Professor Peter Ramadge Scribe: Amy Hua

*** denotes a chapter that is currently being edited.

Contents

1 Preliminaries

This course is an introduction to machine learning and pattern recognition. We will cover fundamental results and illustrate them with a variety of applications.

1.1 Data and learning

1.1.1 Data as vectors

Generally, we assume that data lives in $\mathbb{R}^n$ and labels live in $\mathbb{R}^k$.

Example 1.1. We have a cylinder with nuts, bolts, washers, and ball bearings, and we listen to the sound being produced by interactions in the container. Let's suppose that we can capture analog symbols and sample at 44,100 samples per second (a pretty standard sample rate for sound)—let's call this raw data $w \in \mathbb{R}^n$, $n = 44,100$. Note that time shifts, and thus the phase, should not be important, so we should take a discrete Fourier transform (DFT). $\hat{w} \in \mathbb{C}^n$, $n = 44,100 \to v = |\hat{w}| \in \mathbb{R}^n_t$. Now, our processed data is $x \in \mathbb{R}^m$, $m = 22,051$, one of which is called an "example." Data has now been encoded as a vector in $\mathbb{R}^m$.

Recall that each piece of data has a label (e.g., "washers")—we can convert these to discrete labels in some vector space via **one-hot embedding**. Given q different object types, we can assign an object type j (e.g., "washers") the label $e_j \in \mathbb{R}^q$, $j = 1, \dots, q$, where e_j is the j-th standard basis vector for $\mathbb{R}^q$. Thus, a set of labelled data can be expressed as $\{(x_j, y_j)\}_{j=1}^m$, where $x_j \in \mathbb{R}^n$ represents the j-th signal, and $y_j \in \{e_1, \dots, e_q\}$ is its one-hot label.

A nice feature that comes out of this conversion is that we can express probability distributions in terms of weighted sums of these basis vectors. We can express example k as x_k and its label as y_k, where $(x_k, y_k) \in \mathbb{R}^n \times \mathbb{R}^k$.

Example 1.2. (Email spam). Email $\equiv$ story of words and punctuation. We pick a list of words $L = (w_1, w_2, \dots, w_p)$, called a **dictionary**. Note that w_i can be a punctuation symbol. Each email can be represented as a vector of counts of each word (or punctuation symbol) $\Rightarrow (c_1, c_2, \dots, c_p) \in \mathbb{R}^p$. We want to treat short and long emails in the same way, so our final processing step can be to normalize this to $(q_1, q_2, \dots, q_p) \in \mathbb{R}^p$ (these are starting to look like probabilities).

For the spam problem, labels are "spam" and "not spam"—a binary classification. (Later, we can try assigning probabilities as labels.) The embedding of the labels into $\mathbb{R}$ is just $1, -1 \in \mathbb{R}$, but we can also assign labels by document types, such as "sports," "tech," "economics." With k labels, our one-hot embedding is $y = e_j \in \mathbb{R}^k$.

Labelled data is represented by $\{(x_j, y_j)\}_{j=1}^m$, $x_j \in \mathbb{R}^n$, $y_j \in L$. We can separate the data into **training data** and **testing data**. We want to learn a function $\hat{y} : \mathbb{R}^n \to L$ (an estimate of the true function y), where L is a finite set of labels. Taking a new example x (from the testing subset), we want to predict its label $\hat{y}(x)$, and since we know its true label y, we can then calculate the accuracy of the function. Thus, our goal is to learn such a classifier; note that only the training subset is used to learn the classifier.

We want equal label counts to occur within each the training and the testing data, to try to do the split in a fair and representative way. This works very well as long as m is very big

(let's say 50,000-100,000), but not as well when m is smaller. How can we compensate? With **k-fold cross-validation**: we can divide the data into $k-1$ training "folds" and 1 testing fold, in k different ways, and then average the results. **Stratified k-fold cross-validation** is a variation of k-fold cross-validation that returns stratified folds: each set contains about the same proportion of samples of each target class as the complete set. A compact way of representing this is:

$$X = \begin{pmatrix} x_1 & x_2 & \cdots & x_m \end{pmatrix} \in \mathbb{R}^{n \times m},$$
$$Y = \begin{pmatrix} y_1 & y_2 & \cdots & y_m \end{pmatrix} \in \mathbb{R}^{k \times m}.$$

$z = Xw$ is a linear combination of the columns of X (so X is a matrix, w is a vector, and z is a vector).

In Python—in particular, when using `scikit-learn` (the machine learning toolbox for Python)—each example must be provided as a row:

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix}, \; Y = \begin{pmatrix} y_1^T \\ \vdots \\ y_m^T \end{pmatrix}.$$

A feature is thus represented by a column in X.

Data standardization is an important part of processing the data. For instance, given $x = \begin{pmatrix} x(1) \\ x(2) \end{pmatrix} \in \mathbb{R}^2$, where $x(1)$ is feature 1 (ex: height) and $x(2)$ is feature 2 (ex: weight), one of the first steps after pre-processing is centering the data. Note that these features might have different units. The **sample mean**, or **empirical mean**, is the vector $\hat{\mu} = \frac{1}{m}\sum_{j=1}^{m} x_j$, so **centering** the data just involves calculating $z_j = x_j - \hat{\mu}$. The centered data is the set $\{z_j\}_{j=1}^m$, which just involves moving the origin of the coordinate system to the sample mean.

Since we don't want to overrepresent the importance of one of the features, and to compensate for the different units of the features, we scale the centered data such that there is unit variance along each feature's dimension. The sample variances of the different features are:

$$\hat{\sigma}_1^2 = \frac{1}{m}\sum_{j=1}^{m}(z_j(1))^2,$$
$$\hat{\sigma}_2^2 = \frac{1}{m}\sum_{j=1}^{m}(z_j(2))^2.$$

We can then scale each centered feature z_j by the inverse of the sample standard deviation:

$$w_j = \begin{pmatrix} 1/\hat{\sigma}_1 & 0 \\ 0 & 1/\hat{\sigma}_2 \end{pmatrix} z_j, \; j = 1, 2.$$

1.1.2 Some simple classifiers

Why do we want to embed the data in $\mathbb{R}^n$? We want to exploit the algebraic and geometric structure of the Euclidean space. We can see that even simple classifiers can be used for this effect:

Example 1.3. (Nearest neighbor classifier). Our training data is $\{(x_j, y_j)\}_{j=1}^m$, $x_j \in \mathbb{R}^n$, $y_j \in L$, and our new example is $x \in \mathbb{R}^n$. The nearest neighbor classifier predicts a label for x by finding the closest point to x among the training examples and then using the label of the training point for the label of x. Our learning function is just $\hat{y}(x) = y_{j^*}$, where $j^* = \arg\min_j \|x - x_j\|_2^2$. Note that we may want to pre-process our data to make computing j^* more efficient.

Example 1.4. (Nearest class mean classifier). Let $L = \{+1, -1\}$ (binary classification). Our training data is $\{(x_j, y_j)\}_{j=1}^m$. For $x_1, \ldots, x_{m/2}$, $y = 1$ (Class 1), and for $x_{m/2+1}, \ldots, x_m$, $y = -1$ (Class -1). We are using the labels just to cluster the data into two classes.

$$\hat{\mu}_1 = \frac{1}{m/2} \sum_{j=1}^{m/2} x_j,$$

$$\hat{\mu}_2 = \frac{1}{m/2} \sum_{j=m/2+1}^{m} x_j.$$

With a new x, $\hat{y}(x) = +1$ if $\|x - \hat{\mu}_1\|_2 \leq \|x - \hat{\mu}_2\|_2$, and $\hat{y}(x) = -1$ otherwise. Note that we can also write the classification rule as

$$\hat{y}(x) = \text{sign}\left(\langle \mu_1, x \rangle - \langle \mu_2, x \rangle + \frac{1}{2} \left(\|\mu_2\|_2^2 - \|\mu_1\|_2^2 \right) \right).$$

Are there any situations in which this is the optimal classifier? Yes. They could be very narrowly-defined situations, but there are such situations. The Naive Bayes classifier, a slight generalization of this classifier, is quite robust under some circumstances. We could have two circularly symmetric covariance matrices for each of the two clusters, drawn from Gaussian distributions; if we know the means, which are Gaussian, and variances are equal, the Naive Bayes classifier is actually the optimal classifier.

1.2 The algebraic structure of $\mathbb{R}^n$

A vector space is $(\mathbb{R}^n, \mathbb{R})$. $x \in \mathbb{R}^n$ is just an n-tuple of real numbers, or scalars from the field $\mathbb{R}$. By default, we refer to column vectors; if we want to refer to a row vector, we'll put an arrow to denote the vector. $\mathbb{R}^n$ has two algebraic operations:

1. **Vector addition**:
$$(x + y)(j) = x(j) + y(j).$$

2. **Scalar multiplication**:
$$(\alpha x)(j) = \alpha x(j),$$
where $x(j)$ is the j-th element of vector x.

Note that notation is not that important—eventually, we will often deviate from our notation, so we must rely on context for meaning rather than relying on convention. But, we'll generally use $u, v, x, y, w, \ldots$ for vectors, $\alpha, \beta, \gamma, \ldots$ for scalars, and $\mathcal{U}, \mathcal{V}, \ldots$ for subspaces. We will also often use $\mathbb{R}^n$ as the example, but note that these concepts also work for other vector spaces, such as:

- $(\mathbb{C}^n, \mathbb{C})$, the set of n-tuples of complex numbers with scalars in the field $\mathbb{C}$
- $(\mathbb{R}^{m\times n}, \mathbb{R})$, the set of $m \times n$ real matrices with scalars in the field $\mathbb{R}$
- $(\mathbb{C}^{m\times n}, \mathbb{C})$, the set of $m \times n$ complex matrices with scalars in the field $\mathbb{C}$.

Definition 1.5. (Linear combination). A linear combination of vectors $x_1, \ldots, x_k \in \mathbb{R}^n$ and scalars $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$ yields the vector $x = \sum_{j=1}^{k} \alpha_j x_j$.

Definition 1.6. (Span). The span of a set of vectors is the set of all linear combinations of the vectors:

$$\text{span}(x_1, \ldots, x_k) = \{\text{all linear combinations of } x_1, \ldots, x_k\}.$$

Definition 1.7. (Subspace). A subspace of $\mathbb{R}^n$ is a subset $\mathcal{U} \subseteq \mathbb{R}^n$ containing the span of any of its elements, i.e. a subspace is a subset that is closed under vector addition and scalar multiplication. A subspace is itself a vector space.

Every vector space always has (at least) the 0 vector (take a vector in the vector space that is not the 0 vector, scale by 0, and we get the 0 vector—thus, the 0 vector must be in the subspace). Thus, the smallest possible vector space is the subspace containing only the zero vector. Two trivial subspaces of $\mathbb{R}^1$ are the 0 vector (the zero subspace) and the vector space itself. In $\mathbb{R}^2$, a subspace is any line through the origin (we can add two vectors that go through the origin in $\mathbb{R}^2$, and the sum vector will be in the vector space; we can scale any vector that goes through the origin, and the resulting vector will be in the vector space). In $\mathbb{R}^3$, a subspace is any plane through the origin (we can specify the plane by its normal vector); other subspaces of $\mathbb{R}^3$ are the 0 vector, lines through the origin, and $\mathbb{R}^3$.

We can specify a subspace in two ways:

1. For any set of vectors $x_1, \ldots, x_k \in \mathbb{R}^n$, $\mathcal{U} = \text{span}(x_1, \ldots, x_k)$ is a subspace of $\mathbb{R}^n$.
2. For fixed scalars $\alpha_i,\ i = 1, \ldots, k,\ \beta_i, i = 1, \ldots, k, \ldots$, the set

$$\mathcal{U} = \left\{ x \in \mathbb{R}^n : \sum \alpha_j x(j) = 0,\ x : \sum \beta_j x(j) = 0,\ \ldots \right\}$$

   is a subspace of $\mathbb{R}^n$.

Definition 1.8. (Subspace sum and subspace intersection). Let $\mathcal{U}$ and $\mathcal{V}$ be subspaces of $\mathbb{R}^n$.

- $\mathcal{U} + \mathcal{V}$ is the set of all vectors formed by a linear combination of a vector in $\mathcal{U}$ and a vector in $\mathcal{V}$:

$$\mathcal{U} + \mathcal{V} = \{x : x = u + v,\ u \in \mathcal{U},\ v \in \mathcal{V}\}.$$

- $\mathcal{U} \cap \mathcal{V}$ is the set of all vectors in both $\mathcal{U}$ and $\mathcal{V}$:

$$\mathcal{U} \cap \mathcal{V} = \{x : x \in \mathcal{U} \text{ and } x \in \mathcal{V}\}.$$

Lemma 1.9. *Take two subspaces $\mathcal{U}, \mathcal{V} \in \mathbb{R}^n$. $\mathcal{U} + \mathcal{V}$ and $\mathcal{U} \cap \mathcal{V}$ are also subspaces of $\mathbb{R}^n$. $\mathcal{U} + \mathcal{V}$ is the smallest subspace containing both $\mathcal{U} + \mathcal{V}$. $\mathcal{U} \cap \mathcal{V}$ is the largest subspace contained in both $\mathcal{U}$ and $\mathcal{V}$.*

Definition 1.10. (Linear independence). A set of vectors $x_1, \ldots, x_k$ is linearly independent if

$$\sum_{j=1}^{k} \alpha_j x_j = 0 \Longrightarrow \alpha_j = 0,\ j = 1, \ldots, k.$$

Note that a linearly independent set cannot contain the zero vector. We can also specify x with

$$x = \sum_{j=1}^{k} \alpha_j x_j,$$

where α_j are unique and called the **coordinates** of x.

Lemma 1.11. *The following conditions are equivalent:*

1. $\{x_1, \ldots, x_k\}$ *is linearly independent.*
2. *Each* $x \in$ *span*$\{x_1, \ldots, x_k\}$ *is a unique linear combination of* $x_1, \ldots, x_k$*; that is,*

$$\sum_{j=1}^{k} \alpha_j x_j = \sum_{j=1}^{k} \beta_j x_j \Longrightarrow \alpha_j = \beta_j,\ j = 1, \ldots, k.$$

3. *No element of* $\{x_1, \ldots, x_k\}$ *can be written as a linear combination of the others.*

Definition 1.12. (Basis). A basis for $\mathcal{U}$ is linearly independent set of vectors that spans $\mathcal{U}$. Every vector in $\mathcal{U}$ can be represented as a linear combination of the basis vectors, and linear independence ensures that this representation is unique.

Definition 1.13. (Coordinates). Let $\{x_j\}_{j=1}^n$ be a basis for $\mathbb{R}^n$. The coordinates of $x \in \mathbb{R}^n$ with respect to this basis are the unique scalars α_j such that $x = \sum_{j=1}^n \alpha_j x_j$. Every vector uniquely determines and is uniquely determined by its coordinates.

Definition 1.14. The **standard basis** for $\mathbb{R}^n$ is $\{e_1, e_2, \ldots, e_n\}$, where

$$e_j(k) = \begin{cases} 1, \text{ if } k = j; \\ 0, \text{ otherwise.} \end{cases}$$

Lemma 1.15. *Every nonzero subspace* $\mathcal{U} \subseteq \mathbb{R}^n$ *has a basis, and every such basis has the same number of elements.*

Definition 1.16. (Dimension). Every basis for the same subspace $\mathcal{U}$ has the same number of elements, which we call the dimension of $\mathcal{U}$.

1.3 Geometry: inner products and norms

We will briefly cover the geometry of $\mathbb{R}^n$, defined by the Euclidean inner product and norm. These concepts also generalize to the vector spaces of complex and real matrices.

Definition 1.17. (Inner product). For two vectors x and y, the inner product is the scalar

$$\langle x, y \rangle = \sum_{j=1}^{n} x(j)y(j).$$

($\langle \cdot, \cdot \rangle : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$.) We can generalize to the complex space $\mathbb{C}^n$:

$$\langle x, y \rangle = \sum_{j=1}^{n} x(j)\overline{y}(j) = x^T \overline{y},$$

where $\overline{y}$ is the complex conjugate of y and $x^T\overline{y}$ denotes the matrix product.

Lemma 1.18. *(Properties of the inner product). For $x, y, z \in \mathbb{C}^n$ and $\alpha \in \mathbb{C}$,*

1. $\langle x, x \rangle \geq 0$ *with equality if and only if* $x = 0$.
2. $\langle x, y \rangle = \overline{\langle y, x \rangle}$.
3. $\langle \alpha x, y \rangle = \alpha \langle x, y \rangle$.
4. $\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$.

Definition 1.19. (Euclidean norm). The inner product specifies the corresponding Euclidean norm via

$$\|x\| = (\langle x, x \rangle)^{1/2} = \left(\sum_{j=1}^{n} |x(j)|^2 \right)^{1/2}.$$

The Euclidean norm is also often denoted by $\| \cdot \|_2$ and referred to as the 2-norm.

Lemma 1.20. *(Properties of the norm). A norm satisfies three properties. For $x, y \in \mathbb{C}^n$ and $\alpha \in \mathbb{C}$,*

1. $\|x\| \geq 0$ *with equality if and only if* $\Longleftrightarrow x = 0$ ***(positivity)***.
2. $\|\alpha x\| = |\alpha| \ \|x\|$ ***(scaling)***.
3. $\|x + y\| \leq \|x\| + \|y\|$ ***(triangle inequality)***.

Example 1.21. Some examples of norms follow:

- (1-norm). $\|x\|_1 = (\sum_{i=1}^{n} |x(i)|)$.
- (2-norm). $\|x\|_2 = (\sum_{i=1}^{n} x(i)^2)^{1/2}$.
- (p-norm). $\|x\|_p = (\sum_{i=1}^{n} |x(i)|^p)^{1/p}$.
- (Max-norm). $\|x\|_\infty = \max_i |x(i)|$.
- (P-norm). $\|x\|_P = (x^T P x)^{1/2}$, where $P \in \mathbb{R}^{n \times n}$ is symmetric positive definite.

Remark 1.22. Note the following properties of norms:

- Every norm is convex.
- $\{x : \|x\| \leq \alpha\}$, the α-sublevel set, is convex.

Definition 1.23. (Cauchy-Schwarz inequality). For all $x, y \in \mathbb{C}^n$,

$$|\langle x, y\rangle| \leq \|x\| \ \|y\|.$$

Equivalently,

$$-1 \leq \frac{\langle x, y\rangle}{\|x\| \ \|y\|} \leq 1.$$

From this point forward, we drop down to the real space $\mathbb{R}^n$.

Remark 1.24. The inner product gives us angles, while the norm gives us distances. Note that the norm $\|x\|$ is the "size" of x. $\|x - y\|$ is the distance between x and y. The triangle inequality can be seen easily with an illustration. If $\|x\| = 1$, x is a **unit vector**. The set $\{x : \|x\| = 1\}$ is a **unit sphere**.

Definition 1.25. (Orthogonal). A set of vectors $\{x_1, \ldots, x_k\}$ are orthogonal if $\forall\ x_i, x_j,\ i \neq j$, $\langle x_i, x_j\rangle = 0$.

Remark 1.26. Note that an orthogonal set could include the zero vector, so it does not have to be a basis.

Theorem 1.27. *(Pythagoras's theorem). If $\{x_1, \ldots, x_k\}$ is an orthogonal set, then*

$$\left\|\sum_{j=1}^{k} x_j\right\|^2 = \sum_{j=1}^{k} \|x_j\|^2.$$

Definition 1.28. (Orthonormal). $\{x_1, \ldots, x_k\}$ is an orthonormal set if it is orthogonal and $\|x_j\| = 1,\ j = 1, \ldots, k$.

Lemma 1.29. *An orthonormal set is linearly independent.*

Proof. Let $\{x_1, \ldots, x_k\}$ be an orthonormal set. Suppose that $\sum_{j=1}^{k} \alpha_j x_j = 0$. For each x_i, $i = 1, \ldots, k$, we have

$$0 = \left\langle \sum_{j=1}^{k} \alpha_j x_j, x_i \right\rangle = \alpha_i.$$

□

Definition 1.30. (Orthonormal basis). An orthonormal basis for $\mathbb{C}^n$ is a basis of orthonormal vectors. Since an orthonormal set is always linearly independent, any set of n orthonormal vectors is an orthonormal basis for $\mathbb{C}^n$.

Remark 1.31. $\{x_1, \ldots, x_k\}$ is an orthonormal basis of $\text{span}\{x_1, \ldots, x_k\} = \mathcal{U}$, so by Pythagoras,

$$\left\|\sum_{i=1}^{k} \alpha_i x_i\right\|^2 = \left\langle \sum_{i=1}^{k} \alpha_i x_i, \sum_{j=1}^{k} \alpha_j x_j \right\rangle = \sum_{i=1}^{k} \langle \alpha_i x_i, \alpha_i x_i\rangle = \sum_{i=1}^{k} \alpha_i^2.$$

Remark 1.32. (Coordinates).

$$\forall\ x \in \mathcal{U},\ x = \sum_{j=1}^{k} \alpha_j x_j,\ \langle x_i, x\rangle = \sum_{j=1}^{k} \alpha_j \langle x_i, x_j\rangle = \alpha_i.$$

Thus, we have a unique formula $\langle x_i, x\rangle = \alpha_i$ to find the coordinate of x with respect to the basis element x_i.

Now, let's generalize to matrices. Take $A, B \in \mathbb{C}^{m\times n}$. We want Euclidean geometry, so how should we express the norm?

Definition 1.33. (Inner product on $\mathbb{C}^{m\times n}$). An inner product on the vector space of complex matrices is given by

$$\langle A, B\rangle = \sum_{i}^{m}\sum_{j}^{n} A(i,j)\overline{B}(i,j).$$

Definition 1.34. (Frobenius norm). The Euclidean norm induced from an inner product is

$$\|A\|_F = (\langle A, A\rangle)^{1/2} = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} |A(i,j)|^2\right)^{1/2}.$$

Lemma 1.35. *The inner product of matrices A and B is $\langle A, B\rangle = Tr(A^T\overline{B}) = Tr(B^*A)$, where $B^* = \overline{B}^T$.*

Proof. Hint: Check that the elements on the diagonal are $A(i,j)\overline{B}(i,j)$, from the expression in the definition. □

Let's say we have an orthonormal basis $\{X_1, \ldots, X_k\}, X_i \in \mathbb{R}^n$, and form an $n \times k$ matrix $Q = [X_1, X_2, \ldots, X_k]$. Some properties of Q are:

- $Q^TQ = I_k$ (these matrices are "conforming" since Q^T is $k \times n$ and Q is $n \times k$). This property is satisfied since Q has orthonormal columns.
- $QQ^T = \sum_{j=1}^{k} x_j x_j^T \neq I_n$, where QQ^T is $n \times n$. This is not the identity matrix, since the rank is k, but full rank is n. To make this equal to the identity, we can make $Q = [X_1, \ldots, X_n]$, an orthonormal set of size n. Then, we would have $QQ^T = I$.

Definition 1.36. (Orthogonal matrix). A square real matrix $Q \in \mathbb{R}^{n\times n}$ is orthogonal if it satisfies $Q^TQ = I_n$ and $QQ^T = I_n$. If $Q^TQ = I_n$, we say that the columns of Q form an orthonormal basis, and if $QQ^T = I_n$, we say that the rows of Q form an orthonormal basis. Note that $Q^T = Q^{-1}$. We denote the set of $n \times n$ orthogonal matrices by $\mathcal{O}_n$.

Lemma 1.37. *Let $Q \in \mathcal{O}_n$. Then, for any $x, y \in \mathbb{R}^n$,*

$$\langle Qx, Qy\rangle = \langle x, y\rangle,$$
$$\|Qx\| = \|x\|.$$

Definition 1.38. (Unitary matrix). We previously defined an orthogonal matrix in the real space, but for complex matrices, we need to deal with complex conjugates. A square complex matrix $Q \in \mathbb{C}^{n\times n}$ is called a unitary matrix if $Q^*Q = QQ^* = I_n$, where $Q^* = \overline{Q}^T$.

Lemma 1.39. *If Q is a unitary matrix, we also have that for any $x, y \in \mathbb{C}^n$, $\langle Qx, Qy\rangle = \langle x, y\rangle$ and $\|Qx\| = \|x\|$.*

1.4 Orthogonal projection

Our goal is to solve: given a data vector x and a subspace $\mathcal{U}$, what is the closest point to x in $\mathcal{U}$?

1.4.1 Simplest instance

Our vector space is $\mathbb{R}^n$. We first cover the simplest example of orthogonal projection: given a line through the origin, a subspace $\mathcal{U}$ of $\mathbb{R}^n$, we want to minimize the distance between a point and the line. We can set this concept up as an optimization problem:

$$\begin{aligned} \min_{z \in \mathbb{R}^n} \quad & \frac{1}{2}\|x - z\|^2 \\ \text{s.t.} \quad & z \in \mathcal{U}. \end{aligned}$$

(The $\frac{1}{2}$ is there for simplification when we take the derivative.) Another way of expressing $z = \mathcal{U}$ is in terms of a linear combination of the basis vectors of $\mathcal{U}$, i.e., choosing an orthonormal basis for the subspace. This case is trivial: we can just write $z = \alpha u,\ \|u\| = 1$. We can expand the objective function to get

$$\begin{aligned} \frac{1}{2}\|x - z\|^2 &= \frac{1}{2}\langle x - z, x - z\rangle \\ &= \frac{1}{2}\left(\langle x, x\rangle - \langle x, z\rangle - \langle z, x\rangle + \langle z, z\rangle\right) \\ &= \frac{1}{2}\left(\|x\|^2 - 2\langle z, x\rangle + \|z\|^2\right) \\ &= \frac{1}{2}\left(\|x\|^2 - 2\alpha\langle u, x\rangle + \alpha^2\right), \end{aligned}$$

where we use $z = \alpha u$ in the second-to-last step. Differentiating $\frac{1}{2}(\|x\|^2 - 2\alpha\langle u, x\rangle + \alpha^2)$ with respect to α and setting it to 0 gives us $\alpha^* = \langle u, x\rangle$. Hence, the optimal solution is

$$\hat{x} = \langle u, x\rangle u.$$

The **residual** is $r_x = x - \hat{x}$. Note that $\langle u, r_x\rangle = 0$ (this gives the orthogonality condition in the name "orthogonal projection"):

$$\begin{aligned} \langle u, r_x\rangle &= \langle u, x - \hat{x}\rangle \\ &= \langle u, x\rangle - \langle u, \hat{x}\rangle \\ &= \langle u, x\rangle - \langle u, x\rangle \\ &= 0. \end{aligned}$$

By Pythagoras, $\|x\|^2 = \|\hat{x}\|^2 + \|r_x\|^2$. We can also write:

$$\hat{x} = (uu^T)x = Px,$$

$$r_x = (I - uu^T)x = (I - P)x.$$

Remark 1.40. How else can we interpret orthogonal projection? Note that

$$\hat{x} = uu^T x = u(u^T x) = u(u^T(\hat{x} + r_x)) = u(u^T(\hat{x})).$$

These are the coordinates of x with respect to the subspace $\mathcal{U}$ and the orthornomal basis we have chosen. r_x disappears—why? There are two steps involved here:

1. Projection: We project x onto the subspace span(u).
2. Synthesis: We synthesize $\hat{x}$ via $u^T x$, coordinates of the subspace.

1.4.2 Projection of x onto a subspace $\mathcal{U}$

Now, let's move onto a case when the subspace has dimension greater than 1. Our optimization problem is, again: for a given $x \in \mathbb{R}^n$, find a point z in $\mathcal{U}$ that minimizes the distance to x:

$$\begin{aligned} \min_{z \in \mathbb{R}^n} \quad & \frac{1}{2}\|x - z\|^2 \\ \text{s.t.} \quad & z \in \mathcal{U}. \end{aligned}$$

Assume that $\mathcal{U}$ has an orthonormal basis $\{u_1, u_2, \ldots, u_k\}$ so we can uniquely write

$$z = \sum_{j=1}^{k} \alpha_j u_j.$$

We can, again, manipulate $\frac{1}{2}\|x - z\|^2$:

$$\begin{aligned} \frac{1}{2}\|x - z\|^2 &= \frac{1}{2}\langle x - z, x - z\rangle \\ &= \frac{1}{2}\|x\|^2 - \langle z, x\rangle + \frac{1}{2}\|z\|^2 \\ &= \frac{1}{2}\|x\|^2 - \sum_{j=1}^{k} \alpha_j \langle u_j, x\rangle + \frac{1}{2}\sum_{j=1}^{k} \alpha_j^2, \end{aligned}$$

using Pythagoras to write $\|z\|^2 = \sum_{j=1}^{k} \alpha_j^2$. Taking the derivative with respect to α_j and setting it to 0 gives us $\alpha_j = \langle u_j, x\rangle,\ j = 1, \ldots, k$. So the optimal point is

$$\hat{x} = \sum_{j=1}^{k} \langle u_j, x\rangle u_j.$$

Again, we have that $r_x = x - \hat{x}$ and that $r_x \perp \mathcal{U}$. Why? If r_x is orthogonal to all of the basis vectors, then r_x is orthogonal to all linear combinations of the basis vectors. Thus, we simply need to show that $r_x \perp u_j,\ j = 1, \ldots, k$:

$$\langle r_x, u_j\rangle = \langle x - \hat{x}, u_j\rangle = \langle u_j, x\rangle - \langle u_j, \hat{x}\rangle = \langle u_j, x\rangle - \langle u_j, x\rangle = 0.$$

Thus, $\hat{x}$ is the unique orthogonal projection of x onto $\mathcal{U}$, and $\|x\|^2 = \|\hat{x}\|^2 + \|r_x\|^2$ by Pythagoras.

In matrix form, the optimal projection is

$$\hat{x} = \sum_{j=1}^{k} u_j u_j^T x = \left(\sum_{j=1}^{k} u_j u_j^T \right) x = Px,$$

where $P = \sum_{j=1}^{k} u_j u_j^T$. Let $U \in \mathbb{R}^{n \times k}$ be the matrix with $u_1, \ldots, u_k$ as columns. We therefore have

$$P = \sum_{j=1}^{k} u_j u_j^T = UU^T,$$

where P is the sum of k rank-1 matrices. Note that P is symmetric ($(UU^T)^T = UU^T$). We claim that $P^2 = P$; intuitively, once we project x onto $\mathcal{U}$, doing it again will give us the same $\hat{x}$.

$$P^2 = (uu^T)(uu^T) = u(u^T u)u^T = uu^T = P.$$

Definition 1.41. (Projection matrix). A matrix P that is **symmetric** ($P^T = P$) and **idempotent** ($P^2 = P$) is called a projection matrix.

Remark 1.42. Note that we can write

$$UU^T = \begin{pmatrix} u_1 & u_2 & \ldots & u_k \end{pmatrix} \begin{pmatrix} u_1^T \\ \vdots \\ u_k^T \end{pmatrix} = \sum_{j=1}^{k} u_j u_j^T.$$

The columns of $U \in \mathbb{R}^{n \times k}$ form an orthornomal basis for the subspace $\mathcal{U}$, so $U^T U = I_k$.

We can therefore write

$$\hat{x} = UU^T x,$$
$$r_x = (I - UU^T)x.$$

1.4.3 Orthogonal complement of a subspace

The **orthogonal complement** of a subspace $\mathcal{U}$ is

$$\mathcal{U}^\perp = \{z \in \mathbb{R}^n : z \perp \mathcal{U}\}.$$

Lemma 1.43. *$\mathcal{U}^\perp$ is a subspace of $\mathbb{R}^n$.*

Proof. 1. 0, the zero vector, is in $\mathcal{U}^\perp$ as it is orthogonal to everything.

2. $\forall\ u, v \in \mathcal{U}^\perp$, $\forall\ x \in \mathcal{U}$, $\langle u, x \rangle = 0$, $\langle v, x \rangle = 0 \Longrightarrow \langle u + v, x \rangle = 0$. Thus, $u + v \in \mathcal{U}^\perp$.

3. $\forall\ u \in \mathcal{U}^\perp$, $\forall\ x \in \mathcal{U}$, $\langle cu, x \rangle = 0$. Thus, $cu \in \mathcal{U}^\perp$.

□

Lemma 1.44. *$\mathcal{U}^\perp \cap \mathcal{U} = \{0\}$, the zero subspace.*

Proof. If a vector u were in both $\mathcal{U}$ and $\mathcal{U}^\perp$, it would have to be perpendicular to itself ($\langle u, u \rangle = 0$), which is only true for $u = 0$. □

Lemma 1.45. $\mathbb{R}^n = \mathcal{U} + \mathcal{U}^\perp$ *uniquely. (Every vector* $x \in \mathbb{R}^n$ *can be written uniquely in the form* $x = u + v$ *with* $u \in \mathcal{U}$ *and* $v \in \mathcal{U}^\perp$*.)*

Proof. Suppose that $x \neq 0$. $x = \hat{x} + r_x$, and by definition, $r_x \in \mathcal{U}^\perp$. Suppose that there are two ways of expressing x: $x = u_1 + v_1 = u_2 + v_2$, where $u_1, u_2 \in \mathcal{U}$ and $v_1, v_2 \in \mathcal{U}^\perp$. We subtract to get $0 = (u_1 - u_2) + (v_1 - v_2)$, or $(u_1 - u_2) = -(v_1 - v_2)$. $u_1 - u_2 \in \mathcal{U}$, and $-(v_1 - v_2) \in \mathcal{U}^\perp$; thus, by (2), $u_1 = u_2$ and $v_1 = v_2$. □

Lemma 1.46. *We also have the following and leave the proofs as an exercise.*

- $(\mathbb{R}^n)^\perp = \{0\}$.
- $\{0\}^\perp = \mathbb{R}^n$.
- $(\mathcal{U}^\perp)^\perp = \mathcal{U}$.
- $(\mathcal{U} + \mathcal{V})^\perp = \mathcal{U}^\perp \cap \mathcal{V}^\perp$.
- $(\mathcal{U} \cap \mathcal{V})^\perp = \mathcal{U}^\perp + \mathcal{V}^\perp$.
- *If* $dim(\mathcal{U}) = k$*, then* $dim(\mathcal{U}^\perp) = n - k$.
- $\mathcal{U} \subseteq \mathcal{V} \Longrightarrow \mathcal{V}^\perp \subseteq \mathcal{U}^\perp$. *Proof: Take* $v \in \mathcal{V}^\perp$. $v \perp \mathcal{V}$*, and since* $\mathcal{U} \subseteq \mathcal{V}$, $v \perp u \ \forall \ u \in \mathcal{U}$. *Thus,* $v \in \mathcal{U}^\perp$.

Remark 1.47. To prove equality of two subspaces $\mathcal{U}$ and $\mathcal{V}$, show that $\mathcal{U}$ is contained in $\mathcal{V}$ and that $\mathcal{V}$ is contained in $\mathcal{U}$.

2 Singular value decomposition and principal component analysis

2.1 Singular value decomposition (SVD)

Let us start with some preliminaries.

Definition 2.1. (Range and nullspace). Let $A \in \mathbb{R}^{m\times n}$. The range of A is defined by

$$\mathcal{R}(A) = \{y \in \mathbb{R}^m : y = Ax \text{ for some } x \in \mathbb{R}^n\} \subseteq \mathbb{R}^m.$$

The nullspace of A is defined by

$$\mathcal{N}(A) = \{x \in \mathbb{R}^n : Ax = 0\}.$$

Theorem 2.2. *Let $A \in \mathbb{R}^{m\times n}$.*

$$\mathcal{N}(A)^\perp = \mathcal{R}(A^T).$$

The orthogonal complement of the null space of A is a subspace of $\mathbb{R}^n$. A^T maps from $\mathbb{R}^m$ to $\mathbb{R}^n$. We will use this theorem again and again.

Proof. Let $x \in \mathcal{N}(A)$, so $Ax = 0$. Thus, $x^TA^T = 0^T = 0$. $\forall\ y \in \mathbb{R}^m$, $x^TA^Ty = 0$. So $x \in \mathcal{R}(A^T)^\perp$. This shows that $\mathcal{N}(A) \subseteq \mathcal{R}(A^T)^\perp$. Note that for all subspaces, $(\mathcal{U}^\perp)^\perp = \mathcal{U}$ and $\mathcal{U} \subseteq \mathcal{V} \Longrightarrow \mathcal{V}^\perp \subseteq \mathcal{U}^\perp$. Thus, $\mathcal{R}(A^T) \subseteq \mathcal{N}(A)^\perp$.

Now let $x \in \mathcal{R}(A^T)^\perp$. $\forall\ y \in \mathbb{R}^m$, $x^TA^Ty = 0 \Longrightarrow y^TAx = 0$. This imples that $Ax = 0$ and, thus, $x \in \mathcal{N}(A)$. Thus, $\mathcal{R}(A^T)^\perp \subseteq \mathcal{N}(A) \Longrightarrow \mathcal{N}(A)^\perp \subseteq \mathcal{R}(A^T)$. □

Definition 2.3. (Rank). Let $A \in \mathbb{R}^{m\times n}$.

$$\begin{aligned}\text{rank}(A) &= \dim \mathcal{R}(A)\\ &\equiv \#\text{ of linearly independent columns in } A\\ &\equiv \#\text{ of linearly independent rows in } A\\ &\equiv \dim \mathcal{R}(A^T)\\ &\leq \min(m,n).\end{aligned}$$

A is said to be **full rank** if $r = \min(m,n)$.

Definition 2.4. (Rank one matrix). An $m \times n$ rank one matrix has the form yx^T, where $y \in \mathbb{R}^m$ and $x \in \mathbb{R}^n$ are both nonzero.

Remark 2.5. Let $A = yx^T,\ y \in \mathbb{R}^m,\ x \in \mathbb{R}^n,\ y \neq 0,\ x \neq 0,\ A \in \mathbb{R}^{m\times n}$. Note that for any $w \in \mathbb{R}^n$, we can write $(yx^T)w = y(x^Tw)$; thus, $\mathcal{R}(yx^T) = \text{span}(y)$.

Definition 2.6. (Eigenvalues and eigenvectors). Let $A \in \mathbb{C}^{n\times n}$. The eigenvectors of A are the nonzero vectors s such that, for some $\lambda \in$, called the eigenvalue corresponding to x, $Ax = \lambda x$. The set of eigenvalues of A is called the **spectrum** of A, denoted by $\sigma(A)$. The eigenvector for a given eigenvalue λ is not unique, since if x is an eigenvector, so is αx for all nonzero $\alpha \in \mathbb{C}$.

Example 2.7. An example: $A = I$. A has n eigenvalues of value 1. Any vector is an eigenvector since any vector satisfies $Ax = \lambda x$.

Lemma 2.8. *A matrix $A \in \mathbb{C}^{n\times n}$ has n eigenvalues (think about the characteristic polynomial, which has n roots). The eigenvectors corresponding to distinct eigenvalues are linearly independent. Hence, A has at least as many linearly independent eigenvectors as the number of distinct eigenvalues.*

Lemma 2.9. *A symmetric matrix $S = S^T,\ S \in \mathbb{R}^{n\times n}$, has n real eigenvalues and n real orthonormal eigenvectors.*

Remark 2.10. (Eigendecomposition). Let $S \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ and corresponding eigenvectors $x_1, \ldots, x_n$. If we place the n orthonormal eigenvectors of S in the columns of a matrix V (note that V is orthogonal, since $V^TV = I_n$, and V is full rank and thus invertible, so $V^{-1} = V^T \implies VV^T = I_n$) and place the corresponding eigenvalues on the diagonal of a diagonal matrix Λ, then

$$S\begin{pmatrix} x_1 & x_2 & \ldots & x_n \end{pmatrix} = \begin{pmatrix} \lambda_1 x_1 & \lambda_2 x_2 & \ldots & \lambda_n x_n \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & \ldots & x_n \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix},$$

$$\implies SV = V\Lambda,$$
$$\implies S = V\Lambda V^T.$$

Definition 2.11. (Positive semidefinite and positive definite matrices). A positive semidefinite (PSD) matrix satisfies $x^TPx \geq 0\ \forall\ x \in \mathbb{R}^n$. A positive definite (PD) matrix satisfies $x^TPx > 0,\ \forall\ x \in \mathbb{R}^n,\ x \neq 0$.

Remark 2.12. WLOG, we will always assume that a PSD (or PD) square matrix $P \in \mathbb{R}^{n\times n}$ is symmetric, since

$$\frac{1}{2}x^T(P + P^T)x = \frac{1}{2}\left(\langle x, Px\rangle + \langle Px, x\rangle\right) = x^TPx,$$

and so we can always replace P with $\frac{1}{2}(P + P^T)$, which is symmetric.

Lemma 2.13. *If $P \in \mathbb{R}^{n\times n}$ is symmetric and positive semidefinite (resp. positive definite), then all eigenvalues of P are real and nonnegative (resp. positive), and the eigenvectors of P can be selected to be real and orthonormal.*

Remark 2.14. A symmetric positive semidefinite matrix does not have to have n distinct eigenvalues. (Consider the identity matrix.)

Definition 2.15. (Differentiable function). A function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable if at $x \in \mathbb{R}^n$, there exists a unique linear function $Df(x) : \mathbb{R}^n \to \mathbb{R}$ such that

$$\lim_{h\to 0} \frac{|f(x + h) - f(x) - Df(x)h|}{\|h\|_2} = 0,$$

where $h \in \mathbb{R}^n$, $Df(x)$ is a linear map ($\mathbb{R}^n \to \mathbb{R}$) matrix that depends upon x and is a $1 \times n$ matrix, and $Df(x)h \in \mathbb{R}$. Roughly, $Df(x)$ is the best linear approximation to f at x:

$$f(x+h) \approx f(x) + Df(x)h.$$

We can write $x = (x_1, x_2, \ldots, x_n)$ and $h = (h_1, \ldots, h_n)^T$.

$$Df(x) = \begin{pmatrix} \frac{\partial}{\partial x_1}(x) & \frac{\partial}{\partial x_2}(x) & \ldots & \frac{\partial}{\partial x_n}(x) \end{pmatrix},$$

so

$$Df(x)h = \begin{pmatrix} \frac{\partial}{\partial x_1}(x) & \frac{\partial}{\partial x_2}(x) & \ldots & \frac{\partial}{\partial x_n}(x) \end{pmatrix} \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix} = \sum_{j=1}^{n} \frac{\partial f}{\partial x_j}(x) h_j.$$

The **gradient** of f at x is the vector in $\mathbb{R}^n$ given by

$$\nabla f(x) = \begin{pmatrix} \frac{\partial}{\partial x_1}(x) & \frac{\partial}{\partial x_2}(x) & \ldots & \frac{\partial}{\partial x_n}(x) \end{pmatrix}^T = Df(x)^T.$$

The gradient points in the direction of steepest ascent of f at x. $\|\nabla f(x)\|_2$ is the rate of increase of f in this direction.

Note that the gradient is just a vector in the space where x and h live. The derivative is a function that takes where the dummy h lives and maps it to $\mathbb{R}$.

Example 2.16. Fix $a \in \mathbb{R}^n,\ b \in \mathbb{R}$. Let $f(x) = a^T x + b$. What is the derivative of this function with respect to x?

$$Df(x)h = \begin{pmatrix} a_1 & a_2 & \ldots & a_n \end{pmatrix} h = a^T h.$$

$$\nabla f(x) = a.$$

This is the best linear approximation for the function because the best linear approximation for a linear function $f(x) = a^T x$ is itself.

Example 2.17. Let $A \in \mathbb{R}^{n \times n},\ x \in \mathbb{R}^n$. Let $f(x) = x^T A x$. What is the derivative of this function with respect to x?

$$Df(x)h = x^T A h + h^T A x = x^T (A^T + A) h.$$

$$\nabla f(x) = (A^T + A) x.$$

Definition 2.18. (Induced matrix 2-norm of a matrix). Let $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n,\ x \neq 0$. The gain of A acting on x is $\frac{\|Ax\|_2}{\|x\|_2}$. The maximum gain of A over all $x \in \mathbb{R}^n$ is

$$\|A\|_2 = \max_{\|x\|_2 \neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \max_{x \neq 0} \left\| A \frac{x}{\|x\|_2} \right\| = \max_{\|u\|_2 = 1} \|Au\|_2.$$

$\|A\|_2$ is called the induced matrix 2-norm of A, as it is induced by the Euclidean norms on $\mathbb{R}^m$ and $\mathbb{R}^n$. This is a well-known optimization problem:

$$\begin{aligned} \max_{x \in \mathbb{R}^n} \quad & x^T (A^T A) x \\ \text{s.t.} \quad & x^T x = 1. \end{aligned}$$

Theorem 2.19. *Let $P \in \mathbb{R}^{n\times n}$ be a symmetric positive semidefinite matriix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. The problem*

$$\begin{aligned} \max_{u\in\mathbb{R}^n} \quad & u^T P u \\ s.t. \quad & u^T u = 1 \end{aligned}$$

has the optimal value λ_1, which is achieved if and only if u is a unit norm eigenvector of P for λ_1. If $\lambda_1 > \lambda_2$, then the solution is unique.

Proof. Start by forming the Lagrangian

$$\begin{aligned} L &= f(x) + \mu g(x) \\ &= x^T P x + \mu(1 - x^T x). \\ DL(h) &= h^T P x + x^T P h - \mu(h^T x + x^T h) \\ &= 2h^T P x - 2\mu h^T x \\ &= 2h^T (Px - \mu x). \end{aligned}$$

Thus, a necessary condition is $Px = \mu x$. Note that we could have written the previous set of equations slightly differently, still ending with the same result:

$$\begin{aligned} DL(h) &= h^T P x + x^T P h - \mu(h^T x + x^T h) \\ &= (2x^T P - 2\mu x^T)h. \\ \nabla L &= 2(Px - \mu x) = 0. \\ \implies Px &= \mu x. \end{aligned}$$

Hence, μ must be an eigenvalue of P such that its corresponding eigenvector x satisfies $x^T x = 1$. We are maximizing $x^T P x$, and

$$Px = \mu x \iff x^T P x = \mu x^T x = \mu,$$

so we therefore choose the eigenvector corresponding to the largest eigenvalue. □

Definition 2.20. (Induced matrix 2-norm cont., or spectral norm).

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)},$$

using the preceding theorem. Note that this is a norm (we leave this as an exercise).

Lemma 2.21. *Let $A, B \in \mathbb{R}^{m\times n},\ x \in \mathbb{R}^n$. Then,*

- $\|Ax\|_2 \leq \|A\|_2 \|x\|_2$.
- $\|AB\|_2 \leq \|A\|_2 \|B\|_2$.

Remark 2.22. The induced matrix 2-norm and the matrix Euclidean norm are distinct norms on $\mathbb{R}^{m\times n}$—the Euclidean norm on $\mathbb{R}^{m\times n}$ is called the Frobenius norm and is denoted by $\|A\|_F$.

Theorem 2.23. *(Compact SVD). For any $A \in \mathbb{R}^{m\times n}$ with rank$(A) = r \leq \min(m,n)$, $\exists\, U \in \mathbb{R}^{m\times r}$ with $U^TU = I_r$, $V \in \mathbb{R}^{n\times r}$ with $V^TV = I_r$, and diagonal $\Sigma \in \mathbb{R}^{r\times r}$ with diagonal entries $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ such that*

$$A = U\Sigma V^T = \sum_{j=1}^{r} \sigma_j u_j v_j^T.$$

*$\sigma_j,\ j = 1,\ldots,r$ are the **singular values** of A. The r orthonormal columns of U are called the **left, or output, singular vectors** of A, and the r orthonormal columns of V are called the **right, or input, singular vectors** of A. Though $U^TU = I_r$ and $V^TV = I_r$, in general, U and V are not orthogonal matrices, i.e., $UU^T \neq I_m$ and $VV^T \neq I_n$.*

Remark 2.24. Note that while Σ is unique, the SVD formed is not unique (we can choose U and V in different ways).

Corollary 2.25. *The matrices U and V in the compact SVD have the following additional properties:*

- *The columns of U form an orthonormal basis for $\mathcal{R}(A) \subseteq \mathbb{R}^m$.*
- *The columns of V form an orthonormal basis for $\mathcal{N}(A)^\perp \subseteq \mathbb{R}^n$.*

Proof. $Ax = U\Sigma V^Tx$, so $Ax \in \mathcal{R}(U)$ for any $x \in \mathbb{R}^n$. Thus, $\mathcal{R}(A) \subseteq \mathcal{R}(U)$. Now, let $u \in \mathcal{R}(U)$. For some $z \in \mathbb{R}^r$, $u = Uz = U\Sigma V^TV(\Sigma^{-1}z) = A(V\Sigma^{-1}z)$. Thus, $\mathcal{R}(U) \subseteq \mathcal{R}(A)$.

Start with $A^T = V\Sigma U^T$. By the first part, the columns of V form an orthonormal basis for the range of A^T: $\mathcal{R}(V) = \mathcal{R}(A^T)$. Since $\mathcal{N}(A)^\perp = \mathcal{R}(A^T)$, we have our result. □

Remark 2.26. Note that

$$Av_1 = U\Sigma V^Tv_1 = U\Sigma \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{pmatrix} v_1 = U\Sigma e_1 = U \begin{pmatrix} \sigma_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \sigma_1 u_1.$$

More generally, $Av_k = \sigma_k u_k,\ k = 1,\ldots,r$.

Remark 2.27. (Operational interpretation of SVD). We can interpret the SVD as going through the following three steps, where we have a given vector $x \in \mathbb{R}^n$:

- *Analysis step*: V^Tx gives the coordinates with respect to V of the orthogonal projection of x onto the subspace $\mathcal{N}(A)^\perp$, since the columns of V form an orthonormal basis for $\mathcal{N}(A)^\perp$ and since the orthogonal projection is $\hat{x} = VV^Tx$.
- *Scaling step*: $\Sigma(V^Tx)$ individually scales the r coordinates using the r diagonal entries of Σ.
- *Synthesis step*: $U(\Sigma V^Tx)$ synthesizes the output vector by using the r scaled coordinates and the orthonormal basis U for $\mathcal{R}(A)$.

Thus, the r orthonormal basis vectors for $\mathcal{N}(A)^\perp$ are mapped to scaled versions of corresponding orthonormal basis vectors for $\mathcal{R}(A)$, as illustrated in the lecture notes. The map $A : \mathcal{N}(A)^\perp \to \mathcal{R}(A)$ is one-to-one and onto, and thus, bijective and invertible.

Remark 2.28. (Induced matrix 2-norm). We return to the induced matrix 2-norm, $\|A\|_2$. Note that

$$\|A\|_2^2 = \max_{\|x\|_2=1} \|U\Sigma V^T x\|_2^2 = \max_{\|x\|_2=1} x^T V\Sigma U^T U\Sigma V^T x = \max_{\|x\|_2=1} x^T (V\Sigma^2 V^T) x = \lambda_{\max}(V\Sigma^T V^T).$$

We turn to Σ for the eigenvalues, which are sorted; thus, we take $x = v_1$ with corresponding eigenvalue σ_1^2. We can also think of this as taking the input direction with the most gain—v_1—which appears in the output in the direction u_1 with gain of σ_1, since $Av_1 = \sigma_1 u_1$. This implies that *the induced matrix* 2*-norm of* A *equals the maximum singular value of* A:

$$\|A\|_2 = \sigma_1.$$

Remark 2.29. (Frobenius norm). We can also say something about $\|A\|_F$, the Frobenius norm of A. First, we can write

$$A = U\Sigma V^T = \sum_{j=1}^{r} \sigma_j u_j v_j^T,$$

which is a linear combination of r rank one matrices that form an orthonormal set in $\mathbb{R}^{m\times n}$, since the outer product of $u_j v_j^T$ and $u_k v_k^T$ is

$$\langle u_j v_j^T, u_k v_k^T \rangle = \mathrm{Tr}(v_j u_j^T u_k v_k^T).$$

We can use the cyclic property of the trace ($\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$) to write

$$\mathrm{Tr}(v_j u_j^T u_k v_k^T) = \mathrm{Tr}(u_j^T u_k v_k^T v_j) = \begin{cases} 1 \text{ if } j = k \\ 0 \text{ if } j \neq k. \end{cases}$$

These rank 1 matrices $\{u_j v_j^T\}_{j=1}^r$ are thus an **orthonormal set** of $m \times n$ matrices.

$$\begin{aligned}
\|A\|_F &= \left\| \sum_{j=1}^r \sigma_j u_j v_j^T \right\|_F \\
&= \left(\sum_{j=1}^r \left\| \sigma_j u_j v_j^T \right\|_F^2 \right)^{1/2} \\
&= \left(\sum_{j=1}^r \left\| \sigma_j \begin{pmatrix} u_{j1} v_j^T \\ \vdots \\ u_{jm} v_j^T \end{pmatrix} \right\|_F^2 \right)^{1/2} \\
&= \left(\sum_{j=1}^r \sigma_j^2 \sum_{i=1}^m \|u_{ji} \cdot v_j^T\|_2^2 \right)^{1/2} \\
&= \left(\sum_{j=1}^r \sigma_j^2 \|u_j\|_2^2 \right)^{1/2} \\
&= \left(\sum_{j=1}^r \sigma_j^2 \right)^{1/2},
\end{aligned}$$

where the second equation holds by Pythagoras's theorem. Thus, *the Frobenius norm of A is the Euclidean norm of the vector of its singular values.*

Definition 2.30. (Nuclear, or trace, norm). $\|A\|_* = \sum_{j=1}^r \sigma_i$.

Claim 2.31. *Suppose that we have a matrix $A \in \mathbb{R}^{m \times n}$. Think of $Q \in \mathcal{O}_m$ and $R \in \mathcal{O}_n$ as rotations. Now, consider the matrix QAR. We claim that QAR has the same singular values as A has—that $\Sigma_A = \Sigma_{QAR}$. This is a very useful property.*

Proof. We can write $A = U_A \Sigma_A V_A^T$. Thus, $QAR = (QU_A)\Sigma_A(V_A^T R)$, which satisfies the SVD. *** □

Claim 2.32. $\|A\|_F = \|QAR\|_F$.

Proof. Method 1: The singular values of QAR are the same as the singular values of A, so the Frobenius norms are the same.

Method 2: $\|A\|_F^2 = \langle A, A \rangle$.

$$\|QAR\|_F^2 = \langle QAR, QAR \rangle = \mathrm{Tr}(R^T A^T Q^T QAR) = \mathrm{Tr}(R^T A^T AR) = \mathrm{Tr}(A^T A) = \langle A, A \rangle = \|A\|_F^2.$$

□

Claim 2.33. *We have two matrices $A, B \in \mathbb{R}^{m \times n}$ with compact SVDs $U_A \Sigma_A V_A^T$ and $U_B \Sigma_B V_B^T$. We want to measure how different the two matrices are. We could measure $\|A - B\|_F$. Or, we could rotate the matrix B to try to "line up" A and B, and solve*

$$\min_{Q \in \mathcal{O}_m} \|A - QB\|_F = \min_{Q \in \mathcal{O}_m, R \in \mathcal{O}_n} \|A - QBR\|_F.$$

We claim that

$$\|A - QBR\|_F \geq \|\Sigma_A - \Sigma_B\|_F,$$

with equality when Q and R are selected to make $QU_B = U_A$ and $R^T V_B = V_A$.

In one of the exercises, we are asked to solve the first minimization problem (easy). The second minimization problem is hard.

Proof. We sketch out the proof that

$$\min_{Q,R} \|A - QBR\|_F = \|\Sigma_A - \Sigma_B\|_F.$$

We can start by plugging in some SVDs:

$$\|U_A \Sigma_A V_A^T - Q U_B \Sigma_B V_B^T R\|_F.$$

We can use the full SVDs of A and B (an upgrade from the compact SVD—see the following definition) to simplify this term.

$$\|U_A(\Sigma_A - U_A^T Q U_B \Sigma_B V_B^T R V_A) V_A^T\|_F = \|\Sigma_A - (U_A^T Q U_B)\Sigma_B (V_B^T R V_A)\|_F.$$

We have two orthogonal matrices—$S := (U_A^T Q U_B)$ and $Z := (V_B^T R V_A)$, by a previous claim.

$$\min_{S,Z} \|\Sigma_A - S\Sigma_B Z\|_F \leq \|\Sigma_A - \Sigma_B\|_F.$$

This inequality holds since we can take $S, Z = I$. The hard part is proving that

$$\min_{S,Z} \|\Sigma_A - S\Sigma_B Z\|_F \geq \|\Sigma_A - \Sigma_B\|_F.$$

How do we make $S = I$ and $Z = I$? We want both $U_A^T Q U_B$ and $V_B^T R V_A$ to disappear, which we can do by making the left singular vectors (U) *** and the right singular vectors (V) ***.

We get a lower bound by solving $\min_{Q \in \mathcal{O}_m} \|A - QB\|_F$ first, which will give us some properties that we can apply to this situation. We leave the rest of the proof as an exercise. □

Remark 2.34. Why do we even want to do this? Let's say we have data living in a very large-dimensional space, but we hypothesize that the data actually lives in some small-dimensional subspace. How do we find U, an orthornomal basis of the subspace $\mathcal{U}$? We use the above derivations.

Definition 2.35. (Full SVD). Let $A = U_c \Sigma_c V_c^T$ be a compact SVD with $U_c \in \mathbb{R}^{m \times r}$, $\Sigma_c = \mathbb{R}^{r \times r}$, $V_c \in \mathbb{R}^{n \times r}$. Instead of creating matrices that are tall or wide, we want to create square matrices $\mathbb{R}^{m \times m}$ and $\mathbb{R}^{n \times n}$. Thus, to U_c, we add an orthonormal basis for $\mathcal{R}(U_c)^\perp$ to form the orthogonal matrix

$$U = \begin{pmatrix} U_c & U_c^\perp \end{pmatrix} \in \mathbb{R}^{m \times m}.$$

Similarly, to V_c, we add an orthonormal basis for $\mathcal{R}(V_c)^\perp$ to form the orthogonal matrix

$$V = \begin{pmatrix} V_c & V_c^\perp \end{pmatrix} \in \mathbb{R}^{n \times n}.$$

To ensure that these extra columns do not interfere with A's factorization, we form $\Sigma \in \mathbb{R}^{m\times n}$ by padding Σ_c with zero entries:

$$\Sigma = \begin{pmatrix} \Sigma_c & 0_{r\times(n-r)} \\ 0_{(m-r)\times r} & 0_{(m-r)\times(n-r)} \end{pmatrix}.$$

Now, we have a full SVD factorization: $A = U\Sigma V^T$. The utility of the full SVD derives from U and V being orthogonal, and hence intervible, matrices.

Remark 2.36. If P is a symmetric positive semidefinite matrix, then a full SVD of P is simply an eigendecomposition of P: $U\Sigma V^T = Q\Sigma Q^T$, where Q is the orthogonal matrix with eigenvectors of P as columns. Thus, the SVD extends the eigendecomposition by using different orthonormal sets of vectors in the input and output spaces.

2.2 Principal component analysis (PCA)

We cover our first dimensionality reduction technique. We ask: given data $\{x_j \in \mathbb{R}^n\}_{j=1}^p$, is there some subspace $\mathcal{U}$, with dimension $q < n$, such that if we orthogonally project each data point down to this subspace, then the errors we incur will be quite small? In other words, we want to find the best subspace $\mathcal{U}$ with dimension q with the minimum sum of squared norms of the error residuals over all possible subspaces of the vector space. We will use SVD to do computations for PCA.

We form an orthornomal basis $\{u_1, \ldots, u_q\}$ for $\mathcal{U}$. $U = \begin{pmatrix} u_1 & u_2 & \ldots & u_q \end{pmatrix} \in \mathbb{R}^{n\times q}$ with $U^TU = I_q$. We have two cases:

1. $q = n$: $U \in \mathcal{O}_n$.
2. $q < n$: $U \in \mathcal{O}_{n\times q}$ (new notation).

The projection of x_j, the j-th (centered) data point, onto $\mathcal{U}$ is given by $\hat{x}_j = UU^Tx_j \in \mathbb{R}^n$; while this does not reduce the dimension of the data, it maps data points onto a q-dimensional subspace in $\mathbb{R}^n$. The error residual of this projection is given by $x_j = \hat{x}_j + r_j$. We aim to replace x_j with $\hat{x}_j$ and suffer the loss of information via r_j. Note that instead of working with $\hat{x}_j$, we can equivalently work with its coordinates $c_j \in \mathbb{R}^q$ with respect to the basis U, since $\hat{x}_j = Uc_j$. There is no loss of information when we work with c_j instead of $\hat{x}_j$ since $c_j = U^T\hat{x}_j = U^Tx_j$. So now we have $\hat{x}_j = Uc_j, c_j \in \mathbb{R}^q$. To summarize:

$$x_j \in \mathbb{R}^n \xrightarrow{UU^Tx_j} \hat{x}_j \in \mathbb{R}^n \xrightarrow{U^T\hat{x}_j} c_j \in \mathbb{R}^q.$$

$$x_j \xleftarrow{r_j+\hat{x}_j} \hat{x}_j \xleftarrow{Uc_j} c_j.$$

This is referred to as **linear dimensionality reduction**; other methods exist for selecting the subspace $\mathcal{U}$ in different ways.

We now discuss how to go about choosing q and its corresponding subspace. Our first step is to **center the data**. We subtract the **sample mean** $\mu = \frac{1}{p}\sum_{i=1}^p x_j$ from each x_j to form $z_j = x_j - \mu$. In matrix form, we have, prior to centering,

$$X = \begin{pmatrix} x_1 & x_2 & \ldots & x_p \end{pmatrix} \in \mathbb{R}^{n\times p}.$$

The sample mean is

$$\mu = \frac{1}{p}(X\mathbf{1}_p),$$

where $\mathbf{1}_p \in \mathbb{R}^p$ is a vector of all 1's. The new matrix of centered data points is

$$Z = \begin{pmatrix} z_1 & z_2 & \dots & z_p \end{pmatrix} = X - \mu\mathbf{1}_p^T = X\left(I_p - \frac{1}{p}\mathbf{1}_p\mathbf{1}_p^T\right) = X(I_p - uu^T).$$

Note that uu^T is a projection matrix! From this point forward, we assume that the data has been centered.

Now, we need to **parametrize** $\mathcal{U}$; in other words, we now fix q. Pick an orthornomal basis $U = \begin{pmatrix} u_1 & u_2 & \dots & u_q \end{pmatrix} \in \mathcal{O}_{n\times q}$. Note that this orthonormal basis representation is not unique, since there are infinitely many orthonormal bases for $\mathcal{U}$. Take $U_1, U_2 \in \mathcal{O}_{n\times q}$, which both span $\mathcal{U}$. We can write $U_1 = U_2Q$ since they span the same subspace, where $Q \in \mathbb{R}^{q\times q}$. Thus, $U_2^TU_1 = Q$.

$$Q^TQ = U_1^T(U_2U_2^T)U_1 = I_q,$$
$$QQ^T = U_2^T(U_1U_1^T)U_2 = I_q,$$

since $U_2U_2^T$ and $U_1U_1^T$ are projection matrices, so $U_1U_1^T = U_2U_2^T$. So $Q \in \mathcal{O}_q$. So any two orthonormal basis representations U_1, U_2 of $\mathcal{U}$ are related via a $q \times q$ orthogonal matrix Q: $U_1 = U_2Q$ and $U_2 = U_1Q^T$.

2.2.1 An optimal projection subspace

We fix $q : U \in \mathcal{O}_{n\times q}$, and our optimization problem is

$$\min_{U\in\mathcal{O}_{n\times q}} \|X - UU^TX\|_F^2.$$

The solution is not unique since if U is a solution, then so is UQ for every $Q \in \mathcal{O}_q$, which corresponds to a different parametrization of the same subspace. We can rewrite the objective function:

$$\begin{aligned}\|(I - UU^T)X\|_F^2 &= \mathrm{Tr}(X^T(I - UU^T)(I - UU^T)X) \\ &= \mathrm{Tr}(X^TX) - \mathrm{Tr}(U^TXX^TU).\end{aligned}$$

Let $P \in \mathbb{R}^{n\times n}$ be the symmetric positive semidefinite matrix XX^T, so we can equivalently solve

$$\max_{U\in\mathcal{O}_{n\times q}} \mathrm{Tr}(U^TPU).$$

Note that we have seen this problem for one-dimensional U (the induced 2-norm optimization problem).

Theorem 2.37. *Let the symmetric positive semidefinite matrix $P \in \mathbb{R}^{n\times n}$ have eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Then, the optimal value of*

$$\max_{U\in\mathcal{O}_{n\times q}} Tr(U^TPU)$$

is $\sum_{j=1}^p \lambda_j$. Moreover, this is achieved if the columns of U are q orthonormal eigenvectors for the largest q eigenvalues of P.

Remark 2.38. It is a standard result that if we are free to select the left and right singular vectors of a matrix B, then the inner product $\langle A, B\rangle$ is maximized when the left and right singular vectors of B are chosen to equal the left and right singular vectors of A, respectively.

Proof. We can rewrite P as its eigendecomposition $P = V\Sigma V^T$, where $V \in \mathcal{O}_n$ and Σ is diagonal with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$ in decreasing order down the diagonal.

$$\begin{aligned} U^T P U &= U^T V \Sigma V^T U. \\ \operatorname{Tr}(U^T P U) &= \operatorname{Tr}(U^T V \Sigma V^T U) \\ &= \operatorname{Tr}(\Sigma V^T U U^T V) \\ &= \langle \Sigma, W W^T \rangle, \end{aligned}$$

where $W \in \mathcal{O}_{n\times q}$, $W = V^T U$, and $U = VW$, since $U, V \in \mathcal{O}_n$ from the eigendecomposition. Let $Z \in \mathcal{O}_{n\times q}$ be an orthonormal basis for $\mathcal{R}(W)^\perp$. Then, the matrices Σ and WW^T have the following full SVDs:

$$\Sigma = \begin{pmatrix} I_q & \mathbf{0} \\ \mathbf{0} & I_{n-q} \end{pmatrix} \Sigma \begin{pmatrix} I_q & \mathbf{0} \\ \mathbf{0} & I_{n-q} \end{pmatrix}^T, \; WW^T = \begin{pmatrix} W & Z \end{pmatrix} \begin{pmatrix} I_q & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{n-q} \end{pmatrix} \begin{pmatrix} W^T \\ Z^T \end{pmatrix}.$$

Recall the previous remark. The answer to this problem is to choose $W = \begin{pmatrix} I_q & \mathbf{0} \end{pmatrix}^T$, so $U = V_q$, the first q columns of V. This results in the optimal objective value $\sum_{j=1}^q \lambda_j$. So we choose $U^* = \begin{pmatrix} u_1 & \dots & u_q \end{pmatrix}$, the q eigenvectors of $P = XX^T$ corresponding to the q largest eigenvalues of P. This is called **principal component analysis (PCA)**! □

Remark 2.39. Note that there is nothing special about U^* beyond the fact that it spans $\mathcal{U}^*$. Any basis of the form U^*Q with $Q \in \mathcal{O}_k$ spans the same optimal subspace $\mathcal{U}^*$. We also note that the subspace $\mathcal{U}^*$ might not be unique: consider when $\lambda_q = \lambda_{q+1}$.

To summarize, to solve $\min_{U\in\mathcal{O}_{n\times q}} \|X - UU^T X\|_F^2$, we find the q largest eigenvalues of XX^T and a corresponding set of orthonormal eigenvectors U^*. Then over all q dimensional subspaces, $\mathcal{U}^* = \mathcal{R}(U^*)$ minimizes the sum of the squared norms of the projection residuals. By projecting each x_j to $\hat{x}_j = U^*(U^*)^T x_j$, we obtain an approximation of the data as points in the subspace $\mathcal{U}^*$. However, we can also represent $\hat{x}_j$ with its coordinates $c_j = (U^*)^T x_j$ with respect to the orthonormal basis U^*. The vectors $c_j \in \mathbb{R}^q$ uniquely specify the corresponding points $\hat{x}_j$. Thus, we have linearly mapped the data into q-dimensional space.

2.2.2 An alternative approach

We consider another way of thinking about PCA. Assume that the data has been centered. Now, we examine how the data is spread around $\mathbf{0}$. Select a unit norm vector $u \in \mathbb{R}^n$ and project x_j onto the line through $\mathbf{0}$ in the direction u, yielding $\hat{x}_j = uu^T x_j,\ j = 1, \dots, p$. Since the direction is fixed to be u, the projected data is effectively specified by the set of scalars $u^T x_j$, which has zero sample mean. The **sample variance**, the spread of the data,

in direction u is

$$\begin{aligned}\sigma^2(u) &= \frac{1}{p}\sum_{j=1}^{p}(u^T x_j)^2 \\ &= \frac{1}{p}\sum_{j=1}^{p} u^T x_j x_j^T u \\ &= u^T\left(\frac{1}{p}\sum_{j=1}^{p} x_j x_j^T\right)u \\ &= u^T R u,\end{aligned}$$

where we define the **sample covariance matrix** as $R := \frac{1}{p}\sum_{j=1}^{p} x_j x_j^T$, a symmetric positive semidefinite $n \times n$ matrix. More generally, if the data has not been centered and has sample mean μ, then the sample covariance is

$$R = \frac{1}{p}\sum_{j=1}^{p}(x_j - \mu)(x_j - \mu)^T.$$

Thus, the variance of the data in direction u is given by

$$\sigma^2(u) = u^T R u.$$

It might be interesting to observe the directions of largest variance of the data, for they capture most of the variability in the data. The direction of maximum variance is

$$u_1 = \arg\max_{\|u\|=1} u^T R u.$$

As we have seen before, u_1 is a unit norm eigenvector corresponding to the largest eigenvalue σ_1 of R. Let's say we want to find a second direction with the second largest variance; we do not want it to be arbitrarily close to u_1, so we can constrain the second direction to be orthogonal to the first. For q orthogonal directions, then, we want to find $U = \begin{pmatrix} u_1 & \dots & u_q \end{pmatrix} \in \mathcal{O}_{n\times q}$ to maximize $\sum_{j=1}^{q} u_j^T R u_j = \mathrm{Tr}(U^T R U)$. We have seen this problem before! The solution is attained by taking the q directions to be unit norm eigenvectors $u_1, \dots, u_q$ for the largest q eigenvalues of R. Generalizing, we can see that we obtain n orthonormal directions of maximum variance in the data: these directions $v_1, \dots, v_n$ and the corresponding variances $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_n^2$ are eigenvectors and corresponding eigenvalues of R, with $Rv_j = \sigma_j^2 v_j,\ j = 1, \dots, n$. The vectors v_j are called the **principal components** of the data, and this decomposition is called PCA. Let V be the matrix with the v_j's as its columns and $\Sigma^2 = \mathrm{diag}(\sigma_1^2, \dots, \sigma_n^2)$. So PCA is an ordered eigendecomposition of the sample covariance matrix $R = V\Sigma^2 V^T$.

We can quickly see that this interpretation of PCA is the same as the previous approach of finding a subspace that minimizes the sum of squared norms of the residuals. We can write

$$R = \frac{1}{p}\sum_{j=1}^{p} x_j x_j^T = \frac{1}{p} X X^T = \frac{1}{p} U\Sigma V^T V \Sigma U^T = \frac{1}{p} U \Sigma^2 U^T.$$

So the principal components are simply the eigenvectors of XX^T listed in order of decreasing eigenvalues, and in particular, the first q principal components are the first q eigenvectors of XX^T, which define the orthonormal basis U^* and, thus, an optimal q-dimensional projection subspace $\mathcal{U}^*$. *The leading q principal components gives a particular orthonormal basis for an optimal q-dimensional projection subspace.*

Like in our previous discussion of the optimal subspace, we can form $V_q = \begin{pmatrix} v_1 & \dots & v_q \end{pmatrix}$, which defines a subspace capturing the q directions of largest variance. We can project the data onto the span of the columns of V_q: $\hat{x}_j = V_q(V_q^T x_j)$, where $c_j = (V_q^T)x_j$ gives the coordinates of x_j with respect to V_q, and $V_q c_j$ synthesizes $\hat{x}_j$ using these coefficients to form the appropriate linear combination of the columns of V_q.

Thus, PCA is essentially looking around to find orthogonal directions of maximum variance, assuming that maximum variance is important. This is also a pitfall of PCA: these dimensions might not be the most significant representations of the data, just that these dimensions have highest variance. Note that feature selection is different—feature selection focuses on determining which features are important, while PCA focuses on bundling features together and finding directions of highest variance. Extensions of PCA include online PCA and probabilistic PCA, which puts Bayesian priors on the components.

Question 2.40. How do we interpret the results of PCA?

Question 2.41. How does one select the dimension q? Clearly, this involves a tradeoff between the size of q and the amount of variation in the original data captured in the projection. Note that there is no universally good way of choosing q. We can basically do some grid search with cross validation to find the best q.

2.2.3 PCA computation

We have shown that the q-dimensional PCA projection subspace is spanned by the leading eigenvectors of XX^T or, equivalently, of $R = \frac{1}{p}XX^T$. In practice, it is usually more efficient to compute a compact SVD of X and use this to find the principal components. Let $X = U\Sigma V^T$ be a compact SVD. Then,

$$XX^T = U\Sigma V^T V\Sigma U^T = U\Sigma^2 U^T.$$

Thus, the principal components with nonzero variances are the left singular vectors U in a compact SVD of X, and the variance of the data in direction u_j is the square of the singular value σ_j.

3 Convexity and least squares regression

3.1 Learning a function

The classification problem deals with training data $\{x_i\}_{i=1}^m$, $x_i \in \mathbb{R}^n$, with labels $\{y_i\}_{i=1}^m$, $y_i \in \mathcal{L}$, a finite discrete set. A **classifier** (or **estimator** or **predictor**) is a function $\hat{y} : \mathbb{R}^n \to L$. With a new data point $x \in \mathbb{R}^n$, $\hat{y}(x)$ is the assigned label, or the **estimate**.

Note that learning a linear function and classification have many similarities, but one key difference is that we can exploit the label space while learning a linear function, whereas the label space of classification is simply a finite discrete set.

Let $\mathcal{G} = \{g : \mathbb{R}^n \to \mathbb{R} \text{ with } g \text{ satisfying some constraint}\}$, so $\mathcal{G}$ is a set of estimators, predictors, or models. For instance, we could have $g_w(x) = w^T x,\ x \in \mathbb{R}^n,\ w \in \mathbb{R}^n$, where w is a parameter that we wish to learn. An example is the function $y = f(x) + v$, where $f(x)$ is the **latent function** and v is noise. Note that f itself might not even be inside $\mathcal{G}$. Thus, this results in forms of error. We could define some notion of the "best" $\hat{g} \in \mathcal{G}$, incurring an error $e(x) = \hat{g}(x) - f(x)$. To quantify the error, we could set up a loss function $L(e) = (\hat{g}(x) - f(x))^2$. We want to find the "average" of the loss function across x, particularly across testing data $\{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^k$, and we can accomplish this by calculating the **empirical loss** of the testing data:

$$\frac{1}{k}\sum_{i=1}^{k}(\hat{g}(\tilde{x}_i) - f(\tilde{x}_i))^2 = \frac{1}{k}\sum_{i=1}^{k}(\hat{g}(\tilde{(x)}_i) - \tilde{y}_i)^2 > 0,$$

which we call our **structural error**. Additionally, we have **generalization error** due to noise (assuming $e(x) > 0$).

Let's take a closer look at $g = \{g_w : g_w(x) = w^T x\},\ g_w : \mathbb{R}^n \to \mathbb{R}$. We take training data $X = \begin{pmatrix} x_1 & x_2 & \dots & x_m \end{pmatrix} \in \mathbb{R}^{n \times m}$ and put our labels in a column vector

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{R}^m.$$

Our function is $g_w(X) = X^T w \in \mathbb{R}^m$. We can write our loss as

$$L(w) = \|y - X^T w\|_2^2 = \|y - Aw\|_2^2,$$

where

$$A = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{pmatrix}.$$

The columns of A are correspond to the features. In general, we can add additional constraints in addition to g_w being linear; for instance, we could solve a problem

$$\begin{aligned} \min_{w \in \mathbb{R}^n} \quad & \|y - Aw\|_2^2 \\ \text{s.t.} \quad & h(w) \leq 0. \end{aligned}$$

Note that our **objective function** $\|y - Aw\|_2^2$ is convex. If $h(w)$ is also convex, this problem is called a **convex program**. There are many loss metrics that we could use. An example is $L(\hat{g}) = (\hat{g}(x) - y)^2$. The **sum of residual squares**, or **least squares**, is $L(\hat{g}) = \frac{1}{k}\sum_{i=1}^{k}(\hat{g}(x) - y)^2$.

3.2 Convexity

How do we recognize a convex function as convex? What does the derivative of the function tell us about its convexity? We can note, for instance, that the tangent of a convex function is a lower function for the entire function. In this section, we will develop a set of rules that will help us classify functions as convex.

Example 3.1. The classic **ordinary least squares problem** is $\min_{w\in\mathbb{R}^n} \|y - Aw\|_2^2$. This particular objective function $f(w) = \|y - Aw\|_2^2$ is a convex function.

Definition 3.2. (Bounded, closed, and compact sets). A subset $S \subset \mathbb{R}^n$ is **bounded** if there exists $B > 0$ such that for each $x \in S$, $\|x\|_2 \leq B$. S is **closed** if, roughly speaking, it contains its boundary. More precisely: if $\{x_k\}$ is a sequence of points in S that converges as $k \to \infty$ to a point $x \in \mathbb{R}^n$, then $x \in S$. So, a closed set contains the limits of its convergent sequences. S is **compact** if both closed and bounded. The implications of these definitions are very important; for one, every real valued continuous function defined on a compact subset $S \subset \mathbb{R}^n$ achieves a minimum value and maximum value over S.

Definition 3.3. (Convex set). Let $S \subseteq \mathbb{R}^n$ and $\alpha \in [0, 1]$. The line segment joining $x, y \in S$ is $z_\alpha = (1 - \alpha)x + \alpha y$. The set S is convex if $\forall x, y \in S,\ \forall\ \alpha \in [0, 1],\ z_\alpha \in S$.

Example 3.4. Here are some examples of convex sets.

- The empty set and $\mathbb{R}^n$ are trivial convex subsets of $\mathbb{R}^n$.
- Any subspace $\mathcal{U} \subset \mathbb{R}^n$ is a convex set.
- An **affine manifold** is a set of the form $S = s + \mathcal{U} = \{x = s + u,\ u \in \mathcal{U}\}$, where $s \in \mathbb{R}^n$ and $\mathcal{U}$ is a subspace of $\mathbb{R}^n$ and is a convex set.
- A **closed half space** $H \subset \mathbb{R}^n$ is a set of the form $\{x : a^T x \leq b\}$, where $a \in \mathbb{R}^n,\ a \neq 0$ and $b \in \mathbb{R}$. A closed half space is a (closed) convex set and is one side of an $n - 1$ dimensional plane in $\mathbb{R}^n$, including the plane itself.

Theorem 3.5. *Some properties of convex sets are:*

- *Closure under intersection: If, for each $a \in A$, $S_a \subset \mathbb{R}^n$ is convex, then $\cap_{a\in A} S_a$ is convex.*
- *Image under a linear map: If $S \subset \mathbb{R}^n$ is convex, and F is a linear map from $\mathbb{R}^n$ to $\mathbb{R}^m$, then $F(S) = \{z : z = Fs,\ s \in S\}$ is convex.*

Corollary 3.6. *Let $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Then $\{x : Ax \leq b\}$ is a convex set.*

Proof. Let $S = \{x : Ax \leq b\}$. S is the intersection of the half spaces $\{x : a_j^T x \leq b_j\}$, where a_j^T is the j-th row of A and b_j is the j-th entry of b. Since half spaces are convex, the result follows by the previous theorem. □

Definition 3.7. (Convex function). The "bowl shaped function property" is

$$f((1-\alpha)x + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y)$$

must hold for all $x, y \in \mathbb{R}^n$ and for all $\alpha \in [0, 1]$.

Definition 3.8. (Strictly convex function). A function $f : \mathbb{R}^n \to \mathbb{R}$ is strictly convex if for all $x, y \in \mathbb{R}^n$ with $x \neq y$, and for all $\alpha \in (0, 1)$,

$$f((1-\alpha)x + \alpha y) < (1-\alpha)f(x) + \alpha f(y).$$

Theorem 3.9. *For any $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$, the affine function $f(x) = w^T x + b$ is a convex function.*

Proof. Take $z_\alpha = (1-\alpha)x + \alpha y$.

$f(z_\alpha) = w^T((1-\alpha)x + \alpha y) + b = (1-\alpha)w^T x + \alpha w^T y + (1-\alpha)b + \alpha b = (1-\alpha)f(x) + \alpha f(y)$.

Thus, f is a convex function. □

Theorem 3.10. *Every norm on $\mathbb{R}^n$ is a convex function.*

Proof. By the triangle inequality and scaling, for any $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$,

$$\|\lambda x + (1-\lambda)y\| \leq \|\lambda x\| + \|(1-\lambda)y\| = \lambda\|x\| + (1-\lambda)\|y\|.$$

□

Theorem 3.11. *Let f, g be convex functions on $\mathbb{R}^m$. Then, the following functions are convex:*

- *$h(x) = \beta f(x)$, where $\beta \geq 0$.*
- *$h(x) = f(x) + g(x)$.*
- *$h(x) = \max\{f(x), g(x)\}$.*
- *$h(x) = f(Ax + b)$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$.*
- *$h(x) = g(f(x))$, where in addition to being convex, g is nondecreasing on range(f).*

Proof. We leave this as an exercise. □

Theorem 3.12. *If $P \in \mathbb{R}^{n \times n}$ is symmetric positive semidefinite, $w, q \in \mathbb{R}^n$, and $r \in R$, then the quadratic function $f(w) = \frac{1}{2}w^T Pw + q^T w + r$ is convex.*

Proof. $q^T w + r$ is convex as it is affine. $\|\cdot\|_2$ is convex since it is a norm. Thus, $\|\sqrt{P}w\|_2$ is convex and nonnegative. $g(x) = x^2$ is convex and monotonically increasing on the nonnegative reals. Thus, $w^T Pw = \|\sqrt{P}w\|_2^2$ is a convex function. Finally, nonnegative scaling preserves convexity, and the sum of convex functions is convex, so f is convex. □

Theorem 3.13. *Let f be a convex function on $\mathbb{R}^n$. For each $c \in \mathbb{R}$, the sublevel set $L_c = \{x : f(x) \leq c\}$ is a convex set.*

Proof. If L_c is empty, then it is convex. Otherwise, let $x, y \in L_c$. Then,

$$f((1-\alpha x) + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y) \leq c.$$

□

Theorem 3.14. *If a convex function f has a finite valued local minimum at $x^* \in \mathbb{R}^n$, then x^* is a global minimum of f.*

Proof. Let $f(x^*) = c > -\infty$. Then, there exists $r > 0$ such that for all x with $\|x - x^*\|_2 < r$, $f(x) \geq c$. Suppose that for some $z \in \mathbb{R}^n$, $f(z) < c$. Let $x_\alpha = (1-\alpha)x^* + \alpha z$, with $\alpha \in (0,1)$. Then for $\alpha > 0$ sufficiently small, $\|x_\alpha - x^*\|_2 < r$, and $f(x_\alpha) \leq (1-\alpha)f(x^*) + \alpha f(z) < c$, which is a contradiction. □

Theorem 3.15. *The set of all global minima of a convex function is a convex set.*

Proof. Let x^* be a global minimum with $f(x^*) = c > -\infty$. The set of all global minima is the sublevel set $L_c = \{x : f(x) \leq c\}$, which is convex. □

Theorem 3.16. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be strictly convex. If f has a finite valued local minimum at x^*, then x^* is the unique global minimum of f.*

Proof. Let x^* be a local minimum with $f(x^*) = c > -\infty$. Then x^* is a global minimum. If there are two distinct global minima, say at x^* and y^*, then all points on the line joining x^* and y^* are global minima, but this violates strict convexity of f. □

Theorem 3.17. *A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if for all $u, v \in \mathbb{R}^n$,*

$$f(v) - f(u) \geq Df(u)(v-u),$$

i.e., the best local linear approximation to f at every point is a global lower bound for f.

Proof. Assume that f is convex. Then, $f(x + \alpha(y-x)) \leq f(x) + \alpha(f(y) - f(x))$. So,

$$\frac{f(x + \alpha(y-x)) - f(x)}{\alpha} \leq f(y) - f(x).$$

Taking the limit as $\alpha \to 0$ yields $Df(x)(y-x) \leq f(y) - f(x)$.

For the reverse direction, assume that the bound holds. Let $x, y \in \mathbb{R}^n$, and consider the point $x_\alpha = (1-\alpha)x + \alpha y$ for some $\alpha \in [0,1]$. We want to show that $(1-\alpha)f(x) + \alpha f(y) \geq f(x_\alpha)$. Applying at $u = x_\alpha$, we get the lower bound $g(v) = f(x_\alpha) + Df(x_\alpha)(v - x_\alpha)$ to $f(v)$. Evaluating this bound at $v = x$ and $v = y$ gives $f(x) \geq g(x)$ and $f(y) \geq g(y)$. Hence, $(1-\alpha)f(x) + \alpha f(y) \geq (1-\alpha)g(x) + \alpha g(y)$, and we can write

$$\begin{aligned}(1-\alpha)f(x) + \alpha f(y) &\geq (1-\alpha)(f(x_\alpha) + Df(x_\alpha)(x - x_\alpha)) + \alpha(f(x_\alpha) + Df(x_\alpha)(y - x_\alpha)) \\ &= f(x_\alpha) + Df(x_\alpha)[(1-\alpha)(x - x_\alpha) + \alpha(y - x_\alpha)] \\ &= f(x_\alpha).\end{aligned}$$

□

Theorem 3.18. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable convex function, and let S be a convex subset of $\mathbb{R}^n$. A point x^* minimizes f over S if and only if $x^* \in S$ and for each $y \in S$,*

$$Df(x^*)(y - x^*) \geq 0.$$

Proof. Suppose that $x^* \in S$ and that $Df(x^*)(y-x^*) \geq 0$ forall $y \in S$. Since f is convex, for each $y \in S$, $f(y) - f(x^*) \geq Df(x^*)(y - x^*)$ by the previous theorem. Thus, $f(y) \geq f(x^*)$.

Now suppose that x^* minimizes f over S. Clearly, $x^* \in S$. Assume that $Df(x^*)(y-x^*) < 0$ for some $y \in S$. Let $h = y - x^*$, and note that for $\alpha \in [0, 1]$, $x^* + \alpha h = (1 - \alpha)x^* + \alpha y \in S$.

$$Df(x^*)h = \lim_{\alpha \to 0} \frac{f(x^* + \alpha h) - f(x^*)}{\alpha} < 0.$$

By the definition of a derivative and a limit, there exists $\alpha_0 > 0$ such that for all $0 < \alpha \leq \alpha_0$, $f(x^* + \alpha h) - f(x^*) < 0$. For such α, $x^* + \alpha h \in S$, and $f(x^* + \alpha h) < f(x^*)$, which is a contradiction. □

Lemma 3.19. *(Jensen's inequality). The value of a convex function at a (weighted) average of a set of points is less than the same average of the function values at these points.*

If $\{x_i\}_{i=1}^k \subset I$, $\alpha_i > 0$ with $\sum_{i=1}^k \alpha_i = 1$, then

$$f\left(\alpha_i x_i\right) \leq \sum_{i=1}^{k} \alpha_i f(x_i).$$

Lemma 3.20. *Let I be an interval of $\mathbb{R}$. If $f : I \to \mathbb{R}$ is convex, then f is continuous on the interior of I.*

Lemma 3.21. *Let $f : I \to \mathbb{R}$ be twice differentiable on an open interval I. Then f is convex on I if and only if $f''(x) \geq 0$ at each point $x \in I$.*

3.3 Least squares regression

3.3.1 Learning a linear function

Let $\mathcal{G}$ be the family of linear functions $\{g : g(x) = w^T x,\ w \in \mathbb{R}^n\}$. We aim to use a set of training examples $\{(x_j, y_j) \in \mathbb{R}^n \times \mathbb{R}\}_{j=1}^m$ to learn $\hat{w} \in \mathbb{R}^n$ so that $\hat{g}(x) = \hat{w}^T x$ best approximates the hidden relationship between x and y. Let $X = \begin{pmatrix} x_1 & \dots & x_m \end{pmatrix} \in \mathbb{R}^{n \times m}$ be the matrix of input training data and $y = \begin{pmatrix} y_j \end{pmatrix} \in \mathbb{R}^m$ be the vector of labels. The predicted values are $\hat{y} = X^T w \in \mathbb{R}^m$, with a **residual** vector of $\varepsilon = y - \hat{y} = y - X^T w$. Now, let $A = X^T$, so each input example is now a row and each column is a **feature**. For **linear regression problems**, A is called the **regression matrix** or **design matrix**, and a column of A is called a **regressor**.

3.3.2 Ordinary least squares and related problems

Least squares is a linear regression method to select the parameter w that minimizes $\|\varepsilon\|_2^2 = \|y - Aw\|_2^2$. This metric is called the **least squares** or **residual sum of squares** objective. The **least squares problem** is thus

$$\min_{w \in \mathbb{R}^n} \|y - Aw\|_2^2.$$

We can show that a lot of problems can be reduced to the least squares problem. We will also derive the optimal solution of the least squares problem in this section. Once we derive this solution, we can derive many insights applicable to the other related problems, which we highlight here.

(1) Learning an affine function
We can rewrite the related minimization involving an affine function as

$$\min_{w\in\mathbb{R}^n,\ \beta\in\mathbb{R}} \|y - Aw - \beta\|_2^2 \equiv \min_{z\in\mathbb{R}^{n+1}} \|y - \tilde{A}z\|_2^2,$$

with $\tilde{A} = \begin{pmatrix} A & \mathbf{1} \end{pmatrix}$ and $z = \begin{pmatrix} w \\ \beta \end{pmatrix}$.

(2) Quadratic performance metric
Our problem is

$$\min_{w\in\mathbb{R}^n} \|y - Aw\|_P^2,$$

where $P \in \mathbb{R}^{n\times n}$ is symmetric positive definite, and $\|x\|_P = \sqrt{x^T P x}$ is the **Mahalanobis norm**. Recall that we can write $\sqrt{P}\sqrt{P} = P \iff \sqrt{P}^T = \sqrt{P} \iff \sqrt{P}$ is positive definite, since we can write

$$\begin{aligned} P &= U\Sigma U^T \\ &= (U\sqrt{\Sigma})(\sqrt{\Sigma}U^T) \\ &= (U\sqrt{\Sigma}U^T)(U\sqrt{\Sigma}U^T) \\ &= \sqrt{P}\sqrt{P}, \end{aligned}$$

Thus,

$$\begin{aligned} \|y - Aw\|_P^2 &= (y - Aw)^T P (y - Aw) \\ &= (y - Aw)^T \sqrt{P}^T \sqrt{P}(y - Aw) \\ &= \|\tilde{y} - \tilde{A}w\|_2, \end{aligned}$$

where $\tilde{y} = \sqrt{P}y$ and $\tilde{A} = \sqrt{P}A$.

(3) Multiple least squares objectives
We can rewrite

$$\min_{w\in\mathbb{R}^n} \|y_1 - A_1 w\|_2^2 + \|y_2 - A_2 w\|_2^2,$$

which has two potentially competing objectives in the minimization, as

$$\begin{aligned} &\min_{w\in\mathbb{R}^n} \left\| \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} w \right\|_2^2 \\ &= \min_{w\in\mathbb{R}^n} \left\| \tilde{y} - \tilde{A}w \right\|, \end{aligned}$$

where $\tilde{y} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in \mathbb{R}^{m_1+m_2}$ and $\tilde{A} = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} \in \mathbb{R}^{(m_1+m_2)\times n}$.

(4) Ridge regression

$$\min_{w\in\mathbb{R}^n} \|y - Aw\|_2^2 + \lambda\|w\|_2^2,\ \lambda > 0.$$

Note that **ridge regression** is a special case of the previous case (multiple least squares objectives), where $y_1 = y$, $A_1 = A$, $A_2 = \sqrt{\lambda}I_n$, and $y_2 = \mathbf{0}$. So the problem can be rewritten as

$$\min_{w\in\mathbb{R}^n} \|\tilde{y} - \tilde{A}w\|_2^2,\ \lambda > 0.$$

where $\tilde{y} = \begin{pmatrix} y \\ \mathbf{0} \end{pmatrix} \in \mathbb{R}^{m+n}$ and $\tilde{A} = \begin{pmatrix} A \\ \sqrt{\lambda}I_n \end{pmatrix} \in \mathbb{R}^{(m+n)\times n}$.

3.3.3 The least squares solution

Let's rewrite our minimization problem (note that it is a convex optimization problem and that we can essentially rewrite it as a quadratic function of w):

$$\begin{aligned} L(w) = \|y - Aw\|_2^2 &= (y - Aw)^T(y - Aw) \\ &= y^Ty - y^TAw - w^TA^Ty + w^TA^TAw \\ &= w^TA^TAw - 2y^TAw + y^Ty. \\ DL(w)h &= w^TA^TAh + h^TA^TAw - 2y^TAh \\ &= 2(w^TA^TA - y^TA)h = 0\ \forall\ h \\ &\implies w^TA^TA - y^TA = 0 \\ &\implies A^TAw = A^Ty, \end{aligned}$$

a set of equations called the **normal equations**.

Let's check the second derivative:

$$D^2L(w)(h) = 2h(A^TA)h.$$

Since A^TA is symmetric positive semidefinite at every stationary point of the derivative, every solution of the normal equations $A^TAw^* = A^Ty$ is a local (and hence global) minimum. We refer to any such vector $w^* \in \mathbb{R}^n$ satisfying $\hat{y} = Aw^*$ as a **least squares solution**. Every least squares solution w^* gives an exact representation of $\hat{y}$ as a linear combination of the columns of A. Thus, w^* is unique if and only if the columns of A are linearly independent; if $(A^TA)^{-1}$ exists, or if $\text{rank}(A) = n \leq m$, then w^* is unique, and

$$w^* = (A^TA)^{-1}A^Ty.$$

$(A^TA)^{-1}$ exists when the columns of A are linearly independent or if A^TA is positive definite (if A^TA is only positive semidefinite, then we have more work to do). On the other hand, if $\mathcal{N}(A)$ is nontrivial, then we have a class of solutions. In this case, the set of solutions is translated from $\mathcal{N}(A)$, and the optimal solution is referred to as the **least norm solution**.

Let w^* be a solution of $A^TAw = A^Ty$. Our predicted labels are $\hat{y} = Aw^*$ (recall that our original prediction is Aw, and our minimization problem is $\min_{w\in\mathbb{R}^n} \|y - Aw\|_2^2$). The residual is $\varepsilon = y - \hat{y}$.

$$\begin{aligned} A^T(\varepsilon) &= A^T(y - \hat{y}) \\ &= A^T(y - Aw^*) \\ &= A^Ty - A^TAw^* \\ &= 0. \end{aligned}$$

So the residual is in the null space of A^T, and since the rows of A^T are the transposed regressors, the residual is orthogonal to each regressor and, hence, to $\mathcal{R}(A)$. So we can write

$$y = \hat{y} + \varepsilon \text{ with } \hat{y} \in \mathcal{R}(A) \text{ and } \varepsilon \in \mathcal{R}(A)^\perp,$$

and $\hat{y}$ is the unique orthogonal projection of y onto $\mathcal{R}(A)$. While $\hat{y}$ is unique, if A has a nontrivial nullspace, then the *representation* of $\hat{y}$ as a linear combination of the columns of A is *not* unique; the representation is unique if and only if $\text{rank}(A) = n$.

How do we compute a solution? Let $A = U\Sigma V^T$ be a compact SVD of A. Substituting this into the normal equations gives

$$\begin{aligned} V\Sigma U^TU\Sigma V^Tw &= V\Sigma U^Ty \\ \implies \Sigma^2V^Tw &= \Sigma U^Ty \\ \implies VV^Tw &= V\Sigma^{-1}U^Ty. \end{aligned}$$

Recall that the columns of V span $\mathcal{N}(A)^\perp$. If the columns of A are linearly independent, then $\mathcal{N}(A) = \mathbf{0}$ and $\mathcal{N}(A)^\perp = \mathbb{R}^n$. In this case, $V \in \mathcal{O}_n$, and the unique **least squares solution** is

$$w^* = V\Sigma^{-1}U^Ty.$$

If the columns of A are linearly dependent, $\mathcal{N}(A)$ is nontrivial, and so if w^* is a solution of the normal equations, then so is $w^* + v$ for every $v \in \mathcal{N}(A)$. Conversely, if w is a solution of the normal equations, then $A^TA(w - w^*) = 0$, so $w = w^* + v$ with $v \in \mathcal{N}(A)$. Thus, the set of solutions is a linear manifold formed by a translation of the subspace $\mathcal{N}(A)$ by a particular solution.

Lemma 3.22. *Let A have compact SVD $U\Sigma V^T$. The unique **least norm solution** is*

$$w^*_{ln} = V\Sigma^{-1}U^Ty.$$

Proof. First, we prove that w^*_{ln} is a solution. Note that

$$\|y - Aw\|_2^2 = \|(I - UU^T)y + UU^Ty - U(\Sigma V^Tw)\|_2^2 = \|(I - UU^T)y\|_2^2 + \|U(U^Ty - \Sigma V^Tw)\|_2^2.$$

The first term is a constant, and we can make the second term zero by setting $w = w^*_{\text{ln}}$. So w^*_{ln} is a solution. If w^* is any solution, then we can write $w^* = w^*_{\text{ln}} + w_0$ for some $w_0 \in \mathcal{N}(A)$. Since $w^*_{\text{ln}} \in \mathcal{N}(A)^\perp$, we have that $w^*_{\text{ln}} \perp w_0$, so $\|w^*\|_2^2 = \|w^*_{\text{ln}}\|_2^2 + \|w_0\|_2^2$. $\|w^*\|_2^2 \geq \|w^*_{\text{ln}}\|_2^2$ with equality if and only if $w^* = w^*_{\text{ln}}$. □

Remark 3.23. w_{ln}^* is obtained by projecting y onto the r principal components of the features, $U^T y$, then using $V\Sigma^{-1}$ to map these coordinates to the vector $w_{\text{ln}}^* \in \mathcal{N}(A)^\perp$. Let U_k and V_k denote the matrices consisting of the first k columns of U and V respectively and Σ_k denote the top left $k \times k$ submatrix of Σ. The least squares solution based on the rank k approximation $U_k \Sigma_k V_k^T$ to A is

$$w_k^* = V_k \Sigma_k^{-1} U_k^T y = \sum_{j=1}^{k} \frac{1}{\sigma_j} v_j u_j^T y.$$

This gives us a sequence of solutions $w_k^*,\ k = 1, \ldots, r$, with $w_r^* = w_{\text{ln}}^*$.

3.3.4 Tikhonov and ridge regularization

Another way to ensure a unique solution is to add a new term to the objective function. An example of such an approach is

$$\min_{w \in \mathbb{R}^n} \|y - Aw\|_2^2 + \lambda \|w\|_2^2,\ \lambda > 0.$$

Imposing such an additional component is called **regularization**. **Tikhonov regularized least squares** is another approach and can be written as

$$\min_{w \in \mathbb{R}^n} \|y - Aw\|_2^2 + \lambda \|g - Fw\|_2^2,\ \lambda > 0.$$

$g \in \mathbb{R}^k$ and $F \in \mathbb{R}^{k \times n}$ are the new elements of the regularization penalty. **Ridge regularization** is a special case with $F = I_n$ and $g = \mathbf{0}$. Note that we could assume a Gaussian prior on w to represent the prior knowledge of w. Also, note that there is an optimal value of λ, which has to do with the size of the dataset and the noisiness of the dataset.

As it is a sum of squares, the objective can be expressed as

$$\|y - Aw\|_2^2 + \lambda \|g - Fw\|_2^2 = \left\| \begin{pmatrix} y - Aw \\ \sqrt{\lambda}(g - Fw) \end{pmatrix} \right\|_2^2 = \|\tilde{y} - \tilde{A}w\|_2^2,$$

where

$$\tilde{y} = \begin{pmatrix} y \\ \sqrt{\lambda} g \end{pmatrix} \in \mathbb{R}^{m+k},\ \tilde{A} = \begin{pmatrix} A \\ \sqrt{\lambda} F \end{pmatrix} \in \mathbb{R}^{(m+k) \times n}.$$

Thus, we just need to solve a basic least squares problem with an augmented vector $\tilde{y}$ and matrix $\tilde{A}$. The corresponding augmented normal equations are

$$\tilde{A}^T \tilde{A} w = \tilde{A}^T \tilde{y} = \begin{pmatrix} A^T & \sqrt{\lambda} F^T \end{pmatrix} \begin{pmatrix} y \\ \sqrt{\lambda} g \end{pmatrix}.$$

$$\implies (A^T A + \lambda F^T F) w = A^T y + \lambda F^T g.$$

If $\text{rank}(\tilde{A}) = n$, then the augmented problem has the unique solution

$$w^*(\lambda) = (A^T A + \lambda F^T F)^{-1} (A^T y + \lambda F^T g).$$

Note that a sufficient condition ensuring that $\text{rank}(\tilde{A}) = n$ is $\text{rank}(F) = n$.

Ridge regression, which has $k = n$, $F = I_n$, and $g = \mathbf{0}$, always has a unique solution:

$$w^*_{\text{rr}}(\lambda) = (A^TA + \lambda I_n)^{-1}A^Ty.$$

This is because A^TA is positive semidefinite, and adding λI_n ensures that the sum is positive definite and thus invertible. These solutions are functions of the regularization parameter λ: as λ varies, the solutions trace out a curve in $\mathbb{R}^n$ called the **regularization path**.

Remark 3.24. The residual $\varepsilon = y - Aw^*$ under Tikhonov and ridge regularization is generally not orthogonal to the columns of A. Take, for instance, ridge regularization: using the normal equations, we can write:

$$\begin{aligned} A^TAw^*_{\text{rr}} + \lambda w^*_{\text{rr}} &= A^Ty \\ \implies A^T(Aw^*_{\text{rr}} - y) &= -\lambda w^*_{\text{rr}} \\ \implies A^T(y - Aw^*_{\text{rr}}) &= \lambda w^*_{\text{rr}}, \end{aligned}$$

where the residual is $y - Aw^*_{\text{rr}}$. However, for all points on the ridge regularization path, $w^*_{\text{rr}}(\lambda) \in \mathcal{N}(A)^\perp$.

Lemma 3.25. *For $\lambda > 0$, $w^*_{rr}(\lambda) \in \mathcal{N}(A)^\perp$.*

Proof. Let $w^*_{\text{rr}}(\lambda) = v + w$ with $v \in \mathcal{N}(A)^\perp$ and $w \in \mathcal{N}(A)$. Then, $Aw^*_{\text{rr}}(\lambda) = A(v+w) = Av$, and $\|Aw^*_{\text{rr}}(\lambda) - y\|_2^2 = \|Av - y\|_2^2$. Thus, $w^*_{\text{rr}}(\lambda)$ and v have the same least squares cost. By Pythagoras, $\|w^*_{\text{rr}}(\lambda)\|_2^2 = \|v\|_2^2 + \|w\|_2^2$. So $w = 0$. □

Let's bring in a compact SVD $A = U\Sigma V^T$. The columns of U form an orthonormal basis for $\mathcal{R}(A)$, and the columns of V form an orthonormal basis for $\mathcal{N}(A)^\perp$. We get

$$(A^TA + \lambda I_n)w^*_{\text{rr}} = (V\Sigma^2V^T + \lambda I_n)w^*_{\text{rr}},$$

$$A^Ty = V\Sigma U^Ty.$$

By the previous lemma, $VV^Tw^*_{\text{rr}} = w^*_{\text{rr}}$. Thus,

$$(V\Sigma^2V^T + \lambda I_n)w^*_{\text{rr}} = (V\Sigma^2V^T + \lambda I_n)VV^Tw^*_{\text{rr}} = V(\Sigma^2 + \lambda I_r)V^Tw^*_{\text{rr}}.$$

We can write the normal equations as

$$\begin{aligned} V(\Sigma^2 + \lambda I_r)V^Tw^*_{\text{rr}} &= V\Sigma U^Ty \\ \implies w^*_{\text{rr}} &= V(\Sigma^2 + \lambda I_r)^{-1}\Sigma U^Ty \\ \implies w^*_{\text{rr}} &= V\text{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda}\right)U^Ty \\ \implies w^*_{\text{rr}} &= \sum_{j=1}^{r} \frac{\sigma_i}{\sigma_i^2 + \lambda} v_i u_i^T y_i. \end{aligned}$$

Lemma 3.26. $\lim_{\lambda \to 0} w^*_{rr}(\lambda) = w^*_{ln}$.

Proof.

$$\lim_{\lambda \to 0} w_{\text{rr}}^*(\lambda) = \lim_{\lambda \to 0} V(\Sigma^2 + \lambda I_r)^{-1} \Sigma U^T y = V \Sigma^{-1} U^T y = w_{\text{ln}}^*.$$

□

So as λ increases from 0, the ridge solution $w_{\text{rr}}^*(\lambda)$ starts at w_{ln}^*, moves along the regularization path within the subspace $\mathcal{N}(A)^\perp$, and gradually shrinks to $\mathbf{0}$ as $\lambda \to \infty$.

Remark 3.27. Think before you blindly apply the pseudo-inverse. Often, it is much more optimal to think in terms of the SVD.

3.3.5 Online least squares

What if we receive our data (x_j, y_j) one at a time? How can we solve the least squares problem as efficiently as possible? We cover a process referred to as **online, or recursive, least squares (RLS)**.

Let $A_m \in \mathbb{R}^{m \times n}$ be the design matrix, $y_m \in \mathbb{R}^m$ be the data vector, and assume that A_m has n linearly independent columns. We are adding rows to a narrow regression matrix ($n < m$), so we write the normal equations in terms of the rows of A_m. We have $(A^T A)w^* = A^T y$.

$$P_m = A_m^T A_m = \begin{pmatrix} x_1 & \dots & x_m \end{pmatrix} \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix} = \sum_{j=1}^{m} x_j x_j^T$$

and

$$s_m = A_m^T y_m = \sum_{j=1}^{m} x_j y(j).$$

Every time we add a new data point, we can just update via the following:

$$P_{m+1} = A_{m+1}^T A_{m+1} = P_m + x_{m+1} x_{m+1}^T,$$

$$s_{m+1} = A_{m+1}^T y_{m+1} = s_m + y(m+1) x_{m+1}.$$

P_{m+1} and s_{m+1} have the same dimensions as $P_m \in \mathbb{R}^{n \times n}$ and $s_m \in \mathbb{R}^n$, respectively, so solving the updated normal equations still takes $O(n^3)$ time. Can we use our previous computation of P_m^{-1} to help us out?

Lemma 3.28. *Let $P \in \mathbb{R}^{k \times k}$ be an invertible symmetric matrix and $d \in \mathbb{R}^k$. If $1 + d^T P^{-1} d \neq 0$, then the matrix $P + dd^T$ has the inverse*

$$(P + dd^T)^{-1} = P^{-1} - \frac{1}{1 + d^T P^{-1} d} (P^{-1} d)(P^{-1} d)^T.$$

Proof. Multiply the RHS by $(P + dd^T)$:

$$(P + dd^T) \left(P^{-1} - \frac{1}{1 + d^T P^{-1} d} (P^{-1} d)(P^{-1} d)^T \right)$$

$$= I_n + dd^T P^{-1} - \frac{1}{1 + d^T P^{-1} d} dd^T P^{-1} - \frac{1}{1 + d^T P^{-1} d} d(d^T P^{-1} d) d^T P^{-1}$$

$$= I_n + \left(1 - \frac{1}{1 + d^T P^{-1} d} - \frac{d^T P^{-1} d}{1 + d^T P^{-1} d}\right) d d^T P^{-1}$$
$$= I_n.$$

□

We assume that the columns of A_m are linearly independent. Thus, the least squares solution is given by $w_m^* = P_m^{-1} s_m$. When a new training example is added, A_{m+1} still has n linearly independent columns—adding a row does not change this—so P_{m+1} is still invertible. Hence, we can simply compute

$$P_{m+1}^{-1} = P_m^{-1} - \frac{(P_m^{-1} x_{m+1})(P_m^{-1} x_{m+1})^T}{1 + x_{m+1}^T P_m^{-1} x_{m+1}},$$

$$w_{m+1}^* = P_{m+1}^{-1} s_{m+1}.$$

Simplifying, we get

$$\hat{y}(m+1) = x_{m+1}^T w_m^*,$$

$$w_{m+1}^* = w_m^* + \frac{P_m^{-1} x_{m+1}}{1 + x_{m+1}^T P_m^{-1} x_{m+1}} (y(m+1) - \hat{y}(m+1)).$$

4 Probabilistic models

4.1 The multivariate Gaussian

Let X, which takes values in $\mathbb{R}^n$, have density $f_X(x)$, mean μ_X, and covariance Σ_X. Similarly, let Y, which takes values in $\mathbb{R}^q$, have density $f_Y(y)$, mean μ_Y, and covariance Σ_Y. Σ_X describes the second order dependencies among the components of $X - \mu_X$, and same with Y. We bring in the joint vector $Z = \begin{pmatrix} X \\ Y \end{pmatrix}$, which has mean

$$\mu_Z = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}$$

and covariance

$$E[(Z - \mu_Z)(Z - \mu_Z)^T] = \Sigma_Z = \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix},$$

where $\Sigma_{XY} \in \mathbb{R}^{n \times q}$ is the **cross variance of X and Y** and $\Sigma_{YX} \in \mathbb{R}^{q \times n}$ is the **cross variance of Y and X**. $\Sigma_{XY}^T = \Sigma_{YX}$. The conditional density of Y given $X = x$ is

$$f_{Y|X}(y|x) = \frac{f_{XY}(x, y)}{f_X(x)},$$

for all x for which $f_X(x) > 0$.

4.1.1 The multivariate Gaussian density

Let the random vector X, which takes values in $\mathbb{R}^n$, have mean $\mu \in \mathbb{R}^n$ and covariance $\Sigma \in \mathbb{R}^{n \times n}$. We say that X is a **nondegenerate Gaussian random vector** if Σ is positive definite and X has the **multivariate Gaussian density**

$$f_X(x) = \frac{1}{(2\pi)^{\frac{n}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}.$$

We call $K := \Sigma^{-1}$ the **precision matrix** of the density. Note that

$$\ln f_X(x) = -\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu) + C = -\frac{1}{2} x^T K x + x^T K \mu + C'.$$

Lemma 4.1. *If $f_X(x)$ is a density and $\ln f_X(x)$ has the form above, then $f_X(x)$ is a Gaussian density.*

4.1.2 Jointly Gaussian random vectors

Denote the density of Z as $f_{XY}(x, y)$ and the marginal densities of X and Y by $f_X(x)$ and $f_Y(y)$.

Theorem 4.2. *When $f_{XY}(x, y)$ is a Gaussian density, the conditional density $f_{Y|X}(y|x)$ is a Gaussian density with mean $\mu_{Y|X}$ and covariance $\Sigma_{Y|X}$ given by*

$$\mu_{Y|X} = \mu_Y + \Sigma_{YX} \Sigma_X^{-1}(x - \mu_X).$$

$$\Sigma_{Y|X} = \Sigma_Y - \Sigma_{YX} \Sigma_X^{-1} \Sigma_{XY}.$$

Proof. Note that $\ln f_{XY}(x,y) = \ln f_{Y|X}(y|x) + \ln f_X(x)$.

$$\ln f_{XY}(x,y) = -\frac{1}{2}\begin{pmatrix} x-\mu_X \\ y-\mu_Y \end{pmatrix}^T \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix}^{-1} \begin{pmatrix} x-\mu_X \\ y-\mu_Y \end{pmatrix} + C,$$

where C does not depend on x or y. Under the assumption of the existence of the inverses below, using the lemmas in the appendix, we can write

$$\begin{aligned}
\ln f_{XY}(x,y) &= -\frac{1}{2}\begin{pmatrix} x-\mu_X \\ y-\mu_Y \end{pmatrix} \begin{pmatrix} I & -\Sigma_X^{-1}\Sigma_{XY} \\ \mathbf{0} & I \end{pmatrix} \begin{pmatrix} \Sigma_X^{-1} & \mathbf{0} \\ \mathbf{0} & S_{\Sigma_X}^{-1} \end{pmatrix} \begin{pmatrix} I & \mathbf{0} \\ -\Sigma_{YX}\Sigma_X^{-1} & I \end{pmatrix} \begin{pmatrix} x-\mu_X \\ y-\mu_Y \end{pmatrix} + C \\
&= -\frac{1}{2}\begin{pmatrix} x-\mu_X \\ (y-\mu_Y) - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X) \end{pmatrix}^T \begin{pmatrix} \Sigma_X^{-1} & \mathbf{0} \\ \mathbf{0} & S_{\Sigma_X}^{-1} \end{pmatrix} \begin{pmatrix} x-\mu_X \\ (y-\mu_Y) - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X) \end{pmatrix} + C \\
&= -\frac{1}{2}(x-\mu_X)^T\Sigma_X^{-1}(x-\mu_X) + C \\
&= -\frac{1}{2}(y-\mu_Y - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X))^T S_{\Sigma_X}^{-1}(y-\mu_Y - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X)) + C,
\end{aligned}$$

where $S_{\Sigma_X} = \Sigma_Y - \Sigma_{YX}\Sigma_X^{-1}\Sigma_{XY}$. *** Now, fix x, so we can write

$$\ln f_{X|Y}(x|y) = -\frac{1}{2}(y-\mu_Y - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X))^T S_{\Sigma_X}^{-1}(y-\mu_Y - \Sigma_{YX}\Sigma_X^{-1}(x-\mu_X)) + C + C',$$

which as a function of y, is a Gaussian density. □

Lemma 4.3. *If the joint density $f_{XY}(x,y)$ is Gaussian, then so are the marginal densities $f_X(x)$ and $f_Y(y)$. Moreover, the means and covariances are μ_X, Σ_X and μ_Y, Σ_Y, respectively.*

Proof. From the expression in the previous lemma,

$$f_{XY}(x,y) = Ce^{-1/2(x-\mu_X)^T\Sigma_X^{-1}(x-\mu_X)}e^{-1/2(y-\mu_Y-\Sigma_{YX}\Sigma_X^{-1}(x-\mu_X))^T S_{\Sigma_X}^{-1}(y-\mu_Y-\Sigma_{YX}\Sigma_X^{-1}(x-\mu_X))}.$$

Integrating this over y gives

$$f_X(x) = CC'e^{-1/2(x-\mu_X)^T\Sigma_X^{-1}(x-\mu_X)}.$$

We can do the same thing for $f_Y(y)$. □

4.1.3 Appendix: Matrix inversion using the Schur complement

Let $A \in \mathbb{R}^{p\times p}$ and $D \in \mathbb{R}^{q\times q}$. Consider block matrices of the form

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}.$$

Assuming that A is invertible, we can zero the block below A by left matrix multiplication:

$$\begin{pmatrix} I_p & \mathbf{0} \\ -CA^{-1} & I_q \end{pmatrix}\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} A & B \\ \mathbf{0} & D-CA^{-1}B \end{pmatrix}.$$

We can also zero the block to the right of A by right matrix multiplication:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}\begin{pmatrix} I_p & -A^{-1}B \\ \mathbf{0} & I_q \end{pmatrix} = \begin{pmatrix} A & \mathbf{0} \\ C & D-CA^{-1}B \end{pmatrix}.$$

The same matrices operating together gives us a block digaonal result:

$$\begin{pmatrix} I_p & \mathbf{0} \\ -CA^{-1} & I_q \end{pmatrix} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} I_p & -A^{-1}B \\ \mathbf{0} & I_q \end{pmatrix} = \begin{pmatrix} A & \mathbf{0} \\ \mathbf{0} & S_A \end{pmatrix},$$

where $S_A = D - CA^{-1}B$ is called the **Schur complement of A in M.**

Lemma 4.4. **** For all $X \in \mathbb{R}^{q\times p}$ and $Y \in \mathbb{R}^{p\times q}$,*

$$\begin{pmatrix} I_p & \mathbf{0} \\ X & I_q \end{pmatrix}^{-1} = \begin{pmatrix} I_p & \mathbf{0} \\ -X & I_q \end{pmatrix} \text{ and } \begin{pmatrix} I_p & Y \\ \mathbf{0} & I_q \end{pmatrix}^{-1} = \begin{pmatrix} I_p & -Y \\ \mathbf{0} & I_q \end{pmatrix}.$$

Lemma 4.5. **** If M and A are invertible, then S_A is invertible, and*

$$\begin{aligned} \begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} &= \begin{pmatrix} I_p & -A^{-1}B \\ \mathbf{0} & I_q \end{pmatrix} \begin{pmatrix} A^{-1}\mathbf{0} & \\ \mathbf{0} & S_A^{-1} \end{pmatrix} \begin{pmatrix} I_p & \mathbf{0} \\ -CA^{-1} & I_q \end{pmatrix} \\ &= \begin{pmatrix} A^{-1} + A^{-1}BS_A^{-1}CA^{-1} & -A^{-1}BS_A^{-1} \\ -S_A^{-1}CA^{-1} & S_A^{-1} \end{pmatrix}. \end{aligned}$$

*** If D is invertible, then we can say

$$\begin{pmatrix} I_p & -BD^{-1}\,\mathbf{0} & I_q \end{pmatrix} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} I_p & \mathbf{0} \\ -D^{-1}C & I_q \end{pmatrix} = \begin{pmatrix} S_D & \mathbf{0} \\ \mathbf{0} & D \end{pmatrix},$$

where $S_D = A - BD^{-1}C$ is the **Schur complement of D in A.**

Lemma 4.6. **** If M and D are invertible, then S_D is invertible, and*

$$\begin{aligned} \begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} &= \begin{pmatrix} I_p & \mathbf{0} \\ -D^{-1}C & I_q \end{pmatrix} \begin{pmatrix} S_D^{-1} & \mathbf{0} \\ \mathbf{0} & D^{-1} \end{pmatrix} \begin{pmatrix} I_p & -BD^{-1} \\ \mathbf{0} & I_q \end{pmatrix} \\ &= \begin{pmatrix} S_D^{-1} & -S_D^{-1}BD^{-1} \\ -D^{-1}CS_D^{-1} & D^{-1} + D^{-1}CS_D^{-1}BD^{-1} \end{pmatrix}. \end{aligned}$$

4.2 Maximum likelihood estimation

Let X be a nondegenerate Gaussian random vector with density

$$f_X(x) = \frac{1}{(2\pi)^{\frac{n}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}.$$

Σ is symmetric positive definite. If μ and Σ are unknown, we can try to learn these parameters via a set of training data $\{x_i \in \mathbb{R}^n\}_{i=1}^m$, where each example is drawn independently under $f_X(x)$. We can model this via a **likelihood function** with fixed data points and variables μ and Σ:

$$L(x_1, \ldots, x_m; \mu, \Sigma) = \prod_{i=1}^m f_X(x_i) = \prod_{i=1}^m \frac{1}{(2\pi)^{\frac{n}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)}.$$

With **maximum likelihood estimation (MLE)**, we aim to estimate μ and Σ by maximizing the likelihood function or, equivalently, the log-likelihood

$$\ln(L) = -\frac{1}{2}m \ln \det(\Sigma) - \frac{1}{2}\sum_{i=1}^{m}(x_i - \mu)^T \Sigma^{-1}(x_i - \mu) + C.$$

Thus, we want to solve

$$\begin{aligned} \min_{\mu \in \mathbb{R}^n,\ \Sigma \in \mathbb{R}^{n\times n}} \quad & J(\mu, \Sigma) = m \ln \det(\Sigma) + \sum_{i=1}^{m}(x_i - \mu)^T \Sigma^{-1}(x_i - \mu) \\ \text{s.t.} \quad & \Sigma \text{ is symmetric positive definite.} \end{aligned}$$

$$D_\mu J(\mu, \Sigma)(h) = \sum_{i=1}^{m} -h^T \Sigma^{-1}(x_i - \mu) - (x_i - \mu)^T \Sigma^{-1} h = -2\left(\sum_{i=1}^{m}(x_i - \mu)^T \Sigma^{-1}\right) h = 0$$

$$\implies \sum_{i=1}^{m}(x_i - \mu)^T \Sigma^{-1} = \mathbf{0}$$

$$\implies \hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x_i.$$

Now, to find the maximum likelihood estimate of Σ, we rewrite our problem, with the substitutions $z_i = x_i - \hat{\mu}$ and $S = \sum_{i=1}^{m} z_i z_i^T$. Note that $z_i^T \Sigma^{-1} z_i = \langle z_i, \Sigma^{-1} z_i \rangle = \mathrm{Tr}(z_i z_i^T \Sigma^{-1})$. Thus, $\sum_{i=1}^{m} z_i^T \Sigma^{-1} z_i = \sum_{i=1}^{m} \langle z_i z_i^T, \Sigma^{-1} \rangle = \mathrm{Tr}(S\Sigma^{-1})$.

$$\begin{aligned} \min_{\Sigma \in \mathbb{R}^{n\times n}} \quad & J(\Sigma) = m \ln \det(\Sigma) + \mathrm{Tr}(S\Sigma^{-1}) \\ \text{s.t.} \quad & \Sigma \text{ is symmetric positive definite.} \end{aligned}$$

Note that the symmetric matrices in $\mathbb{R}^{n\times n}$ form a subspace of dimension $\frac{(n+1)n}{2}$, and the symmetric positive semidefinite matrices form a closed subset C of this subspace. The symmetric positive definite matrices form the interior of C, so if we have a positive definite solution for Σ^*, then it lies in the interior of C.

Remark 4.7. It is important to carry arround the dummy variable H, especially now that the matrices might not commute.

Lemma 4.8. *For any invertible $M \in \mathbb{R}^{n\times n}$, let $g(M) = M^{-1}$. Then,*

$$Dg(M)(H) = -M^{-1} H M^{-1}.$$

Proof. $g(M)M = I_n$. Taking the derivative of both sides gives $Dg(M)(H)M = -g(M)H$. □

So, if we are at M and move a small step ε along H to $M + \varepsilon H$, then to first order in ε,

$$(M + \varepsilon H)^{-1} = M^{-1} + Dg(M)(\varepsilon H) = M^{-1} - \varepsilon M^{-1} H M^{-1}.$$

Lemma 4.9. *For any invertible $M \in \mathbb{R}^{n\times n}$, let $f(M) = det(M)$.*

$$Df(M)(H) = det(M)\, Tr(M^{-1}H).$$

Proof. Using the definition of the derivative,

$$\begin{aligned} Df(M)(H) &= \lim_{\varepsilon\to 0} \frac{f(M+\varepsilon H) - f(M)}{\varepsilon} \\ &= \lim_{\varepsilon\to 0} \frac{\det(M(I+\varepsilon M^{-1}H)) - \det(M)}{\varepsilon} \\ &= \det(M) \lim_{\varepsilon\to 0} \frac{\det(I+\varepsilon M^{-1}H) - 1}{\varepsilon} \\ &= \det(M) \lim_{\varepsilon\to 0} \frac{\det(I+\varepsilon M^{-1}H) - 1}{\varepsilon} = \det(M)\mathrm{Tr}(M^{-1}H), \end{aligned}$$

where $\det(I+\varepsilon A) = 1 + \varepsilon \mathrm{Tr}(A)$ as seen in the following lemma. □

If we are at M and move a small step ε along H to $M+\varepsilon H$, then to first order in ε,

$$\det(M+\varepsilon H) = \det(M) + Df(M)(\varepsilon H) = \det(M) + \varepsilon \det(M)\mathrm{Tr}(M^{-1}H).$$

Lemma 4.10. *For $A \in \mathbb{R}^{n\times n}$ and small $\varepsilon \in \mathbb{R}$, $|I+\varepsilon A| = 1 + \varepsilon\, Tr(A) + O(\varepsilon^2)$.*

Proof. *** □

Let's return to estimating Σ:

$$DJ(\Sigma)(H) = m\frac{1}{\det(\Sigma)}\det(\Sigma)\mathrm{Tr}(\Sigma^{-1}H) - \mathrm{Tr}(S\Sigma^{-1}H\Sigma^{-1}) = \mathrm{Tr}((m\Sigma^{-1} - \Sigma^{-1}S\Sigma^{-1})H) = 0$$

$$\implies \langle m\Sigma^{-1} - \Sigma^{-1}S\Sigma^{-1}, H\rangle = 0$$

$$\implies m\Sigma^{-1} - \Sigma^{-1}S\Sigma^{-1} = \mathbf{0}$$

$$\implies \hat{\Sigma} = \frac{1}{m}S = \frac{1}{m}\sum_{i=1}^{m}(x_i - \hat{mu})(x_i - \hat{mu})^T.$$

Theorem 4.11. *Let $\{x_i\}_{i=1}^m$ be independent samples from a nondegenerate multivariate Gaussian density with mean $\mu \in \mathbb{R}^n$ and covariance $\Sigma \in \mathbb{R}^{n\times n}$. The maximum likelihood estimates of μ and Σ are*

$$\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x_i,$$

$$\hat{\Sigma} = \frac{1}{m}\sum_{i=1}^{m}(x_i - \hat{\mu})(x_i - \hat{\mu})^T.$$

Remark 4.12. Let $B \in \mathbb{R}^{p\times n}$ be given, $w \in \mathbb{R}^n$ be fixed but unknown, and V be a Gaussian random vector with covariance Σ. Let $Y = Bw + V$. So Y is a Gaussian random vector with mean $\mu = Bw$, covariance Σ, and density

$$f_Y(y; w) = \frac{1}{(2\pi)^{\frac{p}{2}}}\frac{1}{|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(y-Bw)^T\Sigma^{-1}(y-Bw)}.$$

Let's say we want to estimate w. We can maximize the log-likelihood:

$$\begin{aligned}\hat{w} &= \arg\max_{w\in\mathbb{R}^n} -\frac{1}{2}(y-Bw)^T\Sigma^{-1}(y-Bw)\\ &= \arg\min_{w\in\mathbb{R}^n}(y-Bw)^T\Sigma^{-1}(y-Bw)\\ &= \arg\min_{w\in\mathbb{R}^n}\|y-Bw\|^2_{\Sigma^{-1}}\\ &= \arg\min_{w\in\mathbb{R}^n}\|y-Bw\|_2^2 \text{ if } \Sigma=\sigma^2 I_p.\end{aligned}$$

So if the noise covariance is known to be i.i.d, then the MLE of w given y can be found by solving a least squares problem.

Lemma 4.13. *For invertible $A\in\mathbb{R}^{n\times n}$ and any $H\in\mathbb{R}^{n\times n}$,*

$$(D_A|A|)H = |A|\,Tr(A^{-1}H).$$

Proof. *** □

4.3 Estimating the value of a random vector

Given a random vector X, how can we estimate the value of a related random vector Y?

4.3.1 Estimate the value of Y given its density $f_Y(y)$

Let Y be a random vector with density $f_Y(y)$. We want to infer the value assumed by Y with just the density. One option is to minimize the expected value of the squared error, $E[\|Y-\hat{y}\|_2^2]$, also called the **mean squared error (MSE)** cost.

$$E[\|Y-\hat{y}\|_2^2] = E[\|Y-\mu_Y+\mu_Y-\hat{y}\|_2^2] \le E[\|Y-\mu_Y\|_2^2]+\|\mu_Y-\hat{y}\|_2^2 = ***\mathrm{Tr}(\Sigma_Y)+\|\mu_Y-\hat{y}\|_2^2.$$

Lemma 4.14. *Assume that the components of Y have finite first and second order moments. Then, $\hat{y}=\mu_Y$ minimizes the MSE $E[(Y-\hat{y})^2]$.*

Proof. We want to set the gradient of $E[(Y-\hat{y})^2]$ with respect to $\hat{y}$ equal to 0. First, exchange the order of the gradient and expectation operators (***). Thus, $2E[-Y+\hat{y}]=0$, so the optimal selection is $\hat{y}=E[Y]=\mu_Y$. □

4.3.2 Estimate the value of Y given $f_{XY}(x,y)$ and $X=x$

We now have additional available information: let X and Y have a joint density $f_{XY}(x,y)$ Suppose that a realization of (X,Y) is determined, but we only know $X=x$. Assuming that X and Y are dependent, this information allows us to more accurately infer Y.

Lemma 4.15. *Assume that the conditional density $f_{Y|X}(y|x)$ has finite first and second order moments. Then, the optimal MSE estimate of the value of Y given that $X=x$ is the mean of the conditional density $f_{Y|X}(y|X)$:*

$$\hat{y}(x) = E[Y|X=x] = \int_{\mathbb{R}} y f_{Y|X}(y|x)dy = \mu_{Y|X}(x).$$

Proof. The conditional density adjusts the density of Y to fully exploit the information in the observation $X = x$. Any remaining uncertainty in the value of Y is described by $f_{Y|X}(y|x)$. By the previous lemma, the optimal MSE estimate of the value of Y is the mean of the conditional density. □

This estimator exists and is optimal for the MSE cost, but finding an expression for the conditional mean estimator is a challenge.

Conditional mean estimator for jointly Gaussian random vectors

Let $f_{XY}(x, y)$ be a Gaussian density; then, as previously shown, the conditional mean estimator is

$$\hat{y}(x) = \hat{\mu}_{Y|X} = \mu_Y + \Sigma_{YX}\Sigma_X^{-1}(x - \mu_X),$$

which is an affine function of x. This is optimal for jointly Gaussian X and Y and is a lower bound for performance.

What if the first and second order statistics of X and Y, Σ_X and Σ_{XY}, are not known? In place of these, we have training data $\{(x_i, y_i)\}_{i=1}^m$, and assume that $X_i \in \mathbb{R}^n$ and the simplifying assumption that $y_i \in \mathbb{R}$. Can we estimate the quantities in $\hat{y}(x)$? The MLE of μ_X is

$$\hat{\mu}_X = \frac{1}{m}\sum_{i=1}^m x_i.$$

Similarly, the MLE of μ_Y is

$$\hat{\mu}_Y = \frac{1}{m}\sum_{i=1}^m y_i.$$

The core problem is $\hat{y}(x) = Wx = w^T x$, where $W = \Sigma_{YX}\Sigma_X^{-1}$. We want to find $w \in \mathbb{R}^n$. Note that

$$w^T = \Sigma_{YX}\Sigma_X^{-1} \Longrightarrow w = \Sigma_X^{-1}\Sigma_{XY} \Longrightarrow \Sigma_X w = \Sigma_{XY}.$$

Let

$$A = \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix}, \; A^T = \begin{pmatrix} x_1 & \dots & x_m \end{pmatrix}, \; y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}.$$

So

$$\hat{\Sigma}_X = \frac{1}{m}A^T A * **,$$

$$\hat{\Sigma}_{XY} = \frac{1}{m}A^T y * * * .$$

Thus, we have

$$A^T A w = A^T y,$$

which are the normal equations of OLS! So all least squares is doing is assuming a jointly Gaussian distribution, knowing the conditional mean estimator, doing MLE based on the training data, and plugging the results into the conditional mean estimator to get w.

Using a fixed affine function to estimate Y given $X = x$
Now, we consider only estimators that are affine functions of x, i.e., $\hat{y} = Wx + b$ with $W \in \mathbb{R}^{q\times n}$ and $b \in \mathbb{R}^q$ that do not depend on x. We solve

$$\min_{W\in\mathbb{R}^{q\times n},\ b\in\mathbb{R}^q} E\left[\sum_{j=1}^{q}(Y_j - (WX)_j - b_j)^2\right] = E[\|Y - WX - b\|_2^2].$$

Let X have mean $\mu_X \in \mathbb{R}^n$ and covariance $\Sigma_X \in \mathbb{R}^{n\times n}$, Y have mean $\mu_Y \in \mathbb{R}^p$ and covariance $\Sigma_Y \in \mathbb{R}^{p\times p}$, and let the cross covariance of X and Y be $\Sigma_{XY} \in \mathbb{R}^{n\times p}$.

Theorem 4.16. *The minimum MSE affine estimator of Y given $X = x$ is*

$$\hat{y}(x) = W^*(x - \mu_X) + \mu_Y,$$

where $W^* = \Sigma_{YX}\Sigma_X^{-1}$ if Σ_X is invertible and is any solution of $W^*\Sigma_X = \Sigma_{YX}$ otherwise.

Proof. Assume for the moment that $\mu_X = \mathbf{0}$ and $\mu_Y = \mathbf{0}$. Note that

$$\begin{aligned} E[\|Y - WX - b\|_2^2] &= E[(Y - WX)^T(Y - WX)] + E[-2(Y - WX)^T b + b^T b] \\ &= E[(Y - WX)^T(Y - WX)] + b^T b. \end{aligned}$$

We can minimize the expression with $b = \mathbf{0}$. With some steps skipped, we can simplify the first term to be

$$\begin{aligned} E[(Y - WX)^T(Y - WX)] &= E[\mathrm{Tr}(YY^T - 2WXY^T + WXX^TW^T)] \\ &= \mathrm{Tr}(\Sigma_Y - 2W\Sigma_{XY} + W\Sigma_X W^T). \end{aligned}$$

The derivative of $\mathrm{Tr}(\Sigma_Y - 2W\Sigma_{XY} + W\Sigma_X W^T)$ with respect to W acting on $H \in \mathbb{R}^{n\times n}$ is

$$\begin{aligned} &\mathrm{Tr}(-2H\Sigma_{XY} + H\Sigma_X W^T + W\Sigma_X H^T) = 2\mathrm{Tr}(H\Sigma_X W^T - H\Sigma_{XY}) = 0 \\ &\implies \mathrm{Tr}(H^T(W\Sigma_X - \Sigma_{YX})) = \langle H, W\Sigma_X - \Sigma_{YX}\rangle = 0 \\ &\implies W\Sigma_X = \Sigma_{YX}. \end{aligned}$$

The second derivative with respect to W is $\mathrm{Tr}(H\Sigma_X H^T)$. We can choose H to be any $n \times n$ matrix, so if we replace it with H^T, we get the equivalent expression $\mathrm{Tr}(H^T\Sigma_X H)$. Σ_X is positive semidefinite, so $\mathrm{Tr}(H^T\Sigma_X H) \geq 0$. Thus, all solutions of $W\Sigma_X = \Sigma_{YX}$ minimize the MSE cost function.

If μ_X and μ_Y are not zero, then we can apply the reasoning above to predict $Y - \mu_Y$ given the value of $X - \mu_X$. The least MSE predictor of Y is thus $\hat{y}(x) = W^*(x - \mu_X) + \mu_Y$. □

4.3.3 Application to label prediction ***

Let's suppose we observe a data vector X that takes values in $\mathbb{R}^n$ and want to estimate a corresponding label vector Y that takes values in $\mathbb{R}^q$. Set $Z = \begin{pmatrix} X \\ Y \end{pmatrix}$, and assume that we know μ_Z and Σ_Z:

$$\mu_Z = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix},\ \Sigma_Z = \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix}.$$

We can use the minimum MSE affine estimator to infer the value of the label vector Y given that $X = x$:

$$\hat{y}(x) = \Sigma_{YX}\Sigma_X^{-1}(x - \mu_X) + \mu_Y = W^*(x - \mu_X) + \mu_Y,$$

or

$$\hat{y}(x) - \mu_Y = W^*(x - \mu_X).$$

If the joint density of the examples and labels is Gaussian, then this is the optimal MSE estimator.

Special case: Label denoising

Let's cover a special case. Let Y be a random label vector with mean μ_Y and covariance Σ_Y. We observe $X = Y + N$, where N is noise, and N and Y are independent and latent (unobserved) random vectors. Assume that the noise vector N has mean $\mathbf{0}$ and covariance Σ_N. To denoise the labels, we do

$$\mu_X = E[Y + N] = \mu_Y + E(N) = 0,$$

$$\Sigma_X = E[(Y - \mu_Y + N)(Y - \mu_Y + N)^T] = \Sigma_Y + \Sigma_N,$$

$$\Sigma_{XY} = E[(Y - \mu_Y + N)(Y - \mu_Y)^T] = \Sigma_Y.$$

So, the minimum MSE affine estimate of the value of Y given that $X = x$ is

$$\hat{y}(x) = \Sigma_Y(\Sigma_Y + \Sigma_N)^{-1}(x - \mu_X) + \mu_Y.$$

Now, assume the special case when $\Sigma_N = \sigma_N^2 I_q$. Let $\Sigma_Y = UDU^T$ be an eigendecomposition, where D is diagonal with entries $\sigma_1^2, \ldots, \sigma_q^2$.

Special case: Estimating a latent vector from linear, noisy observations

4.3.4 Appendix: Exchanging differentiation with expectation ***

5 Sparse regression

5.1 Sparse least squares

We have training data $\{(x_i, y_i)\}_{i=1}^m,\ x_i \in \mathbb{R}^n,\ y_i \in \mathbb{R}$. We compose

$$A = \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix} \in \mathbb{R}^{m \times n},\ y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{R}^m.$$

We solve the optimization problem

$$\min_{w \in \mathbb{R}^n} \quad \|y - Aw\|_2^2 + \lambda \|w\|_2^2,$$

with $w^* \in \mathbb{R}^n\ \forall\ w_i^* > 0$, generically. $x_i = \begin{pmatrix} x_i(1) & \dots & x_i(n) \end{pmatrix}^T$ represents the n features of the ith training example.

We want to force some of the features to have zero weight in order to select features that are truly informative and not just noise, thus finding a **sparse representation** of y. Basically, we want to incorporate into the model some measure of cost for selecting a feature i such that $\|y - Aw\|_2^2$ must be reduced enough to justify the cost. Let

$$\alpha \in \mathbb{R},\ |\alpha|_0 = \begin{cases} 1 \text{ if } |\alpha| > 0 \\ 0 \text{ otherwise.} \end{cases}$$

We could define

$$\|w\|_0 = 1^T [|w(i)|_0]$$

as the **sparsity** of w, where $\|w\|_0$ is the number of nonzero entries in w. Define the **support** of $w \in \mathbb{R}^n$ to be the set

$$S(w) = \{i : w(i) \neq 0\}.$$

Clearly, $|S(w) = \|w\|_0$. For an integer $k \geq 0$, we say that w is k**-sparse** if $\|w\|_0 \leq k$. More generally, a vector w is **sparse** if $\|w\|_0$ is relatively small. A new idea would be to solve

$$\begin{aligned} \min_{w \in \mathbb{R}^n} \quad & \|y - Aw\|^2 \\ \text{s.t.} \quad & \|w\|_0 \leq k. \end{aligned}$$

Thus, we want to choose the features that have the maximum benefit through reducing the value of the objective function the most, subject to a budget of a certain number of components that we can use.

We could also solve a second optimization problem

$$\begin{aligned} \min_{w \in \mathbb{R}^n} \quad & \|w\|_0 \\ \text{s.t.} \quad & \|y - Aw\|_2^2 \leq \varepsilon. \end{aligned}$$

There must be at least one such w such that this optimization problem is feasible, and we can then choose the best w.

A third optimization problem to solve could be

$$\min_{w \in \mathbb{R}^n} \quad \|y - Aw\|^2 + \lambda \|w\|_0$$

for some $\lambda > 0$. For smaller λ, there is a smaller incremental cost for making $\|w\|_0$ larger. But for larger λ, the problem incorporates larger costs for choosing more components. This process is sometimes referred to as **subset selection**.

By varying k or λ, we can make these optimization problems equivalent! Thus, let's focus our discussion on the third optimization problem for now. What is a major downside? This optimization problem faces hard decisions about which features to keep. Suppose there are two highly correlated features—this model would simply select one, whereas we might ideally like to average the two features or take a look at the covariance matrix. Perhaps PCA would help here by grouping highly correlated features. Another hybrid solution is **block sparsity**—grouping features together and having a groupwise version of subset selection (within the group, we want averaging, and between the groups, we want selection).

The fact that these problems are computationally challenging (NP-hard, actually, though some subproblems can be solved efficiently) has led to a range of **greedy algorithms**, which might be efficient but not optimal and typically operate on the first optimization problem above. Is there a convex relaxation of this problem that could be solved efficiently?

The obvious aspect of this problem that we want to address is to make it convex. First, why is it nonconvex? Note that $\|x\|_0 \leq 1$ is nonconvex. We can visualize this in $\mathbb{R}^2$: While $(0, 1)$ and $(1, 0)$ are in the 1-sublevel set (in fact, the entirety of the x-axis and y-axis are in the 1-sublevel set), other points along $y = 1 - x$ are not in the 1-sublevel set. Furthermore, scaling is not satisfied: $\|\alpha x\|_0 = |\alpha| \|x\|_0$ does not always hold.

Definition 5.1. (Sparse approximation).

$$\min_{x \in \mathbb{R}^n} \|y - x\|_2^2 + \lambda \|x\|_0, \ \lambda > 0.$$

This problem is interesting because it is a compression problem. It is easy to solve: we basically capture the largest k entries of y and, thus, it is a hard thresholding problem. We will analyze this problem in more detail later.

5.1.1 Sparse least squares problem

Let us focus on the third optimization problem above,

$$\min_{w \in \mathbb{R}^n} \|y - Aw\|_2^2 + \lambda \|w\|_0.$$

We will refer to the columns of A as **atoms**. Without loss of generality, we can assume that $y \in \mathcal{R}(A)$ and that y and the atoms of A have unit norm. We see why the first claim is true: let $y = \hat{y} + \tilde{y}$ with $\hat{y} \in \mathcal{R}(A)$ and $\tilde{y} \in \mathcal{R}(A)^\perp$. Then

$$\begin{aligned} \|y - Aw\|_2^2 + \lambda \|w\|_0 &= \|\tilde{y} + \hat{y} - Aw\|_2^2 + \lambda \|w\|_0 \\ &= \|\tilde{y}\|_2^2 + \|\hat{y} - Aw\|_2^2 + \lambda \|w\|_0 \\ &\equiv \|\hat{y} - Aw\|_2^2 + \lambda \|w\|_0. \end{aligned}$$

So solving the problem with $\hat{y}$ gives us a solution of the original problem. Conversely, if w^* is a solution of the original problem, then it is also a solution for the problem with $\hat{y}$, so WLOG, we assume $y \in \mathcal{R}(A)$.

Now, let's see why the second claim is true: if the atoms do not have unit norm, let $\tilde{A} = AD$, where the atoms of $\tilde{A}$ have unit norm and D is diagonal with positive diagonal entries. If $z = Dw$, then $\|w\|_0 = \|D^{-1}z\|_0 = \|z\|_0$. Then,

$$\begin{aligned}\|y - Aw\|_2^2 + \lambda\|w\|_0 &= \|y - \tilde{A}Dw\|_2^2 + \lambda\|w\|_0 \\ &= \|y - \tilde{A}z\|_2^2 + \lambda\|D^{-1}z\|_0 \\ &= \|y - \tilde{A}z\|_2^2 + \lambda\|z\|_0.\end{aligned}$$

Solving this sparse regression problem with $\tilde{A}$ to obtain z^* gets us $w^* = D^{-1}z^*$, the solution to the original problem. Conversely, solving w^* for the original problem gets us Dw^*, a solution for the problem using $\tilde{A}$. Thus, without loss of generality, we can assume that the atoms of A have unit norm.

We modify the objective function now, multiplying it by $c^2 > 0$ and setting $u = cy, z = cw, \tilde{\lambda} = c^2\lambda$. Since $\|cw\|_0 = \|w\|_0$, we have

$$\begin{aligned}c^2(\|y - Aw\|_2^2 + \lambda\|w\|_0) &= \|cy - Acw\|_2^2 + \lambda c^2\|cw\|_0 \\ &= \|u - Az\|_2^2 + \lambda c^2\|z\|_0 \\ &= \|u - Az\|_2^2 + \tilde{\lambda}\|z\|_0.\end{aligned}$$

Note that if z^* solves the modified problem, then $w^* = \frac{z^*}{c}$ solves the original problem. Conversely, if w^* solves the original problem, then $z^* = cw^*$ solves the modified problem. Choosing $c = \frac{1}{\|y\|_2}$ ensures that u has unit norm, so without loss of generality, we can assume that y has unit norm.

$\min_{w\in\mathbb{R}^n} \|y - Aw\|_2^2 + \lambda\|w\|_0$ is easily solved when $A \in \mathbb{R}^{n\times k},\ k \leq n$, has an SVD factorization $A = U\Sigma P^T$, where $U \in \mathcal{O}_{n\times k},\ \Sigma \in \mathbb{R}^{k\times k}$ is diagonal with a positive diagonal, and $P \in \mathbb{R}^{k\times k}$ is a generalized permutation matrix ***. Special cases of this include:

1. **Sparse approximation**: $\min_{x\in\mathbb{R}^n} \|y - x\|_2^2 + \lambda\|x\|_0$.
2. **Sparse weighted approximation**: For $D = \text{diag}(d)$ with $d \in \mathbb{R}^n$ and $d > 0$, $\min_{x\in\mathbb{R}^n} \|y - Dx\|_2^2 + \lambda\|x\|_0$.
3. **Sparse representation in an orthonormal basis**: For $Q \in \mathcal{O}_{n\times k}$, $\min_{x\in\mathbb{R}^k} \|y - Qx\|_2^2 + \lambda\|x\|_0$.

Problem (1) is an easily solved sparse approximation problem that we will analyze in the following section. The other special cases reduce to problem (1). Beyond these special cases, the general sparse least squares problem presents a computational bottleneck; we will discuss some greedy algorithms for efficiently finding an approximate solution.

5.1.2 k-sparse approximation

We consider the **sparse approximation problem** of finding a k-sparse vector $x \in \mathbb{R}^n$ that is "closest" to a given $y \in \mathbb{R}^n$, where $k < n$. We can state the problem as

$$\begin{aligned}\min_{x\in\mathbb{R}^n} \quad & f(y - w) \\ \text{s.t.} \quad & \|x\|_0 \leq k,\end{aligned}$$

where $f(z) = g(\|z\|)$ for some norm $\|\cdot\|$ on $\mathbb{R}^n$ and g is a strictly monotone increasing function $g : \mathbb{R}_+ \to \mathbb{R}_+$ with $g(0) = 0$.

Example 5.2. Some example functions f include:

- $f(z) = \|z\|_1 = \sum_{j=1}^n |z(j)|$.
- $f(z) = (\|z\|_2)^2 = \sum_{j=1}^n |z(j)|^2$.
- $f(z) = (\|z\|_p)^p = \sum_{j=1}^n |z(j)|^p$.
- $f(z) = \|z\|_\infty = \max_j\{|z(j)|\}$.
- $f(x) = x^T P x$, where $P \in \mathbb{R}^{n\times n}$ is symmetric positive definite.

We can also bring in a parameter $\lambda > 0$ and add a penalty term $\lambda\|x\|_0$ to the objective function to form the unconstrained problem

$$\min_{x\in\mathbb{R}^n} f(y - x) + \lambda\|x\|_0.$$

$f(y - x)$ and $\lambda\|x\|_0$ are competing terms. Selecting a small value for λ encourages less sparsity and a better match between x and y, while selecting a large value for λ encourages higher sparsity but is potentially a worse match between x and y.

Separable objective function
We can easily solve

$$\min_{x\in\mathbb{R}^n} f(y - x) + \lambda\|x\|_0$$

when $f(z)$ is a separable function. To be specific, assume that $f(z) = \sum_{j=1}^n h_j(|z(j)|)$ with functions $h_j : \mathbb{R}_+ \to \mathbb{R}_+$ that are strictly monotone increasing functions with $h_j(0) = 0$. In this case, we would minimize

$$\sum_{j=1}^{n}(h_j(|y(j) - x(j)|) + \lambda|x(j)|_0),$$

which is the sum of n subproblems that can each be solved independently. Subproblem j can be expressed as

$$\min_\alpha(h_j(|y(j) - \alpha|) + \lambda|\alpha|_0).$$

We can consider cases:

1. $y(j) = 0$: Set $\alpha = 0$.
2. $y(j) \neq 0$: Set $\alpha = y(j)$ and incur a cost λ, or set $\alpha = 0$ and incur a cost $h_j(|y(j)|)$.

Theorem 5.3. *Assume that $f(z) = \sum_{j=1}^n h_j(|z(j)|)$, where the functions $h_j : \mathbb{R}_+ \to \mathbb{R}_+$ are strictly monotone increasing functions with $h_j(0) = 0$. Then the solution of $\min_{x\in\mathbb{R}^n} f(y - x) + \lambda\|x\|_0$ is*

$$x^*(j) = \begin{cases} y(j) & \text{if } h_j(|y(j)|) \geq \lambda \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for separable functions f, the solution to the sparse approximation problem is obtained by **hard thresholding** $y(j)$ based on a comparison of $h_j(|y(j)|)$ and λ. Since $h_j(\cdot)$ is strictly monotone increasing, we can also write the thresholding condition as $|y(j)| \geq t_j(\lambda) = h_j^{-1}(\lambda)$. Using the generic scalar **hard thresholding operator** defined for $z \in \mathbb{R}$ by

$$H_t(z) = \begin{cases} z & \text{if } |z| \geq t \\ 0 & \text{otherwise,} \end{cases}$$

we can write

$$x^* = \left[H_{t_j(\lambda)}(y(j))\right] = \begin{bmatrix} y(j) & \text{if } |y(j)| \geq t_j(\lambda) \\ 0 & \text{otherwise} \end{bmatrix}.$$

Example 5.4. Now consider the special case: $\min_{x\in\mathbb{R}^n} \|y - x\|_2^2 + \lambda\|x\|_0$. For this problem, the appropriate hard threshold is $t = \sqrt{\lambda}$. So

$$x^* = \left[H_{\sqrt{\lambda}}(y)\right] = \begin{bmatrix} y(j) & \text{if } |y(j)| \geq \sqrt{\lambda} \\ 0 & \text{otherwise} \end{bmatrix}.$$

Theorem 5.5. *Let S be the indices of any set of k largest values of $h_j(|y(j)|)$. Then x^* defined by*

$$x^*(j) = \begin{cases} y(j) & \text{if } j \in S \\ 0 & \text{otherwise.} \end{cases}$$

is a solution to the sparse approximation problem. This solution is unique if and only if the k-th largest value of $h_j(|y(j)|)$ is strictly larger than the $(k+1)$-st value.

Lemma 5.6. *For symmetric $P \in \mathbb{R}^{n\times n}$, the function $f(x) = x^T Px$ is separable if and only if P is diagonal.*

Proof. *** □

Non-separable objective function

As an example, let's consider k-sparse approximation under the non-separable max-norm:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \|y - x\|_\infty \\ \text{s.t.} \quad & \|x\|_0 \leq k, \end{aligned}$$

Assume, for simplicity, that the entries of y are nonzero with distinct absolute values and that the entries are arranged from largest to smallest by absolute value. So $\|y\|_\infty = |y(1)| > |y(2)| > \cdots > |y(n)| > 0$. We can use up to k nonzero elements in x to minimize $\|y-x\|_\infty$. The optimal allocation is to use $x(1), \ldots, x(k)$ to make $|y(j)-x(j)|$ no larger than $|y(k+1)|, j = 1, \ldots, k$. This yields $\|y - x^*\|_\infty = |y(k+1)|$, the smallest achievable value of $\|y - x\|_\infty$ with a k-sparse x. Any x^* satisfying $|y(j) - x^*(j)| \leq |y(k+1)|, j = 1, \ldots, k$, is an optimal solution. Thus, in general, this problem does not have a unique solution.

The general solution, without the simplifying assumptions, can be obtained by finding the k largest values of $|y(j)|$ and setting the corresponding elements of x^* with the same indices to be zero. Note that the solution for the separable objective function and the solution for the non-separable objective function look remarkably similar, suggesting that separability is not critical in defining a solution.

5.1.3 Greedy solution methods for sparse least squares

Recall that the support set of $x \in \mathbb{R}^n$ is the set $S(x) = \{i : x(i) \neq 0\}$. $S(x) = \|x\|_0$. Greedy solution methods generally operate by iteratively alternating between:

- Adding an atom (or atoms) to the support set of the estimated solution w^*.
- Updating the weights assigned to atoms indexed by this support set.

Let $c = 1, 2, 3, \ldots$ count the iterations used, $\hat{S}$ denote the indices of the currently selected atoms, $\hat{y}$ denote the associated sparse approximation to y, and $r = y - \hat{y}$ denote the associated residual.

Matching pursuit

The **matching pursuit (MP) algorithm** iteratively either adds one new atom to the estimated support set and assigns a new weight to this atom or updates the weight of an atom already in the estimated support set. The steps are outlined below:

1. Initialize the iteration count $c = 0$, the set of selected atoms $\hat{S} = \emptyset$, the sparse approximation $\hat{y} = \mathbf{0}$, and the residual $r = y$.
2. (a) Update the iteration count: $c = c + 1$.
   (b) Select an atom a_{p_c} with index $p_c \in \arg\max_i |a_i^T r|$. (Intuitively, we are picking the atom most aligned with the residual, i.e., the atom that can best explain the residual.)
   (c) Update the indices of the selected atoms: $\hat{S} = \hat{S} \cup \{p_c\}$.
   (d) Set $w(p_c) = a_{p_c}^T r$.
   (e) The projection of r onto a_{p_c} is $\hat{r} = w(p_c) a_{p_c}$.
   (f) Update the sparse approximation $\hat{y} = \hat{y} + w(p_c) a_{p_c}$ and the residual $r = y - \hat{y}$.
3. Check if a termination condition is satisfied: construction terminates after a desired number of distinct atoms have been selected or the size of the residual falls below some threshold. Upon termination, we have a list of the indices of the selected atoms $a_{p_1}, \ldots, a_{p_k}$ with weights $w(p_1), \ldots, w(p_k)$. These give the sparse approximation $\hat{y} = \sum_{i=1}^{k} w(p_i) a_{p_i}$ to y with residual $r = y - \hat{y}$. If a termination condition is not satisfied, go to step (2).

Orthogonal matching pursuit

The **orthogonal matching pursuit (OMP) algorithm** is similar to MP except that at each iteration, the weights are updated jointly by orthogonal projection of y onto the span of the atoms selected so far. Let $A_{\hat{S}}$ denote the matrix consisting of the columns of A with indices in $\hat{S}$. The steps are outlined below:

1. Initialize the iteration count $c = 0$, the set of selected atoms $\hat{S} = \emptyset$, and the residual $r = y$.
2. (a) Update the iteration count: $c = c + 1$.

(b) Select an atom a_{p_c} with index $p_c \in \arg\max_i |a_i^T r|$.

(c) Update the indices of the selected atoms: $\hat{S} = \hat{S} \cup \{p_c\}$.

(d) Determine the orthogonal projection $\hat{y}$ of y onto the range of $A_{\hat{S}}$. This step requires solving a least squares problem with one additional column than at the previous iteration.

(e) Update the residual $r = y - \hat{y}$.

After this step is complete, $r \perp \text{span}(A_{\hat{S}})$. So once an atom has been selected, it cannot be selected a second time.

3. Check if a termination condition is satisfied: construction terminates after a desired number of distinct atoms have been selected, the size of the residual falls below some threshold, or no atom can be found that has a nonzero correlation with the residual. Upon termination, we have a list of the indices of the selected atoms $a_{p_1}, \ldots, a_{p_k}$ with weights $w(p_1), \ldots, w(p_k)$. These give the sparse approximation $\hat{y} = \sum_{i=1}^{k} w(p_i) a_{p_i}$ to y with residual $r = y - \hat{y}$. If a termination condition is not satisfied, go to step (2).

5.1.4 Appendix: Sparse approximation under a symmetric norm ***

A norm on $\mathbb{R}^n$ is **symmetric** if it has the following two properties:

1. For any permutation matrix P and for all $x \in \mathbb{R}^n$, $\|Px\| = \|x\|$.
2. For any diagonal matrix D with diagonal entries in $\{+1, -1\}$, and for all $x \in \mathbb{R}^n$, $\|Dx\| = \|x\|$.

A norm satisfying the first property is called a **permutation invariant norm**, and a norm satisfying the second property is called an **absolute norm**. A square matrix of the form DP, where $D = \text{diag}[+1, -1]$ and P is a permutation matrix, is called a **generalized permutation**. Let $\text{GP}(n)$ denote the set of $n \times n$ generalized permutation matrices. Then a norm is symmetric if and only if it is invariant under all generalized permutations.

Lemma 5.7. *$GP(n)$ is closed under matrix multiplication, contains the identiy, and every $P \in GP(n)$ has an inverse $P^{-1} \in GP(n)$.*

Example 5.8. The following are some examples of symmetric norms. $Q \in \text{GP}(n)$.

- Every p-norm.
- The max-norm.
- The c-norm defined by $\|x\|_c = \max_{P \in \text{GP}(n)} \{x^T P x\}$, where $c \in \mathbb{R}^n$ is nonzero.

Lemma 5.9. *For any symmetric norm $\|\cdot\|$ on $\mathbb{R}^n$, if $\mathbf{0} \leq x \leq y$, then $\|x\| \leq \|y\|$. (We use the notation $\mathbf{0}$ meaning if $x > \mathbf{0}$, then the vector $x \in \mathbb{R}^n$ has all positive entries.)*

Theorem 5.10. *Let $\|\cdot\|$ be a symmetric norm, $y \in \mathbb{R}^n$, and S be the indices of the k largest values of $|y(j)|, j = 1, \ldots, n$. Then*

$$x^*(j) = \begin{cases} y(j) & \text{if } j \in S \\ 0 & \text{otherwise} \end{cases}$$

gives a solution to

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \|y-x\| \\ s.t. \quad & \|x\|_0 \le k, \end{aligned}$$

5.2 Convex relaxation: the LASSO

An alternative approach to solving

$$\min_{w\in\mathbb{R}^n} \|y-Aw\|_2^2 + \lambda\|w\|_0$$

is solving the **ℓ_1-regularized last squares problem**, also known as the **LASSO problem**:

$$\min_{w\in\mathbb{R}^n} \|y-Aw\|_2^2 + \lambda\|w\|_1.$$

The objective function is the sum of two competing convex terms, $\|Aw-b\|_2^2$ and $\lambda\|w\|_1$, and, thus, is convex. Equivalently, we can consider

$$\begin{aligned} \min_{w\in\mathbb{R}^n} \quad & \|y-Aw\|_2^2 \\ \text{s.t.} \quad & \|w\|_1 \le \varepsilon, \end{aligned}$$

and

$$\begin{aligned} \min_{w\in\mathbb{R}^n} \quad & \|w\|_1 \\ \text{s.t.} \quad & \|y-Aw\|_2^2 \le \delta. \end{aligned}$$

In this second optimization problem, we minimize $\|y-Aw\|_2^2$ over a fixed sublevel set of the ℓ_1-norm. The optimal solution occurs at the point w_ε^* where a level set of $\|y-Aw\|_2^2$ first intersects the ε-ball of $\|\cdot\|_1$.

In this third optimization problem, we minimize $\|w\|_1$ over a fixed sublevel set of $\|y-Aw\|_2^2$. The optimal solution occurs at the point w_δ^* where a level set of $\|\cdot\|_1$ first intersects the δ-sublevel set of $\|y-Aw\|_2^2$. For the appropriate choice of δ and ε, the second and third problems have the same solution.

Note that if we multiply the objective function of the first optimization problem by $\alpha^2 > 0$, we get

$$\min_{w\in\mathbb{R}^n} \|\tilde{y}-\tilde{A}w\|_2^2 + \tilde{\lambda}\|w\|_1,$$

where $\tilde{y} = \alpha y$, $\tilde{A} = \alpha A$, and $\tilde{\lambda} = \alpha^2\lambda$. Thus, to account for possible scaling, we can use the ratio

$$\frac{\lambda}{\lambda_{\max}} = \frac{\lambda}{\max_{j=1}^m |a_j^T y|}.$$

No closed-form solutions for these optimization problems are known; however, since they are convex optimization problems, they are amenable to solution via efficient numerical algorithms.

5.2.1 ℓ_1-sparse approximation and soft thresholding

A special case of the LASSO problem, $\min_{w\in\mathbb{R}^n} \|y - Aw\|_2^2 + \lambda\|w\|_1$, is the **$\ell_1$-sparse approximation problem**:

$$x^* = \arg\min_{x\in\mathbb{R}^n} \|y - x\|_2^2 + \lambda\|x\|_1.$$

The objective function is convex and separable:

$$\|y - x\|_2^2 + \lambda\|x\|_1 = \sum_{j=1}^{n}[(y(j) - x(j))^2 + \lambda|x(j)|].$$

We can optimize each term separately, thus solving n scalar problems of the form

$$\min_{z\in\mathbb{R}}[(y(j) - z)^2 + \lambda|z|].$$

The optimal solution of each scalar problem is

$$x^*(j) = \begin{cases} y(j) - \lambda/2 & \text{if } y(j) > \lambda/2 \\ 0 & \text{if } -\lambda/2 \le y(j) \le \lambda/2 \\ y(j) + \lambda/2 & \text{if } y(j) < -\lambda/2. \end{cases}$$

This operation is referred to as **shrinkage** since the component $x^*(j)$ is set to 0 if $y(j)$ is smaller in magnitude than $\lambda/2$, introducing sparsity in x^*, and the remaining nonzero components of x^* are formed by reducing the corresponding y values in magnitude by $\lambda/2$.

Using the generic scalar **soft thresholding operator** defined for $z \in \mathbb{R}$ by

$$S_t(z) = \begin{cases} z - t & \text{if } z \ge t \\ 0 & \text{if } -t < z < t \\ z + t & \text{if } z \le -t, \end{cases}$$

we can write

$$x^* = \left[S_{\lambda/2}(y)\right].$$

More generally, we can use the same approach to solve

$$\min_{x\in\mathbb{R}^n} \|y - x\|_2^2 + \lambda\|Dx\|_1,$$

where $D \in \mathbb{R}^{n\times n}$ is diagonal with all positive diagonal entries. The solution is again a soft thresholding operation on y, with a threshold for component j of $\lambda d(j)/2$. We can generalize even further to solve any problem of the form

$$\min_{x\in\mathbb{R}^n} \|y - UDP^Tx\|_2^2 + \lambda\|x\|_1,$$

where $U \in \mathcal{O}_{n\times k}$ has orthonormal columns, $D \in \mathbb{R}^{k\times k}$ is diagonal with positive diagonal entries, and $P \in \mathbb{R}^{k\times k}$ is a generalized permutation matrix.

Remark 5.11. The sparse approximation problems

$$\min_{x\in\mathbb{R}^n} \|y-x\|_2^2 + \lambda\|x\|_0 \text{ and } \min_{x\in\mathbb{R}^n} \|y-x\|_2^2 + \lambda\|x\|_1$$

have the solutions

$$x_0^* = [H_{\sqrt{\lambda}}(y)] \text{ and } x_1^* = [S_{\lambda/2}(y)].$$

The two solutions use similar but distinct thresholding functions (hard vs. soft) and distinct threshold values ($\sqrt{\lambda}$ vs. $\lambda/2$). In both solutions, components of y with magnitudes above the threshold will remain nonzero, else be set to zero. Thus, the higher the threshold, the sparser the solution.

5.2.2 Subgradients and the subdifferential

We consider a scalar valued differentiable function $f:\mathbb{R}^n\to\mathbb{R}$. The Derivative of f at x is a linear function from $\mathbb{R}^n$ into $\mathbb{R}$, so we can represent it by a row vector g_x^T, where $g_x\in\mathbb{R}^n$ is the gradient $\nabla f(x)$.

Recall when we discussed differentiable convex functions that we can write

$$f(z)\geq f(x)+g_x^T(z-x).$$

A vector $g\in\mathbb{R}^n$ is called a **subgradient** of f at x if for all $z\in\mathbb{R}^n$,

$$f(z)\geq f(x)+g^T(z-x).$$

The function f is **subdifferentiable** at x if it has a nonempty set of subgradients. The set of subgradients, if nonempty, is denoted by $\partial f(x)$ and is called the **subdifferential** of f at x.

Example 5.12.

$$g\in\partial\|w\|_1 \iff g(j)=\begin{cases}1 & \text{if } w(j)>0\\ \gamma\in[-1,1] & \text{if } w(j)=0\\ -1 & \text{if } w(j)<0.\end{cases}$$

Lemma 5.13. *The point w^* minimizes the convex function $f:\mathbb{R}^n\to\mathbb{R}$ if and only if $\mathbf{0}\in\partial f(w^*)$.*

Proof. (1) If w^* minimizes f, then for every z, $f(z)\geq f(w^*)=f(w^*)+\mathbf{0}^T(z-w^*)$. So $\mathbf{0}\in\partial f(w^*)$. (2) Conversely, if $\mathbf{0}\in\partial f(w^*)$, then for all z, $f(z)\geq f(w^*)+\mathbf{0}(z-w^*)=f(w^*)$. Thus, w^* minimizes f. □

5.2.3 Application to ℓ_1-regularized least squares

Note that the subdifferential of $f(w)=\|y-Aw\|_2^2+\lambda\|w\|_1$ is

$$\partial f(w)=2A^T(Aw-y)+\lambda\partial\|w\|_1.$$

Theorem 5.14. *w^* is a solution of $\min_{w\in\mathbb{R}^n}\|y-Aw\|_2^2+\lambda\|w\|_1$ if and only if*

$$A^T(y-Aw^*)=\frac{\lambda}{2}g \text{ for some } g\in\partial\|w^*\|_1.$$

Note that $y - Aw^*$ is the residual term and that the rows of A^T (columns of A) are the atoms.

$$a_j^T(y - Aw^*) = \begin{cases} \lambda/2 & \text{if } w^*(j) > 0 \\ \gamma \in [-\lambda/2, \lambda/2] & \text{if } w^*(j) = 0 \\ -\lambda/2 & \text{if } w^*(j) < 0. \end{cases}$$

Note that the residual is never orthogonal to an atom used by w^*.

We have previously seen corresponding conditions for least squares and ridge regression. For least squares, the condition is $A^T(y - Aw) = 0$ (the normal equations), or for any atom a_i, $a_i^T(y - Aw^*) = 0$. For ridge, the condition is $A^T(y - Aw^*) = \lambda w^*$, or for any atom a_i, $a_i^T(y - A^T w^*) = \lambda w^*(i)$.

Example 5.15. Applying the necessary and sufficient conditions from the prevceding theorem to the ℓ_1-sparse approximation problem to obtain the known solution gives

$$y - x^* \in \frac{\lambda}{2} \partial \|x^*\|_1.$$

Componentwise, we get

$$x^*(i) = y(i) - \begin{cases} \frac{\lambda}{2} & \text{if } x^*(i) > 0 \ \equiv \ y(i) > \frac{\lambda}{2} \\ \gamma \in \left[-\frac{\lambda}{2}, \frac{\lambda}{2}\right] & \text{if } x^*(i) = 0 \ \equiv \ y(i) \in \left[-\frac{\lambda}{2}, \frac{\lambda}{2}\right] \text{ with } \gamma = y(i) \\ -\frac{\lambda}{2} & \text{if } x^*(i) < 0 \ \equiv \ y(i) < -\frac{\lambda}{2}. \end{cases}$$

5.2.4 Appendix: Related problems and concepts ***

6 Convex optimization ***

6.1 Convex programs

6.1.1 Linear programs

6.1.2 Quadratic programs

6.2 The Lagrangian and the dual problem

6.3 Weak duality, strong duality, and Slater's condition

6.4 Complementary slackness

6.5 The KKT conditions

7 The linear support vector machine ***

7.1 A simple linear SVM

7.2 The linear SVM for general training data

7.2.1 The primal linear SVM problem

7.2.2 The dual linear SVM problem

7.3 Appendix 1: The ν-SVM

7.4 Appendix 2: One class SVMs

8 Feature maps and kernels

While the linear SVM learns a hyperplane classification boundary in the ambient space $\mathbb{R}^n$ of the examples, the general SVM learns a nonlinear classification boundary in the same ambient space. Conceptually, we can think of this process as having two steps:

1. Finding a nonlinear function ϕ to map the training document into a higher dimensional space equipped with an inner product. (Nonlinearity will be critical to the feature map.)

2. Using the linear SVM to learn a classification hyperplane boundary in the higher dimensional space.

We will discuss feature maps and the concept of kernels of feature maps.

8.1 Introduction to feature maps and kernels

We aim to find a reasonably computationally feasible solution for the general SVM. The rough idea is to learn a *feature map* or *embedding* ϕ that maps each classification observation $x \in \mathbb{R}^n$ to $\phi(x) \in \mathcal{H}$, a higher dimensional (Hilbert) space $\mathcal{H}$.

8.1.1 From feature maps to kernels

Definition 8.1. (Feature map). Let the set $\mathcal{X}$ be where the examples of interest are drawn. A feature map is a nonlinear function from the space of examples $\mathcal{X}$ to a feature space, such as a Hilbert space $\mathcal{H}$ in the infinite dimensional function space case.

$$\phi : \mathcal{X} \to \mathcal{H}.$$

More concretely, let us consider a set $\{(x_i, y_i)\}_{i=1}^m$ of binary labelled training data, with $x_i \in \mathbb{R}^m,\ i = 1, \ldots, m$. We could directly use this training data to learn a linear classifier, but we could also learn a linear classifier using the mapped training data, $\{(\phi(x_i), y_i)\}_{i=1}^m$, using a feature map $\phi : \mathbb{R}^n \to \mathbb{R}^q$, where q is the dimension of $\mathcal{H}$. We could then classify a new test point $x \in \mathbb{R}^n$ using its image $\phi(x) \in \mathbb{R}^q$:

$$\hat{y}(\phi(x)) = \begin{cases} 1 & \text{if } w^{*T}\phi(x) + b^* \geq 0, \\ -1 & \text{otherwise.} \end{cases}$$

In general, a feature map is a very powerful tool: the feature map can warp and fold the original space to bring examples with the same label closer together (reducing within class variation) while moving examples with distinct labels further apart (increasing between class distances). In order to do this, the data needs to be moved to a higher dimensional space, so that the map can have the necessary degrees of freedom to warp and fold the data surface. We, of course, need to consider the feasibility of these computations, especially since n can be large.

8.1.2 Kernels

Definition 8.2. (Kernel). Given a feature map $\phi : \mathcal{X} \to \mathbb{R}^q$ on $\mathbb{R}^q$, we have an inner product. For $x, z \in \mathcal{X}$, the kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ of ϕ is

$$k_\phi(x, z) = \langle \phi(x), \phi(z) \rangle.$$

Note that, by Cauchy-Schwarz, we have

$$k_\phi(x, z)^2 = (\langle \phi(x), \phi(z) \rangle)^2 \leq \|\phi(x)\|^2 \|\phi(z)\|^2 = k_\phi(x, x) k_\phi(z, z)$$

$$\implies -1 \leq \frac{\langle \phi(x), \phi(z) \rangle}{\frac{\|\phi(x)\|}{\sqrt{k_\phi(x,x)}} \frac{\|\phi(z)\|}{\sqrt{k_\phi(z,z)}}} \leq 1$$

$$\implies -1 \leq \frac{\langle \phi(x), \phi(z) \rangle}{\sqrt{\|\phi(x)\| \|\phi(z)\|}} \leq 1.$$

Thus, the kernel is a similarity measure on $\mathcal{X} \times \mathcal{X}$. The kernel also lets us directly evaluate inner products of pairs of mapped examples $\phi(x), \phi(z) \in \mathcal{H}$ without first computing the feature mappings of x and z.

Example 8.3. The following are some examples of kernels.

- For $x = (x_1, x_2)^T, z = (z_1, z_2)^T \in \mathbb{R}^2$, let $\phi(x) = x_1 x_2 \in \mathbb{R}$. Then, $k_\phi(x, z) = x_1 x_2 z_1 z_2$.
- For $x = (x_1, x_2)^T \in \mathbb{R}^2$, let $\phi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2)^T \in \mathbb{R}^3$. Why is this a good kernel choice? Note that

  $$k_\phi(x, z) = \langle \phi(x), \phi(z) \rangle = x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 = (x_1 z_1 + x_2 z_2)^2 = (\langle x, z \rangle)^2.$$

  How pretty!
- For $x \in \mathbb{R}$, let $\phi(x) = x$. $k_\phi(x, z) = xz$.
- For $x \in \mathbb{R}$, let $\phi(x) = (x/\sqrt{2}, x/\sqrt{2})^T$. $k_\phi(x, z) = (xz/2) + (xz/2) = xz$.

Thus, different feature maps can have the same kernel (see the last two examples), so perhaps kernels are the more fundamental factor at play here, capturing the geometry of the inner product space, and not necessarily the feature maps.

8.1.3 Properties of kernels

What properties must a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ have in order to be the kernel of some feature map?

Definition 8.4. (Gram matrix). The Gram matrix of a kernel k on the set of points $\{x_i\}_{i=1}^m$ is the matrix $K \in \mathbb{R}^{m \times m}$ with $K = [\langle \phi(x_i), \phi(x_j) \rangle]$. K is always a real symmetric positive semidefinite matrix. We show that it is positive semidefinite with the following: let $u \in \mathbb{R}^m$.

$$u^T K u = \sum_{i=1}^m \sum_{j=1}^m u(i) u(j) \langle \phi(x_i), \phi(x_j) \rangle = \left\langle \sum_{i=1}^m u(i) \phi(x_i), \sum_{j=1}^m u(j) \phi(x_j) \right\rangle \geq 0.$$

So if we want to pick a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that k is the kernel of some embedding, then for every integer $m \geq 1$ and every set of m points $\{x_j \in \mathcal{X}\}_{j=1}^m$, the matrix $K = [k(x_i, x_j)] \in \mathbb{R}^{m \times m}$ must be symmetric positive semidefinite.

Definition 8.5. (Positive semidefinite (PSD) kernel). A kernel function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ satisfying the condition that for every integer $m \geq 1$ and every set of m points $\{x_j \in \mathcal{X}\}_{j=1}^m$, the matrix $K = [k(x_i, x_j)] \in \mathbb{R}^{m \times m}$ is symmetric positive semidefinite. This is a sufficient condition for k to be the kernel of some feature map—see the following theorem.

Theorem 8.6. *A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the kernel of some feature map $\phi : \mathcal{X} \to \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space, if and only if for every integer $m \geq 1$ and every set of m points $\{x_j \in \mathcal{X}\}_{j=1}^m$, the matrix $K = [k(x_i, x_j)]$ is symmetric positive semidefinite.*

Proof. We proved necessity. The proof of sufficiency involves the construction of a reproducing kernel Hilbert space $\mathcal{H}$ of functions with reproducing kernel $k(\cdot, \cdot)$. The required feature map is then $\phi(x_i) = k(x_i, \cdot) \in \mathcal{H}$. □

Theorem 8.7. *(Schur product theorem). If $A, B \in \mathbb{R}^{n \times n}$ are symmetric positive semidefinite, then so is $A \otimes B$. Moreover, if A, B are symmetric positive definite, then so is $A \otimes B$.*

Proof. *** □

Theorem 8.8. *Additional properties of PSD kernels:*

- *If $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a PSD kernel and $\phi : \mathcal{Y} \to \mathcal{X}$, then $k'(y, z) = k(\phi(y), \phi(z))$ is a PSD kernel on $\mathcal{Y} \times \mathcal{Y}$.*
- *If k is a PSD kernel on $\mathcal{X} \times \mathcal{X}$, and $\alpha > 0$, then αk is also a PSD kernel.*
- *If k_1 and k_2 are PSD kernels on $\mathcal{X} \times \mathcal{X}$, then so is $k_1 + k_2$.*
- *If $f : \mathcal{X} \to \mathbb{R}$, then $k(x, z) = f(x)f(z)$ is a PSD kernel on $\mathcal{X} \times \mathcal{X}$.*
- *If k_1 and k_2 are PSD kernels on $\mathcal{X} \times \mathcal{X}$, then so is $k_1 k_2$.*
- *If $\{k_p\}_{p \geq 1}$ is a sequence of PSD kernels on $\mathcal{X} \to \mathcal{X}$ and $\lim_{p \to \infty} k_p(x, z) = k(x, z)$, then $k(x, z)$ is a PSD kernel on $\mathcal{X} \times \mathcal{X}$.*

Proof. Let $m \geq 1$ be an integer and $\{x_j\}_{j=1}^m \subset \mathcal{X}$.

- For any $m \geq 1$ and set of points $\{y_j\}_{j=1}^m \in \mathcal{Y}$, $\{\phi(y_j)\}_{j=1}^m$ constitutes m points in $\mathcal{X}$. Thus, since $K = [k(\phi(y_i), \phi(y_i))]$ is symmetric PSD, $K' = [k'(y_i, y_j)] = K$ is symmetric PSD.
- $K = [\alpha k(x_i, x_j)] = \alpha[k(x_i, x_j)]$ is symmetric PSD.
- $K = [k_1(x_i, x_j)] + [k_2(x_i, x_j)]$ is symmetric PSD.
- Let $K = [f(x_i)f(x_j)]$ and $h = \begin{pmatrix} f(x_1) & \dots & f(x_m) \end{pmatrix}^T$. Then, $K = hh^T$, which is symmetric PSD.
- Let $K = [k_1(x_i, x_j)k_2(x_i, x_j)] = [k(x_i, x_j)] \otimes [k_2(x_i, x_j)]$. The result follows by Schur product theorem.
- $K_p = [k_p(x_i, x_j)]$ is symmetric positive semidefinite. Then, $\lim_{p \to \infty} K_p = [\lim_{p \to \infty} k_p(x_i, x_j)] = [k(x_i, x_j)]$. So $\lim_{p \to \infty} K_p$ exists, and since the limit of a sequence of symmetric positive semidefinite matrices is symmetric positive semidefinite, k is a PSD kernel.

□

Example 8.9. Let $x, z \in \mathbb{R}^2$ and $f(x) = x_1 x_2$.

- $k(x,z) = \langle x, z \rangle$ is a kernel. A corresponding feature map is $\phi(x) = x$.
- $k(x,z) = f(x)f(z)$ is a kernel. A corresponding feature map is $\phi(x) = x_1 x_2$.
- $k(x,z) = \langle x, z \rangle + f(x)f(z)$ is a kernel. A corresponding feature map is $\phi(x) = (x_1, x_2, x_1 x_2)$.
- $k(x,z) = (\langle x, z \rangle)^2$ is a kernel. A corresponding feature map is $\phi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2)$.

Theorem 8.10. *The following are PSD kernels:*

- $k(x,z) = 1$.
- $k(x,z) = x^T z$, *with* $x, z \in \mathbb{R}^n$.
- $k(x,z) = x^T P z$, *where* P *is a symmetric positive semidefinite matrix and* $x, z \in \mathbb{R}^n$.
- $k(x,z)^d$, *where* k *is a PSD kernel and* $d \geq 1$ *is an integer.*
- $k'(x,z) = q(k(x,z))$, *where* $q(s)$ *is a polynomial with positive coefficients and* k *is a PSD kernel.*
- $k'(x,z) = e^{k(x,z)}$, *where* k *is a PSD kernel.*
- $k(x,z) = e^{-\gamma \|x - z\|_2^2}$, *where* $\gamma > 0$ *and* $x, z \in \mathbb{R}^n$.

Proof. Let $m \geq 1$ be an integer and $\{x_j \in \mathcal{X}\}_{j=1}^m$. Assume that $\mathcal{X} \subseteq \mathbb{R}^n$ where necessary.

- $K = [k(x_i, x_j)] = 11^T$ is symmetric and positive semidefinite.
- $K(x,z) = \phi(x)^T \phi(z)$ where $\phi(x) = x$ is symmetric and positive semidefinite.
- Set $\phi(x) = \sqrt{P} x$. Then, $k(x,z) = \phi(x)^T \phi(z)$.
- By the fourth point in the preceding theorem, $k^2 = k \cdot k$ is a PSD kernel. By induction, the claim follows.
- k^j, where $j > 1$ is an integer, is a PSD kernel. For $\alpha_j > 0$, $\alpha_j k^j$ is a PSD kernel. The sum of PSD kernels is a PSD kernel. Thus, $k'(x,z) = q(k(x,z))$ is a PSD kernel.
- $k'(x,z) = \lim_{p \to \infty} \sum_{j=0}^p \frac{k(x,z)^j}{j!}$, so k' is a PSD kernel.
- Note that $k(x,z) = e^{-\gamma \|x\|_2^2} e^{2\gamma x^T z} e^{-\gamma \|z\|_2^2} = f(x) f(z) e^{2\gamma x^T z}$. From the previous results, k is a PSD kernel.

□

Definition 8.11. (Polynomial kernels). The kernels $k_1(x,z) = (x^T z)^2$ and $k_2(x,z) = (x^T z + 1)^2$ are examples of polynomial kernels. k_1 is of the form $k(x,z)^d$, and k_2 is of the form $q(k(x,z))$, where $k(x,z) = x^T z$. k_1 is called a **second order homogeneous polynomial kernel**, and k_2 is called a **second order inhomogeneous polynomial kernel**. More generally, $k(x,z) = (x^T z)^d$ is the **d-th order homogeneous polynomial kernel**, and $k(x,z) = (x^T z + 1)^d$ is the **d-th order inhomogeneous polynomial kernel**.

Definition 8.12. (Radial basis function (rbf) or Gaussian kernel). The kernel $k(x,z) = e^{-\gamma\|x-z\|_2^2}$, where $\gamma > 0$ and $x, z \in \mathbb{R}^n$, is called the radial basis function (rbf) or Gaussian kernel and corresponds to an embedding into an infinite dimensional space.

Remark 8.13. *** Consider the embedding that maps $x \in \mathbb{R}^n$ into the function $g_x(\theta) = \sqrt{C}e^{-\gamma\|\theta-x\|_2^2}$ for $\theta \in \mathbb{R}^n$. So,

$$\phi(x) = g_x(\theta) = \sqrt{C}e^{-\gamma\|\theta-x\|_2^2}.$$

Select C such that $C\int_{\mathbb{R}^2} e^{-2\gamma\|\theta\|_2^2} = 1$. $g_x(\theta)$ lies in the linear space $L_2(\mathbb{R}^n)$ of real valued square integrable functions on $\mathbb{R}^n$. This space has the inner product

$$\langle f(\theta), g(\theta)\rangle = \int_{\mathbb{R}^n} f(\theta)g(\theta)d\theta.$$

The kernel of this embedding is a RBF since:

$$\begin{aligned}
k(x,z) &= \langle \phi(x), \phi(z)\rangle \\
&= C\int_{\mathbb{R}^n} e^{-\gamma\|\theta-x\|_2^2} e^{-\gamma\|\theta-z\|_2^2} d\theta \\
&= e^{-\gamma(x^T x + z^T z)} e^{\frac{\gamma}{2}(x+z)^T(x+z)} C\int_{\mathbb{R}^n} e^{-2\gamma(\theta^T\theta - \theta^T(x+z) + \frac{1}{4}(x+z)^T(x+z))} d\theta \\
&= e^{-\frac{\gamma}{2}(x^T x - 2x^T z + z^T z)} C\int_{\mathbb{R}^n} e^{-2\gamma\left\|\theta - \frac{1}{2}(x+z)\right\|_2^2} d\theta \\
&= e^{-\frac{\gamma}{2}\|x-z\|_2^2}.
\end{aligned}$$

8.2 Classification with kernels ***

We explore how we can extend machine learning algorithms using kernels. As usual, we have training data $\{(x_i, y_i)\}_{i=1}^n$, with $x_i \in \mathbb{R}^n$, $y_i \in \mathcal{Y}$, a finite set, and aim to find a classifier $\hat{y}(x)$ to label new data. We consider feature maps $\phi : \mathbb{R}^n \to \mathbb{R}^q$ for finite q (finite for now).

8.2.1 Kernel nearest neighbor classification

Given a test point $x \in \mathbb{R}^n$, recall that we find

$$i^* = \arg\min_i \|x - x_i\|_2^2$$

and assign $\hat{y}(x) = y_{i^*}$. Note that

$$\min_i \|x - x_i\|_2^2 = \min_i (x - x_i)^T(x - x_i) = \min_i x^T x - 2x_i^T x + x_i^T x_i.$$

$$\implies \max_i 2x_i^T x - x_i^T x_i.$$

Do we have room for kernels here? We argue yes.

$$\max_i 2K(x_i, x) - K(x_i, x_i),$$

where the second term is the diagonal cluster of K.

8.2.2 Kernel nearest mean classification

We have classes $\{1, 2, \ldots, c\} = Y$. $X = \begin{pmatrix} X_1 & X_2 & \ldots & X_c \end{pmatrix}$. $\mu_1 = \frac{1}{m_1} X_1 \mathbf{1}$, $\mu_i = \frac{1}{m_i} X_i \mathbf{1}$.

Notice the pattern. We take the existing classifier and try to write it in terms of an inner product of the test point and the data in the i-th path. We find the kernel. We extrapolate up to any finite-dimensional space.

We can also do the generic embedding first, find where the inner products are, and exploit the inner products by replacing them with kernel evaluations.

8.2.3 Kernel SVM classification

We want to put a kernel inside the SVM to get a non-linear version of the SVM. The easiest way to accomplish this is to examine the dual problem:

$$\max_{\alpha \in \mathbb{R}^m} \mathbf{1}^T \alpha - \frac{1}{2} \alpha^T Z^T Z \alpha$$

$$\text{s.t. } \alpha^T y = 0, \alpha \leq c\mathbf{1}, \alpha \geq 0.$$

We don't have a cloesd-form computation to do and instead have to solve the optimization problem.

8.3 Kernel PCA ***

8.4 Kernel ridge regression ***

8.5 Reproducing kernel Hilbert spaces (RKHS) ***

8.5.1 Hilbert spaces

8.5.2 Reproducing kernel Hilbert spaces

8.5.3 RKHS representer theorems

1. $h \in \mathcal{X} \implies h = \sum_{i=1}^m \sigma(i) x_i$. $a = [a(i)] \in \mathbb{R}^m$.
2. $h \in \mathcal{H} \implies h = \hat{h} + \tilde{h}$, with $\hat{h} \in \mathcal{X}, \tilde{h} \in x^\perp$.
3. $G = [\langle x_i, x_j \rangle] \in \mathbb{R}^{n \times m}$.
4. $\forall\ h \in \mathcal{H}$, $k(h) = [\langle x_i, h \rangle] \in \mathbb{R}^m$.
5. $\forall\ h \in \mathcal{H}\ \forall x_i$, $\langle x_i, h \rangle = \langle x_i, \hat{h} \rangle$.

6. $\forall\ h \in \mathcal{H}\ \exists\ a \in \mathbb{R}^m$ that depends only on $\hat{y}$ such that $\forall\ x_j$, $\langle x_j, h\rangle = \sum_{i=1}^m a(i)\langle x_i, x_j\rangle$.

7. $\forall\ h \in \mathcal{H}\ \exists\ a \in \mathbb{R}^m$ such that $k(h) = Ga$.

We now discuss the loss function to use. We have data $X = \{x_i\}_{i=1}^m, y = [y_i]$. Our loss function is $J(h, b, X, y)$, $b \in \mathbb{R}^p, h \in \mathcal{H}$, and we aim to solve $\min_{h,b} J(h, b, X, y)$. We make the following big assumption:

$$J(h, b, X, y) = J'(k(h), b, G, y).$$

Example 8.14.

$y_i \in \mathbb{R}$, $y = [y_i] \in \mathbb{R}^m$, with no auxiliary variables. $J(h, X, y) = \sum_{i=1}^m (y_i - \langle x_i, h\rangle)^2 = \|y - k(h)\|_2^2$.

$y_i \in \{+1, -1\}$. $H(z) = \max(0, z) = \begin{cases} 0 & z \geq 0 \\ -z & z < 0 \end{cases}$. $J(h, b, X, y) = \sum_{i=1}^m H(y_i(\langle x_i, h\rangle + b) - 1)$.

Theorem 8.15. *Let $J(h, b, X, y)$ be a loss function satisfying $J(h, b, X, y) = J'(k(h), b, G, y)$ and $\min_{h,b} J(h, b, X, y)$ having a solution (h^*, b^*). Then there exists $h^* \in \mathcal{X}$ such that (h^*, b^*) is a solution, and $h^* = \sum_{i=1}^m \sigma(i)x_i$.*

The loss function depends on the training data, but now to seek regularization, we need the regularized loss function to not depend on the data. We introduce a term $R(h) = \|h\|_{\mathcal{H}}^2$ $(\hat{h} + \tilde{h} = h)$. So $C(h, b) = J(h, b, X, y) + \lambda R(h)$. $(h^*, b^*), h^* \in \mathcal{R}, h^* = \sum_{i=1}^m a(i)x_i$.

Example 8.16.

$$\begin{aligned} J(h, X, y) &= \sum_{i=1}^m (y_i - \langle x_i, h\rangle)^2 + \|h\|_{\mathcal{H}}^2 \\ &= \|y - k(h)\|_2^2 + \lambda\|h\|_{\mathcal{H}}^2 \\ &= \|y - Ga\|_2^2 + \lambda a^T G a. \end{aligned}$$

$\frac{1}{2}\|h\|_{\mathcal{H}}^2 + C\sum_{i=1}^m s_i$. $y_i\langle x_i, h\rangle + y_i b + s_i - 1 \geq 0$. $s_i \geq 0$. $s_0 = 0$ if $y_i(\langle x_i, h\rangle + b) \geq 1$. $s_i = 1 - y_i(\langle x_i, h\rangle + b)y_i(\langle x_i, h\rangle) + b < 1$.

8.5.4 Kernel regularized least squares

9 Deep learning today ***

9.1 Bayesian generative models (graphical models)

We cover the following subtopics:

- Model
- Expectation maximization
- Variational inference.

To keep things simple, suppose that we want to do classification with training data $\{X_j\}_{j=1}^m$. An example data is

$$(x, y),\ x \sim X,\ y \sim Y.$$

We get to observe X but do not get to observe Y; thus, our problem is to predict Y, where, say, $Y \in \{0, 1\}$. We assume that X depends on Y, thus depending on **hyperparameters** $\mu_0, \Sigma_0, \mu_1, \Sigma_1$, and that Y depends on a hyperparameter π_1. Assume that

$$f_{X|Y}(x|y) = \mathcal{N}(\mu_y, \Sigma_y).$$

We call this model "generative" because we can actually generate data based on this model.

Remark 9.1. (Bayes rule).

$$f_{X|Y}(x|y) = \frac{1}{(2\pi)^{n/2}} \frac{1}{|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)}.$$

The posterior probability is

$$\mathbb{P}_{Y|X}(k|x) = \frac{f_{X|Y}(x|k)\mathbb{P}_Y(k)}{f_X(x)} = \frac{\frac{1}{(2\pi)^{n/2}} \frac{1}{|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)}}{f_X(x)}.$$

The nice thing about these Bayesian models is that we have a posterior distribution and, therefore, an estimate of the likelihood that we are correct.

Remark 9.2. How do we choose k? There can be many criteria, depending on the weights on the outcomes. One criteria could be to minimize error—to get as many predictions right as possible—or to maximize the probability that we are right.

To find the outcome that maximizes the posterior probability, we want to find

$$\begin{aligned}\hat{y}(x) &= \arg\max_k -\frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k) - \frac{1}{2}\ln|\Sigma_k| + \ln(\pi_k) \\ &= \arg\min_k \frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k) + \frac{1}{2}\ln|\Sigma_k| - \ln(\pi_k).\end{aligned}$$

This is called the **Bayes classifier** and maximizes the probability that we are right. We can estimate the parameters by, for instance, maximum likelihood estimation on the training data.

Question 9.3. Can this classifier be improved? We make some assumptions that we know all of these parameters, but what if we don't?

Remark 9.4. How many parameters are in the Bayes classifier? μ_k, for $k \in \{0, 1\}$, has $2n$ parameters. Σ_k has $O(n^2)$ parameters. π_k has 2 parameters. While μ_k is linear in n, Σ_k is quadratic in n—we should regularize this as the class of classifiers is too large. We want to control the complexity in the family by either penalizing the complex classifiers or, more harshly, not considering the complex classifiers. For instance, we could assume that Σ_k is diagonal, resulting in the **Naive Bayes classifier**, which has a linear number of parameters in n.

Remark 9.5. One step up: we could assume that we have a *mixture* of Gaussians, rather than just one Gaussian. We can also add a level of complexity with expectation maximization, and again with variational inference.

9.2 Gaussian processes

Coming out of the statistics community, this topic covers how to learn a function. Suppose that we can control a function (e.g., the smoothness), which models some random process; after training the model using training data (values of the function), we can then do prediction. We can get both a mean estimate for the function we are modeling and a confidence interval estimate around the mean.

Suppose that we have some Gaussian process $\{X_t\}, t \in T$, with some finite number of indices $t_1, t_2, \ldots, t_p$.

$$f_{X_{t_1,\ldots,t_p}} \sim \mathcal{N}(\mu_{t_1,\ldots,t_p}, \Sigma).$$

Some examples of T include:

- $T = \{1, 2, \ldots, n\}$.
- $T = \{0, 1, \ldots\}$.
- $T = [0, 1]$.
- $T = [0, \infty)$.

We get to see some examples of deterministic outcomes drawn from the process and try to fit the parameters of this probabilistic model to best match the training data. We can control the smoothness of the random function drawn from the Gaussian process by controlling the covariance Σ, or indirectly, controlling the kernel function that controls the covariance.

$$k(X_s, X_t).$$

$$k(X, Z) = ce^{-\sum_{j=1}^{n}\left(\frac{x(j)-Z(j)}{\sigma_j}\right)^2}.$$

9.3 Deep learning

Suppose we have an input layer vector x with a vector of weights w and an offset b. We can compute $w^T x + b$, put it through some non-linearization (e.g., the sigmoid function), and get an output y. We call the simple model the **perceptron**. To make this "deep" learning, we add additional hidden layers, resulting in a **fully connected neural network**. We can extend our discussion to the following categories of neural networks:

- Fully connected neural nets
- Convolutional neural nets
- Autoencoders
- Recurrent neural nets.

9.3.1 Fully connected neural networks (FCNNs)

We introduce the general framework for this problem. We want the outcome probabilities to be differentiated, but how do we tune the parameters such that the probability of class 1 assignment is high and the probability of class 2 assignment is low?

We can incorporate some cost function for each example:

$$c(x) = \|y - a^L\|_2^2.$$

The total cost is then

$$\frac{1}{M} \sum_{j=1}^{M} \frac{1}{a} \|y_j - a_j^L(x_j)\|_2^2.$$

Instead of backpropagation, another technique is to put all of the examples in one big batch, figure out the total cost, and try to get the gradient of the total cost. The problem is that M can be very large, so it is very time consuming to run through all the data before finding one gradient and doing one update. Thus, rather than doing gradient descent, we can do stochastic gradient descent by generating "mini-batches" of examples (say, 10 examples) to move against the direction of the noise of the gradient. Note that we can introduce randomization by randomly selecting mini-batches. We work out the cost for each mini-batch and then ask how we can change the parameters to reduce the gradient cost for that mini-batch. It usually requires an order of 1,000 epochs to train a network.

How do we know when to stop training? When we take a step in the negative gradient, how big of a step should we take (what should our learning rate be)?

9.3.2 Convolutional neural networks (CNNs)

While a fully connected network would have connections from any element in x to any element in L^2 (layer 2), a convolutional neural network ...

This is a feature map—it is looking for what strength each feature is present at a specific location in the input.

NN vs CNNs:

- Structured weights vs. replicated filters.
-

9.4 Final examination

- 3 hours.
- 4 parts: three parts on course material and one bonus part with a bonus question on final on the two lectures in reading period.

Example 9.6. (Part 2). Solve $\frac{x^T P x}{x^T Q x}$. (We know how to solve the numerator and denominator, separately.) This is called the generalized Raleigh quotient.