

CS221 Exam

Spring 2020

Read all of the following information before starting the exam:

- This test has 4 problems on 29 pages for a total of 140 possible points. **However, we will only be grading this exam out of 120 points.** Any points you get over a score of 120 will be used as extra credit.
- Note that different questions are worth different amounts of points. Budget your time accordingly!
- Keep your answers precise and concise. We may award partial credit so show all your work clearly and in order.
- Don't spend too much time on one problem. Read through all the problems carefully and do the easier ones first.
- This exam is open-book; you may use any resources, including the course website.
- Good luck!

Problem	Part	Max Score
1	a	14
	b	10
	c	6
	Total	30
2	a	4
	b	12
	c	4
	Total	20
3	a	6
	b	10
	c	12
	d	12
	Total	40
4	a	6
	b	8
	c	10
	d	4
	e	12
	f	10
	Total	50

1. Classification (30 points)

Suppose we have a dataset of N points (x_i, y_i) , where $y_i \in \{-1, +1\}$ for all i . Consider the binary classification problem on this dataset with the pointwise loss function $Loss(\mathbf{x}_i, y_i, \mathbf{w})$ defined as follows:

$$Loss(\mathbf{x}_i, y_i, \mathbf{w}) = \frac{1}{2N} \sum_{j=1}^d (w^{(j)})^2 + \lambda \max \left\{ 0, 1 - y_i \left(\sum_{j=1}^d w^{(j)} \phi(\mathbf{x}_i)^{(j)} \right) \right\} \quad \forall i \in \{1, 2, \dots, N\} \quad (1)$$

where

- $\phi(\mathbf{x}_i)$ is the feature vector corresponding to the i th data point
- $\mathbf{w} \in \mathbb{R}^d$ is the learned weight vector, with $w^{(j)}$ being its j th dimension.
- λ is a regularization parameter

The overall training loss $TrainLoss(\mathbf{w})$ is defined as follows:

$$TrainLoss(\mathbf{w}) = \sum_{i=1}^N Loss(\mathbf{x}_i, y_i, \mathbf{w})$$

a. (14 points) Stochastic Gradient Descent

Consider the following implementation of the stochastic gradient algorithm:

Algorithm 1: Stochastic Gradient Descent

```
1: Randomly shuffle the training data
2: initialize  $\mathbf{w} = \mathbf{0}, i = 1$ 
3: for  $t = 1, 2, \dots, T$  do
4:    $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \text{Loss}(\mathbf{x}_i, y_i, \mathbf{w})$ 
5:    $i \leftarrow (i \bmod N) + 1$ 
6: end for
```

where η is the learning rate, t is the iteration number (different from lecture slides, where t represented the epoch number), and T is the maximum number of iterations. The rest are as defined as in Equation (1).

(i) [3 points] Compute the gradient of the pointwise loss function above with respect to \mathbf{w} , i.e., $\nabla_{\mathbf{w}} \text{Loss}(\mathbf{x}_i, y_i, \mathbf{w})$. Express your answer **in vector notation**.

(ii) [3 points] You train a binary classifier by running stochastic gradient descent, and obtain a low training loss. However, you notice that the loss at test time is quite high. In 1-2 sentences, explain how you would change the hyperparameter λ to improve performance at test time and why this change would help.

(iii) [8 points] Consider a training dataset with two datapoints $(x_a, +1)$ and $(x_b, -1)$. You use a feature extractor $\phi(x)$ on these datapoints, and obtain $\phi(x_a) = [4 \ 2]^T$ and $\phi(x_b) = [1 \ 3]^T$.

You run stochastic gradient descent on $(x_a, +1)$, followed by $(x_b, -1)$ (in that order) with $\lambda = 0.1$ and $\eta = 0.1$. What is the value of the weight vector at the end of these two iterations? Show your work.

b. (10 points) Modifying Margin

You now consider adding a hyperparameter $\hat{\delta} > 0$ to increase the minimum margin of the binary classifier. In other words, you define a modified pointwise loss function $\hat{Loss}(\mathbf{x}_i, y_i, \hat{\mathbf{w}})$ with hyperparameters $\hat{\lambda}$ and $\hat{\delta}$ as follows:

$$\hat{Loss}(\mathbf{x}_i, y_i, \hat{\mathbf{w}}) = \frac{1}{2N} \sum_{j=1}^d (\hat{w}^{(j)})^2 + \hat{\lambda} \max \left\{ 0, \hat{\delta} - y_i \left(\sum_{j=1}^d \hat{w}^{(j)} \phi(\mathbf{x}_i)^{(j)} \right) \right\} \quad \forall i \in \{1, 2, \dots, N\} \quad (2)$$

The weight vector $\hat{\mathbf{w}}$ is learned using stochastic gradient descent on the above loss function. Prove that $\hat{\delta}$ is an unnecessary hyperparameter, and our objective can be accomplished by tweaking the hyperparameter λ to some $\hat{\lambda}$. Specifically:

1. Prove the following by induction: for $\hat{\lambda} = \hat{\delta}\lambda$, show that $\hat{\mathbf{w}} = \hat{\delta}\mathbf{w}$ for iteration 1 and all subsequent iterations of the Stochastic Gradient Descent algorithm described in Algorithm 1. Note that \mathbf{w} and $\hat{\mathbf{w}}$ are both initialized to $\mathbf{0}$.
2. Prove that, for $\hat{\mathbf{w}} = \hat{\delta}\mathbf{w}$, weight vectors \mathbf{w} and $\hat{\mathbf{w}}$ will yield the same prediction labels on all data points \mathbf{x} . Note that predictions are given by $y_{pred}(\mathbf{x}; \mathbf{w}) = \text{sign}(\mathbf{w} \cdot \phi(\mathbf{x}))$

c. (6 points) Stopping Criteria

Our objective for this problem is to design stopping criteria that stop training once training loss has converged.

We define the average training loss at iteration \tilde{t} as follows:

$$L_{avg}(\mathbf{w})^{(\tilde{t})} = \frac{1}{\tilde{t}} \sum_{t=1}^{\tilde{t}} Loss(\mathbf{x}_i, y_i, \mathbf{w})^{(t)}$$

where $Loss(\mathbf{x}_i, y_i, \mathbf{w})^{(t)}$ is the pointwise loss obtained at iteration t .

We modify Algorithm 1 as follows (the only change to the algorithm is the calculation of average training loss at each iteration):

Algorithm 2: Modified Stochastic Gradient Descent

```
Randomly shuffle the training data
initialize  $\mathbf{w} = \mathbf{0}, i = 1, L_{avg} = 0$ 
while stopping criteria not reached do
  for  $t = 1, 2, \dots, T$  do
     $L_{avg} \leftarrow \frac{1}{t} (L_{avg} \times (t - 1) + Loss(\mathbf{x}_i, y_i, \mathbf{w}))$ 
     $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} Loss(\mathbf{x}_i, y_i, \mathbf{w})$ 
     $i \leftarrow (i \bmod N) + 1$ 
  end for
end while
```

(i) [3 points] Consider the following stopping criteria:

$$L_{avg}^{(\tilde{t})} < \epsilon \quad \text{and} \quad \tilde{t} \geq N$$

for some hyperparameter $\epsilon > 0$.

Is the criteria above good for our objective (i.e., designing stopping criteria that stop training once training loss has converged)? Explain why or why not.

(ii) [3 points] Now consider the following stopping criteria:

$$\frac{|L_{avg}^{(\tilde{t})} - L_{avg}^{(\tilde{t}-1)}|}{L_{avg}^{(\tilde{t}-1)}} < \epsilon \quad \text{and} \quad \tilde{t} \geq N$$

for some hyperparameter $\epsilon > 0$. What issues might you run into by using the above criteria for stochastic gradient descent? (Hint: this new criteria might work well for regular gradient descent.)

2. Cluster Away (20 points)

Consider performing k -means clustering with $k = 2$ clusters on the following set of 3 one-dimensional data points:

$$\{-8, 0, 14\} \tag{3}$$

a. (4 points)

What is the globally optimal clustering and the associated reconstruction loss for the above dataset? Your answer should specify the following:

- Centroids μ_1 and μ_2 for clusters 1 and 2 respectively.
- Assignments of points $x_1 = -8, x_2 = 0$ and $x_3 = 14$ to clusters.
- Reconstruction loss (sum of squared distances) of this globally optimal clustering.

b. (12 points)

Consider the following algorithm (henceforth referred to as **k-means++**) to initialize the k-means centroids intelligently rather than at random:¹

- I. Choose one centroid μ_1 , uniformly at random from among the data points.
- II. For each data point x , compute $D^*(x)$, the Euclidean distance between x and the nearest centroid that has already been chosen. In other words $D^*(x) = \min_{\mu_i} D(x, \mu_i)$
- III. Choose one new data point at random as a new centroid, using a weighted probability distribution where a point x is chosen with probability proportional to $D^*(x)$. In other words,

$$P[x_i \text{ is chosen as } \mu] = \frac{D^*(x_i)}{\sum_{j \in J} D^*(x_j)}$$

where J is the set of all points that haven't been chosen as a centroid yet.

- IV. Repeat Steps II and III until k centroids have been chosen.
- V. Now that the initial centroids have been chosen, proceed using standard k -means clustering.

Now consider the dataset specified in equation (3) with $k = 2$.

Fill in the table on the following page. For each row, you should specify (1) the probability of the centroids being chosen in this order with the k-means++ algorithm, (2) the **final** clusters that would result from these initial centroids (at the conclusion of k-means clustering), and (3) whether or not this clustering is optimal. You may leave your probabilities as unsimplified fractions and/or sums of fractions, e.g. $\frac{1}{1+2+3}$.

Note that, in the table on the following page, μ_1 is the first centroid chosen and μ_2 is the second centroid chosen.

¹Adapted from the following work: Arthur, David, and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Stanford, 2006.

μ_1	μ_2	Probability	Final Clusters	Optimal (Y/N)
-8	0			
-8	14			
0	-8			
0	14			
14	-8			
14	0			

What is the overall probability that using the k-means++ algorithm returns the globally optimal k -means cluster for this dataset with $k = 2$? (You may leave your answer as an unsimplified fraction or a sum of fractions.)

c. (4 points)

For part (c), consider an unspecified dataset in \mathbb{R}^d with n datapoints, and with $k \in \{2, 3, \dots, n\}$.

An alternative initialization method discussed in class is "multiple random initialization":
(1) run multiple k-means clustering runs with random initializations of centroids on each run,
and (2) choose the clustering with the lowest reconstruction loss.

Under what circumstances might you choose: (i) k-means++ initialization over multiple random initialization, (ii) multiple random initialization over k-means++ initialization?
Describe no more than one circumstance in each case.

The circumstances you describe can include: the problem or application that clustering is being applied to, specific prior knowledge of the dataset, compute constraints, and/or the importance of getting the globally optimal clustering.

3. Weeding a lawn (*40 points*)

You are hired to weed a large lawn. Suppose the lawn is a $m \times n$ grid, and grass randomly grows in these grids (see Figure 1). Starting at a grid point, you turn on your weeding machine and weed the 3×3 area around you with no cost. Then, at each timestep, you must move to an adjacent cell and weed the 3×3 area centered at your new location at a total cost of 1 per timestep, until all grass is weeded. Near boundaries, you still weed the 3×3 area intersecting with the lawn (see Figure 2).

To formulate this as a state-based problem, each action is defined as a combination of move and weed, that is, a move from your current location followed by a weed of the 3×3 area centered at the location that you moved to, and you can assume there is no grass in the 3×3 area around the start location. Every move comes with cost 1.

You are expected to weed all the grass in the lawn. Let G_0 to be set of indices of all the grass in the initial lawn. For following questions, assume you start at $(1, 1)$ if not explicitly specified.

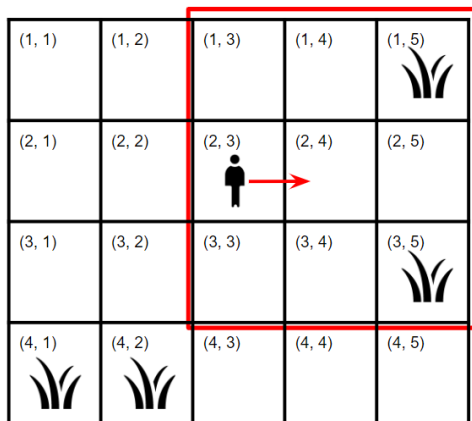


Figure 1: Example of a 4×5 lawn. Here, $G_0 = \{(4, 1), (4, 2), (1, 5), (3, 5)\}$. Standing on $(2, 3)$, if you move to $(2, 4)$, you can weed the entire red region.

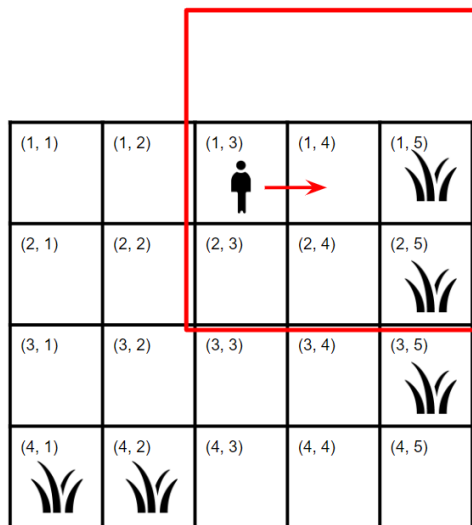


Figure 2: Boundary example with the same lawn. Standing on $(1, 3)$, if you move to $(1, 4)$, you can weed the 6 squares in the intersection between the red region and the lawn.

a. (6 points)

Fill out the components of the search problem corresponding to the above problem setting. You can use the notation $G_{(i,j)}^{\text{weed}}$ to represent the set of locations of weeds that you can weed when moving to location (i, j) . You might find set difference notation useful: $A \setminus B = \{a : a \in A \text{ and } a \notin B\}$.

- $s_{\text{start}} = ((1, 1), G_0)$.
- $\text{Actions}(((i, j), G)) = \{a \in \{(-1, 0), (+1, 0), (0, -1), (0, +1)\} : (i, j) + a \text{ is in bounds}\}$.
- $\text{IsEnd}(((i, j), G)) =$

- $\text{Succ}(((i, j), G), a) =$

- $\text{Cost}(((i, j), G), a) =$

b. (10 points)

You are very impatient for the original search problem to finish, so you want to apply A^* with "hierarchical pathfinding". To relax the problem, you divide the lawn into adjacent $k \times k$ blocks and in each move and weed action you weed all the grass in blocks that your 3×3 weeding area has intersection with. The original problem is equivalent to the relaxed problem with $k = 1$. **For simplicity, you can assume m and n are both multiples of k .**

Solving this relaxed search problem directly using UCS is as hard as the original problem. However, you can crack this relaxed problem using a modified UCS algorithm: when visiting a state, instead of moving one step, you will walk all the way to the border of next adjacent block with cost equal to the walked distance. Even though this algorithm only searches a subset of all possible paths, e.g. it will miss a zigzag path inside a block, it will always find a path with equal cost to any of optimal paths of the relaxed problem. In your answer you can assume the correctness of this algorithm and you don't need all details about this algorithm to solve following questions.

(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 8)
(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 8)
(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 8)
(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 8)
(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 8)
(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 8)
(7, 1)	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 8)
(8, 1)	(8, 2)	(8, 3)	(8, 4)	(8, 5)	(8, 6)	(8, 7)	(8, 8)

Figure 3: Example of a 8×8 lawn with block length $k = 4$. In the relaxed problem with the modified UCS algorithm, you can move to one of (1, 3), (2, 4), (4, 3) or (2, 1) with cost 1, 1, 2, 2 respectively. If you move to (4, 3), then your weeding area will intersect the lower-left gray block and you will weed the grass in (5, 1), (5, 4), and (7, 2).

(i) [6 points] Prove that we can use the future cost of this relaxed problem as a consistent heuristic for the original problem in part (a).

(ii) [4 points] What is one advantage for using large k for this relaxed problem? What is one advantage of using small k ? Please limit your answer to a maximum of two sentences for each question.

c. (12 points)

Suppose you are standing at (3, 4) and you've managed to weed the entire lawn! Before you celebrate your success, however, you see that grass is about to re-grow in exactly one of three possible places: (1, 1), (4, 1), and (2, 9) *with uniform probability*. You can see this in Figure 4. You need to make one action before they re-grow, but you don't know which spot the grass will pop up in! After this action you will know the location of the newly grown grass and head to weeding it.





(1, 1) 	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 8)	(1, 9)
(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 8)	(2, 9) 
(3, 1)	(3, 2)	(3, 3)	(3, 4) 	(3, 5)	(3, 6)	(3, 7)	(3, 8)	(3, 9)
(4, 1) 	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 8)	(4, 9)

Figure 4: Lawn of this problem. You stand on (3, 4) and grass will grow in exactly one of the three marked locations with uniform probability after your next move.

(i) [6 points] What is the optimal action you can take to minimize the *expected future cost*? If there is a tie between optimal actions, state all of them. What is the expected future cost *after* you take this action?

(ii) [6 points] What is the optimal action you can take to minimize the *worst case future cost*? If there is a tie between optimal actions, state all of them. What is the worst case future cost *after* you take this action?

d. (12 points)

You find that grass randomly grows in certain locations on the lawn, denoted in the set H . More specifically, after every move and weed you make, grass will regrow in one location from H , chosen uniformly at random. If there is already grass at that location, nothing will happen. Grass can also re-grow at the location that you have just weeded.

Because it is now impossible to weed all the grass, you are paid at a per-piece basis: you will receive reward q for each grass you weed, and you can quit at any time. The moving cost 1 (or equivalently reward of -1) still applies to this problem.

Due to the randomness of regrown grass, taking an action might transit to a state indeterminately, so you want to form it as a MDP problem instead.

(i) [6 points] Modify the original search problem into a MDP. You can use the notation $G_{(i,j)}^{\text{weed}}$ to represent the set of locations of weeds that you can weed when moving to location (i, j) .

- $s_{\text{start}} = ((1, 1), G_0)$.
- $\text{Actions}(((i, j), G)) = \{a \in \{(-1, 0), (+1, 0), (0, -1), (0, +1)\} : (i, j) + a \text{ is in bounds}\} \cup \{\text{Quit}\}$
- For $a \neq \text{Quit}$, $\text{Reward}(((i, j), G), a, ((i, j) + a, G')) =$

- For $a \neq \text{Quit}$, $T(((i, j) + a, G') \mid ((i, j), G), a) =$

(ii) [6 points] You are thinking about using value iteration on this MDP, and you are interested in its time complexity.

- For any state, what is the largest possible number of successor states? Provide an exact expression.
- What is the total number of states possible for this problem, using Big-O notation?
- What is the time complexity of one iteration of value iteration of all states, using Big-O notation?

For all of these questions, you may express your answer using m , n , G , and H , though you may not need all of these.

4. Olympics (50 points)

You are the president of the small nation of Inferencia, and you have been charged with choosing which of your country's two rival soccer teams - the Bayesians or the Markovians - should represent Inferencia at the upcoming Olympics. You'd like to send whichever team is more popular, so you decide to model the monthly evolution of the two teams' fanbases during the months leading up to the Olympics using a dynamic Bayesian network.

Let B_t denote the number of fans that the Bayesians have in month t , and let M_t denote the number of fans that the Markovians have in month t . You have no way of observing these quantities directly, but you can observe two other quantities which they influence: let J_t denote the number of jerseys sold by the Bayesians in month t , and let A_t denote the attendance of the monthly exhibition game between the Bayesians and the Markovians in month t .

The fanbases of the two teams evolve according to the following model, where each month a fan is either gained or lost with equal probability:

$$\Pr(M_{t+1}|M_t) = \begin{cases} \frac{1}{2} & \text{if } M_{t+1} = M_t - 1 \\ \frac{1}{2} & \text{if } M_{t+1} = M_t + 1 \\ 0 & \text{otherwise} \end{cases} \quad \Pr(B_{t+1}|B_t) = \begin{cases} \frac{1}{2} & \text{if } B_{t+1} = B_t - 1 \\ \frac{1}{2} & \text{if } B_{t+1} = B_t + 1 \\ 0 & \text{otherwise} \end{cases}$$

The Bayesian fans are big spenders - almost every fan buys a jersey each month! We model the fanbase size's influence on jersey sales by:

$$\Pr(J_t|B_t) = \begin{cases} 0.3 & \text{if } J_t = B_t \\ 0.25 & \text{if } J_t = B_t - 1 \\ 0.2 & \text{if } J_t = B_t - 2 \\ 0.15 & \text{if } J_t = B_t - 3 \\ 0.1 & \text{if } J_t = B_t - 4 \\ 0 & \text{otherwise} \end{cases}$$

Lastly, because most fans attend each monthly exhibition (although sometimes more, and sometimes fewer), we model the influence of the fanbase sizes on the exhibition attendance by:

$$\Pr(A_t|B_t, M_t) = \begin{cases} 0.14 & \text{if } A_t = B_t + M_t \\ 0.13 & \text{if } |A_t - (B_t + M_t)| = 1 \\ 0.11 & \text{if } |A_t - (B_t + M_t)| = 2 \\ 0.09 & \text{if } |A_t - (B_t + M_t)| = 3 \\ 0.06 & \text{if } |A_t - (B_t + M_t)| = 4 \\ 0.04 & \text{if } |A_t - (B_t + M_t)| = 5 \\ 0 & \text{otherwise} \end{cases}$$

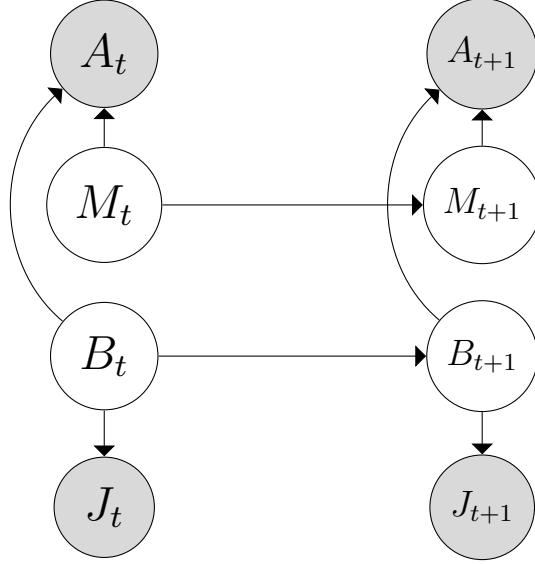


Figure 5: The changing fanbases process modeled as a dynamic Bayesian network. The unshaded nodes correspond to the latent/hidden fanbase counts, and the shaded nodes correspond to the observable emissions.

Note that the assumptions and inferences made in individual parts (i.e. **(a)**, **(b)**, etc.) of this problem do *not* carry over from one to the next; the only assumptions you may make in a given part are those which are explicitly stated in that part's description.

a. (6 points) (Conditional) Independences

Mark each of the following as True or False.

(i) [1 point] $B_t \perp\!\!\!\perp J_{t+1}$

(ii) [1 point] $B_t \perp\!\!\!\perp J_{t+1} \mid B_{t+1}$

(iii) [1 point] $B_t \perp\!\!\!\perp M_t$

(iv) [1 point] $B_t \perp\!\!\!\perp M_t \mid A_t$

(v) [1 point] $A_t \perp\!\!\!\perp M_{t+1}$

(vi) [1 point] $A_t \perp\!\!\!\perp M_{t+1} \mid M_t$

b. (8 points) Domain Consistencies

As a first step, we will not concern ourselves with which fanbase counts are *probable*, but instead which counts are even *possible*. Suppose that we observe, in our first month of collecting data, that $J_1 = 75$ and $A_1 = 100$. Give the domains for M_1 and B_1 that are consistent with these observations.

You need only give the consistent domains (using either set notation or inequality notation) in order to receive full credit.

c. (10 points) Inference

Suppose the Bayesian's manager took a nationwide poll in month t that concluded they had exactly 75 fans. Suppose additionally that in month $t + 2$, the Bayesians sell 73 jerseys. What is the probability that in month $t + 2$ the Bayesians have 77 fans?

$$\Pr(B_{t+2} = 77 | B_t = 75, J_{t+2} = 73) =$$

d. (4 points) Gibbs Sampling

Inference is exhausting; you decide that you'd be satisfied with simply being able to draw samples from distributions rather than specifying them exactly. In particular, you want to sample joint assignments to the variables $\{B_t, M_t, A_t, J_t\}_{t=1}^T$ for some time horizon T . You decide to implement Gibbs sampling for this purpose, but something's not right! What additional information, beyond what we've given you, would allow you to perform Gibbs sampling? Briefly explain.

e. (12 points) Exact Filtering

You now want to begin making inferences as to the sizes of the teams' fanbases given only observations of attendances and jersey sales. Recall that exact inference of this kind in dynamic Bayesian networks can be achieved using a dynamic programming approach - for example, in the context of Hidden Markov Models, we used the forward-backward algorithm to do filtering and smoothing.

Give recursive expressions for the following filtering queries. Leave your expressions in terms of known probabilities.

(i) [4 points] Let's start by making inferences based only on observed jersey sales. Denote $F_t(b_t) = \Pr(B_t = b_t | J_1 = j_1, \dots, J_t = j_t)$. Give a recursive expression for $F_t(b_t)$ assuming that you've already computed $F_{t-1}(b_{t-1})$ for all b_{t-1} .

(ii) [8 points] Let's bring in the observed attendances as well! Now, denote $F_t(b_t, m_t) = \Pr(B_t = b_t, M_t = m_t | J_1 = j_1, \dots, J_t = j_t, A_1 = a_1, \dots, A_t = a_t)$. Give a recursive expression for $F_t(b_t, m_t)$ assuming that you've already computed $F_{t-1}(b_{t-1}, m_{t-1})$ for all b_{t-1} and all m_{t-1} .

f. (10 points) Particle Filtering

Throughout this problem, you are free to leave quantities in terms of unevaluated expressions (i.e. you may write $0.75 \cdot 0.5$ instead of 0.375).

Computing all of those terms exactly seems tedious, so you instead decide to employ particle filtering to quickly and painlessly provide you with approximate solutions. You're fine with a (very) crude approximation, so you only use two particles.

(i) [2 points] Suppose you begin with the two particles $(B_1 = 80, M_1 = 75)$ and $(B_1 = 82, M_1 = 74)$. You then observe that $J_1 = 79$ and $A_1 = 154$. Compute the weights that you should assign to the two particles based on this evidence.

(ii) [2 points] Using these weights, we now resample two new particles. Provide this sampling distribution.

Probability of sampling a new particle to be $(B_1 = 80, M_1 = 75) =$

Probability of sampling a new particle to be $(B_1 = 82, M_1 = 74) =$

(iii) [3 points] Suppose both of our new particles are sampled to be $(B_1 = 80, M_1 = 75)$. We now extend these particles using our dynamics models. What is the probability that a particular one of these two particles is extended to:

$$(B_1 = 80, M_1 = 75, B_2 = 78, M_2 = 76)?$$

$$(B_1 = 80, M_1 = 76, B_2 = 79, M_2 = 75)?$$

$$(B_1 = 80, M_1 = 75, B_2 = 79, M_2 = 76)?$$

(iv) [3 points] Suppose now that you have access to a large number of particles which are approximating the distribution over $(B_1, \dots, B_n, M_1, \dots, M_n)$. The Olympics are happening in 6 months, but you have to decide now which team to send so that they can start preparing! You decide to make predictions of B_{n+6} and M_{n+6} in order to send whichever team you predict to be more popular during the month in which the Olympics will be held. Explain in a few sentences how you would use your particles for making this decision.