Chapter 7

A Best Nonorthogonal Basis

In this section, we examine a particular question whose solution will involve getting an optimal *nonorthogonal* basis, which is quite contrary to our earlier chapters- in fact, the reader should ask why we would ever want to use a nonorthogonal basis when it would be quite easy (using Gram-Schmidt) to construct an orthogonal version of the same basis.

To answer this question, consider the synthetic data example in Figure 7.1. Here there is a definite "natural" basis appearing in the data- and the basis vectors are not orthogonal. While the data was synthetic, we do get similar types of data appearing in the problem of *Blind Signal Separation*. Consider the following tasks:

1. We have a patient that is pregnant. Our overall goal is to listen to the fetus heartbeat, but when we try, the sound of the mother's heartbeat is mixed with the heartbeat of the fetus. Symmetrically, if we were to try to listen to the heartbeat of the mother, we would also hear the heartbeat of the fetus. Is it possible to gather these sounds on microphones and manipulate the data so that the mother's (or fetus) heartbeat has been isolated?

2. We have two microphones placed at random, but distinct, places in a room. We also have two people speaking in the room (the placement of the people is distinct from the placement of the microphones- we do not assume that each microphone is placed in front of each speaker). Is it possible to manipulate the two mixtures of voices so that we can isolate each speaker's voice?

The answer lies in a fairly new technique called *Independent Component Analysis* (ICA) (versus what we studied earlier, *principal components analysis*, or PCA). In ICA, we assume that we have some underlying, statistically inde-

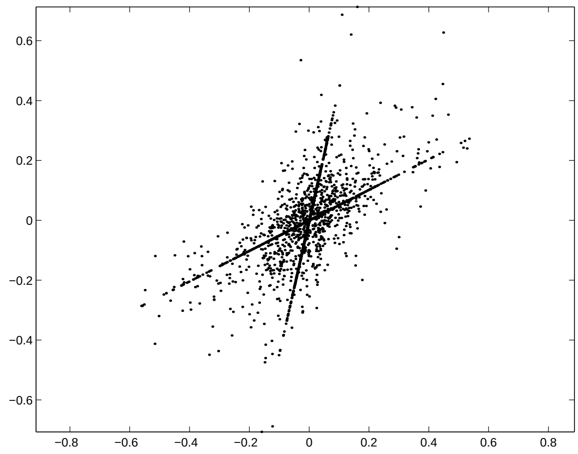

Figure 7.1: A synthetic data example of a naturally emerging set of basis vectors from data- These are not orthogonal.

pendent, processes and that we are observing mixtures of these processes. Our goal is to separate the mixtures.

This problem is also known as *Blind Signal Separation*, where we assume some unknown mixture of signals, and we attempt at separating them.

This process (the problem and the solution) can also be framed in other terms- we will focus on the geometric meaning of the problem, and will solve it using the techniques of linear algebra.

7.1 Set up the Signal Separation Problem

We will assume that there exists a "clean" separation of our observed mixture of two signals (we will be explicit in what we mean by that momentarily, and we will discuss the more general case in a moment). These signals, as time series, are two columns of a matrix S, so that $S \in \mathbb{R}^{p\times 2}$, where p is the length of the sample.

We will further assume that the mixtures we are observing are *linear* mixtures, so that the mixtures we observe may be modeled as:

$$\boldsymbol{x}_1 = \alpha_1 \boldsymbol{s}_1 + \alpha_2 \boldsymbol{s}_2$$

$$\boldsymbol{x}_2 = \alpha_3 \boldsymbol{s}_1 + \alpha_4 \boldsymbol{s}_2$$

so that $\boldsymbol{x}_1$ is the observed mixture in microphone 1, and $\boldsymbol{x}_2$ is the observed mixture in microphone 2. In linear algebra terms, we can state the problem as follows:

Given $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ as columns of $X \in \mathbb{R}^{p\times 2}$, solve the following equation for

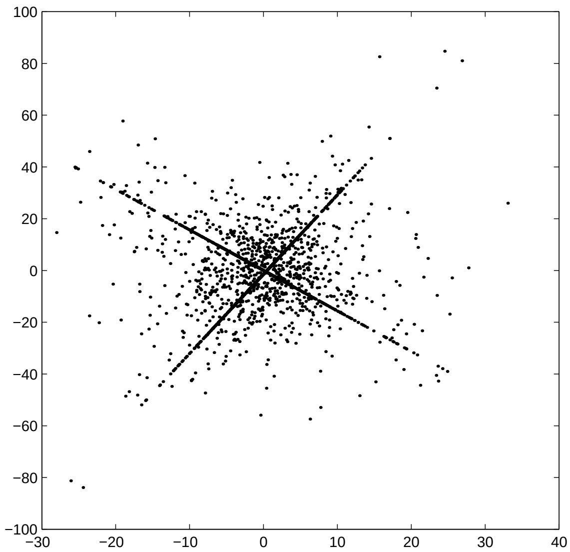

Figure 7.2: The synthetic data set after the SVD transformation as described in the text. In this case, the principal components analysis has left the desired basis vectors in a rotated position. We require one more rotation (multiplication by an orthogonal matrix) to get the desirable results.

$A \in \mathbb{R}^{2\times 2}$ and $S \in \mathbb{R}^{p\times 2}$:

$$X = SA$$

We assume that the rows of X have been mean-subtracted.

If you look at this equation, something should be occurring to you- this is not a well defined problem! There are an infinite number of solutions for A, S. In fact, one solution would be to let A be the 2×2 identity matrix, and $S = X$.

We could also give a solution in terms of the SVD of X, which is what we would do in Principal Component Analysis:

$$X = U_x \Sigma_x V_x^T$$

so that $S = U_x$, and $A = \Sigma_x V_x^T$. In this case the data is "separated" in the sense that the columns of U_x are orthogonal (or *uncorrelated*). Figure 7.2 shows the result of this operation on our synthetic data. It also shows that the desired basis vectors are still rotated.

This process, while not the desired one, is a good first step, but somehow we need the signals to be *independent*, and not just uncorrelated.

Alternatively, let us consider the SVD of the unknown mixing matrix A, $A = U_m \Sigma_m V_m^T$. The problem will be solved if we knew this matrix, as S could be computed by using the inverse of A (we assume that A is full rank- see the exercises for a discussion). Let's try some sample computations now by taking our original equation and substituting the SVD of A:

$$X = SA \Rightarrow X = SU_m\Sigma_m V_m^T$$

Now compute the 2×2 covariance of X:

$$X^TX = V_m\Sigma_m U_m^T S^T S U_m \Sigma_m V_m^T$$

Now, if the rows of S are statistically independent, then they certainly should be uncorrelated. We will assume therefore that SS^T is some scalar multiple of the identity:

$$SS^T = cI_{2\times 2}$$

which also assumes that the variances of the signals are the same. In particular, we'll assume that the signals in S have been scaled so that $S^TS = I$. In the exercises, we will examine this assumption in more detail.

Using this, we see that:

$$X^TX = V_m\Sigma_m^2 V_m^T = V_x\Sigma_x V_x^T$$

This tells us that V_m and Σ_m are recoverable from the SVD of X: If

$$X = U_x\Sigma_x V_x^T$$

then

$$\Sigma_m = \Sigma_x^{1/2}, \qquad V_m = V_x$$

If we were to stop here and take:

$$Y = XV_x\Sigma_x^{-1/2} = SU_m\Sigma_m V_m^T V_x \Sigma_x^{-1/2} = SU_m$$

we obtain the standard PCA solution. However, the signals are still rotated. We cannot perform another covariance computation on Y, since now:

$$Y^TY = U_m^T S^T S U_m = I_{2\times 2}$$

How can we compute U_m? There is some justification for what we're about to do- Let's do it first and then discuss it.

Define dA to be the difference matrix for the matrix A. That is, if A has been organized so that it comprises p samples of k time series of data, then let A be $p \times k$. We compute the difference as:

$$dA = A(2:p,:) - A(1:p-1,:)$$

so that dA is now $p-1 \times k$ and

$$(dA)_{ij} = A_{i+1,j} - A_{i,j}$$

If the data in A were the sample of some differentiable function, then dA is an approximation to the derivative using $\Delta t = 1$.

Note that the following matrix equation holds:

$$X = SA \quad \Rightarrow \quad dX = dS\,A$$

so that

$$dX^T dX = A^T\, dS^T dS\, A$$

More particularly, let

$$Y = XV_x\Sigma_x^{-1/2}, \text{ or } \; dY = dXV_x\Sigma_x^{-1/2} = dS\,U_m$$

so that $dY^T dY = U_m^T dS^T dS\, U_m$. We now make our second assumption: While $S^T S = I$, we assume that $dS^T dS \neq I$, but is diagonal. In this case, we can recover U_m from the SVD of dY:

$$dY = U_y \Sigma_y V_y^T$$

and $U_m = V_y^T$. Note that the singular vectors are transposed when making this computation!

This finishes our problem, since we have now computed the SVD of the mixing matrix A. The clean signal is now found by taking:

$$S = YV_y$$

We are approximating the mixing matrix by:

$$A \approx V_y^T \Sigma_x^{1/2} V_x = U_m \Sigma_m V_m^T$$

so that the approximate inverse is found via:

$$A^{-1} = V_m \Sigma_m^{-1} U_m^T = V_x^T \Sigma_x^{-1/2} V_y$$

This process of two SVDs, one on the matrix of data X, and another on the data dY can be brought together as a single command. In fact, this process is equivalent to using the *Generalized Singular Value Decomposition*, which will sometimes go under the name of *Quotient Singular Value Decomposition* (GSVD or QSVD, respectively). This simplifies the coding so that you only have to use the following Matlab commands. Let X be the $p \times k$ matrix of k mixtures of signals (this is transposed from our earlier notation). Then signal separation is simply:

```
dX=diff(X);
[U,V,B,C,S]=gsvd(X,dX,0);
```

where the clean mixtures are in the columns of U.

We'll try out both versions in the examples below, then in the next section we'll define the GSVD.

Example

In this example, we will take three columns of data. The first two columns will comprise a circle and the last will be white noise (that is, random data from a normal distribution). We will then multiply this by 3×3 mixing matrix A, which was originally taken so that the elements are from a normal random variable then subsequently hard coded for you to replicate. Denote the mixed data matrix as X, as we did previously. Note that with column-wise data, the mixing matrix equations become:

$$X = SA \Rightarrow X = SU\Sigma V^T$$

where we try to determine U, Σ, V.

Here is the script file to produce the signal separation:

```
%Script file to produce Example 1, ICA
numpts=400;
t=linspace(0,3*pi,numpts);
S=[cos(t'), sin(t'), randn(numpts,1)]; %Separated Signals
A =[  -0.0964   -0.1680    1.6777
    0.4458    0.1795    1.9969
   -0.2958    0.4211    0.6970]; %Mixing Matrix
X=S*A;

dX=diff(X);
[U,V,B,C,S] = gsvd(X,dX,0); %Clean Signal in U

%The double SVD code, equivalent to the GSVD:
[Ux,Sx,Vx]=svd(X,0);
Y=X*Vx*diag(1./sqrt(diag(Sx)));  %Stopping here is basic PCA
dY=diff(Y);
[Uy,Sy,Vy]=svd(dY,0);
S2=Y*Vy;  %S2 is also the clean signal


%Plotting routines below:
figure(1)
for j=1:3
   subplot(3,1,j)
   plot(U(:,j));  %Clean Signals from GSVD
end
figure(2)
for j=1:3
   subplot(3,1,j);
   plot(S2(:,j));  %Clean Signals from double SVD
end
figure(3)
for j=1:3
   subplot(3,1,j)
```

```
   plot(Y(:,j));  %Results of PCA (or KL)
end
```

7.2 Signal Separation of Voice Data

In this example, we will linearly mix two voice signals, then separate them using the techniques we previously described. This example is best done using a computer with a good sound card, but can be done without it.

Matlab comes with several sound files. For this example, we will use `handel` (a sample of the chorus to Handel's "Messiah"), and `laughter` (a sample of people laughing). The two files have different lengths, so we'll have to cut the longer file off so that they match.

Here is the Matlab code:

```
%Script for sound files
load handel
y1=y;
load laughter
S=[y y1(1:52634)];  %Clean samples in the columns of S
A =[ -0.5883   -0.1364; 2.1832    0.1139]; %Mixing Matrix
X=S*A;

mX=mean(X);
X=X-repmat(mX,52634,1);

figure(1)
plot(X(:,1),X(:,2),'.');
title('Mixed Signals');

%For comparison purposes, here's the SVD
[Ux,Sx,Vx]=svd(X,0);
Y=X*Vx*(1./sqrt(diag(Sx)));

figure(2)
plot(Y(:,1),Y(:,2),'.');
title('KL Results')
%Listen to the results:  You'll still hear mixtures
%soundsc(Y(:,1));
%soundsc(Y(:,2));

dX=diff(X);
[Y2,V,B,C,S3]=gsvd(X,dX,0);
figure(3)
plot(Y2(:,1),Y2(:,2),'.');
title('ICA Results');
%Listen to the results:  They will be clean!
```

```
%soundsc(Y2(:,1));
%soundsc(Y2(:,2));
```

More to Think About: There are a lot of experiments you can try with this data. Here are some things you might try:

- Plot the signals as time series if you've never looked at voice data before. Also plot the differenced signal. You might also look at the histograms of the two voice signals using `hist`. Do the signals look like they are normally distributed or do they follow a Laplace distribution?
- Change the mixing matrix to a random matrix. Will you always get good results?
- Listen to the difference matrix dX. Does it still sound like the original? Listen to the second, third, fourth difference. Why does the "derivative" of the signal sound just like the original (perhaps with a different timbre quality, but recognizable just the same)?
- Check the assumptions on the clean signal S and the differenced signal dS- Are the assumptions met?

7.3 A Closer Look at the GSVD

Suppose that we have two matrices $X \in \mathbb{R}^{m\times n}$ and $Z \in \mathbb{R}^{p\times n}$. The GSVD of matrices X, Z is a decomposition where we determine $\hat{U}, \hat{V}, W, C, S$ so that:

$$X = \hat{U}CW^T \qquad Z = \hat{V}SW^T$$

where $\hat{U}, \hat{V}$ have orthonormal columns, C, S are diagonal matrices such that $C^TC + S^TS = I$, and W is an invertible matrix. In Matlab, the command is:

```
[Ux,Vz,W,C,S]=gsvd(X,Z)
```

The values of C, S and W satisfy the following generalized eigenvalue problem:

$$s_i^2 A^T A w_i = c_i^2 B^T B w_i$$

The solution we use for the signal separation is now either:

$$X = \hat{U}CW^T = \hat{U}(CW^T) = SA \text{ or } X = \hat{U}CW^T = (\hat{U}C)W^T = SA$$

In the special case that $s_i \neq 0$, we'll show that we can find x_i using two regular SVD's as we did in the signal separation.

In this case, the eigenvector problem can be written as:

$$X^TXw_i = \lambda Z^T Z w_i \Rightarrow \left(X^TX - \lambda Z^TZ\right) w_i = 0$$

If we let $X = U\Sigma_x V_x^T$ be the SVD of X, then we can rewrite this as:

$$\left(V_x\Sigma_x^2 V_x^T - \lambda Z^TZ\right) x_i = 0$$

This can be factored as:

$$\left[V_x\Sigma_x\left(I - \lambda(\Sigma_x^{-1}V_x^T)Z^TZ(V_x\Sigma_x^{-1})\right)\Sigma_xV_x^T\right]w_i = 0$$

or equivalently, if V_x is square and Σ_x is invertible:

$$\left(I - \lambda(\Sigma_x^{-1}V_x^T)Z^TZ(V_x\Sigma_x^{-1})\right)\Sigma_xV_x^Tw_i = 0$$

If we let $Y = ZV_x\Sigma_x^{-1}$, and $q_i = \Sigma_xV_x^Tw_i$, the previous equation can be written as:

$$(I - \lambda Y^TY)q_i = 0$$

Therefore, q_i is an eigenvector of Y^TY, or a right singular vector of Y. We can compute $w_i = V_x\Sigma_x^{-1}q_i$, or

$$W = V_x\Sigma_x^{-1}Q = V_x\Sigma_x^{-1}V_y$$

To summarize, the GSVD is equivalent to two SVDs as follows:

- Let $X = U\Sigma_xV_x^T$ be the SVD of A
- Let $Y = ZV_x\Sigma_x^{-1}$ be a "whitening" transformation.
- Let $Y = U_y\Sigma_yV_y^T$ be the second SVD.
- Final answer: $W = V_x\Sigma_x^{-1}V_y$

To connect this process to our previous signal separation solution, let's recall what we did there: Let X be the mixed signal, dX be the differenced signal (we are now thinking of $Z = dX$):

- Let $X = U_x\Sigma_xV_x^T$ be the SVD of X.
- Let $dY = dXV_x\Sigma_x^{-1/2}$ be the "whitened" signal.
- Let $dY = U_y\Sigma_yV_y^T$ be the second SVD.
- Then $S = YV_y = XV_x\Sigma_x^{-1/2}V_y = XW$ is the clean signal.