ECE59500RL Exam

Robert (Cars) Chandler — chandl71@purdue.edu

Problem 1

1.1

The set of trajectories described is the set of all trajectories such that we start at $X_0 = 1$ and all following states are either 5, 6, or 7. Because the transitions between 5, 6, and 7 are periodic, there is no way to return back to 1 once 5 is reached. As a result, we only have to analyze the first step in the trajectory. If the first step takes us to 5, then the trajectory must belong to the set described. The probability of transitioning to state 5 from the initial state, 1, is labeled as 0.5. If we let $\mathcal{T}$ be the set of trajectories described in the problem statement, then:

$$\mathbb{P}(X_1 = 5|X_0 = 1) = \mathbb{P}(\tau \in \mathcal{T}|X_0 = 1) = 0.5$$

Where τ is any trajectory resulting from $X_0 = 1$.

Final answer: 0.5

1.2

We can solve this by finding the left eigenvectors of the $\mathbf{P}$ matrix:

```
import numpy as np

p = np.array(
    [
        [0, 0.1, 0, 0, 0.5, 0, 0, 0.4, 0, 0],
        [0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0.6, 0.4, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0.75, 0.25, 0, 0, 0, 0],
```

```
        [0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
        [0, 0, 0, 0, 0.2, 0.8, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
        [0, 0, 0, 0, 0, 0, 0, 0.9, 0, 0.1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
    ]
)

# Using p.T gives us left eigenvectors instead of right
eigvals, eigvecs = np.linalg.eig(p.T)

# Get the indices where the eigenvalues are 1 (accounting for floating point
# math error)
i_ones = np.argwhere(np.abs(eigvals - 1) < 1e-10).squeeze()

# Get the eigenvectors with eigenvalues of 1 (each column of vecs1 is an eigenvector)
vecs1 = eigvecs[:, i_ones]

# The vectors are normalized by default to have a magnitude of 1, but we want
# them to sum to 1
vecs1 /= np.sum(vecs1, axis=0)

# Make each row an eigenvector
# vecs1 = vecs1.T

print("Stationary distributions:")

# Ensure theese eigenvectors display the desired properties
with np.printoptions(precision=4):
    for vec in vecs1.T:
        # Should sum to 1
        assert np.allclose(np.sum(vec), 1)

        # Should yield the same state after applying P
        assert np.allclose(vec @ p, vec)

        print(vec, "\n")
```

```
Stationary distributions:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[0.    0.    0.375 0.625 0.    0.    0.    0.    0.    0.   ]
```

```
[0.     0.     0.     0.     0.2857 0.3571 0.3571 0.     0.     0.    ]
```

The vectors above represent the stationary distributions of the Markov chain. We have proven that they are indeed stationary distributions by proving that they all sum to 1 and that $\bar{\boldsymbol{\mu}}\mathbf{P} = \bar{\boldsymbol{\mu}}$ for each vector.

1.3

If $\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2$ is a stationary distribution, then it will satisfy the following properties:

$$\alpha\bar{\mu}^1(s) + (1-\alpha)\bar{\mu}^2(s) \geq 0, \quad \forall s \in \mathcal{S}$$
$$\sum_{s\in\mathcal{S}} \alpha\bar{\mu}^1(s) + (1-\alpha)\bar{\mu}^2(s) = 1$$
$$\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2 = \left(\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2\right)\mathbf{P}$$

Note that if the quantity above is a convex combination, then both α and $(1-\alpha)$ must be nonnegative, and therefore:

$$0 \leq \alpha \leq 1$$

First we show that for every element $\bar{\mu}_i$ of any stationary distribution $\bar{\boldsymbol{\mu}}$:

$$0 \leq \bar{\mu}_i \leq 1$$

The left side of the inequality holds by the definition of a stationary distribution, and the right side holds because $\sum_{s\in\mathcal{S}} \bar{\mu}_i = 1$ and since each $\bar{\mu}_i$ is positive, the maximum value for any one $\bar{\mu}_i$ is 1.

With this in mind, it becomes apparent that the first of the three properties above holds, because if each element of μ^1, μ^2 is between 0 and 1 and α and $(1-\alpha)$ are also between 0 and 1, then we just have a sum of two products of positive values, the result of which will always be positive.

Next:

$$
\begin{aligned}
&\sum_{s\in\mathcal{S}} \alpha\bar{\mu}^1(s) + (1-\alpha)\bar{\mu}^2(s) \\
&= \alpha \sum_{s\in\mathcal{S}} \bar{\mu}^1(s) + (1-\alpha)\sum_{s\in\mathcal{S}} \bar{\mu}^2(s) \qquad \text{by linearity of summation} \\
&= \alpha \cdot 1 + (1-\alpha)\cdot 1 \quad \text{since } \sum_{s\in\mathcal{S}} \bar{\mu} = 1 \text{ by definition} \\
&= 1
\end{aligned}
$$

And finally:

$$
\begin{aligned}
\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2 &= \left(\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2\right)\mathbf{P} \\
\alpha\bar{\boldsymbol{\mu}}^1 + (1-\alpha)\bar{\boldsymbol{\mu}}^2 &= \alpha\bar{\boldsymbol{\mu}}^1\mathbf{P} + (1-\alpha)\bar{\boldsymbol{\mu}}^2\mathbf{P} \\
\alpha\bar{\boldsymbol{\mu}}^1 - \alpha\bar{\boldsymbol{\mu}}^1\mathbf{P} &= (1-\alpha)\bar{\boldsymbol{\mu}}^2\mathbf{P} - (1-\alpha)\bar{\boldsymbol{\mu}}^2 \\
\alpha\left(\bar{\boldsymbol{\mu}}^1 - \bar{\boldsymbol{\mu}}^1\mathbf{P}\right) &= (1-\alpha)\left(\bar{\boldsymbol{\mu}}^2\mathbf{P} - \bar{\boldsymbol{\mu}}^2\right) \\
\alpha\left(\bar{\boldsymbol{\mu}}^1 - \bar{\boldsymbol{\mu}}^1\right) &= (1-\alpha)\left(\bar{\boldsymbol{\mu}}^2 - \bar{\boldsymbol{\mu}}^2\right) \qquad \text{since } \bar{\boldsymbol{\mu}}\mathbf{P} = \bar{\boldsymbol{\mu}} \text{ by definition} \\
\alpha\cdot 0 &= (1-\alpha)\cdot 0 \\
0 &= 0
\end{aligned}
$$

■

Having proven that the convex combination satisfies all the properties of a stationary distribution, we have proven that it is itself a stationary distribution

1.4

Initializing our chain at $\alpha\mu_0^1 + (1-\alpha)\mu_0^2$ will cause μ_t to converge to

$$\alpha\bar{\mu}^1 + (1-\alpha)\bar{\mu}^2$$

as $t \to \infty$. Intuitively, the pieces of the initial state distribution that were carried to their respective stationary distribution will still be carried to the same distribution, but in proportion based on the fractions α and $(1-\alpha)$.

Problem 2

We will begin with the right side of the property and work our way back to the left side to show they are equivalent:

$$
\begin{aligned}
&= \frac{1}{1-\gamma}\mathbb{E}_{S\sim\bar{\rho}^{\pi}_{\mu_0}}\left[\mathbb{E}_{A\sim\pi(\cdot|S)}\left[f(S,A)\right]\right] \\
&= \frac{1}{1-\gamma}\sum_{S\in\mathcal{S}}\left[\bar{\rho}^{\pi}_{\mu_0}(S)\cdot\mathbb{E}_{A\sim\pi(\cdot|S)}\left[f(S,A)\right]\right] \\
&= \frac{1}{1-\gamma}\sum_{S\in\mathcal{S}}\left[(1-\gamma)\sum_{t=0}^{\infty}\gamma^t\mathbb{P}\left(S_t=S|\mu_0,P,\pi\right)\cdot\mathbb{E}_{A\sim\pi(\cdot|S)}\left[f(S,A)\right]\right] \\
&= \left[\mathbb{E}_{A\sim\pi(\cdot|S)}\left[\sum_{t=0}^{\infty}\gamma^t\sum_{S\in\mathcal{S}}\mathbb{P}\left(S_t=S|\mu_0,P,\pi\right)f(S_t,A_t)\right]\right] \\
&= \mathbb{E}_{A\sim\pi(\cdot|S)}\left[\sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\mu_0,P(\cdot|S_t,A_t)}\left[f(S_t,A_t)\right]\right] \\
&= \mathbb{E}_{S_0\sim\mu_0,S_{t+1}\sim P(\cdot|S_t,A_t),A\sim\pi(\cdot|S)}\left[\sum_{t=0}^{\infty}\gamma^t f(S_t,A_t)\right] \quad \blacksquare
\end{aligned}
$$

Problem 3

3.1

The objective of the primal linear program is:

$$\min_{V\in\mathbb{R}^{|\mathcal{S}|}}\sum_{s\in\mathcal{S}}\mu_0(s)V(s) = \min_{V\in\mathbb{R}^{|\mathcal{S}|}}\left[0.4V(b)+0.6V(c)\right]$$

and this objective is subject to the following constraint:

$$V(s)\geq R(s,a)+\gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}\left[V(s')\right], \quad \forall s\in\mathcal{S},\forall a\in\mathcal{A}$$

We can expand this constraint using our known values. We will first define it for $s=b, a=x$:

$$
\begin{aligned}
V(b) &\geq R(b,x)+\gamma\mathbb{E}_{s'\sim P(\cdot|b,x)}\left[V(s')\right] \\
V(b) &\geq R(b,x)+\gamma\sum_{s'\in\mathcal{S}}P(s'|b,x)V(s') \\
V(b) &\geq R(b,x)+\gamma\left[P(b|b,x)V(b)+P(c|b,x)V(c)\right] \\
V(b) &\geq R(b,x)+\gamma\left[P(b|b,x)V(b)+P(c|b,x)V(c)\right]
\end{aligned}
$$

We can now carry this final line through for each of the four state-action combinations and substitute in our known values to yield four simultaneous constraints that must be met:

$$
\begin{aligned}
V(b) &\geq R(b,x) + \gamma\left[P(b|b,x)V(b) + P(c|b,x)V(c)\right] \\
V(b) &\geq 0.5 + 0.9\left[1 \cdot V(b) + 0 \cdot V(c)\right] \\
V(b) &\geq 0.5 + 0.9V(b) \\
0.1V(b) &\geq 0.5 \\
V(b) &\geq 5
\end{aligned}
$$

$$
\begin{aligned}
V(b) &\geq R(b,y) + \gamma\left[P(b|b,y)V(b) + P(c|b,y)V(c)\right] \\
V(b) &\geq 0.2 + 0.9\left[0.2V(b) + 0.8V(c)\right] \\
V(b) &\geq 0.2 + 0.18V(b) + 0.72V(c) \\
0.82V(b) &\geq 0.2 + 0.72V(c) \\
V(b) &\geq 0.2439 + 0.878V(c)
\end{aligned}
$$

$$
\begin{aligned}
V(c) &\geq R(c,x) + \gamma\left[P(b|c,x)V(b) + P(c|c,x)V(c)\right] \\
V(c) &\geq 0.5 + 0.9\left[0.1V(b) + 0.9V(c)\right] \\
V(c) &\geq 0.5 + 0.09V(b) + 0.81V(c) \\
0.19V(c) &\geq 0.5 + 0.09V(b) \\
V(c) &\geq 2.6316 + 0.47368V(b)
\end{aligned}
$$

$$
\begin{aligned}
V(c) &\geq R(c,y) + \gamma\left[P(b|c,y)V(b) + P(c|c,y)V(c)\right] \\
V(c) &\geq 1 + 0.9\left[0.6V(b) + 0.4V(c)\right] \\
V(c) &\geq 1 + 0.54V(b) + 0.36V(c) \\
0.64V(c) &\geq 1 + 0.54V(b) \\
V(c) &\geq 1.5625 + 0.84375V(b)
\end{aligned}
$$

So the last lines of the four sections above are the final four constraints that must be upheld.

3.2

The objective of the dual linear program is:

$$\max_{\nu \geq 0} \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \nu(s,a)R(s,a) = \max_{\nu \geq 0} \left[0.5\nu(b,x) + 0.2\nu(b,y) + 0.5\nu(c,x) + \nu(c,y)\right]$$

and this objective is subject to the following constraint:

$$\sum_{a\in\mathcal{A}} \nu(s,a) = \mu_0(s) + \gamma \sum_{s'\in\mathcal{S},a'\in\mathcal{A}} P(s|s',a')\nu(s',a'), \quad \forall s \in \mathcal{S}$$

Beginning with $s = b$:

$$\begin{aligned}
\sum_{a\in\mathcal{A}} \nu(b,a) &= \mu_0(b) + \gamma \sum_{s'\in\mathcal{S},a'\in\mathcal{A}} P(b|s',a')\nu(s',a') \\
\nu(b,x) + \nu(b,y) &= 0.4 + 0.9\left[\nu(b,x) + 0.2\nu(b,y) + 0.1\nu(c,x) + 0.6\nu(c,y)\right] \\
\nu(b,x) + \nu(b,y) &= 0.4 + 0.9\nu(b,x) + 0.18\nu(b,y) + 0.09\nu(c,x) + 0.54\nu(c,y) \\
0.1\nu(b,x) + 0.82\nu(b,y) - 0.09\nu(c,x) - 0.54\nu(c,y) &= 0.4
\end{aligned}$$

and for $s = c$:

$$\begin{aligned}
\sum_{a\in\mathcal{A}} \nu(c,a) &= \mu_0(c) + \gamma \sum_{s'\in\mathcal{S},a'\in\mathcal{A}} P(c|s',a')\nu(s',a') \\
\nu(c,x) + \nu(c,y) &= 0.6 + 0.9\left[0\cdot\nu(b,x) + 0.8\nu(b,y) + 0.9\nu(c,x) + 0.4\nu(c,y)\right] \\
\nu(c,x) + \nu(c,y) &= 0.6 + 0.72\nu(b,y) + 0.81\nu(c,x) + 0.36\nu(c,y) \\
0.19\nu(c,x) + 0.64\nu(c,y) - 0.72\nu(b,y) &= 0.6
\end{aligned}$$

So in conclusion, the dual LP is subject to the two following constraints:

$$\begin{aligned}
0.1\nu(b,x) + 0.82\nu(b,y) - 0.09\nu(c,x) - 0.54\nu(c,y) &= 0.4 \\
0.19\nu(c,x) + 0.64\nu(c,y) - 0.72\nu(b,y) &= 0.6
\end{aligned}$$

Problem 4

4.1

The return for each trajectory is calculated as:

$$\sum_{t=0}^{T} \gamma^t R(S_t, A_t)$$

We can calculate these in Python:

```
import numpy as np
from IPython.display import Markdown

trajectories = [
    dict(
        states=np.array(list("cbdeec")),
        actions=np.array(list("zxyxx")),
        rewards=np.array([1, 0, 1, 2, 2]),
    ),
    dict(
        states=np.array(list("cdecee")),
        actions=np.array(list("yxzzx")),
        rewards=np.array([0, 1, -1, 1, 2]),
    ),
    dict(
        states=np.array(list("cbcebe")),
        actions=np.array(list("yzyxy")),
        rewards=np.array([0, 0.5, 0, 2, 0]),
    ),
    dict(
        states=np.array(list("ccebed")),
        actions=np.array(list("zxxyy")),
        rewards=np.array([1, 0.5, 2, 0, -1]),
    ),
]

gamma = 0.9

gammas = gamma ** np.arange(5)

returns = [np.sum(traj["rewards"] * gammas) for traj in trajectories]

returns_latex = Markdown(
    r"\\".join([f"G_{i+1} &= {ret:0.4f}" for i, ret in enumerate(returns)])
)
```

The returns for each trajectory are:

$$G_1 = 4.5802$$
$$G_2 = 2.1312$$
$$G_3 = 1.9080$$
$$G_4 = 2.4139$$

4.2

We can determine the probabilities of taking each action in τ_2 at the corresponding states using the policy tables defined in the problem statement. We can recreate these tables in Python and then determine the probability of the trajectory by multiplying out the probability of taking each individual step and compare the result for each policy. We will go ahead and do this for τ_4 as well to obtain the answer for problem 4.3.

```
from copy import deepcopy

import numpy as np
import xarray as xr

# Define a multidimensional array containing probabilities of state-action
# pairs. This is just like a regular numpy array but with labeled "coordinates"
# and axes which make it easy for us to index it for a given state and/or
# action.
policy_t = xr.DataArray(
    data=np.array(
        [
            [0.1, 0.1, 0.8],
            [0.05, 0.2, 0.75],
            [0.25, 0.5, 0.25],
            [0.9, 0.02, 0.08],
        ]
    ),
    coords=dict(state=list("bcde"), action=list("xyz")),
)

policy_b = xr.DataArray(
    data=np.array(
        [
            [0.5, 0.1, 0.4],
            [0.05, 0.2, 0.75],
            [0.5, 0.2, 0.3],
            [0.9, 0.02, 0.08],
```

```
        ]
    ),
    coords=dict(state=list("bcde"), action=list("xyz")),
)


# Double-check that our probabilities for a given state all sum to one
assert np.allclose(policy_t.sum(dim="action"), 1)
assert np.allclose(policy_b.sum(dim="action"), 1)

from p4_1 import trajectories

p_target = {2: 0.0, 4: 0.0}
p_behavior = {2: 0.0, 4: 0.0}


for prob, policy in zip([p_target, p_behavior], [policy_t, policy_b]):
    # Walk each step of the trajectory, determine the probability of taking an
    # action at a given state according to the policy of interest, and take the
    # product of all the probabilities to get the total probability of the
    # trajectory.
    for traj, i_traj in zip([trajectories[1], trajectories[3]], [2, 4]):
        prob[i_traj] = float(
            np.prod(
                [
                    policy.sel(state=s, action=a)
                    # We included the final state in the definition, but we
                    # don't need it here, so we skip it in our iteration (i.e.
                    # traj["states"][:-1])
                    for s, a in zip(traj["states"][:-1], traj["actions"])
                ]
            )
        )
```

The probability of taking τ_2 according to π^t is 0.0027 while the probability according to π^b is 0.0054, so τ_2 **is more probable under π^b than under π^t**.

4.3

As described above, we already calculated the probabilities for τ_4 in 4.2.

The probability of taking τ_4 according to π^t is $6.75e-05$ while the probability according to π^b is $6.75e-05$, so τ_4 **is equally probable under π^b or π^t.**

4.4

We start with the (on-policy) evaluation of $V^{\pi^b}(c)$ of the behavior policy using an every-visit Monte Carlo method:

```
import numpy as np
from IPython.display import Markdown
from p4_1 import gammas, trajectories

vhat = {state: 0.0 for state in "bcde"}
num_visits = {state: 0.0 for state in "bcde"}
returns = {state: [] for state in "bcde"}


def return_at_t(t, traj, gammas):
    return float(np.sum(traj["rewards"][t:] * gammas[t:]))


def mc_evaluation(vhat, returns, num_visits, return_function):
    for traj in trajectories:
        # For each state-action pair... (excluding state c since we end at that state)
        for state in np.unique(traj["states"][:-1]):
            # For each step in the trajectory...
            for t, (s, a, r) in enumerate(
                zip(traj["states"][:-1], traj["actions"], traj["rewards"])
            ):
                # We only want to evaluate each time where state is visited
                if s != state:
                    continue

                ret = return_function(t, traj, gammas)

                returns[state].append(ret)

                num_visits[state] += 1

            vhat[state] = float(
                (1 / num_visits[state]) * np.sum(returns[state])
            )
```

```
    return vhat, returns, num_visits


mc_evaluation(vhat, returns, num_visits, return_function=return_at_t)

vhat_latex = Markdown(
    r"\\".join(
        [rf"V^{{\pi^b}}({s}) &\approx {v:.5f}" for s, v in vhat.items()]
    )
)
```

The resulting value function for each state is:

$$V^{\pi^b}(b) \approx 1.20803$$
$$V^{\pi^b}(c) \approx 2.27806$$
$$V^{\pi^b}(d) \approx 2.85570$$
$$V^{\pi^b}(e) \approx 1.19880$$

4.5

We repeat the same every-visit Monte Carlo estimation of the value function, but this time we perform an off-policy estimate for the target policy using the behavior policy:

```
import numpy as np
from IPython.display import Markdown
from p4_1 import gammas, trajectories
from p4_2 import policy_b, policy_t
from p4_4 import mc_evaluation, return_at_t

vhat = {state: 0.0 for state in "bcde"}
num_visits = {state: 0.0 for state in "bcde"}
returns = {state: [] for state in "bcde"}


def weighted_return_at_t(t, traj, gammas):
    weight = np.prod(
        [
            (
                policy_t.sel(state=s, action=a)
```

```
                / policy_b.sel(state=s, action=a)
            )
            for s, a in zip(traj["states"][:-1][t:], traj["actions"][t:])
        ]
    )

    return weight * return_at_t(t, traj, gammas)


mc_evaluation(vhat, returns, num_visits, return_function=weighted_return_at_t)

vhat_latex = Markdown(
    r"\\".join(
        [rf"V^{{\pi^b}}({s}) &\approx {v:.5f}" for s, v in vhat.items()]
    )
)
```

The resulting value function for each state is:

$$V^{\pi^b}(b) \approx 1.23750$$
$$V^{\pi^b}(c) \approx 2.07124$$
$$V^{\pi^b}(d) \approx 5.00805$$
$$V^{\pi^b}(e) \approx 1.19880$$

Problem 5

5.1

We follow the same logic as we did in Homework 3, but we use the following definition of the off-policy value function:

$$\hat{V}^{\pi}(s) = \frac{1}{N^s} \sum_{i=1}^{N^s} w_i^s G_i^s$$

Where w_i^s is the weight applied to the return using the sample trajectory starting at s:

$$w_i^s = \prod_{t=0}^{T-1} \frac{\pi^t(a_t|s_t)}{\pi^b(a_t|s_t)}$$

We are given the maximum value of the weight as W, and we can use this to bound $w_i^s G_i^s$:

$$w_i^s G_i^s \leq W \sum_{t=0}^{T} \gamma^t R_{max} = W R_{max} \frac{1-\gamma^{T+1}}{1-\gamma}$$

Note that in Homework 3, we assumed infinite trajectories, but we can actually use T as the limit of our sum since we are told that the length of all trajectories is bounded by T, which will give us a tighter bound than the infinite case. Similarly for the lower bound:

$$w_i^s G_i^s \geq W \sum_{t=0}^{T} -\gamma^t R_{max} = -W R_{max} \frac{1-\gamma^{T+1}}{1-\gamma}$$

Therefore:

$$-W R_{max} \frac{1-\gamma^{T+1}}{1-\gamma} \leq w_i^s G_i^s \leq W R_{max} \frac{1-\gamma^{T+1}}{1-\gamma}$$

If we define $E(s)$ as

$$E(s) = \sum_{i=1}^{N^s} w_i^s G_i^s$$

Then:

$$\begin{aligned}
\mathbb{E}_{A_t \sim \pi, S_{t+1} \sim P}\left[E(s)\right] &= \mathbb{E}_{A_t \sim \pi, S_{t+1} \sim P}\left[\sum_{i=1}^{N^s} w_i^s G_i^s\right] \\
&= \sum_{i=1}^{N^s} \mathbb{E}\left[w_i^s G_i^s\right] \\
&= \sum_{i=1}^{N^s} V^\pi(s) \\
&= N^s V^\pi(s)
\end{aligned}$$

Since the off-policy value function is the expected value of the weighted return.

Therefore, we can bound the probability that $E(s)$ differs from its expectation using Hoeffding's inequality and our bound $w_i^s G_i^s$:

$$\mathbb{P}\left(|E(s) - \mathbb{E}[E(s)]| \geq \varepsilon\right) \leq 2\exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^{N^s}\left(2WR_{max}\frac{1-\gamma^{T+1}}{1-\gamma}\right)^2}\right)$$
$$\mathbb{P}\left(|E(s) - \mathbb{E}[E(s)]| \geq \varepsilon\right) \leq 2\exp\left(-\frac{(1-\gamma)^2\varepsilon^2}{2N^s\left(WR_{max}(1-\gamma^{T+1})\right)^2}\right)$$

And since $\hat{V}^\pi(s) = \frac{1}{N^s}E(s)$:

$$\mathbb{P}\left(\left|\hat{V}^\pi(s) - V^\pi(s)\right| \geq \varepsilon'\right) \leq 2\exp\left(-\frac{N^s(1-\gamma)^2\varepsilon^2}{2\left(WR_{max}(1-\gamma^{T+1})\right)^2}\right),\ \forall\varepsilon' > 0$$

5.2

We start with a realization that the L1 norm is simply a sum of absolute differences:

$$\|\hat{\mathbf{V}}^\pi - \mathbf{V}^\pi\|_1 = \sum_{s\in\mathcal{S}}|\hat{V}^\pi(s) - V^\pi(s)|$$

and we have already bounded the probability of a single absolute difference in 5.1. So, we would like to bound the following probability:

$$\begin{aligned}\mathbb{P}\left(\|\hat{\mathbf{V}}^\pi - \mathbf{V}^\pi\|_1 \geq \varepsilon'\right) &= \mathbb{P}\left(\sum_{s\in\mathcal{S}}|\hat{V}^\pi(s) - V^\pi(s)| \geq \varepsilon'\right)\\ &\leq \mathbb{P}\left(|\mathcal{S}||\hat{V}^\pi(s) - V^\pi(s)| \geq \varepsilon'\right)\\ &= \mathbb{P}\left(|\hat{V}^\pi(s) - V^\pi(s)| \geq \frac{\varepsilon'}{|\mathcal{S}|}\right)\end{aligned}$$

Now we have a scaled version of the bound from 5.1. If we substitute $\varepsilon'/|\mathcal{S}|$ in for ε' in the original bound, then we can calculate the final upper bound:

$$\mathbb{P}\left(\left|\hat{V}^\pi(s) - V^\pi(s)\right| \geq \frac{\varepsilon'}{|\mathcal{S}|}\right) \leq 2\exp\left(-\frac{N^s(1-\gamma)^2\frac{\varepsilon}{|\mathcal{S}|}^2}{2\left(WR_{max}(1-\gamma^{T+1})\right)^2}\right),\ \forall\varepsilon' > 0$$

So finally:

$$\mathbb{P}\left(\|\hat{\mathbf{V}}^\pi - \mathbf{V}^\pi\|_1 \geq \varepsilon'\right) \leq 2\exp\left(-\frac{N^s(1-\gamma)^2\varepsilon^2}{2|\mathcal{S}|^2\left(WR_{max}(1-\gamma^{T+1})\right)^2}\right),\ \forall\varepsilon' > 0$$

Problem 6

6.1

We will estimate the true state transitions using with the frequency of the transitions observed in these samples. We take the number of titmes a particular state is observed when starting at a given state-action pair and divide it by the total number of samples at that pair to get a probability estimate:

$$\begin{aligned} P(b|b,x) &\approx 0.7, \quad P(c|b,x) \approx 0.3 \\ P(b|b,y) &\approx 0.2, \quad P(c|b,y) \approx 0.8 \\ P(b|c,x) &\approx 0.1, \quad P(c|c,x) \approx 0.9 \\ P(b|c,y) &\approx 0.8, \quad P(c|c,y) \approx 0.2 \end{aligned}$$

6.2

The L1-differences are:

$$\begin{aligned} \|\hat{P}(\cdot|b,x) - P(\cdot|b,x)\|_1 &= |\hat{P}(b|b,x) - P(b|b,x)| + |\hat{P}(c|b,x) - P(c|b,x)| \\ &= |0.7 - 0.8| + |0.3 - 0.2| = 0.1 + 0.1 \\ &= 0.2 \end{aligned}$$

$$\begin{aligned} \|\hat{P}(\cdot|b,y) - P(\cdot|b,y)\|_1 &= |\hat{P}(b|b,y) - P(b|b,y)| + |\hat{P}(c|b,y) - P(c|b,y)| \\ &= |0.2 - 0.1| + |0.8 - 0.9| = 0.1 + 0.1 \\ &= 0.2 \end{aligned}$$

$$\begin{aligned} \|\hat{P}(\cdot|c,x) - P(\cdot|c,x)\|_1 &= |\hat{P}(b|c,x) - P(b|c,x)| + |\hat{P}(c|c,x) - P(c|c,x)| \\ &= |0.1 - 0.05| + |0.9 - 0.95| = 0.05 + 0.05 \\ &= 0.1 \end{aligned}$$

$$\begin{aligned} \|\hat{P}(\cdot|c,y) - P(\cdot|c,y)\|_1 &= |\hat{P}(b|c,y) - P(b|c,y)| + |\hat{P}(c|c,y) - P(c|c,y)| \\ &= |0.8 - 0.9| + |0.2 - 0.1| = 0.1 + 0.1 \\ &= 0.2 \end{aligned}$$

6.3

The simulation lemma states that

$$|\hat{V}^{\pi}(s_0) - V^{\pi}(s_0)| \leq \frac{\gamma R_{max}}{(1-\gamma)^2} \mathbb{E}_{s \sim \hat{\rho}^{\pi}_{s_0}} \left[\| \hat{P}(\cdot|s, \pi(s)) - P(\cdot|s, \pi(s)) \|_1 \right]$$

where $R_{max} = \max_{s,a} |R(s,a)|$.

We can evaluate this for state $s_0 = b$ usign the policy and state occupancy measure values given in the problem statement to bound the difference between the value functions. We know $R_{max} = 1$ since it is on the interval $[0, 1]$.

$$\begin{aligned}
|\hat{V}^{\pi}(b) - V^{\pi}(b)| &\leq \frac{0.9 \cdot 1}{(1-0.9)^2} \sum_{s \in \mathcal{S}} \left[\hat{\rho}^{\pi}_b(s) \cdot \| \hat{P}(\cdot|s, \pi(s)) - P(\cdot|s, \pi(s)) \|_1 \right] \\
|\hat{V}^{\pi}(b) - V^{\pi}(b)| &\leq 90 \left[\hat{\rho}^{\pi}_b(b) \cdot \| \hat{P}(\cdot|b, y) - P(\cdot|b, y) \|_1 + \hat{\rho}^{\pi}_b(c) \cdot \| \hat{P}(\cdot|c, x) - P(\cdot|c, x) \|_1 \right] \\
|\hat{V}^{\pi}(b) - V^{\pi}(b)| &\leq 90 \left[1.5 \cdot 0.2 + 8.5 \cdot 0.1 \right] \\
|\hat{V}^{\pi}(b) - V^{\pi}(b)| &\leq 103.5
\end{aligned}$$

6.4

We should allocate the most samples to pair (c, x) if our goal is to lower the upper bound from the simulation lemma, because this is the pair that has the higher coefficient multiplied against its L1 error, $\| \hat{P}(\cdot|c, x) - P(\cdot|c, x) \|_1$. This is because $\rho^{\pi}_b(c) = 8.5 > 1.5 = \rho^{\pi}_b(b)$, so c is the action with the greater weight in the bound, and the policy dictates that $\pi(c) = x$, so it is the action that will be evaluated with c in the L1 error term. By prioritizing this pair, we will decrease our L1 error for the pair since we will approach the true probability as our number of samples increases. So, driving down the L1 error with the greater weight is the best way to drive down the upper bound above.