# CSE 512: Homework 1 Due Monday, Sept. 14

1. *Bayes' rule.*

   (a) I look out of my window and see 10 people in the street. 8 have umbrellas, 2 do not. None of these people know each other; they make decisions independently.

   In general, the area is very windy, so people do not always use an umbrella when it rains. But each day has a 50% chance of getting rain. In fact, even when it rains, people only use an umbrella 75% of the time. When it's not raining, people sometimes use an umbrella as a parasol, 5% of the time.

   i. What is the probability that it is raining, given that I am using an umbrella?

   ii. What is the probability that it is raining today, given my observation?

   (b) According to the CDC[1], the current percentage of positive COVID tests in the US is 9.1%. Let's use this as a marker for $\mathbf{Pr}(\text{COVID}|\text{took a test})$.

   There are many available tests, each with different performance metrics. For example, if I take a combined IgG/IgM seriology test, then for one company [2], the PPV (the chance of a positive test when COVID is present) is estimated at 82.5%, and the NPV (the chance of a negative test when COVID is not present) is estimated at 99.9%.

   i. If a person goes in for a test and gets a positive result, what are the chances that that person has COVID?

   ii. If a person goes in for a test and gets a negative result, what are the chances that that person does not have COVID?

2. *Logistic regression for Binary MNIST*

   (a) For a twice-differentiable function $f : \mathbb{R}^d \to \mathbb{R}$, the *gradient* and *Hessian* are defined as

$$\nabla f(\theta) = \begin{bmatrix} \frac{\partial f}{\partial \theta_1} \\ \frac{\partial f}{\partial \theta_2} \\ \vdots \\ \frac{\partial f}{\partial \theta_d} \end{bmatrix} \in \mathbb{R}^d, \qquad \nabla^2 f(\theta) = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_d} \\ \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_2^2} & \cdots & \frac{\partial^2 f}{\partial \theta_2 \partial \theta_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial \theta_d \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_d \partial \theta_2} & \cdots & \frac{\partial^2 f}{\partial \theta_d^2} \end{bmatrix} \in \mathbb{R}^{d \times d}.$$

   For example, for the function $f(\theta) = \theta_1^2 + 2\theta_1\theta_2 + \theta_3^3$, the gradient and Hessian are

$$\nabla f(\theta) = \begin{bmatrix} 2\theta_1 + 2\theta_2 \\ 2\theta_1 \\ 3\theta_3^2 \end{bmatrix}, \qquad \nabla^2 f(\theta) = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 6\theta_3 \end{bmatrix}.$$

   What is the gradient and Hessian of the logistic loss function

$$\mathcal{L}(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \log(\sigma(y_i x_i^T \theta)), \quad \sigma(s) = \frac{1}{1+e^{-s}}$$

   where $y_i \in \{-1, 1\}$?

   (b) **Coding.**

   - Download mnist.mat [2]. We will use logistic regression to diffrentiate 4's from 9's, a notoriously tricky problem. If you are using python, you can use `scipy.io.loadmat` to read the matrix. If you are using MATLAB, just `load mnist.mat` should suffice.

[1] `https://www.cdc.gov/coronavirus/2019-ncov/covid-data/covidview/index.html`

[2] `https://www.fda.gov/medical-devices/coronavirus-disease-2019-covid-19-emergency-use-authorizations-medical-devices/eua-authorized-serology-test-performance`

- The matrices $X$ and $y$ should contain the vectorized images and corresponding labels. To take a look at how the data is stored, run the following code:

  Python

```
data = sio.loadmat('mnist.mat')
for k in xrange(9):
    plt.subplot(3,3,k+1)
    plt.imshow(np.reshape(data['trainX'][k,:],(28,28)))
    plt.title(data['trainY'][0,k])
plt.tight_layout()
```

  Matlab

```
load mnist.mat
for k = 1:9:
    subplot(3,3,k)
    imshow(reshape(trainX(k,:),28,28))
    title(trainY(k))
end
```

- Select only the data rows corresponding to the labels 4 and 9, and set the remaining labels to be binary.

  Python

```
idx = np.logical_or(np.equal(y,4) ,  np.equal(y,9))
X = X[idx,:]
y = y[idx]
y[np.equal(y,4)] = -1
y[np.equal(y,9)] = 1
```

  Matlab

```
idx = (y == 4) || (y == 9)
X = X(idx,:)
y = y(idx)
y(y==4) = -1
y(y==9)= 1
```

  You should be left with 11791 train images and 1991 test images. Make sure they are stored separately, e.g. $X$ = train images, $X_t$ = test images.

- Normalize the data matrix by first rescaling the pixel values to all be between 0 and 1 (effectively, divide by 255), and then translating all the values so that the sum of all the images in the *train set* is 0. To do this, first compute the average pixel for the training set, e.g.

$$x_{\mathbf{mean}}[k] = \frac{1}{m_{\text{train}}} \sum_{i=1}^{m_{\text{train}}} x_i^{\text{train}}[k], \qquad x_{\mathbf{mean}} \in \mathbb{R}^{784}.$$

  Then translate both the test and train data using this offset:

$$x_i^{\text{train}} \leftarrow x_i^{\text{train}} - x_{\mathbf{mean}}$$

$$x_i^{\text{test}} \leftarrow x_i^{\text{test}} - x_{\mathbf{mean}}$$

  [3]

- Use gradient descent to minimize the logistic loss for this classification problem. Use a step size of 0.001, and run for 5000 iterations. Plot the train / test loss, and train/test misclassification rate, and also report these final values.
- Comment a bit on what you see.

3. Using the properties of norms, verify that the following are norms, or prove that they are not norms

[3] There are other ways to normalize, but this is a reasonable one I have found works in practice, and in the interest of "normalizing" the assignment, it's what we'll go with.

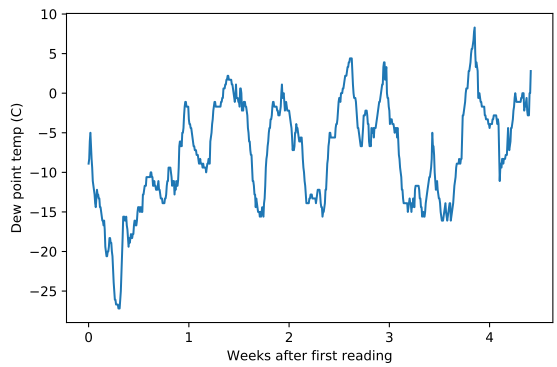
(a) *Direct sum.* $f : \mathbb{R}^d \to \mathbb{R}$, $f(x) = \sum_k x[k]$

(b) *Sum of square roots, squared.* $f : \mathbb{R}^d \to \mathbb{R}$, $f(x) = \left(\sum_{k=1}^d \sqrt{|x[k]|}\right)^2$

(c) *Weighted 2-norm.* $f : \mathbb{R}^d \to \mathbb{R}$, $f(x) = \sqrt{\sum_{k=1}^d \frac{|x[k]|^2}{k}}$

4. *Polyfit via linear regression.*

(a) Download weatherDewTmp.mat. Plot the data (`plot(weeks,dew)`). It should look like the following




(b) We want to form a polynomial regression of this data. That is, given $w$ = weeks and $d$ = dew readings, we want to find $\theta_1, ..., \theta_p$ as the solution to

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2}\sum_{i=1}^{m}(\theta_1 + \theta_2 w_i + \theta_3 w_i^2 + \cdots + \theta_p w_i^{p-1} - d_i)^2. \tag{1}$$

Find $X$ and $y$ such that (1) is equivalent to the least squares problem

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2}\|X\theta - y\|_2^2. \tag{2}$$

(c) What are the normal equations for problem (2)? In particular, if $\theta^*$ is the minimizer of (2), then $\theta^*$ solves a linear system $A\theta^* = b$. What are $A$ and $b$?

(d) *Ridge regression* Oftentimes, it is helpful to add a *regularization term* to (2), to improve stability. This also has an interpretation as Bayesian linear regression with a Gaussian 0-mean prior. In other words, we solve

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\alpha}{2}\|\theta\|_2^2. \tag{3}$$

for some $\alpha > 0$. The $\theta^*$ that minimizes (3) is the solution to a different linear system $A_{\text{reg}}\theta^* = b_{\text{reg}}$. What are $A_{\text{reg}}$ and $b_{\text{reg}}$?

(e) If $A$ is a positive semidefinite matrix with condition number 5 and largest eigenvalue 1, what is the condition number of $A + \alpha I$ for some $\alpha > 0$?

(f) In MATLAB or Python, write a function that takes as argument $p$ and returns $X$ and $y$ so that (1) is equivalent to (2). Report the *condition numbers* for $A$ and $A_{\text{reg}}$ by filling out this table:

| p | $A$ ($\alpha = 0$) | $A_{\text{reg}}$, $\alpha = 0.1 \cdot m$ | $A_{\text{reg}}$, $\alpha = m$ | $A_{\text{reg}}$, $\alpha = 10 \cdot m$ | $A_{\text{reg}}$, $\alpha = 100 \cdot m$ |
|---|---|---|---|---|---|
| 2 | | | | | |
| 3 | | | | | |
| 6 | | | | | |
| 11 | | | | | |

(g) Compute a polynomial fit by solving (2) for polynomials of order $1, 2, 10, 100, 150$, and $200$. Plot all the fits on separate plots (use `subplot`). Comment on your observations.

(h) Now compute a *regularized* polynomial fit by solving (3) for polynomials of order $1, 2, 10, 100, 150$, and $200$, for $\alpha = 0.0001$. Plot all the fits on separate plots (use `subplot`). Comment on your observations. How does this compare to the unregularized polynomial fit?

(i) Picking your favorite set of hyperparameters ($p$, $\alpha$), forecast the next week's dew point temperature. Plot the forecasted data over the current observations. Do you believe your forecast? Why?

## Challenge!

In this problem we will investigate a *sparse* regularizer, in which we replace the 2-norm regularizer with a 1-norm regularizer. In other words, given $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $\lambda \in \mathbb{R}$, we will solve

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \tfrac{1}{2m}\|Ax-b\|_2^2 + \lambda\|x\|_1 \tag{4}$$

1. This objective function is composed of a smooth (everywhere differentiable) and nonsmooth (not everywhere differentiable) term. Show that $\|x\|_1$ is nonsmooth by describing all the points $x$ where $g(x) = \|x\|_1$ is not differentiable.
2. Because the objective has a nonsmooth point, gradient descent will not converge to the global minimum. To see that this is true, consider the case of $m = n = 1$, with $A = b = 1$, $\lambda = 2$. In other words, we consider

   $$\underset{x}{\text{minimize}} \quad \frac{1}{2}(x-1)^2 + 2|x|. \tag{5}$$

   Start with $x^{(0)} = 1$, and with a step size $t = 1$, write out the iterates $x^{(k)}$ for $k = 1, 2, 3, 4$. Is there a limit point $\lim_{k\to+\infty} x^{(k)}$? If so, is this limit point the problem's global minima?
3. We therefore will introduce a new method called the *proximal gradient descent* method. This method is similar to gradient descent, except we break away the nonsmooth term and deal with it separately. Explicitly, for solving

   $$\underset{x}{\text{minimize}} \quad f(x) + g(x)$$

   where $f(x)$ is smooth and $g(x)$ is nonsmooth, the proximal gradient descent method picks a random point $x^{(0)}$ and iterates

   $$x^{(k+1)} = \mathbf{Pr}ox_{tg}(x^{(k)} - t\nabla f(x^{(t)}))$$

   where the mapping $\mathbf{Pr}ox_{tg}$ is the *proximal operator*

   $$\mathbf{Pr}ox_{tg}(z) = \underset{x}{\text{argmin}}\ g(x) + \frac{1}{2t}\|x-z\|_2^2.$$

   We can interpret this as finding the variable $x$ that trades off minimizing the nonsmooth term $g(x)$ and a proximity term (e.g. doesn't want to deviate too far from $z$).

   Show that the proximal operator of the 1-norm can be computed in closed form, as

   $$\mathbf{Pr}ox_{tg}(z) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \qquad x_k = \begin{cases} (|z_k| - t)\mathbf{sign}(z_k) & \text{if } |z_k| > t \\ 0 & \text{else.} \end{cases}$$

   This operator is called the "shrinkage operator".
4. Again consider the scalar problem (5). Start with $x^{(0)} = 2$, and with a step size $t = 1/2$, write out the iterates $x^{(k)}$ for $k = 1, 2, 3$, but following the proximal gradient scheme. What is the limit point $\lim_{k\to+\infty} x^{(k)}$? Is this limit point the problem's global minima?
5. **Coding.** In MATLAB or Python, generate a sample problem with `A = randn(m,n)` and `b = randn(m,1)`. Pick $m = 100$, $n = 1000$. Solve (4). For $\lambda = 0, 0.001, 0.1$, show histograms of the elements of $x^*$. Comment on the sparsifying property of the 1-norm.

   **Comparison with 2-norm regularization.** We can also consider a 2-norm regularized version as well, where we solve

   $$\underset{x}{\text{minimize}} \quad \frac{1}{2m}\|Ax-b\|_2^2 + \lambda\|x\|_2 \tag{6}$$

6. Show that this regularization term $\|x\|_2$ is also nonsmooth.
7. Derive the proximal operator $\mathbf{Pr}ox_{tg}$ for $g(x) = \|x\|_2$.
8. Use proximal gradient descent to solve (6), using the same choices of $A$ and $b$ as in the previous section. Histogram the final solutions $x^*$ for $\lambda = 0, 1, 2, 5, 10$. Comment on the sparsifying properties of the 2-norm vs the 1-norm.

# References


[1] A. Beck, *Introduction to nonlinear optimization: theory, algorithms, and applications with MATLAB*, vol. 19, Siam, 2014.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, vol. 86, no. 11 (1998): 2278-2324.