Review Notes and Supplementary Notes
CS229 Course
Machine Learning
Standford University

Topics Covered:

1. Linear Algebra
2. Probability Theory
3. Convex Optimization
4. Hidden Markov Models
5. Maximum Entropy and Exponential Families
6. The Multivariate Gaussian Distribution
7. Gaussian Processes
8. Hoeffding's inequality

Reference:

[1] http://cs229.stanford.edu/

[2] https://see.stanford.edu/Course/CS229

Mail: tanujitisi@gmail.com Webpage: www.ctanujit.org

Linear Algebra Review and Reference

Zico Kolter

October 16, 2007

1 Basic Concepts and Notation

Linear algebra provides a way of compactly representing and operating on sets of linear equations. For example, consider the following system of equations:

$$\begin{array}{rcrcl} 4x_1 & - & 5x_2 & = & -13 \\ -2x_1 & + & 3x_2 & = & 9 \end{array} .$$

This is two equations and two variables, so as you know from high school algebra, you can find a unique solution for x_1 and x_2 (unless the equations are somehow degenerate, for example if the second equation is simply a multiple of the first, but in the case above there is in fact a unique solution). In matrix notation, we can write the system more compactly as:

$$Ax = b$$

$$\text{with } A = \begin{bmatrix} 4 & -5 \\ -2 & 3 \end{bmatrix}, \; b = \begin{bmatrix} 13 \\ -9 \end{bmatrix}.$$

As we will see shortly, there are many advantages (including the obvious space savings) to analyzing linear equations in this form.

1.1 Basic Notation

We use the following notation:

- By $A \in \mathbb{R}^{m\times n}$ we denote a matrix with m rows and n columns, where the entries of A are real numbers.

- By $x \in \mathbb{R}^n$, we denote a vector with n entries. Usually a vector x will denote a ***column vector*** — i.e., a matrix with n rows and 1 column. If we want to explicitly represent a ***row vector*** — a matrix with 1 row and n columns — we typically write x^T (here x^T denotes the transpose of x, which we will define shortly).

- The ith element of a vector x is denoted x_i:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

- We use the notation a_{ij} (or A_{ij}, $A_{i,j}$, etc) to denote the entry of A in the ith row and jth column:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

- We denote the jth column of A by a_j or $A_{:,j}$:

$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix}.$$

- We denote the ith row of A by a_i^T or $A_{i,:}$:

$$A = \begin{bmatrix} — & a_1^T & — \\ — & a_2^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix}.$$

- Note that these definitions are ambiguous (for example, the a_1 and a_1^T in the previous two definitions are *not* the same vector). Usually the meaning of the notation should be obvious from its use.

2 Matrix Multiplication

The product of two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times p}$ is the matrix

$$C = AB \in \mathbb{R}^{m\times p},$$

where

$$C_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}.$$

Note that in order for the matrix product to exist, the number of columns in A must equal the number of rows in B. There are many ways of looking at matrix multiplication, and we'll start by examining a few special cases.

2.1 Vector-Vector Products

Given two vectors $x, y \in \mathbb{R}^n$, the quantity $x^T y$, sometimes called the ***inner product*** or ***dot product*** of the vectors, is a real number given by

$$x^T y \in \mathbb{R} = \sum_{i=1}^{n} x_i y_i.$$

Note that it is always the case that $x^T y = y^T x$.

Given vectors $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$ (they no longer have to be the same size), xy^T is called the ***outer product*** of the vectors. It is a matrix whose entries are given by $(xy^T)_{ij} = x_i y_j$, i.e.,

$$xy^T \in \mathbb{R}^{m \times n} = \begin{bmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_m y_1 & x_m y_2 & \cdots & x_m y_n \end{bmatrix}.$$

2.2 Matrix-Vector Products

Given a matrix $A \in \mathbb{R}^{m \times n}$ and a vector $x \in \mathbb{R}^n$, their product is a vector $y = Ax \in \mathbb{R}^m$. There are a couple ways of looking at matrix-vector multiplication, and we will look at them both.

If we write A by rows, then we can express Ax as,

$$y = \begin{bmatrix} \text{—} & a_1^T & \text{—} \\ \text{—} & a_2^T & \text{—} \\ & \vdots & \\ \text{—} & a_m^T & \text{—} \end{bmatrix} x = \begin{bmatrix} a_1^T x \\ a_2^T x \\ \vdots \\ a_m^T x \end{bmatrix}.$$

In other words, the ith entry of y is equal to the inner product of the ith *row* of A and x, $y_i = a_i^T x$.

Alternatively, lets write A in column form. In this case we see that,

$$y = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_1 \end{bmatrix} x_1 + \begin{bmatrix} a_2 \end{bmatrix} x_2 + \ldots + \begin{bmatrix} a_n \end{bmatrix} x_n \ .$$

In other words, y is a ***linear combination*** of the *columns* of A, where the coefficients of the linear combination are given by the entries of x.

So far we have been multiplying on the right by a column vector, but it is also possible to multiply on the left by a row vector. This is written, $y^T = x^T A$ for $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^m$, and $y \in \mathbb{R}^n$. As before, we can express y^T in two obvious ways, depending on whether we express A in terms on its rows or columns. In the first case we express A in terms of its columns, which gives

$$y^T = x^T \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} = \begin{bmatrix} x^T a_1 & x^T a_2 & \cdots & x^T a_n \end{bmatrix}$$

which demonstrates that the ith entry of y^T is equal to the inner product of x and the ith *column* of A.

Finally, expressing A in terms of rows we get the final representation of the vector-matrix product,

$$\begin{aligned} y^T &= \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} \text{—} & a_1^T & \text{—} \\ \text{—} & a_2^T & \text{—} \\ & \vdots & \\ \text{—} & a_m^T & \text{—} \end{bmatrix} \\ &= x_1 \begin{bmatrix} \text{—} & a_1^T & \text{—} \end{bmatrix} + x_2 \begin{bmatrix} \text{—} & a_2^T & \text{—} \end{bmatrix} + ... + x_n \begin{bmatrix} \text{—} & a_n^T & \text{—} \end{bmatrix} \end{aligned}$$

so we see that y^T is a linear combination of the *rows* of A, where the coefficients for the linear combination are given by the entries of x.

2.3 Matrix-Matrix Products

Armed with this knowledge, we can now look at four different (but, of course, equivalent) ways of viewing the matrix-matrix multiplication $C = AB$ as defined at the beginning of this section. First we can view matrix-matrix multiplication as a set of vector-vector products. The most obvious viewpoint, which follows immediately from the definition, is that the i, j entry of C is equal to the inner product of the ith row of A and the jth row of B. Symbolically, this looks like the following,

$$C = AB = \begin{bmatrix} \text{—} & a_1^T & \text{—} \\ \text{—} & a_2^T & \text{—} \\ & \vdots & \\ \text{—} & a_m^T & \text{—} \end{bmatrix} \begin{bmatrix} | & | & & | \\ b_1 & b_2 & \cdots & b_p \\ | & | & & | \end{bmatrix} = \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdots & a_1^T b_p \\ a_2^T b_1 & a_2^T b_2 & \cdots & a_2^T b_p \\ \vdots & \vdots & \ddots & \vdots \\ a_m^T b_1 & a_m^T b_2 & \cdots & a_m^T b_p \end{bmatrix}.$$

Remember that since $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times p}$, $a_i \in \mathbb{R}^n$ and $b_j \in \mathbb{R}^n$, so these inner products all make sense. This is the most "natural" representation when we represent A by rows and B by columns. Alternatively, we can represent A by columns, and B by rows, which leads to the interpretation of AB as a sum of outer products. Symbolically,

$$C = AB = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} \text{—} & b_1^T & \text{—} \\ \text{—} & b_2^T & \text{—} \\ & \vdots & \\ \text{—} & b_n^T & \text{—} \end{bmatrix} = \sum_{i=1}^{n} a_i b_i^T \ .$$

Put another way, AB is equal to the sum, over all i, of the outer product of the ith column of A and the ith row of B. Since, in this case, $a_i \in \mathbb{R}^m$ and $b_i \in \mathbb{R}^p$, the dimension of the outer product $a_i b_i^T$ is $m \times p$, which coincides with the dimension of C.

Second, we can also view matrix-matrix multiplication as a set of matrix-vector products. Specifically, if we represent B by columns, we can view the columns of C as matrix-vector products between A and the columns of B. Symbolically,

$$C = AB = A \left[\begin{array}{cccc} | & | & & | \\ b_1 & b_2 & \cdots & b_p \\ | & | & & | \end{array} \right] = \left[\begin{array}{cccc} | & | & & | \\ Ab_1 & Ab_2 & \cdots & Ab_p \\ | & | & & | \end{array} \right].$$

Here the ith column of C is given by the matrix-vector product with the vector on the right, $c_i = Ab_i$. These matrix-vector products can in turn be interpreted using both viewpoints given in the previous subsection. Finally, we have the analogous viewpoint, where we represent A by rows, and view the rows of C as the matrix-vector product between the rows of A and C. Symbolically,

$$C = AB = \left[\begin{array}{ccc} — & a_1^T & — \\ — & a_2^T & — \\ & \vdots & \\ — & a_m^T & — \end{array} \right] B = \left[\begin{array}{ccc} — & a_1^T B & — \\ — & a_2^T B & — \\ & \vdots & \\ — & a_m^T B & — \end{array} \right].$$

Here the ith row of C is given by the matrix-vector product with the vector on the left, $c_i^T = a_i^T B$.

It may seem like overkill to dissect matrix multiplication to such a large degree, especially when all these viewpoints follow immediately from the initial definition we gave (in about a line of math) at the beginning of this section. However, virtually all of linear algebra deals with matrix multiplications of some kind, and it is worthwhile to spend some time trying to develop an intuitive understanding of the viewpoints presented here.

In addition to this, it is useful to know a few basic properties of matrix multiplication at a higher level:

- Matrix multiplication is associative: $(AB)C = A(BC)$.
- Matrix multiplication is distributive: $A(B + C) = AB + AC$.
- Matrix multiplication is, in general, *not* commutative; that is, it can be the case that $AB \neq BA$.

3 Operations and Properties

In this section we present several operations and properties of matrices and vectors. Hopefully a great deal of this will be review for you, so the notes can just serve as a reference for these topics.

3.1 The Identity Matrix and Diagonal Matrices

The ***identity matrix***, denoted $I \in \mathbb{R}^{n\times n}$, is a square matrix with ones on the diagonal and zeros everywhere else. That is,

$$I_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

It has the property that for all $A \in \mathbb{R}^{m\times n}$,

$$AI = A = IA$$

where the size of I is determined by the dimensions of A so that matrix multiplication is possible.

A ***diagonal matrix*** is a matrix where all non-diagonal elements are 0. This is typically denoted $D = \mathrm{diag}(d_1, d_2, \ldots, d_n)$, with

$$D_{ij} = \begin{cases} d_i & i = j \\ 0 & i \neq j \end{cases}$$

Clearly, $I = \mathrm{diag}(1, 1, \ldots, 1)$.

3.2 The Transpose

The ***transpose*** of a matrix results from "flipping" the rows and columns. Given a matrix $A \in \mathbb{R}^{m\times n}$, is transpose, written A^T, is defined as

$$A^T \in \mathbb{R}^{n\times m},\ (A^T)_{ij} = A_{ji}\ \ .$$

We have in fact already been using the transpose when describing row vectors, since the transpose of a column vector is naturally a row vector.

The following properties of transposes are easily verified:

- $(A^T)^T = A$
- $(AB)^T = B^T A^T$
- $(A+B)^T = A^T + B^T$

3.3 Symmetric Matrices

A square matrix $A \in \mathbb{R}^{n\times n}$ is ***symmetric*** if $A = A^T$. It is ***anti-symmetric*** if $A = -A^T$. It is easy to show that for any matrix $A \in \mathbb{R}^{n\times n}$, the matrix $A + A^T$ is symmetric and the matrix $A - A^T$ is anti-symmetric. From this it follows that any square matrix $A \in \mathbb{R}^{n\times n}$ can be represented as a sum of a symmetric matrix and an anti-symmetric matrix, since

$$A = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T)$$

and the first matrix on the right is symmetric, while the second is anti-symmetric. It turns out that symmetric matrices occur a great deal in practice, and they have many nice properties which we will look at shortly. It is common to denote the set of all symmetric matrices of size n as $\mathbb{S}^n$, so that $A \in \mathbb{S}^n$ means that A is a symmetric $n \times n$ matrix;

3.4 The Trace

The ***trace*** of a square matrix $A \in \mathbb{R}^{n\times n}$, denoted $\mathrm{tr}(A)$ (or just $\mathrm{tr}A$ if the parentheses are obviously implied), is the sum of diagonal elements in the matrix:

$$\mathrm{tr}A = \sum_{i=1}^{n} A_{ii}.$$

As described in the CS229 lecture notes, the trace has the following properties (included here for the sake of completeness):

- For $A \in \mathbb{R}^{n\times n}$, $\mathrm{tr}A = \mathrm{tr}A^T$.
- For $A, B \in \mathbb{R}^{n\times n}$, $\mathrm{tr}(A + B) = \mathrm{tr}A + \mathrm{tr}B$.
- For $A \in \mathbb{R}^{n\times n}$, $t \in \mathbb{R}$, $\mathrm{tr}(tA) = t\,\mathrm{tr}A$.
- For A, B such that AB is square, $\mathrm{tr}AB = \mathrm{tr}BA$.
- For A, B, C such that ABC is square, $\mathrm{tr}ABC = \mathrm{tr}BCA = \mathrm{tr}CAB$, and so on for the product of more matrices.

3.5 Norms

A ***norm*** of a vector $\|x\|$ is informally measure of the “length” of the vector. For example, we have the commonly-used Euclidean or ℓ_2 norm,

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Note that $\|x\|_2^2 = x^T x$.

More formally, a norm is any function $f : \mathbb{R}^n \to \mathbb{R}$ that satisfies 4 properties:

1. For all $x \in \mathbb{R}^n$, $f(x) \geq 0$ (non-negativity).
2. $f(x) = 0$ if and only if $x = 0$ (definiteness).
3. For all $x \in \mathbb{R}^n$, $t \in \mathbb{R}$, $f(tx) = |t|f(x)$ (homogeneity).
4. For all $x, y \in \mathbb{R}^n$, $f(x + y) \leq f(x) + f(y)$ (triangle inequality).

Other examples of norms are the ℓ_1 norm,

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

and the ℓ_∞ norm,

$$\|x\|_\infty = \max_i |x_i|.$$

In fact, all three norms presented so far are examples of the family of ℓ_p norms, which are parameterized by a real number $p \geq 1$, and defined as

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}.$$

Norms can also be defined for matrices, such as the Frobenius norm,

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2} = \sqrt{\operatorname{tr}(A^T A)}.$$

Many other norms exist, but they are beyond the scope of this review.

3.6 Linear Independence and Rank

A set of vectors $\{x_1, x_2, \ldots x_n\}$ is said to be ***(linearly) independent*** if no vector can be represented as a linear combination of the remaining vectors. Conversely, a vector which *can* be represented as a linear combination of the remaining vectors is said to be ***(linearly) dependent***. For example, if

$$x_n = \sum_{i=1}^{n-1} \alpha_i x_i$$

for some $\{\alpha_1, \ldots, \alpha_{n-1}\}$ then x_n is dependent on $\{x_1, \ldots, x_{n-1}\}$; otherwise, it is independent of $\{x_1, \ldots, x_{n-1}\}$.

The ***column rank*** of a matrix A is the largest number of columns of A that constitute linearly independent set. This is often referred to simply as the number of linearly independent columns, but this terminology is a little sloppy, since it is possible that any vector in some set $\{x_1, \ldots x_n\}$ can be expressed as a linear combination of the remaining vectors, even though some subset of the vectors might be independent. In the same way, the ***row rank*** is the largest number of rows of A that constitute a linearly independent set.

It is a basic fact of linear algebra, that for any matrix A, $\operatorname{columnrank}(A) = \operatorname{rowrank}(A)$, and so this quantity is simply refereed to as the ***rank*** of A, denoted as $\operatorname{rank}(A)$. The following are some basic properties of the rank:

- For $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank}(A) \leq \min(m, n)$. If $\operatorname{rank}(A) = \min(m, n)$, then A is said to be ***full rank***.

- For $A \in \mathbb{R}^{m \times n}$, $\text{rank}(A) = \text{rank}(A^T)$.
- For $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $\text{rank}(AB) \leq \min(\text{rank}(A), \text{rank}(B))$.
- For $A, B \in \mathbb{R}^{m \times n}$, $\text{rank}(A + B) \leq \text{rank}(A) + \text{rank}(B)$.

3.7 The Inverse

The ***inverse*** of a square matrix $A \in \mathbb{R}^{n \times n}$ is denoted A^{-1}, and is the unique matrix such that

$$A^{-1}A = I = AA^{-1}.$$

It turns out that A^{-1} may not exist for some matrices A; we say A is ***invertible*** or ***non-singular*** if A^{-1} exists and ***non-invertible*** or ***singular*** otherwise. One condition for invertibility we already know: it is possible to show that A^{-1} exists if and only if A is full rank. We will soon see that there are many alternative sufficient and necessary conditions, in addition to full rank, for invertibility. The following are properties of the inverse; all assume that $A, B \in \mathbb{R}^{n \times n}$ are non-singular:

- $(A^{-1})^{-1} = A$
- If $Ax = b$, we can multiply by A^{-1} on both sides to obtain $x = A^{-1}b$. This demonstrates the inverse with respect to the original system of linear equalities we began this review with.
- $(AB)^{-1} = B^{-1}A^{-1}$
- $(A^{-1})^T = (A^T)^{-1}$. For this reason this matrix is often denoted A^{-T}.

3.8 Orthogonal Matrices

Two vectors $x, y \in \mathbb{R}^n$ are ***orthogonal*** if $x^T y = 0$. A vector $x \in \mathbb{R}^n$ is ***normalized*** if $\|x\|_2 = 1$. A square matrix $U \in \mathbb{R}^{n \times n}$ is ***orthogonal*** (note the different meanings when talking about vectors versus matrices) if all its columns are orthogonal to each other and are normalized (the columns are then referred to as being ***orthonormal***).

It follows immediately from the definition of orthogonality and normality that

$$U^T U = I = UU^T.$$

In other words, the inverse of an orthogonal matrix is its transpose. Note that if U is not square — i.e., $U \in \mathbb{R}^{m \times n}$, $n < m$ — but its columns are still orthonormal, then $U^T U = I$, but $UU^T \neq I$. We generally only use the term orthogonal to describe the previous case, where U is square.

Another nice property of orthogonal matrices is that operating on a vector with an orthogonal matrix will not change its Euclidean norm, i.e.,

$$\|Ux\|_2 = \|x\|_2$$

for any $x \in \mathbb{R}^n$, $U \in \mathbb{R}^{n \times n}$ orthogonal.

3.9 Range and Nullspace of a Matrix

The ***span*** of a set of vectors $\{x_1, x_2, \ldots x_n\}$ is the set of all vectors that can be expressed as a linear combination of $\{x_1, \ldots, x_n\}$. That is,

$$\operatorname{span}(\{x_1, \ldots x_n\}) = \left\{ v : v = \sum_{i=1}^{n} \alpha_i x_i, \;\; \alpha_i \in \mathbb{R} \right\}.$$

It can be shown that if $\{x_1, \ldots, x_n\}$ is a set of n linearly independent vectors, where each $x_i \in \mathbb{R}^n$, then $\operatorname{span}(\{x_1, \ldots x_n\}) = \mathbb{R}^n$. In other words, *any* vector $v \in \mathbb{R}^n$ can be written as a linear combination of x_1 through x_n. The ***projection*** of a vector $y \in \mathbb{R}^m$ onto the span of $\{x_1, \ldots, x_n\}$ (here we assume $x_i \in \mathbb{R}^m$) is the vector $v \in \operatorname{span}(\{x_1, \ldots x_n\})$, such that v as close as possible to y, as measured by the Euclidean norm $\|v - y\|_2$. We denote the projection as $\operatorname{Proj}(y; \{x_1, \ldots, x_n\})$ and can define it formally as,

$$\operatorname{Proj}(y; \{x_1, \ldots x_n\}) = \operatorname{argmin}_{v \in \operatorname{span}(\{x_1, \ldots, x_n\})} \|y - v\|_2.$$

The ***range*** (sometimes also called the columnspace) of a matrix $A \in \mathbb{R}^{m \times n}$, denoted $\mathcal{R}(A)$, is the the span of the columns of A. In other words,

$$\mathcal{R}(A) = \{v \in \mathbb{R}^m : v = Ax, x \in \mathbb{R}^n\}.$$

Making a few technical assumptions (namely that A is full rank and that $n < m$), the projection of a vector $y \in \mathbb{R}^m$ onto the range of A is given by,

$$\operatorname{Proj}(y; A) = \operatorname{argmin}_{v \in \mathcal{R}(A)} \|v - y\|_2 = A(A^T A)^{-1} A^T y \;\; .$$

This last equation should look extremely familiar, since it is almost the same formula we derived in class (and which we will soon derive again) for the least squares estimation of parameters. Looking at the definition for the projection, it should not be too hard to convince yourself that this is in fact the same objective that we minimized in our least squares problem (except for a squaring of the norm, which doesn't affect the optimal point) and so these problems are naturally very connected. When A contains only a single column, $a \in \mathbb{R}^m$, this gives the special case for a projection of a vector on to a line:

$$\operatorname{Proj}(y; a) = \frac{a a^T}{a^T a} y \;\; .$$

The ***nullspace*** of a matrix $A \in \mathbb{R}^{m \times n}$, denoted $\mathcal{N}(A)$ is the set of all vectors that equal 0 when multiplied by A, i.e.,

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

Note that vectors in $\mathcal{R}(A)$ are of size m, while vectors in the $\mathcal{N}(A)$ are of size n, so vectors in $\mathcal{R}(A^T)$ and $\mathcal{N}(A)$ are both in $\mathbb{R}^n$. In fact, we can say much more. It turns out that

$$\left\{ w : w = u + v, u \in \mathcal{R}(A^T), v \in \mathcal{N}(A) \right\} = \mathbb{R}^n \text{ and } \mathcal{R}(A^T) \cap \mathcal{N}(A) = \emptyset \;\; .$$

In other words, $\mathcal{R}(A^T)$ and $\mathcal{N}(A)$ are disjoint subsets that together span the entire space of $\mathbb{R}^n$. Sets of this type are called ***orthogonal complements***, and we denote this $\mathcal{R}(A^T) = \mathcal{N}(A)^\perp$.

3.10 The Determinant

The ***determinant*** of a square matrix $A \in \mathbb{R}^{n\times n}$, is a function $\det : \mathbb{R}^{n\times n} \to \mathbb{R}$, and is denoted $|A|$ or $\det A$ (like the trace operator, we usually omit parentheses). The full formula for the determinant gives little intuition about its meaning, so we instead first give three defining properties of the determinant, from which all the rest follow (including the general formula):

1. The determinant of the identity is 1, $|I| = 1$.

2. Given a matrix $A \in \mathbb{R}^{n\times n}$, if we multiply a single row in A by a scalar $t \in \mathbb{R}$, then the determinant of the new matrix is $t|A|$,

$$\left| \begin{bmatrix} — & t\, a_1^T & — \\ — & a_2^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} \right| = t|A| \quad .$$

3. If we exchange any two rows a_i^T and a_j^T of A, then the determinant of the new matrix is $-|A|$, for example

$$\left| \begin{bmatrix} — & a_2^T & — \\ — & a_1^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} \right| = -|A| \quad .$$

These properties, however, also give very little intuition about the nature of the determinant, so we now list several properties that follow from the three properties above:

- For $A \in \mathbb{R}^{n\times n}$, $|A| = |A^T|$.

- For $A, B \in \mathbb{R}^{n\times n}$, $|AB| = |A||B|$.

- For $A \in \mathbb{R}^{n\times n}$, $|A| = 0$ if and only if A is singular (i.e., non-invertible).

- For $A \in \mathbb{R}^{n\times n}$ and A non-singular, $|A|^{-1} = 1/|A|$.

Before given the general definition for the determinant, we define, for $A \in \mathbb{R}^{n\times n}$, $A_{\setminus i,\setminus j} \in \mathbb{R}^{(n-1)\times(n-1)}$ to be the *matrix* that results from deleting the ith row and jth column from A. The general (recursive) formula for the determinant is

$$\begin{aligned} |A| &= \sum_{i=1}^{n} (-1)^{i+j} a_{ij} |A_{\setminus i,\setminus j}| \quad \text{(for any } j \in 1,\ldots,n) \\ &= \sum_{j=1}^{n} (-1)^{i+j} a_{ij} |A_{\setminus i,\setminus j}| \quad \text{(for any } i \in 1,\ldots,n) \end{aligned}$$

with the initial case that $|A| = a_{11}$ for $A \in \mathbb{R}^{1\times 1}$. If we were to expand this formula completely for $A \in \mathbb{R}^{n\times n}$, there would be a total of $n!$ (n factorial) different terms. For this reason, we hardly even explicitly write the complete equation of the determinant for matrices bigger than 3×3. However, the equations for determinants of matrices up to size 3×3 are fairly common, and it is good to know them:

$$\begin{aligned}
|[a_{11}]| &= a_{11} \\
\left| \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \right| &= a_{11}a_{22} - a_{12}a_{21} \\
\left| \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \right| &= \begin{array}{l} a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} \\ \quad -a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31} \end{array}
\end{aligned}$$

The ***classical adjoint*** (often just called the adjoint) of a matrix $A \in \mathbb{R}^{n\times n}$, is denoted $\mathrm{adj}(A)$, and defined as

$$\mathrm{adj}(A) \in \mathbb{R}^{n\times n}, \quad (\mathrm{adj}(A))_{ij} = (-1)^{i+j}|A_{\setminus j,\setminus i}|$$

(note the switch in the indices $A_{\setminus j,\setminus i}$). It can be shown that for any nonsingular $A \in \mathbb{R}^{n\times n}$,

$$A^{-1} = \frac{1}{|A|}\mathrm{adj}(A) \ .$$

While this is a nice "explicit" formula for the inverse of matrix, we should note that, numerically, there are in fact much more efficient ways of computing the inverse.

3.11 Quadratic Forms and Positive Semidefinite Matrices

Given a matrix square $A \in \mathbb{R}^{n\times n}$ and a vector $x \in \mathbb{R}$, the scalar value x^TAx is called a ***quadratic form***. Written explicitly, we see that

$$x^TAx = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}x_ix_j \ .$$

Note that,

$$x^TAx = (x^TAx)^T = x^TA^Tx = x^T(\frac{1}{2}A + \frac{1}{2}A^T)x$$

i.e., only the symmetric part of A contributes to the quadratic form. For this reason, we often implicitly assume that the matrices appearing in a quadratic form are symmetric.

We give the following definitions:

- A symmetric matrix $A \in \mathbb{S}^n$ is ***positive definite*** (PD) if for all non-zero vectors $x \in \mathbb{R}^n$, $x^TAx > 0$. This is usually denoted $A \succ 0$ (or just $A > 0$), and often times the set of all positive definite matrices is denoted $\mathbb{S}^n_{++}$.

- A symmetric matrix $A \in \mathbb{S}^n$ is ***position semidefinite*** (PSD) if for all vectors $x^T Ax \geq 0$. This is written $A \succeq 0$ (or just $A \geq 0$), and the set of all positive semidefinite matrices is often denoted $\mathbb{S}^n_+$.
- Likewise, a symmetric matrix $A \in \mathbb{S}^n$ is ***negative definite*** (ND), denoted $A \prec 0$ (or just $A < 0$) if for all non-zero $x \in \mathbb{R}^n$, $x^T Ax < 0$.
- Similarly, a symmetric matrix $A \in \mathbb{S}^n$ is ***negative semidefinite*** (NSD), denoted $A \succeq 0$ (or just $A \leq 0$) if for all $x \in \mathbb{R}^n$, $x^T Ax \leq 0$.
- Finally, a symmetric matrix $A \in \mathbb{S}^n$ is ***indefinite***, if it is neither positive semidefinite nor negative semidefinite — i.e., if there exists $x_1, x_2 \in \mathbb{R}^n$ such that $x_1^T Ax_1 > 0$ and $x_2^T Ax_2 < 0$.

It should be obvious that if A is positive definite, then $-A$ is negative definite and vice versa. Likewise, if A is positive semidefinite then $-A$ is negative semidefinite and vice versa. If A is indefinite, then so is $-A$. It can also be shown that positive definite and negative definite matrices are always invertible.

Finally, there is one type of positive definite matrix that comes up frequently, and so deserves some special mention. Given any matrix $A \in \mathbb{R}^{m\times n}$ (not necessarily symmetric or even square), the matrix $G = A^T A$ (sometimes called a ***Gram matrix***) is always positive semidefinite. Further, if $m \geq n$ (and we assume for convenience that A is full rank), then $G = A^T A$ is positive definite.

3.12 Eigenvalues and Eigenvectors

Given a square matrix $A \in \mathbb{R}^{n\times n}$, we say that $\lambda \in \mathbb{C}$ is an ***eigenvalue*** of A and $x \in \mathbb{C}^n$ is the corresponding ***eigenvector***[1] if

$$Ax = \lambda x, \quad x \neq 0 \ .$$

Intuitively, this definition means that multiplying A by the vector x results in a new vector that points in the same direction as x, but scaled by a factor λ. Also note that for any eigenvector $x \in \mathbb{C}^n$, and scalar $t \in \mathbb{C}$, $A(cx) = cAx = c\lambda x = \lambda(cx)$, so cx is also an eigenvector. For this reason when we talk about "the" eigenvector associated with λ, we usually assume that the eigenvector is normalized to have length 1 (this still creates some ambiguity, since x and $-x$ will both be eigenvectors, but we will have to live with this).

We can rewrite the equation above to state that (λ, x) is an eigenvalue-eigenvector pair of A if,

$$(\lambda I - A)x = 0, \quad x \neq 0 \ .$$

[1]Note that λ and the entries of x are actually in $\mathbb{C}$, the set of complex numbers, not just the reals; we will see shortly why this is necessary. Don't worry about this technicality for now, you can think of complex vectors in the same way as real vectors.

But $(\lambda I - A)x = 0$ has a non-zero solution to x if and only if $(\lambda I - A)$ has a non-empty nullspace, which is only the case if $(\lambda I - A)$ is singular, i.e.,

$$|(\lambda I - A)| = 0 \ \ .$$

We can now use the previous definition of the determinant to expand this expression into a (very large) polynomial in λ, where λ will have maximum degree n. We then find the n (possibly complex) roots of this polynomial to find the n eigenvalues $\lambda_1, \ldots, \lambda_n$. To find the eigenvector corresponding to the eigenvalue λ_i, we simply solve the linear equation $(\lambda_i I - A)x = 0$. It should be noted that this is not the method which is actually used in practice to numerically compute the eigenvalues and eigenvectors (remember that the complete expansion of the determinant has $n!$ terms); it is rather a mathematical argument.

The following are properties of eigenvalues and eigenvectors (in all cases assume $A \in \mathbb{R}^{n\times n}$ has eigenvalues $\lambda_i, \ldots, \lambda_n$ and associated eigenvectors $x_1, \ldots x_n$):

- The trace of a A is equal to the sum of its eigenvalues,
$$\mathrm{tr}A = \sum_{i=1}^{n} \lambda_i \ \ .$$
- The determinant of A is equal to the product of its eigenvalues,
$$|A| = \prod_{i=1}^{n} \lambda_i \ \ .$$
- The rank of A is equal to the number of non-zero eigenvalues of A.
- If A is non-singular then $1/\lambda_i$ is an eigenvalue of A^{-1} with associated eigenvector x_i, i.e., $A^{-1}x_i = (1/\lambda_i)x_i$.
- The eigenvalues of a diagonal matrix $D = \mathrm{diag}(d_1, \ldots d_n)$ are just the diagonal entries $d_1, \ldots d_n$.

We can write all the eigenvector equations simultaneously as

$$AX = X\Lambda$$

where the columns of $X \in \mathbb{R}^{n\times n}$ are the eigenvectors of A and Λ is a diagonal matrix whose entries are the eigenvalues of A, i.e.,

$$X \in \mathbb{R}^{n\times n} = \left[\begin{array}{cccc} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{array}\right], \ \ \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n) \ \ .$$

If the eigenvectors of A are linearly independent, then the matrix X will be invertible, so $A = X\Lambda X^{-1}$. A matrix that can be written in this form is called ***diagonalizable***.

3.13 Eigenvalues and Eigenvectors of Symmetric Matrices

Two remarkable properties come about when we look at the eigenvalues and eigenvectors of a symmetric matrix $A \in \mathbb{S}^n$. First, it can be shown that all the eigenvalues of A are real. Secondly, the eigenvectors of A are orthonormal, i.e., the matrix X defined above is an orthogonal matrix (for this reason, we denote the matrix of eigenvectors as U in this case). We can therefore represent A as $A = U\Lambda U^T$, remembering from above that the inverse of an orthogonal matrix is just its transpose.

Using this, we can show that the definiteness of a matrix depends entirely on the sign of its eigenvalues. Suppose $A \in \mathbb{S}^n = U\Lambda U^T$. Then

$$x^T A x = x^T U \Lambda U^T x = y^T \Lambda y = \sum_{i=1}^{n} \lambda_i y_i^2$$

where $y = U^T x$ (and since U is full rank, any vector $y \in \mathbb{R}^n$ can be represented in this form). Because y_i^2 is always positive, the sign of this expression depends entirely on the λ_i's. If all $\lambda_i > 0$, then the matrix is positive definite; if all $\lambda_i \geq 0$, it is positive semidefinite. Likewise, if all $\lambda_i < 0$ or $\lambda_i \leq 0$, then A is negative definite or negative semidefinite respectively. Finally, if A has both positive and negative eigenvalues, it is indefinite.

An application where eigenvalues and eigenvectors come up frequently is in maximizing some function of a matrix. In particular, for a matrix $A \in \mathbb{S}^n$, consider the following maximization problem,

$$\max_{x \in \mathbb{R}^n} \; x^T A x \quad \text{subject to } \|x\|_2^2 = 1$$

i.e., we want to find the vector (of norm 1) which maximizes the quadratic form. Assuming the eigenvalues are ordered as $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, the optimal x for this optimization problem is x_1, the eigenvector corresponding to λ_1. In this case the maximal value of the quadratic form is λ_1. Similarly, the optimal solution to the minimization problem,

$$\min_{x \in \mathbb{R}^n} \; x^T A x \quad \text{subject to } \|x\|_2^2 = 1$$

is x_n, the eigenvector corresponding to λ_n, and the minimal value is λ_n. This can be proved by appealing to the eigenvector-eigenvalue form of A and the properties of orthogonal matrices. However, in the next section we will see a way of showing it directly using matrix calculus.

4 Matrix Calculus

While the topics in the previous sections are typically covered in a standard course on linear algebra, one topic that does not seem to be covered very often (and which we will use extensively) is the extension of calculus to the vector setting. Despite the fact that all the actual calculus we use is relatively trivial, the notation can often make things look much more difficult than they are. In this section we present some basic definitions of matrix calculus and provide a few examples.

4.1 The Gradient

Suppose that $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ is a function that takes as input a matrix A of size $m \times n$ and returns a real value. Then the ***gradient*** of f (with respect to $A \in \mathbb{R}^{m\times n}$) is the matrix of partial derivatives, defined as:

$$\nabla_A f(A) \in \mathbb{R}^{m\times n} = \begin{bmatrix} \frac{\partial f(A)}{\partial A_{11}} & \frac{\partial f(A)}{\partial A_{12}} & \cdots & \frac{\partial f(A)}{\partial A_{1n}} \\ \frac{\partial f(A)}{\partial A_{21}} & \frac{\partial f(A)}{\partial A_{22}} & \cdots & \frac{\partial f(A)}{\partial A_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f(A)}{\partial A_{m1}} & \frac{\partial f(A)}{\partial A_{m2}} & \cdots & \frac{\partial f(A)}{\partial A_{mn}} \end{bmatrix}$$

i.e., an $m \times n$ matrix with

$$(\nabla_A f(A))_{ij} = \frac{\partial f(A)}{\partial A_{ij}}.$$

Note that the size of $\nabla_A f(A)$ is always the same as the size of A. So if, in particular, A is just a vector $x \in \mathbb{R}^n$,

$$\nabla_x f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}.$$

It is very important to remember that the gradient of a function is *only* defined if the function is real-valued, that is, if it returns a scalar value. We can not, for example, take the gradient of $Ax, A \in \mathbb{R}^{n\times n}$ with respect to x, since this quantity is vector-valued.

It follows directly from the equivalent properties of partial derivatives that:

- $\nabla_x(f(x) + g(x)) = \nabla_x f(x) + \nabla_x g(x)$.
- For $t \in \mathbb{R}$, $\nabla_x(t\ f(x)) = t\nabla_x f(x)$.

It is a little bit trickier to determine what the proper expression is for $\nabla_x f(Ax)$, $A \in \mathbb{R}^{n\times n}$, but this is doable as well (if fact, you'll have to work this out for a homework problem).

4.2 The Hessian

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is a function that takes a vector in $\mathbb{R}^n$ and returns a real number. Then the ***Hessian*** matrix with respect to x, written $\nabla_x^2 f(x)$ or simply as H is the $n \times n$ matrix of partial derivatives,

$$\nabla_x^2 f(x) \in \mathbb{R}^{n\times n} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}.$$

In other words, $\nabla_x^2 f(x) \in \mathbb{R}^{n\times n}$, with

$$(\nabla_x^2 f(x))_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}.$$

Note that the Hessian is always symmetric, since

$$\frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \frac{\partial^2 f(x)}{\partial x_j \partial x_i}.$$

Similar to the gradient, the Hessian is defined only when $f(x)$ is real-valued.

It is natural to think of the gradient as the analogue of the first derivative for functions of vectors, and the Hessian as the analogue of the second derivative (and the symbols we use also suggest this relation). This intuition is generally correct, but there a few caveats to keep in mind.

First, for real-valued functions of one variable $f : \mathbb{R} \to \mathbb{R}$, it is a basic definition that the second derivative is the derivative of the first derivative, i.e.,

$$\frac{\partial^2 f(x)}{\partial x^2} = \frac{\partial}{\partial x}\frac{\partial}{\partial x} f(x).$$

However, for functions of a vector, the gradient of the function is a vector, and we cannot take the gradient of a vector — i.e.,

$$\nabla_x \nabla_x f(x) = \nabla_x \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_1} \end{bmatrix}$$

and this expression is not defined. Therefore, it is *not* the case that the Hessian is the gradient of the gradient. However, this is *almost* true, in the following sense: If we look at the ith entry of the gradient $(\nabla_x f(x))_i = \partial f(x)/\partial x_i$, and take the gradient with respect to x we get

$$\nabla_x \frac{\partial f(x)}{\partial x_i} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_i \partial x_1} \\ \frac{\partial^2 f(x)}{\partial x_i \partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_i \partial x_n} \end{bmatrix}$$

which is the ith column (or row) of the Hessian. Therefore,

$$\nabla_x^2 f(x) = \left[\begin{array}{cccc} \nabla_x (\nabla_x f(x))_1 & \nabla_x (\nabla_x f(x))_2 & \cdots & \nabla_x (\nabla_x f(x))_n \end{array} \right].$$

If we don't mind being a little bit sloppy we can say that (essentially) $\nabla_x^2 f(x) = \nabla_x (\nabla_x f(x))^T$, so long as we understand that this really means taking the gradient of each entry of $(\nabla_x f(x))^T$, not the gradient of the whole vector.

Finally, note that while we can take the gradient with respect to a matrix $A \in \mathbb{R}^n$, for the purposes of this class we will only consider taking the Hessian with respect to a vector $x \in \mathbb{R}^n$. This is simply a matter of convenience (and the fact that none of the calculations we do require us to find the Hessian with respect to a matrix), since the Hessian with respect to a matrix would have to represent all the partial derivatives $\partial^2 f(A)/(\partial A_{ij} \partial A_{k\ell})$, and it is rather cumbersome to represent this as a matrix.

4.3 Gradients and Hessians of Quadratic and Linear Functions

Now let's try to determine the gradient and Hessian matrices for a few simple functions. It should be noted that all the gradients given here are special cases of the gradients given in the CS229 lecture notes.

For $x \in \mathbb{R}^n$, let $f(x) = b^T x$ for some known vector $b \in \mathbb{R}^n$. Then

$$f(x) = \sum_{i=1}^{n} b_i x_i$$

so

$$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^{n} b_i x_i = b_k.$$

From this we can easily see that $\nabla_x b^T x = b$. This should be compared to the analogous situation in single variable calculus, where $\partial/(\partial x)\ ax = a$.

Now consider the quadratic function $f(x) = x^T A x$ for $A \in \mathbb{S}^n$. Remember that

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

so

$$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j = \sum_{i=1}^{n} A_{ik} x_i + \sum_{j=1}^{n} A_{kj} x_j = 2 \sum_{i=1}^{n} A_{ki} x_i$$

where the last equality follows since A is symmetric (which we can safely assume, since it is appearing in a quadratic form). Note that the kth entry of $\nabla_x f(x)$ is just the inner product of the kth row of A and x. Therefore, $\nabla_x x^T A x = 2Ax$. Again, this should remind you of the analogous fact in single-variable calculus, that $\partial/(\partial x)\ ax^2 = 2ax$.

Finally, lets look at the Hessian of the quadratic function $f(x) = x^T A x$ (it should be obvious that the Hessian of a linear function $b^T x$ is zero). This is even easier than determining the gradient of the function, since

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_\ell} = \frac{\partial^2}{\partial x_k \partial x_\ell} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j = A_{k\ell} + A_{\ell k} = 2A_{k\ell}.$$

Therefore, it should be clear that $\nabla_x^2 x^T A x = 2A$, which should be entirely expected (and again analogous to the single-variable fact that $\partial^2/(\partial x^2)\ ax^2 = 2a$).

To recap,

- $\nabla_x b^T x = b$
- $\nabla_x x^T A x = 2Ax$ (if A symmetric)
- $\nabla_x^2 x^T A x = 2A$ (if A symmetric)

4.4 Least Squares

Lets apply the equations we obtained in the last section to derive the least squares equations. Suppose we are given matrices $A \in \mathbb{R}^{m\times n}$ (for simplicity we assume A is full rank) and a vector $b \in \mathbb{R}^m$ such that $b \notin \mathcal{R}(A)$. In this situation we will not be able to find a vector $x \in \mathbb{R}^n$, such that $Ax = b$, so instead we want to find a vector x such that Ax is as close as possible to b, as measured by the square of the Euclidean norm $\|Ax - b\|_2^2$.

Using the fact that $\|x\|_2^2 = x^T x$, we have

$$\begin{aligned} \|Ax - b\|_2^2 &= (Ax - b)^T (Ax - b) \\ &= x^T A^T A x - 2b^T A x + b^T b \end{aligned}$$

Taking the gradient with respect to x we have, and using the properties we derived in the previous section

$$\begin{aligned} \nabla_x (x^T A^T A x - 2b^T A x + b^T b) &= \nabla_x x^T A^T A x - \nabla_x 2b^T A x + \nabla_x b^T b \\ &= 2A^T A x - 2A^T b \end{aligned}$$

Setting this last expression equal to zero and solving for x gives the normal equations

$$x = (A^T A)^{-1} A^T b$$

which is the same as what we derived in class.

4.5 Gradients of the Determinant

Now lets consider a situation where we find the gradient of a function with respect to a matrix, namely for $A \in \mathbb{R}^{n\times n}$, we want to find $\nabla_A |A|$. Recall from our discussion of determinants that

$$|A| = \sum_{i=1}^{n} (-1)^{i+j} A_{ij} |A_{\backslash i, \backslash j}| \quad \text{(for any } j \in 1, \ldots, n)$$

so

$$\frac{\partial}{\partial A_{k\ell}} |A| = \frac{\partial}{\partial A_{k\ell}} \sum_{i=1}^{n} (-1)^{i+j} A_{ij} |A_{\backslash i, \backslash j}| = (-1)^{k+\ell} |A_{\backslash k, \backslash \ell}| = (\mathrm{adj}(A))_{\ell k}.$$

From this it immediately follows from the properties of the adjoint that

$$\nabla_A |A| = (\mathrm{adj}(A))^T = |A| A^{-T}.$$

Now lets consider the function $f : \mathbb{S}^n_{++} \to \mathbb{R}$, $f(A) = \log |A|$. Note that we have to restrict the domain of f to be the positive definite matrices, since this ensures that $|A| > 0$, so that the log of $|A|$ is a real number. In this case we can use the chain rule (nothing fancy, just the ordinary chain rule from single-variable calculus) to see that

$$\frac{\partial \log |A|}{\partial A_{ij}} = \frac{\partial \log |A|}{\partial |A|} \frac{\partial |A|}{\partial A_{ij}} = \frac{1}{|A|} \frac{\partial |A|}{\partial A_{ij}}.$$

From this is should be obvious that

$$\nabla_A \log |A| = \frac{1}{|A|} \nabla_A |A| = A^{-1},$$

where we can drop the transpose in the last expression because A is symmetric. Note the similarity to the single-valued case, where $\partial/(\partial x) \log x = 1/x$.

4.6 Eigenvalues as Optimization

Finally, we use matrix calculus to solve an optimization problem in a way that leads directly to eigenvalue/eigenvector analysis. Consider the following, equality constrained optimization problem:

$$\max_{x \in \mathbb{R}^n} \; x^T A x \qquad \text{subject to } \|x\|_2^2 = 1$$

for a symmetric matrix $A \in \mathbb{S}^n$. A standard way of solving optimization problems with equality constraints is by forming the ***Lagrangian***, an objective function that includes the equality constraints.[2] The Lagrangian in this case can be given by

$$\mathcal{L}(x, \lambda) = x^T A x - \lambda x^T x$$

where λ is called the Lagrange multiplier associated with the equality constraint. It can be established that for x^* to be a optimal point to the problem, the gradient of the Lagrangian has to be zero at x^* (this is not the only condition, but it is required). That is,

$$\nabla_x \mathcal{L}(x, \lambda) = \nabla_x (x^T A x - \lambda x^T x) = 2A^T x - 2\lambda x = 0.$$

Notice that this is just the linear equation $Ax = \lambda x$. This shows that the only points which can possibly maximize (or minimize) $x^T A x$ assuming $x^T x = 1$ are the eigenvectors of A.

[2] Don't worry if you haven't seen Lagrangians before, as we will cover them in greater detail later in CS229.

Probability Theory Review for Machine Learning

Samuel Ieong

November 6, 2006

1 Basic Concepts

Broadly speaking, probability theory is the mathematical study of uncertainty. It plays a central role in machine learning, as the design of learning algorithms often relies on probabilistic assumption of the data. This set of notes attempts to cover some basic probability theory that serves as a background for the class.

1.1 Probability Space

When we speak about probability, we often refer to the probability of an *event* of uncertain nature taking place. For example, we speak about the probability of rain next Tuesday. Therefore, in order to discuss probability theory formally, we must first clarify what the possible events are to which we would like to attach probability.

Formally, a *probability space* is defined by the triple $(\Omega, \mathcal{F}, P)$, where

- Ω is the *space of possible outcomes* (or *outcome space*),
- $\mathcal{F} \subseteq 2^{\Omega}$ (the power set of Ω) is the *space of (measurable) events* (or *event space*),
- P is the *probability measure* (or *probability distribution*) that maps an event $E \in \mathcal{F}$ to a real value between 0 and 1 (think of P as a function).

Given the outcome space Ω, there is some restrictions as to what subset of 2^{Ω} can be considered an event space $\mathcal{F}$:

- The trivial event Ω and the empty event $\emptyset$ is in $\mathcal{F}$.
- The event space $\mathcal{F}$ is closed under (countable) union, i.e., if $\alpha, \beta \in \mathcal{F}$, then $\alpha \cup \beta \in \mathcal{F}$.
- The even space $\mathcal{F}$ is closed under complement, i.e., if $\alpha \in \mathcal{F}$, then $(\Omega \setminus \alpha) \in \mathcal{F}$.

Example 1. *Suppose we throw a (six-sided) dice. The space of possible outcomes* $\Omega = \{1, 2, 3, 4, 5, 6\}$. *We may decide that the events of interest is whether the dice throw is odd or even. This event space will be given by* $\mathcal{F} = \{\emptyset, \{1, 3, 5\}, \{2, 4, 6\}, \Omega\}$.

Note that when the outcome space Ω is finite, as in the previous example, we often take the event space $\mathcal{F}$ to be 2^{Ω}. This treatment is not fully general, but it is often sufficient for practical purposes. However, when the outcome space is infinite, we must be careful to define what the event space is.

Given an event space $\mathcal{F}$, the probability measure P must satisfy certain axioms.

- (non-negativity) For all $\alpha \in \mathcal{F}$, $P(\alpha) \geq 0$.
- (trivial event) $P(\Omega) = 1$.
- (additivity) For all $\alpha, \beta \in \mathcal{F}$ and $\alpha \cap \beta = \emptyset$, $P(\alpha \cup \beta) = P(\alpha) + P(\beta)$.

Example 2. *Returning to our dice example, suppose we now take the event space $\mathcal{F}$ to be 2^{Ω}. Further, we define a probability distribution P over $\mathcal{F}$ such that*

$$P(\{1\}) = P(\{2\}) = \cdots = P(\{6\}) = 1/6$$

then this distribution P completely specifies the probability of any given event happening (through the additivity axiom). For example, the probability of an even dice throw will be

$$P(\{2, 4, 6\}) = P(\{2\}) + P(\{4\}) + P(\{6\}) = 1/6 + 1/6 + 1/6 = 1/2$$

since each of these events are disjoint.

1.2 Random Variables

Random variables play an important role in probability theory. The most important fact about random variables is that they are **not** variables. They are actually **functions** that map outcomes (in the outcome space) to real values. In terms of notation, we usually denote random variables by a capital letter. Let's see an example.

Example 3. *Again, consider the process of throwing a dice. Let X be a random variable that depends on the outcome of the throw. A natural choice for X would be to map the outcome i to the value i, i.e., mapping the event of throwing an "one" to the value of 1. Note that we could have chosen some strange mappings too. For example, we could have a random variable Y that maps all outcomes to 0, which would be a very boring function, or a random variable Z that maps the outcome i to the value of 2^i if i is odd and the value of $-i$ if i is even, which would be quite strange indeed.*

In a sense, random variables allow us to abstract away from the formal notion of event space, as we can define random variables that capture the appropriate events. For example, consider the event space of odd or even dice throw in Example 1. We could have defined a random variable that takes on value 1 if outcome i is odd and 0 otherwise. These type of binary random variables are very common in practice, and are known as *indicator variables*, taking its name from its use to indicate whether a certain event has happened. So why did we introduce event space? That is because when one studies probability theory (more rigorously) using measure theory, the distinction between outcome space and event space will be very important. This topic is too advanced to be covered in this short review note. In any case, it is good to keep in mind that event space is not always simply the power set of the outcome space.

From here onwards, we will talk mostly about probability with respect to random variables. While some probability concepts can be defined meaningfully without using them, random variables allow us to provide a more uniform treatment of probability theory. For notations, the probability of a random variable X taking on the value of a will be denoted by either

$$P(X = a) \qquad \text{or} \qquad P_X(a)$$

We will also denote the range of a random variable X by $Val(X)$.

1.3 Distributions, Joint Distributions, and Marginal Distributions

We often speak about the *distribution* of a variable. This formally refers to the probability of a random variable taking on certain values. For example,

Example 4. *Let random variable X be defined on the outcome space Ω of a dice throw (again!). If the dice is fair, then the distribution of X would be*

$$P_X(1) = P_X(2) = \cdots = P_X(6) = 1/6$$

Note that while this example resembles that of Example 2, they have different semantic meaning. The probability distribution defined in Example 2 is over **events**, whereas the one here is defined over **random variables**.

For notation, we will use $P(X)$ to denote the distribution of the random variable X.

Sometimes, we speak about the distribution of more than one variables at a time. We call these distributions *joint distributions*, as the probability is determined jointly by all the variables involved. This is best clarified by an example.

Example 5. *Let X be a random variable defined on the outcome space of a dice throw. Let Y be an indicator variable that takes on value 1 if a coin flip turns up head and 0 if tail. Assuming both the dice and the coin are fair, the joint distribution of X and Y is given by*

P	$X = 1$	$X = 2$	$X = 3$	$X = 4$	$X = 5$	$X = 6$
$Y = 0$	1/12	1/12	1/12	1/12	1/12	1/12
$Y = 1$	1/12	1/12	1/12	1/12	1/12	1/12

As before, we will denote the probability of X taking value a and Y taking value b by either the long hand of $P(X = a, Y = b)$, or the short hand of $P_{X,Y}(a, b)$. We refer to their joint distribution by $P(X, Y)$.

Given a joint distribution, say over random variables X and Y, we can talk about the *marginal distribution* of X or that of Y. The marginal distribution refers to the probability distribution of a random variable on its own. To find out the marginal distribution of a random variable, we *sum out* all the other random variables from the distribution. Formally, we mean

$$P(X) = \sum_{b \in Val(Y)} P(X, Y = b) \tag{1}$$

The name of marginal distribution comes from the fact that if we add up all the entries of a row (or a column) of a joint distribution, and write the answer at the end (i.e., margin) of it, this will be the probability of the random variable taking on that value. Of course, thinking in this way only helps when the joint distribution involves two variables.

1.4 Conditional Distributions

Conditional distributions are one of the key tools in probability theory for reasoning about uncertainty. They specify the distribution of a random variable when the value of another random variable is known (or more generally, when some event is known to be true).

Formally, conditional probability of $X = a$ *given* $Y = b$ is defined as

$$P(X = a|Y = b) = \frac{P(X = a, Y = b)}{P(Y = b)} \tag{2}$$

Note that this is not defined when the probability of $Y = b$ is 0.

Example 6. *Suppose we know that a dice throw was odd, and want to know the probability of an "one" has been thrown. Let X be the random variable of the dice throw, and Y be an indicator variable that takes on the value of* 1 *if the dice throw turns up odd, then we write our desired probability as follows:*

$$P(X = 1|Y = 1) = \frac{P(X = 1, Y = 1)}{P(Y = 1)} = \frac{1/6}{1/2} = 1/3$$

The idea of conditional probability extends naturally to the case when the distribution of a random variable is conditioned on several variables, namely

$$P(X = a|Y = b, Z = c) = \frac{P(X = a, Y = b, Z = c)}{P(Y = b, Z = c)}$$

As for notations, we write $P(X|Y = b)$ to denote the distribution of random variable X when $Y = b$. We may also write $P(X|Y)$ to denote a set of distributions of X, one for each of the different values that Y can take.

1.5 Independence

In probability theory, *independence* means that the distribution of a random variable does *not* change on learning the value of another random variable. In machine learning, we often make such assumptions about our data. For example, the training samples are assumed to be drawn independently from some underlying space; the label of sample i is assumed to be independent of the features of sample j ($i \neq j$).

Mathematically, a random variable X is independent of Y when

$$P(X) = P(X|Y)$$

(Note that we have dropped what values X and Y are taking. This means the statement holds true for any values X and Y may take.)

Using Equation (2), it is easy to verify that if X is independent of Y, then Y is also independent of X. As a notation, we write $X \perp Y$ if X and Y are independent.

An equivalent mathematical statement about the independence of random variables X and Y is

$$P(X, Y) = P(X)P(Y)$$

Sometimes we also talk about *conditional independence*, meaning that if we know the value of a random variable (or more generally, a set of random variables), then some other random variables will be independent of each other. Formally, we say "X and Y are *conditionally* independent given Z" if

$$P(X|Z) = P(X|Y, Z)$$

or, equivalently,

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

An example of conditional independence that we will se in class is the *Naïve Bayes* assumption. This assumption is made in the context of a learning algorithm for learning to classify emails as spams or non-spams. It assumes that the probability of a word x appearing in the email is conditionally independent of a word y appearing given whether the email is spam or not. This clearly is not without loss of generality, as some words almost invariably comes in pair. However, as it turns out, making this simplifying assumption does not hurt the performance much, and in any case allow us to learn to classify spams rapidly. Details can be found in Lecture Notes 2.

1.6 Chain Rule and Bayes Rule

We now present two basic yet important rules for manipulating that relates joint distributions and conditional distributions. The first is known as the *Chain Rule*. It can be seen as a generalization of Equation (2) to multiple random variables.

Theorem 1 (Chain Rule)**.**

$$P(X_1, X_2, \ldots, X_n) = P(X_1)P(X_2|X_1) \cdots P(X_n|X_1, X_2, \ldots, X_{n-1}) \tag{3}$$

The Chain Rule is often used to evaluate the joint probability of some random variables, and is especially useful when there are (conditional) independence across variables. Notice there is a choice in the order we unravel the random variables when applying the Chain Rule; picking the right order can often make evaluating the probability much easier.

The second rule we are going to introduce is the *Bayes Rule*. The Bayes Rule allows us to compute the conditional probability $P(X|Y)$ from $P(Y|X)$, in a sense "inverting" the conditions. It can be derived simply from Equation (2) as well.

Theorem 2 (Bayes Rule)**.**

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)} \tag{4}$$

And recall that if $P(Y)$ is not given, we can always apply Equation (1) to find it.

$$P(Y) = \sum_{a \in Val(X)} P(X = a, Y) = \sum_{a \in Val(X)} P(Y|X = a)P(X = a)$$

This application of Equation (1) is sometimes referred to as the *law of total probability*.

Extending the Bayes Rule to the case of multiple random variables can sometimes be tricky. Just to be clear, we would give a few examples. When in doubt, one can always refer to how conditional probabilities are defined and work out the details.

Example 7. *Let's consider the following conditional probabilities:* $P(X, Y|Z)$ *and* $(X|Y, Z)$*.*

$$P(X, Y|Z) = \frac{P(Z|X, Y)P(X, Y)}{P(Z)} = \frac{P(Y, Z|X)P(X)}{P(Z)}$$
$$P(X|Y, Z) = \frac{P(Y|X, Z)P(X, Z)}{P(Y, Z)} = \frac{P(Y|X, Z)P(X|Z)P(Z)}{P(Y|Z)P(Z)} = \frac{P(Y|X, Z)P(X|Z)}{P(Y|Z)}$$

2 Defining a Probability Distribution

We have been talking about probability distributions for a while. But how do we define a distribution? In a broad sense, there are two classes of distribution that require seemingly different treatments (these can be unified using measure theory). Namely, *discrete* distributions and *continuous* distributions. We will discuss how distributions are specified next.

Note that this discussion is distinct from how we can efficiently *represent* a distribution. The topic of efficient representation of probability distribution is in fact a very important and active research area that deserves its own course. If you are interested to learn more about how to efficiently represent, reason, and perform learning on distributions, you are advised to take CS228: Probabilistic Models in Artificial Intelligence.

2.1 Discrete Distribution: Probability Mass Function

By a discrete distribution, we mean that the random variable of the underlying distribution can take on only *finitely many* different values (or that the outcome space is finite).

To define a discrete distribution, we can simply enumerate the probability of the random variable taking on each of the possible values. This enumeration is known as the *probability mass function*, as it divides up a unit mass (the total probability) and places them on the different values a random variable can take. This can be extended analogously to joint distributions and conditional distributions.

2.2 Continuous Distribution: Probability Density Function

By a continuous distribution, we mean that the random variable of the underlying distribution can take on *infinitely many* different values (or that the outcome space is infinite).

This is arguably a trickier situation than the discrete case, since if we place a non-zero amount of mass on each of the values, the total mass will add up to infinity, which violates the requirement that the total probaiblity must sum up to one.

To define a continuous distribution, we will make use of *probability density function* (PDF). A probability density function, f, is a *non-negative*, *integrable* function such that

$$\int_{Val(X)} f(x)dx = 1$$

The probability of a random variable X distributed according to a PDF f is computed as follows

$$P(a \leq X \leq b) = \int_a^b f(x)dx$$

Note that this, in particular, implies that the probability of a continuously distributed random variable taking on any given single value is zero.

Example 8 (Uniform distribution)**.** *Let's consider a random variable X that is uniformly distributed in the range $[0, 1]$. The corresponding PDF would be*

$$f(x) = \begin{cases} 1 & \textit{if } 0 \leq x \leq 1 \\ 0 & \textit{otherwise} \end{cases}$$

We can verify that $\int_0^1 1\ dx$ is indeed 1, and therefore f is a PDF. To compute the probability of X smaller than a half,

$$P(X \leq 1/2) = \int_0^{1/2} 1\ dx = [x]_0^{1/2} = 1/2$$

More generally, suppose X is distributed uniformly over the range $[a, b]$, then the PDF would be

$$f(x) = \begin{cases} \frac{1}{b-a} & \textit{if } a \leq x \leq b \\ 0 & \textit{otherwise} \end{cases}$$

Sometimes we will also speak about *cumulative distribution function.* It is a function that gives the probability of a random variable being smaller than some value. A cumulative distribution function F is related to the underlying probability density function f as follows:

$$F(b) = P(X \leq b) = \int_{-\infty}^{b} f(x)dx$$

and hence $F(x) = \int f(x)dx$ (in the sense of indefinite integral).

To extend the definition of continuous distribution to joint distribution, the probability density function is extended to take multiple arguments, namely,

$$P(a_1 \leq X_1 \leq b_1, a_2 \leq X_2 \leq b_2, \ldots, a_n \leq X_n \leq n_1) = \int_{a_1}^{b_1} \int_{a_2}^{b_2} \cdots \int_{a_n}^{b_n} f(x_1, x_2, \ldots, x_n) dx_1 dx_2 \ldots dx_n$$

To extend the definition of conditional distribution to continuous random variables, we ran into the problem that the probability of a continuous random variable taking on a single value is 0, so Equation (2) is not well defined, since the denominator equals 0. To define the conditional distribution of a continuous variable, let $f(x, y)$ be the joint distribution of X and Y. Through application of analysis, we can show that the PDF, $f(y|x)$, underlying the distribution $P(Y|X)$ is given by

$$f(y|x) = \frac{f(x, y)}{f(x)}$$

For example,

$$P(a \leq Y \leq b | X = c) = \int_{a}^{b} f(y|c)dy = \int_{a}^{b} \frac{f(c, y)}{f(c)} dy$$

3 Expectations and Variance

3.1 Expectations

One of the most common operations we perform on a random variable is to compute its *expectation*, also known as its *mean*, *expected value*, or *first moment*. The expectation of a random variable, denoted by $E(X)$, is given by

$$E(X) = \sum_{a \in Val(X)} aP(X = a) \qquad \text{or} \qquad E(X) = \int_{a \in Val(X)} xf(x)\, dx \tag{5}$$

Example 9. *Let X be the outcome of rolling a fair dice. The expectation of X is*

$$E(X) = (1)\frac{1}{6} + (2)\frac{1}{6} + \cdots + 6\frac{1}{6} = 3\frac{1}{2}$$

We may sometimes be interested in computing the expected value of some function f of a random variable X. Recall, however, that a random variable is also a function itself, so the easiest way to think about this is that we define a new random variable $Y = f(X)$, and compute the expected value of Y instead.

When working with indicator variables, a useful identify is the following:

$$E(X) = P(X = 1) \qquad \text{for indicator variable } X$$

When working with the sums of random variables, one of the most important rule is the *linearity of expectations.*

Theorem 3 (Linearity of Expectations)**.** *Let $X_1, X_2, \ldots, X_n$ be (possibly dependent) random variables,*

$$E(X_1 + X_2 + \cdots + X_n) = E(X_1) + E(X_2) + \cdots + E(X_n) \tag{6}$$

The linearity of expectations is very powerful because there are no restrictions on whether the random variables are independent or not. When we work on products of random variables, however, there is very little we can say in general. However, when the random variables are independent, then

Theorem 4. *Let X and Y be independent random variables,*

$$E(XY) = E(X)E(Y)$$

3.2 Variance

The *variance* of a distribution is a measure of the "spread" of a distribution. Sometimes it is also referred to as the *second moment.* It is defined as follows:

$$Var(X) = E\left((X - E(X))^2\right) \tag{7}$$

The variance of a random variable is often denoted by σ^2. The reason that this is squared is because we often want to find out σ, known as the *standard deviation.* The variance and the standard deviation is related (obviously) by $\sigma = \sqrt{Var(X)}$.

To find out the variance of a random variable X, it's often easier to compute the following instead

$$Var(X) = E(X^2) - (E(X))^2$$

Note that unlike expectation, variance is not a linear function of a random variable X. In fact, we can verify that the variance of $(aX + b)$ is

$$Var(aX + b) = a^2 Var(X)$$

If random variables X and Y are independent, then

$$Var(X + Y) = Var(X)Var(Y) \qquad \text{if } X \perp Y$$

Sometimes we also talk about the *covariance* of two random variables. This is a measure of how "closely related" two random variables are. Its definition is as follows.

$$Cov(X, Y) = E((X - E(X))(Y - E(Y)))$$

4 Some Important Distributions

In this section, we will review some of the probability distributions that we will see in this class. This is by no means a comprehensive list of distribution that one should know. In particular, distributions such as the geoemtric, hypergeometric, and binomial distributions, which are very useful in their own right and studied in introductory probability theory, are not reviewed here.

4.1 Bernoulli

The *Bernoulli distribution* is one of the most basic distribution. A random variable distributed according to the Bernoulli distribution can take on two possible values, $\{0, 1\}$. It can be specified by a single parameter p, and by convention we take p to be $P(X = 1)$. It is often used to indicate whether a trail is successful or not.

Sometimes it is useful to write the probability distribution of a Bernoulli random variable X as follows

$$P(X) = p^x(1-p)^{1-x}$$

An example of the Bernoulli distribution in action is the classification task in Lecture Notes 1. To develop the logistic regression algorithm for the task, we assume that the labels are distributed according to the Bernoulli distribution given the features.

4.2 Poisson

The *Poisson distribution* is a very useful distribution that deals with the arrival of events. It measures probaiblity of the number of events happening over a fixed period of time, given a fixed average rate of occurrence, and that the events take place independently of the time since the last event. It is parametrized by the average arrival rate λ. The probability mass function is given by:

$$P(X = k) = \frac{\exp(-\lambda)\lambda^k}{k!}$$

The mean value of a Poisson random variable is λ, and its variance is also λ.

We will get to work on a learning algorithm that deals with Poisson random variables in Homework 1, Problem 3.

4.3 Gaussian

The *Gaussian distribution*, also known as the *normal distribution*, is one of the most "versatile" distributions in probability theory, and appears in a wide variety of contexts. For example, it can be used to approximate the binomial distribution when the number of experiments is large, or the Poisson distribution when the average arrival rate is high. It is also related to the Law of Large Numbers. For many problems, we will also often assume that when noise in the system is Gaussian distributed. The list of applications is endless.

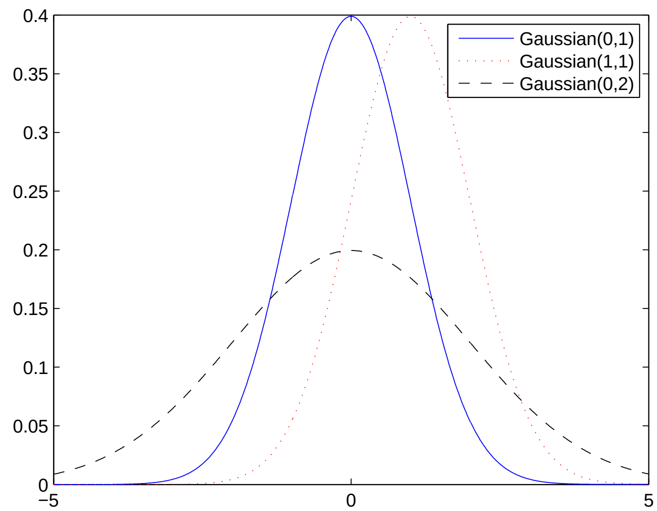


Figure 1: Gaussian distributions under different mean and variance

The Gaussian distribution is determined by two parameters: the mean μ and the variance σ^2. The probability density function is given by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{8}$$

To get a better sense of how the distribution changes with respect to the mean and the variance, we have plotted three different Gaussian distributions in Figure 1.

In our class, we will sometimes work with multi-variate Gaussian distributions. A k-dimensional multi-variate Gaussian distribution is parametrized by (μ, Σ), where μ is now a *vector* of means in $\mathbb{R}^k$, and Σ is the *covariance matrix* in $\mathbb{R}^{k\times k}$, in other words, $\Sigma_{ii} = Var(X_i)$ and $\Sigma_{ij} = Cov(X_i, X_j)$. The probability density function is now defined over vectors of input, given by

$$f(\mathbf{x}) = \frac{1}{\sqrt{2\pi^k|\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right) \tag{9}$$

(Recall that we denote the determinant of a matrix A by $|A|$, and its inverse by A^{-1})

To get a better sense of how a multi-variate Gaussian distribution depends on the covariance matrix, we can look at the figures in Lecture Notes 2, Pages 3—4.

Working with a multi-variate Gaussian distribution can be tricky and daunting at times. One way to make our lives easier, at least as a way to get intuition on a problem, is to assume that the covariances are zero when we first attempt a problem. When the covariances are zero, the determinant $|\Sigma|$ will simply be the product of the variances, and the inverse Σ^{-1} can be found by taking the inverse of the diagonal entries of Σ.

5 Working with Probabilities

As we will be working with probabilities and distributions a lot in this class, listed below are a few tips about efficient manipulation of distributions.

5.1 The log trick

In machine learning, we generally assume the independence of different samples. Therefore, we often have to deal with the product of a (large) number of distributions. When our goal is to optimize functions of such products, it is often easier if we first work with the logarithm of such functions. As the logarithmic function is a strictly increasing function, it will not distort where the maximum is located (although, most certainly, the maximum value of the function before and after taking logarithm will be different).

As an example, consider the likelihood function in Lecture Notes 1, Page 17.

$$L(\theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}}$$

I dare say this is a pretty mean-looking function. But by taking the logarithm of it, termed log-likelihood function, we have instead

$$\ell(\theta) = \log L(\theta) = \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

Not the world's prettiest function, but at least it's more manageable. We can now work on one term (i.e., one training sample) at a time, because they are summed together rather than multiplied together.

5.2 Delayed Normalization

Because probability has to sum up to one, we often have to deal with normalization, especially with continuous distribution. For example, for Gaussian distributions, the term outside of the exponent is to ensure that the integral of the PDF evaluates to one. When we are sure that the end product of some algebra will be a probability distribution, or when we are finding the optimum of some distributions, it's often easier to simply denote the normalization constant to be Z, and not worry about computing the normalization constant all the time.

5.3 Jenson's Inequality

Sometimes when we are evaluating the expectation of a function of a random variable, we may only need a bound rather than its exact value. In these situations, if the function is convex or concave, Jenson's inequality allows us to derive a bound by evaluating the value of the function at the expectation of the random variable itself.

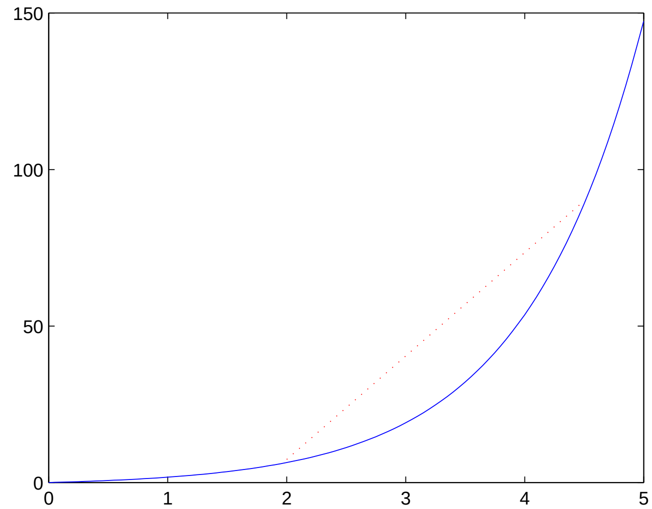


Figure 2: Illustration of Jenson's Inequality

Theorem 5 (Jenson's Inequality)**.** *Let X be a random variable, and f be a convex function. Then*

$$f(E(X)) \leq E(f(X))$$

If f is a concave function, then

$$f(E(X)) \geq E(f(X))$$

While we can show Jenson's inequality by algebra, it's easiest to understand it through a picture. The function in Figure 2 is a convex function. We can see that a straight line between any two points on the function always lie above the function. This shows that if a random variable can take on only two values, then Jenson's inequality holds. It is relatively straight forward to extend this to general random variables.

Convex Optimization Overview

Zico Kolter

October 19, 2007

1 Introduction

Many situations arise in machine learning where we would like to ***optimize*** the value of some function. That is, given a function $f : \mathbb{R}^n \to \mathbb{R}$, we want to find $x \in \mathbb{R}^n$ that minimizes (or maximizes) $f(x)$. We have already seen several examples of optimization problems in class: least-squares, logistic regression, and support vector machines can all be framed as optimization problems.

It turns out that in the general case, finding the global optimum of a function can be a very difficult task. However, for a special class of optimization problems, known as ***convex optimization problems***, we can efficiently find the global solution in many cases. Here, "efficiently" has both practical and theoretical connotations: it means that we can solve many real-world problems in a reasonable amount of time, and it means that theoretically we can solve problems in time that depends only *polynomially* on the problem size.

The goal of these section notes and the accompanying lecture is to give a very brief overview of the field of convex optimization. Much of the material here (including some of the figures) is heavily based on the book *Convex Optimization* [1] by Stephen Boyd and Lieven Vandenberghe (available for free online), and EE364, a class taught here at Stanford by Stephen Boyd. If you are interested in pursuing convex optimization further, these are both excellent resources.

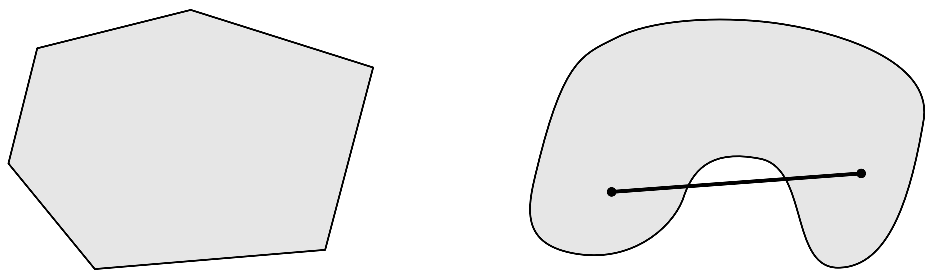
2 Convex Sets

We begin our look at convex optimization with the notion of a ***convex set***.

Definition 2.1 *A set C is convex if, for any $x, y \in C$ and $\theta \in \mathbb{R}$ with $0 \leq \theta \leq 1$,*

$$\theta x + (1 - \theta)y \in C.$$

Intuitively, this means that if we take any two elements in C, and draw a line segment between these two elements, then every point on that line segment also belongs to C. Figure 1 shows an example of one convex and one non-convex set. The point $\theta x + (1 - \theta)y$ is called a ***convex combination*** of the points x and y.



(a) (b)

Figure 1: Examples of a convex set (a) and a non-convex set (b).

2.1 Examples

- **All of $\mathbb{R}^n$.** It should be fairly obvious that given any $x, y \in \mathbb{R}^n$, $\theta x + (1-\theta)y \in \mathbb{R}^n$.

- **The non-negative orthant, $\mathbb{R}^n_+$.** The non-negative orthant consists of all vectors in $\mathbb{R}^n$ whose elements are all non-negative: $\mathbb{R}^n_+ = \{x : x_i \geq 0 \;\; \forall i = 1, \ldots, n\}$. To show that this is a convex set, simply note that given any $x, y \in \mathbb{R}^n_+$ and $0 \leq \theta \leq 1$,

$$(\theta x + (1-\theta)y)_i = \theta x_i + (1-\theta)y_i \geq 0 \;\; \forall i.$$

- **Norm balls.** Let $\|\cdot\|$ be some norm on $\mathbb{R}^n$ (e.g., the Euclidean norm, $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$). Then the set $\{x : \|x\| \leq 1\}$ is a convex set. To see this, suppose $x, y \in \mathbb{R}^n$, with $\|x\| \leq 1, \|y\| \leq 1$, and $0 \leq \theta \leq 1$. Then

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

where we used the triangle inequality and the positive homogeneity of norms.

- **Affine subspaces and polyhedra.** Given a matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$, an affine subspace is the set $\{x \in \mathbb{R}^n : Ax = b\}$ (note that this could possibly be empty if b is not in the range of A). Similarly, a polyhedron is the (again, possibly empty) set $\{x \in \mathbb{R}^n : Ax \preceq b\}$, where '$\preceq$' here denotes componentwise inequality (i.e., all the entries of Ax are less than or equal to their corresponding element in b).[1] To prove this, first consider $x, y \in \mathbb{R}^n$ such that $Ax = Ay = b$. Then for $0 \leq \theta \leq 1$,

$$A(\theta x + (1-\theta)y) = \theta Ax + (1-\theta)Ay = \theta b + (1-\theta)b = b.$$

Similarly, for $x, y \in \mathbb{R}^n$ that satisfy $Ax \leq b$ and $Ay \leq b$ and $0 \leq \theta \leq 1$,

$$A(\theta x + (1-\theta)y) = \theta Ax + (1-\theta)Ay \leq \theta b + (1-\theta)b = b.$$

[1]Similarly, for two vectors $x, y \in \mathbb{R}^n$, $x \succeq y$ denotes that each element of X is greater than or equal to the corresponding element in b. Note that sometimes '$\leq$' and '$\geq$' are used in place of '$\preceq$' and '$\succeq$'; the meaning must be determined contextually (i.e., both sides of the inequality will be vectors).

- **Intersections of convex sets.** Suppose $C_1, C_2, \ldots, C_k$ are convex sets. Then their intersection

$$\bigcap_{i=1}^{k} C_i = \{x : x \in C_i \ \ \forall i = 1, \ldots, k\}$$

is also a convex set. To see this, consider $x, y \in \bigcap_{i=1}^{k} C_i$ and $0 \leq \theta \leq 1$. Then,

$$\theta x + (1-\theta)y \in C_i \ \ \forall i = 1, \ldots, k$$

by the definition of a convex set. Therefore

$$\theta x + (1-\theta)y \in \bigcap_{i=1}^{k} C_i.$$

Note, however, that the *union* of convex sets in general will not be convex.

- **Positive semidefinite matrices.** The set of all symmetric positive semidefinite matrices, often times called the *positive semidefinite cone* and denoted $\mathbb{S}_+^n$, is a convex set (in general, $\mathbb{S}^n \subset \mathbb{R}^{n\times n}$ denotes the set of symmetric $n \times n$ matrices). Recall that a matrix $A \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite if and only if $A = A^T$ and for all $x \in \mathbb{R}^n$, $x^T Ax \geq 0$. Now consider two symmetric positive semidefinite matrices $A, B \in \mathbb{S}_+^n$ and $0 \leq \theta \leq 1$. Then for any $x \in \mathbb{R}^n$,

$$x^T(\theta A + (1-\theta)B)x = \theta x^T Ax + (1-\theta)x^T Bx \geq 0.$$

The same logic can be used to show that the sets of all positive definite, negative definite, and negative semidefinite matrices are each also convex.

3 Convex Functions

A central element in convex optimization is the notion of a ***convex function***.

Definition 3.1 *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its domain (denoted $\mathcal{D}(f)$) is a convex set, and if, for all $x, y \in \mathcal{D}(f)$ and $\theta \in \mathbb{R}$, $0 \leq \theta \leq 1$,*

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

Intuitively, the way to think about this definition is that if we pick any two points on the graph of a convex function and draw a straight line between then, then the portion of the function between these two points will lie below this straight line. This situation is pictured in Figure 2.[2]

We say a function is ***strictly convex*** if Definition 3.1 holds with strict inequality for $x \neq y$ and $0 < \theta < 1$. We say that f is ***concave*** if $-f$ is convex, and likewise that f is ***strictly concave*** if $-f$ is strictly convex.

[2]Don't worry too much about the requirement that the domain of f be a convex set. This is just a technicality to ensure that $f(\theta x + (1-\theta)y)$ is actually defined (if $\mathcal{D}(f)$ were not convex, then it could be that $f(\theta x + (1-\theta)y)$ is undefined even though $x, y \in \mathcal{D}(f)$).

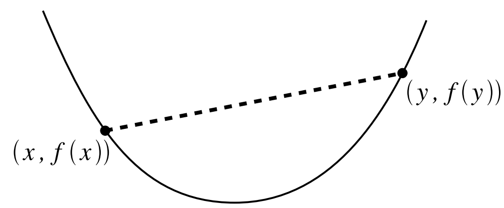


Figure 2: Graph of a convex function. By the definition of convex functions, the line connecting two points on the graph must lie above the function.

3.1 First Order Condition for Convexity

Suppose a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable (i.e., the gradient[3] $\nabla_x f(x)$ exists at all points x in the domain of f). Then f is convex if and only if $\mathcal{D}(f)$ is a convex set and for all $x, y \in \mathcal{D}(f)$,

$$f(y) \geq f(x) + \nabla_x f(x)^T (y - x).$$

The function $f(x) + \nabla_x f(x)^T (y - x)$ is called the ***first-order approximation*** to the function f at the point x. Intuitively, this can be thought of as approximating f with its tangent line at the point x. The first order condition for convexity says that f is convex if and only if the tangent line is a global underestimator of the function f. In other words, if we take our function and draw a tangent line at any point, then every point on this line will lie below the corresponding point on f.

Similar to the definition of convexity, f will be strictly convex if this holds with strict inequality, concave if the inequality is reversed, and strictly concave if the reverse inequality is strict.

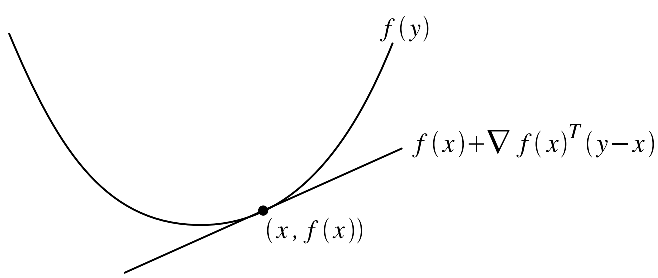


Figure 3: Illustration of the first-order condition for convexity.

[3]Recall that the gradient is defined as $\nabla_x f(x) \in \mathbb{R}^n, (\nabla_x f(x))_i = \frac{\partial f(x)}{\partial x_i}$. For a review on gradients and Hessians, see the previous section notes on linear algebra.

3.2 Second Order Condition for Convexity

Suppose a function $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable (i.e., the Hessian[4] $\nabla_x^2 f(x)$ is defined for all points x in the domain of f). Then f is convex if and only if $\mathcal{D}(f)$ is a convex set and its Hessian is positive semidefinite: i.e., for any $x \in \mathcal{D}(f)$,

$$\nabla_x^2 f(x) \succeq 0.$$

Here, the notation '$\succeq$' when used in conjunction with matrices refers to positive semidefiniteness, rather than componentwise inequality. [5] In one dimension, this is equivalent to the condition that the second derivative $f''(x)$ always be positive (i.e., the function always has positive curvature).

Again analogous to both the definition and first order conditions for convexity, f is strictly convex if its Hessian is positive definite, concave if the Hessian is negative semidefinite, and strictly concave if the Hessian is negative definite.

3.3 Jensen's Inequality

Suppose we start with the inequality in the basic definition of a convex function

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) \quad \text{for} \quad 0 \leq \theta \leq 1.$$

Using induction, this can be fairly easily extended to convex combinations of more than one point,

$$f\left(\sum_{i=1}^k \theta_i x_i\right) \leq \sum_{i=1}^k \theta_i f(x_i) \quad \text{for} \quad \sum_{i=1}^k \theta_i = 1,\ \theta_i \geq 0 \ \ \forall i.$$

In fact, this can also be extended to infinite sums or integrals. In the latter case, the inequality can be written as

$$f\left(\int p(x) x dx\right) \leq \int p(x) f(x) dx \quad \text{for} \quad \int p(x) dx = 1,\ p(x) \geq 0 \ \ \forall x.$$

Because $p(x)$ integrates to 1, it is common to consider it as a probability density, in which case the previous equation can be written in terms of expectations,

$$f(\mathbf{E}[x]) \leq \mathbf{E}[f(x)].$$

This last inequality is known as *Jensen's inequality*, and it will come up later in class.[6]

---

[4]Recall the Hessian is defined as $\nabla_x^2 f(x) \in \mathbb{R}^{n\times n}$, $(\nabla_x^2 f(x))_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$

[5]Similarly, for a symmetric matrix $X \in \mathbb{S}^n$, $X \preceq 0$ denotes that X is negative semidefinite. As with vector inequalities, '$\leq$' and '$\geq$' are sometimes used in place of '$\preceq$' and '$\succeq$'. Despite their notational similarity to vector inequalities, these concepts are very different; in particular, $X \succeq 0$ does not imply that $X_{ij} \geq 0$ for all i and j.

[6]In fact, all four of these equations are sometimes referred to as Jensen's inequality, due to the fact that they are all equivalent. However, for this class we will use the term to refer specifically to the last inequality presented here.

3.4 Sublevel Sets

Convex functions give rise to a particularly important type of convex set called an ***α-sublevel set***. Given a convex function $f : \mathbb{R}^n \to R$ and a real number $\alpha \in \mathbb{R}$, the α-sublevel set is defined as

$$\{x \in \mathcal{D}(f) : f(x) \leq \alpha\}.$$

In other words, the α-sublevel set is the set of all points x such that $f(x) \leq \alpha$.

To show that this is a convex set, consider any $x, y \in \mathcal{D}(f)$ such that $f(x) \leq \alpha$ and $f(y) \leq \alpha$. Then

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

3.5 Examples

We begin with a few simple examples of convex functions of one variable, then move on to multivariate functions.

- **Exponential.** Let $f : \mathbb{R} \to \mathbb{R}$, $f(x) = e^{ax}$ for any $a \in \mathbb{R}$. To show f is convex, we can simply take the second derivative $f''(x) = a^2 e^{ax}$, which is positive for all x.

- **Negative logarithm.** Let $f : \mathbb{R} \to \mathbb{R}$, $f(x) = -\log x$ with domain $\mathcal{D}(f) = \mathbb{R}_{++}$ (here, $\mathbb{R}_{++}$ denotes the set of strictly positive real numbers, $\{x : x > 0\}$). Then $f''(x) = 1/x^2 > 0$ for all x.

- **Affine functions.** Let $f : \mathbb{R}^n \to \mathbb{R}$, $f(x) = b^T x + c$ for some $b \in \mathbb{R}^n$, $c \in \mathbb{R}$. In this case the Hessian, $\nabla_x^2 f(x) = 0$ for all x. Because the zero matrix is both positive semidefinite and negative semidefinite, f is both convex and concave. In fact, affine functions of this form are the *only* functions that are both convex and concave.

- **Quadratic functions.** Let $f : \mathbb{R}^n \to \mathbb{R}$, $f(x) = \frac{1}{2}x^T A x + b^T x + c$ for a symmetric matrix $A \in \mathbb{S}^n$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$. In our previous section notes on linear algebra, we showed the Hessian for this function is given by

  $$\nabla_x^2 f(x) = A.$$

  Therefore, the convexity or non-convexity of f is determined entirely by whether or not A is positive semidefinite: if A is positive semidefinite then the function is convex (and analogously for strictly convex, concave, strictly concave). If A is indefinite then f is neither convex nor concave.

  Note that the squared Euclidean norm $f(x) = \|x\|_2^2 = x^T x$ is a special case of quadratic functions where $A = I$, $b = 0$, $c = 0$, so it is therefore a strictly convex function.

- **Norms.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be some norm on $\mathbb{R}^n$. Then by the triangle inequality and positive homogeneity of norms, for $x, y \in \mathbb{R}^n$, $0 \leq \theta \leq 1$,

$$f(\theta x + (1-\theta)y) \leq f(\theta x) + f((1-\theta)y) = \theta f(x) + (1-\theta)f(y).$$

This is an example of a convex function where it is *not* possible to prove convexity based on the second or first order conditions, because norms are not generally differentiable everywhere (e.g., the 1-norm, $||x||_1 = \sum_{i=1}^n |x_i|$, is non-differentiable at all points where any x_i is equal to zero).

- **Nonnegative weighted sums of convex functions.** Let $f_1, f_2, \ldots, f_k$ be convex functions and $w_1, w_2, \ldots, w_k$ be nonnegative real numbers. Then

$$f(x) = \sum_{i=1}^k w_i f_i(x)$$

is a convex function, since

$$\begin{aligned} f(\theta x + (1-\theta)y) &= \sum_{i=1}^k w_i f_i(\theta x + (1-\theta)y) \\ &\leq \sum_{i=1}^k w_i(\theta f_i(x) + (1-\theta)f_i(y)) \\ &= \theta \sum_{i=1}^k w_i f_i(x) + (1-\theta)\sum_{i=1}^k w_i f_i(y) \\ &= \theta f(x) + (1-\theta)f(x). \end{aligned}$$

4 Convex Optimization Problems

Armed with the definitions of convex functions and sets, we are now equipped to consider ***convex optimization problems***. Formally, a convex optimization problem in an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in C \end{array}$$

where f is a convex function, C is a convex set, and x is the optimization variable. However, since this can be a little bit vague, we often write it often written as

$$\begin{array}{lll} \text{minimize} & f(x) & \\ \text{subject to} & g_i(x) \leq 0, & i = 1, \ldots, m \\ & h_i(x) = 0, & i = 1, \ldots, p \end{array}$$

where f is a convex function, g_i are convex functions, and h_i are affine functions, and x is the optimization variable.

Is it imporant to note the direction of these inequalities: a convex function g_i must be *less* than zero. This is because the 0-sublevel set of g_i is a convex set, so the feasible region, which is the intersection of many convex sets, is also convex (recall that affine subspaces are convex sets as well). If we were to require that $g_i \geq 0$ for some convex g_i, the feasible region would no longer be a convex set, and the algorithms we apply for solving these problems would not longer be guaranteed to find the global optimum. Also notice that only affine functions are allowed to be equality constraints. Intuitively, you can think of this as being due to the fact that an equality constraint is equivalent to the two inequalities $h_i \leq 0$ and $h_i \geq 0$. However, these will both be valid constraints if and only if h_i is both convex and concave, i.e., h_i must be affine.

The ***optimal value*** of an optimization problem is denoted $p^\star$ (or sometimes $f^\star$) and is equal to the minimum possible value of the objective function in the feasible region[7]

$$p^\star = \min\{f(x) : g_i(x) \leq 0, i = 1, \ldots, m, h_i(x) = 0, i = 1, \ldots, p\}.$$

We allow $p^\star$ to take on the values $+\infty$ and $-\infty$ when the problem is either *infeasible* (the feasible region is empty) or *unbounded below* (there exists feasible points such that $f(x) \to -\infty$), respectively. We say that $x^\star$ is an ***optimal point*** if $f(x^\star) = p^\star$. Note that there can be more than one optimal point, even when the optimal value is finite.

4.1 Global Optimality in Convex Problems

Before stating the result of global optimality in convex problems, let us formally define the concepts of local optima and global optima. Intuitively, a feasible point is called ***locally optimal*** if there are no "nearby" feasible points that have a lower objective value. Similarly, a feasible point is called ***globally optimal*** if there are no feasible points at all that have a lower objective value. To formalize this a little bit more, we give the following two definitions.

Definition 4.1 *A point x is locally optimal if it is feasible (i.e., it satisfies the constraints of the optimization problem) and if there exists some $R > 0$ such that all feasible points z with $\|x - z\|_2 \leq R$, satisfy $f(x) \leq f(z)$.*

Definition 4.2 *A point x is globally optimal if it is feasible and for all feasible points z, $f(x) \leq f(z)$.*

We now come to the crucial element of convex optimization problems, from which they derive most of their utility. The key idea is that ***for a convex optimization problem all locally optimal points are globally optimal***.

Let's give a quick proof of this property by contradiction. Suppose that x is a locally optimal point which is not globally optimal, i.e., there exists a feasible point y such that

[7] Math majors might note that the min appearing below should more correctly be an inf. We won't worry about such technicalities here, and use min for simplicity.

$f(x) > f(y)$. By the definition of local optimality, there exist no feasible points z such that $\|x - z\|_2 \le R$ and $f(z) < f(x)$. But now suppose we choose the point

$$z = \theta y + (1-\theta)x \quad \text{with} \quad \theta = \frac{R}{2\|x-y\|_2}.$$

Then

$$\begin{aligned}
\|x - z\|_2 &= \left\| x - \left(\frac{R}{2\|x-y\|_2} y + \left(1 - \frac{R}{2\|x-y\|_2}\right) x \right) \right\|_2 \\
&= \left\| \frac{R}{2\|x-y\|_2}(x-y) \right\|_2 \\
&= R/2 \le R.
\end{aligned}$$

In addition, by the convexity of f we have

$$f(z) = f(\theta y + (1-\theta)x) \le \theta f(y) + (1-\theta) f(x) < f(x).$$

Furthermore, since the feasible set is a convex set, and since x and y are both feasible $z = \theta y + (1-\theta)$ will be feasible as well. Therefore, z is a feasible point, with $\|x - z\|_2 < R$ and $f(z) < f(x)$. This contradicts our assumption, showing that x cannot be locally optimal.

4.2 Special Cases of Convex Problems

For a variety of reasons, it is often times convenient to consider special cases of the general convex programming formulation. For these special cases we can often devise extremely efficient algorithms that can solve very large problems, and because of this you will probably see these special cases referred to any time people use convex optimization techniques.

- **Linear Programming.** We say that a convex optimization problem is a ***linear program*** (LP) if both the objective function f and inequality constraints g_i are affine functions. In other words, these problems have the form

$$\begin{array}{rl}
\text{minimize} & c^T x + d \\
\text{subject to} & Gx \preceq h \\
& Ax = b
\end{array}$$

  where $x \in \mathbb{R}^n$ is the optimization variable, $c \in \mathbb{R}^n$, $d \in \mathbb{R}$, $G \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$ are defined by the problem, and '$\preceq$' denotes elementwise inequality.

- **Quadratic Programming.** We say that a convex optimization problem is a ***quadratic program*** (QP) if the inequality constraints g_i are still all affine, but if the objective function f is a convex quadratic function. In other words, these problems have the form,

$$\begin{array}{rl}
\text{minimize} & \frac{1}{2} x^T P x + c^T x + d \\
\text{subject to} & Gx \preceq h \\
& Ax = b
\end{array}$$

where again $x \in \mathbb{R}^n$ is the optimization variable, $c \in \mathbb{R}^n$, $d \in \mathbb{R}$, $G \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$ are defined by the problem, but we also have $P \in \mathbb{S}^n_+$, a symmetric positive semidefinite matrix.

- **Quadratically Constrained Quadratic Programming.** We say that a convex optimization problem is a ***quadratically constrained quadratic program*** (QCQP) if both the objective f and the inequality constraints g_i are convex quadratic functions,

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^T P x + c^T x + d \\ \text{subject to} & \frac{1}{2}x^T Q_i x + r_i^T x + s_i \leq 0, \quad i = 1, \ldots, m \\ & Ax = b \end{array}$$

  where, as before, $x \in \mathbb{R}^n$ is the optimization variable, $c \in \mathbb{R}^n$, $d \in \mathbb{R}$, $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$, $P \in \mathbb{S}^n_+$, but we also have $Q_i \in \mathbb{S}^n_+$, $r_i \in \mathbb{R}^n$, $s_i \in \mathbb{R}$, for $i = 1, \ldots, m$.

- **Semidefinite Programming.** This last example is a bit more complex than the previous ones, so don't worry if it doesn't make much sense at first. However, semidefinite programming is become more and more prevalent in many different areas of machine learning research, so you might encounter these at some point, and it is good to have an idea of what they are. We say that a convex optimization problem is a ***semidefinite program*** (SDP) if it is of the form

$$\begin{array}{ll} \text{minimize} & \mathrm{tr}(CX) \\ \text{subject to} & \mathrm{tr}(A_i X) = b_i, \quad i = 1, \ldots, p \\ & X \succeq 0 \end{array}$$

  where the symmetric matrix $X \in \mathbb{S}^n$ is the optimization variable, the symmetric matrices $C, A_1, \ldots, A_p \in \mathbb{S}^n$ are defined by the problem, and the constraint $X \succeq 0$ means that we are constraining X to be positive semidefinite. This looks a bit different than the problems we have seen previously, since the optimization variable is now a matrix instead of a vector. If you are curious as to why such a formulation might be useful, you should look into a more advanced course or book on convex optimization.

It should be fairly obvious from the definitions that quadratic programs are more general than linear programs (since a linear program is just a special case of a quadratic program where $P = 0$), and likewise that quadratically constrained quadratic programs are more general than quadratic programs. However, what is not obvious at all is that semidefinite programs are in fact more general than all the previous types. That is, any quadratically constrained quadratic program (and hence any quadratic program or linear program) can be expressed as a semidefinte program. We won't discuss this relationship further in this document, but this might give you just a small idea as to why semidefinite programming could be useful.

4.3 Examples

Now that we've covered plenty of the boring math and formalisms behind convex optimization, we can finally get to the fun part: using these techniques to solve actual problems. We've already encountered a few such optimization problems in class, and in nearly every field, there is a good chance that someone has tried to apply convex optimization to solve some problem.

- **Support Vector Machines.** One of the most prevalent applications of convex optimization methods in machine learning is the support vector machine classifier. As discussed in class, finding the support vector classifier (in the case with slack variables) can be formulated as the optimization problem

$$\begin{array}{rll} \text{minimize} & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^m \xi_i & \\ \text{subject to} & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, & i = 1, \ldots, m \\ & \xi_i \geq 0, & i = 1, \ldots, m \end{array}$$

with optimization variables $w \in \mathbb{R}^n$, $\xi \in \mathbb{R}^m$, $b \in \mathbb{R}$, and where $C \in \mathbb{R}$ and $x^{(i)}, y^{(i)}, i = 1, \ldots m$ are defined by the problem. This is an example of a quadratic program, which we try to put the problem into the form described in the previous section. In particular, if define $k = m + n + 1$, let the optimization variable be

$$x \in \mathbb{R}^k \equiv \begin{bmatrix} w \\ \xi \\ b \end{bmatrix}$$

and define the matrices

$$P \in \mathbb{R}^{k \times k} = \begin{bmatrix} I & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad c \in \mathbb{R}^k = \begin{bmatrix} 0 \\ C \cdot \mathbf{1} \\ 0 \end{bmatrix},$$

$$G \in \mathbb{R}^{2m \times k} = \begin{bmatrix} -\mathrm{diag}(y)X & -I & -y \\ 0 & -I & 0 \end{bmatrix}, \quad h \in \mathbb{R}^{2m} = \begin{bmatrix} -\mathbf{1} \\ 0 \end{bmatrix}$$

where I is the identity, $\mathbf{1}$ is the vector of all ones, and X and y are defined as in class,

$$X \in \mathbb{R}^{m \times n} = \begin{bmatrix} x^{(1)T} \\ x^{(2)T} \\ \vdots \\ x^{(m)T} \end{bmatrix}, \quad y \in \mathbb{R}^m = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}.$$

You should try to convince yourself that the quadratic program described in the previous section, when using these matrices defined above, is equivalent to the SVM optimization problem. In reality, it is fairly easy to see that there the SVM optimization problem has a quadratic objective and linear constraints, so we typically don't need to put it into standard form to "prove" that it is a QP, and would only do so if we are using an off-the-shelf solver that requires the input to be in standard form.

- **Constrained least squares.** In class we have also considered the least squares problem, where we want to minimize $\|Ax - b\|_2^2$ for some matrix $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. As we saw, this particular problem can actually be solved analytically via the normal equations. However, suppose that we also want to constrain the entries in the solution x to lie within some predefined ranges. In other words, suppose we weanted to solve the optimization problem,

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|Ax - b\|_2^2 \\ \text{subject to} & l \preceq x \preceq u \end{array}$$

  with optimization variable x and problem data $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $l \in \mathbb{R}^n$, and $u \in \mathbb{R}^n$. This might seem like a fairly simple additional constraint, but it turns out that there will no longer be an analytical solution. However, you should be able to convince yourself that this optimization problem is a quadratic program, with matrices defined by

$$P \in \mathbb{R}^{n\times n} = \frac{1}{2}A^TA, \;\; c \in \mathbb{R}^n = -b^TA, \;\; d \in \mathbb{R} = \frac{1}{2}b^Tb,$$
$$G \in \mathbb{R}^{2n\times 2n} = \begin{bmatrix} -I & 0 \\ 0 & I \end{bmatrix}, \;\; h \in \mathbb{R}^{2n} = \begin{bmatrix} -l \\ u \end{bmatrix}.$$

- **Maximum Likelihood for Logistic Regression.** For homework one, you were required to show that the log-likelihood of the data in a logistic model was concave. This log likehood under such a model is

$$\ell(\theta) = \sum_{i=1}^{n} \left\{ y^{(i)} \ln g(\theta^T x^{(i)}) + (1 - y^{(i)}) \ln(1 - g(\theta^T x^{(i)})) \right\}$$

  where $g(z)$ denotes the logistic function $g(z) = 1/(1 + e^{-z})$. Finding the maximum likelihood estimate is then a task of maximizing the log-likelihood (or equivalently, minimizing the negative log-likelihood, a convex function), i.e.,

$$\text{minimize} \quad -\ell(\theta)$$

  with optimization variable $\theta \in \mathbb{R}^n$ and no constraints.

  Unlike the previous two examples, it turns out that it is not so easy to put this problem into a "standard" form optimization problem. Nevertheless, you've seen on the homework that the fact that ℓ is a concave function means that you can very efficiently find the global solution using an algorithm such as Newton's method.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge UP, 2004. Online: `http://www.stanford.edu/~boyd/cvxbook/`

Convex Optimization Overview (cnt'd)

Chuong B. Do

October 26, 2007

1 Recap

During last week's section, we began our study of ***convex optimization***, the study of mathematical optimization problems of the form,

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n}{\text{minimize}} & f(x) \\ \text{subject to} & g_i(x) \leq 0, \quad i = 1, \ldots, m, \\ & h_i(x) = 0, \quad i = 1, \ldots, p, \end{array} \tag{1}$$

where $x \in \mathbb{R}^n$ is the optimization variable, $f : \mathbb{R}^n \to \mathbb{R}$ and $g_i : \mathbb{R}^n \to \mathbb{R}$ are convex functions, and $h_i : \mathbb{R}^n \to \mathbb{R}$ are affine functions. In a convex optimization problem, the convexity of both the objective function f and the feasible region (i.e., the set of x's satisfying all constraints) allows us to conclude that any feasible locally optimal point must also be globally optimal. This fact provides the key intuition for why convex optimization problems can in general be solved efficiently.

In these lecture notes, we continue our foray into the field of convex optimization. In particular, we will introduce the theory of Lagrange duality for convex optimization problems with inequality and equality constraints. We will also discuss generic yet efficient algorithms for solving convex optimization problems, and then briefly mention directions for further exploration.

2 Duality

To explain the fundamental ideas behind duality theory, we start with a motivating example based on CS 229 homework grading. We prove a simple weak duality result in this setting, and then relate it to duality in optimization. We then discuss strong duality and the KKT optimality conditions.

2.1 A motivating example: CS 229 homework grading

In CS 229, students must complete four homeworks throughout the quarter, each consisting of five questions apiece. Suppose that during one year that the course is offered, the TAs decide to economize on their work load for the quarter by grading only one problem on each submitted problem set. Nevertheless, they also require that every student submit an attempted solution to every problem (a requirement which, if violated, would lead to automatic failure of the course).

Because they are extremely cold-hearted[1], the TAs always try to ensure that the students lose as many points as possible; if the TAs grade a problem that the student did not attempt, the number of points lost is set to $+\infty$ to denote automatic failure in the course. Conversely, each student in the course seeks to minimize the number of points lost on his or her assignments, and thus must decide on a strategy—i.e., an allocation of time to problems—that minimizes the number of points lost on the assignment.

The struggle between student and TAs can be summarized in a matrix $A = (a_{ij}) \in \mathbb{R}^{n\times m}$, whose columns correspond to different problems that the TAs might grade, and whose rows correspond to different strategies for time allocation that the student might use for the problem set. For example, consider the following matrix,

$$A = \begin{bmatrix} 5 & 5 & 5 & 5 & 5 \\ 8 & 8 & 1 & 8 & 8 \\ +\infty & +\infty & +\infty & 0 & +\infty \end{bmatrix},$$

Here, the student must decide between three strategies (corresponding to the three rows of the matrix, A):

- $i = 1$: she invests an equal effort into all five problems and hence loses at most 5 points on each problem,
- $i = 2$: she invests more time into problem 3 than the other four problems, and
- $i = 3$: she skips four problems in order to guarantee no points lost on problem 4.

Similarly, the TAs must decide between five strategies ($j \in \{1, 2, 3, 4, 5\}$) corresponding to the choice of problem graded.

If the student is forced to submit the homework without knowing the TAs choice of problem to be graded, and if the TAs are allowed to decide which problem to grade after having seen the student's problem set, then the number of points she loses will be:

$$p^* = \min_i \max_j a_{ij} \qquad (= 5 \text{ in the example above}) \tag{P}$$

where the order of the minimization and maximization reflect that for each fixed student time allocation strategy i, the TAs will have the opportunity to choose the worst scoring problem $\max_j a_{ij}$ to grade. However, if the TAs announce beforehand which homework problem will be graded, then the the number of points lost will be:

$$d^* = \max_j \min_i a_{ij} \qquad (= 0 \text{ in the example above}) \tag{D}$$

where this time, for each possible announced homework problem j to be graded, the student will have the opportunity to choose the optimal time allocation strategy, $\min_i a_{ij}$, which loses

[1]Clearly, this is a fictional example. The CS 229 TAs want you to succeed. Really, we do.

her the fewest points. Here, (P) is called the ***primal*** optimization problem whereas (D) is called the ***dual*** optimization problem. Rows containing $+\infty$ values correspond to strategies where the student has flagrantly violated the TAs demand that all problems be attempted; for reasons, which will become clear later, we refer to these rows as being ***primal-infeasible***.

In the example, the value of the dual problem is lower than that of the primal problem, i.e., $d^* = 0 < 5 = p^*$. This intuitively makes sense: the second player in this adversarial game has the advantage of knowing his/her opponent's strategy. This principle, however, holds more generally:

Theorem 2.1 (Weak duality)**.** *For any matrix $A = (a_{ij}) \in \mathbb{R}^{m \times n}$, it is always the case that*

$$\max_j \min_i a_{ij} = d^* \leq p^* = \min_i \max_j a_{ij}.$$

Proof. Let (i_d, j_d) be the row and column associated with d^*, and let (i_p, j_p) be the row and column associated with p^*. We have,

$$d^* = a_{i_d j_d} \leq a_{i_p j_d} \leq a_{i_p j_p} = p^*.$$

Here, the first inequality follows from the fact that $a_{i_d j_d}$ is the smallest element in the j_dth column (i.e., i_d was the strategy chosen by the student after the TAs chose problem j_d, and hence, it must correspond to the fewest points lost in that column). Similarly, the second inequality follow from the fact that $a_{i_p j_p}$ is the largest element in the i_pth row (i.e., j_p was the problem chosen by the TAs after the student picked strategy i_p, so it must correspond to the most points lost in that row). $\square$

2.2 Duality in optimization

The task of constrained optimization, it turns out, relates closely with the adversarial game described in the previous section. To see the connection, first recall our original optimization problem,

$$\begin{aligned}
\underset{x}{\text{minimize}} \quad & f(x) \\
\text{subject to} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m, \\
& h_i(x) = 0, \quad i = 1, \ldots, p.
\end{aligned}$$

Define the ***generalized Lagrangian*** to be

$$\mathcal{L}(x, \lambda, \nu) := f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{i=1}^{p} \nu_i h_i(x).$$

Here, the variables λ and ν are called the the ***dual variables*** (or ***Lagrange multipliers***). Analogously, the variables x are known as the ***primal variables***.

The correspondence between primal/dual optimization and game playing can be pictured informally using an infinite matrix whose rows are indexed by $x \in \mathbb{R}^n$ and whose columns are indexed by $(\lambda, \nu) \in \mathbb{R}^m_+ \times \mathbb{R}^p$ (i.e., $\lambda_i \geq 0$, for $i = 1, \ldots, m$). In particular, we have

$$A = \begin{bmatrix} \ddots & \vdots & \iddots \\ \cdots & \mathcal{L}(x, \lambda, \nu) & \cdots \\ \iddots & \vdots & \ddots \end{bmatrix}$$

Here, the "student" manipulates the primal variables x in order to minimize the Lagrangian $\mathcal{L}(x, \lambda, \nu)$ while the "TAs" manipulate the dual variables (λ, ν) in order to maximize the Lagrangian.

To see the relationship between this game and the original optimization problem, we formulate the following ***primal*** problem:

$$\begin{aligned} p^* &= \min_x \max_{\lambda,\nu:\lambda_i \geq 0} \quad \mathcal{L}(x, \lambda, \nu) \\ &= \min_x \quad \theta_P(x) \end{aligned} \tag{P'}$$

where $\theta_P(x) := \max_{\lambda,\nu:\lambda_i \geq 0} \mathcal{L}(x, \lambda, \nu)$. Computing p^* is equivalent to our original convex optimization primal in the following sense: for any candidate solution x,

- if $g_i(x) > 0$ for some $i \in \{1, \ldots, m\}$, then setting $\lambda_i = \infty$ gives $\theta_P(x) = \infty$.
- if $h_i(x) \neq 0$ for some $i \in \{1, \ldots, m\}$, then setting $\lambda_i = \infty \cdot \text{Sign}(h_i(x))$ gives $\theta_P(x) = \infty$.
- if x is feasible (i.e., x obeys all the constraints of our original optimization problem), then $\theta_P(x) = f(x)$, where the maximum is obtained, for example, by setting all of the λ_i's and ν_i's to zero.

Intuitively then, $\theta_P(x)$ behaves conceptually like an "unconstrained" version of the original constrained optimization problem in which the infeasible region of f is "carved away" by forcing $\theta_P(x) = \infty$ for any infeasible x; thus, only points in the feasible region are left as candidate minimizers. This idea of using penalties to ensure that minimizers stay in the feasible region will come up later when talk about barrier algorithms for convex optimization.

By analogy to the CS 229 grading example, we can form the following ***dual*** problem:

$$\begin{aligned} d^* &= \max_{\lambda,\nu:\lambda_i \geq 0} \min_x \quad \mathcal{L}(x, \lambda, \nu) \\ &= \max_{\lambda,\nu:\lambda_i \geq 0} \quad \theta_D(\lambda, \nu) \end{aligned} \tag{D'}$$

where $\theta_D(\lambda, \nu) := \min_x \mathcal{L}(x, \lambda, \nu)$. Dual problems can often be easier to solve than their corresponding primal problems. In the case of SVMs, for instance, SMO is a dual optimization algorithm which considers joint optimization of pairs of dual variables. Its simple form derives largely from the simplicity of the dual objective and the simplicity of the corresponding constraints on the dual variables. Primal-based SVM solutions are indeed possible, but when the number of training examples is large and the kernel matrix K of inner products $K_{ij} = K(x^{(i)}, x^{(j)})$ is large, dual-based optimization can be considerably more efficient.

Using an argument essentially identical to that presented in Theorem (2.1), we can show that in this setting, we again have $d^* \leq p^*$. This is the property of ***weak duality*** for general optimization problems. Weak duality can be particularly useful in the design of optimization algorithms. For example, suppose that during the course of an optimization algorithm we have a candidate primal solution x and dual-feasible vector (λ, ν) such that $\theta_P(x) - \theta_D(\lambda, \nu) \leq \epsilon$. From weak duality, we have that

$$\theta_D(\lambda, \nu) \leq d^* \leq p^* \leq \theta_P(x),$$

implying that x and (λ, ν) must be ϵ-optimal (i.e., their objective functions differ by no more than ϵ from the objective functions of the true optima x^* and (λ^*, ν^*).

In practice, the dual objective $\theta_D(\lambda, \nu)$ can often be found in closed form, thus allowing the dual problem (D') to depend only on the dual variables λ and ν. When the Lagrangian is differentiable with respect to x, then a closed-form for $\theta_D(\lambda, \nu)$ can often be found by setting the gradient of the Lagrangian to zero, so as to ensure that the Lagrangian is minimized with respect to x.[2] An example derivation of the dual problem for the L_1 soft-margin SVM is shown in the Appendix.

2.3 Strong duality

For any primal/dual optimization problems, weak duality will always hold. In some cases, however, the inequality $d^* \leq p^*$ may be replaced with equality, i.e., $d^* = p^*$; this latter condition is known as ***strong duality***. Strong duality does not hold in general. When it does however, the lower-bound property described in the previous section provide a useful termination criterion for optimization algorithms. In particular, we can design algorithms which simultaneously optimize both the primal and dual problems. Once the candidate solutions x of the primal problem and (λ, ν) of the dual problem obey $\theta_P(x) - \theta_D(\lambda, \nu) \leq \epsilon$, then we know that both solutions are ϵ-accurate. This is guaranteed to happen provided our optimization algorithm works properly, since strong duality guarantees that the optimal primal and dual values are equal.

Conditions which guarantee strong duality for convex optimization problems are known as ***constraint qualifications***. The most commonly invoked constraint qualification, for example, is ***Slater's condition***:

Theorem 2.2. *Consider a convex optimization problem of the form* (1)*, whose corresponding primal and dual problems are given by* (P') *and* (D')*. If there exists a primal feasible x for*

[2]Often, differentiating the Lagrangian with respect to x leads to the generation of additional requirements on dual variables that must hold at any fixed point of the Lagrangian with respect to x. When these constraints are not satisfied, one can show that the Lagrangian is unbounded below (i.e., $\theta_D(\lambda, \nu) = -\infty$).

Since such points are clearly not optimal solutions for the dual problem, we can simply exclude them from the domain of the dual problem altogether by adding the derived constraints to the existing constraints of the dual problem. An example of this is the derived constraint, $\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$, in the SVM formulation. This procedure of incorporating derived constraints into the dual problem is known as ***making dual constraints explicit*** (see [1], page 224).

which each inequality constraint is strictly satisfied (i.e., $g_i(x) < 0$), then $d^* = p^*$.[3]

The proof of this theorem is beyond the scope of this course. We will, however, point out its application to the soft-margin SVMs described in class. Recall that soft-margin SVMs were found by solving

$$\begin{array}{lll} \underset{w,b,\xi}{\text{minimize}} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i & \\ \text{subject to} & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, & i = 1,\ldots,m, \\ & \xi_i \geq 0, & i = 1,\ldots,m. \end{array}$$

Slater's condition applies provided we can find at least one primal feasible setting of w, b, and ξ where all inequalities are strict. It is easy to verify that $w = \mathbf{0}$, $b = 0$, $\xi = 2 \cdot \mathbf{1}$ satisfies these conditions (where $\mathbf{0}$ and $\mathbf{1}$ denote the vector of all 0's and all 1's, respectively), since

$$y^{(i)}(w^T x^{(i)} + b) = y^{(i)}(\mathbf{0}^T x^{(i)} + 0) = 0 > -1 = 1 - 2 = 1 - \xi_i, \qquad i = 1,\ldots,m,$$

and the remaining m inequalities are trivially strictly satisfied. Hence, strong duality holds, so the optimal values of the primal and dual soft-margin SVM problems will be equal.

2.4 The KKT conditions

In the case of differentiable unconstrained convex optimization problems, setting the gradient to "zero" provides a simple means for identifying candidate local optima. For constrained convex programming, do similar criteria exist for characterizing the optima of primal/dual optimization problems? The answer, it turns out, is provided by a set of requirements known as the ***Karush-Kuhn-Tucker (KKT) necessary and sufficient conditions*** (see [1], pages 242-244).

Suppose that the constraint functions $g_1,\ldots,g_m,h_1,\ldots,h_p$ are not only convex (the h_i's must be affine) but also differentiable.

Theorem 2.3. *If $\tilde{x}$ is primal feasible and $(\tilde{\lambda},\tilde{\nu})$ are dual feasible, and if*

$$\nabla_x \mathcal{L}(\tilde{x},\tilde{\lambda},\tilde{\nu}) = \mathbf{0}, \tag{KKT1}$$

$$\tilde{\lambda}_i g_i(\tilde{x}) = 0, \qquad i = 1,\ldots,m, \tag{KKT2}$$

then $\tilde{x}$ is primal optimal, $(\tilde{\lambda},\tilde{\nu})$ are dual optimal, and strong duality holds.

Theorem 2.4. *If Slater's condition holds, then conditions of Theorem 2.3 are necessary for any (x^*,λ^*,ν^*) such that x^* is primal optimal and (λ^*,ν^*) are dual feasible.*

[3] One can actually show a more general version of Slater's inequality, which requires only strict satisfaction of non-affine inequality constraints (but allowing affine inequalities to be satisfied with equality). See [1], page 226.

(KKT1) is the standard gradient stationarity condition found for unconstrained differentiable optimization problems. The set of inequalities corresponding to (KKT2) are known as the ***KKT complementarity (or complementary slackness) conditions***. In particular, if x^* is primal optimal and (λ^*, ν^*) is dual optimal, then (KKT2) implies that

$$\begin{aligned} \lambda_i^* > 0 \quad &\Rightarrow \quad g_i(x^*) = 0 \\ g_i(x^*) < 0 \quad &\Rightarrow \quad \lambda_i^* = 0 \end{aligned}$$

That is, whenever λ_i^* is greater than zero, its corresponding inequality constraint must be tight; conversely, any strictly satisfied inequality must have have λ_i^* equal to zero. Thus, we can interpret the dual variables λ_i^* as measuring the "importance" of a particular constraint in characterizing the optimal point.

This interpretation provides an intuitive explanation for the difference between hard-margin and soft-margin SVMs. Recall the dual problems for a hard-margin SVM:

$$\begin{aligned} \underset{\alpha,\beta}{\text{maximize}} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_i y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle \\ \text{subject to} \quad & \alpha_i \geq 0, \qquad\qquad\qquad\qquad i = 1, \ldots, m, \\ & \sum_{i=1}^m \alpha_i y^{(i)} = 0, \end{aligned} \tag{2}$$

and the L_1 soft-margin SVM:

$$\begin{aligned} \underset{\alpha,\beta}{\text{maximize}} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_i y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \qquad\qquad\qquad i = 1, \ldots, m, \\ & \sum_{i=1}^m \alpha_i y^{(i)} = 0. \end{aligned} \tag{3}$$

Note that the only difference in the soft-margin formulation is the introduction of upper bounds on the dual variables α_i. Effectively, this upper bound constraint limits the influence of any single primal inequality constraint (i.e., any single training example) on the decision boundary, leading to improved robustness for the L_1 soft-margin model.

What consequences do the KKT conditions have for practical optimization algorithms? When Slater's conditions hold, then the KKT conditions are both necessary and sufficient for primal/dual optimality of a candidate primal solution $\tilde{x}$ and a corresponding dual solution $(\tilde{\lambda}, \tilde{\nu})$. Therefore, many optimization algorithms work by trying to guarantee that the KKT conditions are satisfied; the SMO algorithm, for instance, works by iteratively identifying Lagrange multipliers for which the corresponding KKT conditions are unsatisfied and then "fixing" KKT complementarity.[4]

[4] See [1], pages 244-245 for an example of an optimization problem where the KKT conditions can be solved directly, thus skipping the need for primal/dual optimization altogether.

3 Algorithms for convex optimization

Thus far, we have talked about convex optimization problems and their properties. But how does one solve a convex optimization problem in practice? In this section, we describe a generic strategy for solving convex optimization problems known as the ***interior-point*** method. This method combines a safe-guarded variant of Newton's algorithm with a "barrier" technique for enforcing inequality constraints.

3.1 Unconstrained optimization

We consider first the problem of unconstrained optimization, i.e.,

$$\underset{x}{\text{minimize}} \quad f(x).$$

In Newton's algorithm for unconstrained optimization, we consider the Taylor approximation $\tilde{f}$ of the function f, centered at the current iterate x_t. Discarding terms of higher order than two, we have

$$\tilde{f}(x) = f(x_t) + \nabla_x f(x_t)^T (x - x_t) + \frac{1}{2}(x - x_t)\nabla_x^2 f(x_t)(x - x_t).$$

To minimize $\tilde{f}(x)$, we can set its gradient to zero. In particular, if x_{nt} denotes the minimum of $\tilde{f}(x)$, then

$$\begin{aligned}
\nabla_x f(x_t) + \nabla_x^2 f(x_t)(x_{\text{nt}} - x_t) &= 0 \\
\nabla_x^2 f(x_t)(x_{\text{nt}} - x_t) &= -\nabla_x f(x_t) \\
x_{\text{nt}} - x_t &= -\nabla_x^2 f(x_t)^{-1} \nabla_x f(x_t) \\
x_{\text{nt}} &= x_t - \nabla_x^2 f(x_t)^{-1} \nabla_x f(x_t)
\end{aligned}$$

assuming $\nabla_x^2 f(x_t)^T$ is positive definite (and hence, full rank). This, of course, is the standard Newton algorithm for unconstrained minimization.

While Newton's method converges quickly if given an initial point near the minimum, for points far from the minimum, Newton's method can sometimes diverge (as you may have discovered in problem 1 of Problem Set #1 if you picked an unfortunate initial point!). A simple fix for this behavior is to use a ***line-search*** procedure. Define the search direction d to be,

$$d := \nabla_x^2 f(x_t)^{-1} \nabla_x f(x_t).$$

A line-search procedure is an algorithm for finding an appropriate step size $\gamma \geq 0$ such that the iteration

$$x_{t+1} = x_t - \gamma \cdot d$$

will ensure that the function f decreases by a sufficient amount (relative to the size of the step taken) during each iteration.

One simple yet effective method for doing this is called a ***backtracking line search***. In this method, one initially sets γ to 1 and then iteratively reduces γ by a multiplicative factor β until $f(x_t + \gamma \cdot d)$ is sufficiently smaller than $f(x_t)$:

Backtracking line-search

- Choose $\alpha \in (0, 0.5)$, $\beta \in (0, 1)$.
- Set $\gamma \leftarrow 1$.
- While $f(x_t + \gamma \cdot d) > f(x_t) + \gamma \cdot \alpha \nabla_x f(x_t)^T d$, do $\gamma \leftarrow \beta\gamma$.
- Return γ.

Since the function f is known to decrease locally near x_t in the direction of d, such a step will be found, provided γ is small enough. For more details, see [1], pages 464-466.

In order to use Newton's method, one must be able to compute and invert the Hessian matrix $\nabla_x^2 f(x_t)$, or equivalently, compute the search direction d indirectly without forming the Hessian. For some problems, the number of primal variables x is sufficiently large that computing the Hessian can be very difficult. In many cases, this can be dealt with by clever use of linear algebra. In other cases, however, we can resort to other nonlinear minimization schemes, such as ***quasi-Newton*** methods, which initially behave like gradient descent but gradually construct approximations of the inverse Hessian based on the gradients observed throughout the course of the optimization.[5] Alternatively, ***nonlinear conjugate gradient*** schemes (which augment the standard conjugate gradient (CG) algorithm for solving linear least squares systems with a line-search) provide another generic blackbox tool for multivariable function minimization which is simple to implement, yet highly effective in practice.[6]

3.2 Inequality-constrained optimization

Using our tools for unconstrained optimization described in the previous section, we now tackle the (slightly) harder problem of constrained optimization. For simplicity, we consider convex optimization problems without equality constraints[7], i.e., problems of the form,

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m. \end{aligned}$$

[5] For more information on Quasi-Newton methods, the standard reference is Jorge Nocedal and Stephen J. Wright's textbook, *Numerical Optimization.*

[6] For an excellent tutorial on the conjugate gradient method, see Jonathan Shewchuk's tutorial, available at: `http://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf`

[7] In practice, there are many of ways of dealing with equality constraints. Sometimes, we can eliminate equality constraints by either reparameterizing of the original primal problem, or converting to the dual problem. A more general strategy is to rely on equality-constrained variants of Newton's algorithms which ensure that the equality constraints are satisfied at every iteration of the optimization. For a more complete treatment of this topic, see [1], Chapter 10.

We will also assume knowledge of a feasible starting point x_0 which satisfies all of our constraints with strict inequality (as needed for Slater's condition to hold).[8]

Recall that in our discussion of the Lagrangian-based formulation of the primal problem,

$$\min_x \max_{\lambda:\lambda_i \geq 0} \quad \mathcal{L}(x, \lambda).$$

we stated that the inner maximization, $\max_{\lambda:\lambda_i \geq 0} \ \mathcal{L}(x, \lambda)$, was constructed in such a way that the infeasible region of f was "carved away", leaving only points in the feasible region as candidate minima. The same idea of using penalties to ensure that minimizers stay in the feasible region is the basis of ***barrier***-based optimization. Specifically, if $B(z)$ is the barrier function

$$B(z) = \begin{cases} 0 & z < 0 \\ \infty & z \geq 0, \end{cases}$$

then the primal problem is equivalent to

$$\min_x \quad f(x) + \sum_{i=1}^{m} B(g_i(x)). \tag{4}$$

When $g_i(x) < 0$, the objective of the problem is simply $f(x)$; infeasible points are "carved away" using the barrier function $B(z)$.

While conceptually correct, optimization using the straight barrier function $B(x)$ is numerically difficult. To ameliorate this, the ***log-barrier*** optimization algorithm approximates the solution to (4) by solving the unconstrained problem,

$$\underset{x}{\text{minimize}} \quad f(x) - \frac{1}{t} \sum_{i=1}^{m} \log(-g_i(x)).$$

for some fixed $t > 0$. Here, the function $-(1/t) \log(-z) \approx B(z)$, and the accuracy of the approximation increases as $t \to \infty$. Rather than using a large value of t in order to obtain a good approximation, however, the log-barrier algorithm works by solving a sequence of unconstrained optimization problems, increasing t each time, and using the solution of the previous unconstrained optimization problem as the initial point for the next unconstrained optimization. Furthermore, at each point in the algorithm, the primal solution points stay strictly in the interior of the feasible region:

[8]For more information on finding feasible starting points for barrier algorithms, see [1], pages 579-585. For inequality-problems where the primal problem is feasible but not strictly feasible, ***primal-dual interior point*** methods are applicable, also described in [1], pages 609-615.

Log-barrier optimization

- Choose $\mu > 1$, $t > 0$.
- $x \leftarrow x_0$.
- Repeat until convergence:

  (a) Compute $x' = \min_x f(x) - \frac{1}{t}\sum_{i=1}^{m} \log(-g_i(x))$ using x as the initial point.

  (b) $t \leftarrow \mu \cdot t$, $x \leftarrow x'$.

One might expect that as t increases, the difficulty of solving each unconstrained minimization problem also increases due to numerical issues or ill-conditioning of the optimization problem. Surprisingly, Nesterov and Nemirovski showed in 1994 that this is not the case for certain types of barrier functions, including the log-barrier; in particular, by using an appropriate barrier function, one obtains a general convex optimization algorithm which takes time polynomial in the dimensionality of the optimization variables and the desired accuracy!

4 Directions for further exploration

In many real-world tasks, 90% of the challenge involves figuring out how to write an optimization problem in a convex form. Once the correct form has been found, a number of pre-existing software packages for convex optimization have been well-tuned to handle different specific types of optimization problems. The following constitute a small sample of the available tools:

- commerical packages: CPLEX, MOSEK
- MATLAB-based: CVX, Optimization Toolbox (linprog, quadprog), SeDuMi
- libraries: CVXOPT (Python), GLPK (C), COIN-OR (C)
- SVMs: LIBSVM, SVM-light
- machine learning: Weka (Java)

In particular, we specifically point out CVX as an easy-to-use generic tool for solving convex optimization problems easily using MATLAB, and CVXOPT as a powerful Python-based library which runs independently of MATLAB.[9] If you're interested in looking at some of the other packages listed above, they are easy to find with a web search. In short, if you need a specific convex optimization algorithm, pre-existing software packages provide a rapid way to prototype your idea without having to deal with the numerical trickiness of implementing your own complete convex optimization routines.

[9] CVX is available at `http://www.stanford.edu/~boyd/cvx` and CVXOPT is available at `http://www.ee.ucla.edu/~vandenbe/cvxopt/`.

Also, if you find this material fascinating, make sure to check out Stephen Boyd's class, EE364: Convex Optimization I, which will be offered during the Winter Quarter. The textbook for the class (listed as [1] in the References) has a wealth of information about convex optimization and is available for browsing online.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge UP, 2004. Online: `http://www.stanford.edu/~boyd/cvxbook/`

Appendix: The soft-margin SVM

To see the primal/dual action in practice, we derive the dual of the soft-margin SVM primal presented in class, and corresponding KKT complementarity conditions. We have,

$$\begin{array}{lll} \underset{w,b,\xi}{\text{minimize}} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^m \xi_i & \\ \text{subject to} & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, & i = 1, \ldots, m, \\ & \xi_i \geq 0, & i = 1, \ldots, m. \end{array}$$

First, we put this into our standard form, with "≤ 0" inequality constraints and no equality constraints. That is,

$$\begin{array}{lll} \underset{w,b,\xi}{\text{minimize}} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^m \xi_i & \\ \text{subject to} & 1 - \xi_i - y^{(i)}(w^T x^{(i)} + b) \leq 0, & i = 1, \ldots, m, \\ & -\xi_i \leq 0, & i = 1, \ldots, m. \end{array}$$

Next, we form the generalized Lagrangian,[10]

$$\mathcal{L}(w, b, \xi, \alpha, \beta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^m \xi_i + \sum_{i=1}^m \alpha_i(1 - \xi_i - y^{(i)}(w^T x^{(i)} + b)) - \sum_{i=1}^m \beta_i \xi_i,$$

which gives the primal and dual optimization problems:

$$\max_{\alpha,\beta:\alpha_i \geq 0, \beta_i \geq 0} \theta_D(\alpha, \beta) \qquad \text{where } \theta_D(\alpha, \beta) := \min_{w,b,\xi} \mathcal{L}(w, b, \xi, \alpha, \beta), \tag{SVM-D}$$

$$\min_{w,b,\xi} \theta_P(w, b, \xi) \qquad \text{where } \theta_P(w, b, \xi) := \max_{\alpha,\beta:\alpha_i \geq 0, \beta_i \geq 0} \mathcal{L}(w, b, \xi, \alpha, \beta). \tag{SVM-P}$$

To get the dual problem in the form shown in the lecture notes, however, we still have a little more work to do. In particular,

[10] Here, it is important to note that (w, b, ξ) collectively play the role of the x primal variables. Similarly, (α, β) collectively play the role of the λ dual variables used for inequality constraints. There are no "ν" dual variables here since there are no affine constraints in this problem.

1. **Eliminating the primal variables.** To eliminate the primal variables from the dual problem, we compute $\theta_D(\alpha, \beta)$ by noticing that

$$\theta_D(\alpha, \beta) = \min_{w,b,\xi} \quad \mathcal{L}(w, b, \xi, \alpha, \beta)$$

is an unconstrained optimization problem, where the objective function $\mathcal{L}(w, b, \xi, \alpha, \beta)$ is differentiable. Therefore, for any fixed (α, β), if $(\hat{w}, \hat{b}, \hat{\xi})$ minimize the Lagrangian, it must be the case that

$$\nabla_w \mathcal{L}(\hat{w}, \hat{b}, \hat{\xi}, \alpha, \beta) = \hat{w} - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \tag{5}$$

$$\frac{\partial}{\partial b} \mathcal{L}(\hat{w}, \hat{b}, \hat{\xi}, \alpha, \beta) = - \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \tag{6}$$

$$\frac{\partial}{\partial \xi_i} \mathcal{L}(\hat{w}, \hat{b}, \hat{\xi}, \alpha, \beta) = C - \alpha_i - \beta_i = 0. \tag{7}$$

Adding (6) and (7) to the constraints of our dual optimizaton problem, we obtain,

$$\begin{aligned}
\theta_D(\alpha, \beta) &= \mathcal{L}(\hat{w}, \hat{b}, \hat{\xi}) \\
&= \frac{1}{2}\|\hat{w}\|^2 + C \sum_{i=1}^{m} \hat{\xi}_i + \sum_{i=1}^{m} \alpha_i (1 - \hat{\xi}_i - y^{(i)}(\hat{w}^T x^{(i)} + \hat{b})) - \sum_{i=1}^{m} \beta_i \hat{\xi}_i \\
&= \frac{1}{2}\|\hat{w}\|^2 + C \sum_{i=1}^{m} \hat{\xi}_i + \sum_{i=1}^{m} \alpha_i (1 - \hat{\xi}_i - y^{(i)}(\hat{w}^T x^{(i)})) - \sum_{i=1}^{m} \beta_i \hat{\xi}_i \\
&= \frac{1}{2}\|\hat{w}\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y^{(i)}(\hat{w}^T x^{(i)})).
\end{aligned}$$

where the first equality follows from the optimality of $(\hat{w}, \hat{b}, \hat{\xi})$ for fixed (α, β), the second equality uses the definition of the generalized Lagrangian, and the third and fourth equalities follow from (6) and (7), respectively. Finally, to use (5), observe that

$$\begin{aligned}
\frac{1}{2}\|\hat{w}\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y^{(i)}(\hat{w}^T x^{(i)})) &= \sum_{i=1}^{m} \alpha_i + \frac{1}{2}\|\hat{w}\|^2 - \hat{w}^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \\
&= \sum_{i=1}^{m} \alpha_i + \frac{1}{2}\|\hat{w}\|^2 - \|\hat{w}\|^2 \\
&= \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\|\hat{w}\|^2 \\
&= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_i y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle.
\end{aligned}$$

Therefore, our dual problem (with no more primal variables) is simply

$$\begin{aligned}
\max_{\alpha,\beta} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_i y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle \\
\text{subject to} \quad & \alpha_i \geq 0, & i = 1, \ldots, m, \\
& \beta_i \geq 0, & i = 1, \ldots, m, \\
& \alpha_i + \beta_i = C, & i = 1, \ldots, m, \\
& \sum_{i=1}^m \alpha_i y^{(i)} = 0.
\end{aligned}$$

2. **KKT complementary.** KKT complementarity requires that for any primal optimal (w^*, b^*, ξ^*) and dual optimal (α^*, β^*),

$$\begin{aligned}
\alpha_i^*(1 - \xi_i^* - y^{(i)}(w^{*T} x^{(i)} + b^*)) &= 0 \\
\beta_i^* \xi_i^* &= 0
\end{aligned}$$

for $i = 1, \ldots, m$. From the first condition, we see that if $\alpha_i > 0$, then in order for the product to be zero, then $1 - \xi_i^* - y^{(i)}(w^{*T} x^{(i)} + b^*) = 0$. It follows that

$$y^{(i)}(w^{*T} x^{(i)} + b^*) \leq 1$$

since $\xi^* \geq 0$ by primal feasibility. Similarly, if $\beta_i^* > 0$, then $\xi_i^* = 0$ to ensure complementarity. From the primal constraint, $y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i$, it follows that

$$y^{(i)}(w^{*T} x^{(i)} + b^*) \geq 1.$$

Finally, since $\beta_i^* > 0$ is equivalent to $\alpha_i^* < C$ (since $\alpha^* + \beta_i^* = C$), we can summarize the KKT conditions as follows:

$$\begin{aligned}
\alpha_i^* = 0 \quad &\Rightarrow \quad y^{(i)}(w^{*T} x^{(i)} + b^*) \geq 1, \\
0 < \alpha_i^* < C \quad &\Rightarrow \quad y^{(i)}(w^{*T} x^{(i)} + b^*) = 1, \\
\alpha_i^* = C \quad &\Rightarrow \quad y^{(i)}(w^{*T} x^{(i)} + b^*) \leq 1.
\end{aligned}$$

3. **Simplification.** We can tidy up our dual problem slightly by observing that each pair of constraints of the form

$$\beta_i \geq 0 \qquad\qquad \alpha_i + \beta_i = C$$

is equivalent to the single constraint, $\alpha_i \leq C$; that is, if we solve the optimization problem

$$\begin{aligned}
\max_{\alpha,\beta} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_i y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle \\
\text{subject to} \quad & 0 \leq \alpha_i \leq C, & i = 1, \ldots, m, \\
& \sum_{i=1}^m \alpha_i y^{(i)} = 0.
\end{aligned} \tag{8}$$

and subsequently set $\beta_i = C - \alpha_i$, then it follows that (α, β) will be optimal for the previous dual problem above. This last form, indeed, is the form of the soft-margin SVM dual given in the lecture notes.

Hidden Markov Models Fundamentals

Daniel Ramage
CS229 Section Notes

December 1, 2007

Abstract

How can we apply machine learning to data that is represented as a sequence of observations over time? For instance, we might be interested in discovering the sequence of words that someone spoke based on an audio recording of their speech. Or we might be interested in annotating a sequence of words with their part-of-speech tags. These notes provides a thorough mathematical introduction to the concept of Markov Models — a formalism for reasoning about states over time — and Hidden Markov Models — where we wish to recover a series of states from a series of observations. The final section includes some pointers to resources that present this material from other perspectives.

1 Markov Models

Given a set of states $S = \{s_1, s_2, ...s_{|S|}\}$ we can observe a series over time $\vec{z} \in S^T$. For example, we might have the states from a weather system $S = \{sun, cloud, rain\}$ with $|S| = 3$ and observe the weather over a few days $\{z_1 = s_{sun}, z_2 = s_{cloud}, z_3 = s_{cloud}, z_4 = s_{rain}, z_5 = s_{cloud}\}$ with $T = 5$.

The observed states of our weather example represent the output of a random process over time. Without some further assumptions, state s_j at time t could be a function of any number of variables, including all the states from times 1 to $t-1$ and possibly many others that we don't even model. However, we will make two Markov assumptions that will allow us to tractably reason about time series.

The limited horizon assumption is that the probability of being in a state at time t depends only on the state at time $t-1$. The intuition underlying this assumption is that the state at time t represents "enough" summary of the past to reasonably predict the future. Formally:

$$P(z_t|z_{t-1}, z_{t-2}, ..., z_1) = P(z_t|z_{t-1})$$

The stationary process assumption is that the conditional distribution over next state given current state does not change over time. Formally:

$$P(z_t|z_{t-1}) = P(z_2|z_1);\ t \in 2...T$$

As a convention, we will also assume that there is an initial state and initial observation $z_0 \equiv s_0$, where s_0 represents the initial probability distribution over states at time 0. This notational convenience allows us to encode our belief about the prior probability of seeing the first real state z_1 as $P(z_1|z_0)$. Note that $P(z_t|z_{t-1}, ..., z_1) = P(z_t|z_{t-1}, ..., z_1, z_0)$ because we've defined $z_0 = s_0$ for any state sequence. (Other presentations of HMMs sometimes represent these prior believes with a vector $\pi \in \mathbb{R}^{|S|}$.)

We parametrize these transitions by defining a state transition matrix $A \in \mathbb{R}^{(|S|+1)\times(|S|+1)}$. The value A_{ij} is the probability of transitioning from state i to state j at any time t. For our sun and rain example, we might have following transition matrix:

$$A = \begin{array}{c|cccc} & s_0 & s_{sun} & s_{cloud} & s_{rain} \\ s_0 & 0 & .33 & .33 & .33 \\ s_{sun} & 0 & .8 & .1 & .1 \\ s_{cloud} & 0 & .2 & .6 & .2 \\ s_{rain} & 0 & .1 & .2 & .7 \end{array}$$

Note that these numbers (which I made up) represent the intuition that the weather is self-correlated: if it's sunny it will tend to stay sunny, cloudy will stay cloudy, etc. This pattern is common in many Markov models and can be observed as a strong diagonal in the transition matrix. Note that in this example, our initial state s_0 shows uniform probability of transitioning to each of the three states in our weather system.

1.1 Two questions of a Markov Model

Combining the Markov assumptions with our state transition parametrization A, we can answer two basic questions about a sequence of states in a Markov chain. What is the probability of a particular sequence of states $\vec{z}$? And how do we estimate the parameters of our model A such to maximize the likelihood of an observed sequence $\vec{z}$?

1.1.1 Probability of a state sequence

We can compute the probability of a particular series of states $\vec{z}$ by use of the chain rule of probability:

$$\begin{aligned} P(\vec{z}) &= P(z_t, z_{t-1}, ..., z_1; A) \\ &= P(z_t, z_{t-1}, ..., z_1, z_0; A) \\ &= P(z_t|z_{t-1}, z_{t-2}, ..., z_1; A)P(z_{t-1}|z_{t-2}, ..., z_1; A)...P(z_1|z_0; A) \\ &= P(z_t|z_{t-1}; A)P(z_{t-1}|z_{t-2}; A)...P(z_2|z_1; A)P(z_1|z_0; A) \end{aligned}$$

$$
\begin{aligned}
&= \prod_{t=1}^{T} P(z_t|z_{t-1}; A) \\
&= \prod_{t=1}^{T} A_{z_{t-1}\, z_t}
\end{aligned}
$$

In the second line we introduce z_0 into our joint probability, which is allowed by the definition of z_0 above. The third line is true of any joint distribution by the chain rule of probabilities or repeated application of Bayes rule. The fourth line follows from the Markov assumptions and the last line represents these terms as their elements in our transition matrix A.

Let's compute the probability of our example time sequence from earlier. We want $P(z_1 = s_{sun}, z_2 = s_{cloud}, z_3 = s_{rain}, z_4 = s_{rain}, z_5 = s_{cloud})$ which can be factored as $P(s_{sun}|s_0)P(s_{cloud}|s_{sun})P(s_{rain}|s_{cloud})P(s_{rain}|s_{rain})P(s_{cloud}|s_{rain}) = .33 \times .1 \times .2 \times .7 \times .2$.

1.1.2 Maximum likelihood parameter assignment

From a learning perspective, we could seek to find the parameters A that maximize the log-likelihood of sequence of observations $\vec{z}$. This corresponds to finding the likelihoods of transitioning from sunny to cloudy versus sunny to sunny, etc., that make a set of observations most likely. Let's define the log-likelihood a Markov model.

$$
\begin{aligned}
l(A) &= \log P(\vec{z}; A) \\
&= \log \prod_{t=1}^{T} A_{z_{t-1}\, z_t} \\
&= \sum_{t=1}^{T} \log A_{z_{t-1}\, z_t} \\
&= \sum_{i=1}^{|S|} \sum_{j=1}^{|S|} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \log A_{ij}
\end{aligned}
$$

In the last line, we use an indicator function whose value is one when the condition holds and zero otherwise to select the observed transition at each time step. When solving this optimization problem, it's important to ensure that solved parameters A still make a valid transition matrix. In particular, we need to enforce that the outgoing probability distribution from state i always sums to 1 and all elements of A are non-negative. We can solve this optimization problem using the method of Lagrange multipliers.

$$
\max_{A} \quad l(A)
$$

$$
\begin{aligned}
s.t. \quad & \sum_{j=1}^{|S|} A_{ij} = 1, \;\; i = 1..|S| \\
& A_{ij} \geq 0, \;\; i, j = 1..|S|
\end{aligned}
$$

This constrained optimization problem can be solved in closed form using the method of Lagrange multipliers. We'll introduce the equality constraint into the Lagrangian, but the inequality constraint can safely be ignored — the optimal solution will produce positive values for A_{ij} anyway. Therefore we construct the Lagrangian as:

$$
\mathcal{L}(A, \alpha) = \sum_{i=1}^{|S|} \sum_{j=1}^{|S|} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \log A_{ij} + \sum_{i=1}^{|S|} \alpha_i (1 - \sum_{j=1}^{|S|} A_{ij})
$$

Taking partial derivatives and setting them equal to zero we get:

$$
\begin{aligned}
\frac{\partial \mathcal{L}(A, \alpha)}{\partial A_{ij}} &= \frac{\partial}{\partial A_{ij}} (\sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \log A_{ij}) + \frac{\partial}{\partial A_{ij}} \alpha_i (1 - \sum_{j=1}^{|S|} A_{ij}) \\
&= \frac{1}{A_{ij}} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} - \alpha_i \equiv 0 \\
&\Rightarrow \\
A_{ij} &= \frac{1}{\alpha_i} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\}
\end{aligned}
$$

Substituting back in and setting the partial with respect to α equal to zero:

$$
\begin{aligned}
\frac{\partial \mathcal{L}(A, \beta)}{\partial \alpha_i} &= 1 - \sum_{j=1}^{|S|} A_{ij} \\
&= 1 - \sum_{j=1}^{|S|} \frac{1}{\alpha_i} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \equiv 0 \\
&\Rightarrow \\
\alpha_i &= \sum_{j=1}^{|S|} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \\
&= \sum_{t=1}^{T} 1\{z_{t-1} = s_i\}
\end{aligned}
$$

Substituting in this value for α_i into the expression we derived for A_{ij} we obtain our final maximum likelihood parameter value for $\hat{A}_{ij}$.

$$\hat{A}_{ij} = \frac{\sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\}}{\sum_{t=1}^{T} 1\{z_{t-1} = s_i\}}$$

This formula encodes a simple intuition: the maximum likelihood probability of transitioning from state i to state j is just the number of times we transition from i to j divided by the total number of times we are in i. In other words, the maximum likelihood parameter corresponds to the fraction of the time when we were in state i that we transitioned to j.

2 Hidden Markov Models

Markov Models are a powerful abstraction for time series data, but fail to capture a very common scenario. How can we reason about a series of states if we cannot observe the states themselves, but rather only some probabilistic function of those states? This is the scenario for part-of-speech tagging where the words are observed but the parts-of-speech tags aren't, and for speech recognition where the sound sequence is observed but not the words that generated it. For a simple example, let's borrow the setup proposed by Jason Eisner in 2002 [1], "Ice Cream Climatology."

> The situation: You are a climatologist in the year 2799, studying the history of global warming. You can't find any records of Baltimore weather, but you do find my (Jason Eisner's) diary, in which I assiduously recorded how much ice cream I ate each day. *What can you figure out from this about the weather that summer?*

A Hidden Markov Model (HMM) can be used to explore this scenario. We don't get to observe the actual sequence of states (the weather on each day). Rather, we can only observe some outcome generated by each state (how many ice creams were eaten that day).

Formally, an HMM is a Markov model for which we have a series of *observed* outputs $x = \{x_1, x_2, ..., x_T\}$ drawn from an output alphabet $V = \{v_1, v_2, ..., v_{|V|}\}$, i.e. $x_t \in V$, $t = 1..T$. As in the previous section, we also posit the existence of series of states $z = \{z_1, z_2, ..., z_T\}$ drawn from a state alphabet $S = \{s_1, s_2, ...s_{|S|}\}$, $z_t \in S$, $t = 1..T$ but in this scenario the values of the states are *unobserved*. The transition between states i and j will again be represented by the corresponding value in our state transition matrix A_{ij}.

We also model the probability of generating an output observation as a function of our hidden state. To do so, we make the OUTPUT INDEPENDENCE ASSUMPTION and define $P(x_t = v_k | z_t = s_j) = P(x_t = v_k | x_1, ..., x_T, z_1, ..., z_T) = B_{jk}$. The matrix B encodes the probability of our hidden state generating output v_k given that the state at the corresponding time was s_j.

Returning to the weather example, imagine that you have logs of ice cream consumption over a four day period: $\vec{x} = \{x_1 = v_3, x_2 = v_2, x_3 = v_1, x_4 = v_2\}$

where our alphabet just encodes the number of ice creams consumed, i.e. $V = \{v_1 = 1\, ice\, cream, v_2 = 2\, ice\, creams, v_3 = 3\, ice\, creams\}$. What questions can an HMM let us answer?

2.1 Three questions of a Hidden Markov Model

There are three fundamental questions we might ask of an HMM. What is the probability of an observed sequence (how likely were we to see $3, 2, 1, 2$ ice creams consumed)? What is the most likely series of states to generate the observations (what was the weather for those four days)? And how can we learn values for the HMM's parameters A and B given some data?

2.2 Probability of an observed sequence: Forward procedure

In an HMM, we assume that our data was generated by the following process: posit the existence of a series of states $\vec{z}$ over the length of our time series. This state sequence is generated by a Markov model parametrized by a state transition matrix A. At each time step t, we select an output x_t as a function of the state z_t. Therefore, to get the probability of a sequence of observations, we need to add up the likelihood of the data $\vec{x}$ given every possible series of states.

$$\begin{aligned}
P(\vec{x}; A, B) &= \sum_{\vec{z}} P(\vec{x}, \vec{z}; A, B) \\
&= \sum_{\vec{z}} P(\vec{x}|\vec{z}; A, B) P(\vec{z}; A, B)
\end{aligned}$$

The formulas above are true for any probability distribution. However, the HMM assumptions allow us to simplify the expression further:

$$\begin{aligned}
P(\vec{x}; A, B) &= \sum_{\vec{z}} P(\vec{x}|\vec{z}; A, B) P(\vec{z}; A, B) \\
&= \sum_{\vec{z}} \left(\prod_{t=1}^{T} P(x_t|z_t; B)\right) \left(\prod_{t=1}^{T} P(z_t|z_{t-1}; A)\right) \\
&= \sum_{\vec{z}} \left(\prod_{t=1}^{T} B_{z_t\, x_t}\right) \left(\prod_{t=1}^{T} A_{z_{t-1}\, z_t}\right)
\end{aligned}$$

The good news is that this is a simple expression in terms of our parameters. The derivation follows the HMM assumptions: the output independence assumption, Markov assumption, and stationary process assumption are all used to derive the second line. The bad news is that the sum is over every possible assignment to $\vec{z}$. Because z_t can take one of $|S|$ possible values at each time step, evaluating this sum directly will require $O(|S|^T)$ operations.

Algorithm 1 Forward Procedure for computing $\alpha_i(t)$

1. Base case: $\alpha_i(0) = A_{0\,i},\ i = 1..|S|$
2. Recursion: $\alpha_j(t) = \sum_{i=1}^{|S|} \alpha_i(t-1) A_{ij} B_{j\,x_t},\ \ j = 1..|S|,\ t = 1..T$

Fortunately, a faster means of computing $P(\vec{x}; A, B)$ is possible via a dynamic programming algorithm called the Forward Procedure. First, let's define a quantity $\alpha_i(t) = P(x_1, x_2, ..., x_t, z_t = s_i; A, B)$. $\alpha_i(t)$ represents the total probability of all the observations up through time t (by any state assignment) and that we are in state s_i at time t. If we had such a quantity, the probability of our full set of observations $P(\vec{x})$ could be represented as:

$$\begin{aligned} P(\vec{x}; A, B) &= P(x_1, x_2, ..., x_T; A, B) \\ &= \sum_{i=1}^{|S|} P(x_1, x_2, ..., x_T, z_T = s_i; A, B) \\ &= \sum_{i=1}^{|S|} \alpha_i(T) \end{aligned}$$

Algorithm 2.2 presents an efficient way to compute $\alpha_i(t)$. At each time step we must do only $O(|S|)$ operations, resulting in a final algorithm complexity of $O(|S| \cdot T)$ to compute the total probability of an observed state sequence $P(\vec{x}; A, B)$.

A similar algorithm known as the Backward Procedure can be used to compute an analogous probability $\beta_i(t) = P(x_T, x_{T-1}, .., x_{t+1}, z_t = s_i; A, B)$.

2.3 Maximum Likelihood State Assignment: The Viterbi Algorithm

One of the most common queries of a Hidden Markov Model is to ask what was the most likely series of states $\vec{z} \in S^T$ given an observed series of outputs $\vec{x} \in V^T$. Formally, we seek:

$$\arg\max_{\vec{z}} P(\vec{z}|\vec{x}; A, B) = \arg\max_{\vec{z}} \frac{P(\vec{x}, \vec{z}; A, B)}{\sum_{\vec{z}} P(\vec{x}, \vec{z}; A, B)} = \arg\max_{\vec{z}} P(\vec{x}, \vec{z}; A, B)$$

The first simplification follows from Bayes rule and the second from the observation that the denominator does not directly depend on $\vec{z}$. Naively, we might try every possible assignment to $\vec{z}$ and take the one with the highest joint probability assigned by our model. However, this would require $O(|S|^T)$ operations just to enumerate the set of possible assignments. At this point, you might think a dynamic programming solution like the Forward Algorithm might save the day, and you'd be right. Notice that if you replaced the $\arg\max_{\vec{z}}$ with $\sum_{\vec{z}}$, our current task is exactly analogous to the expression which motivated the forward procedure.

Algorithm 2 Naive application of EM to HMMs

Repeat until convergence {

(E-Step) For every possible labeling $\vec{z} \in S^T$, set

$$Q(\vec{z}) \quad := \quad p(\vec{z}|\vec{x}; A, B)$$

(M-Step) Set

$$\begin{aligned} A, B \quad &:= \quad \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \log \frac{P(\vec{x}, \vec{z}; A, B)}{Q(\vec{z})} \\ & \quad s.t. \quad \sum_{j=1}^{|S|} A_{ij} = 1,\ i = 1..|S|;\ A_{ij} \geq 0,\ \ i, j = 1..|S| \\ & \qquad\quad \sum_{k=1}^{|V|} B_{ik} = 1,\ i = 1..|S|;\ B_{ik} \geq 0,\ \ i = 1..|S|, k = 1..|V| \end{aligned}$$

}

The VITERBI ALGORITHM is just like the forward procedure except that instead of tracking the total probability of generating the observations seen so far, we need only track the *maximum* probability and record its corresponding state sequence.

2.4 Parameter Learning: EM for HMMs

The final question to ask of an HMM is: given a set of observations, what are the values of the state transition probabilities A and the output emission probabilities B that make the data most likely? For example, solving for the maximum likelihood parameters based on a speech recognition dataset will allow us to effectively train the HMM before asking for the maximum likelihood state assignment of a candidate speech signal.

In this section, we present a derivation of the Expectation Maximization algorithm for Hidden Markov Models. This proof follows from the general formulation of EM presented in the CS229 lecture notes. Algorithm 2.4 shows the basic EM algorithm. Notice that the optimization problem in the M-Step is now constrained such that A and B contain valid probabilities. Like the maximum likelihood solution we found for (non-Hidden) Markov models, we'll be able to solve this optimization problem with Lagrange multipliers. Notice also that the E-Step and M-Step both require enumerating all $|S|^T$ possible labellings of $\vec{z}$. We'll make use of the Forward and Backward algorithms mentioned earlier to compute a set of sufficient statistics for our E-Step and M-Step tractably.

First, let's rewrite the objective function using our Markov assumptions.

$$
\begin{aligned}
A, B &= \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \log \frac{P(\vec{x}, \vec{z}; A, B)}{Q(\vec{z})} \\
&= \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \log P(\vec{x}, \vec{z}; A, B) \\
&= \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \log(\prod_{t=1}^{T} P(x_t|z_t; B)) \, (\prod_{t=1}^{T} P(z_t|z_{t-1}; A)) \\
&= \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} \log B_{z_t\, x_t} + \log A_{z_{t-1}\, z_t} \\
&= \arg\max_{A,B} \sum_{\vec{z}} Q(\vec{z}) \sum_{i=1}^{|S|} \sum_{j=1}^{|S|} \sum_{k=1}^{|V|} \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} \log B_{jk} + 1\{z_{t-1} = s_i \wedge z_t = s_j\} \log A_{ij}
\end{aligned}
$$

In the first line we split the log division into a subtraction and note that the denominator's term does not depend on the parameters A, B. The Markov assumptions are applied in line 3. Line 5 uses indicator functions to index A and B by state.

Just as for the maximum likelihood parameters for a visible Markov model, it is safe to ignore the inequality constraints because the solution form naturally results in only positive solutions. Constructing the Lagrangian:

$$
\begin{aligned}
\mathcal{L}(A, B, \delta, \epsilon) &= \sum_{\vec{z}} Q(\vec{z}) \sum_{i=1}^{|S|} \sum_{j=1}^{|S|} \sum_{k=1}^{|V|} \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} \log B_{jk} + 1\{z_{t-1} = s_i \wedge z_t = s_j\} \log A_{ij} \\
&\quad + \sum_{j=1}^{|S|} \epsilon_j (1 - \sum_{k=1}^{|V|} B_{jk}) + \sum_{i=1}^{|S|} \delta_i (1 - \sum_{j=1}^{|S|} A_{ij})
\end{aligned}
$$

Taking partial derivatives and setting them equal to zero:

$$
\begin{aligned}
\frac{\partial \mathcal{L}(A, B, \delta, \epsilon)}{\partial A_{ij}} &= \sum_{\vec{z}} Q(\vec{z}) \frac{1}{A_{ij}} \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} - \delta_i \equiv 0 \\
A_{ij} &= \frac{1}{\delta_i} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}(A, B, \delta, \epsilon)}{\partial B_{jk}} &= \sum_{\vec{z}} Q(\vec{z}) \frac{1}{B_{jk}} \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} - \epsilon_j \equiv 0 \\
B_{jk} &= \frac{1}{\epsilon_j} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\}
\end{aligned}
$$

Taking partial derivatives with respect to the Lagrange multipliers and substituting our values of A_{ij} and B_{jk} above:

$$\begin{aligned}
\frac{\partial \mathcal{L}(A,B,\delta,\epsilon)}{\partial \delta_i} &= 1 - \sum_{j=1}^{|S|} A_{ij} \\
&= 1 - \sum_{j=1}^{|S|} \frac{1}{\delta_i} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \equiv 0 \\
\delta_i &= \sum_{j=1}^{|S|} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \\
&= \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i\}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{L}(A,B,\delta,\epsilon)}{\partial \epsilon_j} &= 1 - \sum_{k=1}^{|V|} B_{jk} \\
&= 1 - \sum_{k=1}^{|V|} \frac{1}{\epsilon_j} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} \equiv 0 \\
\epsilon_j &= \sum_{k=1}^{|V|} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} \\
&= \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j\}
\end{aligned}$$

Substituting back into our expressions above, we find that parameters $\hat{A}$ and $\hat{B}$ that maximize our predicted counts with respect to the dataset are:

$$\begin{aligned}
\hat{A}_{ij} &= \frac{\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\}}{\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i\}} \\
\hat{B}_{jk} &= \frac{\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\}}{\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j\}}
\end{aligned}$$

Unfortunately, each of these sums is over all possible labellings $\vec{z} \in S^T$. But recall that $Q(\vec{z})$ was defined in the E-step as $P(\vec{z}|\vec{x}; A, B)$ for parameters A and B at the last time step. Let's consider how to represent first the numerator of $\hat{A}_{ij}$ in terms of our forward and backward probabilities, $\alpha_i(t)$ and $\beta_j(t)$.

$$\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\}$$

$$
\begin{aligned}
&= \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_{t-1} = s_i \wedge z_t = s_j\} Q(\vec{z}) \\
&= \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_{t-1} = s_i \wedge z_t = s_j\} P(\vec{z}|\vec{x}; A, B) \\
&= \frac{1}{P(\vec{x}; A, B)} \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_{t-1} = s_i \wedge z_t = s_j\} P(\vec{z}, \vec{x}; A, B) \\
&= \frac{1}{P(\vec{x}; A, B)} \sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\, x_t} \beta_j(t+1)
\end{aligned}
$$

In the first two steps we rearrange terms and substitute in for our definition of Q. Then we use Bayes rule in deriving line four, followed by the definitions of α, β, A, and B, in line five. Similarly, the denominator can be represented by summing out over j the value of the numerator.

$$
\begin{aligned}
& \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i\} \\
&= \sum_{j=1}^{|S|} \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_{t-1} = s_i \wedge z_t = s_j\} \\
&= \frac{1}{P(\vec{x}; A, B)} \sum_{j=1}^{|S|} \sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\, x_t} \beta_j(t+1)
\end{aligned}
$$

Combining these expressions, we can fully characterize our maximum likelihood state transitions $\hat{A}_{ij}$ without needing to enumerate all possible labellings as:

$$
\hat{A}_{ij} = \frac{\sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\, x_t} \beta_j(t+1)}{\sum_{j=1}^{|S|} \sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\, x_t} \beta_j(t+1)}
$$

Similarly, we can represent the numerator for $\hat{B}_{jk}$ as:

$$
\begin{aligned}
& \sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j \wedge x_t = v_k\} \\
&= \frac{1}{P(\vec{x}; A, B)} \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_t = s_j \wedge x_t = v_k\} P(\vec{z}, \vec{x}; A, B) \\
&= \frac{1}{P(\vec{x}; A, B)} \sum_{i=1}^{|S|} \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_{t-1} = s_i \wedge z_t = s_j \wedge x_t = v_k\} P(\vec{z}, \vec{x}; A, B)
\end{aligned}
$$

Algorithm 3 Forward-Backward algorithm for HMM parameter learning

Initialization: Set A and B as random valid probability matrices
where $A_{i0} = 0$ and $B_{0k} = 0$ for $i = 1..|S|$ and $k = 1..|V|$.

Repeat until convergence {

(E-Step) Run the Forward and Backward algorithms to compute α_i and β_i for $i = 1..|S|$. Then set:

$$\gamma_t(i,j) := \alpha_i(t) A_{ij} B_{j\,x_t} \beta_j(t+1)$$

(M-Step) Re-estimate the maximum likelihood parameters as:

$$A_{ij} := \frac{\sum_{t=1}^{T} \gamma_t(i,j)}{\sum_{j=1}^{|S|} \sum_{t=1}^{T} \gamma_t(i,j)}$$

$$B_{jk} := \frac{\sum_{i=1}^{|S|} \sum_{t=1}^{T} 1\{x_t = v_k\}\, \gamma_t(i,j)}{\sum_{i=1}^{|S|} \sum_{t=1}^{T} \gamma_t(i,j)}$$

}

$$= \frac{1}{P(\vec{x}; A, B)} \sum_{i=1}^{|S|} \sum_{t=1}^{T} 1\{x_t = v_k\} \alpha_i(t) A_{ij} B_{j\,x_t} \beta_j(t+1)$$

And the denominator of $\hat{B}_{jk}$ as:

$$\begin{aligned}
&\sum_{\vec{z}} Q(\vec{z}) \sum_{t=1}^{T} 1\{z_t = s_j\} \\
= \; & \frac{1}{P(\vec{x}; A, B)} \sum_{i=1}^{|S|} \sum_{t=1}^{T} \sum_{\vec{z}} 1\{z_{t-1} = s_i \wedge z_t = s_j\} P(\vec{z}, \vec{x}; A, B) \\
= \; & \frac{1}{P(\vec{x}; A, B)} \sum_{i=1}^{|S|} \sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\,x_t} \beta_j(t+1)
\end{aligned}$$

Combining these expressions, we have the following form for our maximum likelihood emission probabilities as:

$$\hat{B}_{jk} = \frac{\sum_{i=1}^{|S|} \sum_{t=1}^{T} 1\{x_t = v_k\} \alpha_i(t) A_{ij} B_{j\,x_t} \beta_j(t+1)}{\sum_{i=1}^{|S|} \sum_{t=1}^{T} \alpha_i(t) A_{ij} B_{j\,x_t} \beta_j(t+1)}$$

Algorithm 2.4 shows a variant of the Forward-Backward Algorithm, or the Baum-Welch Algorithm for parameter learning in HMMs. In the

E-Step, rather than explicitly evaluating $Q(\vec{z})$ for all $\vec{z} \in S^T$, we compute a sufficient statistics $\gamma_t(i,j) = \alpha_i(t)A_{ij}B_{j\,x_t}\beta_j(t+1)$ that is proportional to the probability of transitioning between sate s_i and s_j at time t given all of our observations $\vec{x}$. The derived expressions for A_{ij} and B_{jk} are intuitively appealing. A_{ij} is computed as the expected number of transitions from s_i to s_j divided by the expected number of appearances of s_i. Similarly, B_{jk} is computed as the expected number of emissions of v_k from s_j divided by the expected number of appearances of s_j.

Like many applications of EM, parameter learning for HMMs is a non-convex problem with many local maxima. EM will converge to a maximum based on its initial parameters, so multiple runs might be in order. Also, it is often important to smooth the probability distributions represented by A and B so that no transition or emission is assigned 0 probability.

2.5 Further reading

There are many good sources for learning about Hidden Markov Models. For applications in NLP, I recommend consulting Jurafsky & Martin's draft second edition of *Speech and Language Processing*[1] or Manning & Schütze's *Foundations of Statistical Natural Language Processing*. Also, Eisner's HMM-in-a-spreadsheet [1] is a light-weight interactive way to play with an HMM that requires only a spreadsheet application.

References

[1] Jason Eisner. An interactive spreadsheet for teaching the forward-backward algorithm. In Dragomir Radev and Chris Brew, editors, *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL*, pages 10–18, 2002.

[1] http://www.cs.colorado.edu/~martin/slp2.html

Maximum Entropy and Exponential Families

April 9, 2019

Abstract

The goal of this note is to derive the exponential form of probability distribution from more basic considerations, in particular Entropy. It follows a description by ET Jaynes in Chapter 11 of his book *Probability Theory: the Logic of Science* [1].[1]

1 Motivating the Exponential Model

This section will motivate the exponential model form that we've seen in lecture.

The Setup The setup for our problem is that we are given a finite set of instances $\mathcal{I}$ and a set of m statistics (f_i, c_i) in which $f_i : \mathcal{I} \to \mathbb{R}$ and $c_i \in \mathbb{R}$. An instance (or possible world) is just an element in a set. We can think about a statistic as a measurement of an instance, it tells us the important features of that instance that are important for our model. More precisely, the only information we have about the instances is the values of f_i on these instances. Our goal is to find a probability function p such that

$$p : \mathcal{I} \to [0, 1] \text{ such that } \sum_{I \in \mathcal{I}} p(I) = 1.$$

The main goal of this note is to provide a set of assumptions under which such distributions have a specific functional form, the exponential family, that we saw in generalized linear model:

$$p_\theta(I) = Z^{-1}(\theta) \exp \{\theta \cdot f(I)\}$$

in which $\theta \in \mathbb{R}^m$ and $f(I) \in \mathbb{R}^m$ and $f(I)_i = f_i(I)$. Notice that there is exactly one parameter for each statistic. As we'll see for discrete distributions, we are able to derive this exponential form as a consequences of a maximizing entropy subject to matching the statistics.[2]

1.1 The problem: Too many distributions!

We'll see the problem of defining a distribution from statistics (measurements). We'll see that often there are often many probability distributions that satisfy our constraints, and we'll be forced to pick among them.[3]

[1] This work is available online in many places including `http://omega.albany.edu:8008/ETJ-PS/cc11g.ps`.

[2] Unfortunately, for continuous distributions, such a derivation does not work due to some technical issues with Entropy–this hasn't stopped folks from using it as justification.

[3] Throughout this section, it will be convenient to view p and f_j as functions from $\mathcal{I} \to \mathbb{R}$–and also as vectors indexed by $\mathcal{I}$. Their use should be clear from the context.

The Constraints We interpret a statistic as a constraint on p of the following form:

$$\mathbb{E}_p[f_i] = c_i \text{ i.e., } \sum_{I \in \mathcal{I}} f_i(I)p(I) = c_i$$

Let's get some notation to describe these constraints. Let $N = |\mathcal{I}|$ then the probability we are after is $p \in \mathbb{R}^N$ subject to constraints.

- There are m constraints of the form
$$f_j^T p = c_j \text{ for } j = 1, \ldots, m.$$
- A single constraint of the form $\sum_{i=1}^N p_i = 1$ to ensure that p is a probability distribution. We can write this more succinctly as $1_N^T p = 1$.
- We also have that $p_i \geq 0$ for $i = 1, \ldots, N$.

More compactly, we can write $F \in \mathbb{R}^{m \times N}$ such that $F_i = f_i$ for $i = 1, \ldots, m$. Then, we can compactly write all constraints in a matrix G as

$$G = \begin{pmatrix} 1_N \\ F \end{pmatrix} \in \mathbb{R}^{(m+1) \times N} \text{ so that } Gp = \begin{pmatrix} 1 \\ c \end{pmatrix}.$$

If $\mathsf{N}(G) = \emptyset$, then this means that p is uniquely defined as G has an inverse. In this case, $p = G^{-1}c$. However often m is much smaller than N, so that $\mathsf{N}(G) \neq \emptyset$–and there are many solutions that satisfy the constraints.

Example 1.1. *Suppose we have three possible worlds, i.e.,* $\mathcal{I} = \{I_1, I_2, I_3\}$ *and one statistic* $f(I_i) = i$ *and* $c = 2.5$. *Then, we have:*

$$G = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix} \text{ and } \mathsf{N}(G) = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$$

Let $p^{(1)} = (1/12, 1/3, 7/12)$ *then* $Gp = (1, 2.5)^T$*–but so do (infinitely) many others, in particular* $q(\alpha) = p^{(1)} + \alpha(1, -2, 1)$ *is valid so long as* $\alpha \in [-1/12, 1/6]$ *(due to positivity).*

Picking a probability distribution p In the case $\emptyset \neq \mathsf{N}(G)$, there are *many* probability distributions we can pick. All of these distributions can be written as follows:

$$p = p^{(0)} + p^{(1)} \text{ in which } p^{(0)} \in \mathsf{N}(G) \text{ and } p^{(1)} \text{ satisfies } Gp^{(1)} = \begin{pmatrix} 1 \\ c \end{pmatrix}$$

Example 1.2. *Continuing the computation above, we see* $p^{(0)} = \alpha(1, -2, 1)$ *is a vector in* $\mathsf{N}(G)$.

Which p should we pick? Well, we'll use one method called the method of maximum entropy. In turn, this will lead to the fact that our function p has a very special form–the form of exponential models!

1.2 Entropy

To pick among the distributions, we'll need some scoring method.[4] We'll cut to the chase here and define the entropy, which is a function on probability distributions $p \in \mathbb{R}^N$ such that $p \geq 0$ and $p^T 1_N = 1$.

$$H(p) = -\sum_{i=1}^N p_i \log p_i$$

[4] A few natural methods don't work as we might think they should (minimizing variance, etc.) See [1, Ch.11] for a description of these alternative approaches.

Effectively, the entropy rewards one for "spreading" the distribution out more. One can motivate Entropy from axioms, and either Jaynes or the Wikipedia page is pretty good on this account.[5]. The intuition should be that entropy can be used to select the *least informative* prior, it's a way of making as few additional assumptions as possible. In other words, we want to encode the prior information given by the constraints on the statistics while being as "objective" or "agnostic" as possible. This is called the maximum entropy principle.

For example, one can verify that under no constraints, $H(p)$ is maximized with $p_i = N^{-1}$–that is all alternatives have equal probability. This is what we mean by spread out.

We'll pick the distribution that maximizes entropy subject to our constraints. Mathematically, we'll examine:

$$\max_{p \in \mathbb{R}^N} H(p) \text{ s.t. } p^T 1 = 1, p \geq 0, \text{ and } Fp = c$$

We will not discuss it, but under appropriate conditions there is a unique solution p.

1.3 The Lagrangian

We'll create a function called the Lagrangian that has the property that any critical point of the Lagrangian is a critical point of the *constrained* problem. We will show that all critical points of the Lagrangian (and so our original problem) can be written in the exponential format we described above.

To simplify our discussion, let's imagine that $p > 0$, i.e,. there are no possible worlds I such that $p(I) = 0$. In this case, our problem reduces to:

$$\max_{p \in \mathbb{R}^N} H(p) \text{ s.t. } Fp = c \text{ and } 1_N^T p = 1$$

We can write the Lagrangian $\Lambda : \mathbb{R}^N \times (\mathbb{R}^m \times \mathbb{R}) \to \mathbb{R}$ as follows:

$$\Lambda(p; \theta, \lambda) = H(p) + \theta^T (Fp - c) + \lambda(1^T p - 1)$$

The special property of Λ is that any critical point of our original solution, in particular any maximum or minimum corresponds to a critical point of the Lagrangian. Thus, if we prove something about critical points of the Lagrangian, we prove something about the critical points of the original function. Later in the course, we'll see more sophisticated uses of Lagrangians but for now we include a simple derivation below to give a hint what's going on. For this section, we'll assume this special property is true.

Due to that special property, we find the critical points of Λ by differentiating with respect to p_i and setting the resulting equations to 0.

$$\frac{\partial}{\partial p_i} \left[H(p) + \theta^T (Fp - c) + \lambda(1^T p - 1) \right] = -(\log p_i + 1) + \sum_{j=1}^{m} \theta_j f_j(I_i) + \lambda = -(\log p_i + 1) + \theta^T f(I_i) + \lambda$$

Setting this expression equal to 0 and solving for p_i we learn:

$$p_i = e^{\lambda - 1} \exp \left\{ \theta^T f(I_i) \right\}$$

which is of the right form–except that we have one too many parameters, namely λ. Nevertheless, this is remarkable: at a critical point, it's always the case that the exponential family "pops out"!

[5] https://en.wikipedia.org/wiki/Entropy_(information_theory)#Rationale

Eliminating λ The parameter λ can be eliminated, which is the final step to match our original claimed exponential form. To do so, we sum over all the p_i which we know on one hand is equal to 1, and the other hand, we have the above expression for p_i. This gives us the following equation:

$$\sum_{i=1}^{N} p_i = 1 \text{ and } \sum_{i=1}^{N} p_i = e^{\lambda-1}\left(\sum_{i=1}^{N} \exp\left\{\theta^T f(I_i)\right\}\right) \text{ thus } e^{-\lambda+1} = \left(\sum_{i=1}^{N} \exp\left\{\theta^T f(I_i)\right\}\right)$$

Thus, we have expressed λ as a function of θ and we can eliminate it. To do so, we write:

$$Z(\theta) = \sum_{i=1}^{N} \exp\left\{\theta^T f(I_i)\right\} \text{ and } p_i = Z(\theta)^{-1} \exp\{\theta^T f(I_i)\}$$

This function Z is called the *partition function* that we saw in lecture. The above form is the claimed exponential form that has one parameter per constraint.

2 Why the Lagrangian?

We observe that this is a constrained optimization problem with *linear* constraints.[6]

Let r be the rank of G and so $\mathsf{dim}(\mathsf{N}(G)) = N - r$. We create a function $\phi : \mathbb{R}^{N-r} \to \mathbb{R}$ such that there is a map between any point in the domain of ϕ and a feasible solution to our constrained problem, and moreover ϕ will take the same value as H. In contrast to our original constrained problem, ϕ has an unconstrained domain (all of $\mathbb{R}^{N-r}$), and so we can apply standard calculus to find its critical points. To that end, we define a (linear) map $B \in \mathbb{R}^{N\times(N-r)}$ that has rank $N - r$. We also insist that $B^T B = I_{N-r}$. Such a B exists, as it is simply the first $N-r$ columns of a change of basis matrix from the standard basis to an orthonormal basis for $\mathsf{N}(G)$.

$$\phi(x) = H(Bx + p^{(1)}),$$

where $p^{(1)}$ is a fixed vector satisfying $Gp^{(1)} = \begin{pmatrix}1\\c\end{pmatrix}$.

Observe that for any $x \in \mathbb{R}^{N-r}$, $Bx \in \mathsf{N}(G)$ so that $G(Bx + p^{(1)}) = Gp^{(1)} = \begin{pmatrix}1\\c\end{pmatrix}$ and so $Bx + p^{(1)}$ is feasible. Moreover, B is a bijection from $\mathbb{R}^{N-r}$ to the set of feasible solutions.[7] Importantly, ϕ is now *unconstrained*, and so any saddle point (and so any maximum or minimum) must satisfy:

$$\nabla_x \phi(x) = 0$$

Gradient Decomposition Any critical point of H yields a critical point of ϕ, that is, if $p = p^{(0)} + p^{(1)}$ is a critical point of H then $x = B^T p^{(0)}$ is a critical point of ϕ. Consider any critical point p, then we can uniquely decompose the gradient as:

$$\nabla_p H(p) = g_0 + g_1 \text{ in which } g_0 \in \mathsf{N}(G) \text{ and } g_1 \in \mathsf{N}(G)^\perp.$$

We claim $g_0 = B\nabla\phi(B^T p)$ or equivalently $B^T g = \nabla_x \phi(B^T p)$. From direct calculation, $\nabla_x \phi(x) = \nabla_x H(Bx+p^{(1)}) = B^T \nabla_p H(p^{(0)}+p^{(1)}) = B^T \nabla_p H(p) = B^T g_0$, where the last equality is due to $g_1 \in \mathsf{N}(G)^\perp$. A critical point of H satisfying the constraints must not change along any direction that satisfies the constraints, which is to say that we must have $g_0 = 0$. Very roughly, one can have the intuition that if p were a maximum (or minimum), then if g_0 were non-zero there would be a way to strictly increase (or decrease) the function in a neighbor around p–contradicting p being a maximum (minimum).

[6]One can form the Lagrangian for non-linear constraints, but to derive it we need to use fancier math like the implicit function theorem. We only need linear constraints for our applications.

[7]For contradiction, suppose p, q are distinct feasible solutions then, $p \neq q$ but $B^T p = B^T q$ but we can write $p = p^{(0)} + p^{(1)}$ and $q = q^{(0)} + p^{(1)}$ from the above. However, $B^T p = B^T q$ implies that $B^T p^{(0)} = B^T q^{(0)}$. In turn since B is a bijection on $\mathsf{N}(G)$ this implies that $p^{(0)} = q^{(0)}$.

Lagrangian Since $g_1 \in \mathsf{N}(G)^\perp = \mathsf{R}(G^T)$ (see the fundamental theorem of linear algebra), we can find a $\theta(p)$ such that $g_1 = -G^T\theta(p)$, which motivates the following functional form:

$$\Lambda(p, \theta(p)) = H(p) + \theta(p)^T(Gp - c)$$

By the definition of $\theta(p)$, we have:

$$\nabla_p \Lambda(p, \theta(p)) = g_0 + g_1 + \theta(p)^T G = g_0.$$

That is, for any critical point p of the original function (which corresponds to $g_0 = 0$) we can select $\theta(p)$ so that it is a critical point of $\Lambda(p, \theta)$. Informally, the multipliers combines the rows of G to cancel g_1, the component of the gradient in the direction of the constraints. This establishes that any critical point of the original constrained function is also a critical point of the Lagrangian.

References

[1] Jaynes, Edwin T, *Probability theory: The logic of science*, Cambridge University Press, 2003.

The Multivariate Gaussian Distribution

Chuong B. Do

October 10, 2008

A vector-valued random variable $X = \begin{bmatrix} X_1 & \cdots & X_n \end{bmatrix}^T$ is said to have a **multivariate normal (or Gaussian) distribution** with mean $\mu \in \mathbf{R}^n$ and covariance matrix $\Sigma \in \mathbf{S}^n_{++}$[1] if its probability density function[2] is given by

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

We write this as $X \sim \mathcal{N}(\mu, \Sigma)$. In these notes, we describe multivariate Gaussians and some of their basic properties.

1 Relationship to univariate Gaussians

Recall that the density function of a **univariate normal (or Gaussian) distribution** is given by

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right).$$

Here, the argument of the exponential function, $-\frac{1}{2\sigma^2}(x-\mu)^2$, is a quadratic function of the variable x. Furthermore, the parabola points downwards, as the coefficient of the quadratic term is negative. The coefficient in front, $\frac{1}{\sqrt{2\pi}\sigma}$, is a constant that does not depend on x; hence, we can think of it as simply a "normalization factor" used to ensure that

$$\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) = 1.$$

[1] Recall from the section notes on linear algebra that $\mathbf{S}^n_{++}$ is the space of symmetric positive definite $n \times n$ matrices, defined as

$$\mathbf{S}^n_{++} = \left\{A \in \mathbf{R}^{n\times n} : A = A^T \text{ and } x^T A x > 0 \text{ for all } x \in \mathbf{R}^n \text{ such that } x \neq 0\right\}.$$

[2] In these notes, we use the notation $p(\bullet)$ to denote density functions, instead of $f_X(\bullet)$ (as in the section notes on probability theory).

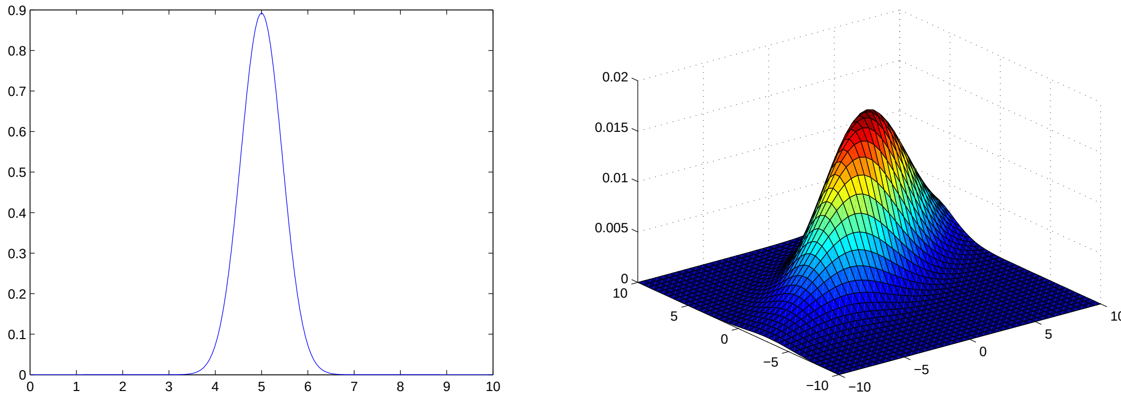


Figure 1: The figure on the left shows a univariate Gaussian density for a single variable X. The figure on the right shows a multivariate Gaussian density over two variables X_1 and X_2.

In the case of the multivariate Gaussian density, the argument of the exponential function, $-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)$, is a **quadratic form** in the vector variable x. Since Σ is positive definite, and since the inverse of any positive definite matrix is also positive definite, then for any non-zero vector z, $z^T\Sigma^{-1}z > 0$. This implies that for any vector $x \neq \mu$,

$$(x-\mu)^T\Sigma^{-1}(x-\mu) > 0$$
$$-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu) < 0.$$

Like in the univariate case, you can think of the argument of the exponential function as being a downward opening quadratic bowl. The coefficient in front (i.e., $\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}$) has an even more complicated form than in the univariate case. However, it still does not depend on x, and hence it is again simply a normalization factor used to ensure that

$$\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx_1dx_2\cdots dx_n = 1.$$

2 The covariance matrix

The concept of the **covariance matrix** is vital to understanding multivariate Gaussian distributions. Recall that for a pair of random variables X and Y, their **covariance** is defined as

$$Cov[X,Y] = E[(X-E[X])(Y-E[Y])] = E[XY] - E[X]E[Y].$$

When working with multiple variables, the covariance matrix provides a succinct way to summarize the covariances of all pairs of variables. In particular, the covariance matrix, which we usually denote as Σ, is the $n \times n$ matrix whose (i,j)th entry is $Cov[X_i, X_j]$.

The following proposition (whose proof is provided in the Appendix A.1) gives an alternative way to characterize the covariance matrix of a random vector X:

Proposition 1. *For any random vector X with mean μ and covariance matrix Σ,*

$$\Sigma = E[(X-\mu)(X-\mu)^T] = E[XX^T] - \mu\mu^T. \tag{1}$$

In the definition of multivariate Gaussians, we required that the covariance matrix Σ be symmetric positive definite (i.e., $\Sigma \in \mathbf{S}^n_{++}$). Why does this restriction exist? As seen in the following proposition, the covariance matrix of *any* random vector must always be symmetric positive semidefinite:

Proposition 2. *Suppose that Σ is the covariance matrix corresponding to some random vector X. Then Σ is symmetric positive semidefinite.*

Proof. The symmetry of Σ follows immediately from its definition. Next, for any vector $z \in \mathbf{R}^n$, observe that

$$\begin{aligned}
z^T\Sigma z &= \sum_{i=1}^n \sum_{j=1}^n (\Sigma_{ij} z_i z_j) && (2)\\
&= \sum_{i=1}^n \sum_{j=1}^n (Cov[X_i, X_j] \cdot z_i z_j)\\
&= \sum_{i=1}^n \sum_{j=1}^n (E[(X_i - E[X_i])(X_j - E[X_j])] \cdot z_i z_j)\\
&= E\left[\sum_{i=1}^n \sum_{j=1}^n (X_i - E[X_i])(X_j - E[X_j]) \cdot z_i z_j\right]. && (3)
\end{aligned}$$

Here, (2) follows from the formula for expanding a quadratic form (see section notes on linear algebra), and (3) follows by linearity of expectations (see probability notes).

To complete the proof, observe that the quantity inside the brackets is of the form $\sum_i \sum_j x_i x_j z_i z_j = (x^T z)^2 \geq 0$ (see problem set #1). Therefore, the quantity inside the expectation is always nonnegative, and hence the expectation itself must be nonnegative. We conclude that $z^T \Sigma z \geq 0$. □

From the above proposition it follows that Σ must be symmetric positive semidefinite in order for it to be a valid covariance matrix. However, in order for Σ^{-1} to exist (as required in the definition of the multivariate Gaussian density), then Σ must be invertible and hence full rank. Since any full rank symmetric positive semidefinite matrix is necessarily symmetric positive definite, it follows that Σ must be symmetric positive definite.

3 The diagonal covariance matrix case

To get an intuition for what a multivariate Gaussian is, consider the simple case where $n = 2$, and where the covariance matrix Σ is diagonal, i.e.,

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$

In this case, the multivariate Gaussian density has the form,

$$\begin{aligned} p(x; \mu, \Sigma) &= \frac{1}{2\pi \begin{vmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{vmatrix}^{1/2}} \exp\left(-\frac{1}{2} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix}^T \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}^{-1} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix} \right) \\ &= \frac{1}{2\pi(\sigma_1^2 \cdot \sigma_2^2 - 0 \cdot 0)^{1/2}} \exp\left(-\frac{1}{2} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix}^T \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 \\ 0 & \frac{1}{\sigma_2^2} \end{bmatrix} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix} \right), \end{aligned}$$

where we have relied on the explicit formula for the determinant of a 2×2 matrix[3], and the fact that the inverse of a diagonal matrix is simply found by taking the reciprocal of each diagonal entry. Continuing,

$$\begin{aligned} p(x; \mu, \Sigma) &= \frac{1}{2\pi\sigma_1\sigma_2} \exp\left(-\frac{1}{2} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix}^T \begin{bmatrix} \frac{1}{\sigma_1^2}(x_1 - \mu_1) \\ \frac{1}{\sigma_2^2}(x_2 - \mu_2) \end{bmatrix} \right) \\ &= \frac{1}{2\pi\sigma_1\sigma_2} \exp\left(-\frac{1}{2\sigma_1^2}(x_1 - \mu_1)^2 - \frac{1}{2\sigma_2^2}(x_2 - \mu_2)^2 \right) \\ &= \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left(-\frac{1}{2\sigma_1^2}(x_1 - \mu_1)^2 \right) \cdot \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left(-\frac{1}{2\sigma_2^2}(x_2 - \mu_2)^2 \right). \end{aligned}$$

The last equation we recognize to simply be the product of two independent Gaussian densities, one with mean μ_1 and variance σ_1^2, and the other with mean μ_2 and variance σ_2^2.

More generally, one can show that an n-dimensional Gaussian with mean $\mu \in \mathbf{R}^n$ and diagonal covariance matrix $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2)$ is the same as a collection of n independent Gaussian random variables with mean μ_i and variance σ_i^2, respectively.

4 Isocontours

Another way to understand a multivariate Gaussian conceptually is to understand the shape of its **isocontours**. For a function $f : \mathbf{R}^2 \to \mathbf{R}$, an isocontour is a set of the form

$$\left\{ x \in \mathbf{R}^2 : f(x) = c \right\}.$$

for some $c \in \mathbf{R}$.[4]

[3] Namely, $\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$.

[4] Isocontours are often also known as **level curves**. More generally, a **level set** of a function $f : \mathbf{R}^n \to \mathbf{R}$, is a set of the form $\left\{ x \in \mathbf{R}^2 : f(x) = c \right\}$ for some $c \in \mathbf{R}$.

4.1 Shape of isocontours

What do the isocontours of a multivariate Gaussian look like? As before, let's consider the case where $n = 2$, and Σ is diagonal, i.e.,

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$

As we showed in the last section,

$$p(x;\mu,\Sigma) = \frac{1}{2\pi\sigma_1\sigma_2} \exp\left(-\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2 - \frac{1}{2\sigma_2^2}(x_2-\mu_2)^2 \right). \tag{4}$$

Now, let's consider the level set consisting of all points where $p(x;\mu,\Sigma) = c$ for some constant $c \in \mathbf{R}$. In particular, consider the set of all $x_1, x_2 \in \mathbf{R}$ such that

$$\begin{aligned}
c &= \frac{1}{2\pi\sigma_1\sigma_2} \exp\left(-\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2 - \frac{1}{2\sigma_2^2}(x_2-\mu_2)^2 \right) \\
2\pi c\sigma_1\sigma_2 &= \exp\left(-\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2 - \frac{1}{2\sigma_2^2}(x_2-\mu_2)^2 \right) \\
\log(2\pi c\sigma_1\sigma_2) &= -\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2 - \frac{1}{2\sigma_2^2}(x_2-\mu_2)^2 \\
\log\left(\frac{1}{2\pi c\sigma_1\sigma_2}\right) &= \frac{1}{2\sigma_1^2}(x_1-\mu_1)^2 + \frac{1}{2\sigma_2^2}(x_2-\mu_2)^2 \\
1 &= \frac{(x_1-\mu_1)^2}{2\sigma_1^2 \log\left(\frac{1}{2\pi c\sigma_1\sigma_2}\right)} + \frac{(x_2-\mu_2)^2}{2\sigma_2^2 \log\left(\frac{1}{2\pi c\sigma_1\sigma_2}\right)}.
\end{aligned}$$

Defining

$$r_1 = \sqrt{2\sigma_1^2 \log\left(\frac{1}{2\pi c\sigma_1\sigma_2}\right)} \qquad r_2 = \sqrt{2\sigma_2^2 \log\left(\frac{1}{2\pi c\sigma_1\sigma_2}\right)},$$

it follows that

$$1 = \left(\frac{x_1-\mu_1}{r_1}\right)^2 + \left(\frac{x_2-\mu_2}{r_2}\right)^2. \tag{5}$$

Equation (5) should be familiar to you from high school analytic geometry: it is the equation of an **axis-aligned ellipse**, with center (μ_1, μ_2), where the x_1 axis has length $2r_1$ and the x_2 axis has length $2r_2$!

4.2 Length of axes

To get a better understanding of how the shape of the level curves vary as a function of the variances of the multivariate Gaussian distribution, suppose that we are interested in

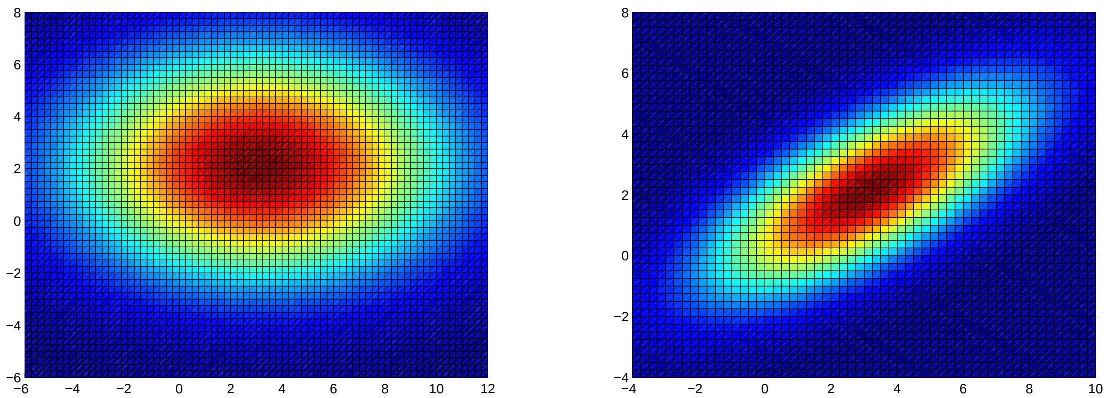


Figure 2:
The figure on the left shows a heatmap indicating values of the density function for an axis-aligned multivariate Gaussian with mean $\mu = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$ and diagonal covariance matrix $\Sigma = \begin{bmatrix} 25 & 0 \\ 0 & 9 \end{bmatrix}$. Notice that the Gaussian is centered at $(3, 2)$, and that the isocontours are all elliptically shaped with major/minor axis lengths in a 5:3 ratio. The figure on the right shows a heatmap indicating values of the density function for a non axis-aligned multivariate Gaussian with mean $\mu = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$ and covariance matrix $\Sigma = \begin{bmatrix} 10 & 5 \\ 5 & 5 \end{bmatrix}$. Here, the ellipses are again centered at $(3, 2)$, but now the major and minor axes have been rotated via a linear transformation.

the values of r_1 and r_2 at which c is equal to a fraction $1/e$ of the peak height of Gaussian density.

First, observe that maximum of Equation (4) occurs where $x_1 = \mu_1$ and $x_2 = \mu_2$. Substituting these values into Equation (4), we see that the peak height of the Gaussian density is $\frac{1}{2\pi\sigma_1\sigma_2}$.

Second, we substitute $c = \frac{1}{e}\left(\frac{1}{2\pi\sigma_1\sigma_2}\right)$ into the equations for r_1 and r_2 to obtain

$$r_1 = \sqrt{2\sigma_1^2 \log\left(\frac{1}{2\pi\sigma_1\sigma_2 \cdot \frac{1}{e}\left(\frac{1}{2\pi\sigma_1\sigma_2}\right)}\right)} = \sigma_1\sqrt{2}$$
$$r_2 = \sqrt{2\sigma_2^2 \log\left(\frac{1}{2\pi\sigma_1\sigma_2 \cdot \frac{1}{e}\left(\frac{1}{2\pi\sigma_1\sigma_2}\right)}\right)} = \sigma_2\sqrt{2}.$$

From this, it follows that the axis length needed to reach a fraction $1/e$ of the peak height of the Gaussian density in the ith dimension grows in proportion to the standard deviation σ_i. Intuitively, this again makes sense: the smaller the variance of some random variable x_i, the more "tightly" peaked the Gaussian distribution in that dimension, and hence the smaller the radius r_i.

4.3 Non-diagonal case, higher dimensions

Clearly, the above derivations rely on the assumption that Σ is a diagonal matrix. However, in the non-diagonal case, it turns out that the picture is not all that different. Instead of being an axis-aligned ellipse, the isocontours turn out to be simply **rotated ellipses**. Furthermore, in the n-dimensional case, the level sets form geometrical structures known as **ellipsoids** in $\mathbf{R}^n$.

5 Linear transformation interpretation

In the last few sections, we focused primarily on providing an intuition for how multivariate Gaussians with diagonal covariance matrices behaved. In particular, we found that an n-dimensional multivariate Gaussian with diagonal covariance matrix could be viewed simply as a collection of n independent Gaussian-distributed random variables with means and variances μ_i and σ_i^2, respectvely. In this section, we dig a little deeper and provide a quantitative interpretation of multivariate Gaussians when the covariance matrix is not diagonal.

The key result of this section is the following theorem (see proof in Appendix A.2).

Theorem 1. *Let $X \sim \mathcal{N}(\mu, \Sigma)$ for some $\mu \in \mathbf{R}^n$ and $\Sigma \in \mathbf{S}^n_{++}$. Then, there exists a matrix $B \in \mathbf{R}^{n\times n}$ such that if we define $Z = B^{-1}(X - \mu)$, then $Z \sim \mathcal{N}(0, I)$.*

To understand the meaning of this theorem, note that if $Z \sim \mathcal{N}(0, I)$, then using the analysis from Section 4, Z can be thought of as a collection of n independent standard normal random variables (i.e., $Z_i \sim \mathcal{N}(0, 1)$). Furthermore, if $Z = B^{-1}(X - \mu)$ then $X = BZ + \mu$ follows from simple algebra.

Consequently, the theorem states that any random variable X with a multivariate Gaussian distribution can be interpreted as the result of applying a linear transformation ($X = BZ + \mu$) to some collection of n independent standard normal random variables (Z).

Appendix A.1

Proof. We prove the first of the two equalities in (1); the proof of the other equality is similar.

$$
\begin{aligned}
\Sigma &= \begin{bmatrix} Cov[X_1, X_1] & \cdots & Cov[X_1, X_n] \\ \vdots & \ddots & \vdots \\ Cov[X_n, X_1] & \cdots & Cov[X_n, X_n] \end{bmatrix} \\
&= \begin{bmatrix} E[(X_1 - \mu_1)^2] & \cdots & E[(X_1 - \mu_1)(X_n - \mu_n)] \\ \vdots & \ddots & \vdots \\ E[(X_n - \mu_n)(X_1 - \mu_1)] & \cdots & E[(X_n - \mu_n)^2] \end{bmatrix} \\
&= E \begin{bmatrix} (X_1 - \mu_1)^2 & \cdots & (X_1 - \mu_1)(X_n - \mu_n) \\ \vdots & \ddots & \vdots \\ (X_n - \mu_n)(X_1 - \mu_1) & \cdots & (X_n - \mu_n)^2 \end{bmatrix} && (6) \\
&= E \left[\begin{bmatrix} X_1 - \mu_1 \\ \vdots \\ X_n - \mu_n \end{bmatrix} \begin{bmatrix} X_1 - \mu_1 & \cdots & X_n - \mu_n \end{bmatrix} \right] && (7) \\
&= E \left[(X - \mu)(X - \mu)^T \right].
\end{aligned}
$$

Here, (6) follows from the fact that the expectation of a matrix is simply the matrix found by taking the componentwise expectation of each entry. Also, (7) follows from the fact that for any vector $z \in \mathbf{R}^n$,

$$
zz^T = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \begin{bmatrix} z_1 & z_2 & \cdots z_n \end{bmatrix} = \begin{bmatrix} z_1 z_1 & z_1 z_2 & \cdots & z_1 z_n \\ z_2 z_1 & z_2 z_2 & \cdots & z_2 z_n \\ \vdots & \vdots & \ddots & \vdots \\ z_n z_1 & z_n z_2 & \cdots & z_n z_n \end{bmatrix}.
$$

□

Appendix A.2

We restate the theorem below:

Theorem 1. *Let $X \sim \mathcal{N}(\mu, \Sigma)$ for some $\mu \in \mathbf{R}^n$ and $\Sigma \in \mathbf{S}^n_{++}$. Then, there exists a matrix $B \in \mathbf{R}^{n \times n}$ such that if we define $Z = B^{-1}(X - \mu)$, then $Z \sim \mathcal{N}(0, I)$.*

The derivation of this theorem requires some advanced linear algebra and probability theory and can be skipped for the purposes of this class. Our argument will consist of two parts. First, we will show that the covariance matrix Σ can be factorized as $\Sigma = BB^T$ for some invertible matrix B. Second, we will perform a "change-of-variable" from X to a different vector valued random variable Z using the relation $Z = B^{-1}(X - \mu)$.

Step 1: Factorizing the covariance matrix. Recall the following two properties of symmetric matrices from the notes on linear algebra[5]:

1. Any real symmetric matrix $A \in \mathbf{R}^{n\times n}$ can always be represented as $A = U\Lambda U^T$, where U is a full rank orthogonal matrix containing of the eigenvectors of A as its columns, and Λ is a diagonal matrix containing A's eigenvalues.

2. If A is symmetric positive definite, all its eigenvalues are positive.

Since the covariance matrix Σ is positive definite, using the first fact, we can write $\Sigma = U\Lambda U^T$ for some appropriately defined matrices U and Λ. Using the second fact, we can define $\Lambda^{1/2} \in \mathbf{R}^{n\times n}$ to be the diagonal matrix whose entries are the square roots of the corresponding entries from Λ. Since $\Lambda = \Lambda^{1/2}(\Lambda^{1/2})^T$, we have

$$\Sigma = U\Lambda U^T = U\Lambda^{1/2}(\Lambda^{1/2})^T U^T = U\Lambda^{1/2}(U\Lambda^{1/2})^T = BB^T,$$

where $B = U\Lambda^{1/2}$.[6] In this case, then $\Sigma^{-1} = B^{-T}B^{-1}$, so we can rewrite the standard formula for the density of a multivariate Gaussian as

$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|BB^T|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T B^{-T}B^{-1}(x-\mu)\right). \tag{8}$$

Step 2: Change of variables. Now, define the vector-valued random variable $Z = B^{-1}(X-\mu)$. A basic formula of probability theory, which we did not introduce in the section notes on probability theory, is the "change-of-variables" formula for relating vector-valued random variables:

> Suppose that $X = \begin{bmatrix} X_1 & \cdots & X_n \end{bmatrix}^T \in \mathbf{R}^n$ is a vector-valued random variable with joint density function $f_X : \mathbf{R}^n \to \mathbf{R}$. If $Z = H(X) \in \mathbf{R}^n$ where H is a bijective, differentiable function, then Z has joint density $f_Z : \mathbf{R}^n \to \mathbf{R}$, where
>
> $$f_Z(z) = f_X(x) \cdot \left| \det\left(\begin{bmatrix} \frac{\partial x_1}{\partial z_1} & \cdots & \frac{\partial x_1}{\partial z_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_n}{\partial z_1} & \cdots & \frac{\partial x_n}{\partial z_n} \end{bmatrix} \right) \right|.$$

Using the change-of-variable formula, one can show (after some algebra, which we'll skip) that the vector variable Z has the following joint density:

$$p_Z(z) = \frac{1}{(2\pi)^{n/2}} \exp\left(-\frac{1}{2}z^T z\right). \tag{9}$$

The claim follows immediately. $\square$

[5]See section on "Eigenvalues and Eigenvectors of Symmetric Matrices."

[6]To show that B is invertible, it suffices to observe that U is an invertible matrix, and right-multiplying U by a diagonal matrix (with no zero diagonal entries) will rescale its columns but will not change its rank.

More on Multivariate Gaussians

Chuong B. Do

November 21, 2008

Up to this point in class, you have seen multivariate Gaussians arise in a number of applications, such as the probabilistic interpretation of linear regression, Gaussian discriminant analysis, mixture of Gaussians clustering, and most recently, factor analysis. In these lecture notes, we attempt to demystify some of the fancier properties of multivariate Gaussians that were introduced in the recent factor analysis lecture. The goal of these notes is to give you some intuition into where these properties come from, so that you can use them with confidence on your homework (hint hint!) and beyond.

1 Definition

A vector-valued random variable $x \in \mathbf{R}^n$ is said to have a **multivariate normal (or Gaussian) distribution** with mean $\mu \in \mathbf{R}^n$ and covariance matrix $\Sigma \in \mathbf{S}^n_{++}$[1] if its probability density function is given by

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

We write this as $x \sim \mathcal{N}(\mu, \Sigma)$.

2 Gaussian facts

Multivariate Gaussians turn out to be extremely handy in practice due to the following facts:

- **Fact #1:** If you know the mean μ and covariance matrix Σ of a Gaussian random variable x, you can write down the probability density function for x directly.

[1]Recall from the section notes on linear algebra that $\mathbf{S}^n_{++}$ is the space of symmetric positive definite $n \times n$ matrices, defined as

$$\mathbf{S}^n_{++} = \left\{A \in \mathbf{R}^{n\times n} : A = A^T \text{ and } x^T Ax > 0 \text{ for all } x \in \mathbf{R}^n \text{ such that } x \neq 0\right\}.$$

- **Fact #2:** The following Gaussian integrals have closed-form solutions:

$$\int_{x\in\mathbf{R}^n} p(x;\mu,\Sigma)dx = \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} p(x;\mu,\Sigma)dx_1\ldots dx_n = 1$$

$$\int_{x\in\mathbf{R}^n} x_i p(x;\mu,\sigma^2)dx = \mu_i$$

$$\int_{x\in\mathbf{R}^n} (x_i-\mu_i)(x_j-\mu_j)p(x;\mu,\sigma^2)dx = \Sigma_{ij}.$$

- **Fact #3:** Gaussians obey a number of *closure* properties:
  - The sum of independent Gaussian random variables is Gaussian.
  - The marginal of a joint Gaussian distribution is Gaussian.
  - The conditional of a joint Gaussian distribution is Gaussian.

At first glance, some of these facts, in particular facts #1 and #2, may seem either intuitively obvious or at least plausible. What is probably not so clear, however, is why these facts are so powerful. In this document, we'll provide some intuition for how these facts can be used when performing day-to-day manipulations dealing with multivariate Gaussian random variables.

3 Closure properties

In this section, we'll go through each of the closure properties described earlier, and we'll either prove the property using facts #1 and #2, or we'll at least give some type of intuition as to why the property is true.

The following is a quick roadmap of what we'll cover:

	sums	marginals	conditionals
why is it Gaussian?	no	yes	yes
resulting density function	yes	yes	yes

3.1 Sum of independent Gaussians is Gaussian

The formal statement of this rule is:

> Suppose that $y \sim \mathcal{N}(\mu,\Sigma)$ and $z \sim \mathcal{N}(\mu',\Sigma')$ are independent Gaussian distributed random variables, where $\mu,\mu' \in \mathbf{R}^n$ and $\Sigma,\Sigma' \in \mathbf{S}^n_{++}$. Then, their sum is also Gaussian:
>
> $$y+z \sim \mathcal{N}(\mu+\mu',\Sigma+\Sigma').$$

Before we prove anything, here are some observations:

1. The first thing to point out is that the importance of the independence assumption in the above rule. To see why this matters, suppose that $y \sim \mathcal{N}(\mu, \Sigma)$ for some mean vector μ and covariance matrix Σ, and suppose that $z = -y$. Clearly, z also has a Gaussian distribution (in fact, $z \sim \mathcal{N}(-\mu, \Sigma)$, but $y + z$ is identically zero!

2. The second thing to point out is a point of confusion for many students: if we add together two Gaussian densities ("bumps" in multidimensional space), wouldn't we get back some bimodal (i.e., "two-humped" density)? Here, the thing to realize is that the density of the random variable $y + z$ in this rule is NOT found by simply adding the densities of the individual random variables y and z. Rather, the density of $y + z$ will actually turn out to be a *convolution* of the densities for y and z.[2] To show that the convolution of two Gaussian densities gives a Gaussian density, however, is beyond the scope of this class.

Instead, let's just use the observation that the convolution does give some type of Gaussian density, along with Fact #1, to figure out what the density, $p(y + z|\mu, \Sigma)$ would be, if we were to actually compute the convolution. How can we do this? Recall that from Fact #1, a Gaussian distribution is fully specified by its mean vector and covariance matrix. If we can determine what these are, then we're done.

But this is easy! For the mean, we have

$$E[y_i + z_i] = E[y_i] + E[z_i] = \mu_i + \mu'_i$$

from linearity of expectations. Therefore, the mean of $y + z$ is simply $\mu + \mu'$. Also, the (i, j)th entry of the covariance matrix is given by

$$\begin{aligned}
&E[(y_i + z_i)(y_j + z_j)] - E[y_i + z_i]E[y_j + z_j] \\
&\quad = E[y_i y_j + z_i y_j + y_i z_j + z_i z_j] - (E[y_i] + E[z_i])(E[y_j] + E[z_j]) \\
&\quad = E[y_i y_j] + E[z_i y_j] + E[y_i z_j] + E[z_i z_j] - E[y_i]E[y_j] - E[z_i]E[y_j] - E[y_i]E[z_j] - E[z_i][z_j] \\
&\quad = (E[y_i y_j] - E[y_i]E[y_j]) + (E[z_i z_j] - E[z_i]E[z_j]) \\
&\qquad + (E[z_i y_j] - E[z_i]E[y_j]) + (E[y_i z_j] - E[y_i]E[z_j]).
\end{aligned}$$

Using the fact that y and z are independent, we have $E[z_i y_j] = E[z_i]E[y_j]$ and $E[y_i z_j] = E[y_i]E[z_j]$. Therefore, the last two terms drop out, and we are left with,

$$\begin{aligned}
&E[(y_i + z_i)(y_j + z_j)] - E[y_i + z_i]E[y_j + z_j] \\
&\quad = (E[y_i y_j] - E[y_i]E[y_j]) + (E[z_i z_j] - E[z_i]E[z_j]) \\
&\quad = \Sigma_{ij} + \Sigma'_{ij}.
\end{aligned}$$

[2] For example, if y and z were univariate Gaussians (i.e., $y \sim \mathcal{N}(\mu, \sigma^2)$, $z \sim \mathcal{N}(\mu', \sigma'^2)$), then the convolution of their probability densities is given by

$$\begin{aligned}
p(y + z; \mu, \mu', \sigma^2, \sigma'^2) &= \int_{-\infty}^{\infty} p(w; \mu, \sigma^2) p(y + z - w; \mu', \sigma'^2) dw \\
&= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(w - \mu)^2\right) \cdot \frac{1}{\sqrt{2\pi}\sigma'} \exp\left(-\frac{1}{2\sigma'^2}(y + z - w - \mu')^2\right) dw
\end{aligned}$$

From this, we can conclude that the covariance matrix of $y + z$ is simply $\Sigma + \Sigma'$.

At this point, take a step back and think about what we have just done. Using some simple properties of expectations and independence, we have computed the mean and covariance matrix of $y + z$. Because of Fact #1, we can thus write down the density for $y + z$ immediately, without the need to perform a convolution![3]

3.2 Marginal of a joint Gaussian is Gaussian

The formal statement of this rule is:

> Suppose that
>
> $$\begin{bmatrix} x_A \\ x_B \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}, \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix} \right),$$
>
> where $x_A \in \mathbf{R}^m$, $x_B \in \mathbf{R}^n$, and the dimensions of the mean vectors and covariance matrix subblocks are chosen to match x_A and x_B. Then, the marginal densities,
>
> $$p(x_A) = \int_{x_B \in \mathbf{R}^n} p(x_A, x_B; \mu, \Sigma) dx_B$$
> $$p(x_B) = \int_{x_A \in \mathbf{R}^m} p(x_A, x_B; \mu, \Sigma) dx_A$$
>
> are Gaussian:
>
> $$x_A \sim \mathcal{N}(\mu_A, \Sigma_{AA})$$
> $$x_B \sim \mathcal{N}(\mu_B, \Sigma_{BB}).$$

To justify this rule, let's just focus on the marginal distribution with respect to the variables x_A.[4]

First, note that computing the mean and covariance matrix for a marginal distribution is easy: simply take the corresponding subblocks from the mean and covariance matrix of the joint density. To make sure this is absolutely clear, let's look at the covariance between $x_{A,i}$ and $x_{A,j}$ (the ith component of x_A and the jth component of x_A). Note that $x_{A,i}$ and $x_{A,j}$ are also the ith and jth components of

$$\begin{bmatrix} x_A \\ x_B \end{bmatrix}$$

[3] Of course, we needed to know that $y + z$ had a Gaussian distribution in the first place.

[4] In general, for a random vector x which has a Gaussian distribution, we can always permute entries of x so long as we permute the entries of the mean vector and the rows/columns of the covariance matrix in the corresponding way. As a result, it suffices to look only at x_A, and the result for x_B follows immediately.

(since x_A appears at the top of this vector). To find their covariance, we need to simply look at the (i, j)th element of the covariance matrix,

$$\begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}.$$

The (i, j)th element is found in the Σ_{AA} subblock, and in fact, is precisely $\Sigma_{AA,ij}$. Using this argument for all $i, j \in \{1, \dots, m\}$, we see that the covariance matrix for x_A is simply Σ_{AA}. A similar argument can be used to find that the mean of x_A is simply μ_A. Thus, the above argument tells us that if we knew that the marginal distribution over x_A is Gaussian, then we could immediately write down a density function for x_A in terms of the appropriate submatrices of the mean and covariance matrices for the joint density!

The above argument, though simple, however, is somewhat unsatisfying: how can we actually be sure that x_A has a multivariate Gaussian distribution? The argument for this is slightly long-winded, so rather than saving up the punchline, here's our plan of attack up front:

1. Write the integral form of the marginal density explicitly.
2. Rewrite the integral by partitioning the inverse covariance matrix.
3. Use a "completion-of-squares" argument to evaluate the integral over x_B.
4. Argue that the resulting density is Gaussian.

Let's see each of these steps in action.

3.2.1 The marginal density in integral form

Suppose that we wanted to compute the density function of x_A directly. Then, we would need to compute the integral,

$$\begin{aligned} p(x_A) &= \int_{x_B \in \mathbf{R}^n} p(x_A, x_B; \mu, \Sigma) dx_B \\ &= \frac{1}{(2\pi)^{\frac{m+n}{2}} \begin{vmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{vmatrix}^{1/2}} \int_{x_B \in \mathbf{R}^n} \exp\left(-\frac{1}{2} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix}^T \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}^{-1} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix} \right) dx_B. \end{aligned}$$

3.2.2 Partitioning the inverse covariance matrix

To make any sort of progress, we'll need to write the matrix product in the exponent in a slightly different form. In particular, let us define the matrix $V \in \mathbf{R}^{(m+n)\times(m+n)}$ as[5]

$$V = \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} = \Sigma^{-1}.$$

[5]Sometimes, V is called the "precision" matrix.

It might be tempting to think that

$$V = \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} = \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}^{-1} \text{“}{=}\text{”} \begin{bmatrix} \Sigma_{AA}^{-1} & \Sigma_{AB}^{-1} \\ \Sigma_{BA}^{-1} & \Sigma_{BB}^{-1} \end{bmatrix}$$

However, the rightmost equality does not hold! We'll return to this issue in a later step; for now, though, it suffices to define V as above without worrying what actual contents of each submatrix are.

Using this definition of V, the integral expands to

$$p(x_A) = \frac{1}{Z} \int_{x_B \in \mathbf{R}^n} \exp\Big(- \Big[\frac{1}{2}(x_A - \mu_A)^T V_{AA} (x_A - \mu_A) + \frac{1}{2}(x_A - \mu_A)^T V_{AB} (x_B - \mu_B) \\ + \frac{1}{2}(x_B - \mu_B)^T V_{BA} (x_A - \mu_A) + \frac{1}{2}(x_B - \mu_B)^T V_{BB} (x_B - \mu_B) \Big] \Big) dx_B,$$

where Z is some constant not depending on either x_A or x_B that we'll choose to ignore for the moment. If you haven't worked with partitioned matrices before, then the expansion above may seem a little magical to you. It is analogous to the idea that when defining a quadratic form based on some 2×2 matrix A, then

$$x^T A x = \sum_i \sum_j A_{ij} x_i x_j = x_1 A_{11} x_1 + x_1 A_{12} x_2 + x_2 A_{21} x_1 + x_2 A_{22} x_2.$$

Take some time to convince yourself that the matrix generalization above also holds.

3.2.3 Integrating out x_B

To evaluate the integral, we'll somehow want to integrate out x_B. In general, however, Gaussian integrals are hard to compute by hand. Is there anything we can do to save time? There are, in fact, a number of Gaussian integrals for which the answer is already known (see Fact #2). The basic idea in this section, then, will be to transform the integral we had in the last section into a form where we can apply one of the results from Fact #2 in order to perform the required integration easily.

The key to this is a mathematical trick known as "completion of squares." Consider the quadratic function $z^T A z + b^T z + c$ where A is a symmetric, nonsingular matrix. Then, one can verify directly that

$$\frac{1}{2} z^T A z + b^T z + c = \frac{1}{2}\big(z + A^{-1}b\big)^T A \big(z + A^{-1}b\big) + c - \frac{1}{2} b^T A^{-1} b.$$

This is the multivariate generalization of the "completion of squares" argument used in single variable algebra:

$$\frac{1}{2} a z^2 + bz + c = \frac{1}{2} a \left(z + \frac{b}{a}\right)^2 + c - \frac{b^2}{2a}$$

To apply the completion of squares in our situation above, let

$$
\begin{aligned}
z &= x_B - \mu_B \\
A &= V_{BB} \\
b &= V_{BA}(x_A - \mu_A) \\
c &= \frac{1}{2}(x_A - \mu_A)^T V_{AA}(x_A - \mu_A).
\end{aligned}
$$

Then, it follows that the integral can be rewritten as

$$
\begin{aligned}
p(x_A) = \frac{1}{Z}\int_{x_B\in\mathbf{R}^n} \exp\bigg(-\bigg[&\frac{1}{2}\big(x_B - \mu_B + V_{BB}^{-1}V_{BA}(x_A - \mu_A)\big)^T V_{BB}\big(x_B - \mu_B + V_{BB}^{-1}V_{BA}(x_A - \mu_A)\big) \\
&+ \frac{1}{2}(x_A - \mu_A)^T V_{AA}(x_A - \mu_A) - \frac{1}{2}(x_A - \mu_A)^T V_{AB}V_{BB}^{-1}V_{BA}(x_A - \mu_A)\bigg]\bigg) dx_B
\end{aligned}
$$

We can factor out the terms not including x_B to obtain,

$$
\begin{aligned}
p(x_A) = &\exp\left(-\frac{1}{2}(x_A - \mu_A)^T V_{AA}(x_A - \mu_A) + \frac{1}{2}(x_A - \mu_A)^T V_{AB}V_{BB}^{-1}V_{BA}(x_A - \mu_A)\right) \\
&\cdot \frac{1}{Z}\int_{x_B\in\mathbf{R}^n} \exp\bigg(-\frac{1}{2}\bigg[\big(x_B - \mu_B + V_{BB}^{-1}V_{BA}(x_A - \mu_A)\big)^T V_{BB}\big(x_B - \mu_B + V_{BB}^{-1}V_{BA}(x_A - \mu_A)\big)\bigg]\bigg) dx_B
\end{aligned}
$$

At this point, we can now apply Fact #2. In particular, we know that generically speaking, for a multivariate Gaussian distributed random variable x with mean μ and covariance matrix Σ, the density function normalizes, i.e.,

$$
\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}\int_{\mathbf{R}^n} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) = 1,
$$

or equivalently,

$$
\int_{\mathbf{R}^n} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) = (2\pi)^{n/2}|\Sigma|^{1/2}.
$$

We use this fact to get rid of the remaining integral in our expression for $p(x_A)$:

$$
p(x_A) = \frac{1}{Z}\cdot(2\pi)^{n/2}|V_{BB}|^{1/2}\cdot\exp\left(-\frac{1}{2}(x_A - \mu_A)^T(V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})(x_A - \mu_A)\right).
$$

3.2.4 Arguing that resulting density is Gaussian

At this point, we are almost done! Ignoring the normalization constant in front, we see that the density of x_A is the exponential of a quadratic form in x_A. We can quickly recognize that our density is none other than a Gaussian with mean vector μ_A and covariance matrix $(V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})^{-1}$. Although the form of the covariance matrix may seem a bit complex, we have already achieved what we set out to show in the first place—namely, that x_A has a marginal Gaussian distribution. Using the logic before, we can conclude that this covariance matrix must somehow reduce to Σ_{AA}.

But, in case you are curious, it's also possible to show that our derivation is consistent with this earlier justification. To do this, we use the following result for partitioned matrices:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} M^{-1} & -M^{-1}BD^{-1} \\ -D^{-1}CM^{-1} & D^{-1} + D^{-1}CM^{-1}BD^{-1} \end{bmatrix}.$$

where $M = A - BD^{-1}C$. This formula can be thought of as the multivariable generalization of the explicit inverse for a 2×2 matrix,

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Using the formula, it follows that

$$\begin{aligned} \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix} &= \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}^{-1} \\ &= \begin{bmatrix} (V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})^{-1} & -(V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})^{-1}V_{AB}V_{BB}^{-1} \\ -V_{BB}^{-1}V_{BA}(V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})^{-1} & (V_{BB} - V_{BA}V_{AA}^{-1}V_{AB})^{-1} \end{bmatrix} \end{aligned}$$

We immediately see that $(V_{AA} - V_{AB}V_{BB}^{-1}V_{BA})^{-1} = \Sigma_{AA}$, just as we expected!

3.3 Conditional of a joint Gaussian is Gaussian

The formal statement of this rule is:

> Suppose that
>
> $$\begin{bmatrix} x_A \\ x_B \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}, \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix} \right),$$
>
> where $x_A \in \mathbf{R}^m$, $x_B \in \mathbf{R}^n$, and the dimensions of the mean vectors and covariance matrix subblocks are chosen to match x_A and x_B. Then, the conditional densities
>
> $$\begin{aligned} p(x_A \mid x_B) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_A \in \mathbf{R}^m} p(x_A, x_B; \mu, \Sigma) dx_A} \\ p(x_B \mid x_A) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_B \in \mathbf{R}^n} p(x_A, x_B; \mu, \Sigma) dx_B} \end{aligned}$$
>
> are also Gaussian:
>
> $$\begin{aligned} x_A \mid x_B &\sim \mathcal{N}\big(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}\big) \\ x_B \mid x_A &\sim \mathcal{N}\big(\mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(x_A - \mu_A), \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}\big). \end{aligned}$$

As before, we'll just examine the conditional distribution $x_B \mid x_A$, and the other result will hold by symmetry. Our plan of attack will be as follows:

1. Write the form of the conditional density explicitly.
2. Rewrite the expression by partitioning the inverse covariance matrix.
3. Use a "completion-of-squares" argument.
4. Argue that the resulting density is Gaussian.

Let's see each of these steps in action.

3.3.1 The conditional density written explicitly

Suppose that we wanted to compute the density function of x_B given x_A directly. Then, we would need to compute

$$\begin{aligned} p(x_B \mid x_A) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_B \in \mathbf{R}^m} p(x_A, x_B; \mu, \Sigma) dx_A} \\ &= \frac{1}{Z'} \exp\left(-\frac{1}{2} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix}^T \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}^{-1} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix} \right) \end{aligned}$$

where Z' is a normalization constant that we used to absorb factors not depending on x_B. Note that this time, we don't even need to compute any integrals – the value of the integral does not depend on x_B, and hence the integral can be folded into the normalization constant Z'.

3.3.2 Partitioning the inverse covariance matrix

As before, we reparameterize our density using the matrix V, to obtain

$$\begin{aligned} p(x_B \mid x_A) &= \frac{1}{Z'} \exp\left(-\frac{1}{2} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix}^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix} \right) \\ &= \frac{1}{Z'} \exp\Big(-\Big[\frac{1}{2}(x_A - \mu_A)^T V_{AA} (x_A - \mu_A) + \frac{1}{2}(x_A - \mu_A)^T V_{AB} (x_B - \mu_B) \\ &\qquad + \frac{1}{2}(x_B - \mu_B)^T V_{BA} (x_A - \mu_A) + \frac{1}{2}(x_B - \mu_B)^T V_{BB} (x_B - \mu_B) \Big] \Big). \end{aligned}$$

3.3.3 Use a "completion of squares" argument

Recall that

$$\frac{1}{2} z^T A z + b^T z + c = \frac{1}{2}\big(z + A^{-1}b\big)^T A \big(z + A^{-1}b\big) + c - \frac{1}{2} b^T A^{-1} b$$

provided A is a symmetric, nonsingular matrix. As before, to apply the completion of squares in our situation above, let

$$\begin{aligned} z &= x_B - \mu_B \\ A &= V_{BB} \\ b &= V_{BA}(x_A - \mu_A) \\ c &= \frac{1}{2}(x_A - \mu_A)^T V_{AA}(x_A - \mu_A). \end{aligned}$$

Then, it follows that the expression for $p(x_B \mid x_A)$ can be rewritten as

$$p(x_B \mid x_A) = \frac{1}{Z'} \exp\Bigg(- \bigg[\frac{1}{2}\big(x_B - \mu_B + V_{BB}^{-1} V_{BA}(x_A - \mu_A)\big)^T V_{BB} \big(x_B - \mu_B + V_{BB}^{-1} V_{BA}(x_A - \mu_A)\big) \\ + \frac{1}{2}(x_A - \mu_A)^T V_{AA}(x_A - \mu_A) - \frac{1}{2}(x_A - \mu_A)^T V_{AB} V_{BB}^{-1} V_{BA}(x_A - \mu_A) \bigg] \Bigg)$$

Absorbing the portion of the exponent which does not depend on x_B into the normalization constant, we have

$$p(x_B \mid x_A) = \frac{1}{Z''} \exp\Bigg(-\frac{1}{2}\big(x_B - \mu_B + V_{BB}^{-1} V_{BA}(x_A - \mu_A)\big)^T V_{BB} \big(x_B - \mu_B + V_{BB}^{-1} V_{BA}(x_A - \mu_A)\big) \Bigg)$$

3.3.4 Arguing that resulting density is Gaussian

Looking at the last form, $p(x_B \mid x_A)$ has the form of a Gaussian density with mean $\mu_B - V_{BB}^{-1} V_{BA}(x_A - \mu_A)$ and covariance matrix V_{BB}^{-1}. As before, recall our matrix identity,

$$\begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix} = \begin{bmatrix} (V_{AA} - V_{AB} V_{BB}^{-1} V_{BA})^{-1} & -(V_{AA} - V_{AB} V_{BB}^{-1} V_{BA})^{-1} V_{AB} V_{BB}^{-1} \\ -V_{BB}^{-1} V_{BA}(V_{AA} - V_{AB} V_{BB}^{-1} V_{BA})^{-1} & (V_{BB} - V_{BA} V_{AA}^{-1} V_{AB})^{-1} \end{bmatrix}.$$

From this, it follows that

$$\mu_{B|A} = \mu_B - V_{BB}^{-1} V_{BA}(x_A - \mu_A) = \mu_B + \Sigma_{BA} \Sigma_{AA}^{-1}(x_A - \mu_A).$$

Conversely, we can also apply our matrix identity to obtain:

$$\begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} = \begin{bmatrix} (\Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA})^{-1} & -(\Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA})^{-1} \Sigma_{AB} \Sigma_{BB}^{-1} \\ -\Sigma_{BB}^{-1} \Sigma_{BA}(\Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA})^{-1} & (\Sigma_{BB} - \Sigma_{BA} \Sigma_{AA}^{-1} \Sigma_{AB})^{-1} \end{bmatrix},$$

from which it follows that

$$\Sigma_{B|A} = V_{BB}^{-1} = \Sigma_{BB} - \Sigma_{BA} \Sigma_{AA}^{-1} \Sigma_{AB}.$$

And, we're done!

4 Summary

In these notes, we used a few simple properties of multivariate Gaussians (plus a couple matrix algebra tricks) in order to argue that multivariate Gaussians satisfy a number of closure properties. In general, multivariate Gaussians are exceedingly useful representations of probability distributions because the closure properties ensure that most of the types of operations we would ever want to perform using a multivariate Gaussian can be done in closed form. Analytically, integrals involving multivariate Gaussians are often nice in practice since we can rely on known Gaussian integrals to avoid having to ever perform the integration ourselves.

5 Exercise

Test your understanding! Let $A \in \mathbf{R}^{n \times n}$ be a symmetric nonsingular square matrix, $b \in \mathbf{R}^n$, and c. Prove that

$$\int_{x \in \mathbf{R}^n} \exp\left(-\frac{1}{2}x^T A x - x^T b - c\right) dx = \frac{(2\pi)^{n/2}}{|A|^{1/2} \exp(c - b^T A^{-1} b)}.$$

References

For more information on multivariate Gaussians, see

Bishop, Christopher M. *Pattern Recognition and Machine Learning.* Springer, 2006.

Gaussian processes

Chuong B. Do (updated by Honglak Lee)

November 22, 2008

Many of the classical machine learning algorithms that we talked about during the first half of this course fit the following pattern: given a training set of i.i.d. examples sampled from some unknown distribution,

1. solve a convex optimization problem in order to identify the single "best fit" model for the data, and

2. use this estimated model to make "best guess" predictions for future test input points.

In these notes, we will talk about a different flavor of learning algorithms, known as **Bayesian methods**. Unlike classical learning algorithm, Bayesian algorithms do not attempt to identify "best-fit" models of the data (or similarly, make "best guess" predictions for new test inputs). Instead, they compute a posterior distribution over models (or similarly, compute posterior predictive distributions for new test inputs). These distributions provide a useful way to quantify our uncertainty in model estimates, and to exploit our knowledge of this uncertainty in order to make more robust predictions on new test points.

We focus on **regression** problems, where the goal is to learn a mapping from some input space $\mathcal{X} = \mathbf{R}^n$ of n-dimensional vectors to an output space $\mathcal{Y} = \mathbf{R}$ of real-valued targets. In particular, we will talk about a kernel-based fully Bayesian regression algorithm, known as Gaussian process regression. The material covered in these notes draws heavily on many different topics that we discussed previously in class (namely, the probabilistic interpretation of linear regression[1], Bayesian methods[2], kernels[3], and properties of multivariate Gaussians[4]).

The organization of these notes is as follows. In Section 1, we provide a brief review of multivariate Gaussian distributions and their properties. In Section 2, we briefly review Bayesian methods in the context of probabilistic linear regression. The central ideas underlying Gaussian processes are presented in Section 3, and we derive the full Gaussian process regression model in Section 4.

[1]See course lecture notes on "Supervised Learning, Discriminative Algorithms."
[2]See course lecture notes on "Regularization and Model Selection."
[3]See course lecture notes on "Support Vector Machines."
[4]See course lecture notes on "Factor Analysis."

1 Multivariate Gaussians

A vector-valued random variable $x \in \mathbf{R}^n$ is said to have a **multivariate normal (or Gaussian) distribution** with mean $\mu \in \mathbf{R}^n$ and covariance matrix $\Sigma \in \mathbf{S}^n_{++}$ if

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right). \tag{1}$$

We write this as $x \sim \mathcal{N}(\mu, \Sigma)$. Here, recall from the section notes on linear algebra that $\mathbf{S}^n_{++}$ refers to the space of symmetric positive definite $n \times n$ matrices.[5]

Generally speaking, Gaussian random variables are extremely useful in machine learning and statistics for two main reasons. First, they are extremely common when modeling "noise" in statistical algorithms. Quite often, noise can be considered to be the accumulation of a large number of small independent random perturbations affecting the measurement process; by the Central Limit Theorem, summations of independent random variables will tend to "look Gaussian." Second, Gaussian random variables are convenient for many analytical manipulations, because many of the integrals involving Gaussian distributions that arise in practice have simple closed form solutions. In the remainder of this section, we will review a number of useful properties of multivariate Gaussians.

Consider a random vector $x \in \mathbf{R}^n$ with $x \sim \mathcal{N}(\mu, \Sigma)$. Suppose also that the variables in x have been partitioned into two sets $x_A = [x_1 \cdots x_r]^T \in \mathbf{R}^r$ and $x_B = [x_{r+1} \cdots x_n]^T \in \mathbf{R}^{n-r}$ (and similarly for μ and Σ), such that

$$x = \begin{bmatrix} x_A \\ x_B \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}.$$

Here, $\Sigma_{AB} = \Sigma_{BA}^T$ since $\Sigma = E[(x-\mu)(x-\mu)^T] = \Sigma^T$. The following properties hold:

1. **Normalization.** The density function normalizes, i.e.,

$$\int_x p(x; \mu, \Sigma) dx = 1.$$

This property, though seemingly trivial at first glance, turns out to be immensely useful for evaluating all sorts of integrals, even ones which appear to have no relation to probability distributions at all (see Appendix A.1)!

2. **Marginalization.** The marginal densities,

$$p(x_A) = \int_{x_B} p(x_A, x_B; \mu, \Sigma) dx_B$$
$$p(x_B) = \int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A$$

[5] There are actually cases in which we would want to deal with multivariate Gaussian distributions where Σ is positive semidefinite but not positive definite (i.e., Σ is not full rank). In such cases, Σ^{-1} does not exist, so the definition of the Gaussian density given in (1) does not apply. For instance, see the course lecture notes on "Factor Analysis."

are Gaussian:

$$\begin{aligned} x_A &\sim \mathcal{N}(\mu_A, \Sigma_{AA}) \\ x_B &\sim \mathcal{N}(\mu_B, \Sigma_{BB}). \end{aligned}$$

3. **Conditioning.** The conditional densities

$$\begin{aligned} p(x_A \mid x_B) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A} \\ p(x_B \mid x_A) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_B} p(x_A, x_B; \mu, \Sigma) dx_B} \end{aligned}$$

   are also Gaussian:

$$\begin{aligned} x_A \mid x_B &\sim \mathcal{N}\big(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}\big) \\ x_B \mid x_A &\sim \mathcal{N}\big(\mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(x_A - \mu_A), \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}\big). \end{aligned}$$

   A proof of this property is given in Appendix A.2. (See also Appendix A.3 for an easier version of the derivation.)

4. **Summation.** The sum of independent Gaussian random variables (with the same dimensionality), $y \sim \mathcal{N}(\mu, \Sigma)$ and $z \sim \mathcal{N}(\mu', \Sigma')$, is also Gaussian:

$$y + z \sim \mathcal{N}(\mu + \mu', \Sigma + \Sigma').$$

2 Bayesian linear regression

Let $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ be a training set of i.i.d. examples from some unknown distribution. The standard probabilistic interpretation of linear regression states that

$$y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}, \qquad i = 1, \ldots, m$$

where the $\varepsilon^{(i)}$ are i.i.d. "noise" variables with independent $\mathcal{N}(0, \sigma^2)$ distributions. It follows that $y^{(i)} - \theta^T x^{(i)} \sim \mathcal{N}(0, \sigma^2)$, or equivalently,

$$P(y^{(i)} \mid x^{(i)}, \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right).$$

For notational convenience, we define

$$X = \begin{bmatrix} — & (x^{(1)})^T & — \\ — & (x^{(2)})^T & — \\ & \vdots & \\ — & (x^{(m)})^T & — \end{bmatrix} \in \mathbf{R}^{m \times n} \qquad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbf{R}^m \qquad \vec{\varepsilon} = \begin{bmatrix} \varepsilon^{(1)} \\ \varepsilon^{(2)} \\ \vdots \\ \varepsilon^{(m)} \end{bmatrix} \in \mathbf{R}^m.$$

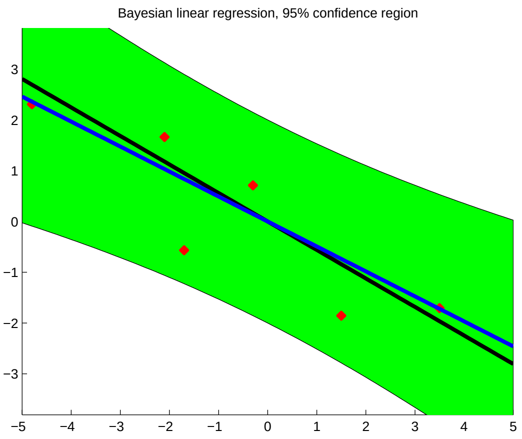


Figure 1: Bayesian linear regression for a one-dimensional linear regression problem, $y^{(i)} = \theta x^{(i)} + \epsilon^{(i)}$, with $\epsilon^{(i)} \sim \mathcal{N}(0,1)$ i.i.d. noise. The green region denotes the 95% confidence region for predictions of the model. Note that the (vertical) width of the green region is largest at the ends but narrowest in the middle. This region reflects the uncertain in the estimates for the parameter θ. In contrast, a classical linear regression model would display a confidence region of constant width, reflecting only the $\mathcal{N}(0,\sigma^2)$ noise in the outputs.

In Bayesian linear regression, we assume that a **prior distribution** over parameters is also given; a typical choice, for instance, is $\theta \sim \mathcal{N}(0, \tau^2 I)$. Using Bayes's rule, we obtain the **parameter posterior**,

$$p(\theta \mid S) = \frac{p(\theta)p(S \mid \theta)}{\int_{\theta'} p(\theta')p(S \mid \theta')d\theta'} = \frac{p(\theta)\prod_{i=1}^m p(y^{(i)} \mid x^{(i)}, \theta)}{\int_{\theta'} p(\theta')\prod_{i=1}^m p(y^{(i)} \mid x^{(i)}, \theta')d\theta'}. \tag{2}$$

Assuming the same noise model on testing points as on our training points, the "output" of Bayesian linear regression on a new test point x_* is not just a single guess "y_*", but rather an entire probability distribution over possible outputs, known as the **posterior predictive distribution**:

$$p(y_* \mid x_*, S) = \int_\theta p(y_* \mid x_*, \theta)p(\theta \mid S)d\theta. \tag{3}$$

For many types of models, the integrals in (2) and (3) are difficult to compute, and hence, we often resort to approximations, such as MAP estimation (see course lecture notes on "Regularization and Model Selection").

In the case of Bayesian linear regression, however, the integrals actually are tractable! In particular, for Bayesian linear regression, one can show (after much work!) that

$$\theta \mid S \sim \mathcal{N}\left(\frac{1}{\sigma^2}A^{-1}X^T\vec{y}, A^{-1}\right)$$
$$y_* \mid x_*, S \sim \mathcal{N}\left(\frac{1}{\sigma^2}x_*^T A^{-1}X^T\vec{y}, x_*^T A^{-1}x_* + \sigma^2\right)$$

where $A = \frac{1}{\sigma^2}X^TX + \frac{1}{\tau^2}I$. The derivation of these formulas is somewhat involved.[6] Nonetheless, from these equations, we get at least a flavor of what Bayesian methods are all about: the posterior distribution over the test output y_* for a test input x_* is a Gaussian distribution—this distribution reflects the uncertainty in our predictions $y_* = \theta^T x_* + \varepsilon_*$ arising from both the randomness in ε_* and the uncertainty in our choice of parameters θ. In contrast, classical probabilistic linear regression models estimate parameters θ directly from the training data but provide no estimate of how reliable these learned parameters may be (see Figure 1).

3 Gaussian processes

As described in Section 1, multivariate Gaussian distributions are useful for modeling finite collections of real-valued variables because of their nice analytical properties. **Gaussian processes** are the extension of multivariate Gaussians to infinite-sized collections of real-valued variables. In particular, this extension will allow us to think of Gaussian processes as distributions not just over random vectors but in fact distributions over **random functions**.[7]

3.1 Probability distributions over functions with finite domains

To understand how one might paramterize probability distributions over functions, consider the following simple example. Let $\mathcal{X} = \{x_1, \ldots, x_m\}$ be any finite set of elements. Now, consider the set $\mathcal{H}$ of all possible functions mapping from $\mathcal{X}$ to $\mathbf{R}$. For instance, one example of a function $f_0(\cdot) \in \mathcal{H}$ is given by

$$f_0(x_1) = 5, \quad f_0(x_2) = 2.3, \quad f_0(x_2) = -7, \quad \ldots, \quad f_0(x_{m-1}) = -\pi, \quad f_0(x_m) = 8.$$

Since the domain of any $f(\cdot) \in \mathcal{H}$ has only m elements, we can always represent $f(\cdot)$ compactly as an m-dimensional vector, $\vec{f} = \begin{bmatrix} f(x_1) & f(x_2) & \cdots & f(x_m) \end{bmatrix}^T$. In order to specify a probability distribution over functions $f(\cdot) \in \mathcal{H}$, we must associate some "probability density" with each function in $\mathcal{H}$. One natural way to do this is to exploit the one-to-one correspondence between functions $f(\cdot) \in \mathcal{H}$ and their vector representations, $\vec{f}$. In particular, if we specify that $\vec{f} \sim \mathcal{N}(\vec{\mu}, \sigma^2 I)$, then this in turn implies a probability distribution over functions $f(\cdot)$, whose probability density function is given by

$$p(h) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(f(x_i) - \mu_i)^2\right).$$

[6]For the complete derivation, see, for instance, [1]. Alternatively, read the Appendices, which gives a number of arguments based on the "completion-of-squares" trick, and derive this formula yourself!

[7]Let $\mathcal{H}$ be a class of functions mapping from $\mathcal{X} \to \mathcal{Y}$. A random function $f(\cdot)$ from $\mathcal{H}$ is a function which is randomly drawn from $\mathcal{H}$, according to some probability distribution over $\mathcal{H}$. One potential source of confusion is that you may be tempted to think of random functions as functions whose outputs are in some way stochastic; this is not the case. Instead, a random function $f(\cdot)$, once selected from $\mathcal{H}$ probabilistically, implies a deterministic mapping from inputs in $\mathcal{X}$ to outputs in $\mathcal{Y}$.

In the example above, we showed that probability distributions over functions with finite domains can be represented using a finite-dimensional multivariate Gaussian distribution over function outputs $f(x_1), \ldots, f(x_m)$ at a finite number of input points $x_1, \ldots, x_m$. How can we specify probability distributions over functions when the domain size may be infinite? For this, we turn to a fancier type of probability distribution known as a Gaussian process.

3.2 Probability distributions over functions with infinite domains

A stochastic process is a collection of random variables, $\{f(x) : x \in \mathcal{X}\}$, indexed by elements from some set $\mathcal{X}$, known as the index set.[8] A **Gaussian process** is a stochastic process such that any finite subcollection of random variables has a multivariate Gaussian distribution.

In particular, a collection of random variables $\{f(x) : x \in \mathcal{X}\}$ is said to be drawn from a Gaussian process with **mean function** $m(\cdot)$ and **covariance function** $k(\cdot, \cdot)$ if for any finite set of elements $x_1, \ldots, x_m \in \mathcal{X}$, the associated finite set of random variables $f(x_1), \ldots, f(x_m)$ have distribution,

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_m) \end{bmatrix}, \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix} \right).$$

We denote this using the notation,

$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)).$$

Observe that the mean function and covariance function are aptly named since the above properties imply that

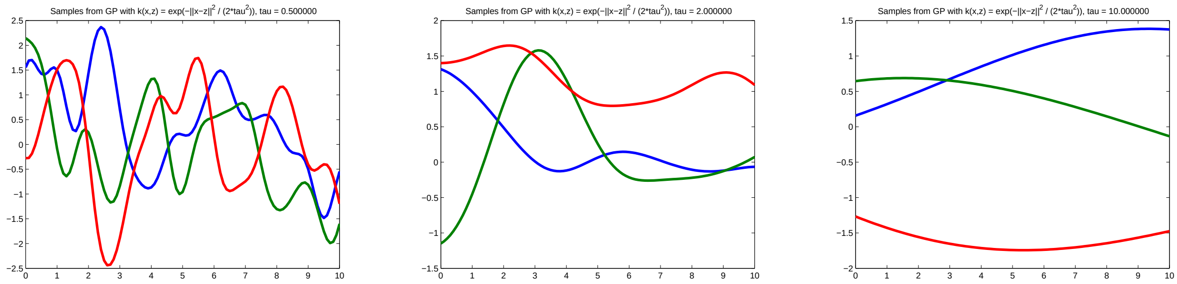
$$\begin{aligned} m(x) &= E[x] \\ k(x, x') &= E[(x - m(x))(x' - m(x')). \end{aligned}$$

for any $x, x' \in \mathcal{X}$.

Intuitively, one can think of a function $f(\cdot)$ drawn from a Gaussian process prior as an extremely high-dimensional vector drawn from an extremely high-dimensional multivariate Gaussian. Here, each dimension of the Gaussian corresponds to an element x from the index set $\mathcal{X}$, and the corresponding component of the random vector represents the value of $f(x)$. Using the marginalization property for multivariate Gaussians, we can obtain the marginal multivariate Gaussian density corresponding to any finite subcollection of variables.

What sort of functions $m(\cdot)$ and $k(\cdot, \cdot)$ give rise to valid Gaussian processes? In general, any real-valued function $m(\cdot)$ is acceptable, but for $k(\cdot, \cdot)$, it must be the case that for any

[8] Often, when $\mathcal{X} = \mathbf{R}$, one can interpret the indices $x \in \mathcal{X}$ as representing times, and hence the variables $f(x)$ represent the temporal evolution of some random quantity over time. In the models that are used for Gaussian process regression, however, the index set is taken to be the input space of our regression problem.



(a) (b) (c)

Figure 2: Samples from a zero-mean Gaussian process prior with $k_{SE}(\cdot, \cdot)$ covariance function, using (a) $\tau = 0.5$, (b) $\tau = 2$, and (c) $\tau = 10$. Note that as the bandwidth parameter τ increases, then points which are farther away will have higher correlations than before, and hence the sampled functions tend to be smoother overall.

set of elements $x_1, \ldots, x_m \in \mathcal{X}$, the resulting matrix

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix}$$

is a valid covariance matrix corresponding to some multivariate Gaussian distribution. A standard result in probability theory states that this is true provided that K is positive semidefinite. Sound familiar?

The positive semidefiniteness requirement for covariance matrices computed based on arbitrary input points is, in fact, identical to Mercer's condition for kernels! A function $k(\cdot, \cdot)$ is a valid kernel provided the resulting kernel matrix K defined as above is always positive semidefinite for any set of input points $x_1, \ldots, x_m \in \mathcal{X}$. Gaussian processes, therefore, are kernel-based probability distributions in the sense that any valid kernel function can be used as a covariance function!

3.3 The squared exponential kernel

In order to get an intuition for how Gaussian processes work, consider a simple zero-mean Gaussian process,

$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot)).$$

defined for functions $h : \mathcal{X} \to \mathbf{R}$ where we take $\mathcal{X} = \mathbf{R}$. Here, we choose the kernel function $k(\cdot, \cdot)$ to be the **squared exponential**[9] kernel function, defined as

$$k_{SE}(x, x') = \exp\left(-\frac{1}{2\tau^2}||x - x'||^2\right)$$

[9]In the context of SVMs, we called this the Gaussian kernel; to avoid confusion with "Gaussian" processes, we refer to this kernel here as the squared exponential kernel, even though the two are formally identical.

for some $\tau > 0$. What do random functions sampled from this Gaussian process look like?

In our example, since we use a zero-mean Gaussian process, we would expect that for the function values from our Gaussian process will tend to be distributed around zero. Furthermore, for any pair of elements $x, x' \in \mathcal{X}$.

- $f(x)$ and $f(x')$ will tend to have high covariance x and x' are "nearby" in the input space (i.e., $||x - x'|| = |x - x'| \approx 0$, so $\exp(-\frac{1}{2\tau^2}||x - x'||^2) \approx 1$).
- $f(x)$ and $f(x')$ will tend to have low covariance when x and x' are "far apart" (i.e., $||x - x'|| \gg 0$, so $\exp(-\frac{1}{2\tau^2}||x - x'||^2) \approx 0$).

More simply stated, functions drawn from a zero-mean Gaussian process prior with the squared exponential kernel will tend to be "locally smooth" with high probability; i.e., nearby function values are highly correlated, and the correlation drops off as a function of distance in the input space (see Figure 2).

4 Gaussian process regression

As discussed in the last section, Gaussian processes provide a method for modelling probability distributions over functions. Here, we discuss how probability distributions over functions can be used in the framework of Bayesian regression.

4.1 The Gaussian process regression model

Let $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ be a training set of i.i.d. examples from some unknown distribution. In the Gaussian process regression model,

$$y^{(i)} = f(x^{(i)}) + \varepsilon^{(i)}, \qquad i = 1, \dots, m$$

where the $\varepsilon^{(i)}$ are i.i.d. "noise" variables with independent $\mathcal{N}(0, \sigma^2)$ distributions. Like in Bayesian linear regression, we also assume a **prior distribution** over functions $f(\cdot)$; in particular, we assume a zero-mean Gaussian process prior,

$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$$

for some valid covariance function $k(\cdot, \cdot)$.

Now, let $T = \{(x_*^{(i)}, y_*^{(i)})\}_{i=1}^{m_*}$ be a set of i.i.d. testing points drawn from the same unknown distribution as S.[10] For notational convenience, we define

$$X = \begin{bmatrix} — & (x^{(1)})^T & — \\ — & (x^{(2)})^T & — \\ & \vdots & \\ — & (x^{(m)})^T & — \end{bmatrix} \in \mathbf{R}^{m\times n} \qquad \vec{f} = \begin{bmatrix} f(x^{(1)}) \\ f(x^{(2)}) \\ \vdots \\ f(x^{(m)}) \end{bmatrix}, \quad \vec{\varepsilon} = \begin{bmatrix} \varepsilon^{(1)} \\ \varepsilon^{(2)} \\ \vdots \\ \varepsilon^{(m)} \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbf{R}^m,$$

$$X_* = \begin{bmatrix} — & (x_*^{(1)})^T & — \\ — & (x_*^{(2)})^T & — \\ & \vdots & \\ — & (x_*^{(m_*)})^T & — \end{bmatrix} \in \mathbf{R}^{m_*\times n} \quad \vec{f}_* = \begin{bmatrix} f(x_*^{(1)}) \\ f(x_*^{(2)}) \\ \vdots \\ f(x_*^{(m_*)}) \end{bmatrix}, \quad \vec{\varepsilon}_* = \begin{bmatrix} \varepsilon_*^{(1)} \\ \varepsilon_*^{(2)} \\ \vdots \\ \varepsilon_*^{(m_*)} \end{bmatrix}, \quad \vec{y}_* = \begin{bmatrix} y_*^{(1)} \\ y_*^{(2)} \\ \vdots \\ y_*^{(m_*)} \end{bmatrix} \in \mathbf{R}^{m_*}.$$

Given the training data S, the prior $p(h)$, and the testing inputs X_*, how can we compute the posterior predictive distribution over the testing outputs $\vec{y}_*$? For Bayesian linear regression in Section 2, we used Bayes's rule in order to compute the paramter posterior, which we then used to compute posterior predictive distribution $p(y_* \mid x_*, S)$ for a new test point x_*. For Gaussian process regression, however, it turns out that an even simpler solution exists!

4.2 Prediction

Recall that for any function $f(\cdot)$ drawn from our zero-mean Gaussian process prior with covariance function $k(\cdot,\cdot)$, the marginal distribution over any set of input points belonging to $\mathcal{X}$ must have a joint multivariate Gaussian distribution. In particular, this must hold for the training and test points, so we have

$$\begin{bmatrix} \vec{f} \\ \vec{f}_* \end{bmatrix} \Bigg| X, X_* \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right),$$

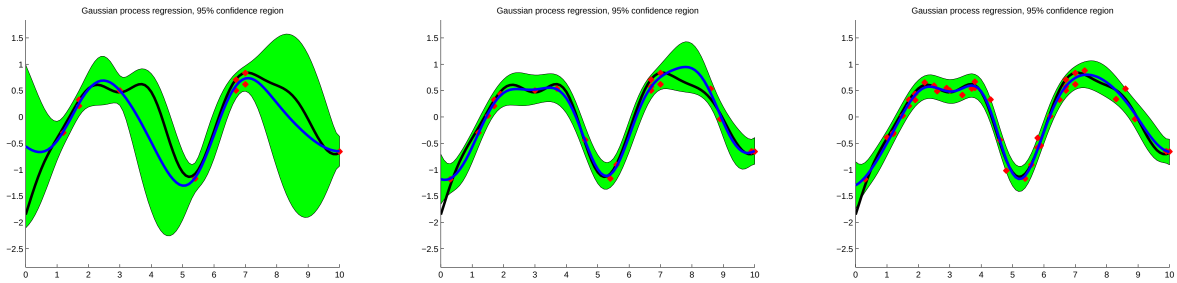
where

$$\begin{aligned} \vec{f} \in \mathbf{R}^m & \text{ such that } \vec{f} = \begin{bmatrix} f(x^{(1)}) & \cdots & f(x^{(m)}) \end{bmatrix}^T \\ \vec{f}_* \in \mathbf{R}^{m_*} & \text{ such that } \vec{f}_* = \begin{bmatrix} f(x_*^{(1)}) & \cdots & f(x_*^{(m)}) \end{bmatrix}^T \\ K(X,X) \in \mathbf{R}^{m\times m} & \text{ such that } (K(X,X))_{ij} = k(x^{(i)}, x^{(j)}) \\ K(X,X_*) \in \mathbf{R}^{m\times m_*} & \text{ such that } (K(X,X_*))_{ij} = k(x^{(i)}, x_*^{(j)}) \\ K(X_*,X) \in \mathbf{R}^{m_*\times m} & \text{ such that } (K(X_*,X))_{ij} = k(x_*^{(i)}, x^{(j)}) \\ K(X_*,X_*) \in \mathbf{R}^{m_*\times m_*} & \text{ such that } (K(X_*,X_*))_{ij} = k(x_*^{(i)}, x_*^{(j)}). \end{aligned}$$

From our i.i.d. noise assumption, we have that

$$\begin{bmatrix} \vec{\varepsilon} \\ \vec{\varepsilon}_* \end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} \sigma^2 I & \vec{0} \\ \vec{0}^T & \sigma^2 I \end{bmatrix}\right).$$

[10] We assume also that T are S are mutually independent.



(a) (b) (c)

Figure 3: Gaussian process regression using a zero-mean Gaussian process prior with $k_{SE}(\cdot,\cdot)$ covariance function (where $\tau = 0.1$), with noise level $\sigma = 1$, and (a) $m = 10$, (b) $m = 20$, and (c) $m = 40$ training examples. The blue line denotes the mean of the posterior predictive distribution, and the green shaded region denotes the 95% confidence region based on the model's variance estimates. As the number of training examples increases, the size of the confidence region shrinks to reflect the diminishing uncertainty in the model estimates. Note also that in panel (a), the 95% confidence region shrinks near training points but is much larger far away from training points, as one would expect.

The sums of independent Gaussian random variables is also Gaussian, so

$$\begin{bmatrix}\vec{y}\\ \vec{y}_*\end{bmatrix}\Bigg| X, X_* = \begin{bmatrix}\vec{f}\\ \vec{f}_*\end{bmatrix} + \begin{bmatrix}\vec{\varepsilon}\\ \vec{\varepsilon}_*\end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix}K(X,X)+\sigma^2 I & K(X,X_*)\\ K(X_*,X) & K(X_*,X_*)+\sigma^2 I\end{bmatrix}\right).$$

Now, using the rules for conditioning Gaussians, it follows that

$$\vec{y_*} \mid \vec{y}, X, X_* \sim \mathcal{N}(\mu^*, \Sigma^*)$$

where

$$\begin{aligned}\mu^* &= K(X_*,X)\left(K(X,X)+\sigma^2 I\right)^{-1}\vec{y}\\ \Sigma^* &= K(X_*,X_*)+\sigma^2 I - K(X_*,X)\left(K(X,X)+\sigma^2 I\right)^{-1}K(X,X_*).\end{aligned}$$

And that's it! Remarkably, performing prediction in a Gaussian process regression model is very simple, despite the fact that Gaussian processes in themselves are fairly complicated![11]

5 Summary

We close our discussion of our Gaussian processes by pointing out some reasons why Gaussian processes are an attractive model for use in regression problems and in some cases may be preferable to alternative models (such as linear and locally-weighted linear regression):

[11]Interestingly, it turns out that Bayesian linear regression, when "kernelized" in the proper way, turns out to be exactly equivalent to Gaussian process regression! But the derivation of the posterior predictive distribution is far more complicated for Bayesian linear regression, and the effort needed to kernelize the algorithm is even greater. The Gaussian process perspective is certainly much easier!

1. As Bayesian methods, Gaussian process models allow one to quantify uncertainty in predictions resulting not just from intrinsic noise in the problem but also the errors in the parameter estimation procedure. Furthermore, many methods for model selection and hyperparameter selection in Bayesian methods are immediately applicable to Gaussian processes (though we did not address any of these advanced topics here).

2. Like locally-weighted linear regression, Gaussian process regression is non-parametric and hence can model essentially arbitrary functions of the input points.

3. Gaussian process regression models provide a natural way to introduce kernels into a regression modeling framework. By careful choice of kernels, Gaussian process regression models can sometimes take advantage of structure in the data (though, we also did not examine this issue here).

4. Gaussian process regression models, though perhaps somewhat tricky to understand conceptually, nonetheless lead to simple and straightforward linear algebra implementations.

References

[1] Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, 2006. Online: `http://www.gaussianprocess.org/gpml/`

Appendix A.1

In this example, we show how the normalization property for multivariate Gaussians can be used to compute rather intimidating multidimensional integrals without performing any real calculus! Suppose you wanted to compute the following multidimensional integral,

$$I(A,b,c) = \int_x \exp\left(-\frac{1}{2}x^T Ax - x^T b - c\right)dx,$$

for some $A \in \mathbf{S}^m_{++}$, $b \in \mathbf{R}^m$, and $c \in \mathbf{R}$. Although one could conceivably perform the multidimensional integration directly (good luck!), a much simpler line of reasoning is based on a mathematical trick known as "completion-of-squares." In particular,

$$\begin{aligned}
I(A,b,c) &= \exp(-c) \cdot \int_x \exp\left(-\frac{1}{2}x^T Ax - x^T AA^{-1}b\right)dx \\
&= \exp(-c) \cdot \int_x \exp\left(-\frac{1}{2}(x - A^{-1}b)^T A(x - A^{-1}b) - b^T A^{-1}b\right)dx \\
&= \exp\left(-c - b^T A^{-1}b\right) \cdot \int_x \exp\left(-\frac{1}{2}(x - A^{-1}b)^T A(x - A^{-1}b)\right)dx.
\end{aligned}$$

Defining $\mu = A^{-1}b$ and $\Sigma = A^{-1}$, it follows that $I(A,b,c)$ is equal to

$$\frac{(2\pi)^{m/2}|\Sigma|^{1/2}}{\exp(c + b^T A^{-1}b)} \cdot \left[\frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \int_x \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx\right].$$

However, the term in brackets is identical in form to the integral of a multivariate Gaussian! Since we know that a Gaussian density normalizes, it follows that the term in brackets is equal to 1. Therefore,

$$I(A,b,c) = \frac{(2\pi)^{m/2}|A^{-1}|^{1/2}}{\exp(c + b^T A^{-1}b)}.$$

Appendix A.2

We derive the form of the distribution of x_A given x_B; the other result follows immediately by symmetry. Note that

$$\begin{aligned}
p(x_A \mid x_B) &= \frac{1}{\int_{x_A} p(x_A, x_B; \mu, \Sigma)dx_A} \cdot \left[\frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)\right] \\
&= \frac{1}{Z_1}\exp\left\{-\frac{1}{2}\left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)\right\}
\end{aligned}$$

where Z_1 is a proportionality constant which does not depend on x_A, and

$$\Sigma^{-1} = V = \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}.$$

To simplify this expression, observe that

$$\left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)$$
$$= (x_A - \mu_A)^T V_{AA}(x_A - \mu_A) + (x_A - \mu_A)^T V_{AB}(x_B - \mu_B)$$
$$+ (x_B - \mu_B)^T V_{BA}(x_A - \mu_A) + (x_B - \mu_B)^T V_{BB}(x_B - \mu_B).$$

Retaining only terms dependent on x_A (and using the fact that $V_{AB} = V_{BA}^T$), we have

$$p(x_A \mid x_B) = \frac{1}{Z_2} \exp\left(-\frac{1}{2}\left[x_A^T V_{AA} x_A - 2x_A^T V_{AA}\mu_A + 2x_A^T V_{AB}(x_B - \mu_B)\right]\right)$$

where Z_2 is a new proportionality constant which again does not depend on x_A. Finally, using the "completion-of-squares" argument (see Appendix A.1), we have

$$p(x_A \mid x_B) = \frac{1}{Z_3} \exp\left(-\frac{1}{2}(x_A - \mu')^T V_{AA}(x_A - \mu')\right)$$

where Z_3 is again a new proportionality constant not depending on x_A, and where $\mu' = \mu_A - V_{AA}^{-1} V_{AB}(x_B - \mu_B)$. This last statement shows that the distribution of x_A, conditioned on x_B, again has the form of a multivariate Gaussian. In fact, from the normalization property, it follows immediately that

$$x_A \mid x_B \sim \mathcal{N}(\mu_A - V_{AA}^{-1} V_{AB}(x_B - \mu_B), V_{AA}^{-1}).$$

To complete the proof, we simply note that

$$\begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} = \begin{bmatrix} (\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1} & -(\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1}\Sigma_{AB}\Sigma_{BB}^{-1} \\ -\Sigma_{BB}^{-1}\Sigma_{BA}(\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1} & (\Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB})^{-1} \end{bmatrix}$$

follows from standard formulas for the inverse of a partitioned matrix. Substituting the relevant blocks into the previous expression gives the desired result. □

Appendix A.3

In this section, we present an alternative (and easier) derivation of the conditional distribution of multivariate Gaussian distribution. Note that, as in Appendix A.2, we can write $p(x_A \mid x_B)$ as following:

$$p(x_A \mid x_B) = \frac{1}{\int_{x_A} p(x_A, x_B; \mu, \Sigma)dx_A} \cdot \left[\frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)\right] \tag{4}$$
$$= \frac{1}{Z_1} \exp\left\{-\frac{1}{2}\left(\begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix}\right)^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \begin{bmatrix} x_A - \mu_A \\ x_B - \mu_B \end{bmatrix}\right\} \tag{5}$$

where Z_1 is a proportionality constant which does not depend on x_A.

This derivation uses an additional assumption that the conditional distribution is a multivariate Gaussian distribution; in other words, we assume that $p(x_A \mid x_B) \sim \mathcal{N}(\mu^*, \Sigma^*)$ for some μ^*, Σ^*. (Alternatively, you can think about this derivation as another way of finding "completion-of-squares".)

The key intuition in this derivation is that $p(x_A \mid x_B)$ will be maximized when $x_A = \mu^* \triangleq x_A^*$. To maximize $p(x_A \mid x_B)$, we compute the gradient of $\log p(x_A \mid x_B)$ w.r.t. x_A and set it to zero. Using Equation (5), we have

$$\nabla_{x_A} \log p(x_A \mid x_B)|_{x_A = x_A^*} \tag{6}$$
$$= -V_{AA}(x_A^* - \mu_A) - V_{AB}(x_B - \mu_B) \tag{7}$$
$$= 0. \tag{8}$$

This implies that

$$\mu^* = x_A^* = \mu_A - V_{AA}^{-1} V_{AB}(x_B - \mu_B). \tag{9}$$

Similarly, we use the fact that the inverse covariance matrix of a Gaussian distribution $p(\cdot)$ is a negative Hessian of $\log p(\cdot)$. In other words, the inverse covariance matrix of a Gaussian distribution $p(x_A|x_B)$ is a negative Hessian of $\log p(x_A|x_B)$. Using Equation (5), we have

$$\Sigma^{*-1} = -\nabla_{x_A} \nabla_{x_A}^T \log p(x_A \mid x_B) \tag{10}$$
$$= V_{AA}. \tag{11}$$

Therefore, we get

$$\Sigma^* = V_{AA}^{-1}. \tag{12}$$

□

CS229 Supplemental Lecture notes
Hoeffding's inequality

John Duchi

1 Basic probability bounds

A basic question in probability, statistics, and machine learning is the following: given a random variable Z with expectation $\mathbb{E}[Z]$, how likely is Z to be close to its expectation? And more precisely, how close is it likely to be? With that in mind, these notes give a few tools for computing bounds of the form

$$\mathbb{P}(Z \geq \mathbb{E}[Z] + t) \quad \text{and} \quad \mathbb{P}(Z \leq \mathbb{E}[Z] - t) \tag{1}$$

for $t \geq 0$.

Our first bound is perhaps the most basic of all probability inequalities, and it is known as Markov's inequality. Given its basic-ness, it is perhaps unsurprising that its proof is essentially only one line.

Proposition 1 (Markov's inequality)**.** *Let $Z \geq 0$ be a non-negative random variable. Then for all $t \geq 0$,*

$$\mathbb{P}(Z \geq t) \leq \frac{\mathbb{E}[Z]}{t}.$$

Proof We note that $\mathbb{P}(Z \geq t) = \mathbb{E}[\mathbf{1}\{Z \geq t\}]$, and that if $Z \geq t$, then it must be the case that $Z/t \geq 1 \geq \mathbf{1}\{Z \geq t\}$, while if $Z < t$, then we still have $Z/t \geq 0 = \mathbf{1}\{Z \geq t\}$. Thus

$$\mathbb{P}(Z \geq t) = \mathbb{E}[\mathbf{1}\{Z \geq t\}] \leq \mathbb{E}\left[\frac{Z}{t}\right] = \frac{\mathbb{E}[Z]}{t},$$

as desired. $\square$

Essentially all other bounds on the probabilities (1) are variations on Markov's inequality. The first variation uses second moments—the variance—of a random variable rather than simply its mean, and is known as Chebyshev's inequality.

Proposition 2 (Chebyshev's inequality)**.** *Let Z be any random variable with* $\mathrm{Var}(Z) < \infty$*. Then*

$$\mathbb{P}(Z \geq \mathbb{E}[Z] + t \textit{ or } Z \leq \mathbb{E}[Z] - t) \leq \frac{\mathrm{Var}(Z)}{t^2}$$

for $t \geq 0$.

Proof The result is an immediate consequence of Markov's inequality. We note that if $Z \geq \mathbb{E}[Z] + t$, then certainly we have $(Z - \mathbb{E}[Z])^2 \geq t^2$, and similarly if $Z \leq \mathbb{E}[Z] - t$ we have $(Z - \mathbb{E}[Z])^2 \geq t^2$. Thus

$$\begin{aligned}\mathbb{P}(Z \geq \mathbb{E}[Z] + t \text{ or } Z \leq \mathbb{E}[Z] - t) &= \mathbb{P}((Z - \mathbb{E}[Z])^2 \geq t^2) \\ &\stackrel{(i)}{\leq} \frac{\mathbb{E}[(Z - \mathbb{E}[Z])^2]}{t^2} = \frac{\mathrm{Var}(Z)}{t^2},\end{aligned}$$

where step (i) is Markov's inequality. □

A nice consequence of Chebyshev's inequality is that averages of random variables with finite variance converge to their mean. Let us give an example of this fact. Suppose that Z_i are i.i.d. and satisfy $\mathbb{E}[Z_i] = 0$. Then $\mathbb{E}[Z_i] = 0$, while if we define $\bar{Z} = \frac{1}{n}\sum_{i=1}^n Z_i$ then

$$\mathrm{Var}(\bar{Z}) = \mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i\right)^2\right] = \frac{1}{n^2}\sum_{i,j\leq n}\mathbb{E}[Z_i Z_j] = \frac{1}{n^2}\sum_{i=1}^n \mathbb{E}[Z_i^2] = \frac{\mathrm{Var}(Z_1)}{n}.$$

In particular, for any $t \geq 0$ we have

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^n Z_i\right| \geq t\right) \leq \frac{\mathrm{Var}(Z_1)}{nt^2},$$

so that $\mathbb{P}(|\bar{Z}| \geq t) \to 0$ for any $t > 0$.

2 Moment generating functions

Often, we would like sharper—even exponential—bounds on the probability that a random variable Z exceeds its expectation by much. With that in mind, we need a stronger condition than finite variance, for which moment generating functions are natural candidates. (Conveniently, they also play nicely with sums, as we will see.) Recall that for a random variable Z, the *moment generating function* of Z is the function

$$M_Z(\lambda) := \mathbb{E}[\exp(\lambda Z)], \tag{2}$$

which may be infinite for some λ.

2.1 Chernoff bounds

Chernoff bounds use of moment generating functions in an essential way to give exponential deviation bounds.

Proposition 3 (Chernoff bounds). *Let Z be any random variable. Then for any $t \geq 0$,*

$$\mathbb{P}(Z \geq \mathbb{E}[Z] + t) \leq \min_{\lambda \geq 0} \mathbb{E}[e^{\lambda(Z-\mathbb{E}[Z])}]e^{-\lambda t} = \min_{\lambda \geq 0} M_{Z-\mathbb{E}[Z]}(\lambda)e^{-\lambda t}$$

and

$$\mathbb{P}(Z \leq \mathbb{E}[Z] - t) \leq \min_{\lambda \geq 0} \mathbb{E}[e^{\lambda(\mathbb{E}[Z]-Z)}]e^{-\lambda t} = \min_{\lambda \geq 0} M_{\mathbb{E}[Z]-Z}(\lambda)e^{-\lambda t}.$$

Proof We only prove the first inequality, as the second is completely identical. We use Markov's inequality. For any $\lambda > 0$, we have $Z \geq \mathbb{E}[Z] + t$ if and only if $e^{\lambda Z} \geq e^{\lambda \mathbb{E}[Z] + \lambda t}$, or $e^{\lambda(Z-\mathbb{E}[Z])} \geq e^{\lambda t}$. Thus, we have

$$\mathbb{P}(Z - \mathbb{E}[Z] \geq t) = \mathbb{P}(e^{\lambda(Z-\mathbb{E}[Z])} \geq e^{\lambda t}) \stackrel{(i)}{\leq} \mathbb{E}[e^{\lambda(Z-\mathbb{E}[Z])}]e^{-\lambda t},$$

where the inequality (i) follows from Markov's inequality. As our choice of $\lambda > 0$ did not matter, we can take the best one by minizing the right side of the bound. (And noting that certainly the bound holds at $\lambda = 0$.) $\square$

The important result is that Chernoff bounds "play nicely" with summations, which is a consequence of the moment generating function. Let us assume that Z_i are independent. Then we have that

$$M_{Z_1+\cdots+Z_n}(\lambda) = \prod_{i=1}^{n} M_{Z_i}(\lambda),$$

which we see because

$$\mathbb{E}\left[\exp\left(\lambda\sum_{i=1}^{n} Z_i\right)\right] = \mathbb{E}\left[\prod_{i=1}^{n}\exp(\lambda Z_i)\right] = \prod_{i=1}^{n}\mathbb{E}[\exp(\lambda Z_i)],$$

by of the independence of the Z_i. This means that when we calculate a Chernoff bound of a sum of i.i.d. variables, we need only calculate the moment generating function for *one* of them. Indeed, suppose that Z_i are i.i.d. and (for simplicity) mean zero. Then

$$\begin{aligned}\mathbb{P}\left(\sum_{i=1}^{n} Z_i \geq t\right) &\leq \frac{\prod_{i=1}^{n}\mathbb{E}\left[\exp(\lambda Z_i)\right]}{e^{\lambda t}} \\ &= (\mathbb{E}[e^{\lambda Z_1}])^n e^{-\lambda t},\end{aligned}$$

by the Chernoff bound.

2.2 Moment generating function examples

Now we give several examples of moment generating functions, which enable us to give a few nice deviation inequalities as a result. For all of our examples, we will have very convienent bounds of the form

$$M_Z(\lambda) = \mathbb{E}[e^{\lambda Z}] \leq \exp\left(\frac{C^2\lambda^2}{2}\right) \quad \text{for all } \lambda \in \mathbb{R},$$

for some $C \in \mathbb{R}$ (which depends on the distribution of Z); this form is *very* nice for applying Chernoff bounds.

We begin with the classical normal distribution, where $Z \sim \mathcal{N}(0, \sigma^2)$. Then we have

$$\mathbb{E}[\exp(\lambda Z)] = \exp\left(\frac{\lambda^2\sigma^2}{2}\right),$$

which one obtains via a calculation that we omit. (You should work this out if you are curious!)

A second example is known as a Rademacher random variable, or the random sign variable. Let $S = 1$ with probability $\frac{1}{2}$ and $S = -1$ with probability $\frac{1}{2}$. Then we claim that

$$\mathbb{E}[e^{\lambda S}] \le \exp\left(\frac{\lambda^2}{2}\right) \quad \text{for all } \lambda \in \mathbb{R}. \tag{3}$$

To see inequality (3), we use the Taylor expansion of the exponential function, that is, that $e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$. Note that $\mathbb{E}[S^k] = 0$ whenever k is odd, while $\mathbb{E}[S^k] = 1$ whenever k is even. Then we have

$$\begin{aligned}\mathbb{E}[e^{\lambda S}] &= \sum_{k=0}^{\infty} \frac{\lambda^k \mathbb{E}[S^k]}{k!} \\ &= \sum_{k=0,2,4,\ldots} \frac{\lambda^k}{k!} = \sum_{k=0}^{\infty} \frac{\lambda^{2k}}{(2k)!}.\end{aligned}$$

Finally, we use that $(2k)! \ge 2^k \cdot k!$ for all $k = 0, 1, 2, \ldots$, so that

$$\mathbb{E}[e^{\lambda S}] \le \sum_{k=0}^{\infty} \frac{(\lambda^2)^k}{2^k \cdot k!} = \sum_{k=0}^{\infty} \left(\frac{\lambda^2}{2}\right)^k \frac{1}{k!} = \exp\left(\frac{\lambda^2}{2}\right).$$

Let us apply inequality (3) in a Chernoff bound to see how large a sum of i.i.d. random signs is likely to be.

We have that if $Z = \sum_{i=1}^{n} S_i$, where $S_i \in \{\pm 1\}$ is a random sign, then $\mathbb{E}[Z] = 0$. By the Chernoff bound, it becomes immediately clear that

$$\mathbb{P}(Z \ge t) \le \mathbb{E}[e^{\lambda Z}]e^{-\lambda t} = \mathbb{E}[e^{\lambda S_1}]^n e^{-\lambda t} \le \exp\left(\frac{n\lambda^2}{2}\right) e^{-\lambda t}.$$

Applying the Chernoff bound technique, we may minimize this in $\lambda \ge 0$, which is equivalent to finding

$$\min_{\lambda \ge 0} \left\{ \frac{n\lambda^2}{2} - \lambda t \right\}.$$

Luckily, this is a convenient function to minimize: taking derivatives and setting to zero, we have $n\lambda - t = 0$, or $\lambda = t/n$, which gives

$$\mathbb{P}(Z \ge t) \le \exp\left(-\frac{t^2}{2n}\right).$$

In particular, taking $t = \sqrt{2n \log \frac{1}{\delta}}$, we have

$$\mathbb{P}\left(\sum_{i=1}^{n} S_i \geq \sqrt{2n \log \frac{1}{\delta}}\right) \leq \delta.$$

So $Z = \sum_{i=1}^{n} S_i = O(\sqrt{n})$ with extremely high probability—the sum of n independent random signs is essentially never larger than $O(\sqrt{n})$.

3 Hoeffding's lemma and Hoeffding's inequality

Hoeffding's inequality is a powerful technique—perhaps the most important inequality in learning theory—for bounding the probability that sums of bounded random variables are too large or too small. We will state the inequality, and then we will prove a weakened version of it based on our moment generating function calculations earlier.

Theorem 4 (Hoeffding's inequality)**.** *Let $Z_1, \ldots, Z_n$ be independent bounded random variables with $Z_i \in [a, b]$ for all i, where $-\infty < a \leq b < \infty$. Then*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right) \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

and

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \leq -t\right) \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

for all $t \geq 0$.

We prove Theorem 4 by using a combination of (1) Chernoff bounds and (2) a classic lemma known as Hoeffding's lemma, which we now state.

Lemma 5 (Hoeffding's lemma)**.** *Let Z be a bounded random variable with $Z \in [a, b]$. Then*

$$\mathbb{E}[\exp(\lambda(Z - \mathbb{E}[Z]))] \leq \exp\left(\frac{\lambda^2(b-a)^2}{8}\right) \quad \textit{for all } \lambda \in \mathbb{R}.$$

Proof We prove a slightly weaker version of this lemma with a factor of 2 instead of 8 using our random sign moment generating bound and an inequality known as *Jensen's inequality* (we will see this very important inequality later in our derivation of the EM algorithm). Jensen's inequality states the following: if $f : \mathbb{R} \to \mathbb{R}$ is a *convex* function, meaning that f is bowl-shaped, then

$$f(\mathbb{E}[Z]) \leq \mathbb{E}[f(Z)].$$

The simplest way to remember this inequality is to think of $f(t) = t^2$, and note that if $\mathbb{E}[Z] = 0$ then $f(\mathbb{E}[Z]) = 0$, while we generally have $\mathbb{E}[Z^2] > 0$. In any case, $f(t) = \exp(t)$ and $f(t) = \exp(-t)$ are convex functions.

We use a clever technique in probability theory known as *symmetrization* to give our result (you are not expected to know this, but it is a very common technique in probability theory, machine learning, and statistics, so it is good to have seen). First, let Z' be an independent copy of Z with the same distribution, so that $Z' \in [a, b]$ and $\mathbb{E}[Z'] = \mathbb{E}[Z]$, but Z and Z' are independent. Then

$$\mathbb{E}_Z[\exp(\lambda(Z - \mathbb{E}_Z[Z]))] = \mathbb{E}_Z[\exp(\lambda(Z - \mathbb{E}_{Z'}[Z']))] \stackrel{(i)}{\leq} \mathbb{E}_Z[\mathbb{E}_{Z'} \exp(\lambda(Z - Z'))],$$

where $\mathbb{E}_Z$ and $\mathbb{E}_{Z'}$ indicate expectations taken with respect to Z and Z'. Here, step (i) uses Jensen's inequality applied to $f(x) = e^{-x}$. Now, we have

$$\mathbb{E}[\exp(\lambda(Z - \mathbb{E}[Z]))] \leq \mathbb{E}\left[\exp\left(\lambda(Z - Z')\right)\right].$$

Now, we note a curious fact: the difference $Z - Z'$ is symmetric about zero, so that if $S \in \{-1, 1\}$ is a random sign variable, then $S(Z - Z')$ has exactly the same distribution as $Z - Z'$. So we have

$$\begin{aligned}\mathbb{E}_{Z,Z'}[\exp(\lambda(Z - Z'))] &= \mathbb{E}_{Z,Z',S}[\exp(\lambda S(Z - Z'))] \\ &= \mathbb{E}_{Z,Z'}\left[\mathbb{E}_S\left[\exp(\lambda S(Z - Z')) \mid Z, Z'\right]\right].\end{aligned}$$

Now we use inequality (3) on the moment generating function of the random sign, which gives that

$$\mathbb{E}_S\left[\exp(\lambda S(Z - Z')) \mid Z, Z'\right] \leq \exp\left(\frac{\lambda^2 (Z - Z')^2}{2}\right).$$

But of course, by assumption we have $|Z - Z'| \leq (b - a)$, so $(Z - Z')^2 \leq (b - a)^2$. This gives

$$\mathbb{E}_{Z,Z'}[\exp(\lambda(Z - Z'))] \leq \exp\left(\frac{\lambda^2 (b - a)^2}{2}\right).$$

This is the result (except with a factor of 2 instead of 8). $\square$

Now we use Hoeffding's lemma to prove Theorem 4, giving only the upper tail (i.e. the probability that $\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t$) as the lower tail has a similar proof. We use the Chernoff bound technique, which immediately tells us that

$$\begin{aligned}\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right) &= \mathbb{P}\left(\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq nt\right)\\ &\leq \mathbb{E}\left[\exp\left(\lambda\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i])\right)\right]e^{-\lambda nt}\\ &= \left(\prod_{i=1}^{n}\mathbb{E}[e^{\lambda(Z_i - \mathbb{E}[Z_i])}]\right)e^{-\lambda nt} \stackrel{(i)}{\leq} \left(\prod_{i=1}^{n}e^{\frac{\lambda^2(b-a)^2}{8}}\right)e^{-\lambda nt}\end{aligned}$$

where inequality (i) is Hoeffding's Lemma (Lemma 5). Rewriting this slightly and minimzing over $\lambda \geq 0$, we have

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right) \leq \min_{\lambda\geq 0}\exp\left(\frac{n\lambda^2(b-a)^2}{8} - \lambda nt\right) = \exp\left(-\frac{2nt^2}{(b-a)^2}\right),$$

as desired.