An Introduction to Empirical Modeling

Douglas Robert Hundley
Mathematics Department
Whitman College

August 28, 2018

Contents

Chapter 1

Learning and Modeling

1.1 What is Learning?

Here are some definitions of learning:[1]

- "Learning involves strengthening correct responses and weakening incorrect responses. Learning involves adding new information to your memory. Learning involves making sense of the presented material by attending to relevant information, mentally reorganizing it, and connecting it with what you already know."

  From eLearning and the Science of Instruction by Ruth C. Clark and Richard E. Mayer

- "Learning is the relatively permanent change in a persons knowledge or behavior due to experience. This definition has three components: 1) the duration of the change is long-term rather than short-term; 2) the locus of the change is the content and structure of knowledge in memory or the behavior of the learner; 3) the cause of the change is the learners experience in the environment rather than fatigue, motivation, drugs, physical condition or physiologic intervention."

  From Learning in Encyclopedia of Educational Research, Richard E. Mayer

- Definition of Learning:

  - The act or experience of one that learns.
  - Knowledge or skill acquired by instruction or study.
  - Modification of a behavioral tendency by experience (such as exposure to conditioning).

  From the Merriam-Webster Dictionary.

This class will generally perform **modeling** via **machine learning**, so we should understand what some of these terms mean. The definitions, which work well if we're thinking primarily of human beings, don't lead us to a satisfactory definition that is easy to check against. Our definition of learning will tend to be a bit more on the behaviorist side of the theory:

In a broad sense, *learning is the process of building a "desirable" association between stimulus and response (domain and range), and is measured through resulting behavior on stimulus that has not been previously seen.*

[1] Downloaded from http://theelearningcoach.com/learning/10-definitions-learning/

Some vocabulary in Machine Learning

In machine learning, problems are typically classified in one of two ways: *supervised* or *unsupervised* learning.

In *supervised learning*, we are given examples of proper behavior, and we want the computer to emulate (and extrapolate from) that behavior.

In *unsupervised learning*, no specific outputs are given per input, but rather an overall goal is given. Here are some examples to help with the definition:

- We might have measurement data on flowers, and we want to classify each set of measurements as to which flower it is. Normally, we have lots of examples of each type of flower.
- We have taken samples of time and temperature of a certain object over time. We want to build a function that inputs time and outputs temperature (so that the function predicts the temp).
- Suppose you have an old clunker of a car that doesn't have much of an engine. You're stuck in a valley, and so the only way out will be to go as fast as you can for a while, then let gravity take you down the hill, then perhaps coast up the other side of the hill. We would then accelerate again, and coast, and so on. You hope that you can build up enough momentum to get out of the valley (that's the goal).
- Suppose you're driving a tractor-trailer, and you need to back the trailer into a loading dock (that's your goal).
- In a game of chess, the input would be the position of each of the chess pieces. The overall goal is to win the game.

In general, supervised learning is MUCH easier than unsupervised learning. That's because in supervised learning, we are given exemplars of "proper" behavior, and we just have to do the function building.

In unsupervised learning, we don't know anything about the environment that we're in, and we have no examples to go on. Therefore, typically through some trial-and-error exploration, we have to build up experience before we can do any function building. We'll see this in detail when we discuss the "n-armed bandit" later on.

Right now, there is a lot of machine learning going on around us. Some examples include spam filtering your email, machine-reading addresses at the post office, credit card fraud detection, movie recommendations from Netflix, are among some of the most well known applications. In fact, you might note that these are probably all *supervised* learning problems.

For the rest of the this chapter, and for the next, we'll stick primarily with supervised learning. We'll then go to our first unsupervised example with the $n-$armed bandit.

Some Matlab Related Vocabulary

Suppose we are given input-output pairs of data (in two dimensions with 100 data points, for example, we might be given a matrix that is 2×100). We would then build a function that takes an input to an output.

- The **targets** are the outputs that we want from the model (the "$y-$values of a function). We'll usually denote these using t's rather than y's.
- The **model output**, or just output, is what we actually get from the function. These are typically denoted by y.
- In function building, we want $t = y$, but usually we will have some error to deal with.

1.1.1 Questions for Discussion:

1. Consider the concept of *superstition*: This is a belief that one must engage in certain behaviors in order to receive a certain reward, where in reality, the reward did not depend on those behaviors. Is it possible for a computer to engage in superstitious activity? Discuss in terms of the supervised versus unsupervised learning paradigms.

2. A signal light comes on and is followed by one of two other lights. The goal is to predict which of the lights comes on given that the signal light comes on. The experimenter is free to arrange the pattern of the two response lights in any way- for example, one might come on 75% of the time.

   Let E_1, E_2 denote the event that the first (second) light comes on, and let A_1, A_2 denote the prediction that the first (second) light comes on (respectively). Let π be the probability that E_1 occurs.

   (a) If the goal is to maximize your reward through accurate predictions, what should you do in this experiment? Just give a heuristic answer- you do not have to formally justify it.

   (b) How would you program a machine to maximize it's prediction accuracy? Can you state this in mathematical terms?

   (c) What do you think happens with actual subject (human) trials?

Chapter 2

Statistics

As we have discussed previously, statistics will naturally emerge when we have to deal either with models that have random characteristics, or with data that may incorporate some kind of random noise- and in fact, we will usually assume that the error present in the system is some kind of random number.

You'll notice that in this section we won't cover much probability, but we will be looking at statistical measures of data like the mean and variance. We might also need to look at different ways random numbers are produced, and that will require a bit of probability.

With those caveats, we'll bring in a few ideas that we'll need for later.

2.1 Functions that Define Data

The basic way of defining "random" data is through the use of a *probability density function*, or pdf.

Example 1: Given a fair dice, the probablity of rolling any number from 1 to 6 is equally likely. If we let $f(x)$ be the probability that we get x, then in this case:

$$f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{1}{6}$$

In this case, f is our pdf.

Example 2: Given the 14 points:

$$\{1, 2, 3, 2, 1, 1, 2, 3, 3, 2, 2, 1, 1, 1\}$$

we might deduce that the probability of having "1" is 6/14, the probability of getting a "2" is 5/14, and the probability of getting a 3 is 3/14. Thus the pdf is:

$$f(1) = \frac{6}{14} \qquad f(2) = \frac{5}{14} \qquad f(3) = \frac{3}{14}$$

In this case, we might assume that the process generating these numbers is adequately represented by this set- For example, the process is only generating the numbers 1, 2 and 3, and we have sampled long enough to get a good approximation to the frequency of each number. This is what is called a "frequentist" approach to building a probability density function.

2.1.1 Matlab and Histograms

A histogram is a frequency plot- That is, the possible outcomes of an experiment are binned, and we run the experiment many times. We will then count how many times the outcomes are in each bin. For example, if we roll our dice 50 times, we can plot the frequency of each output number using Matlab:

```
A=randi(6,[1,50]); %A is a vector 1 x 50 with random integers
histogram(A);      % histogram should be used in place of hist.
```

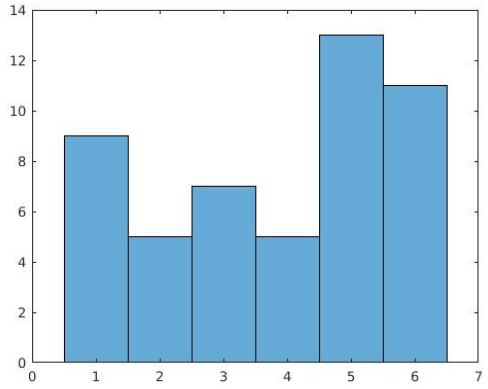

Figure 2.1: The histogram of data in vector A.

2.2 Standard Density Functions

Some pdf's are so common that it is a good idea to be at least a little familiar with some of them. Here are some formal definitions to start, followed by some examples.

Definition: A (continuous) function $f(x)$ is said to be a probability density function if it satisfies the following conditions:

1. $f(x)$ is always non-negative.
2. $\int_{-\infty}^{\infty} f(x)\,dx = 1$
3. The probability of an event between values $x = a$ and $x = b$ is given by:

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

Definition: A discrete probability density function will be a finite set of numbers, $\{P_1, P_2, \ldots, P_k\}$, so that:

1. P_i is non-negative, for all i.

2. $\sum_{i=1}^{k} P_k = 1$

Let's take a look at some template probability distributions:

- **The Uniform Distribution**
  - The Continuous Version:

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

  - The Discrete Version (using N bins over the same interval):

$$\Pr\left(a + (i-1)\frac{b-a}{N} \le x \le a + i\frac{b-a}{N}\right) = \frac{1}{N} = P_i, i = 1, 2, \ldots, N.$$

  - In Matlab, to obtain a value from a uniform distribution over $[0, 1]$, we type `rand`. To get integers with a uniform distribution, use `randi`.

- **The Normal (or Gaussian) Distribution**
  - The Continuous Version: The Normal distribution with mean μ and variance σ^2 (to be defined shortly) is defined as:

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

    This is the common "bell-shaped curve"; the constant in the front is needed to make the integral evaluate to 1. Note that the normal distribution with zero mean and variance 1 simplifies to:

$$f(x) = \mathcal{N}(0, 1) = \frac{1}{\sqrt{2\pi}}\,\mathrm{e}^{-x^2}$$

    The plot of this function is given below.
  - In Matlab, we can obtain values from a normal distribution with zero mean and unit variance by `randn`.

2.3 Descriptive Statistics

The most basic way to characterize a data set is through one number- the mean (or median or mode).

- The *Sample Mean* for a discrete set of m numbers, $x_1, \ldots, x_m$ is given by:

$$\bar{x} = \frac{1}{m}\sum_{k=1}^{m} x_k$$

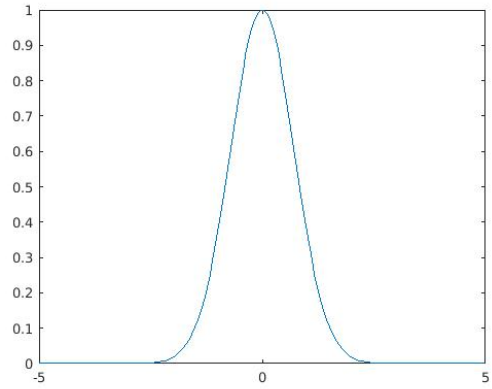

Figure 2.2: The normal distribution. In this case, $y = \mathrm{e}^{-x^2}$.

We will typically use the sample mean rather than the population mean, which is defined using the underlying pdf:

$$\mu = E(X) = \sum_{\text{all x}} xf(x)$$

You might remember this formula by seeing that this is really a weighted average, with those events most probable getting a higher weight than events less probable.

Also, if your data is being drawn independently from a fixed p.d.f., then the sample mean will converge to the population mean, as the number of samples gets very large.

Suppose we have m vectors in $\mathbb{R}^n$. we can similarly define the (sample) mean, just replace the scalar x_k with the k^{th} vector:

$$\bar{\boldsymbol{x}} = \frac{1}{m}\sum_{k=1}^{m} \boldsymbol{x}^{(k)}$$

The j^{th} element of the sample mean vector is just the sample mean of the (scalar) data in the j^{th} dimension of your collection of vectors.

In Matlab, the mean is a built-in function. The command is `mean`, and the output depends on whether you input a vector or a matrix of data.

For vectors, `mean(x)` outputs a scalar.

```
m=mean(X,1);  %Returns a row vector
m=mean(X,2);  %Returns a column vector
```

For matrices, one can compute a row mean (which is a row), a column mean (which is a column) or a **grand mean**. In this context, the grand mean of a matrix is found by taking the mean of all the entries of the matrix, so the grand mean of a matrix is a scalar.

See the end of this section for more on mean subtracting a matrix of data.

- The *Median* is a number so that exactly half the data is above that number, and half the data is below that number. Although the median does not have to be unique, we follow the definitions below if we are given a finite sample:

If there are an odd number of data points, the median is the middle point. If there is an even number of data points, then there are two numbers in the middle- the median is the average of these.

Although not used terribly often, Matlab will perform the median as well as the mean:

```
m=median(X);
```

where the output is a scalar if X is a vector, or a row vector if X is a matrix.

- The *Mode* is the value taken the most number of times. In the case of ties, the data is multi-modal.

We'll compare these definitions in the Exercises.

2.3.1 Centering and Double Centering Data

Let matrix A be $n \times m$, which may be considered n points in R^m or m points in $\mathbb{R}^n$. If we wish to look at A both ways, a double-centering may be appropriate.

The result of the double-centering will be that (in Matlab), we determine $\hat{A}$ so that

$$\texttt{mean}(\hat{A}, 1) = 0, \qquad \texttt{mean}(\hat{A}, 2) = 0$$

The algorithm is (in Matlab):

```
%Let A be n times m
[n,m]=size(A);
rowmean=mean(A);
A1=A-repmat(rowm,n,1);
colmean=mean(A1,2);
Ahat=A1-repmat(colmean,1,m);
```

or, equivalently:

```
%Let A be n times m
[n,m]=size(A);
colmean=mean(A,2);
A1=A-repmat(colmean,1,m);
rowmean=mean(A1,1);
Ahat=A1-repmat(rowmean,n,1);
```

Proof: For the first version (row mean first):

Let A_1 be the matrix A with the row mean $\mathbf{b}$ subtracted:

$$A_1 = \begin{bmatrix} a_{11} - b_1 & a_{12} - b_2 & \cdots & a_{1m} - b_m \\ a_{21} - b_1 & a_{22} - b_2 & \cdots & a_{2m} - b_m \\ \vdots & & & \vdots \\ a_{n1} - b_1 & a_{n2} - b_2 & \cdots & a_{nm} - b_m \end{bmatrix}$$

with

$$b_j = \frac{1}{n}\sum_{i=1}^{n} a_{ij}$$

Now define $\mathbf{c}$ as the column mean of A_1. Mean subtraction of this column results in the $\hat{A}$, written explicitly as:

$$\hat{A} = \begin{bmatrix} a_{11}-b_1-c_1 & a_{12}-b_2-c_1 & \cdots & a_{1m}-b_m-c_1 \\ a_{21}-b_1-c_2 & a_{22}-b_2-c_2 & \cdots & a_{2m}-b_m-c_2 \\ \vdots & & & \vdots \\ a_{n1}-b_1-c_n & a_{n2}-b_2-c_n & \cdots & a_{nm}-b_m-c_n \end{bmatrix}$$

By definition, the column mean of $\hat{A}$ is zero. Is the new row mean zero? It is clear that the new row mean is zero iff $\sum_k c_k = 0$, which we now show:

Proof that $\sum_{k=1}^{n} c_k = 0$

We explicitly write down what c_k is:

$$c_k = \frac{1}{m}\sum_{j=1}^{m}(a_{kj} - b_j)$$

and substitute the expression for b_j,

$$c_k = \frac{1}{m}\sum_{j=1}^{m}\left(a_{kj} - \frac{1}{n}\sum_{i=1}^{n} a_{ij}\right) = \frac{1}{m}\sum_{j=1}^{m} a_{kj} - \frac{1}{mn}\sum_{j=1}^{m}\sum_{i=1}^{n} a_{ij}$$

Now sum over k:

$$\sum_{k=1}^{n} c_k = \sum_{k=1}^{n}\left(\frac{1}{m}\sum_{j=1}^{m} a_{kj} - \frac{1}{mn}\sum_{j=1}^{m}\sum_{i=1}^{n} a_{ij}\right) =$$

$$\frac{1}{m}\sum_{k=1}^{n}\sum_{j=1}^{m} a_{kj} - \frac{n}{mn}\sum_{j=1}^{m}\sum_{i=1}^{n} a_{ij} = 0$$

It may be clear that these two methods produce the same result (e.g., row subtract first, then column subtract or vice-versa). If we examine the (i,j)th entry of $\hat{A}$,

$$\hat{A}_{ij} = a_{ij} - b_j - c_i = a_{ij} - \frac{1}{n}\sum_{k=1}^{n} a_{kj} - \frac{1}{m}\sum_{k=1}^{m} a_{ik} + \sum_{r=1}^{m}\sum_{s=1}^{n} a_{rs}$$

Therefore, to double center a matrix of data, each element has subtracted from it its corresponding row mean and column mean, and we add back the average of all the elements.

As a final note, this technique is only suitable if it is reasonable that the $m \times n$ matrix may be data in either $\mathbb{R}^n$ or $\mathbb{R}^m$. For example, you probably would not double center a data matrix that is 5000×2 unless there is a specific reason to do so.

Example

Let the matrix be defined below. Verify that the row mean, column mean are vectors with all 1's and the grand mean is 1 as well. After that, double center the array.

$$\begin{bmatrix} 3 & 0 & -1 & 2 \\ -1 & 2 & 3 & 0 \end{bmatrix}$$

SOLUTION: Computing the means, we see that the row mean is [1111] the column mean is $[1,1]^T$ and the grand mean is also 1! Double centering means we'll subtract the row and column mean, then add back in the grand mean. In this special matrix, that means that we'll subtract 1 from every value:

$$\begin{bmatrix} 3 & 0 & -1 & 2 \\ -1 & 2 & 3 & 0 \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} 2 & -1 & -2 & 1 \\ -2 & 1 & 2 & -1 \end{bmatrix}$$

Now you should notice that the new row mean and new column mean are vectors with all zeros.

2.4 The Variance and Standard Deviation

The number that is used to describe the spread of the data about its mean is the *variance*. As with the mean, we rarely know the underlying distribution, so again we'll focus on the sample variance.

Let $\{x_1, \ldots, x_m\}$ be m real numbers. Then the **sample variance** is:

$$s^2 = \frac{1}{m-1} \sum_{k=1}^{m} (x_k - \bar{x})^2$$

where $\bar{x}$ is the mean of the data. If we think of the data as a vector of length m, then this formula becomes:

$$s^2 = \frac{1}{m-1} \|\boldsymbol{x} - \bar{x}\|^2$$

To be complete, we include the population variance below:

$$\sigma^2 = E((x-\mu)^2) = \sum_{\text{all x}} (x-\mu)^2 f(x)$$

Finally, the **standard deviation** is the square root of the variance, so the standard deviation is σ.

Quick Example

Let's take some template data to look at what the variance (and standard deviation) measure: Consider the data:

$$-\frac{2}{n}, -\frac{1}{n}, 0, \frac{1}{n}, \frac{2}{n}$$

If n is large, our data is tightly packed together about the mean, 0. If n is small, the data are spread out. The variance of this sample is:

$$s^2 = \frac{1}{4}\left(\frac{4+1+0+1+4}{n^2}\right) = \frac{5}{2}\frac{1}{n^2}$$

so that the standard deviation is:

$$s = \sqrt{\frac{5}{2}}\frac{1}{n}$$

and this is in agreement with our heuristic: If n is large, our data is tightly packed about the mean, and the standard deviation is small. If n is small, our data is loosely distributed about the mean, and the standard deviation is large. Another way to look at the standard deviation is in linear algebra terms: If the data is put into a vector of length m (call it $\boldsymbol{x}$), then the (sample) standard deviation can be computed as:

$$s = \frac{\|\boldsymbol{x}\|}{\sqrt{m-1}}$$

2.4.1 Covariance and Correlation Coefficients

If we have two data sets, sometimes we would like to compare them to see how they relate to each other. In this case, it is important that the two data sets be ordered so that x_1 is being compared to y_1, then x_2 is compared to y_2, and so on.

Definition: Let $X = \{x_1, \ldots, x_n\}, Y = \{y_1, \ldots, y_n\}$ be two ordered data sets with means m_x, m_y respectively. Then the *sample covariance* of the data sets is given by:

$$\mathrm{Cov}(X,Y) = s_{xy}^2 = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - m_x)(y_k - m_y)$$

There are exercises at the end of the chapter that will reinforce the notation and give you some methods for manipulating the covariance. In the meantime, it is easy to remember this formula if you think of the following:

If X and Y have mean zero, and we think of X and Y as vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, then the covariance is just the dot product between the vectors, divided by $n-1$:

$$\mathrm{Cov}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n-1}\boldsymbol{x}^T\boldsymbol{y}$$

We can then interpret what it means for X, Y to have a covariance of zero: $\boldsymbol{x}$ is "orthogonal" to $\boldsymbol{y}$. Continuing with this analogy, if we normalized by the size of $\boldsymbol{x}$ and the size of $\boldsymbol{y}$, we'd get the cosine of the angle between them. This is the definition of the correlation coefficient, and gives the relationship between the covariance and correlation coefficient:

Definition: The *correlation coefficient* between x and y is given by:

$$r_{xy} = \frac{s_{xy}^2}{s_x s_y} = \frac{\sum_{k=1}^{n}(x_k - m_x)(y_k - m_y)}{\sqrt{\sum_{k=1}^{n}(x_k - m_x)^2 \cdot \sum_{k=1}^{n}(y_k - m_y)^2}}$$

Again, thinking of X, Y as having zero mean and placing the data in vectors $\boldsymbol{x}, \boldsymbol{y}$, then this formula becomes:

$$r_{xy} = \frac{\boldsymbol{x}^T\boldsymbol{y}}{\|\boldsymbol{x}\| \cdot \|\boldsymbol{y}\|} = \cos(\theta)$$

This works out so nicely because we have a $\frac{1}{n-1}$ in both the numerator and denominator, so they cancel each other out.

We also see immediately that r_{xy} can only take on the real numbers between -1 and 1. Some interesting values of r_{xy}:

If r_{xy} is:	Then the data is:
1	Perfectly correlated ($\theta = 0$)
0	Uncorrelated ($\theta = \frac{\pi}{2}$)
-1	Perfectly (negatively) correlated ($\theta = \pi$)

One last comment before we leave this section: The covariance s_{xy}^2 and correlation coefficient r_{xy} only look for *linear* relationships between data sets!

For example, we know that $\sin(x)$ and $\cos(x)$ (as functions, or as data points sampled at equally spaced intervals) will be uncorrelated, but, because $\sin^2(x) + \cos^2(x) = 1$, we see that $\sin^2(x)$ and $\cos^2(x)$ are perfectly correlated.

This difference is the difference between the words "correlated" and "statistically independent". Statistical independence (not defined here) and correlations are not the same thing! We will look at this difference closely in a later section.

2.5 The Covariance Matrix

If we have p data points in $\mathbb{R}^n$, we can think of the data as a $p \times n$ matrix. Let X denote the *mean-subtracted* data matrix (as we defined previously). A natural question to ask is then how the i^{th} and j^{th} dimensions (columns) covary- so we'll compute the covariance between the i, j columns to define:

$$s_{ij}^2 = \frac{1}{p-1} \sum_{k=1}^{p} X(k,i) \cdot X(k,j)$$

Computing this for all i, j will result in an $n \times n$ symmetric matrix, C, for which:

$$C_{ij} = s_{ij}^2$$

In the exercises, you'll show that an alternative way of computing the covariance matrix is by using what we'll refer to as its definition:

Definition: Let X denote a matrix of data, so that, if X is $p \times n$, then we have p data points in $\mathbb{R}^n$. Furthermore, we assume that the data in X has been mean subtracted (so the mean in $\mathbb{R}^n$ is the zero vector). Then the *covariance matrix* associated with X is given by:

$$C = \frac{1}{p-1} X^T X$$

In Matlab, it is easy to compute the covariance matrix. For your convenience, we repeat the mean-subtraction routine here:

```
%X is a pxn matrix of data:
[p,n]=size(X);
m = mean(X);
Xm = X-repmat(m,p,1);
C=(1/(p-1))*X'*X;
```

Matlab also has a built-in covariance function. It will automatically do the mean-subtraction (which is a lot of extra work if you've already done it!).

```
C=cov(X);
```

If you forget which sizes Matlab uses, you might want to just compute the covariance yourself. It assumes, as we did, that the matrix is $p \times n$, and returns an $n \times n$ covariance, and it will divide by $p - 1$ (some algorithm divide only by p).

2.6 Exercises

1. By hand, compute the mean and variance of the following set of data:

$$1, 2, 9, 6, 3, 4, 3, 8, 4, 2$$

2. Obtain a sampling of 1000 points using the uniform distribution: and 1000 points using the normal distribution:

```
x=rand(1000,1);
y=randn(1000,1);
```

Compare the distributions using Matlab's *hist* command: `hist([x y],100)` and print the results. You'll note that the histograms have not been scaled so that the areas sum to 1, but we do get an indication of the nature of the data.

3. Compute the value of K in the double Laplacian function so that f is a p.d.f.

4. Next, load a sample of human voice: `load laughter` If you type `whos`, you'll see that you have a vector y with the sound data. The computers in the lab do have sound cards, but they don't work very well with Matlab, so we won't listen to the sample. Before continuing, you might be curious about what the data in y looks like, so feel free to plot it. We want to look at the distribution of the data in the vector y, and compare it to the normal distribution. The mean of y is already approximately zero, but to get a good comparison, we'll take a normal distribution with the same variance:

```
clear
load laughter
whos
sound(y,Fs);  %This only works if there's a good sound card
s=std(y);
x=s*randn(size(y));
hist([x y],100);  %Blue is "normal", Red is Voice
```

Print the result. Note that the normal distribution is much flatter than the distribution of the voice signal.

5. Compute the covariance between the following data sets:

x	-1.0	-0.7	-0.4	-0.1	0.2	0.5	0.8
y	-1.3	-0.7	-0.1	0.5	1.1	1.7	2.3

(2.1)

6. Let $\mathbf{x}$ be a vector of data with mean μ, and let a, b be scalars. What is the mean of $a\mathbf{x}$? What is the mean of $\mathbf{x} + b$? What is the mean of $a\mathbf{x} + b$?

   NOTE: Formally, the addition of a vector and a scalar is not defined. Here, we are utilizing Matlab notation: The result of a vector plus a scalar is addition done component-wise. This is only done with scalars- for example, a matrix added to a vector is still not defined, while it is valid to add a matrix and a scalar.

7. Let $\mathbf{x}$ be a vector of data with variance σ^2, and let a, b be scalars. What is the variance of $a\mathbf{x}$? What is the variance of $\mathbf{x} + b$? What is the variance of $a\mathbf{x} + b$?

8. Show that, for data in vectors $\mathbf{x}$, $\mathbf{y}$ and a real scalar a,

$$\text{Cov}(ax, y) = a\text{Cov}(x, y) \quad \text{Cov}(x, by) = b\text{Cov}(x, y)$$

9. Show that, for data in $\mathbf{x}$ and a vector consisting only of the scalar a,

$$\text{Cov}(x, a) = 0$$

10. Show that, for a and b fixed scalars, and data in vectors $\mathbf{x}, \mathbf{y}$,

$$\text{Cov}(x + a, y + b) = \text{Cov}(x, y)$$

11. If the data sets X and Y are the same, what is the covariance? What is the correlation coefficient? What if $Y = mX$? What if $Y = mX + b$?

12. Let X be a $p \times n$ matrix of data, where we n columns of p data points (you may assume each column has zero mean). Show that the $(i, j)^{\text{th}}$ entry of $\dfrac{1}{p-1}X^T X$ is the covariance between the i^{th} and j^{th} columns of X. HINT: It might be convenient to write X in terms of its columns,

$$X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$$

    Also show that $\frac{1}{p-1}X^T X$ is a symmetric matrix.

13. This exercise shows us that our geometric insight might not extend to high dimensional space. We examine how points are distributed in high dimensional hypercubes and unit balls. Before we begin, let us agree that a hypercube of dimension n has the edges:

$$(\pm 1, \pm 1, \pm 1, \ldots, \pm 1)^T$$

    so, for example, a 2-d hypercube (a square) has edges:

$$(1, 1)^T, (-1, 1)^T, (1, -1)^T, (-1, -1)^T$$

    (a) Show that the distance (standard Euclidean) from the origin to a corner of a hypercube of dimension d is $\sqrt{d}$. What does this imply about the shape of the "cube", as $d \to \infty$?

(b) The volume of a $d-$dimensional hypersphere of radius a can be written as:

$$V_d = \frac{S_d a^d}{d}$$

where S_d is the $d-$dimensional surface area of the unit sphere.

First, compute the volume between hyperspheres of radius a and radius $a - \epsilon$.

Next, show that the ratio of this volume to the full volume is given by:

$$1 - \left(1 - \frac{\epsilon}{a}\right)^d$$

What happens as $d \to \infty$?

If we have 100,000 data points "uniformly distributed" in a hypersphere of dimension 10,000, where are "most" of the points?

2.7 Linear Regression

In this section, we examine the simplest case of fitting data to a function. We are given p pairs of data (t is for "target", we'll use y for something else):

$$(x_1, t_1), (x_2, t_2), \ldots (x_p, t_p)$$

We wish to find a line through the data. That is, we want to find scalars m, b so that

$$mx_i + b = t_i$$

for each pair (x_i, t_i). Of course, if the data actually was on a line, we would not need p points- only two are needed.

Thus, we assume that there is something going on so that the data is not exactly linear (statisticians would add ϵ_i to the end of our line to account for noise). Thus, for each point, we now have an error. We are distinguishing now between the point on the line:

$$y_i = mx_i + b$$

and the *desired* value t_i. Now the error at the ith point is defined as:

$$(t_i - y_i)^2 = (t_i - (mx_i + b))^2$$

and the overall error is the sum of squares error (summed over the p points):

$$E(m, b) = \sum_{k=1}^{p} (t_k - (mx_k + b))^2$$

We have now translated our problem into a Calculus problem- Find the minimum of $E(m, b)$. Here are some exercises to lead you to the solution:

Exercises with the Regression Error

1. E is a function of m and b, so the minimum value occurs where

$$\frac{\partial E}{\partial m} = 0 \qquad \frac{\partial E}{\partial b} = 0$$

Show that this leads to the system of equations: (the summation index is 1 to p)

$$\begin{aligned} m\sum x_k^2 + b\sum x_k &= \sum x_k t_k \\ m\sum x_k + bn &= \sum t_k \end{aligned}$$

2. Using linear algebra, given the data, then in finding the line of best fit, we are trying to solve the system of equations below, where we then write the system in matrix-vector form:

$$\begin{array}{rl} mx_1 + b &= t_1 \\ mx_2 + b &= t_2 \\ mx_3 + b &= t_3 \\ \vdots & \vdots \\ mx_p + b &= t_p \end{array} \quad \Rightarrow \quad \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ \vdots & \vdots \\ x_p & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_p \end{bmatrix} \quad \Rightarrow \quad A\mathbf{c} = \mathbf{t}$$

As we said before, the data does not lie exactly on a line (otherwise we would only need two points). That means that this system of equations *has no solution*. However, if we think of varying the vector of unknowns $\boldsymbol{c}$, we might define the model output as $\mathbf{y}$:

$$\mathbf{y} = A\mathbf{c}$$

Now we can define the error as:

$$E(\mathbf{c}) = \|\mathbf{t} - \mathbf{y}\|^2 = \|\mathbf{t} - A\mathbf{c}\|^2$$

and now we will find $\mathbf{c}$ that minimizes this error- In linear algebra, this is known as the *least squares solution* to the equation $A\mathbf{c} = \mathbf{t}$. We will revisit this again later using projections, but for now, we can solve this problem using the **normal equations**. That is, we will multiply both sides of our equation by A^T to get:

$$A\mathbf{c} = \mathbf{t} \quad \Rightarrow \quad A^T A\mathbf{c} = A^T\mathbf{t}$$

Originally, A was $p \times 2$, so now $A^T A$ is 2×2, which we can invert. EXERCISE: Show that this system of two equations in two variables is the same as the system we obtained by setting the partial derivatives to zero.

3. Consider the following data set [11] which relates the index of exposure to radioactive contamination from Hanford to the number of cancer deaths per $100,000$ residents. We would

like to get a relationship between these data.

County/City	Index	Deaths
Umatilla	2.5	147
Morrow	2.6	130
Gilliam	3.4	130
Sherman	1.3	114
Wasco	1.6	138
Hood River	3.8	162
Portland	11.6	208
Columbia	6.4	178
Clatsop	8.3	210