

Linear Algebra
Fall 2002

SVD and Cryptograms

by Tim Honn & Seth Stone

College of the Redwoods
Eureka, CA
Math dept.

email: timhonn@cox.net

email: lamentofseth@hotmail.com



1/28



Introduction

Cryptology is the study of the processes used to encode and decode messages for the purpose keeping the content of the messages secret. Ideas developed in Linear Algebra can provide techniques to aid in the breaking of these codes.

Of course there are many ways to encode a particular piece of writing, each with it's own level of complexity. One of the most basic methods of encoding is the simple substitution cipher which we will be discussing here.



Methods of Cryptology

When employing the method of a substitution cipher we simply rearrange the order of the alphabet and map the letters of a message to the letter found in the corresponding position of the newly ordered alphabet. For example, we use a simple reversed alphabet here where a is mapped to z . As depicted below

$$a \rightarrow z, b \rightarrow y, \dots, z \rightarrow a$$

[abcdefghijklmnopqrstuvwxyz]
[zyxwvutsrqponmlkjihgfedcba]





Then through the use of the permuted alphabet we can encode a simple message,

see spot run

as,

hvv hklg ifm

The recipient of the message has only the simple task of re-mapping the letters to decode the secret message.





The Digram Frequency Matrix

The digram Frequency Matrix is the $n \times n$ array A where a_{ij} is the number of occurrences of the i th letter followed by the j th letter. For a simple example we use restricted alphabet consisting of only

[a b c d e]

To demonstrate we use this short text

aabcd ddab ddace addeca babcbdeba abcdba ebad

to obtain the digram matrix

$$A = \begin{matrix} & \begin{matrix} a & b & c & d & e \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix} & \begin{pmatrix} 2 & 5 & 1 & 2 & 1 \\ 4 & 0 & 3 & 2 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 3 & 1 & 0 & 4 & 2 \\ 1 & 2 & 1 & 0 & 0 \end{pmatrix} \end{matrix}$$



aabcd ddab ddace addeca babcbdeba abcdba ebad

$$A = \begin{matrix} & a & b & c & d & e \\ a & \left(\begin{array}{c} 2 \\ 4 \\ 1 \\ 3 \\ 1 \end{array} \right. & \begin{array}{c} 5 \\ 0 \\ 1 \\ 1 \\ 2 \end{array} & \begin{array}{c} 1 \\ 3 \\ 0 \\ 0 \\ 1 \end{array} & \begin{array}{c} 2 \\ 2 \\ 2 \\ 4 \\ 0 \end{array} & \begin{array}{c} 1 \\ 0 \\ 1 \\ 2 \\ 0 \end{array} \end{matrix}$$

Notice that the a_{13} entry is 1, the number of the occurrences of a followed by c and the a_{14} entry is 2, the number of occurrences of a followed by d .





And of course this idea generalizes to larger texts using the complete alphabet.

Four score and seven years ago our fathers brought forth on this continent, a new nation, conceived in Liberty, and dedicated to the proposition that all men are created equal...

.
. .
.

and that government of the people, by the people, for the people, shall not perish from the earth.

yields the digram matrix below,





0	1	2	5	0	1	4	0	2	0	1	9	0	15	0	1	0	10	5	36	1	8	0	0	1	0
1	0	0	0	5	0	0	0	1	0	0	1	0	0	1	0	0	2	0	0	2	0	0	0	1	0
12	0	0	0	4	0	0	2	1	0	0	0	0	0	7	0	0	4	0	1	0	0	0	0	0	0
2	0	1	6	4	0	0	3	13	0	0	0	0	0	4	1	0	0	4	1	1	4	0	0	0	0
16	3	8	26	3	5	2	7	6	0	0	4	5	10	5	4	1	22	9	12	2	4	8	0	3	0
3	0	0	1	0	1	0	0	5	0	0	0	0	0	10	0	0	3	0	3	1	0	0	0	0	0
5	1	0	0	5	0	1	4	0	0	0	1	0	0	3	1	0	6	0	0	0	0	1	0	0	0
24	0	0	0	32	1	0	0	7	0	0	1	0	0	8	0	0	0	0	5	1	0	0	0	0	0
0	1	8	1	3	0	2	0	0	0	0	2	0	16	9	0	0	2	9	8	0	7	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
3	0	0	4	6	0	0	1	6	0	0	8	2	3	3	0	0	1	0	1	0	1	1	0	2	0
2	1	0	0	7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
10	0	5	9	4	1	9	0	2	0	0	3	2	4	12	0	0	0	4	8	1	1	0	0	2	0
1	3	1	3	0	6	1	1	0	0	0	1	4	20	2	5	0	17	3	13	7	2	3	0	0	0
0	0	0	0	5	0	0	0	0	0	0	4	0	0	4	0	0	2	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
8	0	0	1	26	4	3	0	1	0	1	3	0	1	6	2	0	0	5	12	3	0	3	0	0	0
4	2	2	0	10	2	1	6	1	0	1	0	0	1	4	0	0	1	0	8	1	0	0	0	0	0
4	1	4	1	11	5	1	47	18	0	0	3	0	2	11	1	0	2	0	9	0	0	5	0	1	0
1	0	0	0	0	0	3	0	0	0	0	2	0	3	0	0	0	5	5	2	0	0	0	0	0	0
2	0	0	0	17	0	0	0	3	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	11	0	0	8	1	0	0	0	0	1	2	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



The Digram Function

The digram frequency matrix above was produced via this function.

```
function A=digram(filename,n) A=zeros(n);
longline=''; fid=fopen(filename,'rt'); while(~feof(fid))
    line=fgetl(fid);
    line=upper(line);
    k=isletter(line);
    line=line(k);
    longline=strcat(longline,line);
end longline=double(longline)-64;
for j=1:length(longline)-1
    A(longline(j),longline(j+1))=...
        A(longline(j),longline(j+1))+1;
end
fclose(fid)
```



The Digram Frequency Matrix

But for the time being let's return to our more manageable example.

The sum of each row is found by Ae , where $e = (1, 1, 1, 1, 1)^T$

$$Ae = \begin{pmatrix} 2 & 5 & 1 & 2 & 1 \\ 4 & 0 & 3 & 2 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 3 & 1 & 0 & 4 & 2 \\ 1 & 2 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 11 \\ 9 \\ 5 \\ 5 \\ 4 \end{pmatrix} = f$$

Multiplying on the right by e sums the entries of each row of A , giving f . Note that the first entry in f is 11, the total number of occurrences where a is followed by another letter and thus is the total number of a 's in the text.

Similarly, $A^T e$ sums the entries of each column of A , giving the same frequency vector f .





The Singular Value Decomposition

When faced with the problem of decoding a cipher, often the cryptanalyst's first approach is to try a comparison of the frequency of the encoded letters with the known frequencies of typical un-coded text. A singular value decomposition of the frequency matrix A will prove useful in this pursuit.

For some $n \times n$ matrix A , the singular value decomposition is,

$$A = X\Sigma Y^T,$$

where X is an $n \times n$ matrix whose columns are the left singular vectors, Y is an $n \times n$ matrix whose columns are the right singular vectors, and Σ is a diagonal $n \times n$ matrix whose entries are the singular values.





An expansion gives the following,

$$\begin{aligned} A &= X\Sigma Y^T \\ &= \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \dots & \\ & & & \sigma_n \end{pmatrix} \begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_n^T \end{pmatrix} \\ &= \sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T + \dots + \sigma_n x_n y_n^T \end{aligned}$$

The digram frequency matrix A equals the finite series above. The first term of the series,

$$\sigma_1 x_1 y_1^T,$$

is called the *rank one approximation*. If σ_1 is significantly larger than the remaining singular values, then the rank one approximations closely resembles A .





Rank One Approximation

Via the rank one approximation, we can obtain some useful information about the digram frequency matrix A . Since

$$Ae = A^T e = f,$$

we can substitute $A \approx \sigma_1 x_1 y_1^T$ and write

$$(\sigma_1 x_1 y_1^T) e = (\sigma_1 x_1 y_1^T)^T e = f$$

$$(\sigma_1 x_1 y_1^T) e = (\sigma_1 y_1 x_1^T) e = f$$

Reordering,

$$(\sigma_1 y_1^T e) x_1 = (\sigma_1 x_1^T e) y_1 = f.$$

In the last equation the left and right singular vectors are simply being multiplied by the scalars $\sigma_1 y_1^T e$ and $\sigma_1 x_1^T e$, so x_1 and y_1 are proportional to f .

Now, let's compare the first left and right singular vectors of the Gettysburg Address digram frequency matrix to f .



$$\begin{array}{l}
 x_1 = \begin{pmatrix} -0.3279 \\ -0.0471 \\ -0.1201 \\ -0.2012 \\ -0.4397 \\ -0.0875 \\ -0.0966 \\ -0.3468 \\ -0.1832 \\ 0.0000 \\ -0.0099 \\ -0.1204 \\ -0.0607 \\ -0.2166 \\ -0.2387 \\ -0.0523 \\ -0.0007 \\ -0.2954 \\ -0.1683 \\ -0.4455 \\ -0.0533 \\ -0.1167 \\ -0.1211 \\ 0.0000 \\ -0.0340 \\ 0.0000 \end{pmatrix}
 \end{array}$$

$$\begin{array}{l}
 y_1 = \begin{pmatrix} -0.3219 \\ -0.0442 \\ -0.1136 \\ -0.2261 \\ -0.4515 \\ -0.1023 \\ -0.0800 \\ -0.3381 \\ -0.2340 \\ -0.0000 \\ -0.0097 \\ -0.1219 \\ -0.0468 \\ -0.2438 \\ -0.2564 \\ -0.0565 \\ -0.0054 \\ -0.2496 \\ -0.1393 \\ -0.4371 \\ -0.0597 \\ -0.0818 \\ -0.1045 \\ 0.0000 \\ -0.0344 \\ 0.0000 \end{pmatrix}
 \end{array}$$

$$\begin{array}{l}
 f = \begin{pmatrix} 102.0000 \\ 14.0000 \\ 31.0000 \\ 58.0000 \\ 165.0000 \\ 26.0000 \\ 28.0000 \\ 80.0000 \\ 68.0000 \\ 0.0000 \\ 3.0000 \\ 42.0000 \\ 13.0000 \\ 77.0000 \\ 93.0000 \\ 15.0000 \\ 1.0000 \\ 79.0000 \\ 44.0000 \\ 126.0000 \\ 21.0000 \\ 24.0000 \\ 28.0000 \\ 0.0000 \\ 10.0000 \\ 0.0000 \end{pmatrix}
 \end{array}$$





At first glance not very impressive. While x_1 and y_1 show a fair amount of correlation, f appears to have nothing to do with x_1 and y_1 . But if we look at the frequencies of these entries the correlation is remarkable.

$$x'_1 = \frac{x_1}{\sum x_1}$$

$$y'_1 = \frac{y_1}{\sum y_1}$$

$$f' = \frac{f}{\sum f}$$



$$\begin{array}{r}
 x'_1 = \begin{pmatrix} 0.0867 \\ 0.0124 \\ 0.0317 \\ 0.0532 \\ 0.1162 \\ 0.0231 \\ 0.0255 \\ 0.0917 \\ 0.0484 \\ 0.0000 \\ 0.0026 \\ 0.0318 \\ 0.0160 \\ 0.0572 \\ 0.0631 \\ 0.0138 \\ 0.0002 \\ 0.0781 \\ 0.0445 \\ 0.1177 \\ 0.0141 \\ 0.0308 \\ 0.0320 \\ 0.0000 \\ 0.0090 \\ 0.0000 \end{pmatrix}
 \end{array}$$

$$\begin{array}{r}
 y'_1 = \begin{pmatrix} 0.0856 \\ 0.0117 \\ 0.0302 \\ 0.0602 \\ 0.1201 \\ 0.0272 \\ 0.0213 \\ 0.0900 \\ 0.0622 \\ 0.0000 \\ 0.0026 \\ 0.0324 \\ 0.0125 \\ 0.0649 \\ 0.0682 \\ 0.0150 \\ 0.0014 \\ 0.0664 \\ 0.0371 \\ 0.1163 \\ 0.0159 \\ 0.0218 \\ 0.0278 \\ 0.0000 \\ 0.0092 \\ 0.0000 \end{pmatrix}
 \end{array}$$

$$\begin{array}{r}
 f' = \begin{pmatrix} 0.0889 \\ 0.0122 \\ 0.0270 \\ 0.0505 \\ 0.1437 \\ 0.0226 \\ 0.0244 \\ 0.0697 \\ 0.0592 \\ 0.0000 \\ 0.0026 \\ 0.0366 \\ 0.0113 \\ 0.0671 \\ 0.0810 \\ 0.0131 \\ 0.0009 \\ 0.0688 \\ 0.0383 \\ 0.1098 \\ 0.0183 \\ 0.0209 \\ 0.0244 \\ 0.0000 \\ 0.0087 \\ 0.0000 \end{pmatrix}
 \end{array}$$





Rank Two Approximation

Recall that,

$$\begin{aligned} A &= X\Sigma Y^T \\ &= \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \dots & \\ & & & \sigma_n \end{pmatrix} \begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_n^T \end{pmatrix} \\ &= \sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T + \dots + \sigma_n x_n y_n^T \end{aligned}$$

A rank two approximation of A is obtained by keeping only the first two terms of the series,

$$\sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T$$

which is an even better approximation of A than the rank one approximation and is integral to our second method.





Before we can proceed we need to make this transition into Linear Algebra complete by thinking of the alphabet as two vectors, v and c .

As before, we return to our simplified example.

$$c = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix} \qquad v = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix}$$

With these vectors we can mathematically express some important properties of any text.





For example,

$v^T Av$ is the number of instances where a vowel is followed by a vowel.

$$\begin{aligned} v^T Av &= (1 \ 0 \ 0 \ 0 \ 1) \begin{pmatrix} 2 & 5 & 1 & 2 & 1 \\ 4 & 0 & 3 & 2 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 3 & 1 & 0 & 4 & 2 \\ 1 & 2 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\ &= (3 \ 7 \ 2 \ 2 \ 1) \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\ &= 4 \end{aligned}$$

aabcd ddab ddace addeca babcbdeba abcdba ebad



And in a similar fashion it can be shown that:

- $c^T Av$ is the number of instances where a consonant is followed by a vowel.

aabcd ddab ddace addeca babcbdeba abcdba ebad

- $v^T A(v + c)$ is the total number of vowels.

aabcd ddab ddace addeca babcbdeba abcdba ebad

- $c^T A(v + c)$ is the total number of consonants.

aabcd ddab ddace addeca babcbdeba abcdba ebad





The vfc rule and partitioning

It is a characteristic of many languages that their texts follow a simple rule called the vfc rule. The vfc rule says that consonants are followed by vowels more often than vowels are followed by vowels.

$$\frac{\text{number of vowel-vowel pairs}}{\text{number of vowels}} < \frac{\text{number of consonant-vowel pairs}}{\text{number of consonants}}$$

Some languages adhere more strictly to the vfc rule than others. For instance, Hawaiian texts are completely vfc: every consonant is followed by a vowel.

Although in English text vowels do occasionally follow vowels, English is still a predominantly vfc language. We will use this fact in the next procedure.





Using the vfc Rule

Another approach to deciphering an encoded message is to attempt a partitioning of the encoded alphabet into what we think are the vowel and consonant categories.

Now that we have developed the symbolism for the number of consonants, vowels, and pairs, we use the vfc rule to test the accuracy of our partitioning attempt. If our partition is correct the following inequality will be true.

$$\frac{\text{number of vowel-vowel pairs}}{\text{number of vowels}} < \frac{\text{number of consonant-vowel pairs}}{\text{number of consonants}}$$

and symbolically,

$$\frac{v^T A v}{v^T A (v + c)} < \frac{c^T A v}{c^T A (v + c)}$$





Using the common denominator we can simplify to,

$$(v^T Av)(c^T Ac) - (v^T Ac)(c^T Av) < 0.$$

It has been deemed “the cryptanalyst’s problem” to find a partitioning of the encoded alphabet such that the inequality above will hold.



Returning to the rank two approximation, we propose that we can use the signs of the components of x_2 and y_2 to partition the alphabet into v , c , and n vectors by,

$$c_i = \begin{cases} 1, & \text{if } x_{i2} > 0 \text{ and } y_{i2} < 0, \\ 0, & \text{otherwise} \end{cases}$$

$$v_i = \begin{cases} 1, & \text{if } x_{i2} < 0 \text{ and } y_{i2} > 0, \\ 0, & \text{otherwise} \end{cases}$$

$$n_i = \begin{cases} 1, & \text{if sign } x_{i2} = \text{sign } y_{i2}, \\ 0, & \text{otherwise} \end{cases}$$

where n is the vector of letters that we cannot categorize in either the v or c vectors by this method. When applied to the Gettysburg Address the rank two approximation yields the following partitioning,



	v	c	n
a	1	0	0
b	0	1	0
c	0	1	0
d	0	1	0
e	1	0	0
f	0	0	1
g	0	1	0
h	0	0	1
i	1	0	0
j	0	1	0
k	0	1	0
l	0	1	0
m	0	1	0
n	0	0	1
o	1	0	0
p	0	1	0
q	0	0	1
r	0	1	0
s	0	1	0
t	0	1	0
u	0	0	1
v	0	1	0
w	0	1	0
x	0	0	0
y	0	0	1
z	0	0	0



The reason why this partitioning scheme works is because we are using the rank two approximation, $A \approx \sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T$.

Recall the final form of the vfc equation,

$$D = (v^T A v)(c^T A c) - (v^T A c)(c^T A v).$$

Substituting this approximation in for A gives,

$$D = v^T (\sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T) v c^T (\sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T) c \\ - v^T (\sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T) c c^T (\sigma_1 x_1 y_1^T + \sigma_2 x_2 y_2^T) v.$$

After some messy algebra, of which we will spare you the agony, four of the eight terms cancel and we are left with,

$$D = \sigma_1 \sigma_2 [(v^T x_1)(y_1^T v)(c^T x_2)(y_2^T c) + (v^T x_2)(y_2^T v)(c^T x_1)(y_1^T c) \\ - (v^T x_1)(y_1^T c)(c^T x_2)(y_2^T v) - (v^T x_2)(y_2^T c)(c^T x_1)(y_1^T v)].$$

Though we have four terms remaining, it is not hard to show that all are negative, and thus D is negative, satisfying the vfc rule. Notice that each term is grouped into four factors, a v or a c times an x_j or a y_j .





First, we know that the c and v are vectors of ones and zeros. Because all the terms in our first right and left singular vectors, x_1 and y_1 , are negative, any dot product in which these appear must be negative. This takes care of twelve of the factors. Each of the remaining four is either a v times an x_2 or a c times a y_2 . Both of these dot products will produce negative answers because of the definition of v and c .

$$v = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ \vdots \end{pmatrix} \quad x_2 = \begin{pmatrix} -0.5090 \\ 0.0413 \\ 0.0722 \\ 0.1439 \\ -0.3304 \\ \vdots \end{pmatrix} \quad c = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \quad y_1 = \begin{pmatrix} 0.1582 \\ -0.0386 \\ -0.0787 \\ -0.2161 \\ 0.5316 \\ \vdots \end{pmatrix}$$





Therefore, a cryptanalyst's methodology when confronted with a simple substitution cipher might go something like this:

- A text is represented by a frequency digram matrix.
- Perform a singular value decomposition.
- Use the rank one approximation to analyze the frequency of occurrence of each letter.
- Use the rank two approximation to partition the encoded alphabet into vowel and consonant categories.

Applying this method to the Gettysburg Address we find that eighteen out of the twenty-six letters are accurately assigned. In conjunction with the frequency of occurrence analysis via the rank one approximation, it is plain to see that the cryptanalyst's job will prove considerably simpler with the help of the singular value decomposition.

