CS532 Final Project Activity - Solution
Climate Data Fitting and Local Warming Justification

Junda Chen, Haoruo Zhao, Doris Duan

December 14, 2018

Our Philosophy in this activity

1. **Never think that machine learning is THE silver bullet**. Although it is a bit cruel to design such an activity, we tend to feel that people are usually too excited about machine learning and artificial intelligence (just as they were 10 years ago when Big Data is introduced, 20+ years ago when JAVA and Python were introduced). Therefore, we follow the "tricky track" when designing our problem: these problem are prone to err, but not significantly result in the failure to the next step. Students will not be easily getting a **great result** that they would expect from the activity because there is no perfect result in such a complicated system as local warming. We respect the scientific result produced by Robert, and, as a great chance, discourage our student a little bit so that they are more willing to objectively reason about **why my model fail** and **what is behind the model**, not just focus on the technique itself. We think this is a very important skill to master whether the student go to industry, academia, or simply just want to see the beauty of machine learning. It's a great tool, but it is not the tool that determines every fate.

2. **Machine learning is also about empirical knowledge.** One of our friend in the group has been excessively talk about the synergy of machine learning in his field, by injecting the **domain knowledge** into his model and use these knowledge to let the model becomes a "smart model". Many students at this stage don't realize that under the beauty of machine learning is a accumulation of laborious and feature-oriented engineering product that supports most of the model. Empirical knowledge, hence, become very important for human user to develop the model in a sensible way. In this activity, we inject multiple "empirical" choices for student to explore. Although the problems are simple at the first glance, it requires some effort to get the satisfactory answer. We don't want to put so many effort for students to process the data, so we shortened our activity, and put actually the more interesting yet open questions to the optional activity section and allow students to solve the problems with more reasoning upon the science itself. It's a bold decision to shorten the activity and switch a great part of the effort to let the student "unhappy" about their result. We don't know how you feel, but we think it's worth it.

Section 3. Warm-up Solution

Problem 1. Basis functions should include trigonometric functions (e.g. cos(x), sin(x)) that can naturally explain the periodicity of the data. Also, an offset term shoul also include to allow the function to shift up-wards and down-wards. (Alternatively, one who had well-studied Math Analysis would say he wants a big polynomial series to fit the data. However, it is not obvious for researchers to reckon the physical meaning behind. Thus, even if it surely is a set of orthogonal basis, we would not consider that.)

Problem 2. This is actually a pretty tricky question. A common answer should be $\sin(x) + \cos(x)$. But if student observe that the coefficient of the functions are hard-code in matlab, they would not actually derive the answer with the smallest error easily. Although the question itself is tricky, it sort of give both students a perspective of how these function fit the data.

Problem 3. Student should notice that the input is a time series, and hence for each scaler value x, there is a vector output (in this case, a row vector) associated with it. Assume the $\mathbf{x}$ is n-by-1 column vector. Then the matrix is represented as follows:

$$\mathbf{A} = \begin{bmatrix} \sin(2\pi x_1/T_1) & \sin(2\pi x_1/T_2) & \cos(2\pi x_1/T_1) & \cos(2\pi x_1/T_2) \\ \sin(2\pi x_2/T_1) & \sin(2\pi x_2/T_2) & \cos(2\pi x_2/T_1) & \cos(2\pi x_2/T_2) \\ \vdots & \vdots & \vdots & \vdots \\ \sin(2\pi x_n/T_1) & \sin(2\pi x_n/T_1) & \cos(2\pi x_n/T_1) & \cos(2\pi x_n/T_2) \end{bmatrix}$$

If we simplify the x values, we get

$$\mathbf{A} = \begin{bmatrix} \sin(2\pi/T_1) & \sin(2\pi/T_2) & \cos(2\pi/T_1) & \cos(2\pi/T_2) \\ \sin(4\pi/T_1) & \sin(4\pi/T_2) & \cos(4\pi/T_1) & \cos(4\pi/T_2) \\ \vdots & \vdots & \vdots & \vdots \\ \sin(2\pi n/T_1) & \sin(2\pi n/T_1) & \cos(2\pi n/T_1) & \cos(2\pi n/T_2) \end{bmatrix}$$

Problem 4. Don't forget to try our `Cool Model Visualization` and slide those sliders to get a result.

Section 4. Main Activity

Problem 1. In theory, LASSO performs better than Ridge Regression in terms of feature selection. However, if we actually run the code, by default it is not gonna have a significant difference. We select features that are not significantly influence the model as the default value, so you might have a bad feeling for betray your prior knowledge about Ridge Regression and LASSO. Soonner you might notice in the later part of the activity that all T are not equal – some are truely better than others when used as the basis function.

Problem 2. 1. Refer to the code we get:

```
x = 1:length(temperature); % x: the x range
x = x';                    % x: a column vector
y = temperature(x);        % y: the corresponding temperature
T_yr = 365.25;             % T_yr: a year
c = (T_yr/(2*pi));         % c: constant term
t = c * [
     0.50  % Undergrad Final Exam Cycle
     1.00  % Seasonal Cycle
     4.00  % US President Election]
    10.78  % Solar Cycle
    18.60  % Moon Declination angle changing cycle
]';
u = x * (1 ./ t);
o = ones(size(x));

A        = [ sin(u) cos(u) o ];
A = A .* (1 ./ max(A)); % Regularization
```

Notice the expression of `t`. We have that

$$A_0 = \begin{bmatrix} sin(2\pi/(T_{yr} \cdot T_1)) & \dots & sin(2\pi/(T_{yr} \cdot T_5)) & cos(2\pi/(T_{yr} \cdot T_1)) & \dots & cos(2\pi/(T_{yr} \cdot T_5)) & 1 \\ sin(4\pi/(T_{yr} \cdot T_1)) & \dots & sin(4\pi/(T_{yr} \cdot T_5)) & cos(4\pi/(T_{yr} \cdot T_1)) & \dots & cos(4\pi/(T_{yr} \cdot T_5)) & 1 \\ \vdots & \ddots & & & & & \vdots \\ sin(2n\pi/(T_{yr} \cdot T_1)) & \dots & sin(2n\pi/(T_{yr} \cdot T_5)) & cos(2n\pi/(T_{yr} \cdot T_1)) & \dots & cos(2n\pi/(T_{yr} \cdot T_5)) & 1 \end{bmatrix}$$

Since `max` operator will take the maximum of each column, we get a row vector when taking the `max(A)`. The next step regularization

```
A = A .* (1 ./ max(A));
```

yield that

```
(1 ./ max(A)) === [ 1 1 . . . 1 ];
```

since the maximum of each column is 1, the final matrix representation of A is

$$A = A_0$$

which is defined as above.

(why sin/cos also have the value 1? becuase we sort of use the fact the n is sufficiently large and it is easy to come up a value 1 or bunch of values that is closed to 1).

2 3. This is a little open question for you to choose the set of cycle and see the outcome. If you happen to choose a large array of continuous integer, you might have a good time observing something fun; if you happen to stick with the parameters we give, that might be a little bit troublesome. So in question 3 we invite you to choose a large set ot `t` to amerliorate the problem. The code is shown as below

```
c = (T_yr/(2*pi));
t = c * (1:100)'; % Create a 1 - 100 column vector (bigger is possible)
u = x * (1 ./ t);
```

Problem 3. **1.** You wait for a long time, and the result still diverge. Why? See the problem below.

2. Definitely you need to regularize. Otherwise you might wonder why you run so long in the previous question. Since the linear term is extremely large, it requires the corresponding coefficient to remain extremely small so that it is able to converge quick and also remain justified as a feature. So next time when you build a new model you will never forget to consider whether you should consider to regularize your data :-J.

3. **(a)** The code is shown as below:

```
x = 1:length(temperature); % x: the x range
x = x';
y = temperature(x);         % y: the corresponding temperature
T_yr = 365.25;              % T_yr: a year
c = (T_yr/(2*pi));          % c: constant term

t = c * [ 1  10.7  ]'; % <--- TODO

u = x * (1 ./ t);
o = ones(size(x));

A = [ sin(u) cos(u) o x ]              % <--- TODO
% And do you want to regularize A?
A = A * (1 ./ max(A) );

reg_params = [0.1 1 10]     % <--- TODO
wdefault =   [ ones(length(t) * 2,1) * 2000; mean(y); 10; ];      % <--- TODO
for lambda = reg_params
    compare_methods(A, x, y, wdefault, lambda);
end
```

3. **(b)** The linear term is positive, and it appears to have a reasonable value once you get the program right. The result that Robert final get in his paper for the weights of w is

$$w = \begin{bmatrix} 52.6 \\ 9.95 \cdot 10^{-5} \\ -20.4 \\ -8.31 \\ -0.197 \\ 0.211 \\ 0.992 \end{bmatrix}$$

Notice the w is denoted according to the sequence represented in the question. You might have a different sequence to construct w. If you use the construction as above, the final term is the linear term.

A rough calculation getting the linear term $9.95 \cdot 10^{-5}$ yield 3.63^{o} F per century.

Section 5. Optional Activity