Mathematical Tools for Neuroscience (NEU 314)
Princeton University, Spring 2016
Jonathan Pillow

Lecture 10: Least Squares Squares

1 Calculus with Vectors and Matrices

Here are two rules that will help us out with the derivations that come later. First of all, let's define what we mean by the gradient of a function $f(\vec{x})$ that takes a vector ($\vec{x}$) as its input. This is just a vector whose components are the derivatives with respect to each of the components of $\vec{x}$:

$$\nabla f \triangleq \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{bmatrix}$$

Where ∇ (the "nabla" symbol) is what we use to denote gradient, though in practice I will often be lazy and write simply $\frac{df}{d\vec{x}}$ or maybe $\frac{\partial}{\partial \vec{x}} f$.

(Also, in case you didn't know it, $\triangleq$ is the symbol denoting "is defined as").

Ok, here are the two useful identities we'll need:

1. Derivative of a linear function:

$$\frac{\partial}{\partial \vec{x}} \vec{a} \cdot \vec{x} \quad = \quad \frac{\partial}{\partial \vec{x}} \vec{a}^\top \vec{x} \quad = \quad \frac{\partial}{\partial \vec{x}} \vec{x}^\top \vec{a} \quad = \quad \vec{a} \tag{1}$$

   (If you think back to calculus, this is just like $\frac{d}{dx} ax = a$).

2. Derivative of a quadratic function: if A is symmetric, then

$$\frac{\partial}{\partial \vec{x}} \vec{x}^\top A \vec{x} = 2A\vec{x} \tag{2}$$

   (Again, thinking back to calculus this is just like $\frac{d}{dx} ax^2 = 2ax$).

   If you ever need it, the more general rule (for non-symmetric A) is:

$$\frac{\partial}{\partial \vec{x}} \vec{x}^\top A \vec{x} = (A + A^\top)\vec{x},$$

   which of course is the same thing as $2A\vec{x}$ when A is symmetric.

2 Least Squares Regression

Ok, let's get down to it!

Suppose someone hands you a stack of N vectors, $\{\vec{x}_1, \ldots \vec{x}_N\}$, each of dimension d, and an associated scalar observation $\{y_1, \ldots, y_N\}$. You'd like to estimate a linear function that allows us to predict y from $\vec{x}$ as well as possible:

$$y_i \approx \vec{w}^\top \vec{x}_i$$

for some weight vector $\vec{w}$.

Specifically, we'd like to minimize the squared prediction error, so we'd like to find the $\vec{w}$ that minimizes

$$\text{squared error} = \sum_{i=1}^{N} (y_i - \vec{x}_i \cdot \vec{w})^2 \tag{3}$$

We're going to write this as a vector equation to make it easier to derive the solution. Let Y be a vector composed of the stacked observations $\{y_i\}$, and let X be the vector whose rows are the vectors $\{\vec{x}_i\}$ (which is known as the *design matrix*):

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \qquad X = \begin{bmatrix} — & \vec{x}_1 & — \\ & \vdots & \\ — & \vec{x}_N & — \end{bmatrix}$$

Then we can rewrite the squared error given above as the squared vector norm of the residual error between Y and $X\vec{w}$:

$$\text{squared error} = ||Y - X\vec{w}||^2 \tag{4}$$

2.1 Derivation #1: calculus

We can call our first derivation of the solution (i.e., the $\vec{w}$ vector that minimizes the squared error defined above) the "straightforward calculus" derivation. We will differentiate the error with respect to $\vec{w}$, set it equal to zero (i.e., implying we have a local optimum of the error), and solve for $\vec{w}$. All we're going to need is some algebra for pushing around terms in the error, and the vector calculus identities we put at the top.

Let's go!

$$\frac{\partial}{\partial \vec{w}} SE = \frac{\partial}{\partial \vec{w}} (Y - X\vec{w})^\top (Y - X\vec{w}) \tag{5}$$

$$= \frac{\partial}{\partial \vec{w}} \left(Y^\top Y - 2\vec{w}^\top X^\top + \vec{w}^\top X^\top X Y \right) \tag{6}$$

$$= -2X^\top Y + 2X^\top X\vec{w} \quad = \quad 0. \tag{7}$$

We can then solve this for $\vec{w}$ as follows:

$$X^\top X\vec{w} = X^\top Y \tag{8}$$

$$\implies \quad \vec{w} = (X^\top X)^{-1} X^\top Y \tag{9}$$

Easy, right?

(Note: we're assuming that $X^\top X$ is full rank so that its inverse exists, implying that $N > d$ and the rows are not all linearly dependent with each other.)

2.2 Derivation #2: orthogonality

Our second derivation is even easier, and it has the added advantage that it gives us some geomtrix insight.

Let's think about the design matrix X in terms of its d columns instead of its N rows. Let $\{X_j\}$ denote the $j'th$ column, i.e.,

$$X = \begin{bmatrix} | & & | \\ X_1 & \cdots & X_d \\ | & & | \end{bmatrix} \tag{10}$$

The columns of X span a d-dimensional subspace within the larger N-dimensional vector space that contains the vector Y. Generally Y does not lie exactly within this subspace. Least squares regression is therefore trying to find the linear combination of these vectors, $X\vec{w}$, that gets as close to possible to Y.

What we know about the optimal linear combination is that it corresponds to dropping a line down from Y to the subspace spanned by $\{X_1, \ldots X_D\}$ at a right angle. In other words, the error vector $(Y - X\vec{w})$ (also known as the *residual error*) should be orthogonal to every column of X:

$$(Y - X\vec{w}) \cdot X_j = 0, \tag{11}$$

for all j columns. Written as a matrix equation this means:

$$(Y - X\vec{w})^\top X = \vec{0} \tag{12}$$

where $\vec{0}$ is d-component vector of zeros.

We should quickly be able to see that solving this for $\vec{w}$ gives us the solution we were looking for:

$$X^\top(Y - X\vec{w}) = X^\top Y - X^\top X\vec{w} = 0 \tag{13}$$

$$\implies \quad (X^\top X)\vec{w} = X^\top Y \tag{14}$$

$$\implies \quad \vec{w} = (X^\top X)^{-1} X^\top Y. \tag{15}$$

So to summarize: the requirement that the residual errors between Y and $X\vec{w}$ be orthogonal to the columns of X was all we needed to derive the optimal $\vec{w}$!

3 Derivation for PCA

In the last lecture on PCA we showed that *if* we restricted ourselves to considering eigenvectors of the $X^\top X$, then the eigenvector with largest eigenvalue captured the largest projected-sum-of-

squares of the vectors contained in X. But we didn't show that eigenvectors themselves correspond to optima of the PCA loss function.

To recap briefly, we want to find the maximum of

$$\vec{v}^\top C\vec{v},$$

where $C = X^\top X$ is the (scaled) covariance of zero-centered data vectors X, subject to the constraint that $\vec{v}$ is a unit vector ($\vec{v}^\top\vec{v} = 1$).

We can solve this kind of optimization problem using the method of Lagrange multipliers. The basic idea is that we minimize a function that is our original function plus a lagrange multiplier λ times an expression that is zero when our constraint is satisfied. For this problem we can define the Lagrangian:

$$L = \vec{v}^\top C\vec{v} + \lambda(\vec{v}^\top\vec{v} - 1). \tag{16}$$

We will want solutions for which

$$\frac{\partial}{\partial\vec{v}}\ L = 0 \tag{17}$$

$$\frac{\partial}{\partial\lambda}\ L = 0. \tag{18}$$

Note that the second of these is satisfied if and only if $\vec{v}$ is a unit vector (which is reassuring).

The first equation gives us:

$$\frac{\partial}{\partial\vec{v}}\ L = \frac{\partial}{\partial\vec{v}}\ \vec{v}C\vec{v} + \lambda(\vec{v}^\top\vec{v} - 1) = 2C\vec{v} - 2\lambda\vec{v} \quad = \quad 0, \tag{19}$$

which implies

$$C\vec{v} = -\lambda\vec{v}. \tag{20}$$

What is this? It's the eigenvector equation! This implies that the derivative of the Lagrangian is zero when $\vec{v}$ is an eigenvector of C. So this establishes, combined with the argument from last week, that the unit vector that captures the greatest squared projection of the raw data is the top eigenvector of C.