

Lecture Notes 1: Vector spaces

In this chapter we review certain basic concepts of linear algebra, highlighting their application to signal processing.

1 Vector spaces

Embedding signals in a vector space essentially means that we can add them up or scale them to produce new signals.

Definition 1.1 (Vector space). *A vector space consists of a set \mathcal{V} , a scalar field that is usually either the real or the complex numbers and two operations $+$ and \cdot satisfying the following conditions.*

1. For any pair of elements $\vec{x}, \vec{y} \in \mathcal{V}$ the vector sum $\vec{x} + \vec{y}$ belongs to \mathcal{V} .
2. For any $\vec{x} \in \mathcal{V}$ and any scalar α , $\alpha \cdot \vec{x} \in \mathcal{V}$.
3. There exists a zero vector $\vec{0}$ such that $\vec{x} + \vec{0} = \vec{x}$ for any $\vec{x} \in \mathcal{V}$.
4. For any $\vec{x} \in \mathcal{V}$ there exists an additive inverse \vec{y} such that $\vec{x} + \vec{y} = \vec{0}$, usually denoted by $-\vec{x}$.
5. The vector sum is commutative and associative, i.e. for all $\vec{x}, \vec{y}, \vec{z} \in \mathcal{V}$

$$\vec{x} + \vec{y} = \vec{y} + \vec{x}, \quad (\vec{x} + \vec{y}) + \vec{z} = \vec{x} + (\vec{y} + \vec{z}). \quad (1)$$

6. Scalar multiplication is associative, for any scalars α and β and any $\vec{x} \in \mathcal{V}$

$$\alpha(\beta \cdot \vec{x}) = (\alpha\beta) \cdot \vec{x}. \quad (2)$$

7. Scalar and vector sums are both distributive, i.e. for any scalars α and β and any $\vec{x}, \vec{y} \in \mathcal{V}$

$$(\alpha + \beta) \cdot \vec{x} = \alpha \cdot \vec{x} + \beta \cdot \vec{x}, \quad \alpha \cdot (\vec{x} + \vec{y}) = \alpha \cdot \vec{x} + \alpha \cdot \vec{y}. \quad (3)$$

A subspace of a vector space \mathcal{V} is any subset of \mathcal{V} that is also itself a vector space.

From now on, for ease of notation we ignore the symbol for the scalar product \cdot , writing $\alpha \cdot \vec{x}$ as $\alpha \vec{x}$.

Depending on the signal of interest, we may want to represent it as an array of real or complex numbers, a matrix or a function. All of these mathematical objects can be represented as vectors in a vector space.

Example 1.2 (Real-valued and complex-valued vectors). \mathbb{R}^n with \mathbb{R} as its associated scalar field is a vector space where each vector consists of a set of n real-valued numbers. This is by far the most useful vector space in data analysis. For example, we can represent images with n pixels as vectors in \mathbb{R}^n , where each pixel is assigned to an entry.

Similarly, \mathbb{C}^n with \mathbb{C} as its associated scalar field is a vector space where each vector consists of a set of n complex-valued numbers. In both \mathbb{R}^n and \mathbb{C}^n , the zero vector is a vector containing zeros in every entry. \triangle

Example 1.3 (Matrices). Real-valued or complex-valued matrices of fixed dimensions form vector spaces with \mathbb{R} and \mathbb{C} respectively as their associated scalar fields. Adding matrices and multiplying matrices by scalars yields matrices of the same dimensions. In this case the zero vector corresponds to a matrix containing zeros in every entry. \triangle

Example 1.4 (Functions). Real-valued or complex-valued functions form a vector space (with \mathbb{R} and \mathbb{C} respectively as their associated scalar fields), since we can obtain new functions by adding functions or multiplying them by scalars. In this case the zero vector corresponds to a function that maps any number to zero. \triangle

Linear dependence indicates when a vector can be represented in terms of other vectors.

Definition 1.5 (Linear dependence/independence). A set of m vectors $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_m$ is linearly dependent if there exist m scalar coefficients $\alpha_1, \alpha_2, \dots, \alpha_m$ which are not all equal to zero and such that

$$\sum_{i=1}^m \alpha_i \vec{x}_i = \vec{0}. \quad (4)$$

Otherwise, the vectors are linearly independent. Equivalently, any vector in a linearly dependent set can be expressed as a linear combination of the rest, which is not the case for linearly independent sets.

We define the span of a set of vectors $\{\vec{x}_1, \dots, \vec{x}_m\}$ as the set of all possible linear combinations of the vectors:

$$\text{span}(\vec{x}_1, \dots, \vec{x}_m) := \left\{ \vec{y} \mid \vec{y} = \sum_{i=1}^m \alpha_i \vec{x}_i \quad \text{for some scalars } \alpha_1, \alpha_2, \dots, \alpha_m \right\}. \quad (5)$$

It is not difficult to check that the span of any set of vectors belonging to a vector space \mathcal{V} is a subspace of \mathcal{V} .

When working with a vector space, it is useful to consider the set of vectors with the smallest cardinality that spans the space. This is called a basis of the vector space.

Definition 1.6 (Basis). A basis of a vector space \mathcal{V} is a set of independent vectors $\{\vec{x}_1, \dots, \vec{x}_m\}$ such that

$$\mathcal{V} = \text{span}(\vec{x}_1, \dots, \vec{x}_m) \quad (6)$$

An important property of all bases in a vector space is that they have the same cardinality.

Theorem 1.7 (Proof in Section 8.1). *If a vector space \mathcal{V} has a basis with finite cardinality then every basis of \mathcal{V} contains the same number of vectors.*

This result allows us to define the dimension of a vector space.

Definition 1.8 (Dimension). *The dimension $\dim(\mathcal{V})$ of a vector space \mathcal{V} is the cardinality of any of its bases, or equivalently the number of linearly independent vectors that span \mathcal{V} .*

This definition coincides with the usual geometric notion of dimension in \mathbb{R}^2 and \mathbb{R}^3 : a line has dimension 1, whereas a plane has dimension 2 (as long as they contain the origin). Note that there exist infinite-dimensional vector spaces, such as the continuous real-valued functions defined on $[0, 1]$ (we will define a basis for this space later on).

The vector space that we use to model a certain problem is usually called the ambient space and its dimension the ambient dimension. In the case of \mathbb{R}^n the ambient dimension is n .

Lemma 1.9 (Dimension of \mathbb{R}^n). *The dimension of \mathbb{R}^n is n .*

Proof. Consider the set of vectors $\vec{e}_1, \dots, \vec{e}_n \subseteq \mathbb{R}^n$ defined by

$$\vec{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \vec{e}_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad \dots, \quad \vec{e}_n = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}. \quad (7)$$

One can easily check that this set is a basis. It is in fact the **standard basis** of \mathbb{R}^n . \square

2 Inner product

Up to now, the only operations we have considered are addition and multiplication by a scalar. In this section, we introduce a third operation, the inner product between two vectors.

Definition 2.1 (Inner product). *An inner product on a vector space \mathcal{V} is an operation $\langle \cdot, \cdot \rangle$ that maps a pair of vectors to a scalar and satisfies the following conditions.*

- If the scalar field associated to \mathcal{V} is \mathbb{R} , it is symmetric. For any $\vec{x}, \vec{y} \in \mathcal{V}$

$$\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle. \quad (8)$$

If the scalar field is \mathbb{C} , then for any $\vec{x}, \vec{y} \in \mathcal{V}$

$$\langle \vec{x}, \vec{y} \rangle = \overline{\langle \vec{y}, \vec{x} \rangle}, \quad (9)$$

where for any $\alpha \in \mathbb{C}$ $\bar{\alpha}$ is the complex conjugate of α .

- It is linear in the first argument, i.e. for any $\alpha \in \mathbb{R}$ and any $\vec{x}, \vec{y}, \vec{z} \in \mathcal{V}$

$$\langle \alpha \vec{x}, \vec{y} \rangle = \alpha \langle \vec{x}, \vec{y} \rangle, \quad (10)$$

$$\langle \vec{x} + \vec{y}, \vec{z} \rangle = \langle \vec{x}, \vec{z} \rangle + \langle \vec{y}, \vec{z} \rangle. \quad (11)$$

Note that if the scalar field is \mathbb{R} , it is also linear in the second argument by symmetry.

- It is positive definite: $\langle \vec{x}, \vec{x} \rangle$ is nonnegative for all $\vec{x} \in \mathcal{V}$ and if $\langle \vec{x}, \vec{x} \rangle = 0$ then $\vec{x} = 0$.

Definition 2.2 (Dot product). The dot product between two vectors $\vec{x}, \vec{y} \in \mathbb{R}^n$

$$\vec{x} \cdot \vec{y} := \sum_i \vec{x}[i] \vec{y}[i], \quad (12)$$

where $\vec{x}[i]$ is the i th entry of \vec{x} , is a valid inner product. \mathbb{R}^n endowed with the dot product is usually called a Euclidean space of dimension n .

Similarly, the dot product between two vectors $\vec{x}, \vec{y} \in \mathbb{C}^n$

$$\vec{x} \cdot \vec{y} := \sum_i \vec{x}[i] \overline{\vec{y}[i]} \quad (13)$$

is a valid inner product.

Definition 2.3 (Sample covariance). In statistics and data analysis, the sample covariance is used to quantify the joint fluctuations of two quantities or features. Let $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ be a data set where each example consists of a measurement of the two features. The sample covariance is defined as

$$\text{cov}((x_1, y_1), \dots, (x_n, y_n)) := \frac{1}{n-1} \sum_{i=1}^n (x_i - \text{av}(x_1, \dots, x_n))(y_i - \text{av}(y_1, \dots, y_n)) \quad (14)$$

where the average or sample mean of a set of n numbers is defined by

$$\text{av}(a_1, \dots, a_n) := \frac{1}{n} \sum_{i=1}^n a_i. \quad (15)$$

Geometrically the covariance is the scaled dot product of the two feature vectors after centering. The normalization constant is set so that if the measurements are modeled as independent samples $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)$ following the same distribution as two random variables \mathbf{x} and \mathbf{y} , then the sample covariance of the sequence is an unbiased estimate of the covariance of \mathbf{x} and \mathbf{y} ,

$$\mathbb{E}(\text{cov}((\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n))) = \text{Cov}(\mathbf{x}, \mathbf{y}) := \mathbb{E}((\mathbf{x} - \mathbb{E}(\mathbf{x}))(\mathbf{y} - \mathbb{E}(\mathbf{y}))). \quad (16)$$

Definition 2.4 (Matrix inner product). *The inner product between two $m \times n$ matrices A and B is*

$$\langle A, B \rangle := \text{tr}(A^T B) \quad (17)$$

$$= \sum_{i=1}^m \sum_{j=1}^n A_{ij} B_{ij}, \quad (18)$$

where the trace of an $n \times n$ matrix is defined as the sum of its diagonal

$$\text{tr}(M) := \sum_{i=1}^n M_{ii}. \quad (19)$$

The following lemma shows a useful property of the matrix inner product.

Lemma 2.5. *For any pair of $m \times n$ matrices A and B*

$$\text{tr}(B^T A) := \text{tr}(AB^T). \quad (20)$$

Proof. Both sides are equal to (18). \square

Note that the matrix inner product is equivalent to the inner product of the vectors with mn entries obtained by vectorizing the matrices.

Definition 2.6 (Function inner product). *A valid inner product between two complex-valued square-integrable functions f, g defined in an interval $[a, b]$ of the real line is*

$$\vec{f} \cdot \vec{g} := \int_a^b f(x) \overline{g(x)} dx. \quad (21)$$

3 Norms

The norm of a vector is a generalization of the concept of *length* in Euclidean space.

Definition 3.1 (Norm). *Let \mathcal{V} be a vector space, a norm is a function $\|\cdot\|$ from \mathcal{V} to \mathbb{R} that satisfies the following conditions.*

- It is homogeneous. For any scalar α and any $\vec{x} \in \mathcal{V}$

$$\|\alpha \vec{x}\| = |\alpha| \|\vec{x}\|. \quad (22)$$

- It satisfies the triangle inequality

$$\|\vec{x} + \vec{y}\| \leq \|\vec{x}\| + \|\vec{y}\|. \quad (23)$$

In particular, it is nonnegative (set $\vec{y} = -\vec{x}$).

- $\|\vec{x}\| = 0$ implies that \vec{x} is the zero vector $\vec{0}$.

A vector space equipped with a norm is called a normed space. Inner-product spaces are normed spaces because we can define a valid norm using the inner product.

Definition 3.2 (Inner-product norm). *The norm induced by an inner product is obtained by taking the square root of the inner product of the vector with itself,*

$$\|\vec{x}\|_{\langle \cdot, \cdot \rangle} := \sqrt{\langle \vec{x}, \vec{x} \rangle}. \quad (24)$$

Definition 3.3 (ℓ_2 norm). *The ℓ_2 norm is the norm induced by the dot product in \mathbb{R}^n or \mathbb{C}^n ,*

$$\|\vec{x}\|_2 := \sqrt{\vec{x} \cdot \vec{x}} = \sqrt{\sum_{i=1}^n \vec{x}[i]^2}. \quad (25)$$

In the case of \mathbb{R}^2 or \mathbb{R}^3 it is what we usually think of as the length of the vector.

Definition 3.4 (Sample variance and standard deviation). *Let $\{x_1, x_2, \dots, x_n\}$ be a set of real-valued data. The sample variance is defined as*

$$\text{var}(x_1, x_2, \dots, x_n) := \frac{1}{n-1} \sum_{i=1}^n (x_i - \text{av}(x_1, x_2, \dots, x_n))^2 \quad (26)$$

The sample standard deviation is the square root of the sample variance

$$\text{std}(x_1, x_2, \dots, x_n) := \sqrt{\text{var}(x_1, x_2, \dots, x_n)}. \quad (27)$$

Definition 3.5 (Sample variance and standard deviation). *In statistics and data analysis, the sample variance is used to quantify the fluctuations of a quantity around its average. Assume that we have n real-valued measurements x_1, x_2, \dots, x_n . The sample variance equals*

$$\text{var}(x_1, x_2, \dots, x_n) := \frac{1}{n-1} \sum_{i=1}^n (x_i - \text{av}(x_1, x_2, \dots, x_n))^2 \quad (28)$$

The normalization constant is set so that if the measurements are modeled as independent samples $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ following the same distribution as a random variable \mathbf{x} then the sample variance is an unbiased estimate of the variance of \mathbf{x} ,

$$\mathbb{E}(\text{var}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)) = \text{Var}(\mathbf{x}) := \mathbb{E}((\mathbf{x} - \mathbb{E}(\mathbf{x}))^2). \quad (29)$$

The sample standard deviation is the square root of the sample variance

$$\text{std}(x_1, x_2, \dots, x_n) := \sqrt{\text{var}(x_1, x_2, \dots, x_n)}. \quad (30)$$

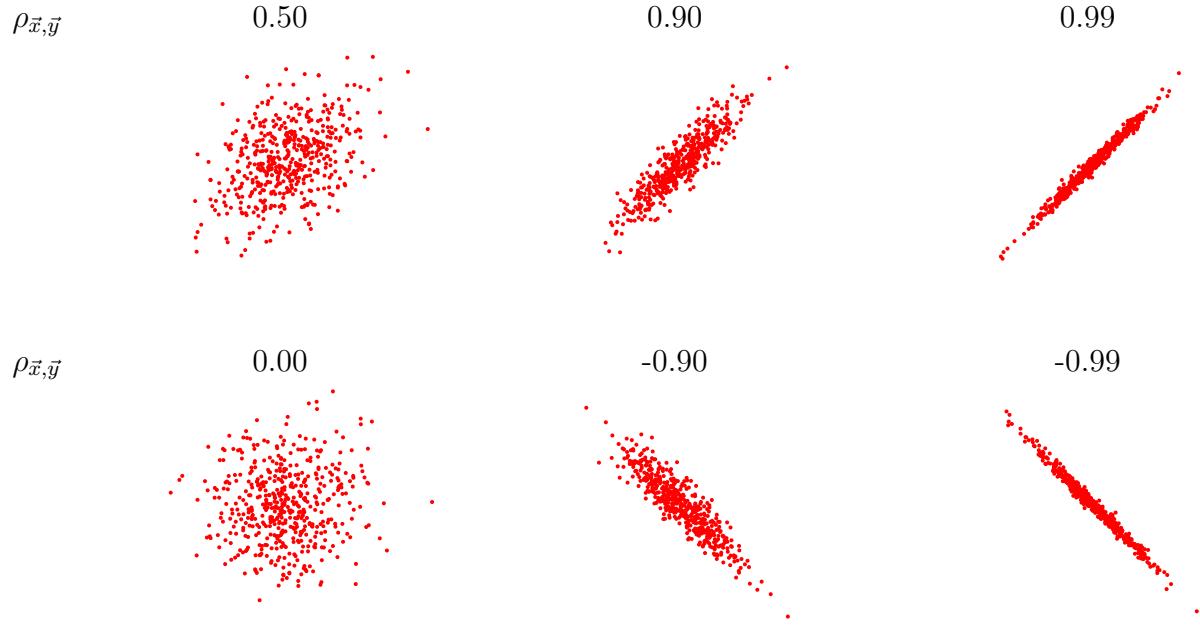


Figure 1: Scatter plot of the points $(\vec{x}_1, \vec{y}_1), (\vec{x}_2, \vec{y}_2), \dots, (\vec{x}_n, \vec{y}_n)$ for vectors with different correlation coefficients.

Definition 3.6 (Correlation coefficient). *When computing the sample covariance of two features the unit in which we express each quantity may severely affect the result. If one of the features is a distance, for example, expressing it in meters instead of kilometers increases the sample covariance by a factor of 1000! In order to obtain a measure of joint fluctuations that is invariant to scale, we normalize the covariance using the sample standard deviation of the features. This yields the correlation coefficient of the two quantities*

$$\rho_{(x_1, y_1), \dots, (x_n, y_n)} := \frac{\text{cov}((x_1, y_1), \dots, (x_n, y_n))}{\text{std}(x_1, \dots, x_n) \text{std}(y_1, \dots, y_n)}. \quad (31)$$

As illustrated in Figure 1 the correlation coefficient quantifies to what extent the entries of the two vectors are linearly related. Corollary 3.12 below shows that it is always between -1 and 1. If it is positive, we say that the two quantities are correlated. If it is negative, we say they are negatively correlated. If it is zero, we say that they are uncorrelated. In the following example we compute the correlation coefficient of some temperature data.

Example 3.7 (Correlation of temperature data). In this example we analyze temperature data gathered at a weather station in Oxford over 150 years.¹ We first compute the correlation between the temperature in January and the temperature in August. The correlation coefficient is $\rho = 0.269$. This means that the two quantities are positively correlated: warmer temperatures in January tend to correspond to warmer temperatures

¹The data is available at <http://www.metoffice.gov.uk/pub/data/weather/uk/climate/stationdata/oxforddata.txt>.

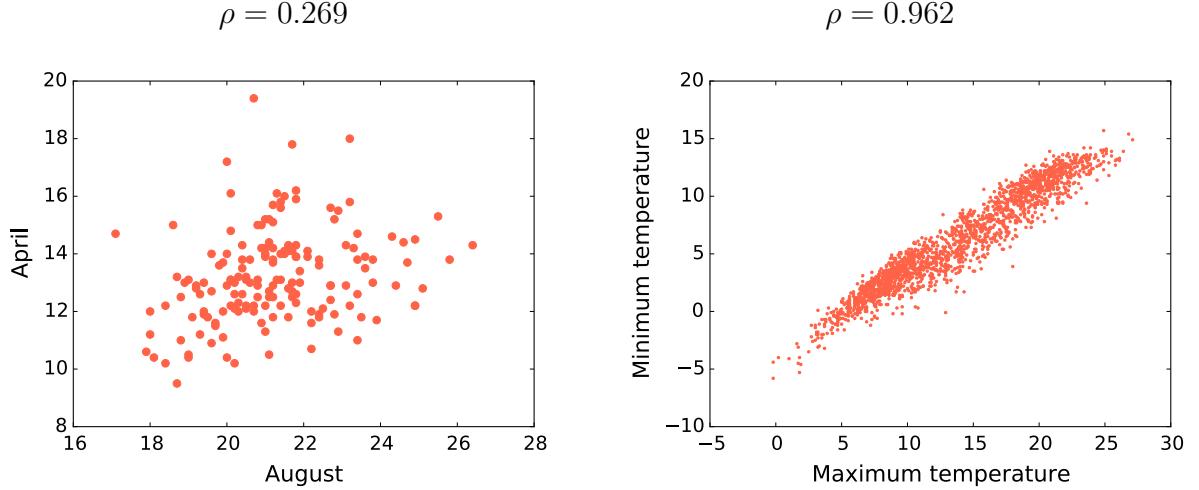


Figure 2: Scatterplot of the temperature in January and in August (left) and of the maximum and minimum monthly temperature (right) in Oxford over the last 150 years.

in August. The left image in Figure 2 shows a scatter plot where each point represents a different year. We repeat the experiment to compare the maximum and minimum temperature in the same month. The correlation coefficient between these two quantities is $\rho = 0.962$, indicating that the two quantities are extremely correlated. The right image in Figure 2 shows a scatter plot where each point represents a different month. \triangle

Definition 3.8 (Frobenius norm). *The Frobenius norm is the norm induced by the matrix inner product. For any matrix $A \in \mathbb{R}^{m \times n}$*

$$\|A\|_{\text{F}} := \sqrt{\text{tr}(A^T A)} = \sqrt{\sum_{i=1}^m \sum_{j=1}^n A_{ij}^2}. \quad (32)$$

It is equal to the ℓ_2 norm of the vectorized matrix.

Definition 3.9 (\mathcal{L}_2 norm). *The \mathcal{L}_2 norm is the norm induced by the dot product in the inner-product space of square-integrable complex-valued functions defined on an interval $[a, b]$,*

$$\|f\|_{\mathcal{L}_2} := \sqrt{\langle f, f \rangle} = \sqrt{\int_a^b |f(x)|^2 dx}. \quad (33)$$

The inner-product norm is clearly homogeneous by linearity and symmetric of the inner product. $\|\vec{x}\|_{\langle \cdot, \cdot \rangle} = 0$ implies $\vec{x} = 0$ because the inner product is positive semidefinite. We only need to establish that the triangle inequality holds to ensure that the inner-product is a valid norm. This follows from a classic inequality in linear algebra, which is proved in Section 8.2.

Theorem 3.10 (Cauchy-Schwarz inequality). *For any two vectors \vec{x} and \vec{y} in an inner-product space*

$$|\langle \vec{x}, \vec{y} \rangle| \leq \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle}. \quad (34)$$

Assume $\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \neq 0$, then

$$\langle \vec{x}, \vec{y} \rangle = -\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \iff \vec{y} = -\frac{\|\vec{y}\|_{\langle \cdot, \cdot \rangle}}{\|\vec{x}\|_{\langle \cdot, \cdot \rangle}} \vec{x}, \quad (35)$$

$$\langle \vec{x}, \vec{y} \rangle = \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \iff \vec{y} = \frac{\|\vec{y}\|_{\langle \cdot, \cdot \rangle}}{\|\vec{x}\|_{\langle \cdot, \cdot \rangle}} \vec{x}. \quad (36)$$

Corollary 3.11. *The norm induced by an inner product satisfies the triangle inequality.*

Proof.

$$\|\vec{x} + \vec{y}\|_{\langle \cdot, \cdot \rangle}^2 = \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2 + 2 \langle \vec{x}, \vec{y} \rangle \quad (37)$$

$\leq \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2 + 2 \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle}$ by the Cauchy-Schwarz inequality

$$= \left(\|\vec{x}\|_{\langle \cdot, \cdot \rangle} + \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \right)^2. \quad (38)$$

□

Another corollary of the Cauchy-Schwarz theorem is that the correlation coefficient is always between -1 and 1 and that if it equals either 1 or -1 then the two vectors are linearly dependent.

Corollary 3.12. *The correlation coefficient of two vectors \vec{x} and \vec{y} in \mathbb{R}^n satisfies*

$$-1 \leq \rho_{(x_1, y_1), \dots, (x_n, y_n)} \leq 1. \quad (39)$$

In addition,

$$\rho_{\vec{x}, \vec{y}} = -1 \iff y_i = \text{av}(y_1, \dots, y_n) - \frac{\text{std}(y_1, \dots, y_n)}{\text{std}(x_1, \dots, x_n)} (x_i - \text{av}(x_1, \dots, x_n)), \quad (40)$$

$$\rho_{\vec{x}, \vec{y}} = 1 \iff y_i = \text{av}(y_1, \dots, y_n) + \frac{\text{std}(y_1, \dots, y_n)}{\text{std}(x_1, \dots, x_n)} (x_i - \text{av}(x_1, \dots, x_n)). \quad (41)$$

Proof. The result follows from applying the Cauchy-Schwarz inequality to the vectors

$$\vec{a} := [x_1 - \text{av}(x_1, \dots, x_n) \ x_2 - \text{av}(x_1, \dots, x_n) \ \dots \ x_n - \text{av}(x_1, \dots, x_n)], \quad (42)$$

$$\vec{b} := [y_1 - \text{av}(y_1, \dots, y_n) \ y_2 - \text{av}(y_1, \dots, y_n) \ \dots \ y_n - \text{av}(y_1, \dots, y_n)], \quad (43)$$

since

$$\text{std}(x_1, x_2, \dots, x_n) = \|\vec{a}\|_2, \quad (44)$$

$$\text{std}(y_1, y_2, \dots, y_n) = \|\vec{b}\|_2, \quad (45)$$

$$\text{cov}((x_1, y_1), \dots, (x_n, y_n)) = \langle \vec{a}, \vec{b} \rangle. \quad (46)$$

□

Norms are not always induced by an inner product. The parallelogram law provides a simple identity that allows to check whether this is the case.

Theorem 3.13 (Parallelogram law). *A norm $\|\cdot\|$ on a vector space \mathcal{V} is induced by an inner product if and only if*

$$2\|\vec{x}\|^2 + 2\|\vec{y}\|^2 = \|\vec{x} - \vec{y}\|^2 + \|\vec{x} + \vec{y}\|^2, \quad (47)$$

for any $\vec{x}, \vec{y} \in \mathcal{V}$.

Proof. If the norm is induced by an inner product then

$$\|\vec{x} - \vec{y}\|^2 + \|\vec{x} + \vec{y}\|^2 = \langle \vec{x} - \vec{y}, \vec{x} - \vec{y} \rangle + \langle \vec{x} + \vec{y}, \vec{x} + \vec{y} \rangle \quad (48)$$

$$= 2\|\vec{x}\|^2 + 2\|\vec{y}\|^2 - (\vec{x}, \vec{y}) - (\vec{y}, \vec{x}) + (\vec{x}, \vec{y}) + (\vec{y}, \vec{x}) \quad (49)$$

$$= 2\|\vec{x}\|^2 + 2\|\vec{y}\|^2. \quad (50)$$

If the identity holds then it can be shown that

$$\langle \vec{x}, \vec{y} \rangle := \frac{1}{4} (\|\vec{x} + \vec{y}\|^2 - \|\vec{x} - \vec{y}\|^2) \quad (51)$$

is a valid inner product for real scalars and

$$\langle \vec{x}, \vec{y} \rangle := \frac{1}{4} (\|\vec{x} + \vec{y}\|^2 - \|\vec{x} - \vec{y}\|^2 - i (\|\vec{x} + i\vec{y}\|^2 - \|\vec{x} - i\vec{y}\|^2)) \quad (52)$$

is a valid inner product for complex scalars. \square

The following two norms do not satisfy the parallelogram identity and therefore are not induced by an inner product. Figure 3 compares their unit-norm balls with that of the ℓ_2 norm. Recall that the unit-norm ball of a norm $\|\cdot\|$ is the set of vectors \vec{x} such that $\|\vec{x}\| \leq 1$.

Definition 3.14 (ℓ_1 norm). *The ℓ_1 norm of a vector in \mathbb{R}^n or \mathbb{C}^n is the sum of the absolute values of the entries,*

$$\|\vec{x}\|_1 := \sum_{i=1}^n |\vec{x}[i]|. \quad (53)$$

Definition 3.15 (ℓ_∞ norm). *The ℓ_∞ norm of a vector in \mathbb{R}^n or \mathbb{C}^n is the maximum absolute value of its entries,*

$$\|\vec{x}\|_\infty := \max_i |\vec{x}[i]|. \quad (54)$$

Although they do not satisfy the Cauchy-Schwarz inequality, as they are not induced by any inner product, the ℓ_1 and ℓ_∞ norms can be used to bound the inner product between two vectors.

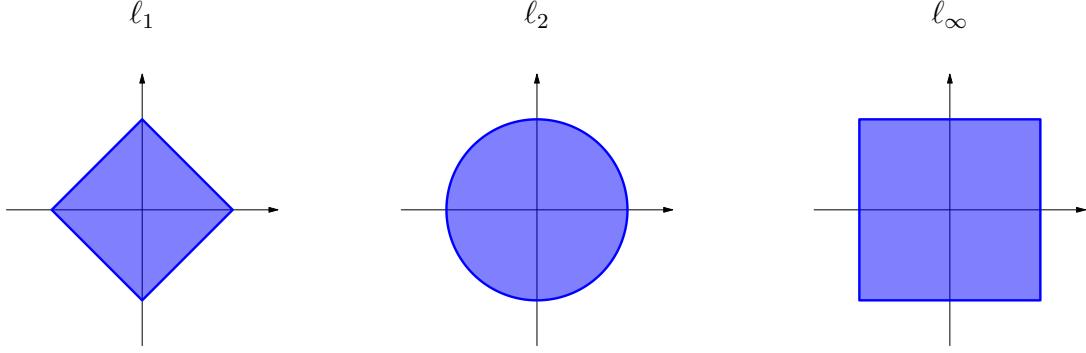


Figure 3: Unit ℓ_1 , ℓ_2 and ℓ_∞ norm balls.

Theorem 3.16 (Hölder's inequality). *For any two vectors \vec{x} and \vec{y} in \mathbb{R}^n or \mathbb{C}^n*

$$|\langle \vec{x}, \vec{y} \rangle| \leq \|\vec{x}\|_1 \|\vec{y}\|_\infty. \quad (55)$$

Proof.

$$|\langle \vec{x}, \vec{y} \rangle| \leq \sum_{i=1}^n |\vec{x}[i]| |\vec{y}[i]| \quad (56)$$

$$\leq \max_i |\vec{y}[i]| \sum_{i=1}^n |\vec{x}[i]| \quad (57)$$

$$= \|\vec{x}\|_1 \|\vec{y}\|_\infty. \quad (58)$$

□

Distances in a normed space can be measured using the norm of the difference between vectors.

Definition 3.17 (Distance). *The distance between two vectors \vec{x} and \vec{y} induced by a norm $\|\cdot\|$ is*

$$d(\vec{x}, \vec{y}) := \|\vec{x} - \vec{y}\|. \quad (59)$$

4 Nearest-neighbor classification

If we represent signals as vectors in a vector space, the distance between them quantifies their similarity. In this section we show how to exploit this to perform classification.

Definition 4.1 (Classification). *Given a set of k predefined classes, the classification problem is to decide what class a signal belongs to. The assignment is done using a training set of examples, each of which consists of a signals and its corresponding label.*

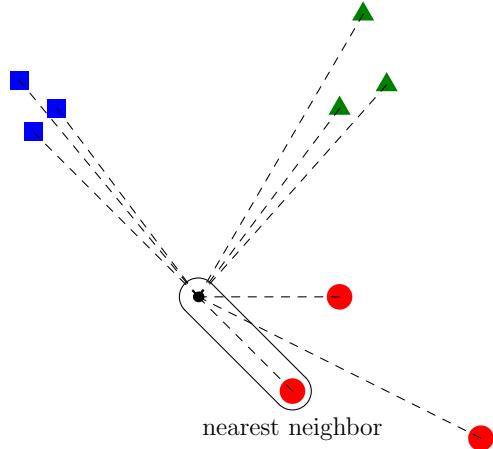


Figure 4: The nearest neighbor algorithm classifies points by assigning them the class of the closest point. In the diagram, the black point is assigned the *red circle* class because its nearest neighbor is a red circle.

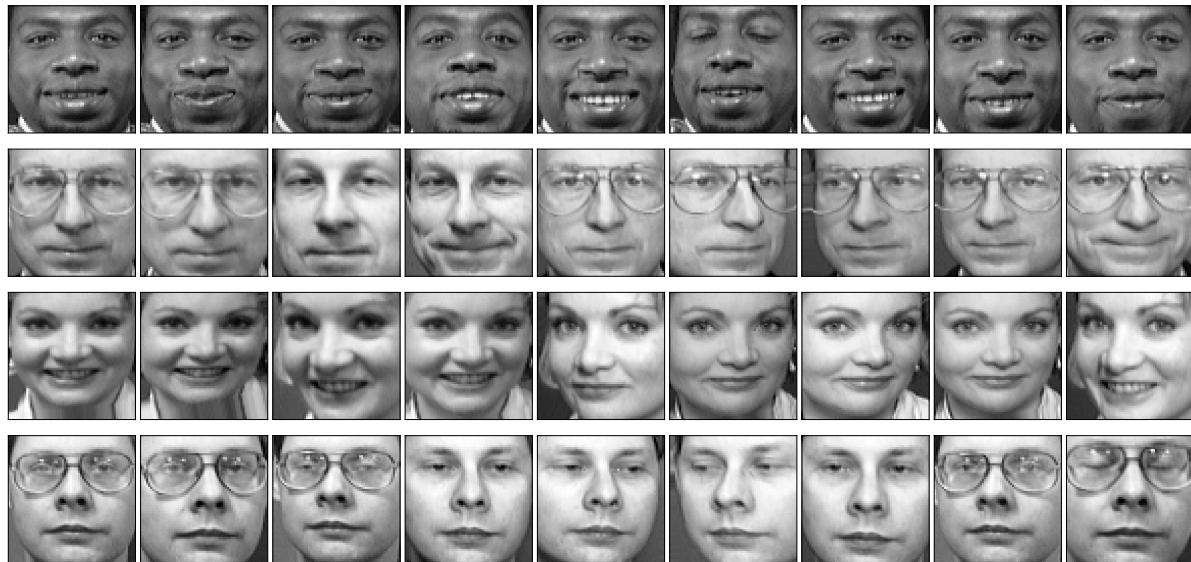


Figure 5: Training examples for four of the people in Example 4.3.

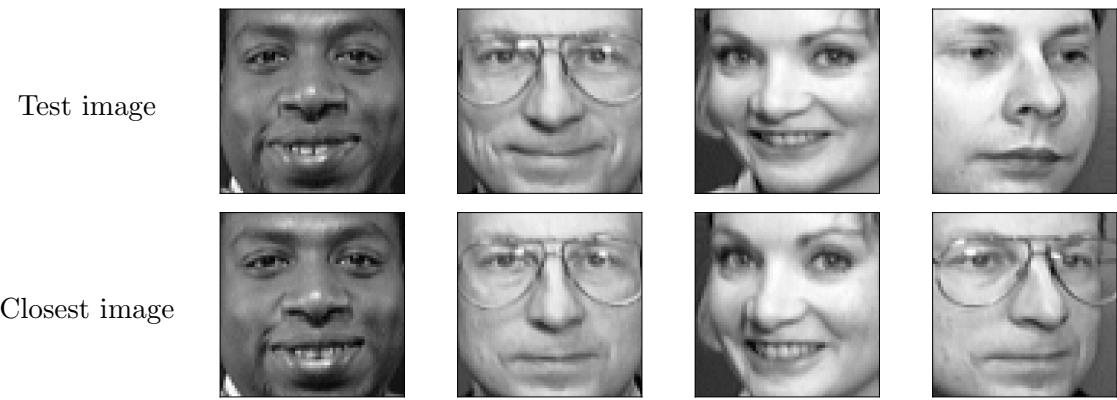


Figure 6: Results of nearest-neighbor classification for four of the people in Example 4.3. The assignments of the first three examples are correct, but the fourth is wrong.

The nearest-neighbor algorithm classifies signals by looking for the closest signal in the training set. Figure 4 shows a simple example.

Algorithm 4.2 (Nearest-neighbor classification). *Assume that the signals of interest can be represented by vectors in a vector space endowed with a norm denoted by $\|\cdot\|$. The training set consequently consists of n pairs of vectors and labels: $\{\vec{x}_1, l_1\}, \dots, \{\vec{x}_n, l_n\}$. To classify a test signal \vec{y} we find the closest signal in the training set in terms of the distance induced by the norm,*

$$i^* := \arg \min_{1 \leq i \leq n} \|\vec{y} - \vec{x}_i\|, \quad (60)$$

and assign the corresponding label l_{i^*} to \vec{y} .

Example 4.3 (Face recognition). The problem of face recognition consists of classifying images of faces to determine what person they correspond to. In this example we consider the Olivetti Faces data set². The training set consists of 360 64×64 images taken from 40 different subjects (9 per subject). Figure 5 shows some of the faces in the training set. The test set consists of an image of each subject, which is different from the ones in the training set. We apply nearest-neighbor algorithm to classify the faces in the test set, modeling each image as a vector in \mathbb{R}^{4096} and using the distance induced by the ℓ_2 norm. The algorithm classifies 36 of the 40 subjects correctly. Some of the results are shown in Figure 6. \triangle

5 Orthogonality

When the inner product between two vectors is zero, we say that the vectors are orthogonal.

²Available at <http://www.cs.nyu.edu/~roweis/data.html>

Definition 5.1 (Orthogonality). Two vectors \vec{x} and \vec{y} are orthogonal if and only if

$$\langle \vec{x}, \vec{y} \rangle = 0. \quad (61)$$

A vector \vec{x} is orthogonal to a set \mathcal{S} , if

$$\langle \vec{x}, \vec{s} \rangle = 0, \quad \text{for all } \vec{s} \in \mathcal{S}. \quad (62)$$

Two sets of $\mathcal{S}_1, \mathcal{S}_2$ are orthogonal if for any $\vec{x} \in \mathcal{S}_1, \vec{y} \in \mathcal{S}_2$

$$\langle \vec{x}, \vec{y} \rangle = 0. \quad (63)$$

The orthogonal complement of a subspace \mathcal{S} is

$$\mathcal{S}^\perp := \{\vec{x} \mid \langle \vec{x}, \vec{y} \rangle = 0 \quad \text{for all } \vec{y} \in \mathcal{S}\}. \quad (64)$$

Distances between orthogonal vectors measured in terms of the norm induced by the inner product are easy to compute.

Theorem 5.2 (Pythagorean theorem). If \vec{x} and \vec{y} are orthogonal vectors

$$\|\vec{x} + \vec{y}\|_{\langle \cdot, \cdot \rangle}^2 = \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2. \quad (65)$$

Proof. By linearity of the inner product

$$\|\vec{x} + \vec{y}\|_{\langle \cdot, \cdot \rangle}^2 = \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2 + 2 \langle \vec{x}, \vec{y} \rangle \quad (66)$$

$$= \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2. \quad (67)$$

□

If we want to show that a vector is orthogonal to a certain subspace, it is enough to show that it is orthogonal to every vector in a basis of the subspace.

Lemma 5.3. Let \vec{x} be a vector and \mathcal{S} a subspace of dimension n . If for any basis $\vec{b}_1, \vec{b}_2, \dots, \vec{b}_n$ of \mathcal{S} ,

$$\langle \vec{x}, \vec{b}_i \rangle = 0, \quad 1 \leq i \leq n, \quad (68)$$

then \vec{x} is orthogonal to \mathcal{S} .

Proof. Any vector $v \in \mathcal{S}$ can be represented as $v = \sum_i \alpha_{i=1}^n \vec{b}_i$ for $\alpha_1, \dots, \alpha_n \in \mathbb{R}$, from (68)

$$\langle \vec{x}, v \rangle = \left\langle \vec{x}, \sum_i \alpha_{i=1}^n \vec{b}_i \right\rangle = \sum_i \alpha_{i=1}^n \langle \vec{x}, \vec{b}_i \rangle = 0. \quad (69)$$

□

If the vectors in a basis are normalized and mutually orthogonal, then the norm is said to be orthonormal.

Definition 5.4 (Orthonormal basis). *A basis of mutually orthogonal vectors with inner-product norm equal to one is called an orthonormal basis.*

It is very easy to find the coefficients of a vector in an orthonormal basis: we just need to compute the dot products with the basis vectors.

Lemma 5.5 (Coefficients in an orthonormal basis). *If $\{\vec{u}_1, \dots, \vec{u}_n\}$ is an orthonormal basis of a vector space \mathcal{V} , for any vector $\vec{x} \in \mathcal{V}$*

$$\vec{x} = \sum_{i=1}^n \langle \vec{u}_i, \vec{x} \rangle \vec{u}_i. \quad (70)$$

Proof. Since $\{\vec{u}_1, \dots, \vec{u}_n\}$ is a basis,

$$\vec{x} = \sum_{i=1}^m \alpha_i \vec{u}_i \quad \text{for some } \alpha_1, \alpha_2, \dots, \alpha_m \in \mathbb{R}. \quad (71)$$

Immediately,

$$\langle \vec{u}_i, \vec{x} \rangle = \left\langle \vec{u}_i, \sum_{i=1}^m \alpha_i \vec{u}_i \right\rangle = \sum_{i=1}^m \alpha_i \langle \vec{u}_i, \vec{u}_i \rangle = \alpha_i \quad (72)$$

because $\langle \vec{u}_i, \vec{u}_i \rangle = 1$ and $\langle \vec{u}_i, \vec{u}_j \rangle = 0$ for $i \neq j$. \square

We can construct an orthonormal basis for any subspace in a vector space by applying the Gram-Schmidt method to a set of linearly independent vectors spanning the subspace.

Algorithm 5.6 (Gram-Schmidt). *Consider a set of linearly independent vectors $\vec{x}_1, \dots, \vec{x}_m$ in \mathbb{R}^n . To obtain an orthonormal basis of the span of these vectors we:*

1. Set $\vec{u}_1 := \vec{x}_1 / \|\vec{x}_1\|_2$.
2. For $i = 1, \dots, m$, compute

$$\vec{v}_i := \vec{x}_i - \sum_{j=1}^{i-1} \langle \vec{u}_j, \vec{x}_i \rangle \vec{u}_j. \quad (73)$$

and set $\vec{u}_i := \vec{v}_i / \|\vec{v}_i\|_2$.

It is not difficult to show that the resulting set of vectors $\vec{u}_1, \dots, \vec{u}_m$ is an orthonormal basis for the span of $\vec{x}_1, \dots, \vec{x}_m$: they are orthonormal by construction and their span is the same as that of the original set of vectors.

6 Orthogonal projection

If two subspaces are disjoint, i.e. their only common point is the origin, then a vector that can be expressed as a sum of a vector from each subspace is said to belong to their direct sum.

Definition 6.1 (Direct sum). *Let \mathcal{V} be a vector space. For any subspaces $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{V}$ such that*

$$\mathcal{S}_1 \cap \mathcal{S}_2 = \{0\} \quad (74)$$

the direct sum is defined as

$$\mathcal{S}_1 \oplus \mathcal{S}_2 := \{\vec{x} \mid \vec{x} = \vec{s}_1 + \vec{s}_2 \quad \vec{s}_1 \in \mathcal{S}_1, \vec{s}_2 \in \mathcal{S}_2\}. \quad (75)$$

The representation of a vector in the direct sum of two subspaces as the sum of vectors from the subspaces is unique.

Lemma 6.2. *Any vector $\vec{x} \in \mathcal{S}_1 \oplus \mathcal{S}_2$ has a unique representation*

$$\vec{x} = \vec{s}_1 + \vec{s}_2 \quad \vec{s}_1 \in \mathcal{S}_1, \vec{s}_2 \in \mathcal{S}_2. \quad (76)$$

Proof. If $\vec{x} \in \mathcal{S}_1 \oplus \mathcal{S}_2$ then by definition there exist $\vec{s}_1 \in \mathcal{S}_1, \vec{s}_2 \in \mathcal{S}_2$ such that $\vec{x} = \vec{s}_1 + \vec{s}_2$. Assume $\vec{x} = \vec{v}_1 + \vec{v}_2$, $\vec{v}_1 \in \mathcal{S}_1, \vec{v}_2 \in \mathcal{S}_2$, then $\vec{s}_1 - \vec{v}_1 = \vec{s}_2 - \vec{v}_2$. This implies that $\vec{s}_1 - \vec{v}_1$ and $\vec{s}_2 - \vec{v}_2$ are in \mathcal{S}_1 and also in \mathcal{S}_2 . However, $\mathcal{S}_1 \cap \mathcal{S}_2 = \{0\}$, so we conclude $\vec{s}_1 = \vec{v}_1$ and $\vec{s}_2 = \vec{v}_2$. \square

Given a vector x and a subspace \mathcal{S} , the orthogonal projection of \vec{x} onto \mathcal{S} is the vector that we reach when we go from x to \mathcal{S} following a direction that is orthogonal to \mathcal{S} . This allows to express \vec{x} as the sum of a component that belongs to \mathcal{S} and another that belongs to its orthogonal complement. This is illustrated by a simple example in Figure 7.

Definition 6.3 (Orthogonal projection). *Let \mathcal{V} be a vector space. The orthogonal projection of a vector $\vec{x} \in \mathcal{V}$ onto a subspace $\mathcal{S} \subseteq \mathcal{V}$ is a vector denoted by $\mathcal{P}_{\mathcal{S}} \vec{x}$ such that $\vec{x} - \mathcal{P}_{\mathcal{S}} \vec{x} \in \mathcal{S}^{\perp}$.*

Theorem 6.4 (Properties of the orthogonal projection). *Let \mathcal{V} be a vector space. Every vector $\vec{x} \in \mathcal{V}$ has a unique orthogonal projection $\mathcal{P}_{\mathcal{S}} \vec{x}$ onto any subspace $\mathcal{S} \subseteq \mathcal{V}$ of finite dimension. In particular \vec{x} can be expressed as*

$$\vec{x} = \mathcal{P}_{\mathcal{S}} \vec{x} + \mathcal{P}_{\mathcal{S}^{\perp}} \vec{x}. \quad (77)$$

For any vector $s \in \mathcal{S}$

$$\langle \vec{x}, s \rangle = \langle \mathcal{P}_{\mathcal{S}} \vec{x}, s \rangle. \quad (78)$$

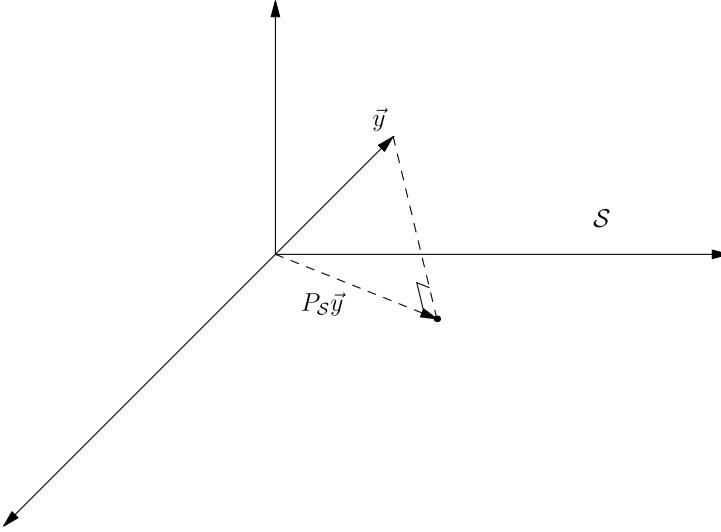


Figure 7: Orthogonal projection of a vector $\vec{x} \in \mathbb{R}^2$ on a two-dimensional subspace \mathcal{S} .

For any orthonormal basis $\vec{b}_1, \dots, \vec{b}_m$ of \mathcal{S} ,

$$\mathcal{P}_{\mathcal{S}} \vec{x} = \sum_{i=1}^m \langle \vec{x}, \vec{b}_i \rangle \vec{b}_i. \quad (79)$$

The orthogonal projection is a linear operation. For any vectors \vec{x} and \vec{y} and any subspace \mathcal{S}

$$\mathcal{P}_{\mathcal{S}} (\vec{x} + \vec{y}) = \mathcal{P}_{\mathcal{S}} \vec{x} + \mathcal{P}_{\mathcal{S}} \vec{y}. \quad (80)$$

Proof. Let us denote the dimension of \mathcal{S} by m . Since m is finite, there exists an orthonormal basis of \mathcal{S} : $\vec{b}'_1, \dots, \vec{b}'_m$. Consider the vector

$$\vec{p} := \sum_{i=1}^m \langle \vec{x}, \vec{b}'_i \rangle \vec{b}'_i. \quad (81)$$

It turns out that $\vec{x} - \vec{p}$ is orthogonal to every vector in the basis. For $1 \leq j \leq m$,

$$\langle \vec{x} - \vec{p}, \vec{b}'_j \rangle = \left\langle \vec{x} - \sum_{i=1}^m \langle \vec{x}, \vec{b}'_i \rangle \vec{b}'_i, \vec{b}'_j \right\rangle \quad (82)$$

$$= \langle \vec{x}, \vec{b}'_j \rangle - \sum_{i=1}^m \langle \vec{x}, \vec{b}'_i \rangle \langle \vec{b}'_i, \vec{b}'_j \rangle \quad (83)$$

$$= \langle \vec{x}, \vec{b}'_j \rangle - \langle \vec{x}, \vec{b}'_j \rangle = 0, \quad (84)$$

so $\vec{x} - \vec{p} \in \mathcal{S}^\perp$ and \vec{p} is an orthogonal projection. Since $\mathcal{S} \cap \mathcal{S}^\perp = \{0\}$ ³ there cannot be two other vectors $\vec{x}_1 \in \mathcal{S}, \vec{x}_2 \in \mathcal{S}^\perp$ such that $\vec{x} = \vec{x}_1 + \vec{x}_2$ so the orthogonal projection is unique.

³For any vector \vec{v} that belongs to both \mathcal{S} and \mathcal{S}^\perp $\langle \vec{v}, \vec{v} \rangle = \|\vec{v}\|_2^2 = 0$, which implies $\vec{v} = 0$.

Notice that $\vec{o} := \vec{x} - \vec{p}$ is a vector in \mathcal{S}^\perp such that $\vec{x} - \vec{o} = \vec{p}$ is in S and therefore in $(\mathcal{S}^\perp)^\perp$. This implies that \vec{o} is the orthogonal projection of \vec{x} onto \mathcal{S}^\perp and establishes (77).

Equation (78) follows immediately from the orthogonality of any vector in \mathcal{S} and $\mathcal{P}_{\mathcal{S}^\perp} \vec{x}$.

Equation (79) follows from (78).

Finally, linearity follows from (79) and linearity of the inner product

$$\mathcal{P}_S (\vec{x} + \vec{y}) = \sum_{i=1}^m \langle \vec{x} + \vec{y}, \vec{b}_i \rangle \vec{b}_i \quad (85)$$

$$= \sum_{i=1}^m \langle \vec{x}, \vec{b}_i \rangle \vec{b}_i + \sum_{i=1}^m \langle \vec{y}, \vec{b}_i \rangle \vec{b}_i \quad (86)$$

$$= \mathcal{P}_S \vec{x} + \mathcal{P}_S \vec{y}. \quad (87)$$

□

The following corollary relates the dimensions of a subspace and its orthogonal complement within a finite-dimensional vector space.

Corollary 6.5 (Dimension of orthogonal complement). *Let \mathcal{V} be a finite-dimensional vector space, for any subspace $\mathcal{S} \subseteq \mathcal{V}$*

$$\dim(\mathcal{S}) + \dim(\mathcal{S}^\perp) = \dim(\mathcal{V}). \quad (88)$$

Proof. Consider a set of vectors \mathcal{B} defined as the union of a basis of \mathcal{S} , which has $\dim(\mathcal{S})$ elements, and a basis of \mathcal{S}^\perp , which has $\dim(\mathcal{S}^\perp)$ elements. Due to the orthogonality of \mathcal{S} and \mathcal{S}^\perp all the $\dim(\mathcal{S}) + \dim(\mathcal{S}^\perp)$ vectors in \mathcal{B} are linearly independent and by (77) they span the whole space, which establishes the result. □

Computing the inner-product norm of the projection of a vector onto a subspace is easy if we have access to an orthonormal basis.

Lemma 6.6 (Norm of the projection). *The norm of the projection of an arbitrary vector $\vec{x} \in \mathcal{V}$ onto a subspace $\mathcal{S} \subseteq \mathcal{V}$ of dimension d can be written as*

$$\|\mathcal{P}_S \vec{x}\|_{\langle \cdot, \cdot \rangle} = \sqrt{\sum_i^d \langle \vec{b}_i, \vec{x} \rangle^2} \quad (89)$$

for any orthonormal basis $\vec{b}_1, \dots, \vec{b}_d$ of \mathcal{S} .

Proof. By (79)

$$\|\mathcal{P}_S \vec{x}\|_{\langle \cdot, \cdot \rangle}^2 = \langle \mathcal{P}_S \vec{x}, \mathcal{P}_S \vec{x} \rangle \quad (90)$$

$$= \left\langle \sum_i^d \langle \vec{b}_i, \vec{x} \rangle \vec{b}_i, \sum_j^d \langle \vec{b}_j, \vec{x} \rangle \vec{b}_j \right\rangle \quad (91)$$

$$= \sum_i^d \sum_j^d \langle \vec{b}_i, \vec{x} \rangle \langle \vec{b}_j, \vec{x} \rangle \langle \vec{b}_i, \vec{b}_j \rangle \quad (92)$$

$$= \sum_i^d \langle \vec{b}_i, \vec{x} \rangle^2. \quad (93)$$

□

The orthogonal projection of a vector \vec{x} onto a subspace \mathcal{S} has a very intuitive interpretation that generalizes to other sets: it is the vector in \mathcal{S} that is closest to \vec{x} in the distance associated to the inner-product norm.

Theorem 6.7 (The orthogonal projection is closest). *The orthogonal projection $\mathcal{P}_S \vec{x}$ of a vector \vec{x} onto a subspace \mathcal{S} is the solution to the optimization problem*

$$\underset{\vec{u}}{\text{minimize}} \quad \|\vec{x} - \vec{u}\|_{\langle \cdot, \cdot \rangle} \quad (94)$$

$$\text{subject to} \quad \vec{u} \in \mathcal{S}. \quad (95)$$

Proof. Take any point $\vec{s} \in \mathcal{S}$ such that $\vec{s} \neq \mathcal{P}_S \vec{x}$

$$\|\vec{x} - \vec{s}\|_{\langle \cdot, \cdot \rangle}^2 = \|\vec{x} - \mathcal{P}_S \vec{x} + \mathcal{P}_S \vec{x} - \vec{s}\|_{\langle \cdot, \cdot \rangle}^2 \quad (96)$$

$$= \|\vec{x} - \mathcal{P}_S \vec{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\mathcal{P}_S \vec{x} - \vec{s}\|_{\langle \cdot, \cdot \rangle}^2 \quad (97)$$

$$> \|\vec{x} - \mathcal{P}_S \vec{x}\|_{\langle \cdot, \cdot \rangle}^2 \quad \text{because } \vec{s} \neq \mathcal{P}_S \vec{x}, \quad (98)$$

where (97) follows from the Pythagorean theorem since because $\mathcal{P}_{S^\perp} \vec{x} = \vec{x} - \mathcal{P}_S \vec{x}$ belongs to S^\perp and $\mathcal{P}_S \vec{x} - \vec{s}$ to S . □

7 Denoising

In this section we consider the problem of denoising a signal that has been corrupted by an unknown perturbation.

Definition 7.1 (Denoising). *The aim of denoising is to estimate a signal from perturbed measurements. If the noise is assumed to be additive, the data are modeled as the sum of the signal \vec{x} and a perturbation \vec{z}*

$$\vec{y} := \vec{x} + \vec{z}. \quad (99)$$

The goal is to estimate \vec{x} from \vec{y} .

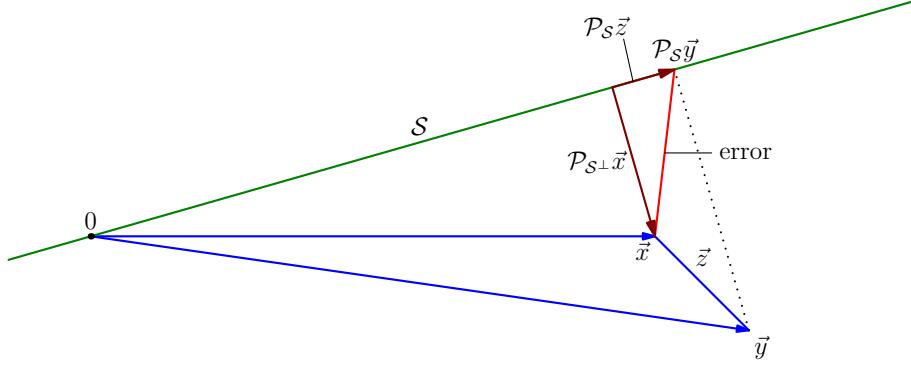


Figure 8: Illustration of the two terms in the error decomposition of Lemma 7.3 for a simple denoising example, where the data vector is denoised by projecting onto a 1D subspace.

In order to denoise a signal, we need to have some prior information about its structure. For instance, we may suspect that the signal is well approximated as belonging to a predefined subspace. This suggests estimating the signal by projecting the noisy data onto the subspace.

Algorithm 7.2 (Denoising via orthogonal projection). *Denoising a data vector \vec{y} via orthogonal projection onto a subspace \mathcal{S} , consists of setting the signal estimate to $\mathcal{P}_{\mathcal{S}} \vec{y}$, the projection of the noisy data onto \mathcal{S} .*

The following lemma gives a simple decomposition of the error incurred by this denoising technique, which is illustrated in Figure 8.

Lemma 7.3. *Let $\vec{y} := \vec{x} + \vec{z}$ and let \mathcal{S} be an arbitrary subspace, then*

$$\|\vec{x} - \mathcal{P}_{\mathcal{S}} \vec{y}\|_2^2 = \|\mathcal{P}_{\mathcal{S}^\perp} \vec{x}\|_2^2 + \|\mathcal{P}_{\mathcal{S}} \vec{z}\|_2^2. \quad (100)$$

Proof. By linearity of the orthogonal projection

$$\vec{x} - \mathcal{P}_{\mathcal{S}} \vec{y} = \vec{x} - \mathcal{P}_{\mathcal{S}} \vec{x} - \mathcal{P}_{\mathcal{S}} \vec{z} \quad (101)$$

$$= \mathcal{P}_{\mathcal{S}^\perp} \vec{x} - \mathcal{P}_{\mathcal{S}} \vec{z}, \quad (102)$$

so the result follows by the Pythagorean theorem. \square

The error is divided into two terms. The first term is the projection of the signal onto the orthogonal complement of the chosen subspace \mathcal{S} . For this term to be small, the signal must be well represented by its projection onto \mathcal{S} . The second term is the projection of the noise onto \mathcal{S} . This term will be small if the noise is mostly orthogonal to \mathcal{S} . This makes sense: denoising via projection will only be effective if the projection preserves the signal but eliminates the noise.

$$\mathcal{S}_1 := \text{span} \left(\begin{array}{c} \text{Image 1} \\ \text{Image 2} \\ \text{Image 3} \\ \text{Image 4} \\ \text{Image 5} \\ \text{Image 6} \\ \text{Image 7} \\ \text{Image 8} \\ \text{Image 9} \end{array} \right)$$

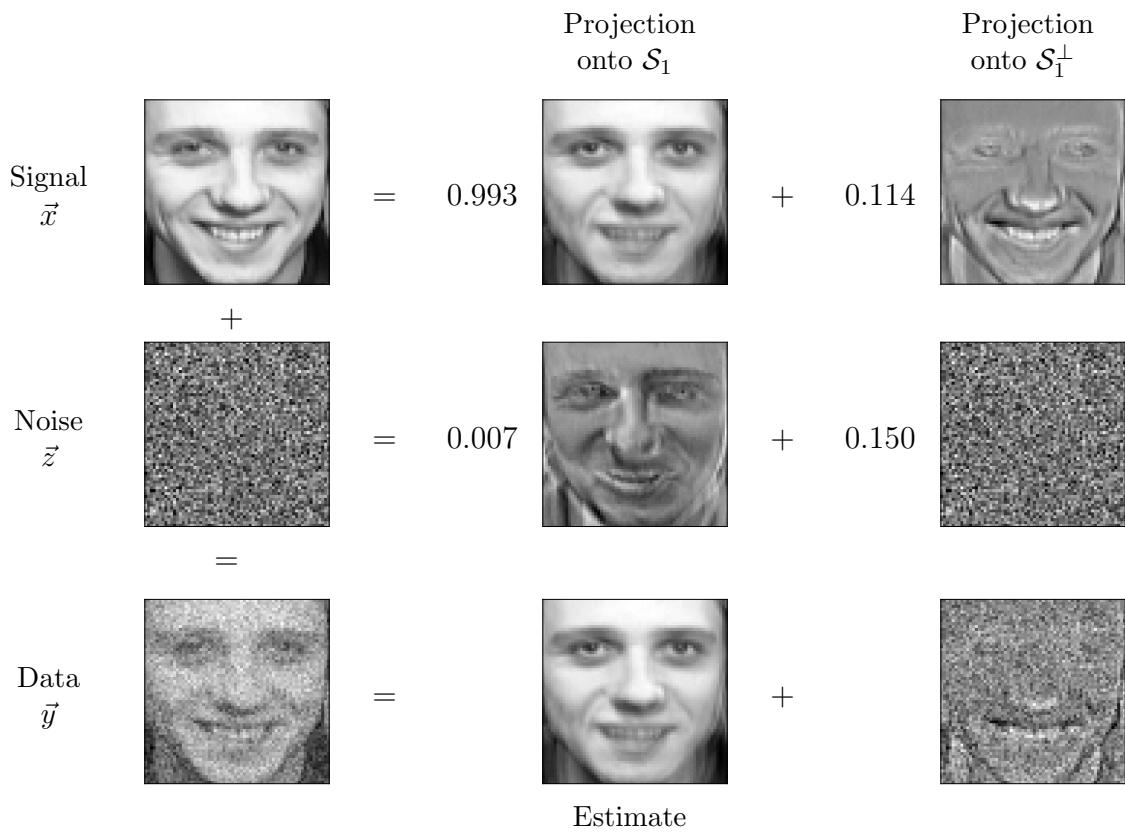


Figure 9: Denoising of the image of a face by projection onto the span of 9 other images of the same person, denoted by \mathcal{S}_1 . The original image is normalized to have ℓ_2 norm equal to one. The noise has ℓ_2 norm equal to 0.1. The ℓ_2 norms of the projections of the original image and of the noise onto \mathcal{S}_1 and its orthogonal complement are indicated beside the corresponding images. The estimate is the projection of the noisy image onto \mathcal{S}_1 .

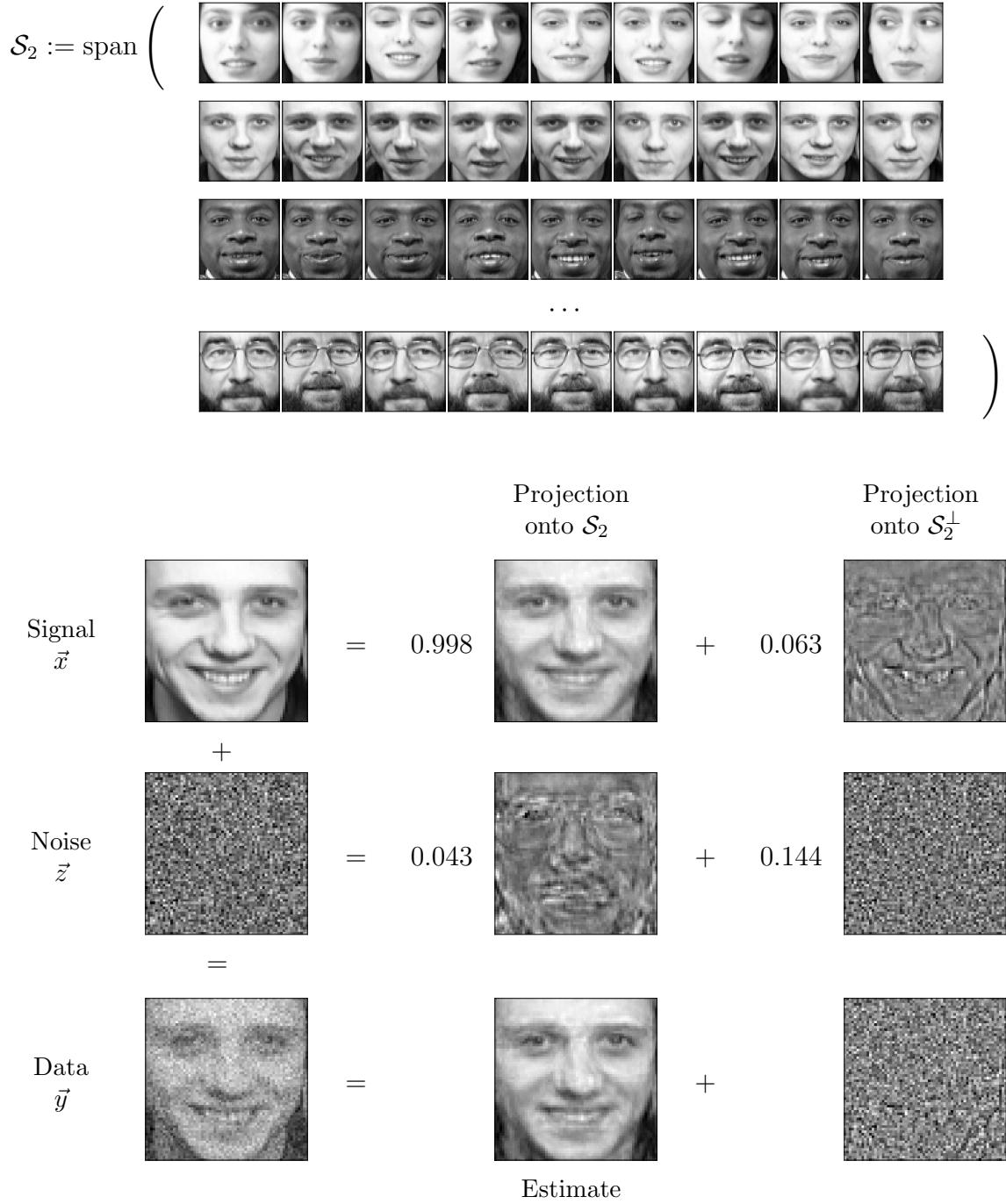


Figure 10: Denoising of the image of a face by projection onto the span of 360 other images of different people (including 9 of the same person), denoted by \mathcal{S}_2 . The original image is normalized to have ℓ_2 norm equal to one. The noise has ℓ_2 norm equal to 0.1. The ℓ_2 norms of the projections of the original image and of the noise onto \mathcal{S}_2 and its orthogonal complement are indicated beside the corresponding images. The estimate is the projection of the noisy image onto \mathcal{S}_2 .

Example 7.4 (Denoising of face images). In this example we again consider the Olivetti Faces dataset⁴, with a training set of 360 64×64 images taken from 40 different subjects (9 per subject). The goal is to denoise a test image \vec{x} of the same dimensions that is not in the training set. The data \vec{y} are obtained by adding noise to the test image. The entries of the noise vector z are sampled independently from a Gaussian distribution and scaled so that the signal-to-noise ratio equals 10,

$$\text{SNR} := \frac{\|\vec{x}\|_2}{\|\vec{z}\|_2} = 6.67. \quad (103)$$

We denoise the image by projecting onto two subspaces:

- \mathcal{S}_1 : the span of the 9 images in the training set that correspond to the same subject.
- \mathcal{S}_2 : the span of the 360 images in the training set.

Figure 9 and 10 show the results for \mathcal{S}_1 and \mathcal{S}_2 respectively. The relative ℓ_2 -norm error of both estimates is:

$$\frac{\|\vec{x} - \mathcal{P}_{\mathcal{S}_1} \vec{y}\|_2}{\|\vec{x}\|_2} = 0.114, \quad (104)$$

$$\frac{\|\vec{x} - \mathcal{P}_{\mathcal{S}_2} \vec{y}\|_2}{\|\vec{x}\|_2} = 0.078. \quad (105)$$

The two estimates look very different. To interpret the results we separate the error into two components, as in Lemma 7.3. The norm of the projection of the noise onto \mathcal{S}_1 is smaller than its projection onto \mathcal{S}_2

$$0.007 = \frac{\|\mathcal{P}_{\mathcal{S}_1} \vec{z}\|_2}{\|\vec{x}\|_2} < \frac{\|\mathcal{P}_{\mathcal{S}_2} \vec{z}\|_2}{\|\vec{x}\|_2} = 0.043. \quad (106)$$

The reason is that \mathcal{S}_1 has lower dimension. The ratio between the two projections ($0.043/0.007 = 6.14$) is close to the square root of the ratio of the dimensions of the subspaces (6.32). This is not a coincidence, as we will see later on. However, the projection of the signal onto \mathcal{S}_1 is not as close to \vec{x} as the projection onto \mathcal{S}_2 , which is particularly obvious in the lower half of the face,

$$0.063 = \frac{\|\mathcal{P}_{\mathcal{S}_2^\perp} \vec{x}\|_2}{\|\vec{x}\|_2} < \frac{\|\mathcal{P}_{\mathcal{S}_1^\perp} \vec{x}\|_2}{\|\vec{x}\|_2} = 0.114. \quad (107)$$

The conclusion is that the projection onto \mathcal{S}_2 produces a noisier looking image (because the noise-component of the error is larger), which nevertheless looks more similar to the original signal (because the signal-component of the error is smaller). This illustrates an important tradeoff when using projection-based denoising: subspaces with larger dimension approximate the signal better, but don't suppress the noise as much. \triangle

⁴Available at <http://www.cs.nyu.edu/~roweis/data.html>

8 Proofs

8.1 Proof of Theorem 1.7

We prove the claim by contradiction. Assume that we have two bases $\{\vec{x}_1, \dots, \vec{x}_m\}$ and $\{\vec{y}_1, \dots, \vec{y}_n\}$ such that $m < n$ (or the second set has infinite cardinality). The proof follows from applying the following lemma m times (setting $r = 0, 1, \dots, m-1$) to show that $\{\vec{y}_1, \dots, \vec{y}_m\}$ spans \mathcal{V} and hence $\{\vec{y}_1, \dots, \vec{y}_n\}$ must be linearly dependent.

Lemma 8.1. *Under the assumptions of the theorem, if $\{\vec{y}_1, \vec{y}_2, \dots, \vec{y}_r, \vec{x}_{r+1}, \dots, \vec{x}_m\}$ spans \mathcal{V} then $\{\vec{y}_1, \dots, \vec{y}_{r+1}, \vec{x}_{r+2}, \dots, \vec{x}_m\}$ also spans \mathcal{V} (possibly after rearranging the indices $r+1, \dots, m$) for $r = 0, 1, \dots, m-1$.*

Proof. Since $\{\vec{y}_1, \vec{y}_2, \dots, \vec{y}_r, \vec{x}_{r+1}, \dots, \vec{x}_m\}$ spans \mathcal{V}

$$\vec{y}_{r+1} = \sum_{i=1}^r \beta_i \vec{y}_i + \sum_{i=r+1}^m \gamma_i \vec{x}_i, \quad \beta_1, \dots, \beta_r, \gamma_{r+1}, \dots, \gamma_m \in \mathbb{R}, \quad (108)$$

where at least one of the γ_j is non zero, as $\{\vec{y}_1, \dots, \vec{y}_n\}$ is linearly independent by assumption. Without loss of generality (here is where we might need to rearrange the indices) we assume that $\gamma_{r+1} \neq 0$, so that

$$\vec{x}_{r+1} = \frac{1}{\gamma_{r+1}} \left(\sum_{i=1}^r \beta_i \vec{y}_i - \sum_{i=r+2}^m \gamma_i \vec{x}_i \right). \quad (109)$$

This implies that any vector in the span of $\{\vec{y}_1, \vec{y}_2, \dots, \vec{y}_r, \vec{x}_{r+1}, \dots, \vec{x}_m\}$, i.e. in \mathcal{V} , can be represented as a linear combination of vectors in $\{\vec{y}_1, \dots, \vec{y}_{r+1}, \vec{x}_{r+2}, \dots, \vec{x}_m\}$, which completes the proof. \square

8.2 Proof of Theorem 3.10

If $\|\vec{x}\|_{\langle \cdot, \cdot \rangle} = 0$ then $\vec{x} = \vec{0}$ because the inner product is positive semidefinite, which implies $\langle \vec{x}, \vec{y} \rangle = 0$ and consequently that (34) holds with equality. The same is true if $\|\vec{y}\|_{\langle \cdot, \cdot \rangle} = 0$.

Now assume that $\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \neq 0$ and $\|\vec{y}\|_{\langle \cdot, \cdot \rangle} \neq 0$. By semidefiniteness of the inner product,

$$0 \leq \left\| \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \vec{x} + \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \vec{y} \right\|^2 = 2 \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2 + 2 \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \langle \vec{x}, \vec{y} \rangle, \quad (110)$$

$$0 \leq \left\| \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \vec{x} - \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \vec{y} \right\|^2 = 2 \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 \|\vec{y}\|_{\langle \cdot, \cdot \rangle}^2 - 2 \|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle} \langle \vec{x}, \vec{y} \rangle. \quad (111)$$

These inequalities establish (34).

Let us prove (40) by proving both implications.

(\implies) Assume $\langle \vec{x}, \vec{y} \rangle = -\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle}$. Then (110) equals zero, so $\|\vec{y}\|_{\langle \cdot, \cdot \rangle} \vec{x} = -\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \vec{y}$ because the inner product is positive semidefinite.

(\Leftarrow) Assume $\|\vec{y}\|_{\langle \cdot, \cdot \rangle} \vec{x} = -\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \vec{y}$. Then one can easily check that (110) equals zero, which implies $\langle \vec{x}, \vec{y} \rangle = -\|\vec{x}\|_{\langle \cdot, \cdot \rangle} \|\vec{y}\|_{\langle \cdot, \cdot \rangle}$.

The proof of (41) is identical (using (111) instead of (110)).

Lecture Notes 2: Matrices

Matrices are rectangular arrays of numbers, which are extremely useful for data analysis. They can be interpreted as vectors in a vector space, linear functions or sets of vectors.

1 Basic properties

1.1 Column and row space

A matrix can be used to represent a set of vectors stored as columns or rows. The span of these vectors are called the column and row space of the matrix respectively.

Definition 1.1 (Column and row space). *The column space $\text{col}(A)$ of a matrix A is the span of its columns. The row space $\text{row}(A)$ is the span of its rows.*

Interestingly, the row space and the column space of all matrices have the same dimension. We name this quantity the rank of the matrix.

Definition 1.2 (Rank). *The rank of a matrix is the dimension of its column and of its row space.*

Theorem 1.3 (Proof in Section 5.1). *The rank is well defined. For any matrix A*

$$\dim(\text{col}(A)) = \dim(\text{row}(A)). \quad (1)$$

If the dimension of the row and column space of an $m \times n$ matrix where $m < n$ is equal to m then the the rows are all linearly independent. Similarly, if $m > n$ and the rank is n then the columns are all linearly independent. In general, when the rank equals $\min\{n, m\}$ we say that the matrix is *full rank*.

Recall that the inner product between two matrices $A, B \in \mathbb{R}^{m \times n}$ is given by the trace of $A^T B$, and the norm induced by this inner product is the Frobenius norm. If the column spaces of two matrices are orthogonal, then the matrices are also orthogonal.

Lemma 1.4. *If the column spaces of any pair of matrices $A, B \in \mathbb{R}^{m \times n}$ are orthogonal then*

$$\langle A, B \rangle = 0. \quad (2)$$

Proof. We can write the inner product as a sum of products between the columns of A

and B , which are all zero under the assumption of the lemma

$$\langle A, B \rangle := \text{tr} (A^T B) \quad (3)$$

$$= \sum_{i=1}^n \langle A_{:,i}, B_{:,i} \rangle \quad (4)$$

$$= 0. \quad (5)$$

□

The following corollary follows immediately from Lemma 1.4 and the Pythagorean theorem.

Corollary 1.5. *If the column spaces of any pair of matrices $A, B \in \mathbb{R}^{m \times n}$ are orthogonal*

$$\|A + B\|_F^2 = \|A\|_F^2 + \|B\|_F^2. \quad (6)$$

1.2 Linear maps

A map is a transformation that assigns a vector to another vector, possibly belonging to a different vector space. The transformation is linear if it maps any linear combination of input vectors to the same linear combination of the corresponding outputs.

Definition 1.6 (Linear map). *Given two vector spaces \mathcal{V} and \mathcal{R} associated to the same scalar field, a linear map $f : \mathcal{V} \rightarrow \mathcal{R}$ is a map from vectors in \mathcal{V} to vectors in \mathcal{R} such that for any scalar α and any vectors $\vec{x}_1, \vec{x}_2 \in \mathcal{V}$*

$$f(\vec{x}_1 + \vec{x}_2) = f(\vec{x}_1) + f(\vec{x}_2), \quad (7)$$

$$f(\alpha \vec{x}_1) = \alpha f(\vec{x}_1). \quad (8)$$

Every complex or real matrix of dimensions $m \times n$ defines a map from the space of n -dimensional vectors to the space of m -dimensional vectors through an operation called matrix-vector product. We denote the i th row of a matrix A by $A_{i:}$, the j th column by $A_{:j}$ and the (i, j) entry by A_{ij} .

Definition 1.7 (Matrix-vector product). *The product of a matrix $A \in \mathbb{C}^{m \times n}$ and a vector $\vec{x} \in \mathbb{C}^n$ is a vector $A\vec{x} \in \mathbb{C}^m$, such that*

$$(A\vec{x})[i] = \sum_{j=1}^n A_{ij}\vec{x}[j]. \quad (9)$$

For real matrices, each entry in the matrix-vector product is the dot product between a row of the matrix and the vector,

$$(A\vec{x})[i] = \langle A_{i:}, \vec{x} \rangle. \quad (10)$$

The matrix-vector product can also be interpreted in terms of the column of the matrix,

$$A\vec{x} = \sum_{j=1}^n \vec{x}[j] A_{:j}. \quad (11)$$

$A\vec{x}$ is a linear combination of the columns of A weighted by the entries in \vec{x} .

Matrix-vector multiplication is clearly linear. Perhaps surprisingly, the converse is also true: *any* linear map between \mathbb{C}^n and \mathbb{C}^m (or between \mathbb{R}^n and \mathbb{R}^m) can be represented by a matrix.

Theorem 1.8 (Equivalence between matrices and linear maps). *For finite m, n every linear map $f : \mathbb{C}^m \rightarrow \mathbb{C}^n$ can be uniquely represented by a matrix $F \in \mathbb{C}^{m \times n}$.*

Proof. The matrix is

$$F := [f(\vec{e}_1) \ f(\vec{e}_2) \ \cdots \ f(\vec{e}_n)], \quad (12)$$

i.e., the columns of the matrix are the result of applying f to the standard basis. Indeed, for any vector $\vec{x} \in \mathbb{C}^n$

$$f(x) = f\left(\sum_{i=1}^n \vec{x}[i]\vec{e}_i\right) \quad (13)$$

$$= \sum_{i=1}^n \vec{x}[i]f(\vec{e}_i) \quad \text{by (7) and (8)} \quad (14)$$

$$= F\vec{x}. \quad (15)$$

The i th column of any matrix that represents the linear map must equal $f(\vec{e}_i)$ by (11), so the representation is unique. \square

When a matrix $\mathbb{C}^{m \times n}$ is *fat*, i.e., $n > m$, we often say that it *projects* vectors onto a lower dimensional space. Note that such projections are not the same as the orthogonal projections we described in Lecture Notes 1. When a matrix is *tall*, i.e., $m > n$, we say that it *lifts* vectors to a higher-dimensional space.

1.3 Adjoint

The adjoint of a linear map f from an inner-product space \mathcal{V} and another inner product space \mathcal{R} maps elements of \mathcal{R} back to \mathcal{V} in a way that preserves their inner product with images of f .

Definition 1.9 (Adjoint). *Given two vector spaces \mathcal{V} and \mathcal{R} associated to the same scalar field with inner products $\langle \cdot, \cdot \rangle_{\mathcal{V}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{R}}$ respectively, the adjoint $f^* : \mathcal{R} \rightarrow \mathcal{V}$ of a linear map $f : \mathcal{V} \rightarrow \mathcal{R}$ satisfies*

$$\langle f(\vec{x}), \vec{y} \rangle_{\mathcal{R}} = \langle \vec{x}, f^*(\vec{y}) \rangle_{\mathcal{V}} \quad (16)$$

for all $\vec{x} \in \mathcal{V}$ and $\vec{y} \in \mathcal{R}$.

In the case of finite-dimensional spaces, the adjoint corresponds to the conjugate Hermitian transpose of the matrix associated to the linear map.

Definition 1.10 (Conjugate transpose). *The entries of the conjugate transpose $A^* \in \mathbb{C}^{n \times m}$ of a matrix $A \in \mathbb{C}^{m \times n}$ are of the form*

$$(A^*)_{ij} = \overline{A_{ji}}, \quad 1 \leq i \leq n, 1 \leq j \leq m. \quad (17)$$

If the entries of the matrix are all real, this is just the transpose of the matrix.

Lemma 1.11 (Equivalence between conjugate transpose and adjoint). *For finite m, n the adjoint $f^* : \mathbb{C}^n \rightarrow \mathbb{C}^m$ of a linear map $f : \mathbb{C}^m \rightarrow \mathbb{C}^n$ represented by a matrix F corresponds to the conjugate transpose of the matrix F^* .*

Proof. For any $\vec{x} \in \mathbb{C}^n$ and $\vec{y} \in \mathbb{C}^m$,

$$\langle f(\vec{x}), \vec{y} \rangle_{\mathbb{C}^m} = \sum_{i=1}^m f(\vec{x})_i \overline{\vec{y}_i} \quad (18)$$

$$= \sum_{i=1}^m \overline{\vec{y}_i} \sum_{j=1}^n F_{ij} \vec{x}_j \quad (19)$$

$$= \sum_{j=1}^n \vec{x}_j \sum_{i=1}^m \overline{F_{ij}} \vec{y}_i \quad (20)$$

$$= \langle \vec{x}, F^* \vec{y} \rangle_{\mathbb{C}^n}. \quad (21)$$

By Theorem 1.8 a linear map is represented by a unique matrix (you can check that the adjoint map is linear), which completes the proof. \square

A matrix that is equal to its adjoint is called self-adjoint. Self-adjoint real matrices are symmetric: they are equal to their transpose. Self-adjoint complex matrices are Hermitian: they are equal to their conjugate transpose.

1.4 Range and null space

The range of a linear map is the set of all possible vectors that can be reached by applying the map.

Definition 1.12 (Range). *Let \mathcal{V} and \mathcal{R} be vector spaces associated to the same scalar field, the range of a map $f : \mathcal{V} \rightarrow \mathcal{R}$ is the set of vectors in \mathcal{R} that can be reached by applying f to a vector in \mathcal{V} :*

$$\text{range}(f) := \{ \vec{y} \mid \vec{y} = f(\vec{x}) \text{ for some } \vec{x} \in \mathcal{V} \}. \quad (22)$$

The range of a matrix is the range of its associated linear map.

The range of a matrix is the same as its column space.

Lemma 1.13 (The range is the column space). *For any matrix $A \in \mathbb{C}^{m \times n}$*

$$\text{range}(A) = \text{col}(A). \quad (23)$$

Proof. For any \vec{x} , $A\vec{x}$ is a linear combination of the columns of A , so the range is a subset of the column space. In addition, every column of A is in the range, since $A_{:,i} = A\vec{e}_i$ for $1 \leq i \leq n$, so the column space is a subset of the range and both sets are equal. \square

The null space of a map is the set of vectors that are mapped to zero.

Definition 1.14 (Null space). *Let \mathcal{V} and \mathcal{R} be vector spaces associated to the same scalar field, the null space of a map $f : \mathcal{V} \rightarrow \mathcal{R}$ is the set of vectors in \mathcal{V} that are mapped to the zero vector in \mathcal{R} by f :*

$$\text{null}(f) := \left\{ \vec{x} \mid f(\vec{x}) = \vec{0} \right\}. \quad (24)$$

The null space of a matrix is the null space of its associated linear map.

It is not difficult to prove that the null space of a map is a vector space, as long as the map is linear, since in that case scaling or adding elements of the null space yield vectors that are mapped to zero by the map.

The following lemma shows that for real matrices the null space is the orthogonal complement of the row space of the matrix.

Lemma 1.15. *For any matrix $A \in \mathbb{R}^{m \times n}$*

$$\text{null}(A) = \text{row}(A)^\perp. \quad (25)$$

Proof. Any vector \vec{x} in the row space of A can be written as $\vec{x} = A^T \vec{z}$, for some vector $\vec{z} \in \mathbb{R}^m$. If $y \in \text{null}(A)$ then

$$\langle \vec{y}, \vec{x} \rangle = \langle \vec{y}, A^T \vec{z} \rangle \quad (26)$$

$$= \langle A\vec{y}, \vec{z} \rangle \quad (27)$$

$$= 0. \quad (28)$$

So $\text{null}(A) \subseteq \text{row}(A)^\perp$.

If $x \in \text{row}(A)^\perp$ then in particular it is orthogonal to every row of A , so $Ax = 0$ and $\text{row}(A)^\perp \subseteq \text{null}(A)$. \square

An immediate corollary of Lemmas 1.13 and 1.15 is that the dimension of the range and the null space add up to the ambient dimension of the row space.

Corollary 1.16. *Let $A \in \mathbb{R}^{m \times n}$*

$$\dim(\text{range}(A)) + \dim(\text{null}(A)) = n. \quad (29)$$

This means that for every matrix $A \in \mathbb{R}^{m \times n}$ we can decompose any vector in \mathbb{R}^n into two components: one is in the row space and is mapped to a nonzero vector in \mathbb{C}^m , the other is in the null space and is mapped to the zero vector.

1.5 Identity matrix and inverse

The identity matrix is a matrix that maps any vector to itself.

Definition 1.17 (Identity matrix). *The identity matrix of dimensions $n \times n$ is*

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}. \quad (30)$$

For any $\vec{x} \in \mathbb{C}^n$, $I\vec{x} = \vec{x}$.

Square matrices have a unique inverse if they are full rank, since in that case the null space has dimension 0 and the associated linear map is a bijection. The inverse is a matrix that reverses the effect of the matrix on any vector.

Definition 1.18 (Matrix inverse). *The inverse of a square matrix $A \in \mathbb{C}^{n \times n}$ is a matrix $A^{-1} \in \mathbb{C}^{n \times n}$ such that*

$$AA^{-1} = A^{-1}A = I. \quad (31)$$

1.6 Orthogonal and projection matrices

We often use the letters $U \in \mathbb{R}^{m \times n}$ or $V \in \mathbb{R}^{m \times n}$ for matrices with orthonormal columns. If such matrices are square then they are said to be orthogonal. Orthogonal matrices represent linear maps that do not affect the magnitude of a vector, just its direction.

Definition 1.19 (Orthogonal matrix). *An orthogonal matrix is a real-valued square matrix such that its inverse is equal to its transpose,*

$$U^T U = U U^T = I. \quad (32)$$

By definition, the columns $U_{:,1}, U_{:,2}, \dots, U_{:,n}$ of any $n \times n$ orthogonal matrix have unit norm and orthogonal to each other, so they form an orthonormal basis (it's somewhat confusing that orthogonal matrices are not called orthonormal matrices instead). Applying U^T to a vector $\vec{x} \in \mathbb{R}^n$ is equivalent to computing the coefficients of its representation in the basis formed by the columns of U . Applying U to $U^T \vec{x}$ recovers \vec{x} by scaling each basis vector with the corresponding coefficient:

$$\vec{x} = U U^T \vec{x} = \sum_{i=1}^n \langle U_{:,i}, \vec{x} \rangle U_{:,i}. \quad (33)$$

Since orthogonal matrices only rotate vectors, it is quite intuitive that the product of two orthogonal matrices yields another orthogonal matrix.

Lemma 1.20 (Product of orthogonal matrices). *If $U, V \in \mathbb{R}^{n \times n}$ are orthogonal matrices, then UV is also an orthogonal matrix.*

Proof.

$$(UV)^T (UV) = V^T U^T UV = I. \quad (34)$$

□

The following lemma proves that orthogonal matrices preserve the ℓ_2 norms of vectors.

Lemma 1.21. *Let $U \in \mathbb{R}^{n \times n}$ be an orthogonal matrix. For any vector $\vec{x} \in \mathbb{R}^n$,*

$$\|U\vec{x}\|_2 = \|\vec{x}\|_2. \quad (35)$$

Proof. By the definition of orthogonal matrix

$$\|U\vec{x}\|_2^2 = \vec{x}^T U^T U \vec{x} \quad (36)$$

$$= \vec{x}^T \vec{x} \quad (37)$$

$$= \|\vec{x}\|_2^2. \quad (38)$$

□

Matrices with orthonormal columns can also be used to construct orthogonal-projection matrices, which represent orthogonal projections onto a subspace.

Lemma 1.22 (Orthogonal-projection matrix). *Given a subspace $\mathcal{S} \subseteq \mathbb{R}^n$ of dimension d , the matrix*

$$P := UU^T, \quad (39)$$

where the columns of $U_{:1}, U_{:2}, \dots, U_{:d}$ are an orthonormal basis of \mathcal{S} , maps any vector \vec{x} to its orthogonal projection onto \mathcal{S} .

Proof. For any vector $\vec{x} \in \mathbb{R}^n$

$$P\vec{x} = UU^T\vec{x} \quad (40)$$

$$= \sum_{i=1}^d \langle U_{:i}, \vec{x} \rangle U_{:i} \quad (41)$$

$$= \mathcal{P}_{\mathcal{S}} \vec{x} \quad \text{by (64) in the lecture notes on vector spaces.} \quad (42)$$

□

2 Singular-value decomposition

In this section we introduce the singular-value decomposition, a fundamental tool for manipulating matrices, and describe several applications in data analysis.

2.1 Definition

Every real matrix has a singular-value decomposition.

Theorem 2.1. *Every rank r real matrix $A \in R^{m \times n}$, has a singular-value decomposition (SVD) of the form*

$$A = [\vec{u}_1 \ \vec{u}_2 \ \dots \ \vec{u}_r] \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \sigma_r \end{bmatrix} [\vec{v}_1^T \ \vec{v}_2^T \ \dots \ \vec{v}_r^T]^T \quad (43)$$

$$= USV^T, \quad (44)$$

where the singular values $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r$ are positive real numbers, the left singular vectors $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_r$ form an orthonormal set, and the right singular vectors $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_r$ also form an orthonormal set. The SVD is unique if all the singular values are different. If several singular values are the same, their left singular vectors can be replaced by any orthonormal basis of their span, and the same holds for the right singular vectors.

The SVD of an $m \times n$ matrix with $m \geq n$ can be computed in $\mathcal{O}(mn^2)$. We refer to any graduate linear algebra book for the proof of Theorem 2.1 and for the details on how to compute the SVD.

The SVD provides orthonormal bases for the column and row spaces of the matrix.

Lemma 2.2. *The left singular vectors are an orthonormal basis for the column space, whereas the right singular vectors are an orthonormal basis for the row space.*

Proof. We prove the statement for the column space, the proof for the row space is identical. All left singular vectors belong to the column space because $\vec{u}_i = A(\sigma_i^{-1}\vec{v}_i)$. In addition, every column of A is in their span because $A_{:i} = U(SV^T\vec{e}_i)$. Since they form an orthonormal set by Theorem 2.1, this completes the proof. \square

The SVD presented in Theorem 2.1 can be augmented so that the number of singular values equals $\min(m, n)$. The additional singular values are all equal to zero. Their corresponding left and right singular vectors are orthonormal sets of vectors in the orthogonal complements of the column and row space respectively. If the matrix is tall or square, the additional right singular vectors are a basis of the null space of the matrix.

Corollary 2.3 (Singular-value decomposition). *Every rank r real matrix $A \in R^{m \times n}$,*

where $m \geq n$, has a singular-value decomposition (SVD) of the form

$$A := [\underbrace{\vec{u}_1 \vec{u}_2 \cdots \vec{u}_r}_{\text{Basis of range}(A)} \quad \vec{u}_{r+1} \cdots \vec{u}_n] \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 & \cdots & 0 \\ & & \ddots & & & & \\ 0 & 0 & \cdots & \sigma_r & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ & & \ddots & & & & \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix} [\underbrace{\vec{v}_1 \vec{v}_2 \cdots \vec{v}_r}_{\text{Basis of row}(A)} \quad \underbrace{\vec{v}_{r+1} \cdots \vec{v}_n}_{\text{Basis of null}(A)}]^T, \quad (45)$$

where the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are positive real numbers, the left singular vectors $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_m$ form an orthonormal set in \mathbb{R}^m , and the right singular vectors $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n$ form an orthonormal basis for \mathbb{R}^n .

If the matrix is fat, we can define a similar augmentation, where the additional left singular vectors form an orthonormal basis of the orthogonal complement of the range.

By the definition of rank and Lemma 2.2 the rank of a matrix is equal to the number of nonzero singular values.

Corollary 2.4. *The rank of a matrix is equal to the number of nonzero singular values.*

This interpretation of the rank allows to define an alternative definition that is very useful in practice, since matrices are often full rank due to numerical error, even if their columns or rows are *almost* linearly dependent.

Definition 2.5 (Numerical rank). *Given a tolerance $\epsilon > 0$, the numerical rank of a matrix is the number of singular values that are greater than ϵ .*

The SVD decomposes the action of a matrix $A \in \mathbb{R}^{m \times n}$ on a vector $\vec{x} \in \mathbb{R}^n$ into three simple steps:

1. Rotation of \vec{x} to align the component of \vec{x} in the direction of the i th right singular vector \vec{v}_i with the i th axis:

$$V^T \vec{x} = \sum_{i=1}^n \langle \vec{v}_i, \vec{x} \rangle \vec{e}_i. \quad (46)$$

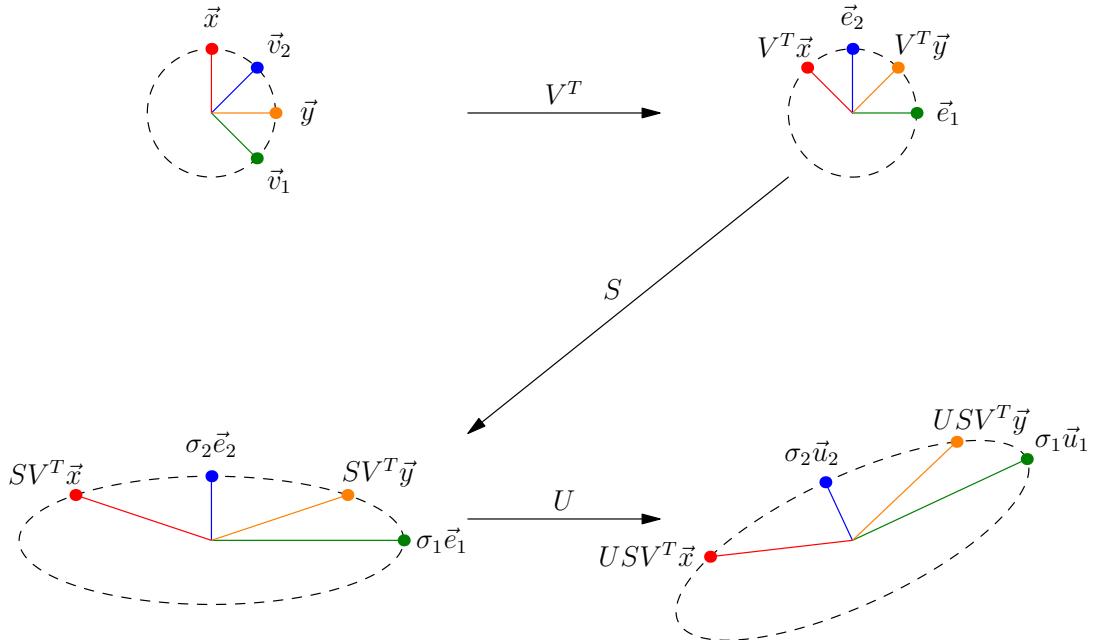
2. Scaling of each axis by the corresponding singular value

$$SV^T \vec{x} = \sum_{i=1}^n \sigma_i \langle \vec{v}_i, \vec{x} \rangle \vec{e}_i. \quad (47)$$

3. Rotation to align the i th axis with the i th left singular vector

$$USV^T \vec{x} = \sum_{i=1}^n \sigma_i \langle \vec{v}_i, \vec{x} \rangle \vec{u}_i. \quad (48)$$

(a) $\sigma_1 = 3, \sigma_2 = 1$.



(b) $\sigma_1 = 3, \sigma_2 = 0$.

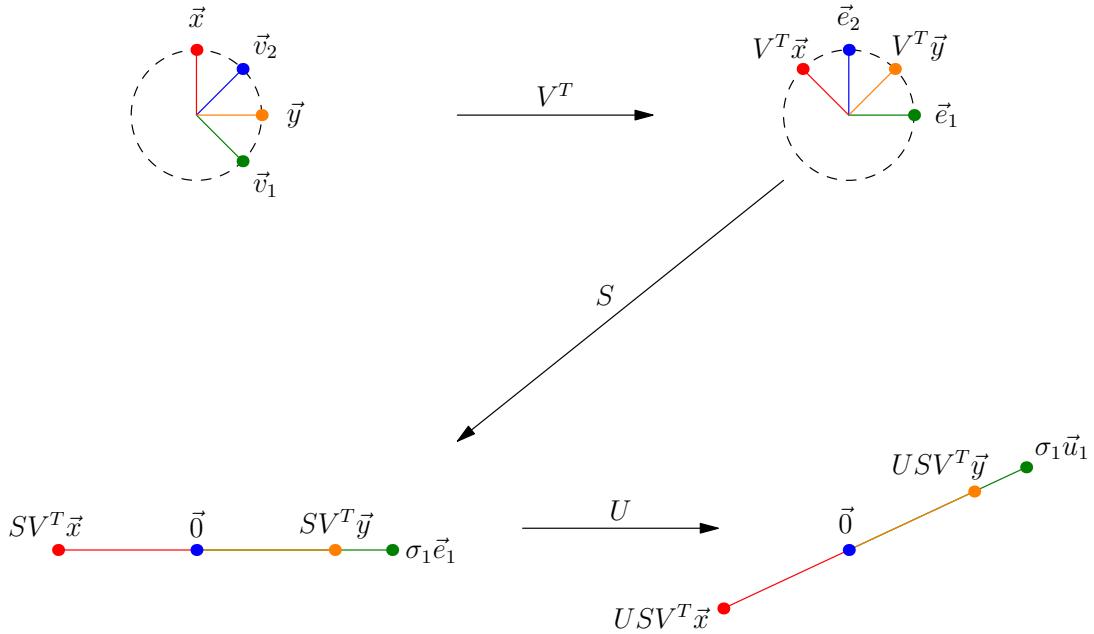


Figure 1: Any linear map can be decomposed into three steps: rotation to align the right singular vectors to the axes, scaling by the singular values and a final rotation to align the axes with the left singular vectors. In image (b) the second singular value is zero, so the linear map projects two-dimensional vectors onto a one-dimensional subspace.

Figure 1 illustrates this geometric analysis of the action of a linear map.

We end the section by showing that multiplying a matrix by an orthogonal matrix does not affect its singular values. This makes sense since it just modifies the rotation carried out by the left or right singular vectors.

Lemma 2.6. *For any matrix $A \in \mathbb{R}^{m \times n}$ and any orthogonal matrices $\tilde{U} \in \mathbb{R}^{m \times m}$ and $\tilde{V} \in \mathbb{R}^{n \times n}$ the singular values of $\tilde{U}A$ and $A\tilde{V}$ are the same as the singular values of A .*

Proof. Let $A = USV^T$ be the SVD of A . By Lemma 1.20 the matrices $\bar{U} := \tilde{U}U$ and $\bar{V}^T := V^T\tilde{V}$ are orthogonal matrices, so $\bar{U}SV^T$ and $US\bar{V}^T$ are valid SVDs for $\tilde{U}A$ and $A\tilde{V}$ respectively. The result follows by unicity of the SVD. \square

2.2 Optimal approximations via the SVD

In the previous section, we show that linear maps rotate vectors, scale them according to the singular values and then rotate them again. This means that the maximum scaling possible is equal to the maximum singular value and occurs in the direction of the right singular vector \vec{v}_1 . The following theorem makes this precise, showing that if we restrict our attention to the orthogonal complement of \vec{v}_1 , then the maximum scaling is the second singular value, due to the orthogonality of the singular vectors. In general, the direction of maximum scaling orthogonal to the first $i - 1$ left singular vectors is equal to the i th singular value and occurs in the direction of the i th singular vector.

Theorem 2.7. *For any matrix $A \in \mathbb{R}^{m \times n}$, with SVD given by (45), the singular values satisfy*

$$\sigma_1 = \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|A\vec{x}\|_2 \quad (49)$$

$$= \max_{\{\|\vec{y}\|_2=1 \mid \vec{y} \in \mathbb{R}^m\}} \|A^T\vec{y}\|_2, \quad (50)$$

$$\sigma_i = \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n, \vec{x} \perp \vec{v}_1, \dots, \vec{v}_{i-1}\}} \|A\vec{x}\|_2, \quad (51)$$

$$= \max_{\{\|\vec{y}\|_2=1 \mid \vec{y} \in \mathbb{R}^m, \vec{y} \perp \vec{u}_1, \dots, \vec{u}_{i-1}\}} \|A^T\vec{y}\|_2, \quad 2 \leq i \leq \min\{m, n\}, \quad (52)$$

the right singular vectors satisfy

$$\vec{v}_1 = \arg \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|A\vec{x}\|_2, \quad (53)$$

$$\vec{v}_i = \arg \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n, \vec{x} \perp \vec{v}_1, \dots, \vec{v}_{i-1}\}} \|A\vec{x}\|_2, \quad 2 \leq i \leq m, \quad (54)$$

and the left singular vectors satisfy

$$\vec{u}_1 = \arg \max_{\{\|\vec{y}\|_2=1 \mid \vec{y} \in \mathbb{R}^m\}} \|A^T \vec{y}\|_2, \quad (55)$$

$$\vec{u}_i = \arg \max_{\{\|\vec{y}\|_2=1 \mid \vec{y} \in \mathbb{R}^m, \vec{y} \perp \vec{u}_1, \dots, \vec{u}_{i-1}\}} \|A^T \vec{y}\|_2, \quad 2 \leq i \leq n. \quad (56)$$

Proof. Consider a vector $\vec{x} \in \mathbb{R}^n$ with unit ℓ_2 norm that is orthogonal to $\vec{v}_1, \dots, \vec{v}_{i-1}$, where $1 \leq i \leq n$ (if $i = 1$ then \vec{x} is just an arbitrary vector). We express \vec{x} in terms of the right singular vectors of A and a component that is orthogonal to their span

$$\vec{x} = \sum_{j=i}^n \alpha_j \vec{v}_j + \mathcal{P}_{\text{row}(A)^\perp} \vec{x} \quad (57)$$

where $1 = \|\vec{x}\|_2^2 \geq \sum_{j=i}^n \alpha_j^2$. By the ordering of the singular values in Theorem 2.1

$$\|A\vec{x}\|_2^2 = \left\langle \sum_{k=1}^n \sigma_k \langle \vec{v}_k, \vec{x} \rangle \vec{u}_k, \sum_{k=1}^n \sigma_k \langle \vec{v}_k, \vec{x} \rangle \vec{u}_k \right\rangle \quad \text{by (48)} \quad (58)$$

$$= \sum_{k=1}^n \sigma_k^2 \langle \vec{v}_k, \vec{x} \rangle^2 \quad \text{because } \vec{u}_1, \dots, \vec{u}_n \text{ are orthonormal} \quad (59)$$

$$= \sum_{k=1}^n \sigma_k^2 \left\langle \vec{v}_k, \sum_{j=i}^n \alpha_j \vec{v}_j + \mathcal{P}_{\text{row}(A)^\perp} \vec{x} \right\rangle^2 \quad (60)$$

$$= \sum_{j=i}^n \sigma_j^2 \alpha_j^2 \quad \text{because } \vec{v}_1, \dots, \vec{v}_n \text{ are orthonormal} \quad (61)$$

$$\leq \sigma_i^2 \sum_{j=i}^n \alpha_j^2 \quad \text{because } \sigma_i \geq \sigma_{i+1} \geq \dots \geq \sigma_n \quad (62)$$

$$\leq \sigma_i^2 \quad \text{by (57)}. \quad (63)$$

This establishes (49) and (51). To prove (53) and (54) we show that \vec{v}_i achieves the maximum

$$\|A\vec{v}_i\|_2^2 = \sum_{k=1}^n \sigma_k^2 \langle \vec{v}_k, \vec{v}_i \rangle^2 \quad (64)$$

$$= \sigma_i^2. \quad (65)$$

The same argument applied to A^T establishes (50), (55), (56) and (52). \square

Given a set of vectors, it is often of interest to determine whether they are oriented in particular directions of the ambient space. This can be quantified in terms of the ℓ_2 norms of their projections on low-dimensional subspaces. The SVD provides an optimal k -dimensional subspace in this sense for *any value of k* .

Theorem 2.8 (Optimal subspace for orthogonal projection). *For any matrix*

$$A := [\vec{a}_1 \quad \vec{a}_2 \quad \cdots \quad \vec{a}_n] \in \mathbb{R}^{m \times n}, \quad (66)$$

with SVD given by (45), we have

$$\sum_{i=1}^n \left\| \mathcal{P}_{\text{span}(\vec{u}_1, \vec{u}_2, \dots, \vec{u}_k)} \vec{a}_i \right\|_2^2 \geq \sum_{i=1}^n \left\| \mathcal{P}_{\mathcal{S}} \vec{a}_i \right\|_2^2, \quad (67)$$

for any subspace \mathcal{S} of dimension $k \leq \min\{m, n\}$.

Proof. Note that

$$\sum_{i=1}^n \left\| \mathcal{P}_{\text{span}(\vec{u}_1, \vec{u}_2, \dots, \vec{u}_k)} \vec{a}_i \right\|_2^2 = \sum_{i=1}^n \sum_{j=1}^k \langle \vec{u}_j, \vec{a}_i \rangle^2 \quad (68)$$

$$= \sum_{j=1}^k \left\| A^T \vec{u}_j \right\|_2^2. \quad (69)$$

We prove the result by induction on k . The base case $k = 1$ follows immediately from (55). To complete the proof we show that if the result is true for $k - 1 \geq 1$ (the induction hypothesis) then it also holds for k . Let \mathcal{S} be an arbitrary subspace of dimension k . The intersection of \mathcal{S} and the orthogonal complement to the span of $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_{k-1}$ contains a nonzero vector \vec{b} due to the following lemma.

Lemma 2.9 (Proof in Section 5.2). *In a vector space of dimension n , the intersection of two subspaces with dimensions d_1 and d_2 such that $d_1 + d_2 > n$ has dimension at least one.*

We choose an orthonormal basis $\vec{b}_1, \vec{b}_2, \dots, \vec{b}_k$ for \mathcal{S} such that $\vec{b}_k := \vec{b}$ is orthogonal to $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_{k-1}$ (we can construct such a basis by Gram-Schmidt, starting with \vec{b}). By the induction hypothesis,

$$\sum_{i=1}^{k-1} \left\| A^T \vec{u}_i \right\|_2^2 = \sum_{i=1}^n \left\| \mathcal{P}_{\text{span}(\vec{u}_1, \vec{u}_2, \dots, \vec{u}_{k-1})} \vec{a}_i \right\|_2^2 \quad (70)$$

$$\geq \sum_{i=1}^n \left\| \mathcal{P}_{\text{span}(\vec{b}_1, \vec{b}_2, \dots, \vec{b}_{k-1})} \vec{a}_i \right\|_2^2 \quad (71)$$

$$= \sum_{i=1}^{k-1} \left\| A^T \vec{b}_i \right\|_2^2. \quad (72)$$

By (56)

$$\left\| A^T \vec{u}_k \right\|_2^2 \geq \left\| A^T \vec{b}_k \right\|_2^2. \quad (73)$$

Combining (72) and (73) we conclude

$$\sum_{i=1}^n \left\| \mathcal{P}_{\text{span}(\vec{u}_1, \vec{u}_2, \dots, \vec{u}_k)} \vec{a}_i \right\|_2^2 = \sum_{i=1}^k \left\| A^T \vec{u}_i \right\|_2^2 \quad (74)$$

$$\geq \sum_{i=1}^k \left\| A^T \vec{b}_i \right\|_2^2 \quad (75)$$

$$= \sum_{i=1}^n \left\| \mathcal{P}_{\mathcal{S}} \vec{a}_i \right\|_2^2. \quad (76)$$

□

The SVD also allows to compute the optimal k -rank approximation to a matrix in Frobenius norm, for any value of k . For any matrix A , we denote by $A_{1:i, 1:j}$ to denote the $i \times j$ submatrix formed by taking the entries that are both in the first i rows and the first j columns. Similarly, we denote by $A_{:, i:j}$ the matrix formed by columns i to j .

Theorem 2.10 (Best rank- k approximation). *Let USV^T be the SVD of a matrix $A \in \mathbb{R}^{m \times n}$. The truncated SVD $U_{:, 1:k} S_{1:k, 1:k} V_{:, 1:k}^T$ is the best rank- k approximation of A in the sense that*

$$U_{:, 1:k} S_{1:k, 1:k} V_{:, 1:k}^T = \arg \min_{\{\tilde{A} \mid \text{rank}(\tilde{A})=k\}} \left\| A - \tilde{A} \right\|_{\text{F}}. \quad (77)$$

Proof. Let \tilde{A} be an arbitrary matrix in $\mathbb{R}^{m \times n}$ with $\text{rank}(\tilde{A}) = k$, and let $\tilde{U} \in \mathbb{R}^{m \times k}$ be a matrix with orthonormal columns such that $\text{col}(\tilde{U}) = \text{col}(\tilde{A})$. By Theorem 2.8,

$$\left\| U_{:, 1:k} U_{:, 1:k}^T A \right\|_{\text{F}}^2 = \sum_{i=1}^n \left\| \mathcal{P}_{\text{col}(U_{:, 1:k})} \vec{a}_i \right\|_2^2 \quad (78)$$

$$\geq \sum_{i=1}^n \left\| \mathcal{P}_{\text{col}(\tilde{U})} \vec{a}_i \right\|_2^2 \quad (79)$$

$$= \left\| \tilde{U} \tilde{U}^T A \right\|_{\text{F}}^2. \quad (80)$$

The column space of $A - \tilde{U} \tilde{U}^T A$ is orthogonal to the column space of \tilde{A} and \tilde{U} , so by Corollary 1.5

$$\left\| A - \tilde{A} \right\|_{\text{F}}^2 = \left\| A - \tilde{U} \tilde{U}^T A \right\|_{\text{F}}^2 + \left\| \tilde{A} - \tilde{U} \tilde{U}^T A \right\|_{\text{F}}^2 \quad (81)$$

$$\geq \left\| A - \tilde{U} \tilde{U}^T A \right\|_{\text{F}}^2 \quad (82)$$

$$= \|A\|_{\text{F}}^2 - \left\| \tilde{U} \tilde{U}^T A \right\|_{\text{F}}^2 \quad \text{also by Corollary 1.5} \quad (83)$$

$$\geq \|A\|_{\text{F}}^2 - \left\| U_{:, 1:k} U_{:, 1:k}^T A \right\|_{\text{F}}^2 \quad \text{by (80)} \quad (84)$$

$$= \left\| A - U_{:, 1:k} U_{:, 1:k}^T A \right\|_{\text{F}}^2 \quad \text{again by Corollary 1.5.} \quad (85)$$

□

2.3 Matrix norms

As we discussed in Lecture Notes 1, the inner-product norm in the vector space of matrices is the Frobenius norm. The following lemma establishes that the Frobenius norm of a matrix equals the ℓ_2 norm of its singular values.

Lemma 2.11. *For any matrix $A \in \mathbb{R}^{m \times n}$, with singular values $\sigma_1, \dots, \sigma_{\min\{m,n\}}$*

$$\|A\|_F = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2}. \quad (86)$$

Proof. Let us denote the SVD of A by USV^T ,

$$\|A\|_F^2 = \text{tr}(A^T A) \quad (87)$$

$$= \text{tr}(VSU^T USV^T) \quad \text{by Lemma 2.5 in Lecture Notes 1} \quad (88)$$

$$= \text{tr}(VSSV^T) \quad \text{because } U^T U = I \quad (89)$$

$$= \text{tr}(V^T V SS) \quad (90)$$

$$= \text{tr}(SS) \quad \text{because } V^T V = I. \quad (91)$$

□

The operator norm quantifies how much a linear map can scale a vector in ℓ_2 norm.

Definition 2.12 (Operator norm). *The operator norm of a linear map and of the corresponding matrix $A \in \mathbb{R}^{m \times n}$ is defined by*

$$\|A\| := \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|A \vec{x}\|_2. \quad (92)$$

By Theorem 2.7 (see equation (49)) the operator norm is equal to the ℓ_∞ norm of the singular values, i.e. the largest one, is also a norm.

Corollary 2.13. *For any matrix $A \in \mathbb{R}^{m \times n}$, with singular values $\sigma_1, \dots, \sigma_{\min\{m,n\}}$*

$$\|A\| := \sigma_1. \quad (93)$$

We end the section by defining an additional matrix norm, this time directly in term of the singular values.

Definition 2.14 (Nuclear norm). *The nuclear norm of a matrix $A \in \mathbb{R}^{m \times n}$ is equal to the ℓ_1 norm of its singular values $\sigma_1, \dots, \sigma_{\min\{m,n\}}$*

$$\|A\|_* := \sum_{i=1}^{\min\{m,n\}} \sigma_i. \quad (94)$$

Any matrix norm that is a function of the singular values of a matrix is preserved after multiplication by an orthogonal matrix. This is a direct corollary of Lemma 2.6.

Corollary 2.15. *For any matrix $A \in \mathbb{R}^{m \times n}$ and any orthogonal matrices $\tilde{U} \in \mathbb{R}^{m \times m}$ and $\tilde{V} \in \mathbb{R}^{n \times n}$ the operator, Frobenius and nuclear norm of $\tilde{U}A$ and $A\tilde{V}$ are the same as those of A .*

The following theorem is analogous to Hölder's inequality for vectors.

Theorem 2.16 (Proof in Section 5.3). *For any matrix $A \in \mathbb{R}^{m \times n}$,*

$$\|A\|_* = \sup_{\{\|B\| \leq 1 \mid B \in \mathbb{R}^{m \times n}\}} \langle A, B \rangle. \quad (95)$$

A direct consequence of the result is that the nuclear norm satisfies the triangle inequality. This implies that it is a norm, since it clearly satisfies the remaining properties.

Corollary 2.17. *For any $m \times n$ matrices A and B*

$$\|A + B\|_* \leq \|A\|_* + \|B\|_*. \quad (96)$$

Proof.

$$\|A + B\|_* = \sup_{\{\|C\| \leq 1 \mid C \in \mathbb{R}^{m \times n}\}} \langle A + B, C \rangle \quad (97)$$

$$\leq \sup_{\{\|C\| \leq 1 \mid C \in \mathbb{R}^{m \times n}\}} \langle A, C \rangle + \sup_{\{\|D\| \leq 1 \mid D \in \mathbb{R}^{m \times n}\}} \langle B, D \rangle \quad (98)$$

$$= \|A\|_* + \|B\|_*. \quad (99)$$

□

2.4 Denoising via low-rank matrix estimation

In this section we consider the problem of denoising a set of n m -dimensional signals $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n \in \mathbb{R}^m$. We model the noisy data as the sum between each signal and a noise vector

$$\vec{y}_i = \vec{x}_i + \vec{z}_i, \quad 1 \leq i \leq n. \quad (100)$$

Our first assumption is that the signals are similar, in the sense that they approximately span a low-dimensional subspace due to the correlations between them. If this is the case, then the matrix

$$X := [\vec{x}_1 \ \vec{x}_2 \ \cdots \ \vec{x}_n] \quad (101)$$

obtained by stacking the signals as columns is approximately low rank. Note that in contrast to the subspace-projection denoising method described in Lecture Notes 1, we do *not* assume that the subspace is known.

Our second assumption is that the noise vectors are independent from each other, so that the noise matrix

$$Z := [\vec{z}_1 \ \vec{z}_2 \ \cdots \ \vec{z}_n] \quad (102)$$

is full rank. If the noise is not too large with respect to the signals, under these assumptions a low-rank approximation to the data matrix

$$Y := [\vec{y}_1 \ \vec{y}_2 \ \cdots \ \vec{y}_n] \quad (103)$$

$$= X + Z \quad (104)$$

should mostly suppress the noise and extract the component corresponding to the signals. Theorem 2.10 establishes that the best rank- k approximation to a matrix in Frobenius norm is achieved by truncating the SVD, for any value of k .

Algorithm 2.18 (Denoising via SVD truncation). *Given n noisy data vectors $\vec{y}_1, \vec{y}_2, \dots, \vec{y}_n \in \mathbb{R}^m$, we denoise the data by*

1. *Stacking the vectors as the columns of a matrix $Y \in \mathbb{R}^{m \times n}$.*
2. *Computing the SVD of $Y = USV^T$.*
3. *Truncating the SVD to produce the low-rank estimate L*

$$L := U_{:,1:k} S_{1:k,1:k} V_{:,1:k}^T, \quad (105)$$

for a fixed value of $k \leq \min\{m, n\}$.

An important decision is what rank k to choose. Higher ranks yield more accurate approximations to the original signals than lower-rank approximations, but they do not suppress the noise component in the data as much. The following example illustrates this tradeoff.

Example 2.19 (Denoising of digit images). In this example we use the MNIST data set¹ to illustrate image denoising using SVD truncation. The signals consist of 6131 28×28 images of the number 3. The images are corrupted by noise sampled independently from a Gaussian distribution and scaled so that the signal-to-noise ratio (defined as the ratio between the ℓ_2 norms of the clean image and the noise) is 0.5 (there is more noise than signal!). Our assumption is that because of their similarities, the images interpreted as vectors in \mathbb{R}^{784} form a low-dimensional (but unknown subspace), whereas the noise is uncorrelated and therefore is not restricted to a subspace. This assumption holds: Figure 11 shows the singular values of matrix formed by stacking the clean images, the noisy images and the noise.

We center each noisy image by subtracting the average of all the noisy images. Subtracting the average is a common preprocessing step in low-rank approximations (see Figure 5 for a geometric justification). We then apply SVD truncation to obtain a low-rank estimate of the image matrix. The noisy average is then added back to the images to produce the final estimate of the images. Figure 3 shows the results for rank-10 and rank-40 estimates. The lower-rank estimate suppresses the noise more, but does not approximate the original signals as effectively. \triangle

¹Available at <http://yann.lecun.com/exdb/mnist/>

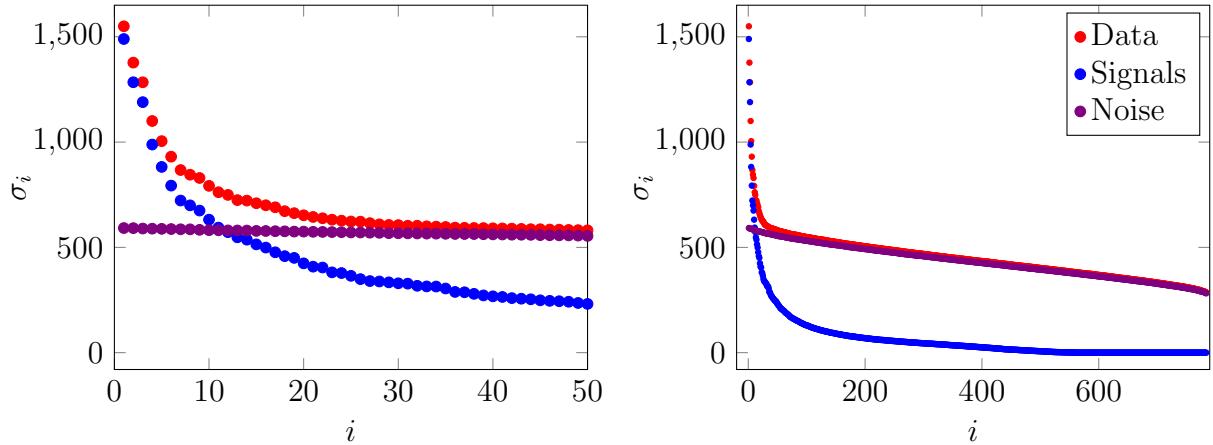


Figure 2: Plots of the singular values of the clean images, the noisy images and the noise in Example 2.19.

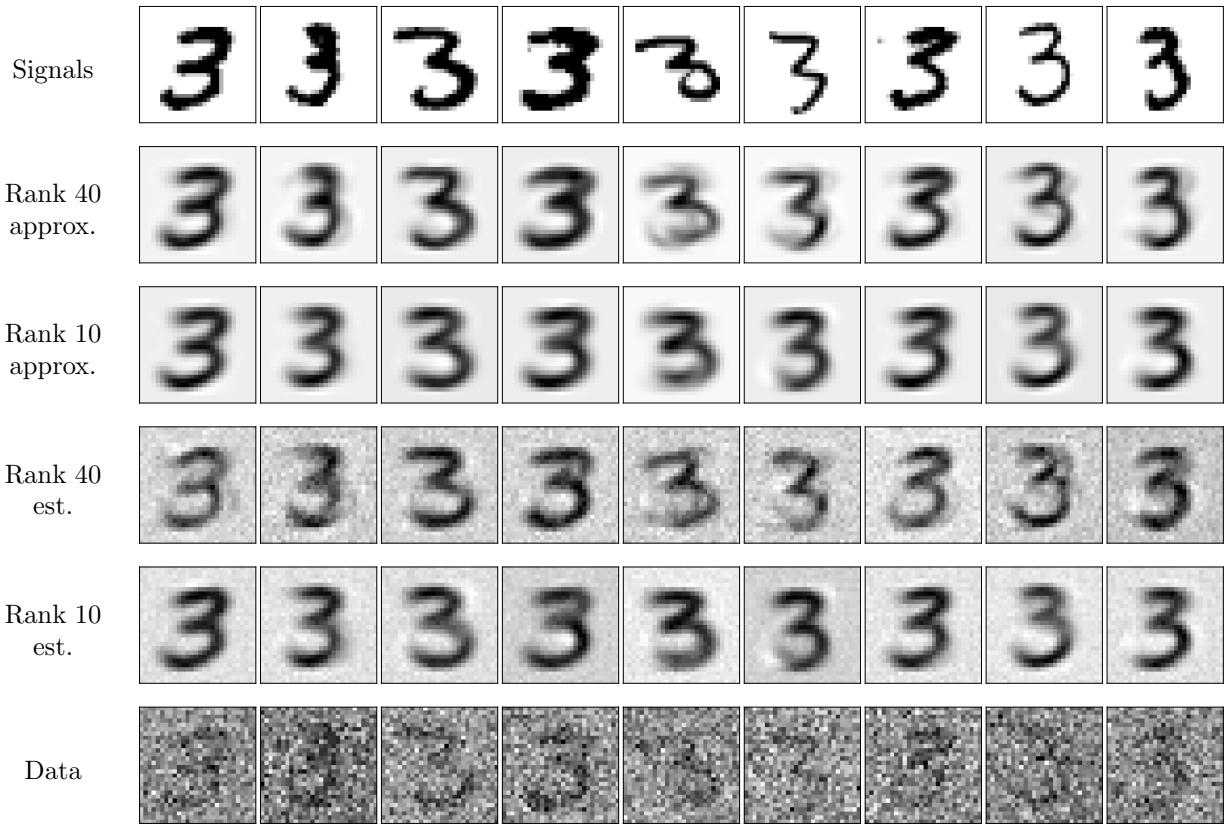


Figure 3: The images show 9 examples from the 6131 images used in Example 2.19. The top row shows the original clean images. The second and third rows show rank-40 and rank-10 approximations to the clean images. The fourth and fifth rows show the results of applying SVD truncation to obtain rank-40 and rank-10 estimates respectively. The sixth row shows the noisy data.

2.5 Collaborative filtering

The aim of collaborative filtering is to pool together information from many users to obtain a model of their behavior. To illustrate the use of low-rank models in this application we consider a toy example. Bob, Molly, Mary and Larry rate the following six movies from 1 to 5,

$$A := \begin{pmatrix} & \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ & 1 & 1 & 5 & 4 \\ & 2 & 1 & 4 & 5 \\ & 4 & 5 & 2 & 1 \\ & 5 & 4 & 2 & 1 \\ & 4 & 5 & 1 & 2 \\ & 1 & 2 & 5 & 5 \end{pmatrix} \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} \quad (106)$$

A common assumption in collaborative filtering is that there are people that have similar tastes and hence produce similar ratings, and that there are movies that are similar and hence elicit similar reactions. Interestingly, this tends to induce low-rank structure in the matrix of ratings. To uncover this low-rank structure, we first subtract the average rating

$$\mu := \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n A_{ij}, \quad (107)$$

from each entry in the matrix to obtain a centered matrix C and then compute its singular-value decomposition

$$A - \mu \vec{1} \vec{1}^T = USV^T = U \begin{bmatrix} 7.79 & 0 & 0 & 0 \\ 0 & 1.62 & 0 & 0 \\ 0 & 0 & 1.55 & 0 \\ 0 & 0 & 0 & 0.62 \end{bmatrix} V^T. \quad (108)$$

where $\vec{1} \in \mathbb{R}^4$ is a vector of ones. 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 indeed the case:

$$\mu \vec{1} \vec{1}^T + \sigma_1 \vec{u}_1 \vec{v}_1^T = \begin{pmatrix} & \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ & 1.34(1) & 1.19(1) & 4.66(5) & 4.81(4) \\ & 1.55(2) & 1.42(1) & 4.45(4) & 4.58(5) \\ & 4.45(4) & 4.58(5) & 1.55(2) & 1.42(1) \\ & 4.43(5) & 4.56(4) & 1.57(2) & 1.44(1) \\ & 4.43(4) & 4.56(5) & 1.57(1) & 1.44(2) \\ & 1.34(1) & 1.19(2) & 4.66(5) & 4.81(5) \end{pmatrix} \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} \quad (109)$$

For ease of comparison the values of A are shown in brackets. The first left singular vector is equal to

$$\vec{u}_1 := \begin{pmatrix} \text{D. Knight} & \text{Spiderman 3} & \text{Love Act.} & \text{B.J.'s Diary} & \text{P. Woman} & \text{Superman 2} \\ -0.45 & -0.39 & 0.39 & 0.39 & 0.39 & -0.45 \end{pmatrix}.$$

This vector allows us to cluster the movies: movies with negative entries are similar (in this case they correspond to action movies) and movies with positive entries are similar (in this case they are romantic movies).

The first right singular vector is equal to

$$\vec{v}_1 = \begin{pmatrix} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ 0.48 & 0.52 & -0.48 & -0.52 \end{pmatrix}. \quad (110)$$

This vector allows to cluster the users: negative entries indicate users that like action movies but hate romantic movies (Bob and Molly), whereas positive entries indicate the contrary (Mary and Larry).

For larger data sets, the model generalizes to a rank- k approximation, which approximates each ranking by a sum of k terms

$$\text{rating (movie } i, \text{ user } j) = \sum_{l=1}^k \sigma_l \vec{u}_l [i] \vec{v}_l [j]. \quad (111)$$

The singular vectors cluster users and movies in different ways, whereas the singular values weight the importance of the different factors.

3 Principal component analysis

In Lecture Notes 1 we introduced the sample variance of a set of one-dimensional data, which measures the variation of measurements in a one-dimensional data set, as well as the sample covariance, which measures the joint fluctuations of two features. We now consider data sets where each example contains m features, and can therefore be interpreted as a vector in an m -dimensional ambient space. We are interested in analyzing the variation of the data in different directions of this space.

3.1 Sample covariance matrix

The sample covariance matrix of a data set contains the pairwise sample covariance between every pair of features in a data set. If the data are sampled from a multivariate distribution, then the sample covariance matrix can be interpreted as an estimate of the covariance matrix (see Section 3.3).

Definition 3.1 (Sample covariance matrix). Let $\{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$ be a set of m -dimensional real-valued data vectors, where each dimension corresponds to a different feature. The sample covariance matrix of these vectors is the $m \times m$ matrix

$$\Sigma(\vec{x}_1, \dots, \vec{x}_n) := \frac{1}{n-1} \sum_{i=1}^n (\vec{x}_i - \text{av}(\vec{x}_1, \dots, \vec{x}_n)) (\vec{x}_i - \text{av}(\vec{x}_1, \dots, \vec{x}_n))^T, \quad (112)$$

where the center or average is defined as

$$\text{av}(\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n) := \frac{1}{n} \sum_{i=1}^n \vec{x}_i \quad (113)$$

contains the sample mean of each feature. The (i, j) entry of the covariance matrix, where $1 \leq i, j \leq d$, is given by

$$\Sigma(\vec{x}_1, \dots, \vec{x}_n)_{ij} = \begin{cases} \text{var}(\vec{x}_1[i], \dots, \vec{x}_n[i]) & \text{if } i = j, \\ \text{cov}((\vec{x}_1[i], \vec{x}_1[j]), \dots, (\vec{x}_n[i], \vec{x}_n[j])) & \text{if } i \neq j. \end{cases} \quad (114)$$

In order to characterize the variation of a multidimensional data set around its center, we consider its variation in different directions. The average variation of the data in a certain direction is quantified by the sample variance of the projections of the data onto that direction. Let $\vec{d} \in \mathbb{R}^m$ be a unit-norm vector aligned with a direction of interest, the sample variance of the data set in the direction of \vec{d} is given by

$$\text{var}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n) = \frac{1}{n-1} \sum_{i=1}^n \left(\vec{d}^T \vec{x}_i - \text{av}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n) \right)^2 \quad (115)$$

$$= \frac{1}{n-1} \sum_{i=1}^n \left(\vec{d}^T (\vec{x}_i - \text{av}(\vec{x}_1, \dots, \vec{x}_n)) \right)^2 \quad (116)$$

$$= \vec{d}^T \left(\frac{1}{n-1} \sum_{i=1}^n (\vec{x}_i - \text{av}(\vec{x}_1, \dots, \vec{x}_n)) (\vec{x}_i - \text{av}(\vec{x}_1, \dots, \vec{x}_n))^T \right) \vec{d} \\ = \vec{d}^T \Sigma(\vec{x}_1, \dots, \vec{x}_n) \vec{d}. \quad (117)$$

Using the sample covariance matrix we can express the variation in every direction! This is a deterministic analog of the fact that the covariance matrix of a random vector encodes its variance in every direction.

3.2 Principal component analysis

Principal-component analysis is a popular tool for data analysis, which consists of computing the singular-value decomposition of a set of vectors grouped as the columns of a matrix.

$$\sigma_1/\sqrt{n-1} = 0.705, \quad \sigma_2/\sqrt{n-1} = 0.690$$

$$\sigma_1/\sqrt{n-1} = 0.983, \quad \sigma_2/\sqrt{n-1} = 0.356$$

$$\sigma_1/\sqrt{n-1} = 1.349, \quad \sigma_2/\sqrt{n-1} = 0.144$$

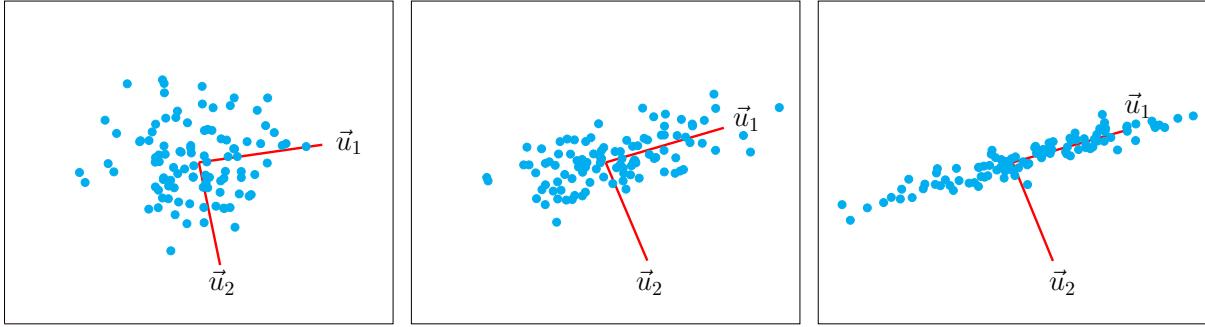


Figure 4: PCA of a dataset with $n = 100$ 2D vectors with different configurations. The two first singular values reflect how much energy is preserved by projecting onto the two first principal directions.

Algorithm 3.2 (Principal component analysis). *Given n data vectors $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n \in \mathbb{R}^d$, we apply the following steps.*

1. Center the data,

$$\vec{c}_i = \vec{x}_i - \text{av}(\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n), \quad 1 \leq i \leq n. \quad (118)$$

2. Group the centered data as columns of a matrix

$$C = [\vec{c}_1 \quad \vec{c}_2 \quad \cdots \quad \vec{c}_n]. \quad (119)$$

3. Compute the SVD of C . The left singular vectors are the principal directions. The principal values are the coefficients of the centered vectors when expressed in the basis of principal directions.

The sample covariance matrix can be expressed in terms of the centered data matrix C

$$\Sigma(\vec{x}_1, \dots, \vec{x}_n) = \frac{1}{n-1} C C^T. \quad (120)$$

This implies that by Theorem 2.7 the principal directions reveal the directions of maximum variation of the data.

Corollary 3.3. *Let $\vec{u}_1, \dots, \vec{u}_k$ be the $k \leq \min\{m, n\}$ first principal directions obtained by applying Algorithm 3.2 to a set of vectors $\vec{x}_1, \dots, \vec{x}_n \in \mathbb{R}^m$. Then the principal directions satisfy*

$$\vec{u}_1 = \arg \max_{\{\|\vec{d}\|_2=1 \mid \vec{d} \in \mathbb{R}^n\}} \text{var}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n), \quad (121)$$

$$\vec{u}_i = \arg \max_{\{\|\vec{d}\|_2=1 \mid \vec{d} \in \mathbb{R}^n, \vec{d} \perp \vec{u}_1, \dots, \vec{u}_{i-1}\}} \text{var}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n), \quad 2 \leq i \leq k, \quad (122)$$

$$\begin{aligned}\sigma_1/\sqrt{n-1} &= 5.077 \\ \sigma_2/\sqrt{n-1} &= 0.889\end{aligned}$$

$$\begin{aligned}\sigma_1/\sqrt{n-1} &= 1.261 \\ \sigma_2/\sqrt{n-1} &= 0.139\end{aligned}$$

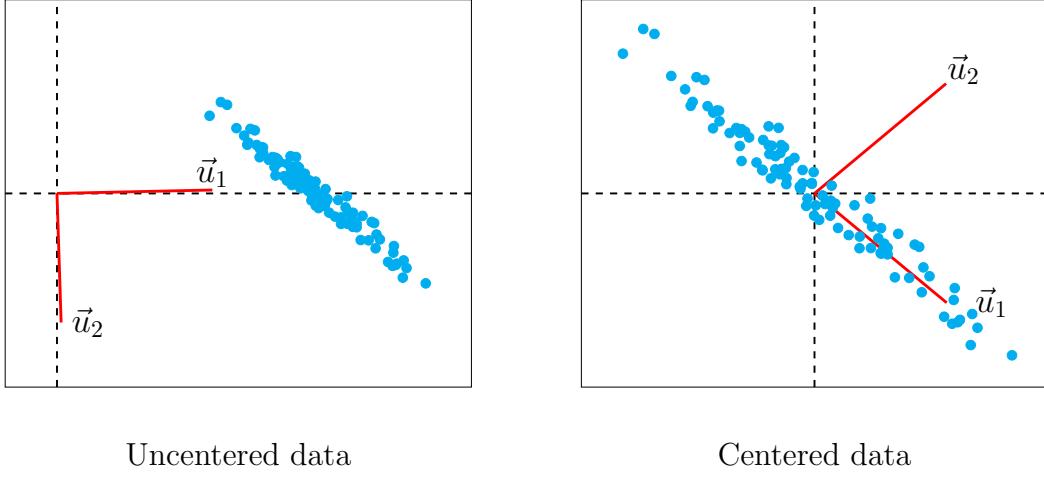


Figure 5: PCA applied to $n = 100$ 2D data points. On the left the data are not centered. As a result the dominant principal direction \vec{u}_1 lies in the direction of the mean of the data and PCA does not reflect the actual structure. Once we center, \vec{u}_1 becomes aligned with the direction of maximal variation.

and the associated singular values satisfy

$$\frac{\sigma_1}{\sqrt{n-1}} = \max_{\{\|\vec{d}\|_2=1 \mid \vec{d} \in \mathbb{R}^n\}} \text{std}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n), \quad (123)$$

$$\frac{\sigma_i}{\sqrt{n-1}} = \max_{\{\|\vec{d}\|_2=1 \mid \vec{d} \in \mathbb{R}^n, \vec{d} \perp \vec{u}_1, \dots, \vec{u}_{i-1}\}} \text{std}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n), \quad 2 \leq i \leq k. \quad (124)$$

Proof. For any vector \vec{d}

$$\text{var}(\vec{d}^T \vec{x}_1, \dots, \vec{d}^T \vec{x}_n) = \vec{d}^T \Sigma(\vec{x}_1, \dots, \vec{x}_n) \vec{d} \quad (125)$$

$$= \frac{1}{n-1} \vec{d}^T C C^T \vec{d} \quad (126)$$

$$= \frac{1}{n-1} \left\| C^T \vec{d} \right\|_2^2, \quad (127)$$

so the result follows from Theorem 2.7 applied to C . \square

In words, \vec{u}_1 is the direction of maximum variation, \vec{u}_2 is the direction of maximum variation orthogonal to \vec{u}_1 , and in general \vec{u}_i is the direction of maximum variation that is orthogonal to $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_{i-1}$. Figure 4 shows the principal directions for several 2D examples.

Figure 5 illustrates the importance of centering, i.e., subtracting the sample mean of each feature, before applying PCA. Theorem 2.7 still holds if the data are not centered.

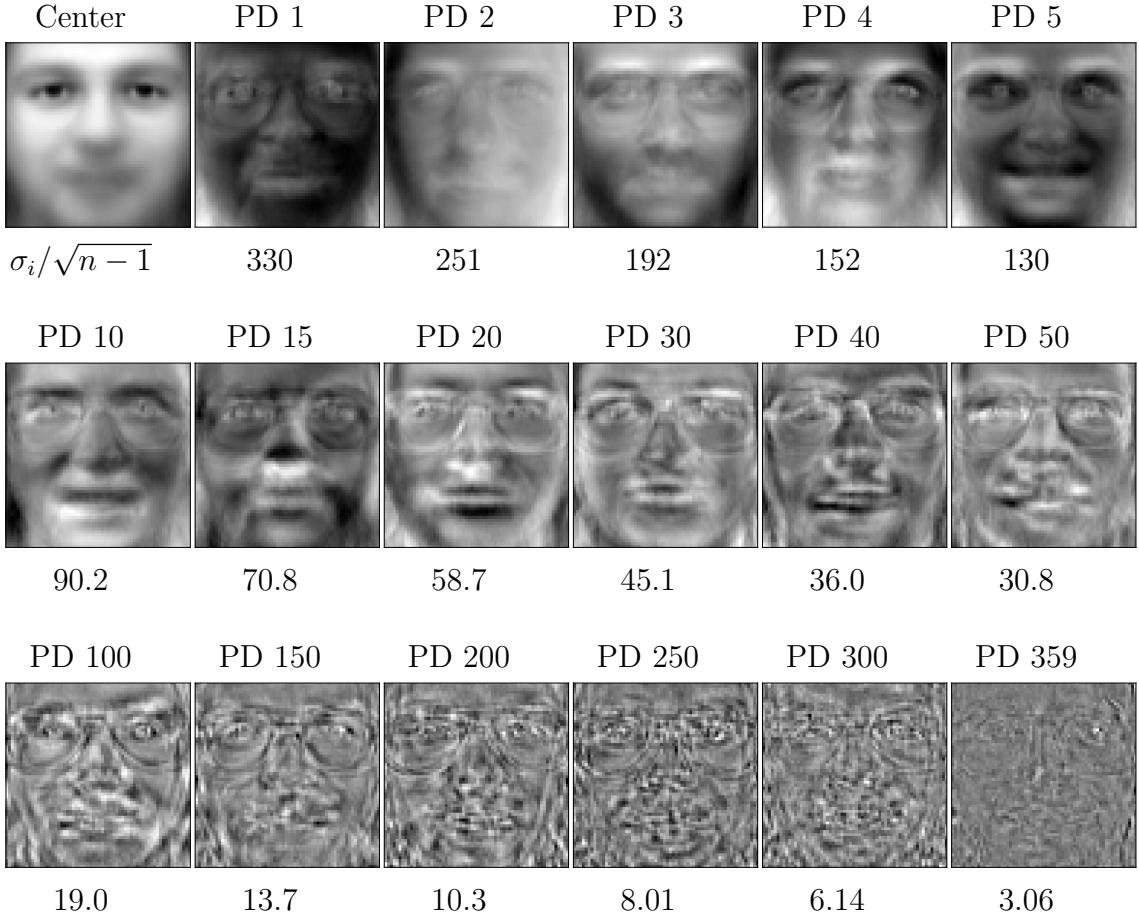


Figure 6: Average and principal directions (PD) of the faces data set in Example 3.4, along with their associated singular values.

However, the norm of the projection onto a certain direction no longer reflects the variation of the data. In fact, if the data are concentrated around a point that is far from the origin, the first principal direction tends to be aligned with that point. This makes sense as projecting onto that direction captures more energy. As a result, the principal directions do not reflect the directions of maximum variation *within* the cloud of data. Centering the data set before applying PCA solves the issue.

In the following example, we apply PCA to a set of images.

Example 3.4 (PCA of faces). In this example we consider the Olivetti Faces data set, which we described in Lecture Notes 1. We apply Algorithm 3.2 to a data set of 400 64×64 images taken from 40 different subjects (10 per subject). We vectorize each image so that each pixel is interpreted as a different feature. Figure 6 shows the center of the data and several principal directions, together with their associated singular values. The principal directions corresponding to the larger singular values seem to capture low-resolution structure, whereas the ones corresponding to the smallest singular values incorporate more intricate details.

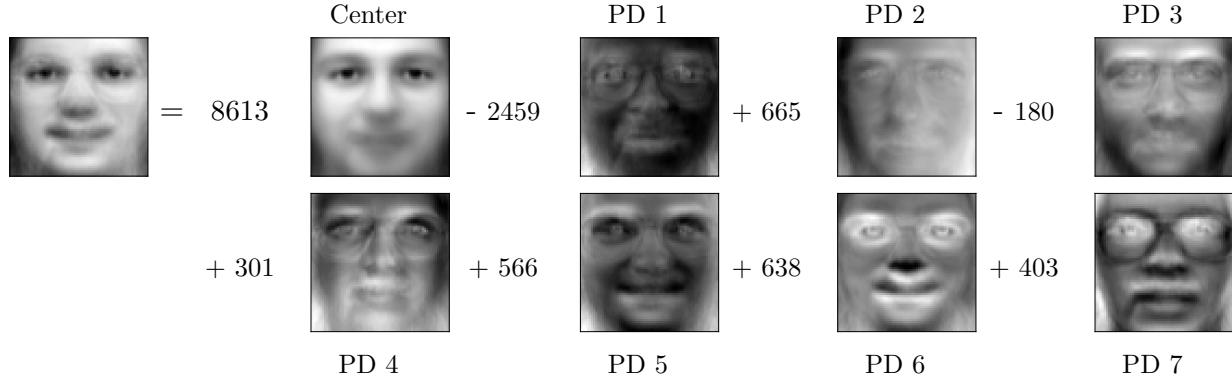


Figure 7: Projection of one of the faces \vec{x} onto the first 7 principal directions and the corresponding decomposition into the 7 first principal components.

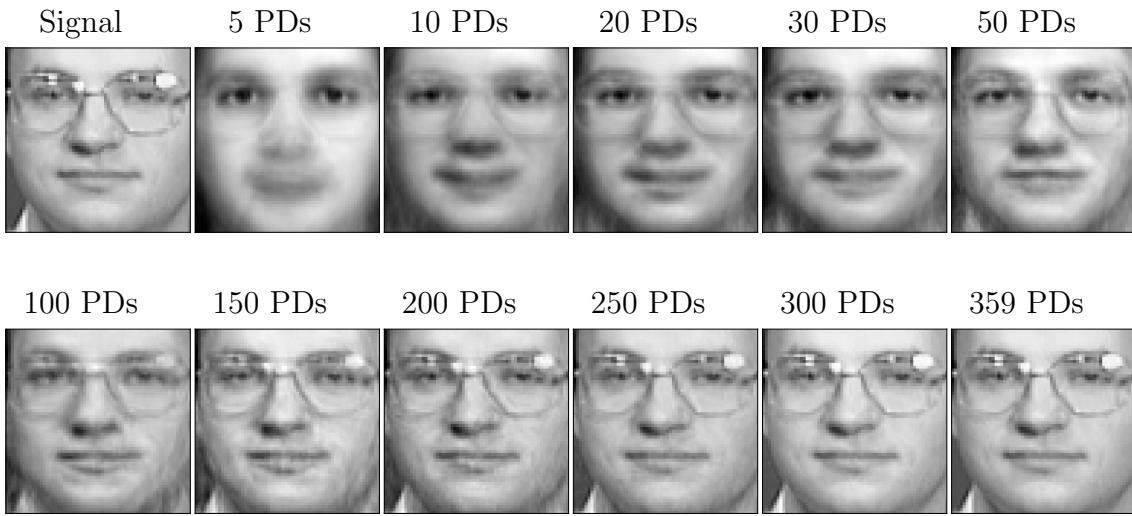


Figure 8: Projection of a face on different numbers of principal directions.

Figure 7 shows the projection of one of the faces onto the first 7 principal directions and the corresponding decomposition into its 7 first principal components. Figure 8 shows the projection of the same face onto increasing numbers of principal directions. As suggested by the visualization of the principal directions in Figure 6, the lower-dimensional projections produce blurry images.

△

3.3 Probabilistic interpretation

To provide a probabilistic interpretation of PCA, we first review some background on covariance matrices. The covariance matrix of a random vector captures the interaction between the components of the vector. It contains the variance of each component in the

diagonal and the covariances between different components in the off diagonals.

Definition 3.5. *The covariance matrix of a random vector \vec{x} is defined as*

$$\Sigma_{\vec{x}} := \begin{bmatrix} \text{Var}(\vec{x}[1]) & \text{Cov}(\vec{x}[1], \vec{x}[2]) & \cdots & \text{Cov}(\vec{x}[1], \vec{x}[n]) \\ \text{Cov}(\vec{x}[2], \vec{x}[1]) & \text{Var}(\vec{x}[2]) & \cdots & \text{Cov}(\vec{x}[2], \vec{x}[n]) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(\vec{x}[n], \vec{x}[1]) & \text{Cov}(\vec{x}[n], \vec{x}[2]) & \cdots & \text{Var}(\vec{x}[n]) \end{bmatrix} \quad (128)$$

$$= E(\vec{x}\vec{x}^T) - E(\vec{x})E(\vec{x})^T. \quad (129)$$

Note that if all the entries of a vector are uncorrelated, then its covariance matrix is diagonal. Using linearity of expectation, we obtain a simple expression for the covariance matrix of the linear transformation of a random vector.

Theorem 3.6 (Covariance matrix after a linear transformation). *Let \vec{x} be a random vector of dimension n with covariance matrix Σ . For any matrix $A \in \mathbb{R}^{m \times n}$,*

$$\Sigma_{A\vec{x}} = A\Sigma_{\vec{x}}A^T. \quad (130)$$

Proof. By linearity of expectation

$$\Sigma_{A\vec{x}} = E((A\vec{x})(A\vec{x})^T) - E(A\vec{x})E(A\vec{x})^T \quad (131)$$

$$= A(E(\vec{x}\vec{x}^T) - E(\vec{x})E(\vec{x})^T)A^T \quad (132)$$

$$= A\Sigma_{\vec{x}}A^T. \quad (133)$$

□

An immediate corollary of this result is that we can easily decode the variance of the random vector *in any direction* from the covariance matrix. Mathematically, the variance of the random vector in the direction of a unit vector \vec{v} is equal to the variance of its projection onto \vec{v} .

Corollary 3.7. *Let \vec{v} be a unit- ℓ_2 -norm vector,*

$$\text{Var}(\vec{v}^T \vec{x}) = \vec{v}^T \Sigma_{\vec{x}} \vec{v}. \quad (134)$$

Consider the SVD of the covariance matrix of an n -dimensional random vector X

$$\Sigma_{\vec{x}} = U\Lambda U^T \quad (135)$$

$$= [\vec{u}_1 \ \vec{u}_2 \ \cdots \ \vec{u}_n] \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \end{bmatrix} [\vec{u}_1 \ \vec{u}_2 \ \cdots \ \vec{u}_n]^T. \quad (136)$$

Covariance matrices are symmetric by definition, so by Theorem 4.3 the eigenvectors $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n$ can be chosen to be orthogonal. These singular vectors and singular values completely characterize the variance of the random vector in different directions. The theorem is a direct consequence of Corollary 3.7 and Theorem 2.7.

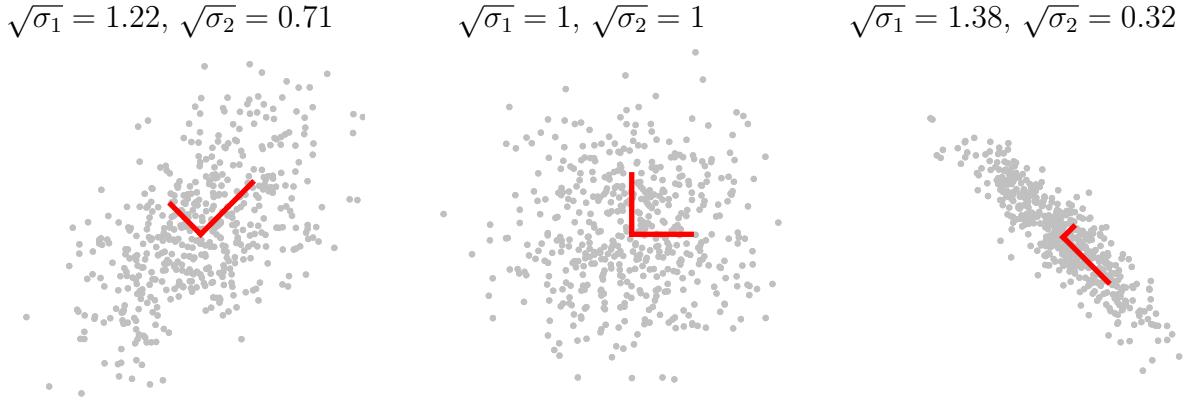


Figure 9: Samples from bivariate Gaussian random vectors with different covariance matrices are shown in gray. The eigenvectors of the covariance matrices are plotted in red. Each is scaled by the square root of the corresponding singular value σ_1 or σ_2 .

Theorem 3.8. Let \vec{x} be a random vector of dimension n with covariance matrix $\Sigma_{\vec{x}}$. The SVD of $\Sigma_{\vec{x}}$ given by (136) satisfies

$$\sigma_1 = \max_{\|\vec{v}\|_2=1} \text{Var}(\vec{v}^T \vec{x}), \quad (137)$$

$$\vec{u}_1 = \arg \max_{\|\vec{v}\|_2=1} \text{Var}(\vec{v}^T \vec{x}), \quad (138)$$

$$\sigma_k = \max_{\|\vec{v}\|_2=1, \vec{v} \perp \vec{u}_1, \dots, \vec{u}_{k-1}} \text{Var}(\vec{v}^T \vec{x}), \quad (139)$$

$$\vec{u}_k = \arg \max_{\|\vec{v}\|_2=1, \vec{v} \perp \vec{u}_1, \dots, \vec{u}_{k-1}} \text{Var}(\vec{v}^T \vec{x}). \quad (140)$$

In words, \vec{u}_1 is the *direction of maximum variance*. The second singular vector \vec{u}_2 is the direction of maximum variation that is orthogonal to \vec{u}_1 . In general, the eigenvector \vec{u}_k reveals the direction of maximum variation that is orthogonal to $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_{k-1}$. Finally, \vec{u}_n is the direction of minimum variance. Figure 9 illustrates this with an example, where $n = 2$.

The sample variance and covariance are consistent estimators of the variance and covariance respectively, under certain assumptions on the higher moments of the underlying distributions. This provides an intuitive interpretation for principal component analysis under the assumption that the data are realizations of an iid sequence of random vectors: the principal components approximate the eigenvectors of the true covariance matrix, and hence the directions of maximum variance of the multidimensional distribution. Figure 10 illustrates this with a numerical example, where the principal directions indeed converge to the singular vectors as the number of data increases.

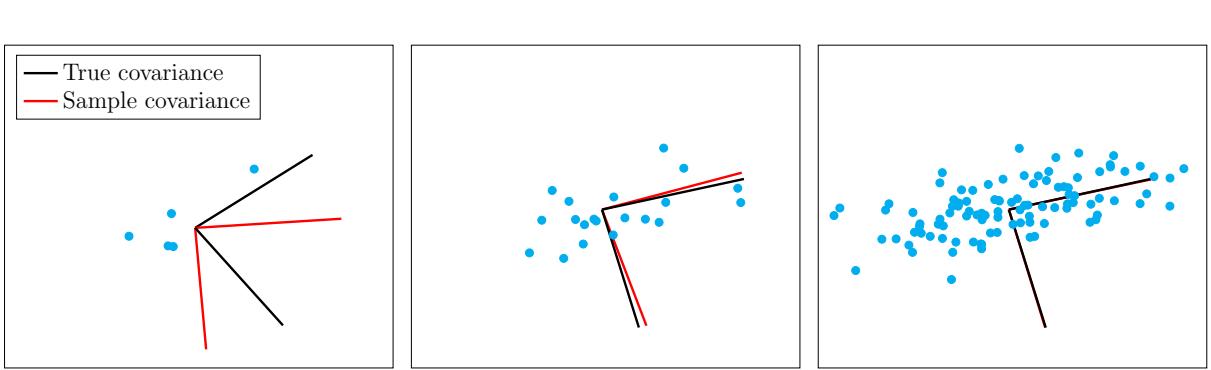


Figure 10: Principal directions of n samples from a bivariate Gaussian distribution (red) compared to the eigenvectors of the covariance matrix of the distribution (black).

3.4 Dimensionality reduction via PCA

Data containing a large number of features can be difficult to analyze or process. Dimensionality reduction is a useful preprocessing step for many data-analysis tasks, which consists of representing the data with a smaller number of variables. For data modeled as vectors in an ambient space \mathbb{R}^m where each dimension corresponds to a feature, this can be achieved by projecting the vectors onto a lower-dimensional space \mathbb{R}^k , where $k < m$. If the projection is orthogonal, the new representation can be computed using an orthogonal basis for the lower-dimensional subspace $\vec{b}_1, \vec{b}_2, \dots, \vec{b}_k$: each data vector $\vec{x} \in \mathbb{R}^m$ is described using the coefficients of its representation in the basis: $\langle \vec{b}_1, \vec{x} \rangle, \langle \vec{b}_2, \vec{x} \rangle, \dots, \langle \vec{b}_k, \vec{x} \rangle$.

Given a data set of n vectors $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n \in \mathbb{R}^m$, the first k principal directions span the subspace that preserves the most energy (measured in ℓ_2 norm) in the centered data among all possible k -dimensional orthogonal projections by Theorem 2.8. This motivates the application of PCA for dimensionality reduction.

Example 3.9 (Nearest neighbors in principal-component space). The nearest neighbors algorithm for classification (Algorithm 4.2 in Lecture Notes 1) requires computing n distances in an m -dimensional space (where m is the number of features) to classify each new example. The computational cost is $\mathcal{O}(nm)$, so if we need to classify p points the total cost is $\mathcal{O}(nmp)$. If we project each of the points onto a lower-dimensional space k computed via PCA before classifying them, then the computational cost is:

- $\mathcal{O}(mn \min\{m, n\})$ to compute the principal directions from the training data.
- kmn operations to project the training data onto the first k principal directions.
- kmp operations to project each point in the test set onto the first k principal directions.
- knp to perform nearest-neighbor classification in the lower-dimensional space.

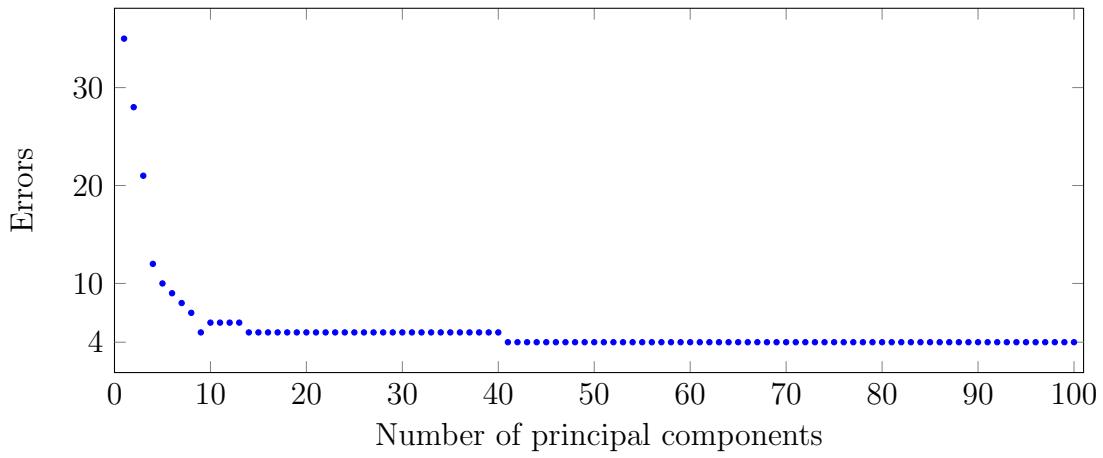


Figure 11: Errors for nearest-neighbor classification combined with PCA-based dimensionality reduction for different dimensions.



Figure 12: Results of nearest-neighbor classification combined with PCA-based dimensionality reduction of order 41 for four of the people in Example 3.9. The assignments of the first three examples are correct, but the fourth is wrong.

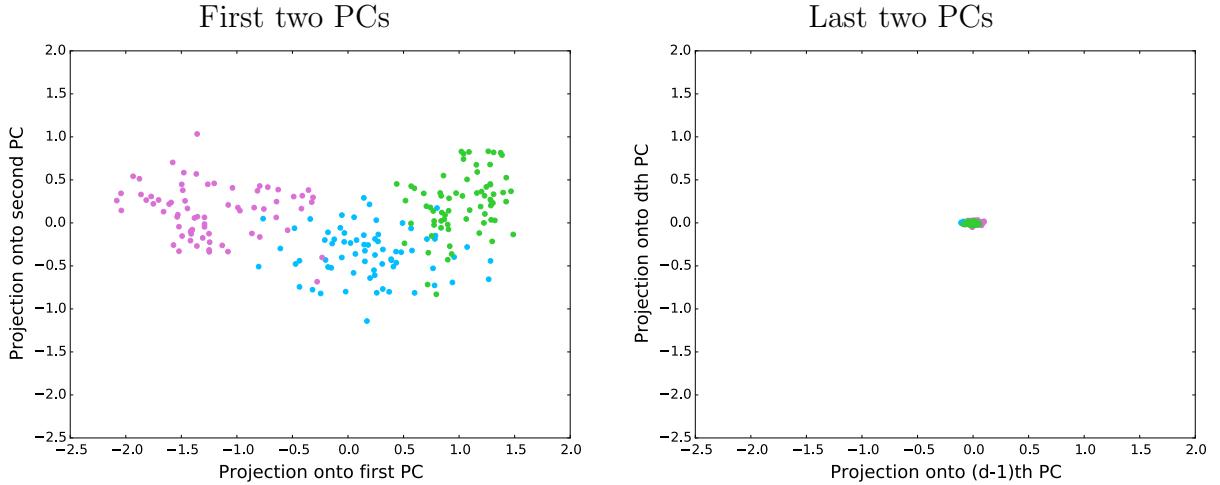


Figure 13: Projection of 7-dimensional vectors describing different wheat seeds onto the first two (left) and the last two (right) principal dimensions of the data set. Each color represents a variety of wheat.

If we have to classify a large number of points (i.e. $p \gg \max\{m, n\}$) the computational cost is reduced by operating in the lower-dimensional space.

Figure 11 shows the accuracy of the algorithm on the same data as Example 4.3 in Lecture Notes 1. The accuracy increases with the dimension at which the algorithm operates. This is not necessarily always the case because projections may actually be helpful for tasks such as classification (for example, factoring out small shifts and deformations). The same precision as in the ambient dimension (4 errors out of 40 test images) is achieved using just $k = 41$ principal components (in this example $n = 360$ and $m = 4096$). Figure 12 shows some examples of the projected data represented in the original m -dimensional space along with their nearest neighbors in the k -dimensional space. \triangle

Example 3.10 (Dimensionality reduction for visualization). Dimensionality reduction is often useful for visualization. The objective is to project the data onto 2D or 3D in a way that preserves its structure as much as possible. In this example, we consider a data set where each data point corresponds to a seed with seven features: area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient and length of kernel groove. The seeds belong to three different varieties of wheat: Kama, Rosa and Canadian.² To visualize the data in 2D, we project each point onto the two first principal dimensions of the data set.

Figure 13 shows the projection of the data onto the first two and the last two principal directions. In the latter case, there is almost no discernible variation. As predicted by our theoretical analysis of PCA, the structure in the data is much better conserved by the two first directions, which allow to clearly visualize the difference between the three types of seeds. Note however that projection onto the first principal directions only ensures

²The data can be found at <https://archive.ics.uci.edu/ml/datasets/seeds>.

$$\vec{x}_1, \dots, \vec{x}_n \quad U^T \vec{x}_1, \dots, U^T \vec{x}_n \quad S^{-1} U^T \vec{x}_1, \dots, S^{-1} U^T \vec{x}_n$$

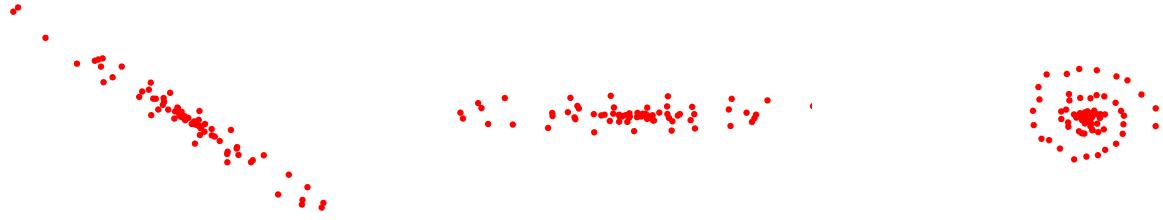


Figure 14: Effect of whitening a set of data. The original data are dominated by a linear skew (left). Applying U^T aligns the axes with the eigenvectors of the sample covariance matrix (center). Finally, S^{-1} reweights the data along those axes so that they have the same average variation, revealing the nonlinear structure that was obscured by the linear skew (right).

that we preserve as much variation as possible, but it does not necessarily preserve useful features for tasks such as clustering or classification. \triangle

3.5 Whitening

The principal directions in a data set do not necessarily capture the most useful features for certain tasks. For instance, in the case of the faces data set in Example 3.4 the principal directions correspond to low-resolution images, so that the corresponding principal components capture low-resolution features. These features do not include important information contained in fine-scale details, which could be useful for tasks such as classification. Whitening is a preprocessing technique that reweights the principal components of every vector so every principal dimension has the same contribution.

Algorithm 3.11 (Whitening). *Given n data vectors $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n \in \mathbb{R}^d$, we apply the following steps.*

1. *Center the data,*

$$\vec{c}_i = \vec{x}_i - \text{av}(\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n), \quad 1 \leq i \leq n. \quad (141)$$

2. *Group the centered data as columns of a matrix*

$$C = [\vec{c}_1 \quad \vec{c}_2 \quad \cdots \quad \vec{c}_n]. \quad (142)$$

3. *Compute the SVD of $C = USV^T$.*
4. *Whiten each centered vector by applying the linear map $US^{-1}U^T$*

$$\vec{w}_i := US^{-1}U^T \vec{c}_i. \quad (143)$$

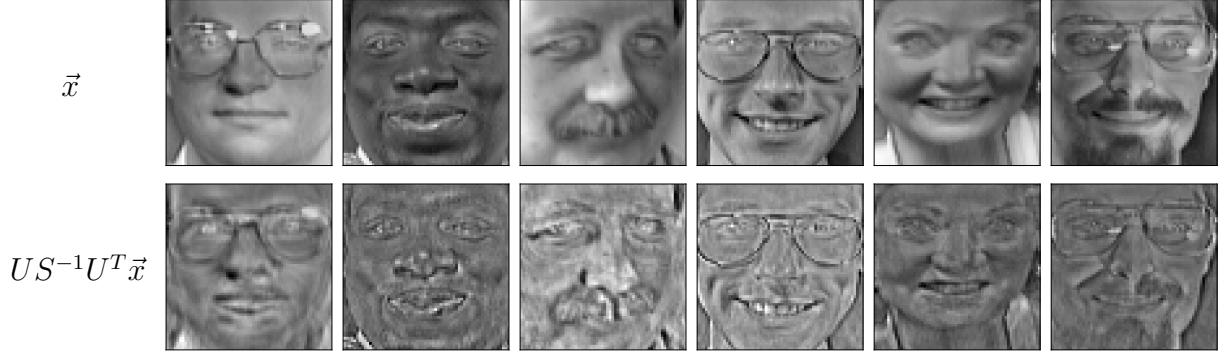


Figure 15: Centered faces in the data set from Example 3.4 before and after whitening.

The linear map $US^{-1}U^T$ scales the components of the centered vector in each principal direction by a factor inversely proportional to its corresponding singular value,

$$\vec{w}_i := \sum_{j=1}^{\min(m,n)} \frac{1}{\sigma_j} \langle \vec{u}_j, \vec{c}_i \rangle \vec{u}_j. \quad (144)$$

If we group the whitened vectors as columns of a matrix, the matrix can be expressed as

$$W = US^{-1}U^T C. \quad (145)$$

The covariance matrix of the whitened data is proportional to the identity

$$\Sigma(\vec{c}_1, \dots, \vec{c}_n) = \frac{1}{n-1} WW^T \quad (146)$$

$$= \frac{1}{n-1} US^{-1}U^T CC^T US^{-1}U^T \quad (147)$$

$$= \frac{1}{n-1} US^{-1}U^T USV^T VSU^T US^{-1}U^T \quad (148)$$

$$= \frac{1}{n-1} I, \quad (149)$$

This means that the whitened data have no linear skews, there are no directions in space that contain more variation than others. As illustrated in Figure 14, this may reveal nonlinear structure in the data. Figure 15 shows some of the centered faces in the data set from Example 3.4 before and after whitening. Whitening enhances fine-detail features of the faces.

4 Eigendecomposition

An eigenvector \vec{q} of a square matrix $A \in \mathbb{R}^{n \times n}$ satisfies

$$A\vec{q} = \lambda\vec{q} \quad (150)$$

for a scalar λ which is the corresponding eigenvalue. Even if A is real, its eigenvectors and eigenvalues can be complex. If a matrix has n linearly independent eigenvectors then it is diagonalizable.

Lemma 4.1 (Eigendecomposition). *If a square matrix $A \in \mathbb{R}^{n \times n}$ has n linearly independent eigenvectors $\vec{q}_1, \dots, \vec{q}_n$ with eigenvalues $\lambda_1, \dots, \lambda_n$ it can be expressed in terms of a matrix Q , whose columns are the eigenvectors, and a diagonal matrix containing the eigenvalues,*

$$A = [\vec{q}_1 \ \vec{q}_2 \ \cdots \ \vec{q}_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} [\vec{q}_1 \ \vec{q}_2 \ \cdots \ \vec{q}_n]^{-1} \quad (151)$$

$$= Q\Lambda Q^{-1} \quad (152)$$

Proof.

$$AQ = [A\vec{q}_1 \ A\vec{q}_2 \ \cdots \ A\vec{q}_n] \quad (153)$$

$$= [\lambda_1\vec{q}_1 \ \lambda_2\vec{q}_2 \ \cdots \ \lambda_n\vec{q}_n] \quad (154)$$

$$= Q\Lambda. \quad (155)$$

If the columns of a square matrix are all linearly independent, then the matrix has an inverse, so multiplying the expression by Q^{-1} on both sides completes the proof. \square

Lemma 4.2. *Not all matrices have an eigendecomposition.*

Proof. Consider the matrix

$$\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}. \quad (156)$$

Assume an eigenvector \vec{q} associated to an eigenvalue λ , then

$$\begin{bmatrix} \vec{q}[2] \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{q}[1] \\ \vec{q}[2] \end{bmatrix} = \begin{bmatrix} \lambda\vec{q}[1] \\ \lambda\vec{q}[2] \end{bmatrix}, \quad (157)$$

which implies that $\vec{q}[2] = 0$ and $\vec{q}[1] = 0$, so the matrix does not have eigenvectors associated to nonzero eigenvalues. \square

Symmetric matrices are always diagonalizable.

Theorem 4.3 (Spectral theorem for symmetric matrices). *If $A \in \mathbb{R}^n$ is symmetric, then it has an eigendecomposition of the form*

$$A = U\Lambda U^T \quad (158)$$

where the matrix of eigenvectors U is an orthogonal matrix.

This is a fundamental result in linear algebra that can be used to prove Theorem 2.1. We refer to any graduate-level linear-algebra text for the proof.

Together, Theorems 2.1 and 4.3 imply that the SVD $A = USV^T$ and the eigendecomposition $A = U\Lambda U^T$ of a symmetric matrix are almost the same. The left singular vectors can be taken to be equal to the eigenvectors. Nonnegative eigenvalues are equal to the singular vectors, and their right singular vectors are equal to the corresponding eigenvectors. The difference is that if an eigenvalue λ_i corresponding to an eigenvector \vec{u}_i is negative, then $\sigma_i = -\lambda_i$ and the corresponding right-singular vector $\vec{v}_i = -\vec{u}_i$.

A useful application of the eigendecomposition is computing successive matrix products. Assume that we are interested in computing

$$AA \cdots A\vec{x} = A^k\vec{x}, \quad (159)$$

i.e., we want to apply A to \vec{x} k times. A^k cannot be computed by taking the power of its entries (try out a simple example to convince yourself). However, if A has an eigendecomposition,

$$A^k = Q\Lambda Q^{-1}Q\Lambda Q^{-1} \cdots Q\Lambda Q^{-1} \quad (160)$$

$$= Q\Lambda^k Q^{-1} \quad (161)$$

$$= Q \begin{bmatrix} \lambda_1^k & 0 & \cdots & 0 \\ 0 & \lambda_2^k & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \lambda_n^k \end{bmatrix} Q^{-1}, \quad (162)$$

using the fact that for diagonal matrices applying the matrix repeatedly is equivalent to taking the power of the diagonal entries. This allows to compute the k matrix products using just 3 matrix products and taking the power of n numbers.

Let $A \in \mathbb{R}^{n \times n}$ be a matrix with eigendecomposition $Q\Lambda Q^{-1}$ and let \vec{x} be an arbitrary vector in \mathbb{R}^n . Since the eigenvectors are linearly independent, they form a basis for \mathbb{R}^n , so we can represent \vec{x} as

$$\vec{x} = \sum_{i=1}^n \alpha_i \vec{q}_i, \quad \alpha_i \in \mathbb{R}, \quad 1 \leq i \leq n. \quad (163)$$

Now let us apply A to \vec{x} k times,

$$A^k \vec{x} = \sum_{i=1}^n \alpha_i A^k \vec{q}_i \quad (164)$$

$$= \sum_{i=1}^n \alpha_i \lambda_i^k \vec{q}_i. \quad (165)$$

If we assume that the eigenvectors are ordered according to their magnitudes and that the magnitude of one of them is larger than the rest, $|\lambda_1| > |\lambda_2| \geq \dots$, and that $\alpha_1 \neq 0$

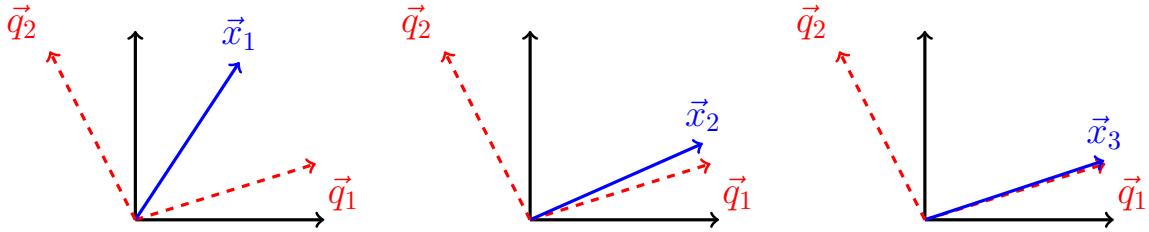


Figure 16: Illustration of the first three iterations of the power method for a matrix with eigenvectors \vec{q}_1 and \vec{q}_2 , with corresponding eigenvalues $\lambda_1 = 1.05$ and $\lambda_2 = 0.1661$.

(which happens with high probability if we draw a random \vec{x}) then as k grows larger the term $\alpha_1 \lambda_1^k \vec{q}_1$ dominates. The term will blow up or tend to zero unless we normalize every time before applying A . Adding the normalization step to this procedure results in the power method or power iteration, an algorithm for estimating the eigenvector of a matrix that corresponds to the largest eigenvalue.

Algorithm 4.4 (Power method).

Set $\vec{x}_1 := \vec{x}/\|\vec{x}\|_2$, where the entries of \vec{x} are drawn at random. For $i = 1, 2, 3, \dots$, compute

$$\vec{x}_i := \frac{A\vec{x}_{i-1}}{\|A\vec{x}_{i-1}\|_2}. \quad (166)$$

Figure 16 illustrates the power method on a simple example, where the matrix is equal to

$$A = \begin{bmatrix} 0.930 & 0.388 \\ 0.237 & 0.286 \end{bmatrix}. \quad (167)$$

The convergence to the eigenvector corresponding to the eigenvalue with the largest magnitude is very fast.

We end this section with an example that applies a decomposition to analyze the evolution of the populations of two animals.

Example 4.5 (Deer and wolfs). A biologist is studying the populations of deer and wolfs in Yellowstone. She concludes that a reasonable model for the populations in year $n + 1$ is the linear system of equations

$$d_{n+1} = \frac{5}{4}d_n - \frac{3}{4}w_n, \quad (168)$$

$$w_{n+1} = \frac{1}{4}d_n + \frac{1}{4}w_n, \quad n = 0, 1, 2, \dots \quad (169)$$

where d_n and w_n denote the number of deer and wolfs in year n . She is interested in determining the evolution of the populations in the future so she computes an eigende-

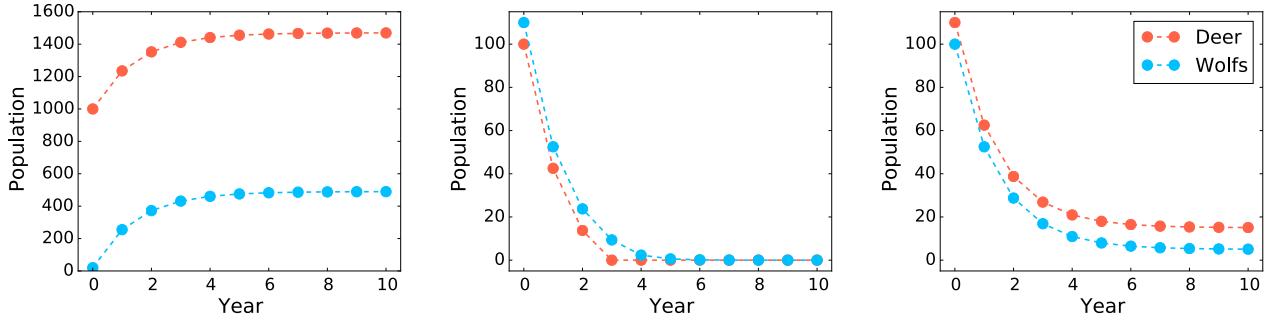


Figure 17: Evolution of the populations of deer and wolfs for different initial populations in Example 4.5.

composition of the matrix

$$A := \begin{bmatrix} 5/4 & -3/4 \\ 1/4 & 1/4 \end{bmatrix} \quad (170)$$

$$= \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix}^{-1} := Q\Lambda Q^{-1}. \quad (171)$$

If we denote the initial populations of deer and wolfs as d_0 and w_0 respectively, the populations in year n are given by

$$\begin{bmatrix} d_n \\ w_n \end{bmatrix} = Q\Lambda^n Q^{-1} \begin{bmatrix} d_0 \\ w_0 \end{bmatrix} \quad (172)$$

$$= \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0.5^n \end{bmatrix} \begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 1.5 \end{bmatrix} \begin{bmatrix} d_0 \\ w_0 \end{bmatrix} \quad (173)$$

$$= \frac{d_0 - w_0}{2} \begin{bmatrix} 3 \\ 1 \end{bmatrix} + \frac{3w_0 - d_0}{8^n} \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \quad (174)$$

As $n \rightarrow \infty$, if the number of deer is larger than the number of wolfs, then the population of deer will converge to be three times the population of wolfs, which will converge to a half of the difference between their original populations. Since the populations cannot be negative, if the original population of wolfs is larger than that of deer, then both species will go extinct. This is confirmed by the simulations shown in Figure 17. \triangle

5 Proofs

5.1 Proof of Theorem 1.3

It is sufficient to prove

$$\dim(\text{row}(A)) \leq \dim(\text{col}(A)) \quad (175)$$

for any arbitrary matrix A . Since the row space of A is equal to the column space of A^T and vice versa, applying (175) to A^T yields $\dim(\text{row}(A)) \geq \dim(\text{col}(A))$ which completes the proof.

To prove (175) let $r := \dim(\text{row}(A))$ and let $\vec{x}_1, \dots, \vec{x}_r$ be a basis for $\text{row}(A)$. Consider the vectors $A\vec{x}_1, \dots, A\vec{x}_r$. They belong to $\text{col}(A)$ by (11), so if they are linearly independent then $\dim(\text{col}(A)) \geq r$. We prove that this is the case by contradiction.

Assume that $A\vec{x}_1, \dots, A\vec{x}_r$ are linearly dependent. Then there exist scalar coefficients $\alpha_1, \dots, \alpha_r$ such that

$$\vec{0} = \sum_{i=1}^r \alpha_i A\vec{x}_i = A \left(\sum_{i=1}^r \alpha_i \vec{x}_i \right) \quad \text{by linearity of the matrix product,} \quad (176)$$

This implies that $\sum_{i=1}^r \alpha_i \vec{x}_i$ is orthogonal to every row of A and hence to every vector in $\text{row}(A)$. However it is in the span of a basis of $\text{row}(A)$ by construction! This is only possible if $\sum_{i=1}^r \alpha_i \vec{x}_i = 0$, which is a contradiction because $\vec{x}_1, \dots, \vec{x}_r$ are assumed to be linearly independent.

5.2 Proof of Lemma 2.9

Let $\vec{a}_1, \vec{a}_2, \dots, \vec{a}_{d_1}$ be a basis for the first subspace and $\vec{b}_1, \vec{b}_2, \dots, \vec{b}_{d_2}$ a basis for the second. Because the dimension of the vector space is n , the set of vectors $\vec{a}_1, \dots, \vec{a}_{d_1}, \vec{b}_1, \dots, \vec{b}_{d_2}$ are not linearly independent. There must exist scalars $\alpha_1, \dots, \alpha_{d_1}, \beta_1, \dots, \beta_{d_2}$, which are not all equal to zero, such that

$$\sum_{i=1}^{d_1} \alpha_i \vec{a}_i + \sum_{j=1}^{d_2} \beta_j \vec{b}_j = 0. \quad (177)$$

The vector

$$\vec{x} := \sum_{i=1}^{d_1} \alpha_i \vec{a}_i = - \sum_{j=1}^{d_2} \beta_j \vec{b}_j \quad (178)$$

cannot equal zero because both $\vec{a}_1, \dots, \vec{a}_{d_1}$ and $\vec{b}_1, \dots, \vec{b}_{d_2}$ are bases by assumption. \vec{x} belongs to the intersection of the two subspaces, which completes the proof.

5.3 Proof of Theorem 2.16

The proof relies on the following lemma.

Lemma 5.1. *For any $Q \in \mathbb{R}^{n \times n}$*

$$\max_{1 \leq i \leq n} |Q_{ii}| \leq \|Q\|. \quad (179)$$

Proof. Since $\|\vec{e}_i\|_2 = 1$,

$$\max_{1 \leq i \leq n} |Q_{ii}| \leq \max_{1 \leq i \leq n} \sqrt{\sum_{j=1}^n Q_{ji}^2} \quad (180)$$

$$= \max_{1 \leq i \leq n} \|Q \vec{e}_i\|_2 \quad (181)$$

$$\leq \|Q\|. \quad (182)$$

□

We denote the SVD of A by USV^T ,

$$\begin{aligned} \sup_{\{\|B\| \leq 1 \mid B \in \mathbb{R}^{m \times n}\}} \text{tr}(A^T B) &= \sup_{\{\|B\| \leq 1 \mid B \in \mathbb{R}^{m \times n}\}} \text{tr}(V S U^T B) \quad (183) \\ &= \sup_{\{\|B\| \leq 1 \mid B \in \mathbb{R}^{m \times n}\}} \text{tr}(S B U^T V) \quad \text{by Lemma 2.5 in Lecture Notes 1} \\ &\leq \sup_{\{\|M\| \leq 1 \mid M \in \mathbb{R}^{m \times n}\}} \text{tr}(S M) \quad \|B\| = \|B U^T V\| \text{ by Corollary 2.15} \\ &\leq \sup_{\{\max_{1 \leq i \leq n} |M_{ii}| \leq 1 \mid M \in \mathbb{R}^{m \times n}\}} \text{tr}(S M) \quad \text{by Lemma 5.1} \\ &\leq \sup_{\{\max_{1 \leq i \leq n} |M_{ii}| \leq 1 \mid M \in \mathbb{R}^{m \times n}\}} \sum_{i=1}^n M_{ii} \sigma_i \quad (184) \\ &\leq \sum_{i=1}^n \sigma_i \quad (185) \\ &= \|A\|_* . \quad (186) \end{aligned}$$

To complete the proof, we need to show that the equality holds. Note that UV^T has operator norm equal to one because its r singular values (recall that r is the rank of A) are equal to one. We have

$$\langle A, UV^T \rangle = \text{tr}(A^T UV^T) \quad (187)$$

$$= \text{tr}(V S U^T UV^T) \quad (188)$$

$$= \text{tr}(V^T V S) \quad \text{by Lemma 2.5 in Lecture Notes 1} \quad (189)$$

$$= \text{tr}(S) \quad (190)$$

$$= \|A\|_* . \quad (191)$$

Lecture Notes 3: Randomness

1 Gaussian random variables

The Gaussian or normal random variable is arguably the most popular random variable in statistical modeling and signal processing. The reason is that sums of independent random variables often converge to Gaussian distributions, a phenomenon characterized by the central limit theorem (see Theorem 1.3 below). As a result any quantity that results from the additive combination of several unrelated factors will tend to have a Gaussian distribution. For example, in signal processing and engineering, noise is often modeled as Gaussian. Figure 1 shows the pdfs of Gaussian random variables with different means and variances. When a Gaussian has mean zero and unit variance, we call it a *standard Gaussian*.

Definition 1.1 (Gaussian). *The pdf of a Gaussian or normal random variable with mean μ and standard deviation σ is given by*

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \quad (1)$$

A Gaussian distribution with mean μ and standard deviation σ is usually denoted by $\mathcal{N}(\mu, \sigma^2)$.

An important property of Gaussian random variables is that scaling and shifting Gaussians preserves their distribution.

Lemma 1.2. *If \mathbf{x} is a Gaussian random variable with mean μ and standard deviation σ , then for any $a, b \in \mathbb{R}$*

$$\mathbf{y} := a\mathbf{x} + b \quad (2)$$

is a Gaussian random variable with mean $a\mu + b$ and standard deviation $|a|\sigma$.

Proof. We assume $a > 0$ (the argument for $a < 0$ is very similar), to obtain

$$F_{\mathbf{y}}(y) = P(\mathbf{y} \leq y) \quad (3)$$

$$= P(a\mathbf{x} + b \leq y) \quad (4)$$

$$= P\left(\mathbf{x} \leq \frac{y-b}{a}\right) \quad (5)$$

$$= \int_{-\infty}^{\frac{y-b}{a}} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \quad (6)$$

$$= \int_{-\infty}^y \frac{1}{\sqrt{2\pi}a\sigma} e^{-\frac{(w-a\mu-b)^2}{2a^2\sigma^2}} dw \quad \text{by the change of variables } w = ax + b. \quad (7)$$

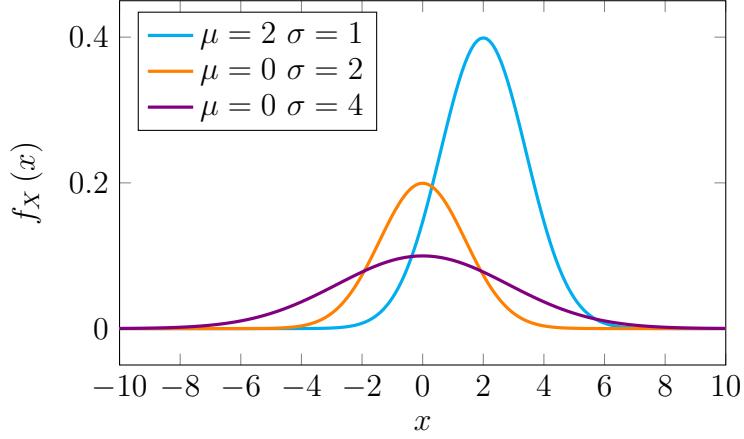


Figure 1: Gaussian random variable with different means and standard deviations.

Differentiating with respect to y yields

$$f_y(y) = \frac{1}{\sqrt{2\pi}a\sigma} e^{-\frac{(y-a\mu-b)^2}{2a^2\sigma^2}} \quad (8)$$

so \mathbf{y} is indeed a standard Gaussian random variable with mean $a\mu + b$ and standard deviation $|a|\sigma$. \square

The distribution of the average of a large number of random variables with bounded variances converges to a Gaussian distribution.

Theorem 1.3 (Central limit theorem). *Let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots$ be a sequence of iid random variables with mean μ and bounded variance σ^2 . We define the sequence of averages $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \dots$, as*

$$\mathbf{a}_i := \frac{1}{i} \sum_{j=1}^i \mathbf{x}_j. \quad (9)$$

The sequence $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \dots$

$$\mathbf{b}_i := \sqrt{i}(\mathbf{a}_i - \mu) \quad (10)$$

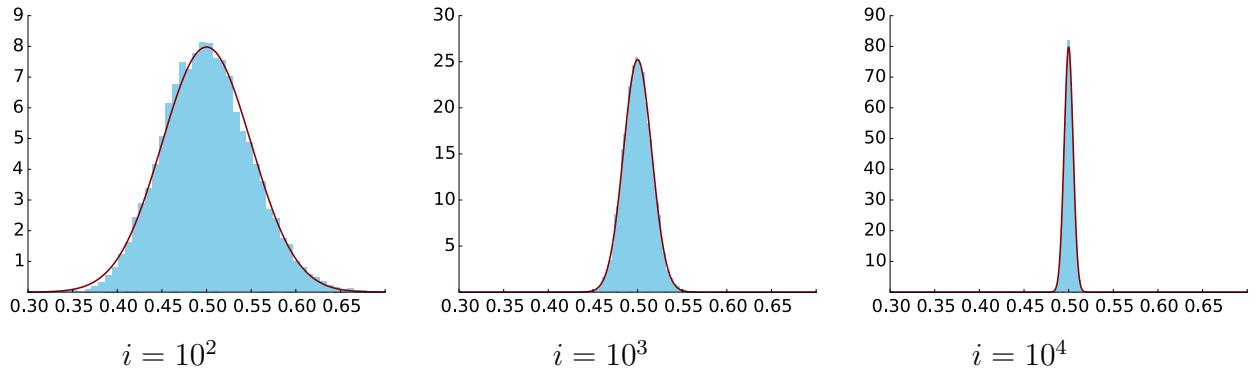
converges in distribution to a Gaussian random variable with mean 0 and variance σ^2 , meaning that for any $x \in \mathbb{R}$

$$\lim_{i \rightarrow \infty} f_{\mathbf{b}_i}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}. \quad (11)$$

For large i the theorem suggests that the average \mathbf{a}_i is approximately Gaussian with mean μ and variance σ/\sqrt{n} . This is verified numerically in Figure 2. Figure 3 shows the histogram of the heights in a population of 25,000 people and how it is very well approximated by a Gaussian random variable¹, suggesting that a person's height may result from a combination of independent factors (genes, nutrition, etc.).

¹The data is available [here](#).

Exponential with $\lambda = 2$ (iid)



Geometric with $p = 0.4$ (iid)

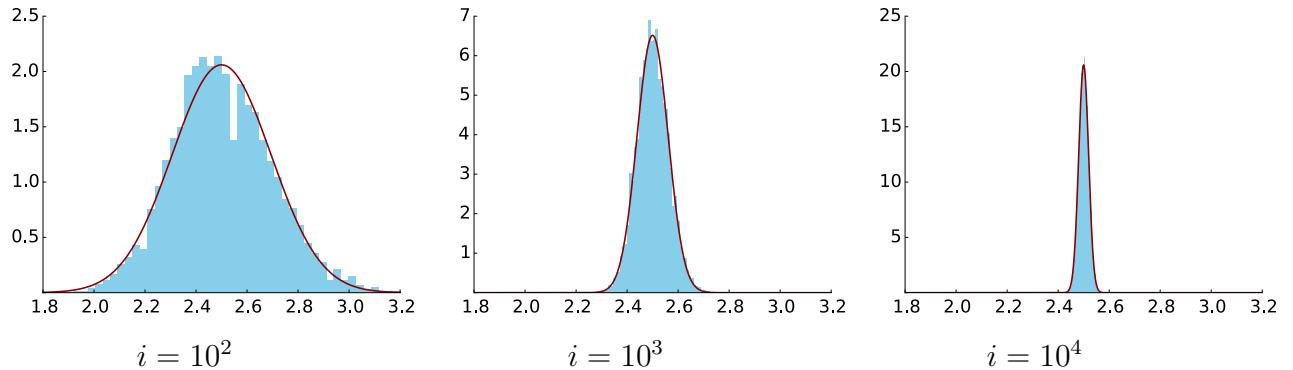


Figure 2: Empirical distribution of the average, defined as in equation (9), of an iid exponential sequence with parameter $\lambda = 2$ (top) and an iid geometric sequence with parameter $p = 0.4$ (bottom). The empirical distribution is computed from 10^4 samples in all cases. The estimate provided by the central limit theorem is plotted in red.

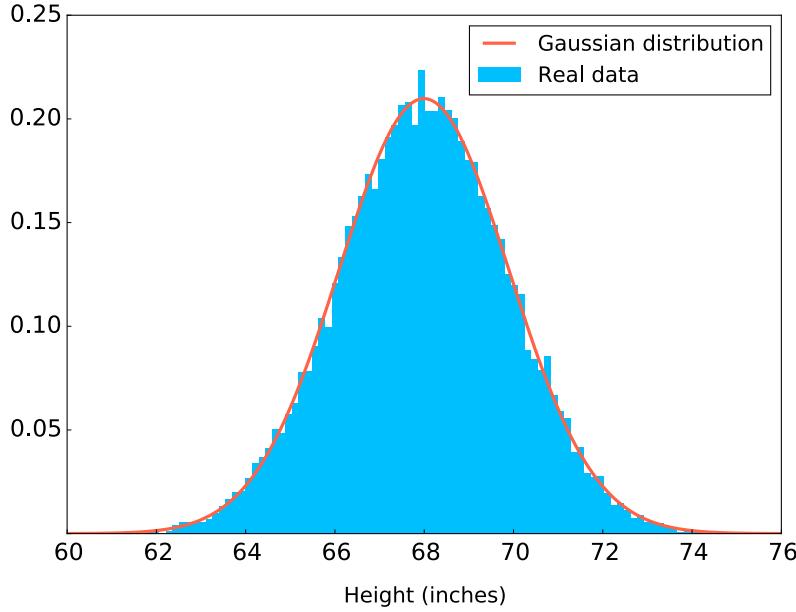


Figure 3: Histogram of heights in a population of 25,000 people (blue) and its approximation using a Gaussian distribution (orange).

2 Gaussian random vectors

2.1 Definition and basic properties

Gaussian random vectors are a multidimensional generalization of Gaussian random variables. They are parametrized by a vector and a matrix that correspond to their mean and covariance matrix.

Definition 2.1 (Gaussian random vector). *A Gaussian random vector \vec{x} is a random vector with joint pdf ($|\Sigma|$ denotes the determinant of Σ)*

$$f_{\vec{x}}(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2} (\vec{x} - \vec{\mu})^T \Sigma^{-1} (\vec{x} - \vec{\mu})\right) \quad (12)$$

where the mean vector $\vec{\mu} \in \mathbb{R}^n$ and the covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$, which is symmetric and positive definite, parametrize the distribution. A Gaussian distribution with mean $\vec{\mu}$ and covariance matrix Σ is usually denoted by $\mathcal{N}(\vec{\mu}, \Sigma)$.

When the covariance matrix of a Gaussian vector is diagonal, then its components are all independent.

Lemma 2.2 (Uncorrelation implies mutual independence for Gaussian random vectors). *If all the components of a Gaussian random vector \vec{x} are uncorrelated, then they are also mutually independent.*

Proof. If all the components are uncorrelated then the covariance matrix is diagonal

$$\Sigma_{\vec{x}} = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix}, \quad (13)$$

where σ_i is the standard deviation of the i th component. Now, the inverse of this diagonal matrix is just

$$\Sigma_{\vec{x}}^{-1} = \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sigma_n^2} \end{bmatrix}, \quad (14)$$

and its determinant is $|\Sigma| = \prod_{i=1}^n \sigma_i^2$ so that

$$f_{\vec{x}}(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2} (\vec{x} - \vec{\mu})^T \Sigma^{-1} (\vec{x} - \vec{\mu})\right) \quad (15)$$

$$= \prod_{i=1}^n \frac{1}{\sqrt{(2\pi)\sigma_i}} \exp\left(-\frac{(\vec{x}_i - \mu_i)^2}{2\sigma_i^2}\right) \quad (16)$$

$$= \prod_{i=1}^n f_{\vec{x}_i}(\vec{x}_i). \quad (17)$$

Since the joint pdf factors into a product of the marginals, the components are all mutually independent. \square

When the covariance matrix of a Gaussian vector is the identity and its mean is zero, then its entries are iid standard Gaussians with mean zero and unit variance. We refer to such vectors as iid standard Gaussian vectors.

A fundamental property of Gaussian random vectors is that performing linear transformations on them always yields vectors with joint distributions that are also Gaussian. This is a multidimensional generalization of Lemma 1.2. We omit the proof, which is similar to that of Lemma 1.2.

Theorem 2.3 (Linear transformations of Gaussian random vectors are Gaussian). *Let \vec{x} be a Gaussian random vector of dimension n with mean $\vec{\mu}$ and covariance matrix Σ . For any matrix $A \in \mathbb{R}^{m \times n}$ and $\vec{b} \in \mathbb{R}^m$, $\vec{Y} = A\vec{x} + \vec{b}$ is a Gaussian random vector with mean $A\vec{\mu} + \vec{b}$ and covariance matrix $A\Sigma A^T$.*

An immediate consequence of Theorem 2.3 is that subvectors of Gaussian vectors are also Gaussian. Figure 4 show the joint pdf of a two-dimensional Gaussian vector together with the marginal pdfs of its entries.

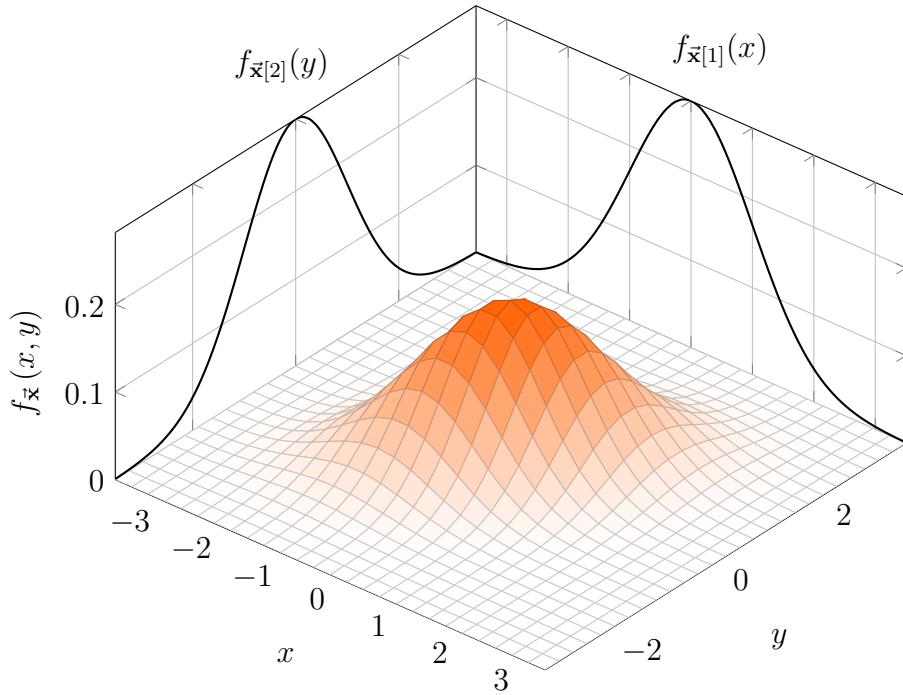


Figure 4: Joint pdf of a two-dimensional Gaussian vector \vec{x} and marginal pdfs of its two entries.

Another consequence of Theorem 2.3 is that an iid standard Gaussian vector is *isotropic*. This means that the vector does not favor any direction in its ambient space. More formally, no matter how you rotate it, its distribution is the same. More precisely, for any orthogonal matrix U , if \vec{x} is an iid standard Gaussian vector, then by Theorem 2.3 $U\vec{x}$ has the same distribution, since its mean equals $U\vec{0} = \vec{0}$ and its covariance matrix equals $UIU^T = UU^T = I$. Note that this is a stronger statement than saying that its variance is the same in every direction, which is true for any vector with uncorrelated entries.

2.2 Concentration in high dimensions

In the previous section we established that the direction of iid standard Gaussian vectors is isotropic. We now consider their magnitude. As we can see in Figure 4, in low dimensions the joint pdf of Gaussian vectors is mostly concentrated around the origin. Interestingly, this is not the case as the dimension of the ambient space grows. The squared ℓ_2 -norm of an iid standard k -dimensional Gaussian vector \vec{x} is the sum of k independent standard Gaussian random variables, which is known as a χ^2 (chi squared) random variable with k degrees of freedom. As shown in Figure 5, as k grows the pdf of this random variable

concentrates around k , which is the mean of the squared ℓ_2 -norm:

$$\mathbb{E} (\|\vec{\mathbf{x}}\|_2^2) = \mathbb{E} \left(\sum_{i=1}^k \vec{\mathbf{x}}[i]^2 \right) \quad (18)$$

$$= \sum_{i=1}^k \mathbb{E} (\vec{\mathbf{x}}[i]^2) \quad (19)$$

$$= k. \quad (20)$$

The following lemma shows that the standard deviation of $\|\vec{\mathbf{x}}\|_2^2$ is $\sqrt{2k}$.

Lemma 2.4 (Variance of the squared ℓ_2 norm of a Gaussian vector). *Let $\vec{\mathbf{x}}$ be an iid Gaussian random vector of dimension k . The variance of $\|\vec{\mathbf{x}}\|_2^2$ is $2k$.*

Proof. Recall that $\text{Var} (\|\vec{\mathbf{x}}\|_2^2) = \mathbb{E} ((\|\vec{\mathbf{x}}\|_2^2)^2) - \mathbb{E} (\|\vec{\mathbf{x}}\|_2^2)^2$. The result follows from

$$\mathbb{E} ((\|\vec{\mathbf{x}}\|_2^2)^2) = \mathbb{E} \left(\left(\sum_{i=1}^k \vec{\mathbf{x}}[i]^2 \right)^2 \right) \quad (21)$$

$$= \mathbb{E} \left(\sum_{i=1}^k \sum_{j=1}^k \vec{\mathbf{x}}[i]^2 \vec{\mathbf{x}}[j]^2 \right) \quad (22)$$

$$= \sum_{i=1}^k \sum_{j=1}^k \mathbb{E} (\vec{\mathbf{x}}[i]^2 \vec{\mathbf{x}}[j]^2) \quad (23)$$

$$= \sum_{i=1}^k \mathbb{E} (\vec{\mathbf{x}}[i]^4) + 2 \sum_{i=1}^{k-1} \sum_{j=i}^k \mathbb{E} (\vec{\mathbf{x}}[i]^2) \mathbb{E} (\vec{\mathbf{x}}[j]^2) \quad (24)$$

$$= 3k + k(k-1) \quad \text{since the 4th moment of a standard Gaussian equals 3} \quad (25)$$

$$= k(k+2). \quad (26)$$

□

The result implies that as k grows the relative deviation of the norm from its mean decreases proportionally to $1/\sqrt{k}$. Consequently, the squared norm is close to k with increasing probability. This is made precise in the following theorem, which yields a concrete non-asymptotic bound on the probability that it deviates by more than a small constant.

Theorem 2.5 (Chebyshev tail bound for the ℓ_2 norm of an iid standard Gaussian vector). *Let $\vec{\mathbf{x}}$ be an iid standard Gaussian random vector of dimension k . For any $\epsilon > 0$ we have*

$$P (k(1-\epsilon) < \|\vec{\mathbf{x}}\|_2^2 < k(1+\epsilon)) \geq 1 - \frac{2}{k\epsilon^2}. \quad (27)$$

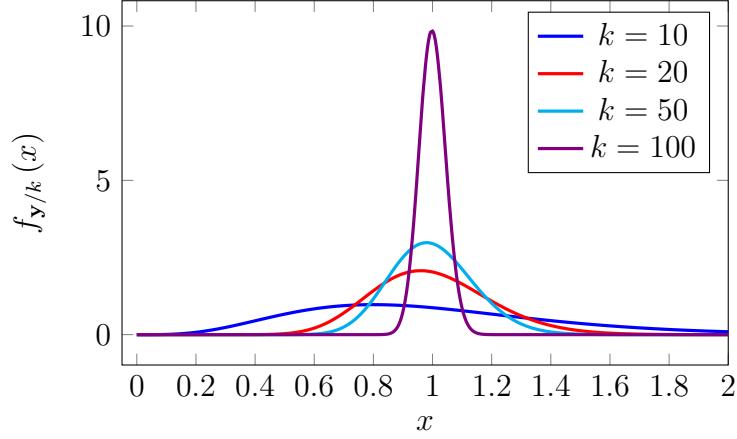


Figure 5: Pdfs of \mathbf{y}/k for different values of k , where \mathbf{y} is a χ^2 random variable with k degrees of freedom.

Proof. The bound is a consequence of Markov's inequality, which quantifies the intuitive idea that if a random variable is nonnegative and small then the probability that it takes large values must be small.

Theorem 2.6 (Markov's inequality, proof in Section 5.1). *Let \mathbf{x} be a nonnegative random variable. For any positive constant $a > 0$,*

$$P(\mathbf{x} \geq a) \leq \frac{E(\mathbf{x})}{a}. \quad (28)$$

Let $\mathbf{y} := \|\vec{\mathbf{x}}\|_2^2$,

$$P(|\mathbf{y} - k| \geq k\epsilon) = P((\mathbf{y} - E(\mathbf{y}))^2 \geq k^2\epsilon^2) \quad (29)$$

$$\leq \frac{E((\mathbf{y} - E(\mathbf{y}))^2)}{k^2\epsilon^2} \quad \text{by Markov's inequality} \quad (30)$$

$$= \frac{\text{Var}(\mathbf{y})}{k^2\epsilon^2} \quad (31)$$

$$= \frac{2}{k\epsilon^2} \quad \text{by Lemma 2.4.} \quad (32)$$

When Markov's inequality is applied to bound the deviation from the mean like this, it is usually called Chebyshev's inequality. \square

The bound in Theorem 2.5 only relies on the variance to bound the probability that the magnitude deviates from its mean. As a result, it is significantly weaker than the following result, which exploits the fact that the higher moments of a standard Gaussian are well behaved.

Theorem 2.7 (Chernoff tail bound for the ℓ_2 norm of an iid standard Gaussian vector). *Let $\vec{\mathbf{x}}$ be an iid standard Gaussian random vector of dimension k . For any $\epsilon \in (0, 1)$ we have*

$$P(k(1 - \epsilon) < \|\vec{\mathbf{x}}\|_2^2 < k(1 + \epsilon)) \geq 1 - 2 \exp\left(-\frac{k\epsilon^2}{8}\right). \quad (33)$$

Proof. Let $\mathbf{y} := \|\vec{\mathbf{x}}\|_2^2$. The result is implied by

$$P(\mathbf{y} > k(1 + \epsilon)) \leq \exp\left(-\frac{k\epsilon^2}{8}\right), \quad (34)$$

$$P(\mathbf{y} < k(1 - \epsilon)) \leq \exp\left(-\frac{k\epsilon^2}{8}\right). \quad (35)$$

We present the proof of (34). The proof of (35) is essentially the same and is presented in Section 5.3. Let $t > 0$ be an arbitrary positive number, and note that

$$P(\mathbf{y} > a) = P(\exp(t\mathbf{y}) > \exp(at)) \quad (36)$$

$$\leq \exp(-at) E(\exp(t\mathbf{y})) \quad \text{by Markov's inequality} \quad (37)$$

$$\leq \exp(-at) E\left(\exp\left(\sum_{i=1}^k t\mathbf{x}_i^2\right)\right) \quad (38)$$

$$\leq \exp(-at) \prod_{i=1}^k E(\exp(t\mathbf{x}_i^2)) \quad \text{by independence of } \mathbf{x}_1, \dots, \mathbf{x}_k \quad (39)$$

$$= \frac{\exp(-at)}{(1 - 2t)^{\frac{k}{2}}}, \quad (40)$$

where the last step is a consequence of the following lemma.

Lemma 2.8 (Proof in Section 5.2). *For \mathbf{x} standard Gaussian and $t < 1/2$,*

$$E(\exp(t\mathbf{x}^2)) = \frac{1}{\sqrt{1 - 2t}}. \quad (41)$$

Note that the lemma implies a bound on the higher-order moments of a standard Gaussian \mathbf{x} , since

$$E(\exp(t\mathbf{x}^2)) = E\left(\sum_{i=0}^{\infty} \frac{(t\mathbf{x}^2)^i}{i!}\right) \quad (42)$$

$$= \sum_{i=0}^{\infty} \frac{E(t^i (\mathbf{x}^{2i}))}{i!}. \quad (43)$$

Bounds that exploit the behavior of higher-order moments to control tail probabilities through the expectation of an exponential are often called Chernoff bounds.

We set $a := k(1 + \epsilon)$ and

$$t := \frac{1}{2} - \frac{1}{2(1 + \epsilon)}, \quad (44)$$

by minimizing over $t \in (0, 1/2)$ in (40). This gives

$$P(\mathbf{y} > k(1 + \epsilon)) \leq (1 + \epsilon)^k 2 \exp\left(-\frac{k\epsilon}{2}\right) \quad (45)$$

$$= \exp\left(-\frac{k}{2}(\epsilon - \log(1 + \epsilon))\right) \quad (46)$$

$$\leq \exp\left(-\frac{k\epsilon^2}{8}\right), \quad (47)$$

where the last step follows from the fact that the function $g(x) := x - \frac{x^2}{4} - \log(1 + x)$ is nonnegative between 0 and 1 (the derivative is nonnegative and $g(0) = 0$). \square

In the next section we apply this result to characterize the projection of an iid standard Gaussian vector on a subspace.

2.3 Projection onto a fixed subspace

In Example 7.4 of Lecture Notes 1 we observed that the ℓ_2 -norm of the projection of iid standard Gaussian noise onto a fixed subspace is proportional to the square root of the dimension of that subspace. In this section we make this precise, using that the fact that the coefficients of the projection in an orthonormal basis of the subspace are themselves iid standard Gaussians.

Lemma 2.9 (Projection of an iid Gaussian vector onto a subspace). *Let \mathcal{S} be a k -dimensional subspace of \mathbb{R}^n and $\vec{\mathbf{z}} \in \mathbb{R}^n$ a vector of iid standard Gaussian noise. $\|\mathcal{P}_{\mathcal{S}} \vec{\mathbf{z}}\|_2^2$ is a χ^2 random variable with k degrees of freedom, i.e. it has the same distribution as the random variable*

$$\mathbf{y} := \sum_{i=1}^k \mathbf{x}_i^2 \quad (48)$$

where $\mathbf{x}_1, \dots, \mathbf{x}_k$ are iid standard Gaussian random variables.

Proof. Let UU^T be a projection matrix for the subspace \mathcal{S} , where the columns of $U \in \mathbb{R}^{n \times k}$

are orthonormal. We have

$$\|\mathcal{P}_{\mathcal{S}} \vec{\mathbf{z}}\|_2^2 = \|UU^T \vec{\mathbf{z}}\|_2^2 \quad (49)$$

$$= \vec{\mathbf{z}}^T UU^T UU^T \vec{\mathbf{z}} \quad (50)$$

$$= \vec{\mathbf{z}}^T UU^T \vec{\mathbf{z}} \quad (51)$$

$$= \vec{\mathbf{w}}^T \vec{\mathbf{w}} \quad (52)$$

$$= \sum_{i=1}^k \vec{\mathbf{w}}[i]^2, \quad (53)$$

where by Theorem 2.3 the random vector $\vec{\mathbf{w}} := U^T \vec{\mathbf{z}}$ is Gaussian with mean zero and covariance matrix

$$\Sigma_{\vec{\mathbf{w}}} = U^T \Sigma_{\vec{\mathbf{z}}} U \quad (54)$$

$$= U^T U \quad (55)$$

$$= I, \quad (56)$$

so the entries are independent standard Gaussians. \square

Since the coefficients are standard Gaussians, we can bound the deviation of their norm using Theorem 2.7.

Theorem 2.10. *Let \mathcal{S} be a k -dimensional subspace of \mathbb{R}^n and $\vec{\mathbf{z}} \in \mathbb{R}^n$ a vector of iid Gaussian noise. For any $\epsilon \in (0, 1)$*

$$\sqrt{k(1-\epsilon)} \leq \|\mathcal{P}_{\mathcal{S}} \vec{\mathbf{z}}\|_2 \leq \sqrt{k(1+\epsilon)} \quad (57)$$

with probability at least $1 - 2 \exp(-k\epsilon^2/8)$.

Proof. The result follows from Theorem 2.7 and Lemma 2.9. \square

3 Gaussian matrices

3.1 Randomized projections

As we discussed in Section 3.3 of Lecture Notes 2, dimensionality reduction via PCA consists of projecting the data on low-dimensional subspaces that are optimal in the sense that they preserve most of the energy. The principal directions are guaranteed to lie in the directions of maximum variation of the data, but finding them requires computing the SVD, which can be computationally expensive or not possible at all if the aim is to project a stream of data in real time. For such cases we need a *non-adaptive* alternative to PCA that chooses the projection before seeing the data. A simple method to achieve this is to project the data using a random linear map, represented by a random matrix \mathbf{A}

built by sampling each entry independently from a standard Gaussian distribution. The following lemma shows that the distribution of the result of applying such a matrix to a fixed deterministic vector is Gaussian.

Lemma 3.1. *Let \mathbf{A} be an $a \times b$ matrix with iid standard Gaussian entries. If $\vec{v} \in \mathbb{R}^b$ is a deterministic vector with unit ℓ_2 norm, then $\mathbf{A}\vec{v}$ is an a -dimensional iid Gaussian vector.*

Proof. By Theorem 2.3, $(\mathbf{A}\vec{v})[i]$, $1 \leq i \leq a$ is Gaussian, since it is the inner product between \vec{v} and the i th row $\mathbf{A}_{i,:}$ (interpreted as a vector in \mathbb{R}^b), which is an iid standard Gaussian vector. The mean of the entry is zero because the mean of $\mathbf{A}_{i,:}$ is zero and the variance equals

$$\text{Var}(\mathbf{A}_{i,:}\vec{v}) = \vec{v}^T \Sigma_{\mathbf{A}_{i,:}} \vec{v} \quad (58)$$

$$= \vec{v}^T I \vec{v} \quad (59)$$

$$= \|\vec{v}\|_2^2 \quad (60)$$

$$= 1, \quad (61)$$

so the entries of $\mathbf{A}\vec{v}$ are all standard Gaussians. Finally, they are independent because each is just a function of a specific row, and all the rows in the matrix are mutually independent. \square

A direct consequence of this result is a non-asymptotic bound on the ℓ_2 norm of $\mathbf{A}\vec{v}$ for any fixed deterministic vector \vec{v} .

Lemma 3.2. *Let \mathbf{A} be a $a \times b$ matrix with iid standard Gaussian entries. For any $\vec{v} \in \mathbb{R}^b$ with unit norm and any $\epsilon \in (0, 1)$*

$$\sqrt{a(1-\epsilon)} \leq \|\mathbf{A}\vec{v}\|_2 \leq \sqrt{a(1+\epsilon)} \quad (62)$$

with probability at least $1 - 2\exp(-a\epsilon^2/8)$.

Proof. The result follows from Theorem 2.7 and Lemma 3.1. \square

Dimensionality-reduction techniques are useful if they preserve the information that we are interested in. In many cases, we would like the projection to conserve the distances between the different data points. This allows us to apply algorithms such as nearest neighbors in the lower-dimensional space. The following lemma guarantees that random projections do not distort the distances between points in a non-asymptotic sense. The result is striking because the lower bound on k – the dimension of the approximate projection – does not depend on n – the ambient dimension of the data – and its dependence on the number of points p in the data set is only logarithmic. The proof is based on the arguments in [3].

Lemma 3.3 (Johnson-Lindenstrauss lemma). *Let \mathbf{A} be a $k \times n$ matrix with iid standard Gaussian entries. Let $\vec{x}_1, \dots, \vec{x}_p \in \mathbb{R}^n$ be any fixed set of p deterministic vectors. For any pair \vec{x}_i, \vec{x}_j and any $\epsilon \in (0, 1)$*

$$(1 - \epsilon) \|\vec{x}_i - \vec{x}_j\|_2^2 \leq \left\| \frac{1}{\sqrt{k}} \mathbf{A} \vec{x}_i - \frac{1}{\sqrt{k}} \mathbf{A} \vec{x}_j \right\|_2^2 \leq (1 + \epsilon) \|\vec{x}_i - \vec{x}_j\|_2^2, \quad (63)$$

with probability at least $\frac{1}{p}$ as long as

$$k \geq \frac{16 \log(p)}{\epsilon^2}. \quad (64)$$

Proof. To prove the result we control the action of the matrix on the normalized difference of the vectors

$$\vec{v}_{ij} := \frac{\vec{x}_i - \vec{x}_j}{\|\vec{x}_i - \vec{x}_j\|_2}, \quad (65)$$

which has unit ℓ_2 -norm unless $\vec{x}_i = \vec{x}_j$ (in which case the norm of the difference is preserved exactly). We denote the event that the norm of the action of \mathbf{A} on \vec{v}_{ij} concentrates around k by

$$\mathcal{E}_{ij} = \{k(1 - \epsilon) < \|\mathbf{A}\vec{v}_{ij}\|_2^2 < k(1 + \epsilon)\} \quad 1 \leq i < p, i < j \leq p.$$

Lemma 3.2 implies that each of the \mathcal{E}_{ij} hold with high probability as long as condition (64) holds

$$P(\mathcal{E}_{ij}^c) \leq \frac{2}{p^2}. \quad (66)$$

However, this is not enough. Our event of interest is the *intersection* of all the \mathcal{E}_{ij} . Unfortunately, the events are dependent (since the vectors are hit by the same matrix), so we cannot just multiply their individual probabilities. Instead, we apply the union bound to control the complement of the intersection.

Theorem 3.4 (Union bound, proof in Section 5.4). *Let S_1, S_2, \dots, S_n be a collection of events in a probability space. Then*

$$P(\cup_i S_i) \leq \sum_{i=1}^n P(S_i). \quad (67)$$

The number of events in the intersection is $\binom{p}{2} = p(p-1)/2$, because that is the number

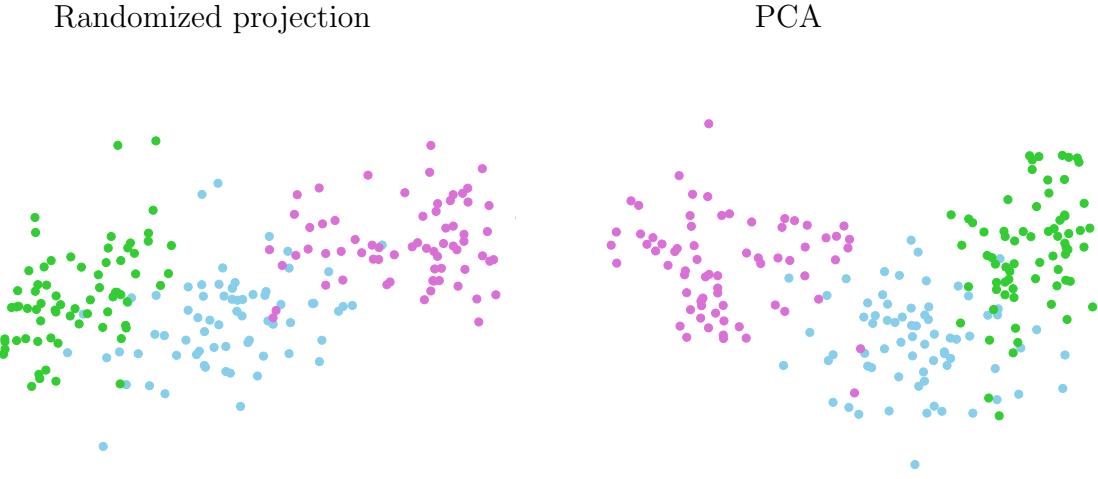


Figure 6: Approximate projection of 7-dimensional vectors describing different wheat seeds onto two random directions. Each color represents a variety of wheat.

of different pairs of vectors in the set $\{\vec{x}_1, \dots, \vec{x}_p\}$. The union bound yields

$$P\left(\bigcap_{i,j} \mathcal{E}_{ij}\right) = 1 - P\left(\bigcup_{i,j} \mathcal{E}_{ij}^c\right) \quad (68)$$

$$\geq 1 - \sum_{i,j} P(\mathcal{E}_{ij}^c) \quad (69)$$

$$\geq 1 - \frac{p(p-1)}{2} \frac{2}{p^2} \quad (70)$$

$$\geq \frac{1}{p}. \quad (71)$$

□

Example 3.5 (Dimensionality reduction via randomized projections). We consider the same data as in Example 3.6 of Lecture Notes 2. Each data point corresponds to a seed with seven features: area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient and length of kernel groove. The seeds belong to three different varieties of wheat: Kama, Rosa and Canadian.² The objective is to project the data onto 2D for visualization. In Figure 6 we compare the result of randomly projecting the data by applying a 2×7 iid standard Gaussian matrix, with the result of PCA-based dimensionality reduction. In terms of keeping the different types of wheat separated, the randomized projection preserves the structure in the data as effectively as PCA. △

²The data can be found at <https://archive.ics.uci.edu/ml/datasets/seeds>.

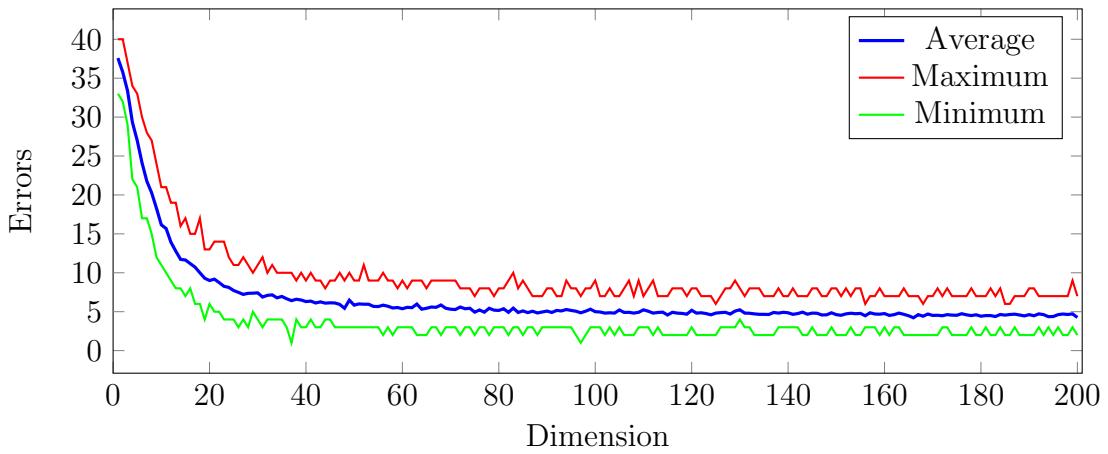


Figure 7: Average, maximum and minimum number of errors (over 50 tries) for nearest-neighbor classification after a randomized dimensionality reduction for different dimensions.



Figure 8: Results of nearest-neighbor classification combined with randomized dimensionality reduction of dimension 50 for four of the people in Example 3.6. The assignments of the first two examples are correct, but the other two are wrong.

Example 3.6 (Nearest neighbors after random projection). The nearest neighbors algorithm for classification (Algorithm 4.2 in Lecture Notes 1) requires computing n distances in an m -dimensional space (where m is the number of features) to classify each new example. The computational cost is $\mathcal{O}(nm)$, so if we need to classify p points the total cost is $\mathcal{O}(nmp)$. If we perform a random projection of each of the points onto a lower-dimensional space k before classifying them, then the computational cost is:

- kmn operations to project the training data using a $k \times m$ iid standard Gaussian matrix.
- kmp operations to project each point in the test set using the same matrix.
- knp to perform nearest-neighbor classification in the lower-dimensional space.

The overall cost is $\mathcal{O}(kp \max\{m, n\})$, which is a significant reduction from $\mathcal{O}(nmp)$. It is also more efficient than the PCA-based approach of Example 3.5 in Lecture Notes 2, which includes an additional $\mathcal{O}(mn \min\{m, n\})$ step to compute an SVD.

Figure 7 shows the accuracy of the algorithm on the same data as Example 4.3 in Lecture Notes 1. A similar average precision as in the ambient dimension (5 errors out of 40 test images compared to 4 out of 40) is achieved for a dimension of $k = 50$. Figure 8 shows some examples of the projected data represented in the original m -dimensional space along with their nearest neighbors in the k -dimensional space. \triangle

3.2 Singular values

In this section we analyze the singular values of matrices with iid standard Gaussian entries. In particular we consider an $k \times n$ matrix \mathbf{A} where $n > k$. Numerically, we observe that as n grows, all k singular values converge to \sqrt{n} , as shown in Figure 9. As a result,

$$\mathbf{A} \approx U(\sqrt{n} I) V^T = \sqrt{n} U V^T, \quad (72)$$

i.e. \mathbf{A} is close to an orthogonal matrix. Geometrically, this implies that if we generate a fixed number of iid Gaussian vectors at increasing ambient dimensions, the vectors will tend to be almost orthogonal as the dimension grows.

The following result establishes a non-asymptotic bound on the singular values using a covering number argument from [1] that can be applied to other distributions and situations. See also [6] for some excellent notes on high-dimensional probability techniques in this spirit.

Theorem 3.7 (Singular values of a Gaussian matrix). *Let \mathbf{A} be a $n \times k$ matrix with iid standard Gaussian entries such that $n > k$. For any fixed $\epsilon > 0$, the singular values of \mathbf{A} satisfy*

$$\sqrt{n}(1 - \epsilon) \leq \sigma_k \leq \sigma_1 \leq \sqrt{n}(1 + \epsilon) \quad (73)$$

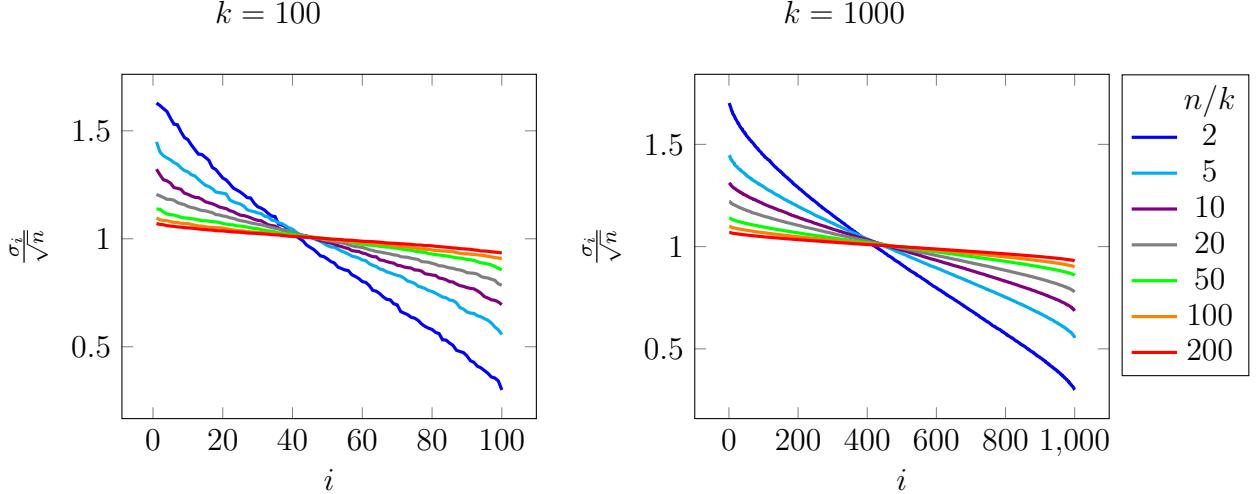


Figure 9: Singular values of $n \times k$ matrices with iid standard Gaussian entries for different values of k and n .

with probability at least $1 - 1/k$ as long as

$$n > \frac{64k}{\epsilon^2} \log \frac{12}{\epsilon}. \quad (74)$$

By Theorem 2.7 in Lecture Notes 2, the bounds on the singular values are equivalent to the following bounds

$$\sqrt{n}(1 - \epsilon) < \|\mathbf{A}\vec{v}\|_2 < \sqrt{n}(1 + \epsilon) \quad (75)$$

where \vec{v} is *any* vector in the k -dimensional sphere \mathcal{S}^{k-1} , which contains the unit- ℓ_2 -norm vectors in \mathbb{R}^k . This set has infinite cardinality, so we cannot apply the union bound to establish the bounds as in the proof of the Johnson-Lindenstrauss lemma. To overcome this obstacle, we consider a set, called an ϵ -net, which covers the sphere in the sense that every other point is not too far from one of its elements. We prove that the bounds hold on the net using the union bound and then establish that as a result they hold for the whole sphere.

Definition 3.8 (ϵ -net). *An ϵ -net of a set $\mathcal{X} \subseteq \mathbb{R}^k$ is a subset $\mathcal{N}_\epsilon \subseteq \mathcal{X}$ such that for every vector $\vec{x} \in \mathcal{X}$ there exists $\vec{y} \in \mathcal{N}_\epsilon$ for which*

$$\|\vec{x} - \vec{y}\|_2 \leq \epsilon. \quad (76)$$

Figure 10 shows an ϵ -net for the two-dimensional sphere \mathcal{S}^1 . The smallest possible number of points in the ϵ -net of a set is called its covering number.

Definition 3.9 (Covering number). *The covering number $\mathcal{N}(\mathcal{X}, \epsilon)$ of a set \mathcal{X} at scale ϵ is the minimal cardinality of an ϵ -net of \mathcal{X} , or equivalently the minimal number of balls of radius ϵ with centers in \mathcal{X} required to cover \mathcal{X} .*

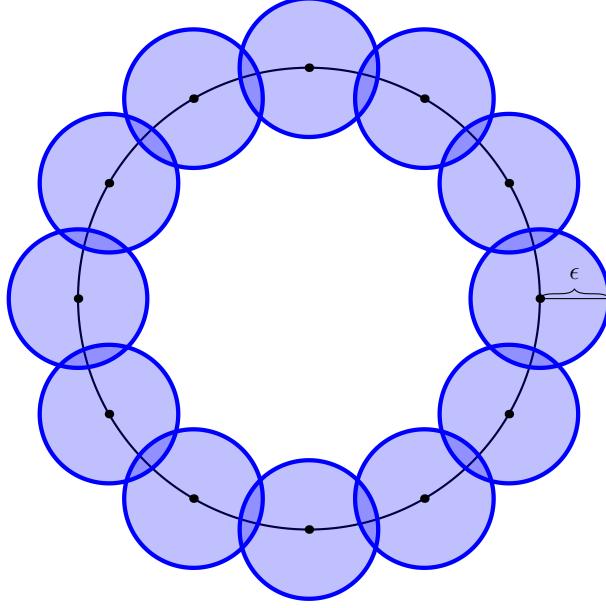


Figure 10: ϵ -net for the two-dimensional sphere S^1 , which is just a circle.

The following theorem, proved in Section 5.5 of the appendix, provides a bound for the covering number of the k -dimensional sphere S^{k-1} .

Theorem 3.10 (Covering number of a sphere). *The covering number of the n -dimensional sphere S^{k-1} at scale ϵ satisfies*

$$\mathcal{N}(S^{k-1}, \epsilon) \leq \left(\frac{2 + \epsilon}{\epsilon} \right)^k \leq \left(\frac{3}{\epsilon} \right)^k. \quad (77)$$

Let $\epsilon_1 := \epsilon/4$ and $\epsilon_2 := \epsilon/2$. Consider an ϵ_1 -net \mathcal{N}_{ϵ_1} of S^{k-1} . We define the event

$$\mathcal{E}_{\vec{v}, \epsilon_2} := \{n(1 - \epsilon_2) \|\vec{v}\|_2^2 \leq \|\mathbf{A}\vec{v}\|_2^2 \leq n(1 + \epsilon_2) \|\vec{v}\|_2^2\}. \quad (78)$$

By Lemma 3.2 for any fixed $\vec{v} \in \mathbb{R}^k$ $P(\mathcal{E}_{\vec{v}, \epsilon_2}^c) \leq 2 \exp(-n\epsilon^2/32)$, so by the union bound

$$P\left(\bigcup_{\vec{v} \in \mathcal{N}_{\epsilon_1}} \mathcal{E}_{\vec{v}, \epsilon_2}^c\right) \leq \sum_{\vec{v} \in \mathcal{N}_{\epsilon_1}} P(\mathcal{E}_{\vec{v}, \epsilon_2}^c) \quad (79)$$

$$\leq |\mathcal{N}_{\epsilon_1}| P(\mathcal{E}_{\vec{v}, \epsilon_2}^c) \quad (80)$$

$$\leq 2 \left(\frac{12}{\epsilon} \right)^k \exp\left(-\frac{n\epsilon^2}{32}\right) \quad (81)$$

$$\leq \frac{1}{k} \quad \text{if (74) holds.} \quad (82)$$

Now, to finish the proof we need to show that if $\bigcup_{\vec{v} \in \mathcal{N}_{\epsilon_1}} \mathcal{E}_{\vec{v}, \epsilon_2}^c$ holds then the bound holds for every element in S^{k-1} , not only for those in the ϵ_1 -net. For any arbitrary vector

$\vec{x} \in \mathcal{S}^{k-1}$ on the sphere there exists a vector in the $\epsilon/4$ -covering set $\vec{v} \in \mathcal{N}(\mathcal{X}, \epsilon_1)$ such that $\|\vec{x} - \vec{v}\|_2 \leq \epsilon/4$. By the triangle inequality this implies

$$\|\mathbf{A}\vec{x}\|_2 \leq \|\mathbf{A}\vec{v}\|_2 + \|\mathbf{A}(\vec{x} - \vec{v})\|_2 \quad (83)$$

$$\leq \sqrt{n} \left(1 + \frac{\epsilon}{2}\right) + \|\mathbf{A}(\vec{x} - \vec{v})\|_2 \quad \text{assuming } \cup_{\vec{v} \in \mathcal{N}_{\epsilon_1}} \mathcal{E}_{\vec{v}, \epsilon_2}^c \text{ holds} \quad (84)$$

$$\leq \sqrt{n} \left(1 + \frac{\epsilon}{2}\right) + \sigma_1 \|\vec{x} - \vec{v}\|_2 \quad \text{by Theorem 2.7 in Lecture Notes 2} \quad (85)$$

$$\leq \sqrt{n} \left(1 + \frac{\epsilon}{2}\right) + \frac{\sigma_1 \epsilon}{4}. \quad (86)$$

By Theorem 2.7 in Lecture Notes 2 σ_1 is the smallest upper bound on $\|\mathbf{A}\vec{x}\|_2$ for all \vec{x} in the sphere, so the bound in equation (86) cannot be smaller:

$$\sigma_1 \leq \sqrt{n} \left(1 + \frac{\epsilon}{2}\right) + \frac{\sigma_1 \epsilon}{4}, \quad (87)$$

so that

$$\sigma_1 \leq \sqrt{n} \left(\frac{1 + \epsilon/2}{1 - \epsilon/4}\right) \quad (88)$$

$$= \sqrt{n} \left(1 + \epsilon - \frac{\epsilon(1 - \epsilon)}{4 - \epsilon}\right) \quad (89)$$

$$\leq \sqrt{n}(1 + \epsilon). \quad (90)$$

The lower bound on σ_k follows from a similar argument combined with (90). By the triangle inequality

$$\|\mathbf{A}\vec{x}\|_2 \geq \|\mathbf{A}\vec{v}\|_2 - \|\mathbf{A}(\vec{x} - \vec{v})\|_2 \quad (91)$$

$$\geq \sqrt{n} \left(1 - \frac{\epsilon}{2}\right) - \|\mathbf{A}(\vec{x} - \vec{v})\|_2 \quad \text{assuming } \cup_{\vec{v} \in \mathcal{N}_{\epsilon_1}} \mathcal{E}_{\vec{v}, \epsilon_2}^c \text{ holds} \quad (92)$$

$$\geq \sqrt{n} \left(1 - \frac{\epsilon}{2}\right) - \sigma_1 \|\vec{x} - \vec{v}\|_2 \quad \text{by Theorem 2.7 in Lecture Notes 2} \quad (93)$$

$$\geq \sqrt{n} \left(1 - \frac{\epsilon}{2}\right) - \frac{\epsilon}{4} \sqrt{n}(1 + \epsilon) \quad \text{by (90)} \quad (94)$$

$$= \sqrt{n}(1 - \epsilon). \quad (95)$$

By Theorem 2.7 in Lecture Notes 2 σ_k is the largest lower bound on $\|\mathbf{A}\vec{x}\|_2$ for all \vec{x} on the sphere, so $\sigma_k \geq \sqrt{n}(1 - \epsilon)$ as long as $\cup_{\vec{v} \in \mathcal{N}_{\epsilon_1}} \mathcal{E}_{\vec{v}, \epsilon_2}^c$ holds.

4 Randomized singular-value decomposition

4.1 Fast SVD

In this section we describe an algorithm to compute the SVD of a low-rank matrix very efficiently assuming that we have access to an orthonormal basis of its column space.

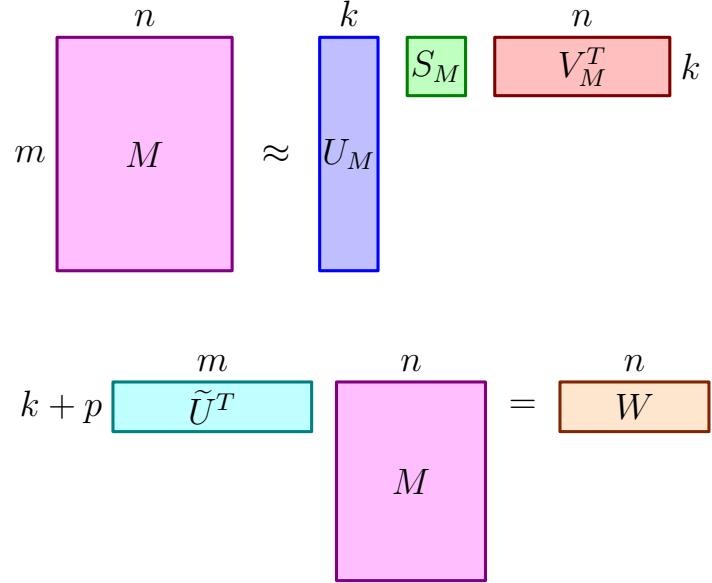


Figure 11: Sketch of the matrices in Algorithm 4.1.

Algorithm 4.1 (Fast SVD). *Given a matrix $M \in \mathbb{R}^{m \times n}$ which is well-approximated as a k -rank matrix:*

1. *Find a matrix $\tilde{U} \in \mathbb{R}^{m \times (k+p)}$ with $k+p$ orthonormal columns that approximately span the column space of M .*
2. *Compute $W \in \mathbb{R}^{(k+p) \times n}$ defined by $W := \tilde{U}^T M$.*
3. *Compute the SVD of $W = U_W S_W V_W^T$.*
4. *Output $U := (\tilde{U} U_W)_{:,1:k}$, $S := (S_W)_{1:k,1:k}$ and $V := (V_W)_{:,1:k}$ as the SVD of M .*

Figure 11 shows the dimensions of the matrices. Computing the SVD of W has complexity $\mathcal{O}(k^2n)$, so overall the complexity of the algorithm is governed by the second step which has complexity $\mathcal{O}(kmn)$. This is a dramatic reduction from $\mathcal{O}(mn \min\{m, n\})$.

The following lemma establishes that the algorithm works in the idealized situation where the matrix is exactly low rank and we have access to an orthonormal basis of its column space.

Lemma 4.2. *Algorithm 4.1 outputs the SVD of a matrix $M \in \mathbb{R}^{m \times n}$ as long as M is rank k and \tilde{U} spans its column space.*

Proof. If \tilde{U} spans the column space of M then

$$M = \tilde{U} \tilde{U}^T M \tag{96}$$

$$= \tilde{U} W \tag{97}$$

$$= \tilde{U} U_W S_W V_W^T, \tag{98}$$

where $U := \tilde{U}U_W$ is an $m \times k$ matrix with orthonormal columns since

$$U^T U = U_W^T \tilde{U}^T \tilde{U} U_W \quad (99)$$

$$= U_W^T U_W \quad (100)$$

$$= I, \quad (101)$$

$S_W \in \mathbb{R}^{k \times k}$ is a diagonal matrix with nonnegative entries and $V_W \in \mathbb{R}^{n \times k}$ has orthonormal columns. We conclude that the output of Algorithm 4.1 is a valid SVD of M . \square

The following sections describe two methods for estimating the column space based on randomization (i.e., step 1 of Algorithm 4.1). In practice, most matrices of interest will only be approximately low rank. In that case, the performance of the column-approximation algorithms depends on the gap between the k first singular values and the $k+1$ th. To enhance the performance, a popular preprocessing procedure is to use *power iterations* to increase the gap. Instead of M , the idea is to apply Step 1 of Algorithm 4.1 to

$$\widetilde{M} := (MM^T)^q M, \quad (102)$$

for a small integer q . Expressing M in terms of its SVD, we have

$$\widetilde{M} = (U_M S_M^2 U_M^T)^q U_M S_M V_M^T \quad (103)$$

$$= U_M S_M^{2q+1} V_M^T. \quad (104)$$

\widetilde{M} has the same singular vectors as M but its singular values are raised to the power of $2q+1$, so the gap between small and large singular values is amplified. The idea is very similar to the power method for computing eigenvectors (Algorithm 4.4 in Lecture Notes 2). More details on the power iterations will be given in the next two subsections.

4.2 Randomized column-space approximation

Random projections make it possible to estimate the column space of a low-rank matrix very efficiently. Here we just outline the method and provide some intuition. We refer to [4, 5] for a more detailed description and theoretical guarantees.

Algorithm 4.3 (Randomized column-space approximation). *Given a matrix $M \in \mathbb{R}^{m \times n}$ which is well-approximated as a k -rank matrix:*

1. Create an $n \times (k+p)$ iid standard Gaussian matrix \mathbf{A} , where p is a small integer (e.g. 5).
2. Compute the $m \times (k+p)$ matrix $\mathbf{B} = M\mathbf{A}$.
3. Compute an orthonormal basis for $\text{col}(\mathbf{B})$ and output them as a matrix $\widetilde{\mathbf{U}} \in \mathbb{R}^{m \times (k+p)}$.



Figure 12: Four randomly selected frames from the video in Example 4.7.

Consider the matrix

$$\mathbf{B} = M\mathbf{A} \quad (105)$$

$$= U_M S_M V_M^T \mathbf{A} \quad (106)$$

$$= U_M S_M \mathbf{C}. \quad (107)$$

If M is exactly low rank, by Theorem 2.3 \mathbf{C} is a $k \times (k + p)$ iid standard Gaussian matrix since $V_M^T V_M = I$. In that case the column space of \mathbf{B} is the same as that of M because \mathbf{C} is full rank with high probability. When M is only approximately low rank, then \mathbf{C} is a $\min\{m, n\} \times (k + p)$ iid standard Gaussian matrix. Surprisingly, for p equal to a small integer, the product with \mathbf{C} conserves the subspace corresponding to the largest k singular values with high probability, as long as the $k + 1$ th singular values is sufficiently small. See the seminal paper [4] for more details.

We can improve the accuracy of Algorithm 4.3 by using power iterations as detailed below.

Algorithm 4.4 (Power Iterations for Randomized column-space approximation). *Given a matrix $M \in \mathbb{R}^{m \times n}$ which is well-approximated as a k -rank matrix:*

1. Create an $n \times (k + p)$ iid standard Gaussian matrix \mathbf{A} , where p is a small integer (e.g. 5).
2. Compute the $m \times (k + p)$ matrix $\mathbf{B} = M\mathbf{A}$ and let $\tilde{\mathbf{U}}_0$ denote the matrix formed by orthonormalizing the columns of \mathbf{B} .
3. For i from 1 to q (inclusive) :
 - (a) Let $\tilde{\mathbf{F}}_i$ be formed by orthonormalizing the columns of $M^T \tilde{\mathbf{U}}_{i-1}$.
 - (b) Let $\tilde{\mathbf{U}}_i$ be formed by orthonormalizing the columns of $M \tilde{\mathbf{F}}_i$.
4. Output $\tilde{\mathbf{U}}_q$.

Example 4.5 (Randomized SVD of a video). In this example we consider a data set that consists of a video with 160 1080×1920 frames. Four sample frames are show in 12. We interpret each frame as a vector in $\mathbb{R}^{20,736,000}$. The matrix obtained by stacking these vectors as columns is approximately low rank due to the correlation between the frames,

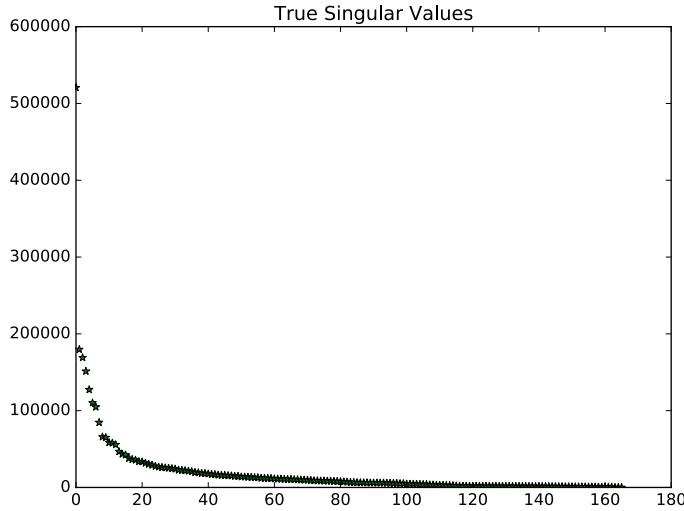


Figure 13: Singular values of the $160 \times 20,736,000$ matrix in Example 4.7.

as can be seen in Figure 13. Computing the SVD of this matrix takes 12 seconds on our machine. Applying Algorithm 4.1 combined with Algorithm 4.4 we obtain a rank-10 approximation in 5.8 seconds. If we consider a larger video with 691 frames, computing the SVD takes 281.1 seconds, whereas the randomized algorithm only takes 10.4 seconds. Figure 14 shows a comparison between the *true* left singular values and the estimate produced by the randomized algorithm, using power iterations with parameter $q = 2$ and setting $p = 7$ in Algorithm 4.4. \triangle

4.3 Random column selection

An alternative procedure for estimating the column space of a low-rank matrix is to randomly select a subset of columns and obtain an orthonormal basis from them.

Algorithm 4.6 (Randomized column-space approximation). *Given a matrix $M \in \mathbb{R}^{m \times n}$ which is well-approximated as a k -rank matrix:*

1. *Select a random subset of column indices $\mathcal{I} := \{\mathbf{i}_1, \mathbf{i}_2, \dots, \mathbf{i}_{k'}\}$ with $k' \geq k$.*
2. *Compute an orthonormal basis for the columns of the submatrix corresponding to \mathcal{I} :*

$$M_{\mathcal{I}} := [M_{:, \mathbf{i}_1} \quad M_{:, \mathbf{i}_2} \quad \cdots \quad M_{:, \mathbf{i}_{k'}}] \quad (108)$$

and output them as a matrix $\tilde{\mathbf{U}} \in \mathbb{R}^{m \times k'}$.

The random submatrix $M_{\mathcal{I}}$ can be expressed in terms of the left singular vectors and singular values of M , and a submatrix of the right-singular-vector matrix,

$$M_{\mathcal{I}} = U_M S_M (V_M)_{\mathcal{I}}. \quad (109)$$

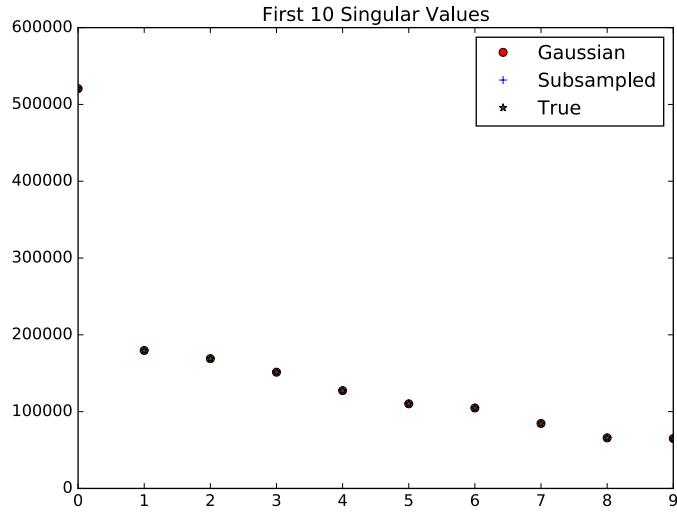


Figure 14: First 10 singular values of the $160 \times 20,736,000$ matrix in Example 4.7 and their estimate using Algorithm 4.1 combined with Algorithm 4.4 and Algorithm 4.6 with power iteration. The estimates are almost perfect.

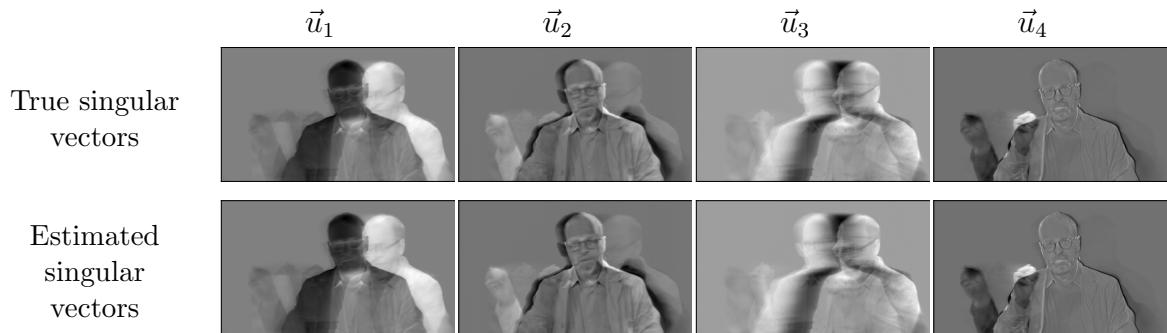


Figure 15: The first 4 left singular vectors of the movie computed using the standard SVD algorithm, and the randomized Algorithm 4.4.

In order for the algorithm to capture the column space of M , we need $(V_M)_{\mathcal{I}}$ to be full rank (and ideally to not have very small singular values). Moreover, if the matrix is only approximately low rank, we need the right singular vectors corresponding to large singular values to be *spread out* so that they are not missed when we subsample. Let us illustrate this with a simple example. Consider the rank-2 matrix

$$M := \begin{bmatrix} -3 & 2 & 2 & 2 \\ 3 & 2 & 2 & 2 \\ -3 & 2 & 2 & 2 \\ 3 & 2 & 2 & 2 \end{bmatrix} \quad (110)$$

and its SVD

$$M = U_M S_M V_M^T = \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & 0.5 \\ 0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 6.9282 & 0 \\ 0 & 6 \end{bmatrix} \begin{bmatrix} 0 & 0.577 & 0.577 & 0.577 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (111)$$

(112)

Any submatrix of columns that do not include the first one will have a column space that only consists of the first left singular vector. For example if $\mathcal{I} = \{2, 3\}$

$$M_{\mathcal{I}} = \begin{bmatrix} 2 & 2 \\ 2 & 2 \\ 2 & 2 \\ 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} 6.9282 [0.577 \ 0.577]. \quad (113)$$

Column subsampling tends to ignore left singular vectors corresponding to sparse (or approximately sparse) right singular vectors. Depending on the application, this may not be a disadvantage: sparse right singular vectors arise due to columns in M that are almost orthogonal to every other column and can consequently be interpreted as outliers. In contrast, column subsampling preserves the part of the column space corresponding to spread-out right singular vectors with high probability (see [2] for a theoretical characterization of this phenomenon).

To use power iteration with column subsampling, define $\mathbf{B} := \tilde{\mathbf{U}}$ obtained from 4.6, and apply steps 3,4 from Algorithm 4.4. Although power iteration can be helpful, it will not alleviate the issue with sparse columns described above.

Example 4.7 (Randomized SVD of a video using subset of columns). In this example we consider the same data set as in Example 4.7. We estimate a rank-10 approximation of the $160 \times 20,736,000$ matrix by randomly selecting $k' = 17$ columns as in Algorithm 4.6 and then applying Algorithm 4.1. The running time is 5.2 seconds, even faster than if we use Algorithm 4.1. Figure 14 shows a comparison between the *true* left singular values and the estimate produced by the randomized algorithm, using power iterations with parameter $q = 2$. In Figure 16 we show the top 4 estimated singular vectors compared to the true ones. The approximation is very precise, indicating that the first 10 right singular vectors of the matrix are not sparse or approximately sparse. \triangle

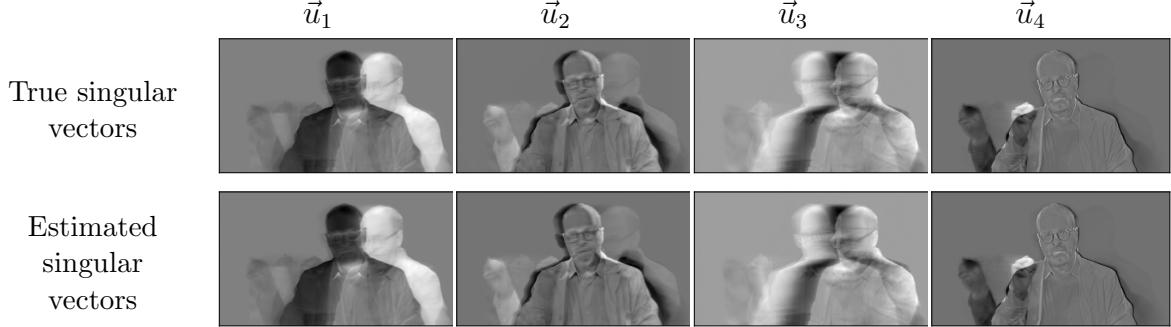


Figure 16: The first 4 left singular vectors of the movie computed using the standard SVD algorithm, and the randomized Algorithm 4.6.

5 Proofs

5.1 Proof of Theorem 2.6

Consider the indicator variable $1_{\mathbf{x} \geq a}$. We have

$$\mathbf{x} - a 1_{\mathbf{x} \geq a} \geq 0. \quad (114)$$

In particular its expectation is nonnegative (as it is the sum or integral of a nonnegative quantity over the positive real line). By linearity of expectation and the fact that $1_{\mathbf{x} \geq a}$ is a Bernoulli random variable with expectation $P(\mathbf{x} \geq a)$ we have

$$E(\mathbf{x}) \geq a E(1_{\mathbf{x} \geq a}) = a P(\mathbf{x} \geq a). \quad (115)$$

5.2 Proof of Lemma 2.8

$$E(\exp(t\mathbf{x}^2)) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\frac{u^2}{2}\right) \exp(tu^2) du \quad (116)$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\frac{(1-2t)u^2}{2}\right) du \quad \text{finite for } 1-2t > 0 \quad (117)$$

$$= \frac{1}{\sqrt{2\pi(1-2t)}} \int_{-\infty}^{\infty} \exp\left(-\frac{v^2}{2}\right) dv \quad \text{change of variables } v = (2-t)u \quad (118)$$

5.3 Proof of (35)

A very similar argument to the one that yields (39) gives

$$P(\mathbf{y} < a') = P(\exp(-t'\mathbf{y}) > \exp(-a't')) \quad (119)$$

$$\leq \exp(a't') \prod_{i=1}^k \mathbb{E}(\exp(-t' \mathbf{x}_i^2)). \quad (120)$$

Setting $t' = t$ in (41), we have

$$\mathbb{E}(\exp(-t' \mathbf{x}^2)) = \frac{1}{\sqrt{1+2t'}}. \quad (121)$$

This implies

$$P(\mathbf{y} < a') \leq \frac{\exp(a't')}{(1+2t')^{\frac{k}{2}}}. \quad (122)$$

Setting

$$t' := -\frac{1}{2} + \frac{1}{2(1-\epsilon)}, \quad (123)$$

$$a' := k(1-\epsilon) \quad (124)$$

we have

$$P(\mathbf{y} < k(1-\epsilon)) \leq (1-\epsilon)^{\frac{k}{2}} \exp\left(\frac{k\epsilon}{2}\right) \quad (125)$$

$$= \exp\left(-\frac{k}{2}(-\epsilon - \log(1-\epsilon))\right). \quad (126)$$

The function $h(x) := -x - \frac{x^2}{2} - \log(1-x)$ is nonnegative between 0 and 1 (the derivative is nonnegative and $g(0) = 0$). We conclude that

$$P(\mathbf{y} < k(1-\epsilon)) \leq \exp\left(-\frac{k\epsilon^2}{2}\right) \quad (127)$$

$$\leq \exp\left(-\frac{k\epsilon^2}{8}\right). \quad (128)$$

5.4 Proof of Theorem 3.4

Let us define the sets:

$$\tilde{S}_i = S_i \cap \cap_{j=1}^{i-1} S_j^c. \quad (129)$$

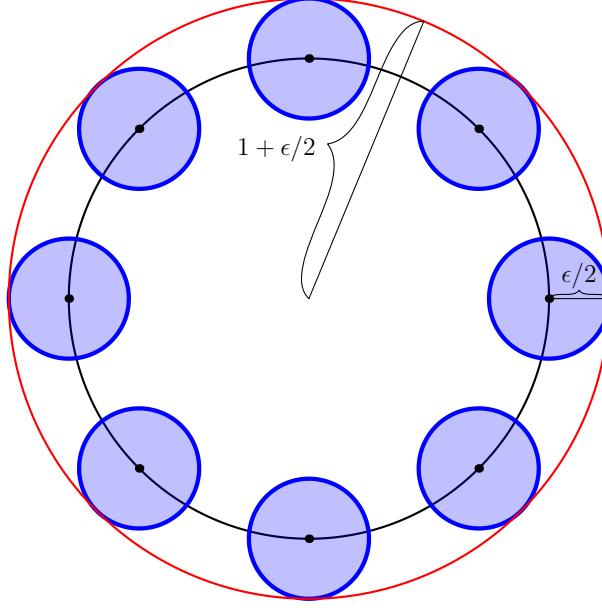


Figure 17: Sketch of the proof of Theorem 3.10 in two dimensions. $\mathcal{B}_{1+\epsilon/2}^k(\vec{0})$ is the big red circle. The smaller shaded circles correspond to $\mathcal{B}_{\epsilon/2}^k(\vec{x})$ for each \vec{x} in the ϵ -net.

It is straightforward to show by induction that $\cup_{j=1}^n S_j = \cup_{j=1}^n \tilde{S}_j$ for any n , so $\cup_i S_i = \cup_i \tilde{S}_i$. The sets $\tilde{S}_1, \tilde{S}_2, \dots$ are disjoint by construction, so

$$P(\cup_i S_i) = P(\cup_i \tilde{S}_i) = \sum_i P(\tilde{S}_i) \quad (130)$$

$$\leq \sum_i P(S_i) \quad \text{because } \tilde{S}_i \subseteq S_i. \quad (131)$$

5.5 Proof of Theorem 3.10

We construct an ϵ -covering set $\mathcal{N}_\epsilon \subseteq \mathcal{S}^{k-1}$ recursively:

- We initialize \mathcal{N}_ϵ to the empty set.
- We choose a point $\vec{x} \in \mathcal{S}^{k-1}$ such that $\|\vec{x} - \vec{y}\|_2 > \epsilon$ for any $\vec{y} \in \mathcal{N}_\epsilon$. We add \vec{x} to \mathcal{N}_ϵ until there are no points in \mathcal{S}^{k-1} that are ϵ away from any point in \mathcal{N}_ϵ .

This algorithm necessarily ends in a finite number of steps because the n -dimensional sphere is compact (otherwise we would have an infinite sequence such that no subsequence converges).

Now, let us consider the balls of radius $\epsilon/2$ centered at each of the points in \mathcal{N}_ϵ . These balls do not intersect since their centers are at least ϵ apart and they are all inside the

ball of radius $1 + \epsilon/2$ centered at the origin $\vec{0}$ because $\mathcal{N}_\epsilon \subseteq \mathcal{S}^{k-1}$. This means that

$$\text{Vol} \left(\mathcal{B}_{1+\epsilon/2}^k \left(\vec{0} \right) \right) \geq \text{Vol} \left(\cup_{\vec{x} \in \mathcal{N}_\epsilon} \mathcal{B}_{\epsilon/2}^k \left(\vec{x} \right) \right) \quad (132)$$

$$= |\mathcal{N}_\epsilon| \text{Vol} \left(\mathcal{B}_{\epsilon/2}^k \left(\vec{0} \right) \right) \quad (133)$$

where $\mathcal{B}_r^k(\vec{x})$ is the ball of radius r centered at \vec{x} . By multivariable calculus

$$\text{Vol} \left(\mathcal{B}_r^k \left(\vec{0} \right) \right) = r^k \text{Vol} \left(\mathcal{B}_1^k \left(\vec{0} \right) \right), \quad (134)$$

so (132) implies

$$(1 + \epsilon/2)^k \geq |\mathcal{N}_\epsilon| (\epsilon/2)^k. \quad (135)$$

References

- [1] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.
- [2] J. Chiu and L. Demanet. Sublinear randomized algorithms for skeleton decompositions. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1361–1383, 2013.
- [3] S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.
- [4] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.
- [5] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.
- [6] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Lecture Notes 4: The Frequency Domain

1 Fourier representations

1.1 Complex exponentials

The complex exponential or complex sinusoids is a complex-valued function. Its real part is a cosine and its imaginary part is a sine.

Definition 1.1 (Complex sinusoid).

$$\exp(ix) := \cos(x) + i \sin(x). \quad (1)$$

The argument x of a complex sinusoid is known as its *phase*. For any value of the phase, the magnitude of the complex exponential is equal to one,

$$|\exp(ix)|^2 = \cos(x)^2 + \sin(x)^2 = 1. \quad (2)$$

If we set the phase of a complex sinusoid to equal ft for fixed f , then the sinusoid is periodic with period $1/f$ and the parameter f is known as the *frequency* of the sinusoid.

Definition 1.2 (Frequency). *A complex sinusoid with frequency f is of the form*

$$h_f(t) := \exp(i2\pi ft). \quad (3)$$

Lemma 1.3 (Periodicity). *The complex exponential $h_f(t)$ with frequency f is periodic with period $1/f$.*

Proof.

$$h_f(t + 1/f) = \exp\left(i2\pi f\left(t + \frac{1}{f}\right)\right) \quad (4)$$

$$= \exp(i2\pi ft) \exp(i2\pi) \quad (5)$$

$$= h_f(t). \quad (6)$$

□

Figure 1 shows a complex sinusoid with frequency 1 together with its real and imaginary parts.

If we consider a unit interval, complex sinusoids with integer frequencies form an orthonormal set. The choice of interval is arbitrary, since all these sinusoids are periodic with period $1/k$ for some integer k and hence also with period one.

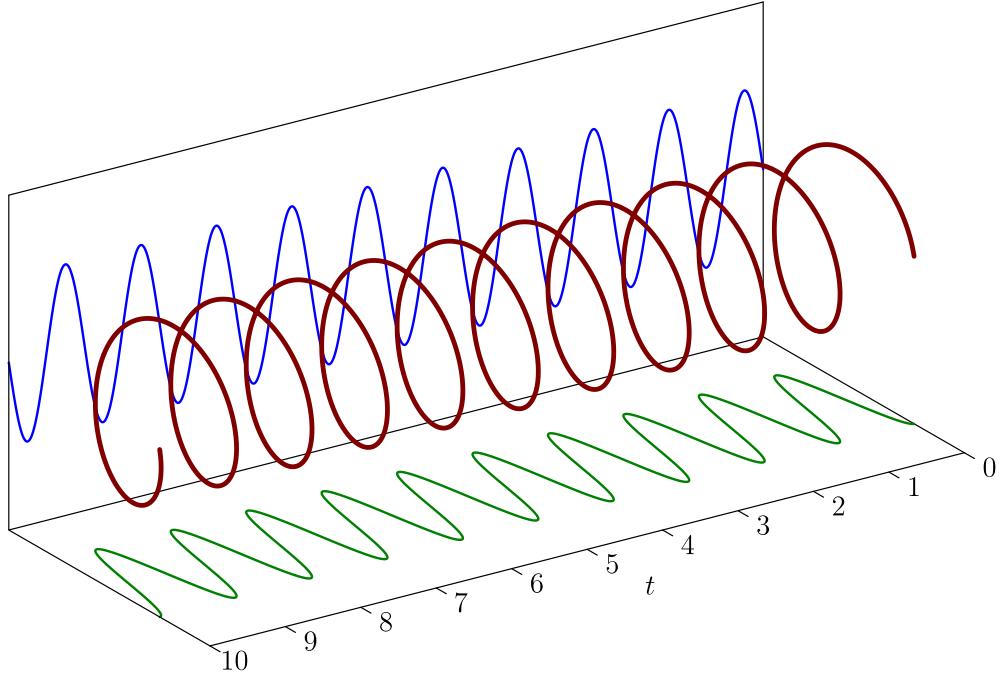


Figure 1: Complex sinusoid h_1 (dark red) plotted between 0 and 10. Its real (green) corresponds to a cosine function and its imaginary part (blue) to a sine function.

Lemma 1.4. *The family of complex sinusoids with integer frequencies*

$$h_k(t) := \exp(i2\pi kt), \quad k \in \mathbb{Z}, \quad (7)$$

is an orthonormal set of functions on the unit interval $[-1/2, 1/2]$.

Proof. Each function has unit \mathcal{L}_2 norm,

$$\|h_k\|_{\mathcal{L}_2}^2 = \int_{-1/2}^{1/2} |h_k(t)|^2 dt \quad (8)$$

$$= 1, \quad (9)$$

and they are all orthogonal

$$\langle h_k, h_j \rangle = \int_{-1/2}^{1/2} h_k(t) \overline{h_j(t)} dt \quad (10)$$

$$= \int_{-1/2}^{1/2} \exp(i2\pi(k-j)t) dt \quad (11)$$

$$= \frac{\exp(i\pi(k-j)) - \exp(-i\pi(k-j))}{i2\pi} \quad (12)$$

$$= \frac{\cos(\pi(k-j)) - \cos(\pi(k-j))}{i2\pi} \quad (13)$$

$$= 0. \quad (14)$$

□

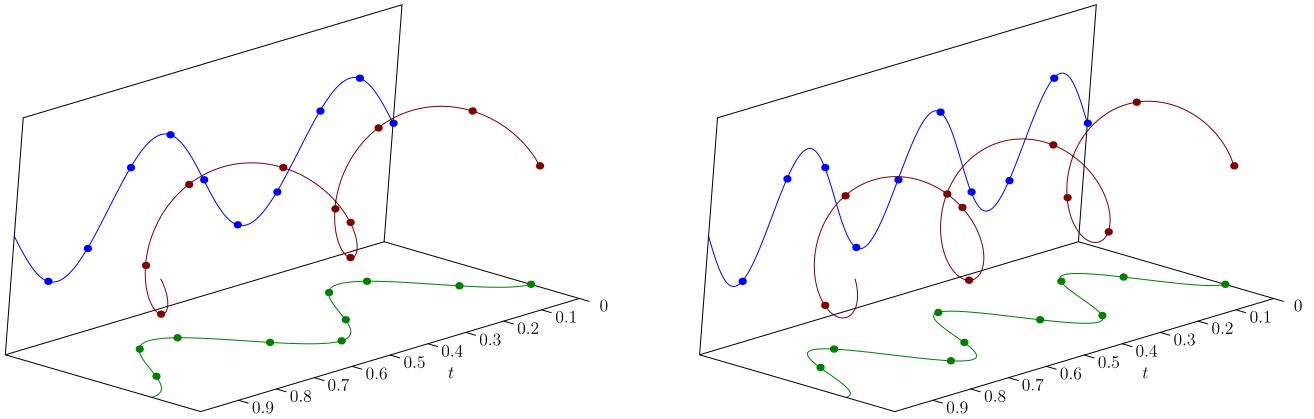


Figure 2: Discrete complex sinusoids $\vec{h}_2^{[10]}$ (left) and $\vec{h}_3^{[10]}$ (right) in red along with their real (green) and imaginary (blue) parts.

Complex sinusoids on the unit interval can be discretized by sampling their values at n equispaced points. This yields a family of discrete complex sinusoids.

Definition 1.5 (Discrete complex exponential). *The discrete complex sinusoid $\vec{h}_k^{[n]} \in \mathbb{C}^n$ with integer frequency k is defined as*

$$\vec{h}_k^{[n]} [j] := \exp\left(\frac{i2\pi kj}{n}\right), \quad 0 \leq j, k \leq n-1 \quad (15)$$

The discrete complex sinusoid is periodic in j with period n/k and in k with period n/j . In particular, they are also periodic with period n , so we only consider values in an interval of length n . Here we have fixed the interval to be from 0 to $n-1$ but this is arbitrary. Figure 2 shows $\vec{h}_2^{[10]}$ and $\vec{h}_3^{[10]}$ along with their real and imaginary parts

Discrete complex sinusoids form an orthonormal basis of \mathbb{C}^n if we scale them by $1/\sqrt{n}$.

Lemma 1.6 (Orthonormal sinusoidal basis). *The discrete complex exponentials $\frac{1}{\sqrt{n}}\vec{h}_0^{[n]}, \dots, \frac{1}{\sqrt{n}}\vec{h}_{n-1}^{[n]}$ form an orthonormal basis of \mathbb{C}^n .*

Proof. Each vector has ℓ_2 norm equal to \sqrt{n} ,

$$\left\| \vec{h}_k^{[n]} \right\|_2^2 = \sum_{j=0}^{n-1} \left| \vec{h}_k^{[n]} [j] \right|^2 \quad (16)$$

$$= \sum_{j=0}^{n-1} 1 \quad (17)$$

$$= n, \quad (18)$$

and they are all orthogonal

$$\left\langle \vec{h}_k^{[n]}, \vec{h}_l^{[n]} \right\rangle = \sum_{j=0}^{n-1} \vec{h}_k^{[n]} [j] \overline{\vec{h}_l^{[n]} [j]} \quad (19)$$

$$= \sum_{j=0}^{n-1} \exp \left(\frac{i2\pi(k-l)j}{n} \right) \quad (20)$$

$$= \frac{1 - \exp \left(\frac{i2\pi(k-l)n}{n} \right)}{1 - \exp \left(\frac{i2\pi(k-l)}{n} \right)} \quad (21)$$

$$= 0. \quad (22)$$

Since there are n vectors in the set and they are linearly independent, they form a basis of \mathbb{C}^n . \square

1.2 Fourier series

The Fourier series of a function defined in the unit interval is the projection of the function onto the span of the complex sinusoids with integer frequencies, which are orthonormal by Lemma 1.4.

Definition 1.7 (Fourier series). *The partial Fourier series of order N of a function $f \in \mathcal{L}_2[-1/2, 1/2]$ is defined as*

$$S_N \{f\} := \sum_{i=-N}^N F[k] h_k, \quad (23)$$

$$F[k] := \langle f, h_k \rangle. \quad (24)$$

The Fourier series of a function $f \in \mathcal{L}_2[-1/2, 1/2]$ is defined as

$$S\{f\} := \sum_{i=-\infty}^{\infty} F[k] h_k, \quad (25)$$

$$F[k] := \langle f, h_k \rangle. \quad (26)$$

For real functions, the Fourier series is a projection onto an orthonormal basis of cosine and sine functions (we omit the proof that these functions are orthogonal, which is very similar to the proof of Lemma 1.4).

Lemma 1.8. *For a real function $f \in \mathcal{L}_2[-1/2, 1/2]$ we have*

$$S\{f\} = \sum_{i=0}^{\infty} a_k \cos(2\pi kt) + b_k \sin(2\pi kt), \quad (27)$$

$$a_k := 2 \langle f, \cos(2\pi kt) \rangle, \quad (28)$$

$$b_k := 2 \langle f, \sin(2\pi kt) \rangle. \quad (29)$$

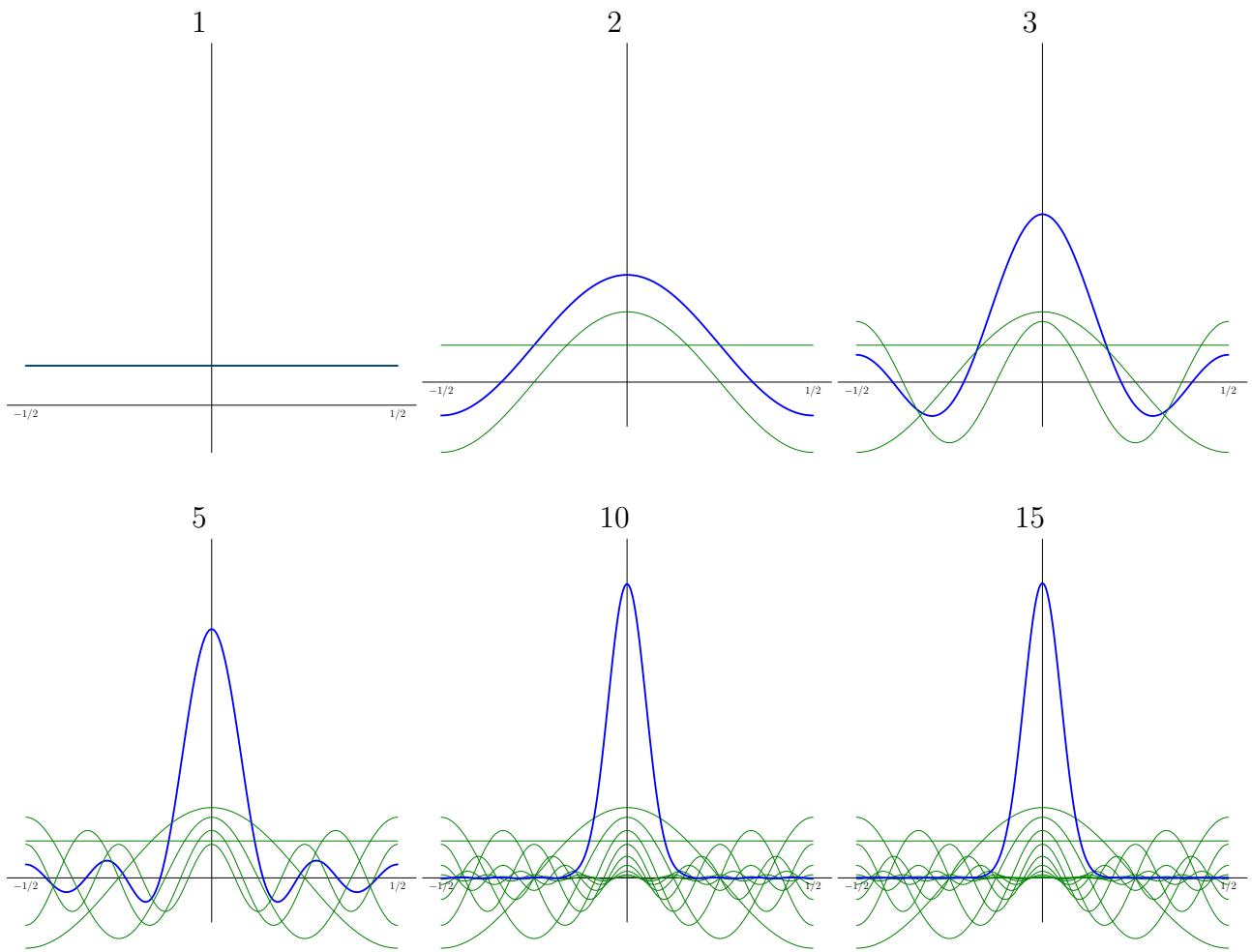


Figure 3: Partial Fourier series (blue) of the Gaussian function in Figure 5 restricted to the unit interval. The complex sinusoids used to produce the approximation are shown in green.

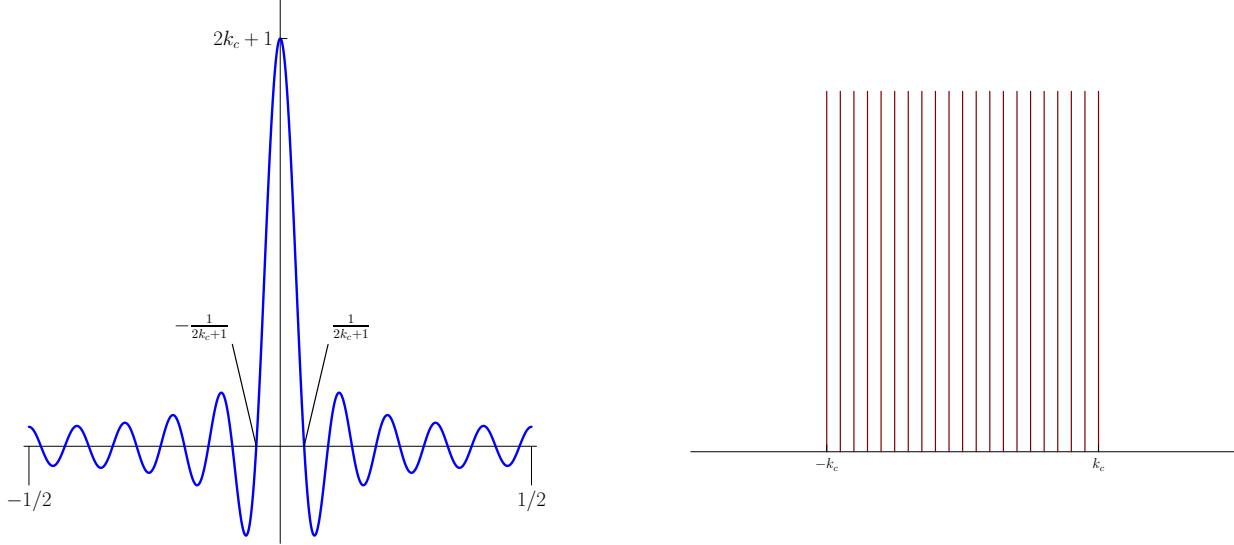


Figure 4: Dirichlet kernel with cut-off frequency k_c (left) and corresponding Fourier coefficients (right).

Figure 3 shows the n th-order partial Fourier series of a Gaussian function restricted to the unit interval for different values of n . As n increases, the approximation obtained by projecting the function onto the span of the complex sinusoids improves. Remarkably, if the function is integrable, the approximation eventually converges to the function. We omit the proof of this result, which is beyond the scope of these notes.

Theorem 1.9 (Convergence of Fourier series). *For any function $f \in \mathcal{L}_2 [-1/2, 1/2]$*

$$\lim_{N \rightarrow \infty} \|f - S_N\{f\}\|_{\mathcal{L}_2} = 0. \quad (30)$$

In addition, if f is continuously differentiable, the convergence is uniform and hence also pointwise, so that

$$\lim_{N \rightarrow \infty} S_N\{f\}(t) = f(t), \quad \text{for any } t. \quad (31)$$

By Theorem 1.9 we can represent any square-integrable function defined on an interval by using its Fourier coefficients, which are known as the *spectrum* of the function. It is worth noting that one can generalize this representation to functions defined on the whole real line by considering real-valued frequencies, which yields the Fourier transform. We will not discuss this further in these notes.

The Dirichlet kernel is an example of a *bandlimited* function, for which the higher end of the spectrum is zero, as illustrated in Figure 4.

Definition 1.10 (Dirichlet kernel). *The Fourier coefficients of the Dirichlet kernel with cut-off frequency k_c are equal to*

$$D[k] := \begin{cases} 1 & \text{if } |k| \leq k_c \\ 0 & \text{otherwise.} \end{cases} \quad (32)$$

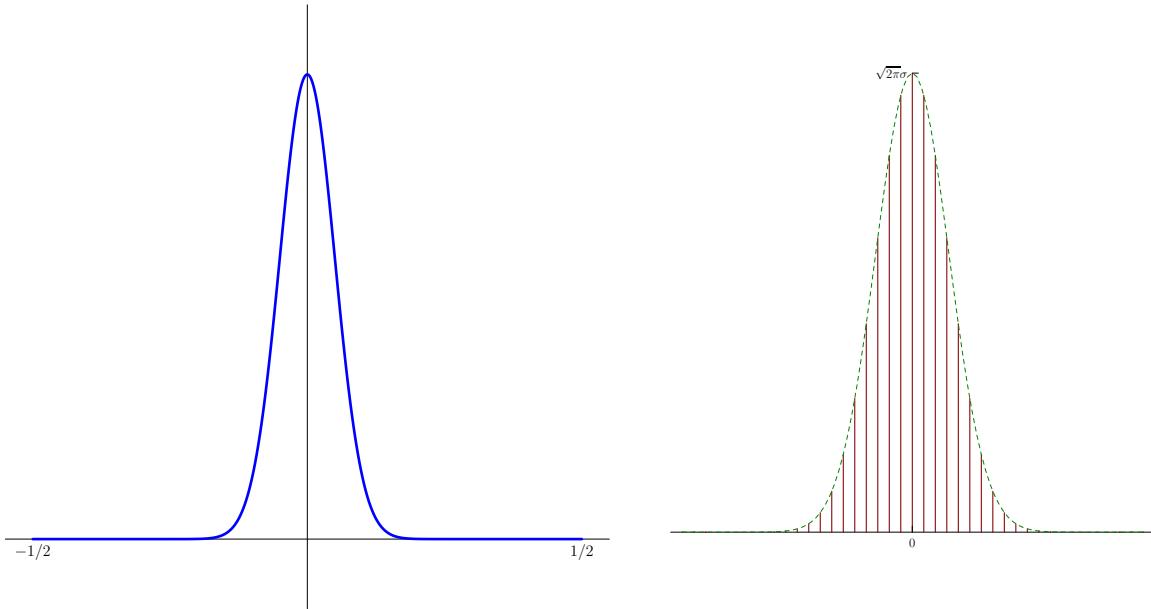


Figure 5: Gaussian function in Example 1.11 and corresponding Fourier coefficients (right).

The corresponding function is given by

$$d(t) = \sum_{n=-k_c}^{k_c} D[k] h_k(t) \quad (33)$$

$$= \sum_{n=-k_c}^{k_c} e^{i2\pi kt} \quad (34)$$

$$= \begin{cases} 2k_c + 1 & \text{if } t = 0, \\ \frac{\sin((2k_c+1)\pi t)}{\sin(\pi t)} & \text{otherwise.} \end{cases} \quad (35)$$

Note that the width of the main lobe of the Dirichlet kernel is inversely proportional to its cut-off frequency, which corresponds to the width of its spectrum. The following example shows that this is also the case for the Gaussian function.

Example 1.11 (Gaussian). Consider the Gaussian function

$$g(t) = \exp\left(-\frac{t^2}{\sigma^2}\right), \quad (36)$$

restricted to the interval $[-1/2, 1/2]$. If σ is small with respect to the length of the interval, then

we can approximate its spectrum by

$$G[k] = \int_{-1/2}^{1/2} \exp\left(-\frac{t^2}{\sigma^2}\right) \exp(-i2\pi kt) dt \quad (37)$$

$$= \int_{1/2}^{1/2} \exp\left(-\frac{t^2}{\sigma^2}\right) \cos 2\pi kt dt \quad \text{because } g \text{ is even and } \sin(2\pi kt) \text{ is odd} \quad (38)$$

$$\approx \int_{-\infty}^{\infty} \exp\left(-\frac{t^2}{\sigma^2}\right) \cos 2\pi kt dt \quad \text{if } \sigma \ll 1 \quad (39)$$

$$= \sqrt{\pi}\sigma \exp(-\pi^2\sigma^2k^2). \quad (40)$$

The envelope of the Fourier coefficients is Gaussian with a standard deviation that is proportional to $1/\sigma$, as shown in Figure 5. \triangle

An important property of the Fourier representation is that shifting a function is equivalent to scaling its Fourier coefficients by a factor that only depends on the shift and the corresponding frequency.

Lemma 1.12 (Time shift). *We define the τ -shifted version of a function $f \in \mathcal{L}_2[-1/2, 1/2]$ by*

$$f_{[\tau]}(t) := f(t - \tau), \quad (41)$$

where the shift is circular (the function wraps around). For any shift τ we have

$$F_{[\tau]}[k] = \exp(-i2\pi k\tau) F[k]. \quad (42)$$

Proof. We interpret f as a periodic function such that $f(t+1) = f(t)$. We have

$$F_{[\tau]}[k] = \int_{-1/2}^{1/2} f(t - \tau) \exp(-i2\pi kt) dt \quad (43)$$

$$= \int_{-1/2-\tau}^{1/2-\tau} f(u) \exp(-i2\pi k(u + \tau)) dt \quad (44)$$

$$= \exp(-i2\pi k\tau) F[k]. \quad (45)$$

\square

1.3 Sampling Theorem

In this section we consider the problem of estimating a bandlimited signal from samples taken at regular intervals. Such situations arise when we digitally store or process an analog signal such as music or speech. Figure 6 shows an example of a bandlimited signal, its spectrum and the corresponding samples. Two fundamental questions are:

1. What sampling rate is necessary to preserve all the information in the signal?
2. How can we reconstruct the original signal from the samples?

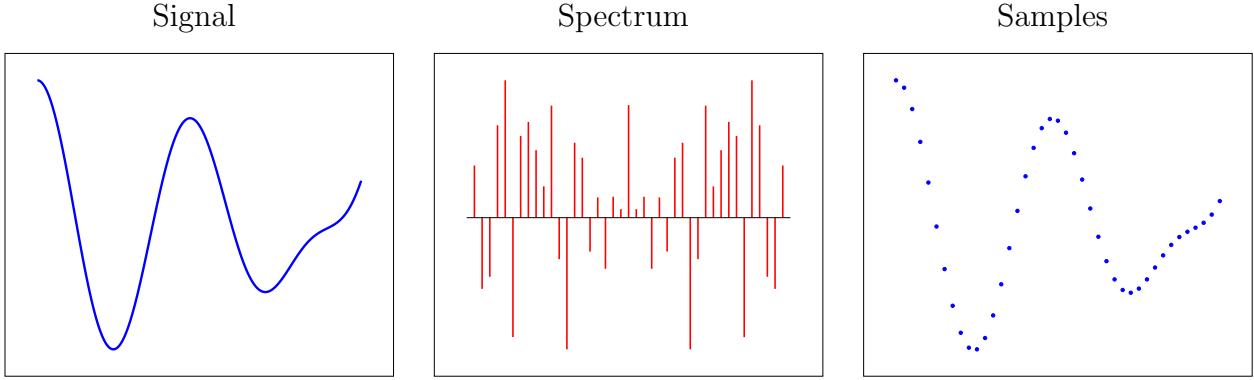


Figure 6: Bandlimited signal (left), corresponding spectrum (center) and regular samples (right).

The answer to the first question is given by the Nyquist-Shannon-Kotelnikov sampling theorem.

Theorem 1.13 (Nyquist-Shannon-Kotelnikov sampling theorem). *Any bandlimited signal $g \in \mathcal{L}_2[0, 1]$ of the form*

$$g(t) := \sum_{k=-k_c}^{k_c} G[k] h_k(t) \quad (46)$$

can be recovered exactly from n uniformly spaced samples $g(0), g(1/n), \dots, g((n-1)/n)$ as long as the sampling rate $f_s := n$ satisfies

$$f_s \geq 2k_c + 1, \quad (47)$$

which is known as the Nyquist rate.

Proof. To simplify the proof, we assume that $n = 2k_s + 1$ for some integer $k_s \geq k_c$. We denote the vector of samples by \vec{g}_n , which equals

$$\vec{g}_n := \begin{bmatrix} g(0) \\ g\left(\frac{2}{n}\right) \\ \vdots \\ g\left(\frac{n-1}{n}\right) \end{bmatrix} = \begin{bmatrix} \sum_{k=-k_c}^{k_c} G[k] h_k(0) \\ \sum_{k=-k_c}^{k_c} G[k] h_k\left(\frac{1}{n}\right) \\ \vdots \\ \sum_{k=-k_c}^{k_c} G[k] h_k\left(\frac{n-1}{n}\right) \end{bmatrix} = \sum_{k=-k_c}^{k_c} G[k] \begin{bmatrix} h_k(0) \\ h_k\left(\frac{1}{n}\right) \\ \vdots \\ h_k\left(\frac{n-1}{n}\right) \end{bmatrix} = \sum_{k=-k_c}^{k_c} G[k] \vec{h}_k^{[n]}. \quad (48)$$

Now let us define a vector $\vec{G} \in \mathbb{C}^n$ containing the Fourier coefficients of g (padded by zeros if $k_s \geq k_c$),

$$\vec{G}[k] := \begin{cases} G[k], & \text{if } |k| \leq k_c, \\ 0, & \text{otherwise.} \end{cases} \quad (49)$$

By equation (48) we have

$$\vec{g}_n = [\vec{h}_{-k_s}^{[n]} \ \vec{h}_{-k_s+1}^{[n]} \ \dots \ \vec{h}_{k_s}^{[n]}] \vec{G} \quad (50)$$

$$= F\vec{G}. \quad (51)$$

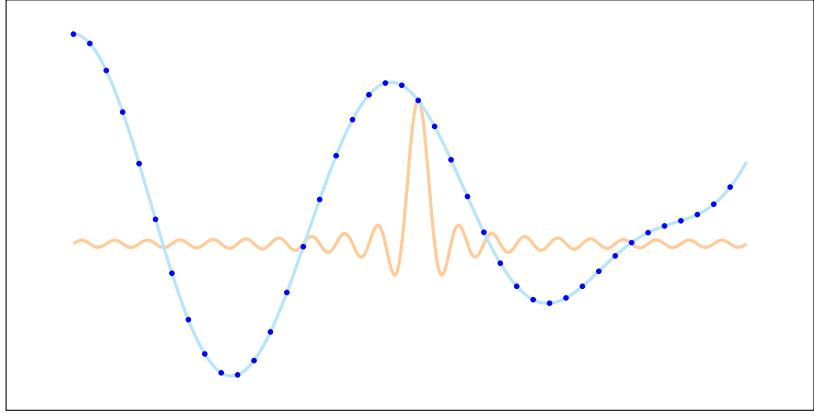


Figure 7: Reconstruction via linear inversion is equivalent to interpolation with a Dirichlet kernel.

By Lemma 1.6 the matrix F has orthogonal columns, so that we can recover \vec{G} and hence g from \vec{g}_n . \square

To recover the signal we can build its Fourier series from its Fourier series obtained by inverting the system of equations given by (51). Interestingly, this is exactly equivalent to interpolating the samples with shifted Dirichlet kernels, as sketched in Figure 7.

Theorem 1.14 (Dirichlet-kernel interpolation). *Let $n := 2k_s + 1$ for some integer k_s . As long as $k_s \geq k_c$, a bandlimited function of the form (46) is equal to*

$$g(t) = \frac{1}{n} \sum_{j=0}^{n-1} g(j/n) d_{[j/n]}(t) \quad (52)$$

where d is a Dirichlet kernel (see Definition 1.10) with cut-off frequency k_s .

Proof. Let us define the vector

$$\vec{a}_t := [\exp(-i2\pi k_s t) \quad \exp(-i2\pi(k_s - 1)t) \quad \cdots \quad \exp(i2\pi k_s t)]^T, \quad (53)$$

such that the adjoint of the matrix F in (51) can be expressed as

$$F^* = [\vec{a}_0 \quad \vec{a}_{1/n} \quad \cdots \quad \vec{a}_{(n-1)/n}]. \quad (54)$$

F scaled by $1/\sqrt{n}$ is orthogonal, so

$$\vec{G} = F^* \vec{g}_n \quad (55)$$

$$= \frac{1}{n} \sum_{j=0}^{n-1} g(j/n) \vec{a}_{j/n}. \quad (56)$$

Now, we express g and the Dirichlet kernel in terms of \vec{a}_t ,

$$g(t) = \frac{1}{n} \sum_{k=-k_c}^{k_c} G[k] e^{-i2\pi kt} \quad (57)$$

$$= \frac{1}{n} \vec{a}_t^* \vec{G}, \quad (58)$$

$$d_{[\tau]}(t) = \sum_{k=-k_c}^{k_c} e^{-i2\pi k(t-\tau)} \quad (59)$$

$$= \vec{a}_t^* \vec{a}_\tau. \quad (60)$$

We conclude

$$g(t) = \frac{1}{n} \vec{a}_t^* \vec{G} \quad \text{by (58)} \quad (61)$$

$$= \frac{1}{n} \sum_{j=0}^n g(j/n) \vec{a}_t^* \vec{a}_{j/n} \quad \text{by (56)} \quad (62)$$

$$= \frac{1}{n} \sum_{j=0}^n g(j/n) d_{[j/n]}(t) \quad \text{by (60).} \quad (63)$$

□

Finally, another interesting question is what occurs when the cut-off frequency of the function we are sampling k_c is too large with respect to the sampling rate $k_c > k_s$ so that we violate the sampling-rate condition (47). The following example illustrates this.

Example 1.15 (Aliasing). We consider a function

$$g(t) := \sum_{k=-k_c}^{k_c+1} G[k] h_k(t). \quad (64)$$

with cut-off frequency $k_c + 1$ defined on the unit interval. We have set $G[-k_c - 1]$ to simplify notation. Imagine that we underestimate its cut-off frequency and sample it at a rate $k_s := k_c/2$ instead of $k_c + 1$, gathering $n := 2k_s + 1 = k_c + 1$ samples. Let us define \vec{G} as the vector containing the Fourier coefficients of g . By (48) we have

$$\vec{g}_n = \begin{bmatrix} \vec{h}_{-k_c}^{[n]} & \dots & \vec{h}_0^{[n]} & \vec{h}_1^{[n]} & \dots & \vec{h}_{k_c+1}^{[n]} \end{bmatrix} \vec{G} \quad (65)$$

$$= \begin{bmatrix} \vec{h}_{-2k_s}^{[n]} & \dots & \vec{h}_0^{[n]} & \vec{h}_1^{[n]} & \dots & \vec{h}_{2k_s+1}^{[n]} \end{bmatrix} \vec{G} \quad (66)$$

$$= \begin{bmatrix} \vec{h}_1^{[n]} & \dots & \vec{h}_n^{[n]} & \vec{h}_1^{[n]} & \dots & \vec{h}_n^{[n]} \end{bmatrix} \vec{G} \quad (67)$$

$$= \begin{bmatrix} \tilde{F} & \tilde{F} \end{bmatrix} \begin{bmatrix} \vec{G}_1 \\ \vec{G}_2 \end{bmatrix}, \quad (68)$$

since for any k $\vec{h}_k^{[n]} = \vec{h}_{k+n}^{[n]}$ by periodicity of the discrete complex sinusoids. Note that we have separated \vec{G}_1 into its first and second half. Now we have a problem because $\begin{bmatrix} \tilde{F} & \tilde{F} \end{bmatrix}$ is clearly not invertible!

Compare this to the situation where $n = 2k_c + 1$. As we showed in the proof of Theorem 1.13 (with a different choice of indices that doesn't affect the result) in that case

$$\vec{G} = \frac{1}{n} \tilde{F}^* \vec{g}_n, \quad (69)$$

since \tilde{F} has orthogonal columns by Lemma 1.6.

What happens if we assume that this holds even if it doesn't and we try to obtain \vec{G} from \vec{g}_n in the same way? Then *aliasing* occurs:

$$\vec{G}_{\text{aliased}} = \frac{1}{n} \tilde{F}^* \vec{g}_n \quad (70)$$

$$= \frac{1}{n} \tilde{F}^* \begin{bmatrix} \tilde{F} & \tilde{F} \end{bmatrix} \begin{bmatrix} \vec{G}_1 \\ \vec{G}_2 \end{bmatrix} \quad (71)$$

$$= \vec{G}_1 + \vec{G}_2. \quad (72)$$

The recovered spectrum is scrambled. We recover a function with cut-off frequency k_c with a spectrum that equals the sum of the spectrum of g and a copy that is shifted by k_c . \triangle

1.4 Discrete Fourier transform

By Lemma 1.6 the discrete complex sinusoids $\frac{1}{\sqrt{n}} \vec{h}_0^{[n]}, \dots, \frac{1}{\sqrt{n}} \vec{h}_{n-1}^{[n]}$ form an orthonormal basis of \mathbb{C}^n . This means that we can express any vector in terms of its coefficients in this representation. The discrete Fourier transform (DFT), which extracts these coefficients, is consequently the discrete counterpart of the Fourier series.

Definition 1.16 (Discrete Fourier transform). *The discrete Fourier transform (DFT) of a vector $\vec{x} \in \mathbb{C}^n$ are given by*

$$\vec{X}[k] := \left\langle \vec{x}, \vec{h}_k^{[n]} \right\rangle, \quad 0 \leq k \leq n - 1. \quad (73)$$

Equivalently,

$$\vec{X} = W \vec{x} \quad (74)$$

where $W^* := \begin{bmatrix} \vec{h}_0^{[n]} & \vec{h}_1^{[n]} & \dots & \vec{h}_{n-1}^{[n]} \end{bmatrix}^*$. W is known as the DFT matrix.

Recovering a vector from its DFT coefficients only requires inverting the linear transformation.

Lemma 1.17. *If \vec{X} contains the DFT coefficients of $\vec{x} \in \mathbb{C}^n$, then*

$$\vec{x} = \frac{1}{n} \sum_{k=0}^{n-1} \vec{X}[k] \vec{h}_k^{[n]} \quad (75)$$

$$= \frac{1}{n} F^* \vec{X}. \quad (76)$$

Proof. By Lemma 1.6 $\frac{1}{\sqrt{n}}F$ is an orthogonal matrix, so

$$F^* \vec{X} = F^* F \vec{x} \quad (77)$$

$$= \vec{x}. \quad (78)$$

□

As in the case of the Fourier series, the DFT coefficients of a shifted vector can be computed by scaling linearly the DFT coefficients of the original vector.

Lemma 1.18 (Discrete time shift). *We define the m -shifted version of a vector $\vec{x} \in \mathbb{C}^n$ by*

$$\vec{x}_{[m]}[j] := \vec{x}(j - m), \quad (79)$$

where the shift is circular, so that $\vec{x}(j + n) = \vec{x}(j)$. For any shift m we have

$$\vec{X}_{[m]}[k] = \exp(-i2\pi km) \vec{X}[k]. \quad (80)$$

Proof.

$$\vec{X}_{[m]}[k] = \sum_{j=0}^{n-1} \vec{x}_{[\tau]}[j] \exp(-i2\pi kj) \quad (81)$$

$$= \sum_{l=-m}^{n-1-m} \vec{x}[l] \exp(-i2\pi k(l + m)) \quad (82)$$

$$= \exp(-i2\pi km) \vec{X}[k]. \quad (83)$$

□

Remark 1.19 (Connection between the Fourier series and the DFT). *If \vec{x} corresponds to samples from a bandlimited function, as in Theorem 1.13, which are measured at the Nyquist rate, then the DFT coefficients of \vec{x} are equal to the Fourier series coefficients of the continuous function. This follows from (51), since if we re-index the columns adequately (recall that $\bar{h}_{k+n}^{[n]} = \bar{h}_k^{[n]}$ for any k) we have $W = F^*$. The DFT, combined with the sampling period, consequently makes it possible to compute the spectrum of continuous signals in practice!*

Figure 8 shows the DFT of an electrocardiogram signal. Its frequency representation makes it possible to locate and remove interference caused by the power-line system.

The DFT of a signal of length n can be obtained with complexity $\mathcal{O}(n \log n)$ by applying the fast-Fourier transform (FFT) algorithm described below. It is difficult to overstate the importance of this method, which makes it extremely efficient to compute the DFT. The algorithm relies on the following lemma, which you will prove in the homework.

Lemma 1.20. *Let $W^{[n]}$ denote the $n \times n$ DFT matrix and assume that n is even. Then for $k < n/2$*

$$W^{[n]} \vec{x}[k] = W^{[n/2]} \vec{x}_{\text{even}}[k] + \exp\left(-\frac{i2\pi k}{n}\right) W^{[n/2]} \vec{x}_{\text{odd}}[k], \quad (84)$$

$$W^{[n]} \vec{x}[k + n/2] = W^{[n/2]} \vec{x}_{\text{even}}[k] - \exp\left(-\frac{i2\pi k}{n}\right) W^{[n/2]} \vec{x}_{\text{odd}}[k]. \quad (85)$$

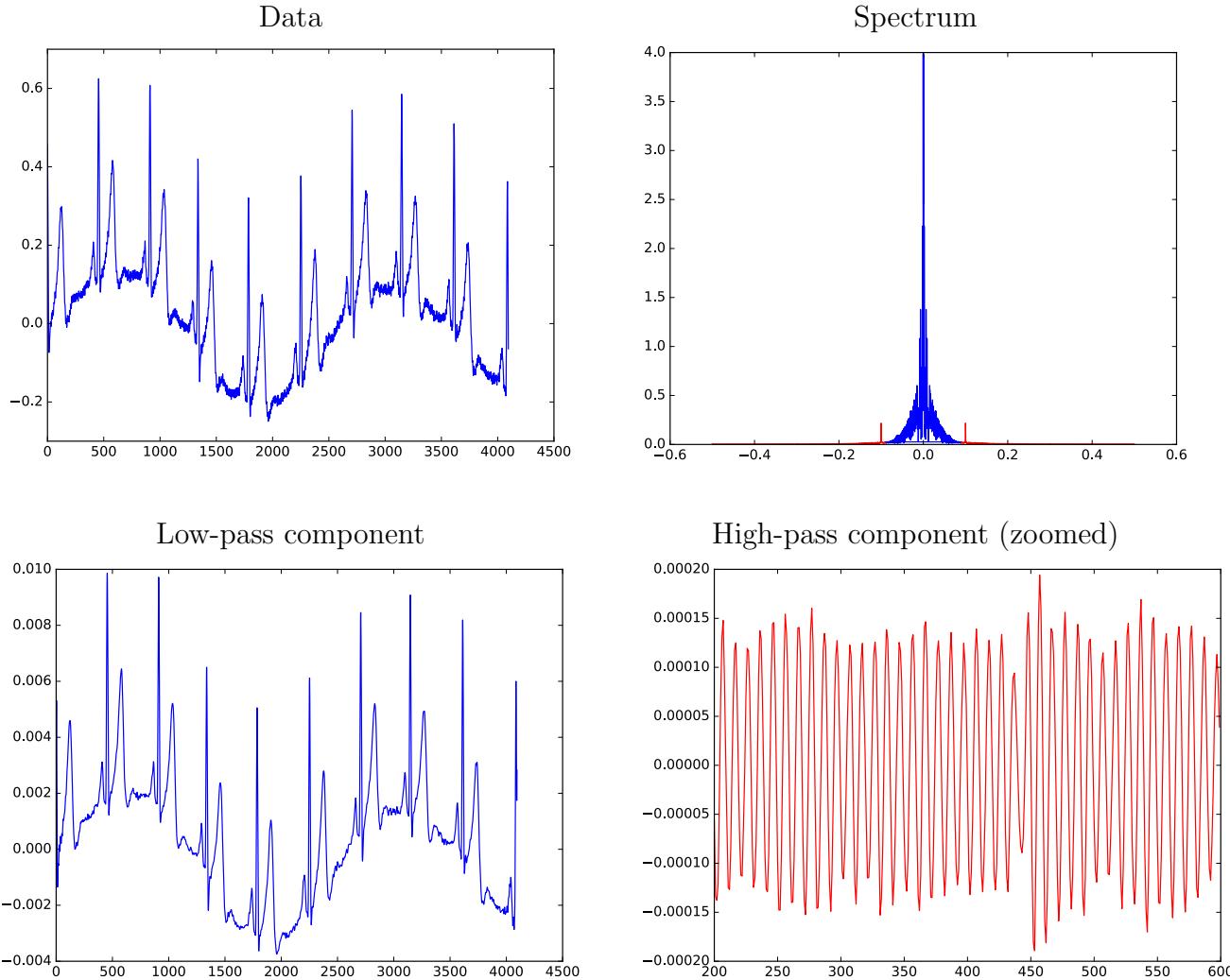


Figure 8: Electrocardiogram data (top left), along with its spectrum (top right). We separate the spectrum into a low-pass (blue) and a high-pass (red) region. The low-pass region corresponds to the signal of interest (bottom left), whereas the high-pass component corresponds to power-line interference (bottom right).

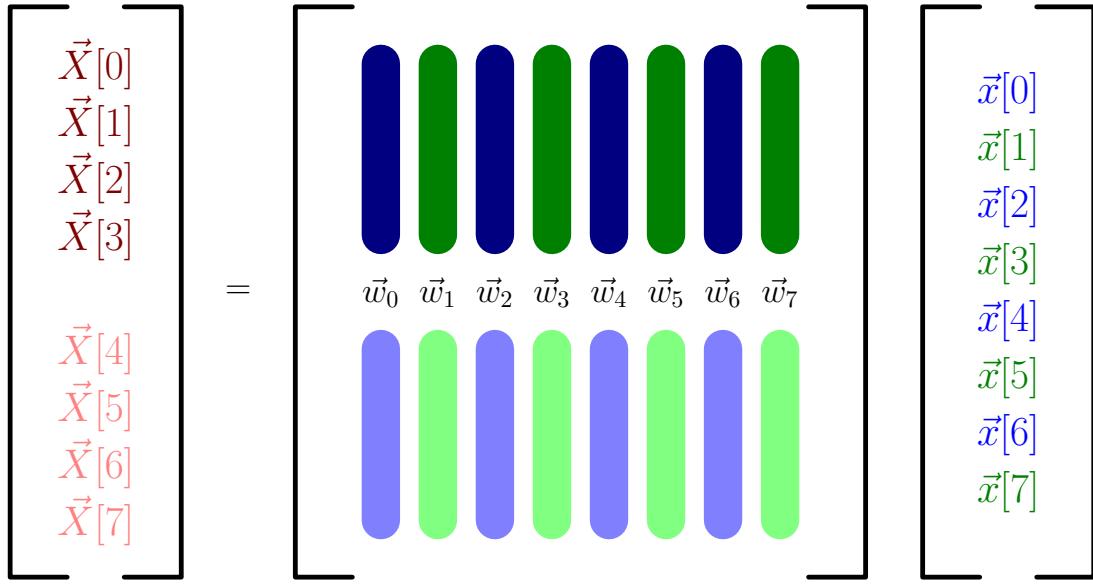


Figure 9: $W^{(8)}$ can be decomposed into even columns (blue) and odd columns (green). As we will see, it is also useful to distinguish the top half (dark) from the bottom half (light).

Algorithm 1.21 (Fast Fourier transform). *If $n = 1$, then set $W^1 \vec{x} := \vec{x}$. Otherwise apply the following steps:*

1. Compute $W^{[n/2]} \vec{x}_{\text{even}}$.
2. Compute $W^{[n/2]} \vec{x}_{\text{odd}}$.
3. For $k = 1, 2, \dots, n/2$ set

$$W^{[n]} \vec{x}[k] := W^{[n/2]} \vec{x}_{\text{even}}[k] + \exp\left(-\frac{i2\pi k}{n}\right) W^{[n/2]} \vec{x}_{\text{odd}}[k], \quad (86)$$

$$W^{[n]} \vec{x}_{k+n/2} := W^{[n/2]} \vec{x}_{\text{even}}[k] - \exp\left(-\frac{i2\pi k}{n}\right) W^{[n/2]} \vec{x}_{\text{odd}}[k]. \quad (87)$$

In Figures 9, 10, 11, 12 we visually depict the FFT algorithm for $n = 8$. Figure 13 shows the recursion tree, which illustrates why the runtime is $O(n \lg n)$.

The DFT can be extended to two dimensions by considering two-dimensional sinusoidal atoms obtained by taking the outer product of the one-dimensional discrete complex sinusoids of different frequencies.

Definition 1.22 (Two-dimensional DFT). *The 2D DFT \widehat{M} of a matrix $M \in \mathbb{C}^{n \times n}$ is given by*

$$\widehat{M}[k] := \langle M, \vec{h}_{k_1, k_2}^{2D} \rangle, \quad 0 \leq k_1, k_2 \leq n - 1, \quad (88)$$

$$\begin{bmatrix}
e^{-2\pi i(0)/8} \\
e^{-2\pi i(1)/8} \\
e^{-2\pi i(2)/8} \\
e^{-2\pi i(3)/8} \\
e^{-2\pi i(4)/8} \\
e^{-2\pi i(5)/8} \\
e^{-2\pi i(6)/8} \\
e^{-2\pi i(7)/8}
\end{bmatrix}
\begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix}
= \begin{bmatrix}
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7 \\
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6
\end{bmatrix}$$

Figure 10: The odd columns of $W^{(8)}$ can be calculated from the evens by scaling the rows.

$$\begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix}
= \begin{bmatrix}
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7 \\
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6
\end{bmatrix}$$

Figure 11: The bottom and top half of the even columns are the same. Both are a DFT matrix of size $4 = n/2$.

$$\begin{aligned}
\begin{bmatrix}
\vec{X}[0] \\
\vec{X}[1] \\
\vec{X}[2] \\
\vec{X}[3]
\end{bmatrix}
&= \begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix} \begin{bmatrix}
x[0] \\
x[2] \\
x[4] \\
x[6]
\end{bmatrix} + \begin{bmatrix}
e^{-2\pi i(0)/8} \\
e^{-2\pi i(1)/8} \\
e^{-2\pi i(2)/8} \\
e^{-2\pi i(3)/8}
\end{bmatrix} \begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix} \begin{bmatrix}
x[1] \\
x[3] \\
x[5] \\
x[7]
\end{bmatrix} \\
\begin{bmatrix}
\vec{X}[4] \\
\vec{X}[5] \\
\vec{X}[6] \\
\vec{X}[7]
\end{bmatrix}
&= \begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix} \begin{bmatrix}
\vec{x}[0] \\
\vec{x}[2] \\
\vec{x}[4] \\
\vec{x}[6]
\end{bmatrix} + \begin{bmatrix}
e^{-2\pi i(4)/8} \\
e^{-2\pi i(5)/8} \\
e^{-2\pi i(6)/8} \\
e^{-2\pi i(7)/8}
\end{bmatrix} \begin{bmatrix}
\vec{w}_0 & \vec{w}_2 & \vec{w}_4 & \vec{w}_6 \\
\vec{w}_1 & \vec{w}_3 & \vec{w}_5 & \vec{w}_7
\end{bmatrix} \begin{bmatrix}
\vec{x}[1] \\
\vec{x}[3] \\
\vec{x}[5] \\
\vec{x}[7]
\end{bmatrix}
\end{aligned}$$

Figure 12: The $n = 8$ FFT calculation represented using only the dark blue portion of the matrix. This reduces the $n = 8$ FFT to two calls of the $n = 4$ FFT and then $O(n)$ work to combine the results.

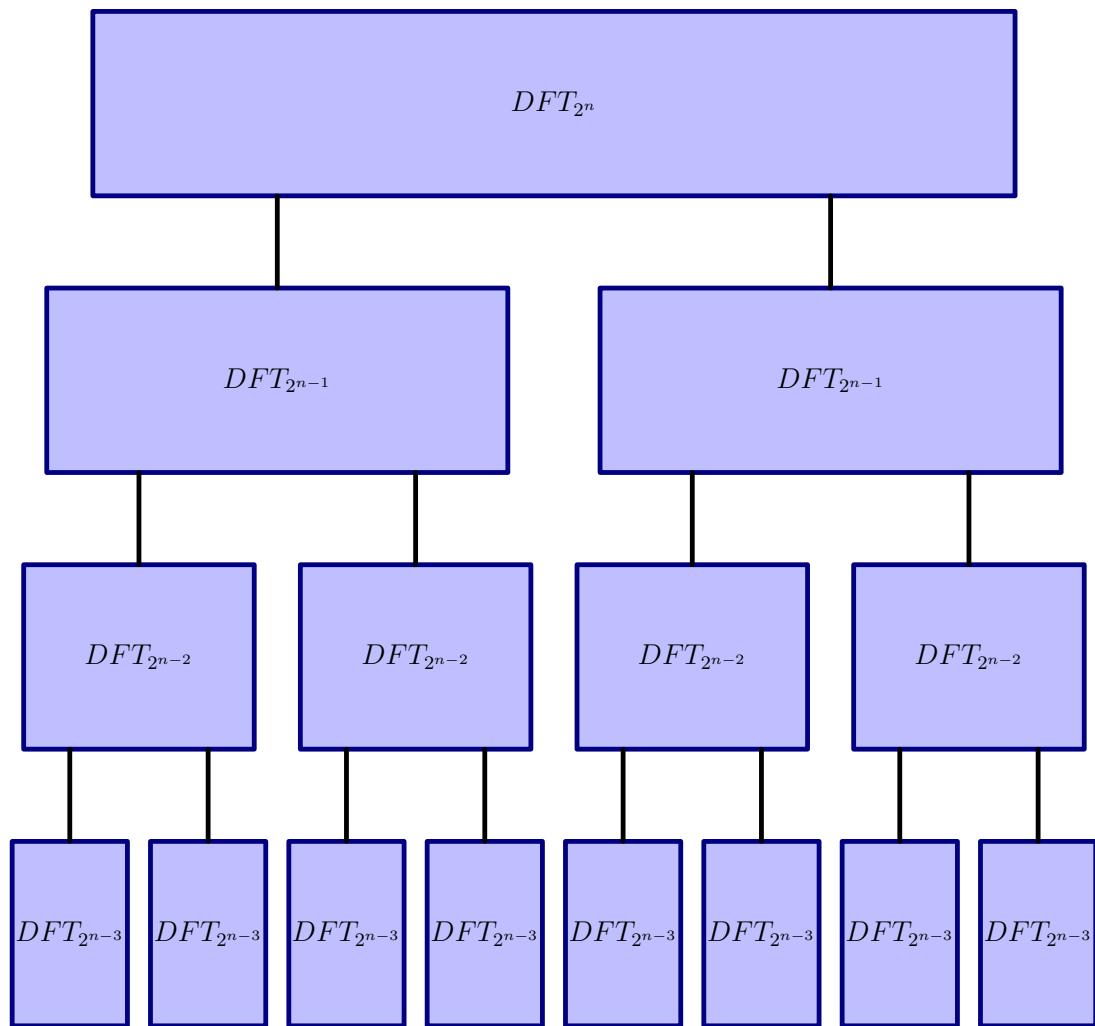


Figure 13: The recursion tree for the FFT.

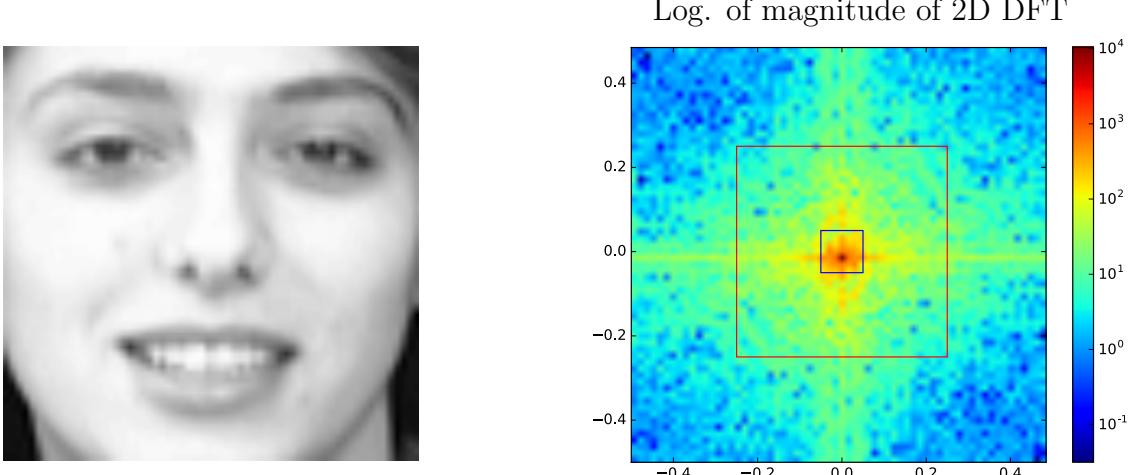


Figure 14: The logarithm of the magnitudes of the 2D DFT coefficients of the image on the left are shown on the right.

where

$$\vec{h}_{k_1, k_2}^{2D} := \vec{h}_{k_1}^{[n]} \left(\vec{h}_{k_2}^{[n]} \right)^T \quad (89)$$

$$= \begin{bmatrix} 1 & e^{\frac{i2\pi k_2}{n}} & \dots & e^{\frac{i2\pi k_2(n-1)}{n}} \\ e^{\frac{i2\pi k_1}{n}} & e^{\frac{i2\pi(k_1+k_2)}{n}} & \dots & e^{\frac{i2\pi(k_1+k_2(n-1))}{n}} \\ & \dots & & \\ e^{\frac{i2\pi k_1(n-1)}{n}} & e^{\frac{i2\pi(k_1(n-1)+k_2)}{n}} & \dots & e^{\frac{i2\pi(k_1(n-1)+k_2(n-1))}{n}} \end{bmatrix}. \quad (90)$$

Equivalently,

$$\widehat{M} = WMW, \quad (91)$$

where $W := \begin{bmatrix} \vec{h}_0^{[n]} & \vec{h}_1^{[n]} & \dots & \vec{h}_{n-1}^{[n]} \end{bmatrix}^*$ is the 1D DFT matrix.

Figure 14 shows an image along with the magnitudes of its 2D DFT coefficients. As is often the case for natural images, most of the energy is concentrated in the lower end of the spectrum. Figure 15 shows the result of projecting different components of the spectrum of the image (see the right image of Figure 14) onto the image domain. The low-pass component captures low-resolution variations, the band-pass component higher-resolution details and the high-pass component high-frequency fluctuations.

1.5 Compression

The discrete cosine transform (DCT) is a variant of the discrete Fourier transform for real discrete signals. The DCT is obtained from the DFT by assuming that the signal is symmetric (but only one half is observed). In that case the DFT components correspond to discrete cosines (the

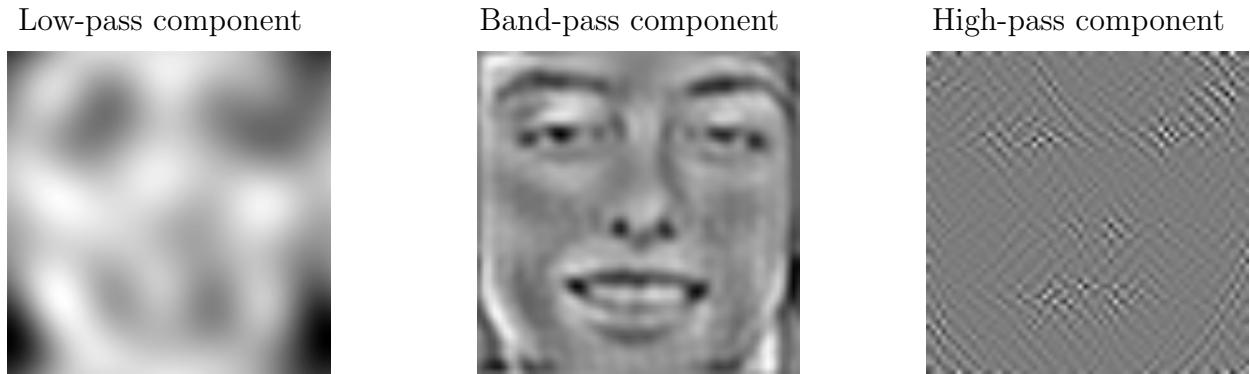


Figure 15: Image corresponding to the low-pass (left), band-pass (center) and high-pass (right) components of the image in Figure 14. The corresponding regions of the spectrum are shown in Figure 14.

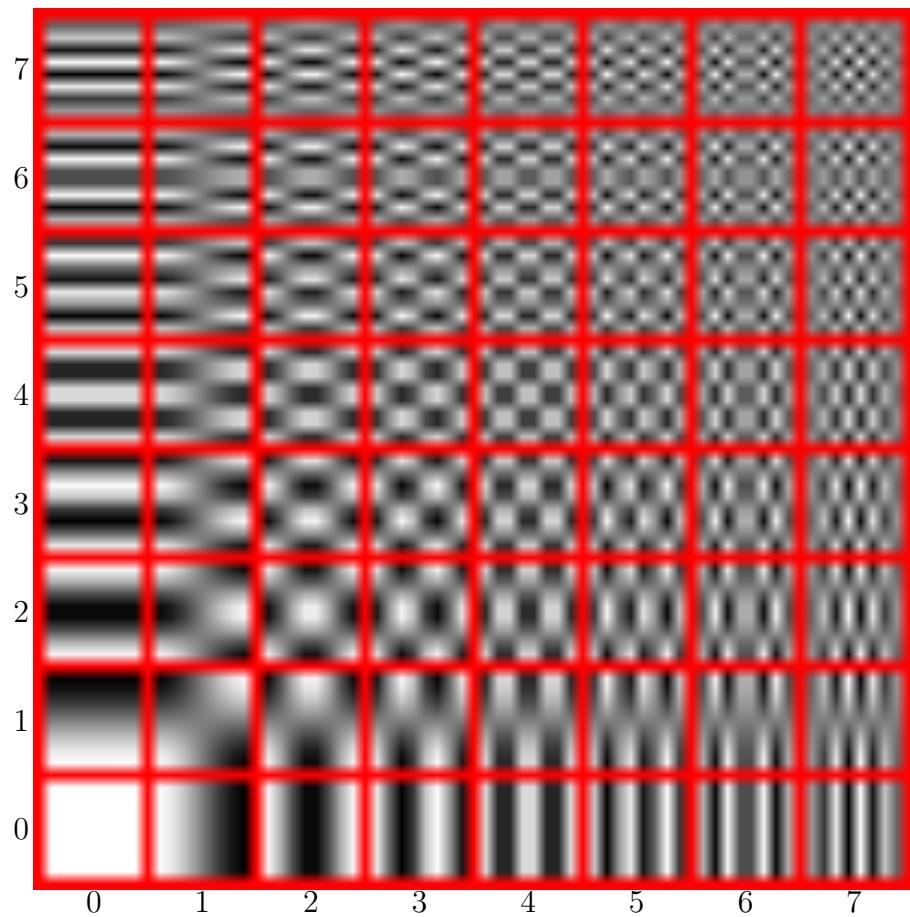


Figure 16: 8×8 DCT basis vectors.

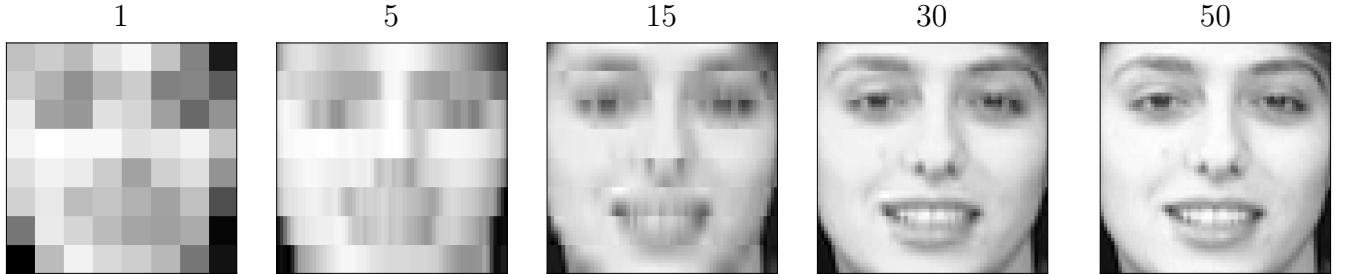


Figure 17: Result of projecting each 64-pixel patch from the natural image in Figure 14 onto the lowest 1, 5, 15, 30 and 50 2D DCT basis functions.

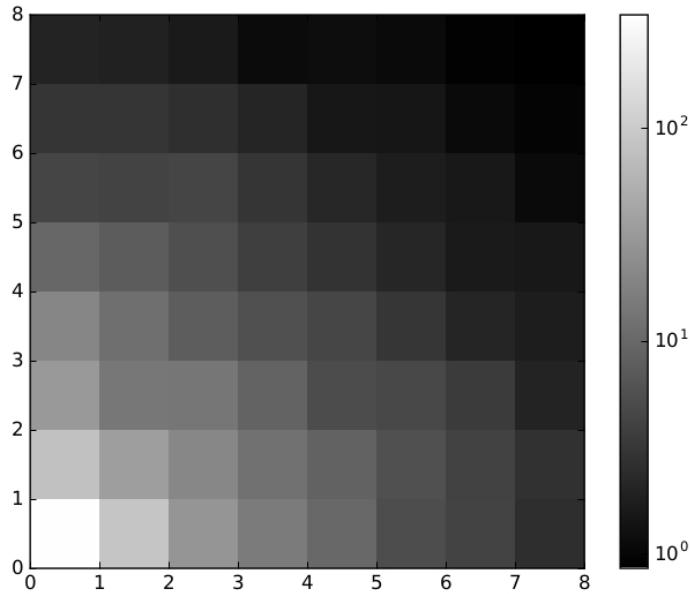


Figure 18: Average magnitudes of each 2D DCT coefficient in a database of patches extracted from natural images.

coefficients corresponding to the sines are zero due to symmetry). The two-dimensional DCT is a very important tool in image processing. Figure 16 shows the basis vectors of the 8×8 2D DCT.

One of the main reasons that the 2D DCT is so useful is that the energy of natural images are highly concentrated in their low-frequency components. Figure 18 shows the average magnitudes of each 2D DCT coefficient in a database of patches extracted from natural images. Figure 17 shows the result of dividing the image into 64-pixel patches and projecting it onto the span of the low-frequency DCT basis functions. Ignoring some of the high-frequency components is almost imperceptible. This is the main insight behind the JPEG method for lossy compression of digital images. JPEG divides the image in 8×8 patches and then quantizes each band differently, using more bits for lower-frequency bands where differences are more apparent.

Algorithm 1.23 (JPEG compression (for grayscale images)). 1. Choose a quality setting $Q \in (0, 100)$.

2. Divide image into a collection of 8×8 pixel patches.

$$M = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Figure 19: JPEG DCT Quantization Matrix



Figure 20: Natural image compared to the result of quantizing the lowest 16 DCT coefficients of each patch, the highest 16 DCT coefficients of each patch and applying JPEG Compression with $Q = 90$ (which yields a compression factor of between 4 and 5).

3. Compute the 2D DCT of each patch.

4. Quantize the 2D DCT of each patch separately. Let $\widehat{P} \in \mathbb{R}^{8 \times 8}$ denote the 2D DCT of a patch and let M denote the JPEG quantization matrix shown in Figure ???. Set

$$\widehat{P}'_{ij} = \text{round} \left(\frac{\widehat{P}_{ij}}{S(Q)M_{ij}} \right) S(Q)M_{ij}, \quad (92)$$

where $S(Q)$ is the quality scaling factor:

$$S(Q) := \begin{cases} \frac{100-Q}{50} & \text{if } Q > 50, \\ \frac{50}{Q} & \text{otherwise.} \end{cases} \quad (93)$$

5. Encode into a file. When viewing, the decoder will compute the inverse 2D DCT of each quantized patch \widehat{P}' to display the image.

2 Convolution

2.1 Continuous convolution

Convolution between functions is a fundamental operation in signal and image processing.

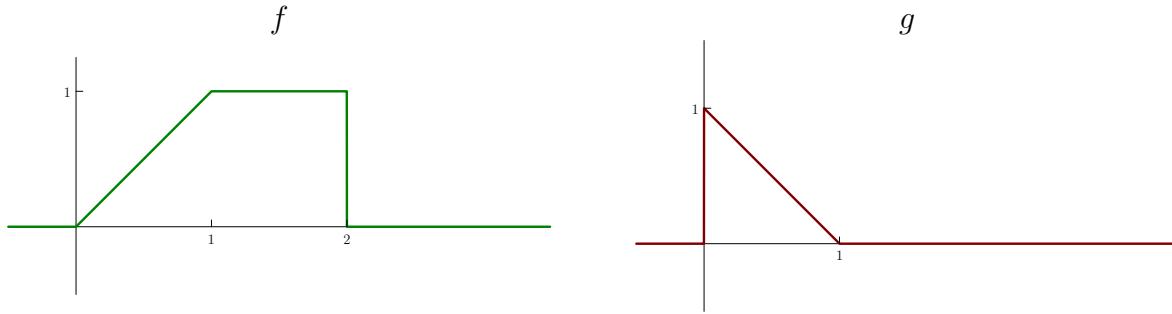


Figure 21: Two functions f and g .

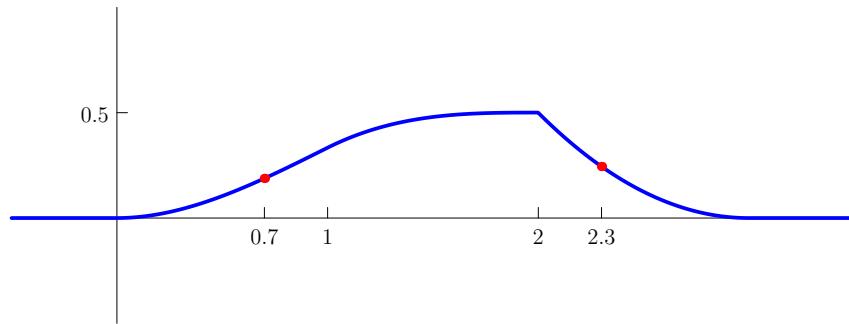


Figure 22: The result of convolving functions f and g from Figure 21.

Definition 2.1 (Convolution). *The convolution of two functions $f, g \in \mathcal{L}_2 [-1/2, 1/2]$ is defined as*

$$f * g (t) := \int_{-1/2}^{1/2} f(u) g(t-u) du. \quad (94)$$

Figures 21, 22 and 23 illustrate the operation with a simple example. To compute the value of the convolution between two functions at point t , we (1) fix one of them ($f(u)$), (2) flip and shift the other by t ($g(t-u)$) and (3) integrate their product.

Convolution in time (or space) is equivalent to multiplication in frequency. This implies that we can compute convolutions very efficiently using the FFT.

Theorem 2.2 (Convolution in time is multiplication in frequency). *Let $r := f * g$ for $f, g \in \mathcal{L}_2 [-1/2, 1/2]$. Then*

$$R[k] = F[k] G[k]. \quad (95)$$

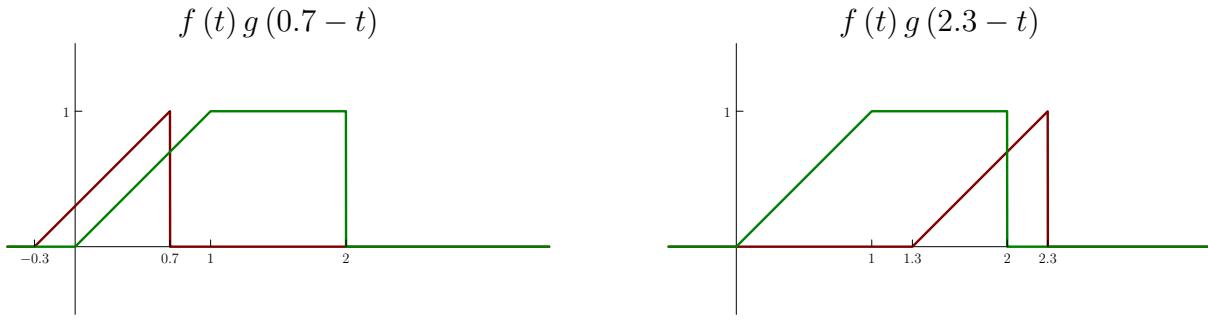


Figure 23: Position of the two functions in Figure 21 when they are multiplied to compute the value of their convolution at 0.7 (left) and 2.3 (right).

Proof.

$$R[k] := \int_{-1/2}^{1/2} \exp(-i2\pi kt) f * g(t) dt \quad (96)$$

$$= \int_{-1/2}^{1/2} f(u) \int_{-1/2}^{1/2} \exp(-i2\pi kt) g(t-u) dt du \quad (97)$$

$$= \int_{-1/2}^{1/2} f(u) G[k] \exp(-i2\pi ku) dt du \quad \text{by Lemma 1.12} \quad (98)$$

$$= F[k] G[k]. \quad (99)$$

□

The convolution theorem shows that we can compute convolutions between continuous functions by just multiplying their Fourier coefficients. It can also be used to prove the central limit theorem.

Example 2.3 (Sketch of a proof of the central limit theorem). The convolution theorem provides insight into why the distribution of sums of independent random variables become Gaussian in the limit.

Theorem 2.4 (Pdf of the sum of two independent random variables). *The pdf of $\mathbf{z} = \mathbf{x} + \mathbf{y}$, where \mathbf{x} and \mathbf{y} are independent random variables is equal to the convolution of their respective pdfs $f_{\mathbf{x}}$ and $f_{\mathbf{y}}$,*

$$f_{\mathbf{z}}(z) = \int_{u=-\infty}^{\infty} f_{\mathbf{x}}(z-u) f_{\mathbf{y}}(u) du. \quad (100)$$

Proof. Note that

$$F_{\mathbf{z}}(z) = P(\mathbf{x} + \mathbf{y} \leq z) \quad (101)$$

$$= \int_{y=-\infty}^{\infty} \int_{x=-\infty}^{z-y} f_{\mathbf{x}, \mathbf{y}}(x, y) dx dy \quad (102)$$

$$= \int_{y=-\infty}^{\infty} \int_{u=-\infty}^z f_{\mathbf{x}, \mathbf{y}}(u-y, y) du dy \quad (u = x + y) \quad (103)$$

$$= \int_{u=-\infty}^z \int_{y=-\infty}^{\infty} f_{\mathbf{x}, \mathbf{y}}(u-y, y) dy du, \quad (104)$$

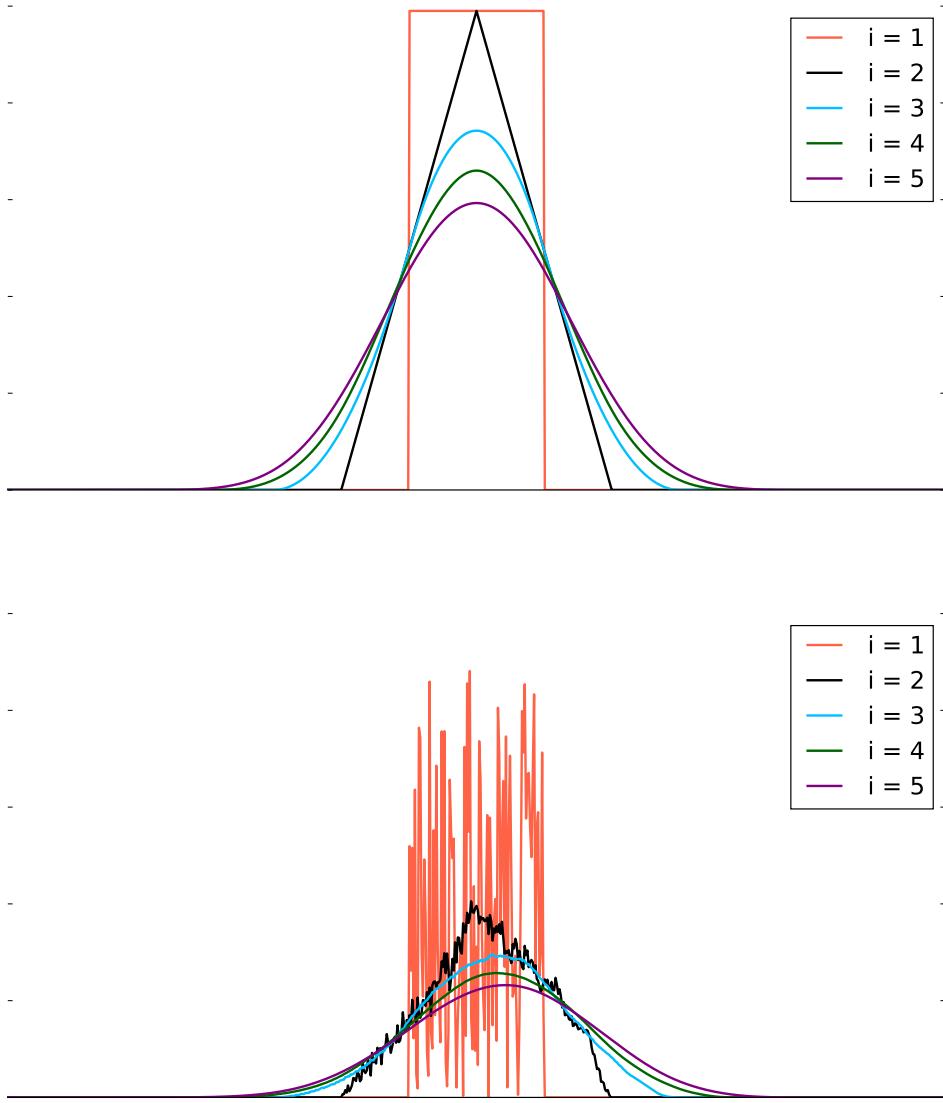


Figure 24: Result of convolving two different distributions with themselves several times. The shapes quickly become Gaussian-like.

where swapping the integrals is justified since the pdfs are nonnegative. Applying independence shows

$$f_{\mathbf{z}}(z) = \int_{y=-\infty}^{\infty} f_{\mathbf{x},\mathbf{y}}(z-y, y) dy = \int_{y=-\infty}^{\infty} f_{\mathbf{x}}(z-y) f_{\mathbf{y}}(y) dy. \quad (105)$$

△

Now let us consider a sequence of iid random variables $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots$ with pdf f . The pdf of their sum is given by

$$f_{\sum_{j=1}^{\infty} \mathbf{x}_j}(x) = (f * f * \dots)(x). \quad (106)$$

Convolutions have a smoothing effect, which eventually transforms the pmf/pdf into a Gaussian! We show this numerically in Figure 24 for two very different distributions: a uniform distribution and a very irregular one. Both converge to Gaussian-like shapes after just 3 or 4 convolutions. The central limit theorem makes this precise, establishing that the shape of the pmf or pdf does indeed become Gaussian asymptotically. △

2.2 Discrete convolution

Definition 2.5 (Discrete convolution). *The circular convolution of two vectors $\vec{x}, \vec{y} \in \mathbb{C}^n$ is defined as*

$$\vec{x} * \vec{y}[j] := \sum_{m=0}^{n-1} \vec{x}[m] \vec{y}[j-m], \quad 0 \leq j \leq n-1, \quad (107)$$

(108)

where the shifts are circular, so that $\vec{x}[j] = \vec{x}[j+n]$ and $\vec{y}[j] = \vec{y}[j+n]$.

Circular convolution can be expressed as multiplication with a convolution matrix.

Definition 2.6 (Convolution matrix). *The convolution matrix corresponding to a vector $\vec{x} \in \mathbb{C}^n$ contains every possible shift of the entries of \vec{y} in its rows*

$$C_{\vec{y}} := \begin{bmatrix} \vec{y}[0] & \vec{y}[n-1] & \cdots & \vec{y}[2] & \vec{y}[1] \\ \vec{y}[1] & \vec{y}[0] & \cdots & \vec{y}[3] & \vec{y}[2] \\ & & & \ddots & \\ \vec{y}[n-1] & \vec{y}[n-2] & \cdots & \vec{y}[1] & \vec{y}[0] \end{bmatrix}. \quad (109)$$

Matrices with this structure are called **circulant** matrices. Assuming that the vectors entries are numbered from 0 to $n-1$, the convolution between \vec{y} and any other vector $\vec{x} \in \mathbb{C}^n$ can be expressed as

$$\vec{x} * \vec{y} = C_{\vec{y}} \vec{x}. \quad (110)$$

Theorem 2.7 (Convolution in time is multiplication in frequency). *Let $\vec{r} := \vec{x} * \vec{y}$ for $\vec{x}, \vec{y} \in \mathbb{C}^n$. Then*

$$\vec{R}[k] = \vec{X}[k] \vec{Y}[k]. \quad (111)$$

Proof.

$$R[k] := \sum_{j=0}^{n-1} \exp(-i2\pi kj) \sum_{m=0}^{n-1} \vec{x}[m] \vec{y}[j-m] \quad (112)$$

$$= \sum_{m=0}^{n-1} \vec{x}[m] \sum_{j=0}^{n-1} \exp(-i2\pi kj) \vec{y}[j-m] \quad (113)$$

$$= \sum_{m=0}^{n-1} \vec{x}[m] \exp(-i2\pi km) \vec{Y}[k] \quad \text{by Lemma 1.18} \quad (114)$$

$$= \vec{X}[k] \vec{Y}[k]. \quad (115)$$

□

Let $\Lambda_{\vec{Y}}$ contain the DFT of an arbitrary vector $\vec{y} \in \mathbb{C}^n$. We can express the convolution of \vec{y} with any vector $\vec{x} \in \mathbb{C}^n$ as

$$C_{\vec{y}} = \vec{x} * \vec{y} \quad (116)$$

$$= \frac{1}{n} W^* \Lambda_{\vec{Y}} \vec{X} \quad (117)$$

$$= \frac{1}{n} W^* \Lambda_{\vec{Y}} W \vec{x}. \quad (118)$$

This immediately implies that the discrete sinusoids in the columns of the DFT matrix are eigenvectors of the convolution matrix.

Corollary 2.8 (Eigendecomposition of circulant matrices). *A circulant matrix $C_{\vec{y}}$ corresponding to a vector \vec{y} has an eigendecomposition of the form*

$$C_{\vec{y}} = \frac{1}{n} W^* \Lambda_{\vec{Y}} W. \quad (119)$$

Convolution can be extended to two dimensions by using 2D shifts.

Definition 2.9 (Two-dimensional discrete convolution). *The circular convolution of two matrices $M_1, M_2 \in \mathbb{C}^{n \times n}$ is defined as*

$$M_1 * M_2[j, l] := \sum_{m=0}^{n-1} \sum_{u=0}^{n-1} M_1[m, u] M_2[j - m, l - u] \quad (120)$$

where the shifts are circular, so that $M_1[j, l] = M_1[j + n, l + n]$.

In two dimensions, convolution is again equivalent to multiplication in the 2D DFT domain. We omit the proof, which is similar to the one for the 1D case.

Theorem 2.10 (Convolution in space is multiplication in frequency). *Let $R := M_1 * M_2$ for $M_1, M_2 \in \mathbb{C}^{n \times n}$. Then*

$$\widehat{R}[k_1, k_2] = \widehat{M}_1[k_1, k_2] \widehat{M}_2[k_1, k_2]. \quad (121)$$

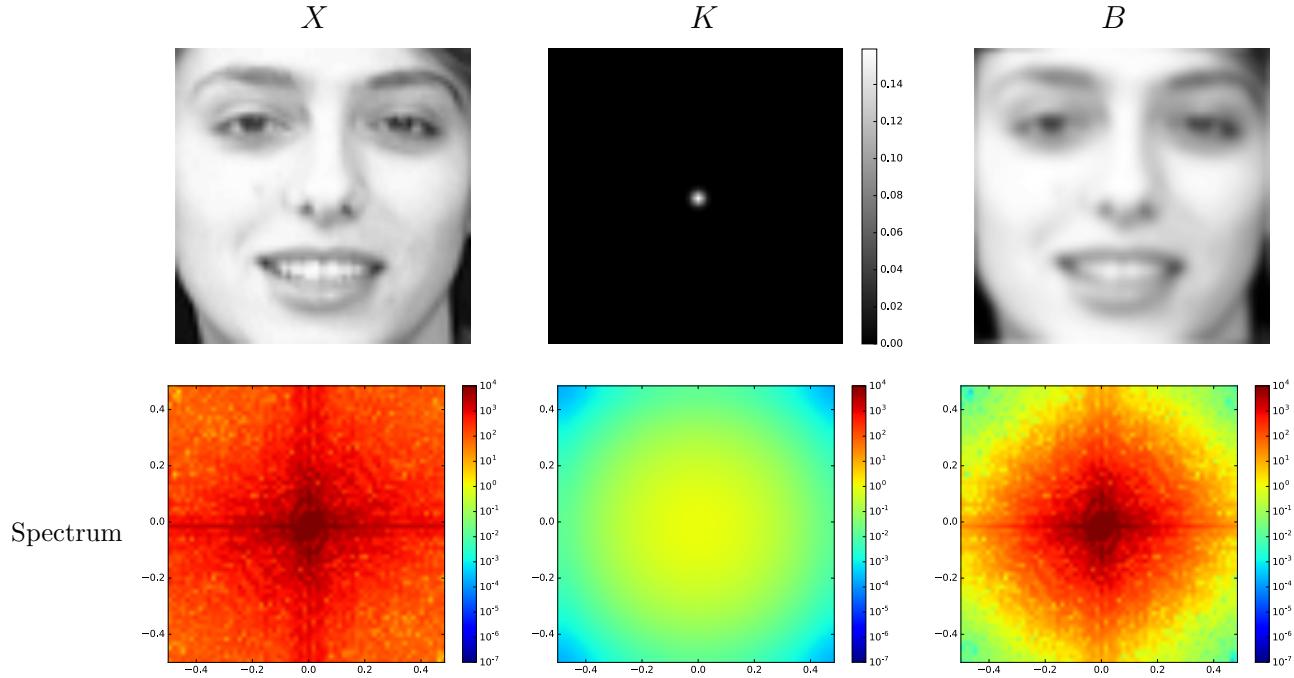


Figure 25: The top row shows a natural image (left), a blurring kernel (center) and the corresponding blurred image (right). The bottom row shows the magnitudes of the 2D DFT coefficients of the three images on a logarithmic scale.

2.3 Wiener deconvolution

In imaging, the resolution of lenses is limited by diffraction. If the resolution is low with respect to the number of pixels in the image, this is perceived as blur. A simplified model for a blurred image B is the convolution of the high-resolution images, which we represent by a matrix $X \in \mathbb{R}^{n \times n}$, and a convolution kernel $K \in \mathbb{R}^{n \times n}$ that depends on the optical system

$$B = K * X. \quad (122)$$

In the frequency domain, the 2D DFT of the image equals the product between the DFTs of the high resolution image and the convolution kernel

$$\widehat{B} = \widehat{K} \circ \widehat{X}, \quad (123)$$

where \circ denotes the Hadamard or entry-wise product. Figure 25 illustrates this model with an example. In the frequency domain, we can see how the high-end of the spectrum of the image is suppressed by the filter. Recovering the high-resolution image from these data is easy. We just need to invert the action of the kernel,

$$\widehat{X}_{\text{est}} = \widehat{K}_{\text{dec}} \circ \widehat{Y} \quad (124)$$

where the spectrum of the *deconvolution* kernel is the inverse of the spectrum of the convolution kernel

$$\widehat{K}_{\text{dec}}[k_1, k_2] := \frac{1}{\widehat{K}_{\text{dec}}[k_1, k_2]}, \quad 0 \leq k_1, k_2 \leq n - 1, \quad (125)$$

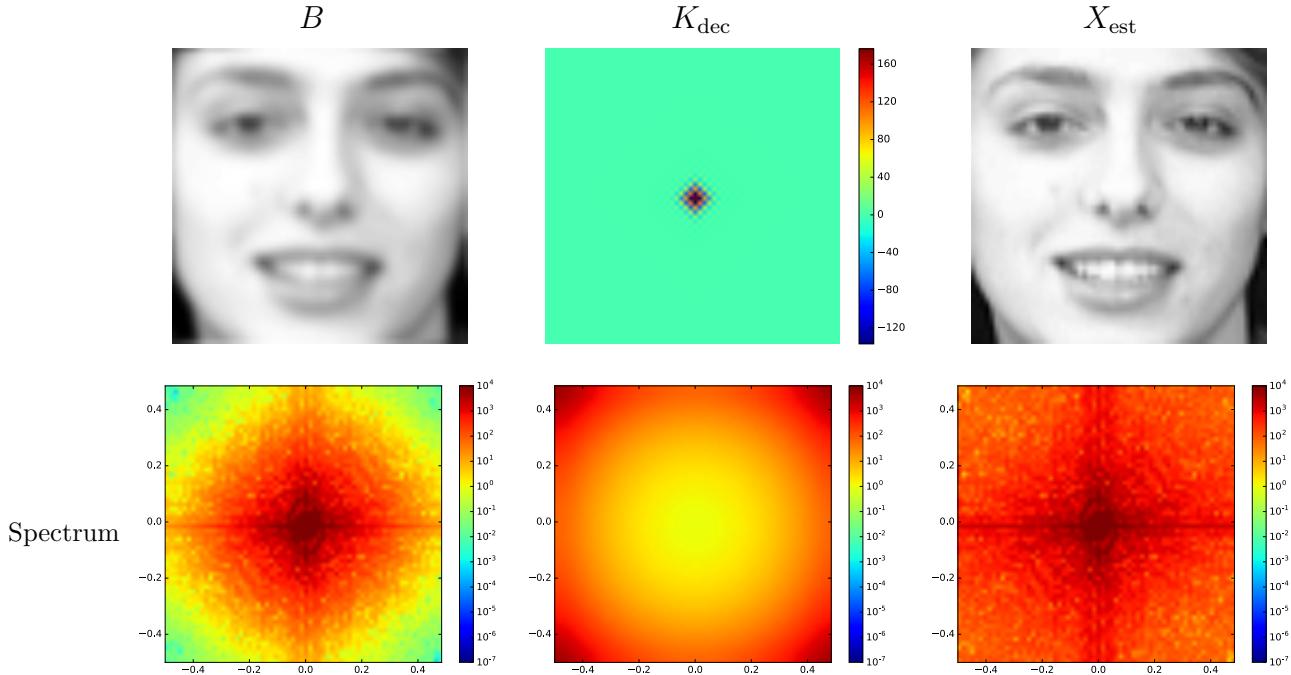


Figure 26: The top row shows a blurred image (left), a deconvolution kernel obtained by inverting the spectrum of the convolution kernel K in Figure 25 (center) and the corresponding deconvolved image (right). The bottom row shows the magnitudes of the 2D DFT coefficients of the three images on a logarithmic scale.

assuming that it is nonzero everywhere. Figure 26 shows that this scheme works perfectly on our simulated data. The deconvolution filter amplifies the high-frequency components to undo the effect of the convolution kernel.

Unfortunately, real data always contain noise. We take this into account by incorporating a noise term to our model

$$B_{\text{noisy}} = K * X + Z, \quad (126)$$

where $Z \in \mathbb{R}^{n \times n}$ represents the noise. Figure 27 shows the noisy data. As opposed to the image, which has most of its energy concentrated in the low frequencies, the energy in the noise is distributed uniformly across all the spectrum. This is not surprising, since the noise is iid Gaussian and the 2D DFT is an orthogonal transformation (up to a constant). By Theorem 2.3 in Lecture Notes 3, the DFT of the noise is consequently also Gaussian with a unit covariance matrix, so that the variance of each corresponding frequency coefficient is the same.

What happens if we apply our deconvolution scheme to the noisy blurred image? The result is catastrophic, as you can see in Figure 28. The reason is obvious when we look at this in the frequency domain. The high-end of the spectrum of the noisy blurred image is dominated by the noise. The deconvolution filter amplifies this high-frequency noise drowning out the actual image! In order to avoid this effect while deconvolving it is necessary to take into account the ratio between the noise level and the signal level at each frequency. Wiener filtering is a principled way of doing this if we can have a prior estimate of the spectral statistics of the signal and the noise.

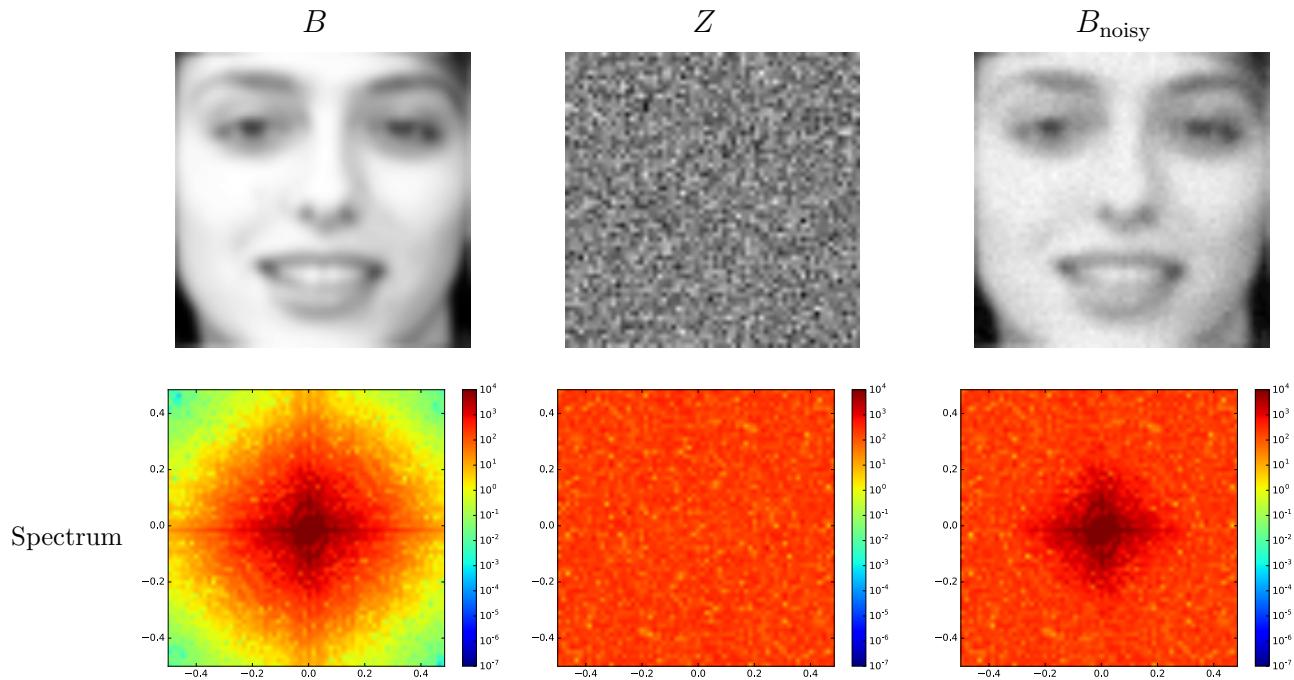


Figure 27: The top row shows a blurred image (left), additive Gaussian noise (center) and the corresponding noisy blurred image (right). The bottom row shows the magnitudes of the 2D DFT coefficients of the three images on a logarithmic scale.

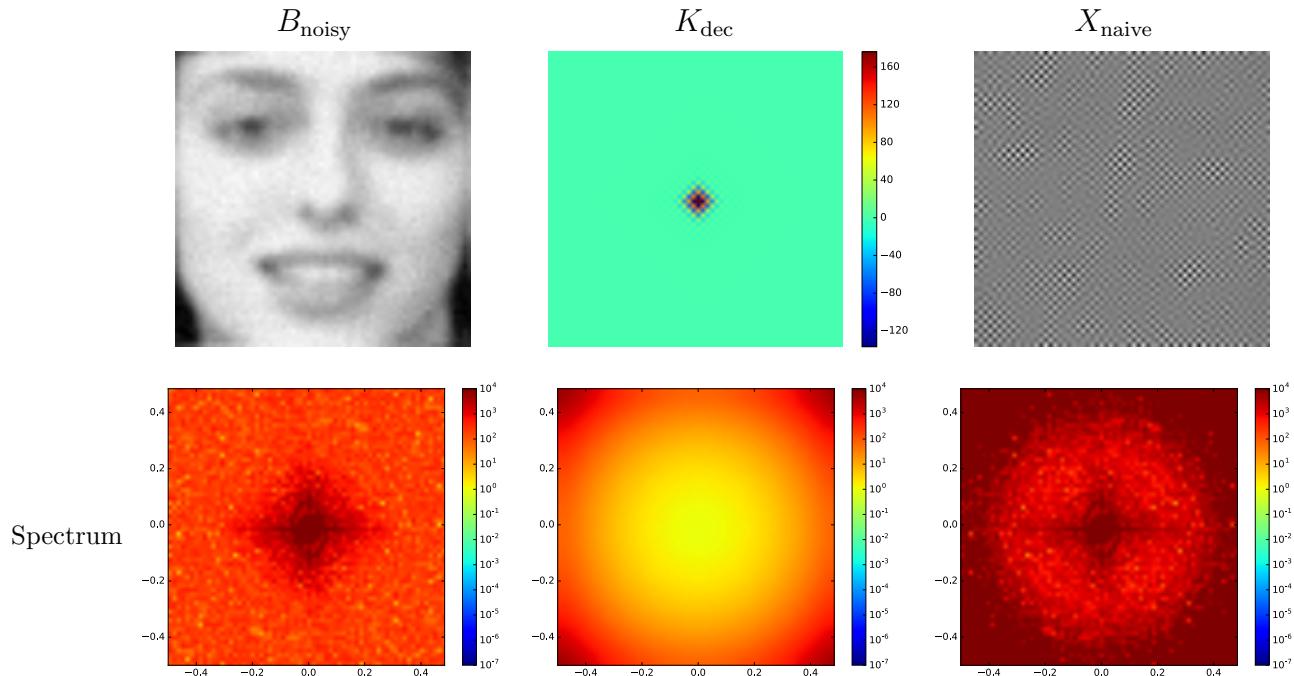


Figure 28: The top row shows a noisy blurred image (left), a deconvolution kernel obtained by inverting the spectrum of the convolution kernel K in Figure 25 (center) and the corresponding deconvolved image (right). The bottom row shows the magnitudes of the 2D DFT coefficients of the three images on a logarithmic scale.

The version of Wiener filtering that we present here makes the assumption that the DFT coefficients of the image and of the noise are uncorrelated. However, the method can be adapted to the case where the correlations are known. Let us model each DFT coefficient of the signal and the noise as random variables with known means and variances. For simplicity, we subtract their respective means, so that both random variables have zero mean. This makes it possible to interpret the random variables as vectors in the vector space of zero-mean random variables.

Theorem 2.11 (Vector space of zero-mean random variables). *Zero-mean complex-valued random variables form a vector space with the usual summing and multiplication operations. The zero vector is the random variable that equals one with probability one.*

Proof. If two random variables have zero mean, any linear combination of the variables also has zero mean because of linearity of expectation. \square

The covariance between two random variables is a valid inner product in this vector space, which means that uncorrelated random variables are orthogonal.

Theorem 2.12 (Inner product for zero-mean random variables). *The covariance*

$$\langle \mathbf{x}, \mathbf{y} \rangle := \text{Cov}(\mathbf{x}, \mathbf{y}) \tag{127}$$

$$= E(\mathbf{x}\bar{\mathbf{y}}) \tag{128}$$

is a valid inner product in the vector space of zero-mean random variables. The corresponding inner-product norm is the variance,

$$\|\mathbf{x}\|_{\langle \cdot, \cdot \rangle} = \text{Var}(\mathbf{x}) \tag{129}$$

Proof. This follows directly from linearity of expectation and the assumption that all the random variables are zero-mean. The fact that $\langle \mathbf{x}, \mathbf{x} \rangle = 0$ implies $\mathbf{x} = 0$ with probability one follows from Chebyshev's inequality. In more detail, if $\|\mathbf{x}\|_{\langle \cdot, \cdot \rangle} = 0$, for any $\epsilon > 0$

$$P(|\mathbf{x}| > \epsilon) \leq \frac{\text{Var}(\mathbf{x})}{\epsilon^2} = 0 \tag{130}$$

so \mathbf{x} with probability one. \square

The problem of estimating each DFT coefficient from the measured DFT coefficients now boils down to estimating a zero-mean random variable \mathbf{x} given a measurement of the form

$$\mathbf{y} = a\mathbf{x} + \mathbf{z} \tag{131}$$

where a represents the value of the DFT coefficient of the convolution kernel. The following theorem derives the best linear estimate of \mathbf{x} given \mathbf{y} assuming that the signal and the noise are uncorrelated.

Theorem 2.13 (Linear estimation). *Let*

$$\mathbf{y} = a\mathbf{x} + \mathbf{z} \tag{132}$$

where a is a known constant and \mathbf{x} and \mathbf{z} are uncorrelated zero-mean random variables with variances $\sigma_{\mathbf{x}}^2$ and $\sigma_{\mathbf{z}}^2$ respectively. The linear estimate of \mathbf{x} given \mathbf{y}

$$\mathbf{x}_{\text{MMSE}} := w\mathbf{y} \quad (133)$$

that minimizes the mean square error

$$\mathbb{E}((\mathbf{x} - \mathbf{x}_{\text{MMSE}})^2) \quad (134)$$

is given by

$$\mathbf{x}_{\text{MMSE}} = \frac{\bar{a}\sigma_{\mathbf{x}}^2\mathbf{y}}{|a|^2\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{z}}^2}. \quad (135)$$

Proof. We need to find the vector $\tilde{\mathbf{x}}$ in the span of \mathbf{y} that minimizes

$$\|\mathbf{x} - \tilde{\mathbf{x}}\|_{\langle \cdot, \cdot \rangle} = \sqrt{\mathbb{E}((\mathbf{x} - \tilde{\mathbf{x}})^2)}. \quad (136)$$

By basic linear algebra, this is just the orthogonal projection of \mathbf{y} onto \mathbf{x} ! The projection is given by

$$\mathcal{P}_{\text{span}(\mathbf{y})}\mathbf{x} = \left\langle \mathbf{x}, \frac{\mathbf{y}}{\|\mathbf{y}\|_{\langle \cdot, \cdot \rangle}} \right\rangle \frac{\mathbf{y}}{\|\mathbf{y}\|_{\langle \cdot, \cdot \rangle}} \quad (137)$$

This implies that

$$w = \frac{1}{\|\mathbf{y}\|_{\langle \cdot, \cdot \rangle}} \left\langle \mathbf{x}, \frac{\mathbf{y}}{\|\mathbf{y}\|_{\langle \cdot, \cdot \rangle}} \right\rangle \quad (138)$$

$$= \frac{\langle \mathbf{x}, a\mathbf{x} + \mathbf{z} \rangle}{\|a\mathbf{x} + \mathbf{z}\|_{\langle \cdot, \cdot \rangle}^2} \quad (139)$$

$$= \frac{\bar{a}\|\mathbf{x}\|_{\langle \cdot, \cdot \rangle}^2 + \bar{a}\langle \mathbf{x}, \mathbf{z} \rangle}{\|a\mathbf{x} + \mathbf{z}\|_{\langle \cdot, \cdot \rangle}^2} \quad (140)$$

$$= \frac{\bar{a}\|\mathbf{x}\|_{\langle \cdot, \cdot \rangle}^2}{|a|^2\|\mathbf{x}\|_{\langle \cdot, \cdot \rangle}^2 + \|\mathbf{z}\|_{\langle \cdot, \cdot \rangle}^2} \quad \text{by orthogonality and the Pythagorean theorem} \quad (141)$$

$$= \frac{\bar{a}\sigma_{\mathbf{x}}^2}{|a|^2\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{z}}^2}. \quad (142)$$

□

Wiener filtering consists of estimating the statistics of the signal and the noise, and then applying the best linear estimate at each frequency coefficient. In this version of the method we assume that the DFT coefficients of the signal and noise are all uncorrelated and that the noise has zero mean.

Algorithm 2.14 (Wiener filtering). *Given a noisy blurred image $B_{\text{noisy}} \in \mathbb{R}^{n \times n}$ and a kernel $K \in \mathbb{R}^{n \times n}$, apply the following steps:*

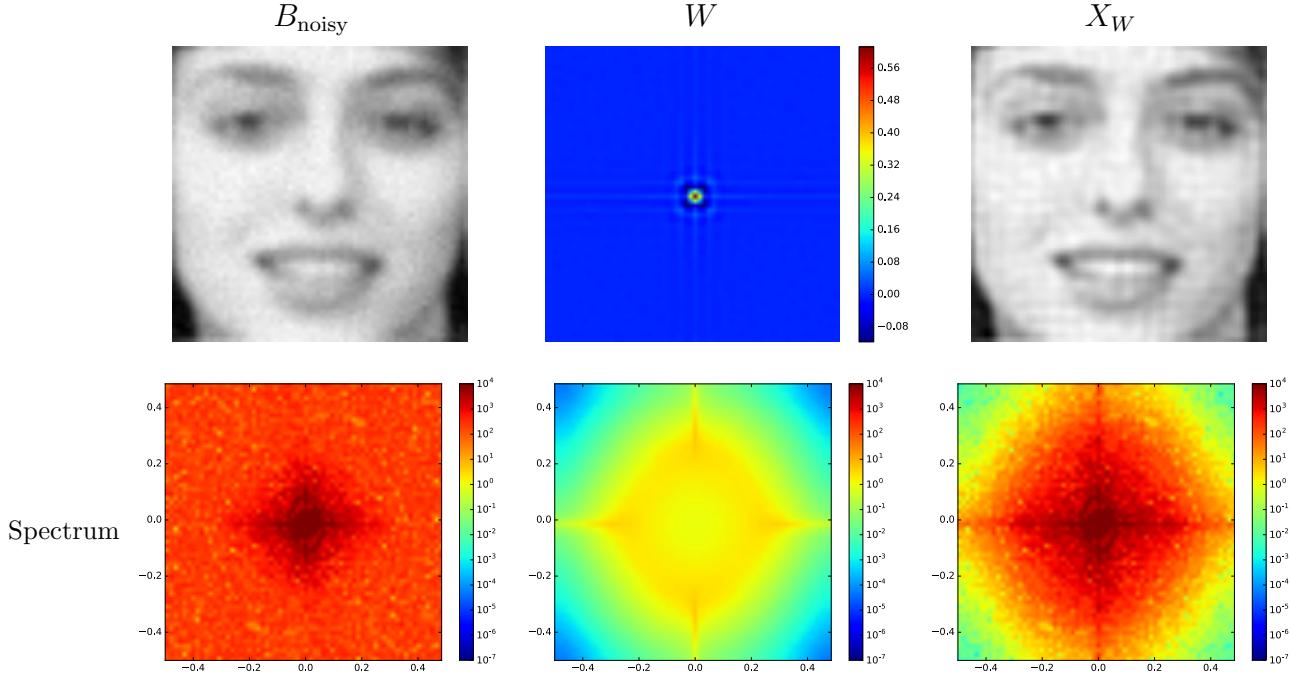


Figure 29: The top row shows a blurred image (left), the Wiener filter obtained by applying Algorithm 2.14 (center) and the corresponding deconvolved image (right). The bottom row shows the magnitudes of the 2D DFT coefficients of the three images on a logarithmic scale.

1. Estimate the variance of each 2D DFT coefficient of the noise $\sigma_Z [k_1, k_2]^2$.
2. Estimate the mean $\mu_X [k_1, k_2]$ and variance $\sigma_X [k_1, k_2]^2$ of each 2D DFT coefficient $\widehat{X} [k_1, k_2]$ of the image using a database of images.
3. Compute the 2D DFT coefficients of the noisy blurred image $\widehat{B}_{\text{noisy}}$ and the kernel \widehat{K} .
4. For $0 \leq k_1, k_2 \leq n - 1$
 - Center $\widehat{B}_{\text{noisy}}[k_1, k_2]$ by subtracting $\widehat{K}[k_1, k_2]\mu_X [k_1, k_2]$.
 - Set

$$W[k_1, k_2] := \frac{\overline{\widehat{K}[k_1, k_2]}\sigma_X [k_1, k_2]^2}{|\widehat{K}[k_1, k_2]|^2\sigma_X [k_1, k_2]^2 + \sigma_Z [k_1, k_2]^2} \quad (143)$$

$$X_W := \mu_X [k_1, k_2] + W[k_1, k_2]\widehat{B}_{\text{noisy}} [k_1, k_2]. \quad (144)$$

5. Compute the inverse 2D DFT of X_W .

Figure 29 shows the result of applying Wiener filtering to our running example.

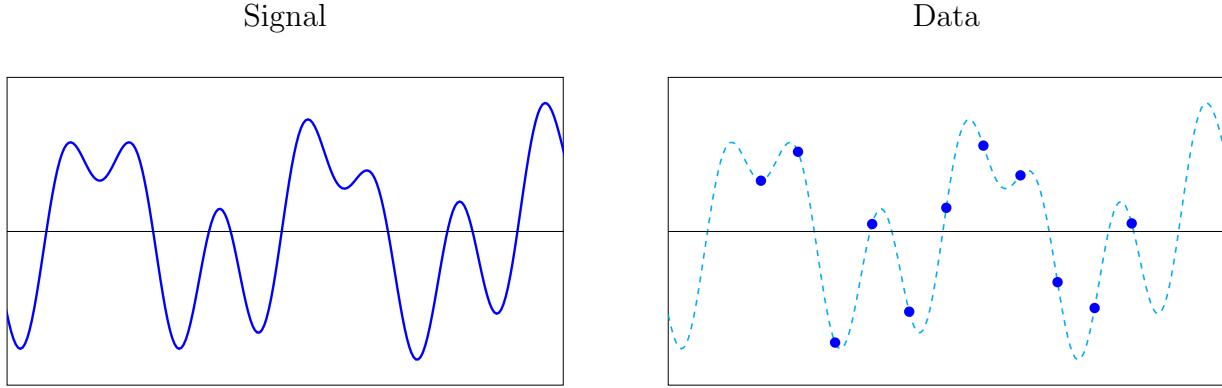


Figure 30: The goal of spectral super-resolution is to estimate the frequencies of a multisinusoidal signal like the one on the left from the finite samples shown on the right.

3 Spectral Super-resolution

3.1 The spectral super-resolution problem

The aim of spectral super-resolution is to estimate the components of a multisinusoidal signal from finite data. This problem arises for example in radar, radio telescopes and other applications where we want to determine the direction of arrival of a propagating wave using an array of sensors. Here we will look at a basic version of the problem. Consider a multisinusoidal signal of the form

$$g(t) := \sum_{j=1}^s \vec{c}[j] \exp(-i2\pi f_j t), \quad (145)$$

which is the weighted sum of s complex sinusoids. The goal of spectral super-resolution is to estimate the frequencies f_1, f_2, \dots, f_s and the corresponding complex-valued amplitudes $\vec{c}[1], \vec{c}[2], \dots, \vec{c}[s]$ from a finite number of samples of g , as illustrated in Figure 30 (in the figure the amplitudes are real for ease of visualization). We assume that the frequencies lie in a unit interval, for example in $[-1/2, 1/2]$ and the samples are measured at integer values from $-(n-1)/2$ to $(n-1)/2$ (n is assumed to be odd to simplify the exposition).

One can model the *spectrum* of g as spikes with amplitudes c_1, c_2, \dots, c_s located at the frequencies f_1, f_2, \dots, f_s . To make this mathematically precise, we introduce the Dirac measure.

Definition 3.1 (Dirac measure). A Dirac measure $\delta_{[\tau]}$ is a measure associated to a point τ , which assigns a value of one to sets that contain τ and zero to sets that do not,

$$\int_{\mathcal{S}} \delta_{[\tau]}(du) = \begin{cases} 1 & \text{if } \tau \in \mathcal{S}, \\ 0 & \text{otherwise.} \end{cases} \quad (146)$$

Even though it is often called a Dirac delta function, the Dirac measure is not a function. It can however be interpreted as a distribution or a generalized function such that for any function h we have

$$\int_{\mathcal{S}} h(u) \delta_{[\tau]}(du) = \begin{cases} h(\tau) & \text{if } \tau \in \mathcal{S}, \\ 0 & \text{otherwise.} \end{cases} \quad (147)$$

Although a rigorous justification is beyond the scope of these notes, one can define the Fourier series of a distribution and in particular of a Dirac measure. The coefficients are obtained by integrating the complex sinusoidal atoms h_k , $k \in \mathbb{Z}$, against the measure. We can now interpret the spectral super-resolution problem as that of estimating the support of the measure

$$\mu_g := \sum_{j=1}^s \vec{c}[j] \delta_{[f_j]} \quad (148)$$

from a finite subset of Fourier coefficients

$$\int_{-1/2}^{1/2} \overline{h_k(u)} \mu_g(\mathrm{d}u) = \sum_{j=1}^s \vec{c}[j] \int_{-1/2}^{1/2} \overline{h_k(u)} \delta_{[f_j]}(\mathrm{d}u) \quad (149)$$

$$= \sum_{j=1}^s \vec{c}[j] \exp(-i2\pi k f_j) \quad (150)$$

$$= g(k), \quad -\frac{n-1}{2} \leq k \leq \frac{n-1}{2}. \quad (151)$$

In the next section it will become apparent why this is a super-resolution problem.

3.2 The periodogram

In order to estimate the frequencies of g from the available samples, we can compute the truncated Fourier series of the measure μ_g . This is known as the periodogram in the signal processing literature.

Definition 3.2 (Periodogram). *The periodogram of a vector of data $\vec{y} \in \mathbb{C}^n$ is defined as*

$$P_{\vec{y}}(u) := \sum_{k=-\frac{n-1}{2}}^{\frac{n-1}{2}} \vec{y}[k] h_k(u), \quad (152)$$

where h_k is a complex sinusoid with integer frequency k and the entries of the data are numbered from $-(n-1)/2$ to $(n-1)/2$ for ease of exposition.

The periodogram can be interpreted as the convolution between a Dirichlet kernel with cut-off frequency $(n-1)/2$ and the measure μ_g .

Lemma 3.3. *The periodogram of $g(-(n-1)/2), \dots, g((n-1)/2)$ where g is the multisinusoidal function defined by equation (145) equals*

$$P_g(u) = \sum_{j=1}^s \vec{c}[j] d_{[f_j]}(u). \quad (153)$$

Proof. By Definition 1.10 and Lemma 1.12 the Fourier coefficients of a Dirichlet kernel with cut-off frequency $(n-1)/2$ shifted by f are equal to

$$D_{[f]}[k] := \begin{cases} \exp(-i2\pi kf) & \text{if } |k| \leq (n-1)/2 \\ 0 & \text{otherwise.} \end{cases} \quad (154)$$

This implies that the samples

$$g(k) = \sum_{j=1}^s \vec{c}[j] D_{[f_j]}[k], \quad -\frac{n-1}{2} \leq k \leq \frac{n-1}{2}, \quad (155)$$

are exactly equal to the Fourier coefficients of the periodogram, so that

$$P_g(u) = \sum_{k=-\frac{n-1}{2}}^{\frac{n-1}{2}} g(k) h_k(u) \quad (156)$$

$$= \sum_{j=1}^s \vec{c}[j] \sum_{k=-\frac{n-1}{2}}^{\frac{n-1}{2}} D_{[f_j]}[k] h_k(u) \quad (157)$$

$$= \sum_{j=1}^s \vec{c}[j] d_{[f_j]}(u). \quad (158)$$

□

Recall that the width of the main lobe of a Dirichlet kernel is inversely proportional to its cut-off frequency, which in this case equals $(n-1)/2$. As we gather more samples, the widths become narrower and narrower, revealing the location of the frequencies of interest. The *resolution* at which we observe the spectrum μ_g is consequentially tied to the number of data. This is illustrated in Figure 31.

Computing the periodogram does not solve the super-resolution problem, it just allows to visualize a lower resolution version of μ_g . If the frequencies f_1, \dots, f_s are far apart, the local maxima of P are a good indication of their location. The problem with this approach is that the side lobes corresponding to large amplitudes may mask the presence of smaller spikes. As a result, the periodogram is not very useful if the spikes are not far enough from each other or if their amplitudes differ substantially, *even if no noise is present in the data*. The image at the center of Figure 32 illustrates this: detecting some of the lower-amplitude spikes from the periodogram is impossible.

In order to alleviate the interference caused by the side lobes of the Dirichlet kernel, one can *window* the data before computing the periodogram.

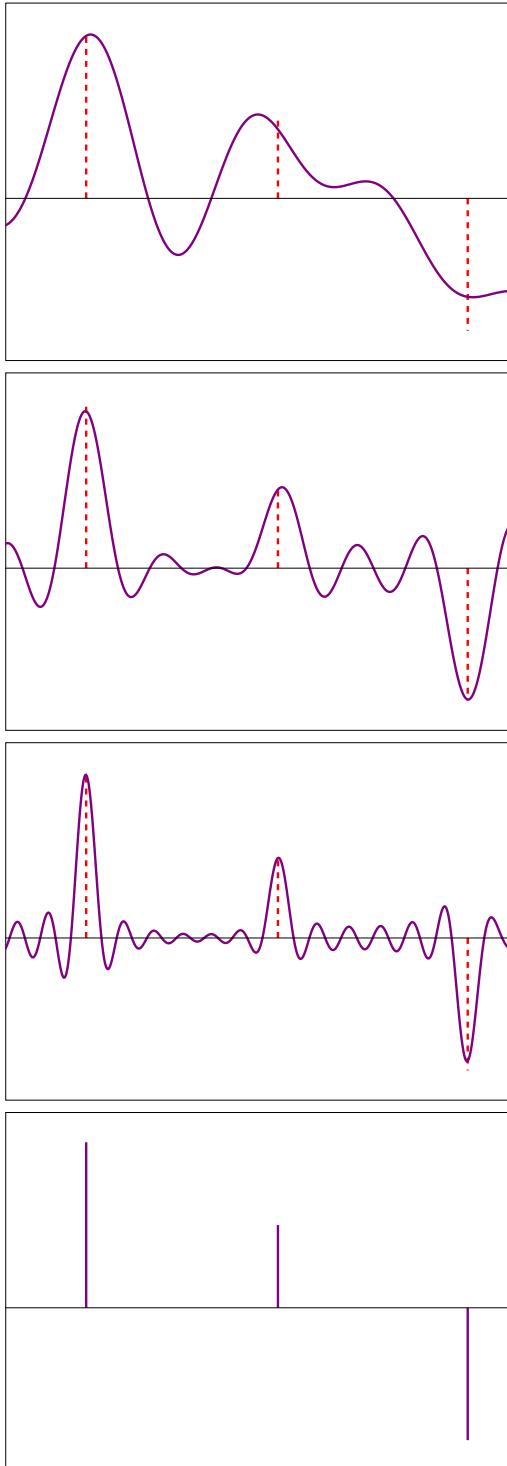
Definition 3.4 (Windowed periodogram). *Let us define a bandlimited window function w with Fourier coefficients $W[k]$ that are only nonzero if $|k| \leq (n-1)/2$. The windowed periodogram of a vector of data $\vec{y} \in \mathbb{C}^n$ is defined as*

$$P_{w,\vec{y}}(u) := \sum_{k=-\frac{n-1}{2}}^{\frac{n-1}{2}} W[k] g(k) h_k(u). \quad (159)$$

Following the exact same reasoning as in the proof of Lemma 3.3,

$$P_{w,\vec{y}}(u) := \sum_{j=1}^s \vec{c}[j] w_{[f_j]}[u]. \quad (160)$$

Periodogram



Data

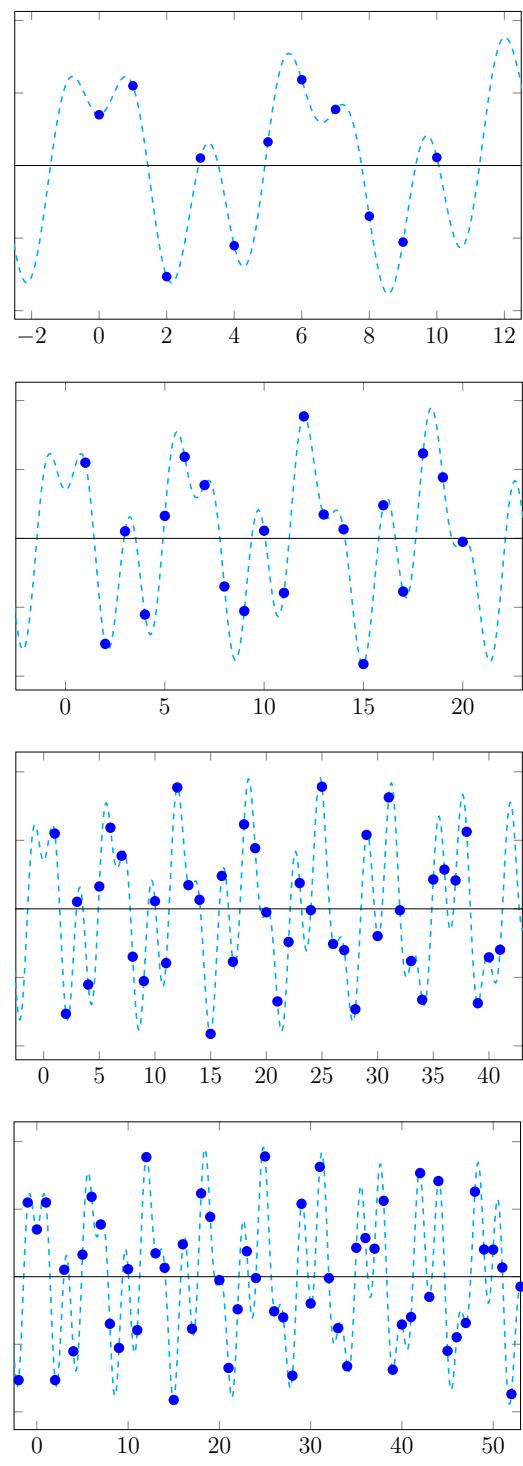


Figure 31: Periodogram (left) and corresponding samples (right) for different numbers of measurements (increasing downwards). In the limit where $n \times \infty$ the frequencies are completely resolved.

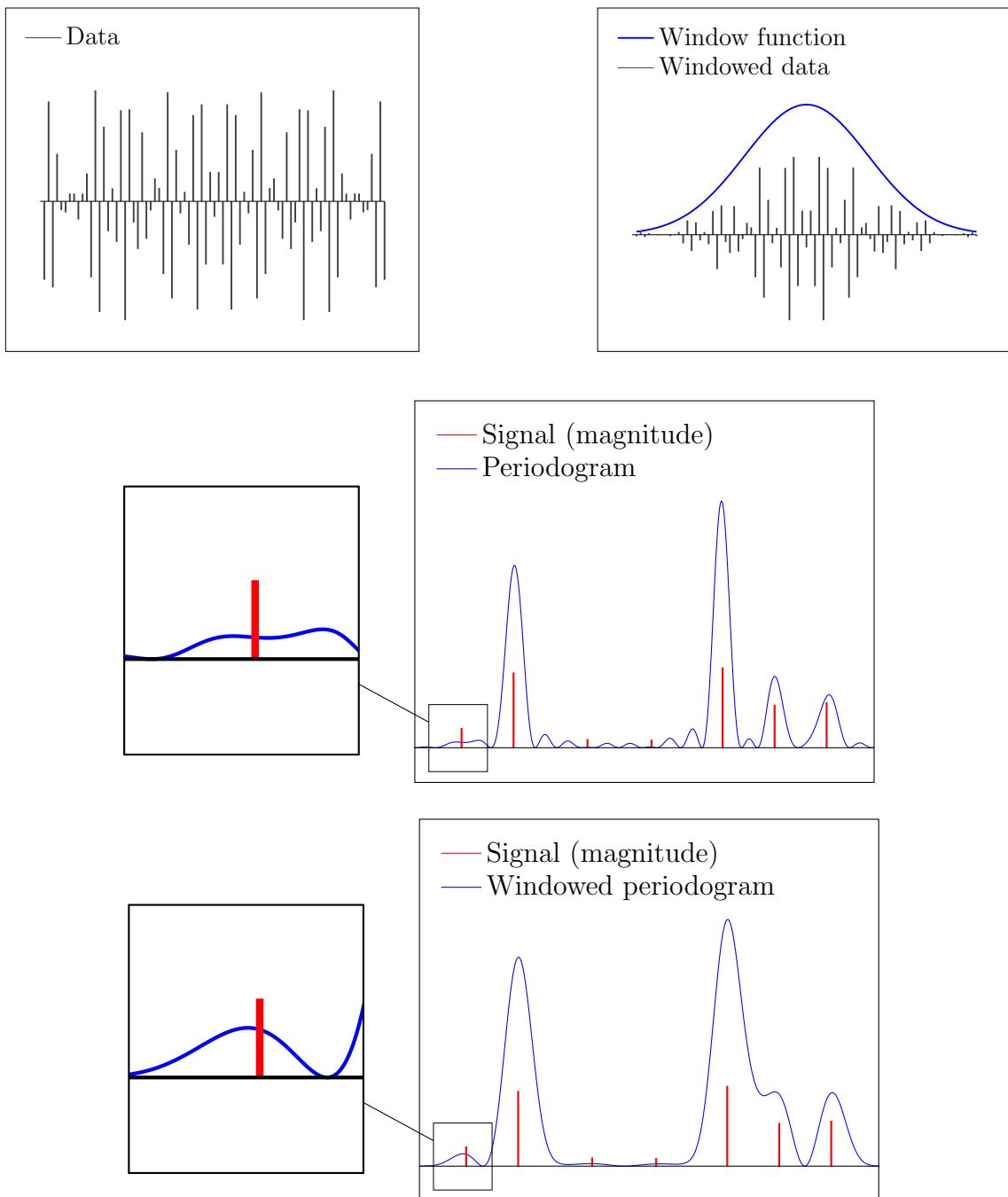


Figure 32: Data before (top left) and after applying a window function (top right). No noise is added to the data. Below we see the periodogram (center) and windowed periodogram (bottom) computed from the data. The scaling of the periodograms is set so that both the large and small peaks can be seen on the plot.

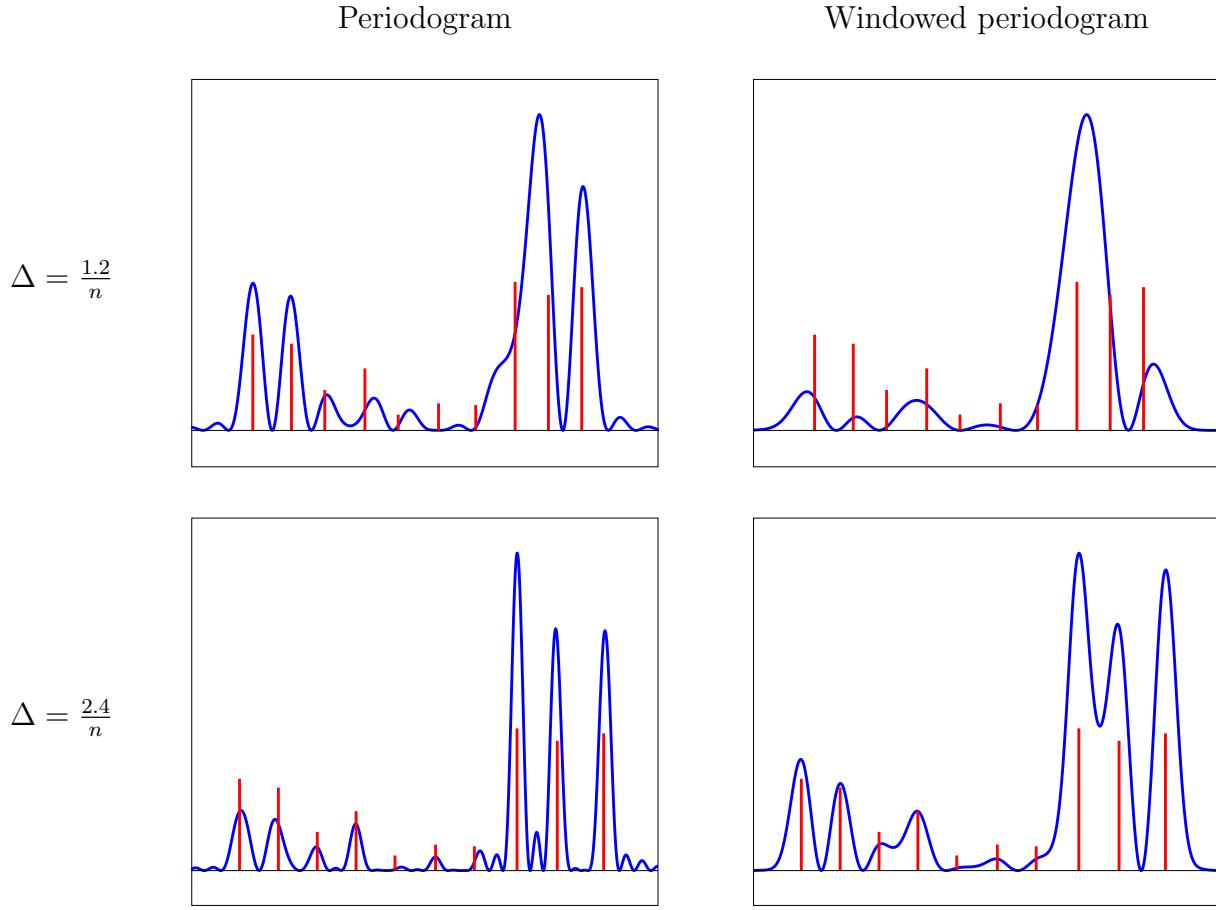


Figure 33: Periodogram (left) and windowed periodogram (right) for two signals with identical amplitudes but different minimum separations.

where w denotes the inverse Fourier transform of the window function. Ideally, w should be as *spiky* as possible to make it easier to locate the frequency locations from the windowed periodogram. However, this is challenging due to the constraint that \hat{w} has finite support and hence w is a low-pass function.

In the image on the top right of Figure 32 we apply a Gaussian window to the data. To be more precise, we set W to be a truncated Gaussian, so that w is also approximately Gaussian. The resulting periodogram, shown at the center of Figure 32, has much less *spectral leakage* from the largest signal components, due to the fact that the Gaussian window has lower side lobes than the periodized sinc. However, the latter is spikier at the origin, which allows to better distinguish neighboring spikes with similar amplitudes. In general, designing an adequate window implies finding a good tradeoff between the width of the main lobe and the height of the side lobes. We refer the reader to [2] for a detailed account of design considerations and types of window function.

3.3 Prony's method

Prony's method solves the spectral super-resolution problem exactly in the absence of noise by encoding the position of the s frequencies of interest as the zeros of a trigonometric polynomial of

order s . The following theorem shows that such a polynomial always exists.

Theorem 3.5 (Prony polynomial). *Given any set of frequencies f_1, f_2, \dots, f_s in the unit interval, there exists a nonzero complex polynomial of order s*

$$p(z) := \sum_{k=0}^s P[k] z^k, \quad (161)$$

which we call a Prony polynomial, such that its s roots are equal to $\exp(2\pi f_1), \exp(2\pi f_2), \dots, \exp(2\pi f_s)$.

Proof. Consider the polynomial

$$p(z) := \prod_{j=1}^s (1 - \exp(-i2\pi f_j) z). \quad (162)$$

If we expand the product, we have

$$p(z) = 1 + \sum_{k=1}^s P[k] z^k \quad (163)$$

for some $P[1], \dots, P[s]$. Since $p(0) = 1$, the polynomial is nonzero and by the fundamental theorem of algebra it has at most s roots. By construction,

$$p(\exp(i2\pi f_j)) = 0 \quad (164)$$

for $1 \leq j \leq s$, which establishes the result. \square

If we are able to compute such a polynomial, then finding its roots immediately reveals the frequencies of interest. Prony's method consists of building a system of linear equations from the available data, such that its solution is equal to the coefficients of the Prony polynomial. The following theorem shows how to build the system.

Theorem 3.6 (Prony system). *Let $P[k]$ denote the k th coefficient of the Prony polynomial defined in Theorem 3.5 and let g be the multisinusoidal function defined by equation (145), for any integer b*

$$\sum_{l=0}^s P[l] g[l-b] = 0. \quad (165)$$

Proof.

$$\sum_{l=0}^s P[l]g[l-b] = \sum_{l=0}^s P[l] \int_{-1/2}^{1/2} \overline{h_{l-b}(u)} \mu_g(du) \quad \text{by (151)} \quad (166)$$

$$= \int_{-1/2}^{1/2} \exp(i2\pi bu) \sum_{l=0}^s P[l] \exp(-i2\pi lu) \mu_g(du) \quad (167)$$

$$= \int_{-1/2}^{1/2} \exp(i2\pi bu) p(\exp(-i2\pi u)) \mu_g(du) \quad (168)$$

$$= \sum_{j=1}^s \bar{c}[j] \exp(i2\pi bf_j) p(\exp(-i2\pi f_j)) \quad (169)$$

$$= 0. \quad (170)$$

□

Notice that equation (165) only involves samples of g between $-b$ and $s - b$. Prony's method consists of setting up enough such equations in a system so that the solution yields the coefficients of the Prony polynomial, which can then be used to estimate the frequencies.

Algorithm 3.7 (Prony's method). *The input is the number of frequencies s that we aim to estimate and $2s + 1$ uniform samples of multisinusoidal function defined by equation (145).*

1. Form the system of equations

$$\begin{bmatrix} g(1) & g(2) & \cdots & g(s) \\ g(0) & g(1) & \cdots & g(s-1) \\ \cdots & \cdots & \cdots & \cdots \\ g(-s+2) & g(-s+3) & \cdots & g(1) \end{bmatrix} \vec{P} = - \begin{bmatrix} g(0) \\ g(-1) \\ \cdots \\ g(-s+1) \end{bmatrix}, \quad (171)$$

where $\vec{P} \in \mathbb{C}^s$.

2. Solve the system to obtain \vec{P} .

3. Root the polynomial

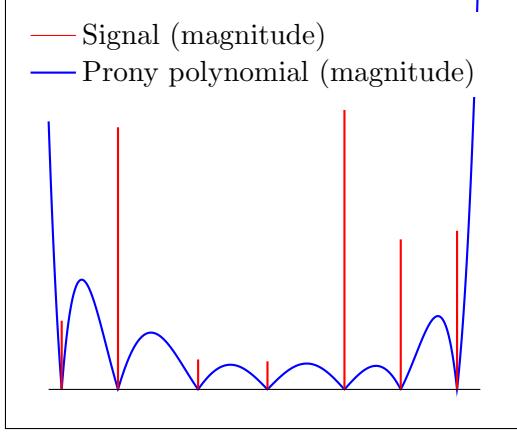
$$p(z) := 1 + \sum_{k=1}^s \vec{P}[k]z^k, \quad (172)$$

to obtain its s roots z_1, \dots, z_s .

4. For every root on the unit circle $z_j = \exp(i2\pi\tau)$ include τ in the set of estimated frequencies.

This procedure is guaranteed to achieve exact recovery of the original signal. This implies that in a noiseless scenario spectral super-resolution is achieved using only $n = 2s + 1$ measurements, which is essentially optimal since we need to estimate $2s$ free parameters (the s frequencies and the corresponding amplitudes).

No noise



SNR = 140 dB

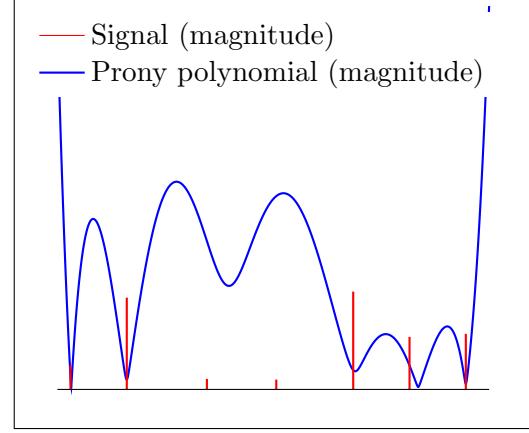


Figure 34: Prony polynomial applied on noiseless data (left). The image on the right shows the effect of adding a very small quantity of noise to the data. The roots of the polynomial no longer coincide with the frequencies of the original signal. Note that the vertical axis is scaled differently in the two images.

Lemma 3.8. *In the absence of noise, Prony's method recovers the frequencies of a multisinusoidal function of the form (145) exactly.*

Proof. The coefficients of the polynomial (161) are a feasible solution for the system of equations (171). In fact, they are the unique solution. To show this we compute the factorization

$$\begin{bmatrix} g(1) & g(2) & \cdots & g(s) \\ g(0) & g(1) & \cdots & g(s-1) \\ \vdots & & & \\ g(-s+2) & g(-s+3) & \cdots & g(1) \end{bmatrix} = \begin{bmatrix} e^{-i2\pi f_1} & e^{-i2\pi f_2} & \cdots & e^{-i2\pi f_s} \\ 1 & 1 & \cdots & 1 \\ \vdots & & & \\ e^{-i2\pi(2-s)f_1} & e^{-i2\pi(2-s)f_2} & \cdots & e^{-i2\pi(2-s)f_s} \end{bmatrix} \begin{bmatrix} \vec{c}[1] & 0 & \cdots & 0 \\ 0 & \vec{c}[2] & \cdots & 0 \\ \vdots & & & \\ 0 & 0 & \cdots & \vec{c}[s] \end{bmatrix} \begin{bmatrix} 1 & e^{-i2\pi f_1} & \cdots & e^{-i2\pi(s-1)f_1} \\ 1 & e^{-i2\pi f_2} & \cdots & e^{-i2\pi(s-1)f_2} \\ \vdots & & & \\ 1 & e^{-i2\pi f_s} & \cdots & e^{-i2\pi(s-1)f_s} \end{bmatrix}. \quad (173)$$

The diagonal matrix is full rank as long as all the coefficients $\vec{c}[j]$ are nonzero, whereas the two remaining matrices are full rank by the following lemma, proved in Section 4.1 of the appendix.

Lemma 3.9 (Vandermonde matrix). *For any distinct set of s nonzero complex numbers z_1, z_2, \dots, z_s*

and any positive integers m_1, m_2, s such that $m_2 - m_1 + 1 \geq s$ the Vandermonde matrix

$$\begin{bmatrix} z_1^{m_1} & z_2^{m_1} & \cdots & z_s^{m_1} \\ z_1^{m_1+1} & z_2^{m_1+1} & \cdots & z_s^{m_1+1} \\ z_1^{m_1+2} & z_2^{m_1+2} & \cdots & z_s^{m_1+2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{m_2} & z_2^{m_2} & \cdots & z_s^{m_2} \end{bmatrix} \quad (174)$$

is full rank.

As a result, the matrix in (171) is full rank, so the system of equations has a unique solution equal to (161). This completes the proof. \square

Unfortunately, Prony's method as presented above cannot be applied to real data even if the signal-to-noise ratio is exceptionally high. The image on the left of Figure 34 shows how the Prony polynomial allows to super-resolve the spectrum to very high accuracy from noiseless data. However, on the right we see the result of applying the method to data that have a very small amount of noise (the ratio between the ℓ_2 norm of the noise and the noiseless data is around 10^{-8} !). The roots of the Prony polynomial are perturbed away from the points of the unit circle that correspond to the true frequencies, so that it is no longer possible to achieve accurate spectral super-resolution. It is consequently necessary to adapt the method to deal with noisy data if there is to be any hope of applying it in any realistic scenario. This is the subject of the following section.

3.4 Subspace methods

In this section we consider the spectral super-resolution problem when noise is added to the data. First, we will generalize Prony's method so that it can use more data than just $2s + 1$ samples. Applying Prony's method is equivalent to finding a nonzero vector in the null space of $Y(s+1)^T$, where $Y(m)$ is defined for any integer m as the Hankel matrix

$$Y(m) := \begin{bmatrix} \vec{y}[0] & \vec{y}[1] & \cdots & \vec{y}[n-m] \\ \vec{y}[1] & \vec{y}[2] & \cdots & \vec{y}[n-m+1] \\ \vdots & \vdots & \ddots & \vdots \\ \vec{y}[m-1] & \vec{y}[m] & \cdots & \vec{y}[n-1] \end{bmatrix}, \quad (175)$$

where

$$\vec{y}[k] = g(-k+1) \quad 0 \leq k \leq n. \quad (176)$$

The vector in the null space of $Y(s+1)^T$ corresponds to the coefficients of the Prony polynomial, which we can root to find the frequencies of g .

In the absence of noise, $Y(m)$ can be decomposed in the following way

$$Y(m) = \begin{bmatrix} \vec{a}_{0:m-1}(f_1) & \vec{a}_{0:m-1}(f_2) & \cdots & \vec{a}_{0:m-1}(f_s) \end{bmatrix} \begin{bmatrix} c_1 & 0 & \cdots & 0 \\ 0 & c_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & c_s \end{bmatrix} \begin{bmatrix} \vec{a}_{0:n-m}(f_1)^T \\ \vec{a}_{0:n-m}(f_2)^T \\ \cdots \\ \vec{a}_{0:n-m}(f_s)^T \end{bmatrix} \\ = A_{0:m-1} C A_{0:m}^T, \quad (177)$$

where for $k > 0$ we define

$$\vec{a}_{0:k}(u) := \begin{bmatrix} 1 & \exp(-i2\pi u) & \exp(-i2\pi 2u) & \cdots & \exp(-i2\pi ku) \end{bmatrix}^T, \quad (178)$$

$$A_{0:k} := \begin{bmatrix} \vec{a}_{0:k}(f_1) & \vec{a}_{0:k}(f_2) & \cdots & \vec{a}_{0:k}(f_s) \end{bmatrix}. \quad (179)$$

This decomposition suggests an alternative way of estimating the spectrum from $Y(m)$: finding sinusoidal atoms $\vec{a}_{0:m-1}(u)$ that are in the column space of $Y(m+1)$. Lemma 3.10 below proves that the only atoms of this form that belong to the column space of $Y(m)$ are precisely $\vec{a}_{0:m-1}(f_1), \vec{a}_{0:m-1}(f_2), \dots, \vec{a}_{0:m-1}(f_s)$.

In order to apply the same idea in the presence of noise, where

$$\vec{y}[k] = g(-k+1) + \vec{z}[k] \quad 0 \leq k \leq n, \quad (180)$$

for some additive perturbation \vec{z} , we check what atoms are *close* to the column space of $Y(m)$. To quantify this we compute the orthogonal complement \mathcal{N} of the column space of $Y(m)$ and construct the *pseudospectrum*

$$P_{\mathcal{N}}(u) = \log \frac{1}{|\mathcal{P}_{\mathcal{N}}(\vec{a}_{0:m-1}(u))|^2}, \quad (181)$$

where $\mathcal{P}_{\mathcal{N}}$ denotes a projection onto \mathcal{N} . If an atom is almost orthogonal to \mathcal{N} then $P_{\mathcal{N}}$ will have a very large value at that point.

The following lemma shows that the maxima of the pseudospectrum reveal the frequencies of interest of the signal in the noiseless case.

Lemma 3.10. *Let \mathcal{N} be the null space of the empirical covariance matrix $\Sigma(m)$ for $m \geq s$. Then*

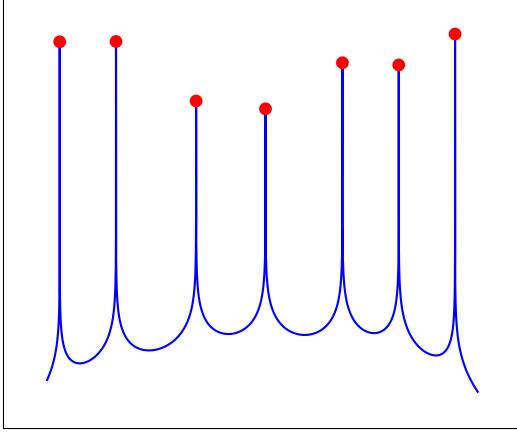
$$P_{\mathcal{N}}(f_j) = \infty, \quad 1 \leq j \leq s, \quad (182)$$

$$P_{\mathcal{N}}(u) < \infty, \quad \text{for } u \notin \{f_1, \dots, f_s\}. \quad (183)$$

Proof. By (177) the atoms $\vec{a}_{0:m-1}(f_1), \dots, \vec{a}_{0:m-1}(f_s)$ span the column space of $Y(m)$ and $\Sigma(m)$. As a result they are orthogonal to the null space \mathcal{N} of the empirical covariance matrix, which proves (182).

We prove (183) by contradiction. The atoms $\vec{a}_{0:m-1}(f_1), \dots, \vec{a}_{0:m-1}(f_s)$ span the orthogonal complement to \mathcal{N} . As a result, if $\vec{a}_{0:m-1}(u)$ is orthogonal to \mathcal{N} for some u then $\vec{a}_{0:m-1}(u)$ is in the span of $\vec{a}_{0:m-1}(f_1), \dots, \vec{a}_{0:m-1}(f_s)$. This would imply that $A_{0:m-1}^T(T \cup \{u\})$ is not full rank, which can only hold if $u \in \{f_1, \dots, f_s\}$ by Lemma 3.9. \square

No noise



SNR = 140 dB

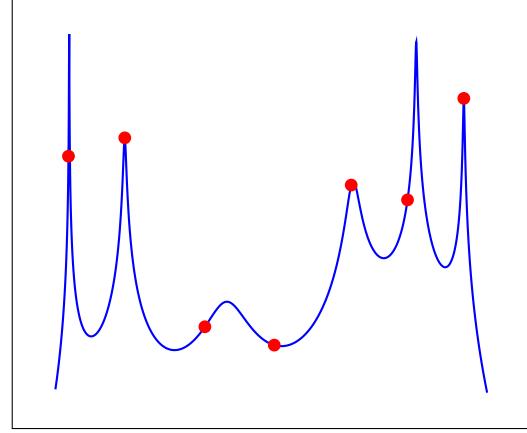


Figure 35: Pseudospectrum corresponding to the data used to construct the Prony polynomials in Figure 34. The true frequencies of interest are marked with red circles.

Figure 35 shows the pseudospectrum corresponding to the data used to construct the Prony polynomials in Figure 34 when $n = 2s + 1$. In the noiseless case, the pseudospectrum achieves exact super-resolution. Unfortunately, the locations of the local maxima are severely perturbed by even a very small amount of noise.

The subspace \mathcal{N} is the null space of the empirical covariance matrix

$$\Sigma(m) = \frac{1}{n-m+1} YY^* \quad (184)$$

$$= \frac{1}{n-m+1} \sum_{j=0}^{n-m} \begin{bmatrix} \vec{y}[j] \\ \vec{y}[j+1] \\ \vdots \\ \vec{y}[j+m-1] \end{bmatrix} \begin{bmatrix} \overline{\vec{y}[j]} & \overline{\vec{y}[j+1]} & \cdots & \overline{\vec{y}[j+m-1]} \end{bmatrix}. \quad (185)$$

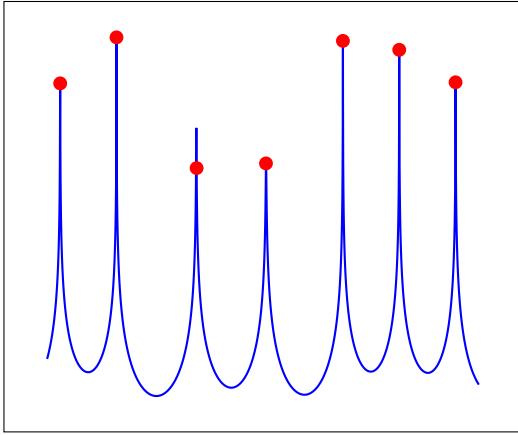
In order to obtain an estimate that is more robust to noise, we can average over more data. If we fix m and increase n , the column space of $\Sigma(m)$ remains the same, but the averaging process will cancel out the noise to some extent. This is the principle underlying the multiple-signal classification (MUSIC) [1, 4].

Algorithm 3.11 (Multiple-signal classification (MUSIC)). *The input is the number of frequencies s , the data y , which are assumed to be of the form (180) and the value of the parameter $m \geq s$.*

1. Build the empirical covariance matrix $\Sigma(m)$ defined in (184).
2. Compute the eigendecomposition of $\Sigma(m)$ to select the subspace \mathcal{N} corresponding to the $m-s$ smallest singular values.
3. Output an estimate of s estimated frequencies by locating the s highest peaks of the pseudospectrum

$$P_{\mathcal{N}}(u) = \log \frac{1}{|\mathcal{P}_{\mathcal{N}}(\vec{a}_{0:m-1})|^2}, \quad (186)$$

SNR = 40 dB



SNR = 1 dB

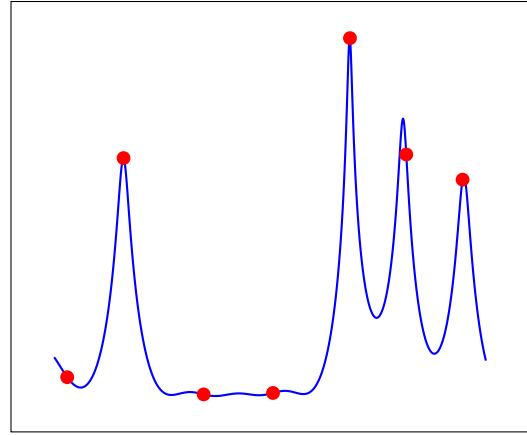


Figure 36: Pseudospectrum constructed by the MUSIC algorithm with $n = 81$ and $m = 30$ for the same signal used in Figure 34 and different noise levels. The true frequencies are marked with red circles.

By Lemma 3.10, in the absence of noise MUSIC achieves perfect super-resolution. When additive Gaussian noise is present in the data, MUSIC is much more robust than Prony's method. Figure 36 shows the result of applying MUSIC algorithm with $n = 81$ and $m = 30$ to the same data used in Figure 34 and different noise levels. The method is able to super-resolve the frequencies at a noise level of 40 dB. At 1 dB the pseudospectrum does not detect the smaller spikes (the true magnitudes are shown in Figure 34), but the estimate for the rest is still rather accurate.

In order to provide a theoretical justification of why MUSIC is stable we study the method in an asymptotic regime where the two following assumptions on the signal and the noise are met:

- **Assumption 1:** Consider the amplitude and phase of each coefficient in the multisinusoidal signal g :

$$\vec{c}[j] = \alpha_j \exp(i\phi_j), \quad 1 \leq j \leq s, \quad (187)$$

We model the phases ϕ_1, \dots, ϕ_s as independent random variables that are uniformly distributed in $[0, 2\pi]$. The amplitudes $\alpha_1, \dots, \alpha_s$ can be arbitrary and deterministic. This implies that $E(\vec{c}[j]) = 0$ and that the covariance matrix equals

$$E[\vec{c}\vec{c}^*] = S_{\vec{c}} := \begin{bmatrix} \alpha_1^2 & 0 & \cdots & 0 \\ 0 & \alpha_2^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \alpha_s^2 \end{bmatrix}. \quad (188)$$

- **Assumption 2:** The noise in the measurements (180) is Gaussian with zero mean and covariance matrix $\sigma^2 I$, which is also independent from the signal.

In summary, we model the data as an m -dimensional random vector of the form

$$\vec{y} = A_{0:m-1} \vec{c} + \vec{z}, \quad (189)$$

where \vec{z} is Gaussian with mean zero and covariance matrix $\sigma^2 I$. The following theorem, proved in Section 4.2 of the appendix, derives the covariance matrix of \vec{y} .

Theorem 3.12. *Let \vec{y} be an m dimensional vector of data satisfying Assumptions 1 and 2. The SVD of the covariance matrix of \vec{y} is equal to*

$$E[\vec{y}\vec{y}^*] = [U_S \ U_N] \begin{bmatrix} S + \sigma^2 I_s & 0 \\ 0 & \sigma^2 I_{n-s} \end{bmatrix} \begin{bmatrix} U_S^* \\ U_N^* \end{bmatrix}, \quad (190)$$

where S is a diagonal matrix containing the singular values of The singular vectors are divided into two unitary matrices.

- $U_S \in \mathbb{C}^{m \times s}$ contains an orthonormal basis of the signal subspace which corresponds to the span of $\vec{a}_{0:m-1}(f_1), \dots, \vec{a}_{0:m-1}(f_s)$.
- $U_N \in \mathbb{C}^{m \times (m-s)}$ is a unitary matrix spanning the noise subspace, which is the orthogonal complement of the signal subspace.

This theorem provides a rather intuitive interpretation of the MUSIC algorithm. The SVD of the covariance matrix of the data allows to estimate a signal subspace and a noise subspace. As a result, the term *subspace methods* is often used to describe MUSIC and related algorithms. Computing the pseudospectrum from these subspaces allows us to locate the support of the signal.

In practice, we approximate the covariance matrix using the empirical covariance matrix $\Sigma(m)$ defined in (184). Asymptotically, if we fix s and m and let $n \rightarrow \infty$, $\Sigma(m)$ converges to the true covariance matrix (see Section 4.9.1 in [5]). However that this does *not* necessarily imply that MUSIC will allow to find the support! To ensure that we can actually identify the noise subspace correctly, the singular values in the matrix S must all be large with respect to the variance of the noise σ^2 . In the case of signals that have a small separation between the frequencies (with respect to n), some of these singular values may be small due to the correlation between $\vec{a}_{0:m-1}(f_1), \dots, \vec{a}_{0:m-1}(f_s)$.

Figure 37 compares the performance of MUSIC for different noise levels and different values of the parameter m . On the left column, we see the decay of the singular values of the empirical covariance matrix. At high signal-to-noise ratios (SNR) there is a clear transition between the singular values corresponding to the signal subspace (in this case $s = 7$) and the noise subspace, but this is no longer necessarily the case when the noise is increased. On the right column, we see the performance of the algorithm for different values of the SNR and the parameter m . At relatively high SNRs MUSIC is an effective algorithm as long as the assumptions on the signal (random phases), noise (Gaussian) and measurement model (equispaced time samples) are satisfied. In Figure 38 we show the result of running the algorithm for the wrong value of the parameter s . If the value is not too different to s and the SNR not too low, the method is still capable of approximately locating the frequencies.

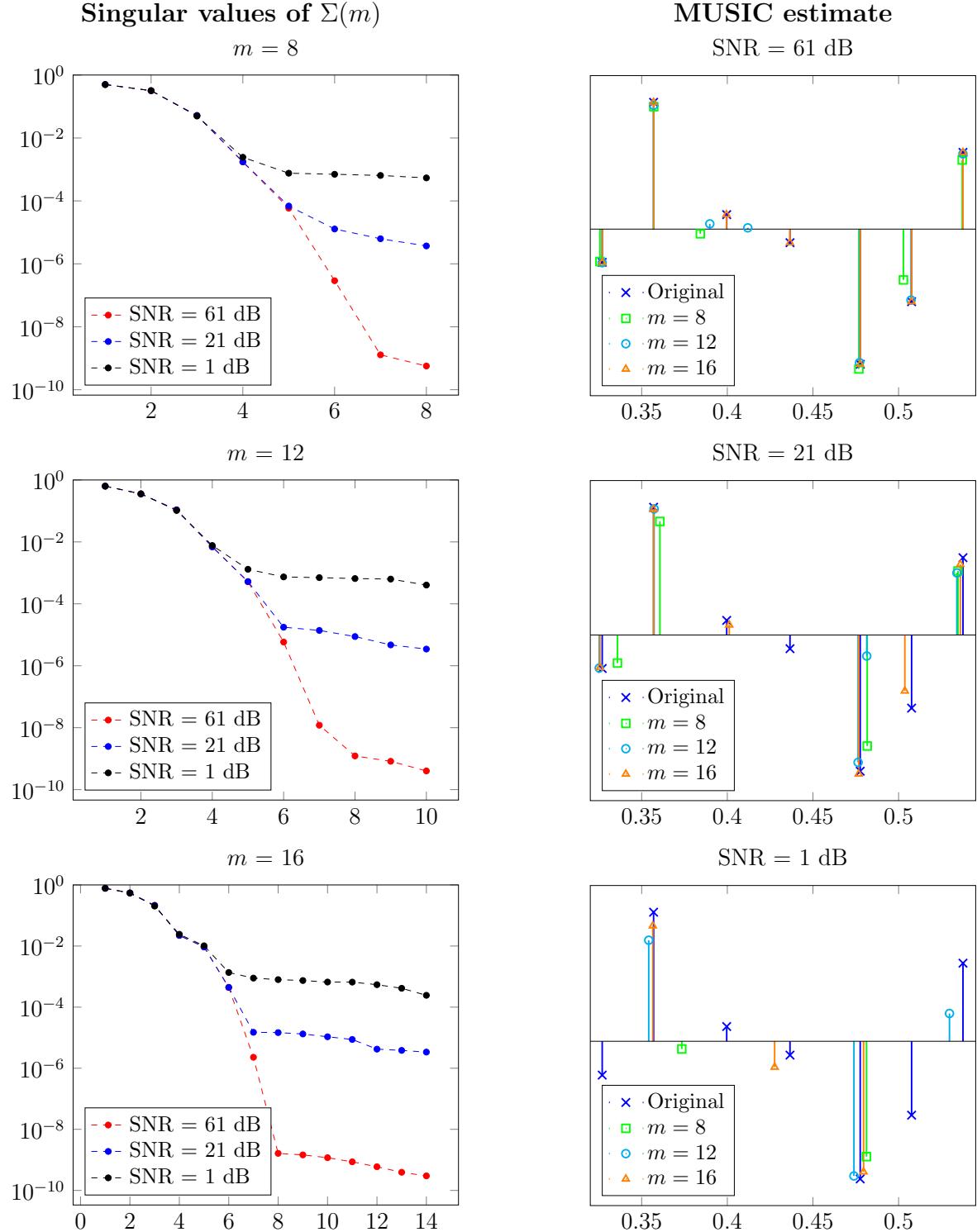


Figure 37: Singular values of the empirical covariance matrix $\Sigma(m)$ used by MUSIC (left) and corresponding estimates (right) for different values of the parameter m and of the SNR. The cardinality of the true support is $s = 7$.

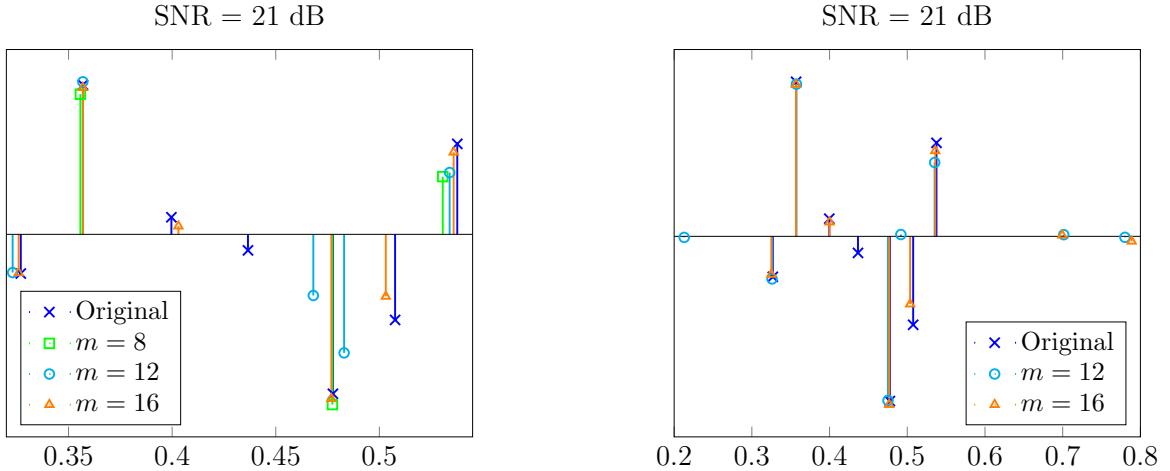


Figure 38: Line-spectra estimates obtained by Root MUSIC when the estimated number of sources is equal to $s - 1$ (left) and $s + 1$ (right) for the same data as in Figure 37.

4 Proofs

4.1 Proof of Lemma 3.9

Let us define

$$Z := \begin{bmatrix} 1 & 1 & \cdots & 1 \\ z_1 & z_2 & \cdots & z_s \\ z_1^2 & z_2^2 & \cdots & z_s^2 \\ \vdots \\ z_1^{s-1} & z_2^{s-1} & \cdots & z_s^{s-1} \end{bmatrix}. \quad (191)$$

The determinant of the first s rows of our matrix of interest is equal to

$$|Z| \prod_{1 \leq i \leq s} z_i = \prod_{1 \leq j < k \leq s} (z_j - z_k) \prod_{1 \leq i \leq s} z_i \neq 0 \quad (192)$$

This implies that the first s rows are linearly independent and consequently that the whole matrix is full rank.

4.2 Proof of Theorem 3.12

Due to the assumptions,

$$\mathbb{E}[\vec{\mathbf{y}}\vec{\mathbf{y}}^*] = \mathbb{E}[A_{0:m-1}\vec{\mathbf{c}}\vec{\mathbf{c}}^*A_{0:m-1}^* + A_{0:m-1}\vec{\mathbf{c}}\vec{\mathbf{z}}^* + \vec{\mathbf{z}}\vec{\mathbf{c}}^*A_{0:m-1}^* + \vec{\mathbf{z}}\vec{\mathbf{z}}^*] \quad (193)$$

$$= A_{0:m-1}\mathbb{E}[\vec{\mathbf{c}}\vec{\mathbf{c}}^*]A_{0:m-1}^* + A_{0:m-1}\mathbb{E}[\vec{\mathbf{c}}]\mathbb{E}[\vec{\mathbf{z}}^*] + \mathbb{E}[\vec{\mathbf{z}}]\mathbb{E}[\vec{\mathbf{c}}^*]A_{0:m-1}^* + \mathbb{E}[\vec{\mathbf{z}}\vec{\mathbf{z}}^*] \quad (194)$$

$$= A_{0:m-1}S_{\vec{\mathbf{c}}}A_{0:m-1}^* + \sigma^2 I. \quad (195)$$

The matrix $A_{0:m-1}S_{\vec{c}}A_{0:m-1}^*$ is symmetric and has rank s . It therefore has a singular value decomposition of the form

$$A_{0:m-1}S_{\vec{c}}A_{0:m-1}^* = \begin{bmatrix} U_{\mathcal{S}} & U_{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{\mathcal{S}}^* \\ U_{\mathcal{N}}^* \end{bmatrix}, \quad (196)$$

where $U_{\mathcal{S}}$ and $U_{\mathcal{N}}$ are as defined in the statement of the theorem.

To complete the proof, we decompose the identity matrix using $U_{\mathcal{S}}$ and $U_{\mathcal{N}}$ to obtain

$$\mathbb{E}[yy^*] = A_{0:m-1}S_{\vec{c}}A_{0:m-1}^* + \sigma^2 I \quad (197)$$

$$= \begin{bmatrix} U_{\mathcal{S}} & U_{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{\mathcal{S}}^* \\ U_{\mathcal{N}}^* \end{bmatrix} + \begin{bmatrix} U_{\mathcal{S}} & U_{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \sigma^2 I_s & 0 \\ 0 & \sigma^2 I_{n-s} \end{bmatrix} \begin{bmatrix} U_{\mathcal{S}}^* \\ U_{\mathcal{N}}^* \end{bmatrix}. \quad (198)$$

References

We recommend the [excellent notes](#) [3] for more background on frequency representations. For more information on spectral super-resolution using the periodogram and subspace methods we refer the reader to [5] and references therein.

- [1] G. Bienvenu. Influence of the spatial coherence of the background noise on high resolution passive methods. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages 306 – 309, 1979.
- [2] F. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the IEEE*, 66(1):51 – 83, 1978.
- [3] B. Osgood. The Fourier transform and its applications. *Lecture notes for EE261*, 2009.
- [4] R. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation*, 34(3):276 – 280, 1986.
- [5] P. Stoica and R. L. Moses. *Spectral Analysis of Signals*. Prentice Hall, 2005.

Lecture Notes 5: Multiresolution Analysis

1 Frames

A frame is a generalization of an orthonormal basis. The inner products between the vectors in a frame and an arbitrary vector preserve the inner-product norm of the vector.

Definition 1.1 (Frame). *Let \mathcal{V} be an inner-product space. A frame of \mathcal{V} is a set of vectors $\mathcal{F} := \{\vec{v}_1, \vec{v}_2, \dots\}$ such that for every $\vec{x} \in \mathcal{V}$*

$$c_L \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 \leq \sum_{\vec{v} \in \mathcal{F}} |\langle \vec{x}, \vec{v} \rangle|^2 \leq c_U \|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2, \quad (1)$$

for fixed positive constants $c_U \geq c_L \geq 0$. The frame is a tight frame if $c_L = c_U$.

A direct consequence of the definition is that frames span the ambient space.

Lemma 1.2 (Frames span the whole space). *Any frame $\mathcal{F} := \{\vec{v}_1, \vec{v}_2, \dots\}$ of a vector space \mathcal{V} spans \mathcal{V} .*

Proof. Assume that there exists a vector \vec{y} that does not belong to the span, then $\mathcal{P}_{\text{span}(\vec{v}_1, \vec{v}_2, \dots)^\perp} \vec{y}$ is nonzero and orthogonal to all the vectors in the frame and cannot satisfy (1). \square

Orthonormal bases are examples of frames. They are frames that contain a minimum number of vectors.

Lemma 1.3 (Orthonormal bases are tight frames). *Any orthonormal basis $\mathcal{B} := \{\vec{b}_1, \vec{b}_2, \dots\}$ of a vector space \mathcal{V} is a tight frame.*

Proof. For any vector $\vec{x} \in \mathcal{V}$, by the Pythagorean theorem

$$\|\vec{x}\|_{\langle \cdot, \cdot \rangle}^2 = \left\| \sum_{\vec{b} \in \mathcal{B}} \langle \vec{x}, \vec{b} \rangle \vec{b} \right\|_{\langle \cdot, \cdot \rangle}^2 \quad (2)$$

$$= \sum_{\vec{b} \in \mathcal{B}} \left| \langle \vec{x}, \vec{b} \rangle \right|^2 \left\| \vec{b} \right\|_{\langle \cdot, \cdot \rangle}^2 \quad (3)$$

$$= \sum_{\vec{b} \in \mathcal{B}} \left| \langle \vec{x}, \vec{b} \rangle \right|^2. \quad (4)$$

\square

The operator that maps vectors to their frame coefficients is called the analysis operator of the frame.

Definition 1.4 (Analysis operator). *The analysis operator Φ of a frame maps a vector to its coefficients in the frame representation*

$$\Phi(\vec{x})[k] = \langle \vec{x}, \vec{v}_k \rangle. \quad (5)$$

For any finite frame $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_m\}$ of \mathbb{C}^n the analysis operator corresponds to the matrix

$$F := \begin{bmatrix} \vec{v}_1^* \\ \vec{v}_2^* \\ \vdots \\ \vec{v}_m^* \end{bmatrix}. \quad (6)$$

In finite-dimensional spaces, any full rank square or tall matrix can be interpreted as the analysis operator of a frame.

Lemma 1.5 (Frames in finite-dimensional spaces). *A set of vectors $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_m\}$ form a frame of \mathbb{C}^n if and only the matrix F defined by equation (6) is full rank. In that case,*

$$c_U = \sigma_1^2, \quad (7)$$

$$c_L = \sigma_n^2, \quad (8)$$

where σ_1 is the largest singular value of F and σ_n is the smallest.

Proof. By Theorem 2.7 in Lecture Notes 2, for any vector $\vec{x} \in \mathbb{C}^n$

$$\sigma_n^2 \leq \|F\vec{x}\|_2^2 = \sum_{j=1}^m \langle \vec{x}, \vec{v}_j \rangle^2 \leq \sigma_1^2. \quad (9)$$

□

To recover a vector from its frame coefficients, we need to invert the action of the analysis operator. In finite dimensions this can be achieved using the pseudoinverse of the corresponding matrix.

Lemma 1.6 (Pseudoinverse). *If an $n \times m$ tall matrix A , $m \geq n$, is full rank, then its pseudoinverse*

$$A^\dagger := (A^* A)^{-1} A^* \quad (10)$$

is well defined, is a left inverse of A

$$A^\dagger A = I \quad (11)$$

and equals

$$A^\dagger = V S^{-1} U^*, \quad (12)$$

where $A = USV^*$ is the SVD of A .

Proof.

$$A^\dagger := (A^* A)^{-1} A^* \quad (13)$$

$$= (V S U^* U S V^* A)^{-1} V S U^* \quad (14)$$

$$= (V S^2 V^*)^{-1} V S U^* \quad (15)$$

$$= V S^{-2} V^* V S U^* \quad (16)$$

$$= V S^{-1} U, \quad (17)$$

where S^{-2} and S^{-1} are diagonal matrices containing σ_j^{-2} and σ_j^{-1} in the j th entry of the diagonal, where σ_j denotes the j th singular value of A . These matrices are well defined as long as all the singular values are nonzero, or equivalently A is full rank. In that case,

$$A^\dagger A = V S^{-1} U V^* U S V^* \quad (18)$$

$$= I. \quad (19)$$

□

Corollary 1.7 (Recovering the signal). *Let \vec{c} be the representation of a vector \vec{x} in terms of a frame $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_m\}$ of \mathbb{C}^n . Then applying the pseudoinverse of F recovers the signal from the coefficients*

$$\vec{x} = F^\dagger \vec{c}. \quad (20)$$

2 Short-time Fourier transform

The motivation to consider frames instead of bases is that they often make it possible to build signal decompositions that are more flexible. An important example is the short-time Fourier transform (STFT). Frequency representations such as the Fourier series and the DFT provide global information about the fluctuations of a signal, but they do not capture *local* information. However, the spectrum of speech, music and other sound signals changes continuously with time. The STFT is designed to describe these localized fluctuations. It consists of computing the frequency representation of a time segment of the signal, extracted through multiplication with a window.

Definition 2.1 (Short-time Fourier transform). *The short-time Fourier transform (STFT) of a function $f \in \mathcal{L}_2[-1/2, 1/2]$ is defined as*

$$\text{STFT}\{f\}(k, \tau) := \int_{-1/2}^{1/2} f(t) \overline{w(t - \tau)} e^{-i2\pi kt} dt, \quad (21)$$

where $w \in \mathcal{L}_2[-1/2, 1/2]$ is a window function. In words, it is equal to the Fourier series coefficients of the pointwise product between f and a shifted window $w_{[l]}$.

The STFT coefficients are equal to the inner product between the signal and vectors of the form $v_{k,\tau}(t) := w(t - \tau) e^{i2\pi kt}$, which corresponds to the window function w shifted by τ in time and by

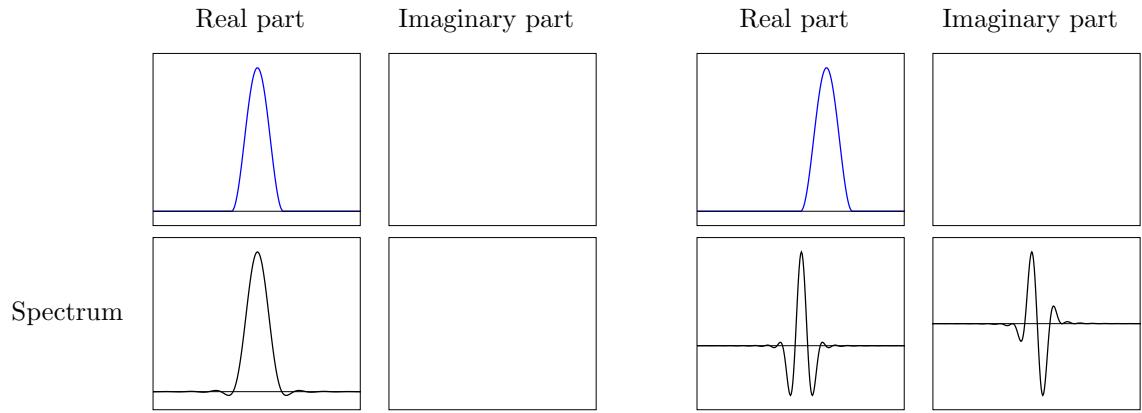
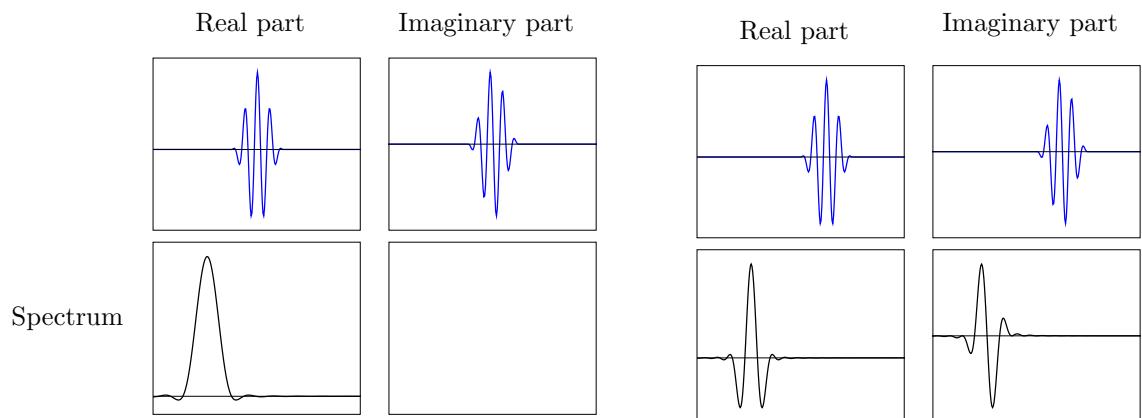
$\tau = 0, k = 0$ $\tau = 1/32, k = 0$  $\tau = 0, k = 64$ $\tau = 1/32, k = 64$ 

Figure 1: Examples of STFT frame vectors along with their Fourier representation.

k in frequency. As long as the shifts are chosen so that the windows overlap, the STFT coefficients form a frame. Figure 1 shows some of these frame vectors.

The discrete version of the short-time Fourier transform acts upon finite-dimensional vectors and is usually also known as STFT.

Definition 2.2 (Discrete short-time Fourier transform). *The STFT of a vector $\vec{x} \in \mathbb{C}^n$ is defined as*

$$\text{STFT } \{f\}(k, l) := \left\langle \vec{x} \circ \vec{w}_{[l]}, \vec{h}_k \right\rangle, \quad (22)$$

where \vec{w} is a window vector and $\vec{h}_k^{[n]}$ is the discrete complex sinusoidal vector from Definition 1.5 in Lecture Notes 4.

As in the continuous case, if the shifts overlap sufficiently, then this transformation is a frame in a finite-dimensional space. This means that there is a tall matrix that represents the analysis operator, and that we can invert it with the pseudoinverse by Lemma 1.6. However this would be very inefficient computationally! The STFT operator is usually applied and inverted using fast algorithms based on the FFT.

The simplest window function that we can use is a rectangular function, i.e. just selecting intervals of coefficients. Unfortunately, this introduces an artificial discontinuity at the ends of the interval. Mathematically, multiplying the coefficients by the rectangular function is equivalent to convolving with a Dirichlet kernel in the spectral domain, which becomes apparent when we compute the Fourier coefficients of the windowed data, as shown in Figure 2. In contrast, Gaussian-like windows that taper off at the ends smooth the borders of the windowed signal and avoid the high-frequency artifacts introduced by the side lobes of the Dirichlet kernel.

The STFT is an important tool for sound processing. Variations in the spectral components of signals are visualized using the *spectrogram*, which is equal to the logarithm of the magnitude of the STFT coefficients. Figure 4 shows the spectrogram of a real speech signal. The time and frequency representation of the same signal are shown in Figure 3. In contrast to these representations, the spectrogram reveals how the frequency components of the speech signal vary over time. The resolution at which we track these variations depends on the width of the window chosen to compute the spectrogram. Shorter windows provide higher temporal resolution (we can track quicker changes), but are not able to detect lower-frequency components whose periods are longer than the chosen window. This motivates using windows of different lengths to extract information at multiple resolutions. The next section discusses signal representations designed to achieve this.

3 Wavelets

3.1 Definition

Wavelets are designed to capture signal structure at different scales. This is achieved with an analysis operator that contains scaled copies of a fixed function called a wavelet.

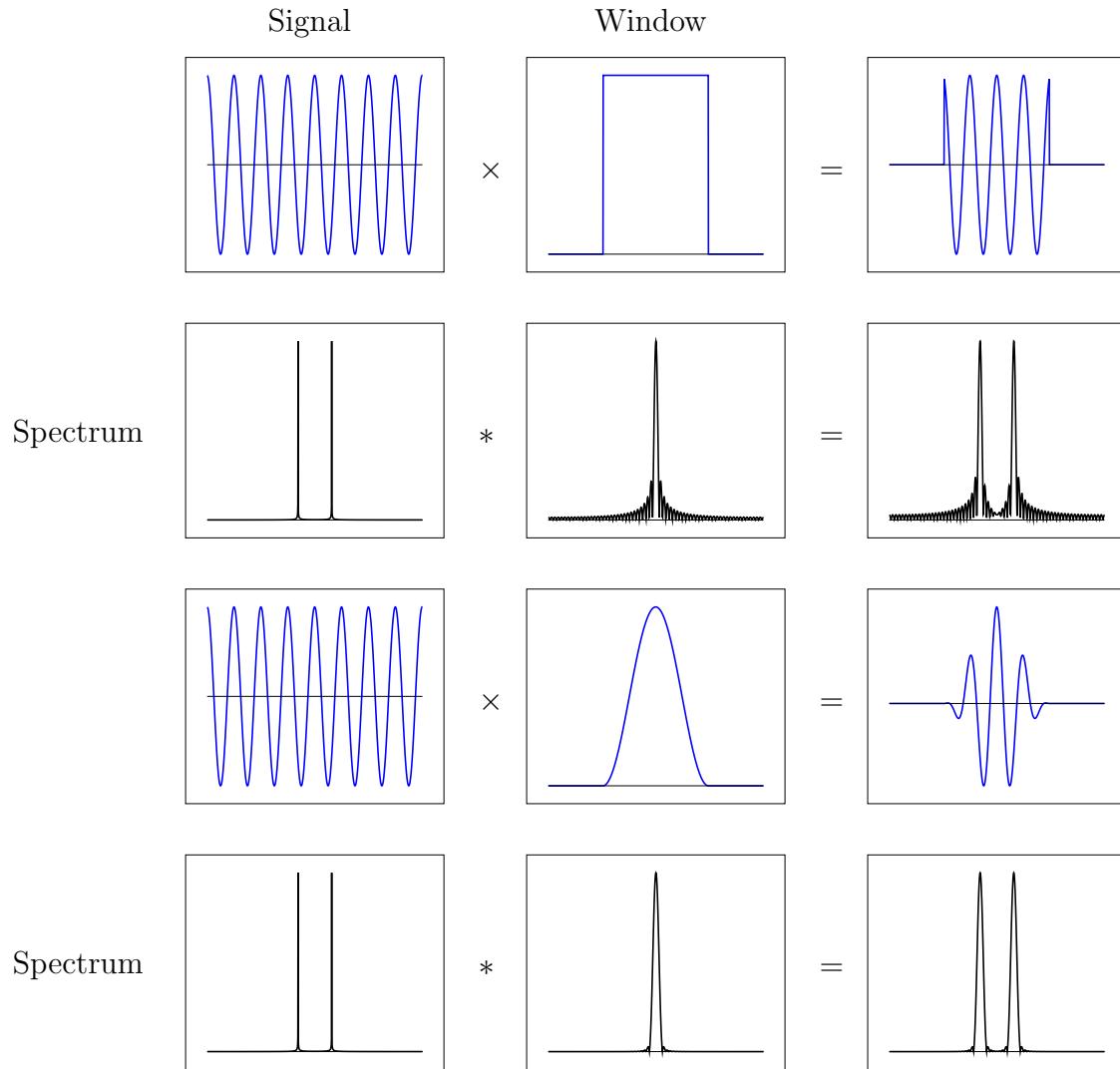


Figure 2: The spectrum of a time segment may contain spurious high-frequency content produced by the sudden transition at the ends of the segment. In the frequency domain, the spectrum is being convolved by a sinc function, which has a very heavy tail. Multiplying the signal by a localized window that has a faster decay in the frequency domain alleviates the problem.

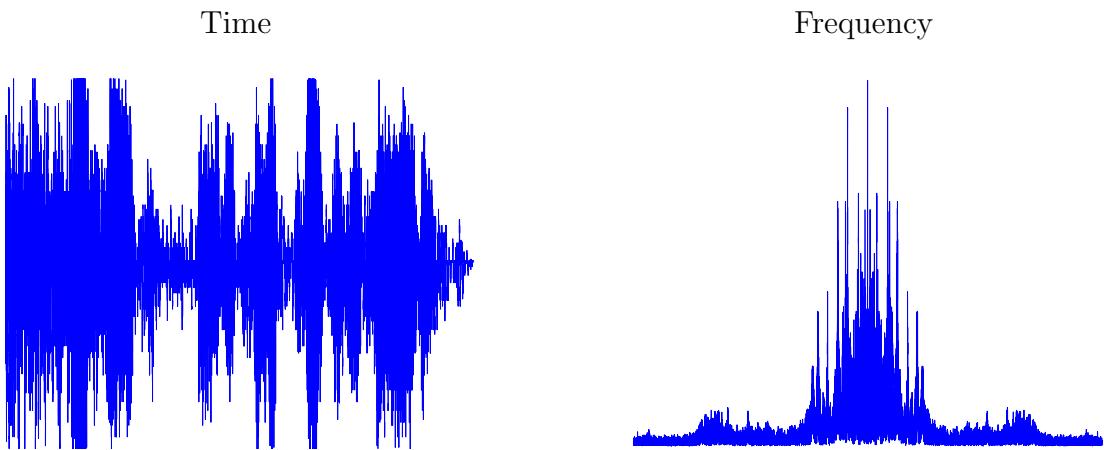


Figure 3: Time and frequency representation of a speech signal.

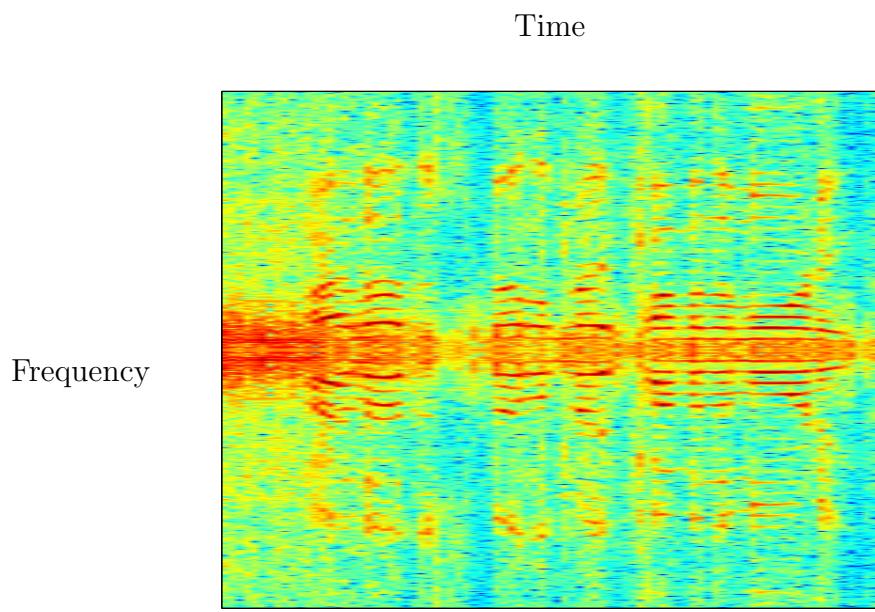


Figure 4: Spectrogram (log magnitude of STFT coefficients) of the speech signal in Figure 3.

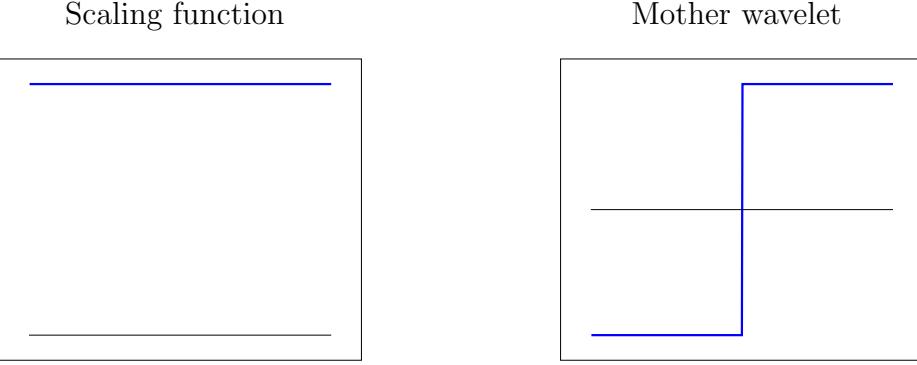


Figure 5: Scaling and wavelet function of the Haar wavelet transform.

Definition 3.1 (Wavelet transform). *The wavelet transform of a function $f \in \mathcal{L}_2$ depends on a choice of wavelet (or mother wavelet) $\psi \in \mathcal{L}_2$ and scaling function $\phi \in \mathcal{L}_2$ (or father wavelet). The scaling coefficients are obtained through an inner product with shifted copies of ϕ*

$$W_\phi \{f\}(\tau) := \frac{1}{\sqrt{s}} \int f(t) \overline{\phi(t-\tau)} dt \quad (23)$$

whereas the wavelet coefficients are obtained through an inner product with dilated, shifted copies of ψ

$$W_\psi \{f\}(s, \tau) := \frac{1}{\sqrt{s}} \int f(t) \overline{\psi\left(\frac{t-\tau}{s}\right)} dt. \quad (24)$$

Intuitively, $W\{f\}(s, \tau)$ captures the information at scale s and location τ . The scaling function can be interpreted as a low-pass filter that extracts the global features that are not captured by the wavelet coefficients. The Haar wavelet is an example of a wavelet. Figure 5 shows its scaling and wavelet functions.

Definition 3.2 (Haar wavelet). *The scaling function for the Haar wavelet is a rectangular function*

$$\phi(t) := 1, \quad -\frac{1}{2} \leq t \leq \frac{1}{2}. \quad (25)$$

The Haar wavelet is of the form

$$\psi(t) := \begin{cases} -1, & -\frac{1}{2} \leq t \leq 0, \\ 1, & 0 \leq t \leq \frac{1}{2}. \end{cases} \quad (26)$$

The discrete wavelet transform acts upon finite-dimensional vectors.

Definition 3.3 (Discrete wavelet transform). *The wavelet transform of a function $f \in \mathbb{C}^n$ depends on a choice of wavelet (or mother wavelet) $\vec{\psi} \in \mathbb{C}^n$ and scaling vector $\vec{\phi} \in \mathbb{C}^n$ (or father wavelet). The scaling coefficients are obtained through an inner product with shifted copies of $\vec{\phi}$*

$$W_{\vec{\phi}} \{f\}(l) := \langle \vec{x}, \vec{\phi}_{[l]} \rangle, \quad (27)$$

whereas the wavelet coefficients are obtained through an inner product with scaled, shifted copies of $\vec{\psi}$

$$W_{\vec{\psi}}\{f\}(s, l) := \left\langle \vec{x}, \vec{\psi}_{[s,l]} \right\rangle, \quad (28)$$

where

$$\vec{\psi}_{[s,l]}[j] := \vec{\psi} \left[\frac{j-l}{s} \right]. \quad (29)$$

The discrete Haar wavelet is the discrete counterpart of the Haar wavelet.

Definition 3.4 (Discrete Haar wavelet). *The scaling function for the Haar wavelet is a rectangular function*

$$\vec{\phi}[j] := 1, \quad 1 \leq j \leq n. \quad (30)$$

The Haar wavelet is of the form

$$\vec{\psi}[j] := \begin{cases} -1, & j = \frac{n}{2}, \\ 1, & j = \frac{n}{2} + 1, \\ 0, & \text{otherwise.} \end{cases} \quad (31)$$

3.2 Multiresolution decomposition

The discrete Haar wavelet and its corresponding scaling vector can be used to construct a basis of \mathbb{C}^n .

Definition 3.5 (Haar wavelet basis). *Let $n := 2^K$ for some integer K . We fix a single scaling vector*

$$\vec{\phi}[j] := \frac{1}{\sqrt{n}}, \quad 1 \leq j \leq n. \quad (32)$$

We fix $K+1$ scales equal to $2^0, 2^1, \dots, 2^K$. The wavelets at the finest scale 2^0 are given by

$$\vec{\psi}[j] := \begin{cases} -\frac{1}{\sqrt{2}}, & j = 1, \\ \frac{1}{\sqrt{2}}, & j = 2, \\ 0, & j > 2, \end{cases} \quad (33)$$

and copies of $\vec{\psi}$ shifted by 2, so that they do not overlap. The wavelets at scale 2^k , $1 \leq k \leq K$ are copies of $\vec{\psi}$ dilated by 2^k , multiplied by a factor of $1/\sqrt{2^k}$ (their ℓ_2 norm equals one) and shifted by multiples of 2^{k+1} (so the basis vectors at each scale do not overlap).

The Haar wavelet basis contains n unit-norm vector that are all orthogonal, so they form an orthonormal basis of \mathbb{C}^n . Figure 6 shows the basis vectors for $n = 8$. Figure 7 shows the coefficients of an electrocardiogram signal in the basis.

Wavelet bases can be interpreted as a multiresolution representation, where the coefficients corresponding to different dilations of the wavelet capture information at the corresponding scales. This is made more precise in the following definition.

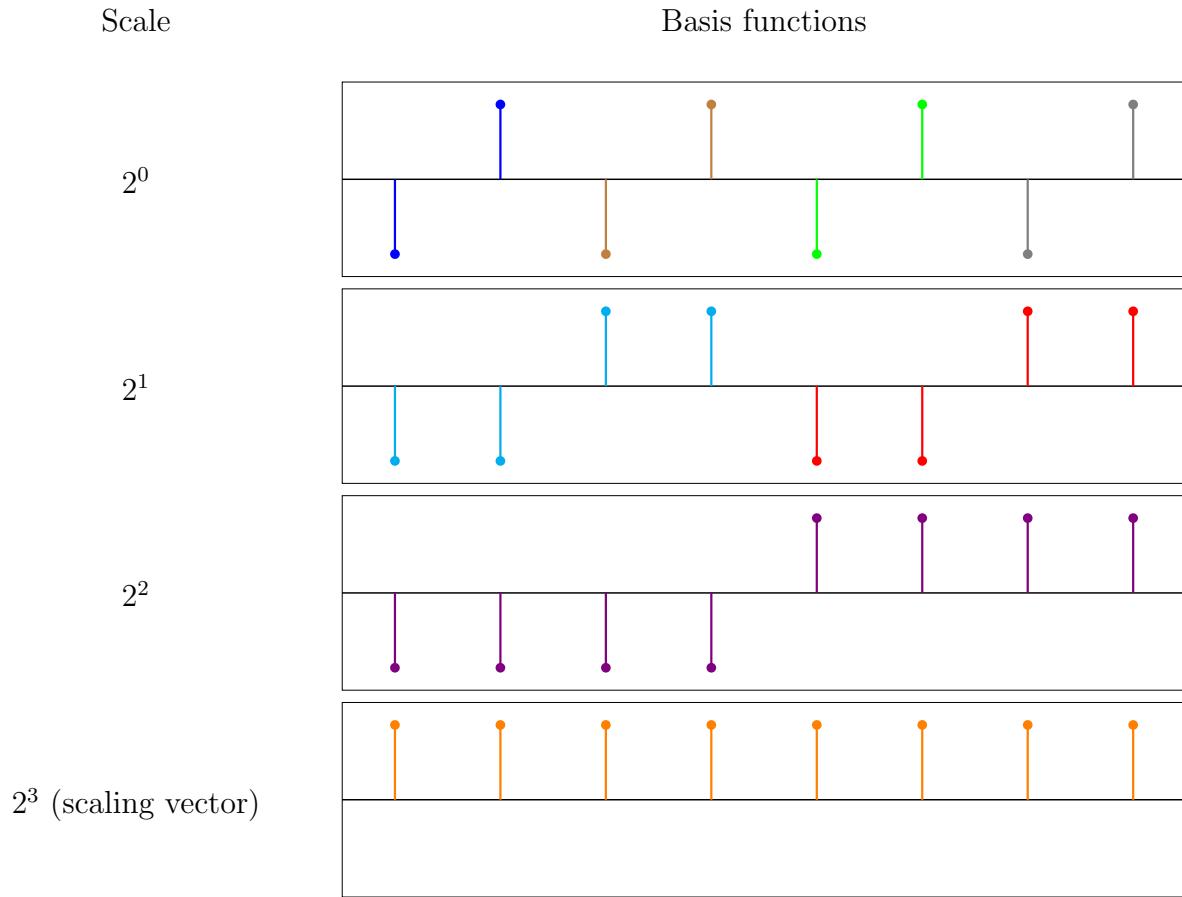


Figure 6: Basis functions in the Haar wavelet basis for \mathbb{C}^8 .

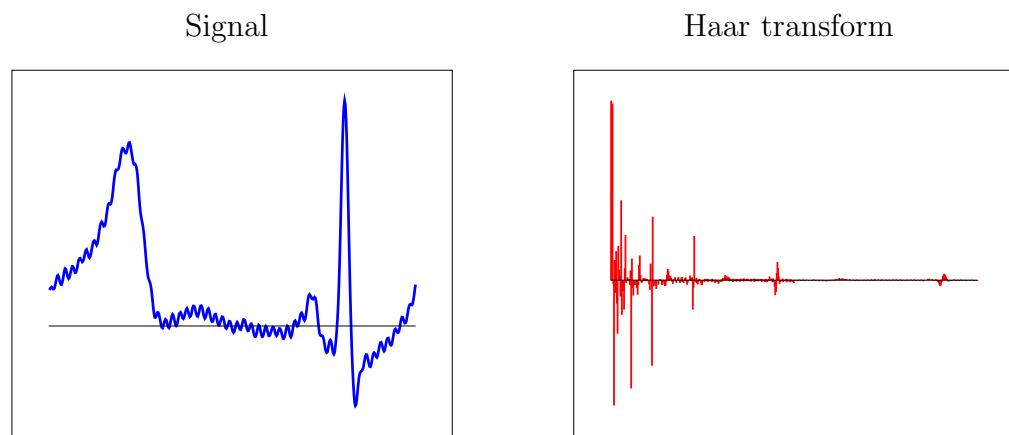


Figure 7: Electrocardiogram signal (left) and its corresponding Haar wavelet coefficients (right).

Definition 3.6 (Multiresolution wavelet decomposition). *Let $n := 2^K$ for some integer K . Given a scaling vector $\vec{\phi} \in \mathbb{C}^n$ and a wavelet $\vec{\psi} \in \mathbb{C}^n$, a multiresolution decomposition of \mathbb{C}^n is a sequence of subspaces $\mathcal{V}_0, \mathcal{V}_1, \dots, \mathcal{V}_K$ where:*

- \mathcal{V}_K is spanned by the scaling vector $\vec{\phi}$.
- $\mathcal{V}_k := \mathcal{W}_k \oplus \mathcal{V}_{k+1}$ where \mathcal{W}_k is the span of ψ dilated by 2^k and shifted by multiples of 2^{k+1} .

For any vector $\vec{x} \in \mathbb{C}^n$, $\mathcal{P}_{\mathcal{V}_k} \vec{x}$ is the approximation of \vec{x} at scale 2^k .

To be a valid multiresolution decomposition, the subspaces must satisfy the following properties:

- $\mathcal{V}_0 = \mathbb{C}^n$, the approximation at scale 2^0 is perfect.
- \mathcal{V}_k is invariant to translations of scale 2^k for $0 \leq k \leq K$. If $\vec{x} \in \mathcal{V}_k$ then

$$\vec{x}_{[2^k l]} \in \mathcal{V}_k \quad \text{for all } l \in \mathbb{Z}, \quad (34)$$

where the shifts are circular.

- Dilating vectors in \mathcal{V}_j by 2 yields vectors in \mathcal{V}_{j+1} . Let $\vec{x} \in \mathcal{V}_j$ be nonzero only between 1 and $n/2$, the dilated vector \vec{y} defined by

$$\vec{y}[j] = \vec{x}[\lceil j/2 \rceil] \quad (35)$$

belongs to \mathcal{V}_{j+1} .

By construction, the Haar wavelet basis in Definition 3.5 provides a multiresolution decomposition of \mathbb{C}^n . In Figure 8 the decomposition is applied to obtain approximations of an electrocardiogram signal at different scales. Many other wavelet bases apart from the Haar yield multiresolution decompositions: Meyer, Daubechies, Battle-Lemarie, ... We refer the interested reader to Chapter 7 in [?] for a detailed and rigorous description of the construction of orthonormal wavelet bases.

3.3 Multidimensional wavelet decompositions

Two-dimensional wavelets can be obtained by taking outer products of one-dimensional wavelets, as in the case of the two-dimensional discrete Fourier transform. 2D wavelets are of the form,

$$\xi_{[s_1, s_2, k_1, k_2]}^{2D} := \xi_{[s_1, k_1]}^{1D} (\xi_{[s_2, k_2]}^{1D})^*, \quad (36)$$

where ξ can refer to both 1D scaling and wavelet functions. We consider shifts k_1, k_2 in two dimensions and a two-dimensional scaling s_1, s_2 . The corresponding two-dimensional transform allows to obtain multiscale representations of images. An example is shown in Figure 10. The coefficients are grouped by their scale (which is decreasing as we move down and to the right) and arranged in two dimensions, according to the location of the corresponding shifted wavelet with respect to the image. Figure 9 shows the vectors in a 2D Haar wavelet basis.

Designing multidimensional transforms that are more effective at providing sparse representations for images has been a vibrant research subject for many years. Some of these extensions include the steerable pyramid, ridgelets, curvelets, and bandlets. We refer to Section 9.3 in [?] for more details.

Scale (2^k)

$\mathcal{P}_{\mathcal{W}_k} \vec{x}$

$\mathcal{P}_{\mathcal{V}_k} \vec{x}$

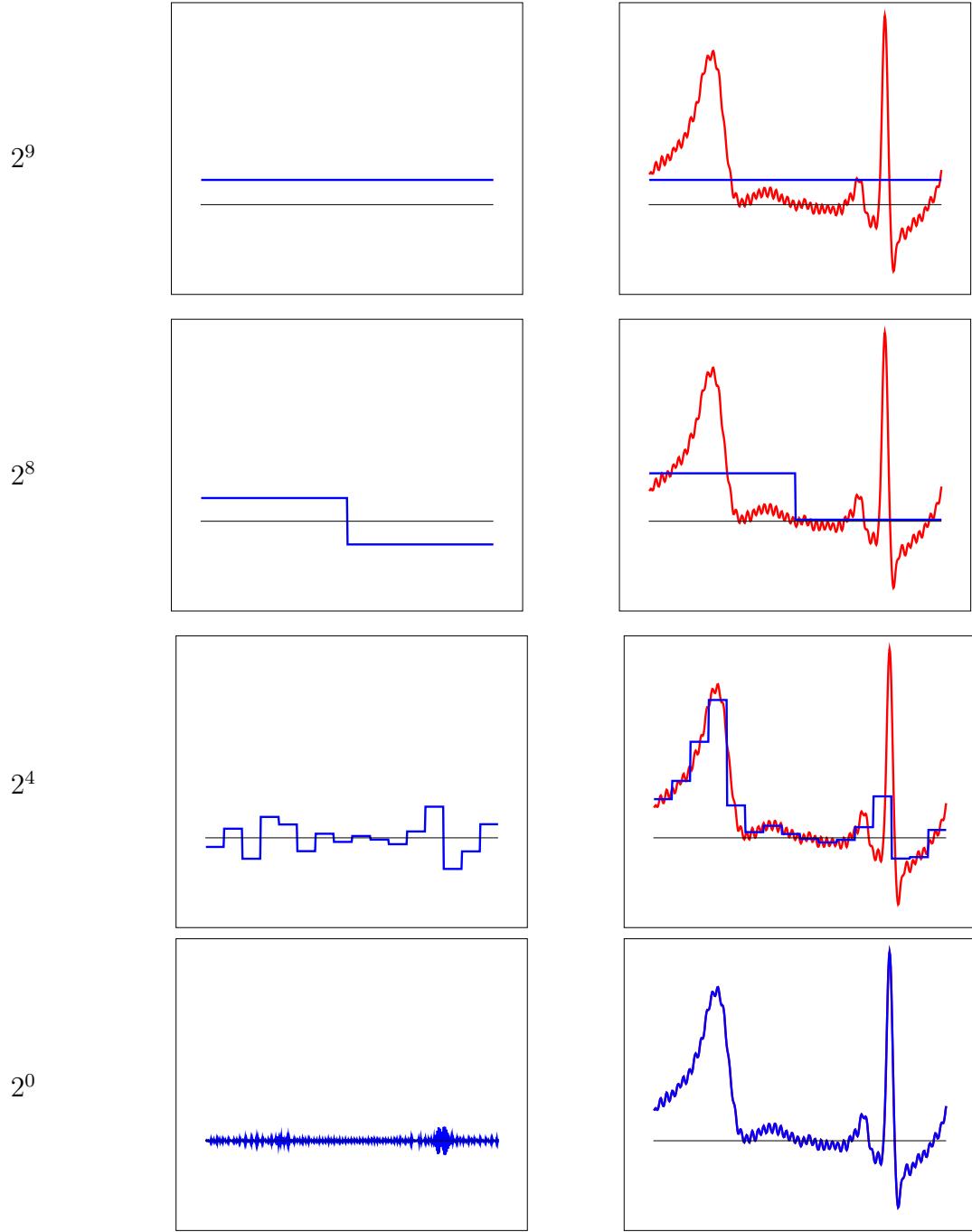


Figure 8: Multiresolution decomposition of the electrocardiogram signal in Figure 7. On the left, the projection of the signal onto \mathcal{W}_k extracts information at scale 2^k . On the right, projection onto \mathcal{V}_k yields an approximation of the signal at scale 2^k .

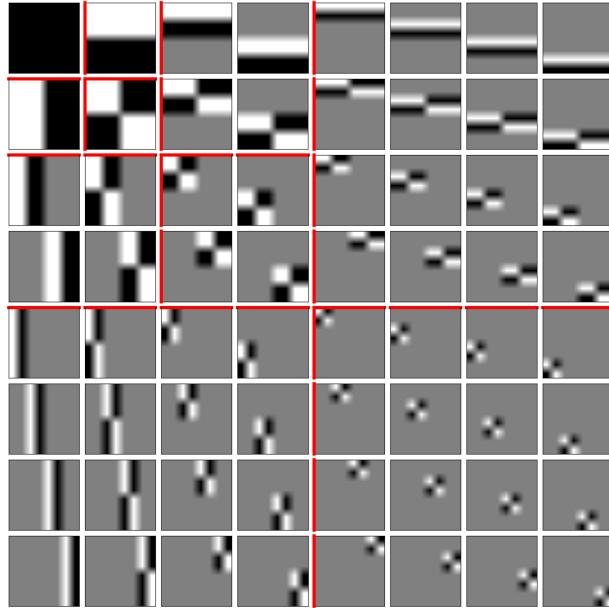


Figure 9: Basis vectors of the 2D Haar wavelet transform.

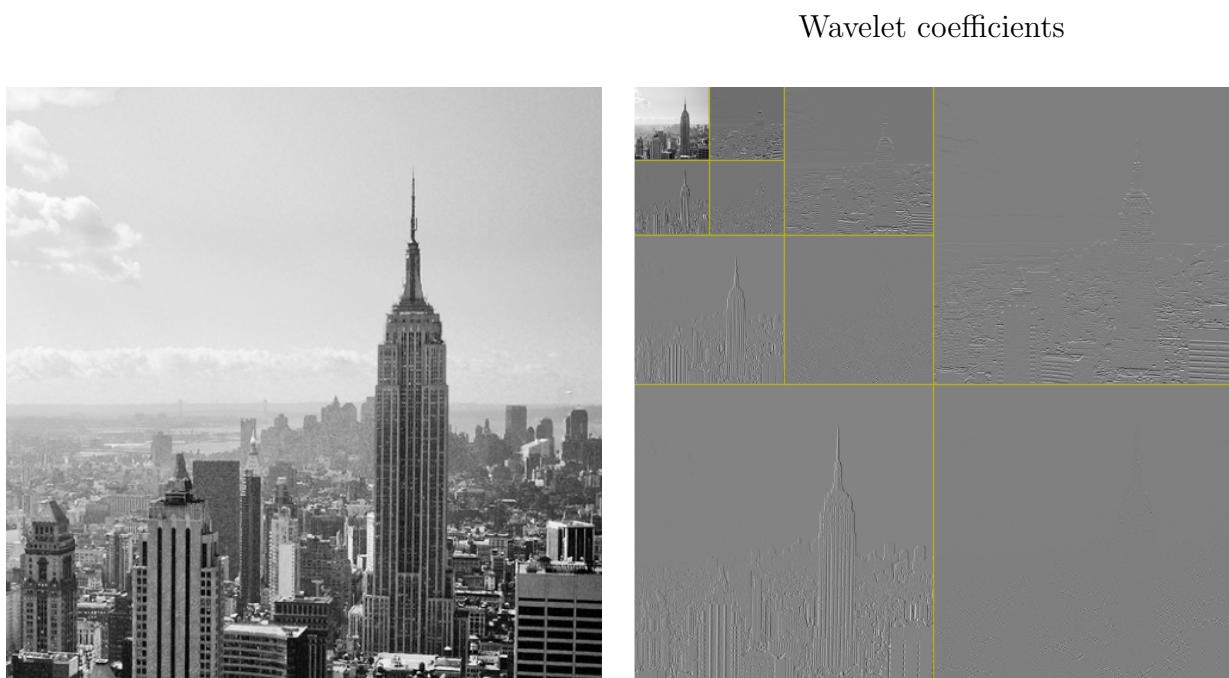
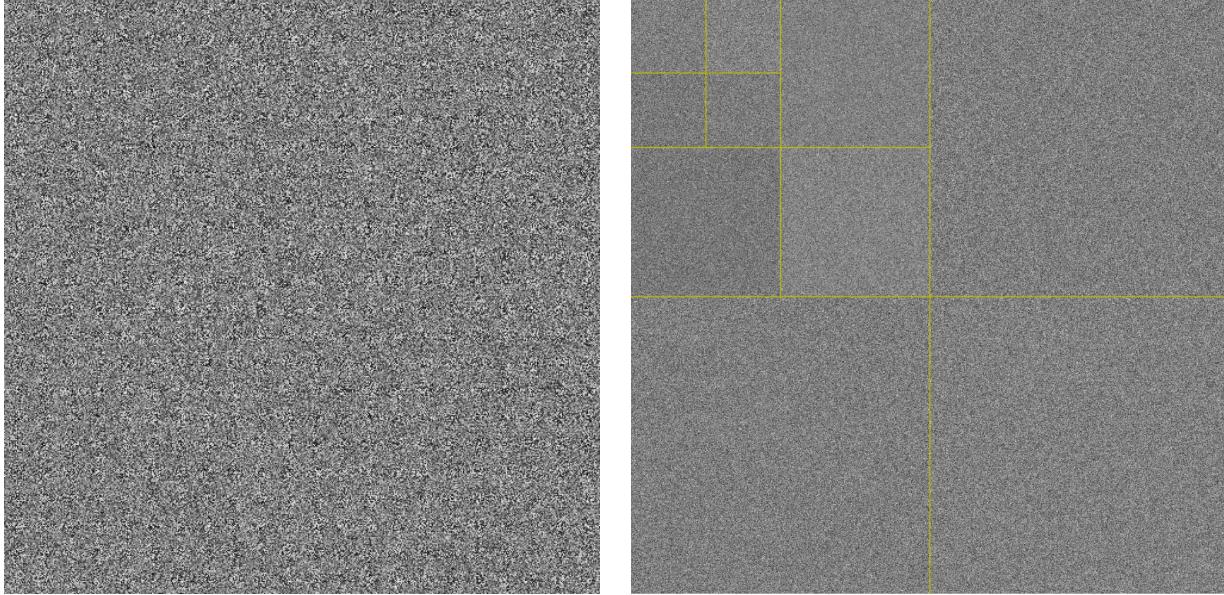


Figure 10: An image (left) and its coefficients in a 2D Haar wavelet basis (right). The coefficients are arranged so that the scaling coefficients are on the top left and coefficients corresponding to increasingly fine scales are situated below and to the right.

Wavelet coefficients

**Figure 11:** Gaussian iid noise (left) and its Haar wavelet coefficients (left).

4 Denoising via thresholding

The STFT and wavelet transforms often yield **sparse** signal representations, meaning that many coefficients are equal to zero. In the case of the STFT, this occurs when only a few spectral components are active at a particular time, which is typical of speech or music signals (see Figure 4). In the case of wavelets, sparsity results from the fact that large regions of natural images (and many other signals) are smooth and mostly contain coarse-scale features, whereas most of the fine-scale features are confined to edges or regions with high-frequency textures.

In contrast, noisy perturbations usually have dense coefficients in any fixed frame or basis. As we establish in Lecture Notes 3, if \vec{z} is a Gaussian random vector with covariance matrix $\sigma^2 I$, for some fixed $\sigma^2 > 0$ then $F\vec{z}$ is a Gaussian random vector with covariance matrix FF^* . In particular, if F is a basis, then $F\vec{z}$ is iid Gaussian, which means that the magnitude of most entries is approximately equal to the standard deviation σ . Figure 11 shows the Haar wavelet coefficients of iid Gaussian noise. As expected, the coefficients are also noisy and dense in this representation.

Let us consider the problem of denoising measurements $\vec{y} \in \mathbb{C}^n$ of a signal $\vec{x} \in \mathbb{C}^n$ corrupted by additive noise $\vec{z} \in \mathbb{C}^n$

$$\vec{y} := \vec{x} + \vec{z}. \quad (37)$$

Under the assumptions that (1) $F\vec{x}$ is sparse representation where F is a certain frame or basis and (2) the entries of $F\vec{z}$ are small and dense, thresholding $F\vec{y}$ makes it possible to suppress the noise while preserving the signal. Figure 12 shows an example of two noisy images with different signal-to-noise ratios (SNR), defined as the ratio between the ℓ_2 norm of the signal and the noise. In the wavelet domain, the coefficients corresponding to the signal lie above a *sea* of noisy coefficients.

SNR

2.5



1



Wavelet coefficients

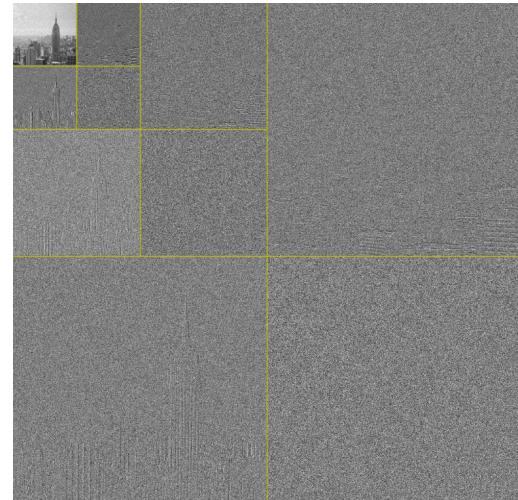
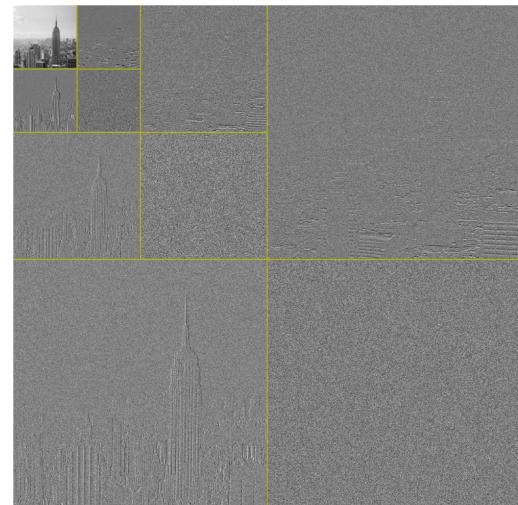


Figure 12: The two noisy images on the left are obtained by adding Gaussian noise to the image in Figure 10 to obtain an SNR of 2.5 (above) and 1 (below). The coefficients of the images in the 2D Haar basis are shown on the right.

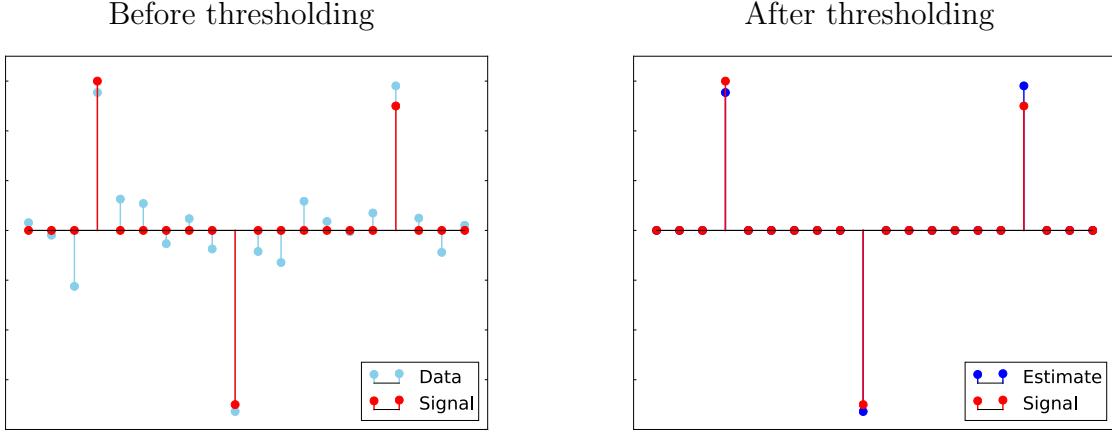


Figure 13: Denoising via hard thresholding.

To motivate threshold-based denoising, consider the case where \vec{x} itself is sparse and \vec{z} . In that case we can denoise by setting to zero the entries in \vec{y} that are below a certain value. Figure 13 illustrates this with a simple example. Most signals are not sparse, but in many cases we can design a linear transform that sparsifies them. We can then apply the same idea to the coefficients of the measurements in this representation.

Algorithm 4.1 (Denoising via hard thresholding). *Let \vec{y} follow the model in equation (37). To estimate the signal we:*

1. *Compute a decomposition $F\vec{y}$, where F is a frame or basis which sparsifies the signal \vec{x} .*
2. *Apply the hard-thresholding operator $\mathcal{H}_\eta : \mathbb{C}^n \rightarrow \mathbb{C}^n$ to $F\vec{y}$*

$$\mathcal{H}_\eta(\vec{v})[j] := \begin{cases} \vec{v}[j] & \text{if } |\vec{v}[j]| > \eta, \\ 0 & \text{otherwise,} \end{cases} \quad (38)$$

for $1 \leq j \leq n$, where η is adjusted according to the standard deviation of $F\vec{z}$. If F is a basis and \vec{z} is iid Gaussian with standard deviation σ , η should be set larger than σ .

3. *Compute the estimate by inverting the transform. If F is a basis, then*

$$\vec{x}_{\text{est}} := F^{-1}\mathcal{H}_\eta(F\vec{y}). \quad (39)$$

If F is a frame,

$$\vec{x}_{\text{est}} := F^\dagger \mathcal{H}_\eta(F\vec{y}), \quad (40)$$

where F^\dagger is the pseudoinverse of F (any other left inverse of F would also work).

Figure 14 shows the result of denoising a multisinusoidal signal by thresholding its Fourier coefficients. Figure 15 shows the result of denoising the images in Figure 12 by thresholding their 2D wavelet coefficients.

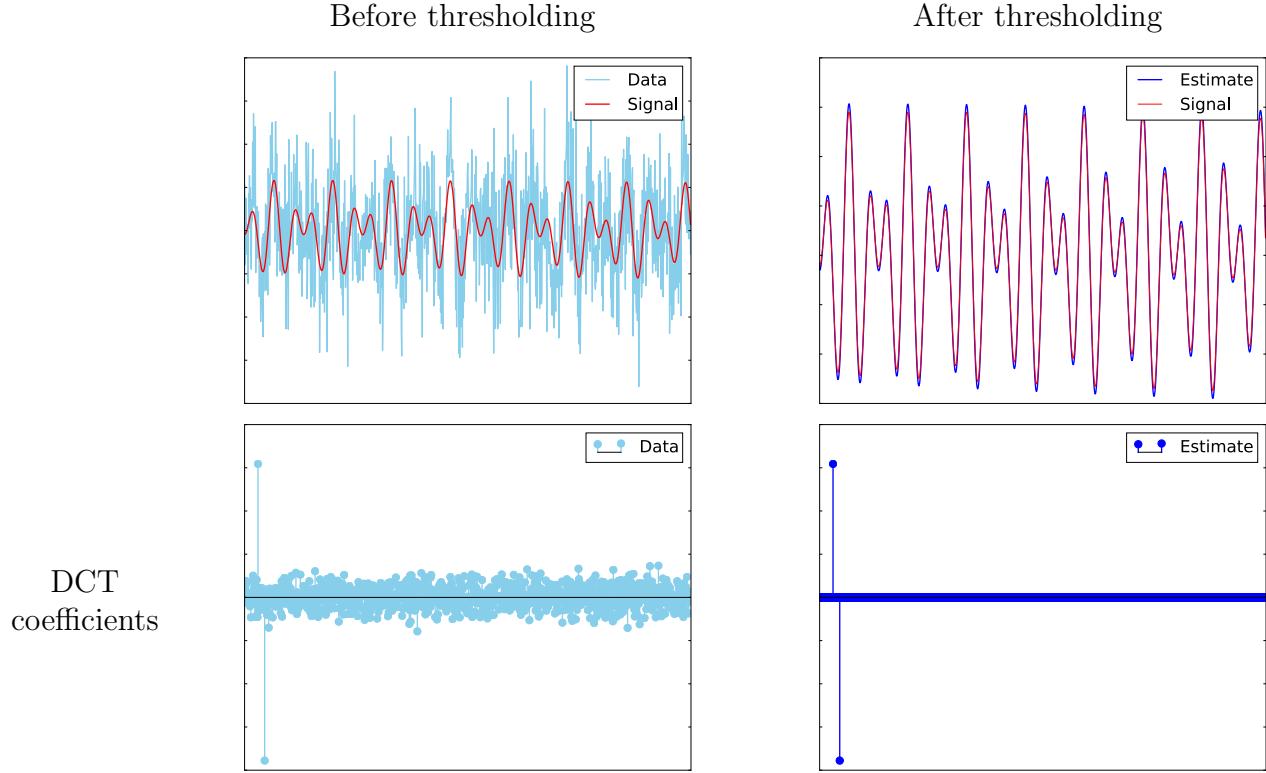


Figure 14: Denoising via hard thresholding in a Fourier basis.

When we apply transforms that capture localized details of signals, such as the wavelet transform or the STFT, sparse representations tend to be highly structured. For example, nonzero wavelet coefficients are often clustered around edges. This is apparent in Figure 10. The reason is that several localized atoms are needed to reproduce sharp variations, whereas a small number of coarse-scale atoms suffice to represent smooth areas of the image.

Thresholding-based denoising can be enhanced by taking into account the *group sparsity* of the signal of interest. If we have a reason to believe that nonzero coefficients in the signal tend to be close to each other, then we should threshold small isolated coefficients, but not similar coefficients that are in the vicinity of large coefficients and therefore may contain useful information. This can be achieved by applying block thresholding.

Algorithm 4.2 (Denoising via block thresholding). *Let \vec{y} follow the model in equation (37). To estimate the signal we:*

1. Compute a decomposition $F\vec{y}$, where F is a frame or basis which sparsifies the signal \vec{x} .
2. Partition the indices of $F\vec{y}$ into blocks $\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_k$.
3. Apply the hard-block-thresholding operator $\mathcal{B}_\eta : \mathbb{C}^n \rightarrow \mathbb{C}^n$ to $F\vec{y}$

$$\mathcal{B}_\eta(\vec{v})[j] := \begin{cases} \vec{v}[j] & \text{if } j \in \mathcal{I}_j \text{ such that } \|\vec{v}_{\mathcal{I}_j}\|_2 > \eta, \\ 0 & \text{otherwise,} \end{cases} \quad (41)$$

SNR

2.5



1



Wavelet coefficients

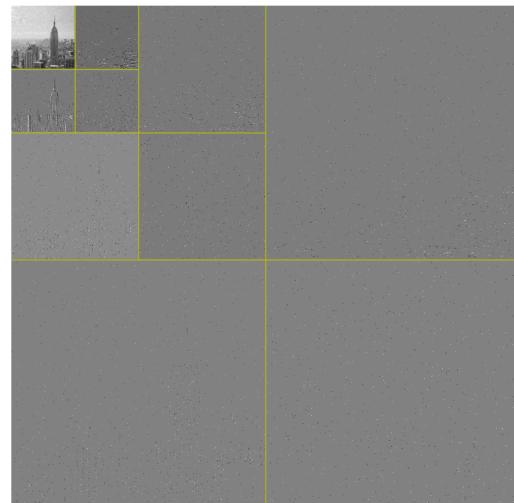
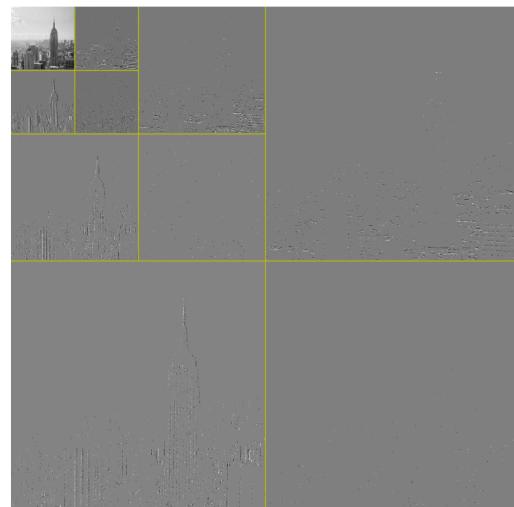


Figure 15: Thresholding-based denoising applied to the images in Figure 12.

SNR

2.5



1



Wavelet coefficients

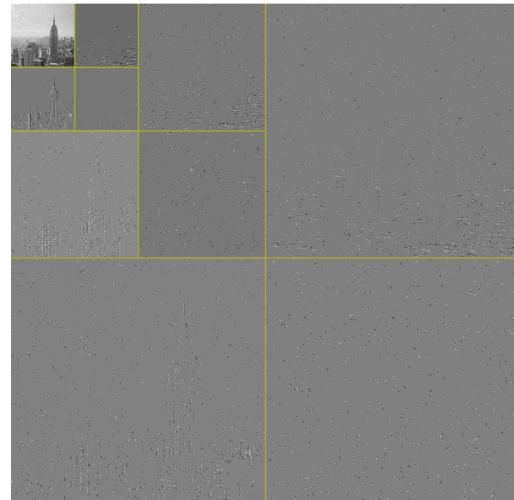
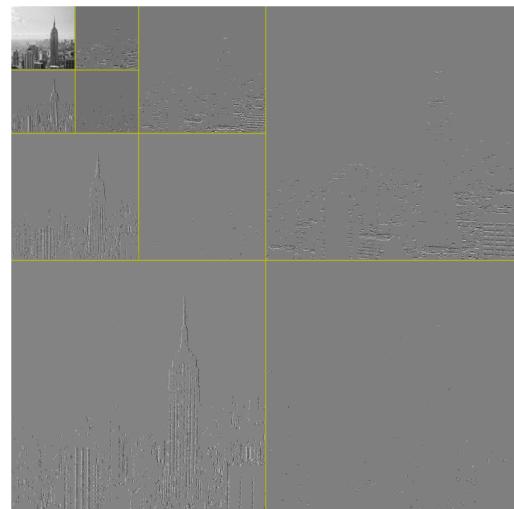


Figure 16: Block-thresholding-based denoising applied to the images in Figure 12.

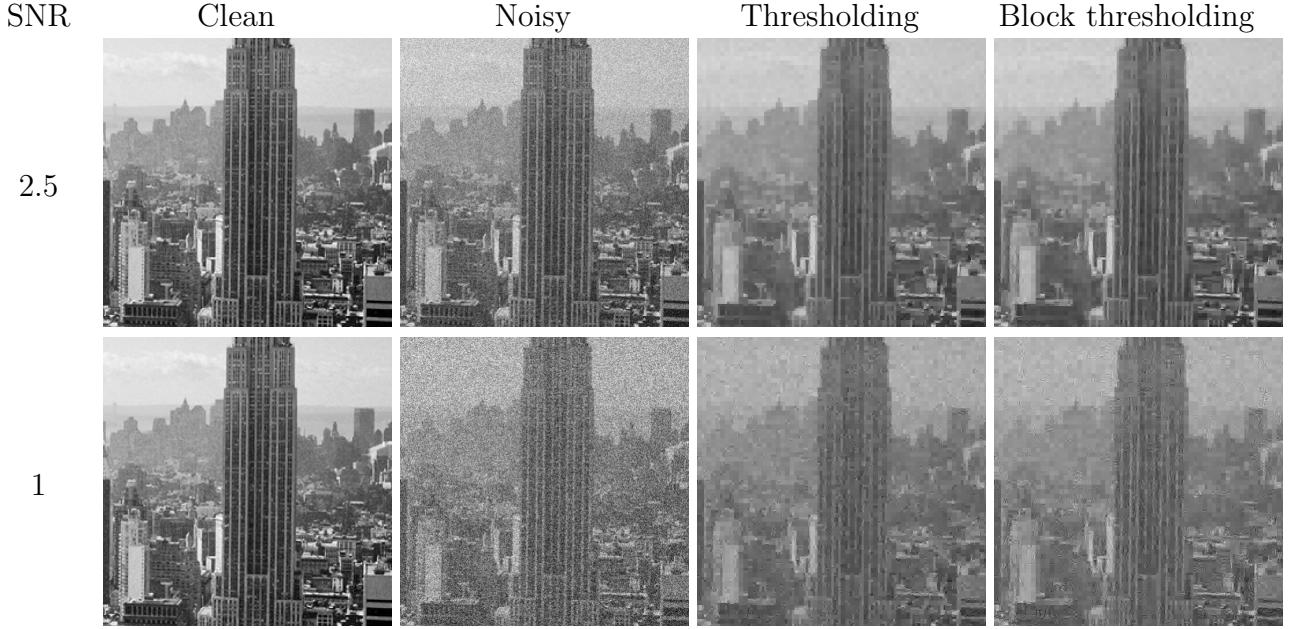


Figure 17: Comparison between thresholding and block-thresholding applied to denoise the images in Figure 12.

where η is adjusted according to the standard deviation of $F\vec{z}$. If F is a basis and \vec{z} is iid Gaussian with standard deviation σ , η should be set larger than $b\sigma$, where b is the number of indices in each block.

4. Compute the estimate by inverting the transform. If F is a basis, then

$$\vec{x}_{\text{est}} := F^{-1}\mathcal{B}_\eta(F\vec{y}). \quad (42)$$

If F is a frame,

$$\vec{x}_{\text{est}} := F^\dagger\mathcal{B}_\eta(F\vec{y}), \quad (43)$$

where F^\dagger is the pseudoinverse of F (any other left inverse of F would also work).

Figure 16 shows the result of denoising the images in Figure 12 by partitioning its 2D Haar coefficients in 4×4 blocks and applying block thresholding. As illustrated by Figure 17 block-thresholding recovers regular such as the vertical lines on the Empire State building more effectively.

We conclude this section with an application of thresholding-based denoising to speech.

Example 4.3 (Speech denoising). The recording shown in Figures 3 and 4 is a short snippet from the movie *Apocalypse Now* where one of the character talks over the noise of a helicopter. We denoise the data using the following methods (click on the links to hear the result):

- **Time thresholding:** The result, which is plotted in Figure 18, sounds terrible because the thresholding eliminates parts of the speech.

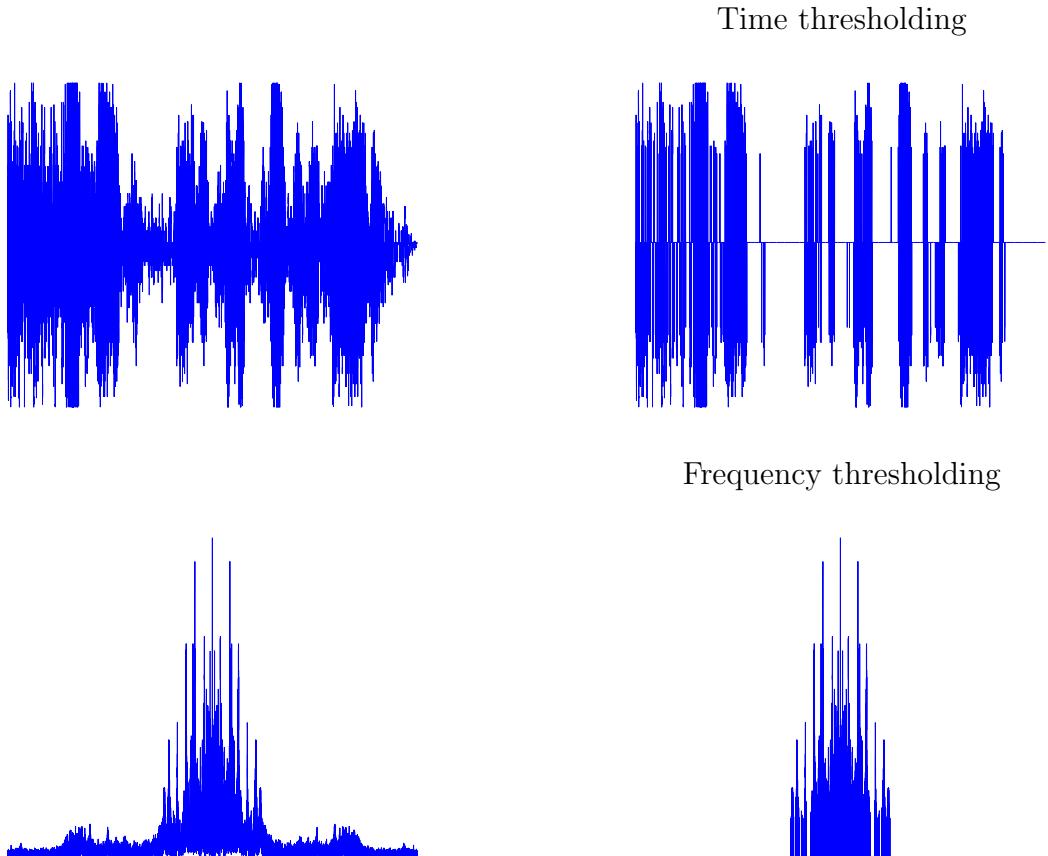


Figure 18: Time thresholding (top row) applied to the noisy data shown in Figure 3. The result sounds terrible because the thresholding eliminates parts of the speech. Below, frequency thresholding is applied to the same data. The result is very low pitch because the thresholding eliminates the high frequencies of both the speech and the noise.

- **Frequency thresholding:** The result has very low pitch because the thresholding eliminates the high frequencies of both the speech and the noise. The spectrum is shown in Figure ?? before and after thresholding.
- **STFT thresholding:** The result is significantly better but isolated STFT coefficients that are not discarded produce *musical noise* artifacts. The corresponding spectrogram is shown in Figure 19.
- **STFT block thresholding:** The result does not suffer from musical noise and retains some of the high-pitch speech. The corresponding spectrogram is shown in Figure 19.

The results are compared visually for a small time segment of the data in Figure 20. △

References

For more information on multiresolution approximations and time-frequency signal processing we refer to the excellent book [?] and references therein.

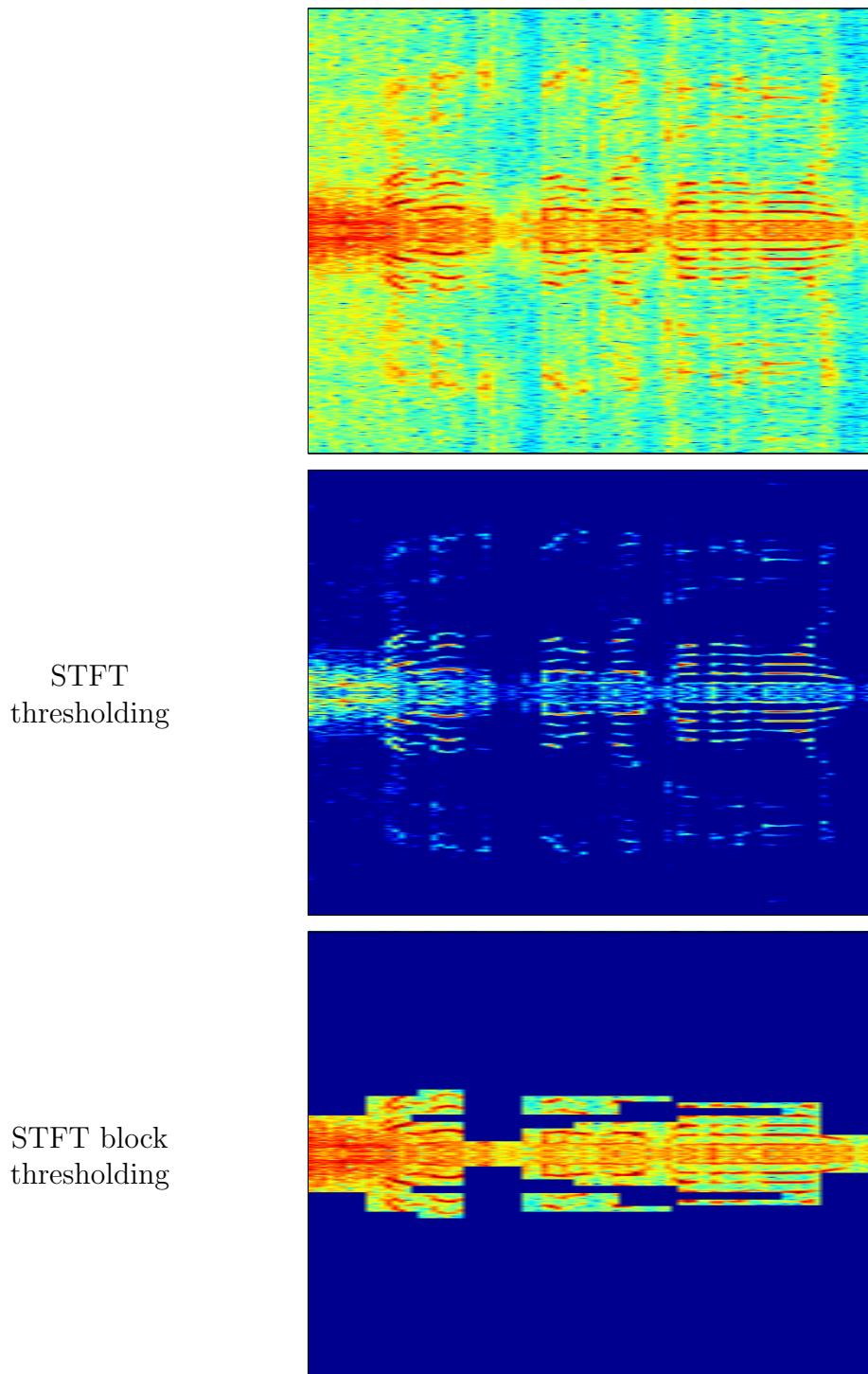


Figure 19: Spectrograms of the noisy signal (above) compared to the estimates obtained by simple thresholding (center) and block thresholding (bottom). The result of [simple thresholding](#) contains musical noise caused by particularly large STFT coefficients caused by the noise that were not thresholded. The result of [block thresholding](#) does not suffer from these artifacts.

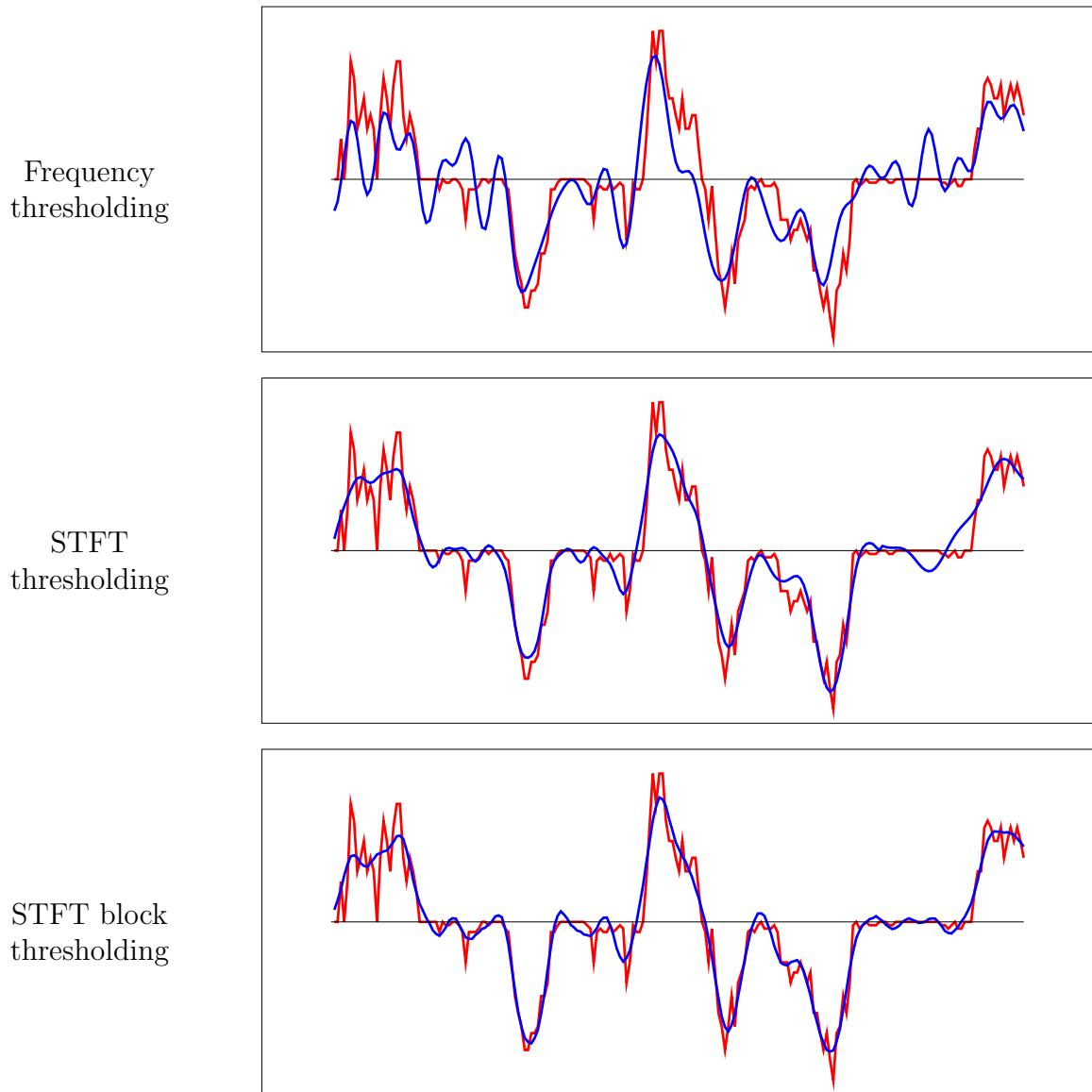


Figure 20: Comparison of the original noisy data (blue) with the denoised signal for the data shown in Figure 3. We compare frequency thresholding (above) and thresholding (center) and block thresholding (below) of STFT coefficients.

Lecture Notes 6: Linear Models

1 Linear regression

1.1 The regression problem

In statistics, regression is the problem of characterizing the relation between a quantity of interest y , called the *response* or the *dependent variable*, and several observed variables x_1, x_2, \dots, x_p , known as *covariates*, *features* or *independent variables*. For example, the response could be the price of a house and the covariates could correspond to the extension, the number of rooms, the year it was built, etc. A regression model would describe how house prices are affected by all of these factors.

More formally, the main assumption in regression models is that the predictor is generated according to a function h applied to the features and then perturbed by some unknown noise z , which is often modeled as additive,

$$y = h(\vec{x}) + z. \quad (1)$$

The aim is to learn h from n examples of responses and their corresponding features

$$(y^{(1)}, \vec{x}^{(1)}), (y^{(2)}, \vec{x}^{(2)}), \dots, (y^{(n)}, \vec{x}^{(n)}). \quad (2)$$

If the regression function h in a model of the form (1) is linear, then the response is modeled as a linear combination of the predictors:

$$y^{(i)} = \langle \vec{x}^{(i)}, \vec{\beta}^* \rangle + z^{(i)}, \quad 1 \leq i \leq n, \quad (3)$$

where $z^{(i)}$ is an entry of the unknown noise vector. The function is parametrized by a vector of coefficients $\vec{\beta}^* \in \mathbb{R}^p$. All we need to fit the linear model to the data is to estimate these coefficients.

Expressing the linear system (3) in matrix form, we have

$$\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix} = \begin{bmatrix} \vec{x}^{(1)}[1] & \vec{x}^{(1)}[2] & \dots & \vec{x}^{(1)}[p] \\ \vec{x}^{(2)}[1] & \vec{x}^{(2)}[2] & \dots & \vec{x}^{(2)}[p] \\ \dots & \dots & \dots & \dots \\ \vec{x}^{(n)}[1] & \vec{x}^{(n)}[2] & \dots & \vec{x}^{(n)}[p] \end{bmatrix} \begin{bmatrix} \vec{\beta}^*[1] \\ \vec{\beta}^*[2] \\ \vdots \\ \vec{\beta}^*[p] \end{bmatrix} + \begin{bmatrix} z^{(1)} \\ z^{(2)} \\ \vdots \\ z^{(n)} \end{bmatrix}. \quad (4)$$

This yields a more succinct representation of the linear-regression model:

$$\vec{y} = X\vec{\beta}^* + \vec{z}, \quad (5)$$

where X is a $n \times p$ matrix containing the features, $\vec{y} \in \mathbb{R}^n$ contains the response and $\vec{z} \in \mathbb{R}^n$ represents the noise.

For simplicity we mostly discuss the linear model (3), but in practice we usually fit an *affine* model that includes a constant term β_0 ,

$$y^{(i)} = \beta_0 + \left\langle \vec{x}^{(i)}, \vec{\beta}^* \right\rangle + z^{(i)}, \quad 1 \leq i \leq n. \quad (6)$$

This term is called an *intercept*, because if there is no noise $y^{(i)}$ is equal to β_0 when the features are all equal to zero. For a least-squares fit (see Section 2 below), β_0 can be shown to equal zero as long as the response \vec{y} and the features $\vec{x}_1, \dots, \vec{x}_p$ are all centered. This is established rigorously in Lemma 2.2. In addition to centering, it is common to normalize the response and the features before fitting a regression model, in order to ensure that all the variables have the same order of magnitude and the model is invariant to changes in units.

Example 1.1 (Linear model for GDP). We consider the problem of building a linear model to predict the gross domestic product (GDP) of a state in the US from its population and unemployment rate. We have available the following data:

	GDP (USD millions)	Population	Unemployment rate (%)
North Dakota	52 089	757 952	2.4
Alabama	204 861	4 863 300	3.8
Mississippi	107 680	2 988 726	5.2
Arkansas	120 689	2 988 248	3.5
Kansas	153 258	2 907 289	3.8
Georgia	525 360	10 310 371	4.5
Iowa	178 766	3 134 693	3.2
West Virginia	73 374	1 831 102	5.1
Kentucky	197 043	4 436 974	5.2
Tennessee	???	6 651 194	3.0

In this example, the GDP is the response, whereas the population and the unemployment rate are the features. Our goal is to fit a linear model to the data so that we can predict the GDP of Tennessee, using a linear model. We begin by centering and normalizing the data. The averages of the response and of the features are

$$\text{av}(\vec{y}) = 179 236, \quad \text{av}(X) = \begin{bmatrix} 3 802 073 & 4.1 \end{bmatrix}. \quad (7)$$

The empirical standard deviations are

$$\text{std}(\vec{y}) = 396 701, \quad \text{std}(X) = \begin{bmatrix} 7 720 656 & 2.80 \end{bmatrix}. \quad (8)$$

We subtract the average and divide by the standard deviations so that both the response and the features are centered and on the same scale,

$$\vec{y} = \begin{bmatrix} -0.321 \\ 0.065 \\ -0.180 \\ -0.148 \\ -0.065 \\ 0.872 \\ -0.001 \\ -0.267 \\ 0.045 \end{bmatrix}, \quad X = \begin{bmatrix} -0.394 & -0.600 \\ 0.137 & -0.099 \\ -0.105 & 0.401 \\ -0.105 & -0.207 \\ -0.116 & -0.099 \\ 0.843 & 0.151 \\ -0.086 & -0.314 \\ -0.255 & 0.366 \\ 0.082 & 0.401 \end{bmatrix}. \quad (9)$$

To obtain the estimate for the GDP of Tennessee we fit the model

$$\vec{y} \approx X\vec{\beta}, \quad (10)$$

rescale according to the standard deviations (8) and recenter using the averages (7). The final estimate is

$$\vec{y}^{\text{Ten}} = \text{av}(\vec{y}) + \text{std}(\vec{y}) \left\langle \vec{x}_{\text{norm}}^{\text{Ten}}, \vec{\beta} \right\rangle \quad (11)$$

where $\vec{x}_{\text{norm}}^{\text{Ten}}$ is centered using $\text{av}(X)$ and normalized using $\text{std}(X)$. \triangle

1.2 Overfitting

Imagine that a friend tells you:

I found a cool way to predict the daily temperature in New York: It's just a linear combination of the temperature in every other state. I fit the model on data from the last month and a half and it's perfect!

Your friend is not lying. The problem is that in this example the number of data points is roughly the same as the number of parameters. If $n \leq p$ we can find a $\vec{\beta}$ such that $\vec{y} = X\vec{\beta}$ exactly, even if \vec{y} and X have nothing to do with each other! This is called *overfitting*: the model is too flexible given the available data. Recall from linear algebra that for a matrix $A \in \mathbb{R}^{n \times p}$ that is full rank, the linear system of equations

$$A\vec{b} = \vec{c} \quad (12)$$

is (1) underdetermined if $n < p$, meaning that it has infinite solutions, (2) determined if $n = p$, meaning that there is a unique solution, and (3) overdetermined if $n > p$. Fitting a linear model without any additional assumptions only makes sense in the overdetermined regime. In that case, an exact solution exists if $\vec{b} \in \text{col}(A)$, which is never the case in practice due to the presence of noise. However, if we manage to find a vector \vec{b} such that $A\vec{b}$ is a good approximation to \vec{c} when $n > p$ then this is an indication that the linear model is capturing some underlying structure in the problem. We make this statement more precise in Section 2.4

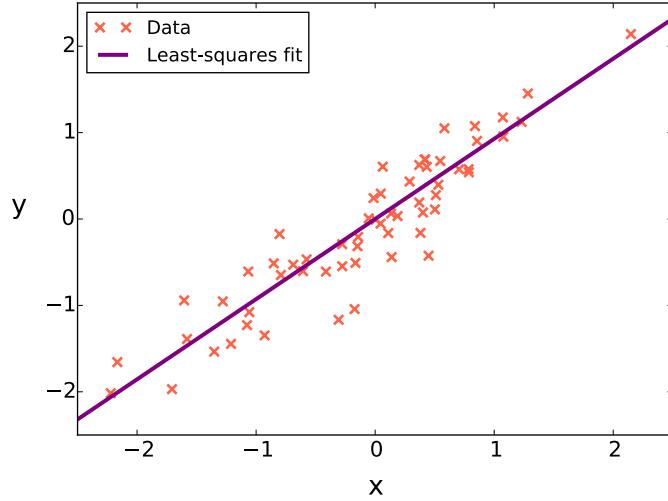


Figure 1: Linear model learned via least-squares fitting for a simple example where there is just one feature ($p = 1$) and 40 examples ($n = 40$).

2 Least-squares estimation

2.1 Minimizing the ℓ_2 -norm approximation error

To calibrate the linear regression model $\vec{y} \approx X\vec{\beta}$ it is necessary to choose a metric to evaluate the fit achieved by the model. By far, the most popular metric is the sum of the squares of the fitting error,

$$\sum_{i=1}^n \left(y^{(i)} - \langle \vec{x}^{(i)}, \vec{\beta} \rangle \right)^2 = \left\| \vec{y} - X\vec{\beta} \right\|_2^2. \quad (13)$$

The least-squares estimate $\vec{\beta}_{\text{LS}}$ is the vector of coefficients that minimizes this cost function,

$$\vec{\beta}_{\text{LS}} := \arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2. \quad (14)$$

The least-squares cost function is convenient from a computational view, since it is convex and can be minimized efficiently (in fact, as we will see in a moment it has a closed-form solution). In addition, it has intuitive geometric and probabilistic interpretations. Figure 1 shows the linear model learned using least squares in a simple example where there is just one feature ($p = 1$) and 40 examples ($n = 40$).

Theorem 2.1. *If X is full rank and $n \geq p$, for any $\vec{y} \in \mathbb{R}^n$ we have*

$$\vec{\beta}_{\text{LS}} := \arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2 \quad (15)$$

$$= VS^{-1}U^T\vec{y} \quad (16)$$

$$= (X^T X)^{-1} X^T \vec{y}, \quad (17)$$

where USV^T is the SVD of X .

Proof. We consider the decomposition of \vec{y} into its orthogonal projection $UU^T\vec{y}$ onto the column space of $X \text{ col}(X)$ and its projection $(I - UU^T)\vec{y}$ onto the orthogonal complement of $\text{col}(X)$. $X\vec{\beta}$ belongs to $\text{col}(X)$ for any β and is consequently orthogonal to $(I - UU^T)\vec{y}$ (as is $UU^T\vec{y}$), so that

$$\arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 = \arg \min_{\vec{\beta}} \left\| (I - UU^T)\vec{y} \right\|_2^2 + \left\| UU^T\vec{y} - X\vec{\beta} \right\|_2^2 \quad (18)$$

$$= \arg \min_{\vec{\beta}} \left\| UU^T\vec{y} - X\vec{\beta} \right\|_2^2 \quad (19)$$

$$= \arg \min_{\vec{\beta}} \left\| UU^T\vec{y} - USV^T\vec{\beta} \right\|_2^2. \quad (20)$$

Since U has orthonormal columns, for any vector $\vec{v} \in \mathbb{R}^p$ $\|U\vec{v}\|_2 = \|\vec{v}\|_2$, which implies

$$\arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 = \arg \min_{\vec{\beta}} \left\| U^T\vec{y} - SV^T\vec{\beta} \right\|_2^2 \quad (21)$$

If X is full rank and $n \geq p$, then SV^T is square and full rank. It therefore has a unique inverse, which is equal to VS^{-1} . As a result $VS^{-1}U^T\vec{y} = (X^TX)^{-1}X^T\vec{y}$ is the unique solution to the optimization problem (it is the only vector that yields a value of zero for the cost function). \square

The following lemma shows that centering the data before computing the least-squares fit is exactly equivalent to fitting an affine model with the same cost function.

Lemma 2.2 (Proof in Section 5.1). *For any matrix $X \in \mathbb{R}^{n \times m}$ and any vector \vec{y} , let*

$$\left\{ \beta_{LS,0}, \vec{\beta}_{LS} \right\} := \arg \min_{\beta_0, \vec{\beta}} \left\| \vec{y} - X\vec{\beta} - \beta_0 \vec{1} \right\|_2^2 \quad (22)$$

be the coefficients corresponding to an affine fit, where $\vec{1}$ is a vector containing n ones, and let

$$\vec{\beta}_{LS}^{\text{cent}} := \arg \min_{\vec{\beta}} \left\| \vec{y}^{\text{cent}} - X^{\text{cent}}\vec{\beta} \right\|_2^2 \quad (23)$$

be the coefficients of a linear fit after centering both X and \vec{y} using their respective averages (in the case of X , the column-wise average). Then,

$$X\vec{\beta}_{LS} + \beta_{LS,0} = X^{\text{cent}}\vec{\beta}_{LS}^{\text{cent}} + \text{av}(y). \quad (24)$$

Example 2.3 (Linear model for GDP (continued)). The least-squares estimate for the regression coefficients in the linear GDP model is equal to

$$\vec{\beta}_{LS} = \begin{bmatrix} 1.019 \\ -0.111 \end{bmatrix}. \quad (25)$$

The GDP seems to be proportional to the population and inversely proportional to the unemployment rate. We now compare the fit provided by the linear model to the original data, as well as its prediction of the GDP of Tennessee:

	GDP	Estimate
North Dakota	52 089	46 241
Alabama	204 861	239 165
Mississippi	107 680	119 005
Arkansas	120 689	145 712
Kansas	153 258	136 756
Georgia	525 360	513 343
Iowa	178 766	158 097
West Virginia	73 374	59 969
Kentucky	197 043	194 829
Tennessee	328 770	345 352

△

Example 2.4 (Global warming). In this example we describe the application of linear regression to climate data. In particular, we analyze temperature data taken in a weather station in Oxford over 150 years.¹ Our objective is not to perform prediction, but rather to determine whether temperatures have risen or decreased during the last 150 years in Oxford.

In order to separate the temperature into different components that account for seasonal effects we use a simple linear with three predictors and an intercept

$$y \approx \beta_0 + \beta_1 \cos\left(\frac{2\pi t}{12}\right) + \beta_2 \sin\left(\frac{2\pi t}{12}\right) + \beta_3 t \quad (26)$$

where t denotes the time in months. The corresponding matrix of predictors is

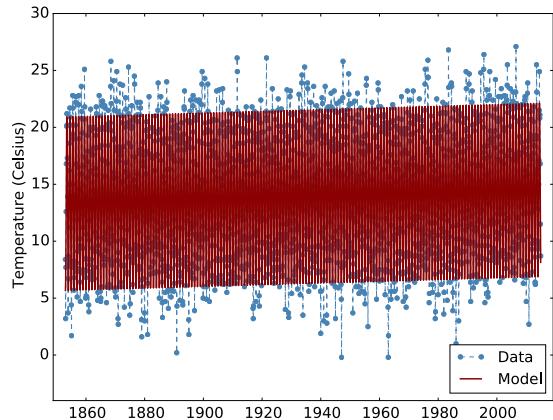
$$X := \begin{bmatrix} 1 & \cos\left(\frac{2\pi t_1}{12}\right) & \sin\left(\frac{2\pi t_1}{12}\right) & t_1 \\ 1 & \cos\left(\frac{2\pi t_2}{12}\right) & \sin\left(\frac{2\pi t_2}{12}\right) & t_2 \\ \dots & \dots & \dots & \dots \\ 1 & \cos\left(\frac{2\pi t_n}{12}\right) & \sin\left(\frac{2\pi t_n}{12}\right) & t_n \end{bmatrix}. \quad (27)$$

The intercept β_0 represents the mean temperature, β_1 and β_2 account for periodic yearly fluctuations and β_3 is the overall trend. If β_3 is positive then the model indicates that temperatures are increasing, if it is negative then it indicates that temperatures are decreasing.

The results of fitting the linear model using least squares are shown in Figures 2 and 3. The fitted model indicates that both the maximum and minimum temperatures have an increasing trend of about 0.8 degrees Celsius (around 1.4 degrees Fahrenheit). △

¹The data are available at <http://www.metoffice.gov.uk/pub/data/weather/uk/climate/stationdata/oxforddata.txt>.

Maximum temperature



Minimum temperature

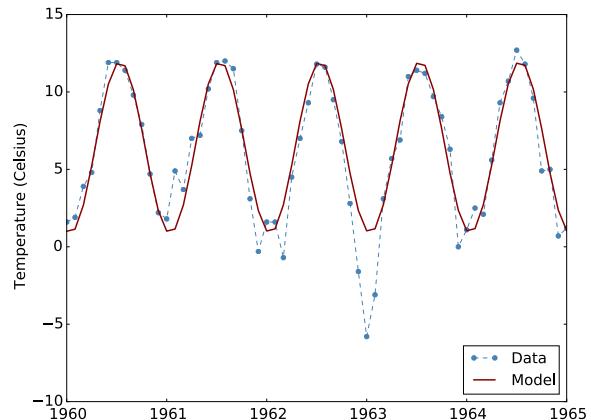
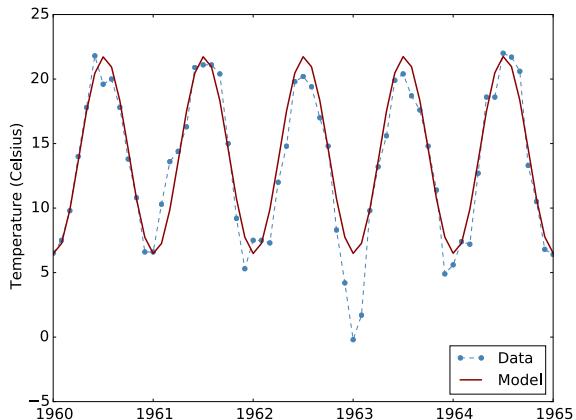
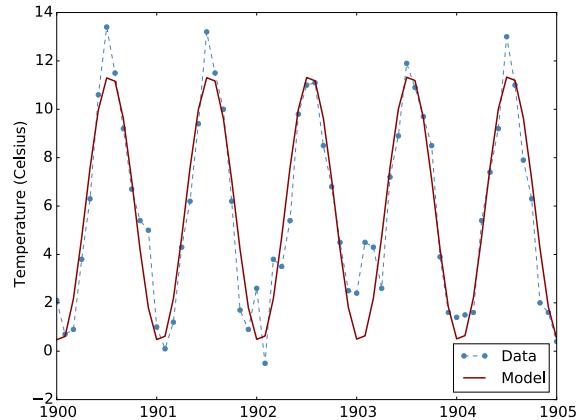
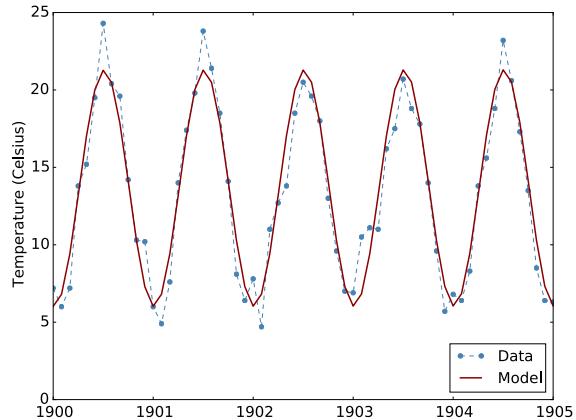
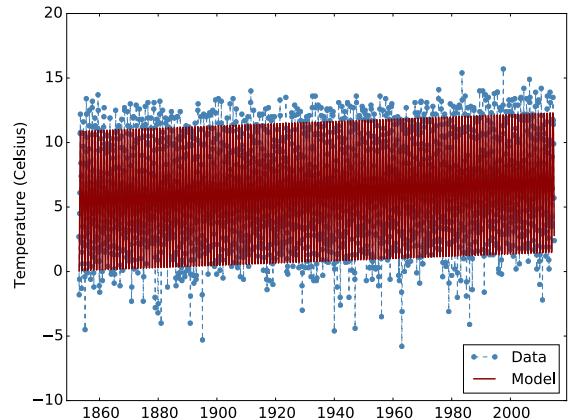


Figure 2: Temperature data together with the linear model described by (26) for both maximum and minimum temperatures.

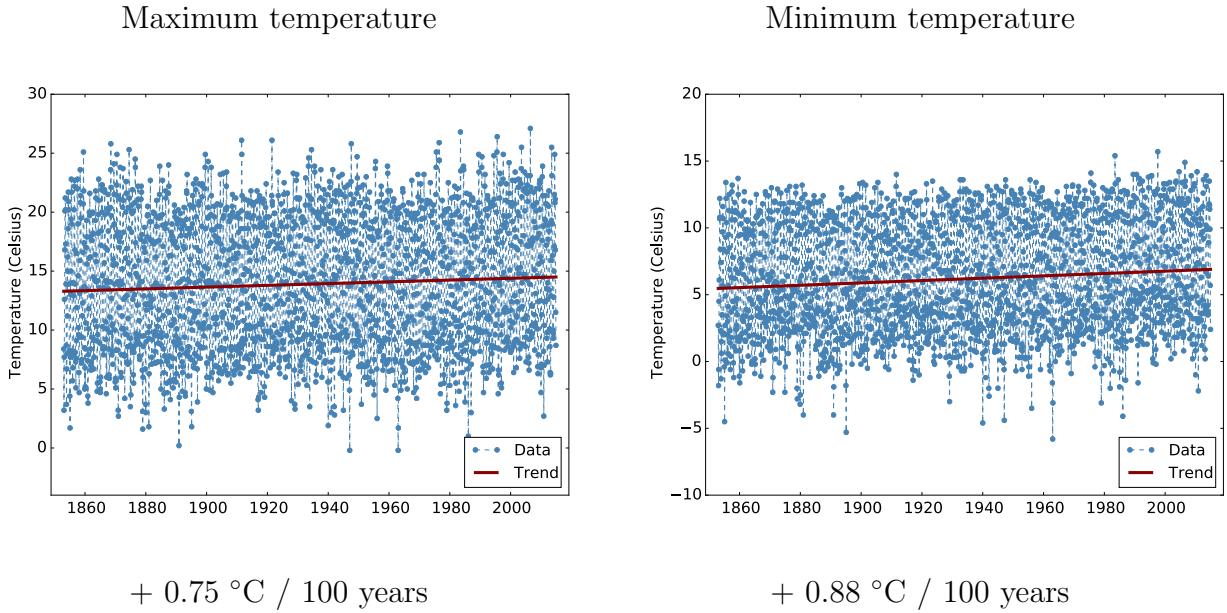


Figure 3: Temperature trend obtained by fitting the model described by (26) for both maximum and minimum temperatures.

2.2 Geometric interpretation of least-squares regression

The following corollary of Theorem 2.1 provides an intuitive geometric interpretation of the linear approximation obtained from a least-squares fit. The least-squares fit yields the vector $X\vec{\beta}$ in the column space $\text{col}(X)$ of the features that is closest to \vec{y} in ℓ_2 norm. $X\vec{\beta}_{\text{LS}}$ is therefore the orthogonal projection of \vec{y} onto $\text{col}(X)$, as depicted in Figure 4.

Corollary 2.5. *The least-squares approximation of \vec{y} obtained by solving problem (14)*

$$\vec{y}_{\text{LS}} = X\vec{\beta}_{\text{LS}} \quad (28)$$

is equal to the orthogonal projection of \vec{y} onto the column space of X .

Proof.

$$X\vec{\beta}_{\text{LS}} = USV^T VS^{-1} U^T \vec{y} \quad (29)$$

$$= UU^T \vec{y} \quad (30)$$

□

Example 2.6 (Denoising of face images). In Example 7.4 of Lecture Notes 1, we denoised a noisy image by projecting it onto the span of a set of clean images. This is equivalent to solving a least-squares linear-regression problem in which the response is the noisy images and the columns of the matrix of features correspond to the clean faces. The regression coefficients are used to combine the different clean faces linearly to produce the estimate. △

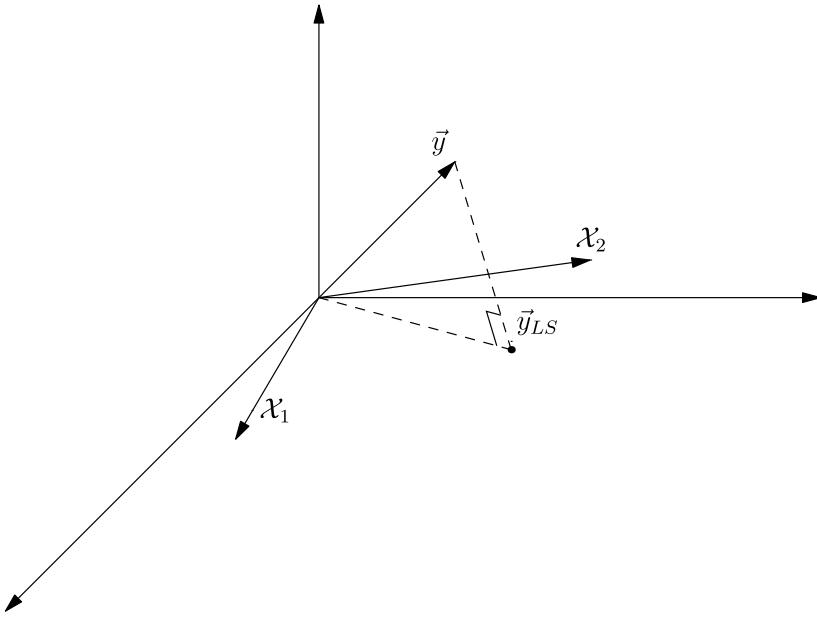


Figure 4: Illustration of Corollary 2.5. The least-squares solution is the orthogonal projection of the data onto the subspace spanned by the columns of X , denoted by X_1 and X_2 .

2.3 Probabilistic interpretation of least-squares regression

In this section we derive the least-squares regression estimate as a maximum-likelihood (ML) estimator. ML estimation is a popular method for learning parametric models. In parametric estimation we assume that the data are sampled from a known distribution that depends on some unknown parameters, which we aim to estimate. The *likelihood* function is the joint pmf or pdf of the data, interpreted as a function of the unknown parameters.

Definition 2.7 (Likelihood function). *Given a realization $\vec{y} \in \mathbb{R}^n$ of random vector \vec{y} with joint pdf $f_{\vec{\beta}}$ parameterized by a vector of parameters $\vec{\beta} \in \mathbb{R}^m$, the likelihood function is*

$$\mathcal{L}_{\vec{y}}(\vec{\beta}) := f_{\vec{\beta}}(\vec{y}). \quad (31)$$

The log-likelihood function is equal to the logarithm of the likelihood function $\log \mathcal{L}_{\vec{y}}(\vec{\beta})$.

The likelihood function represents the probability density of the parametric distribution at the observed data, i.e. it quantifies how *likely* the data are according to the model. Therefore, higher likelihood values indicate that the model is better adapted to the samples. The maximum-likelihood (ML) estimator is a very popular parameter estimator based on maximizing the likelihood (or equivalently the log-likelihood).

Definition 2.8 (Maximum-likelihood estimator). *The maximum likelihood (ML) estimator of the*

vector of parameters $\vec{\beta} \in \mathbb{R}^m$ is

$$\vec{\beta}_{\text{ML}}(\vec{y}) := \arg \max_{\vec{\beta}} \mathcal{L}_{\vec{y}}(\vec{\beta}) \quad (32)$$

$$= \arg \max_{\vec{\beta}} \log \mathcal{L}_{\vec{y}}(\vec{\beta}). \quad (33)$$

The maximum of the likelihood function and that of the log-likelihood function are at the same location because the logarithm is a monotone function.

The following lemma shows that the least-squares estimate can be interpreted as an ML estimator.

Lemma 2.9. *Let $\vec{y} \in \mathbb{R}^n$ be a realization of a random vector*

$$\vec{y} := X\vec{\beta} + \vec{z}, \quad (34)$$

where \vec{z} is iid Gaussian with mean zero and variance σ^2 . If $X \in \mathbb{R}^{n \times m}$ is known, then the ML estimate of $\vec{\beta}$ is equal to the least-squares estimate

$$\vec{\beta}_{\text{ML}} = \arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2^2. \quad (35)$$

Proof. For a fixed $\vec{\beta}$, the joint pdf of \vec{y} is equal to

$$f_{\vec{\beta}}(\vec{y}) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma} \exp \left(-\frac{1}{2\sigma^2} (\vec{y}[i] - (X\vec{\beta})[i])^2 \right) \quad (36)$$

$$= \frac{1}{\sqrt{(2\pi)^n}\sigma^n} \exp \left(-\frac{1}{2\sigma^2} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 \right). \quad (37)$$

The likelihood is the probability density function of \vec{y} evaluated at the observed data \vec{y} and interpreted as a function of the coefficient vector $\vec{\beta}$,

$$\mathcal{L}_{\vec{y}}(\vec{\beta}) = \frac{1}{\sqrt{(2\pi)^n}} \exp \left(-\frac{1}{2} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 \right). \quad (38)$$

To find the ML estimate, we maximize the log likelihood

$$\vec{\beta}_{\text{ML}} = \arg \max_{\vec{\beta}} \mathcal{L}_{\vec{y}}(\vec{\beta}) \quad (39)$$

$$= \arg \max_{\vec{\beta}} \log \mathcal{L}_{\vec{y}}(\vec{\beta}) \quad (40)$$

$$= \arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2^2. \quad (41)$$

□

2.4 Analysis of the least-squares estimate

In this section we analyze the solution of the least-squares regression fit under the assumption that the data are indeed generated according to a linear model with additive noise,

$$\vec{y} := X\vec{\beta}^* + \vec{z}, \quad (42)$$

where $X \in \mathbb{R}^{n \times m}$ and $\vec{z} \in \mathbb{R}^n$. In that case, we can express the least-squares solution in terms of the true coefficients $\vec{\beta}^*$, the feature matrix X and the noise \vec{z} applying Theorem 2.1. The estimation error equals

$$\vec{\beta}_{\text{LS}} - \vec{\beta}^* = (X^T X)^{-1} X^T (X\vec{\beta}^* + \vec{z}) \quad (43)$$

$$= (X^T X)^{-1} X^T \vec{z}, \quad (44)$$

as long as X is full rank.

Equation (44) implies that if the noise is random and has zero mean, then the expected error is equal to zero. In statistics lingo, the least-squares estimate is *unbiased*, which means that the estimator is centered at the true coefficient vector $\vec{\beta}^*$.

Lemma 2.10 (Least-squares estimator is unbiased). *If the noise \vec{z} is a random vector with zero mean, then*

$$\mathbb{E}(\vec{\beta}_{\text{LS}} - \vec{\beta}^*) = 0. \quad (45)$$

Proof. By (44) and linearity of expectation

$$\mathbb{E}(\vec{\beta}_{\text{LS}} - \vec{\beta}^*) = (X^T X)^{-1} X^T \mathbb{E}(\vec{z}) = 0. \quad (46)$$

□

We can bound the error incurred by the least-squares estimate in terms of the noise and the singular values of the feature matrix X .

Theorem 2.11 (Least-squares error). *For data of the form (42), we have*

$$\frac{\|\vec{z}\|_2}{\sigma_1} \leq \left\| \vec{\beta}_{\text{LS}} - \vec{\beta}^* \right\|_2 \leq \frac{\|\vec{z}\|_2}{\sigma_p}, \quad (47)$$

as long as X is full rank, where σ_1 and σ_p denote the largest and smallest singular value of X respectively.

Proof. By (44)

$$\vec{\beta}_{\text{LS}} - \vec{\beta}^* = VS^{-1}U^T \vec{z}. \quad (48)$$

The smallest and largest singular values of $VS^{-1}U$ are $1/\sigma_1$ and $1/\sigma_p$ respectively so by Theorem 2.7 in Lecture Notes 2

$$\frac{\|\vec{z}\|_2}{\sigma_1} \leq \left\| VS^{-1}U^T \vec{z} \right\|_2 \leq \frac{\|\vec{z}\|_2}{\sigma_p}. \quad (49)$$

□

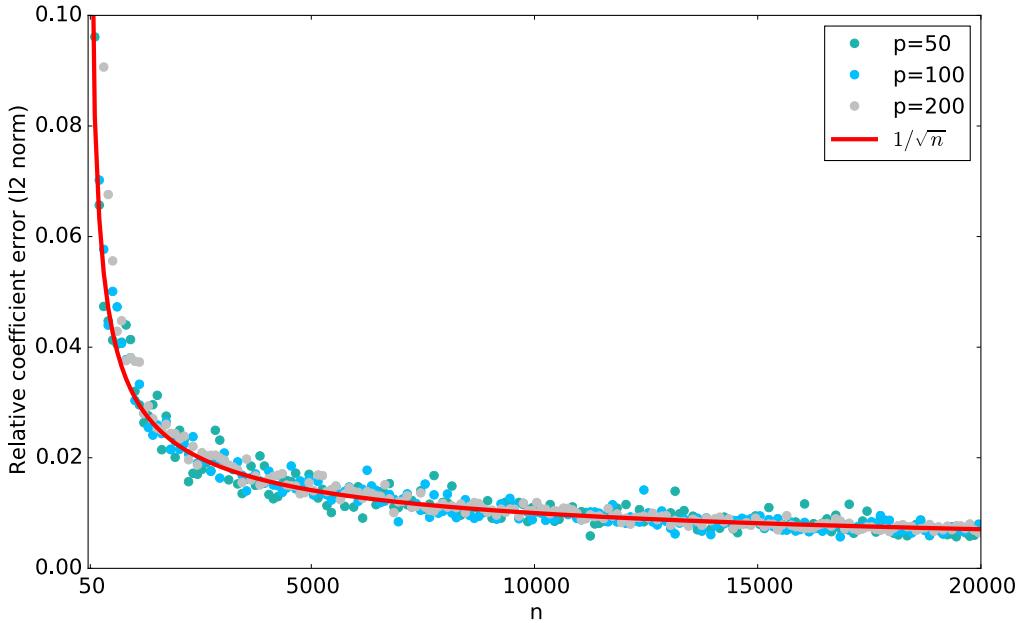


Figure 5: Relative ℓ_2 -norm error of the least-squares coefficient estimate as n grows. The entries of \mathbf{X} , $\vec{\beta}^*$ and \vec{z} are sampled iid from a standard Gaussian distribution. The error scales as $1/\sqrt{n}$ as predicted by Theorem 2.12.

Let us assume that the norm of the noise $\|\vec{z}\|_2$ is fixed. In that case, by (48) the largest error occurs when \vec{z} is aligned with \vec{u}_p , the singular vector corresponding to σ_p , whereas the smallest error occurs when \vec{z} is aligned with \vec{u}_1 , the singular vector corresponding to σ_1 . To analyze what happens in a *typical* linear-regression problem, we can assume that X and \vec{z} are sampled from a Gaussian distribution. The following theorem shows that in this case, the ratio between the norms of the error and the noise (or equivalently the error when the norm of the noise is fixed to one) concentrates around $\sqrt{p/n}$. In particular, for a fixed number of features it decreases as $1/\sqrt{n}$ with the number of available data, becoming arbitrarily small as $n \rightarrow \infty$. This is illustrated by Figure 5, which shows the results of a numerical experiment that match the theoretical analysis very closely.

Theorem 2.12 (Non-asymptotic bound on least-squares error). *Let*

$$\vec{y} := \mathbf{X}\vec{\beta}^* + \vec{z}, \quad (50)$$

where the entries of the $n \times p$ matrix \mathbf{X} and the n -dimensional vector \vec{z} are iid standard Gaussians. The least-squares estimate satisfies

$$\sqrt{\frac{(1-\epsilon)}{(1+\epsilon)}} \sqrt{\frac{p}{n}} \leq \left\| \vec{\beta}_{\text{LS}} - \vec{\beta}^* \right\|_2 \leq \sqrt{\frac{(1+\epsilon)}{(1-\epsilon)}} \sqrt{\frac{p}{n}} \quad (51)$$

with probability at least $1 - 1/p - 2 \exp(-p\epsilon^2/8)$ as long as $n \geq 64p \log(12/\epsilon)/\epsilon^2$.

Proof. By the same argument used to derive (49), we have

$$\frac{\|\mathbf{U}^T \vec{z}\|_2}{\sigma_1} \leq \left\| \mathbf{V} \mathbf{S}^{-1} \mathbf{U}^T \vec{z} \right\|_2 \leq \frac{\|\mathbf{U}^T \vec{z}\|_2}{\sigma_p}. \quad (52)$$

By Theorem 2.10 in Lecture Notes 3 with probability $1 - 2 \exp(-p\epsilon^2/8)$

$$(1 - \epsilon)p \leq \|\mathbf{U}^T \vec{\mathbf{z}}\|_2^2 \leq (1 + \epsilon)p, \quad (53)$$

where \mathbf{U} contains the left singular vectors of \mathbf{X} . By Theorem 3.7 in Lecture Notes 3 with probability $1 - 1/p$

$$\sqrt{n(1 - \epsilon)} \leq \sigma_p \leq \sigma_1 \leq \sqrt{n(1 + \epsilon)} \quad (54)$$

as long as $n \geq 64p \log(12/\epsilon)/\epsilon^2$. The result follows from combining (52) with (53) and (54) which hold simultaneously with probability at least $1 - 1/p - 2 \exp(-p\epsilon^2/8)$ by the union bound. \square

3 Regularization

3.1 Noise amplification

Theorem 2.12 characterizes the performance of least-squares regression when the feature matrix is *well-conditioned*, which means that its smallest singular value is not too small with respect to the largest singular value.

Definition 3.1 (Condition number). *The condition number of a matrix $A \in \mathbb{R}^{n \times p}$, $n \geq p$, is equal to the ratio σ_1/σ_p of its largest and smallest singular values σ_1 and σ_p .*

In numerical linear algebra, a system of equations is said to be *ill conditioned* if the condition number is large. The reason is that perturbations aligned with the singular vector corresponding to the smallest singular value may be amplified dramatically when inverting the system. This is exactly what happens in linear regression problems when the feature matrix X is not well conditioned. The component of the noise that falls in the direction of the singular vector corresponding to the smallest singular value *blows up*, as proven in the following theorem.

Lemma 3.2 (Noise amplification). *Let $X \in \mathbb{R}^{n \times p}$ be a matrix such that m singular values are smaller than η and let*

$$\vec{\mathbf{y}} := X\vec{\beta}^* + \mathbf{z}, \quad (55)$$

where the entries of $\vec{\mathbf{z}}$ are iid standard Gaussians. Then, with probability at least $1 - 2 \exp(-m\epsilon^2/8)$

$$\left\| \vec{\beta}_{\text{LS}} - \vec{\beta}^* \right\|_2 \geq \frac{m\sqrt{1 - \epsilon}}{\eta}. \quad (56)$$

Proof. Let $X = USV^T$ be the SVD of X , $\vec{u}_1, \dots, \vec{u}_p$ the columns of U and $\sigma_1, \dots, \sigma_p$ the singular

values. By (44)

$$\left\| \vec{\beta}_{\text{LS}} - \vec{\beta}^* \right\|_2^2 = \left\| VS^{-1} U^T \vec{\mathbf{z}} \right\|_2^2 \quad (57)$$

$$= \left\| S^{-1} U^T \vec{\mathbf{z}} \right\|_2^2 \quad V \text{ is an orthogonal matrix} \quad (58)$$

$$= \sum_i^p \frac{(\vec{u}_i^T \vec{\mathbf{z}})^2}{\sigma_i^2} \quad (59)$$

$$\geq \frac{1}{\eta^2} \sum_i^m (\vec{u}_i^T \vec{\mathbf{z}})^2. \quad (60)$$

The result follows because $\sum_i^m (\vec{u}_i^T \vec{\mathbf{z}})^2 \geq 1 - \epsilon$ with probability at least $1 - 2 \exp(-m\epsilon^2/8)$ by Theorem 2.10 in Lecture Notes 3 . \square

We illustrate noise amplification in least-squares regression through a simple example.

Example 3.3 (Noise amplification). Consider a linear-regression problem with data of the form

$$\vec{y} := X \vec{\beta}^* + \vec{z}, \quad (61)$$

where

$$X := \begin{bmatrix} 0.212 & -0.099 \\ 0.605 & -0.298 \\ -0.213 & 0.113 \\ 0.589 & -0.285 \\ 0.016 & 0.006 \\ 0.059 & 0.032 \end{bmatrix}, \quad \vec{\beta}^* := \begin{bmatrix} 0.471 \\ -1.191 \end{bmatrix}, \quad \vec{z} := \begin{bmatrix} 0.066 \\ -0.077 \\ -0.010 \\ -0.033 \\ 0.010 \\ 0.028 \end{bmatrix}. \quad (62)$$

The ℓ_2 norm of the noise is 0.11. The feature matrix is ill conditioned, its condition number is 100,

$$X = USV^T = \begin{bmatrix} -0.234 & 0.427 \\ -0.674 & -0.202 \\ 0.241 & 0.744 \\ -0.654 & 0.350 \\ 0.017 & -0.189 \\ 0.067 & 0.257 \end{bmatrix} \begin{bmatrix} 1.00 & 0 \\ 0 & \textcolor{red}{0.01} \end{bmatrix} \begin{bmatrix} -0.898 & 0.440 \\ 0.440 & 0.898 \end{bmatrix}. \quad (63)$$

As a result, the component of \vec{z} in the direction of the second singular vector is amplified by a

factor of 100! By (44), the error in the coefficient estimate is

$$\vec{\beta}_{\text{LS}} - \vec{\beta}^* = VS^{-1}U^T \vec{z} \quad (64)$$

$$= V \begin{bmatrix} 1.00 & 0 \\ 0 & 100.00 \end{bmatrix} U^T \vec{z} \quad (65)$$

$$= V \begin{bmatrix} 0.058 \\ 3.004 \end{bmatrix} \quad (66)$$

$$= \begin{bmatrix} 1.270 \\ 2.723 \end{bmatrix}, \quad (67)$$

so that the norm of the error satisfies

$$\frac{\|\vec{\beta}_{\text{LS}} - \vec{\beta}^*\|_2}{\|\vec{z}\|_2} = 27.00. \quad (68)$$

△

The feature matrix is ill conditioned if any subset of columns is close to being linearly dependent, since in that case there must be a vector that is *almost* in the null space of the matrix. This occurs when some of the feature vectors are highly correlated, a phenomenon known as *multicollinearity* in the statistics lingo. The following lemma shows how two feature vectors being very correlated results in poor conditioning.

Lemma 3.4 (Proof in Section 5.2). *For any matrix $X \in \mathbb{R}^{n \times p}$, with columns normalized to have unit ℓ_2 norm, if any two distinct columns X_i and X_j satisfy*

$$\langle X_i, X_j \rangle^2 \geq 1 - \epsilon^2 \quad (69)$$

then $\sigma_p \leq \epsilon$, where σ_p is the smallest singular value of X .

3.2 Ridge regression

As described in the previous section, if the feature matrix is ill conditioned, then small shifts in the data produce large changes in the least-squares solution. In particular, some of the coefficients may blow up due to noise amplification. In order to avoid this, we can add a term penalizing the norm of the coefficient vector to the least-squares cost function. The aim is to promote solutions that yield a good fit with small coefficients. Incorporating prior assumptions on the desired solution— in this case that the coefficients should not be too large— is called *regularization*. Least-squares regression combined with ℓ_2 -norm regularization is called ridge regression in statistics and Tikhonov regularization in the inverse-problems literature.

Definition 3.5 (Ridge regression / Tikhonov regularization). *For any $X \in \mathbb{R}^{n \times p}$ and $\vec{y} \in \mathbb{R}^p$ the ridge-regression estimate is the minimizer of the optimization problem*

$$\vec{\beta}_{\text{ridge}} := \arg \min_{\vec{\beta}} \left\| \vec{y} - X \vec{\beta} \right\|_2^2 + \lambda \left\| \vec{\beta} \right\|_2^2, \quad (70)$$

where $\lambda > 0$ is a fixed regularization parameter.

As in the case of least-squares regression, the ridge-regression estimate has a closed form solution.

Theorem 3.6 (Ridge-regression estimate). *For any $X \in R^{n \times p}$ and $\vec{y} \in \mathbb{R}^n$ we have*

$$\vec{\beta}_{\text{ridge}} := (X^T X + \lambda I)^{-1} X^T \vec{y}. \quad (71)$$

Proof. The ridge-regression estimate is the solution to a modified least-squares problem

$$\vec{\beta}_{\text{ridge}} = \arg \min_{\vec{\beta}} \left\| \begin{bmatrix} \vec{y} \\ 0 \end{bmatrix} - \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} \vec{\beta} \right\|_2^2. \quad (72)$$

By Theorem 2.1 the solution equals

$$\vec{\beta}_{\text{ridge}} := \left(\begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} \right)^{-1} \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} \vec{y} \\ 0 \end{bmatrix} \quad (73)$$

$$= (X^T X + \lambda I)^{-1} X^T \vec{y}. \quad (74)$$

□

When $\lambda \rightarrow 0$ then $\vec{\beta}_{\text{ridge}}$ converges to the least-squares estimator. When $\lambda \rightarrow \infty$, it converges to zero.

The approximation $X \vec{\beta}_{\text{ridge}}$ corresponding to the ridge-regression estimate is no longer the orthogonal projection of the data onto the column space of the feature matrix. It is a modified projection where the component of the data in the direction of each left singular vector of the feature matrix is shrunk by a factor of $\sigma_i^2 / (\sigma_i^2 + \lambda)$ where σ_i is the corresponding singular value. Intuitively, this reduces the influence of the directions corresponding to the smaller singular values which are the ones responsible for more noise amplification.

Corollary 3.7 (Modified projection). *For any $X \in R^{n \times p}$ and $\vec{y} \in \mathbb{R}^n$ we have*

$$\vec{y}_{\text{ridge}} := X \vec{\beta}_{\text{ridge}} \quad (75)$$

$$= \sum_{i=1}^p \frac{\sigma_i^2}{\sigma_i^2 + \lambda} \langle \vec{y}, \vec{u}_i \rangle \vec{u}_i, \quad (76)$$

where $\vec{u}_1, \dots, \vec{u}_p$ are the left singular vectors of X and $\sigma_1 \geq \dots \geq \sigma_p$ the corresponding singular values.

Proof. Let $X = USV^T$ be the SVD of X . By the theorem,

$$X \vec{\beta}_{\text{ridge}} := X (X^T X + \lambda I)^{-1} X^T \vec{y} \quad (77)$$

$$= USV^T (VS^2 V^T + \lambda VV^T)^{-1} VSU^T \vec{y} \quad (78)$$

$$= USV^T V (S^2 + \lambda I)^{-1} V^T VSU^T \vec{y} \quad (79)$$

$$= US (S^2 + \lambda I)^{-1} SU^T \vec{y}, \quad (80)$$

since V is an orthogonal matrix. □

The following theorem shows that, under the assumption that the data indeed follow a linear model, the ridge-regression estimator can be decomposed into a term that depends on the signal and a term that depends on the noise.

Theorem 3.8 (Ridge-regression estimate). *If $\vec{y} := X\vec{\beta}^* + \vec{z}$, where $X \in \mathbb{R}^{n \times p}$, $\vec{z} \in \mathbb{R}^n$ and $\vec{\beta}^* \in \mathbb{R}^p$, then the solution of Problem (70) is equal to*

$$\vec{\beta}_{\text{ridge}} = V \begin{bmatrix} \frac{\sigma_1^2}{\sigma_1^2 + \lambda} & 0 & \cdots & 0 \\ 0 & \frac{\sigma_2^2}{\sigma_2^2 + \lambda} & \cdots & 0 \\ \vdots & & & \\ 0 & 0 & \cdots & \frac{\sigma_p^2}{\sigma_p^2 + \lambda} \end{bmatrix} V^T \vec{\beta}^* + V \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2 + \lambda} & 0 & \cdots & 0 \\ 0 & \frac{\sigma_2}{\sigma_2^2 + \lambda} & \cdots & 0 \\ \vdots & & & \\ 0 & 0 & \cdots & \frac{\sigma_p}{\sigma_p^2 + \lambda} \end{bmatrix} U^T \vec{z}, \quad (81)$$

where $X = USV^T$ is the SVD of X and $\sigma_1, \dots, \sigma_p$ are the singular values.

Proof. By Theorem 2.1 the solution equals

$$\vec{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T (X \vec{\beta}^* + \vec{z}) \quad (82)$$

$$= (VS^2V^T + \lambda VV^T)^{-1} (VS^2V^T \vec{\beta}^* + VSU^T \vec{z}) \quad (83)$$

$$= V(S^2 + \lambda I)^{-1} V^T (VS^2V^T \vec{\beta}^* + VSU^T \vec{z}) \quad (84)$$

$$= V(S^2 + \lambda I)^{-1} S^2 V^T \vec{\beta}^* + V(S^2 + \lambda I)^{-1} SU^T \vec{z}, \quad (85)$$

because V is an orthogonal matrix. \square

If we consider the difference between the true coefficients $\vec{\beta}^*$ and the ridge-regression estimator, the term that depends on $\vec{\beta}^*$ is usually known as the *bias* of the estimate, whereas the term that depends on the noise is the *variance*. The reason is that if we model the noise as being random and zero mean, then the mean or bias of the ridge-regression estimator equals the first term and the variance is equal to the variance of the second term.

Corollary 3.9 (Bias of ridge-regression estimator). *If the noise vector \vec{z} is random and zero mean,*

$$E(\vec{\beta}_{\text{ridge}} - \vec{\beta}^*) = V \begin{bmatrix} \frac{\lambda}{\sigma_1^2 + \lambda} & 0 & \cdots & 0 \\ 0 & \frac{\lambda}{\sigma_2^2 + \lambda} & \cdots & 0 \\ \vdots & & & \\ 0 & 0 & \cdots & \frac{\lambda}{\sigma_p^2 + \lambda} \end{bmatrix} V^T \vec{\beta}^*. \quad (86)$$

Proof. The result follows from the lemma and linearity of expectation. \square

Increasing λ increases the bias, moving the mean of the estimator farther from the true value of the coefficients, but in exchange dampens the noise component. In statistics jargon, we introduce bias in order to reduce the variance of the estimator. Calibrating the regularization parameter allows us to adapt to the conditioning of the predictor matrix and the noise level in order to achieve a good tradeoff between both terms.

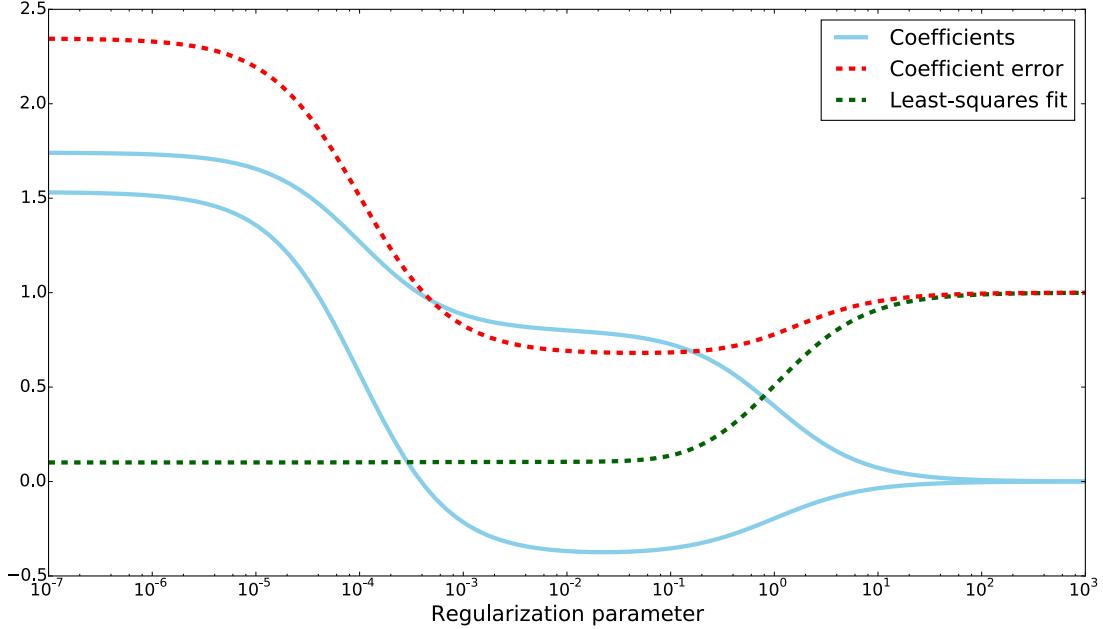


Figure 6: Coefficients in the ridge-regression model (blue) for different values of the regularization parameter λ (horizontal axis). The fit to the data improves as we reduce λ (green). The relative error of the coefficient estimate $\|\vec{\beta}^* - \vec{\beta}_{\text{ridge}}\|_2 / \|\vec{\beta}^*\|_2$ is equal to one when λ is large (because $\vec{\beta}_{\text{ridge}} = 0$), then it decreases as λ is reduced and finally it blows up due to noise amplification (red).

Example 3.10 (Noise amplification (continued)). By Theorem 3.8, the ridge-regression estimator for the regression problem in Example 3.3 equals

$$\vec{\beta}_{\text{ridge}} - \vec{\beta}^* = V \begin{bmatrix} \frac{\lambda}{1+\lambda} & 0 \\ 0 & \frac{\lambda}{0.01^2+\lambda} \end{bmatrix} V^T \vec{\beta}^* - V \begin{bmatrix} \frac{1}{1+\lambda} & 0 \\ 0 & \frac{0.01}{0.01^2+\lambda} \end{bmatrix} U^T \vec{z}, \quad (87)$$

The regularization λ should be set so to achieve a good balance between the two terms in the error. Setting $\lambda = 0.01$

$$\vec{\beta}_{\text{ridge}} - \vec{\beta}^* = -V \begin{bmatrix} 0.001 & 0 \\ 0 & 0.99 \end{bmatrix} V^T \vec{\beta}^* + V \begin{bmatrix} 0.99 & 0 \\ 0 & 0.99 \end{bmatrix} U^T \vec{z} \quad (88)$$

$$= \begin{bmatrix} 0.329 \\ 0.823 \end{bmatrix}. \quad (89)$$

The error is reduced significantly with respect to the least-squares estimate, we have

$$\frac{\|\vec{\beta}_{\text{ridge}} - \vec{\beta}^*\|_2}{\|\vec{z}\|_2} = 7.96. \quad (90)$$

Figure 6 shows the values of the coefficients for different values of the regularization parameter. They vary wildly due to the ill conditioning of the problem. The figure shows how least squares

(to the left where $\lambda \rightarrow 0$) achieves the best fit to the data, but this does not result in a smaller error in the coefficient vector. $\lambda = 0.01$ achieves a good compromise. At that point the coefficients are smaller, while yielding a similar fit to the data as least squares. \triangle

3.3 Ridge regression as maximum-a-posteriori estimation

From a probabilistic point of view, we can view the ridge-regression estimate as a maximum-a-posteriori (MAP) estimate. In Bayesian statistics, the MAP estimate is the mode of the posterior distribution of the parameter that we aim to estimate given the observed data.

Definition 3.11 (Maximum-a-posteriori estimator). *The maximum-a-posteriori (MAP) estimator of a random vector of parameters $\vec{\beta} \in \mathbb{R}^m$ given a realization of the data vector \vec{y} is*

$$\vec{\beta}_{\text{MAP}}(\vec{y}) := \arg \max_{\vec{\beta}} f_{\vec{\beta}|\vec{y}}(\vec{\beta} | \vec{y}), \quad (91)$$

where $f_{\vec{\beta}|\vec{y}}$ is the conditional pdf of the parameter $\vec{\beta}$ given the data \vec{y} .

In contrast to ML estimation, the parameters of interest (in our case the regression coefficients) are modeled as random variables, not as deterministic quantities. This allows us to incorporate prior assumptions about them through their marginal distribution. Ridge regression is equivalent to modeling the distribution of the coefficients as an iid Gaussian random vector.

Lemma 3.12 (Proof in Section 5.3). *Let $\vec{y} \in \mathbb{R}^n$ be a realization of a random vector*

$$\vec{y} := X\vec{\beta} + \vec{z}, \quad (92)$$

where $\vec{\beta}$ and \vec{z} are iid Gaussian random vectors with mean zero and variance σ_1^2 and σ_2^2 , respectively. If $X \in \mathbb{R}^{n \times m}$ is known, then the MAP estimate of $\vec{\beta}$ is equal to the ridge-regression estimate

$$\vec{\beta}_{\text{MAP}} = \arg \min_{\vec{\beta}} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 + \lambda \left\| \vec{\beta} \right\|_2^2, \quad (93)$$

where $\lambda := \sigma_2^2/\sigma_1^2$.

3.4 Cross validation

An important issue when applying ridge regression, and also other forms of regularization, is how to calibrate the regularization parameter λ . With real data, we do not know the true value of the coefficients as in Example 3.3 (otherwise we wouldn't need to do regression in the first place!). In addition, we cannot rely on how well the model fits the data, since this will always occur for $\lambda = 0$, which can lead to overfitting and noise amplification. However, we can evaluate the fit achieved by the model on *new* data, different from the ones used to estimate the regression coefficients. If the fit is accurate, this is a strong indication that the model is not overfitting the noise. Calibrating the regularization parameter using a different set of data is known as cross validation.

Algorithm 3.13 (Cross validation). *Given a set of examples*

$$(y^{(1)}, \vec{x}^{(1)}), (y^{(2)}, \vec{x}^{(2)}), \dots, (y^{(n)}, \vec{x}^{(n)}), \quad (94)$$

which are centered and normalized, to determine the best value for λ we:

1. *Partition the data into a training set $X_{\text{train}} \in \mathbb{R}^{n_{\text{train}} \times p}$, $\vec{y}_{\text{train}} \in \mathbb{R}^{n_{\text{train}}}$ and a validation set $X_{\text{val}} \in \mathbb{R}^{n_{\text{val}} \times p}$, $\vec{y}_{\text{val}} \in \mathbb{R}^{n_{\text{val}}}$, such that $n_{\text{train}} + n_{\text{val}} = n$.*
2. *Fit the model using the training set for every λ in a set Λ (usually a logarithmic grid of values)*

$$\vec{\beta}_{\text{ridge}}(\lambda) := \arg \min_{\vec{\beta}} \left\| \vec{y}_{\text{train}} - X_{\text{train}} \vec{\beta} \right\|_2^2 + \lambda \left\| \vec{\beta} \right\|_2^2 \quad (95)$$

and evaluate the fitting error on the validation set

$$\text{err}(\lambda) := \left\| \vec{y}_{\text{train}} - X_{\text{train}} \vec{\beta}_{\text{ridge}}(\lambda) \right\|_2^2. \quad (96)$$

3. *Choose the value of λ that minimizes the validation-set error*

$$\lambda_{\text{cv}} := \arg \min_{\lambda \in \Lambda} \text{err}(\lambda). \quad (97)$$

In practice, more sophisticated cross-validation procedures are applied to make an efficient use of the data. For example, in *k-fold* cross validation we randomly partition the data into k sets of equal size. Then we evaluate the fitting error k times, each time using one of the k sets as the validation set and the rest as the training set.

Finally, it is important to note that if we have used the validation set to fit the regularization parameter, we *cannot* use it to evaluate our results. This wouldn't be fair, since we have calibrated one the parameter to do well precisely on those data! It is crucial to evaluate the model on a test set that is completely different from both the training and validation tests.

Example 3.14 (Prediction of house prices). In this example we consider the problem of predicting the price of a house². The features that we consider are:

1. Area of the living room.
2. Condition (an integer between 1 and 5 evaluating the state of the house).
3. Grade (an integer between 7 and 12 evaluating the house).
4. Area of the house without the basement.
5. Area of the basement.
6. The year it was built.
7. Latitude.
8. Longitude.

²The data are available at <http://www.kaggle.com/harlfoxem/housesalesprediction>

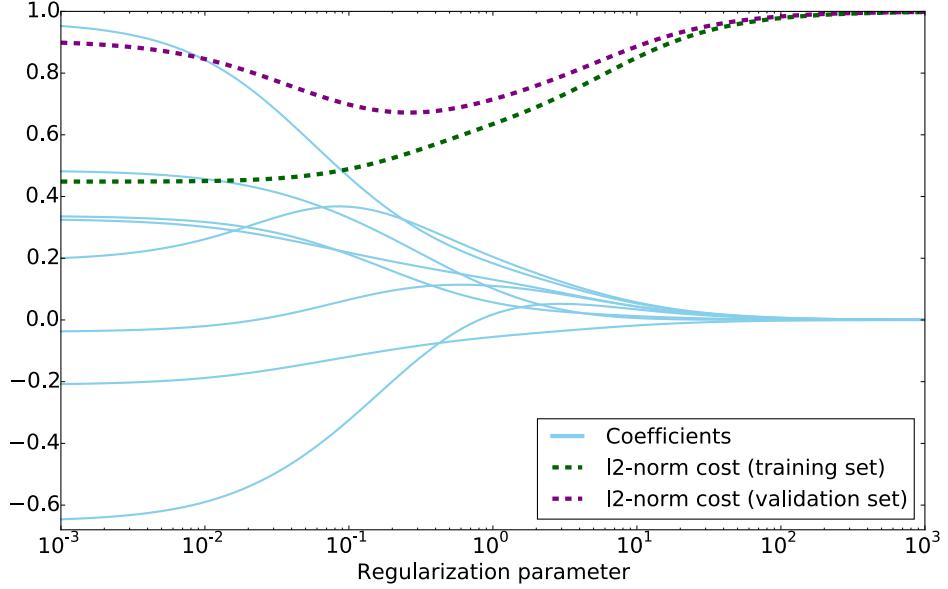


Figure 7: Coefficients in the ridge-regression model (blue) for different values of the regularization parameter λ (horizontal axis). The relative ℓ_2 -norm error evaluated on the training data is shown in green. The relative ℓ_2 -norm error evaluated on the validation data is shown in purple.

9. Average area of the living room of the houses within 15 blocks.

We use 15 houses to train the data, a validation set of 15 houses to calibrate the regularization parameter of the ridge regression model and a test set of 15 houses to evaluate the results. The feature matrix has significant correlations (the condition number is equal to 9.94), so we decide to apply ridge regression. Figure 7 shows the value of the coefficients obtained by fitting the model to the training set for different values of λ . It also shows the corresponding relative ℓ_2 -norm fit

$$\frac{\|\vec{y} - X\vec{\beta}_{\text{ridge}}\|_2}{\|\vec{y}\|_2} \quad (98)$$

to the training and validation sets. For small λ the model fits the training set much better than the validation set, a clear indication that it is overfitting. The validation-set error is minimized for $\lambda = 0.27$. For that value the error is 0.672 on the validation set and 0.799 on the test set. In contrast, the error of the least-squares estimator is 0.906 on the validation set and 1.186 on the test set. Figure 8 shows the prices estimated by the least-squares and the ridge-regression models plotted against the true prices. The least-squares estimate is much more accurate on the training set than on the validation and test sets due to overfitting. Adding regularization and computing a ridge-regression estimate substantially improves the prediction results on the test set. \triangle

4 Classification

In this section, we consider the problem of classification. The goal is to learn a model that assigns one of several predefined categories to a set of examples, represented by the values of certain

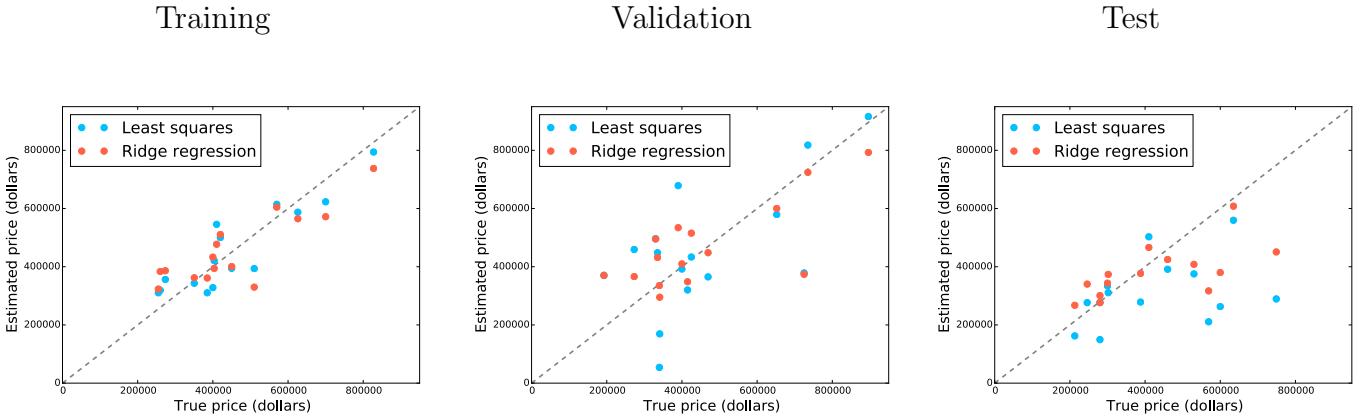


Figure 8: Prices estimated by the least-squares (blue) and the ridge-regression (orange) models plotted against the true prices for the training, validation and test sets.

features, as in the case of regression. To be more precise, we have available n examples of category labels and their corresponding features

$$(y^{(1)}, \vec{x}^{(1)}) , (y^{(2)}, \vec{x}^{(2)}) , \dots , (y^{(n)}, \vec{x}^{(n)}) . \quad (99)$$

The label $y^{(i)}$ indicates what category example i belongs to. Here, we consider the simple case where there are only two categories and set the labels to equal either 0 or 1. Our aim is to predict the label $y^{(i)} \in \{0, 1\}$ from p real-valued features $\vec{x}^{(i)} \in \mathbb{R}^p$. This is a regression problem, where the response is binary.

4.1 Perceptron

Inspired by linear regression, let us consider how to use a linear model to perform classification. A reasonable idea is to fit a vector of coefficients $\vec{\beta}$ such that the label is predicted to equal 1 if $\langle \vec{x}^{(i)}, \vec{\beta} \rangle$ is larger than a certain quantity, and 0 if it is smaller. This requires finding $\vec{\beta} \in \mathbb{R}^p$ and β_0 such that

$$y^{(i)} = \begin{cases} 1 & \text{if } \beta_0 + \langle \vec{x}^{(i)}, \vec{\beta} \rangle > 0 \\ 0 & \text{otherwise} \end{cases} \quad (100)$$

for as many $1 \leq i \leq n$ as possible. This method is called the *perceptron* algorithm. The model is fit by considering each feature vector sequentially and updating $\vec{\beta}$ if the current classification is wrong. This method is guaranteed to converge if the data are *linearly separable*, i.e. if there is a hyperplane in the p -dimensional feature space \mathbb{R}^p separating the two classes. However, if this is not the case, then the method becomes unstable.

4.2 Logistic regression

Logistic regression is an example of a *generalized linear model*. Generalized linear models extend the linear regression paradigm by incorporating a *link function* that performs an entry-wise non-

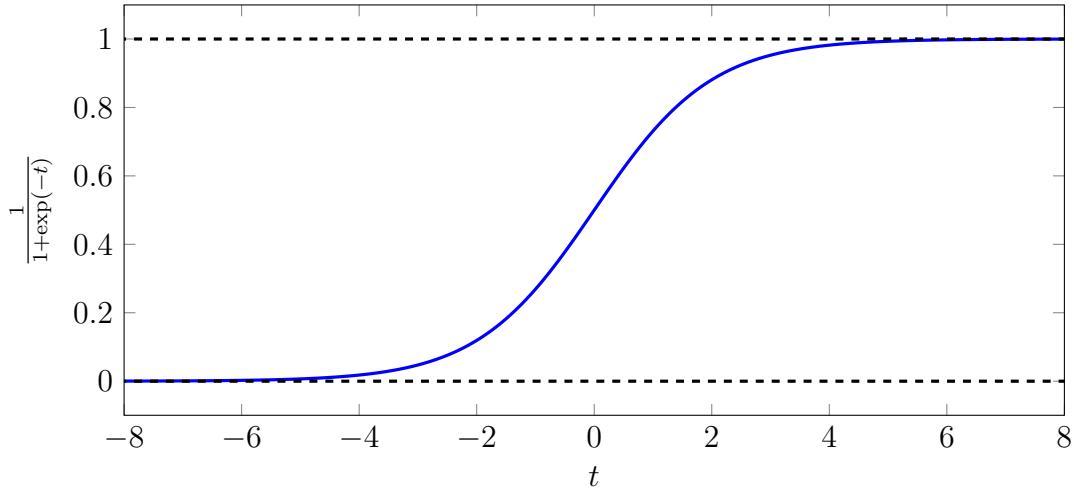


Figure 9: The logistic function used as a link function in logistic regression.

linear transformation of the output of a linear model. In the case of logistic regression, this link function is the *logistic function*

$$g(t) := \frac{1}{1 + \exp(-t)}, \quad (101)$$

depicted in Figure 9. The output of g is always between 0 and 1. We can interpret the function as a smoothed version of the step function used by the perceptron algorithm, as it maps large values to 1 and small values to 0.

The logistic-regression model is of the form

$$y^{(i)} \approx g\left(\beta_0 + \langle \vec{x}^{(i)}, \vec{\beta} \rangle\right). \quad (102)$$

To simplify notation, from now on we assume that one of the feature vectors is equal to a constant, so that β_0 is included in $\vec{\beta}$. The logistic-regression estimator is obtained by calibrating $\vec{\beta}$ in order to optimize the fit to the training data. This can be achieved by maximizing the log-likelihood function derived in the following theorem.

Theorem 4.1 (Logistic-regression cost function). *Assume that $y^{(1)}, \dots, y^{(n)}$ are independent samples from Bernoulli random variables with parameter*

$$p_{\mathbf{y}^{(i)}}(1) := g\left(\langle \vec{x}^{(i)}, \vec{\beta} \rangle\right), \quad (103)$$

where the vectors $\vec{x}^{(1)}, \dots, \vec{x}^{(n)} \in \mathbb{R}^p$ are known. The maximum-likelihood estimate of $\vec{\beta}$ given $y^{(1)}, \dots, y^{(n)}$ is equal to

$$\vec{\beta}_{\text{ML}} := \sum_{i=1}^n y^{(i)} \log g\left(\langle \vec{x}^{(i)}, \vec{\beta} \rangle\right) + (1 - y^{(i)}) \log \left(1 - g\left(\langle \vec{x}^{(i)}, \vec{\beta} \rangle\right)\right). \quad (104)$$

Proof. The likelihood $\mathcal{L}(\vec{\beta})$ is defined as the joint pmf of the random variables $\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(n)}$ interpreted as a function of the coefficient vector. Due to the independence assumption,

$$\mathcal{L}(\vec{\beta}) := p_{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(n)}}(y^{(1)}, \dots, y^{(n)}) \quad (105)$$

$$= \prod_{i=1}^n g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)^{y^{(i)}} (1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle))^{1-y^{(i)}}. \quad (106)$$

Maximizing this nonnegative function is the same as maximizing its logarithm, so the proof is complete. \square

Even though it is quite implausible that the probabilistic assumptions assumed in this theorem actually hold in practice, the corresponding log-likelihood function is very useful. It penalizes classification errors in a smooth way and is easy to optimize (as we will see later on).

Definition 4.2 (Logistic-regression estimator). *Given a set of examples $(y^{(1)}, \vec{x}^{(1)}), (y^{(2)}, \vec{x}^{(2)}), \dots, (y^{(n)}, \vec{x}^{(n)})$, we define the logistic-regression coefficient vector as*

$$\vec{\beta}_{LR} := \sum_{i=1}^n y^{(i)} \log g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle) + (1 - y^{(i)}) \log (1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)), \quad (107)$$

where we assume that one of the features is always equal to one, so we don't have to fit an intercept. For a new feature vector \vec{x} the logistic-regression prediction is

$$y_{LR} := \begin{cases} 1 & \text{if } g(\langle \vec{x}, \vec{\beta}_{LR} \rangle) \geq 0, \\ 0 & \text{otherwise.} \end{cases} \quad (108)$$

The value $g(\langle \vec{x}, \vec{\beta}_{LR} \rangle)$ can be interpreted as the probability under the model that the label of the example equals 1.

Example 4.3 (Flower classification). The [Iris data set](#) was compiled by the statistician Ronald Fisher in 1936. It contains examples of three species of flowers, together with measurements of the length and width of their sepal and petal. In this example, we consider the problem of distinguishing between two of the species using only the sepal lengths and widths.

We assume that we just have access to 5 examples of *Iris setosa* (label 0) with sepal lengths 5.4, 4.3, 4.8, 5.1 and 5.7, and sepal widths 3.7, 3, 3.1, 3.8 and 3.8, and to 5 examples of *Iris versicolor* (label 1) with sepal lengths 6.5, 5.7, 7, 6.3 and 6.1, and sepal widths 2.8, 2.8, 3.2, 2.3 and 2.8. We want to classify two new examples: one has a sepal length of 5.1 and width 3.5, the other has length 5 and width 2. $\beta_0 = 2.06$. After centering and normalizing the data set (note that we ignore the labels to center and normalize), we fit a logistic regression model, where the coefficient vector equals

$$\vec{\beta}_{LR} = \begin{bmatrix} 32.1 \\ -29.6 \end{bmatrix} \quad (109)$$

and the intercept β_0 equals 2.06. The coefficients suggest that *versicolor* has larger sepal length than *setosa*, but smaller sepal width. The following table shows the values of the features, their inner product with $\vec{\beta}_{LR}$ and the output of the logistic function.

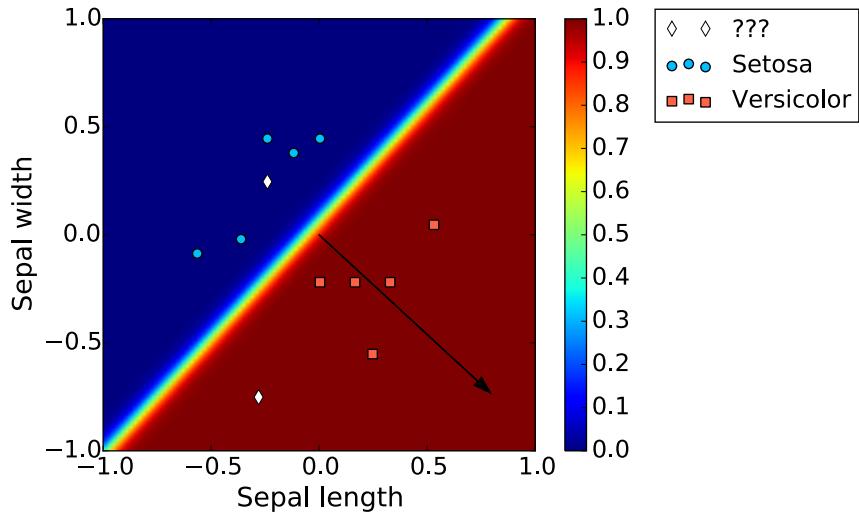


Figure 10: The data used in Example 4.3 is plotted in different colors depending on the corresponding flower species. The direction of $\vec{\beta}_{LR}$ is shown as a black arrow. The heat map corresponds to the value of $g(\langle \vec{x}, \vec{\beta}_{LR} \rangle + \beta_0)$ at every point. The two new examples are depicted as white diamonds.

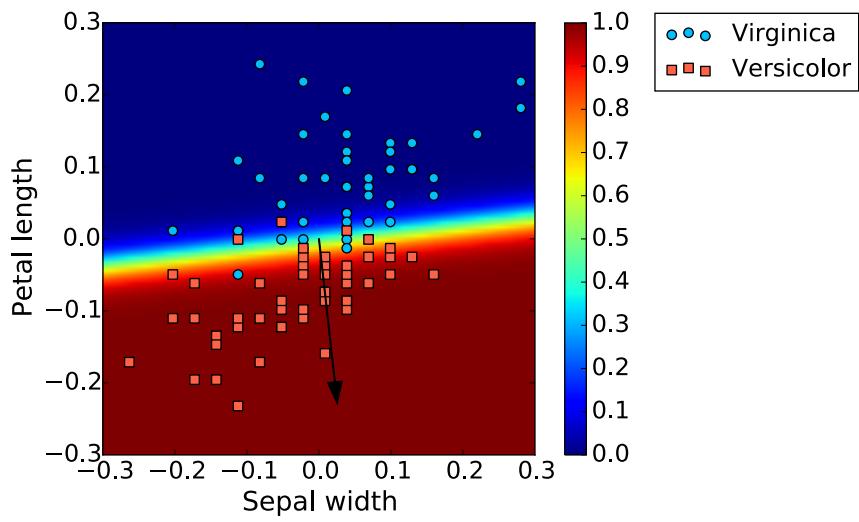


Figure 11: The data from the Iris data set plotted in different colors depending on the corresponding flower species. The direction of $\vec{\beta}_{LR}$ is shown as a black arrow. The heat map corresponds to the value of $g(\langle \vec{x}, \vec{\beta}_{LR} \rangle + \beta_0)$ at every point.

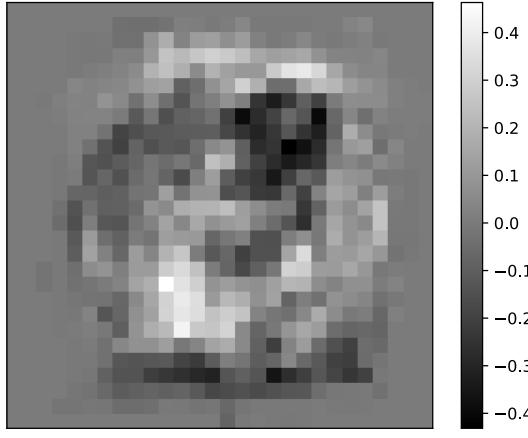


Figure 12: The coefficient vector $\vec{\beta}_{\text{LR}}$ obtained by fitting a logistic-regression model to distinguish between 6 and 9. The vector is reshaped so that each coefficient is shown at the position of the corresponding pixel.

i	1	2	3	4	5	6	7	8	9	10
$\vec{x}^{(i)}[1]$	-0.12	-0.56	-0.36	-0.24	0.00	0.33	0.00	0.53	0.25	0.17
$\vec{x}^{(i)}[2]$	0.38	-0.09	-0.02	0.45	0.45	-0.22	-0.22	0.05	-0.05	-0.22
$\langle \vec{x}^{(i)}, \vec{\beta}_{\text{LR}} \rangle + \beta_0$	-12.9	-13.5	-8.9	-18.8	-11.0	19.1	8.7	17.7	26.3	13.9
$g(\langle \vec{x}^{(i)}, \vec{\beta}_{\text{LR}} \rangle + \beta_0)$	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00

Figure 10 shows the data, which are linearly separable, the direction of $\vec{\beta}_{\text{LR}}$ (black arrow) and a heat map of values for $g(\langle \vec{x}, \vec{\beta}_{\text{LR}} \rangle)$ which shows are assigned to what category and with how much certainty. The two new examples are depicted as white diamonds, the first is assigned to *setosa* and the second to *versicolor* with almost total certainty. Both decisions are correct.

Figure 11 shows the result of trying to classify between *Iris virginica* and *Iris versicolor* based on petal length and sepal width. In this case the data is not linearly separable, but the logistic-regression model still partitions the space in a way that approximately separates the two classes. The value of the likelihood $g(\langle \vec{x}, \vec{\beta}_{\text{LR}} \rangle)$ allows us to quantify the certainty with which the model classifies each example. Note that the examples that are misclassified are assigned low values. \triangle

Example 4.4 (Digit classification). In this example we use the MNIST data set³ to illustrate image classification. We consider the task of distinguishing a digit from another. The feature vector \vec{x}_i contains the pixel values of an image of a 6 ($\vec{y}_i = 1$) or a 9 ($\vec{y}_i = 0$). We use 2000 training examples to fit a logistic regression model. The coefficient vector is shown in Figure 12, the intercept is equal to 0.053. The model manages to fit the training set perfectly. When tested on 2000 new examples, it achieves a test error rate of 0.006. Figure 13 shows some test examples and the corresponding probabilities assigned by the model. \triangle

³ Available at <http://yann.lecun.com/exdb/mnist/>

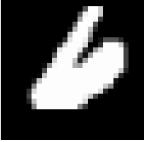
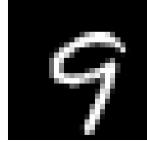
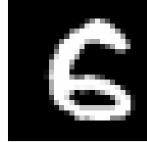
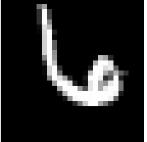
\vec{x}	$\vec{\beta}^T \vec{x}$	$g(\vec{\beta}^T \vec{x} + \beta_0)$	Pred.	True label	\vec{x}	$\vec{\beta}^T \vec{x}$	$g(\vec{\beta}^T \vec{x} + \beta_0)$	Pred.	True label
	20.88	1.00	6	6		18.22	1.00	6	6
	16.41	1.00	6	6		-14.71	0.00	9	9
	-15.83	0.00	9	9		-17.02	0.00	9	9
	7.612	0.9995	6	9		0.434	0.606	6	9
	7.822	0.9996	6	9		-5.984	0.0025	9	6
	-2.384	0.084	9	6		-1.164	0.238	9	6

Figure 13: Examples of digits in the MNIST data set along with the value of $\vec{\beta}^T \vec{x} + \beta_0$ and the probability assigned by the model.

5 Proofs

5.1 Proof of Lemma 2.2

To ease notation let $\tilde{X} := X^{\text{cent}}$ and $\tilde{x} := X^T \vec{1}$. Note that

$$\vec{y}^{\text{cent}} = \vec{y} - \frac{1}{n} \vec{1} \vec{1}^T \vec{y}, \quad (110)$$

$$\tilde{X} = X - \frac{1}{n} \vec{1} \tilde{x}^T. \quad (111)$$

By Theorem 2.1

$$\begin{bmatrix} \vec{\beta}_{\text{LS}} \\ \beta_{\text{LS},0} \end{bmatrix} = \left(\begin{bmatrix} X & \vec{1} \end{bmatrix}^T \begin{bmatrix} X & \vec{1} \end{bmatrix} \right)^{-1} \begin{bmatrix} X & \vec{1} \end{bmatrix}^T \vec{y} \quad (112)$$

$$= \begin{bmatrix} X^T X & \tilde{x} \\ \tilde{x}^T & n \end{bmatrix}^{-1} \begin{bmatrix} X^T \vec{y} \\ \vec{1}^T \vec{y} \end{bmatrix}. \quad (113)$$

We now apply the following lemma.

Lemma 5.1. *For any matrices $A \in \mathbb{R}^{m \times n}$, let*

$$B = A - \frac{1}{n} \tilde{x} \tilde{x}^T \quad (114)$$

be invertible, then

$$\begin{bmatrix} A & \tilde{x} \\ \tilde{x}^T & n \end{bmatrix}^{-1} = \begin{bmatrix} B^{-1} & -\frac{1}{n} B^{-1} \tilde{x} \\ -\frac{1}{n} \tilde{x}^T B^{-1} & \frac{1}{n} + \frac{1}{n^2} \tilde{x}^T B^{-1} \tilde{x} \end{bmatrix} \quad (115)$$

Proof. One can check the result by multiplying the two matrices and verifying that the product is the identity. \square

Setting $A := X^T X$, we have

$$B = X^T X - \frac{1}{n} \tilde{x} \tilde{x}^T \quad (116)$$

$$= \left(X - \frac{1}{n} \vec{1} \tilde{x}^T \right)^T \left(X - \frac{1}{n} \vec{1} \tilde{x}^T \right) \quad (117)$$

$$= \tilde{X}^T \tilde{X}. \quad (118)$$

As a result, by the lemma

$$\begin{bmatrix} \vec{\beta}_{\text{LS}} \\ \beta_{\text{LS},0} \end{bmatrix} = \begin{bmatrix} \left(\tilde{X}^T \tilde{X} \right)^{-1} & -\frac{1}{n} \left(\tilde{X}^T \tilde{X} \right)^{-1} \tilde{x} \\ -\frac{1}{n} \tilde{x}^T \left(\tilde{X}^T \tilde{X} \right)^{-1} & \frac{1}{n} + \frac{1}{n^2} \tilde{x}^T \left(\tilde{X}^T \tilde{X} \right)^{-1} \tilde{x} \end{bmatrix} \begin{bmatrix} X^T \vec{y} \\ \vec{1}^T \vec{y} \end{bmatrix} \quad (119)$$

$$= \begin{bmatrix} \left(\tilde{X}^T \tilde{X} \right)^{-1} X^T \left(\vec{y} - \frac{1}{n} \vec{1} \vec{1}^T \vec{y} \right) \\ -\frac{1}{n} \tilde{x}^T \left(\tilde{X}^T \tilde{X} \right)^{-1} X^T \left(\vec{y} - \frac{1}{n} \vec{1} \vec{1}^T \vec{y} \right) + \frac{\vec{1}^T \vec{y}}{n} \end{bmatrix}, \quad (120)$$

which implies

$$X\vec{\beta}_{\text{LS}} + \beta_{\text{LS},0}\vec{1} = X \left(\tilde{X}^T \tilde{X} \right)^{-1} X^T \vec{y}^{\text{cent}} - \frac{1}{n} \vec{1} \tilde{X}^T \left(\tilde{X}^T \tilde{X} \right)^{-1} X^T \vec{y}^{\text{cent}} + \text{av}(\vec{y}) \vec{1} \quad (121)$$

$$= \tilde{X} \left(\tilde{X}^T \tilde{X} \right)^{-1} X^T \vec{y}^{\text{cent}} + \text{av}(\vec{y}) \vec{1} \quad (122)$$

$$= \tilde{X} \left(\tilde{X}^T \tilde{X} \right)^{-1} \tilde{X}^T \vec{y}^{\text{cent}} + \text{av}(\vec{y}) \vec{1}, \quad (123)$$

where the last inequality follows from

$$\tilde{X}^T \vec{y}^{\text{cent}} = \left(X - \frac{1}{n} \vec{1} \vec{1}^T X \right)^T \left(\vec{y} - \frac{1}{n} \vec{1} \vec{1}^T \vec{y} \right) \quad (124)$$

$$= X^T \vec{y} - \frac{1}{n} X^T \vec{1} \vec{1}^T \vec{y} - \frac{1}{n} X^T \vec{1} \vec{1}^T \vec{y} + \frac{1}{n^2} X^T \vec{1} \vec{1}^T \vec{1} \vec{1}^T \vec{y} \quad (125)$$

$$= X^T \vec{y} - \frac{1}{n} X^T \vec{1} \vec{1}^T \vec{y} \quad (126)$$

$$= X^T \vec{y}^{\text{cent}}. \quad (127)$$

Since $\vec{\beta}_{\text{LS}}^{\text{cent}} = \left(\tilde{X}^T \tilde{X} \right)^{-1} \tilde{X}^T \vec{y}^{\text{cent}}$ the proof is complete.

5.2 Proof of Lemma 3.4

The orthogonal projection of X_i onto the span of X_j equals

$$\mathcal{P}_{\text{span}(X_j)} X_i = \langle X_i, X_j \rangle X_j \quad (128)$$

so

$$\| \mathcal{P}_{\text{span}(X_j)} X_i \|_2^2 = \langle X_i, X_j \rangle^2 \| X_j \|_2^2 = 1 - \epsilon^2 \quad (129)$$

and

$$\| \mathcal{P}_{\text{span}(X_j)^\perp} X_i \|_2^2 = \| X_i \|_2^2 - \| \mathcal{P}_{\text{span}(X_j)} X_i \|_2^2 = \epsilon^2. \quad (130)$$

Consider the unit norm vector $\vec{w} \in \mathbb{R}^p$

$$\vec{w}[l] := \begin{cases} \frac{1}{\sqrt{2}} & \text{if } l = i \\ -\frac{1}{\sqrt{2}} & \text{if } l = j \\ 0 & \text{otherwise.} \end{cases} \quad (131)$$

We have

$$\|X\vec{w}\|_2^2 = \frac{1}{2} \|X_i - X_j\|_2^2 \quad (132)$$

$$= \frac{1}{2} \left\| \mathcal{P}_{\text{span}(X_j)} X_i + \mathcal{P}_{\text{span}(X_j)^\perp} X_i - X_j \right\|_2^2 \quad (133)$$

$$= \frac{1}{2} \left\| \mathcal{P}_{\text{span}(X_j)} X_i - X_j \right\|_2^2 + \frac{1}{2} \left\| \mathcal{P}_{\text{span}(X_j)^\perp} X_i \right\|_2^2 \quad (134)$$

$$= \frac{1}{2} \left\| \langle X_i, X_j \rangle X_j - X_j \right\|_2^2 + \frac{\epsilon^2}{2} \quad (135)$$

$$= \frac{\langle X_i, X_j \rangle^2}{2} \|X_j\|_2^2 + \frac{\epsilon^2}{2} \quad (136)$$

$$= \epsilon^2. \quad (137)$$

Finally by Theorem 2.7 in Lecture Notes 2

$$\sigma_p = \min_{\|v\|_2=1} \|X\vec{v}\|_2 \geq \|X\vec{w}\|_2 = \epsilon. \quad (138)$$

5.3 Proof of Lemma 3.12

By Bayes' rule, the posterior pdf of \vec{x} given \vec{y} is equal to

$$f_{\vec{\beta}|\vec{y}}(\vec{\beta}|\vec{y}) = \frac{f_{\vec{\beta},\vec{y}}(\vec{\beta},\vec{y})}{f_{\vec{y}}(\vec{y})} \quad (139)$$

so for fixed \vec{y}

$$\arg \max_{\vec{\beta}} f_{\vec{\beta}|\vec{y}}(\vec{\beta}|\vec{y}) = \arg \max_{\vec{\beta}} f_{\vec{\beta},\vec{y}}(\vec{\beta},\vec{y}) \quad (140)$$

$$= \arg \max_{\vec{\beta}} f_{\vec{\beta}}(\vec{\beta}) f_{\vec{y}|\vec{\beta}}(\vec{y}|\vec{\beta}). \quad (141)$$

Since all the quantities are nonnegative, we can take logarithms

$$\arg \max_{\vec{\beta}} f_{\vec{\beta}|\vec{y}}(\vec{\beta}|\vec{y}) = \arg \max_{\vec{\beta}} \log f_{\vec{\beta}}(\vec{\beta}) + \log f_{\vec{y}|\vec{\beta}}(\vec{y}|\vec{\beta}). \quad (142)$$

Since, conditioned on $\vec{\beta} = \vec{\beta}$, \vec{y} is iid Gaussian with mean $X\vec{\beta}$ and variance σ_2^2

$$\log f_{\vec{y}|\vec{\beta}}(\vec{y}|\vec{\beta}) = \log \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp \left(-\frac{1}{2\sigma_2^2} (\vec{y}[i] - (X\vec{\beta})[i])^2 \right) \quad (143)$$

$$= \log \frac{1}{\sqrt{(2\pi)^n \sigma_2^n}} \exp \left(-\frac{1}{2\sigma_2^2} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 \right) \quad (144)$$

$$= -\frac{1}{2\sigma_2^2} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 + \log \frac{1}{\sqrt{(2\pi)^n \sigma_2^n}}. \quad (145)$$

Similarly,

$$\log f_{\vec{\beta}}(\vec{\beta}) = -\frac{1}{2\sigma_1^2} \left\| \vec{\beta} \right\|_2^2 + \log \frac{1}{\sqrt{(2\pi)^n} \sigma_1^n}. \quad (146)$$

Setting

$$\lambda := \frac{\sigma_2^2}{\sigma_1^2}, \quad (147)$$

combining (142), (145) and (146) and ignoring the terms that do not depend on $\vec{\beta}$ completes the proof.

Lecture Notes 7: Convex Optimization

1 Convex functions

Convex functions are of crucial importance in optimization-based data analysis because they can be efficiently minimized. In this section we introduce the concept of convexity and then discuss norms, which are convex functions that are often used to design convex cost functions when fitting models to data.

1.1 Convexity

A function is convex if and only if its curve lies below any chord joining two of its points.

Definition 1.1 (Convex function). *A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ and any $\theta \in (0, 1)$,*

$$\theta f(\vec{x}) + (1 - \theta) f(\vec{y}) \geq f(\theta\vec{x} + (1 - \theta)\vec{y}). \quad (1)$$

The function is strictly convex if the inequality is always strict, i.e. if $\vec{x} \neq \vec{y}$ implies that

$$\theta f(\vec{x}) + (1 - \theta) f(\vec{y}) > f(\theta\vec{x} + (1 - \theta)\vec{y}). \quad (2)$$

A concave function is a function f such that $-f$ is convex.

Linear functions are convex, but not strictly convex.

Lemma 1.2. *Linear functions are convex but not strictly convex.*

Proof. If f is linear, for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ and any $\theta \in (0, 1)$,

$$f(\theta\vec{x} + (1 - \theta)\vec{y}) = \theta f(\vec{x}) + (1 - \theta) f(\vec{y}). \quad (3)$$

□

Condition (1) is illustrated in Figure 1. The following lemma shows that when determining whether a function is convex we can restrict our attention to its behavior along lines in \mathbb{R}^n .

Lemma 1.3 (Proof in Section 4.1). *A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only if for any two points $\vec{x}, \vec{y} \in \mathbb{R}^n$ the univariate function $g_{\vec{x}, \vec{y}} : [0, 1] \rightarrow \mathbb{R}$ defined by*

$$g_{\vec{x}, \vec{y}}(\alpha) := f(\alpha\vec{x} + (1 - \alpha)\vec{y}) \quad (4)$$

is convex. Similarly, f is strictly convex if and only if $g_{\vec{x}, \vec{y}}$ is strictly convex for any $\vec{x} \neq \vec{y}$.

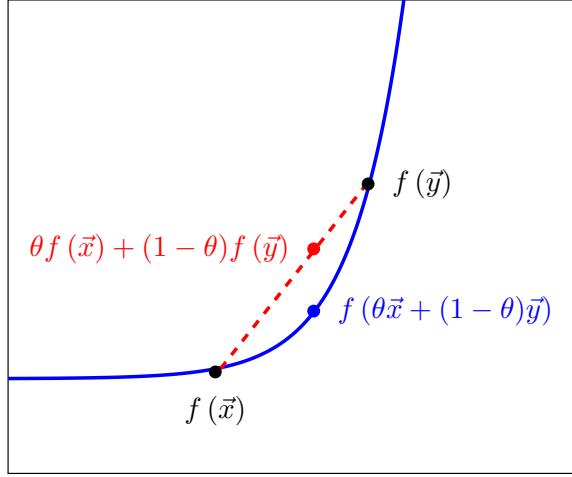


Figure 1: Illustration of condition (1) in Definition 1.1. The curve corresponding to the function must lie below any chord joining two of its points.

Convex functions are easier to optimize than nonconvex functions because once we find a local minimum of the function we are done: every local minimum is guaranteed to be a global minimum.

Theorem 1.4 (Local minima are global). *Any local minimum of a convex function is also a global minimum.*

Proof. We prove the result by contradiction. Let \vec{x}_{loc} be a local minimum and \vec{x}_{glob} a global minimum such that $f(\vec{x}_{\text{glob}}) < f(\vec{x}_{\text{loc}})$. Since \vec{x}_{loc} is a local minimum, there exists $\gamma > 0$ for which $f(\vec{x}_{\text{loc}}) \leq f(\vec{x})$ for all $\vec{x} \in \mathbb{R}^n$ such that $\|\vec{x} - \vec{x}_{\text{loc}}\|_2 \leq \gamma$. If we choose $\theta \in (0, 1)$ small enough, $\vec{x}_\theta := \theta\vec{x}_{\text{loc}} + (1 - \theta)\vec{x}_{\text{glob}}$ satisfies $\|\vec{x}_\theta - \vec{x}_{\text{loc}}\|_2 \leq \gamma$ and therefore

$$f(\vec{x}_{\text{loc}}) \leq f(\vec{x}_\theta) \tag{5}$$

$$\leq \theta f(\vec{x}_{\text{loc}}) + (1 - \theta) f(\vec{x}_{\text{glob}}) \quad \text{by convexity of } f \tag{6}$$

$$< f(\vec{x}_{\text{loc}}) \quad \text{because } f(\vec{x}_{\text{glob}}) < f(\vec{x}_{\text{loc}}). \tag{7}$$

□

1.2 Norms

Many of the cost functions that we consider in data analysis involve norms. Conveniently, all norms are convex.

Lemma 1.5 (Norms are convex). *Any valid norm $\|\cdot\|$ is a convex function.*

Proof. By the triangle inequality inequality and homogeneity of the norm, for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ and any $\theta \in (0, 1)$

$$\|\theta\vec{x} + (1 - \theta)\vec{y}\| \leq \|\theta\vec{x}\| + \|(1 - \theta)\vec{y}\| = \theta \|\vec{x}\| + (1 - \theta) \|\vec{y}\|. \tag{8}$$

□

The following lemma establishes that the composition between a convex function and an affine function is convex. In particular, this means that any function of the form

$$f(\vec{x}) := \left\| A\vec{x} + \vec{b} \right\| \quad (9)$$

is convex for any fixed matrix A and vector \vec{b} with suitable dimensions.

Lemma 1.6 (Composition of convex and affine function). *If $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex, then for any $A \in \mathbb{R}^{n \times m}$ and any $\vec{b} \in \mathbb{R}^n$, the function*

$$h(\vec{x}) := f(A\vec{x} + \vec{b}) \quad (10)$$

is convex.

Proof. By convexity of f , for any $\vec{x}, \vec{y} \in \mathbb{R}^m$ and any $\theta \in (0, 1)$

$$h(\theta\vec{x} + (1 - \theta)\vec{y}) = f\left(\theta\left(A\vec{x} + \vec{b}\right) + (1 - \theta)\left(A\vec{y} + \vec{b}\right)\right) \quad (11)$$

$$\leq \theta f\left(A\vec{x} + \vec{b}\right) + (1 - \theta)f\left(A\vec{y} + \vec{b}\right) \quad (12)$$

$$= \theta h(\vec{x}) + (1 - \theta)h(\vec{y}). \quad (13)$$

□

The number of nonzero entries in a vector is often called the ℓ_0 “norm” of the vector. Despite its name, it is not a valid norm (it is not homogeneous: for any \vec{x} $\|2\vec{x}\|_0 = \|\vec{x}\|_0 \neq \|\vec{x}\|_0$). In fact, the ℓ_0 “norm” is not even convex.

Lemma 1.7 (ℓ_0 “norm” is not convex). *The ℓ_0 “norm” defined as the number of nonzero entries in a vector is not convex.*

Proof. We provide a simple counterexample with vectors in \mathbb{R}^2 that can be easily extended to vectors in \mathbb{R}^n . Let $\vec{x} := (\begin{smallmatrix} 1 \\ 0 \end{smallmatrix})$ and $\vec{y} := (\begin{smallmatrix} 0 \\ 1 \end{smallmatrix})$, then for any $\theta \in (0, 1)$

$$\|\theta\vec{x} + (1 - \theta)\vec{y}\|_0 = 2 > 1 = \theta\|\vec{x}\|_0 + (1 - \theta)\|\vec{y}\|_0. \quad (14)$$

□

Example 1.8 (Promoting sparsity). Finding sparse vectors that are consistent with observed data is often very useful in data analysis. Let us consider a toy problem where the entries of a vector are constrained to be of the form

$$\vec{v}_t := \begin{bmatrix} t \\ t-1 \\ t-1 \end{bmatrix}. \quad (15)$$

Our objective is to fit t so that \vec{v}_t is as sparse as possible or, in other words, minimize $\|\vec{v}_t\|_0$. Unfortunately this function is nonconvex. The graph of the function is depicted in Figure 2. In contrast, if we consider

$$f(t) := \|\vec{v}_t\| \quad (16)$$

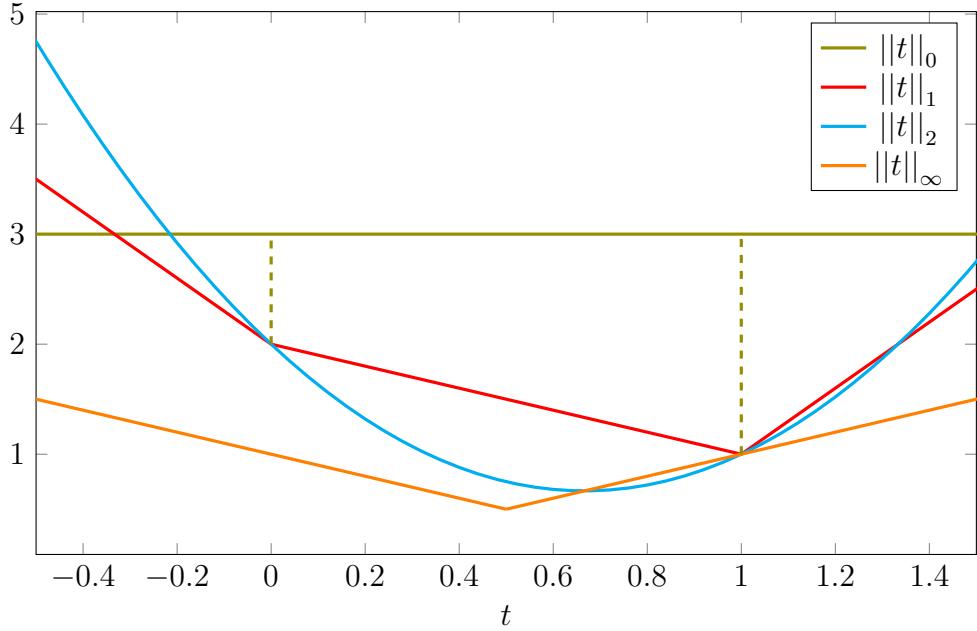


Figure 2: Graph of the function (16) for different norms and for the nonconvex ℓ_0 “norm”.

where $\|\cdot\|$ is a valid norm, then we can exploit local information to find the global minimum (we will discuss how to do this in more detail later on) because the function is convex in t by Lemma 1.6. This is impossible to do for $\|\vec{v}_t\|_0$ because it is constant except at two isolated points. Figure 2 shows f for different norms.

The ℓ_1 norm is the best choice for our purposes: it is convex and its global minimum is at the same location as the minimum ℓ_0 “norm” solution. This is not a coincidence: minimizing the ℓ_1 norm tends to promote sparsity. When compared to the ℓ_2 norm, it penalizes small entries much more (ϵ^2 is much smaller than $|\epsilon|$ for small ϵ), as a result it tends to produce solutions that contain a small number of larger nonzero entries. \triangle

The rank of a matrix interpreted as a function of its entries is also not convex.

Lemma 1.9 (The rank is not convex). *The rank of matrices in $\mathbb{R}^{n \times n}$ interpreted as a function from $\mathbb{R}^{n \times n}$ to \mathbb{R} is not convex.*

Proof. We provide a counterexample that is very similar to the one in the proof of Lemma 1.7. Let

$$X := \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad Y := \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}. \quad (17)$$

For any $\theta \in (0, 1)$

$$\text{rank}(\theta X + (1 - \theta) Y) = 2 > 1 = \theta \text{rank}(X) + (1 - \theta) \text{rank}(Y). \quad (18)$$

\square

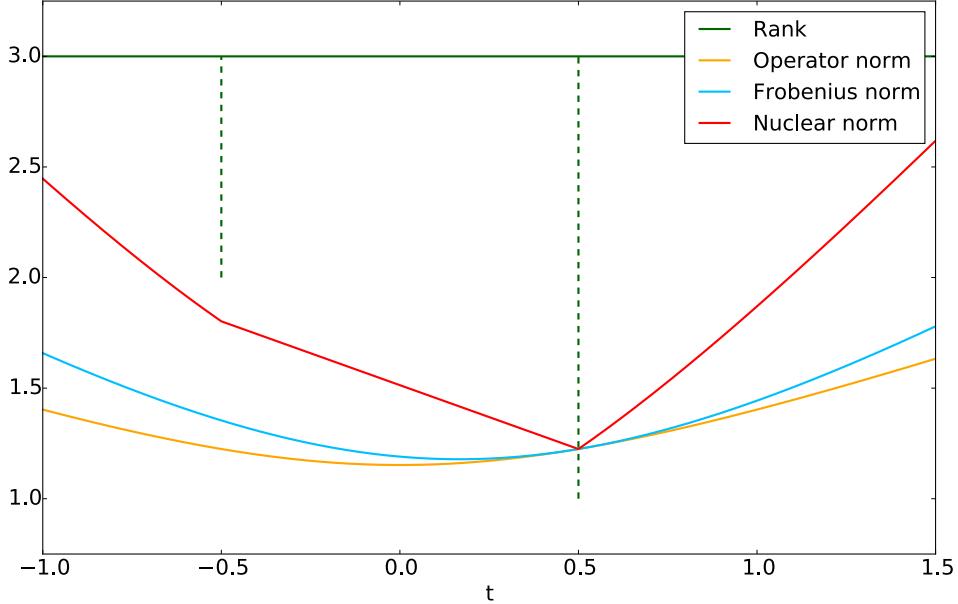


Figure 3: Values of different norms for the matrix $M(t)$ defined by (19). The rank of the matrix for each t is marked in green.

Example 1.10 (Promoting low-rank structure). Finding low-rank matrices that are consistent with data is useful in applications of PCA where data may be corrupted or missing. Let us consider a toy problem where our goal is to find t so that

$$M(t) := \begin{bmatrix} 0.5 + t & 1 & 1 \\ 0.5 & 0.5 & t \\ 0.5 & 1 - t & 0.5 \end{bmatrix}, \quad (19)$$

is as low rank as possible. In Figure 3 we compare the rank, the operator norm, the Frobenius norm and the nuclear norm of $M(t)$ for different values of t . As expected, the rank is highly nonconvex, whereas the norms are all convex, which follows from Lemma 1.6. The value of t that minimizes the rank is the same as the one that minimizes the nuclear norm. In contrast, the values of t that minimize the operator and Frobenius norms are different. Just like the ℓ_1 norm promotes sparsity, the nuclear norm, which is the ℓ_0 “norm” of the singular values, promotes solutions with low rank, which is the ℓ_0 “norm” of the singular values. \triangle

2 Differentiable convex functions

2.1 First-order conditions

The gradient is the generalization of the concept of derivative, which captures the local rate of change in the value of a function, in multiple directions.

Definition 2.1 (Gradient). *The gradient of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ at a point $\vec{x} \in \mathbb{R}^n$ is defined to be the unique vector $\nabla f(\vec{x}) \in \mathbb{R}^n$ satisfying*

$$\lim_{\vec{p} \rightarrow 0} \frac{f(x + \vec{p}) - f(x) - \nabla f(\vec{x})^T \vec{p}}{\|\vec{p}\|_2} = 0,$$

assuming such a vector $\nabla f(\vec{x})$ exists. If $\nabla f(\vec{x})$ exists then it is given by the vector of partial derivatives:

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f(\vec{x})}{\partial \vec{x}[1]} \\ \frac{\partial f(\vec{x})}{\partial \vec{x}[2]} \\ \vdots \\ \frac{\partial f(\vec{x})}{\partial \vec{x}[n]} \end{bmatrix}. \quad (20)$$

If the gradient exists at every point, the function is said to be differentiable.

The gradient encodes the variation of the function in every direction.

Lemma 2.2. *If a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable, the directional derivative $f'_{\vec{u}}$ of f at \vec{x} equals*

$$f'_{\vec{u}}(\vec{x}) := \lim_{h \rightarrow 0} \frac{f(\vec{x} + h\vec{u}) - f(\vec{x})}{h} \quad (21)$$

$$= \langle \nabla f(\vec{x}), \vec{u} \rangle \quad (22)$$

for any unit-norm vector $\vec{u} \in \mathbb{R}^n$.

We omit the proof of the lemma, which is a basic result from multivariable calculus. An important corollary is that the gradient provides the direction of maximum positive and negative variation of the function.

Corollary 2.3. *The direction of the gradient ∇f of a differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the direction of maximum increase of the function. The opposite direction is the direction of maximum decrease.*

Proof. By the Cauchy-Schwarz inequality

$$|f'_{\vec{u}}(\vec{x})| = |\nabla f(\vec{x})^T \vec{u}| \quad (23)$$

$$\leq \|\nabla f(\vec{x})\|_2 \|\vec{u}\|_2 \quad (24)$$

$$= \|\nabla f(\vec{x})\|_2 \quad (25)$$

with equality if and only if $\vec{u} = \pm \frac{\nabla f(\vec{x})}{\|\nabla f(\vec{x})\|_2}$. □

Figure 4 shows the gradient of a function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ at different locations. The gradient is orthogonal to the contour lines of the function. The reason is that by definition the function does not change along the contour lines, so the directional derivatives in those directions are zero.

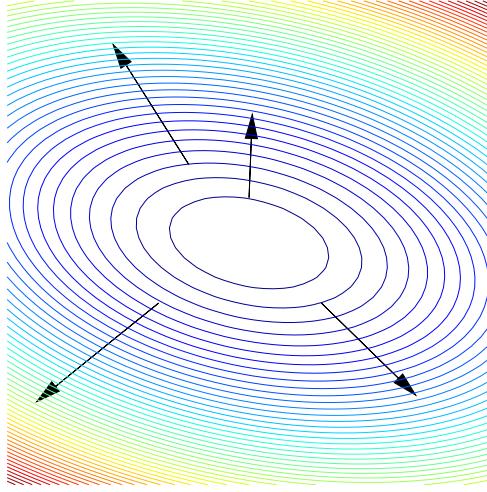


Figure 4: Contour lines of a function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$. The gradients at different points are represented by black arrows, which are orthogonal to the contour lines.

The first-order Taylor expansion of a differentiable function is a linear function that approximates the function around a certain point. Geometrically, in one dimension this linear approximation is a line that is tangent to the curve $(x, f(x))$. In multiple dimensions, it is a hyperplane that is tangent to the hypersurface $(\vec{x}, f(\vec{x}))$ at that point.

Definition 2.4 (First-order approximation). *The first-order or linear approximation of a differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ at \vec{x} is*

$$f_{\vec{x}}^1(\vec{y}) := f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}). \quad (26)$$

By construction, the first-order approximation of a function at a given point is a linear function that has exactly the same directional derivatives at that point. The following theorem establishes that a function f is convex if and only if the linear approximation $f_{\vec{x}}^1$ is a lower bound of f for any $\vec{x} \in \mathbb{R}^n$. Figure 5 illustrates the condition.

Theorem 2.5 (Proof in Section 4.2). *A differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only if for every $\vec{x}, \vec{y} \in \mathbb{R}^n$*

$$f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}). \quad (27)$$

It is strictly convex if and only if

$$f(\vec{y}) > f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}). \quad (28)$$

An immediate corollary is that for a convex function, any point at which the gradient is zero is a global minimum. If the function is strictly convex, the minimum is unique. This is very useful for minimizing such functions, once we find a point where the gradient is zero we are done!

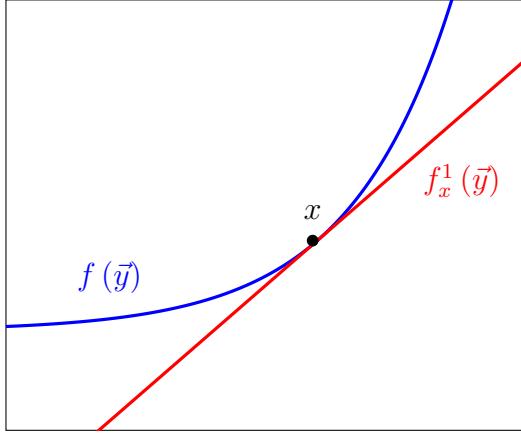


Figure 5: An example of the first-order condition for convexity. The first-order approximation at any point is a lower bound of the function.

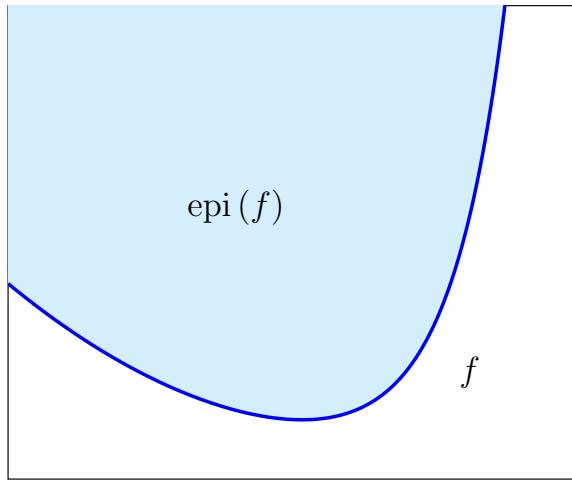


Figure 6: Epigraph of a function.

Corollary 2.6. If a differentiable function f is convex and $\nabla f(\vec{x}) = 0$, then for any $\vec{y} \in \mathbb{R}$

$$f(\vec{y}) \geq f(\vec{x}). \quad (29)$$

If f is strictly convex then for any $\vec{y} \neq \vec{x}$

$$f(\vec{y}) > f(\vec{x}). \quad (30)$$

For any differentiable function f and any $\vec{x} \in \mathbb{R}^n$ let us define the hyperplane $\mathcal{H}_{f,\vec{x}} \subset \mathbb{R}^{n+1}$ that corresponds to the first-order approximation of f at \vec{x} ,

$$\mathcal{H}_{f,\vec{x}} := \left\{ \vec{y} \mid \vec{y}[n+1] = f_{\vec{x}}^1 \begin{pmatrix} \vec{y}[1] \\ \vdots \\ \vec{y}[n] \end{pmatrix} \right\}. \quad (31)$$

The epigraph is the subset of \mathbb{R}^{n+1} that lies above the graph of the function. Recall that the graph is the set of vectors in \mathbb{R}^{n+1} obtained by concatenating $\vec{x} \in \mathbb{R}^n$ and $f(\vec{x})$ for every $\vec{x} \in \mathbb{R}^n$. Figure 6 shows the epigraph of a convex function.

Definition 2.7 (Epigraph). *The epigraph of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the set*

$$\text{epi}(f) := \left\{ \vec{x} \mid f \begin{pmatrix} \vec{x}[1] \\ \vdots \\ \vec{x}[n] \end{pmatrix} \leq \vec{x}[n+1] \right\}. \quad (32)$$

Geometrically, Theorem 2.5 establishes that the epigraph of a convex function always lies above $\mathcal{H}_{f,\vec{x}}$. By construction, $\mathcal{H}_{f,\vec{x}}$ and $\text{epi}(f)$ intersect at \vec{x} . This implies that $\mathcal{H}_{f,\vec{x}}$ is a supporting hyperplane of $\text{epi}(f)$ at \vec{x} .

Definition 2.8 (Supporting hyperplane). *A hyperplane \mathcal{H} is a supporting hyperplane of a set \mathcal{S} at \vec{x} if*

- \mathcal{H} and \mathcal{S} intersect at \vec{x} ,
- \mathcal{S} is contained in one of the half-spaces bounded by \mathcal{H} .

The optimality condition in Corollary 2.6 has a very intuitive geometric interpretation in terms of the supporting hyperplane $\mathcal{H}_{f,\vec{x}}$. $\nabla f = 0$ implies that $\mathcal{H}_{f,\vec{x}}$ is horizontal if the vertical dimension corresponds to the $n + 1$ th coordinate. Since the epigraph lies above hyperplane, the point at which they intersect must be a minimum of the function.

2.2 Second-order conditions

The Hessian matrix of a function contains its second-order partial derivatives.

Definition 2.9 (Hessian matrix). *A differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is twice differentiable at $\vec{x} \in \mathbb{R}^n$ if there is a matrix $\nabla^2 f(\vec{x}) \in \mathbb{R}^{n \times n}$ such that*

$$\lim_{\vec{p} \rightarrow 0} \frac{\|\nabla f(\vec{x} + \vec{p}) - \nabla f(\vec{x}) - \nabla^2 f(\vec{x})\vec{p}\|_2}{\|\vec{p}\|_2} = 0.$$

If $\nabla^2 f(\vec{x})$ exists then it is given by

$$\nabla^2 f(\vec{x}) = \begin{bmatrix} \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]^2} & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]\partial \vec{x}[2]} & \cdots & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]\partial \vec{x}[n]} \\ \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]\partial \vec{x}[2]} & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]^2} & \cdots & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[2]\partial \vec{x}[n]} \\ & & \ddots & \\ \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[1]\partial \vec{x}[n]} & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[2]\partial \vec{x}[n]} & \cdots & \frac{\partial^2 f(\vec{x})}{\partial \vec{x}[n]^2} \end{bmatrix}. \quad (33)$$

If a function has a Hessian matrix at every point, we say that the function is twice differentiable. If each entry of the Hessian is continuous, we say f is twice continuously differentiable.

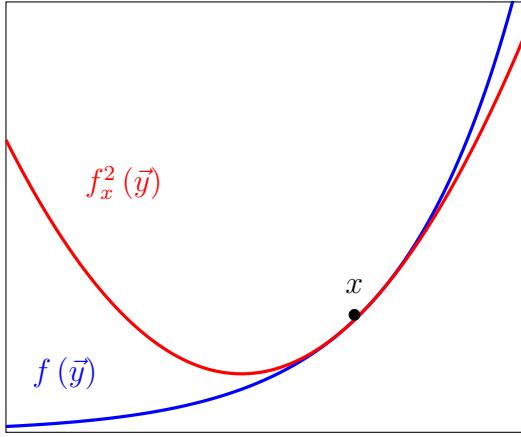


Figure 7: Second-order approximation of a function.

Note that by (33) if $f : \mathbb{R} \rightarrow \mathbb{R}^n$ is differentiable everywhere and twice differentiable at $\vec{x} \in \mathbb{R}^n$ then the Hessian $\nabla^2 f(\vec{x})$ is always a symmetric matrix.

As you might recall from basic calculus, curvature is the rate of change of the slope of the function and is consequently given by its second derivative. The Hessian matrix encodes the curvature of the function in every direction, another basic result from multivariable calculus.

Lemma 2.10. *If a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is twice differentiable, the second directional derivative $f''_{\vec{u}}$ of f at \vec{x} equals*

$$f''_{\vec{u}}(\vec{x}) = \vec{u}^T \nabla^2 f(\vec{x}) \vec{u}, \quad (34)$$

for any unit-norm vector $\vec{u} \in \mathbb{R}^n$.

The Hessian and the gradient of a twice-differentiable function can be used to build a quadratic approximation of the function. This approximation is depicted in Figure 7 for a one-dimensional function.

Definition 2.11 (Second-order approximation). *The second-order or quadratic approximation of f at \vec{x} is*

$$f_{\vec{x}}^2(\vec{y}) := f(\vec{x}) + \nabla f(\vec{x})(\vec{y} - \vec{x}) + \frac{1}{2} (\vec{y} - \vec{x})^T \nabla^2 f(\vec{x}) (\vec{y} - \vec{x}) \quad (35)$$

By construction, the second-order approximation of a function at a given point is a quadratic form that has exactly the same directional derivatives and curvature at that point. This second-order approximation is a quadratic form.

Definition 2.12 (Quadratic functions/forms). *A quadratic function $q : \mathbb{R}^n \rightarrow \mathbb{R}$ is a second-order polynomial in several dimensions. Such polynomials can be written in terms of a symmetric matrix $A \in \mathbb{R}^{n \times n}$, a vector $\vec{b} \in \mathbb{R}^n$ and a constant c*

$$q(\vec{x}) := \vec{x}^T A \vec{x} + \vec{b}^T \vec{x} + c. \quad (36)$$

A quadratic form is a (pure) quadratic function where $\vec{b} = 0$ and $c = 0$.

The quadratic form $f_{\vec{x}}^2(\vec{y})$ becomes an arbitrarily good approximation of f as we approach \vec{x} , even if we divide the error by the squared distance between \vec{x} and \vec{y} . We omit the proof that follows from multivariable calculus.

Lemma 2.13. *The quadratic approximation $f_{\vec{x}}^2 : \mathbb{R}^n \rightarrow \mathbb{R}$ at $\vec{x} \in \mathbb{R}^n$ of a twice differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfies*

$$\lim_{\vec{y} \rightarrow \vec{x}} \frac{f(\vec{y}) - f_{\vec{x}}^2(\vec{y})}{\|\vec{y} - \vec{x}\|_2^2} = 0 \quad (37)$$

To find the maximum curvature of a function at a given point, we can compute an eigendecomposition of its Hessian.

Theorem 2.14. *Let $A = U\Lambda U^T$ be the eigendecomposition of a symmetric matrix A , where $\lambda_1 \geq \dots \geq \lambda_n$ (which can be negative) are the eigenvalues and $\vec{u}_1, \dots, \vec{u}_n$ the corresponding eigenvectors. Then*

$$\lambda_1 = \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \vec{x}^T A \vec{x}, \quad (38)$$

$$\vec{u}_1 = \arg \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \vec{x}^T A \vec{x}, \quad (39)$$

$$\lambda_n = \min_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \vec{x}^T A \vec{x}, \quad (40)$$

$$\vec{u}_n = \arg \min_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \vec{x}^T A \vec{x}. \quad (41)$$

Proof. By Theorem 4.3 in Lecture Notes 2 the eigendecomposition of A is the same as its SVD, except that some of the eigenvalues may be negative (which flips the direction of the corresponding eigenvectors with respect to the singular vectors). The result then follows from Theorem 2.7 in the same lecture notes. \square

Corollary 2.15. *Consider the eigendecomposition of the Hessian matrix of a twice-differentiable function f at a point \vec{x} . The maximum curvature of f at \vec{x} is given by the largest eigenvalue of $\nabla^2 f(\vec{x})$ and is in the direction of the corresponding eigenvector. The smallest curvature, or the largest negative curvature, of f at \vec{x} is given by the smallest eigenvalue of $\nabla^2 f(\vec{x})$ and is in the direction of the corresponding eigenvector.*

If all the eigenvalues of a symmetric matrix $A \in \mathbb{R}^{n \times n}$ are nonnegative, the matrix is said to be *positive semidefinite*. The pure quadratic form corresponding to such matrices is always nonnegative.

Lemma 2.16 (Positive semidefinite matrices). *The eigenvalues of a symmetric matrix $A \in \mathbb{R}^{n \times n}$ are all nonnegative if and only if*

$$\vec{x}^T A \vec{x} \geq 0 \quad (42)$$

for all $\vec{x} \in \mathbb{R}^n$. Such matrices are called *positive semidefinite*.

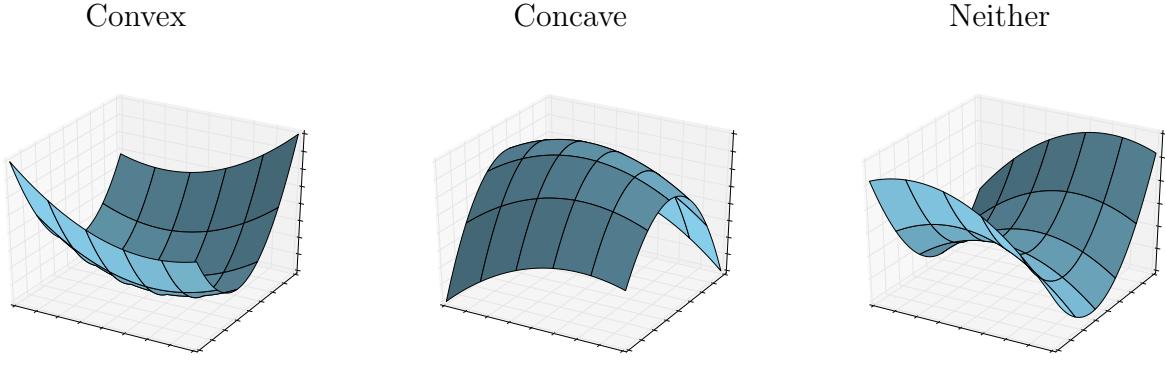


Figure 8: Quadratic forms for which the Hessian is positive definite (left), negative definite (center) and neither positive nor negative definite (right).

Proof. By Theorem 2.14, the matrix has an eigendecomposition $A = U\Lambda U^T$ where the eigenvectors $\vec{u}_1, \dots, \vec{u}_n$ form an orthonormal basis so that

$$\vec{x}^T A \vec{x} = \vec{x}^T U \Lambda U^T \vec{x} \quad (43)$$

$$= \sum_{i=1}^n \lambda_i \langle \vec{u}_i, \vec{x} \rangle^2. \quad (44)$$

□

If the eigenvalues are positive the matrix is *positive definite*. If the eigenvalues are all nonpositive, the matrix is negative semidefinite. If they are negative, the matrix is negative definite. By Corollary 2.15 a twice-differentiable function has positive (resp. nonnegative) curvature in *every* direction if its Hessian is positive definite (resp. semidefinite) and it has negative (resp. nonpositive) curvature in every direction if the Hessian is negative definite (resp. semidefinite). Figure 8 illustrates this in the case of quadratic forms in two dimensions.

For univariate functions that are twice differentiable, convexity is dictated by the curvature. The following lemma establishes that univariate functions are convex if and only if their curvature is always nonnegative.

Lemma 2.17 (Proof in Section 4.4). *A twice-differentiable function $g : \mathbb{R} \rightarrow \mathbb{R}$ is convex if and only if $g''(x) \geq 0$ for all $x \in \mathbb{R}$.*

A corollary of this result is that twice-differentiable functions in \mathbb{R}^n are convex if and only if their Hessian is positive semidefinite at every point.

Corollary 2.18. *A twice-differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only if for every $\vec{x} \in \mathbb{R}^n$, the Hessian matrix $\nabla^2 f(\vec{x})$ is positive semidefinite.*

Proof. By Lemma 1.3 we just need to show that the univariate function $g_{\vec{a}, \vec{b}}$ defined by (4) is convex for all $\vec{a}, \vec{b} \in \mathbb{R}^n$. By Lemma 2.17 this holds if and only if the second derivative of $g_{\vec{a}, \vec{b}}$ is nonnegative. This quantity is nonnegative for all $\vec{a}, \vec{b} \in \mathbb{R}^n$ if and only if $\nabla^2 f(\vec{x})$ is positive semidefinite for any $\vec{x} \in \mathbb{R}^n$. □

Remark 2.19 (Strict convexity). *If the Hessian is positive definite, then the function is strictly convex (the proof is essentially the same). However, there are functions that are strictly convex for which the Hessian may equal zero at some points. An example is the univariate function $f(x) = x^4$, for which $f''(0) = 0$.*

We can interpret Corollary 2.18 in terms of the second-order Taylor expansion of $f : \mathbb{R}^n \rightarrow \mathbb{R}$ at \vec{x} : f is convex if and only if this quadratic approximation is always convex.

3 Minimizing differentiable convex functions

In this section we describe different techniques for solving the optimization problem

$$\min_{\vec{x} \in \mathbb{R}^n} f(\vec{x}), \quad (45)$$

when f is differentiable and convex. By Theorem 1.4 any local minimum of the function is also a global minimum. This motivates trying to make progress towards a solution by exploiting local first and second order information.

3.1 Gradient descent

Gradient descent exploits first-order local information encoded in the gradient to iteratively approach the point at which f achieves its minimum value. The idea is to take steps in the direction of steepest descent, which is $-\nabla f(\vec{x})$ by Corollary 2.3.

Algorithm 3.1 (Gradient descent, aka steepest descent). *Set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Update by setting*

$$\vec{x}^{(k+1)} := \vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)}), \quad (46)$$

where $\alpha_k > 0$ is a nonnegative real number which we call the step size, until a stopping criterion is met.

Examples of stopping criteria include checking whether the relative progress

$$\frac{\|\vec{x}^{(k+1)} - \vec{x}^{(k)}\|_2}{\|\vec{x}^{(k)}\|_2} \quad (47)$$

or the norm of the gradient are below a predefined tolerance. Figure 9 shows two examples in which gradient descent is applied in one and two dimensions. In both cases the method converges to the minimum.

In the examples of Figure 9 the step size is constant. In practice, determining a constant step that is adequate for a particular function can be challenging. Figure 10 shows two examples to illustrate this. In the first, the step size is too small and as a result convergence is extremely slow. In the second the step size is too large which causes the algorithm to repeatedly overshoot the minimum and eventually diverge.

$$f : \mathbb{R} \rightarrow \mathbb{R}$$

$$f : \mathbb{R}^2 \rightarrow \mathbb{R}$$

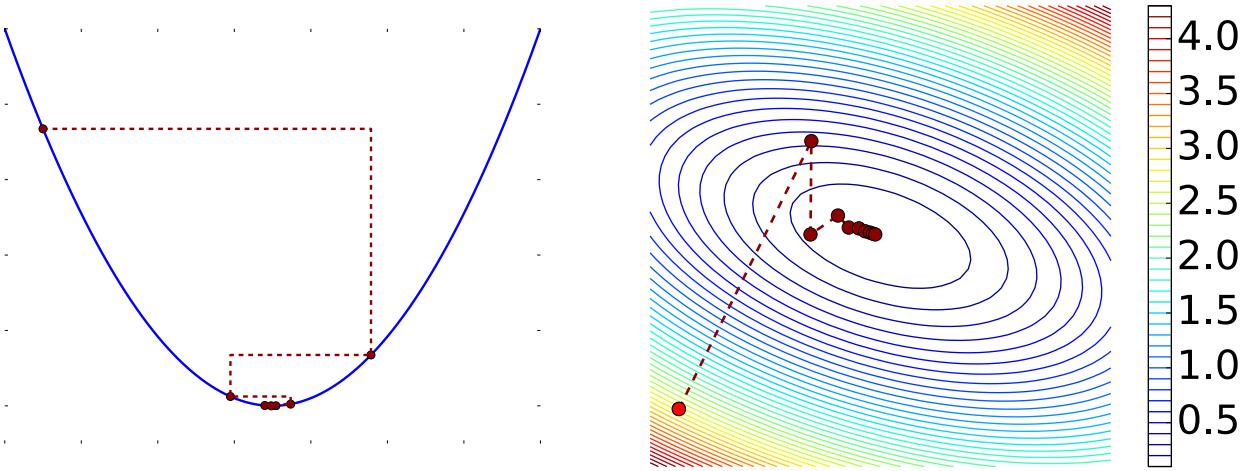


Figure 9: Iterations of gradient descent applied to a univariate (left) and a bivariate (right) function. The algorithm converges to the minimum in both cases.

Ideally, we would like to adapt the step size automatically as the iterations progress. A possibility is to search for the minimum of the function along the direction of the gradient,

$$\alpha_k := \arg \min_{\alpha} h(\alpha) \quad (48)$$

$$= \arg \min_{\alpha \in \mathbb{R}} f(\vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)})) . \quad (49)$$

This is called a line search. Recall that the restriction of an n -dimensional convex function to a line in its domain is also convex. As a result the line-search problem is a one-dimensional convex problem. However, it may still be costly to solve. The backtracking line search is an alternative heuristic that produces very similar results in practice at less cost. The idea is to ensure that we make some progress in each iteration, without worrying about actually minimizing the univariate function.

Algorithm 3.2 (Backtracking line search with Armijo rule). *Given $\alpha^0 \geq 0$ and $\beta, \eta \in (0, 1)$, set $\alpha_k := \alpha^0 \beta^i$ for the smallest integer i such that $\vec{x}^{(k+1)} := \vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)})$ satisfies*

$$f(\vec{x}^{(k+1)}) \leq f(\vec{x}^{(k)}) - \frac{1}{2} \alpha_k \|\nabla f(\vec{x}^{(k)})\|_2^2 , \quad (50)$$

a condition known as Armijo rule

Figure 11 shows the result of applying gradient descent with a backtracking line search to the same example as in Figure 10. In this case, the line search manages to adjust the step size so that the method converges.

Example 3.3 (Gradient descent for least squares). Gradient descent can be used to minimize the least-squares cost function

$$\text{minimize}_{\vec{\beta} \in \mathbb{R}^p} \left\| \vec{y} - X \vec{\beta} \right\|_2^2 , \quad (51)$$

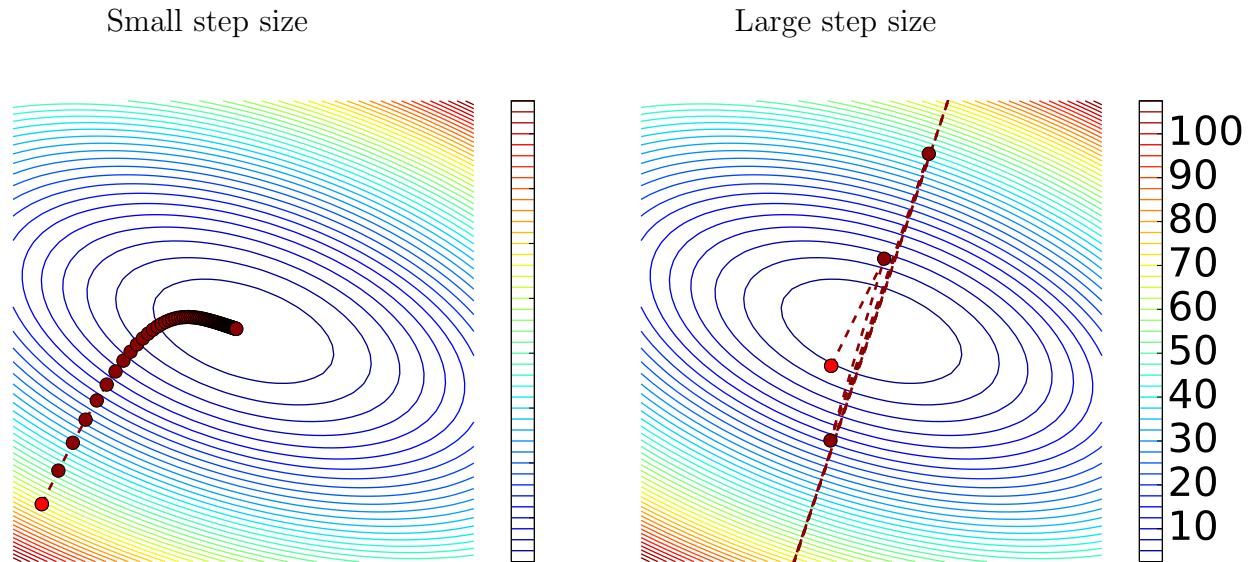


Figure 10: Iterations of gradient descent when the step size is small (left) and large (right). In the first case the convergence is very small, whereas in the second the algorithm diverges away from the minimum. The initial point is bright red.

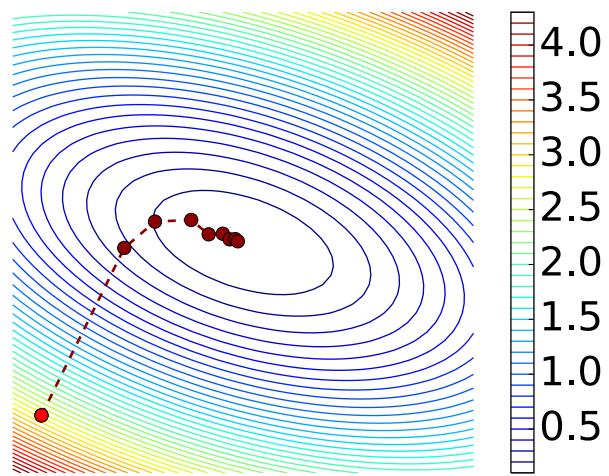


Figure 11: Gradient descent using a backtracking line search based on the Armijo rule. The function is the same as in Figure 10.

described in Section 2 of Lecture Notes 6 to fit a linear regression model from n examples of the form,

$$(y^{(1)}, \vec{x}^{(1)}) , (y^{(2)}, \vec{x}^{(2)}) , \dots , (y^{(n)}, \vec{x}^{(n)}) . \quad (52)$$

The cost function is convex by Lemma 1.6 (also its Hessian $X^T X$ is positive semidefinite). The gradient of the quadratic function

$$f(\vec{\beta}) := \left\| \vec{y} - X\vec{\beta} \right\|_2^2 \quad (53)$$

$$= \vec{\beta}^T X^T X \vec{\beta} - 2\vec{\beta}^T X^T \vec{y} + \vec{y}^T \vec{y} \quad (54)$$

equals

$$\nabla f(\vec{\beta}) = 2X^T X \vec{\beta} - 2X^T \vec{y} \quad (55)$$

so the gradient descent updates are

$$\vec{\beta}^{(k+1)} = \vec{\beta}^{(k)} + 2\alpha_k X^T (\vec{y} - X\vec{\beta}^{(k)}) \quad (56)$$

$$= \vec{\beta}^{(k)} + 2\alpha_k \sum_{i=1}^n \left(\vec{y}^{(i)} - \langle x^{(i)}, \vec{\beta}^{(k)} \rangle \right) x^{(i)}. \quad (57)$$

This has a very intuitive interpretation in terms of the examples: if $\vec{y}^{(i)}$ is larger than $\langle x^{(i)}, \vec{\beta}^{(k)} \rangle$ we add a small multiple of $x^{(i)}$ in order to reduce the difference, if it is smaller we subtract it.

Gradient descent is not the best first-order iterative optimization method for least-squares minimization. The conjugate-gradients method is better suited for this problem. We refer to [5] for an excellent tutorial on this method. \triangle

Example 3.4 (Gradient ascent for logistic regression). Since gradient descent minimizes convex functions, gradient ascent (where we climb in the direction of the gradient) can be used to maximize concave functions. In particular, let us consider the logistic regression log-likelihood cost function

$$f(\vec{\beta}) := \sum_{i=1}^n y^{(i)} \log g\left(\langle \vec{x}^{(i)}, \vec{\beta} \rangle\right) + (1 - y^{(i)}) \log \left(1 - g\left(\langle \vec{x}^{(i)}, \vec{\beta} \rangle\right)\right) \quad (58)$$

from Definition 4.2 of Lecture Notes 6, where $g(t) = (1 - \exp -t)^{-1}$ the labels $y^{(i)}$, $1 \leq i \leq n$, are equal to 0 or 1, and the features $\vec{x}^{(i)}$ are vectors in \mathbb{R}^n . We establish that this cost function is concave in Corollary 3.23. The gradient of this cost function is given in the following lemma.

Lemma 3.5. *The gradient of the function f in equation (58) equals*

$$\nabla f(\vec{\beta}) = \sum_{i=1}^n y^{(i)} \left(1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)\right) \vec{x}^{(i)} - (1 - y^{(i)}) g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle) \vec{x}^{(i)}. \quad (59)$$

Proof. The result follows from the identities

$$g'(t) = g(t)(1 - g(t)), \quad (60)$$

$$(1 - g(t))' = -g(t)(1 - g(t)). \quad (61)$$

and the chain rule. \triangle

The gradient ascent updates

$$\vec{\beta}^{(k+1)} := \vec{\beta}^{(k)} + \alpha_k \sum_{i=1}^n y^{(i)} \left(1 - g(\langle \vec{x}^{(i)}, \vec{\beta}^{(k)} \rangle) \right) \vec{x}^{(i)} - (1 - y^{(i)}) g(\langle \vec{x}^{(i)}, \vec{\beta}^{(k)} \rangle) \vec{x}^{(i)}. \quad (62)$$

have an intuitive interpretation. If $\vec{y}^{(i)}$ equals 1, we add $\vec{x}^{(i)}$ scaled by the error $1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)$ to push $g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)$ up towards $\vec{y}^{(i)}$. Similarly, if $\vec{y}^{(i)}$ equals 0, we subtract $\vec{x}^{(i)}$ scaled by the error $g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)$ to push $g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)$ down towards $\vec{y}^{(i)}$. \triangle

3.2 Convergence of gradient descent

In this section we analyze the convergence of gradient descent. We begin by introducing a notion of continuity for functions from \mathbb{R}^n to \mathbb{R}^m .

Definition 3.6 (Lipschitz continuity). *A function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is Lipschitz continuous with Lipschitz constant L if for any $\vec{x}, \vec{y} \in \mathbb{R}^n$*

$$\|f(\vec{y}) - f(\vec{x})\|_2 \leq L \|\vec{y} - \vec{x}\|_2. \quad (63)$$

We focus on functions that have Lipschitz-continuous gradients. The following theorem shows that these functions are upper bounded by a quadratic function.

Theorem 3.7 (Proof in Section 4.5). *If the gradient of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is Lipschitz continuous with Lipschitz constant L ,*

$$\|\nabla f(\vec{y}) - \nabla f(\vec{x})\|_2 \leq L \|\vec{y} - \vec{x}\|_2 \quad (64)$$

then for any $\vec{x}, \vec{y} \in \mathbb{R}^n$

$$f(\vec{y}) \leq f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}) + \frac{L}{2} \|\vec{y} - \vec{x}\|_2^2. \quad (65)$$

The quadratic upper bound immediately implies a bound on the value of the cost function after k iterations of gradient descent.

Corollary 3.8. *Let $\vec{x}^{(i)}$ be the i th iteration of gradient descent and $\alpha_i \geq 0$ the i th step size, if ∇f is L -Lipschitz continuous,*

$$f(\vec{x}^{(k+1)}) \leq f(\vec{x}^{(k)}) - \alpha_k \left(1 - \frac{\alpha_k L}{2} \right) \|\nabla f(\vec{x}^{(k)})\|_2^2. \quad (66)$$

Proof. Applying the quadratic upper bound we obtain

$$f(\vec{x}^{(k+1)}) \leq f(\vec{x}^{(k)}) + \nabla f(\vec{x}^{(k)})^T (\vec{x}^{(k+1)} - \vec{x}^{(k)}) + \frac{L}{2} \|\vec{x}^{(k+1)} - \vec{x}^{(k)}\|_2^2. \quad (67)$$

The result follows because $\vec{x}^{(k+1)} - \vec{x}^{(k)} = -\alpha_k \nabla f(\vec{x}^{(k)})$. \square

We can now establish that if the step size is small enough, the value of the cost function at each iteration will decrease (unless we are at the minimum, where the gradient is zero).

Corollary 3.9 (Gradient descent is a descent method). *If $\alpha_k \leq \frac{1}{L}$*

$$f(\vec{x}^{(k+1)}) \leq f(\vec{x}^{(k)}) - \frac{\alpha_k}{2} \|\nabla f(\vec{x}^{(k)})\|_2^2. \quad (68)$$

Note that up to now we are *not* assuming that the function we are minimizing is convex. Gradient descent will make local progress even for nonconvex functions if the step size is sufficiently small. We now establish global convergence for gradient descent applied to convex functions with Lipschitz-continuous gradients.

Theorem 3.10. *We assume that f is convex, ∇f is L -Lipschitz continuous and there exists a point \vec{x}^* at which f achieves a finite minimum. If we set the step size of gradient descent to $\alpha_k = \alpha \leq 1/L$ for every iteration,*

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \frac{\|\vec{x}^{(0)} - \vec{x}^*\|_2^2}{2\alpha k} \quad (69)$$

Proof. By the first-order characterization of convexity

$$f(\vec{x}^{(k-1)}) + \nabla f(\vec{x}^{(k-1)})^T (\vec{x}^* - \vec{x}^{(k-1)}) \leq f(\vec{x}^*), \quad (70)$$

which together with Corollary 3.9 yields

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \nabla f(\vec{x}^{(k-1)})^T (\vec{x}^{(k-1)} - \vec{x}^*) - \frac{\alpha}{2} \|\nabla f(\vec{x}^{(k-1)})\|_2^2 \quad (71)$$

$$= \frac{1}{2\alpha} \left(\|\vec{x}^{(k-1)} - \vec{x}^*\|_2^2 - \|\vec{x}^{(k-1)} - \vec{x}^* - \alpha \nabla f(\vec{x}^{(k-1)})\|_2^2 \right) \quad (72)$$

$$= \frac{1}{2\alpha} \left(\|\vec{x}^{(k-1)} - \vec{x}^*\|_2^2 - \|\vec{x}^{(k)} - \vec{x}^*\|_2^2 \right) \quad (73)$$

Using the fact that by Corollary 3.9 the value of f never increases, we have

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \frac{1}{k} \sum_{i=1}^k f(\vec{x}^{(i)}) - f(\vec{x}^*) \quad (74)$$

$$\leq \frac{1}{2\alpha k} \left(\|\vec{x}^{(0)} - \vec{x}^*\|_2^2 - \|\vec{x}^{(k)} - \vec{x}^*\|_2^2 \right) \quad (75)$$

$$\leq \frac{\|\vec{x}^{(0)} - \vec{x}^*\|_2^2}{2\alpha k}. \quad (76)$$

□

The theorem assumes that we know the Lipschitz constant of the gradient beforehand. However, the following lemma establishes that a backtracking line search with the Armijo rule is capable of adjusting the step size adequately.

Lemma 3.11 (Backtracking line search). *If the gradient of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is Lipschitz continuous with Lipschitz constant L the step size obtained by applying a backtracking line search using the Armijo rule with $\eta = 0.5$ satisfies*

$$\alpha_k \geq \alpha_{\min} := \min \left\{ \alpha^0, \frac{\beta}{L} \right\}. \quad (77)$$

Proof. By Corollary 3.8 the Armijo rule with $\eta = 0.5$ is satisfied if $\alpha_k \leq 1/L$. Since there must exist an integer i for which $\beta/L \leq \alpha^0 \beta^i \leq 1/L$ this establishes the result. \square

We can now adapt the proof of Theorem 3.10 to establish convergence when we apply a backtracking line search.

Theorem 3.12 (Convergence with backtracking line search). *If f is convex and ∇f is L -Lipschitz continuous. Gradient descent with a backtracking line search produces a sequence of points that satisfy*

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \frac{\|\vec{x}^{(0)} - \vec{x}^*\|_2^2}{2\alpha_{\min} k}, \quad (78)$$

where $\alpha_{\min} := \min \left\{ \alpha^0, \frac{\beta}{L} \right\}$.

Proof. Following the reasoning in the proof of Theorem 3.10 up until equation (73) we have

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \frac{1}{2\alpha_i} \left(\|x^{(k-1)} - x^*\|_2^2 - \|x^{(k)} - x^*\|_2^2 \right). \quad (79)$$

By Lemma 3.11 $\alpha_i \geq \alpha_{\min}$, so we just mimic the steps at the end of the proof of Theorem 3.10 to obtain

$$f(x^{(k)}) - f(x^*) \leq \frac{1}{k} \sum_{i=1}^k f(x^{(k)}) - f(x^*) \quad (80)$$

$$= \frac{1}{2\alpha_{\min} k} \left(\|x^{(0)} - x^*\|_2^2 - \|x^{(k)} - x^*\|_2^2 \right) \quad (81)$$

$$\leq \frac{\|x^{(0)} - x^*\|_2^2}{2\alpha_{\min} k}. \quad (82)$$

\square

The results that we have proved imply that we need $\mathcal{O}(1/\epsilon)$ to compute a point at which the cost function has a value that is ϵ close to the minimum. However, in practice gradient descent and related methods often converge much faster.

3.3 Accelerated gradient descent

The following theorem by Nesterov shows that no algorithm that uses first-order information can converge faster than $\mathcal{O}(1/\sqrt{\epsilon})$ for the class of functions with Lipschitz-continuous gradients. The proof is constructive, see Section 2.1.2 of [4] for the details.

Theorem 3.13 (Lower bound on rate of convergence). *There exist convex functions with L -Lipschitz-continuous gradients such that for any algorithm that selects $x^{(k)}$ from*

$$x^{(0)} + \text{span} \{ \nabla f(x^{(0)}), \nabla f(x^{(1)}), \dots, \nabla f(x^{(k-1)}) \} \quad (83)$$

we have

$$f(x^{(k)}) - f(x^*) \geq \frac{3L \|x^{(0)} - x^*\|_2^2}{32(k+1)^2}. \quad (84)$$

This rate is in fact optimal. The convergence of $\mathcal{O}(1/\sqrt{\epsilon})$ can be achieved if we modify gradient descent by adding a momentum term.

Algorithm 3.14 (Nesterov's accelerated gradient descent). *Set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Update by setting*

$$y^{(k+1)} = x^{(k)} - \alpha_k \nabla f(x^{(k)}), \quad (85)$$

$$x^{(k+1)} = \beta_k y^{(k+1)} + \gamma_k y^{(k)}, \quad (86)$$

where α_k is the step size and β_k and γ_k are nonnegative real parameters, until a stopping criterion is met.

Intuitively, the momentum term $y^{(k)}$ prevents the algorithm from overreacting to changes in the local slope of the function. We refer the interested reader to [3, 4] for more details.

Example 3.15 (Digit classification). In this example we apply both gradient descent and accelerated gradient descent to train a logistic-regression model on the MNIST data set¹. We consider the task of determining whether a digit is a 5 or not. The feature vector \vec{x}_i contains the pixel values of an image of a 5 ($\vec{y}_i = 1$) or another number ($\vec{y}_i = 0$). We use different numbers of training examples to fit a logistic regression model. The cost function is minimized by running gradient descent and accelerated gradient descent until the gradient is smaller than a certain value. Figure 12 shows the time taken by both algorithms for different training-set sizes. \triangle

3.4 Stochastic gradient descent

Cost functions used to fit models from data are often additive, in the sense that we can write them as a sum of m terms, each of which often depends on just one measurement,

$$f(\vec{x}) = \frac{1}{m} \sum_{i=1}^m f_i(\vec{x}). \quad (87)$$

¹Available at <http://yann.lecun.com/exdb/mnist/>

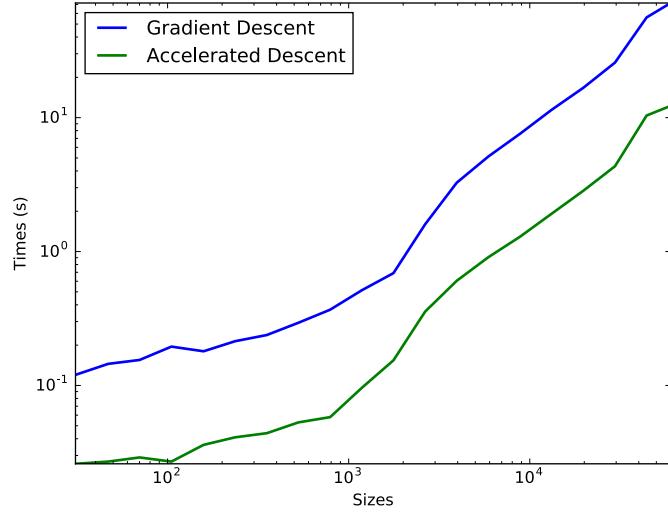


Figure 12: Time taken by gradient descent and accelerated gradient descent to train a logistic-regression model on the MNIST data set for different training-set sizes.

Two important examples are the least-squares and logistic-regression log-likelihood functions discussed in Examples 3.3 and 3.4. In those cases each term f_i corresponds to a different example in the training set. If the training set is extremely large, then computing the whole gradient may be computationally infeasible. Stochastic gradient descent circumvents this issue by using the gradient of individual components instead.

Algorithm 3.16 (Stochastic gradient descent). *Set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Until a stopping criterion is met, update by:*

1. Choosing a random subset of b indices \mathcal{B} , where $b \ll m$ is the batch size.
2. Setting

$$\vec{x}^{(k+1)} := \vec{x}^{(k)} - \alpha_k \sum_{i \in \mathcal{B}} \nabla f_i (\vec{x}^{(k)}) \quad (88)$$

where $\alpha_k > 0$ is the step size.

Apart from its computational efficiency, an advantage of stochastic gradient descent is that it can be applied in *online* settings, where we need to optimize a function that depends on a large data set but only have access to portions of the data set at a time.

Intuitively, stochastic gradient descent replaces the gradient of the whole function by

$$\sum_{i=1}^m 1_{i \in \mathcal{B}} \nabla f_i (\vec{x}^{(k+1)}). \quad (89)$$

If \mathcal{B} is generated so that every index has the same probability p of belonging to it, then this is an

unbiased estimate of ∇f

$$\mathbb{E} \left(\sum_{i=1}^m 1_{i \in \mathcal{B}} \nabla f_i (\vec{x}^{(k)}) \right) = \sum_{i=1}^m \mathbb{E} (1_{i \in \mathcal{B}}) \nabla f_i (\vec{x}^{(k)}) \quad (90)$$

$$= \sum_{i=1}^m \mathbb{P}(i \in \mathcal{B}) \nabla f_i (\vec{x}^{(k)}) \quad (91)$$

$$= mp \nabla f (\vec{x}^{(k)}). \quad (92)$$

Intuitively, the direction of the stochastic-gradient descent is the one of steepest descent *on average*. However, the variation in the estimate can make stochastic gradient descent diverge unless the step size is diminishing. We refer to [1] for more details on this algorithm.

Example 3.17 (Stochastic gradient descent for least squares and logistic regression). By the derivation in Example 3.3, in the case of least squares the stochastic gradient descent update is

$$\vec{\beta}^{(k+1)} := \vec{\beta}^{(k)} + 2\alpha_k \sum_{i \in \mathcal{B}} \left(\vec{y}^{(i)} - \langle \vec{x}^{(i)}, \vec{\beta}^{(k)} \rangle \right) \vec{x}^{(i)}, \quad (93)$$

Similarly, by the derivation in Example 3.4, the update for stochastic gradient ascent applied to logistic regression is

$$\vec{\beta}^{(k+1)} := \vec{\beta}^{(k)} + \alpha_k \sum_{i \in \mathcal{B}} y^{(i)} \left(1 - g(\langle \vec{x}^{(i)}, \vec{\beta}^{(k)} \rangle) \right) \vec{x}^{(i)} - (1 - y^{(i)}) g(\langle \vec{x}^{(i)}, \vec{\beta}^{(k)} \rangle) \vec{x}^{(i)}. \quad (94)$$

In both cases the algorithm is very intuitive: instead of adjusting $\vec{\beta}$ using all of the examples, we just use the ones in the batch. \triangle

Example 3.18 (Digit classification). In this example we apply stochastic gradient descent for to train a logistic-regression model on the MNIST data set for the same task as Example 3.18. Figure 13 shows the convergence of the algorithm for different batch sizes. \triangle

3.5 Newton's method

Newton's method minimizes a convex function by iteratively minimizing its quadratic approximation. The following simple lemma derives a closed form for the minimum of the quadratic approximation at a given point.

Lemma 3.19. *The minimum of the second-order approximation of a convex function f at $\vec{x} \in \mathbb{R}^n$*

$$f_{\vec{x}}^2 (\vec{y}) := f(\vec{x}) + \nabla f(\vec{x})(\vec{y} - \vec{x}) + \frac{1}{2} (\vec{y} - \vec{x})^T \nabla^2 f(\vec{x}) (\vec{y} - \vec{x}), \quad (95)$$

which has a positive definite Hessian at \vec{x} , is equal to

$$\arg \min_{\vec{y} \in \mathbb{R}^n} f_{\vec{x}}^2 (\vec{y}) = \vec{x} - \nabla^2 f(\vec{x})^{-1} \nabla f(\vec{x}). \quad (96)$$

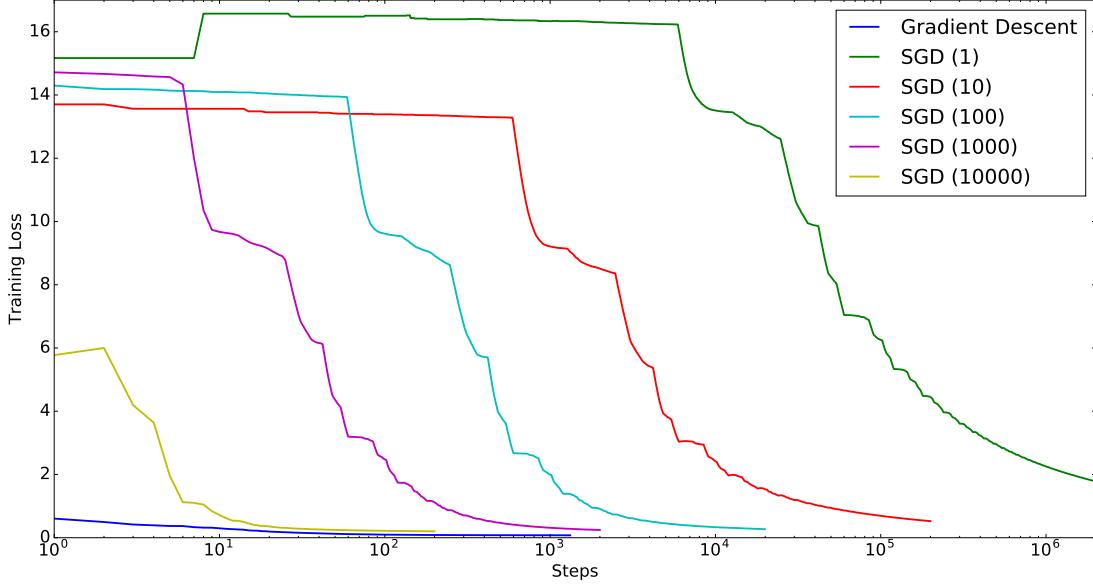


Figure 13: Convergence of stochastic gradient descent when fitting a logistic-regression model on the MNIST data set. The plot shows the value of the cost function on the training set for different batch sizes. Note that the updates for smaller batch sizes are much faster so the horizontal axis does not reflect running time.

Proof. If the Hessian is positive definite, then $f_{\vec{x}}^2$ is strictly convex and it has a unique global minimum. Its gradient equals

$$\nabla f_{\vec{x}}^2(y) = \nabla f(\vec{x}) + \nabla^2 f(\vec{x})(\vec{y} - \vec{x}) \quad (97)$$

so it is equal to zero if

$$\nabla^2 f(\vec{x})(\vec{y} - \vec{x}) = -\nabla f(\vec{x}). \quad (98)$$

If the Hessian is positive definite, then it is also full rank. The only solution to this system of equations is consequently $\vec{y} = \vec{x} - \nabla^2 f(\vec{x})^{-1} \nabla f(\vec{x})$, which must be the minimum of $f_{\vec{x}}^2$ by Corollary 2.6 because the gradient vanishes. \square

The idea behind Newton's method is that convex functions are often well approximated by quadratic functions, especially close to their minimum.

Algorithm 3.20 (Newton's method). *Set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Update by setting*

$$\vec{x}^{(k+1)} := \vec{x}^{(k)} - \nabla^2 f(\vec{x}^{(k)})^{-1} \nabla f(\vec{x}^{(k)}) \quad (99)$$

until a stopping criterion is met.

Figure 14 illustrates Newton's method in a one-dimensional setting. When applied to a quadratic function, the algorithm converges in one step: if we start at the origin it is equivalent to computing

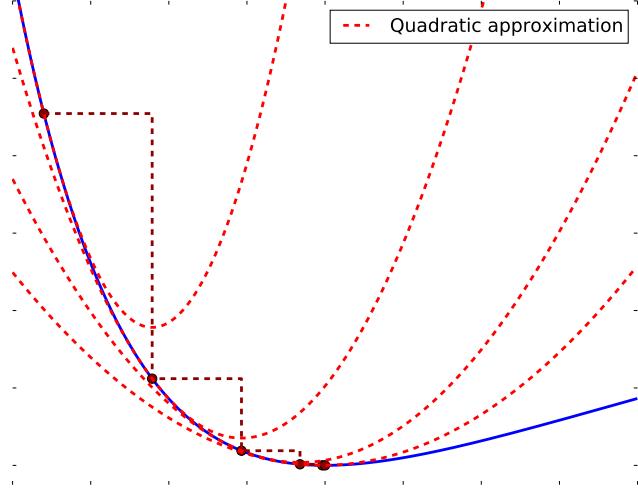


Figure 14: Newton’s method applied to a one-dimensional convex function. The quadratic approximations to the function at each iteration are depicted in red.

the closed-form solution for least squares derived in Theorem 2.1 of Lectures Notes 6. This is illustrated in Figure 15, along with another example where the function is convex but not quadratic. Newton’s method can provide significant acceleration for problems of moderate sizes where the quadratic approximation is accurate, but often inverting the Hessian may be computationally expensive.

Example 3.21 (Newton’s method for logistic regression). The following lemma derives the Hessian of the logistic regression log-likelihood cost function (58).

Lemma 3.22. *The Hessian of the function f in equation (58) equals*

$$\nabla^2 f(\vec{\beta}) = -X^T G(\vec{\beta}) X, \quad (100)$$

where the rows of $X \in \mathbb{R}^{n \times p}$ contain the feature vectors $\vec{x}^{(1)}, \dots, \vec{x}^{(n)}$ and G is a diagonal matrix such that

$$G(\vec{\beta})_{ii} := g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle) \left(1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)\right), \quad 1 \leq i \leq n. \quad (101)$$

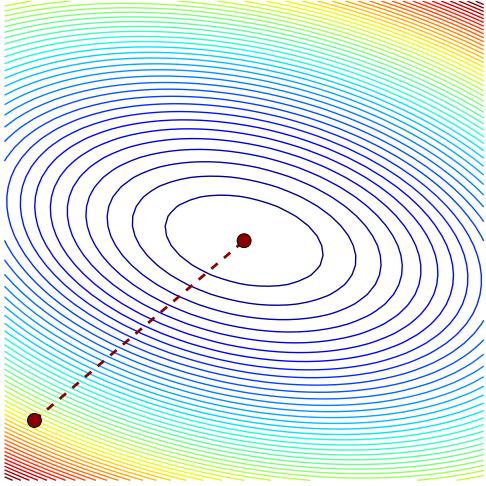
Proof. By the identities (60) and (61) and the chain rule we have

$$\frac{\partial^2 f(\vec{x})}{\partial \vec{x}[j] \partial \vec{x}[l]} = - \sum_{i=1}^n g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle) \left(1 - g(\langle \vec{x}^{(i)}, \vec{\beta} \rangle)\right) \vec{x}^{(i)}[j] \vec{x}^{(i)}[l] \quad (102)$$

for $1 \leq j, l \leq p$. \triangle

Corollary 3.23. *The logistic regression log-likelihood cost function (58) is concave.*

Quadratic function



Convex function

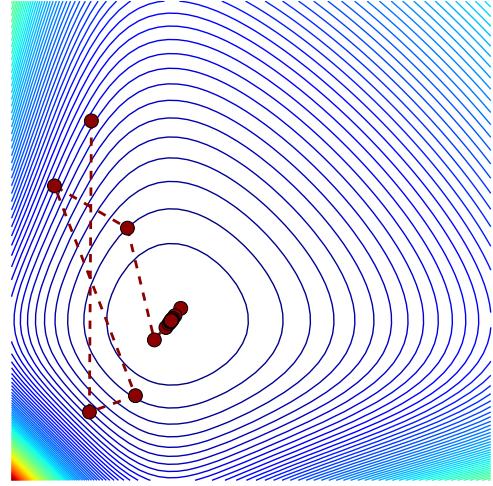


Figure 15: Newton’s method applied to a quadratic function (left) and to a convex function that is not quadratic.

Proof. The Hessian is negative semidefinite, for any arbitrary $\vec{\beta}, \vec{v} \in \mathbb{R}^p$

$$\vec{v}^T \nabla^2 f(\vec{\beta}) \vec{v} = - \sum_{i=1}^n G(\vec{\beta})_{ii} (X \vec{v}) [i]^2 \leq 0 \quad (103)$$

since the entries of $G(\vec{\beta})$ are nonnegative. \triangle

The Newton updates are consequently of the form

$$\vec{\beta}^{(k+1)} := \vec{\beta}^{(k)} - \left(X^T G(\vec{\beta}^{(k)}) X \right)^{-1} \nabla f(\vec{\beta}^{(k)}). \quad (104)$$

A potentially problematic feature of this application of Newton’s method is that the Hessian $X^T G(\vec{\beta}) X$ may become ill conditioned if most of the examples are classified correctly, since in that case the matrix G mostly contains zeros. Intuitively, in this case the cost function is very *flat* in certain directions. \triangle

4 Proofs

4.1 Proof of Lemma 1.3

The proof for strict convexity is exactly the same, replacing the inequalities by strict inequalities.

f being convex implies that $g_{\vec{x}, \vec{y}}$ is convex for any $\vec{x}, \vec{y} \in \mathbb{R}^n$

For any $\alpha, \beta, \theta \in (0, 1)$

$$g_{\vec{x}, \vec{y}}(\theta\alpha + (1 - \theta)\beta) = f((\theta\alpha + (1 - \theta)\beta)\vec{x} + (1 - \theta\alpha - (1 - \theta)\beta)\vec{y}) \quad (105)$$

$$= f(\theta(\alpha\vec{x} + (1 - \alpha)\vec{y}) + (1 - \theta)(\beta\vec{x} + (1 - \beta)\vec{y})) \quad (106)$$

$$\leq \theta f(\alpha\vec{x} + (1 - \alpha)\vec{y}) + (1 - \theta)f(\beta\vec{x} + (1 - \beta)\vec{y}) \quad \text{by convexity of } f \\ = \theta g_{\vec{x}, \vec{y}}(\alpha) + (1 - \theta)g_{\vec{x}, \vec{y}}(\beta). \quad (107)$$

$g_{\vec{x}, \vec{y}}$ being convex for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ implies that f is convex

For any $\alpha, \beta, \theta \in (0, 1)$

$$f(\theta\vec{x} + (1 - \theta)\vec{y}) = g_{\vec{x}, \vec{y}}(\theta) \quad (108)$$

$$\leq \theta g_{\vec{x}, \vec{y}}(1) + (1 - \theta)g_{\vec{x}, \vec{y}}(0) \quad \text{by convexity of } g_{\vec{x}, \vec{y}} \quad (109)$$

$$= \theta f(\vec{x}) + (1 - \theta)f(\vec{y}). \quad (110)$$

4.2 Proof of Theorem 2.5

The proof for strict convexity is almost exactly the same; we omit the details.

The following lemma, proved in Section 4.3 below establishes that the result holds for univariate functions

Lemma 4.1. *A univariate differentiable function $g : \mathbb{R} \rightarrow \mathbb{R}$ is convex if and only if for all $x, y \in \mathbb{R}$*

$$g(y) \geq g'(x)(y - x) + g(x) \quad (111)$$

and strictly convex if and only if for all $x, y \in \mathbb{R}$

$$g(y) > g'(x)(y - x) + g(x). \quad (112)$$

To complete the proof we extend the result to the multivariable case using Lemma 1.3.

If $f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x})$ for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ then f is convex

By Lemma 1.3 we just need to show that the univariate function $g_{\vec{a}, \vec{b}}$ defined by (4) is convex for all $\vec{a}, \vec{b} \in \mathbb{R}^n$. Applying some basic multivariate calculus yields

$$g'_{\vec{a}, \vec{b}}(\alpha) = \nabla f\left(\alpha\vec{a} + (1 - \alpha)\vec{b}\right)^T(\vec{a} - \vec{b}). \quad (113)$$

Let $\alpha, \beta \in \mathbb{R}$. Setting $\vec{x} := \alpha\vec{a} + (1 - \alpha)\vec{b}$ and $\vec{y} := \beta\vec{a} + (1 - \beta)\vec{b}$ we have

$$g_{\vec{a}, \vec{b}}(\beta) = f(\vec{y}) \quad (114)$$

$$\geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x}) \quad (115)$$

$$= f\left(\alpha\vec{a} + (1 - \alpha)\vec{b}\right) + \nabla f\left(\alpha\vec{a} + (1 - \alpha)\vec{b}\right)^T(\vec{a} - \vec{b})(\beta - \alpha) \quad (116)$$

$$= g_{\vec{a}, \vec{b}}(\alpha) + g'_{\vec{a}, \vec{b}}(\alpha)(\beta - \alpha) \quad \text{by (113),} \quad (117)$$

which establishes that $g_{\vec{a}, \vec{b}}$ is convex by Lemma 4.1 above.

If f is convex then $f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x})$ for any $\vec{x}, \vec{y} \in \mathbb{R}^n$

By Lemma 1.3, $g_{\vec{x}, \vec{y}}$ is convex for any $\vec{x}, \vec{y} \in \mathbb{R}^n$.

$$f(\vec{y}) = g_{\vec{x}, \vec{y}}(1) \tag{118}$$

$$\geq g_{\vec{x}, \vec{y}}(0) + g'_{\vec{x}, \vec{y}}(0) \quad \text{by convexity of } g_{\vec{x}, \vec{y}} \text{ and Lemma 4.1} \tag{119}$$

$$= f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}) \quad \text{by (113).} \tag{120}$$

4.3 Proof of Lemma 4.1

g being convex implies $g(y) \geq g'(x)(y - x) + g(x)$ for all $x, y \in \mathbb{R}$

If g is convex then for any $x, y \in \mathbb{R}$ and any $0 \leq \theta \leq 1$

$$\theta(g(y) - g(x)) + g(x) \geq g(x + \theta(y - x)). \tag{121}$$

Rearranging the terms we have

$$g(y) \geq \frac{g(x + \theta(y - x)) - g(x)}{\theta} + g(x). \tag{122}$$

Setting $h = \theta(y - x)$, this implies

$$g(y) \geq \frac{g(x + h) - g(x)}{h}(y - x) + g(x). \tag{123}$$

Taking the limit when $h \rightarrow 0$ yields

$$g(y) \geq g'(x)(y - x) + g(x). \tag{124}$$

If $g(y) \geq g'(x)(y - x) + g(x)$ for all $x, y \in \mathbb{R}$ then g is convex

Let $z = \theta x + (1 - \theta)y$, then by if $g(y) \geq g'(x)(y - x) + g(x)$

$$g(x) \geq g'(z)(x - z) + g(z) \tag{125}$$

$$= g'(z)(1 - \theta)(x - y) + g(z) \tag{126}$$

$$g(y) \geq g'(z)(y - z) + g(z) \tag{127}$$

$$= g'(z)\theta(y - x) + g(z) \tag{128}$$

Multiplying (126) by θ , then (128) by $1 - \theta$ and summing the inequalities, we obtain

$$\theta g(x) + (1 - \theta)g(y) \geq g(\theta x + (1 - \theta)y). \tag{129}$$

4.4 Proof of Lemma 2.17

The second derivative of g is nonnegative anywhere if and only if the first derivative is nondecreasing, because g'' is the derivative of g' .

If g is convex g' is nondecreasing

By Lemma 4.1, if the function is convex then for any $x, y \in \mathbb{R}$ such that $y > x$

$$g(x) \geq g'(y)(x - y) + g(y), \quad (130)$$

$$g(y) \geq g'(x)(y - x) + g(x). \quad (131)$$

Rearranging, we obtain

$$g'(y)(y - x) \geq g(y) - g(x) \geq g'(x)(y - x). \quad (132)$$

Since $y - x > 0$, we have $g'(y) \geq g'(x)$.

If g' is nondecreasing, g is convex

For arbitrary $x, y, \theta \in \mathbb{R}$, such that $y > x$ and $0 < \theta < 1$, let $\eta = \theta y + (1 - \theta)x$. Since $y > \eta > x$, by the mean-value theorem there exist $\gamma_1 \in [x, \eta]$ and $\gamma_2 \in [\eta, y]$ such that

$$g'(\gamma_1) = \frac{g(\eta) - g(x)}{\eta - x}, \quad (133)$$

$$g'(\gamma_2) = \frac{g(y) - g(\eta)}{y - \eta}. \quad (134)$$

Since $\gamma_1 < \gamma_2$, if g' is nondecreasing

$$\frac{g(y) - g(\eta)}{y - \eta} \geq \frac{g(\eta) - g(x)}{\eta - x}, \quad (135)$$

which implies

$$\frac{\eta - x}{y - x}g(y) + \frac{y - \eta}{y - x}g(x) \geq g(\eta). \quad (136)$$

Recall that $\eta = \theta y + (1 - \theta)x$, so that $\theta = (\eta - x) / (y - x)$ and $1 - \theta = (y - \eta) / (y - x)$. (136) is consequently equivalent to

$$\theta g(y) + (1 - \theta)g(x) \geq g(\theta y + (1 - \theta)x). \quad (137)$$

4.5 Proof of Proposition 3.7

Consider the function

$$g(\vec{x}) := \frac{L}{2}\vec{x}^T\vec{x} - f(\vec{x}). \quad (138)$$

We first establish that g is convex using the following lemma, proved in Section 4.6 below.

Lemma 4.2 (Monotonicity of gradient). *A differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only if*

$$(\nabla f(\vec{y}) - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}) \geq 0. \quad (139)$$

By the Cauchy-Schwarz inequality, Lipschitz continuity of the gradient of f implies

$$(\nabla f(\vec{y}) - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}) \leq L \|\vec{y} - \vec{x}\|_2^2, \quad (140)$$

for any $\vec{x}, \vec{y} \in \mathbb{R}^n$. This directly implies

$$(\nabla g(\vec{y}) - \nabla g(\vec{x}))^T (\vec{y} - \vec{x}) = (L\vec{y} - L\vec{x} + \nabla f(\vec{x}) - \nabla f(\vec{y}))^T (\vec{y} - \vec{x}) \quad (141)$$

$$= L \|\vec{y} - \vec{x}\|_2^2 - (\nabla f(\vec{y}) - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}) \quad (142)$$

$$\geq 0 \quad (143)$$

and hence that g is convex. By the first-order condition for convexity,

$$\frac{L}{2} \vec{y}^T \vec{y} - f(\vec{y}) = g(\vec{y}) \quad (144)$$

$$\geq g(\vec{x}) + \nabla g(\vec{x})^T (\vec{y} - \vec{x}) \quad (145)$$

$$= \frac{L}{2} \vec{x}^T \vec{x} - f(\vec{x}) + (L\vec{x} - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}). \quad (146)$$

Rearranging the inequality we conclude that

$$f(\vec{y}) \leq f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}) + \frac{L}{2} \|\vec{y} - \vec{x}\|_2^2. \quad (147)$$

4.6 Proof of Lemma 4.2

Convexity implies $(\nabla f(\vec{y}) - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}) \geq 0$ for all $\vec{x}, \vec{y} \in R^n$

If f is convex, by the first-order condition for convexity

$$f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T (\vec{y} - \vec{x}), \quad (148)$$

$$f(\vec{x}) \geq f(\vec{y}) + \nabla f(\vec{y})^T (\vec{x} - \vec{y}), \quad (149)$$

$$(150)$$

Adding the two inequalities directly implies the result.

$(\nabla f(\vec{y}) - \nabla f(\vec{x}))^T (\vec{y} - \vec{x}) \geq 0$ for all $\vec{x}, \vec{y} \in R^n$ implies convexity

Recall the univariate function $g_{a,b} : [0, 1] \rightarrow \mathbb{R}$ defined by

$$g_{a,b}(\alpha) := f(\alpha a + (1 - \alpha)b), \quad (151)$$

for any $a, b \in \mathbb{R}^n$. By multivariate calculus, $g'_{a,b}(\alpha) = \nabla f(\alpha a + (1 - \alpha)b)^T (a - b)$. For any $\alpha \in (0, 1)$ we have

$$g'_{a,b}(\alpha) - g'_{a,b}(0) = (\nabla f(\alpha a + (1 - \alpha)b) - \nabla f(b))^T (a - b) \quad (152)$$

$$= \frac{1}{\alpha} (\nabla f(\alpha a + (1 - \alpha)b) - \nabla f(b))^T (\alpha a + (1 - \alpha)b - b) \quad (153)$$

$$\geq 0 \quad \text{because } (\nabla f(y) - \nabla f(x))^T (y - x) \geq 0 \text{ for any } x, y. \quad (154)$$

This allows us to prove that the first-order condition for convexity holds. For any \vec{x}, \vec{y}

$$f(\vec{x}) = g_{\vec{x}, \vec{y}}(1) \tag{155}$$

$$= g_{\vec{x}, \vec{y}}(0) + \int_0^1 g'_{\vec{x}, \vec{y}}(\alpha) d\alpha \tag{156}$$

$$\geq g_{\vec{x}, \vec{y}}(0) + g'_{\vec{x}, \vec{y}}(0) \tag{157}$$

$$= f(\vec{y}) + \nabla f(\vec{y})(\vec{x} - \vec{y}). \tag{158}$$

References

A very readable and exhaustive reference on convex optimization is Boyd and Vandenberghe's seminal book [2].

- [1] L. Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.
- [2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [3] S. Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.
- [4] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.
- [5] J. R. Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain, 1994.

Lecture Notes 8:

Convex Nondifferentiable Functions

1 Applications

1.1 Sparse regression

In our description of linear regression in Lecture Notes 6, we assume implicitly that all features are related to the response. However, this is often not the case in applications: some measured features may be unrelated and should not be included in the model. Selecting relevant features is a crucial problem in statistics, which is known as *model selection*. In this section, we consider the problem of selecting a small subset of relevant features that yield a good linear approximation to the data. This is equivalent to finding a *sparse* vector of coefficients $\vec{\beta}$ such that

$$y^{(i)} \approx \langle \vec{x}^{(i)}, \vec{\beta} \rangle. \quad (1)$$

The number of selected features is equal to the number of nonzero entries in $\vec{\beta}$.

When fitting a sparse linear model we have two objectives:

- Achieving a good fit to the data; $\left\| X\vec{\beta} - \vec{y} \right\|_2^2$ should be as small as possible.
- Using a small number of features; $\vec{\beta}$ should be as sparse as possible.

This suggests minimizing a cost function that simultaneously promotes a good fit to the data and sparsity in the vector of coefficients. In Lecture Notes 6 we describe the ridge-regression estimator, that uses an ℓ_2 -norm regularization term to ensure that the norm of the coefficients is not too large. Here we would like to control the number of nonzeros in the support of the coefficient, i.e. its ℓ_0 “norm” instead. However, this “norm” is not convex and very difficult to minimize (see Example 1.8 in Lecture Notes 7). Instead, we incorporate an ℓ_1 -norm regularization that promotes sparsity, but is still convex. In statistics, the solution to an ℓ_1 -norm-regularized least-squares problem is called the *lasso* estimator, introduced in [9] (see also [6]).

Definition 1.1 (The lasso). *For $X \in \mathbb{R}^{n \times p}$ and $\vec{y} \in \mathbb{R}^p$, the lasso estimate is the minimizer of the optimization problem*

$$\vec{\beta}_{\text{lasso}} := \arg \min_{\vec{\beta}} \frac{1}{2} \left\| \vec{y} - X\vec{\beta} \right\|_2^2 + \lambda \left\| \vec{\beta} \right\|_1, \quad (2)$$

where $\lambda > 0$ is a fixed regularization parameter.

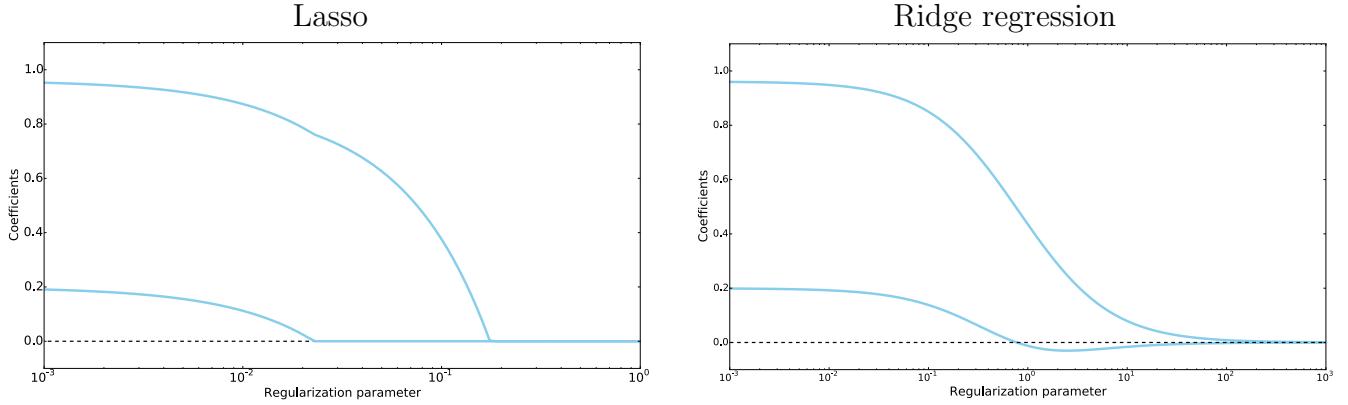


Figure 1: Coefficients of the lasso and ridge-regression estimates in the sparse regression problem in Example 1.4 for $\alpha = 1$, 5 examples ($n = 5$), $\rho := -0.43$ and different values of the regularization parameter λ .

The following lemma shows that sums of convex functions are convex, so the lasso cost function is indeed convex.

Lemma 1.2 (Nonnegative weighted sums). *The weighted sum of m convex functions f_1, \dots, f_m*

$$f := \sum_{i=1}^m \alpha_i f_i \quad (3)$$

is convex as long as the weights $\alpha_1, \dots, \alpha \in \mathbb{R}$ are nonnegative.

Proof. By convexity of f_1, \dots, f_m , for any $\vec{x}, \vec{y} \in \mathbb{R}^m$ and any $\theta \in (0, 1)$

$$f(\theta \vec{x} + (1 - \theta) \vec{y}) = \sum_{i=1}^m \alpha_i f_i(\theta \vec{x} + (1 - \theta) \vec{y}) \quad (4)$$

$$\leq \sum_{i=1}^m \alpha_i (\theta f_i(\vec{x}) + (1 - \theta) f_i(\vec{y})) \quad (5)$$

$$= \theta f(\vec{x}) + (1 - \theta) f(\vec{y}). \quad (6)$$

□

Corollary 1.3 (Regularized least squares). *Regularized least-squares cost functions of the form*

$$\|A\vec{x} - \vec{y}\|_2^2 + \|\vec{x}\|, \quad (7)$$

where $\|\cdot\|$ is an arbitrary norm, are convex.

Example 1.4 (Sparse regression with two features). We consider a simple sparse regression problem where the response only depends on one feature,

$$\vec{y} := \alpha \vec{x}_1 + \vec{z}, \quad (8)$$

where $\vec{y} \in \mathbb{R}^n$ is the response vector, $\vec{x} \in \mathbb{R}^n$ contains the relevant feature and $\vec{z} \in \mathbb{R}^n$ is additive noise. In our data set, there are two features \vec{x}_1 and \vec{x}_2 , which is irrelevant to the response. However, we don't know this a priori, so we use the feature matrix

$$X := [\vec{x}_1 \quad \vec{x}_2] \quad (9)$$

to fit a linear-regression model with both features by minimizing the least-squares cost function. Both features are normalized so that $\|\vec{x}_1\|_2 = \|\vec{x}_2\|_2 = 1$. The correlation between them equals

$$\rho := \langle \vec{x}_1, \vec{x}_2 \rangle. \quad (10)$$

Unfortunately, the least-square estimate of the vector of coefficients is dense

$$\vec{\beta}_{\text{LS}} = (X^T X)^{-1} X^T \vec{y} \quad (11)$$

$$= \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}^{-1} \begin{bmatrix} \vec{x}_1^T \vec{y} \\ \vec{x}_2^T \vec{y} \end{bmatrix} \quad (12)$$

$$= \frac{1}{1 - \rho^2} \begin{bmatrix} 1 & -\rho \\ -\rho & 1 \end{bmatrix} \begin{bmatrix} \alpha + \vec{x}_1^T \vec{z} \\ \alpha\rho + \vec{x}_2^T \vec{z} \end{bmatrix} \quad (13)$$

$$= \begin{bmatrix} \alpha \\ 0 \end{bmatrix} + \frac{1}{1 - \rho^2} \begin{bmatrix} \langle \vec{x}_1 - \rho \vec{x}_2, \vec{z} \rangle \\ \langle \vec{x}_2 - \rho \vec{x}_1, \vec{z} \rangle \end{bmatrix} \quad (14)$$

unless the noise happens to be orthogonal to both \vec{x}_1 and \vec{x}_2 , which is not the case with high probability. Ridge regression also produces a dense estimate. In contrast, the lasso estimate is sparse and correctly identifies the right feature. The value of the coefficients for the ridge-regression and lasso estimators are shown in Figure 1 for $\alpha = 1, 5$ examples ($n = 5$) and $\rho := -0.43$. For large λ both estimators are equal to zero, as the regularization term dominates. For small λ the estimators tend to the least-squares estimators. For intermediate values of λ , the ℓ_1 -norm regularization term promotes a sparse coefficient vector, whereas the ℓ_2 -norm regularization term does not. \triangle

Example 1.5 (Prostate cancer data set). In this example, we apply the lasso to a sparse regression problem related to the study of prostate cancer.¹ The response is the level of prostate-specific antigen (PSA) measured for a specific patient (high levels of PSA are an indication of cancer), whereas the features are characteristics of the patient, including age, weight and other measurements. We fit a sparse linear model to the data using the lasso. The training set contains 60 patients, whereas the test set contains 37 patients. Figure 2 shows the coefficients for different values of the regularization parameter λ . The least-squares estimate ($\lambda \rightarrow 0$) achieves the smallest ℓ_2 error on the training set using all of the features. The lasso estimate with λ between 0.1 and 0.5 incurs in a larger training error but achieves a similar, or better test error, with a smaller number of coefficients (5 instead of 8), suggesting that the 3 remaining features are not related to the response. \triangle

¹The data is available [here](#).

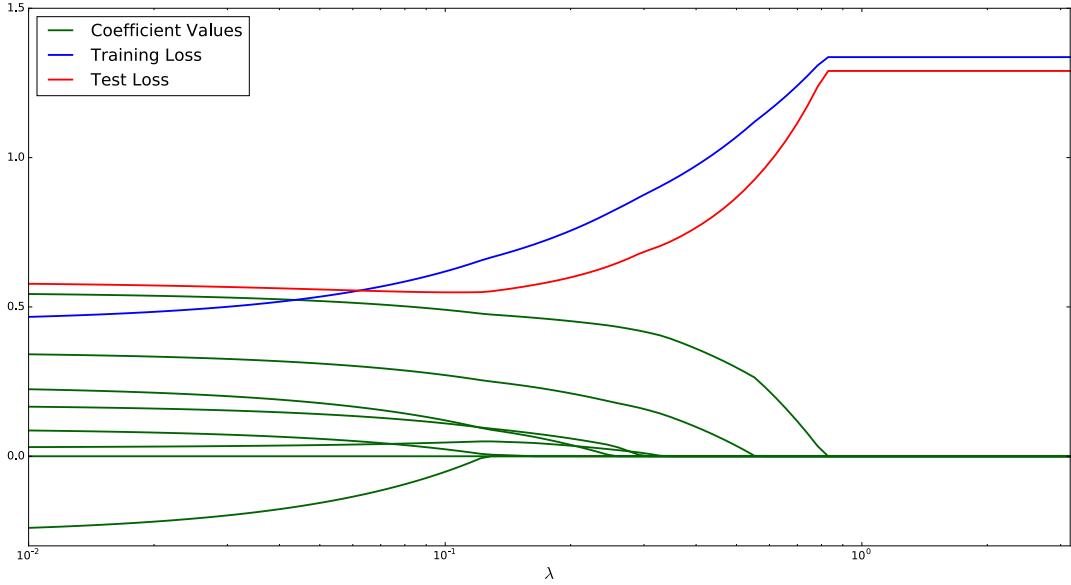


Figure 2: Coefficients, training error and test error of the lasso estimate for different values of the regularization parameter λ when applied to the sparse-regression problem in Example 1.5.

1.2 Robust principal-component analysis

Outliers may severely distort the results of applying principal-component analysis (PCA) to a set of data that lie close to a low-dimensional subspace. Even one outlier can have a significant effect, as illustrated in the following example.

Example 1.6 (PCA with an outlier). A data set contains five examples with three features each. We apply PCA to these data by computing the SVD of the matrix

$$Y := \begin{bmatrix} -2 & -1 & 5 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}. \quad (15)$$

All data points are aligned with the vector $[1 \ 1 \ 1]^T$ except for the third one, due to an outlier that has corrupted one of the features (shown in red). Due of this outlier, the data matrix has rank 2 instead of 1 and the principal directions are not aligned on the line that contains most of the points. Figure 3 shows the data points, as well as the principal directions when the outlier is present and when it is absent. \triangle

This is an example of a general phenomenon where a small number of corrupted entries disrupts low-rank structure in a matrix, making it appear high rank, despite the correlations between columns (or rows). As a result, computing the SVD does not uncover the low-rank component of the data. An alternative is to fit a low rank + sparse model to the data, where the sparse component accounts for the outliers and the low-rank component reveals the underlying correlations.

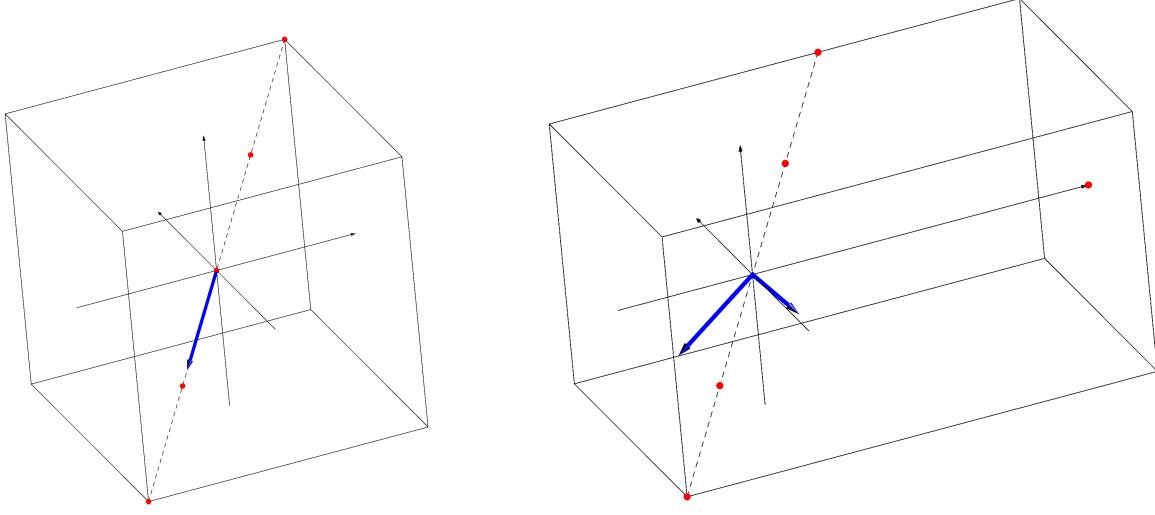


Figure 3: The data points in Example 1.6 are plotted in red. On the left, the data contains no outliers and the principal direction (blue) corresponding to the only nonzero singular value of the SVD of the data matrix is aligned with all the points. On the right, adding the outlier distorts the principal directions (in blue), which are two instead of one because the rank of the matrix increases by one.

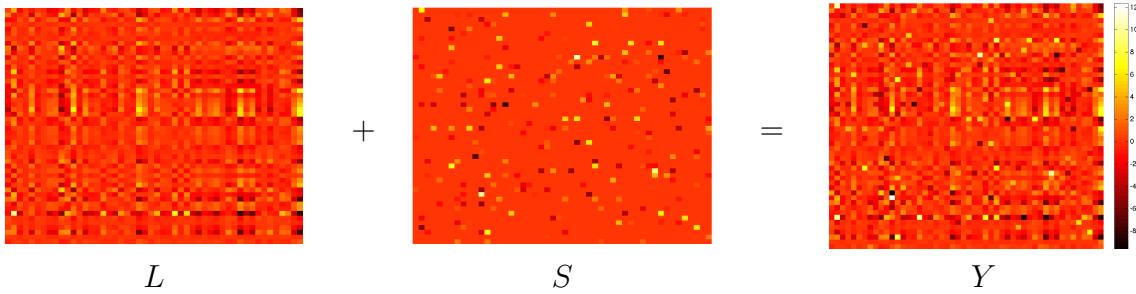


Figure 4: Y is obtained by summing a low-rank matrix L and a sparse matrix S .

Figure 4 shows a simulated example of this model. As illustrated in Examples 1.8 and 1.10, it is usually not tractable to maximize sparsity and minimize rank directly. An alternative that often works well is to penalize the ℓ_1 and nuclear norm respectively. This technique introduced by [3, 5] is often called *robust PCA* (RPCA), since it aims to obtain a low-rank component that is not affected by the presence of outliers.

Algorithm 1.7 (Robust PCA). *For $Y \in \mathbb{R}^{n \times m}$, the robust PCA estimator of the low-rank component in Y is the minimizer of the optimization problem*

$$L_{\text{RPCA}} := \arg \min_L \|L\|_* + \lambda \|Y - L\|_1, \quad (16)$$

where $\lambda > 0$ is a fixed regularization parameter. Here $\|\cdot\|_1$ denotes the sum of absolute values of the entries of the matrix; it is the ℓ_1 norm of the vectorized matrix. $S_{\text{RPCA}} := Y - L_{\text{RPCA}}$ is the estimator of the sparse component.

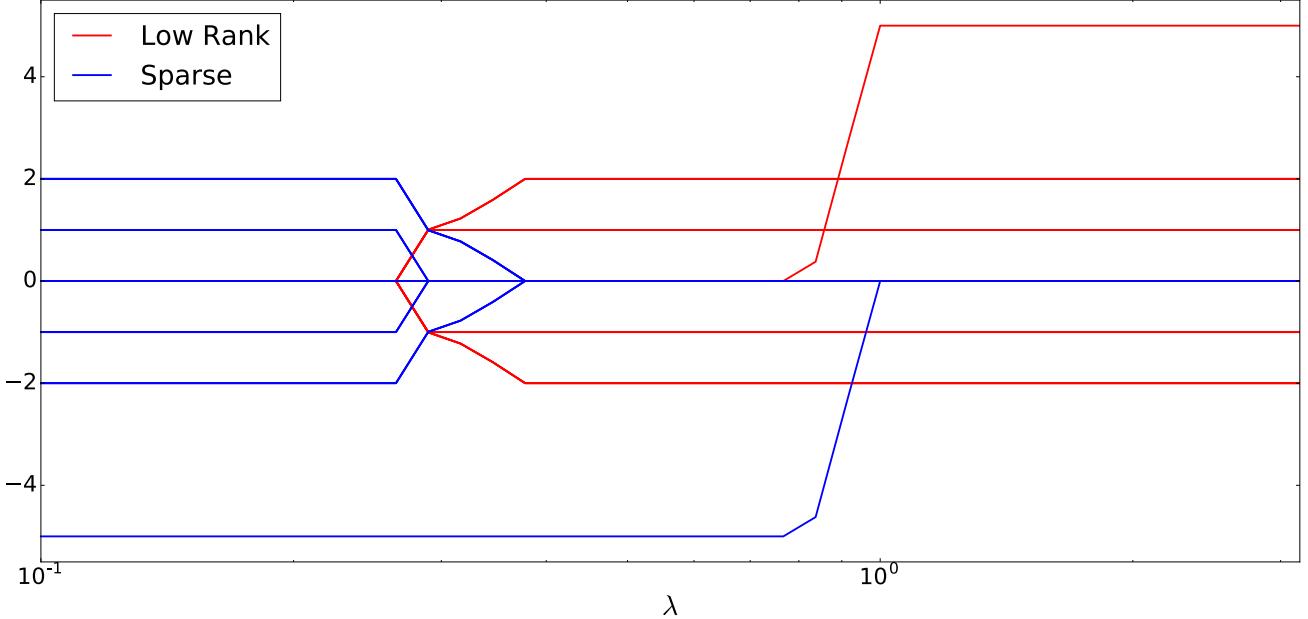


Figure 5: Values of the entries of the low-rank and sparse components for different values of λ computed by applying RPCA to the data in Example 1.6.

Once the low-rank component has been recovered, PCA can be applied to it to determine its principal directions. Figure 5 shows the result of applying RPCA to the data in Example 1.6. If the parameter λ is in certain range, then the low-rank component exactly uncovers the rank-1 structure in the data and the sparse component identifies the outlier. In general, the regularization parameter λ determines the weight of the two structure-inducing terms in the cost function. Figure 6 shows the low-rank and sparse components of the matrix in Figure 4 for different values of λ . If λ is too small, then it is *cheap* to increase the content of the sparse component, which consequently won't be very sparse. Similarly, if λ is too large, then the low-rank component won't be low-rank, as the nuclear-norm term has less influence. Setting λ correctly allows to achieve a perfect decomposition. In practice, the regularization parameter is usually set using cross validation.

Example 1.8 (Background subtraction). In computer vision, the problem of background subtraction is that of separating the background and foreground of a video sequence. In particular we consider a scene with a static background. If we stack the video frames in a matrix Y , where each column corresponds to a vectorized frame, and the background is completely static, then all the frames are equal to a certain vector $\vec{x} \in \mathbb{R}^m$ (m is the number of pixels in each frame) and the matrix is rank 1,

$$Y = [\vec{x} \ \vec{x} \ \cdots \ \vec{x}] = \vec{x} [1 \ 1 \ \cdots \ 1]. \quad (17)$$

If the background is not completely static, but instead experiences gradual changes, then the matrix containing the frames will be approximately low rank. If there are sudden events in the foreground that occupy a small part of the field of view and do not last very long, then this is equivalent to adding a sparse component (most entries are equal to zero) to the low-rank background. The two components can consequently be separated using the robust PCA algorithm. The results of applying this method to a real video sequence are shown in Figure 7. \triangle

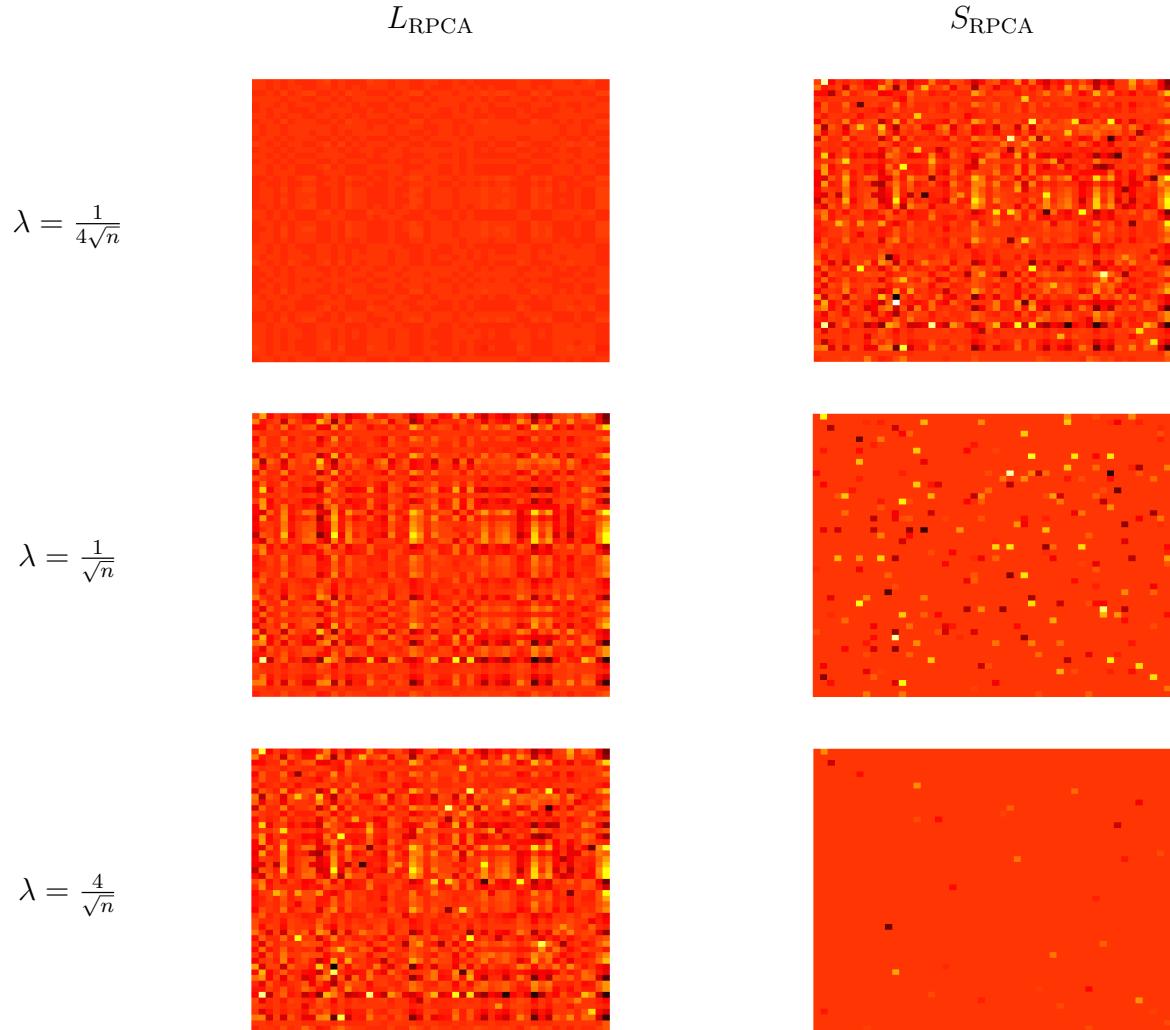


Figure 6: RPCA estimates of the low-rank and sparse components of the matrix in Figure 4 for different values of the regularization parameter. For $\lambda := 1/\sqrt{n}$ the components are recovered perfectly.

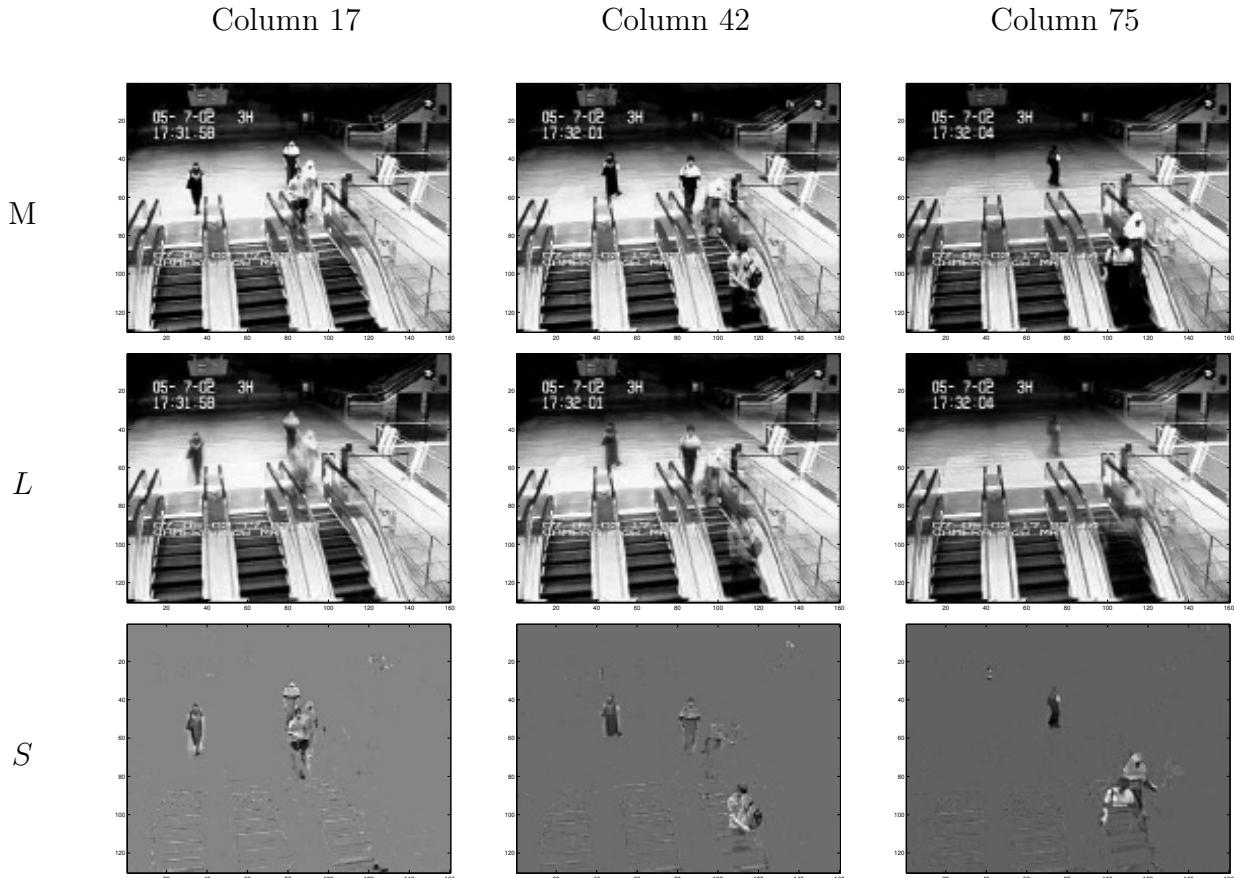


Figure 7: Background subtraction results from a video. This example is due to Stephen Becker. The code is available at <http://cvxr.com/tfocs/demos/rpca>.

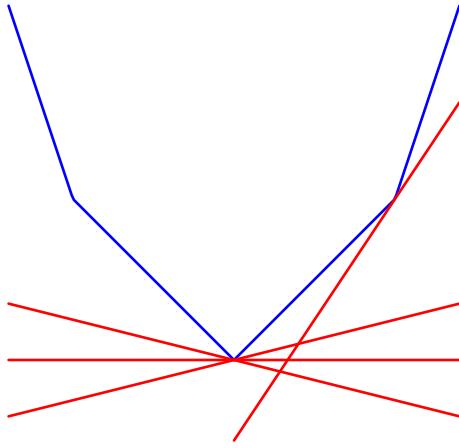


Figure 8: A nondifferentiable convex function (blue). The red supporting lines are specified by subgradients that determine their slope.

2 Subgradients

2.1 Definition and properties

By Theorem 2.5 in Lecture Notes 7, differentiable functions are convex if and only if their epigraph has a supporting hyperplane at every point. In more detail, a differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only at any point $\vec{x} \in \mathbb{R}^n$ there exists a vector \vec{g} such that

$$f(\vec{y}) \geq f(\vec{x}) + \langle \vec{g}, \vec{y} - \vec{x} \rangle \quad (18)$$

for every $\vec{y} \in \mathbb{R}^n$. In the case of differentiable functions, \vec{g} is the gradient of f at \vec{x} . Non-differentiable functions do not have gradients, but the existence of a supporting hyperplane still characterizes convexity. The vector \vec{g} that corresponds to such a hyperplane is called a subgradient.

Definition 2.1 (Subgradient). *The subgradient of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ at $\vec{x} \in \mathbb{R}^n$ is a vector $\vec{g} \in \mathbb{R}^n$ such that*

$$f(\vec{y}) \geq f(\vec{x}) + \vec{g}^T (\vec{y} - \vec{x}), \quad \text{for all } \vec{y} \in \mathbb{R}^n. \quad (19)$$

The set of all subgradients is called the subdifferential of the function at \vec{x} .

Figure 8 shows a one-dimensional nondifferentiable convex function, along with some of the hyperplanes that support its epigraph. The following theorem establishes that a function is convex if and only if a subgradient exists at every point.

Theorem 2.2 (Proof in Section 4.1). *A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if and only if it has a non-empty subdifferential at any $\vec{x} \in \mathbb{R}^n$. It is strictly convex if and only for all $\vec{x} \in \mathbb{R}^n$ there exists a subgradient $\vec{g} \in \mathbb{R}^n$ such that*

$$f(\vec{y}) \geq f(\vec{x}) + \vec{g}^T (\vec{y} - \vec{x}), \quad \text{for all } \vec{y} \in \mathbb{R}^n. \quad (20)$$

Subgradients are a useful tool to characterize the minima of nondifferentiable convex functions.

Theorem 2.3 (Optimality condition). *A convex function attains its minimum value at a vector x if and only if the zero vector is a subgradient of f at x . If the function is strictly convex, then the minimum is unique.*

Proof. By the definition of subgradient, if $\vec{g} := \vec{0}$ is a subgradient at \vec{x} , then for any $\vec{y} \in \mathbb{R}^n$

$$f(\vec{y}) \geq f(\vec{x}) + \vec{g}^T (\vec{y} - \vec{x}) = f(\vec{x}), \quad (21)$$

which is equivalent to \vec{x} being a global minimum of the function. If the function is strictly convex, then the inequality is strict for all $\vec{y} \neq \vec{x}$. \square

A useful property is that the sum of subgradients of two or more functions is a subgradient of their sum.

Lemma 2.4 (Sum of subgradients). *Let \vec{g}_1 and \vec{g}_2 be subgradients at $\vec{x} \in \mathbb{R}^n$ of $f_1 : \mathbb{R}^n \rightarrow \mathbb{R}$ and $f_2 : \mathbb{R}^n \rightarrow \mathbb{R}$ respectively. Then $\vec{g} := \vec{g}_1 + \vec{g}_2$ is a subgradient of $f := f_1 + f_2$ at \vec{x} .*

Proof. For any $\vec{y} \in \mathbb{R}^n$

$$f(\vec{y}) = f_1(\vec{y}) + f_2(\vec{y}) \quad (22)$$

$$\geq f_1(\vec{x}) + \vec{g}_1^T (\vec{y} - \vec{x}) + f_2(\vec{y}) + \vec{g}_2^T (\vec{y} - \vec{x}) \quad (23)$$

$$\geq f(\vec{x}) + \vec{g}^T (\vec{y} - \vec{x}). \quad (24)$$

\square

Another useful property is that the subgradient of a function scaled by a constant can be obtained by scaling the subgradient.

Lemma 2.5 (Subgradient of scaled function). *Let \vec{g}_1 be a subgradient at $\vec{x} \in \mathbb{R}^n$ of $f_1 : \mathbb{R}^n \rightarrow \mathbb{R}$. Then for any nonnegative $\eta \in \mathbb{R}$ $\vec{g}_2 := \eta \vec{g}_1$ is a subgradient of $f_2 := \eta f_1$ at \vec{x} .*

Proof. For any $\vec{y} \in \mathbb{R}^n$

$$f_2(\vec{y}) = \eta f_1(\vec{y}) \quad (25)$$

$$\geq \eta (f_1(\vec{x}) + \vec{g}_1^T (\vec{y} - \vec{x})) \quad (26)$$

$$\geq f_2(\vec{x}) + \vec{g}_2^T (\vec{y} - \vec{x}). \quad (27)$$

\square

2.2 Examples of subdifferentials

If a function is differentiable at a given point, then the gradient is the only subgradient at that point.

Theorem 2.6 (Subdifferential of differentiable functions). *If a convex function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable at $\vec{x} \in \mathbb{R}^n$, then its subdifferential at \vec{x} only contains $\nabla f(\vec{x})$.*

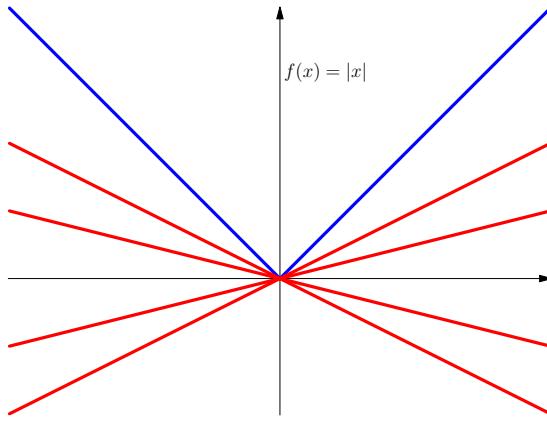


Figure 9: Examples of supporting lines of the absolute value function at the origin. The subgradients at the origin determine the slope of the lines.

Proof. By Theorem 2.5 in Lecture Notes 7 $\nabla f(\vec{x})$ is a subgradient at \vec{x} . Now, let \vec{g} be an arbitrary subgradient at \vec{x} . By the definition of subgradient, for any $1 \leq i \leq n$

$$f(\vec{x} + \alpha \vec{e}_i) \geq f(\vec{x}) + \vec{g}^T \alpha \vec{e}_i \quad (28)$$

$$= f(\vec{x}) + \vec{g}[i] \alpha, \quad (29)$$

$$f(\vec{x}) \geq f(\vec{x} - \alpha \vec{e}_i) + \vec{g}^T \alpha \vec{e}_i \quad (30)$$

$$= f(\vec{x} - \alpha \vec{e}_i) + \vec{g}[i] \alpha, \quad (31)$$

where \vec{e}_i is the i th vector in the standard basis (all its entries are equal to zero, except the i th entry which is equal to one). Combining both inequalities

$$\frac{f(\vec{x}) - f(\vec{x} - \alpha \vec{e}_i)}{\alpha} \leq \vec{g}[i] \leq \frac{f(\vec{x} + \alpha \vec{e}_i) - f(\vec{x})}{\alpha}. \quad (32)$$

If we let $\alpha \rightarrow 0$, this implies $\vec{g}[i] = \frac{\partial f(\vec{x})}{\partial \vec{x}[i]}$. Consequently, $\vec{g} = \nabla f(\vec{x})$. \square

The following lemma characterizes the subdifferential of the absolute value function.

Lemma 2.7 (Subdifferential of absolute value). *The subdifferential of the absolute value function $|\cdot| : \mathbb{R} \rightarrow \mathbb{R}$ at x is equal to $\{\text{sign}(x)\}$ if $x \neq 0$ and to $\{g \in \mathbb{R} \mid |g| \leq 1\}$ if $x = 0$.*

Proof. If $x \neq 0$ the function is differentiable and the only subgradient is equal to the derivative by Theorem 2.6. At $x = 0$, we need

$$|y| = f(0 + y) \quad (33)$$

$$\geq f(0) + g(y - 0) \quad (34)$$

$$\geq gy \quad (35)$$

for all $y \in \mathbb{R}$, which holds if and only if $|g| \leq 1$. \square

As motivated in Section 1.1, the ℓ_1 norm is an important nondifferentiable convex function in data analysis. The following theorem characterizes its subdifferential.

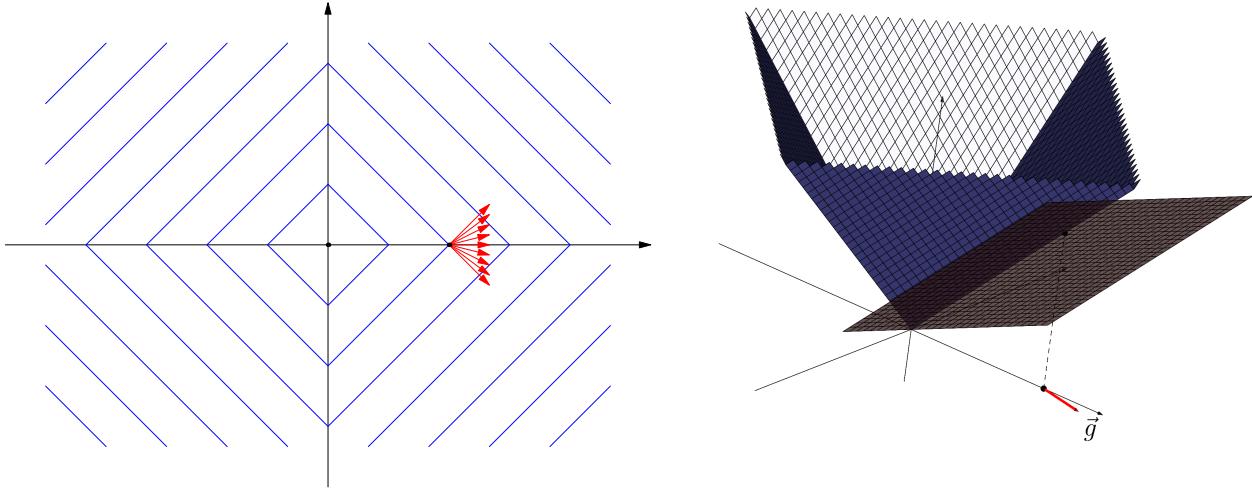


Figure 10: On the left the blue lines are contour lines of the ℓ_1 norm in \mathbb{R}^2 . The red arrows correspond to subgradients at a point where the function is nondifferentiable. On the right the graph of the function is shown in blue, and the supporting hyperplane corresponding to one of the subgradients (denoted by \vec{g} and a red line) is plotted in brown.

Theorem 2.8 (Subdifferential of ℓ_1 norm). *The subdifferential of the ℓ_1 norm at $\vec{x} \in \mathbb{R}^n$ is the set of vectors $\vec{g} \in \mathbb{R}^n$ that satisfy*

$$\vec{g}[i] = \text{sign}(\vec{x}[i]) \quad \text{if } \vec{x}[i] \neq 0, \quad (36)$$

$$|\vec{g}[i]| \leq 1 \quad \text{if } \vec{x}[i] = 0. \quad (37)$$

The theorem is a direct consequence of Lemma 2.7 and the following result.

Lemma 2.9. *The vector $\vec{g} \in \mathbb{R}^n$ is a subgradient of $\|\cdot\|_1 : \mathbb{R}^n \rightarrow \mathbb{R}$ at \vec{x} if and only if $g[i]$ is a subgradient of $|\cdot| : \mathbb{R} \rightarrow \mathbb{R}$ at $\vec{x}[i]$ for all $1 \leq i \leq n$.*

Proof. If \vec{g} is a subgradient of $\|\cdot\|_1$ at \vec{x} then for any $y \in \mathbb{R}$

$$|y| = |\vec{x}[i]| + \|\vec{x} + (y - \vec{x}[i])\vec{e}_i\|_1 - \|\vec{x}\|_1 \quad (38)$$

$$\geq |\vec{x}[i]| + \|\vec{x}\|_1 + \vec{g}^T(y - \vec{x}[i])\vec{e}_i - \|\vec{x}\|_1 \quad (39)$$

$$= |\vec{x}[i]| + \vec{g}[i](y - \vec{x}[i]), \quad (40)$$

so $\vec{g}[i]$ is a subgradient of $|\cdot|$ at $|\vec{x}[i]|$ for any $1 \leq i \leq n$.

If $\vec{g}[i]$ is a subgradient of $|\cdot|$ at $|\vec{x}[i]|$ for $1 \leq i \leq n$ then for any $\vec{y} \in \mathbb{R}^n$

$$\|\vec{y}\|_1 = \sum_{i=1}^n |\vec{y}[i]| \quad (41)$$

$$\geq \sum_{i=1}^n |\vec{x}[i]| + \vec{g}[i](\vec{y}[i] - \vec{x}[i]) \quad (42)$$

$$= \|\vec{x}\|_1 + \vec{g}^T(\vec{y} - \vec{x}) \quad (43)$$

so \vec{g} is a subgradient of $\|\cdot\|_1$ at \vec{x} . □

Another important nondifferentiable convex function in data analysis is the nuclear norm (see Section 1.2). The following theorem characterizes its subdifferential.

Theorem 2.10 (Subdifferential of the nuclear norm). *Let $X \in \mathbb{R}^{m \times n}$ be a rank- r matrix with SVD USV^T , where $U \in \mathbb{R}^{m \times r}$, $V \in \mathbb{R}^{n \times r}$ and $S \in \mathbb{R}^{r \times r}$ contains the nonzero singular values of X . The subdifferential of the nuclear norm at X is the set of matrices of the form*

$$G := UV^T + W \quad (44)$$

where W satisfies

$$\|W\| \leq 1, \quad (45)$$

$$U^T W = 0, \quad (46)$$

$$W V = 0. \quad (47)$$

Proof. We only prove that a matrix of the form (44) is a valid subgradient. For the converse (all subgradients are of this form) see [12]. By Pythagoras' Theorem, for any $\vec{x} \in \mathbb{R}^m$ with unit ℓ_2 norm we have

$$\|\mathcal{P}_{\text{row}(X)} \vec{x}\|_2^2 + \left\| \mathcal{P}_{\text{row}(X)^\perp} \vec{x} \right\|_2^2 = \|\vec{x}\|_2^2 \quad (48)$$

$$= 1. \quad (49)$$

As result, since the rows of UV^T are all in $\text{row}(X)$ and the rows of W are in $\text{row}(X)^\perp$ by Condition (47)

$$\|G\|^2 := \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|G \vec{x}\|_2^2 \quad (50)$$

$$= \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|UV^T \vec{x}\|_2^2 + \|W \vec{x}\|_2^2 \quad (51)$$

$$= \max_{\{\|\vec{x}\|_2=1 \mid \vec{x} \in \mathbb{R}^n\}} \|UV^T \mathcal{P}_{\text{row}(X)} \vec{x}\|_2^2 + \left\| W \mathcal{P}_{\text{row}(X)^\perp} \vec{x} \right\|_2^2 \quad (52)$$

$$\leq \|UV^T\|^2 \|\mathcal{P}_{\text{row}(X)} \vec{x}\|_2^2 + \|W\|^2 \left\| \mathcal{P}_{\text{row}(X)^\perp} \vec{x} \right\|_2^2 \quad (53)$$

$$\leq 1 \quad \text{by condition (45).} \quad (54)$$

Equality (51) follows from Pythagoras' Theorem because the column spaces of U and W are orthogonal by condition (46), which also implies

$$\langle W, X \rangle = 0. \quad (55)$$

By equation (191) in Lecture Notes 2

$$\langle UV^T, X \rangle = \|X\|_* . \quad (56)$$

For any matrix $Y \in \mathbb{R}^{m \times n}$

$$\|Y\|_* \geq \langle G, Y \rangle \quad \text{by (54) and Theorem 2.6 in Lecture Notes 2} \quad (57)$$

$$= \langle G, X \rangle + \langle G, Y - X \rangle \quad (58)$$

$$= \langle UV^T, X \rangle + \langle G, Y - X \rangle \quad \text{by (55)} \quad (59)$$

$$= \|X\|_* + \langle G, Y - X \rangle \quad \text{by (56).} \quad (60)$$

□

2.3 Analysis of the lasso estimator

In this section we derive an exact characterization of the solution to the lasso estimator for Example 1.4. This illustrates how to use the subdifferential of a convex cost function to understand the performance of its minimizer as an estimator.

Lemma 2.11 (Sparse regression with two features). *Assume that $\alpha \geq 0$ and $n \geq 2$. The lasso estimator for the sparse-regression problem in Example 1.4 is of the form*

$$\vec{\beta}_{\text{lasso}} = \begin{bmatrix} \alpha + \vec{x}_1^T \vec{z} - \lambda \\ 0 \end{bmatrix} \quad (61)$$

as long as

$$\frac{|\vec{x}_2^T \vec{z} - \rho \vec{x}_1^T \vec{z}|}{1 - |\rho|} \leq \lambda \leq \alpha + \vec{x}_1^T \vec{z}. \quad (62)$$

Proof. The lasso cost function is strictly convex if $n \geq 2$ and the matrix X is full rank (i.e. $\rho \neq 0$), because the quadratic term corresponds to a positive definite quadratic form. By Theorem 2.3, to establish that $\vec{\beta}_{\text{lasso}}$ is the unique minimizer it suffices to prove that the zero vector is a subgradient of the cost function at $\vec{\beta}_{\text{lasso}}$.

The gradient of the quadratic term

$$q(\vec{\beta}) := \frac{1}{2} \left\| X \vec{\beta} - \vec{y} \right\|_2^2 \quad (63)$$

at $\vec{\beta}_{\text{lasso}}$ equals

$$\nabla q(\vec{\beta}_{\text{lasso}}) = X^T (X \vec{\beta}_{\text{lasso}} - \vec{y}). \quad (64)$$

By Theorem 2.8, if only the first entry of $\vec{\beta}_{\text{lasso}}$ is nonzero and nonnegative, then

$$\vec{g}_{\ell_1} := \begin{bmatrix} 1 \\ \gamma \end{bmatrix} \quad (65)$$

is a subgradient of the ℓ_1 norm at $\vec{\beta}_{\text{lasso}}$ for any $\gamma \in \mathbb{R}$ such that $|\gamma| \leq 1$. By Lemmas 2.4 and 2.5, the sum of $\nabla q(\vec{\beta}_{\text{lasso}})$ and $\lambda \vec{g}_{\ell_1}$ is a subgradient of the lasso cost function at $\vec{\beta}_{\text{lasso}}$. If only the first entry of $\vec{\beta}_{\text{lasso}}$ is nonzero, this subgradient equals

$$\vec{g}_{\text{lasso}} := X^T (X \vec{\beta}_{\text{lasso}} - \vec{y}) + \lambda \begin{bmatrix} 1 \\ \gamma \end{bmatrix} \quad (66)$$

$$= X^T (\vec{\beta}_{\text{lasso}}[1] \vec{x}_1 - \alpha \vec{x}_1 - \vec{z}) + \lambda \begin{bmatrix} 1 \\ \gamma \end{bmatrix} \quad (67)$$

$$= \begin{bmatrix} \vec{x}_1^T (\vec{\beta}_{\text{lasso}}[1] \vec{x}_1 - \alpha \vec{x}_1 - \vec{z}) + \lambda \\ \vec{x}_2^T (\vec{\beta}_{\text{lasso}}[1] \vec{x}_1 - \alpha \vec{x}_1 - \vec{z}) + \lambda \gamma \end{bmatrix} \quad (68)$$

$$= \begin{bmatrix} \vec{\beta}_{\text{lasso}}[1] - \alpha - \vec{x}_1^T \vec{z} + \lambda \\ \rho \vec{\beta}_{\text{lasso}}[1] - \rho \alpha - \vec{x}_2^T \vec{z} + \lambda \gamma \end{bmatrix}. \quad (69)$$

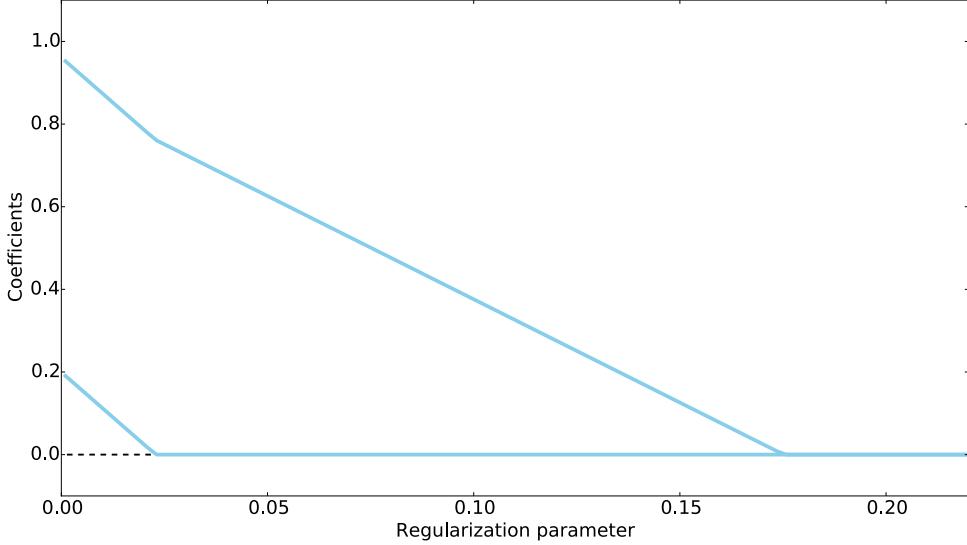


Figure 11: Coefficients of the lasso estimates in the sparse regression problem in Example 1.4 for $\alpha = 1$, 5 examples ($n = 5$), $\rho := -0.43$ and different values of the regularization parameter λ .

The expression is equal to zero if

$$\vec{\beta}_{\text{lasso}}[1] = \alpha + \vec{x}_1^T \vec{z} - \lambda, \quad (70)$$

$$\gamma = \frac{\rho\alpha + \vec{x}_2^T \vec{z} - \rho\vec{\beta}_{\text{lasso}}[1]}{\lambda} \quad (71)$$

$$= \frac{\vec{x}_2^T \vec{z} - \rho\vec{x}_1^T \vec{z}}{\lambda} + \rho. \quad (72)$$

In order to ensure that \vec{g}_{lasso} is a valid subgradient for this choice, we need to check that (1) $\vec{\beta}_{\text{lasso}}[1]$ is indeed nonnegative, which is the case if λ satisfies equation (62), and (2) that $|\gamma| \leq 1$. By the triangle inequality

$$|\gamma| \leq \left| \frac{\vec{x}_2^T \vec{z} - \rho\vec{x}_1^T \vec{z}}{\lambda} \right| + |\rho| \quad (73)$$

$$\leq 1, \quad (74)$$

as long as λ satisfies equation (62). We conclude that $\vec{0}$ is a subgradient of the cost function at $\vec{\beta}_{\text{lasso}}$, which establishes that $\vec{\beta}_{\text{lasso}}$ as given by equation (61) is the unique solution to the optimization problem. \square

The lemma establishes that in this example the lasso estimator detects the relevant feature vector, setting the coefficient of the irrelevant feature vector to zero, for a certain range of λ . Within that range the coefficient corresponding to the relevant predictor scales linearly with λ . This is confirmed in Figure 11.

2.4 Analysis of robust PCA

In this section we analyze the RPCA estimator showing that it succeeds for the data in Example 1.6. This is a cartoon example, but similar arguments can be used to analyze the algorithm in a more general setting [4]. The main idea is to construct a subgradient of the cost function at the ground truth that is equal to zero. This implies that the true low-rank and sparse components are a solution to the problem, but not necessarily the unique solution. The following result shows that if the subgradient satisfies two small additional constraints, then the solution is indeed unique.

Lemma 2.12 (Lemma 2.4 in [4]). *Let $L^*, S^* \in \mathbb{R}^{m \times n}$ and*

$$Y := L^* + S^*. \quad (75)$$

L^* is a rank- r matrix with SVD $U_{L^*}S_{L^*}V_{L^*}^T$, where $U_{L^*} \in \mathbb{R}^{m \times r}$, $V_{L^*} \in \mathbb{R}^{n \times r}$ and $S_{L^*} \in \mathbb{R}^{r \times r}$ contains the nonzero singular values of L^* . Assume there exists a matrix

$$G_* := U_{L^*}V_{L^*}^T + W, \quad (76)$$

where W is a matrix satisfying

$$\|W\| < 1 \quad (77)$$

$$U^T W = 0, \quad (78)$$

$$W V = 0, \quad (79)$$

and there also exists a matrix G_{ℓ_1} satisfying

$$G_{\ell_1}[i, j] = -\text{sign}(S^*[i, j]) \quad \text{if } S^*[i, j] \neq 0, \quad (80)$$

$$|G_{\ell_1}[i, j]| < 1 \quad \text{otherwise}, \quad (81)$$

where $S^* := Y - L^*$, such that

$$G_* + \lambda G_{\ell_1} = 0. \quad (82)$$

Then the solution to the robust PCA problem (16) is unique and equal to L^* .

Proof. By Theorem 2.10 $G_* := U_{L^*}V_{L^*}^T + W$ is a subgradient of the nuclear norm at L^* , whereas by Theorem 2.8 G_{ℓ_1} is a subgradient of $\|\cdot - Y\|_1$ at L^* . As a result by Lemmas 2.4 and 2.5 $G_* + \lambda G_{\ell_1}$ is a subgradient of the RPCA cost function at L^* . By Theorem 2.3 $G_* + \lambda G_{\ell_1} = 0$ consequently implies that L^* is a solution. Uniqueness follows from the strict inequalities (77) and (81). The proof is more involved and can be found in [4]. \square

The following lemma establishes that the RPCA estimator recovers the low-rank and sparse components for the data in Example 1.6 for *any value* of the outlier.

Lemma 2.13. *Let*

$$Y := \begin{bmatrix} -2 & -1 & \alpha & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}. \quad (83)$$

For any value of α the unique solution to the optimization problem

$$\min_L \|L\|_* + \lambda \|Y - L\|_1 \quad (84)$$

is

$$L^* := \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix} \quad (85)$$

as long as

$$\frac{2}{\sqrt{30}} < \lambda < \sqrt{\frac{2}{3}}. \quad (86)$$

Proof. In order to satisfy (80) and (76) at L^* , we set

$$G_* = U_{L^*} V_{L^*}^T + W \quad (87)$$

$$= \frac{1}{\sqrt{30}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \end{bmatrix} + W, \quad (88)$$

$$G_{\ell_1}[1, 3] = -\text{sign}(\alpha). \quad (89)$$

To ensure $G_* + \lambda G_{\ell_1} = 0$ we set

$$W := \lambda \text{sign}(\alpha) \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -0.5 & 0 & 0 \\ 0 & 0 & -0.5 & 0 & 0 \end{bmatrix}, \quad (90)$$

where the entries are chosen so that (78) and (79) both hold, and

$$G_{\ell_1} = \begin{bmatrix} \frac{2}{\lambda\sqrt{30}} & \frac{1}{\lambda\sqrt{30}} & -\text{sign}(\alpha) & -\frac{1}{\lambda\sqrt{30}} & -\frac{2}{\lambda\sqrt{30}} \\ \frac{2}{\lambda\sqrt{30}} & \frac{1}{\lambda\sqrt{30}} & \frac{\text{sign}(\alpha)}{2} & -\frac{1}{\lambda\sqrt{30}} & -\frac{2}{\lambda\sqrt{30}} \\ \frac{2}{\lambda\sqrt{30}} & \frac{1}{\lambda\sqrt{30}} & \frac{\text{sign}(\alpha)}{2} & -\frac{1}{\lambda\sqrt{30}} & -\frac{2}{\lambda\sqrt{30}} \end{bmatrix}. \quad (91)$$

To complete the proof we need to check conditions (77) and (81). We have

$$\|W\|^2 = \frac{3\lambda^2}{2} < 1, \quad \text{if } \lambda < \sqrt{\frac{2}{3}}, \quad (92)$$

$$|G_{\ell_1}[i, j]| \leq \max \left\{ \frac{1}{2}, \frac{2}{\lambda\sqrt{30}} \right\} < 1, \quad \text{if } \lambda > \frac{2}{\sqrt{30}}. \quad (93)$$

□

3 Minimizing nondifferentiable convex functions

3.1 Subgradient method

Consider the optimization problem

$$\text{minimize } f(\vec{x}) \quad (94)$$

where f is convex but nondifferentiable. This implies that we cannot compute a gradient and advance in the steepest descent direction as in gradient descent. However, we can generalize the idea by using subgradients, which exist because f is convex. This is useful as long as it is efficient to compute the subgradient of the function.

Algorithm 3.1 (Subgradient method). *We set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Then we compute*

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - \alpha_k \vec{g}^{(k)}, \quad (95)$$

where $\vec{g}^{(k)}$ is a subgradient of f at $\vec{x}^{(k)}$, until a convergence criterion is satisfied.

Interestingly, the subgradient method is not a descent method. The value of the cost function can actually increase as the iterations progress. However, the method can be shown to converge at a rate of order $\mathcal{O}(1/\epsilon^2)$ as long as the step size decreases along iterations, see [11].

We now apply the subgradient method to solve the lasso problem, i.e. least-squares regression with ℓ_1 -norm regularization. The cost function in the optimization problem,

$$\text{minimize } \frac{1}{2} \|A\vec{x} - \vec{y}\|_2^2 + \lambda \|\vec{x}\|_1, \quad (96)$$

is convex but not differentiable. By Theorem 2.8 $\text{sign}(\vec{x})$ is a subgradient of the ℓ_1 norm at \vec{x} , so

$$\vec{g}^{(k)} = A^T (A\vec{x}^{(k)} - \vec{y}) + \lambda \text{sign}(\vec{x}^{(k)}) \quad (97)$$

is a subgradient of the cost function at $\vec{x}^{(k)}$.

Algorithm 3.2 (Subgradient method for sparse regression). *Set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Update by setting*

$$\vec{x}^{(k+1)} := \vec{x}^{(k)} - \alpha_k (A^T (A\vec{x}^{(k)} - \vec{y}) + \lambda \text{sign}(\vec{x}^{(k)})), \quad (98)$$

where $\alpha_k > 0$ is the step size, until a stopping criterion is met.

Figure 12 shows the result of applying this algorithm to an example in which $A \in \mathbb{R}^{2000 \times 1000}$, $y = A\vec{x}^* + \vec{z}$ where \vec{x}^* is 100-sparse and \vec{z} is iid Gaussian. The example illustrates that decreasing the step size at each iteration achieves faster convergence.

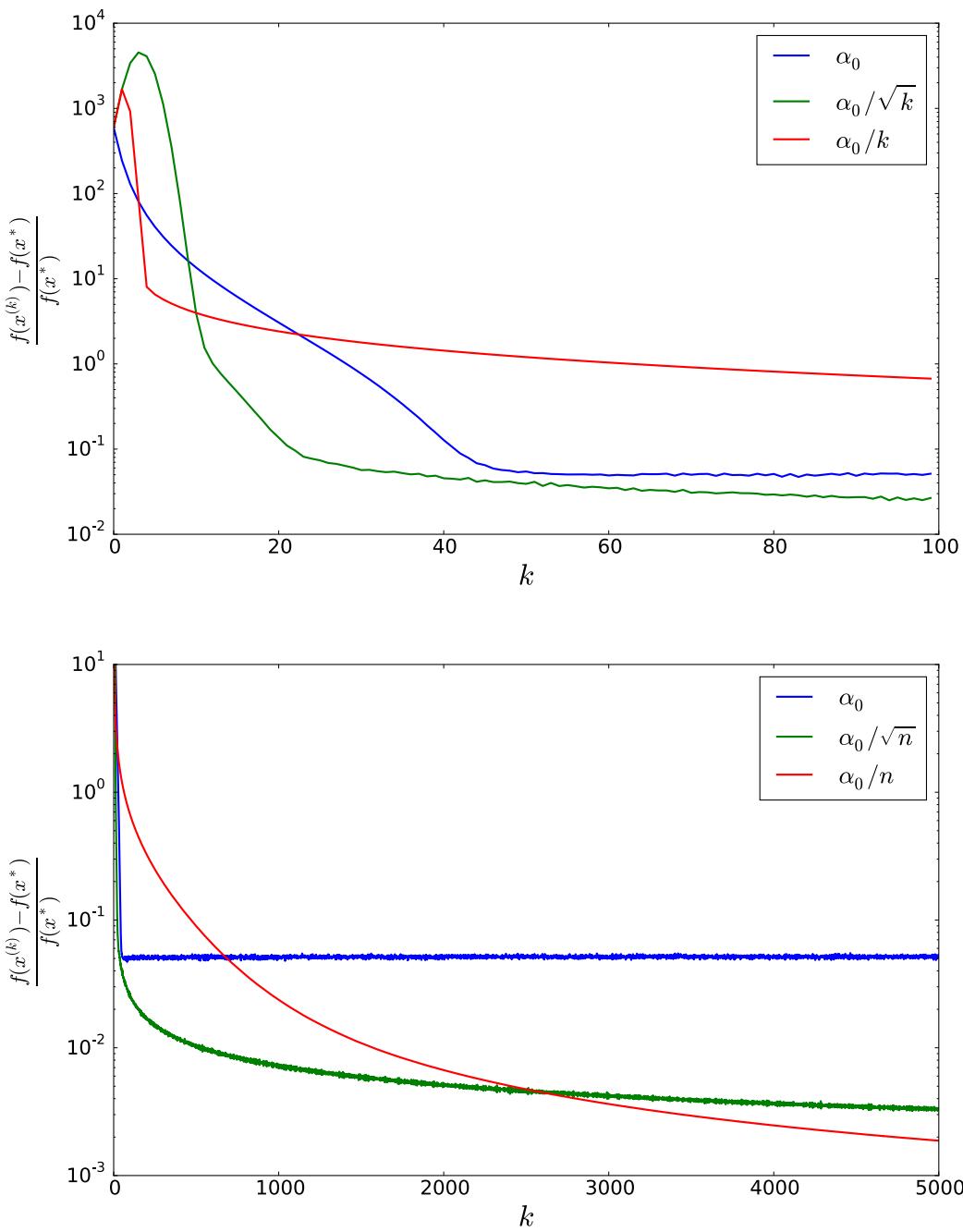


Figure 12: Subgradient method applied to least-squares regression with ℓ_1 -norm regularization for different choices of step size (α_0 is a constant).

3.2 Proximal gradient method

As we saw in the previous section, convergence of subgradient method is slow, both in terms of theoretical guarantees and in the example of Figure 12. In this section we introduce an alternative method that can be applied to a class of functions which is very useful for optimization-based data analysis.

Definition 3.3 (Composite function). *A composite function is a function that can be written as the sum*

$$f(\vec{x}) + h(\vec{x}) \quad (99)$$

where f convex and differentiable and h is convex but not differentiable.

Clearly, the least-squares regression cost function with ℓ_1 -norm regularization is of this form.

In order to motivate proximal methods, let us begin by interpreting the gradient-descent iteration as the solution to a *local* linearization of the function.

Lemma 3.4. *The minimum of the function*

$$h(\vec{x}) := f(\vec{x}^{(k)}) + \nabla f(\vec{x}^{(k)})^T (\vec{x} - \vec{x}^{(k)}) + \frac{1}{2\alpha} \|\vec{x} - \vec{x}^{(k)}\|_2^2 \quad (100)$$

is $\vec{x}^{(k)} - \alpha \nabla f(\vec{x}^{(k)})$.

Proof.

$$\vec{x}^{(k+1)} := \vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)}) \quad (101)$$

$$= \arg \min_{\vec{x}} \|\vec{x} - (\vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)}))\|_2^2 \quad (102)$$

$$= \arg \min_{\vec{x}} f(\vec{x}^{(k)}) + \nabla f(\vec{x}^{(k)})^T (\vec{x} - \vec{x}^{(k)}) + \frac{1}{2\alpha_k} \|\vec{x} - \vec{x}^{(k)}\|_2^2. \quad (103)$$

□

A natural generalization of gradient descent is to minimize the sum of h and the local first-order approximation of f .

$$\vec{x}^{(k+1)} = \arg \min_{\vec{x}} f(\vec{x}^{(k)}) + \nabla f(\vec{x}^{(k)})^T (\vec{x} - \vec{x}^{(k)}) + \frac{1}{2\alpha_k} \|\vec{x} - \vec{x}^{(k)}\|_2^2 + h(\vec{x}) \quad (104)$$

$$= \arg \min_{\vec{x}} \frac{1}{2} \|\vec{x} - (\vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)}))\|_2^2 + \alpha_k h(\vec{x}) \quad (105)$$

$$= \text{prox}_{\alpha_k h}(\vec{x}^{(k)} - \alpha_k \nabla f(\vec{x}^{(k)})). \quad (106)$$

We have written the iteration in terms of the proximal operator of the function h .

Definition 3.5 (Proximal operator). *The proximal operator of a function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ is*

$$\text{prox}_h(\vec{y}) := \arg \min_{\vec{x}} h(\vec{x}) + \frac{1}{2} \|\vec{x} - \vec{y}\|_2^2. \quad (107)$$

Solving the modified local first-order approximation of the composite function iteratively yields the proximal-gradient method, which will be useful if the proximal operator of h can be computed efficiently.

Algorithm 3.6 (Proximal-gradient method). *We set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Then we compute*

$$\vec{x}^{(k+1)} = \text{prox}_{\alpha_k h} (\vec{x}^{(k)} - \alpha_k \nabla f (\vec{x}^{(k)})), \quad (108)$$

until a convergence criterion is satisfied.

This algorithm may be interpreted as a fixed-point method. Indeed, fixed points of the proximal-gradient iteration are a minima of the composite function and vice versa. This suggests applying the iteration repeatedly to minimize the function, although it does not prove convergence (for this we would need to prove that the operator is contractive, see [11]).

Theorem 3.7 (Fixed point of proximal operator). *A vector \vec{x}^* is a solution to*

$$\text{minimize } f(\vec{x}) + h(\vec{x}), \quad (109)$$

if and only if it is a fixed point of the proximal-gradient iteration

$$\vec{x}^* = \text{prox}_{\alpha h} (\vec{x}^* - \alpha \nabla f (\vec{x}^*)) \quad (110)$$

for any $\alpha > 0$.

Proof. \vec{x}^* is a solution to the optimization problem if and only if there exists a subgradient \vec{g} of h at \vec{x}^* such that $\nabla f(\vec{x}^*) + \vec{g} = 0$. \vec{x}^* is the solution to

$$\text{minimize } \alpha h(\vec{x}) + \frac{1}{2} \|\vec{x}^* - \alpha \nabla f(\vec{x}^*) - \vec{x}\|_2^2, \quad (111)$$

which is the case if and only if there exists a subgradient \vec{g} of h at \vec{x}^* such that $\alpha \nabla f(\vec{x}^*) + \alpha \vec{g} = 0$. As long as $\alpha > 0$ the two conditions are equivalent. \square

Proximal methods are very useful for fitting sparse models because the proximal operator of the ℓ_1 norm is very tractable.

Theorem 3.8 (Proximal operator of ℓ_1 norm). *The proximal operator of the ℓ_1 norm weighted by a constant $\alpha > 0$ is the soft-thresholding operator*

$$\text{prox}_{\alpha \|\cdot\|_1} (y) = \mathcal{S}_\alpha (\vec{y}) \quad (112)$$

where

$$\mathcal{S}_\alpha (\vec{y}) [i] := \begin{cases} \vec{y}[i] - \text{sign}(\vec{y}[i]) \alpha & \text{if } |\vec{y}[i]| \geq \alpha, \\ 0 & \text{otherwise.} \end{cases} \quad (113)$$

Proof. Writing the function as a sum,

$$\alpha \|\vec{x}\|_1 + \frac{1}{2} \|\vec{y} - \vec{x}\|_2^2 = \sum_{i=1}^n \alpha |\vec{x}[i]| + \frac{1}{2} (\vec{y}[i] - \vec{x}[i])^2 \quad (114)$$

reveals that it decomposes into independent nonnegative terms. The univariate function

$$h(x) := \alpha |x| + \frac{1}{2} (\vec{y}[i] - x)^2 \quad (115)$$

is strictly convex and consequently has a unique global minimum. It is also differentiable everywhere except at zero. If $x \geq 0$ the derivative is $\lambda + x - \vec{y}[i]$, so if $\vec{y}[i] \geq \alpha$, the minimum is achieved at $\vec{y}[i] - \alpha$. If $\vec{y}[i] < \alpha$ the function is increasing for $x \geq 0$, so the minimizer must be smaller or equal to zero. The derivative for $x < 0$ is $-\alpha + x - \vec{y}[i]$ so the minimum is achieved at $\vec{y}[i] + \alpha$ if $\vec{y}[i] \leq -\alpha$. Otherwise the function is decreasing for all $x < 0$. As a result, if $-\alpha < \vec{y}[i] < \alpha$ the minimum must be at zero. \square

This result yields the following algorithm for least-squares with ℓ_1 -norm regularization.

Algorithm 3.9 (Iterative Shrinkage-Thresholding Algorithm (ISTA)). *We set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Then we compute*

$$\vec{x}^{(k+1)} = \mathcal{S}_{\alpha_k \lambda} (\vec{x}^{(k)} - \alpha_k A^T (A \vec{x}^{(k)} - \vec{y})) , \quad (116)$$

until a convergence criterion is satisfied.

ISTA can be accelerated using a momentum term as in Nesterov's accelerated gradient method. This yields a fast version of the algorithm called FISTA.

Algorithm 3.10 (Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)). *We set the initial point $\vec{x}^{(0)}$ to an arbitrary value in \mathbb{R}^n . Then we compute*

$$\vec{z}^{(0)} = \vec{x}^{(0)} \quad (117)$$

$$\vec{x}^{(k+1)} = \mathcal{S}_{\alpha_k \lambda} (\vec{z}^{(k)} - \alpha_k A^T (A \vec{z}^{(k)} - \vec{y})) , \quad (118)$$

$$\vec{z}^{(k+1)} = \vec{x}^{(k+1)} + \frac{k}{k+3} (\vec{x}^{(k+1)} - \vec{x}^{(k)}) , \quad (119)$$

until a convergence criterion is satisfied.

ISTA and FISTA were proposed by Beck and Teboulle in [1]. ISTA is a descent method. It has the same convergence rate as gradient descent $\mathcal{O}(1/\epsilon)$ both with a constant step size and with a backtracking line search, under the condition that ∇f be L -Lipschitz continuous. FISTA in contrast is not a descent method, but it can be shown to converge in $\mathcal{O}(1/\sqrt{\epsilon})$ to an ϵ -optimal solution.

To illustrate the performance of ISTA and FISTA, we apply them to the same example used in Figure 12. Even without applying a backtracking line search both methods converge to a solution of middle precision (around 10^{-3} or 10^{-4}) much more rapidly than the subgradient method. The results are shown in Figure 13.

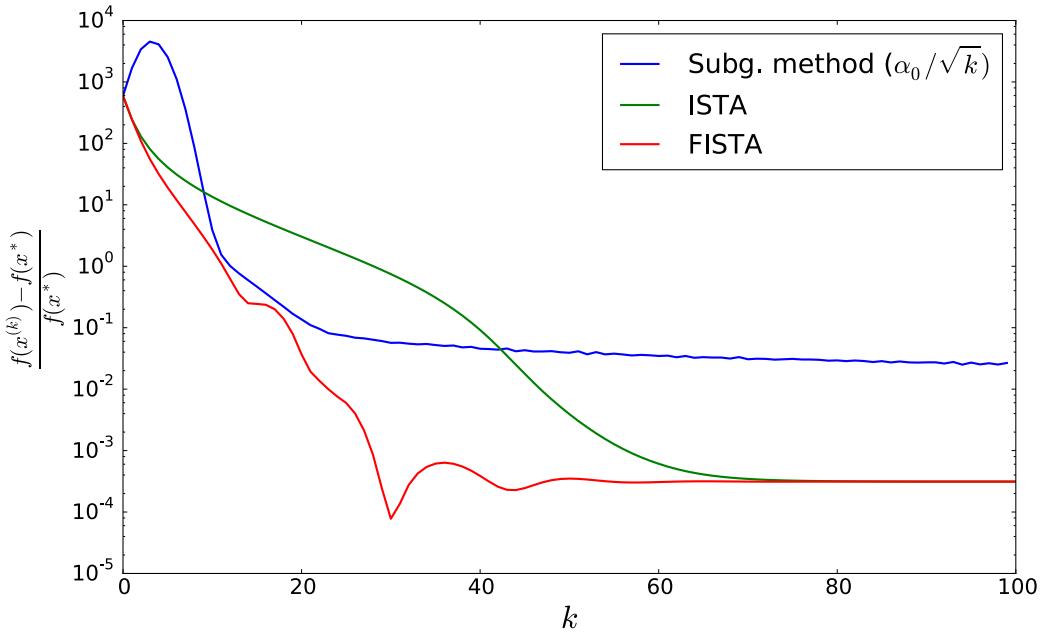


Figure 13: ISTA and FISTA applied to least-squares regression with ℓ_1 -norm regularization.

4 Proofs

4.1 Proof of Theorem 2.2

We prove the statement about convexity. The statement about strict convexity can be proved in a similar way.

The epigraph of a convex function is a convex set, meaning that it contains the line between any of its points. As a consequence of the separating-hyperplane theorem, which states that there is a separating hyperplane between any two disjoint convex sets (we omit the proof which can be found in any text on convex analysis), such sets have a supporting hyperplane at every point. This establishes that convex functions defined on \mathbb{R}^n have a subgradient at every point.

Now assume that a function has a subgradient at every point. Then for any $\vec{x}, \vec{y} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$ there exists a subgradient \vec{g} of f at $\alpha\vec{x} + (1 - \alpha)\vec{y}$. This implies

$$f(\vec{y}) \geq f(\alpha\vec{x} + (1 - \alpha)\vec{y}) + \vec{g}^T(\vec{y} - \alpha\vec{x} - (1 - \alpha)\vec{y}) \quad (120)$$

$$= f(\alpha\vec{x} + (1 - \alpha)\vec{y}) + \alpha\vec{g}^T(\vec{y} - \vec{x}), \quad (121)$$

$$f(\vec{x}) \geq f(\alpha\vec{x} + (1 - \alpha)\vec{y}) + \vec{g}^T(\vec{x} - \alpha\vec{x} - (1 - \alpha)\vec{y}) \quad (122)$$

$$= f(\alpha\vec{x} + (1 - \alpha)\vec{y}) + (1 - \alpha)\vec{g}^T(\vec{y} - \vec{x}). \quad (123)$$

Multiplying equation (121) by $1 - \alpha$ and equation (123) by α and adding them together yields

$$\alpha f(\vec{x}) + (1 - \alpha) f(\vec{y}) \geq f(\alpha\vec{x} + (1 - \alpha)\vec{y}). \quad (124)$$

We conclude that the function is convex.

References

A very readable and exhaustive reference on convex optimization is Boyd and Vandenberghe's seminal book [2], which unfortunately does not cover subgradients.² Nesterov's book [7] and Rockafellar's book [8] do cover subgradients. Chapter 5 of [6] in Hastie, Tibshirani and Wainwright is a great description of proximal-gradient methods, as well as alternative first-order techniques such as coordinate descent, and their application to sparse regression.

- [1] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.
- [2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [3] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.
- [4] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.
- [5] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.
- [6] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC Press, 2015.
- [7] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.
- [8] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*. Springer Science & Business Media, 2009.
- [9] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- [10] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.
- [11] L. Vandenberghe. Notes on optimization methods for large-scale systems.
- [12] G. A. Watson. Characterization of the subdifferential of some matrix norms. *Linear algebra and its applications*, 170:33–45, 1992.

²However, see http://see.stanford.edu/materials/lsocoee364b/01-subgradients_notes.pdf

Lecture Notes 9:

Constrained Optimization

1 Compressed sensing

1.1 Underdetermined linear inverse problems

Linear inverse problems model measurements of the form

$$A\vec{x} = \vec{y} \quad (1)$$

where the data $\vec{y} \in \mathbb{R}^n$ are the result of applying a linear operator represented by the matrix $A \in \mathbb{R}^{m \times n}$ to a signal $\vec{x} \in \mathbb{R}^m$. The aim is to recover \vec{x} from \vec{y} , assuming we know A . Mathematically, this is exactly equivalent to the linear-regression problem discussed in Lecture Notes 6. The difference is that in linear regression the matrix consists of measured features, whereas in inverse problems the linear operator usually has a physical interpretation. For example, in imaging problems the operator depends on the optical system used to obtain the data.

Each entry of \vec{y} can be interpreted as a separate measurement of \vec{x}

$$\vec{y}[i] = \langle A_{i:}, \vec{x} \rangle, \quad 1 \leq i \leq n, \quad (2)$$

where $A_{i:}$ is the i th row of A . In many applications, it is desirable to reduce the number of measurements as much as possible. However, by basic linear algebra, the number of measurements must be at least equal to m . If $m > n$ the system of equations (1) is underdetermined. Even if A is full rank, its null space has dimension $m - n$ by Corollary 1.16 in Lecture Notes 2. Any signal of the form $\vec{x} + \vec{w}$ where \vec{w} belongs to the null space of A is a solution to the system.

As we discussed in Lecture Notes 4 and 5, natural images, speech and other signals are often *compressible*: they can be represented as sparse combinations of predefined atoms such as sinusoids or wavelets. The goal of compressed sensing is to exploit the compressibility of signals in order to reconstruct them from a smaller number of measurements. The idea is that although it is impossible to recover an arbitrary m -dimensional signal from n measurements if $m > n$, it may be possible to recover an m -dimensional signal that is parametrized by an s -dimensional vector, as long as $s < n$. The simplest example of compressible structure is sparsity. We will mostly focus on this case to illustrate the main ideas behind compressed sensing.

Example 1.1 (Compressed sensing in magnetic-resonance imaging). Magnetic resonance imaging (MRI) is a popular medical-imaging technique that measures the response of the atomic nuclei of body tissues to high-frequency radio waves when placed in a strong magnetic field. MRI measurements can be modeled as samples from the 2D or 3D Fourier transform of the object that is being imaged, for example a slice of a human brain. An estimate of the corresponding image

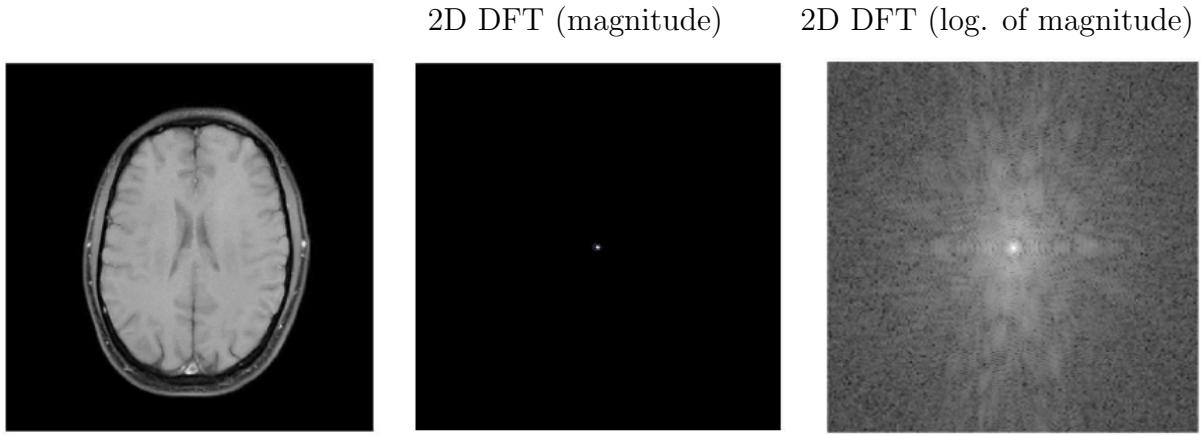


Figure 1: Image of a brain obtained by MRI, along with the magnitude of its 2D discrete Fourier transform (DFT) and the logarithm of this magnitude.

can be obtained by computing the inverse Fourier transform of the data, as shown in Figure 1. An important challenge in MRI is to reduce measurement time: long acquisition times are expensive and bothersome for the patients, especially for those that are seriously ill and for infants. Gathering less measurements, or equivalently undersampling the 2D or 3D Fourier transform of the image of interest, results in shorter data-acquisition times, but poses the challenge of recovering the image from undersampled data. Fortunately, MR images tend to be compressible in the wavelet domain. Compressed sensing of MR images consists of recovering the sparse wavelet coefficients from a small number of Fourier measurements. \triangle

Example 1.2 (1D subsampled Fourier measurements). This cartoon example is inspired by compressed sensing in MRI. We consider the problem of recovering a sparse signal from undersampled Fourier data. The rows of the measurement matrix are a subset of the rows of a DFT matrix, extracted following two strategies: regular and random subsampling. In regular subsampling we select the odd rows of the matrix, whereas in random subsampling we just select the rows uniformly at random. Figure 2 shows the real part of the matrices. Figure 3 shows the underdetermined linear system corresponding to each of the subsampling strategies for a simple example where the signal has sparsity 3. \triangle

1.2 When is sparse estimation well posed?

A first question that arises when we consider sparse recovery from underdetermined measurements is under what conditions the problem is well posed. In other words, is it possible that there may be other sparse signals that produce the same measurements? If that is the case, it is impossible to determine which sparse signal actually generated the data and the problem is ill posed. Whether this situation may arise or not depends on the spark of the measurement matrix.

Definition 1.3 (Spark). *The spark of a matrix is the smallest subset of columns that is linearly dependent.*

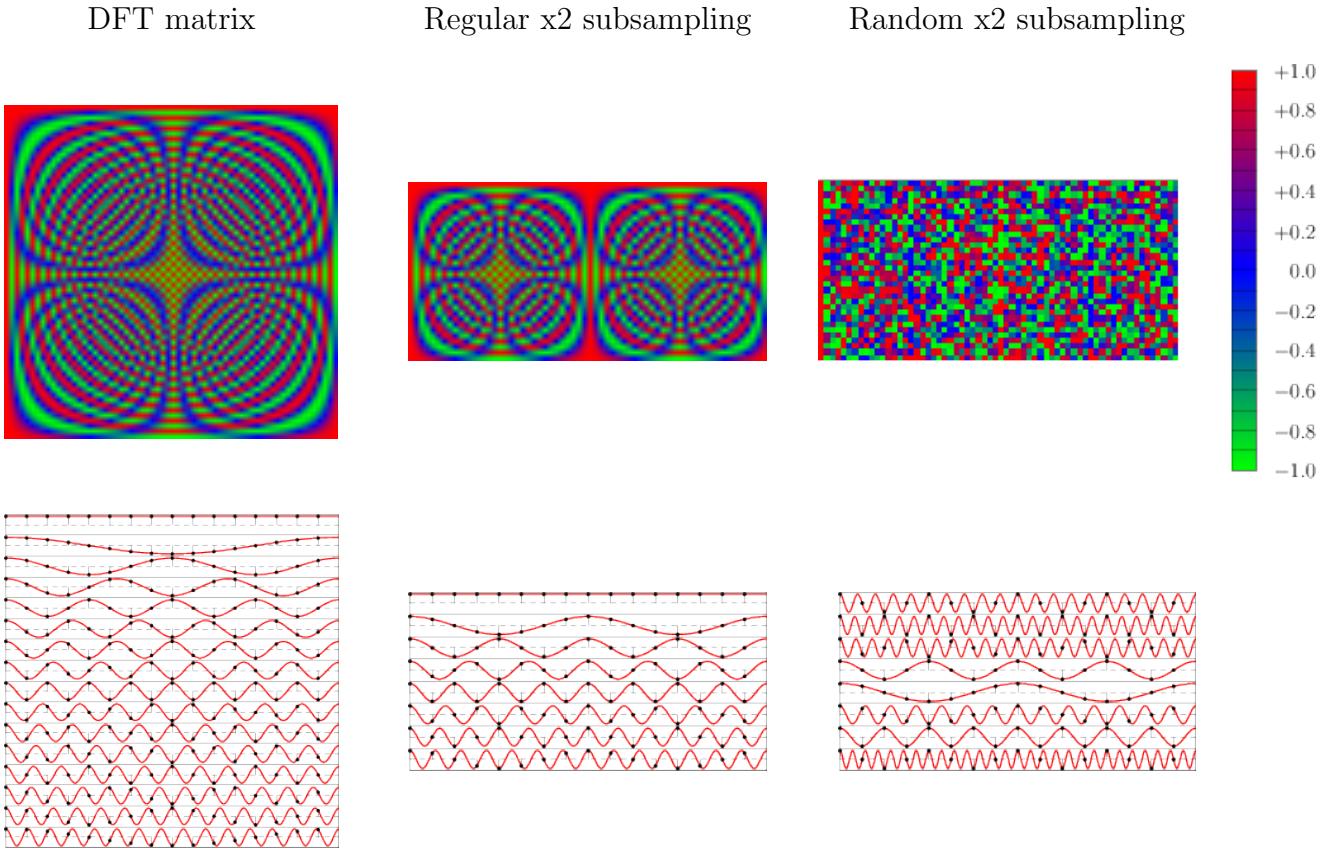


Figure 2: Real part of the DFT matrix, as well as the corresponding regularly-subsampled and randomly-subsampled measurement matrix, represented as a heat map (above) and as samples from continuous sinusoids (below).

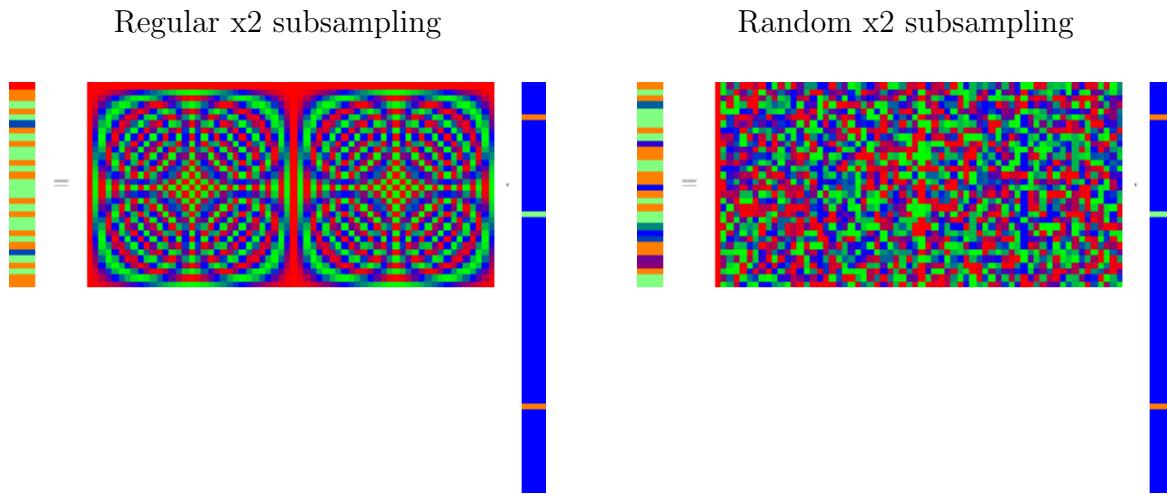


Figure 3: Underdetermined linear system of equations corresponding to the subsampled Fourier matrices in Figure 2.

The spark sets a fundamental limit to the sparsity of vectors that can be recovered uniquely from linear measurements.

Theorem 1.4. *Let $\vec{y} := A\vec{x}^*$, where $A \in \mathbb{R}^{m \times n}$, $\vec{y} \in \mathbb{R}^n$ and $\vec{x}^* \in \mathbb{R}^m$ is a sparse vector with s nonzero entries. The vector \vec{x}^* is guaranteed to be the only vector with sparsity level equal to s consistent with the measurements, i.e. the solution of*

$$\min_{\vec{x}} \|\vec{x}\|_0 \quad \text{subject to} \quad A\vec{x} = \vec{y}, \quad (3)$$

for any choice of \vec{x}^* if and only if

$$\text{spark}(A) > 2s. \quad (4)$$

Proof. \vec{x}^* is the only sparse vector consistent with the data if and only if there is no other vector \vec{x}' with sparsity s such that $A\vec{x}^* = A\vec{x}'$. This occurs for any choice of \vec{x}^* if and only if for any pair of vectors \vec{x}_1 and \vec{x}_2 with sparsity level s , we have

$$A(\vec{x}_1 - \vec{x}_2) \neq \vec{0}. \quad (5)$$

Let T_1 and T_2 denote the support of the nonzero entries of \vec{x}_1 and \vec{x}_2 . Equation (5) can be written as

$$A_{T_1 \cup T_2} \vec{\alpha} \neq \vec{0} \quad \text{for any } \vec{\alpha} \in \mathbb{R}^{|T_1 \cup T_2}|. \quad (6)$$

This is equivalent to all submatrices with at most $2s$ columns (the difference between 2 s -sparse vectors can have at most $2s$ nonzero entries) having no nonzero vectors in their null space and therefore being full rank, which is exactly the meaning of $\text{spark}(A) > 2s$. \square

If the spark of a matrix is greater than $2s$ then the matrix represents a linear operator that is invertible when restricted to act upon s -sparse signals. However, it may still be the case that two different sparse vectors could generate data that are extremely close, which would make it challenging to distinguish them if the measurements are noisy. In order to ensure that stable inversion is possible, we must in addition require that the *distance* between sparse vectors is preserved, so that if \vec{x}_1 is far from \vec{x}_2 then $A\vec{x}_1$ is guaranteed to be far from $A\vec{x}_2$. Mathematically, the linear operator should be an isometry when restricted to act upon sparse vectors.

Definition 1.5 (Restricted-isometry property). *A matrix A satisfies the restricted-isometry property with constant κ_s if for any s -sparse vector \vec{x}*

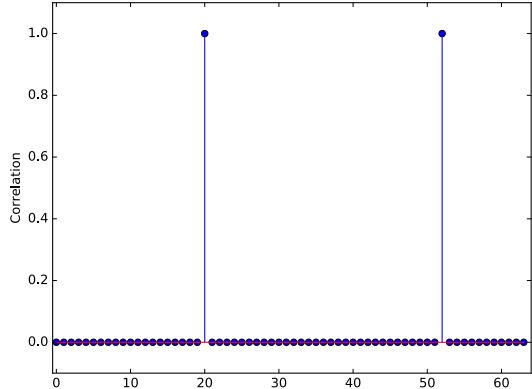
$$(1 - \kappa_s) \|\vec{x}\|_2 \leq \|A\vec{x}\|_2 \leq (1 + \kappa_s) \|\vec{x}\|_2. \quad (7)$$

If a matrix A satisfies the restricted-isometry property (RIP) for a sparsity level $2s$ then for any pair of vectors \vec{x}_1 and \vec{x}_2 with sparsity level s , the distance between their corresponding measurements \vec{y}_1 and \vec{y}_2 is lower bounded by the difference between the two vectors

$$\|\vec{y}_2 - \vec{y}_1\|_2 = \|A(\vec{x}_1 - \vec{x}_2)\|_2 \quad (8)$$

$$\geq (1 - \kappa_{2s}) \|\vec{x}_2 - \vec{x}_1\|_2. \quad (9)$$

Regular x2 subsampling



Random x2 subsampling

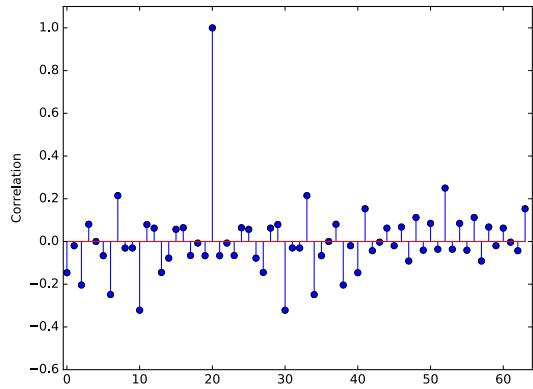
**Figure 4:** Correlation between the 20th column and the rest of the columns for the matrices described in Example 1.2.

Figure 4 shows the correlation between one of the columns in the matrices described in Example 1.2 and the rest of the columns. For the regularly-subsampled Fourier matrix, there exists another column that is exactly the same. No method will be able to distinguish the data corresponding to even 1-sparse vectors, since the contributions of these two columns will be impossible to distinguish. The matrix does not even satisfy the RIP for a sparsity level equal to two.

In the case of the randomly-subsampled Fourier matrix, column 20 is not highly correlated with any other column. This does not immediately mean that the matrix satisfies the restricted-isometry property. Unfortunately, verifying that a matrix satisfies the spark or the restricted-isometry property is not computationally tractable (essentially, one has to check all possible sparse submatrices). However, we can prove that the RIP holds with high probability for random matrices. In the following theorem we prove this statement for Gaussian iid matrices. The proof for random Fourier measurements is more complicated [8, 10].

Theorem 1.6 (Restricted-isometry property for Gaussian matrices). *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a random matrix with iid standard Gaussian entries. $\frac{1}{\sqrt{m}}\mathbf{A}$ satisfies the restricted-isometry property for a constant κ_s with probability $1 - \frac{C_2}{n}$ as long as the number of measurements*

$$m \geq \frac{C_1 s}{\kappa_s^2} \log \left(\frac{n}{s} \right) \quad (10)$$

for two fixed constants $C_1, C_2 > 0$.

Proof. Let us fix an arbitrary support T of size s . The $m \times s$ submatrix \mathbf{A}_T of \mathbf{A} that contains the columns indexed by T has iid Gaussian entries, so by Theorem 3.7 in Lecture Notes 3 (in particular equation (81)), its singular values are bounded by

$$\sqrt{m}(1 - \kappa_s) \leq \sigma_s \leq \sigma_1 \leq \sqrt{m}(1 + \kappa_s) \quad (11)$$

with probability at least

$$1 - 2 \left(\frac{12}{\kappa_s} \right)^s \exp \left(- \frac{m\kappa_s^2}{32} \right). \quad (12)$$

This implies that for any vector \vec{x} with support T

$$\sqrt{1 - \kappa_s} \|\vec{x}\|_2 \leq \frac{1}{\sqrt{m}} \|\mathbf{A}\vec{x}\|_2 \leq \sqrt{1 + \kappa_s} \|\vec{x}\|_2. \quad (13)$$

This is not enough for our purposes, we need this to hold for *all* supports of size s , i.e. on all possible combinations of s columns selected from the n columns in \mathbf{A} . A simple bound on the binomial coefficient yields the following bound on the number of such combinations

$$\binom{n}{s} \leq \left(\frac{en}{s} \right)^s. \quad (14)$$

By the union bound (Theorem 3.4 in Lecture Notes 3), we consequently have that the bounds (13) hold for any sparse- s vector with probability at least

$$\begin{aligned} 1 - 2 \left(\frac{en}{s} \right)^s \left(\frac{12}{\kappa_s} \right)^s \exp \left(- \frac{m\kappa_s^2}{32} \right) &= 1 - \exp \left(\log 2 + s + s \log \left(\frac{n}{s} \right) + s \log \left(\frac{12}{\kappa_s} \right) - \frac{m\kappa_s^2}{2} \right) \\ &\leq 1 - \frac{C_2}{n} \end{aligned} \quad (15)$$

for some constant C_2 as long as m satisfies (10). \square

1.3 Sparse recovery via ℓ_1 -norm minimization

Choosing the sparsest vector consistent with the available data is computationally intractable, due to the nonconvexity of the ℓ_0 “norm” ball. Instead, we can minimize the ℓ_1 norm in order to promote sparse solutions.

Algorithm 1.7 (Sparse recovery via ℓ_1 -norm minimization). *Given data $\vec{y} \in \mathbb{R}^n$ and a matrix $A \in \mathbb{R}^{m \times n}$, the minimum- ℓ_1 -norm estimate is the solution to the optimization problem*

$$\min_{\vec{x}} \|\vec{x}\|_1 \quad \text{subject to} \quad A\vec{x} = \vec{y}. \quad (16)$$

Figure 5 shows the minimum ℓ_2 - and ℓ_1 -norm estimates of the sparse vector in the sparse recovery problem described in Example 1.2. In the case of the regularly-subsampled matrix, both methods yield erroneous solutions that are sparse. As discussed previously, for that matrix the sparse-recovery problem is ill posed. In the case of the randomly-subsampled matrix, ℓ_2 -norm minimization promotes a solution that contains a lot of small entries. The reason is that large entries are very expensive because we are minimizing the square of the magnitudes. Such large entries are not as expensive for the ℓ_1 -norm cost function. As a result, the algorithm produces a sparse solution that is exactly equal to the original signal. Figure 6 provides some geometric intuition as to why the ℓ_1 -norm minimization problem promotes sparse solutions.

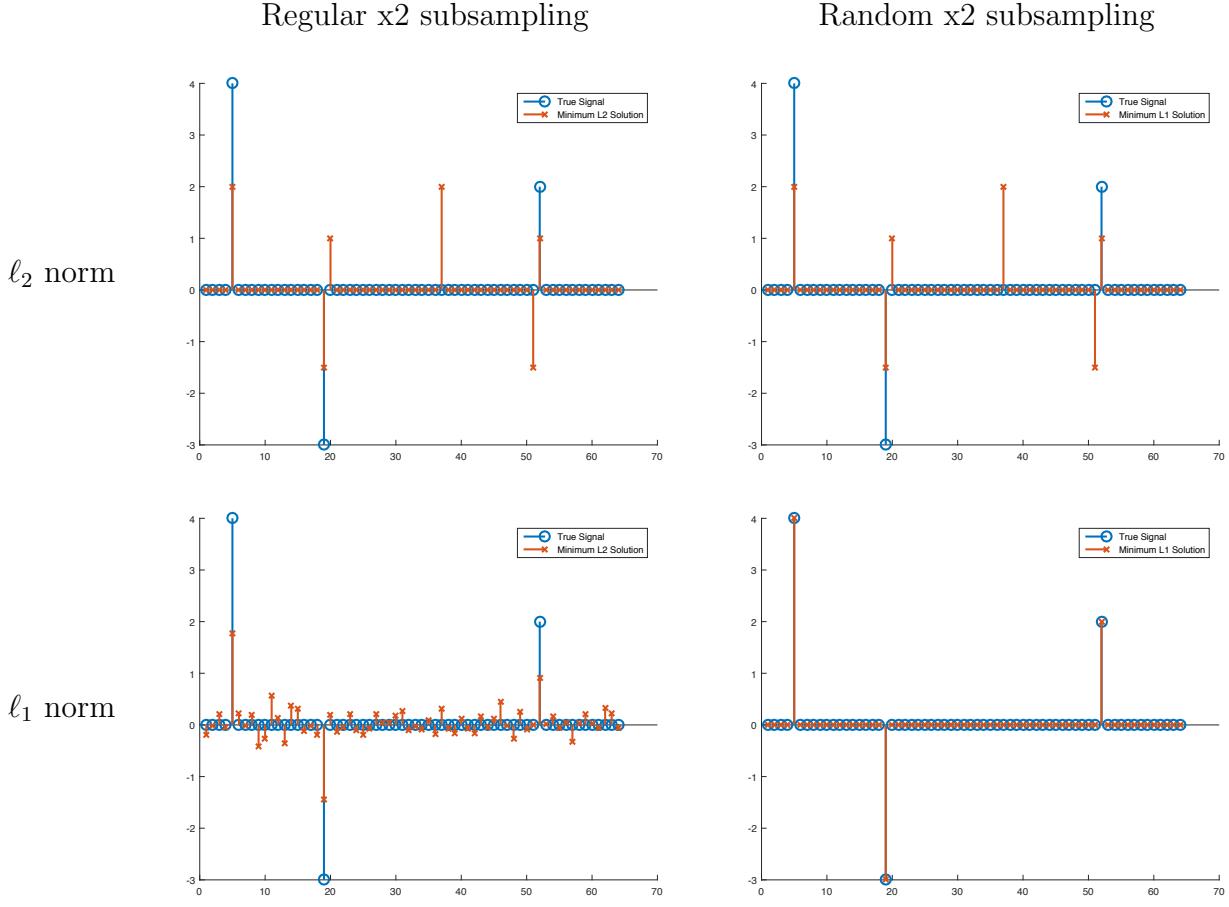


Figure 5: Minimum ℓ_2 - and ℓ_1 -norm estimates of the sparse vector in the sparse recovery problem described in Example 1.2.

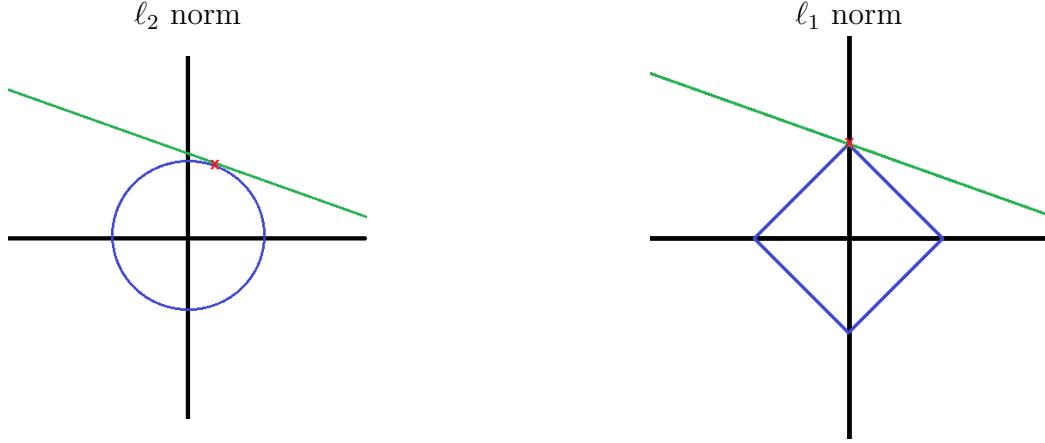


Figure 6: Cartoon of the ℓ_2 - and ℓ_1 -norm minimization problems for a two-dimensional signal. The lines represent the hyperplane of signals such that $A\vec{x} = \vec{y}$. The ℓ_1 -norm ball is spikier, so that as a result the solution lies on a low-dimensional face of the norm ball. In contrast, the ℓ_2 -norm ball is rounded and this does not occur.

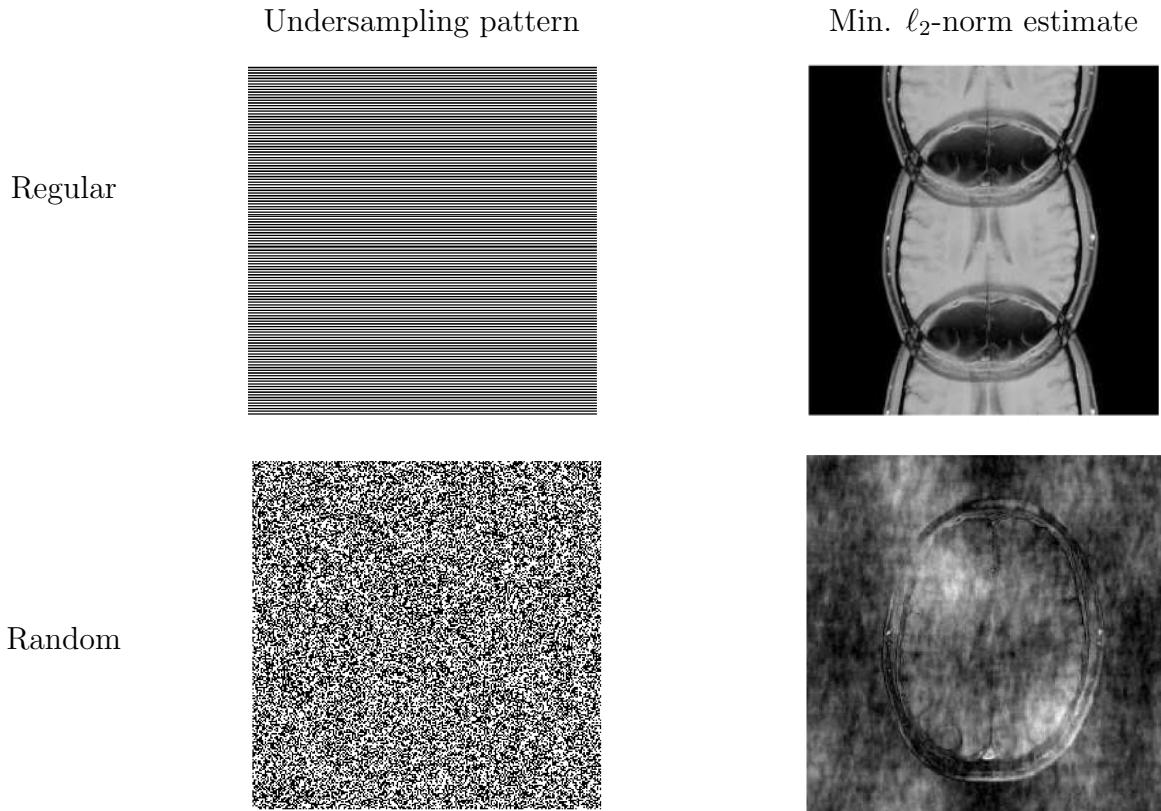


Figure 7: Two different sampling strategies in 2D k space: regular undersampling in one direction (top) and random undersampling (bottom). The original data is the same as in Figure 1. On the right we see the corresponding minimum- ℓ_2 -norm estimate for each undersampling pattern.



Figure 8: Minimum- ℓ_1 -norm for the two undersampling patterns shown in Figure 7.

1.4 Sparsity in a transform domain

If the signal is sparse in a transform domain, then we can modify the optimization problem to take this into account. Let W represent a wavelet transform, such that we assume that the corresponding wavelet coefficients of the image are sparse. In that case, we solve the optimization problem,

$$\min_{\vec{c}} \|\vec{c}\|_1 \quad \text{subject to} \quad AW\vec{c} = \vec{y}. \quad (17)$$

If we want to recover the original \vec{c}^* then we would need to verify that AW should satisfy the RIP, which would require analyzing the inner products between the rows of the measurement A (the measurement vectors) and the columns of W (the sparsifying basis functions). However, we might be fine with any \vec{c}' such that $A\vec{c}' = \vec{x}^*$. In that case, characterizing when the problem is well posed is more challenging.

Figure 8 shows the result of applying ℓ_1 -norm minimization to recover an image from the data corresponding to the images shown in Figure 7. For regular undersampling, then the estimate is essentially the same as the minimum- ℓ_2 -norm estimate. This is not surprising, since the minimum- ℓ_2 -norm estimate is also sparse in the wavelet domain because it is equal to a superposition of two shifted copies of the image. In contrast, ℓ_1 -norm minimization recovers the original image perfectly when coupled with random projections. Intuitively, ℓ_1 -norm minimization *cleans up* the noisy aliasing caused by random undersampling.

2 Constrained optimization

2.1 Convex sets

A set is convex if it contains all segments connecting points that belong to it.

Definition 2.1 (Convex set). *A convex set \mathcal{S} is any set such that for any $\vec{x}, \vec{y} \in \mathcal{S}$ and $\theta \in (0, 1)$*

$$\theta\vec{x} + (1 - \theta)\vec{y} \in \mathcal{S}. \quad (18)$$

Figure 9 shows a simple example of a convex and a nonconvex set.

The following lemma establishes that the intersection of convex sets is convex.

Lemma 2.2 (Intersection of convex sets). *Let $\mathcal{S}_1, \dots, \mathcal{S}_m$ be convex subsets of \mathbb{R}^n , $\cap_{i=1}^m \mathcal{S}_i$ is convex.*

Proof. Any $\vec{x}, \vec{y} \in \cap_{i=1}^m \mathcal{S}_i$ also belong to S_1 . By convexity of S_1 $\theta\vec{x} + (1 - \theta)\vec{y}$ belongs to S_1 for any $\theta \in (0, 1)$ and therefore also to $\cap_{i=1}^m \mathcal{S}_i$. \square

The following theorem shows that projection onto non-empty closed convex sets is unique.

Theorem 2.3 (Projection onto convex set). *Let $\mathcal{S} \subseteq \mathbb{R}^n$ be a non-empty closed convex set. The projection of any vector $v\vec{x} \in \mathbb{R}^n$ onto \mathcal{S}*

$$\mathcal{P}_{\mathcal{S}}(\vec{x}) := \arg \min_{\vec{y} \in \mathcal{S}} \|\vec{x} - \vec{y}\|_2 \quad (19)$$

exists and is unique.

Proof. Existence

Since \mathcal{S} is non-empty we can choose an arbitrary point $\vec{y}' \in \mathcal{S}$. Minimizing $\|\vec{x} - \vec{y}\|_2$ over \mathcal{S} is equivalent to minimizing $\|\vec{x} - \vec{y}\|_2$ over $\mathcal{S} \cap \{\vec{y} \mid \|\vec{x} - \vec{y}\|_2 \leq \|\vec{x} - \vec{y}'\|_2\}$. Indeed, the solution cannot be a point that is farther away from \vec{x} than \vec{y}' . By Weierstrass's extreme-value theorem, the optimization problem

$$\text{minimize } \|\vec{x} - \vec{y}\|_2^2 \quad (20)$$

$$\text{subject to } s \in \mathcal{S} \cap \{\vec{y} \mid \|\vec{x} - \vec{y}\|_2 \leq \|\vec{x} - \vec{y}'\|_2\} \quad (21)$$

has a solution because $\|\vec{x} - \vec{y}\|_2^2$ is a continuous function and the feasibility set is bounded and closed, and hence compact. Note that this also holds if \mathcal{S} is not convex.

Uniqueness

Assume that there are two distinct projections $\vec{y}_1 \neq \vec{y}_2$. Consider the point

$$\vec{y}' := \frac{\vec{y}_1 + \vec{y}_2}{2}, \quad (22)$$

which belongs to \mathcal{S} because \mathcal{S} is convex. The difference between \vec{x} and \vec{y}' and the difference between \vec{y}_1 and \vec{y}' are orthogonal vectors,

$$\langle \vec{x} - \vec{y}', \vec{y}_1 - \vec{y}' \rangle = \left\langle \vec{x} - \frac{\vec{y}_1 + \vec{y}_2}{2}, \vec{y}_1 - \frac{\vec{y}_1 + \vec{y}_2}{2} \right\rangle \quad (23)$$

$$= \left\langle \frac{\vec{x} - \vec{y}_1}{2} + \frac{\vec{x} - \vec{y}_2}{2}, \frac{\vec{x} - \vec{y}_1}{2} - \frac{\vec{x} - \vec{y}_2}{2} \right\rangle \quad (24)$$

$$= \frac{1}{4} (\|\vec{x} - \vec{y}_1\|^2 + \|\vec{x} - \vec{y}_2\|^2) \quad (25)$$

$$= 0, \quad (26)$$

because $\|\vec{x} - \vec{y}_1\| = \|\vec{x} - \vec{y}_2\|$ by assumption. By Pythagoras' theorem this implies

$$\|\vec{x} - \vec{y}_1\|_2^2 = \|\vec{x} - \vec{y}'\|_2^2 + \|\vec{y}_1 - \vec{y}'\|_2^2 \quad (27)$$

$$= \|\vec{x} - \vec{y}'\|_2^2 + \left\| \frac{\vec{y}_1 - \vec{y}_2}{2} \right\|_2^2 \quad (28)$$

$$> \|\vec{x} - \vec{y}'\|_2^2 \quad (29)$$

because $\vec{y}_1 \neq \vec{y}_2$ by assumption. We have reached a contradiction, so the projection is unique. \square

A convex combination of n points is any linear combination of the points with nonnegative coefficients that add up to one. In the case of two points, this is just the segment between the points.

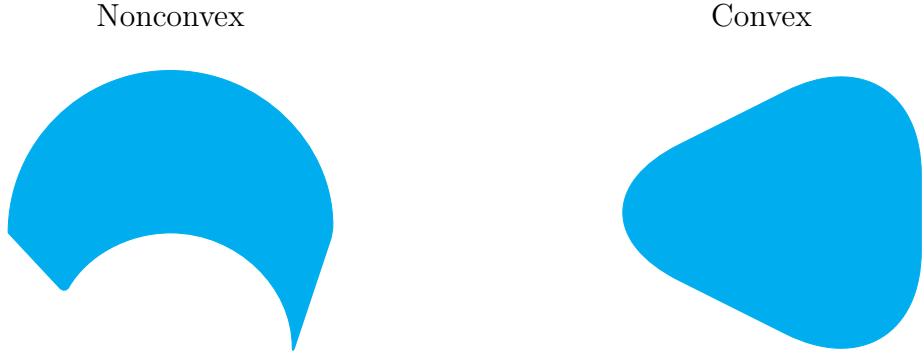


Figure 9: An example of a nonconvex set (left) and a convex set (right).

Definition 2.4 (Convex combination). *Given n vectors $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n \in \mathbb{R}^n$,*

$$\vec{x} := \sum_{i=1}^n \theta_i \vec{x}_i \quad (30)$$

is a convex combination of $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n$ as long as the real numbers $\theta_1, \theta_2, \dots, \theta_n$ are nonnegative and add up to one,

$$\theta_i \geq 0, \quad 1 \leq i \leq n, \quad (31)$$

$$\sum_{i=1}^n \theta_i = 1. \quad (32)$$

The convex hull of a set \mathcal{S} contains all convex combination of points in \mathcal{S} . Intuitively, it is the smallest convex set that contains \mathcal{S} .

Definition 2.5 (Convex hull). *The convex hull of a set \mathcal{S} is the set of all convex combinations of points in \mathcal{S} .*

A possible justification of why we penalize the ℓ_1 -norm to promote sparse structure is that the ℓ_1 -norm ball is the convex hull of 1-sparse vectors with unit norm, which form the intersection between the ℓ_0 “norm” ball and the ℓ_∞ -norm ball. The lemma is illustrated in 2D in Figure 10.

Lemma 2.6 (ℓ_1 -norm ball). *The ℓ_1 -norm ball is the convex hull of the intersection between the ℓ_0 “norm” ball and the ℓ_∞ -norm ball.*

Proof. We prove that the ℓ_1 -norm ball \mathcal{B}_{ℓ_1} is equal to the convex hull of the intersection between the ℓ_0 “norm” ball \mathcal{B}_{ℓ_0} and the ℓ_∞ -norm ball $\mathcal{B}_{\ell_\infty}$ by showing that the sets contain each other.

$$\underline{\mathcal{B}_{\ell_1} \subseteq \mathcal{C}(\mathcal{B}_{\ell_0} \cap \mathcal{B}_{\ell_\infty})}$$

Let \vec{x} be an n -dimensional vector in \mathcal{B}_{ℓ_1} . If we set $\theta_i := |\vec{x}[i]|$, where $\vec{x}[i]$ is the i th entry of \vec{x} by

$\vec{x}[i]$, and $\theta_0 = 1 - \sum_{i=1}^n \theta_i$ we have $\sum_{i=0}^n \theta_i = 1$ by construction, $\theta_i = |\vec{x}[i]| \geq 0$ and

$$\theta_0 = 1 - \sum_{i=1}^{n+1} \theta_i \quad (33)$$

$$= 1 - \|\vec{x}\|_1 \quad (34)$$

$$\geq 0 \quad \text{because } \vec{x} \in \mathcal{B}_{\ell_1}. \quad (35)$$

We can express now \vec{x} as a convex combination of the standard basis vectors multiplied by the sign of the entries of \vec{x} $\text{sign}(\vec{x}[1]) \vec{e}_1, \text{sign}(\vec{x}[2]) \vec{e}_2, \dots, \text{sign}(\vec{x}[n]) \vec{e}_n$, which belong to $\mathcal{B}_{\ell_0} \cap \mathcal{B}_{\ell_\infty}$ since they have a single nonzero entry with magnitude equal to one, and the zero vector $\vec{0}$, which also belongs to $\mathcal{B}_{\ell_0} \cap \mathcal{B}_{\ell_\infty}$,

$$\vec{x} = \sum_{i=1}^n \theta_i \text{sign}(\vec{x}[i]) \vec{e}_i + \theta_0 \vec{0}. \quad (36)$$

$$\underline{\mathcal{C}(\mathcal{B}_{\ell_0} \cap \mathcal{B}_{\ell_\infty}) \subseteq \mathcal{B}_{\ell_1}}$$

Let \vec{x} be an n -dimensional vector in $\mathcal{C}(\mathcal{B}_{\ell_0} \cap \mathcal{B}_{\ell_\infty})$. By the definition of convex hull, we can write

$$\vec{x} = \sum_{i=1}^m \theta_i \vec{y}_i, \quad (37)$$

where $m > 0$, $\vec{y}_1, \dots, \vec{y}_m \in \mathbb{R}^n$ have a single entry bounded by one, $\theta_i \geq 0$ for all $1 \leq i \leq m$ and $\sum_{i=1}^m \theta_i = 1$. This immediately implies $\vec{x} \in \mathcal{B}_{\ell_1}$, since

$$\|\vec{x}\|_1 \leq \sum_{i=1}^m \theta_i \|\vec{y}_i\|_1 \quad \text{by the Triangle inequality} \quad (38)$$

$$\leq \sum_{i=1}^m \theta_i \|\vec{y}_i\|_\infty \quad \text{because each } \vec{y}_i \text{ only has one nonzero entry} \quad (39)$$

$$\leq \sum_{i=1}^m \theta_i \quad (40)$$

$$\leq 1. \quad (41)$$

□

2.2 Constrained convex programs

In this section we discuss convex optimization problems, which are problems in which a convex function is minimized over a convex set.

Definition 2.7 (Convex optimization problem). *An optimization problem is a convex optimization problem if it can be written in the form*

$$\text{minimize } f_0(\vec{x}) \quad (42)$$

$$\text{subject to } f_i(\vec{x}) \leq 0, \quad 1 \leq i \leq m, \quad (43)$$

$$h_i(\vec{x}) = 0, \quad 1 \leq i \leq p, \quad (44)$$

where $f_0, f_1, \dots, f_m, h_1, \dots, h_p : \mathbb{R}^n \rightarrow \mathbb{R}$ are functions satisfying the following conditions

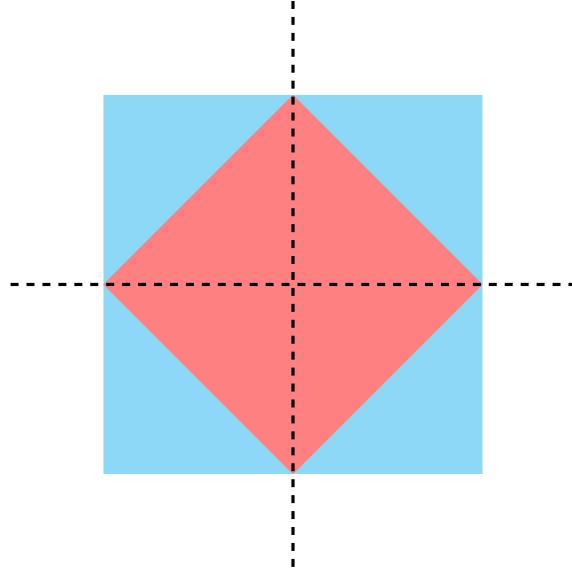


Figure 10: Illustration of Lemma (2.6) The ℓ_0 “norm” ball is shown in black, the ℓ_∞ -norm ball in blue and the ℓ_1 -norm ball in a reddish color.

- The cost function f_0 is convex.
- The functions that determine the inequality constraints f_1, \dots, f_m are convex.
- The functions that determine the equality constraints h_1, \dots, h_p are affine, i.e. $h_i(\vec{x}) = \vec{a}_i^T \vec{x} + b_i$ for some $\vec{a}_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$.

Any vector that satisfies all the constraints in a convex optimization problem is said to be *feasible*. A solution to the problem is any vector \vec{x}^* such that for all feasible vectors \vec{x}

$$f_0(\vec{x}) \geq f_0(\vec{x}^*). \quad (45)$$

If a solution exists $f(\vec{x}^*)$ is the optimal value or optimum of the optimization problem.

Under the conditions in Definition 2.7 we can check that the feasibility set of the optimization problem is indeed convex. Indeed, it corresponds to the intersection of several convex sets: the 0-sublevel sets of f_1, \dots, f_m , which are convex by Lemma 2.9 below, and the hyperplanes $h_i(\vec{x}) = \vec{a}_i^T \vec{x} + b_i$. The intersection is convex by Lemma 2.2.

Definition 2.8 (Sublevel set). *The γ -sublevel set of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$, where $\gamma \in \mathbb{R}$, is the set of points in \mathbb{R}^n at which the function is smaller or equal to γ ,*

$$C_\gamma := \{\vec{x} \mid f(\vec{x}) \leq \gamma\}. \quad (46)$$

Lemma 2.9 (Sublevel sets of convex functions). *The sublevel sets of a convex function are convex.*

Proof. If $\vec{x}, \vec{y} \in \mathbb{R}^n$ belong to the γ -sublevel set of a convex function f then for any $\theta \in (0, 1)$

$$f(\theta \vec{x} + (1 - \theta) \vec{y}) \leq \theta f(\vec{x}) + (1 - \theta) f(\vec{y}) \quad \text{by convexity of } f \quad (47)$$

$$\leq \gamma \quad (48)$$

because both \vec{x} and \vec{y} belong to the γ -sublevel set. We conclude that any convex combination of \vec{x} and \vec{y} also belongs to the γ -sublevel set. \square

If both the cost function and the constraint functions of a convex optimization problem are affine, the problem is a linear program.

Definition 2.10 (Linear program). *A linear program is a convex optimization problem of the form*

$$\text{minimize} \quad \vec{d}^T \vec{x} \quad (49)$$

$$\text{subject to} \quad \vec{c}_i^T \vec{x} \leq d_i, \quad 1 \leq i \leq m, \quad (50)$$

$$A\vec{x} = \vec{b}. \quad (51)$$

It turns out that ℓ_1 -norm minimization can be cast as a linear program.

Theorem 2.11 (ℓ_1 -norm minimization as a linear program). *The optimization problem*

$$\text{minimize} \quad \|\vec{x}\|_1 \quad (52)$$

$$\text{subject to} \quad A\vec{x} = \vec{b} \quad (53)$$

can be recast as the linear program

$$\text{minimize} \quad \sum_{i=1}^n \vec{t}[i] \quad (54)$$

$$\text{subject to} \quad \vec{t}[i] \geq \vec{e}_i^T \vec{x}, \quad (55)$$

$$\vec{t}[i] \geq -\vec{e}_i^T \vec{x}, \quad (56)$$

$$A\vec{x} = \vec{b}. \quad (57)$$

Proof. To show that the linear problem and the ℓ_1 -norm minimization problem are equivalent, we show that they have the same set of solutions.

Let us denote an arbitrary solution of the linear program by $(\vec{x}^{\text{lp}}, \vec{t}^{\text{lp}})$. For any solution \vec{x}^{ℓ_1} of the ℓ_1 -norm minimization problem, we define \vec{t}^{ℓ_1} such that $\vec{t}^{\ell_1}[i] := |\vec{x}^{\ell_1}[i]|$. $(\vec{x}^{\ell_1}, \vec{t}^{\ell_1})$ is feasible for the LP so

$$\|\vec{x}^{\ell_1}\|_1 = \sum_{i=1}^n \vec{t}^{\ell_1}[i] \quad (58)$$

$$\geq \sum_{i=1}^n \vec{t}^{\text{lp}}[i] \quad \text{by optimality of } \vec{t}^{\text{lp}} \quad (59)$$

$$\geq \|\vec{x}^{\text{lp}}\|_1 \quad \text{by constraints (55) and (56)}. \quad (60)$$

This implies that any solution of the linear program is also a solution of the ℓ_1 -norm minimization problem.

To prove the converse, we fix a solution \vec{x}^{ℓ_1} of the ℓ_1 -norm minimization problem. Setting $\vec{t}^{\ell_1}[i] := |\vec{x}^{\ell_1}[i]|$, we show that $(\vec{x}^{\ell_1}, \vec{t}^{\ell_1})$ is a solution of the linear program. Indeed,

$$\sum_{i=1}^n t_i^{\ell_1} = \|\vec{x}^{\ell_1}\|_1 \quad (61)$$

$$\leq \|\vec{x}^{\text{lp}}\|_1 \quad \text{by optimality of } \vec{x}^{\ell_1} \quad (62)$$

$$\leq \sum_{i=1}^n \vec{t}^{\text{lp}}[i] \quad \text{by constraints (55) and (56).} \quad (63)$$

□

If the cost function is a positive semidefinite quadratic form and the constraints are affine a convex optimization problem is called a quadratic program (QP).

Definition 2.12 (Quadratic program). *A quadratic program is a convex optimization problem of the form*

$$\text{minimize} \quad \vec{x}^T Q \vec{x} + \vec{d}^T \vec{x} \quad (64)$$

$$\text{subject to} \quad \vec{c}_i^T \vec{x} \leq d_i, \quad 1 \leq i \leq m, \quad (65)$$

$$A \vec{x} = \vec{b}, \quad (66)$$

where $Q \in \mathbb{R}^{n \times n}$ is positive semidefinite.

A corollary of Theorem 2.11 is that ℓ_1 -norm regularized least squares can be cast as a QP.

Corollary 2.13 (ℓ_1 -norm regularized least squares as a QP). *The optimization problem*

$$\text{minimize} \quad \|A \vec{x} - y\|_2^2 + \lambda \|\vec{x}\|_1 \quad (67)$$

can be recast as the quadratic program

$$\text{minimize} \quad \vec{x}^T A^T A \vec{x} - 2 \vec{y}^T \vec{x} + \lambda \sum_{i=1}^n \vec{t}[i] \quad (68)$$

$$\text{subject to} \quad \vec{t}[i] \geq \vec{e}_i^T \vec{x}, \quad (69)$$

$$\vec{t}[i] \geq -\vec{e}_i^T \vec{x}. \quad (70)$$

2.3 Duality

The Lagrangian of an optimization problem combines the cost function and the constraints.

Definition 2.14. *The Lagrangian of the optimization problem in Definition 2.7 is defined as the cost function augmented by a weighted linear combination of the constraint functions,*

$$L(\vec{x}, \vec{\alpha}, \vec{\nu}) := f_0(\vec{x}) + \sum_{i=1}^m \vec{\alpha}[i] f_i(\vec{x}) + \sum_{j=1}^p \vec{\nu}[j] h_j(\vec{x}), \quad (71)$$

where the vectors $\vec{\alpha} \in \mathbb{R}^m, \vec{\nu} \in \mathbb{R}^p$ are called Lagrange multipliers or dual variables, whereas \vec{x} is the primal variable.

The Lagrangian yields a family of lower bounds to the cost function of the optimization problem at every feasible point.

Lemma 2.15. *As long as $\vec{\alpha}[i] \geq 0$ for $1 \leq i \leq m$, the Lagrangian of the optimization problem in Definition 2.7 lower bounds the cost function at all feasible points, i.e. if \vec{x} is feasible then*

$$L(\vec{x}, \vec{\alpha}, \vec{\nu}) \leq f_0(\vec{x}). \quad (72)$$

Proof. If \vec{x} is feasible and $\vec{\alpha}[i] \geq 0$ for $1 \leq i \leq m$ then

$$\vec{\alpha}[i] f_i(\vec{x}) \leq 0, \quad (73)$$

$$\vec{\nu}[j] h_j(\vec{x}) = 0, \quad (74)$$

which immediately implies (72). \square

Minimizing over the primal variable yields a family of lower bounds that only depends on the dual variables. We call the corresponding function the Lagrange dual function.

Definition 2.16 (Lagrange dual function). *The Lagrange dual function is the infimum of the Lagrangian over the primal variable \vec{x}*

$$l(\vec{\alpha}, \vec{\nu}) := \inf_{\vec{x} \in \mathbb{R}^n} L(\vec{x}, \vec{\alpha}, \vec{\nu}). \quad (75)$$

Theorem 2.17 (Lagrange dual function as a lower bound of the primal optimum). *Let \vec{x}^* denote an optimal value of the optimization problem in Definition 2.7,*

$$l(\vec{\alpha}, \vec{\nu}) \leq \vec{x}^*, \quad (76)$$

as long as $\vec{\alpha}[i] \geq 0$ for $1 \leq i \leq n$.

Proof. The result follows directly from (72),

$$\vec{x}^* = f_0(\vec{x}^*) \quad (77)$$

$$\geq L(\vec{x}^*, \vec{\alpha}, \vec{\nu}) \quad (78)$$

$$\geq l(\vec{\alpha}, \vec{\nu}). \quad (79)$$

\square

Optimizing the lower bound provided by the Lagrange dual function yields an optimization problem that is called the *dual* problem of the original optimization problem. The original problem is called the *primal* problem in this context.

Definition 2.18 (Dual problem). *The dual problem of the optimization problem from Definition 2.7 is*

$$\text{maximize } l(\vec{\alpha}, \vec{\nu}) \quad (80)$$

$$\text{subject to } \vec{\alpha}[i] \geq 0, \quad 1 \leq i \leq m. \quad (81)$$

Note that the cost function is a pointwise supremum of linear (and hence convex) functions.

Lemma 2.19 (Supremum of convex functions). *Pointwise supremum of a family of convex functions indexed by a set \mathcal{I}*

$$f_{\sup}(\vec{x}) := \sup_{i \in \mathcal{I}} f_i(\vec{x}). \quad (82)$$

is convex.

Proof. For any $0 \leq \theta \leq 1$ and any $\vec{x}, \vec{y} \in \mathbb{R}$,

$$f_{\sup}(\theta \vec{x} + (1 - \theta) \vec{y}) = \sup_{i \in \mathcal{I}} f_i(\theta \vec{x} + (1 - \theta) \vec{y}) \quad (83)$$

$$\leq \sup_{i \in \mathcal{I}} \theta f_i(\vec{x}) + (1 - \theta) f_i(\vec{y}) \quad \text{by convexity of the } f_i \quad (84)$$

$$\leq \theta \sup_{i \in \mathcal{I}} f_i(\vec{x}) + (1 - \theta) \sup_{j \in \mathcal{I}} f_j(\vec{y}) \quad (85)$$

$$= \theta f_{\sup}(\vec{x}) + (1 - \theta) f_{\sup}(\vec{y}) \quad (86)$$

□

As a result of the lemma, the dual problem is a convex optimization problem even if the primal is nonconvex! The following result, which is an immediate corollary to Theorem 2.17, states that the optimum of the dual problem is a lower bound for the primal optimum. This is known as weak duality.

Corollary 2.20 (Weak duality). *Let \vec{x}^* denote an optimum of the optimization problem in Definition 2.7 and d^* an optimum of the corresponding dual problem,*

$$d^* \leq \vec{x}^*. \quad (87)$$

In the case of convex functions, the optima of the primal and dual problems are often equal, i.e.

$$d^* = \vec{x}^*. \quad (88)$$

This is known as strong duality. A simple condition that guarantees strong duality for convex optimization problems is Slater's condition.

Definition 2.21 (Slater's condition). *A vector $x \in \mathbb{R}^n$ satisfies Slater's condition for a convex optimization problem if*

$$f_i(\vec{x}) < 0, \quad 1 \leq i \leq m, \quad (89)$$

$$Ax = b. \quad (90)$$

A proof of strong duality under Slater's condition can be found in Section 5.3.2 of [2].

The following theorem derives the dual problem for the ℓ_1 -norm minimization problem with equality constraints.

Theorem 2.22 (Dual of ℓ_1 -norm minimization). *The dual of the optimization problem of*

$$\min_{\vec{x}} \|\vec{x} \in \mathbb{R}^m\|_1 \quad \text{subject to} \quad A\vec{x} = \vec{y} \quad (91)$$

is

$$\max_{\vec{\nu} \in \mathbb{R}^n} \vec{y}^T \vec{\nu} \quad \text{subject to} \quad \|A^T \vec{\nu}\|_\infty \leq 1. \quad (92)$$

Proof. The Lagrangian is equal to

$$L(\vec{x}, \vec{\nu}) = \|\vec{x}\|_1 + \vec{\nu}^T (\vec{y} - A\vec{x}), \quad (93)$$

so the Lagrange dual function equals

$$l(\vec{\alpha}, \vec{\nu}) := \inf_{\vec{x} \in \mathbb{R}^n} \|\vec{x}\|_1 - (A^T \vec{\nu})^T \vec{x} + \vec{\nu}^T \vec{y}. \quad (94)$$

If $(A^T \vec{\nu})[i] > 1$ then one can set $\vec{x}[i]$ arbitrarily large so that $l(\vec{\alpha}, \vec{\nu}) \rightarrow -\infty$. The same happens if $(A^T \vec{\nu})[i] < 1$. If $\|A^T \vec{\nu}\|_\infty \leq 1$, by Hölder's inequality (Theorem 3.16 in Lecture Notes 1)

$$|(A^T \vec{\nu})^T \vec{x}| \leq \|\vec{x}\|_1 \|A^T \vec{\nu}\|_\infty \quad (95)$$

$$\leq \|\vec{x}\|_1, \quad (96)$$

so the Lagrangian is minimized by setting \vec{x} to zero and $l(\vec{\alpha}, \vec{\nu}) = \vec{\nu}^T \vec{y}$. This completes the proof. \square

Interestingly, the solution to the dual of the ℓ_1 -norm minimization problem can often be used to estimate the support of the primal solution. Figure 11 shows that the vector $A^T \vec{\nu}^*$, where A is the underdetermined linear operator and $\vec{\nu}^*$ is a solution to Problem (92), reveals the support of the original signal for the randomly-subsampled data in Example 1.2.

Lemma 2.23. *If there exists a feasible vector for the primal problem, then the solution $\vec{\nu}^*$ to Problem (92) satisfies*

$$(A^T \vec{\nu}^*)[i] = \text{sign}(\vec{x}^*[i]) \quad \text{for all } \vec{x}^*[i] \neq 0 \quad (97)$$

for all solutions \vec{x}^* to the primal problem.

Proof. If there is a feasible vector for the primal problem, then strong duality holds because the optimization problem is a linear program with finite cost function. By strong duality

$$\|\vec{x}^*\|_1 = \vec{y}^T \vec{\nu}^* \quad (98)$$

$$= (A\vec{x}^*)^T \vec{\nu}^* \quad (99)$$

$$= (\vec{x}^*)^T (A^T \vec{\nu}^*) \quad (100)$$

$$= \sum_{i=1}^m (A^T \vec{\nu}^*)[i] \vec{x}^*[i]. \quad (101)$$

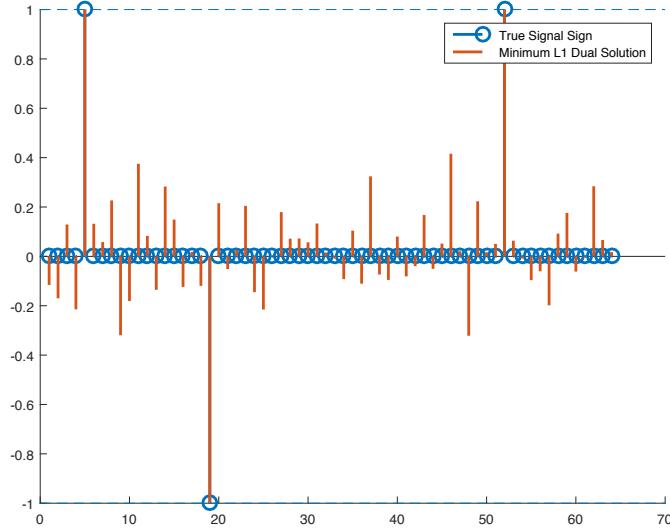


Figure 11: The vector $A^T \vec{\nu}^*$, where A is the underdetermined linear operator and $\vec{\nu}^*$ is a solution to Problem (92), reveals the support of the original signal for the randomly-subsampled data in Example 1.2.

By Hölder's inequality

$$\|\vec{x}^*\|_1 \geq \sum_{i=1}^m (A^T \vec{\nu}^*)[i] \vec{x}^*[i] \quad (102)$$

with equality if and only if

$$(A^T \vec{\nu}^*)[i] = \text{sign}(\vec{x}^*[i]) \quad \text{for all } \vec{x}^*[i] \neq 0. \quad (103)$$

□

Consider the following algorithm for sparse recovery. Our goal is to find nonzero locations of a sparse vector \vec{x} from $\vec{y} = A\vec{x}$. We have access to inner products of \vec{x} and $A^T \vec{w}$ for any \vec{w} , since

$$\vec{y}^T \vec{w} = (A\vec{x})^T \vec{w} \quad (104)$$

$$= \vec{x}^T (A^T \vec{w}). \quad (105)$$

This suggests maximizing $A^T \vec{w}$, while bounding its magnitude entries by 1. In that case, the entries where \vec{x} is nonzero should saturate to 1 or -1. This is exactly Problem (92)!

3 Analysis of constrained convex programs

3.1 Minimum ℓ_2 -norm solution

The best-case scenario for the analysis of constrained convex program is that the optimization problem has a closed-form solution. This is the case for ℓ_2 -norm minimization.

Theorem 3.1. Let $A \in \mathbb{R}^{m \times n}$ be a full rank matrix such that $m < n$. For any $\vec{y} \in \mathbb{R}^n$ the solution to the optimization problem

$$\arg \min_{\vec{x}} \|\vec{x}\|_2 \quad \text{subject to} \quad A\vec{x} = \vec{y}. \quad (106)$$

is

$$\vec{x}^* := VS^{-1}U^T\vec{y} = A^T(A^TA)^{-1}\vec{y}. \quad (107)$$

Proof. Let us decompose \vec{x} into its projection on the row space of A and on its orthogonal complement

$$\vec{x} = \mathcal{P}_{\text{row}(A)}\vec{x} + \mathcal{P}_{\text{row}(A)^\perp}\vec{x}. \quad (108)$$

Let $A = USV^T$ be the reduced SVD of A where S contains the nonzero singular values. Since A is full rank V contains an orthonormal basis of $\text{row}(A)$ and we can write $\mathcal{P}_{\text{row}(A)}\vec{x} = V\vec{c}$ for some vector $\vec{c} \in \mathbb{R}^n$. We have

$$A\vec{x} = A\mathcal{P}_{\text{row}(A)}\vec{x} \quad (109)$$

$$= USV^TV\vec{c} \quad (110)$$

$$= US\vec{c}. \quad (111)$$

So that the equality constraint is equivalent to

$$US\vec{c} = \vec{y}, \quad (112)$$

where US is square and invertible because A is full rank, so that

$$\vec{c} = S^{-1}U^T\vec{y} \quad (113)$$

and hence for all feasible vectors \vec{x}

$$\mathcal{P}_{\text{row}(A)}\vec{x} = VS^{-1}U^T\vec{y}. \quad (114)$$

By Pythagoras' theorem, minimizing $\|\vec{x}\|_2$ is equivalent to minimizing

$$\|\vec{x}\|_2^2 = \left\| \mathcal{P}_{\text{row}(A)}\vec{x} \right\|_2^2 + \left\| \mathcal{P}_{\text{row}(A)^\perp}\vec{x} \right\|_2^2. \quad (115)$$

Since $\mathcal{P}_{\text{row}(A)}\vec{x}$ is fixed by the equality constraint, the minimum is achieved by setting $\mathcal{P}_{\text{row}(A)^\perp}\vec{x}$ to zero and the solution equals

$$\vec{x}^* := VS^{-1}U^T\vec{y} = A^T(A^TA)^{-1}\vec{y}. \quad (116)$$

□

The next lemma exploits the closed-form solution of the minimum ℓ_2 -norm to explain the aliasing that occurs for the regularly-subsampled data in Figure 5.

Lemma 3.2 (Regular subsampling). *Let A be the regularly-subsampled DFT matrix in Example 1.2 and let*

$$\vec{x} := \begin{bmatrix} \vec{x}_{\text{up}} \\ \vec{x}_{\text{down}} \end{bmatrix} \quad (117)$$

be the original signal. The minimum ℓ_2 -norm estimate equals

$$\vec{x}_{\ell_2} := \arg \min_{A\vec{x} = \vec{y}} \|\vec{x}\|_2 \quad (118)$$

$$= \frac{1}{2} \begin{bmatrix} \vec{x}_{\text{up}} + \vec{x}_{\text{down}} \\ \vec{x}_{\text{up}} + \vec{x}_{\text{down}} \end{bmatrix}. \quad (119)$$

Proof. As is obvious in Figure 2 and we discussed in Example 1.15 of Lecture Notes 4, the matrix A is equal to two concatenated DFT matrices of size $m/2$ (for simplicity we assume m is even)

$$A := \frac{1}{\sqrt{2}} \begin{bmatrix} F_{m/2} & F_{m/2} \end{bmatrix}, \quad (120)$$

where $F_{m/2}^* F_{m/2} = F_{m/2} F_{m/2}^* = I$. By Theorem 3.1

$$\vec{x}_{\ell_2} = A^T (A^T A)^{-1} \vec{y} \quad (121)$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} F_{m/2}^* \\ F_{m/2}^* \end{bmatrix} \left(\frac{1}{\sqrt{2}} \begin{bmatrix} F_{m/2} & F_{m/2} \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} F_{m/2}^* \\ F_{m/2}^* \end{bmatrix} \right)^{-1} \frac{1}{\sqrt{2}} \begin{bmatrix} F_{m/2} & F_{m/2} \end{bmatrix} \begin{bmatrix} \vec{x}_{\text{up}} \\ \vec{x}_{\text{down}} \end{bmatrix} \quad (122)$$

$$= \frac{1}{2} \begin{bmatrix} F_{m/2}^* \\ F_{m/2}^* \end{bmatrix} \left(\frac{1}{2} \begin{bmatrix} F_{m/2} F_{m/2}^* + F_{m/2} F_{m/2}^* \end{bmatrix} \right)^{-1} (F_{m/2} \vec{x}_{\text{up}} + F_{m/2} \vec{x}_{\text{down}}) \quad (123)$$

$$= \frac{1}{2} \begin{bmatrix} F_{m/2}^* \\ F_{m/2}^* \end{bmatrix} I^{-1} (F_{m/2} \vec{x}_{\text{up}} + F_{m/2} \vec{x}_{\text{down}}) \quad (124)$$

$$= \frac{1}{2} \begin{bmatrix} F_{m/2}^* (F_{m/2} \vec{x}_{\text{up}} + F_{m/2} \vec{x}_{\text{down}}) \\ F_{m/2}^* (F_{m/2} \vec{x}_{\text{up}} + F_{m/2} \vec{x}_{\text{down}}) \end{bmatrix} \quad (125)$$

$$= \frac{1}{2} \begin{bmatrix} \vec{x}_{\text{up}} + \vec{x}_{\text{down}} \\ \vec{x}_{\text{up}} + \vec{x}_{\text{down}} \end{bmatrix}. \quad (126)$$

□

3.2 Minimum ℓ_1 -norm solution

Unfortunately, the solution to ℓ_1 -norm minimization with linear constraints does not have a closed-form solution. When we considered unconstrained nondifferentiable convex problems without closed-form solutions in Lecture Notes 8, we characterized the solutions by exploiting the fact that the zero vector is a subgradient of a convex cost function at a point if and only if the point is a minimizer. Here we will use a different argument based on the dual problem (which can often also be interpreted geometrically in terms of subgradients as we discuss below). The main idea is to construct a dual feasible vector whose existence implies that the original signal which we aim to recover is the unique solution to the primal.

Consider a certain sparse vector $\vec{x}^* \in \mathbb{R}^m$ with support T such that $A\vec{x}^* = \vec{y}$. If there exists a vector $\vec{\nu} \in \mathbb{R}^n$ such that $A^T \vec{\nu}$ is equal to the sign of \vec{x} on T and has magnitude smaller than one elsewhere, then ν is feasible for the dual problem 92, so by weak duality $\|\vec{x}\|_1 \geq \vec{y}^T \vec{\nu}$ for any $\vec{x} \in \mathbb{R}^m$ that is feasible for the primal. We then have

$$\|\vec{x}\|_1 \geq \vec{y}^T \vec{\nu} \quad (127)$$

$$= (A\vec{x}^*)^T \vec{\nu} \quad (128)$$

$$= (\vec{x}^*)^T (A^T \vec{\nu}) \quad (129)$$

$$= \sum_{i=1}^m \vec{x}^*[i] \operatorname{sign}(\vec{x}^*[i]) \quad (130)$$

$$= \|\vec{x}^*\|_1. \quad (131)$$

Geometrically, $A^T \vec{\nu}$ is a subgradient of the ℓ_1 norm at \vec{x}^* . The subgradient is orthogonal to the feasibility hyperplane given by $A\vec{x} = \vec{y}$ (any vector \vec{v} within the hyperplane is the difference between two feasible vectors and therefore satisfies $A\vec{v} = \vec{0}$). As a result, for any other feasible vector \vec{x}

$$\|\vec{x}\|_1 \geq \|\vec{x}^*\|_1 + (A^T \vec{\nu})^T (\vec{x} - \vec{x}^*) \quad (132)$$

$$= \|\vec{x}^*\|_1 + \vec{\nu}^T (A\vec{x} - A\vec{x}^*) \quad (133)$$

$$= \|\vec{x}^*\|_1. \quad (134)$$

These two arguments show that the existence of a certain dual vector can be used to establish that a certain primal feasible vector is a solution, but they do not establish uniqueness. It turns out that requiring that the magnitude of $A^T \vec{\nu}$ be strictly smaller than one on T^c is enough to guarantee it (as long as A is full rank). In that case, we call the dual variable $\vec{\nu}$ a dual certificate for the ℓ_1 -norm minimization problem.

Theorem 3.3 (Dual certificate for ℓ_1 -norm minimization). *Let $\vec{x}^* \in \mathbb{R}^m$ with support T such that $A\vec{x}^* = \vec{y}$ and the submatrix A_T containing the columns of A indexed by T is full rank. If there exists a vector $\vec{\nu} \in \mathbb{R}^n$ such that*

$$(A^T \vec{\nu})[i] = \operatorname{sign}(\vec{x}^*[i]) \quad \text{if } \vec{x}^*[i] \neq 0 \quad (135)$$

$$|(A^T \vec{\nu})[i]| < 1 \quad \text{if } \vec{x}^*[i] = 0 \quad (136)$$

then \vec{x}^* is the unique solution to the ℓ_1 -norm minimization problem (16).

Proof. For any feasible $\vec{x} \in \mathbb{R}^m$, let $\vec{h} := \vec{x} - \vec{x}^*$. If A_T is full rank then $\vec{h}_{T^c} \neq 0$ unless $\vec{h} = 0$ because otherwise \vec{h}_T would be a nonzero vector in the null space of A_T . Condition (136) implies

$$\left\| \vec{h}_{T^c} \right\|_1 > (A^T \vec{\nu})^T \vec{h}_{T^c}, \quad (137)$$

where \vec{h}_{T^c} denotes \vec{h} restricted to the entries indexed by T^c . Let $\mathcal{P}_T(\cdot)$ denote a projection that sets to zero all entries of a vector except the ones indexed by T . We have

$$\|\vec{x}\|_1 = \left\| \vec{x}^* + \mathcal{P}_T(\vec{h}) \right\|_1 + \left\| \vec{h}_{T^c} \right\|_1 \quad \text{because } \vec{x}^* \text{ is supported on } T \quad (138)$$

$$> \|\vec{x}^*\|_1 + (A^T \vec{\nu})^T \mathcal{P}_T(\vec{h}) + (A^T \vec{\nu})^T \mathcal{P}_{T^c}(\vec{h}) \quad \text{by (137)} \quad (139)$$

$$= \|\vec{x}^*\|_1 + \vec{\nu}^T A \vec{h} \quad (140)$$

$$= \|\vec{x}^*\|_1. \quad (141)$$

□

A strategy to prove that compressed sensing succeeds for a class of signals is to propose a dual-certificate construction and show that it produces a valid certificate for any signal in the class. We illustrate this with Gaussian matrices, but similar arguments can be extended to random Fourier matrices [7] and other measurements [4] (see also [6] for a more recent proof technique based on approximate dual certificates that provides better guarantees). It is also worth noting that the restricted-isometry property can be directly used to prove exact recovery via ℓ_1 -norm minimization [5], but this technique is less general. Dual certificates can also be used to analyze other problems such as matrix completion, super-resolution and phase retrieval.

Theorem 3.4 (Exact recovery via ℓ_1 -norm minimization). *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a random matrix with iid standard Gaussian entries and $\vec{x}^* \in \mathbb{R}^m$ a vector with s nonzero entries such that $\mathbf{A}\vec{x}^* = \vec{y}$. Then \vec{x}^* is the unique solution to the ℓ_1 -norm minimization problem (16) with probability at least $1 - \frac{1}{n}$ as long as the number of measurements satisfies*

$$m \geq Cs \log n, \quad (142)$$

for a fixed numerical constant C .

Proof. By Theorem 3.3 all we need to show is that for any support T of size s and any possible sign pattern $\vec{w} := \vec{x}_T^* \in \mathbb{R}^s$ of the nonzero entries of \vec{x}^* there exists a valid dual certificate $\vec{\nu}$. The certificate must satisfy

$$\mathbf{A}_T^T \vec{\nu} = \vec{w}. \quad (143)$$

Ideally we would like to analyze the vector $\vec{\nu}$ satisfying this underdetermined system of s equations such that $\mathbf{A}_{T^c}^T \vec{\nu}$ has the smallest possible ℓ_∞ norm. Unfortunately, the solution to the optimization problem

$$\min_{\nu} \|\mathbf{A}_{T^c}^T \vec{\nu}\|_\infty \quad \text{subject to} \quad \mathbf{A}_T^T \vec{\nu} = \vec{w} \quad (144)$$

does not have a closed-form solution. However, the solution to

$$\min_{\vec{\nu}} \|\vec{\nu}\|_2 \quad \text{subject to} \quad \mathbf{A}_T^T \vec{\nu} = \vec{w} \quad (145)$$

does, so we can analyze it instead. By Theorem 3.1 the solution is

$$\vec{\nu}_{\ell_2} := \mathbf{A}_T^T (\mathbf{A}_T^T \mathbf{A}_T)^{-1} \vec{w}. \quad (146)$$

To control $\vec{\nu}_{\ell_2}$ we resort to the bound on the singular values of a fixed $m \times s$ submatrix in equation (11). Setting $\kappa_s := 0.5$ we denote by \mathcal{E} the event that

$$0.5\sqrt{m} \leq \sigma_s \leq \sigma_1 \leq 1.5\sqrt{m}, \quad (147)$$

where

$$P(\mathcal{E}) \geq 1 - \exp\left(-C' \frac{m}{s}\right) \quad (148)$$

for a fixed constant C' . Conditioned on \mathcal{E} \mathbf{A}_T is full rank and $\mathbf{A}_T^T \mathbf{A}_T$ is invertible, so $\vec{\nu}_{\ell_2}$ guarantees condition (135). In order to verify condition (136), we need to bound $\mathbf{A}_i^T \vec{\nu}_{\ell_2}$ for all indices $i \in T^c$. Let $\mathbf{U} \mathbf{S} \mathbf{V}^T$ be the SVD of \mathbf{A}_T . Conditioned on \mathcal{E} we have

$$\|\vec{\nu}_{\ell_2}\|_2 = \|\mathbf{V} \mathbf{S}^{-1} \mathbf{U}^T \vec{w}\|_2 \quad (149)$$

$$\leq \frac{\|\vec{w}\|_2}{\sigma_s} \quad (150)$$

$$\leq 2 \sqrt{\frac{s}{m}}. \quad (151)$$

For a fixed $i \in T^c$ and a fixed vector $\vec{v} \in R^n$, $\mathbf{A}_i^T \vec{v} / \|\vec{v}\|_2$ is a standard Gaussian random variable, which implies

$$P(|\mathbf{A}_i^T \vec{v}| \geq 1) = P\left(\frac{|\mathbf{A}_i^T \vec{v}|}{\|\vec{v}\|_2} \geq \|\vec{v}\|_2\right) \quad (152)$$

$$\leq 2 \exp(-\|\vec{v}\|_2^2/2) \quad (153)$$

by the following lemma.

Lemma 3.5 (Proof in Section 4.1). *For a Gaussian random variable \mathbf{u} with zero mean and unit variance and any $t > 0$*

$$P(|\mathbf{u}| \geq t) \leq 2 \exp\left(-\frac{t^2}{2}\right). \quad (154)$$

Note that if $i \notin T$ then \mathbf{A}_i and $\vec{\nu}_{\ell_2}$ are independent (they depend on different and hence independent entries of \mathbf{A}). This means that due to equation 151

$$P(|\mathbf{A}_i^T \vec{\nu}_{\ell_2}| \geq 1 | \mathcal{E}) = P\left(|\mathbf{A}_i^T \vec{v}| \geq 1 \quad \text{for} \quad \|\nu\|_2 \leq 2 \sqrt{\frac{s}{m}}\right) \quad (155)$$

$$\leq 2 \exp\left(-\frac{m}{8s}\right). \quad (156)$$

As a result,

$$P(|\mathbf{A}_i^T \vec{\nu}_{\ell_2}| \geq 1) \leq P(|\mathbf{A}_i^T \vec{\nu}_{\ell_2}| \geq 1 | \mathcal{E}) + P(\mathcal{E}^c) \quad (157)$$

$$\leq 2 \exp\left(-\frac{m}{8s}\right) + \exp\left(-C' \frac{m}{s}\right). \quad (158)$$

We now apply the union bound to obtain a bound that holds for all $i \in T^c$. Since T^c has cardinality at most n

$$P\left(\bigcup_{i \in T^c} \{|\mathbf{A}_i^T \vec{\nu}_{\ell_2}| \geq 1\}\right) \leq n \left(2 \exp\left(-\frac{m}{8s}\right) + \exp\left(-C' \frac{m}{s}\right)\right). \quad (159)$$

We can consequently choose a constant C so that if the number of measurements satisfies

$$m \geq C s \log n \quad (160)$$

we have exact recovery with probability $1 - \frac{1}{n}$. \square

4 Proofs

4.1 Proof of Lemma 3.5

By symmetry of the Gaussian probability density function, we just need to bound the probability that $u > t$. Applying Markov's inequality (Theorem 2.6 in Lecture Notes 3) we have

$$P(u \geq t) = P(\exp(ut) \geq \exp(t^2)) \quad (161)$$

$$\leq E(\exp(ut - t^2)) \quad (162)$$

$$= \exp\left(-\frac{t^2}{2}\right) \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\frac{(x-t)^2}{2}\right) dx \quad (163)$$

$$= \exp\left(-\frac{t^2}{2}\right). \quad (164)$$

References

The proofs of exact recovery via ℓ_1 -norm minimization and of the restricted isometry property of Gaussian matrices are based on arguments in [3] and [1]. For further reading on mathematical tools used to analyze compressed sensing we refer to [11] and [9].

- [1] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.
- [2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [3] E. Candès and B. Recht. Simple bounds for recovering low-complexity models. *Mathematical Programming*, 141(1-2):577–589, 2013.
- [4] E. Candès and J. Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.
- [5] E. J. Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.
- [6] E. J. Candès and Y. Plan. A probabilistic and ripless theory of compressed sensing. *IEEE Transactions on Information Theory*, 57(11):7235–7254, 2011.
- [7] E. J. Candès, J. K. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.
- [8] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Transactions in Information Theory*, 52:5406–5425, 2006.
- [9] S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing*, volume 1. 2013.
- [10] M. Rudelson and R. Vershynin. On sparse reconstruction from Fourier and Gaussian measurements. *Communications on Pure and Applied Mathematics*, 61(8):1025–1045, 2008.

- [11] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Lecture Notes 10:

Matrix Factorization

1 Low-rank models

1.1 Rank-1 model

Consider the problem of modeling a quantity $y[i, j]$ that depends on two indices i and j . To fix ideas, assume that $y[i, j]$ represents the rating assigned to a movie i by a user j . If we have available a data set of such ratings, how can we predict new ratings for (i, j) that we have not seen before? A possible assumption is that some movies are more popular in general than others, and some users are more generous. This is captured by the following simple model

$$y[i, j] \approx a[i]b[j]. \quad (1)$$

The features a and b capture the respective contributions of the movie and the user to the ranking. If $a[i]$ is large, movie $[i]$ receives good ratings in general. If $b[j]$ is large, then user $[j]$ is generous, they give good ratings in general.

In the model (1) the two unknown parameters $a[i]$ and $b[j]$ are combined multiplicatively. As a result, if we fit these parameters using observed data by minimizing a least-squares cost function, the optimization problem is not convex. Figure 1 illustrates this for the function

$$f(a, b) := (1 - ab)^2, \quad (2)$$

which corresponds to an example where there is only one movie and one user and the rating equals one. Nonconvexity is problematic because if we use algorithms such as gradient descent to minimize the cost function, they may get stuck in local minima corresponding to parameter values that do not fit the data well. In addition, there is a scaling issue: the pairs (a, b) and $(ac, b/c)$ yield the same cost for any constant c . This motivates incorporating a constraint to restrict the magnitude of some of the parameters. For example, in Figure 1 we add the constraint $|a| = 1$, which is a nonconvex constraint set.

Assume that there are m movies and n users in the data set and every user rates every movie. If we store the ratings in a matrix Y such that $Y_{ij} := y[i, j]$ and the movie and user features in the vectors $\vec{a} \in \mathbb{R}^m$ and $\vec{b} \in \mathbb{R}^n$, then equation (1) is equivalent to

$$Y \approx \vec{a}\vec{b}^T. \quad (3)$$

Now consider the problem of fitting the problem by solving the optimization problem

$$\min_{\vec{a} \in \mathbb{R}^m, \vec{b} \in \mathbb{R}^n} \left\| Y - \vec{a}\vec{b}^T \right\|_F \quad \text{subject to} \quad \|\vec{a}\|_2 = 1. \quad (4)$$

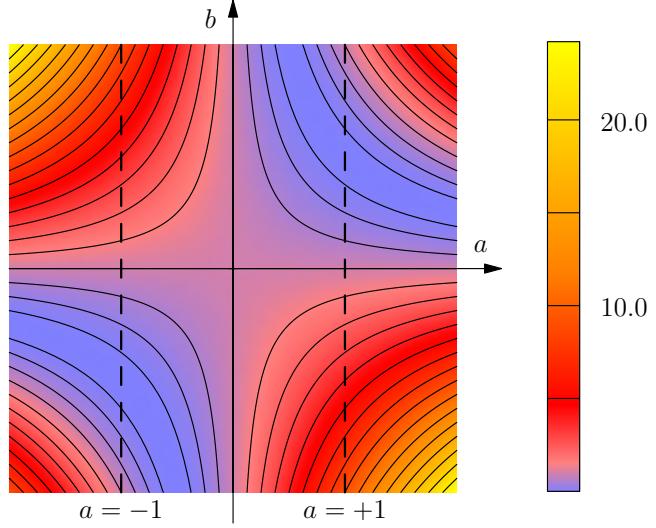


Figure 1: Heat map and contour map for the function $(1 - ab)^2$. The dashed line correspond to the set $|a| = 1$.

Note that $\vec{a}\vec{b}^T$ is a rank-1 matrix and conversely any rank-1 matrix can be written in this form where $\|\vec{a}\|_2 = 1$ (\vec{a} is equal to any of the columns normalized by their ℓ_2 norm). The problem is consequently equivalent to

$$\min_{X \in \mathbb{R}^{m \times n}} \|Y - X\|_F \quad \text{subject to} \quad \text{rank}(X) = 1. \quad (5)$$

By Theorem 2.10 in Lecture Notes 2 the solution X_{\min} to this optimization problem is given by the truncated singular-value decomposition (SVD) of Y

$$X_{\min} = \sigma_1 \vec{u}_1 \vec{v}_1^T, \quad (6)$$

where σ_1 is the largest singular value and \vec{u}_1 and \vec{v}_1 are the corresponding singular vectors. The corresponding solutions \vec{a}_{\min} and \vec{b}_{\min} to problem (4) are

$$\vec{a}_{\min} = \vec{u}_1, \quad (7)$$

$$\vec{b}_{\min} = \sigma_1 \vec{v}_1. \quad (8)$$

Example 1.1 (Movies). Bob, Molly, Mary and Larry rate the following six movies from 1 to 5,

$$A := \begin{pmatrix} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ 1 & 1 & 5 & 4 \\ 2 & 1 & 4 & 5 \\ 4 & 5 & 2 & 1 \\ 5 & 4 & 2 & 1 \\ 4 & 5 & 1 & 2 \\ 1 & 2 & 5 & 5 \end{pmatrix} \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} \quad (9)$$

To fit a low-rank model, we first subtract the average rating

$$\mu := \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n A_{ij}, \quad (10)$$

from each entry in the matrix to obtain a centered matrix C and then compute its singular-value decomposition

$$A - \mu \vec{1} \vec{1}^T = USV^T = U \begin{bmatrix} 7.79 & 0 & 0 & 0 \\ 0 & 1.62 & 0 & 0 \\ 0 & 0 & 1.55 & 0 \\ 0 & 0 & 0 & 0.62 \end{bmatrix} V^T. \quad (11)$$

where $\vec{1} \in \mathbb{R}^4$ is a vector of ones. 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 indeed the case:

$$\mu \vec{1} \vec{1}^T + \sigma_1 \vec{u}_1 \vec{v}_1^T = \begin{pmatrix} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ 1.34(1) & 1.19(1) & 4.66(5) & 4.81(4) \\ 1.55(2) & 1.42(1) & 4.45(4) & 4.58(5) \\ 4.45(4) & 4.58(5) & 1.55(2) & 1.42(1) \\ 4.43(5) & 4.56(4) & 1.57(2) & 1.44(1) \\ 4.43(4) & 4.56(5) & 1.57(1) & 1.44(2) \\ 1.34(1) & 1.19(2) & 4.66(5) & 4.81(5) \end{pmatrix} \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} \quad (12)$$

For ease of comparison the values of A are shown in brackets. The first left singular vector is equal to

$$\vec{u}_1 := \begin{pmatrix} \text{D. Knight} & \text{Spiderman 3} & \text{Love Act.} & \text{B.J.'s Diary} & \text{P. Woman} & \text{Superman 2} \\ -0.45 & -0.39 & 0.39 & 0.39 & 0.39 & -0.45 \end{pmatrix}.$$

This vector allows us to cluster the movies: movies with negative entries are similar (in this case action movies) and movies with positive entries are similar (in this case romantic movies).

The first right singular vector is equal to

$$\vec{v}_1 = \begin{pmatrix} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ 0.48 & 0.52 & -0.48 & -0.52 \end{pmatrix}. \quad (13)$$

This vector allows to cluster the users: negative entries indicate users that like action movies but hate romantic movies (Bob and Molly), whereas positive entries indicate the contrary (Mary and Larry). \triangle

1.2 Rank- r model

Our rank-1 matrix model is extremely simplistic. Different people like different movies. In order to generalize it we can consider r factors that capture the dependence between the ratings and the movie/user

$$y[i, j] \approx \sum_{l=1}^r a_l[i] b_l[j]. \quad (14)$$

The parameters of the model have the following interpretation:

- $a_l[i]$: movie i is positively (> 0), negatively (< 0) or not (≈ 0) associated to factor l .
- $b_l[j]$: user j likes (> 0), hates (< 0) or is indifferent (≈ 0) to factor l .

The model learns the factors directly from the data. In some cases, these factors may be interpretable— for example, they can be associated to the genre of the movie as in Example 1.1 or the age of the user— but this is not always the case.

The model (14) corresponds to a rank- r model

$$Y \approx AB, \quad A \in \mathbb{R}^{m \times r}, \quad B \in \mathbb{R}^{r \times n}. \quad (15)$$

We can fit such a model by computing the SVD of the data and truncating it. By Theorem 2.10 in Lecture Notes 2 the truncated SVD is the solution to

$$\min_{A \in \mathbb{R}^{m \times r}, B \in \mathbb{R}^{r \times n}} \|Y - AB\|_F \quad \text{subject to} \quad \|\vec{a}_1\|_2 = 1, \dots, \|\vec{a}_r\|_2 = 1. \quad (16)$$

However, the SVD provides an estimate of the matrices A and B that constrains the columns of A and the rows of B to be orthogonal. As a result, these vectors do not necessarily correspond to interpretable factors.

2 Matrix completion

2.1 Missing data

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 2. This problem is known as matrix completion.

At first glance, the problem of completing a matrix such as this one

$$\begin{bmatrix} 1 & ? & 5 \\ ? & 3 & 2 \end{bmatrix} \quad (17)$$

may seem completely ill posed. We can just fill in the missing entries arbitrarily! In more mathematical terms, the completion problem is equivalent to an underdetermined system of equations

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} M_{11} \\ M_{21} \\ M_{12} \\ M_{22} \\ M_{13} \\ M_{23} \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 2 \end{bmatrix}. \quad (18)$$



Figure 2: 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.

In order to solve the problem, we need to make an assumption on the structure of the matrix that we aim to complete. In compressed sensing we make the assumption that the original signal is sparse. In the case of matrix completion, we make the assumption that the original matrix is low rank. This implies that there exists a high correlation between the entries of the matrix, which may make it possible to infer the missing entries from the observations. As a very simple example, consider the following matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & ? & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ ? & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (19)$$

Setting the missing entries to 1 yields a rank 1 matrix, whereas setting them to any other number yields a rank 2 or rank 3 matrix.

The low-rank assumption implies that if the matrix has dimensions $m \times n$ then it can be factorized into two matrices that have dimensions $m \times r$ and $r \times n$. This factorization allows to encode the matrix using $r(m + n)$ parameters. If the number of observed entries is larger than $r(m + n)$ parameters then it may be possible to recover the missing entries. However, this is not enough to ensure that the problem is well posed.

2.2 When is matrix completion well posed?

The results of matrix completion will obviously depend on the subset of entries that are observed. For example, completion is impossible unless we observe at least one entry in each row and column.

To see why let us consider a rank 1 matrix for which we do not observe the second row,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ ? & ? & ? & ? \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ ? \\ 1 \\ 1 \end{bmatrix} [1 \ 1 \ 1 \ 1]. \quad (20)$$

As long as we set the missing row to equal the same value, we will obtain a rank-1 matrix consistent with the measurements. In this case, the problem is not well posed.

In general, we need samples that are distributed across the whole matrix. This may be achieved by sampling entries uniformly at random. Although this model does not completely describe matrix completion problems in practice (some users tend to rate more movies, some movies are very popular and are rated by many people), making the assumption that the revealed entries are random simplifies theoretical analysis and avoids dealing with adversarial cases designed to make deterministic patterns fail.

We now turn to the question of what matrices can be completed from a subset of entries samples uniformly at random. Intuitively, matrix completion can be achieved when the information contained in the entries of the matrix is *spread out* across multiple entries. If the information is very localized then it will be impossible to reconstruct the missing entries. Consider a simple example where the matrix is sparse

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 23 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (21)$$

If we don't observe the nonzero entry, we will naturally assume that it was equal to zero.

The problem is not restricted to sparse matrices. In the following matrix the last row does not seem to be correlated to the rest of the rows,

$$M := \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ -3 & 3 & -3 & 3 \end{bmatrix}. \quad (22)$$

This is revealed by the singular-value decomposition of the matrix, which allows to decompose it

into two rank-1 matrices.

$$M = U S V^T \quad (23)$$

$$= \begin{bmatrix} 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 8 & 0 \\ 0 & 6 \end{bmatrix} \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ -0.5 & 0.5 & -0.5 & 0.5 \end{bmatrix} \quad (24)$$

$$= 8 \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0 \end{bmatrix} \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \end{bmatrix} + 6 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} -0.5 & 0.5 & -0.5 & 0.5 \end{bmatrix} \quad (25)$$

$$= \sigma_1 U_1 V_1^T + \sigma_2 U_2 V_2^T. \quad (26)$$

The first rank-1 component of this decomposition has information that is very spread out,

$$\sigma_1 U_1 V_1^T = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (27)$$

The reason is that most of the entries of V_1 are nonzero and have the same magnitude, so that each entry of U_1 affects every single entry of the corresponding row. If one of those entries is missing, we can still recover the information from the other entries.

In contrast, the information in the second rank-1 component is very localized, due to the fact that the corresponding left singular vector is very sparse,

$$\sigma_2 U_2 V_2^T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -3 & 3 & -3 & 3 \end{bmatrix}. \quad (28)$$

Each entry of the right singular vector only affects one entry of the component. If we don't observe that entry then it will be impossible to recover.

This simple example shows that sparse singular vectors are problematic for matrix completion. In order to quantify to what extent the information is spread out across the low-rank matrix we define a coherence measure that depends on the singular vectors.

Definition 2.1 (Coherence). *Let $U S V^T$ be the singular-value decomposition of an $n \times n$ matrix M with rank r . The coherence μ of M is a constant such that*

$$\max_{1 \leq j \leq n} \sum_{i=1}^r U_{ij}^2 \leq \frac{n\mu}{r} \quad (29)$$

$$\max_{1 \leq j \leq n} \sum_{i=1}^r V_{ij}^2 \leq \frac{n\mu}{r}. \quad (30)$$

This condition was first introduced in [1]. Its exact formulation is not too important. The point is that matrix completion from uniform samples only makes sense for matrices which are incoherent, and therefore do not have spiky singular values. There is a direct analogy with the super-resolution problem, where sparsity is not a strong enough constraint to make the problem well posed and the class of signals of interest has to be further restricted to signals with supports that satisfy a minimum separation.

2.3 Minimizing the nuclear norm

We are interested in recovering low-rank matrices from a subset of their entries. Let \vec{y} be a vector containing the revealed entries and let Ω be the corresponding entries. Ideally, we would like to select the matrix with the lowest rank that corresponds to the measurements,

$$\min_{X \in \mathbb{R}^{m \times n}} \text{rank}(X) \quad \text{such that } X_\Omega = \vec{y}. \quad (31)$$

Unfortunately, this optimization problem is computationally hard to solve. Substituting the rank with the nuclear norm yields a tractable alternative:

$$\min_{X \in \mathbb{R}^{m \times n}} \|X\|_* \quad \text{such that } X_\Omega = \vec{y}. \quad (32)$$

The cost function is convex and the constraint is linear, so this is a convex program. In practice, the revealed entries are usually noisy. They do not correspond exactly to entries from a low-rank matrix. We take this into account by removing the equality constraint and adding a data-fidelity term penalizing the ℓ_2 -norm error over the revealed entries in the cost function,

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{2} \|X_\Omega - \vec{y}\|_2^2 + \lambda \|X\|_*, \quad (33)$$

where $\lambda > 0$ is a regularization parameter.

Example 2.2 (Collaborative filtering (simulated)). We now apply this method to the following completion problem:

$$\begin{array}{cccc} \text{Bob} & \text{Molly} & \text{Mary} & \text{Larry} \\ \left(\begin{array}{cccc} 1 & ? & 5 & 4 \\ ? & 1 & 4 & 5 \\ 4 & 5 & 2 & ? \\ 5 & 4 & 2 & 1 \\ 4 & 5 & 1 & 2 \\ 1 & 2 & ? & 5 \end{array} \right) & \begin{array}{l} \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array} & & (34) \end{array}$$

In more detail we apply the following steps:

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 (32).

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 pretty good,

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

For comparison the original ratings are shown in brackets. \triangle

2.4 Algorithms

In this section we describe a proximal-gradient method to solve Problem 33. Recall that proximal-gradient methods allow to solve problems of the form

$$\text{minimize } f(x) + g(x), \quad (36)$$

where f is differentiable and we can apply the proximal operator prox_g efficiently.

Recall that the proximal operator norm of the ℓ_1 norm is a soft-thresholding operator. Analogously, the proximal operator of the nuclear norm is applied by soft-thresholding the singular values of the matrix.

Theorem 2.3 (Proximal operator of the nuclear norm). *The solution to*

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{2} \|Y - X\|_F^2 + \tau \|X\|_* \quad (37)$$

is $\mathcal{D}_\tau(Y)$, obtained by soft-thresholding the singular values of $Y = U S V^T$

$$\mathcal{D}_\tau(Y) := U \mathcal{S}_\tau(S) V^T, \quad (38)$$

$$\mathcal{S}_\tau(S)_{ii} := \begin{cases} S_{ii} - \tau & \text{if } S_{ii} > \tau, \\ 0 & \text{otherwise.} \end{cases} \quad (39)$$

Proof. Due to the Frobenius norm term, the cost function is strictly convex. This implies that any point at which there exists a subgradient that is equal to zero is the solution to the optimization problem. The subgradients of the cost function at X are of the form,

$$X - Y + \tau G, \quad (40)$$

where G is a subgradient of the nuclear norm at X . If we can show that

$$\frac{1}{\tau} (Y - \mathcal{D}_\tau(Y)) \quad (41)$$

is a subgradient of the nuclear norm at $D_\tau(Y)$ then $D_\tau(Y)$ is the solution.

Let us separate the singular-value decomposition of Y into the singular vectors corresponding to singular values greater than τ , denoted by U_0 and V_0 and the rest

$$Y = U S V^T \quad (42)$$

$$= [U_0 \ U_1] \begin{bmatrix} S_0 & 0 \\ 0 & S_1 \end{bmatrix} [V_0 \ V_1]^T. \quad (43)$$

Note that $D_\tau(Y) = U_0(S_0 - \tau I)V_0^T$, so that

$$\frac{1}{\tau}(Y - D_\tau(Y)) = U_0 V_0^T + \frac{1}{\tau} U_1 S_1 V_1^T. \quad (44)$$

By construction all the singular values of $U_1 S_1 V_1^T$ are smaller than τ , so

$$\left\| \frac{1}{\tau} U_1 S_1 V_1^T \right\| \leq 1. \quad (45)$$

In addition, by definition of the singular-value decomposition $U_0^T U_1 = 0$ and $V_0^T V_1 = 0$. As a result, (44) is a subgradient of the nuclear norm at $D_\tau(Y)$ and the proof is complete. \square

Algorithm 2.4 (Proximal-gradient method for nuclear-norm regularization). *Let Y be a matrix such that $Y_\Omega = y$ and let us abuse notation by interpreting $X_\Omega^{(k)}$ as a matrix which is zero on Ω^c . We set the initial point $X^{(0)}$ to Y . Then we iterate the update*

$$X^{(k+1)} = \mathcal{D}_{\alpha_k \lambda} \left(X^{(k)} - \alpha_k \left(X_\Omega^{(k)} - Y \right) \right), \quad (46)$$

where $\alpha_k > 0$ is the step size.

Example 2.5 (Collaborative filtering (real)). The [Movielens data set](#) contains ratings from 671 users for 300 movies. The ratings are between 1 and 10. Figure 3 shows the results of applying algorithm 2.4 (as implemented by [this package](#)) using a training set of 9 135 ratings and evaluating the model on 1 016 test ratings. For large values of λ the model is too low rank and is not able to fit the data, so both the training and test error is high. When λ is too small, the model is not low rank, which results in overfitting: the observed entries are approximated by a high-rank model that is not able to predict the test entries. For intermediate values of λ the model achieves an average error of about 2/10. \triangle

2.5 Alternating minimization

Minimizing the nuclear norm to recover a low-rank matrix is an effective method but it has a drawback: it requires repeatedly computing the singular-value decomposition of the matrix, which can be computationally heavy for large matrices. A more efficient alternative is to parametrize the matrix as AB where $A \in \mathbb{R}^{m \times k}$ and $B \in \mathbb{R}^{k \times n}$, which requires fixing a value for the rank of the matrix k (in practice this can be set by cross validation). The two components A and B can then be fit by solving the optimization problem

$$\min_{\tilde{A} \in \mathbb{R}^{m \times k}, \tilde{B} \in \mathbb{R}^{k \times n}} \left\| (\tilde{A} \tilde{B})_\Omega - \vec{y} \right\|_2. \quad (47)$$

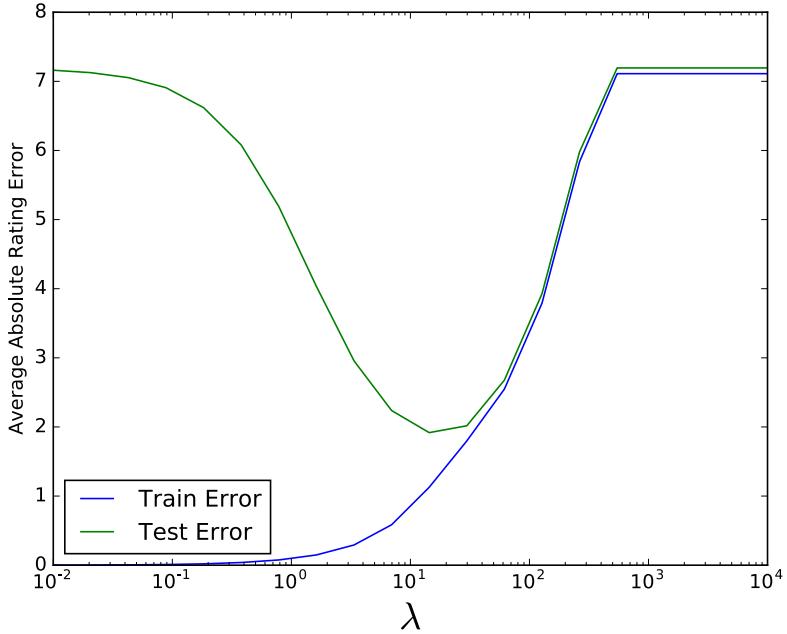


Figure 3: Training and test error of nuclear-norm-based collaborative filtering for the data described in Example 2.5.

This nonconvex problem is usually tackled by alternating minimization. Indeed, if we fix $\tilde{B} = B$ the optimization problem over \tilde{A} is just a least-squares problem

$$\min_{\tilde{A} \in \mathbb{R}^{m \times k}} \left\| (\tilde{A}B)_\Omega - \vec{y} \right\|_2 \quad (48)$$

and the same is true for the optimization problem over \tilde{B} if we fix $\tilde{A} = A$. Iteratively solving these least-squares problems allows to find a local minimum of the cost function. Under certain assumptions, one can even show that a certain initialization coupled with this procedure guarantees exact recovery, see [3] for more details.

3 Structured low-rank models

3.1 Nonnegative matrix factorization

As we discussed in Lecture Notes 2, PCA can be used to compute the main principal directions of a dataset, which can be interpreted as basis vectors that capture as much of the energy as possible. These vectors are constrained to be orthogonal. Unfortunately, as a result they are often not necessarily interpretable. For example, when we compute the principal directions and components of a data set of faces, they both may have negative values, so it is difficult to interpret the directions as *face atoms* that can be added to form a face. This suggests computing a decomposition where both atoms and coefficients are nonnegative, with the hope that this will allow us to learn a more interpretable model.

A nonnegative matrix factorization of the data matrix may be obtained by solving the optimization problem,

$$\text{minimize} \quad \|X - \tilde{A} \tilde{B}\|_F^2 \quad (49)$$

$$\text{subject to} \quad \tilde{A}_{i,j} \geq 0, \quad (50)$$

$$\tilde{B}_{i,j} \geq 0, \quad \text{for all } i, j \quad (51)$$

where $\tilde{A} \in \mathbb{R}^{d \times r}$ and $\tilde{B} \in \mathbb{R}^{r \times n}$ for a fixed r . This is a nonconvex problem which is computationally hard, due to the nonnegative constraint. Several methods to compute local optima have been suggested in the literature, as well as alternative cost functions to replace the Frobenius norm. We refer interested readers to [4].

Figure 4 shows the columns of the left matrix A , which we interpret as atoms that can be combined to form the faces, obtained by applying this method to the faces dataset from Lecture Notes 2 and compares them to the principal directions obtained through PCA. Due to the nonnegative constraint, the atoms resemble portions of faces (the black areas have very small values) which capture features such as the eyebrows, the nose, the eyes, etc.

Example 3.1 (Topic modeling). Topic modeling aims to learn the thematic structure of a text corpus automatically. We will illustrate this application with 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 .

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

Computing the singular-value decomposition of the matrix— after subtracting the mean of each entry as in (11)— we determine that the matrix is approximately low rank

$$A = USV^T = U \begin{bmatrix} 23.64 & 0 & 0 & 0 \\ 0 & 18.82 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14.23 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3.63 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.03 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.36 \end{bmatrix} V^T \quad (52)$$

Unfortunately the singular vectors do not have an intuitive interpretation as in Section ???. In

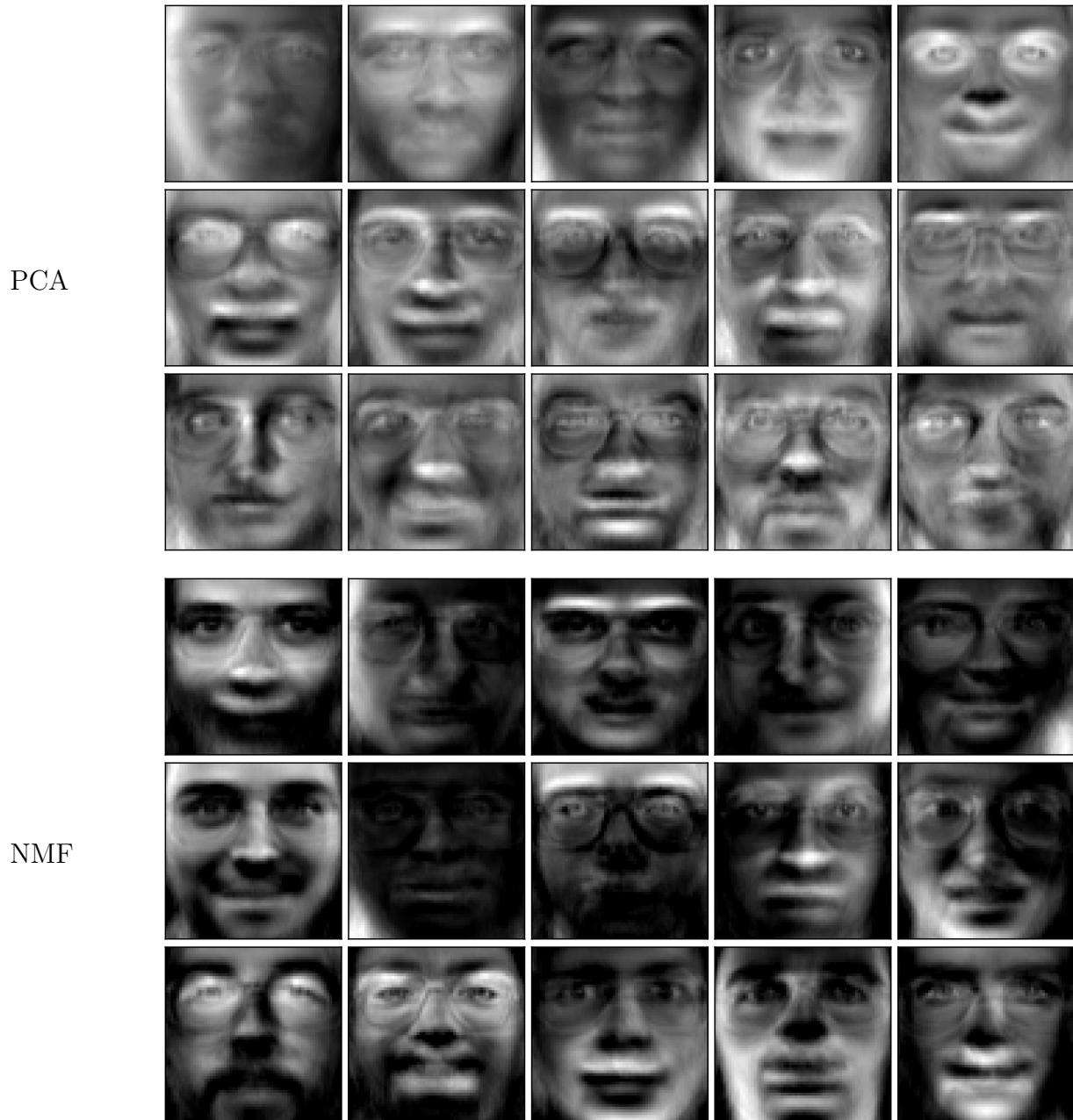


Figure 4: Atoms obtained by applying PCA and nonnegative matrix factorization to the faces dataset in Lecture Notes 2.

particular, they do not allow to cluster the words

$$\begin{aligned} U_1 &= \begin{pmatrix} a & b & c & d & e & f \\ -0.24 & -0.47 & -0.24 & -0.32 & -0.58 & -0.47 \end{pmatrix} \\ U_2 &= \begin{pmatrix} 0.64 & -0.23 & 0.67 & -0.03 & -0.18 & -0.21 \end{pmatrix} \\ U_3 &= \begin{pmatrix} -0.08 & -0.39 & -0.08 & 0.77 & 0.28 & -0.40 \end{pmatrix} \end{aligned} \quad (53)$$

or the articles

$$\begin{aligned} V_1 &= \begin{pmatrix} \text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band} \\ -0.18 & -0.24 & -0.51 & -0.38 & -0.46 & -0.34 & -0.2 & -0.3 & -0.22 \end{pmatrix} \\ V_2 &= \begin{pmatrix} 0.47 & 0.01 & -0.22 & -0.15 & -0.25 & -0.07 & 0.63 & -0.05 & 0.49 \end{pmatrix} \\ V_3 &= \begin{pmatrix} -0.13 & 0.47 & -0.3 & -0.14 & -0.37 & 0.52 & -0.04 & 0.49 & -0.07 \end{pmatrix} \end{aligned} \quad (54)$$

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: it makes sense to add atoms corresponding to different topics to approximate the word count of a document but not to subtract them. Following this intuition, we apply nonnegative matrix factorization to obtain two matrices $W \in \mathbb{R}^{m \times k}$ and $H \in \mathbb{R}^{k \times n}$ such that

$$M \approx WH, \quad W_{i,j} \geq 0, \quad 1 \leq i \leq m, \quad 1 \leq j \leq k, \quad (55)$$

$$H_{i,j} \geq 0, \quad 1 \leq i \leq k, \quad 1 \leq j \leq n. \quad (56)$$

In our example, we set $k = 3$. H_1 , H_2 and H_3 can be interpreted as word-count atoms, but also as coefficients that weigh the contribution of W_1 , W_2 and W_3 .

$$\begin{aligned} H_1 &= \begin{pmatrix} \text{singer} & \text{GDP} & \text{senate} & \text{election} & \text{vote} & \text{stock} & \text{bass} & \text{market} & \text{band} \\ 0.34 & 0 & 3.73 & 2.54 & 3.67 & 0.52 & 0 & 0.35 & 0.35 \end{pmatrix} \\ H_2 &= \begin{pmatrix} 0 & 2.21 & 0.21 & 0.45 & 0 & 2.64 & 0.21 & 2.43 & 0.22 \end{pmatrix} \\ H_3 &= \begin{pmatrix} 3.22 & 0.37 & 0.19 & 0.2 & 0 & 0.12 & 4.13 & 0.13 & 3.43 \end{pmatrix} \end{aligned} \quad (57)$$

The latter interpretation allows to cluster the words into topics. The first topic corresponds 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 the 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)

$$\begin{aligned} W_1 &= \begin{pmatrix} a & b & c & d & e & f \\ 0.03 & 2.23 & 0 & 0 & 1.59 & 2.24 \end{pmatrix} \\ W_2 &= \begin{pmatrix} 0.1 & 0 & 0.08 & 3.13 & 2.32 & 0 \end{pmatrix} \\ W_3 &= \begin{pmatrix} 2.13 & 0 & 2.22 & 0 & 0 & 0.03 \end{pmatrix} \end{aligned} \quad (58)$$

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

singer	GDP	senate	election	vote	stock	bass	market	band	Art.
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)	a
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)	b
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)	c
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)	d
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)	e
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)	f

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

4 Sparse principal-component analysis

In certain cases, it may be desirable to learn sparse atoms that are able to represent a set of signals. In the case of the faces dataset, this may force the representation to isolate specific face features such as the mouth, the eyes, etc. In order to fit such a model, we can incorporate a sparsity constraint on the atoms by using the ℓ_1 norm

$$\text{minimize} \quad \left\| X - \tilde{A} \tilde{B} \right\|_2^2 + \lambda \sum_{i=1}^k \left\| \tilde{A}_i \right\|_1 \quad (59)$$

$$\text{subject to} \quad \left\| \tilde{A}_i \right\|_2 = 1, \quad 1 \leq i \leq k. \quad (60)$$

Due to the sparsity-inducing constraint, this problem is computationally hard, as in the case of nonnegative matrix factorization. We refer the interested reader to [6] for algorithms to compute local minima. Figure 5 shows the atoms obtained by applying this method to the faces dataset from Figure 4. The model indeed learns very localized atoms that represent face features.

References

The tutorial [5] is an excellent reference on the application of matrix-decomposition techniques in machine learning and image processing. Chapters 7 and 8 of [2] describe low-rank models in statistics. The numerical experiments shown in these notes were implemented using scikit-learn, which is available online at <http://scikit-learn.org>. In particular, a script to apply different matrix-decomposition techniques to the faces dataset is available [here](#).

- [1] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.
- [2] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC Press, 2015.

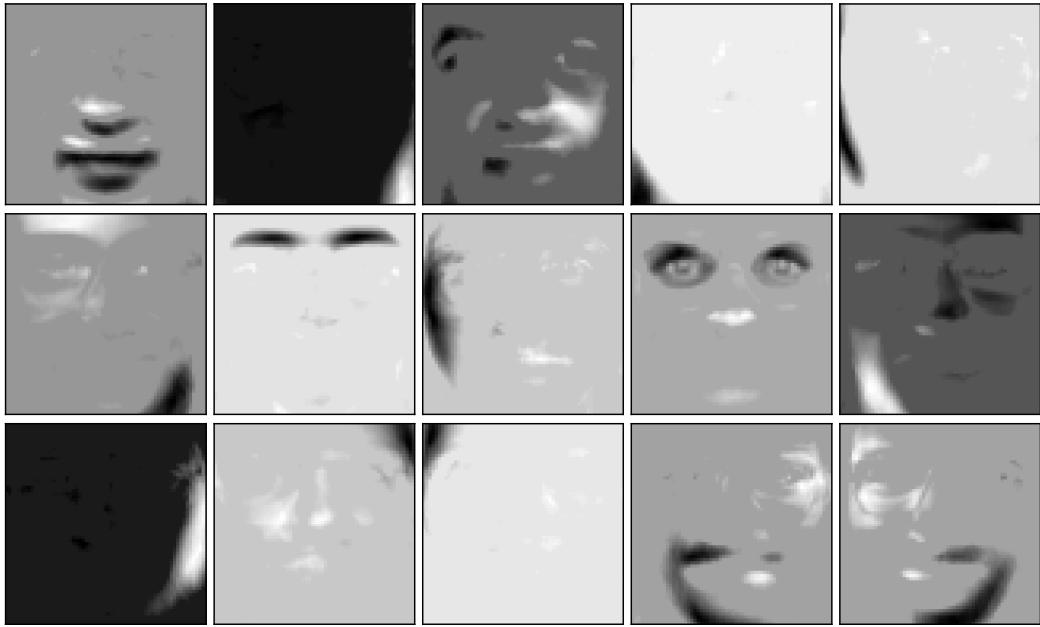


Figure 5: Atoms obtained by applying sparse PCA to the faces dataset from Figure 4.

- [3] P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674. ACM, 2013.
- [4] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.
- [5] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. *arXiv preprint arXiv:1411.3230*, 2014.
- [6] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.