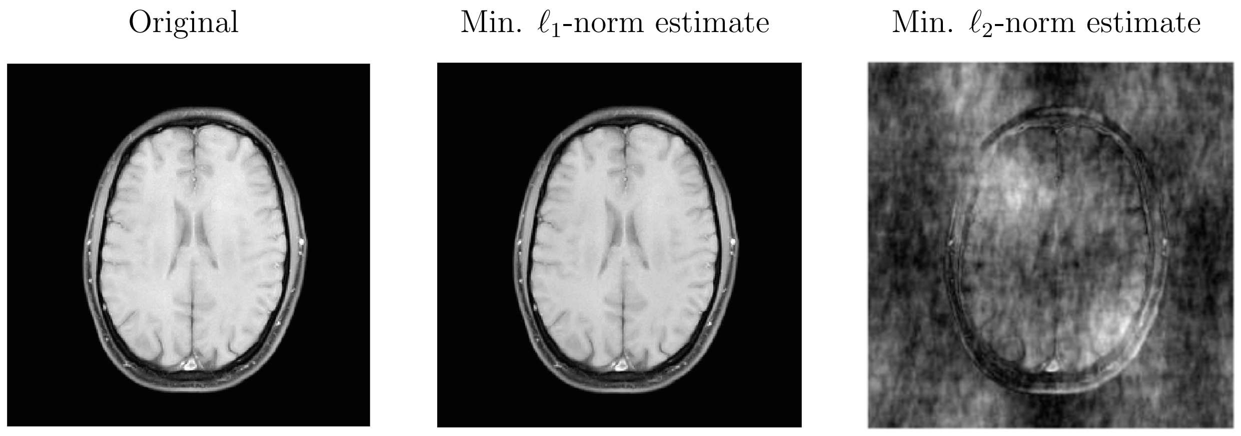

Figure 19: Compressed-sensing reconstruction (center) of a brain image (left) from random k-space samples. Minimizing the ℓ_2-norm recovers a heavily-aliased image (right).

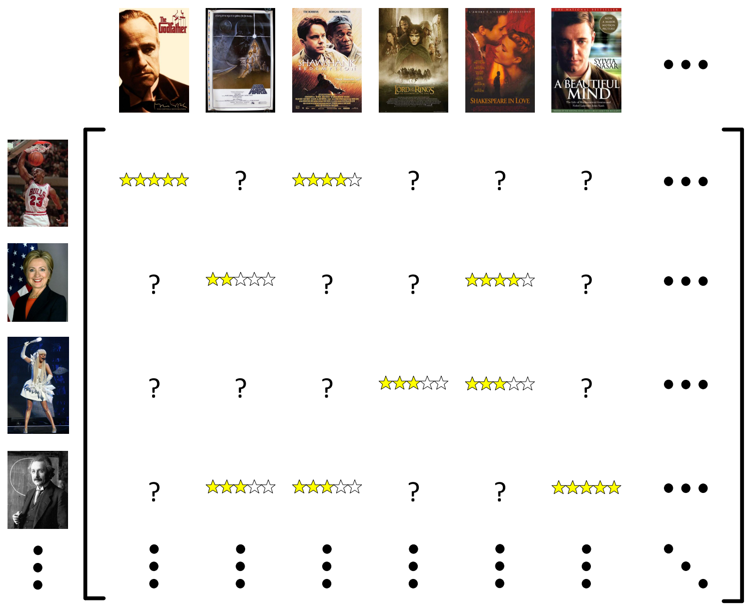

Figure 20: A depiction of the Netflix challenge in matrix form. Each row corresponds to a user that ranks a subset of the movies, which correspond to the columns. The figure is due to Mahdi Soltanolkotabi.

2 Low-rank models

In this section we describe three applications of low-rank models: matrix completion for movie-rating prediction, low rank + sparse models for background subtraction and nonnegative matrix factorization for topic modeling.

2.1 Matrix completion

The Netflix Prize was a contest organized by Netflix from 2007 to 2009 in which teams of data scientists tried to develop algorithms to improve the prediction of movie ratings. The problem of predicting ratings can be recast as that of completing a matrix from some of its entries, as illustrated in Figure 20. It turns out that matrices of ratings are often well modeled as being approximately low rank. We demonstrate this through a simple example.

Bob, Molly, Mary and Larry rate the following six movies from 1 to 5,

$$A := \begin{array}{cccccl} & \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} & \\ \left(\begin{array}{c} \\ \\ \\ \\ \\ \\ \end{array}\right. & \begin{array}{c} 1 \\ 2 \\ 4 \\ 5 \\ 4 \\ 1 \end{array} & \begin{array}{c} 1 \\ 1 \\ 5 \\ 4 \\ 5 \\ 2 \end{array} & \begin{array}{c} 5 \\ 4 \\ 2 \\ 2 \\ 1 \\ 5 \end{array} & \begin{array}{c} 4 \\ 5 \\ 1 \\ 1 \\ 2 \\ 5 \end{array} \left.\begin{array}{c} \\ \\ \\ \\ \\ \\ \end{array}\right) & \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} \end{array} \tag{26}$$

We subtract the average rating,

$$\mu := \frac{1}{n} \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}, \tag{27}$$

$$\tag{28}$$

from each entry in the matrix and then compute its singular value decomposition

$$A - \bar{A} = USV^T = U \begin{bmatrix} 7.79 & 0 & 0 & 0 \\ 0 & 1.62 & 0 & 0 \\ 0 & 0 & 1.55 & 0 \\ 0 & 0 & 0 & 0.62 \end{bmatrix} V^T, \tag{29}$$

where

$$\bar{A} := \begin{bmatrix} \mu & \mu & \cdots & \mu \\ \mu & \mu & \cdots & \mu \\ \cdots & \cdots & \cdots & \cdots \\ \mu & \mu & \cdots & \mu \end{bmatrix}. \tag{30}$$

The fact that the first singular value is significantly larger than the rest suggests that the matrix may be well approximated by a rank-1 matrix. This is the case (for ease of comparison the values of A are shown in brackets):

$$\bar{A} + \sigma_1 U_1 V_1^T = \begin{array}{ccccl} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} & \\ \left(1.34\,(1) \right. & 1.19\,(1) & 4.66\,(5) & \left. 4.81\,(4) \right) & \text{The Dark Knight} \\ 1.55\,(2) & 1.42\,(1) & 4.45\,(4) & 4.58\,(5) & \text{Spiderman 3} \\ 4.45\,(4) & 4.58\,(5) & 1.55\,(2) & 1.42\,(1) & \text{Love Actually} \\ 4.43\,(5) & 4.56\,(4) & 1.57\,(2) & 1.44\,(1) & \text{Bridget Jones's Diary} \\ 4.43\,(4) & 4.56\,(5) & 1.57\,(1) & 1.44\,(2) & \text{Pretty Woman} \\ 1.34\,(1) & 1.19\,(2) & 4.66\,(5) & 4.81\,(5) & \text{Superman 2} \end{array} \tag{31}$$

This rank-1 model has an interesting interpretation. The first left singular vector is equal to

$$U_1 = \begin{array}{cccccc} \text{D. Knight} & \text{Spiderman 3} & \text{Love Act.} & \text{B.J.'s Diary} & \text{P. Woman} & \text{Superman 2} \\ (-0.45 & -0.39 & 0.39 & 0.39 & 0.39 & -0.45) \end{array}.$$

Entries with similar values represent movies that are rated similarly by viewers. The first right singular vector is equal to

$$V_1 = \begin{array}{cccc} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ (0.48 & 0.52 & -0.48 & -0.52) \end{array}. \tag{32}$$

Here, entries with similar values correspond to users that have a similar taste (Bob and Molly vs Mary and Larry).

Now, let us consider the problem of completing the matrix if we only have access to a subset of its entries. This problem is known as **matrix completion**.

$$\begin{array}{ccccl} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} & \\ 1 & ? & 5 & 4 & \text{The Dark Knight} \\ ? & 1 & 4 & 5 & \text{Spiderman 3} \\ 4 & 5 & 2 & ? & \text{Love Actually} \\ 5 & 4 & 2 & 1 & \text{Bridget Jones's Diary} \\ 4 & 5 & 1 & 2 & \text{Pretty Woman} \\ 1 & 2 & ? & 5 & \text{Superman 2} \end{array} \tag{33}$$

Knowing that rating matrices are often approximately low-rank, we would like to compute a low-rank estimate from the available data. Achieving this through an optimization problem that penalizes the rank of the estimate is not computationally tractable. However penalizing the ℓ_1 norm of the singular values of the matrix does promote low-rank estimates. The ℓ_1 norm of the singular values is called the **nuclear norm** of the matrix, and is usually denoted by $||\cdot||_*$. In our example, we can leverage this insight in the following way.

1. We compute the average observed rating and subtract it from each entry in the matrix. We denote the vector of centered ratings by y.
2. We solve the optimization problem

$$\min_{\widetilde{X} \in \mathbb{R}^{m \times n}} \left|\left|\widetilde{X}_\Omega - y\right|\right|_2^2 + \lambda \left|\left|\widetilde{X}\right|\right|_* \tag{34}$$

where $\lambda > 0$ is a regularization parameter. We denote the set of observed indices of the matrix as Ω. For any $m \times n$ matrix M, M_Ω is a vector containing the entries of M indexed by Ω.

3. We add the average observed rating to the solution of the optimization problem and round each entry to the nearest integer.

The result is

$$\begin{array}{ccccc} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} & \\ 1 & 2\ (1) & 5 & 4 & \text{The Dark Knight} \\ 2\ (2) & 1 & 4 & 5 & \text{Spiderman 3} \\ 4 & 5 & 2 & 2\ (1) & \text{Love Actually} \\ 5 & 4 & 2 & 1 & \text{Bridget Jones's Diary} \\ 4 & 5 & 1 & 2 & \text{Pretty Woman} \\ 1 & 2 & 5\ (5) & 5 & \text{Superman 2} \end{array} \tag{35}$$

For comparison the original ratings are shown in brackets.

2.2 Low rank + sparse model

In computer vision, the problem of **background subtraction** is that of separating the background and foreground of a video sequence. Imagine that we take a video of a static background. We then stack the video frames in a matrix M, where each column corresponds to a vectorized frame. If the background is completely static, then all the frames are equal to a certain vector $f \in \mathbb{R}^m$ (m is the number of pixels in each frame) and the matrix is rank 1

$$M = \begin{bmatrix} f & f & \cdots & f \end{bmatrix} = f \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}.$$

If the background is not completely static, but instead experiences gradual changes, then the matrix containing the frames will be approximately low rank. Now, assume that there are sudden events in the foreground. If these events occupy a small part of the field of view and do not last very long, then the corresponding matrix can be modeled as sparse (most entries are equal to zero).

These observations motivate the following method for background subtraction. Stack the frames as columns of a matrix and separate the matrix into a low-rank and a sparse component. Applying the heuristic that the nuclear norm promotes low-rank structure and the ℓ_1 norm promotes sparsity, this suggests solving the optimization problem

$$\min_{\widetilde{L},\widetilde{S} \in \mathbb{R}^{m\times n}} \left|\left|\widetilde{L}\right|\right|_* + \lambda \left|\left|\widetilde{S}\right|\right|_1 \quad \text{such that } \widetilde{L} + \widetilde{S} = Y, \tag{36}$$

where $\lambda > 0$ is a regularization parameter and $||\cdot||_1$ denotes the ℓ_1 norm of a matrix interpreted as a vector. The results of applying this method to a real video sequence are shown in Figure 21.

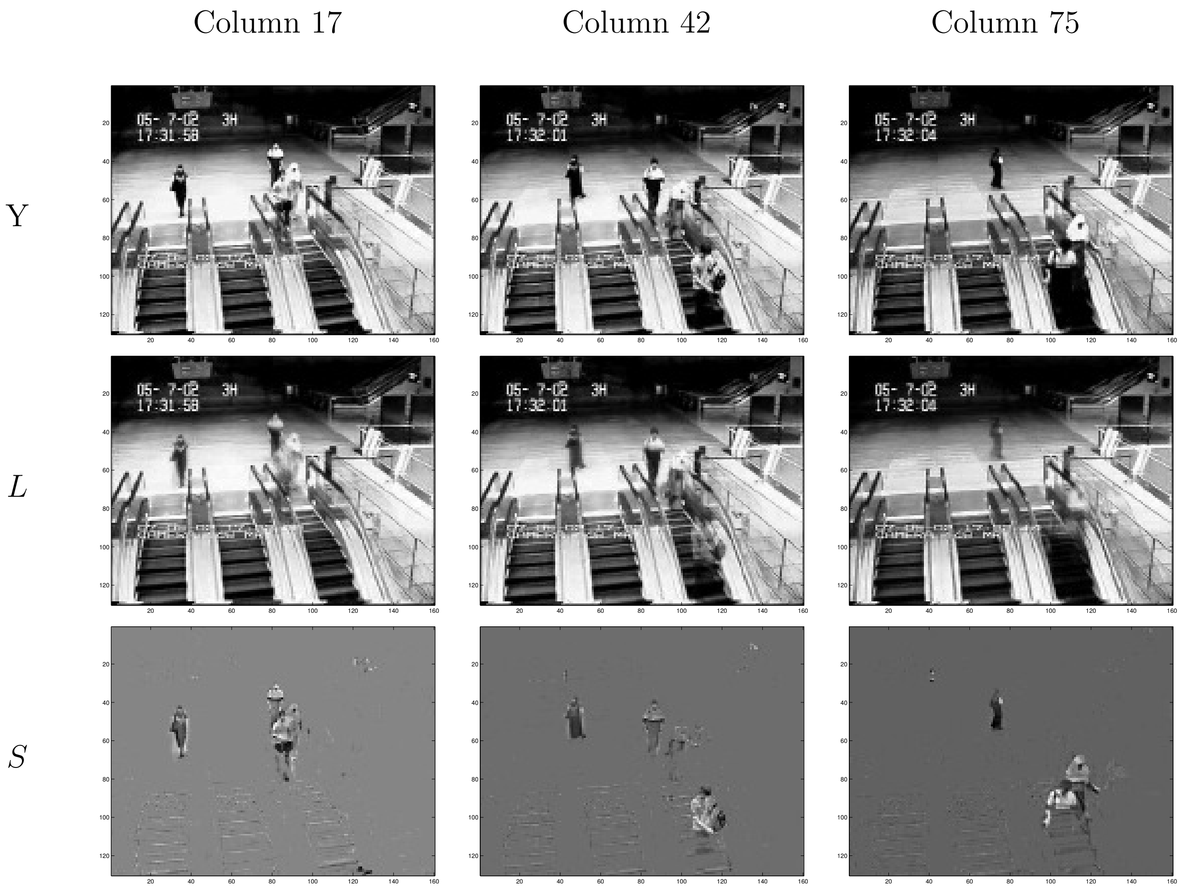

Figure 21: Background subtraction results from a video. This example is due to Stephen Becker. The code is available at http://cvxr.com/tfocs/demos/rpca.

2.3 Nonnegative matrix factorization

Topic modeling aims to learn the thematic structure of a text corpus automatically. Let us work on a simple example. We take six newspaper articles and compute the frequency of a list of words in each of them. Our final goal is to separate the words into different clusters that hopefully correspond to different topics. The following matrix contains the counts for each word and article. Each entry contains the number of times that the word corresponding to column j is mentioned in the article corresponding to row i.

$$A = \begin{array}{c} \begin{array}{ccccccccc} \text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band} \end{array} \\ \left(\begin{array}{ccccccccc} 6 & 1 & 1 & 0 & 0 & 1 & 9 & 0 & 8 \\ 1 & 0 & 9 & 5 & 8 & 1 & 0 & 1 & 0 \\ 8 & 1 & 0 & 1 & 0 & 0 & 9 & 1 & 7 \\ 0 & 7 & 1 & 0 & 0 & 9 & 1 & 7 & 0 \\ 0 & 5 & 6 & 7 & 5 & 6 & 0 & 7 & 2 \\ 1 & 0 & 8 & 5 & 9 & 2 & 0 & 0 & 1 \end{array}\right) \end{array} \begin{array}{c} \textit{Articles} \\ \text{a} \\ \text{b} \\ \text{c} \\ \text{d} \\ \text{e} \\ \text{f} \end{array}$$

Computing the singular-value decomposition of the matrix– after subtracting the mean of each entry as in (29)– we determine that the matrix is approximately low rank

$$A - \bar{A} = USV^T = U \begin{bmatrix} 19.32 & 0 & 0 & 0 & & \\ 0 & 14.46 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4.99 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.77 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.67 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.93 \end{bmatrix} V^T. \tag{37}$$

Unfortunately the singular vectors do not have an intuitive interpretation as in Section 2.1. In particular, they do not allow to cluster the words

$$\begin{array}{rcl} & & \begin{array}{cccccc} \text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{f} \end{array} \\ U_1 & = & (-0.51 \quad -0.40 \quad -0.54 \quad -0.11 \quad -0.38 \quad -0.38) \\ U_2 & = & (\ 0.19 \quad -0.45 \quad -0.19 \quad -0.69 \quad -0.2 \quad -0.46) \\ U_3 & = & (\ 0.14 \quad -0.27 \quad -0.09 \quad -0.58 \quad -0.69 \quad -0.29) \end{array} \tag{38}$$

or the articles

$$\begin{array}{rcl} & & \begin{array}{ccccccccc} \text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band} \end{array} \\ V_1 & = & (-0.38 \quad 0.05 \quad 0.4 \quad 0.27 \quad 0.4 \quad 0.17 \quad -0.52 \quad 0.14 \quad -0.38) \\ V_2 & = & (\ 0.16 \quad -0.46 \quad 0.33 \quad 0.15 \quad 0.38- \quad 0.49 \quad 0.1 \quad -0.47 \quad 0.12\ ) \\ V_3 & = & (-0.18 \quad -0.18 \quad -0.04 \quad -0.74 \quad -0.05 \quad 0.11 \quad -0.1 \quad -0.43 \quad -0.43) \end{array} \tag{39}$$

A problem here is that the singular vectors have negative entries that are difficult to interpret. In the case of rating prediction, negative ratings mean that a person does not like a movie. In contrast articles either are about a topic or they are not. Following this intuition, we can try to obtain a low-rank model with nonnegative entries, i.e. compute two matrices $W \in \mathbb{R}^{m\times r}$ and $H \in \mathbb{R}^{r\times n}$ such that

$$M \approx WH, \quad W_{i,j} \geq 0, \qquad 1 \leq i \leq m,\ 1 \leq j \leq r, \tag{40}$$

$$H_{i,j} \geq 0, \qquad 1 \leq i \leq r,\ 1 \leq i \leq n, \tag{41}$$

where the rank of the model is equal to $r < m, n$. This is known as **nonnegative matrix factorization**. Solving the problem for our example with $r = 3$ yields some interesting results. The entries of H allow to cluster the words into three topics,

$$\begin{array}{rcl ccccccccc l}
 & & & \text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band} & \\
H_1 & = & (& 0.34 & 0 & 3.73 & 2.54 & 3.67 & 0.52 & 0 & 0.35 & 0.35 &) \\
H_2 & = & (& 0 & 2.21 & 0.21 & 0.45 & 0 & 2.64 & 0.21 & 2.43 & 0.22 &) \\
H_3 & = & (& 3.22 & 0.37 & 0.19 & 0.2 & 0 & 0.12 & 4.13 & 0.13 & 3.43 &)
\end{array} \tag{42}$$

The first topic correspond to the entries that are not zero (or very small) in H_1: senate, election and vote. The second corresponds to H_2: GDP, stock and market. The third corresponds to H_3: singer, bass and band.

The entries of W allow to assign topics to articles. b, e and f are about politics (topic 1), d and e about economics (topic 3) and a and c about music (topic 3)

$$\begin{array}{rcl cccccc l}
 & & & \text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{f} & \\
W_1 & = & (& 0.03 & 2.23 & 0 & 0 & 1.59 & 2.24 &) \\
W_2 & = & (& 0.1 & 0 & 0.08 & 3.13 & 2.32 & 0 &) \\
W_3 & = & (& 2.13 & 0 & 2.22 & 0 & 0 & 0.03 &)
\end{array} \tag{43}$$

Finally, we check that the factorization provides a good fit to the data. The product WH is equal to

$$\begin{array}{c c}
\begin{array}{ccccccccc}
\text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band}
\end{array} & \mathit{Art.} \\
\left(\begin{array}{ccccccccc}
6.89\ (6) & 1.01\ (1) & 0.53\ (1) & 0.54\ (0) & 0.10\ (0) & 0.53\ (1) & 8.83\ (9) & 0.53\ (0) & 7.36\ (8) \\
0.75\ (1) & 0\ (0) & 8.32\ (9) & 5.66\ (5) & 8.18\ (8) & 1.15\ (1) & 0\ (0) & 0.78\ (1) & 0.78\ (0) \\
7.14\ (8) & 0.99\ (1) & 0.44\ (0) & 0.47\ (1) & 0\ (0) & 0.47\ (0) & 9.16\ (9) & 0.48\ (1) & 7.62\ (7) \\
0\ (0) & 7\ (6.91) & 0.67\ (1) & 1.41\ (0) & 0\ (0) & 8.28\ (9) & 0.65\ (1) & 7.60\ (7) & 0.69\ (0) \\
0.53\ (0) & 5.12\ (5) & 6.45\ (6) & 5.09\ (7) & 5.85\ (5) & 6.97\ (6) & 0.48\ (0) & 6.19\ (7) & 1.07\ (2) \\
0.86\ (1) & 0.01\ (0) & 8.36\ (8) & 5.69\ (5) & 8.22\ (9) & 1.16\ (2) & 0.14\ (0) & 0.79\ (0) & 0.9\ (1)
\end{array}\right) & \begin{array}{c} \text{a} \\ \text{b} \\ \text{c} \\ \text{d} \\ \text{e} \\ \text{f} \end{array}
\end{array}$$

For ease of comparison the values of A are shown in brackets.