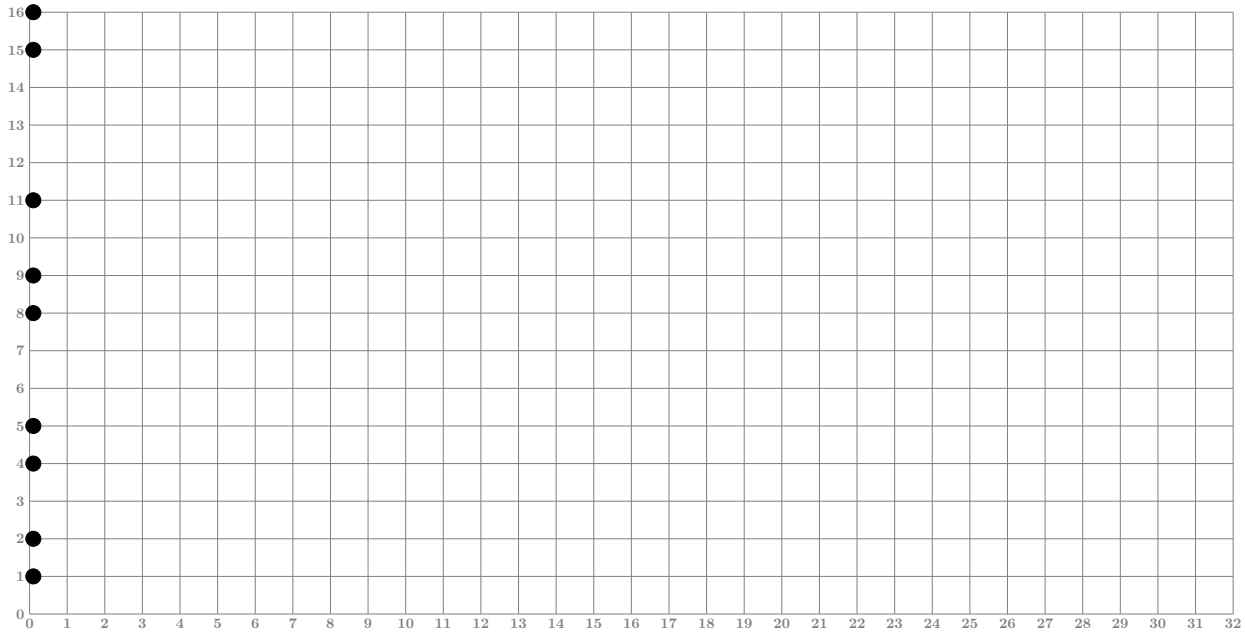
CS 1810 Second Half Practice Problems

1. **Hierarchical Agglomerative Clustering**

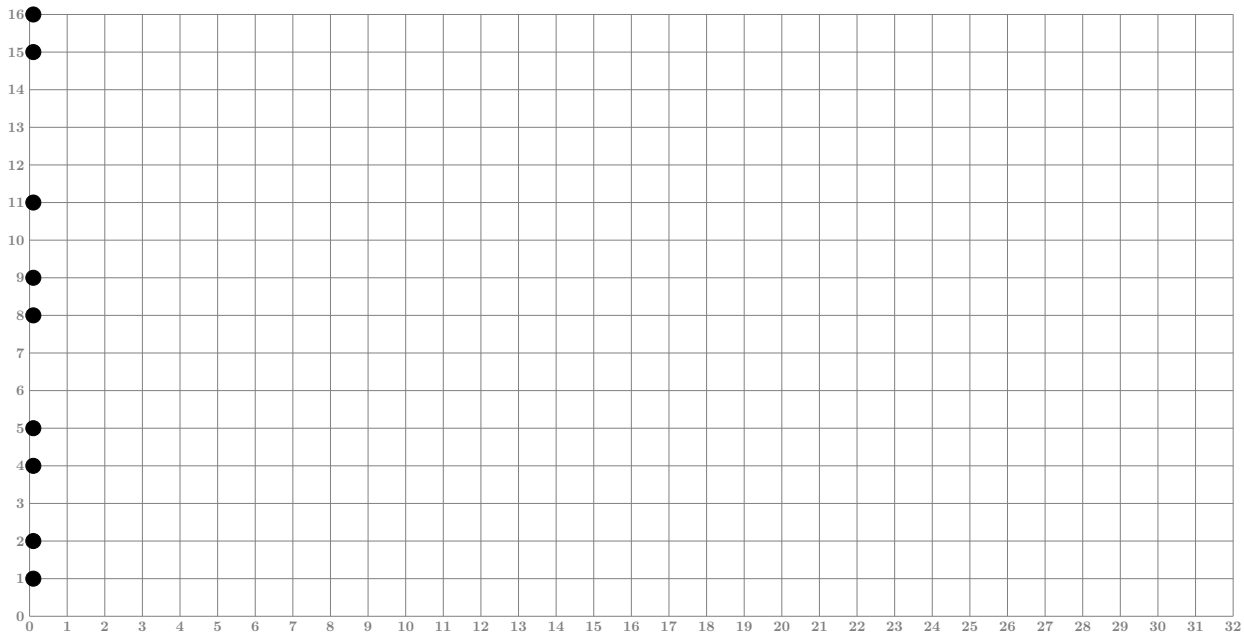
   Consider nine points $x_1, \ldots, x_9$ shown below, where the y-axis provides their values. We define $d(x, x') = |x - x'|$, and consider two different cluster distances.

   Draw the dendrograms for the data. In the top figure, use the min-linkage distance and in the bottom figure use the max-linkage distance. Remember that the x-axis reflects the distance between clusters when they get merged.

   (a) Min Linkage:

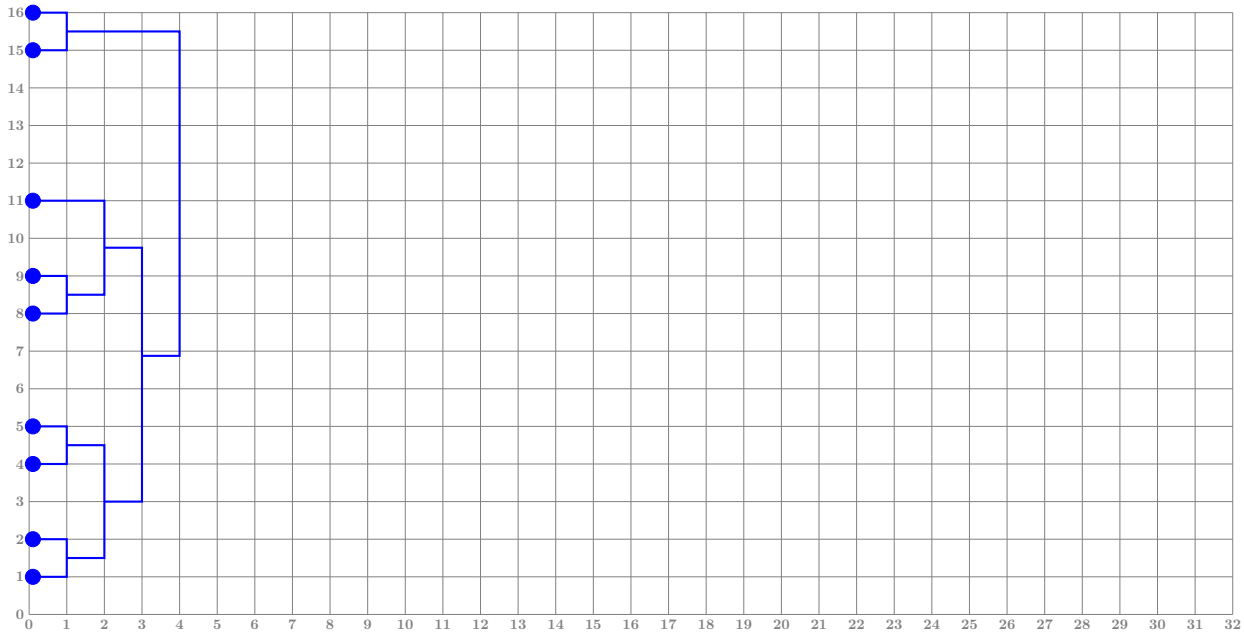


   (b) Max Linkage:

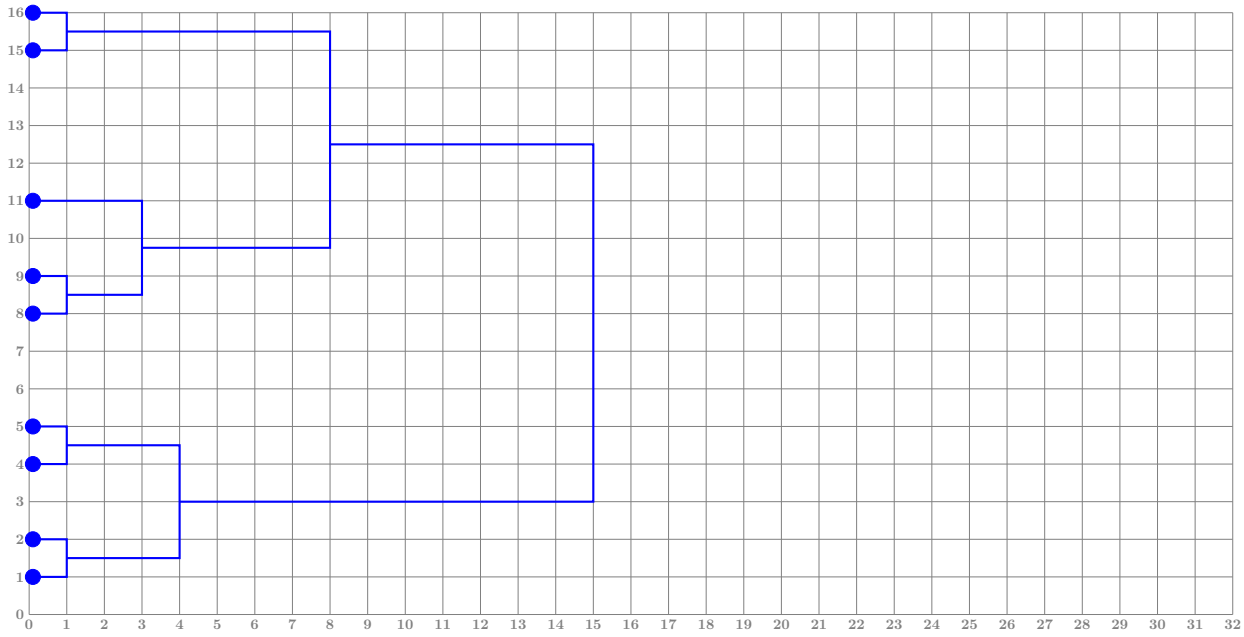

16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32

Solution:

(a) Min Linkage:

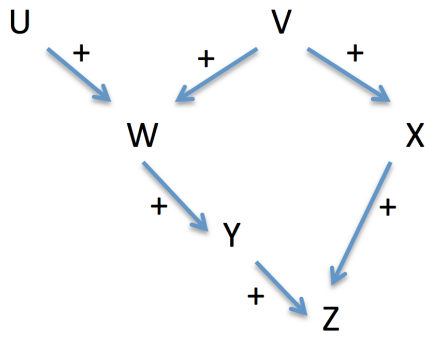


(b) Max Linkage:



2. **Bayesian networks**

Consider the following Bayesian network, where the variables are all Boolean.



The '+' annotations indicate the direction of the local effect; e.g., the '+' from U to W means that for each value v of V,

$$p(W = true \,|\, U = true, V = v) > p(W = true \,|\, U = false, V = v).$$

For each of the following questions, select one of the following, and also state which (if any) undirected paths are blocked (in the sense of d-separation):

- $=$ if the two probabilities are necessarily equal;
- $<$ if the first probability is necessarily smaller;
- $>$ if the first probability is necessarily larger;
- ? if none of these cases hold.

(a) $p(V = true \mid Y = false)$ $p(V = true \mid Y = true)$

(b) $p(V = true \mid Z = false)$ $p(V = true \mid Z = true)$

(c) $p(U = true \mid W = true, Y = false)$ $p(U = true \mid W = true, Y = true)$

(d) $p(Y = true \mid Z = true, X = false)$ $p(Y = true \mid Z = true, X = true)$

(e) $p(U = true \mid Y = true, Z = false)$ $p(U = true \mid Y = true, Z = true)$

Solution:

[We give more explanation here than you would expected to give!]

(a) $p(V = true \mid Y = false) \; < \; p(V = true \mid Y = true)$

First, let's ask whether $V \perp Y$, i.e., are V and Y independent?

Path VWY is not blocked, path $VXZY$ is blocked at Z (converging arrow). So, V is not independent of Y.

Considering the unblocked path, since the $V - W$ effect is positive and the $W - Y$ effect is positive, then we have Y being true makes V more likely to be true. *Notice that this positive correlation goes in both directions, e.g., if V is positively correlated with W, then W is positively correlated with V.* For this reason, we conclude "<".

(b) $p(V = true \mid Z = false) \; < \; p(V = true \mid Z = true)$

First let's ask whether $V \perp Z$? Neither paths V-W-Y-Z or V-X-Z are blocked, so these are not independent.

Now, the effect on each of the paths is positive, for the same reason as the positive correlation argument in in part (a). Since both effects go in the same direction we can answer affirmatively with a "<" inequality.

(c) $p(U = true \mid W = true, Y = false) \; = \; p(U = true \mid W = true, Y = true)$

Now we have W in the evidence. The relevant independent question is $U \perp Y \mid W = True$, i.e., are U and Y conditionally independent, given this evidence W?

There are two paths to check. Path $U - W - Y$ is blocked at W, and path $U - W - V - X - Z - Y$ is blocked at Z (converging arrow, no evidence). Note the second path is no longer blocked at W because there's a converging arrow on this path. But it is any way blocked, because Z is not in the evidence.

Since both paths are blocked, then U and Y are conditionally independent given $W = True$, and we have "=" as the answer.

(d) $p(Y = true \mid Z = true, X = false)$? $p(Y = true \mid Z = true, X = true)$

Now Z in evidence. We're interested in understanding $Y \perp X \mid Z = True$.

Path $YWVX$ is not blocked. In addition, $Y - Z - X$ is not blocked (converging arrows at Z, and we know Z).

There are two paths for information to flow between Y and X. On the first path $YWVX$ this is a positive effect for the reasons as the positive correlation argument in part (a). But on the second path YZX this is a negative effect because we have "explaining away" through Z. That is, knowing X is true reduces the probability that Y is true since it explains Z being true and means that it's less likely we also have the other possible reason of Y being true.

Because the effects go in different directions, the answer is "?".

(e) $p(U = true \mid Y = true, Z = false) \; > \; p(U = true \mid Y = true, Z = true)$

Now Y is in the evidence.

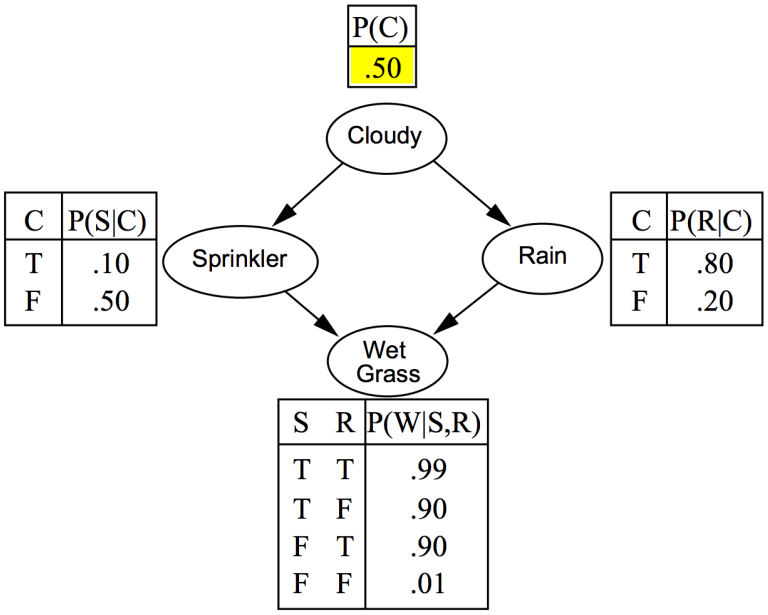
We're interested in understanding $U \perp Z \mid Y$. In this case path $U - W - Y - Z$ is blocked at Y. But path $U - W - V - X - Z$ is not blocked (we have converging arrows at W and a child of W is in the evidence, and so the path is not blocked at W).

We have a single unblocked path, and information can flow along $U - W - V - X - Z$. Now, if Z is true then this makes V more likely to be true by the positive correlation argument in (a), and note that Y being true makes W likely to be true. We now have an "explaining away" pattern at W where having Y be more likely to be true, conditioned on W being likely to be true, makes U less likely to be true.

Hence the "$>$" in the answer.

3. **Bayesian networks**

Consider this example of a Bayesian network with binary variables. It models a garden lawn and whether or not the grass is wet.



(a) Construct an alternative Bayesian network that models the same distribution for variable ordering, S, C, R, W. That is, add S, then C with any required edge, then R with any required edges, then W with any required edges. **Don't specify conditional probability tables.** [Hint: Use the given Bayesian network to determine which conditional Independence properties hold amongst preceding variables, and only include needed edges.]

(b) Is this new Bayesian network a correct model of the distribution? Which network do you consider to be preferable, if any?

(c) Going back to the original network, what is the probability that it is not cloudy, rains, sprinkler doesn't run, and grass is wet?

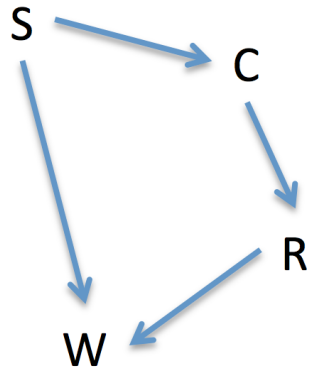
(d) In the original network: write down the first two steps of variable elimination for $p(W)$, eliminating C and then S. Perform the numerical calculations!

Solution:

(a) Construct the Bayesian Network for ordering S, C, R, W. [Note: Written in full here, but you didn't need to provide this detail.]

- Variable S: just add the node
- Variable C: check the following
  - can we add no in-edges? Is $C \perp S$? no!

  Conclude that we need edge $S \to C$.
- Variable R: check the following
  - can we add no in-edges? Is $R \perp C$? no!
  - can we add just one in-edge? Suppose we add C to R. Is $R \perp S \,|\, C$? yes! (in original network: R-C-S blocked at C, R-W-S blocked at W.) Conclude can just add C to R.
- Variable W: check the following
  - can we add no in-edges? Is $W \perp R$? No!
  - can we add just one in-edge?
    what if we just had the R-W edge? Is $W \perp C \,|\, R$? no, path $C - S - W$ not blocked in original network.
    what if we just had the C-W edge? Is $R \perp W \,|\, C$? no!
    what if we just had the S-W edge? Is $R \perp W \,|\, S$? no!
  - can we add just two in-edges? Consider edges S-W and R-W. Is $C \perp W \,|\, S, R$? checking original network, yes! paths CRW and CSW are both blocked. Conclude that it is sufficient to just add these two edges.



(b) Yes, it is correct.[1]
In this case, both have the same number of parameters ($1+2+2+4 = 9$), and thus one is not preferred for reasons of compactness.[2] But we might prefer the first network because it is constructed in a causal order and is therefore more interpretable (the $S \to C$ edge in the new network is hard to interpret!).

[1] Networks are correct for any ordering, but the ordering can affect compactness and interpretability.

[2] Compact networks are generally preferred because they have fewer parameters, need less data to learn, and are less likely to overfit if data is noisy.

(c)

$$
\begin{aligned}
&p(C = false, R = true, S = false, W = true) \\
&= p(C = false)p(S = false \mid C = false)p(R \mid C = false)p(W \mid S = false, R) \\
&= 0.5 \cdot 0.5 \cdot 0.2 \cdot 0.9 = 0.045
\end{aligned}
$$

(d)

$$
\begin{aligned}
p(W) &= \sum_{c,s,r} p(C)p(S \mid C)p(R \mid C)p(W \mid S, R) \\
&= \sum_{s,r} p(W \mid S, R) \sum_{c} p(C)p(S \mid C)p(R \mid C) \\
&= \sum_{s,r} p(W \mid S, R)\psi_1(S, R) \\
&= \sum_{r} \sum_{s} p(W \mid S, R)\psi_1(S, R) \\
&= \sum_{r} \psi_2(W, R)
\end{aligned}
$$

Calculations:

ψ_1		$C = false$				$C = true$				
S	R	$p(C)$	$p(S \mid C)$	$p(R \mid C)$	$\times$	$p(C)$	$p(S \mid C)$	$p(R \mid C)$	$\times$	$\sum$
T	T	0.5	0.5	0.2	0.05	0.5	0.1	0.8	0.04	0.09
T	F	0.5	0.5	0.8	0.2	0.5	0.1	0.2	0.01	0.21
F	T	0.5	0.5	0.2	0.05	0.5	0.9	0.8	0.36	0.41
F	F	0.5	0.5	0.8	0.2	0.5	0.9	0.2	0.09	0.29

ψ_2		$S = false$			$S = true$			
W	R	$p(W \mid S, R)$	$\psi_1(S, R)$	$\times$	$p(W \mid S, R)$	$\psi_1(S, R)$	$\times$	$\sum$
T	T	0.9	0.41	0.369	0.99	0.09	0.0891	0.4581
T	F	0.01	0.29	0.0029	0.9	0.21	0.189	0.1919
F	T	0.1	0.41	0.041	0.01	0.09	0.0009	0.0419
F	F	0.99	0.29	0.2871	0.1	0.21	0.021	0.3081

4. **Markov Decision Process (Modeling)**

You are asked to develop a Markov Decision Process (MDP) to be used for the control of a single elevator. To model:

- There are three floors
- There are three buttons inside the car
- There is a single call button outside on each floor
- The door of the elevator opens and closes.

The "agent" here is the elevator itself, and the aim of the system is to get passengers to their appropriate floors.

(a) Describe in words the states, actions, reward function, and transition model for a suitable MDP model. Make sure that the reward function is clear.

(b) Explain your model as you introduce it. From your explanation the reader should understand the idea for why an optimal policy should lead to an efficient system.

Note: **There is no single correct answer here.**

Solution:

There is no single correct answer to this problem, but your response should clearly identify the sets of states and actions, the reward function, and reasonable transitions.

States

A single state $s \in S$ is represented in a factored way through $s = (F, R, C)$, with

- (floor) $F \in \{1, 2, 3\}$
- (requests from inside car) $R \subseteq \{1, 2, 3\}$
- (call from outside elevator) $C \subseteq \{1, 2, 3\}$

Initial state: $F = 1, R = \emptyset, C = \emptyset$.

Note: We choose not to model whether the door is open or closed, where the elevator was in the past, or how long someone has been waiting.

Actions

An action is one of

- open-close (modeled as a single action, for simplicity)
- up (available if $F < 3$)
- down (available if $F > 1$)
- nothing

The 'nothing' action is to stop the elevator continually doing something

Reward

$$r(s, \text{open-close}) = \begin{cases} 1 & \text{if } (F \in C) \vee (F \in R) \\ -0.01 & \text{o.w.} \end{cases}$$

$$r(s, up) = r(s, down) = -0.01$$

$$r(s, nothing) = 0$$

We include positive reward when the open-close action is taken and the elevator is at a floor where it has been requested to go to or where it was called from. Other rewards are negative to dissuade it from doing things without need.

Transition

Define random variables $X_j \sim [\{j\} : 0.5,\ \emptyset : 0.5]$ that take on set value $\{j\}$ w.p. 0.5, and emptyset otherwise.

Define random variable $Y(C, F)$ that is $\emptyset$ if $F \notin C$ (was not called to this floor), or uniformly sampled from $\{1, 2, 3\} \setminus F$ otherwise (the floor the rider wants to go to.)

We can now define a transition model for next state $s' = (F', R', C')$ reached after action a in state s. We do this in factored way. First, in regard to floor:

- if $a = up$: then $F' = F + 1$
- if $a = down$: then $F' = F - 1$
- if $a = nothing$ or $a =$ open-close then $F' = F$

Second, in regard to calls:

- if $a \in \{up, down, nothing\}$ then $C' := C \cup X_1 \cup X_2 \cup X_3$ because we assume that there's a 50% probability of each floor being called from outside
- if $a =$ open-close then $C' := (C \setminus F) \cup X_1 \cup X_2 \cup X_3$ because open-close will drop any call button at that floor, but someone else can still call before the next period

Third, in regard to requests:

- if $a \in \{up, down, nothing\}$ then $R' := R$
- if $a =$ open-close then $R' := (R \setminus F) \cup Y(C, F)$

Note: This models that the effect of open-close is to drop any request that had been at that floor, and add a request only if there had been a call at the floor, and only going to a different floor.

5. **Alternate Reward Function for MDPs**

We have been assuming that the reward function for an MDP has the form $r(s, a)$. Also recall that we have written value iteration for infinite-horizon problems as:

$$V'(s) \leftarrow \max_a \left[r(s,a) + \gamma \sum_{s'} p(s' \mid s, a) V(s') \right]$$

Now, imagine that we have a reward function that depends on both the current state *and* the next state, i.e., $r(s, a, s')$.

(a) Explain why this kind of reward function can be useful from a modeling perspective

(b) Write an expression for the value iteration step that incorporates this alternative type of reward.

(c) Explain formally why this approach is neither more general nor less general than an MDP model that insists on just using $r(s, a)$.

Solution:

(a) This can be useful for modeling. For example, a room cleaning robot can have $r(s, pick-up, s') = 1$ if next state does not have a broken object and $r(s, pick-up, s') = -10$ if next state does have a broken object. Without this, we'd model $r(s, pick-up)$ as the expected reward for these two outcomes.

(b) The value iteration step becomes:

$$V'(s) \leftarrow \max_a \left[\sum_{s'} p(s' \mid s, a) r(s, a, s') + \gamma \sum_{s'} p(s' \mid s, a) V(s') \right]$$

(c) This approach can represent any $r(s, a)$ reward because we can always define $r(s, a, s') = r(s, a)$, for all values of s'. Any $r(s, a, s')$ function can be represented as an $r(s, a)$ function by defining $r(s, a) = \sum_{s'} p(s' \mid s, a) r(s, a, s')$.

6. **Planning in MDPs**

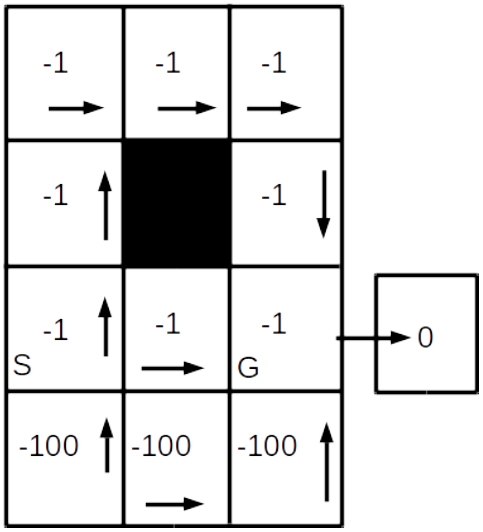
Consider a gridworld with the layout below. From each square, the agent may move into an adjoining square (up, down, left, right) or stay in place. If a policy specifies a move into a square which does not exist (i.e. down from one of the squares in the bottom row), the agent stays in place. Actions are deterministic, that is, they always have their intended effect. We use an infinite horizon with discount $\gamma = 1$. [This keeps the math simple in this example]

The robot starts in the state marked with an S. Upon reaching the state marked G the agent transitions into an absorbing state where it stays forever. **The rewards associated with a state are the reward for taking any action from that state**.

Recall the policy improvement step in policy iteration (where V^π is the value function of the current policy):

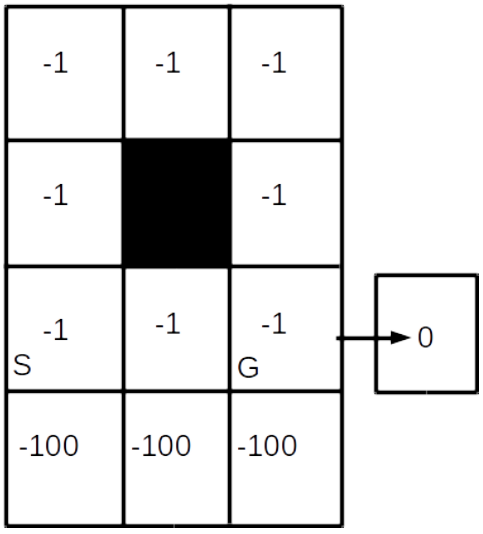
$$\pi'(s) \leftarrow \arg\max_{a \in A} \left[r(s,a) + \gamma \sum_{s' \in S} p(s' \mid s, a) V^\pi(s') \right], \quad \forall s$$

(a) Suppose that we follow the policy given by the arrows. What is the MDP value of each state under this policy? [You can figure this out by inspection of the policy and the environment]



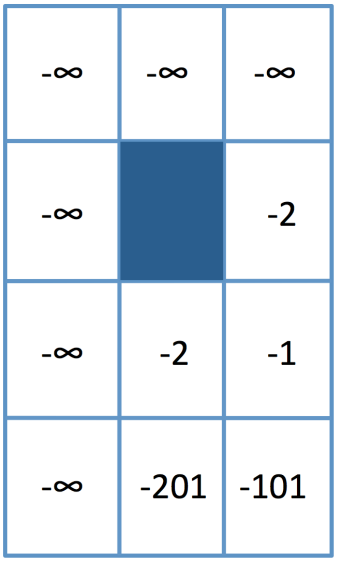
(b) Can this policy be improved? To check this, (1) use policy improvement and draw the adjusted policy and (2) compute the new value function in each state.

(c) Is the new policy optimal? [Hint: You should be able to argue yes/no directly, without doing another round of policy iteration.]



Solution:

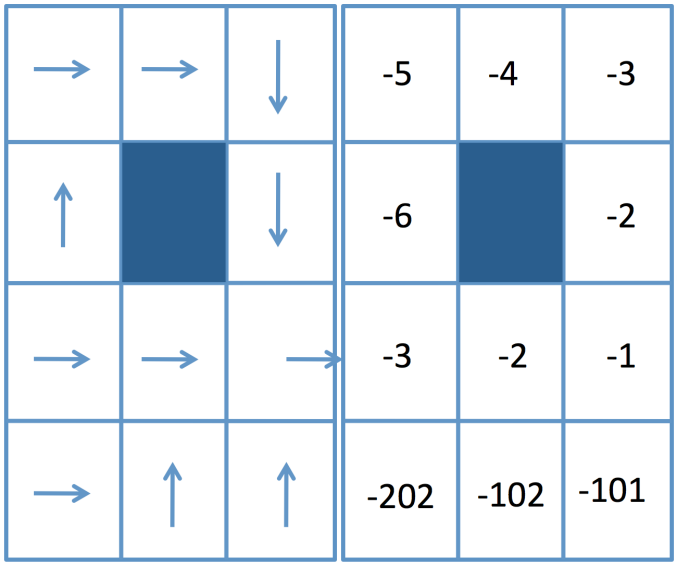
(a) See the following figure for the MDP value of each state under this policy. [Note: It is $-\infty$ in states for which the policy does not escape to the goal state because $\gamma = 1$ and thus there is no discount.]



(b) See the following figure for the updated policy and MDP value function based on one round of policy iteration.

Here, we break ties by leaving the action unchanged if there is no better action relative to the last round of policy iteration.

We see the new policy is better in a number of states.



(c) The new policy is still not optimal; e.g, it moves right in the bottom-left state, whereas it would be optimal to move up. [Similarly, it moves up in the state to the left of the missing position, when it would be better to move down.]

7. **Reinforcement learning**

The update rule for **SARSA** reinforcement learning is:

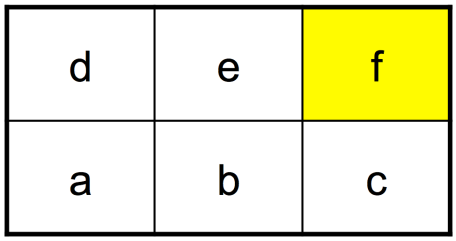
$$Q(s,a) \leftarrow Q(s,a) + \alpha[(r + \gamma Q(s',a')) - Q(s,a)].$$

(a) What are the different quantities, how are they generated (e.g., which by the agent, which from the environment), and what is the idea of the update?

(b) What is meant by 'on-policy' and 'off-policy' reinforcement learning, and is SARSA an on-policy or off-policy method?

(c) What does it mean to **exploit** in the context of reinforcement learning?

(d) Consider this simple MDP world, where the reward is 100 for any action taken in state f and 0 in all other states and actions are deterministic (thus 'up' always moves 'up').



Assume the Q-values are initialized to 0, and the agent is initially in state c. What are the updates made by SARSA following each action (for $\alpha = 0.9$ and $\gamma = 0.9$).

Assume that no update is possible until the values of s, a, r, s', a' are all well-defined.

i. up (to state f)
ii. left (to state e)
iii. right (to state f)
iv. down (to state c)

Solution:

(a) s is current state, a is selected via an agent's policy such as ϵ-greedy, reward is given by the MDP for $r(s,a)$, state s' is given by the MDP for $p(s' \mid s,a)$, action a' is given by the agent's policy.

The idea is to use a single step observation to adjust the Q value for the state-action (s,a) closer to that which is consistent with the policy being followed by the agent. Eventually we hope Bellman equations will hold the and the Q values correspond to those of the optimal policy.

(b) On-policy: with enough observations, and a learning rate that becomes small in the limit, the Q-values learned will correspond to those of the policy we follow while learning (i.e., the one that we converge to as a result of learning, which would involve ϵ-exploration if using ϵ-greedy).

Off-policy: with enough observations, and a learning rate that becomes small in the limit, the Q-values learned will correspond to the optimal policy.

SARSA is on-policy.

(c) Exploit: this means to follow $\max_a Q(s,a)$ in state s, rather than also explore (e.g., by taking some other action with small probability $\epsilon > 0$).

(d) SARSA updates:

- up (to state f): have $s = c, a = up, r = 0, s' = f, a' =$??, and we cannot do an update yet
- left (to state e): now have $s = c, a = up, r = 0, s' = f, a' = left$, and we can do update to $Q(c, up)$:

$$Q'(c, up) \leftarrow Q(c, up) + 0.9((0 + 0.9Q(f, left)) - Q(c, up)) = 0 + 0.9((0 + 0.9(0)) - 0) = 0$$

- right (to state f): now have $s = f, a = left, r = 100, s' = e, a' = right$, and we can do an update to $Q(f, left)$:

$$\begin{aligned} Q'(f, left) &\leftarrow Q(f, left) + 0.9((100 + 0.9Q(e, right)) - Q(f, left)) \\ &= 0 + 0.9((100 + 0.9(0)) - 0) = 90 \end{aligned}$$

- down (to state c): now we have $s = e, a = right, r = 0, s' = f, a' = down$, and we can do an update to $Q(e, right)$:

$$\begin{aligned} Q'(e, right) &\leftarrow Q(e, right) + 0.9((0 + 0.9Q(f, down)) - Q(e, right)) \\ &= 0 + 0.9((0 + 0.9(0)) - 0) = 0 \end{aligned}$$

8. **K-Means**

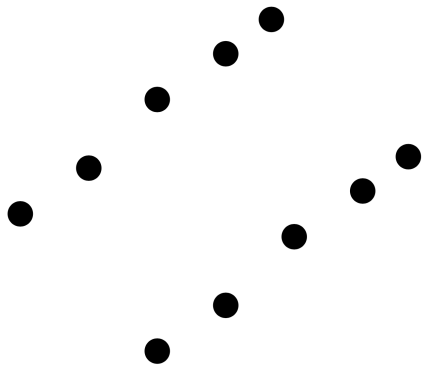
In K-Means, we are given a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and a fixed number of clusters K. Our aim is to find cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ that represent the data.

(a) Define the K-Means loss function.

(b) What two steps does Lloyd's algorithm repeat in order to find a good clustering?

(c) What is the asympotic run-time of each step of Lloyd's algorithm, as a function of the number of examples N and the number of clusters K?

(d) Given data that falls on two parallel diagonal lines as shown below, can Lloyd's algorithm with $K = 2$ find two clusters, such that each line is in one of the clusters?



Solution:

(a) The objective is to find prototypes and an assignment to minimize

$$\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||\mathbf{x}_n - \boldsymbol{\mu}_k||^2$$

where r_n is a one-hot vector with 1 in the position corresponding to the index of the assigned cluster for the point $\mathbf{x}_n$.

(b) Step 1: assign each example to the closest prototype; step 2: for each cluster k, set $\boldsymbol{\mu}_k$ to the centroid of the assigned examples

$$\boldsymbol{\mu}_k := \frac{1}{N_k}\sum_{n} r_{nk}\mathbf{x}_n$$

where $N_k = \sum_n r_{nk}$ and $r_{nk} = 1$ if $\mathbf{x}_n$ assigned to cluster k, and 0 otherwise.

(c) Step 1 takes time NK because each example is checked for the closest of K prototypes.

Step 2 takes time N because each example is in exactly one cluster, and this is used once in the averaging step.

Overall, the complexity is $N + NK$ each iteration, and $O(NK)$. [Note: We choose to ignore the dependence on number of features D.]

(d) Yes, for a suitable initialization.

Consider two clusters assignments, one to the top line and one to the bottom line, where the cluster centers lie approximately at the middle of the respective line segments.

We need to check this is a stable assignment for K-means.

In particular, is each point in a cluster closer to its centroid than the centroid of the other cluster?

This looks to be true, and thus K-means would converge on such a clustering if such centroids were ever found during LLoyd's algorithm.

Thus it is *possible* for K-means to find this clustering— so long as an appropriate initialization is given! One way to see this is that it would work if the prototypes were simply initialized to the cluster centers.

9. **Hidden Markov Models**

Consider a weather domain, with observations $\mathbf{x}_t \in \{D, R\}$ (dry, rain) and hidden state $\mathbf{s}_t \in \{C, S\}$ (cloud, sun). Assume the following parameters:

- initial prob: $p(\mathbf{s}_1 = C) = 0.7$
- transition

$p(\mathbf{s}_{t+1} \mid \mathbf{s}_t)$		Next State C	Next State S
State	C	0.8	0.2
	S	0.1	0.9

- output

$p(\mathbf{x}_t \mid \mathbf{s}_t)$		Output D	Output R
State	C	0.25	.75
	S	0.6	0.4

(a) For a general HMM, if the total number of timesteps is n and $t < n$ is a timestep in the middle of the sequence, why is $p(\mathbf{s}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) \neq p(\mathbf{s}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_t)$? (An informal answer is fine.)

(b) (Forward-backward algorithm). Now suppose we observe $\mathbf{x}_1 = R$, $\mathbf{x}_2 = R$.
We can calculate:

$$\alpha_1(\mathbf{s}_1) = \begin{cases} 0.525 & \text{, if } \mathbf{s}_1 = C \\ 0.12 & \text{, if } \mathbf{s}_1 = S \end{cases}$$

Use

$$\alpha_2(\mathbf{s}_2) = p(\mathbf{x}_2 \mid \mathbf{s}_2) \sum_{\mathbf{s}_1} p(\mathbf{s}_2 \mid \mathbf{s}_1) \alpha_1(\mathbf{s}_1)$$

to compute the α_2-values.

(c) We have $\beta_2(\mathbf{s}_2) = 1$. In addition, we can calculate:

$$\beta_1(\mathbf{s}_1) = \begin{cases} 0.68 & \text{, if } \mathbf{s}_1 = C \\ 0.435 & \text{, if } \mathbf{s}_1 = S \end{cases}$$

Use these quantities, and

$$p(\mathbf{s}_2 \,|\, \mathbf{x}_1, \mathbf{x}_2) \propto \alpha_2(\mathbf{s}_2)$$
$$p(\mathbf{s}_1 \,|\, \mathbf{x}_1, \mathbf{x}_2) \propto \alpha_1(\mathbf{s}_1)\beta_1(\mathbf{s}_1)$$

to infer the values of $p(\mathbf{s}_1 \,|\, \mathbf{x}_1, \mathbf{x}_2)$ and $p(\mathbf{s}_2 \,|\, \mathbf{x}_1, \mathbf{x}_2)$.

(d) Use $p(\mathbf{x}_1, \mathbf{x}_2) = \sum_{\mathbf{s}_t} \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t)$ to calculate the likelihood of the data.

Solution:

(a) The additional observations from $t+1$ to n may also be informative as to the hidden state at period t. For example, the observation at $\mathbf{x}_{t+1}$ might only be possible if $\mathbf{s}_t$ has a value that allows for $\mathbf{s}_{t+1}$ to have a value that can generate $\mathbf{x}_{t+1}$.

(b) Calculations for the $\alpha_2(\mathbf{s}_2)$ values are:

$\mathbf{s}_1$	$\mathbf{s}_2$	$p(\mathbf{s}_2 \,\vert\, \mathbf{s}_1)$	$p(R \mid \mathbf{s}_2)$	$\alpha_1(\mathbf{s}_1)$	$\times$ (product)
C	C	0.8	0.75	0.525	0.315
C	S	0.2	0.4	0.525	0.042
S	C	0.1	0.75	0.12	0.009
S	S	0.9	0.4	0.12	0.0432

and then summing out $\mathbf{s}_1$, we obtain

$\mathbf{s}_2$	$\alpha_2(\mathbf{s}_2)$
C	0.315+0.009=0.324
S	0.042+0.0432=0.0852

(c) For $t=1$, we have

$$p(\mathbf{s}_1 \mid R, R) \propto \alpha_1(\mathbf{s}_1)\beta_1(\mathbf{s}_1)$$

For $\mathbf{s}_1 = C$:

$$\alpha_1(C)\beta_1(C) = (0.525)(0.68) = 0.357$$

For $\mathbf{s}_1 = S$:

$$\alpha_1(S)\beta_1(S) = (0.12)(0.435) = 0.0522$$

We conclude that $p(\mathbf{s}_1 = C \,|\, R, R) \approx 0.872$ and $p(\mathbf{s}_1 = S \,|\, R, R) \approx 0.128$.

For $t=2$, we have

$$p(\mathbf{s}_2 \mid R, R) \propto \alpha_2(\mathbf{s}_2)$$

For $\mathbf{s}_2 = C$:

$$\alpha_2(C) = (0.324) = 0.324$$

For $\mathbf{s}_2 = S$:

$$\alpha_2(S) = (0.0852) = 0.0852$$

We conclude that $p(\mathbf{s}_2 = C \,|\, R, R) \approx 0.792$ and $p(\mathbf{s}_2 = S \,|\, R, R) \approx 0.208$.

(d) The likelihod of the data is

$$p(R, R) = \sum_{\mathbf{s}_1} \alpha_1(\mathbf{s}_1)\beta_1(\mathbf{s}_1) = (0.525)(0.68) + (0.12)(0.435) = 0.4092.$$

Equivalently, we could calculate using $t=2$ alpha and beta values, and get

$$p(R, R) = \sum_{\mathbf{s}_2} \alpha_2(\mathbf{s}_2)\beta_2(\mathbf{s}_2) = (0.324)(1) + (0.0852)(1) = 0.4092$$

10. **Mean of a Mixture Model**

For some class conditional distribution p_{class}, the details of which don't matter for this question, we are given a mixture model of the form

$$p(\mathbf{x}; \{\boldsymbol{\pi}_k\}_{k=1}^K, \theta) = \sum_{k=1}^K \theta_k \, p_{\text{class}}(\mathbf{x} \,|\, \mathbf{z} = C_k; \boldsymbol{\pi}_k)$$

where example $\mathbf{x} \in \mathbb{R}^D$.

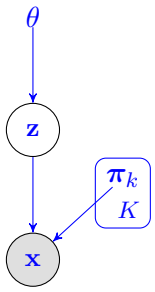
(a) Draw a graphical model with plates to show the form of this mixture distribution for a single example $\mathbf{x}$. [Note: If you're unfamiliar with the idea of "plate notation" for graphical models, take a quick look at p.363-365 in Bishop's book.]

(b) Suppose that the mean of the class-conditional distribution for component k is given by $\boldsymbol{\mu}_k$. Show that the mean of the overall mixture model is given by

$$\mathbb{E}[\mathbf{x}] = \sum_{k=1}^K \theta_k \boldsymbol{\mu}_k.$$

Solution:

(a) We draw this for a single $\mathbf{x}$ variable. Since $\boldsymbol{\pi}_k$ and θ are parameters that are not random variables, we don't have circles around them.



(b) For this, it is convenient to work with latent variable $\mathbf{z}$. We have

$$\begin{aligned}\mathbb{E}[\mathbf{x}] &= \sum_{k=1}^{K} p(\mathbf{z} = C_k; \theta)\mathbb{E}_{\mathbf{x}\sim p_{\text{class}}(\mathbf{x}|\mathbf{z}=C_k;\boldsymbol{\pi}_k)}[\mathbf{x}] \\ &= \sum_{k=1}^{K} \theta_k \boldsymbol{\mu}_k\end{aligned}$$

11. **Expectation Maximization**

We have a collection of binary images $\mathbf{x}_1, \ldots, \mathbf{x}_N$, each of which is 5×5. We treat each image $\mathbf{x}_n$ as a 25-dimensional binary vector where the dth pixel is $x_{n,d}$. We model an image as coming from a mixture distribution, with a product-of-Bernoulli distribution for each component k:

$$p(\mathbf{x}_n | \boldsymbol{z}_n = C_k; \boldsymbol{\mu}_k) = \prod_{d=1}^{25} \mu_{k,d}^{x_{n,d}} (1 - \mu_{k,d})^{1 - x_{n,d}} \qquad \mathbf{x}_n \in \{0,1\}^{25} \qquad \boldsymbol{\mu}_k \in [0,1]^{25} .$$

Each of the K components has parameters $\boldsymbol{\mu}_k$, where each dimension $\mu_{k,d}$ specifies the probability that pixel d is black in an example from this component. The mixture weights are $\{\theta_k\}_{k=1}^K$ and known. You will use EM to estimate the $\{\boldsymbol{\mu}_k\}$ parameters.

(a) Write down the probability of generating a single image $\mathbf{x}$, i.e., $p(\mathbf{x}; \{\boldsymbol{\mu}_k\}_{k=1}^K, \theta)$.

(b) What are the "latent variables" in this model? Draw the plate diagram for this model, writing it for N examples, and indicating what is known and unknown. [Hint: See the previous question for a pointer to plate diagrams.]

(c) In the E-step, you find the probability with which example $\mathbf{x}_n$ belongs to each component fixing the parameters $\{\boldsymbol{\mu}_k\}_{k=1}^K$; i.e., $q_{n,k} = p(\mathbf{z}_n = C_k \,|\, \mathbf{x}_n; \{\boldsymbol{\mu}_k\}, \theta)$ for each k. Derive the expression for $q_{n,k}$.

(d) In the M-step, you update the parameters $\{\boldsymbol{\mu}_k\}_{k=1}^K$. Write down the expression for this, making use of the $\mathbf{q}$ values. [No need to derive the answer. As a hint, for the supervised case with class $\mathbf{z}_n$ of each image (one-hot coded), the MLE for the parameters of class k would be

$$\mu_{k,d} = \frac{\sum_{n=1}^N z_{n,k} x_{n,d}}{\sum_{n=1}^N z_{n,k}}$$

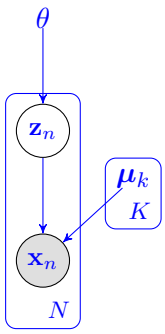
(intuitively, the percentage of times pixel d was black for the data in class k).]

Solution:

(a) Sum out over possible values of the latent variables $\mathbf{z}$:

$$p(\mathbf{x} \mid \{\boldsymbol{\mu}_k\}_{k=1}^K; \theta) = \sum_{k=1}^{K} p(\mathbf{z} = C_k;\ \theta) p(\mathbf{x} \mid \mathbf{z} = C_k; \boldsymbol{\mu}_k) = \sum_{k=1}^{K} \theta_k \prod_{d=1}^{25} \mu_{k,d}^{x_d} (1 - \mu_{k,d})^{1 - x_d}$$

(b) Plate diagram. The mixture weights θ and the parameters $\boldsymbol{\mu}_k$ for each component k are parameters and drawn without a circle.



(c) For example $\mathbf{x}_n$, we need to compute

$$q_{n,k} = p(\mathbf{z}_n = C_k \mid \mathbf{x}_n; \{\boldsymbol{\mu}_k\}, \theta)$$

We do this by applying Bayes' rule:

$$q_{n,k} = p(\mathbf{z}_n = C_k \mid \mathbf{x}_x; \{\boldsymbol{\mu}_k\}, \theta) \quad = \quad \frac{1}{Z} p(\mathbf{z}_n = C_k; \theta) p(\mathbf{x}_n \mid \mathbf{z}_n = C_k; \boldsymbol{\mu}_k)$$

The normalization term can be computed as $Z = \sum_\ell q_{n,\ell}$.

(d) For EM we instead utilize the predicted assignments of each example to each component, and have

$$\mu_{k,d} = \frac{\sum_{n=1}^{N} q_{n,k} x_{n,d}}{\sum_{n=1}^{N} q_{n,k}}$$

12. **PCA**

Consider a mean-centered data set of four points $\mathbf{x}_1 = (1, 0)$, $\mathbf{x}_2 = (-1, 0)$, $\mathbf{x}_3 = (0, -2)$, $\mathbf{x}_4 = (0, 2)$.

(a) Compute the empirical covariance matrix $\mathbf{S}$ for this dataset.

(b) Draw a rough sketch of the distribution $\mathcal{N}(0, \mathbf{S})$ formed with this covariance matrix.

(c) If we were to run PCA on this data, what algebraic properties of the empirical covariance matrix correspond to first and second principal components? What are the first and second principal components in this example? [Hint: You should be able to easily recognize them without computation.]

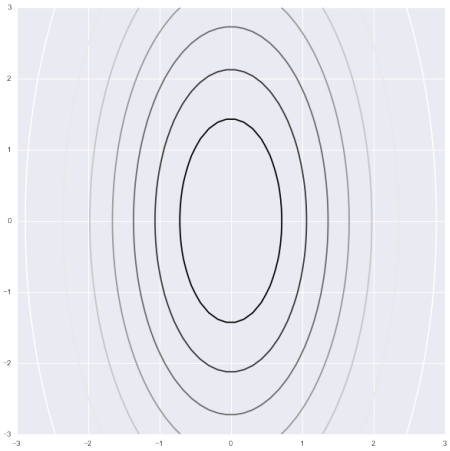
(d) Graph the four points after running PCA and projecting down to a single dimension. What is lost in this transformation?

Solution:

(a) The empirical covariance matrix is defined as $\frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$.

$$\frac{1}{4}\left(\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} + \begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 4\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 4\end{bmatrix}\right) = \begin{bmatrix}0.5 & 0\\ 0 & 2\end{bmatrix}$$

(b) Here's what it looks like. We'd hope for a rough sketch of this form, i.e. that it is an axis-aligned ellipse. The particular position of the contours wouldn't matter.



(c) The first and second principal components correspond to the eigenvectors with the first and second highest eigenvalues of the empirical covariance matrix, respectively. In this case they have a simple form $\mathbf{u}_1 = [0,1]^\top$ with $\mathbf{S}\mathbf{u}_1 = 2\mathbf{u}_1$ and $\mathbf{u}_2 = [1,0]^\top$ with $\mathbf{S}\mathbf{u}_2 = 0.5\mathbf{u}_2$. This can also be seen visually from the data.

(d) Projecting to one dimension corresponds to projecting onto the first principal component,

$$\mathbf{z}_n = (\mathbf{x}_n^\top \mathbf{u}_1)\mathbf{u}_1.$$

Given $\mathbf{u}_1$, the projections take the simple form of lying along the y-axis:

$$\mathbf{z}_1 = [0,0]^\top, \mathbf{z}_2 = [0,0]^\top, \mathbf{z}_3 = [0,-2]^\top, \mathbf{z}_4 = [0,2]^\top$$

This has the effect of collapsing $\mathbf{x}_1$ and $\mathbf{x}_2$ to the same point, but keeps the distinction between $\mathbf{x}_3$ and $\mathbf{x}_4$.

13. **PCA on transformed data**

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be a data matrix, with the j^{th} row corresponding to the j^{th} observation $\mathbf{x}_j^\top \in \mathbb{R}^d$. Assume $\mathbf{X}$ is centered, and suppose that the PCA of this data has principal components (eigenvectors) $\mathbf{v}_1, \ldots, \mathbf{v}_d$ with associated variances (eigenvalues) $\lambda_1 \geq \ldots, \geq \lambda_d \geq 0$. Now, let $\mathbf{Q}$ be a $d \times d$ orthonormal matrix and let $\mathbf{y}_j = \mathbf{Q}\mathbf{x}_j$ for all $j = 1, \ldots, n$.

(a) In words, briefly explain how the matrix $\mathbf{Q}$ transforms the data $\mathbf{x}_j$. Then, find an expression for $\mathbf{Y} \in \mathbb{R}^{n \times d}$, the new data matrix where the j^{th} row corresponds to $\mathbf{y}_j^\top$, in terms of $\mathbf{X}$ and $\mathbf{Q}$.

(b) Show that the PCA of $\mathbf{Y}$ yields principal components $\mathbf{Q}\mathbf{v}_1, \ldots, \mathbf{Q}\mathbf{v}_d$ with associated variances $\lambda_1, \ldots, \lambda_d$. Briefly explain why this result is intuitive.

(c) Now, suppose that $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ is the SVD of $\mathbf{X}$. Noting that the columns of $\mathbf{V}$ contain the eigenvectors of the covariance matrix of $\mathbf{X}$, find an equation that relates the singular values of $\mathbf{S}$, s_j, with the PCA variances λ_j (assuming that the diagonal values of $\mathbf{S}$ are sorted in decreasing order).

(d) With a graphical explanation, show how performing PCA on non-centered data can yield incorrect principal components and variances.

[21] Adapted from Stat 185

Solution:

(a) Orthonormal matrices represent transformations composed of rotations and reflections, so the data is being changed through these length-preserving transformations. Since $\mathbf{y}_j = \mathbf{Q}\mathbf{x}_j$, we know that $\mathbf{y}_j^\top = \mathbf{x}_j^\top \mathbf{Q}^\top$, which implies that $\mathbf{Y} = \mathbf{X}\mathbf{Q}^\top$.

(b) Let $\mathbf{\Sigma}_X = \frac{1}{n}\mathbf{X}^\top\mathbf{X}$ denote the covariance matrix of $\mathbf{X}$. Then, consider the covariance matrix of $\mathbf{Y}$:

$$\begin{aligned}\mathbf{\Sigma}_Y &= \frac{1}{n}\mathbf{Y}^\top\mathbf{Y} \\ &= \frac{1}{n}\left(\mathbf{X}\mathbf{Q}^\top\right)^\top\left(\mathbf{X}\mathbf{Q}^\top\right) \\ &= \frac{1}{n}\mathbf{Q}\mathbf{X}^\top\mathbf{X}\mathbf{Q}^\top \\ &= \mathbf{Q}\mathbf{\Sigma}_X\mathbf{Q}^\top\end{aligned}$$

Note that in order for this derivation to be correct, we must show that $\mathbf{Y}$ is centered given that $\mathbf{X}$ is centered; this is left as an additional exercise. Since we know the principal components and variances of the data $\mathbf{X}$, we can write $\mathbf{\Sigma}_X = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where the columns of $\mathbf{V}$ are the eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_d$ and $\mathbf{\Lambda}$ is diagonal with entries $\lambda_1, \ldots, \lambda_d$. This means that

$$\mathbf{\Sigma}_Y = \mathbf{Q}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top\mathbf{Q}^\top = (\mathbf{Q}\mathbf{V})\mathbf{\Lambda}(\mathbf{Q}\mathbf{V})^\top$$

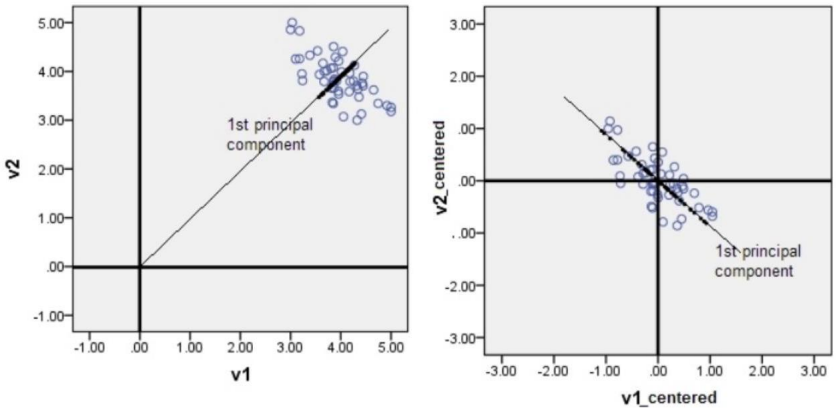
Therefore, the eigendecomposition of $\mathbf{\Sigma}_Y$ yields eigenvectors which are the columns of $\mathbf{Q}\mathbf{V}$ and eigenvalues which are the diagonal entries of $\mathbf{\Lambda}$. This corresponds exactly to $\mathbf{Y}$ having principal components $\mathbf{Q}\mathbf{v}\mathbf{v}_1, \ldots, \mathbf{Q}\mathbf{v}_d$ with associated variances $\lambda_1, \ldots, \lambda_d$. This result makes sense because since $\mathbf{Q}$ transforms the data through rotations and reflections, the principal components will also be transformed via the same transformation. Since $\mathbf{Q}$ is distance-preserving, this also means that the variances are unaffected.

(c) Given the SVD of $\mathbf{X}$, we have

$$\begin{aligned}\mathbf{\Sigma}_X &= \frac{1}{n}\mathbf{X}^\top\mathbf{X} \\ &= \frac{1}{n}\left(\mathbf{U}\mathbf{S}\mathbf{V}^\top\right)^\top\left(\mathbf{U}\mathbf{S}\mathbf{V}^\top\right) \\ &= \frac{1}{n}\mathbf{V}\mathbf{S}^\top\mathbf{U}^\top\mathbf{U}\mathbf{S}\mathbf{V}^\top \\ &= \frac{1}{n}\mathbf{V}\mathbf{S}^\top\mathbf{S}\mathbf{V}^\top \\ &= \mathbf{V}\frac{\mathbf{S}^2}{n}\mathbf{V}^\top\end{aligned}$$

The fourth equality is due to $\mathbf{U}$ being unitary, so $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$, while the fifth equality is due to $\mathbf{S}$ being diagonal, so $\mathbf{S}^\top = \mathbf{S}$. From this derivation, we arrive at the eigendecomposition of $\mathbf{\Sigma}_X$, so we have $\mathbf{\Lambda} = \frac{\mathbf{S}^2}{n}$. This means that the singular values s_j of $\mathbf{X}$ are related to the PCA variances λ_j through the equation $\lambda_j = \frac{s_j^2}{n}$.

(d) Consider the figure below:



The general direction of the principal components can be explained by noting that PCA minimizes the reconstruction error, that is, it minimizes the sum of the distances from the original data and the data projected onto the subspace spanned by the principal component(s). By not centering the data (left), the principal component is very different from the true direction of maximal variance when the data is centered (right).

14. **Expectation maximization on multinomial data** [2]

Recall that the multinomial distribution is a generalization of the binomial distribution where there are $k \geq 3$ possible categories. If $\mathbf{x} = (x_1, \ldots, x_k) \sim \text{Mult}(n, \boldsymbol{\pi})$, its PMF is given by

$$p(\mathbf{x}) = \frac{n!}{x_1! \cdots x_k!} \pi_1^{x_1} \ldots \pi_k^{x_k}$$

where $x_j \geq 0$ for all $j = 1, \ldots, k, \sum_{j=1}^k x_j = n$, and $\sum_{j=1}^k \pi_j = 1$. Suppose we have a single observation $\mathbf{x} = (x_1, x_2, x_3, x_4)$ from a Mult $(n, \boldsymbol{\pi}_\theta)$ distribution, where

$$\boldsymbol{\pi}_\theta = \left(\frac{1}{2} + \frac{1}{4}\theta, \frac{1}{4}(1-\theta), \frac{1}{4}(1-\theta), \frac{1}{4}\theta \right).$$

However, assume that the complete data is given by $\mathbf{z} = (z_0, z_1, x_2, x_3, x_4) \sim$ Mult $(n, \boldsymbol{\pi}_\theta^*)$, where

$$\boldsymbol{\pi}_\theta^* = \left(\frac{1}{2}, \frac{1}{4}\theta, \frac{1}{4}(1-\theta), \frac{1}{4}(1-\theta), \frac{1}{4}\theta \right).$$

That is, we have the latent variables z_0 and z_1, but we only observe $x_1 = z_0 + z_1$.

(a) Write out both the observed data log-likelihood, $\ell(\theta; \mathbf{x})$, and the complete data log-likelihood, $\ell(\theta; \mathbf{x}, z_1)$, up to an additive constant with respect to θ. Why does the complete data log-likelihood not depend on z_0 ?

(b) What is the conditional distribution $p(z_1 \mid \mathbf{x}, \theta)$? Briefly justify your answer.

(c) Let θ^t denote the current value of θ at the t^{th} iteration of the EM algorithm. Write out the ELBO function ELBO $\left(\theta \mid q, \theta^t\right)$, where $q(z_1)$ is the posterior distribution $p\left(z_1 \mid \mathbf{x}, \theta^t\right)$. You may leave your answer in terms of named distributions and the complete data log-likelihood (i.e., you do not need to plug in the PMF/PDFs of any distributions in your answer).

(d) Now, write out the expression which must be maximized in the M-step of the EM algorithm. Simplify as much as possible: your answer should be in the form $\underset{\theta}{\text{argmax}} g(\theta)$, where $g(\theta)$ contains no expectations or other terms not dependent on θ.

(e) Finally, compute θ^{t+1} in terms of θ^t and $\mathbf{x}$ by finding the maximum value of the expression you derived in part (d).

[22] Adapted from http://www.columbia.edu/~mh2078/MachineLearningORFE/EM_Algorithm.pdf

Solution:

(a) The observed data likelihood is given by

$$L(\theta; \mathbf{x}) = \frac{n!}{x_1! x_2! x_3! x_4!} \left(\frac{1}{2} + \frac{1}{4}\theta\right)^{x_1} \left(\frac{1}{4}(1-\theta)\right)^{x_2} \left(\frac{1}{4}(1-\theta)\right)^{x_3} \left(\frac{1}{4}\theta\right)^{x_4}$$

Therefore, the observed data log-likelihood is

$$\ell(\theta; \mathbf{x}) = x_1 \log\left(\frac{1}{2} + \frac{1}{4}\theta\right) + (x_2 + x_3)\log(1-\theta) + x_4 \log\theta + C$$

where C is a constant not depending on θ. Similarly, the complete data log-likelihood is

$$\ell\left(\theta; \mathbf{x}, z_1\right) = z_0 \log\frac{1}{2} + (z_1 + x_4)\log\theta + (x_2 + x_3)\log(1-\theta) + C$$

Note that the first term has no dependence on θ, so it can be absorbed into the constant C. Thus, the complete data log-likelihood does not depend on z_0.

(b) The posterior distribution is

$$p\left(z_1 \mid \mathbf{x}, \theta\right) \sim \text{Bin}\left(x_1, \frac{\theta/4}{1/2 + \theta/4}\right)$$

This is because given x_1, we know that the conditional distribution of z_1 reduces to a binomial distribution, where $x_1 = z_0 + z_1$ is the number of trials, and the probability has been appropriately normalized.

(c) The ELBO function is

$$\begin{aligned}
\text{ELBO}\left(\theta \mid q, \theta^t\right) &= \mathbb{E}_{z_1 \sim q(z_1)}\left[\log\frac{p\left(\mathbf{x}, z_1 \mid \theta\right)}{q\left(z_1\right)}\right] \\
&= \mathbb{E}_{z_1 \sim q(z_1)}\left[\ell\left(\theta; \mathbf{x}, z_1\right)\right] - \mathbb{E}_{z_1 \sim q(z_1)}\left[\log q\left(z_1\right)\right]
\end{aligned}$$

where $q\left(z_1\right) \sim \text{Bin}\left(x_1, \frac{\theta^t/4}{1/2+\theta^t/4}\right)$. Note that $q\left(z_1\right)$, as found in the E-step of the EM algorithm, has no dependence on θ.

(d) In the M-step of the EM algorithm, we want to maximize the ELBO function found in part (c) with respect to θ. The second term $\mathbb{E}_{z_1 \sim q(z_1)}\left[\log q\left(z_1\right)\right]$ has no dependence on θ, so we have

$$\begin{aligned}
\underset{\theta}{\text{argmax}}\, \text{ELBO}\left(\theta \mid q, \theta^t\right) &= \underset{\theta}{\text{argmax}} \mathbb{E}_{z_1 \sim q(z_1)}\left[\ell\left(\theta; \mathbf{x}, z_1\right)\right] \\
&= \underset{\theta}{\text{argmax}} \mathbb{E}_{z_1 \sim q(z_1)}\left[z_1 \log\theta\right] + x_4 \log\theta + (x_2 + x_3)\log(1-\theta) \\
&= \underset{\theta}{\text{argmax}} x_1 p^t \log\theta + x_4 \log\theta + (x_2 + x_3)\log(1-\theta),
\end{aligned}$$

where $p^t = \frac{\theta^t/4}{1/2+\theta^t/4}$.

(e) To maximize the expression found in part (d) with respect to θ, we take the derivative and set it equal to 0 :

$$0 = \frac{x_1 p^t + x_4}{\theta} - \frac{x_2 + x_3}{1 - \theta}$$

Solving for θ yields

$$\theta^{t+1} = \frac{x_4 + x_1 p^t}{x_2 + x_3 + x_4 + x_1 p^t}$$

15. **Markov decision process of a caterpillar** [3]

George the very hungry caterpillar loves to eat. Because George wants to grow up and become a butterfly, he is trying to eat as many calories as possible. At every meal, he decides between eating watermelon and strawberry ice cream. Eating watermelon gives him 4 calories, while eating ice cream gives him 10 calories. However, eating too much ice cream may cause George to become sick, and eating ice cream while sick may cause him to die! (This would be bad because then George can no longer eat.) On the other hand, eating watermelon will keep George healthy and make him healthy if he is sick. George will always be in one of these three states: healthy, sick, or dead-the transitions are given in the table below.

Health condition	Watermelon or Ice Cream?	Next condition	Probability
healthy	watermelon	healthy	1
healthy	ice cream	healthy	1/4
healthy	ice cream	sick	3/4
sick	watermelon	healthy	1/4
sick	watermelon	sick	3/4
sick	ice cream	sick	7/8
sick	ice cream	dead	1/8

(a) Model this problem as an MDP by specifying the states $\mathcal{S}$, actions $\mathcal{A}$, transition functions $T^a(s, s') = P(s' \mid s, a)$, and reward function $R(a)$. Note that in this context, the reward function does not depend on the current state s or subsequent state s'.

(b) Run value iteration for 2 iterations on this MDP with $\gamma = 0.8$, specifying how the functions $Q_t(s, a)$ and $V_t(s)$ change over each iteration. That is, start with $Q_0 = 0, V_0 = 0$ and compute $Q_1(s, a), V_1(s), Q_2(s, a)$, and $V_2(s)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.

(c) Let π_0 be the policy that George always eats ice cream regardless of his health condition. Find the value of this policy, $V^{\pi_0}(s)$, for each state $s \in \mathcal{S}$ in terms of the discount factor γ. Then, for each state-action pair (s, a), find $Q^{\pi_0}(s, a)$ in terms of V^{π_0}.

(d) Now, run policy iteration, starting with the initial policy π_0 where George always eats ice cream. Setting $\gamma = 0.8$, use your results in part (c) to find the optimal policy. How many iterations does your algorithm take?

(e) Run policy iteration again, but now with $\gamma = 0.9$. How many iterations does it take for your policy to converge? If your optimal policy differs that of part (d), briefly describe your intuition for why this might occur.

(f) (Bonus) Does there exist a reward function $R'(a)$ with R' (watermelon) $< R'$ (ice cream) and some $\gamma \in [0, 1)$ such that the optimal policy π^* satisfies π^* (healthy) = watermelon? If so, find R' and γ, or prove that such an R' cannot exist.

[23] Adapted from CS 182

Solution:

(a) We have states $\mathcal{S} = \{H, S, D\}$ representing the health conditions healthy, sick, and dead, respectively. The actions are $\mathcal{A} = \{W, I\}$, which represent eating watermelon and ice cream, respectively. The transition functions are given as matrices below:

$$T^W(s,s') = P(s' \mid s, W) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{ccc} H & S & D \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{4} & \frac{3}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array}, \quad T^I(s,s') = P(s' \mid s, I) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{ccc} H & S & D \end{array} \\ \begin{pmatrix} \frac{1}{4} & \frac{3}{4} & 0 \\ 0 & \frac{7}{8} & \frac{1}{8} \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

where s is given by the row label and s' is given by the column label. Finally, we define $R(W) = 4$ and $R(I) = 10$.

(b) Recall the value iteration algorithm:

(1) Initialize $Q_0(s,a) = 0, V_0(s) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$.

(2) For $t = 1, 2, \ldots$, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, compute

$$Q_{t+1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P\left(s' \mid s, a\right) V_t\left(s'\right)$$

(3) For all $s \in \mathcal{S}$, set

$$V_{t+1}(s) = \max_a Q_{t+1}(s,a)$$

(4) Repeat steps 2 and 3 until convergence.

In all subsequent parts of this question, we will denote the Q function as a 3×2 matrix corresponding to all pairs $(s,a) \in \mathcal{S} \times \mathcal{A}$. We have

$$Q_1(s,a) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{cc} W & I \end{array} \\ \begin{pmatrix} 4 & 10 \\ 4 & 10 \\ 0 & 0 \end{pmatrix} \end{array}$$

and $V_1(H) = V_1(S) = 10$ (we won't specify $V(D)$ because it will always be 0). From this, we get

$$Q_2(s,a) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{cc} W & I \end{array} \\ \begin{pmatrix} 12 & 18 \\ 12 & 17 \\ 0 & 0 \end{pmatrix} \end{array}$$

with $V_2(H) = 18$ and $V_2(S) = 17$.

(c) For a policy π, we know that V^π satisfies the following equations for all $s \in \mathcal{S}$:

$$V^\pi = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} P\left(s' \mid s, \pi(s)\right) V^\pi\left(s'\right)$$

Similarly, the Q function can be evaluated for all $(s,a) \in \mathcal{S} \times \mathcal{A}$:

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P\left(s' \mid s,a\right) Q^{\pi}\left(s', \pi\left(s'\right)\right)$$

Given the policy π_0, we have the following system of equations:

$$\begin{aligned} V^{\pi_0}(H) &= 10 + \frac{3}{4}\gamma V^{\pi_0}(S) + \frac{1}{4}\gamma V^{\pi_0}(H) \\ V^{\pi_0}(S) &= 10 + \frac{7}{8}\gamma V^{\pi_0}(S) + \frac{1}{8}\gamma V^{\pi_0}(D) \\ V^{\pi_0}(D) &= 0. \end{aligned}$$

These yield the solutions

$$V^{\pi_0}(H) = \frac{320 - 40\gamma}{(4-\gamma)(8-7\gamma)}, \quad V^{\pi_0}(S) = \frac{80}{8-7\gamma}$$

Since $Q\left(s, \pi(s) = V(s)\right.$, we know that $Q^{\pi_0}(H,I) = V^{\pi_0}(H)$ and $Q^{\pi_0}(S,I) = V^{\pi_0}(S)$. Finally, we also have

$$\begin{aligned} Q^{\pi_0}(S,W) &= 4 + \frac{1}{4}\gamma V^{\pi_0}(H) + \frac{3}{4}\gamma V^{\pi_0}(S) \\ Q^{\pi_0}(H,W) &= 4 + \gamma V^{\pi_0}(H). \end{aligned}$$

(d) Recall the policy iteration algorithm:

(1) Start with some policy $\pi_0 : \mathcal{S} \to \mathcal{A}$.

(2) For $t = 0, 1, \ldots$, compute V^{π_t} and Q^{π_t} by policy evaluation (via the exact or iterative method).

(3) For all $s \in \mathcal{S}$, set

$$\pi_{t+1}(s) = \underset{a}{\operatorname{argmax}} Q^{\pi_t}(s,a).$$

(4) Repeat until $\pi_{t+1} = \pi_t$.

Using this and the equations from part (c), we have the following Q function for $\gamma = 0.8$:

$$Q^{\pi_0}(s,a) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{cc} W & I \end{array} \\ \begin{pmatrix} 34 & 37.5 \\ 31.5 & 33.3 \\ 0 & 0 \end{pmatrix} \end{array}$$

Therefore, the policy π_1 is to eat ice cream at every state (once again, disregarding state D). Thus, the policy converges after a single iteration.

(e) Now, with $\gamma = 0.9$, we get the following result:

$$Q^{\pi_0}(s,a) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{cc} W & I \end{array} \\ \begin{pmatrix} 52.5 & 53.9 \\ 47.9 & 47.1 \\ 0 & 0 \end{pmatrix} \end{array}$$

Thus, we have $\pi_1(H) = I$, but $\pi_1(S) = W$. Then, evaluating this policy yields

$$Q^{\pi_1}(s,a) = \begin{array}{c} \\ H \\ S \\ D \end{array} \begin{array}{c} \begin{array}{cc} W & I \end{array} \\ \begin{pmatrix} 57.6 & 59.5 \\ 53.5 & 52.1 \\ 0 & 0 \end{pmatrix} \end{array}$$

This yields $\pi_2(H) = I$ and $\pi_2(S) = W$, so at this point, our policy has converged after two iterations. The reason why the policy differs for different values of γ is because values of γ closer to 1 prioritize long-term gains (i.e. not dying), whereas smaller values of γ yield policies that look for immediate gain (i.e. eating as much ice cream immediately).

(f) Suppose we have reward values $R(W)$ and $R(I)$ for the reward of eating watermelon and ice cream, respectively, with $R(W) < R(I)$. Also, suppose that the optimal policy when healthy is to eat watermelon. Then, we must have $V(H) = R(W) + \gamma V(H)$, or $V(H) = (1-\gamma)^{-1}R(W)$. Furthermore, because it is the optimal policy, the value of eating vegetables when healthy must be at least the value of eating ice cream; that is, we have

$$R(W) + \gamma V(H) \geq R(I) + \frac{3}{4}\gamma V(S) + \frac{1}{4}\gamma V(H)$$

This simplifies to $V(H) \geq \frac{4}{3\gamma}(R(I) - R(W)) + V(S)$. In particular, because $R(I) - R(W) > 0$, we have $V(H) > V(S)$. Under the policy of eating vegetables while sick, we also have

$$V(S) = R(W) + \frac{3}{4}\gamma V(S) + \frac{1}{4}\gamma V(H)$$

Substituting $V(H) = \frac{1}{1-\gamma}R(W)$, we get

$$\begin{aligned} V(S) &= \frac{4}{4-3\gamma}\left(R(W) + \frac{\gamma}{4-4\gamma}R(W)\right) \\ &= \frac{4}{4-3\gamma}R(W)\left(1 + \frac{\gamma}{4-4\gamma}\right) \\ &= \frac{4}{4-3\gamma}R(W)\left(\frac{4-3\gamma}{4-4\gamma}\right) \\ &= \frac{1}{1-\gamma}R(W) \\ &= V(H). \end{aligned}$$

In fact, $V(H)$ is a lower bound on $V(S)$ because the optimal policy could be to eat ice cream while sick. In other words, the true value of $V(S)$ under the optimal policy cannot be less than $V(H)$ because existence of the policy of eating watermelon while sick forces $V(S) \geq V(H)$. Thus, we have $V(H) > V(S) \geq V(H)$, which is impossible. Thus, there are no reward values satisfying $R(W) < R(I)$ such that the optimal policy is to eat watermelon while healthy.

16. **Everything is a graphical model**

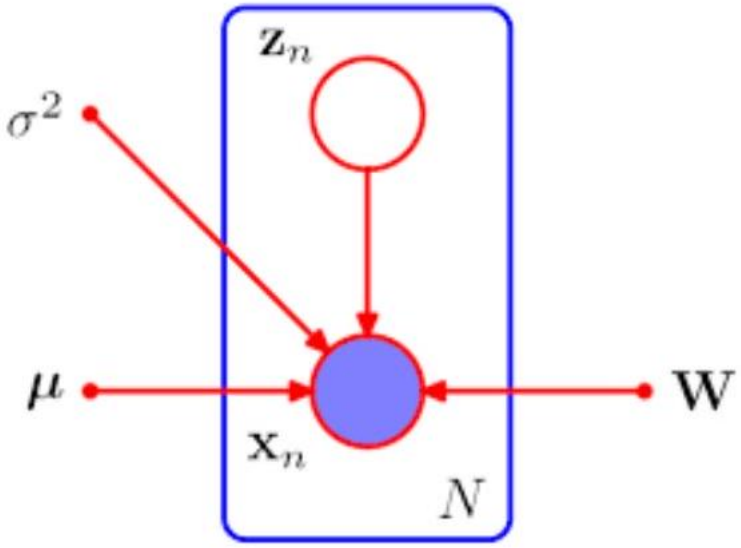
For the following models, write the following: (i) graphical model, (ii) "generative story" of the model, (iii) observed data log likelihood, (iv) ELBO function, (v) E-step (what are you maximizing, with respect to what?), (vi) M-step (what are you maximizing, with respect to what?).

(a) pPCA : latent variables $\mathbf{z}_n \sim \mathcal{N}(0, I)$, $\mathbf{z} \in \mathbb{R}^k$, observed variables $\mathbf{x}_n \mid \mathbf{z}_n \sim \mathcal{N}\left(\mathbf{W}\mathbf{z}_n\sigma^2\mathbf{I}\right)$, $\mathbf{x} \in \mathbb{R}^d$. You may find the multivariate Gaussian $\mathbf{r} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \mathbf{r} \in \mathbb{R}^d$ PDF helpful:

$$p(\mathbf{r}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-d/2}(\det \boldsymbol{\Sigma})^{-1/2} \exp - \left(\frac{(\mathbf{r} - \boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{r} - \boldsymbol{\mu})}{2}\right)$$

(b) Hidden Markov Model: latent variables $(\mathbf{s}_1, \ldots, \mathbf{s}_n), \mathbf{s}_1 \sim \text{Cat}(\boldsymbol{\theta}), \mathbf{s}_t \in [0, 1]^K$ (i.e. the s_t are one-hot encoded vectors), $\boldsymbol{\theta} \in \mathbb{R}^K$, observed variables $(\mathbf{x}_1, \ldots, \mathbf{x}_n), \mathbf{x}_t \mid \mathbf{s}_t \sim p(\mathbf{x}_t \mid \mathbf{s}_t)$ (this is arbitrary, one example is linear Gaussian noise $\mathcal{N}\left(\mathbf{D}s_t + \mathbf{E}, \sigma^2\mathbf{I}\right)$, where $\mathbf{D} \in \mathbb{R}^{K\times d}, \mathbf{E} \in \mathbb{R}^d$, and $\mathbf{I}$ is the $d \times d$ identity matrix), $\mathbf{x}_t \in \mathbb{R}^d$, latent state transition probabilities $\mathbf{T}_{ij} \in \mathbb{R}^{K\times K}, \mathbf{T}_{ij} = P(s_{t+1} = j \mid s_t = i)$.

Solution:



(a) i. Note that the parameter $\boldsymbol{\mu}$ is unnecessary if the data $\mathbf{X}$ is mean-centered.

ii. For each data point, we sample $\mathbf{z}_n$ from the latent Gaussian distribution, and then we linearly map from the latent space to the observation space. Finally, we sample Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \sigma^2 \mathbf{I}\right)$ and add it to observation.

iii. We can marginalize over $\mathbf{z}_n$ by integrating: $p\left(\mathbf{x}_n; \mathbf{W}, \sigma\right) = \int_{-\infty}^{\infty} p\left(\mathbf{x}_n, \mathbf{z}_n\right) d\mathbf{z}_n = \int_{-\infty}^{\infty} p\left(\mathbf{x}_n \mid \mathbf{z}_n\right) p\left(\mathbf{z}_n\right) d\mathbf{z}_n$ and substituting in the PDFs. Alternatively, we can use the property that the marginal over any particular variable in a multivariate Gaussian is going to be Gaussian, meaning we only need to find the mean and variance of $\mathbf{x}_n$. First, note that $\mu_{\mathbf{x}} = \mathbb{E}[\mathbf{x}] = \mathbb{E}[\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon}] = \mathbb{E}[\mathbf{W}\mathbf{z}] + \mathbb{E}[\boldsymbol{\epsilon}] = 0$ since both our noise and our $\mathbf{z}$ are zero centered. Next,

$$
\begin{aligned}
\boldsymbol{\Sigma}_{\mathbf{xx}} &= \mathbb{E}\left[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^\top\right] \\
&= \mathbb{E}\left[(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon} - 0)(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon} - 0)^\top\right] \\
&= \mathbb{E}\left[\mathbf{W}\mathbf{z}\mathbf{z}^T\mathbf{W}^\top + \boldsymbol{\epsilon}\mathbf{z}^\top\mathbf{W}^\top + \mathbf{W}\mathbf{z}\boldsymbol{\epsilon}^\top + \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top\right] \\
&= \mathbf{W}\mathbb{E}\left[\mathbf{z}\mathbf{z}^\top\right]\mathbf{W}^\top + \mathbb{E}\left[\epsilon\mathbf{z}^\top\right]\mathbf{W}^\top + \mathbf{W}\mathbb{E}\left[\mathbf{z}\boldsymbol{\epsilon}^\top\right] + \mathbb{E}\left[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top\right] \\
&= \mathbf{W}(\mathbf{I})\mathbf{W}^T + \sigma^2\mathbf{I}
\end{aligned}
$$

With these two parameters, we have $\mathbf{x} \sim \mathcal{N}\left(0, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}\right)$. The observed data log likelihood is

$$\begin{aligned}
\ell\left(\mathbf{x}_n; \mathbf{W}, \sigma\right) &= \sum_{n=1}^{N} \log p\left(\mathbf{x}_n \mid \mathbf{W}, \sigma\right) \\
&= \sum_{n=1}^{N} \log \left(\frac{1}{(2\pi)^{d/2} \left(\det \boldsymbol{\Sigma}_{\mathbf{xx}}\right)^{1/2}} \exp\left\{ -\frac{1}{2} \left(\mathbf{x}_n^{\top} \boldsymbol{\Sigma}_{\mathbf{xx}}^{-1} \mathbf{x}_n \right) \right\} \right) \\
&= \frac{Nd}{2} \log 2\pi + \log \det \left(\mathbf{W}\mathbf{W}^T + \sigma^2 \mathbf{I} \right) + \sum_{n=1}^{N} -\frac{1}{2} \left(\mathbf{x}_n^{\top} \boldsymbol{\Sigma}_{\mathbf{xx}}^{-1} \mathbf{x}_n \right)
\end{aligned}$$

iv. ELBO: Recall that ELBO is defined as $\sum_{n=1}^{N} \mathbb{E}_{\mathbf{z}_n \sim q(\mathbf{z})} \left[\log \frac{p(\mathbf{x}_n, \mathbf{z}_n; \mathbf{W}, \sigma)}{q(\mathbf{z})} \right]$

v. E-step: In lecture, we showed that $\underset{q}{\text{argmax}} \text{ELBO} = \underset{q}{\text{argmin}} D_{KL}(q \| p(\mathbf{z} \mid \mathbf{x}; \mathbf{W}, \sigma))$. Therefore, assuming some initial values of $\mathbf{W}^t, \sigma^t$, we set $q^*\left(\mathbf{z}_n\right) = p\left(\mathbf{z}_n \mid \mathbf{x}_n; \mathbf{W}^t, \sigma^t\right)$. For some calculation practice (you would not have to do this on an exam): To find the posterior distribution, which we also know is Gaussian, we can use the following formula (found in the STAT 110 textbook, but would be provided on exam if needed): a joint Gaussian distribution $p\left(\mathbf{x}_1, \mathbf{x}_2\right) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ has conditional distribution

$$p\left(\mathbf{x}_1 \mid \mathbf{x}_2\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\left(\mathbf{x}_2 - \boldsymbol{\mu}_2\right), \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\right)$$

So $p(\mathbf{z} \mid \mathbf{x}; \mathbf{W}, \boldsymbol{\sigma}) \sim \mathcal{N}\left(\boldsymbol{\Sigma}_{zx}\boldsymbol{\Sigma}_{xx}^{-1}(\mathbf{x}), \boldsymbol{\Sigma}_{zz}\boldsymbol{\Sigma}_{xx}^{-1}\boldsymbol{\Sigma}_{xz}\right)$, where $\boldsymbol{\Sigma}_{zz} = \mathbf{I}$, and $\boldsymbol{\Sigma}_{zx} = \mathbf{W}, \boldsymbol{\Sigma}_{xz} = \mathbf{W}^{\top}$.

vi. M-step: $\underset{\mathbf{W}, \sigma}{\text{argmax}}\, \text{ELBO}\left(\mathbf{x}_n, \mathbf{z}_n, q^*, \mathbf{W}, \sigma\right) = \underset{\mathbf{W}, \sigma}{\text{argmax}} \mathbb{E}_{z_n \sim q(\mathbf{z})} \left[\log p\left(\mathbf{x}_n, \mathbf{z}_n; \mathbf{W}, \sigma\right)\right]$.

(b) See HMM section notes