Lecture 14
Singular Value Decomposition

02 November 2015

Taylor B. Arnold
Yale Statistics
STAT 312/612

Yale

Goals for today

- singular value decomposition
- condition numbers
- application to mean squared errors
- (time permitting) image dataset

A matrix $A \in \mathbb{R}^{n \times m}$ can be thought of as a linear mapping between two spaces:

$$A : \mathbb{R}^m \to \mathbb{R}^n$$

This interpretation requires no assumptions on the shape or structure of the matrix A.

The singular value decomposition writes the matrix A as a product of three matricies:

$$A = U\Sigma V^t$$

Where $U \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{m \times m}$ are orthonormal matricies and Σ is the rectangular diagonal matrix $\text{diag}\big(\sigma_1, \sigma_2, \ldots, \sigma_{\min(n,m)}\big)$.

This decompositon exists for any real matrix A.

By convention, the values of Σ are arranged in decending order: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(n,m)}$.

These are called the **singular values** of the matrix A.

The number of non-zero singular values is equal to the rank of the matrix A.

The singular value decomposition allows us to write the matrix A as a sum of r, rank 1 matricies:

$$A = \sum_{i}^{r=\text{rank}(A)} \sigma_i u_i v_i^t$$

A useful way of viewing the singular value decomposition is to think about what would happen when projecting columns of U and V:

$$Av_i = \sigma_i u_i$$
$$A^t u_i = \sigma_i v_i$$

Notice that both equations use σ_i!

Therefore, if we have an arbitrary vector $z \in \mathbb{R}^m$ and we write it in the basis of V:

$$z = \sum_i \alpha_i v_i$$

The mapping of A can be easily calculated in the coordinate system of U:

$$Az = \sum_i \alpha_i \sigma_i u_i$$

Due to the linearity of the matrix operation.

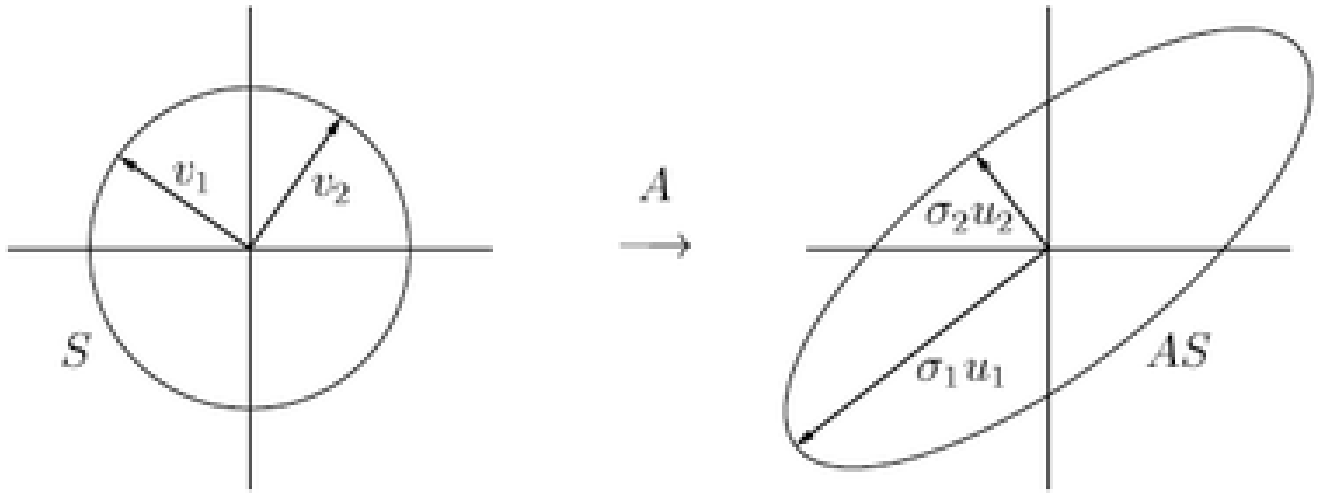


Figure 4.1. *SVD of a* 2×2 *matrix.*

As an example, let's construct a 2-by-3 matrix:

```
> A <- matrix(1:6,ncol=3)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

The singular value decomposition can be calculated by the `svd` function in R. By default only $\min(n, m)$ columns of U and V are calculated, but we'll ask for all of them here.

```
> svd(A, nu=2, nv=3)
$d
[1] 9.5255181 0.5143006

$u
           [,1]       [,2]
[1,] -0.6196295 -0.7848945
[2,] -0.7848945  0.6196295

$v
           [,1]       [,2]       [,3]
[1,] -0.2298477  0.8834610  0.4082483
[2,] -0.5247448  0.2407825 -0.8164966
[3,] -0.8196419 -0.4018960  0.4082483
```

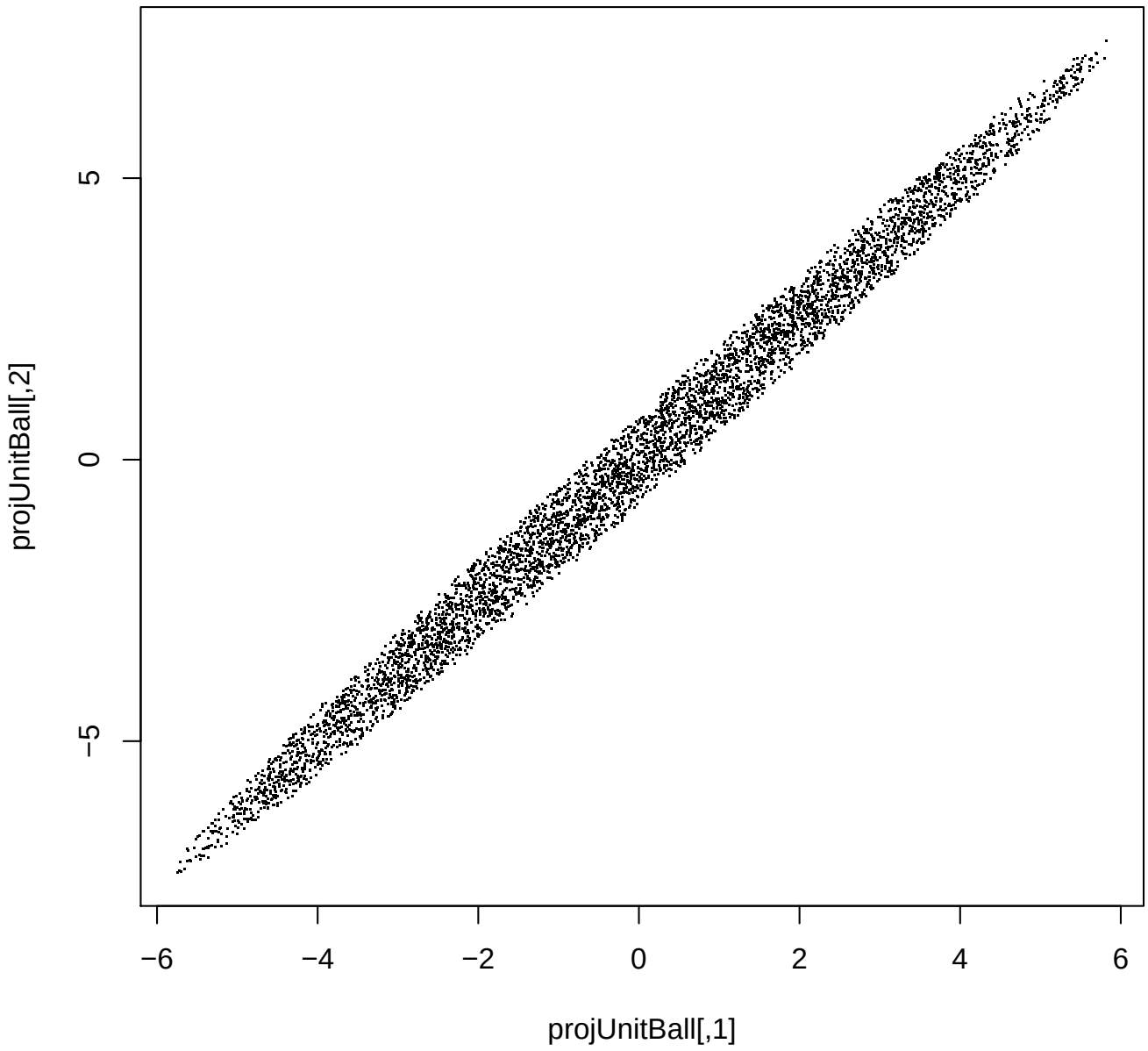
We can extract the three components of the SVD and verify that the matrix A is returned:

```
> SVD <- svd(A, nu=2, nv=3)
> Sigma <- cbind(diag(SVD$d),0)
> U <- SVD$u
> V <- SVD$v
> A - U %*% Sigma %*% t(V)
              [,1]          [,2] [,3]
[1,] 2.220446e-16 4.440892e-16    0
[2,] 0.000000e+00 4.440892e-16    0
```

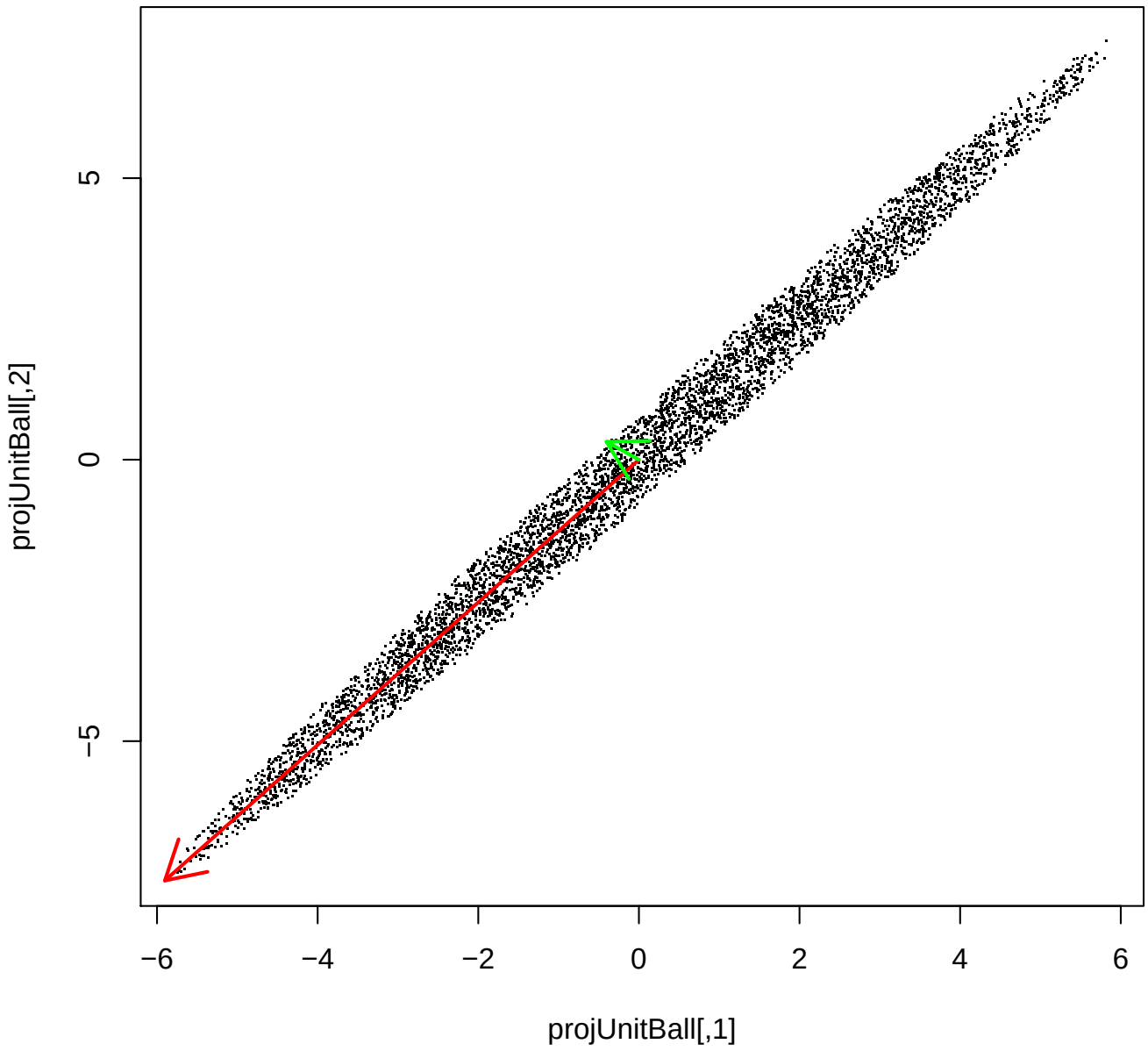
Notice that there are some small errors in the decomposition.

```
> N <- 1e5
> p <- 3
> unitBall <- matrix(runif(N * p, -1, 1), nrow=3)
> unitBall <- unitBall[,apply(unitBall^2, 2, sum) < 1]
> unitBall[,1:5]
          [,1]       [,2]       [,3]       [,4]        [,5]
[1,] 0.5287973 -0.7040859 -0.7248478 -0.4671987  0.05705657
[2,] 0.4255332 -0.3016897  0.3072752  0.5290554  0.78391679
[3,] 0.5349525 -0.4057218 -0.5348268 -0.3940231 -0.51866525
```

```
> projUnitBall <- t(A %*% unitBall)
> head(projUnitBall)
           [,1]       [,2]
[1,]  4.4801594  5.9694424
[2,] -3.6377640 -5.0492614
[3,] -2.4771562 -3.4295556
[4,] -0.8501476 -1.1823139
[5,] -0.1845193  0.1377888
[6,]  4.8409047  5.9547905
> plot(projUnitBall,pch=".")
```


projUnitBall[,2]
5
0
−5
−6
−4
−2
0
2
4
6
projUnitBall[,1]

```
> A %*% V
           [,1]       [,2]          [,3]
[1,] -5.902292 -0.4036717 8.881784e-16
[2,] -7.476526  0.3186758 4.440892e-16
> sqrt(apply((A %*% V)^2, 2, sum))
 [1] 9.525518e+00 5.143006e-01 9.930137e-16
>
> v1 <- (A %*% V)[,1]
> v2 <- (A %*% V)[,2]
> arrows(0,0,v1[1],v1[2],col="red",lwd=2)
> arrows(0,0,v2[1],v2[2],col="green",lwd=2)
```


projUnitBall[,2]
5
0
−5
−6
−4
−2
0
2
4
6
projUnitBall[,1]

Now consider the following quantity for a full rank matrix A:

$$\frac{||A\delta||_2}{||\delta||_2}$$

Let $\delta = \sum_i \alpha_i v_i$. Then:

$$\frac{||A\delta||_2}{||\delta||_2} = \sqrt{\frac{\sum_i \sigma_i^2 \alpha_i^2 v_i^2}{\sum_i \alpha_i^2 v_i^2}}$$

We can see that the minimum occurs when δ is equal to $v_{\min(n,m)}$.

Likewise, the maximum occurs when δ is equal to v_1.

So, we have the following inequality for all $\delta \neq 0$:

$$\sigma_{\min} \leq \left\{ \frac{||A\delta||_2}{||\delta||_2} \right\} \leq \sigma_{\max}$$

For the singular values $\sigma_{\min}$ and $\sigma_{\max}$.

Finally, notice that the inner product can be written compactly in terms of the singular values:

$$\begin{aligned} A^t A &= V\Sigma^t U^t U \Sigma V^t \\ &= V\Sigma^2 V^t \end{aligned}$$

This just the eigenvalue decomposition of the matrix $A^t A$. Notice that the eigenvalues of $A^t A$ are the squares of the singular values of A.

Back to linear models

In a linear model, we only observe $X\beta$, rather than β itself. We have already seen that numerical problems can lead to multiple solutions for which the $X\beta$'s is very similar but the regression vectors β are quite different.

Say that we have an error (or noise) Δ in the term β. Formally, we wish to control the ratio of the relative error in estimation to that of projection:

$$\frac{\text{rel. error estimation}}{\text{rel. error projection}} = \frac{||\beta + \Delta||_2 / ||\beta||_2}{||X(\beta + \Delta)||_2 / ||X\beta||_2} < \epsilon$$

So we do not want large changes in Δ to yield relatively small changes in the prediction space $X\beta$.

Notice that we can re-arrange the equation as:

$$\frac{||\beta + \Delta||_2 / ||X(\beta + \Delta)||_2}{||\beta||_2 / ||X\beta||_2}$$

And now we have an upper bound on the numerator and an lower bound on the denominator via the singular values:

$$\frac{\text{rel. error estimation}}{\text{rel. error projection}} \leq \frac{\sigma_{max}}{\sigma_{min}}$$

This is called the *condition number* of the matrix A, and was the quantity R complained about last week when I tried to invert an ill-conditioned matrix.

The Δ in this equation could be (amongst other things) numerical error, measurment error, or statistical noise. In any case, badly conditions matrices X make solving linear systems difficult.

Mean squared error

Completely switching topics for the moment, consider the mean squared error (MSE) of an estimator $\widehat{\beta}$:

$$\begin{aligned}
\mathbb{E}||\beta - \widehat{\beta}||_2^2 &= \mathbb{E}\sum_j (\beta_j - \widehat{\beta})^2 \\
&= \sum_j \mathbb{E}(\beta_j - \widehat{\beta})^2 \\
&= \sum_j \text{Var}(\widehat{\beta}_j) + \left[\mathbb{E}(\beta_j - \widehat{\beta})\right]^2 \\
&= \text{tr}(\text{Var}(\widehat{\beta})) + ||\text{Bias}(\widehat{\beta})||_2^2
\end{aligned}$$

This is the multivariate version of the version you (hopefully) saw in introductory statistics.

The mean squared error of the ordinary least squares estimator can be calculated as follows, given the formula we derived for the variance and the fact that it is unbiased:

$$\begin{aligned} \text{MSE}(\widehat{\beta}_{OLS}) &= \text{tr}\left[\sigma^2 (X^t X)^{-1}\right] \\ &= \sigma^2 \text{tr}\left[(X^t X)^{-1}\right] \end{aligned}$$

On the last homework, I ask you to look at an estimator which has been shrunk towards zero by a factor of α:

$$\widehat{\beta}_{\alpha} = \alpha \cdot \widehat{\beta}_{OLS}$$

You then compared the mean squared error of this to the standard ordinary least squares solution using a series of simulations.

Let's calculate the mean squared error directly.

We see quickly that:

$$\begin{aligned}\text{MSE}(\widehat{\beta}_\alpha) &= \text{tr}(\text{Var}(\widehat{\beta}_\alpha)) + ||\text{Bias}(\widehat{\beta}_\alpha)||_2^2 \\ &= \alpha^2\sigma^2\text{tr}\left[(X^tX)^{-1}\right] + (1-\alpha)^2||\beta||_2^2\end{aligned}$$

What is the relationship between the optimal α and the quantities σ^2 and $||\beta||_2^2$?

Sacrificing bias for a reduction in variance is, generally, a very good idea, but we are not doing so in a very intelligent way here.

What is the quantity $\text{tr}\left[(X^tX)^{-1}\right]$? Well, in terms of singular values we have:

$$\begin{aligned}
\text{tr}\left[(X^tX)^{-1}\right] &= \text{tr}\left[(V\Sigma^2V^t)^{-1}\right] \\
&= \text{tr}\left[V\Sigma^{-2}V^t\right] \\
&= \text{tr}\left[\Sigma^{-2}V^tV\right] \\
&= \text{tr}\left[\Sigma^{-2}\right] \\
&= \sum_{i=1}^{r}\frac{1}{\sigma_i^2}
\end{aligned}$$

So a disproportionate amount of variance is coming from the smallest singular values.

What if we could just shrink the variance in the directions of the lowest singular values?

Two possible solutions:

(1) **Principal component regression** uses only the first k singular vectors of the data matrix X in the regression model.

(2) **Ridge regression** scales the ordinary least squares solution by shrinking a small amount in the direction of the largest singular values and a large amount in the direction of the smallest singular values.

Summary of today's lecture

1. Singular value decomposition can be applied to any real matrix A without regard to its shape or structure.
2. Singular values are a generalization of eigenvalues.
3. The ratio of the largest singular value to the smallest singular value indicates how difficult it is to solve the linear system $y = Ab$ by least squares. The quantity is called the condition number of the matrix.
4. The smallest singular value directions contribute a disproportionate amount of variance in the estimation of the regression vector using ordinary least squares.