Assignment 10

Benjamin Jakubowski

December 13, 2015

1. Statements

a. Subspaces

Statement 1: For any subspace $\mathcal{S}$ belonging to a vector space $\mathcal{V}$ of dimension n

$$\dim(\mathcal{S}) + \dim(\mathcal{S}^{\perp}) = n$$

Proof:
Let $\{s_1, s_2, ..., s_j\}$ be an orthonormal basis for the subspace $\mathcal{S}$. Note that $\dim(\mathcal{S}) = j$. Now let $\{v_1, v_2, ..., v_n\}$ be a basis for the vector space $\mathcal{V}$. Then, using Gram-Schmidt, we can construct an orthonormal basis for $\mathcal{V}$ that includes the basis vectors for $\mathcal{S}$:

$$\{s_1, s_2, ..., s_j, w_1, w_2, ..., w_{n-j}\}$$

where the w_i's are unit norm, linearly independent vectors constructed as linear combinations of the original basis vectors vectors $v_1, v_2, ..., v_n$.

Now take $y \in \mathcal{S}^{\perp}$. Then, since $y \in \mathcal{V}$, we know:

$$y = \sum_{i=1}^{j} \alpha_i s_i + \sum_{k=1}^{n-j} \beta_k w_k$$

Since $y \in \mathcal{S}^{\perp}$, we know that $\alpha_i = 0$ for all $i \in \{1, ..., j\}$. Thus y is a linear combination of the w_k's. But then, since y was arbitrary, we know that $\mathcal{S}^{\perp}$ is spanned by $\{w_1, w_2, ..., w_{n-j}\}$. Finally, by construction we know the w_i's are linearly independent, so $\{w_1, w_2, ..., w_{n-j}\}$ is a basis for $\mathcal{S}^{\perp}$. Thus, $\dim(\mathcal{S}^{\perp}) = n - j$ and

$$\dim(\mathcal{S}) + \dim(\mathcal{S}^{\perp}) = n$$

□

B. Dimensions of the null space and range of A

Statement 2:

$$\dim(\text{null}(A)) + \dim(\text{range}(A)) = n$$

Proof:

By Lemma 1.5 (in lecture notes 10),

$$\text{null}(A) = \text{row}(A)^{\perp}$$

Then, by statement (a) (proved above),

$$\dim(\text{row}(A)^{\perp}) + \dim(\text{row}(A)) = n$$

But (by Theorem 6.2 in lecture notes 9), $\dim(\text{row}(A)) = \dim(\text{col}(A))$. Thus,

$$\begin{aligned} & \dim(\text{row}(A)^{\perp}) + \dim(\text{row}(A)) = n \\ \implies \quad & \dim(\text{row}(A)^{\perp}) + \dim(\text{col}(A)) = n \\ \implies \quad & \dim(\text{null}(A)) + \dim(\text{col}(A)) = n \quad \square \end{aligned}$$

C. $A^T(AA^T)^{-1}Ax$ as projection onto row space of full-rank, fat A

Statement: For any full-rank matrix $A \in \mathbb{R}^{m \times n}, m \leq n, A^T(AA^T)^{-1}Ax$ is the projection of $x \in \mathbb{R}^n$ onto the row space of A.

Proof:

First, note A^T is a full-rank matrix in $\mathbb{R}^{n \times m}, n \geq m$. Therefore, by Theorem 2.2 in Lecture notes 10, we know:

$$\mathcal{P}_{\text{range}(A^T)}x = (A^T)((A^T)^T(A^T))^{-1}(A^T)^T x = A^T(AA^T)^{-1}Ax$$

Then merely noting $\text{range}(A^T) = \text{row}(A)$ yields the desired result:

$$\mathcal{P}_{\text{row}(A)}x = A^T(AA^T)^{-1}Ax$$

$\square$

As a side note, we can perhaps better understand this projection by decomposing the expression into two pieces:

$$\begin{aligned} \mathcal{P}_{\text{row}(A)}x &= A^T(AA^T)^{-1}Ax \\ &= \begin{bmatrix} A_{\text{row 1}} & A_{\text{row 2}} & \cdots & A_{\text{row m}} \end{bmatrix} \underbrace{(AA^T)^{-1}Ax}_{\text{Coordinates of } x \text{ w.r.t. row's of } A} \end{aligned}$$

d. Regression with an orthogonal matrix

Statement 4: If the columns of $A \in \mathbb{R}^{m \times n}, m \geq n$, are orthonormal then the solution to the least-squares problem is of the form

$$x = \arg\min_z ||y - Az||_2 = A^T y$$

Proof:

If the columns of $A \in \mathbb{R}^{m \times n}, m \geq n$ are orthonormal, than A is an orthogonal matrix. But then,

$$AA^T = \mathbf{1}_n$$

so

$$A(A^T y) = (AA^T)y = \mathbf{1}_n y = y$$

So the least squares solution to the equation $Ax = y$ is

$$x_{LS} = \arg\min_z ||y - Az||_2 = A^T y$$

□

2. Global Warming

a. Completed hw10_pb2.py script

Here are the scripts necessary to fit the models:

```
# First build matrix A:
import pandas as pd


t = np.arange(n)
A = pd.concat([pd.Series(np.ones(n).T, name='One'),\
               pd.Series(np.cos(2.0*np.pi*t/12.0).T, name='Cos'),\
               pd.Series(np.sin(2.0*np.pi*t/12.0).T, name='Sin'),\
               pd.Series(t.T, name='Month')], axis=1)

#Then fit models
max_model_coef = np.linalg.lstsq(A.values, max_temp)[0]
def max_model(t):
    y_fit = max_model_coef[0] +\
    max_model_coef[1]*np.cos(2.0*np.pi*t/12.0) +\
    max_model_coef[2]*np.sin(2.0*np.pi*t/12.0) +\
    max_model_coef[3]*t
    return y_fit
```

```
def max_trend(t):
    y_fit = max_model_coef[0] + max_model_coef[3]*t
    return y_fit

min_model_coef = np.linalg.lstsq(A.values, min_temp)[0]
def min_model(t):
    y_fit = min_model_coef[0] +\
    min_model_coef[1]*np.cos(2.0*np.pi*t/12.0) +\
    min_model_coef[2]*np.sin(2.0*np.pi*t/12.0) +\
    min_model_coef[3]*t
    return y_fit

def min_trend(t):
    y_fit = min_model_coef[0] + min_model_coef[3]*t
    return y_fit

reconstruction_max = max_model(t)
reconstruction_min = min_model(t)
trend_max =  max_trend(t)
trend_min =  min_trend(t)
```

b. Hypothesis Test

Recall we fit the same model to data from 100 stations and observe the slope is positive for 65 stations. We are interested in testing:

$$\begin{aligned} \mathcal{H}_0 : &\quad \beta_{linear} = 0 \\ \mathcal{H}_A : &\quad \beta_{linear} \neq 0 \end{aligned}$$

Under our null hypothesis, the number of observations with $\beta_{linear} > 0$ is distributed as a binomial random variable X with $n = 100$ and $p = 0.5$. Thus, conservatively using a two-sided hypothesis test, we get

$$\begin{aligned} P &= P(X \leq 35 \text{ or } X \geq 65) \\ &= P(X \leq 35) + P(X \geq 65) \\ &= \sum_{i=0}^{35} \binom{100}{i} \cdot 0.5^{100} + \sum_{i=65}^{100} \binom{100}{35} \cdot 0.5^{100} \\ &= 0.00176 + 0.00176 = 0.00352 \end{aligned}$$

Thus we have strong evidence of warming if 65 of 100 stations record data yield positive β_{linear}.

3. Weight prediction

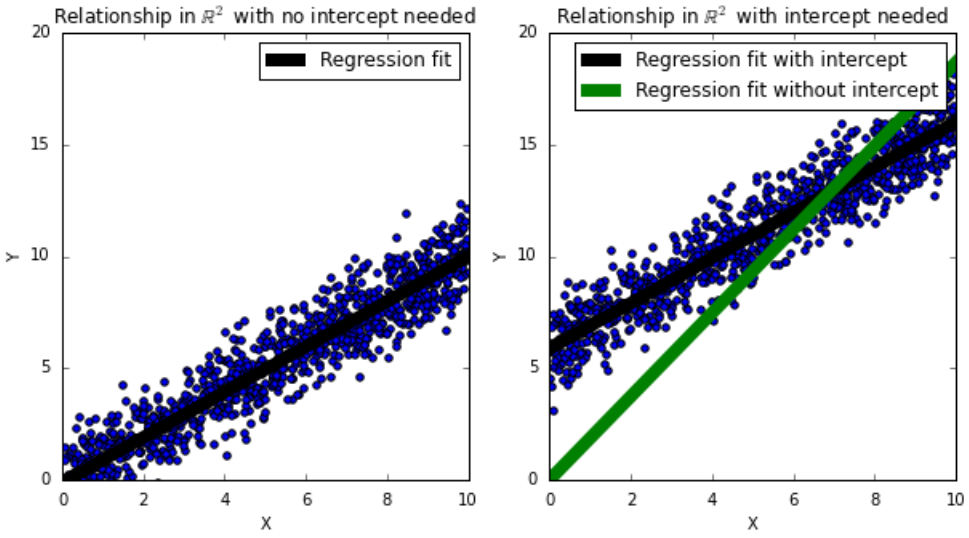
A. Least-squares estimate of α without intercept

Using theorem 2.2, the least squares estimate of α is:

$$\alpha_{LS} = (w^T \cdot w)^{-1} \cdot w^T \cdot h$$

B. Adding an intercept

The point of adding an intercept is to give the linear model an additional degree of freedom. Without the intercept term (in a 2D data set), the regression line is constrained to pass through the origin. Adding an intercept terms removes this constraint, allowing the regression model to capture any linear relationship in $\mathbb{R}^2$. This is illustrated below using simulated data:



C. Models with and without intercept for heigh/weight

After completing the script *hw*10_*pb*2.*py*, the relative errors achieved by the two models on the test dataset are:

1. Model 1 (no intercept) relative error: 0.06295
2. Model 2 (with intercept) relative error: 0.01951

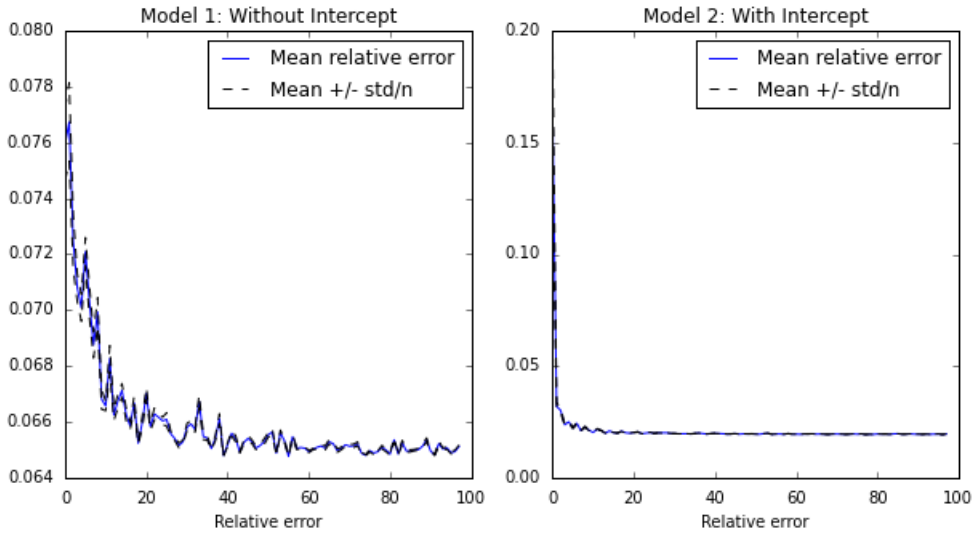
The model with the intercept terms has much lower relative error, so adding the intercept clearly improves the model.

D. Models with and without intercept for height/weight

To compare the performance of the two models on the height weight test set, the following simulations were conducted:

- For $n \in \{2, 100\}, n$ observations were randomly sampled from the data to form the test set.
- Both models were fit, and the relative error was determined using $21000-n$ held-out observations.
- This was repeated 10 times, and the mean and standard deviation of the relative error was calculated for each n.

Results from these simulations are shown below:



As you can see, including the intercept results in improved model performance on all but the very smallest n. In fact, the no-intercept model only outperformed the intercept model when $n = 2$. Given this n is absurd (in no realistic situation would you try fitting a model to two observations), these results suggest including an intercept produces better results when less data is available.

In general, I would favor linear models when less data is available. This is because allowing non-linearity may lead to overfitting, and this is more likely when less data is available. When more data is available, we are better able to control overfitting (through use of hold-out validation sets, for example) and can introduce more complexity into our model. When less data is available, the constraint imposed by the linearity requirement helps avoid overfitting.

4. Noise Amplification

Note: This problem was completed with significant assistance from Professor Fernandez-Granda, Nora, Mike, Filipe, and the other students present at the Professor's office hours on 12/2/15.

a. Least-squares solution

Recall we are interested in estimating a vector $x \in \mathbb{R}^n$ from data $y \in \mathbb{R}^m, m \geq n$, that we know follows the model

$$y = Ax + z$$

where $A \in \mathbb{R}^{m \times n}$ is full rank.
We aim to find the least-squares solution x_{LS} in terms of the SVD of A, z, and x.
First, we know:

$$x_{LS} = VS^{-1}U^T y$$

where USV^T is the SVD of A. Then, substituting for y yields

$$x_{LS} = VS^{-1}U^T(Ax + z) = VS^{-1}U^T(USV^T x + z) = x + VS^{-1}U^T z$$

b. Maximizing the estimation error

Now we are interested in finding z with unit ℓ_2 norm that maximizes the estimation error $||x_{LS} - x||_2$. Note this is equavelent to maximizing $||x_{LS} - x||_2^2$.
First, substituting in the expression from (a) yields:

$$||x_{LS} - x||_2^2 = ||(x + VS^{-1}U^T z) - x||_2^2 = ||VS^{-1}U^T z||_2^2$$

Now, since V is an orthogonal matrix, we know

$$||VS^{-1}U^T z||_2^2 = ||S^{-1}U^T z||_2^2$$

Next, before we proceed, let's introduce some notation- let u_i represent the i^{th} column of U, such that $U = \begin{bmatrix} u_1 & u_2 & \dots & u_n \end{bmatrix}$. Then expanding yields

$$\begin{aligned} ||S^{-1}U^T z||_2^2 &= \left|\left| \begin{bmatrix} \frac{1}{\sigma_1} & & \\ & \ddots & \\ & & \frac{1}{\sigma_n} \end{bmatrix} \begin{bmatrix} u_1^T z \\ \vdots \\ u_n^T z \end{bmatrix} \right|\right|_2^2 \\ &= \left|\left| \begin{bmatrix} \frac{u_1^T z}{\sigma_1} \\ \vdots \\ \frac{u_n^T z}{\sigma_n} \end{bmatrix} \right|\right|_2^2 \\ &= \sum_{i=1}^{n} \left(\frac{u_n^T z}{\sigma_i} \right)^2 \end{aligned}$$

Now (noting the columns of U form an orthonormal basis for $\mathbb{R}^n$), let

$$z = \sum_{i=1}^{n} \alpha_i u_i$$

Then note, for all $1 \leq k \leq n$,

$$u_k^T z = u_k^T \sum_{i=1}^{n} \alpha_i u_i = \sum_{i=1}^{n} \alpha_i u_k^T u_i = \alpha_k$$

so

$$\begin{aligned} ||S^{-1}U^T z||_2^2 &= \sum_{i=1}^{n} \left(\frac{u_n^T z}{\sigma_n} \right)^2 \\ &= \sum_{i=1}^{n} \left(\frac{\alpha_i}{\sigma_i} \right)^2 \end{aligned}$$

Then, note that for all $1 \leq i \leq n$, $\left(\frac{\alpha_i}{\sigma_i}\right)^2 \leq \left(\frac{\alpha_i}{\sigma_{min}}\right)^2$. Thus,

$$\begin{aligned} \sum_{i=1}^{n} \left(\frac{\alpha_i}{\sigma_i} \right)^2 &\leq \sum_{i=1}^{n} \left(\frac{\alpha_i}{\sigma_{min}} \right)^2 \\ &= \left(\sum_{i=1}^{n} \alpha_i^2 \right) / \sigma_{min}^2 \end{aligned}$$

But, since the z has unit norm, we know $\sum_{i=1}^{n} \alpha_i^2 = 1$. Thus, we have

$$||x_{LS} - x||_2^2 = ||S^{-1}U^T z||_2^2 \leq \frac{1}{\sigma_{min}^2}$$

Thus, to maximize the error, we want z to be in the direction of u_{min}, the left singular vector corresponding to the least singular value- in other words, we want

$$z = \sum_{i=1}^{n} \alpha_i u_i = 1 \cdot u_{min} + \sum_{\substack{i=1 \\ i \neq \min}}^{n} 0 \cdot u_i$$

In the case of our example matrix

$$A = \begin{bmatrix} 2.1 & 1.1 \\ 3.2 & 1.6 \\ 2.4 & 1.2 \end{bmatrix}$$

the corresponding left singular vector (found using WolframAlpha) is

$$u_{\min} = \begin{bmatrix} 0.883558 \\ -0.374658 \\ -0.280993 \end{bmatrix}$$

This vector corresponds to the minimum singular value of $\sigma_{min} = 0.0395142$. Hence, the maximum error is

$$||x_{LS} - x||_{2_{\max}} = \frac{1}{0.0395142} = 25.3$$

C. Ridge regression

In ridge regression, we optimize the cost function

$$\min_x ||Ax - y||_2^2 + \gamma^2 ||x||_2^2$$

Our goal is to rewrite this as a least-squares problem of the form

$$\min_x ||Bx - c||_2$$

First, note for arbitrary matrices G, H $||G||_2^2 + ||H||_2^2 = \left|\left|\begin{bmatrix} G \\ H \end{bmatrix}\right|\right|_2^2$. Thus,

$$\begin{aligned} ||Ax - y||_2^2 + \gamma^2 ||x||_2^2 &= ||Ax - y||_2^2 + ||\gamma x||_2^2 \\ &= ||Ax - y||_2^2 + ||\gamma \mathbb{I}_n x||_2^2 \\ &= \left|\left|\begin{bmatrix} Ax - y \\ \gamma \mathbb{I}_n x \end{bmatrix}\right|\right|_2^2 \\ &= \left|\left|\begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix} x - \begin{bmatrix} y \\ \mathbf{0} \end{bmatrix}\right|\right|_2^2 \end{aligned}$$

where $\mathbf{0}$ is $n \times 1$. Moreover, note that

- $\begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix}$ is $(m+n) \times n$
- $\begin{bmatrix} y \\ \mathbf{0} \end{bmatrix}$ is $(m+n)$.

D. Expression for x_{RR}

We aim to show that

$$x_{RR} := \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2} (\sigma_i v_i^T x + u_i^T z) v_i$$

We proceed by expanding the least-squares form of ridge regression:

$$
\begin{aligned}
x_{RR} &= \min_x ||Ax - y||_2^2 + \gamma^2 ||x||_2^2 \\
&= \min_x \left|\left| \begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix} x - \begin{bmatrix} y \\ \mathbf{0} \end{bmatrix} \right|\right|_2^2 \\
&= \left(\begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix}^T \begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ \gamma \mathbb{I}_n \end{bmatrix}^T \begin{bmatrix} y \\ \mathbf{0} \end{bmatrix} \\
&= \left(\left[A^T A + \gamma^2 \mathbb{I}_n \right] \right)^{-1} \left[A^T y \right] \\
&= \left(\left[(VSU^T)(USV^T) + \gamma^2 \mathbb{I}_n \right] \right)^{-1} \left[A^T y \right] \\
&= \left(\left[VS^2 V^T + \gamma^2 \mathbb{I}_n \right] \right)^{-1} \left[A^T y \right] \\
&= \left(\left[VS^2 V^T + \gamma^2 VV^T \right] \right)^{-1} \left[A^T y \right] \\
&= \left(V \left[S^2 + \gamma^2 \mathbb{I}_n \right] V^T \right)^{-1} \left[A^T y \right] \\
&= V^{-T} \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} V^{-1} \left[A^T y \right] \\
&= V \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} V^T \left[A^T y \right] \\
&= V \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} V^T \left[(VSU^T) y \right] \\
&= V \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} SU^T y \\
&= V \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} SU^T (Ax + z) \\
&= V \left[S^2 + \gamma^2 \mathbb{I}_n \right]^{-1} SU^T (USV^T x + z) \\
&= V \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2 + \gamma^2} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{\sigma_n}{\sigma_n^2 + \gamma^2} \end{bmatrix} U^T (USV^T x + z) \\
&= V \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2 + \gamma^2} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{\sigma_n}{\sigma_n^2 + \gamma^2} \end{bmatrix} (SV^T x + U^T z) \\
&= \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2} (\sigma_i v_i^T x + u_i^T z) v_i
\end{aligned}
$$

e. Separating the error term into two components

Our goal is to separate the error term $||x - x_{RR}||_2$ into two components- one that increases and one that decreases with γ.

$$
\begin{aligned}
x - x_{RR} &= x - \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(\sigma_i v_i^T x + u_i^T z)v_i \\
&= \left[x - \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(\sigma_i v_i^T x v_i)\right] + \left[\sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(u_i^T z)v_i\right]
\end{aligned}
$$

Now, note that the v_i's form an orthonormal basis- thus, we can express

$$
x = \sum_{i=1}^{n} v_i^T x v_i
$$

Substituting yields

$$
\begin{aligned}
x - x_{RR} &= \left[\sum_{i=1}^{n} v_i^T x v_i - \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(\sigma_i v_i^T x v_i)\right] + \left[\sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(u_i^T z)v_i\right] \\
&= \left[\sum_{i=1}^{n} \left(1 - \frac{\sigma_i^2}{\sigma_i^2 + \gamma^2}\right)(v_i^T x v_i)\right] + \left[\sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(u_i^T z)v_i\right] \\
&= \underbrace{\left[\sum_{i=1}^{n} \left(\frac{\gamma^2}{\sigma_i^2 + \gamma^2}\right)(v_i^T x v_i)\right]}_{\text{Component 1: Increases with } \gamma} + \underbrace{\left[\sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \gamma^2}(u_i^T z)v_i\right]}_{\text{Component 2: Decreases with } \gamma}
\end{aligned}
$$

f. Value of ridge regression when matrix A is badly conditioned

As shown above, there are two components of the error. If the matrix is badly conditioned, then (holding γ constant) component 2 is expected to be large. In response, we can increase γ to reduce the contribution of component 2 to error. This biases our estimate, but reduces our overall error (up until the point when the error reduction achieved by decreasing component 2 is less than the error augmentation caused by increasing component 1). Because there is this trade off, we should optimize γ through cross-validation (for example).