CSE 546 Homework #4
Fall 2016

Janet Matsen

December 12, 2016

For this assignment, I went to office hours and developed strategies with Serena Liu and David Flemming.

My code is available at `https://github.com/JanetMatsen/Machine_Learning_CSE_546/tree/master/HW4`

1 Manual calculation of one round of EM for a GMM [15 points]

In this question we consider clustering 1D data with a mixture of 2 Gaussians using the EM algorithm. You are given the 1-D data points $x = [1\ \ 10\ \ 20]$.

Note: I previously solved this question in CSE 446.

1.1 M step [10 points]

Suppose the output of the E step is the following matrix:

$$R = \begin{pmatrix} 1 & 0 \\ 0.4 & 0.6 \\ 0 & 1 \end{pmatrix}$$

where entry $R_{i,c}$ is the probability of observation x_i belonging to cluster c (the responsibility of cluster c for data point i). You just have to compute the M step. You may state the equations for maximum likelihood estimates of these quantities (which you should know) without proof; you just have to apply the equations to this data set. You may leave your answer in fractional form. Show your work.

1.1.1 *[2 points]* Write down the likelihood function you are trying to optimize.

The general form for the M step of EM is:

$$\theta = \text{argmax}_\theta \sum_j \sum_{i=1}^{k} P(y^j = i | x^j, \theta) \log P(y^j = i, x^j | \theta)$$

Where the right hand term is describing the probability of generating the point x^j from the ith Gaussian cluster.

So the entire likelihood of the M step is:

$$\theta = \text{argmax}_\theta \sum_j \sum_{i=1}^{k} P(y^j = i | x^j, \theta) \log P(x^j | \mu_i, \sigma_i)$$

$P(x^j | \mu_i, \sigma_i)$ is the probability of x^j under the Gaussian likelihood with parameters μ_i, σ_i.

Notation:
x^j represents data point j is an index over the data. We have 3 j values for x^1, x^2, and x^3.
y^i represents cluster i. i is an index over the clusters, so it takes 2 values.
θ is all parameters in the model, which includes $\pi_1, \pi_2, \mu_1, \mu_2, \sigma_1, \sigma_2$.

1.1.2 ***[2 points]*** **After performing the M step for the mixing weights π_1, π_2, what are the new values?**

In CSE 446, we defined $\pi_i = \frac{1}{m}\sum_{j=1}^{m} P(y = i|x^j; \theta_t)$.

$$R = \begin{pmatrix} 1 & 0 \\ 0.4 & 0.6 \\ 0 & 1 \end{pmatrix}$$

So $\pi_1 = \frac{1}{3}(1 + 0.4 + 0) = 1.4/3 = 0.4667$.
And $\pi_2 = \frac{1}{3}(0 + 0.6 + 1) = 1.6/3 = 0.533$.
These sum to 1 as expected.

1.1.3 ***[3 points]*** **After performing the M step for the means μ_1 and μ_2, what are the new values?**

i	j	x^j	$P(y = i\|x^j, \theta)$	$P(y^j = i\|x^j, \theta)$
1	1	1	$P(y = 1\|x^1, \theta)$	1
1	2	10	$P(y = 1\|x^2, \theta)$	0.4
1	3	20	$P(y = 1\|x^3, \theta)$	0
2	1	1	$P(y = 2\|x^1, \theta)$	0
2	2	10	$P(y = 2\|x^2, \theta)$	0.6
2	3	20	$P(y = 2\|x^3, \theta)$	1

$$\mu_i = \frac{\sum_{j=1}^{m} p(y = i|x^j; \theta_t)x^j}{\sum_{j=1}^{m} p(y = i|x^j; \theta_t)}$$

Calculate μ_1:

$$\begin{aligned}
\mu_1 &= \frac{\sum_{j=1}^{m} p(y = 1|x^j; \theta_t)x^j}{\sum_{j=1}^{m} p(y = 1|x^j; \theta_t)} \\
&= \frac{p(y = 1|x^1; \theta_t)x^1 + p(y = 1|x^2; \theta_t)x^2 + p(y = 1|x^3; \theta_t)x^3}{p(y = 1|x^1; \theta_t) + p(y = 1|x^2; \theta_t) + p(y = 1|x^3; \theta_t)} \\
&= \frac{1 \cdot 1 + 0.4 \cdot 10 + 0 \cdot 20}{1 + 0.4 + 0} \\
&= \frac{1 + 4}{1.4} \\
&= 5/1.4 \\
&= 25/7 = 3.57
\end{aligned}$$

Calculate μ_2:

$$\begin{aligned}
\mu_2 &= \frac{\sum_{j=1}^{m} p(y=2|x^j;\theta_t)x^j}{\sum_{j=1}^{m} p(y=2|x^j;\theta_t)} \\
&= \frac{p(y=2|x^1;\theta_t)x^1 + p(y=2|x^2;\theta_t)x^2 + p(y=2|x^3;\theta_t)x^3}{p(y=2|x^1;\theta_t) + p(y=2|x^2;\theta_t) + p(y=2|x^3;\theta_t)} \\
&= \frac{0\cdot 1 + 0.6\cdot 10 + 1\cdot 20}{0 + 0.6 + 1} \\
&= \frac{0 + 6 + 20}{0.6 + 1} \\
&= 26/1.6 \\
&= 64/4 = 16.25
\end{aligned}$$

1.1.4 *[3 points]* After performing the M step for the standard deviations σ_1 and σ_2, what are the new values?

From CSE 446:

$$\Sigma_i = \frac{\sum_{j=1}^{m} p(y=i|x^j;\theta_t)(x^j-\mu_i)(x^j-\mu_i)^T}{\sum_{j=1}^{m} p(y=i|x^j;\theta_t)}$$

Drop the $|\theta_t$ notation for convenience.

$$\begin{aligned}
\Sigma_1 &= \frac{p(y=1|x^1)(x^1-3.57)(x^1-3.57) + p(y=1|x^2)(x^2-3.57)(x^2-3.57) + p(y=1|x^3)(x^3-3.57)(x^2-}{p(y=1|x^1) + p(y=1|x^2) + p(y=1|x^3)} \\
&= \frac{1(1-3.57)(1-3.57) + 0.4(10-3.57)(10-3.57) + 0(20-3.57)(20-3.57)}{0 + 0.4 + 1} \\
&= \frac{(-2.57)(-2.57) + 0.4(6.43)(6.43)}{1.4} \\
&= 16.53
\end{aligned}$$

$$\begin{aligned}
\Sigma_2 &= \frac{p(y=2|x^1)(x^1-16.25)(x^1-16.25) + p(y=2|x^2)(x^2-16.25)(x^j-16.25) + p(y=2|x^3)(x^3-16.25)}{p(y=2|x^1) + p(y=2|x^2) + p(y=2|x^3)} \\
&= \frac{0(1-16.25)(1-16.25) + 0.6(10-16.25)(10-16.25) + 1(20-16.25)(20-16.25)}{0 + 0.6 + 1} \\
&= \frac{0 + 0.6(-6.25)(-6.25) + (3.75)(3.75)}{1.6} \\
&= \frac{37.5}{1.6} \\
&= 23.44
\end{aligned}$$

E step [5 points]

Suppose the output of the M step is your previous answer. Let us compute the subsequent E step.

1.1.5 *[5 points]* Write down the formula for the probability of observation x_i belonging to cluster c.

$$P(y = i|x^j; \theta_t) \propto \frac{1}{(2\pi)^m |\Sigma_i|} \exp\left(-\frac{1}{2}(x^j - \mu_i)^T \Sigma_i^{-1}(x^j - \mu_i)\right)$$

In CSE 446 we saw that for 1D data this is more simply expressed as:

$$P(y = i|x^j; \theta_t) \propto \exp\left(-\frac{1}{2\sigma^2}(x^j - \mu_j)^2\right)$$

1.1.6 *[3 points]* After performing the E step, what is the new value of R?

i	j	x^j	P $\propto$	$P(y^j = i\|x^j, \theta)$
1	1	1	$\exp\left(-\frac{1}{2 \cdot 16.53}(1 - 3.57)^2\right)$	0.99
1	2	10	$\exp\left(-\frac{1}{2 \cdot 16.53}(10 - 3.57)^2\right)$	0.40
1	3	20	$\exp\left(-\frac{1}{2 \cdot 16.53}(20 - 3.57)^2\right)$	0.00
2	1	1	$\exp\left(-\frac{1}{2 \cdot 23.44}(1 - 16.25)^2\right)$	0.01
2	2	10	$\exp\left(-\frac{1}{2 \cdot 23.44}(10 - 16.25)^2\right)$	0.60
2	3	20	$\exp\left(-\frac{1}{2 \cdot 23.44}(20 - 16.25)^2\right)$	1.00

Screenshot of calculations:

j	i	x^j	mu_i	var_i	sigma_i	-1/(2*sigma^2)	(x^j - mu)^2	exp(-1/(2*sigma^2) (x - mu)^2)	P (normalized!)
1	1	1	3.57	16.53	4.0657	-0.030248034	6.6049	0.818906603	0.99
1	2	1	16.25	23.44	4.8415	-0.021331058	232.5625	0.00700729	0.01
2	1	10	3.57	16.53	4.0657	-0.030248034	41.3449	0.286332391	0.40
2	2	10	16.25	23.44	4.8415	-0.021331058	39.0625	0.434636837	0.60
3	1	20	3.57	16.53	4.0657	-0.030248034	269.9449	0.000284351	0.00
3	2	20	16.25	23.44	4.8415	-0.021331058	14.0625	0.740841925	1.00

$$\text{R} = \begin{pmatrix} 0.99 & 0.01 \\ 0.40 & 0.60 \\ 0.00 & 1.00 \end{pmatrix}$$

2 Neural Nets and Backprop [45 points]

2.1 With sigmoid hidden units [15 points]

The input layer should have 50 dimensions (as the image is 50 dimensions, after PCA). Let us use 500 nodes for the hidden layer, with a sigmoid transfer function. The output layer can simply consist of the predictions of the network. Let us make the output nodes to be linear in the hidden layer.

2.1.1 (3 points) Specify all your parameter choices: your learning rate (or learning rate scheme), mini-batch size, initialization scheme.

- **learning rate:** 0.0005. This is normalized by the minibatch size at each update.
- **mini-batch size:** 10
- **learning rate decay:** none
- **epochs:** 11
- **W1 initialization scheme:** Xavier/1e8. Weights were normal with magnitude around $1e-9$
- **W2 initialization scheme:** sampled from np.random.normal(0, 1, size=(self.n_nodes, self.n_in)), then divided by $1e8$. Magnitudes were on order of $1e-9$.

2.1.2 (6 points) Plot the squared loss after every half epoch (starting with your initial squared error).

Please label your axes in a readable way. Plot the loss of both the training and test losses on the same plot. For the 0/1 loss, do the same, except start your plots a little later (e.g. when the 0/1 loss is below 7%) so they are more readable.

Figure 1: Tanh hidden, linear output. Squared loss after every half epoch

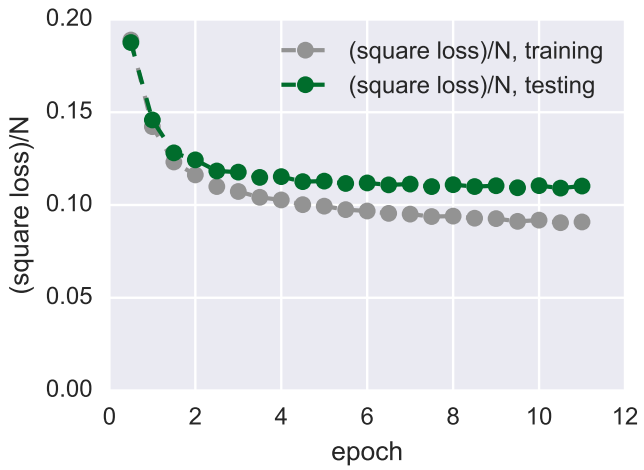
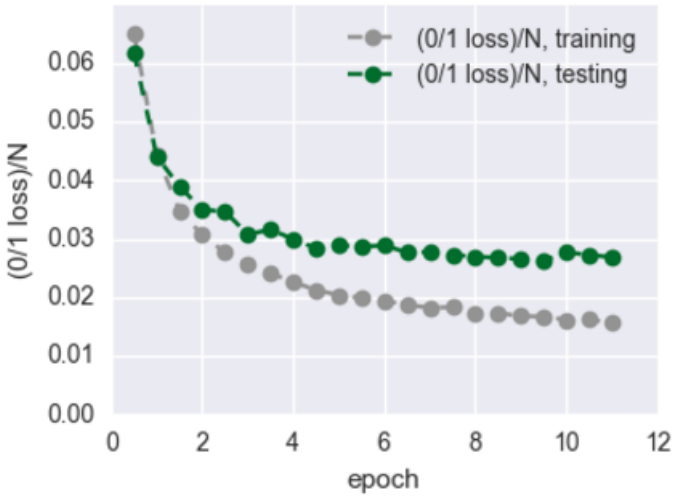


Figure 2: Tanh hidden, linear output. 0/1 loss after every half epoch



2.1.3 (6 points) What is your final squared loss and 0/1 loss for both the training and test sets?

Data Set	squared loss	(squared loss)/N	0/1 loss	(0/1 loss)/N
Training	5449.77	0.0908295	941	0.0156833
Test	1097.84	0.109784	269	0.0269

Table 1: Question 2.1: Sigmoid with hidden units

2.2 (3 points) Choose 10 hidden layer nodes at random and display the learned weights (these are the 50 dimensional weights, visualized back into image space).

Figure 3: Tanh hidden, linear output. 10 random weight vectors.

While I expected to see non-interpretable weights, I didn't expect them to look like lovely little flowers.

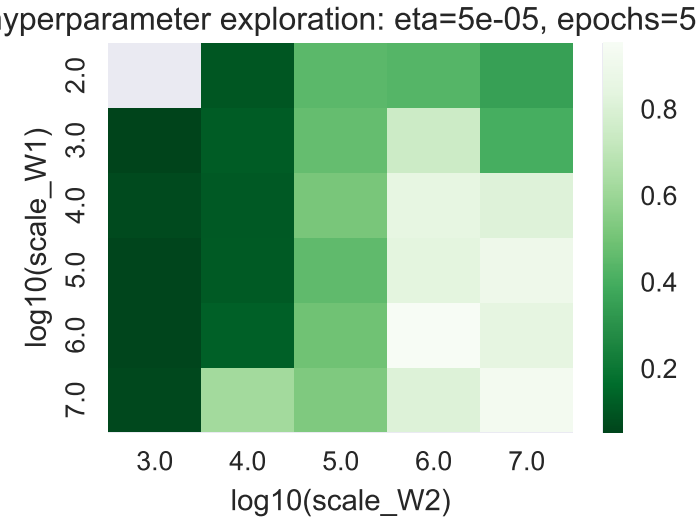
2.3 With ReLu hidden units [15 points]

The input layer should have 50 dimensions (as the image is 50 dimensions, after PCA). Let us use 500 nodes for the hidden layer, with a rectified linear (ReLu) transfer function. The output layer can simply consist of the predictions of the network. Let us make the output nodes to be linear in the hidden layer.

2.3.1 (3 points) Specify all your parameter choices: your learning rate (or learning rate scheme), mini-batch size, initialization scheme.

Motivated by observing ReLu's sensitivity to the scale of the initial weights, I explored the effect of scaling my initial weights by orders of magnitude. For this process, I held the learning rate, learning rate decay scheme, epochs, and weight initialization scheme constant. The only thing that varied was dividing the initial weights, W1 and W2 by different constants. This got me in the right neighborhood of a good combination of learning rates and initial matrix weights.

Figure 4: Sample hyperparameter exploration for hidden ReLu nodes and Linear output nodes. Each rectangle represents the 0/1 test loss given different initial weight scaling. Dark green → good fit. Gray → model not run or exception raised due to fit instability



- **learning rate:** $1e-4$
- **learning rate decay:** $1/epochs^{0.5}$
- **epochs:** 9
- **W1 initialization scheme:** Sample each element from $\mathrm{N}(0, ||X||_2^2)$; divide by $10**5$ to prevent weights from blowing up.
- **W2 initialization scheme:** Sample each element from $\mathrm{N}(0, 1)$; divide by $10**3$ to prevent weights from blowing up.

2.3.2 (6 points) Plot the squared loss after every half epoch (starting with your initial squared error).

Please label your axes in a readable way. Plot the loss of both the training and test losses on the same plot. For the 0/1 loss, do the same, except start your plots a little later (e.g. when the 0/1 loss is below 7%) so they are more readable.

Figure 5: Square loss during hidden = ReLu, output = linear neural net training.

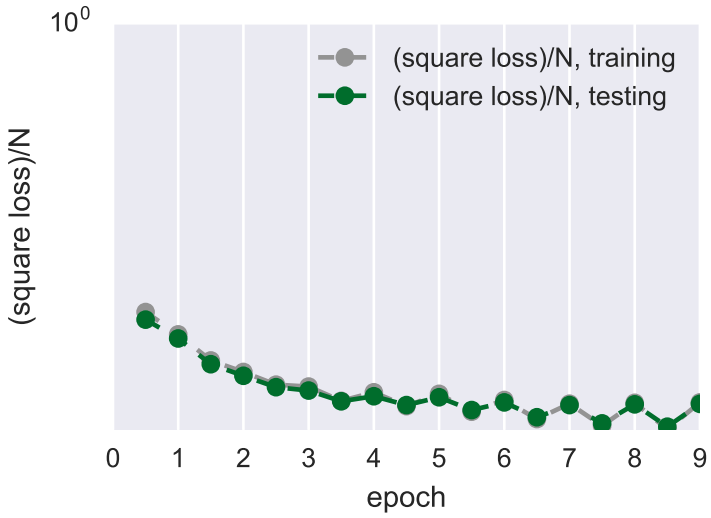
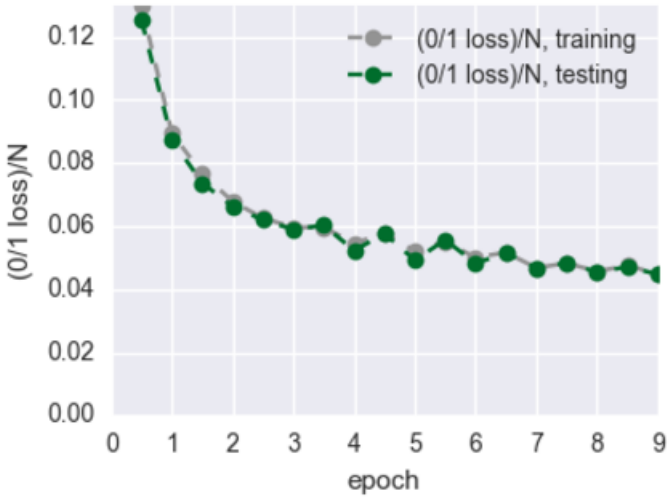


Figure 6: 0/1 loss during hidden = ReLu, output = linear neural net training.



2.3.3 (6 points) What is your final squared loss and 0/1 loss for both the training and test sets?

Data Set	squared loss	(squared loss)/N	0/1 loss	(0/1 loss)/N
Training	13052.2	0.217536	2695	0.0449167
Test	2162.18	0.216218	448	0.0448

Table 2: Question 2.2: ReLu hidden units, Linear output units

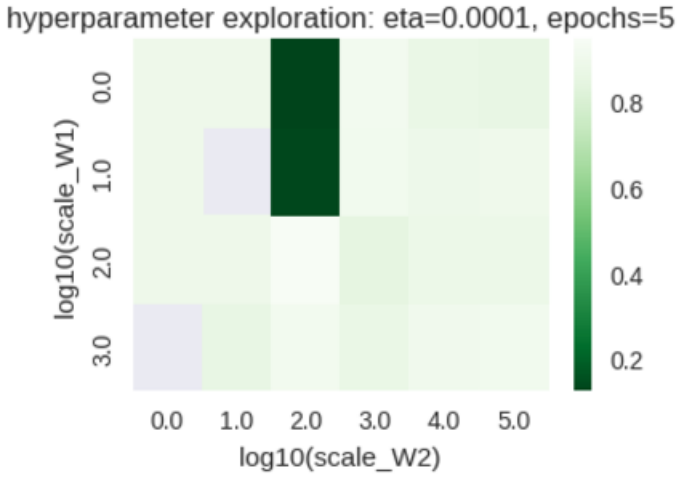
2.4 (3 points) Choose 10 hidden layer nodes at random and display the learned weights (these are the 50 dimensional weights, visualized back into image space).

Figure 7: 10 hidden nodes transformed back to image space for hidden = ReLu, output = linear.

2.5 With ReLu hidden units + ReLu output units [15 points]

The input layer should have 50 dimensions, and let us use 500 nodes for the hidden layer, with a rectified linear (ReLu) transfer function. Now the output layer the ten predictions made by the network should be a ReLu unit where each output is a ReLu taking as input a linear combination of the hidden layer outputs.

Figure 8: Sample hyperparameter exploration for hidden = ReLu, output = ReLu.



2.5.1 (2 points) Specify all your parameter choices: your learning rate (or learning rate scheme), mini-batch size, initialization scheme.

- **learning rate:** 10^{-4}
- **learning rate decay:** $1/epochs^{0.5}$
- **epochs:** 12
- **W1 initialization scheme:** Sample each element from $\mathrm{N}(0, ||X||_2^2)$
- **W2 initialization scheme:** Sample each element from $\mathrm{N}(0, 1)$; divide by 10^2 to prevent weights from blowing up.

2.5.2 (6 points) Plot the squared loss after every half epoch (starting with your initial squared error). Please label your axes in a readable way. Plot the loss of both the training and test losses on the same plot. For the 0/1 loss, do the same, except start your plots a little later (e.g. when the 0/1 loss is below 7%) so they are more readable.

Figure 9: Square Loss during fitting for hidden = ReLu, output = ReLu.

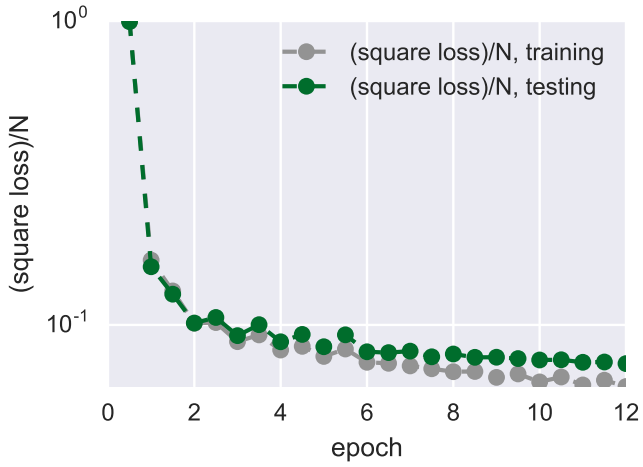
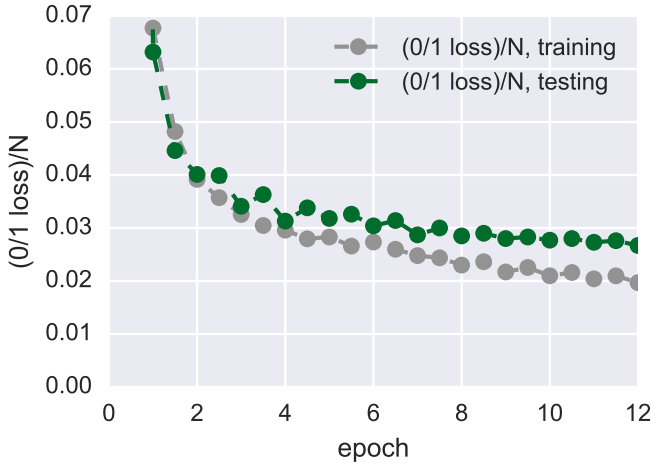


Figure 10: 0/1 Loss during fitting for hidden = ReLu, output = ReLu.



2.5.3 (4 points) What is your final squared loss and 0/1 loss for both the training and test sets?

Data Set	squared loss	(squared loss)/N	0/1 loss	(0/1 loss)/N
Training	3762	0.0627	1181	0.0197
Test	746	0.0746	267	0.0267

Table 3: Question 2.3: ReLu hidden units + ReLu output units

2.6 (3 points) Choose 10 hidden layer nodes at random and display the learned weights (these are the 50 dimensional weights, visualized back into image space).

Figure 11: 10 hidden nodes transformed back to image space for hidden = ReLu, output = linear.

3 EM v.s. Gradient Descent [15 points]

3.1 (8 points) Show that: $\nabla L(\theta) = E_{Z\sim \Pr(Z|X,\theta)} \nabla \log \Pr(X, Z|\theta)$

$$
\begin{aligned}
L(\theta) &= log \sum_Z P(X, Z|\theta) \\
\frac{d}{d\theta} \mathrm{L}(\theta) &= \frac{d}{d\theta} \log \sum_Z P(X, Z|\theta) \\
&= \frac{1}{\sum_Z P(X, Z|\theta)} \frac{d}{d\theta} \sum_Z P(X, Z|\theta) \\
&= \frac{1}{P(X|\theta)} \frac{d}{d\theta} \sum_Z P(X, Z|\theta) \\
&= \sum_Z \frac{1}{P(X|\theta)} \frac{d}{d\theta} P(X, Z|\theta) \\
&= \sum_Z \frac{P(Z|X,\theta)}{P(X, Z|\theta)} \frac{d}{d\theta} P(X, Z|\theta) \\
\text{use} \quad \frac{d}{d\theta} \log P(X, Z|\theta) &= \frac{1}{P(X, Z|\theta)} \frac{d}{d\theta} P(X, Z|\theta) \\
&= \sum_Z P(Z|X,\theta) \frac{d}{d\theta} \log P(X, Z|\theta) \\
\nabla L(\theta) &= E_{Z\sim \Pr(Z|X,\theta)} \nabla \log \Pr(X, Z|\theta)
\end{aligned}
$$

3.2 (3 points) Suppose starting from θ, Alice does one gradient update. Now suppose Bob, also starting with θ, performs an E step, and then, instead of doing an exact M-step, Bob does a gradient update on the objective function in the M-step. Both Alice and Bob use the same learning rates. Are Alice and Bob doing the same thing? Give a brief justification of your answer.

Figure 11.16 (page 365) of Murphy shows a sketch of EM's lower bound for approximating the log likelihood function. Alice will do her update using the gradient at theta. If Bob was doing a normal M-step, he would not be using this slope; instead he would move theta to the place that maximizes the current lower bound approximation of the log likelihood. This step could be different than Alice's step side.

However, Bob is doing an unconventional M step that is using the gradient instead of the lower bound approximation. Thus he will take the same step as Alice.

3.3 (4 points) Suppose now that you run the EM algorithm to convergence (assume it converged). Do you reach a critical point of the likelihood function? (i.e. do you reach a point where the gradient of the log likelihood function is 0). Give a brief justification of your answer.

Murphy Section 11.4 describes how EM can be thought of as a process of computing successive lower bounds on the log likelihood function, then picking parameter values that maximize those lower bounds. At each time when we approximate the log likelihood, this approximation touches the true log likelihood. Because the approximation is a lower bound of the true log likelihood, we know that the derivatives of these two functions are exactly the same at the point of contact.

If EM has converged to a critical point, the derivative of the approximation is zero. By the above argument, that also means the derivative of the log likelihood function is zero.

4 Markov Decision Processes and Dynamic Programming [15 points]

$$\tilde{V}(x) = \max_a \left(R(x,a) + \gamma \sum_{x'} \Pr(x'|x,a) V(x') \right)$$

4.1 (8 points) Show that the Bellman operator is a contraction mapping. Specifically, show that for any two value functions $||Bell(V_1) - Bell(V_2)||_\infty \leq \gamma ||V_1 - V_2||_\infty$ where the $||Z||_\infty$ denotes the infinity norm of a vector Z, i.e. $||Z||_\infty = \max_s |Z(s)|$.

Useful:

$$\begin{aligned}
\max_x f_1(x) - \max_x f_2(x) &= f_1(x^*) - \max_x f_2(x) \\
&\qquad (x^* \text{ is the optimal value for } f_1) \\
&\leq f_1(x^*) - f_2(x^*) \\
&\max_x |f_1(x) - f_2(x)|
\end{aligned}$$

Let $f_1 = \max_x \left(R(x,a_1) + \gamma \sum_{x'} Pr(x'|x,a_1) V_1(x') \right)$

Proof:

$$
\begin{aligned}
&||\text{Bell}(V_1) - \text{Bell}(V_2)|| \\
&\quad = \max_x \left| \max_{a_1} \left(R(x,a_1) + \gamma \sum_{x'} Pr(x'|x,a_1) V_1(x') \right) \right. \\
&\qquad\qquad \left. - \max_{a_2} \left(R(x,a_2) + \gamma \sum_{x'} Pr(x'|x,a_2) V_2(x') \right) \right| \\
&\quad = \max_x \left| f_1(x_1,a_1) - \max_{a_2} \left(R(x,a_2) + \gamma \sum_{x'} Pr(x'|x,a_2) V_2(x') \right) \right| \\
&\quad = \max_x \left| f_1(x_1,a_1) - \max_{a_2} \left(R(x,a_2) + \gamma \sum_{x'} Pr(x'|x,a_2) (V_2(x') + V_1(x') - V_1(x')) \right) \right| \\
&\qquad \text{use useful rule above; use the same a for both } a_1 \text{ and } a_2; = \text{ becomes } \leq \\
&\quad \leq \max_{x,a} \left| f_1(x_1,a) - \left(R(x,a) + \gamma \sum_{x'} Pr(x'|x,a) (V_2(x') + V_1(x') - V_1(x')) \right) \right| \\
&\quad \leq \max_{x,a} \left| f_1(x_1,a) - f_1(x_1,a) + \left(\gamma \sum_{x'} Pr(x'|x,a) (V_2(x') - V_1(x')) \right) \right| \\
&\quad \leq \max_{x,a} \left| \left(\gamma \sum_{x'} Pr(x'|x,a) (V_2(x') - V_1(x')) \right) \right| \\
&\quad \leq \gamma \max_{x,a} \left| \sum_{x'} Pr(x'|x,a) (V_2(x') - V_1(x')) \right| \\
&\quad \leq \gamma \max_x |V_2(x') - V_1(x')| \\
&\quad \leq \gamma \max_x |V_1(x') - V_2(x')| \\
&\quad \leq \gamma ||V_1(x') - V_2(x')||_\infty
\end{aligned}
$$

4.2 (3 points) Suppose that upon repeated updating, we reach a fixed point, i.e. we find a V such that Bell$(V) = V$. Show that this V is unique. This V is the value of the optimal policy.

Assume we have two distinct fixed points V_1 and V_2. Since they are fixed points, $\text{Bell}(V_1) = V_1$ and $\text{Bell}(V_2) = V_2$.

$$
\begin{aligned}
||\text{Bell}(V_1) - \text{Bell}(V_2)||_\infty &\leq \gamma ||V_1 - V_2||_\infty \\
\max_x |\text{Bell}(V_1) - \text{Bell}(V_2)| &\leq \gamma \max_x |V_1 - V_2| \\
\max_x |V_1 - V_2| &\leq \gamma \max_x |V_1 - V_2|
\end{aligned}
$$

This is only true when $V_1 = V_2$, meaning the V found when $\text{Bell}(V) = V$ is unique.

4.3 (4 points) Suppose we find a V such that $\text{Bell}(V) = V$. Specify the optimal policy $\Pi(x)$, which specifies the action to be taken in state x, in terms of V and other relevant quantities.

When $\text{Bell}(V) = V$, we have found the a for each x that yields the highest value. The optimal policy is thus the argmax:

Thus $\Pi(x) = \text{argmax}_a \left(R(x,a) + \gamma \sum_{x'} \Pr(x'|x,a) V(x') \right)$