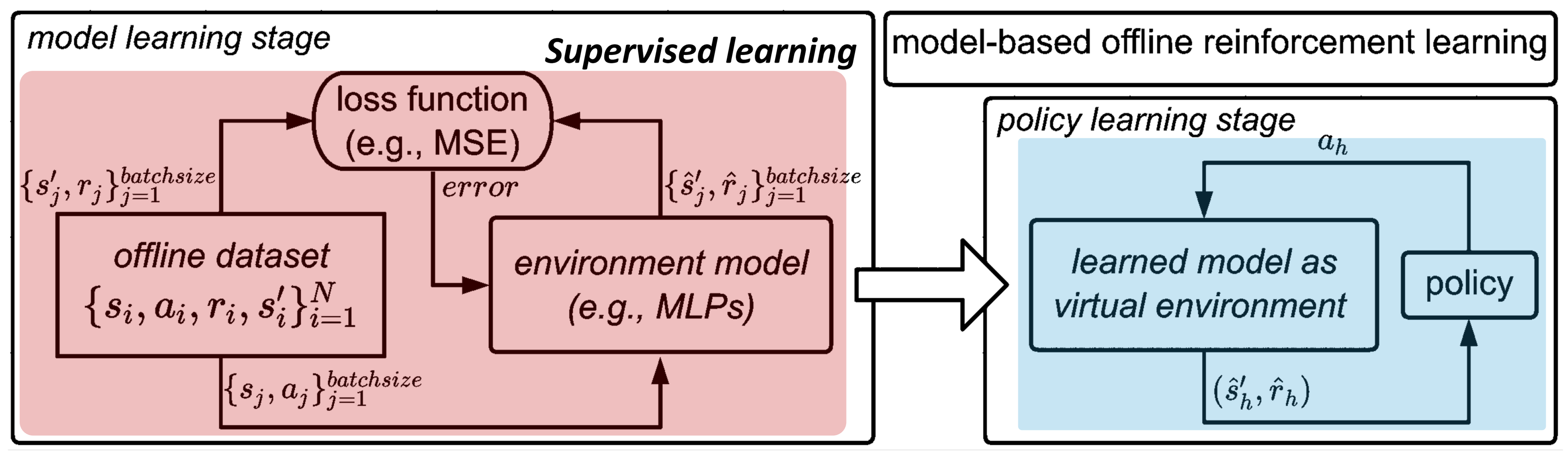
Pareto Policy Pool for Model-based Offline Reinforcement Learning

Yijun Yang[1,4], Jing Jiang[1], Tianyi Zhou[2,3], Jie Ma[1], Yuhui Shi[4]

[1]Australian AI Institute, University of Technology Sydney; [2]University of Washington, Seattle; [3]University of Maryland, College Park; [4]CSE, Southern University of Science and Technology

{yijun.yang-1, jie.ma-5}@student.uts.edu.au, jing.jiang@uts.edu.au, tianyizh@uw.edu, shiyh@sustech.edu.cn*

Model-based Offline RL with Uncertainty Regularization



Model-based RL is a promising paradigm for **offline** policy learning *because*

- the learned model fully exploits the pre-collected data (***offline dataset***);
- the agent avoids costly interactions with real environments required by online RL, but instead interacts with a ***virtual environment*** and aims to maximize:

$$\max_\pi \hat{R}_{\hat{\rho}_0}(\pi, \widehat{\mathcal{M}}) = \mathbb{E}_{s_0\sim\hat{\rho}_0,\pi}\left[\sum_{h=0}^{H-1} \hat{r}(\hat{s}_h, a_h)\right], \text{ i.e., model-estimated return.}$$

- However, directly maximizing $\hat{R}$ usually **fails** due to the ***epistemic uncertainty*** of model on out-of-distribution (OOD) state-action pairs.
- The policy may suffer from "**model exploitation**", i.e., it achieves a ***high model return by repeatedly visiting some OOD pairs*** but ***performs poorly*** in the real environment.

A widely-used solution to this problem is uncertainty regularization:

$$\max_\pi \tilde{R}_{\hat{\rho}_0}(\pi, \widehat{\mathcal{M}}) = \mathbb{E}_{s_0\sim\hat{\rho}_0,\pi}\left[\sum_{h=0}^{H-1} (\hat{r}(\hat{s}_h, a_h) - \lambda u(\hat{s}_h, a_h))\right]$$

***Hyperparameter** controls the trade-off between $\hat{r}$ and u.* ***Regularization term:** the epistemic uncertainty of model.*

- Its performance **significantly relies on the trade-off** between model reward and uncertainty in the optimization objective.
- However, it is usually **challenging or intractable to determine the optimal trade-off under *offline RL*.**

Bi-objective Formulation for Model-based Offline RL

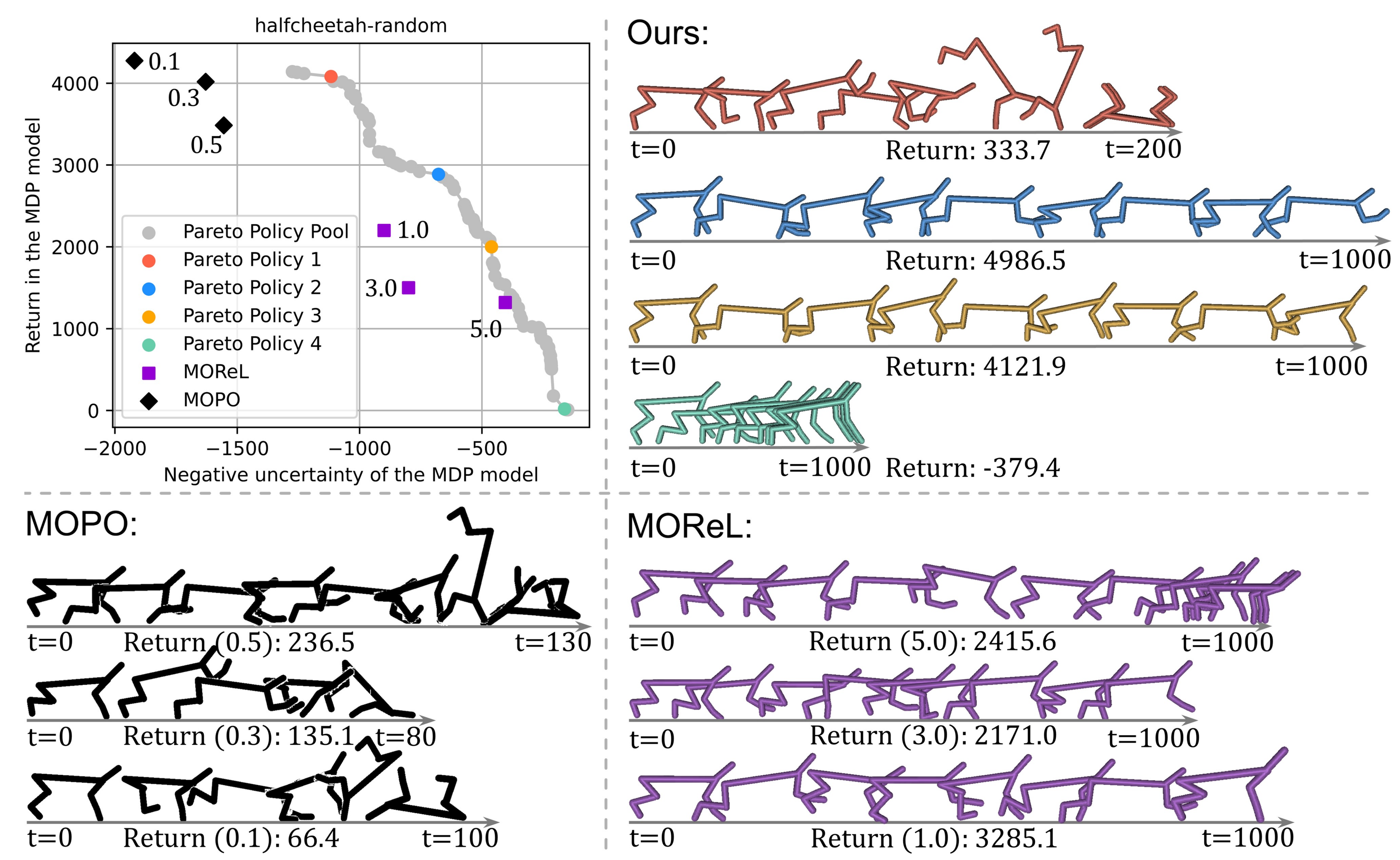
To avoid those challenges caused by **uncertainty-regularized methods**, we study a ***bi-objective formulation*** for model-based offline RL

$$\max_\pi \tilde{R}_{\hat{\rho}_0}(\pi, \widehat{\mathcal{M}}) = \mathbb{E}_{s_0\sim\hat{\rho}_0,\pi}\left[\sum_{h=0}^{H-1} (\hat{r}(\hat{s}_h, a_h) - \lambda u(\hat{s}_h, a_h))\right] \Rightarrow \textit{Single-objective optimization}$$

$$\max_\pi \mathbf{J}_{\hat{\rho}_0}(\pi, \widehat{\mathcal{M}}) = \mathbb{E}_{s_0\sim\hat{\rho}_0,\pi}\left[\sum_{h=0}^{H-1} (\hat{r}(\hat{s}_h, a_h), -u(\hat{s}_h, a_h))^\intercal\right] \Rightarrow \textit{Our Bi-objective optimization}$$

- that aims at **producing a pool of diverse policies on the Pareto front** performing different levels of trade-offs,
- thus it provides **the flexibility to select the best trade-off policy** for the testing environment from the pool.

An Example of Pareto Policy Pool



- **Pareto policy 1 is overly optimal on the model return,** so it runs fast at the beginning but quickly falls to the ground due to the "model exploitation".
- **Pareto policy 4 suppressing model uncertainty is overly conservative,** which keeps standing because it avoids taking exploratory actions that potentially increase the uncertainty.
- **Pareto policy 2&3 with the more balanced trade-off** between the model return and uncertainty perform better and achieve higher scores in the testing environment.
- By running multiple instances with different regularization weights, **MOPO and MOReL can only produce a few separated policies,** and it is difficult to find a promising policy that outperforms the policies trained by our methods.

Our Method: Pareto Policy Pool (P3)

Algorithm 1 Pareto policy pool (P3) for model-based offline RL

input: dataset D, constraint $\psi < 0$, step size η, num. reference vectors n, $T_g \gg T_l$
initialize: environment models, Pareto policy pool $\mathcal{P} = \varnothing$, $0 < \tau_a < \tau_b < 1$ for Eq. (5), $0 < \epsilon < \tau_a$ for Eq. (8), number of updates: $T = n(T_g + 2T_l)$
Train the model on D using supervised learning;
Generate n reference vectors $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ by Eq. (5);
for $i \in \{1, \ldots, n\}$ (in parallel) **do** ▷ Diverse Pareto Policies
  Initialize a policy π_i
  for $j = 0, 1, \ldots, T_g - 1$ **do**
    Update the parameters of π_i by Alg. 2 with $\boldsymbol{v}_i$;
  Generate $\{\boldsymbol{v}_i^+, \boldsymbol{v}_i^-\}$ to $\boldsymbol{v}_i$ by Eq. (8); ▷ Local Pareto Extension
  for $\boldsymbol{v}' \in \{\boldsymbol{v}_i^+, \boldsymbol{v}_i^-\}$ **do**
    for $j' = 0, 1, \ldots, T_l - 1$ **do**
      $\mathcal{P} = \mathcal{P} \cup \{\pi_i\}$; ▷ Store Pareto policies into the pool
      Update the parameters of π_i by Alg. 2 with $\boldsymbol{v}'$;
output: $\mathcal{P}$;

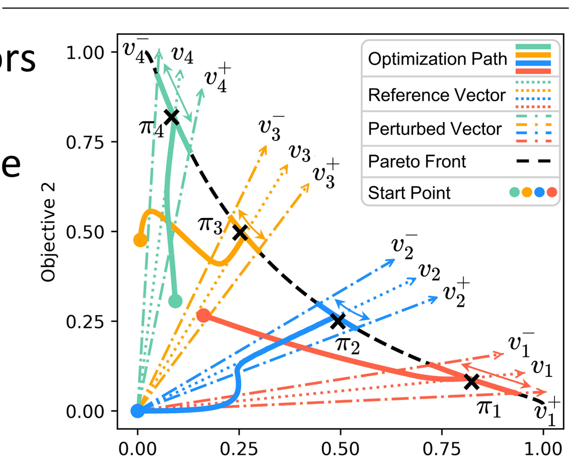
Algorithm 2 A two-stage method for solving constrained bi-objective optimization

input: $\pi_{\theta_t}, \boldsymbol{v}_i, \psi$
if $\Psi(\pi_{\theta_t}, \boldsymbol{v}_i) < \psi$ **then** ▷ Correction stage
  if $J^{\hat{r}}(\pi_{\theta_t})/J^u(\pi_{\theta_t}) < v_i^1/v_i^2$ **then**
    Compute $\nabla_\theta J^{\hat{r}}(\pi_{\theta_t})$;
    $\theta_{t+1} = \theta_t + \eta\nabla_\theta J^{\hat{r}}(\pi_{\theta_t})$;
  else
    Compute $\nabla_\theta J^u(\pi_{\theta_t})$;
    $\theta_{t+1} = \theta_t + \eta\nabla_\theta J^u(\pi_{\theta_t})$;
else ▷ Ascending stage
  Compute $\nabla_\theta \mathbf{F}(\pi_{\theta_t})$;
  Find $\boldsymbol{\alpha}_t^*$ to Eq. (7);
  $\theta_{t+1} = \theta_t + \eta\boldsymbol{\alpha}_t^*\nabla_\theta \mathbf{F}(\pi_{\theta_t})$;
$t \leftarrow t + 1$
output: π_{θ_t}

(Diverse Pareto Polices) P3 generates multiple reference vectors $\{v_i\}_{i=1}^n$ in the objective space, each forming a constraint to the bi-objective optimization and targeting a different region on the Pareto front, and optimizes the policy π_i by Alg. 2 towards v_i

$$\max_\pi \mathbf{J}_{\hat{\rho}_0}(\pi, \widehat{\mathcal{M}}) = \mathbb{E}_{s_0\sim\hat{\rho}_0,\pi}\left[\sum_{h=0}^{H-1} (\hat{r}(\hat{s}_h, a_h), -u(\hat{s}_h, a_h))^\intercal\right]$$

$$\text{s.t. } \Psi(\pi, v_i) \triangleq -D_{\mathrm{KL}}\left(\frac{v_i}{\|v_i\|_1} \middle\| \frac{\mathbf{J}(\pi)}{\|\mathbf{J}(\pi)\|_1}\right) \geq \psi$$



(Local Pareto Extension to Reduce Training Cost) Each v_i is perturbed in opposing directions, and then these perturbed vectors are further optimized via Alg. 2, with intermediate policies being added to the policy pool.

Experiments on D4RL Gym Benchmark

		BCQ	BEAR	CQL	UWAC*	TD3+BC	MOPO	MOPO*	MOReL	COMBO*	P3+FQE	P3
Random	HalfCheetah	2.2 ±0.1	2.3 ±0.1	21.7 ±0.6	14.5 ±3.3	10.6 ±1.7	35.9 ±2.9	35.4 ±2.5	30.3 ±5.9	38.8	37.4 ±5.1	40.6 ±3.7
	Hopper	8.1 ±0.5	3.9 ±2.3	8.1 ±1.4	22.4 ±12.1	8.6 ±0.4	16.7 ±12.2	11.7 ±0.4	44.8 ±4.8	17.9	33.8 ±0.4	35.4 ±0.8
	Walker2d	4.6 ±0.7	12.8 ±10.2	0.5 ±1.3	15.5 ±11.7	1.5 ±1.4	4.2 ±5.7	13.6 ±2.6	17.3 ±8.2	7.0	19.7 ±0.5	22.9 ±0.6
Medium	HalfCheetah	45.4 ±1.7	42.9 ±0.2	49.2 ±0.3	46.5 ±2.5	47.8 ±0.4	73.1 ±2.4	42.3 ±1.6	20.4 ±13.8	54.2	61.4 ±2.0	64.7 ±1.6
	Hopper	53.9 ±3.7	51.8 ±3.9	62.7 ±3.7	88.9 ±12.2	69.1 ±4.5	38.3 ±34.9	28.0 ±12.4	53.2 ±32.1	94.9	105.9 ±1.4	106.8 ±0.7
	Walker2d	74.5 ±3.7	-0.2 ±0.1	57.5 ±8.3	57.5 ±7.8	81.3 ±3.0	41.2 ±30.8	17.8 ±19.3	10.3 ±8.9	75.5	71.1 ±3.5	81.3 ±2.0
Medium -replay	HalfCheetah	40.9 ±1.1	36.3 ±3.1	47.2 ±0.4	46.8 ±3.0	44.8 ±0.5	69.2 ±1.1	53.1 ±2.0	31.9 ±6.1	55.1	43.4 ±1.1	48.2 ±0.6
	Hopper	40.9 ±16.7	52.2 ±19.3	28.6 ±0.9	39.4 ±6.1	57.8 ±17.3	32.7 ±9.4	67.5 ±24.7	54.2 ±32.1	73.1	89.5 ±2.0	94.6 ±1.4
	Walker2d	42.5 ±13.7	6.9 ±7.8	45.3 ±2.7	27.0 ±6.3	81.9 ±2.7	73.7 ±9.4	39.0 ±9.6	13.7 ±8.1	56	60.1 ±9.5	64.0 ±8.2
	Mean	34.8 ±4.7	23.2 ±5.2	35.6 ±2.2	39.8 ±7.2	44.8 ±3.5	42.8 ±12.1	34.3 ±8.3	30.7 ±13.3	52.5	58.0 ±2.8	62.1 ±2.2
Expert	HalfCheetah	92.7 ±2.5	92.7 ±0.6	97.5 ±1.8	128.6 ±2.9	96.3 ±0.9	81.3 ±21.8	–	2.2 ±5.4	–	81.4 ±1.72	88.8 ±0.4
	Hopper	105.3 ±8.1	54.6 ±21.1	105.4 ±5.9	135.0 ±14.1	109.5 ±4.1	62.5 ±28.9	–	26.2 ±13.9	–	110.6 ±1.2	111.3 ±0.5
	Walker2d	109.1 ±0.4	106.8 ±6.8	108.9 ±0.4	121.1 ±22.4	110.3 ±0.4	62.4 ±3.2	–	-0.3 ±0.3	–	102.0 ±3.4	106.7 ±0.2
Medium -expert	HalfCheetah	93.9 ±1.2	46.1 ±4.7	70.6 ±13.6	127.4 ±3.7	88.9 ±5.3	70.3 ±21.9	63.3 ±38.0	35.9 ±19.2	90	57.1 ±16.0	69.9 ±10.5
	Hopper	108.6 ±5.9	50.6 ±25.3	111.0 ±1.2	134.7 ±21.2	102.0 ±10.1	60.6 ±32.5	23.7 ±6.0	52.1 ±27.7	111.1	109.4 ±1.3	110.8 ±0.5
	Walker2d	109.7 ±0.6	22.1 ±44.5	109.7 ±0.3	99.7 ±12.2	110.5 ±0.3	77.4 ±27.9	44.6 ±12.9	3.9 ±2.8	96.1	90.3 ±4.2	98.9 ±3.4
	Mean	103.2 ±3.1	62.2 ±17.2	100.5 ±3.9	124.4 ±12.8	102.9 ±3.5	69.1 ±22.7	43.9 ±19.0	20.0 ±7.7	99.1	91.8 ±4.6	97.7 ±2.6
Total	Mean	62.2 ±4.1	38.8 ±10.0	61.6 ±2.9	73.6 ±9.4	68.0 ±3.5	53.3 ±16.3	36.8 ±11.0	26.4 ±11.1	64.2	71.5 ±3.5	76.3 ±2.4

Table 1: **Results on D4RL Gym experiments.** Normalized score (mean±std) over the final 10 evaluations and 5 seeds. ∗ marks previously reported results. Dataset quality gradually improves from Random to Medium-expert.

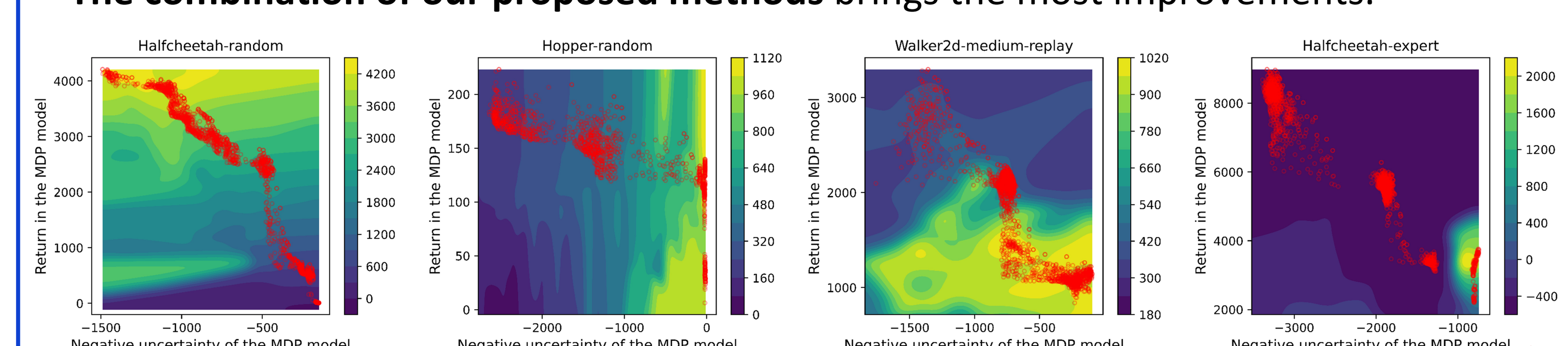
- P3 achieves the **highest average-score over all datasets** compared to recently proposed methods, including model-based and -free ones.
- P3 significantly **outperforms the baseline methods in 5 out of the 9 low/medium-quality datasets**, showing its advantages on learning from non-expert experiences.
- Online selection of multiple policies required by P3 can be expensive. We replace the online selection with FQE, an offline policy evaluation method, which (approximately) evaluates each Pareto policy using offline data only.
- We surprisingly find that **"P3+FQE (offline policy selection)" only slightly degrades from the original P3** on the performance but results in the same inference cost as other baselines.

Ablation Study

Data Quality	Random			Medium-replay			Medium-expert		
Environment	HalfCheetah	Hopper	Walker2d	HalfCheetah	Hopper	Walker2d	HalfCheetah	Hopper	Walker2d
P3: scalarization	15.5 ±0.8	32.3 ±1.5	15.2 ±5.0	40.1 ±1.4	88.5 ±8.3	49.9 ±15.0	52.4 ±7.3	77.3 ±22.9	84.7 ±8.5
P3: no StateNorm	35.3 ±2.5	34.9 ±0.2	21.8 ±0.3	41.7 ±0.4	82.3 ±12.9	61.6 ±9.4	47.1 ±0.3	99.9 ±6.0	90.3 ±2.2
P3: no RankShaping	37.6 ±4.4	33.6 ±0.3	27.3 ±6.2	44.3 ±0.7	95.6 ±1.7	64.7 ±3.9	66.3 ±1.9	108.3 ±1.2	97.0 ±2.6
P3: no ParetoExtension	31.2 ±2.4	5.2 ±0.4	0.1 ±0.2	43.4 ±1.6	91.3 ±4.9	2.0 ±0.6	4.7 ±3.2	88.2 ±16.4	0.3 ±0.1
P3: no BehaviorCloning	38.2 ±1.4	35.5 ±0.5	24.1 ±1.1	45.4 ±1.8	97.1 ±2.1	26.1 ±4.9	52.2 ±3.5	89.8 ±16.6	69.1 ±9.1
P3: our version	40.6 ±3.7	35.4 ±0.8	22.9 ±0.6	48.2 ±0.6	94.6 ±1.4	64.0 ±8.2	69.9 ±10.5	110.8 ±0.5	98.9 ±3.4
MOPO	35.9 ±2.9	16.7 ±12.2	4.2 ±5.7	69.2 ±1.1	32.7 ±9.4	73.7 ±9.4	70.3 ±21.9	60.6 ±32.5	77.4 ±27.9
TD3+BC	10.6 ±1.7	8.6 ±0.4	1.5 ±1.4	44.8 ±0.5	57.8 ±17.3	81.9 ±2.7	88.9 ±5.3	102.0 ±10.1	110.5 ±0.3

We conduct a thorough ablation study towards five variants of P3, each removing/changing one component used in P3.

- **The combination of our proposed methods** brings the most improvements.



Low ***The quality of datasets*** ***High***

Model-based offline RL's performance in the real environment (**heatmap**) under different trade-offs between the model return (**y-axis**) and uncertainty (**x-axis**).

- Exploring the whole Pareto front (**P3 does**) is essential to our appealing results on low-quality datasets.
- Carefully tuning the trade-off (**other baselines do**) suffices to find a good policy for high-quality datasets since the optimal policies gather in a small region and associate with one trade-off level.