

## A. Appendix

### A.1. Bootstrap Convergence

In this section we provide a high-level discussion of the bootstrap procedure and its asymptotic validity. We refer the readers to the works by (Cao, 1993; Hall, 1990) for a more fine-grained analysis and convergence rates when estimating MSE using statistical bootstrap. Individual treatment of bias (Efron, 1990; Efron and Tibshirani, 1994; Hong, 1999; Shi, 2012; Mikusheva, 2013) and variance (Chen, 2017b; Gamero et al., 1998; Shao, 1990; Ghosh et al., 1984; Li and Maddala, 1999) can also be found.

In the following, we will discuss the consistency of  $\hat{A}$  estimated using bootstrap,

$$\hat{A}_{i,j} - A_{i,j} \xrightarrow{a.s.} 0.$$

Towards this goal, we will consider the following conditions imposed on the set of the base estimators  $\{\hat{\theta}_i\}_{i=1}^k$ ,

- $\forall i, \hat{\theta}_i$  is uniformly bounded.
- $\forall i, \hat{\theta}_i \xrightarrow{a.s.} c_i$ .
- $\forall i, \hat{\theta}_i$  is smooth with respect to data distribution.
- $\exists \hat{\theta}_k : \hat{\theta}_k \xrightarrow{a.s.} c_k = \theta^*$ .

Recall from (2),

$$\begin{aligned} A_{i,j} &= \mathbb{E} \left[ \left( \hat{\theta}_i - \theta^* \right) \left( \hat{\theta}_j - \theta^* \right) \right] \\ &= \mathbb{E} \left[ \left( \hat{\theta}_i - \mathbb{E}[\hat{\theta}_i] + \mathbb{E}[\hat{\theta}_i] - \theta^* \right) \right. \\ &\quad \left. \left( \hat{\theta}_j - \mathbb{E}[\hat{\theta}_j] + \mathbb{E}[\hat{\theta}_j] - \theta^* \right) \right] \\ &= \mathbb{E} \left[ \left( \hat{\theta}_i - \mathbb{E}[\hat{\theta}_i] \right) \left( \hat{\theta}_j - \mathbb{E}[\hat{\theta}_j] \right) \right] \\ &\quad + \mathbb{E} \left[ \left( \mathbb{E}[\hat{\theta}_i] - \theta^* \right) \left( \mathbb{E}[\hat{\theta}_j] - \theta^* \right) \right]. \end{aligned}$$

Let  $X_n := \left( \hat{\theta}_i - \mathbb{E}[\hat{\theta}_i] \right)$  and  $Y_n := \left( \hat{\theta}_j - \mathbb{E}[\hat{\theta}_j] \right)$ . As  $\hat{\theta}_i \xrightarrow{a.s.} c_i$  and  $\hat{\theta}_i$  is uniformly bounded, using (Thomas and Brunskill, 2016, Lemma 2), we have  $\mathbb{E}[\hat{\theta}_i] \xrightarrow{a.s.} c_i$ . Similarly, we have  $\mathbb{E}[\hat{\theta}_j] \xrightarrow{a.s.} c_j$  as  $\hat{\theta}_j \xrightarrow{a.s.} c_j$ . Then using continuous mapping theorem,

$$X_n Y_n \xrightarrow{a.s.} (c_i - c_i)(c_j - c_j) = 0.$$

Now using (Thomas and Brunskill, 2016, Lemma 2),

$$\mathbb{E} \left[ \left( \hat{\theta}_i - \mathbb{E}[\hat{\theta}_i] \right) \left( \hat{\theta}_j - \mathbb{E}[\hat{\theta}_j] \right) \right] = \mathbb{E}[X_n Y_n] \xrightarrow{a.s.} 0. \quad (9)$$

Similarly,

$$\left( \mathbb{E}[\hat{\theta}_i] - \theta^* \right) \left( \mathbb{E}[\hat{\theta}_j] - \theta^* \right) \xrightarrow{a.s.} (c_i - \theta^*)(c_j - \theta^*) \quad (10)$$

Therefore, using (9) and (10),

$$\hat{A}_{i,j} \xrightarrow{a.s.} 0 + (c_i - \theta^*)(c_j - \theta^*). \quad (11)$$

Now we consider the asymptotic property of the bootstrap estimate  $\hat{A}$  of  $A$ .

$$\hat{A}_{i,j} = \mathbb{E}_{D_{n_1}^* | D_n} \left[ \left( \hat{\theta}_i(D_{n_1}^*) - \hat{\theta}_k \right) \left( \hat{\theta}_j(D_{n_1}^*) - \hat{\theta}_k \right) \right] \quad (12)$$

where  $\hat{\theta}_k$  is known to be a consistent estimator, i.e.,  $\hat{\theta}_k \xrightarrow{a.s.} \theta^*$ . Here,  $\hat{\theta}_k$  could be the WIS or IS or doubly-robust estimators that are known to provide consistent estimates of  $\theta^* = J(\pi)$ . For brevity, we drop the conditional notation on the subscript, and write (12) as,

$$\hat{A}_{i,j} = \mathbb{E}_{D_{n_1}^*} \left[ \left( \hat{\theta}_i(D_{n_1}^*) - \hat{\theta}_k \right) \left( \hat{\theta}_j(D_{n_1}^*) - \hat{\theta}_k \right) \right] \quad (13)$$

Simplifying (13),

$$\begin{aligned} \hat{A}_{i,j} &= \mathbb{E}_{D_{n_1}^*} \left[ \left( \hat{\theta}_i(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_i(D_{n_1}^*) \right] \right. \right. \\ &\quad \left. \left. + \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_i(D_{n_1}^*) \right] - \hat{\theta}_k \right) \right. \\ &\quad \left. \left( \hat{\theta}_j(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_j(D_{n_1}^*) \right] \right. \right. \\ &\quad \left. \left. + \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_j(D_{n_1}^*) \right] - \hat{\theta}_k \right) \right] \\ &= \mathbb{E}_{D_{n_1}^*} \left[ \left( \hat{\theta}_i(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_i(D_{n_1}^*) \right] \right) \right. \\ &\quad \left. \left( \hat{\theta}_j(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_j(D_{n_1}^*) \right] \right) \right] \\ &\quad + \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_i(D_{n_1}^*) - \hat{\theta}_k \right] \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_j(D_{n_1}^*) - \hat{\theta}_k \right] \quad (14) \end{aligned}$$

Let  $X_{n_1} := \left( \hat{\theta}_i(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_i(D_{n_1}^*) \right] \right)$  and  $Y_{n_1} := \left( \hat{\theta}_j(D_{n_1}^*) - \mathbb{E}_{D_{n_1}^*} \left[ \hat{\theta}_j(D_{n_1}^*) \right] \right)$ . As the empirical distribution  $D_{n_1}^*$  converges to the population distribution, i.e.,  $D_n \xrightarrow{a.s.} D$ , the resampled distribution  $D_{n_1}^*$  from  $D_n$  also converges to the population distribution, i.e.,  $D_{n_1}^* \xrightarrow{a.s.} D$ . Therefore, when the estimator  $\hat{\theta}_i(D_{n_1}^*)$  is smooth, using the continuous mapping theorem,

$$\forall i, \lim_{n_1 \rightarrow \infty} \hat{\theta}_i(D_{n_1}^*) = \hat{\theta}_i \left( \lim_{n_1 \rightarrow \infty} D_{n_1}^* \right) = \hat{\theta}_i(D) = c_i.$$

Therefore, similar to before,

$$X_{n_1} Y_{n_1} \xrightarrow{a.s.} (c_i - c_i)(c_j - c_j) = 0,$$

and subsequently,

$$\mathbb{E}_{D_{n_1}^*} [X_{n_1} Y_{n_1}] \xrightarrow{a.s.} 0. \quad (15)$$

Further, as  $\hat{\theta}_k \xrightarrow{a.s.} \theta^*$ ,

$$\hat{\theta}_i(D_{n_1}^*) - \hat{\theta}_k \xrightarrow{a.s.} c_i - \theta^*.$$

Therefore,

$$\begin{aligned} \mathbb{E}_{D_{n_1}^*} [\hat{\theta}_i(D_{n_1}^*) - \hat{\theta}_k] \mathbb{E}_{D_{n_1}^*} [\hat{\theta}_j(D_{n_1}^*) - \hat{\theta}_k] \\ \xrightarrow{a.s.} (c_i - \theta^*)(c_j - \theta^*). \end{aligned} \quad (16)$$

Using (15) and (16) in (14),

$$\hat{A}_{i,j} \xrightarrow{a.s.} 0 + (c_i - \theta^*)(c_j - \theta^*). \quad (17)$$

Finally, combining (11) and (17),

$$\hat{A}_{i,j} - A_{i,j} \xrightarrow{a.s.} 0.$$

which gives the desired result. It is worth highlighting that, theoretically, this result relies upon assumptions that the base estimators satisfy regularity conditions and are consistent. In practice, such assumptions might not hold (for e.g., when using FQE to do policy evaluation if the function approximation is under-parameterized). Nonetheless, in Section 5 we empirically illustrate that even when these assumptions are not directly satisfied, OPERA can be effective.

## A.2. Proofs on Properties of OPERA

### A.2.1. INVARIANCE

In the following, we illustrate an important property of OPERA, that the resulting combined estimate  $\hat{\theta}$  is invariant to the addition of redundant copies of the base estimators  $\{\hat{\theta}_i\}_{i=1}^n$ . Without loss of generality, let  $\hat{\Theta}_\beta \in \mathbb{R}^{(K+1) \times 1}$  be the stack of unique estimators  $\{\hat{\theta}_i\}_{i=1}^k$  with  $\hat{\theta}_{k+1}$  being a redundant copy of the  $\hat{\theta}_k$ ,

**Theorem 4** (Invariance). *If  $\hat{A}$  is positive definite, then  $\hat{\theta}_\beta = \hat{\theta}$ , where,*

$$\hat{\theta}_\beta := \sum_{i=1}^{k+1} \beta_i^* \hat{\theta}_i \in \mathbb{R}, \quad \text{where, } \beta^* \in \arg \min_{\beta \in \mathbb{R}^{(k+1) \times 1}} \beta^\top B \beta.$$

*Proof.* We prove this by contradiction. Recall that  $\hat{\alpha} \in \mathbb{R}^k$  are the weights that minimize the bootstrap estimate of MSE of  $\hat{\theta}$  consisting of  $k$  estimators.

$$\widehat{\text{MSE}}(\hat{\alpha}_1 \hat{\theta}_1 + \dots + \hat{\alpha}_k \hat{\theta}_k) = \hat{\alpha}^\top \hat{A} \hat{\alpha}. \quad (18)$$

As  $\hat{\theta}_{k+1}$  is a redundant copy of  $\hat{\theta}_k$ ,

$$\begin{aligned} \widehat{\text{MSE}}(\beta_1^* \hat{\theta}_1 + \dots + \beta_k^* \hat{\theta}_k + \beta_{k+1}^* \hat{\theta}_{k+1}) \\ = \widehat{\text{MSE}}(\beta_1^* \hat{\theta}_1 + \dots + (\beta_k^* + \beta_{k+1}^*) \hat{\theta}_k) \end{aligned} \quad (19)$$

Finally, as  $\beta^* \in \mathbb{R}^{k+1}$  is the weight that minimizes the bootstrap estimate of MSE of  $\hat{\theta}_\beta$ . Now, if (18) < (19), then one could assign  $\beta_i^* := \hat{\alpha}_i$  for  $i \in \{1, \dots, k\}$ , and  $\beta_{k+1}^* = 0$  to make (19) = (18). Further, notice that as both  $\hat{\alpha}$  and  $\beta^*$  are within the same feasible set of solutions, the above reassignment is also within the feasible set of solutions. Similarly, if (18) > (19), then one could assign  $\hat{\alpha}_i := \beta_i^*$  for  $i \in \{1, \dots, k-1\}$ , and  $\hat{\alpha}_k = \beta_k^* + \beta_{k+1}^*$  to make (19) = (18). Hence, if (18) does not equal (19), then either  $\hat{\alpha}$  or  $\beta^*$  is not optimal and that would be a contradiction. This ensures that  $\widehat{\text{MSE}}(\hat{\theta}_\beta) = \widehat{\text{MSE}}(\hat{\theta})$ .

As  $\hat{A}$  is positive definite, it implies that (8) is strictly convex with linear constraints. Thus the minimizer  $\hat{\alpha}$  of (8) is unique, and  $\hat{\theta}_\beta = \hat{\theta}$ . Note that due to redundancy,  $B$  will not be PD despite  $\hat{A}$  being PD. This would imply that there can be multiple values of  $\beta_k^*$  and  $\beta_{k+1}^*$ . Nonetheless, since  $\beta_k^* + \beta_{k+1}^* = \hat{\alpha}_k$ , it implies that  $\hat{\theta}_\beta = \hat{\theta}$ .  $\square$

### A.2.2. PERFORMANCE IMPROVEMENT

**Theorem 5** (Performance improvement). *If  $\hat{\alpha} = \alpha^*$ ,*

$$\forall i \in \{1, \dots, k\}, \quad \text{MSE}(\hat{\theta}) \leq \text{MSE}(\hat{\theta}_i).$$

*Proof.* With a slight overload of notation, we make the dependency of weights  $\alpha$  explicit and let  $\bar{\theta}(\alpha) = \sum_{i=1}^k \alpha_i \hat{\theta}_i$ . Let  $\text{MSE}(\bar{\theta}(\alpha)) := \alpha^\top A \alpha$ , where  $A$  is defined as in (2).

Now from (1) and (2), we know that for  $\sum_{i=1}^k \alpha_i = 1$ ,

$$\alpha^* \in \arg \min_{\alpha \in \mathbb{R}^{k \times 1}} \text{MSE}(\bar{\theta}(\alpha)).$$

Therefore, for any  $\lambda \in \mathbb{R}^{k \times 1}$  such that  $\sum_{i=1}^k \lambda_i = 1$ ,

$$\begin{aligned} \text{MSE}(\bar{\theta}(\hat{\alpha})) &= \text{MSE}(\bar{\theta}(\alpha^*)) & \because \hat{\alpha} = \alpha^* \\ &\leq \text{MSE}(\bar{\theta}(\lambda)). \end{aligned}$$

Notice that for  $e_i := [0, 0, \dots, 1, \dots, 0]$ , where there is a 1 in the  $i^{\text{th}}$  position and zero otherwise,  $\bar{\theta}(e_i) = \hat{\theta}_i$ . Therefore,

$$\begin{aligned} \text{MSE}(\bar{\theta}(\hat{\alpha})) &\leq \text{MSE}(\bar{\theta}(e_i)) & \forall i \\ &= \text{MSE}(\hat{\theta}_i) & \forall i. \end{aligned}$$

Therefore, as  $\hat{\theta} = \bar{\theta}(\hat{\alpha})$ , we have the desired result that  $\forall i \in \{1, \dots, k\}, \quad \text{MSE}(\hat{\theta}) \leq \text{MSE}(\hat{\theta}_i)$ .  $\square$

### A.3. OPERA Algorithm

We show an illustration of the OPERA algorithm in Figure 1 and we describe the pseudo-code below.

---

**Algorithm 1:** OPERA Score Computation with Bootstrap
 

---

**Input:** offline RL data  $\mathcal{D}$ ; evaluation policy  $\pi$ ; a set of OPE estimators  $[\text{OPE}_1, \text{OPE}_2, \dots, \text{OPE}_k]$ ; number of bootstrap  $B$ ; a subsample coefficient  $\eta \in [0, 1]$ .

**Output:** estimated  $\pi$  performance  $s_{\text{OPERA}}$

```

for  $i \leftarrow 1 \dots K$  do
     $s_i^* = \text{OPE}_i(\mathcal{D})$ 
     $\tilde{s}_i = \emptyset$ 
    for  $j \leftarrow 1 \dots B$  do
         $\tilde{n} = |\mathcal{D}|^\eta$ 
         $\tilde{\mathcal{D}}_j \leftarrow \text{Bootstrap}(\mathcal{D}, \tilde{n})$ 
         $\tilde{s}_i = \tilde{s}_i \cup \text{OPE}_i(\tilde{\mathcal{D}}_j)$ 
    end
end
 $\tilde{M} \leftarrow [\tilde{s}_1, \tilde{s}_2, \dots, \tilde{s}_k] \in \mathbb{R}^{K \times B}$ 
 $M \leftarrow [s_1^*, s_2^*, \dots, s_k^*] \in \mathbb{R}^{K \times 1}$ 
 $\delta \leftarrow [(\tilde{s}_1 - s_1^*, \tilde{s}_2 - s_2^*, \dots, \tilde{s}_k - s_k^*)] \in \mathbb{R}^{K \times B}$ 
 $A \leftarrow \frac{1}{B} \tilde{n} \delta \delta^\top \in \mathbb{R}^{K \times K}$ 
 $\alpha = \arg \min_{\alpha} \alpha A \alpha^\top \quad s.t. \sum \alpha = 1$ 
 $s_{\text{OPERA}} = \alpha^\top M$ 
return  $s_{\text{OPERA}}$ 
    
```

---

#### A.4. Comparison to MAGIC

An important component of our algorithm is accurately estimating the MSE of each OPE estimator. In Sec 5.1, we showed that our  $\widehat{\text{MSE}}(\theta)$  is close to the  $\text{MSE}(\theta)$ . Here, we discuss an alternative method of estimating an OPE estimator’s MSE in a related work (Thomas and Brunskill, 2016). In this work, the bias and variance of an OPE estimator are computed through per-trajectory OPE scores. This leads to several issues: most notably, this method cannot estimate the MSE of self-normalizing estimators (such as WIS) or minimax-style estimators (such as any estimator in the DICE family (Yang et al., 2020)). We denote this estimator as  $\widehat{\text{MSE}}_{\text{MAGIC}}(\theta)$ .

In our experiment, we evaluate FQE and IS on Sepsis-POMDP and Sepsis-MDP domains. We choose percentile bootstrap to construct the CI around WIS and use it to compute the bias of the other two estimators. We use a 50% confidence interval to get an upper and lower bound on the WIS estimates, and compute the bias of FQE and IS by subtracting the average over trajectory with the closest upper or lower bound of WIS. We show the comparison results in Table A1. Our procedure is able to provide a consistently better estimate for FQE’s MSE. We suspect that this is due to MAGIC’s unique way of computing bias. Specifically, MAGIC computes bias by comparing two estimates (in this case, FQE and WIS); if these two estimators do not agree

with each other, then the bias will be large.

#### A.5. Different MSE Estimation Strategies

We explore two alternative strategies to estimate the MSE of each estimator. The first strategy is, instead of using the estimator’s own score as the centering variable  $\hat{\Theta}$ , we use a consistent and unbiased estimator’s score as  $\hat{\Theta}$ . We call this OPERA-IS. Another strategy is to use idea from MAGIC’s guided importance sampling, where the bias estimate of each estimator is an upper bound over the true bias. We call this OPERA-MAGIC. The results on Sepsis domain is presented in Table A2. While using an unbiased consistent estimator as the centering variable can help further improve OPERA’s estimate, sometimes it also hurts the performance (MDP N=1000 setting). OPERA-MAGIC however almost always performs worse than the best estimator in the ensemble. Using an upper bound on bias is a good idea if we are performing a conservative (safe) selection between different OPEs – it is however a bad idea when we want to combine OPE scores together, as an upper bound is inherently a distorted estimate of the estimator bias.

#### A.6. D4RL Experiment

**Setup** D4RL (Fu et al., 2020) is an offline RL standardized benchmark designed and commonly used to evaluate the progress of offline RL algorithms. We use 6 datasets of different quality from three environments: Hopper, HalfCheetah, and Walker2d. We choose the medium and medium-replay datasets. Medium dataset has 200k samples from a policy trained to approximately 1/3 the performance of a policy trained to completion with SAC. Medium-replay dataset takes the transitions stored in the experience replay buffer of policy – this dataset can be thought of as a dataset sampled by a mixture of policies.

**Policy Training** We train 6 policies from these three algorithms with 2 different hyperparameters for the neural network, Q-learning (CQL) (Kumar et al., 2020), implicit Q-learning (Kostrikov et al., 2021), and TD3+BC (Fujimoto et al., 2018). We initialize all neural networks (including both actor and critics, if the algorithm uses both) with the hidden dimensions of [256, 256, 256]. We train with a batch size of 512, with Adam Optimizer. We train for 100 epochs on each dataset. We only change one important hyperparameter per algorithm. We report the discounted return of each policy in Table A8, A7, A9. We report these scores because they are the prediction target of the FQE algorithm. We report the un-discounted return of each policy in Table A12, A13, A14.

**FQE Training** We train Fitted Q learning for each policy. As discussed in the main text, FQE has a few hyperparameter choices. We choose 4 hyperparameters for Hopper and HalfCheetah. We choose another 4 hyperparameters for

Table A1. We compare two styles of MSE estimations and how well they can estimate the true MSE of each estimator. We report averaged results over 10 trials, with N=200.

	Sepsis-POMDP			Sepsis-MDP		
	MSE( $\theta$ )	$\widehat{\text{MSE}}_{\text{MAGIC}}(\theta)$	$\widehat{\text{MSE}}(\theta)$	MSE( $\theta$ )	$\widehat{\text{MSE}}_{\text{MAGIC}}(\theta)$	$\widehat{\text{MSE}}(\theta)$
IS	0.0161	0.0281	<b>0.0088</b>	0.3445	<b>0.0485</b>	0.0056
FQE	0.0979	0.4953	<b>0.0163</b>	0.0077	0.0771	<b>0.0011</b>

Table A2. We report the Mean-Squared Error (MSE) for the Sepsis domain. We additionally present two variants of OPERA where we experimented with different MSE estimation strategies.

Sepsis	N	OPERA	OPERA-IS	OPERA-MAGIC	IS	WIS	FQE
MDP	200	0.2205	<b>0.2181</b>	0.2657	0.2753	0.2998	0.2448
MDP	1000	<b>0.1705</b>	0.1779	0.1848	<u>0.1720</u>	0.2948	0.2995
POMDP	200	<b>0.2750</b>	0.2768	0.2827	<u>0.2804</u>	0.2850	0.3931
POMDP	1000	0.2749	<b>0.2720</b>	0.2802	<u>0.2799</u>	0.3092	0.4078

Table A3. Root Mean-Squared Error (RMSE) of the FQE estimators with different hyperparameter configurations.

Env/Dataset	FQE 1	FQE 2	FQE 3	FQE 4
<b>Hopper</b>				
medium-replay	30.2	15.5	133.5	153.4
medium	52.2	12.5	242.9	237.6
<b>HalfCheetah</b>				
medium-replay	126.0	65.0	439.7	318.8
medium	158.6	111.8	491.6	386.5
<b>Walker2d</b>				
medium-replay	185.8	167.4	301.6	167.7
medium	184.9	192.0	406.7	183.8

Walker2D. The reason is that we noticed the Q-value for Walker2D exploded if we used the same hyperparameters for the two other tasks. We should note that since OPERA does not require OPEs to be the same across tasks. The hyperparameter choices are around the Q-function neural network’s hidden sizes and how many epochs we train each Q-function. Generally, training too long / over-training leads to exploding Q-values.

## A.7. Sepsis and Graph Experiment Details

### A.7.1. SEPSIS

The first domain is based on the simulator and works by (Oberst and Sontag, 2019) and revolves around treating sepsis patients. The goal of the policy for this simulator is to discharge patients from the hospital. There are three treatments the policy can choose from antibiotics, vasopressors,

and mechanical ventilation. The policy can choose multiple treatments at the same time or no treatment at all, creating 8 different unique actions.

The simulator models patients as a combination of four vital signs: heart rate, blood pressure, oxygen concentration and glucose levels, all with discrete states (for example, for heart rate low, normal and high). There is a latent variable called diabetes that is present with a 20% probability which drives the likelihood of fluctuating glucose levels. When a patient has at least 3 of the vital signs simultaneously out of the normal range, the patient dies. If all vital signs are within normal ranges and the treatments are all stopped, the patient is discharged. The reward function is +1 if a patient is discharged, -1 if a patient dies, and 0 otherwise. We truncate the trajectory to 20 actions (H=20). For this simulator, early termination means we don’t get to observe a positive or negative return on the patient.

We follow the process described by (Oberst and Sontag, 2019) to marginalize an optimal policy’s action over 2 states: glucose level and whether the patient has diabetes. This creates the Sepsis-POMDP environment. We sample 200 and 1000 patients (trajectories) from Sepsis-POMDP environment with the optimal policy that has 5% chance of taking a random action. We also sample trajectories from the original MDP using the same policy; we call this the Sepsis-MDP environment.

**FQE Training** We use tabular FQE. Therefore, there is no representation mismatch. We additionally use cross-fitting, a form of procedure commonly used in causal inference (Chernozhukov et al., 2016). Cross-fitting is a sample-splitting procedure where we swap the roles of main and auxiliary samples to obtain multiple estimates and then av-

OPERA: Re-weighted Aggregates of Multiple Offline Policy Evaluation Estimators

Alg	Initial $\alpha$
CQL 1	1.0
CQL 2	10

Table A4.

Alg	Expectile
IQL 1	0.7
IQL 2	0.5

Table A5.

Alg	Alpha
TD3+BC 1	0.7
TD3+BC 2	0.5

Table A6.

Policy	Hopper (medium-replay)	Hopper (medium)
CQL 1	193.47	242.24
CQL 2	123.76	243.57
IQL 1	239.20	246.26
IQL 2	239.85	240.05
TD3+BC 1	183.48	231.81
TD3+BC 2	208.16	234.19

Table A7. Discounted perf of different policies on Hopper task.

Policy	Walker2D (medium-replay)	Walker2D (medium)
CQL 1	252.68	85.39
IQL 1	238.77	253.19
IQL 2	130.29	243.51
CQL 2	247.03	198.92
TD3+BC 1	211.28	247.22
TD3+BC 2	183.38	237.85

Table A8. Discounted perf of different policies on Walker2D task.

erage the results. The main goal of cross-fitting is to reduce overfitting. We notice significant performance improvement of our FQE estimator after using cross-fitting. We present the RMSE of each of our trained FQE estimator in Table A3.

#### A.7.2. GRAPH

For the graph environment, we set the horizon  $H=4$ , with either POMDP or MDP and ablate on the stochasticity of transition and reward function. The optimal policy for the Graph domain is simply the policy that chooses action 0. All the experiments reported have 512 trajectories.

Policy	HalfCheetah (medium-replay)	HalfCheetah (medium)
CQL 1	363.35	601.59
IQL 1	394.06	436.52
IQL 2	362.65	423.37
CQL 2	354.23	539.03
TD3+BC 1	407.96	441.20
TD3+BC 2	318.22	422.65

Table A9. Discounted perf of different policies on HalfCheetah task.

Hopper/ HalfCheetah	Q-Function Network	Training Epochs
FQE 1	[256, 256, 256]	2
FQE 2	[256, 256, 256]	3
FQE 3	[512, 512]	1
FQE 4	[512, 512]	2

Table A10. FQE Hyperparameters. Training epochs were chosen to be an early checkpoint and a late checkpoint (before exploding Q-values).

Walker2D	Q-Function Network	Training Epochs
FQE 1	[128, 256, 512]	2
FQE 2	[128, 256, 512]	5
FQE 3	[512, 512]	1
FQE 4	[512, 512]	2

Table A11. FQE Hyperparameters. Training epochs were chosen to be an early checkpoint and a late checkpoint (before exploding Q-values).

Policy	Hopper (medium-replay-v2)	Hopper (medium-v2)
CQL 1	433.40	2550.03
CQL 2	439.56	2787.95
IQL 1	3144.02	1768.19
IQL 2	2177.90	2028.27
TD3 1	1104.04	1977.88
TD3 2	910.26	1751.87

Table A12. Undiscounted perf of different policies on Hopper task.

Policy	Walker2D (medium-replay-v2)	Walker2D (medium-v2)
CQL 1	3732.01	145.25
IQL 1	2383.09	3044.03
IQL 2	776.79	3194.87
CQL 2	3073.49	1409.01
TD3 1	2250.07	3920.79
TD3 2	1656.82	3732.23

Table A13. Undiscounted perf of different policies on Walker2D task.

Policy	HalfCheetah (medium-replay-v2)	HalfCheetah (medium-v2)
CQL 1	4053.04	7894.69
CQL 2	4192.01	6875.66
IQL 1	4995.02	5704.12
IQL 2	4657.00	5475.88
TD3 1	5324.46	5758.83
TD3 2	5002.90	5420.27

Table A14. Undiscounted perf of different policies on HalfCheetah task.