SVD and Cryptograms

Tim Honn and Seth Stone

December 13, 2002

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. To encode a message we map the letters of the un-coded 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, etc..

$$a \to z,\ b \to y,\ \ldots,\ z \to a$$

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

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 ith letter followed by the jth letter. For demonstration purposes we will use a restricted alphabet consisting of only,

[a b c d e]

and this short text of gibberish:

aabcd ddab ddace addeca babcbdeba abcdba ebad to obtain the digram frequency matrix.

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

```
aabcd ddab ddace addeca babcbdeba abcdba ebad
```

Notice that the a_{13} entry is 1, the number of the occurrences of a followed by c. Also, 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. Here we use Abraham Lincoln's Gettysburg Address to exemplify the extension of this idea.

```
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.
```

This text 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	14	0	0	3	13	0	0	0	0	0	4	1	0	0	4	4	1	1	4	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	0	1	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

Notice that as in the previous example, the ijth entry is the number of occurrences of the ith letter followed by the jth letter. While this 26×26 matrix is rather unwieldy, there is an interesting point to be made. The j, x, and z vectors (both vertically and horizontally) are all zeros. This follows from the fact that there are no j's, x's, or z's used in the text.

But for the time being let's return to our more manageable example to show you that the sum of each row of A is found by Ae, where $e = (1, 1, 1, 1, 1)^T$

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

Multiplying on the right by e sums the columns of A, giving the *occurrence vector* 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 rows of A, giving the same vector f. This is easily checked by summing straight across. *Row* one is,

$$2 + 5 + 1 + 2 + 1 = 11$$

as well, summing *column* one to get,

$$2 + 4 + 1 + 3 + 1 = 11,$$

which is equal to the first entry in f.

The Singular Value Decomposition

When faced with the problem of decoding a cipher, often the cryptalalyst's first approach is to try a comparison of the frequency of the encoded letters with the known frequencies of typical un-coded text. We will refer to this as method 1. 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. We will refer to this as method 2.

An important tool that will aid in both methods is the *singular value decomposition*, or svd. Performing the svd on the frequency matrix A factors it into three matrices, each of which contains useful information about the encoded text.

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

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

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

An expansion gives the following.

$$\begin{aligned} A &= X\Delta Y^T \\ &= \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix} \begin{pmatrix} \delta_1 & & & \\ & \delta_2 & & \\ & & \ddots & \\ & & & \delta_n \end{pmatrix} \begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_n^T \end{pmatrix} \\ &= \delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T + \cdots + \delta_n x_n y_n^T \end{aligned}$$

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

$$\delta_1 x_1 y_1^T,$$

is called the *rank one approximation*. If δ_1 is significantly larger than the remaining singular values, then the rank one approximation closely resembles A.

Rank One Approximation

Via the rank one approximation, we can obtain some useful information about the digram frequency matrix A. Since by equation (1),

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

we can substitute $A \approx \delta_1 x_1 y_1^T$ and write

$$\begin{aligned} (\delta_1 x_1 y_1^T)e &= (\delta_1 x_1 y_1^T)^T e = f \\ (\delta_1 x_1 y_1^T)e &= (\delta_1 y_1 x_1^T)e = f \end{aligned}$$

Reordering,
$$(\delta_1 y_1^T e)x_1 = (\delta_1 x_1^T e)y_1 = f.$$

In the last equation the left and right singular vectors are simply being multiplied by the scalars $(\delta_1 y_1^T e)$, and $(\delta_1 x_1^T e)$. This means that x_1 and y_1 are proportional to f .

Having constructed A from the Gettysburg Address, and performing the svd, we will now compare the first left and right singular vectors to f (see table 1).

At first glance this is not very impressive. While x_1 and y_1 show a fair amount of correlation, f appears to have nothing to do with the other two, although we do have conformation of the absence of j's, x's, and z's by the zeros in those positions. However, if we look at the frequencies of these entries the correlation is remarkable.

To obtain the relative frequency of each entry in any vector we just divide it by the sum of all the entries in that vector (see table 2).

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

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

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

Amazingly x' and y' closely approximate f'. Why is this significant? Given an encoded text one can generate the digram frequency matrix, and then factor it by svd. Using the first left and right singular vectors to approximate the frequency of the occurrence for each letter in the cipher, a skilled cryptanalyst could compare them to the frequencies of each letter in a typical, un-coded text and begin making guesses as to what each coded letter might represent. For instance, since e is the most frequently

$$x_1 = \begin{pmatrix} -0.3279 \\ -0.0471 \\ -0.1201 \\ -0.2012 \\ -0.4397 \\ -0.0875 \\ -0.0966 \\ -0.3468 \\ -0.1832 \\ 0.0000 \\ -0.0099 \\ -0.1204 \\ -0.0607 \\ -0.2166 \\ -0.2387 \\ -0.0523 \\ -0.0007 \\ -0.2954 \\ -0.1683 \\ -0.4455 \\ -0.0533 \\ -0.1167 \\ -0.1211 \\ 0.0000 \\ -0.0340 \\ 0.0000 \end{pmatrix} y_1 = \begin{pmatrix} -0.3219 \\ -0.0442 \\ -0.1136 \\ -0.2261 \\ -0.4515 \\ -0.1023 \\ -0.0800 \\ -0.3381 \\ -0.2340 \\ -0.0000 \\ -0.0097 \\ -0.1219 \\ -0.0468 \\ -0.2438 \\ -0.2564 \\ -0.0565 \\ -0.0054 \\ -0.2496 \\ -0.1393 \\ -0.4371 \\ -0.0597 \\ -0.0818 \\ -0.1045 \\ 0.0000 \\ -0.0344 \\ 0.0000 \end{pmatrix} f = \begin{pmatrix} 102.0000 \\ 14.0000 \\ 31.0000 \\ 58.0000 \\ 165.0000 \\ 26.0000 \\ 28.0000 \\ 80.0000 \\ 68.0000 \\ 0.0000 \\ 3.0000 \\ 42.0000 \\ 13.0000 \\ 77.0000 \\ 93.0000 \\ 15.0000 \\ 1.0000 \\ 79.0000 \\ 44.0000 \\ 126.0000 \\ 21.0000 \\ 24.0000 \\ 28.0000 \\ 0.0000 \\ 10.0000 \\ 0.0000 \end{pmatrix}$$

Table 1: Comparing the first left and right singular vectors with the occurrence vector f.

$$x_1' = \begin{pmatrix} 0.0867 \\ 0.0124 \\ 0.0317 \\ 0.0532 \\ 0.1162 \\ 0.0231 \\ 0.0255 \\ 0.0917 \\ 0.0484 \\ 0.0000 \\ 0.0026 \\ 0.0318 \\ 0.0160 \\ 0.0572 \\ 0.0631 \\ 0.0138 \\ 0.0002 \\ 0.0781 \\ 0.0445 \\ 0.1177 \\ 0.0141 \\ 0.0308 \\ 0.0320 \\ 0.0000 \\ 0.0090 \\ 0.0000 \end{pmatrix} \quad y_1' = \begin{pmatrix} 0.0856 \\ 0.0117 \\ 0.0302 \\ 0.0602 \\ 0.1201 \\ 0.0272 \\ 0.0213 \\ 0.0900 \\ 0.0622 \\ 0.0000 \\ 0.0026 \\ 0.0324 \\ 0.0125 \\ 0.0649 \\ 0.0682 \\ 0.0150 \\ 0.0014 \\ 0.0664 \\ 0.0371 \\ 0.1163 \\ 0.0159 \\ 0.0218 \\ 0.0278 \\ 0.0000 \\ 0.0092 \\ 0.0000 \end{pmatrix} \quad f' = \begin{pmatrix} 0.0889 \\ 0.0122 \\ 0.0270 \\ 0.0505 \\ 0.1437 \\ 0.0226 \\ 0.0244 \\ 0.0697 \\ 0.0592 \\ 0.0000 \\ 0.0026 \\ 0.0366 \\ 0.0113 \\ 0.0671 \\ 0.0810 \\ 0.0131 \\ 0.0009 \\ 0.0688 \\ 0.0383 \\ 0.1098 \\ 0.0183 \\ 0.0209 \\ 0.0244 \\ 0.0000 \\ 0.0087 \\ 0.0000 \end{pmatrix}$$

Table 2: Comparing the frequencies of the first left and right singular vectors with the frequency of occurrence vector f.

used letter in English texts, if the rank one approximation of an encoded text shows z to be the most frequently occurring letter, it would be a pretty safe bet that z should map to e.

As you might guess, this would be a long, arduous process. However, the cryptanalyst's job can be made much easier by looking at the *rank two approximation*. In conjunction with what we have learned using the rank one approximation, this second method will get us much closer to deciphering an encoded message.

Rank Two Approximation

Recall that

$$\begin{aligned} A &= X\Delta Y^T \\ &= \begin{pmatrix} x_1 & x_2 & \cdots & x_n \end{pmatrix} \begin{pmatrix} \delta_1 & & & \\ & \delta_2 & & \\ & & \ddots & \\ & & & \delta_n \end{pmatrix} \begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_n^T \end{pmatrix} \\ &= \delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T + \cdots + \delta_n x_n y_n^T. \end{aligned}$$

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

$$\delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T,$$

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

However, before we proceed, let's make this transition into Linear Algebra complete by thinking of the alphabet as two vectors, v and c. The entries of v signify a vowel as a 1 and the entries of c signify a consonant as a 1.

As before, we return to our restricted alphabet to demonstrate.

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

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.

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

Indeed, as depicted below in red, there are only four instances in our previous example where a vowel is followed by a vowel.

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

Using the vfc Rule

As stated earlier, a second 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. For our partition to be correct, the vfc rule must be satisfied. Now that we have developed the symbolism for the number of consonants, vowels, and pairs, we can express the vfc rule mathematically.

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

and symbolically,

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

Using the common denominator we can simplify.

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

It has been deemed "the cryptanalyst's problem" to find a partitioning of the encoded alphabet such that the above inequality will hold. Returning to the rank two approximation, we propose to use the signs of the components of x_2 and y_2 to partition the alphabet into v, c, and n vectors.

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

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

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

Here n is the vector of letters that we cannot categorize as either vowels or consonants. The reason why this partitioning scheme works is because we are using the rank two approximation, $A \approx \delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T$.

Recall the final form of the vfc equation,

$$D = (v^T A v)(c^T A c) - (v^T A c)(c^T A v) < 0. \tag{2}$$

Substituting this approximation in for A gives

$$\begin{aligned} D = v^T(\delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T)vc^T(\delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T)c \\ - v^T(\delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T)cc^T(\delta_1 x_1 y_1^T + \delta_2 x_2 y_2^T)v. \end{aligned} \tag{3}$$

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

$$\begin{aligned} D = \delta_1\delta_2[(v^T x_1)(y_1^T v)(c^T x_2)(y_2^T c) + (v^T x_2)(y_2^T v)(c^T x_1)(y_1^T c) \\ - (v^T x_1)(y_1^T c)(c^T x_2)(y_2^T v) - (v^T x_2)(y_2^T c)(c^T x_1)(y_1^T v)]. \end{aligned} \tag{4}$$

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. Also, notice that the entries in our first left and right singular vectors, (see table 1) are all negative. This is explained by the Perron-Frobenius theorem which states that if A is a non-negative matrix, then the first right and left singular vectors will both have either all non-positive entries or all non-negative entries. Since in our case, they are all non-positive any of the above factors in which x_1 or y_1 appear will be negative. This takes care of eight of the factors.

By our definition of the partition vectors you can see that v times a y_2 or a c times an x_2 will yield a positive factor. Each of the remaining four factors are either a v times an x_2 or a c times a y_2. Both of the these dot products will produce negative answers by the definition. Observing where each of these factors appears in equation (4) shows that D will be the sum of 4 negative terms.

$$v = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ \vdots \end{pmatrix}, \quad x_2 = \begin{pmatrix} -0.5090 \\ 0.0413 \\ 0.0722 \\ 0.1439 \\ -0.3304 \\ \vdots \end{pmatrix}, \quad c = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \quad \text{and} \quad y_2 = \begin{pmatrix} 0.1582 \\ -0.0386 \\ -0.0787 \\ -0.2161 \\ 0.5316 \\ \vdots \end{pmatrix}$$

When applied to the Gettysburg Address the rank two approximation yields the following partitioning (see table (3)).

As you can see the rank two approximation produces a surprisingly accurate partition. In fact, the chosen text does not contain any x's, or z's and consequently the rank two approximation accurately assigns a zero in each of the categories v, c, and n since there was no opportunity to test for their categorization. This is nice that it works for an un-coded text such as the Gettysburg Address, but how can we be sure that it will work for a coded text?

If a text is encoded using a simple substitution cipher, then the new alphabet is represented by p. The vector p is a permutation of the numbers $1, 2, \ldots 26$, representing letters $a, b, \ldots z$. C is the 26 by 26 identity matrix whose rows have been permuted in

	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

Table 3: Partition attempt based on the rank two approximation.

the same way as p. If A is the digram frequency matrix for the original text and B is the digram frequency matrix for the encoded text, then $B = C^T AC$. It can be shown that the matrices $B^T B$, and $A^T A$ have the same eigenvalues. To find the eigenvalues of any matrix we find the roots of it's characteristic polynomial. Using $B^T B = C^T A^T AC$ we can find the characteristic polynomial for $B^T B$.

$$\begin{aligned} P_{B^T B}(\lambda) &= |B^T B - \lambda I| \\ &= |C^T A^T AC - \lambda I| \\ &= |C^T A^T AC - C^T \lambda I C| \\ &= |C^T||A^T A - \lambda I||C| \\ &= |A^T A - \lambda I| \\ &= P_{A^T A}(\lambda). \end{aligned}$$

This is important because the eigenvalues for any matrix $M^T M$ are the squares of the singular values of M. Therefore, if $B^T B$ and $A^T A$ share the same eigenvalues, then we know that A and B share the same singular values. This allows us to write,

$$B = C^T AC = C^T X_A \Delta_A Y_A^T C = (C^T X_A)\Delta_A(C^T Y_A)^T.$$

But, $\Delta_B = \Delta_A$, so we can write $B = (C^T X_A)\Delta_B(C^T Y_A)^T$.

Hence, $X_B = C^T X_A$ and $Y_B = C^T Y_A$ means that the left and right singular vectors of B are just a permutation of the left and right singular vectors of A. The algorithm that assigns vowels and consonants depends solely on the relationships between the signs of the corresponding entries in the x_2 and y_2 vectors. Because these relationships will be maintained through the permutation by C^T we can use the same partitioning scheme.

Suppose we encode the Gettysburg Address using the simple reversed alphabet as mentioned earlier.

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

Four score and seven years ago...
Ulfi hxliv zmw hvevm bvzih ztl...

Generating the digram frequency matrix for the cipher and then applying the partition algorithm produces the following partitioning (see table (4)).

Again, the algorithm has assigned the correct letters as vowels and consonants. We can check because we know that a has been mapped to z, and z is now assigned as a vowel. In fact both of the new vowel and consonant columns are just the original ones inverted. This is what we expected because to encode the Gettysburg Address, we simply inverted the alphabet.

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

- A text is represented by a digram frequency 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.

The Matlab functions used herein can be found at the following link:

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

Table 4: Partition attempt of encoded text

http://online.redwoods.edu/instruct/darnold/laproj/fall2002/TimSeth/matlab.tar.gz.

References

[1] Arnold, David. Mathetics, LaTeX, Matlab, Patients, Dedication, Support.

[2] Garrett, Paul. Making, Breaking Codes: An Introduction to Cryptology 2001, Prentice Hall, Inc., Upper Saddle River, NJ, pages 40-42.

[3] Moler,Cleve, Donald Morrison. *American Mathematical Monthly*, Volume 90, Issue 2 (Feb.,1983), pages 78-87.

[4] Strang, Gilbet. Introduction to Linear Algebra, Second Edition. Wellesley Cambridge Press, 1998.