Homework 4

EE 263 Stanford University Summer 2019

Due: July 24, 2019

Question **5**, and **6** are **optional** questions (they will not be graded) and you do not need to submit them to gradescope. However, doing them will help improve your understanding and grasp of the material. Q5 is good if you want a simple practice question for application of least squares and Q6 looks at logistic regression and maybe good for students in fields such as ML and such.

1. Fitting a model for hourly temperature. You are given a set of temperature measurements (in degrees C), $y_t \in \mathbb{R}$, $t = 1, \ldots, N$, taken hourly over one week (so $N = 168$). An expert says that over this week, an appropriate model for the hourly temperature is a trend (*i.e.*, a linear function of t) plus a diurnal component (*i.e.*, a 24-periodic component):

$$\hat{y}_t = at + p_t,$$

where $a \in \mathbb{R}$ and $p \in \mathbb{R}^N$ satisfies $p_{t+24} = p_t$, for $t = 1, \ldots, N - 24$. We can interpret a (which has units of degrees C per hour) as the warming or cooling trend (for $a > 0$ or $a < 0$, respectively) over the week.

a) Explain how to find $a \in \mathbb{R}$ and $p \in \mathbb{R}^N$ (which is 24-periodic) that minimize the RMS value of $y - \hat{y}$.

b) Carry out the procedure described in part (a) on the data set found in `tempfit_data.m`.

Give the value of the trend parameter a that you find. Plot the model $\hat{y}$ and the measured temperatures y on the same plot. (The matlab code to do this is in the data file, but commented out.)

c) *Temperature prediction.* Use the model found in part (b) to predict the temperature for the next 24-hour period (*i.e.*, from $t = 169$ to $t = 192$). The file `tempdata.m` also contains a 24 long vector `ytom` with tomorrow's temperatures. Plot tomorrow's temperature and your prediction of it, based on the model found in part (b), on the same plot. What is the RMS value of your prediction error for tomorrow's temperatures?

2. Identifying a system from input/output data. Suppose $y = Ax + v$, where $x \in \mathbb{R}^n$ is the input, $y \in \mathbb{R}^m$ is the output, $A \in \mathbb{R}^{m \times n}$ is the sensor matrix, and $v \in \mathbb{R}^m$ is measurement noise. Suppose we are given N input/output pairs

$$(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)}).$$

a) Explain how to choose A in order to minimize

$$J = \sum_{k=1}^{N} \|Ax^{(k)} - y^{(k)}\|^2.$$

State any assumptions that are needed for your method to work.

b) Apply your method to the data defined in `system_identification_data.m`. Report the average relative approximation error:

$$\frac{1}{N} \sum_{k=1}^{N} \frac{\|Ax^{(k)} - y^{(k)}\|}{\|y^{(k)}\|}.$$

3. Robust regression using the Huber penalty function. The Huber penalty function is

$$H_\delta(d) = \begin{cases} \frac{1}{2}d^2 & |d| \leq \delta, \\ \delta(|d| - \frac{1}{2}\delta) & \text{otherwise}, \end{cases}$$

where $\delta > 0$ is a parameter. Observe that the Huber penalty function is quadratic for small values of d, and linear for large values of d. Thus, the Huber penalty function attempts to combine the sensitivity of the squared-error loss function to small errors, and the robustness of the absolute-error loss function to large errors.

a) Suppose you want to fit a line to given data points $(t_1, x_1), \ldots, (t_N, x_N) \in \mathbb{R}^2$. Explain how to use iteratively reweighted least squares to choose the parameters a and b in order to minimize the total Huber loss:

$$J = \sum_{i=1}^{N} H_\delta(at_i + b - x_i).$$

In particular, what is the weight function, and what is the update equation?

b) Apply your method to the data defined in `huber_penalty_function_data.m` using $\delta = 1$. Report your estimates of the parameters a and b, and the corresponding value of the total Huber loss. Make a plot of the data, the line corresponding to your estimates of a and b, and the line obtained using least-squares. Briefly comment on your results.

4. Estimating a signal with interference. This problem concerns three proposed methods for estimating a signal, based on a measurement that is corrupted by a small noise and also by an interference, that need not be small. We have

$$y = Ax + Bv + w,$$

where $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{m \times p}$ are known. Here $y \in \mathbb{R}^m$ is the measurement (which is known), $x \in \mathbb{R}^n$ is the signal that we want to estimate, $v \in \mathbb{R}^p$ is the interference, and w is a noise. The noise is unknown, and can be assumed to be small. The interference is unknown, but cannot be assumed to be small. You can assume that the matrices A and B are skinny

and full rank (*i.e.*, $m > n$, $m > p$), and that the ranges of A and B intersect only at 0. (If this last condition does not hold, then there is no hope of finding x, even when $w = 0$, since a nonzero interference can masquerade as a signal.) Each of the EE263 TAs proposes a method for estimating x. These methods, along with some informal justification from their proposers, are given below. Nikola proposes the **ignore and estimate method.** He describes it as follows:

> We don't know the interference, so we might as well treat it as noise, and just ignore it during the estimation process. We can use the usual least-squares method, for the model $y = Ax + z$ (with z a noise) to estimate x. (Here we have $z = Bv + w$, but that doesn't matter.)

Almir proposes the **estimate and ignore method.** He describes it as follows:

> We should simultaneously estimate both the signal x *and* the interference v, based on y, using a standard least-squares method to estimate $[x^T \ v^T]^T$ given y. Once we've estimated x and v, we simply ignore our estimate of v, and use our estimate of x.

Miki proposes the **estimate and cancel method.** He describes it as follows:

> Almir's method makes sense to me, but I can improve it. We should simultaneously estimate both the signal x and the interference v, based on y, using a standard least-squares method, exactly as in Almir's method. In Almir's method, we then throw away $\hat{v}$, our estimate of the interference, but I think we should use it. We can form the "pseudo-measurement" $\tilde{y} = y - B\hat{v}$, which is our measurement, with the effect of the estimated interference subtracted off. Then, we use standard least-squares to estimate x from $\tilde{y}$, from the simple model $\tilde{y} = Ax + z$. (This is exactly as in Nikola's method, but here we have subtracted off or cancelled the effect of the estimated interference.)

These descriptions are a little vague; part of the problem is to translate their descriptions into more precise algorithms.

a) Give an explicit formula for each of the three estimates. (That is, for each method give a formula for the estimate $\hat{x}$ in terms of A, B, y, and the dimensions n, m, p.)

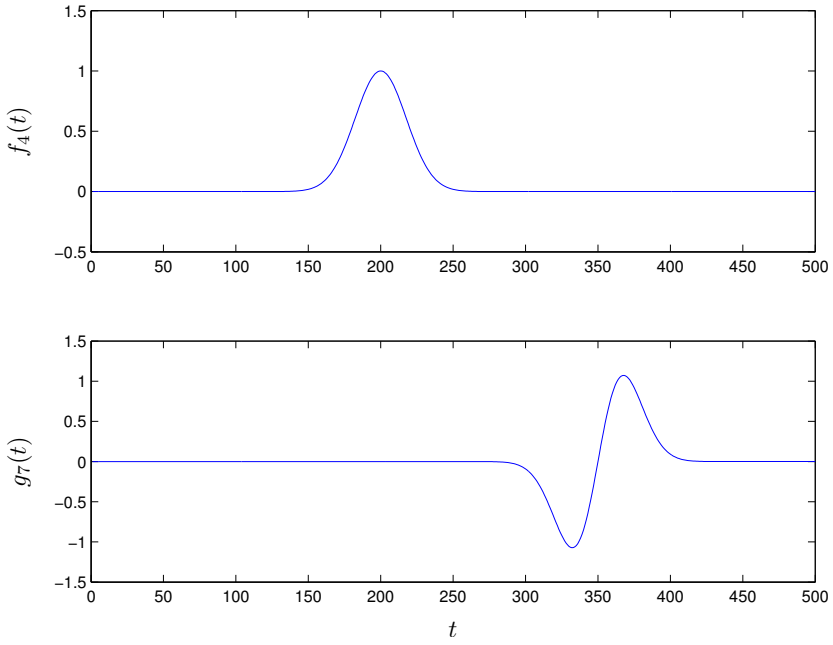
b) Are the methods really different? Identify any pairs of the methods that coincide (*i.e.*, always give exactly the same results). If they are all three the same, or all three different, say so. Justify your answer. To show two methods are the same, show that the formulas given in part (a) are equal (even if they don't appear to be at first). To show two methods are different, give a specific numerical example in which the estimates differ.

c) Which method or methods do you think work best? Give a very brief explanation. (If your answer to part (b) is "The methods are all the same" then you can simply repeat here, "The methods are all the same".)

5. Signal estimation using least-squares. This problem concerns discrete-time signals defined for $t = 1, \ldots, 500$. We'll represent these signals by vectors in $\mathbb{R}^{500}$, with the index corresponding to the time. We are given a noisy measurement $y_{\text{meas}}(1), \ldots, y_{\text{meas}}(500)$, of a signal $y(1), \ldots, y(500)$ that is thought to be, at least approximately, a linear combination of the 22 signals

$$f_k(t) = e^{-(t-50k)^2/25^2}, \qquad g_k(t) = \left(\frac{t-50k}{10}\right) e^{-(t-50k)^2/25^2},$$

where $t = 1, \ldots, 500$ and $k = 0, \ldots, 10$. Plots of f_4 and g_7 (as examples) are shown below.



As our estimate of the original signal, we will use the signal $\hat{y} = (\hat{y}(1), \ldots, \hat{y}(500))$ in the span of $f_0, \ldots, f_{10}, g_0, \ldots, g_{10}$, that is closest to $y_{\text{meas}} = (y_{\text{meas}}(1), \ldots, y_{\text{meas}}(500))$ in the RMS (root-mean-square) sense. Explain how to find $\hat{y}$, and carry out your method on the signal y_{meas} given in `sig_est_data.m` on the course web site. Plot y_{meas} and $\hat{y}$ on the same graph. Plot the residual (the difference between these two signals) on a different graph, and give its RMS value.

6. Logistic regression. Consider a data set in which each observation belongs to exactly one of two classes, labeled 0 and 1. In particular, suppose we are given observations $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{0, 1\}$, where $x_k \in \mathbb{R}^n$ is a vector of attributes describing the kth observation, and $y_k \in \{0, 1\}$ is a label indicating whether the kth observation belongs to class 0 or class 1. We can represent the data set compactly using the matrix

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_m^T \end{bmatrix} \in \mathbb{R}^{m \times n},$$

and the vector $y = (y_1, \ldots, y_m) \in \{0,1\}^m$. In a logistic regression model, we assume that

$$\text{Prob}(y = 1 \,|\, x) = \phi(\beta^\mathsf{T} x),$$

where $\phi : \mathbb{R} \to [0,1]$ is the logistic function:

$$\phi(z) = \frac{1}{1 + \exp(-z)},$$

and $\beta \in \mathbb{R}^n$ is a vector of parameters. In other words, the conditional probability that an observation belongs to class 1 given the vector of attributes describing the observation is equal to $\phi(\beta^\mathsf{T} x)$. A plot of the logistic function is given in **??**.

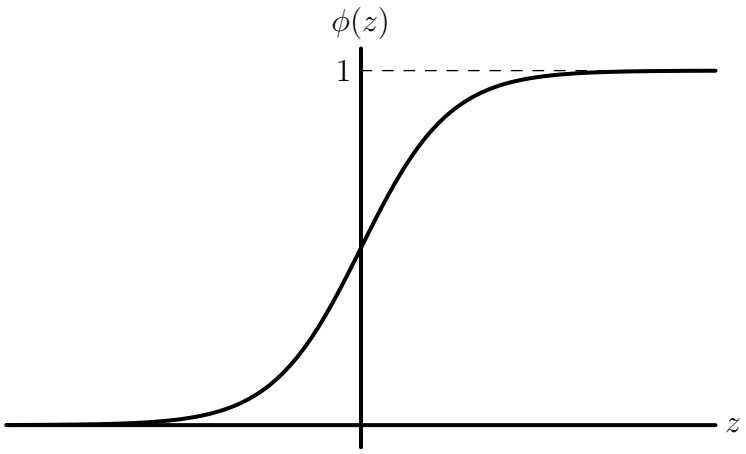


Figure 1: the logistic function

Because probabilities must lie between 0 and 1, it makes sense to assume that $\text{Prob}(y = 1 \,|\, x)$ is equal to $\phi(\beta^\mathsf{T} x)$ rather than $\beta^\mathsf{T} x$. An equivalent way of formulating a logistic regression model is

$$\phi^{-1}(\text{Prob}(y = 1 \,|\, x)) = \log\left(\frac{\text{Prob}(y = 1 \,|\, x)}{1 - \text{Prob}(y = 1 \,|\, x)}\right) = \beta^\mathsf{T} x.$$

Thus, we assume that the quantity $\phi^{-1}(\text{Prob}(y = 1 \,|\, x))$ (which is called the log-odds) is a linear function of the vector x of attributes.

It is common to choose the vector β of parameters in order to maximize the likelihood of the model given the data. This is equivalent to minimizing

$$J(\beta) = -\sum_{k=1}^{m} \Big(y_k \log(\phi(\beta^\mathsf{T} x_k)) + (1 - y_k) \log(1 - \phi(\beta^\mathsf{T} x_k)) \Big).$$

In this problem you will use Newton's method to minimize $J(\beta)$. In order to solve the problem, you do not need to understand what the likelihood of a model is, or why it is a good thing to maximize – we only provide this information for motivation.

a) Show that

$$\phi'(z) = \phi(z)(1 - \phi(z)), \quad \nabla J(\beta) = X^\mathsf{T}(p - y), \quad \text{and} \quad \nabla^2 J(\beta) = X^\mathsf{T} W X,$$

where we define the vector $p \in \mathbb{R}^m$ and the matrix $W \in \mathbb{R}^{m \times m}$ such that

$$p = \begin{bmatrix} \phi(\beta^\mathsf{T} x_1) \\ \vdots \\ \phi(\beta^\mathsf{T} x_m) \end{bmatrix} \qquad \text{and} \qquad W = \begin{bmatrix} p_1(1-p_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & p_m(1-p_m) \end{bmatrix}.$$

b) Show that the update equation for Newton's method can be written as

$$\beta^{(\ell+1)} = (X^\mathsf{T} W^{(\ell)} X)^{-1} X^\mathsf{T} W^{(\ell)} z^{(\ell)},$$

where $p^{(\ell)}$ and $W^{(\ell)}$ are the values of p and W corresponding to the parameter vector $\beta^{(\ell)}$, and

$$z^{(\ell)} = X\beta^{(\ell)} - (W^{(\ell)})^{-1}(p^{(\ell)} - y)$$

is called the adjusted response. Thus, when used to estimate the parameters in a logistic regression model, we can think of Newton's method as a form of iteratively reweighted least-squares: the weight assigned to the kth observation is $p_k(1-p_k)$, which is small when we are relatively confident about our classification (that is, p_k is close to either 0 or 1), and large when we are not very confident about our classification (that is, p_k is close to $\frac{1}{2}$). This weighting scheme allows us to focus on the observations that are difficult to classify.

c) The file `logistic_regression_data.m` defines the following variables.

- `m` and `n`, the number of observations and the length of the attribute vector, respectively
- `y`, a vector of length m whose kth component is the label for the kth observation
- `X`, an $m \times n$ matrix whose kth row is the attribute vector for the kth observation

Use Newton's method to fit a logistic regression model to the data; use $\beta^{(0)} = 0$ for your initial guess. Report your final vector of parameter estimates, and the corresponding value of $J(\beta)$.

Note that the third component of attribute vector is a constant; make a scatter plot of (x_{k1}, x_{k2}), and add the classification boundary to your plot (the classification boundary is the line such that $\text{Prob}(y = 1 \,|\, x) = \frac{1}{2}$). In your scatter plot, indicate each of the observations from class 0 with a blue circle, and each of the observations from class 1 with a red square.

d) Use your fitted model to classify the data: that is, evaluate the estimated probability that each observation belongs to class 1; assign an observation to class 1 if this probability is greater than or equal to $\frac{1}{2}$. What is the misclassification rate of the fitted model (that is, the percentage of observations assigned to the wrong class)? Additionally, report $\text{Prob}(y = 1 \,|\, x_k)$ for $k = 12$.