IMITATION BEYOND EXPECTATION VIA SECOND-ORDER STOCHASTIC DOMINANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Expected utility theory provides an ordering for policies when the utilities of behavior sequences are given. Many imitation learning methods extend this perspective by seeking policies that match or exceed the utility of demonstrations in expectation, despite the exact utility function only being assumed to be Markovian (and linear). However, this reduces a distribution of demonstrations to its average properties, ignoring the distinct characteristics of individual demonstrations that may be important to imitate. We adopt second-order multivariate stochastic dominance, a stricter notion for partially ordering distributions, to guide imitation learning. It broadens the expected and risk-sensitive performance guarantees of imitation learning to all non-decreasing and concave utility functions. We investigate both left-sided (i.e., risk-avoidant) and right-sided (i.e., risk-seeking) variants of this stochastic dominance to demonstrate the advantages compared to employing stricter first-order stochastic dominance and other imitation learning baselines.

1 INTRODUCTION

Imitation learning is the task of learning a policy π for an imitator to produce highly desirable trajectories $\xi \sim \pi$ in a decision process given desirable demonstrated trajectories $\{\tilde{\xi}_i\} \sim \tilde{\pi}$. It differs from traditional reinforcement learning because a reward signal r for evaluating the utility of trajectories (e.g., based on state-action pairs defining the trajectories: $\sum_{(s,a)\in\xi} \mathrm{r}(s,a)$) is not assumed or available. Many imitation learning methods seek guarantees with respect to **expected utility** by either (approximately) matching (Ng and Russell, 2000; Abbeel and Ng, 2004; Ziebart et al., 2008; Ho and Ermon, 2016) or exceeding (Syed and Schapire, 2007) the expected utility received by the demonstrator, $\mathbb{E}_{\xi\sim\pi}[u(\xi)] \geq \mathbb{E}_{\tilde{\xi}\sim\tilde{\pi}}[u(\tilde{\xi})]$, despite not knowing the reward/utility function u. By assuming a (positive) linear family of utility functions, $R_{\mathbf{w}}(\xi) = \mathbf{w}\cdot\mathbf{F}_\xi$ with $\mathbf{w} \succeq \mathbf{0}$ given vector-valued trajectory features $\mathbf{F}_\xi$, an expected reward no worse than the demonstrator's is guaranteed:

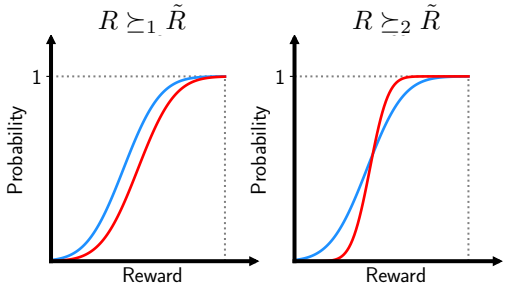

Figure 1: First-order univariate stochastic dominance (left) requires the cumulative distribution function (CDF) of R (red) to be no larger than that of $\tilde{R}$ (blue), while second-order univariate stochastic dominance (right) requires the total area under the CDF curve of R, at any point, is no larger than that of $\tilde{R}$. Both are stricter criteria than dominance in expected value, $R \succeq_{\mathbb{E}} \tilde{R}$, which simply requires $\mathbb{E}_{R\sim\pi}[R] \geq \mathbb{E}_{\tilde{R}\sim\tilde{\pi}}[\tilde{R}]$.

$$\mathbf{F} \succeq_{\mathbb{E}} \tilde{\mathbf{F}} \iff \mathbb{E}_{\mathbf{F}\sim\pi}\left[\mathbf{F}\right] \succeq \mathbb{E}_{\tilde{\mathbf{F}}\sim\tilde{\pi}}\left[\tilde{\mathbf{F}}\right] \implies \forall u \in \mathcal{U}_{\text{linear}+}\ \mathbb{E}_{\mathbf{F}\sim\pi}[u(\mathbf{F})] \geq \mathbb{E}_{\tilde{\mathbf{F}}\sim\tilde{\pi}}[u(\tilde{\mathbf{F}})]. \quad (1)$$

Univariate stochastic dominance (Levy, 1992) is a stricter criteria for partially ordering distributions. As illustrated by Figure 1, univariate first-order stochastic dominance (FSD) requires higher expected values over any range of percentiles for each distribution, while univariate second-order stochastic dominance (SSD) requires higher expected values for any range that includes the left-most tail. Both variants guarantee no worse conditional-value-at-risk (CVaR) for the returns produced by policy π compared to those from policy $\tilde{\pi}$. **Multivariate stochastic dominance** (Müller and Stoyan,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2002; Shaked and Shanthikumar, 2007) provides expected utility (and generalized CVaR tail risk) guarantees for a broader range of utility functions $u : \mathbb{R}^K \to \mathbb{R}$:

$$\mathbf{F} \succeq_i \tilde{\mathbf{F}} \iff \forall u \in \mathcal{U}_i \; \mathbb{E}_{\mathbf{F}\sim\pi}[u(\mathbf{F})] \geq \mathbb{E}_{\tilde{\mathbf{F}}\sim\tilde{\pi}}[u(\tilde{\mathbf{F}})], \tag{2}$$

that are non-decreasing ($\mathcal{U}_1$) for FSD; or non-decreasing and concave ($\mathcal{U}_2$) for SSD.

Previous work has investigated imitation learning based on first-order stochastic dominance (FSD) (Farajzadeh et al., 2025), demonstrating the multimodal coverage that is induced in the resulting policy. However, FSD can be challenging for a policy model to satisfy. For example, sufficient multimodality may not be easily produced by conditional Gaussian policy models commonly employed for continuous control environments. Additionally, FSD provides strict constraints over the entire distribution of returns, while imitation that focuses more closely on one side of the distribution (e.g., risk-avoiding or risk-seeking) may be more aligned with particular applications. We instead focus on second-order stochastic dominance (SSD), which relaxes the FSD requirements on the policy—particularly away from the (left or right) tail. This enables imitation learning that focuses on robustness with respect to the worst demonstrations (left-sided SSD) or outperformance of the best demonstrations (right-sided SSD).

Focusing on sample distributions arising from imitation learning settings, we develop imitative policy optimization for multivariate SSD guided by a linear program. It seamlessly interpolates between directly imitating demonstrations (i.e., behavioral cloning (Pomerleau, 1988)) and seeking better-than-demonstrator trajectories, often across multiple demonstrated behavior modes. Compared to recent work guiding imitation learning towards first-order stochastic dominance (Farajzadeh et al., 2025), second-order stochastic dominance is easier to achieve. Left-sided SSD focuses imitation on avoiding underperforming against the worst tails of the distribution of demonstrations, which is important for safety applications. Right-sided SSD seeks tail distribution improvements compared to the best demonstrations. Unlike RAIL (Santara et al., 2018) and RS-GAIL (Lacotte et al., 2019), which choose a specific expected tail return (e.g., CVaR_α) to constrain or trade-off with expected utility maximization, second-order stochastic dominance provides these tail guarantees for all tail thresholds.

2 Stochastic Dominance Background

We now review the key characteristics of stochastic dominance for univariate (§2.1) and multivariate (§2.2) distributions, their relationships, and related work on stochastic dominance verification and optimization (§2.3).

2.1 Univariate definitions and relationships

Stochastic dominance (Def. 1) provides a means for comparing two random variables. It is a stricter definition than using the expectation of the random variables, which we denote $R \succeq_{\mathbb{E}} \tilde{R} \iff \mathbb{E}_{R\sim\pi}[R] \geq \mathbb{E}_{\tilde{R}\sim\tilde{\pi}}[\tilde{R}] \triangleq \sum_r r\tilde{\mathbb{P}}(R = r)$, and thus only provides a partial ordering.

Definition 1. Scalar random variable R has **first-order stochastic dominance (FSD)** or **second-order stochastic dominance (SSD)** over $\tilde{R}$, denoted $R \succeq_1 \tilde{R}$ or $R \succeq_2 \tilde{R}$, respectively, based on the following equivalent FSD definitions (3a-3c) and SSD definitions (4a-4c):

$$\mathbb{P}_{R\sim\pi}(R \leq c) \leq \mathbb{P}_{\tilde{R}\sim\tilde{\pi}}(\tilde{R} \leq c); \tag{3a}$$

$$\int_{-\infty}^{c} \mathbb{P}_{R\sim\pi}(R \leq x)\, dx \leq \int_{-\infty}^{c} \mathbb{P}_{\tilde{R}\sim\tilde{\pi}}(\tilde{R} \leq x)\, dx; \tag{4a}$$

$$\text{RVaR}_{\alpha,\beta}(R) \geq \text{RVaR}_{\alpha,\beta}(\tilde{R}); \tag{3b}$$

$$\text{CVaR}_\alpha(R) \geq \text{CVaR}_\alpha(\tilde{R}); \tag{4b}$$

$$\forall u \in \mathcal{U}_1 \; \mathbb{E}_{R\sim\pi}\left[u(R)\right] \geq \mathbb{E}_{\tilde{R}\sim\tilde{\pi}}\left[u(\tilde{R})\right]; \tag{3c}$$

$$\forall u \in \mathcal{U}_2 \; \mathbb{E}_{R\sim\pi}\left[u(R)\right] \geq \mathbb{E}_{\tilde{R}\sim\tilde{\pi}}\left[u(\tilde{R})\right]; \tag{4c}$$

$\forall c \in \mathbb{R}$ and $\alpha, \beta \in (0, 1)$, where: $\mathcal{U}_1$ is the set of all non-decreasing utility functions $u : \mathbb{R} \to \mathbb{R}$; $\mathcal{U}_2$ is the set of all non-decreasing and concave utility functions $u : \mathbb{R} \to \mathbb{R}$; the **range value at risk** is $\text{RVaR}_{\alpha,\beta}(R) \triangleq \int_\alpha^\beta \text{VaR}_{1-\gamma}(R)\, d\gamma$ with **value at risk** defined as $\text{VaR}_{1-\alpha}(R) = \inf_{t\in\mathbb{R}}\{t : \mathbb{P}(R \leq t) \geq 1 - \alpha\}$; and the **conditional value at risk** (CVaR) is defined by a range that includes the left-most distribution tail: $\text{CVaR}_\alpha(R) \triangleq \text{RVaR}_{0,\alpha}(R) = \int_0^\alpha \text{VaR}_{1-\gamma}(R)\, d\gamma$.

As shown in Figure 1, FSD corresponds to the cumulative distribution function of R being no larger anywhere than the cumulative distribution function of $\tilde{R}$ equation 3a, while SSD corresponds to a smaller supercumulative distribution function equation 4a, which is the area under the cumulative distribution function up to threshold c.

Stochastic dominance is typically defined in terms of the left or lower tail of the distribution, and generally requires lower tails that are no worse or thicker than the dominated random variable. Right-sided stochastic dominance instead seeks better performance in the more desirable right or upper tail of the reward distribution, and is symmetrically defined as: $-\tilde{R} \succeq_2 -R$. The relationships between different variants of dominance from both left and right tails are shown in equation 5, with strictness decreasing from $\succeq_1$ to $\succeq_2$ to $\succeq_{\mathbb{E}}$:

$$\begin{array}{ccccc} R \succeq_1 \tilde{R} & \Longrightarrow & R \succeq_2 \tilde{R} & \Longrightarrow & R \succeq_{\mathbb{E}} \tilde{R} \\ \Updownarrow & \nearrow\!\!\searrow & & \nearrow\!\!\searrow & \Updownarrow \\ -\tilde{R} \succeq_1 -R & \Longrightarrow & -\tilde{R} \succeq_2 -R & \Longrightarrow & -\tilde{R} \succeq_{\mathbb{E}} -R. \end{array} \tag{5}$$

Unlike $\succeq_1$ and $\succeq_{\mathbb{E}}$, left-sided and right-sided dominance are not equivalent for the second-order variant (i.e., $R \succeq_2 \tilde{R} \not\Longleftrightarrow -\tilde{R} \succeq_2 -R$), and either of the two versions may be of interest depending on whether lower tail-risk-avoidance or upper tail-reward-seeking is desired for the application.

2.2 Multivariate definitions and relationships

In the multivariate setting, a K-dimensional feature vector $\mathbf{f} \in \mathbb{R}^K$ characterizes each trajectory sample from the imitator $\mathbf{F} \sim \pi$, and $\tilde{\mathbf{f}}$ similarly denotes features of samples drawn from the demonstrator $\tilde{\mathbf{F}} \sim \tilde{\pi}$. Multivariate extensions of the univariate stochastic dominance definitions are not equivalent (Armbruster and Luedtke, 2015; Kopa and Petrová, 2018). For example, multivariate extensions of the cumulative density function definition of FSD equation 3a are less strict than the utility-based definition equation 3c (Kopa and Petrová, 2018). In addition to the generalizations of equation 3c and equation 4c that Eq. equation 1 provides, multivariate definitions based on reductions to univariate stochastic dominance for each coordinate equation 6 or for all positive linear functions equation 7 (Dentcheva and Wolfhagen, 2015) are related (Armbruster and Luedtke, 2015) as shown in Eq. equation 8:

$$\mathbf{F} \succeq_i^{\text{coord}} \tilde{\mathbf{F}} \iff \forall k \in [K]\ F_k \succeq_i \tilde{F}_k; \tag{6}$$

$$\mathbf{F} \succeq_i^{\text{lin}} \tilde{\mathbf{F}} \iff \forall u \in \mathcal{U}_{\text{lin}+}\ u(\mathbf{F}) \succeq_i u(\tilde{\mathbf{F}}). \tag{7}$$

$$\begin{array}{ccccc} \mathbf{F} \succeq_1 \tilde{\mathbf{F}} & \Rightarrow & \mathbf{F} \succeq_1^{\text{lin}} \tilde{\mathbf{F}} & \Rightarrow & \mathbf{F} \succeq_1^{\text{coord}} \tilde{\mathbf{F}} \\ \Downarrow & & \Downarrow & & \Downarrow \\ \mathbf{F} \succeq_2 \tilde{\mathbf{F}} & \Rightarrow & \mathbf{F} \succeq_2^{\text{lin}} \tilde{\mathbf{F}} & \Rightarrow & \mathbf{F} \succeq_2^{\text{coord}} \tilde{\mathbf{F}}. \end{array} \tag{8}$$

From the imitation learning perspective, linear multivariate stochastic dominance equation 7 provides risk-sensitivity guarantees (RVaR or CVaR) for all non-negative linear utility functions.

The more general definition of multivariate stochastic dominance of Eq. equation 2 can similarly be viewed as a reduction to univariate stochastic dominance,

$$\mathbf{F} \succeq_i \tilde{\mathbf{F}} \iff \forall u \in \mathcal{U}_i\ u(\mathbf{F}) \succeq_i u(\tilde{\mathbf{F}}), \tag{9}$$

for the broader set of non-decreasing (FSD) and non-decreasing/concave (SSD) utility functions. This follows from the composition of non-decreasing ($\mathcal{U}_1$) [and concave ($\mathcal{U}_2$)] functions also exactly comprising the set of non-decreasing [and concave] functions. The risk-sensitive properties of univariate stochastic dominance are naturally extended to the multivariate setting under this perspective. For example, the expected deficiency below some performance threshold c is no greater for $\mathbf{F}$ than $\tilde{\mathbf{F}}$ under SSD equation 9:

$$\mathbf{F} \succeq_2 \tilde{\mathbf{F}} \iff \forall u \in \mathcal{U}_2\ \forall c \in \mathbb{R}\ \mathbb{E}_{\mathbf{F}\sim\pi}\left[[c - u(\mathbf{F})]_+\right] \leq \mathbb{E}_{\tilde{\mathbf{F}}\sim\tilde{\pi}}\left[[c - u(\tilde{\mathbf{F}})]_+\right]. \tag{10}$$

Previous work has characterized multivariate FSD and SSD for sample distributions of equal size (Armbruster and Luedtke, 2015):

$$\exists \sigma \in \mathcal{S} : \forall i \ \mathbf{f}_i \succeq \tilde{\mathbf{f}}_{\sigma_i} \iff \mathbf{F} \succeq_1 \tilde{\mathbf{F}}; \tag{11}$$

$$\exists \gamma \in \Gamma : \forall i \ \mathbf{f}_i \succeq \sum_j \gamma_{i,j} \tilde{\mathbf{f}}_j \iff \mathbf{F} \succeq_2 \tilde{\mathbf{F}}, \tag{12}$$

where $\mathcal{S}$ is the set of hard one-to-one matchings (i.e., permutations) and Γ is the set of all soft one-to-one matchings: $\forall i \in [N] \sum_j \gamma_{i,j} = 1$ and $\forall j \in [N] \sum_i \gamma_{i,j} = 1$ (illustrated by Figure 2). Though $\mathbf{F} \succeq_2 \tilde{\mathbf{F}}$ is stricter than $\mathbf{F} \succeq_2^{\text{lin}} \tilde{\mathbf{F}}$, it is computationally more tractable to verify using equation 12, while equation 7 is verified by using a (small) set of utility functions that are challenging to identify.

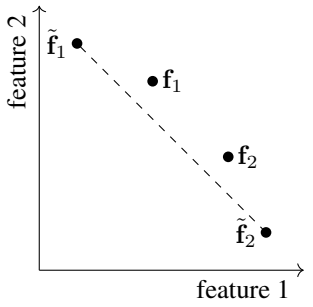

Figure 2: $\mathbf{F} \succeq_2 \tilde{\mathbf{F}}$ verified by interpolating $\tilde{\mathbf{f}}_1$ and $\tilde{\mathbf{f}}_2$, but $\mathbf{F} \not\succeq_1 \tilde{\mathbf{F}}$ since $\mathbf{f}_i \not\succeq \tilde{\mathbf{f}}_j \ \forall i, j$.

2.3 Related work

Univariate stochastic dominance has been most widely applied to portfolio optimization and resource allocation problems (Kuosmanen, 2004; Dentcheva and Ruszczyński, 2006; Armbruster and Luedtke, 2015; Post and Kopa, 2017; Sharma and Mehra, 2017) to produce allocations that are broadly preferable over reference distributions. Machine learning approaches have been developed to improve scalability of optimization methods under stochastic dominance constraints (Dai et al., 2023). Univariate stochastic dominance has also been applied to AI alignment problems so that the distribution of risks/costs are preferable to a reference distribution, e.g., in large language model applications (Chittepu et al., 2026). Related to this, Rioux et al. (2024) develop statistical tests using multivariate first-order stochastic dominance for comparing LLM performance.

Our work is most similar to Farajzadeh et al. (2025), which uses optimal transport to guide policy optimization towards achieving multivariate FSD over demonstrated trajectory samples. We instead pursue second-order stochastic dominance to more narrowly focus imitation learning to align in the tail (left or right) of distribution returns. Other risk-sensitive imitation learning methods (Santara et al., 2018; Majumdar et al., 2017) focus more narrowly yet on optimizing specific conditional-value-at-risk thresholds—all of which are implied by second-order stochastic dominance.

3 Approach

3.1 Problem setting

We consider the imitation learning setting in which we seek an imitator policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ that performs well compared to a demonstrator policy $\tilde{\pi}$ given demonstrator sample trajectories $\tilde{\xi} \sim \tilde{\pi}$ from a sequential decision process $(\mathcal{S}, \mathcal{A}, \mathcal{T})$ with state-transition dynamics $\mathcal{T}$ and unknown reward function. Seeking to mimic the demonstrator policy, $\pi = \tilde{\pi}$, e.g., by memorizing the most prevalent demonstrator action for any state, faces limitations due to a finite number of demonstrated trajectory samples. Instead, we seek to outperform the demonstrator, which requires additional assumptions to guarantee. Specifically, we assume a given feature representations $\mathbf{f} : \Xi \to \mathbb{R}^K$ for characterizing each trajectory with a known direction for improvement or sign (Syed and Schapire, 2007). Such feature representations are common in engineered systems (e.g., autonomous vehicles, robotics) for which the more desirable direction of optimizing each feature (e.g., proximity to obstacles, high frequency control changes) is known, but how features should be aggregated to define a utility function is unknown. We assume only sample trajectories are available from both the imitator's policy π and the demonstrator's $\tilde{\pi}$. We also assume simulator access for sampling trajectories from the imitator's policy.

3.2 Slack-augmented stochastic dominance

We formulate our approach for policy optimization using second-order stochastic dominance based on the following question: *Given sample distributions of size N, what sub-probability measure*

for sample distribution $\mathbf{F}$ *has second-order dominance over the largest sub-probability measure bounded by the sample distribution* $\tilde{\mathbf{F}}$*?* The answer ultimately guides our policy improvement.

Expanding from the soft matching constraints for defining SSD equation 12, we employ the following linear program to measure second-order stochastic dominance with slack variables $\{\epsilon_j\}$ that allow a portion of each demonstration to be unmatched to imitator trajectory mixtures.

Definition 2. The slack-augmented (left-sided) SSD linear program is:

$$\min_{\epsilon \succeq \mathbf{0}\ \gamma \succeq \mathbf{0}} \sum_{j=1}^{M} \epsilon_j \text{ such that: } \sum_{j=1}^{M} \tilde{\mathbf{f}}_j \gamma_{ij} \leq \mathbf{f}_i \sum_{j=1}^{M} \gamma_{ij}\ \forall i \in [N] \text{ and } \epsilon_j + \sum_{i=1}^{N} \gamma_{ij} = \frac{1}{M}\ \forall j \in [M]. \quad (13)$$

When $\mathbf{F} \succeq_2 \tilde{\mathbf{F}}$, this optimization is solved with zero slack (i.e., with an objective value of zero). It can similarly solved with an objective of zero sometimes even when $\mathbf{F} \not\succeq_2 \tilde{\mathbf{F}}$ using a new distribution over the imitator samples $\mathbb{P}_{\hat{\mathbf{F}}}(\mathbf{f}_i) = \sum_j \gamma_{i,j}$, in which case $\hat{\mathbf{F}} \succeq_2 \tilde{\mathbf{F}}$. More generally, when slack is required for the solution, the resulting sub-probability measure has important second-order stochastic dominance properties.

Proposition 3. *Using the solution variables from Def. 2 as* γ^* *and* ϵ^**, and assuming* $\sum_{i,j} \gamma_{i,j} > 0$*, let random feature vector* $\hat{\mathbf{F}}$ *have re-normalized probability* $\frac{\sum_j \gamma^*_{i,j}}{\sum_{i,j} \gamma^*_{i,j}}$ *assigned to each* $\hat{\mathbf{f}}_i$ *and let random feature vector* $\tilde{\mathbf{F}}'$ *have re-normalized probability* $\frac{\frac{1}{N} - \epsilon^*_i}{1 - \sum_i \epsilon^*_i}$ *assigned to each* $\tilde{\mathbf{f}}_j$*. Then second-order stochastic dominance between these distributions is implied:* $\hat{\mathbf{F}} \succeq_2 \tilde{\mathbf{F}}'$.

3.3 Policy optimization via target distribution matching

We define a target trajectory distribution from the imitator and demonstrator trajectory samples and our slack-augmented linear program's solution (Def. 2) to guide imitation learning towards SSD:

$$\mathbb{P}_{\text{target}}(\xi) \triangleq \mathbb{I}[\xi = \xi_i] \sum_j \gamma^*_{i,j} + \mathbb{I}[\xi = \tilde{\xi}_j] \epsilon^*_j, \quad (14)$$

where the indicator function $\mathbb{I}$ outputs 1 when its argument is true and 0 otherwise. For the $(1 - \sum_j \epsilon_j)$ portion of the sample distribution of demonstrations that can be stochastically dominated, $\hat{\mathbf{F}}$ defines the target, and for each remaining $\epsilon_j > 0$, a demonstrator sample is employed as part of the target distribution.

Corollary 4. *Following from Proposition 3, the random vector of features defined by* $\mathbb{P}_{trg}$ *also has second-order stochastic dominance over the demonstrations:* $\mathbf{F}_{trg} \succeq_2 \tilde{\mathbf{F}}$.

We optimize the imitator policy π_θ based on the target probability distribution:

$$\theta \leftarrow \theta + \eta \sum_{\xi \in \{\xi_i\} \cup \{\tilde{\xi}_j\}} \mathbb{P}_{\text{target}}(\xi) \nabla_\theta \log \pi_\theta(\xi). \quad (15)$$

This distribution-matching optimization is very similar to policy gradient optimization (e.g., REINFORCE) (Williams, 1992), but uses the target probability as the optimization signal rather than the reward or advantage function.

Algorithm 1 SSD policy optimization

Input: Set of M demonstrated trajectories $\{\tilde{\xi}_j\}$, feature function $\mathbf{f} : \Xi \to \mathbb{R}^K$, initial imitator policy π_θ, state transition dynamics $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$.

Output: Policy π_θ optimized for SSD.

1: **while** π_θ not converged **do**
2: Sample N trajectories $\{\xi_i\}$ from $\pi_\theta \times \mathcal{T}$.
3: Solve the LP of Def. 2 to obtain γ^* and ϵ^*.
4: Define $\mathbb{P}_{\text{target}}$ according to Eq. equation 14 or equation 16.
5: Optimize π_θ towards $\mathbb{P}_{\text{target}}$ using Eq. equation 15.
6: **end while**

Our entire training procedure is outlined in Algorithm 1. In practice, over-sampling the imitator trajectories (e.g., five imitator trajectories for each demonstrator trajectory) provides empirical benefits. We note that in the extreme case that no matching between demonstration trajectory mixtures and imitation trajectories is possible, our optimization becomes equivalent to (maximum likelihood) behavioral cloning Pomerleau (1988).

For right-sided second-order stochastic dominance (i.e., $-\tilde{\mathbf{F}} \succeq_2 -\mathbf{F}$), the roles of the imitator trajectories and demonstrator trajectories in the linear program are reversed, producing a target distribution of:

$$\mathbb{P}_{\text{target}}(\xi) \triangleq \mathbb{I}[\xi = \tilde{\xi}_i] \sum_j \gamma^*_{i,j} + \mathbb{I}[\xi = \xi_j]\epsilon^*_j. \tag{16}$$

4 Experiments

4.1 Environments and evaluation metrics

We conduct experiments on three different environments: `GridWorld` (discrete states and actions); `DrivingSim` (continuous states, discrete actions); and `PointBot` (continuous states and actions). Since right-sided SSD seeks performance guarantees for the right return-seeking tail of the distribution, we explicitly differentiate between two tail expectations by calling them LCVaR (left-sided CVaR, which would just be called CVaR in the literature as the left tail represents 'risk') and RCVaR (right-sided tail expectation). They are defined as:

$$\nu_\alpha(X) = \inf_x \{x \mid \mathbb{P}(X \leq x) \geq \alpha\}; \tag{17}$$

$$\text{LCVaR}_\alpha(X) = \mathbb{E}[X | X \leq \nu_\alpha(X)]; \quad \text{RCVaR}_\alpha(X) = -\text{LCVaR}_\alpha(-X). \tag{18}$$

4.2 GridWorld

We first evaluate the efficacy of our method in a simpler, model-based setting of a 10-by-10 discrete `GridWorld` environment. There are three types of cells in the environment: {`white, gray, red`}. The agent aims to prevent encounters with unsafe `gray` or `red` cells as much as possible while quickly trying to reach the `goal` cell.

We utilize synthetic demonstrations covering three modes: reaching the goal cell disregarding both `gray` or `red` cells, reaching the goal while avoiding the `red` cells, and reaching the goal while avoiding both `red` and `white` cells. A three-dimensional reward feature vector, accumulated across time-steps, helps characterize trajectory quality: $(is_gray, is_red, 1)^T$. The last feature simply counts the episode duration.

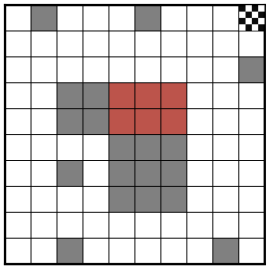

Figure 3: `GridWorld` Environment.

Target probability outputs from the SSD linear program are then utilized as surrogate returns for current (stochastic) policy rollouts, which are then used with Monte Carlo control to iteratively estimate the optimal Q value function for each target probability assignment. A stochastic policy is estimated using softmax over the learned Q values complemented with a small entropy temperature hyperparameter α. Evaluation is made against MaxEntIRL (Ziebart et al., 2008), LPAL (Syed et al., 2008), and FSD (Farajzadeh et al., 2025) policies.

Table 1: `GridWorld` results on demonstration reward feature weights. (Mean±SD for 10 runs).

Policy	Mean	LCVaR$_{0.01}$	LCVaR$_{0.05}$	LCVaR$_{0.1}$	LCVaR$_{0.25}$	RCVaR$_{0.25}$	RCVaR$_{0.1}$	RCVaR$_{0.05}$	RCVaR$_{0.01}$
L-SSD	91.55 ± 0.09	80.58 ± 6.63	79.98 ± 6.72	88.74 ± 1.33	**91.66 ± 0.04**	91.73 ± 0.00	91.73 ± 0.00	91.73 ± 0.00	91.73 ± 0.00
R-SSD	**91.72 ± 0.05**	**84.82 ± 2.98**	**85.11 ± 1.98**	**90.44 ± 0.58**	91.31 ± 0.28	**91.88 ± 0.01**	**91.93 ± 0.0**	91.93 ± 0.00	91.94 ± 0.01
FSD	68.95 ± 0.30	51.62 ± 6.79	65.79 ± 3.21	69.21 ± 0.09	69.19 ± 0.13	69.744 ± 0.24	69.75 ± 0.23	73.42 ± 2.56	76.57 ± 2.91
MaxEntIRL	69.31 ± 0.13	67.62 ± 1.06	65.88 ± 2.27	68.09 ± 1.102	69.21 ± 0.11	69.70 ± 0.22	69.46 ± 0.07	70.39 ± 0.95	70.85 ± 0.97
LPAL	22.28 ± 0.32	-126 ± 3.10	-80.02 ± 1.90	-56.89 ± 2.07	-26.69 ± 1.23	52.53 ± 0.12	55.68 ± 0.12	57.50 ± 0.26	62.99 ± 0.57
Demos	81.70	55.40	61.00	62.72	65.18	91.10	91.85	**92.01**	**92.08**

As shown in Table 1, both L-SSD and R-SSD provide comparatively strong performance across all metrics. MaxEntIRL suffers from collapse to a single mode of demonstrator behaviors, decreasing performance on this averaged metric, while LPAL mixes the different modes of behavior together, leading to even worse performance.

It is interesting to note that LCVaR displays higher variance for almost every threshold compared to RCVaR. We hypothesize that this happens due to episodes in the right-tail of rewards terminating early and thereby exhibiting less variance due to non-activity, as opposed to episodes in the left-tail of rewards that continue acting in the environment, accumulating variance.

4.3 DRIVINGSIM

The `DrivingSim` environment (Majumdar et al., 2017; Greenberg et al., 2022) models a two-lane road where the follower's car has to follow a leader car at a safe distance. The leader can randomly speed up, slow down or change lanes. The state vector in $\mathbb{R}^5$ representing relative distances, relative speeds, current acceleration of the agent, and the angle of the direction it is facing; all computed with a fixed amount of lag to model reaction delay. Each step in the environment represents 0.5 seconds of time where the agents takes one of five discrete actions representing accelerating, braking, maintaining speed or turning left or right. Following the leader closely can give high reward but is also risky because the leader can brake, and following too far behind is safe but does not give high reward. However, our methods do not have access to the reward. Instead, we have access to reward features $\in \mathbb{R}^7$ where elements of the reward feature vector represent desirable aspects of the agent's state: avoiding collision, staying behind leader, not falling too far behind, matching leader's speed, smooth acceleration, following in the same lane, and not veering off-road.

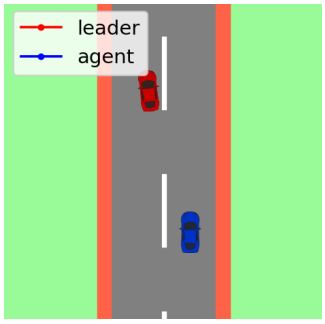

Figure 4: Vehicles in the `DrivingSim` environment.

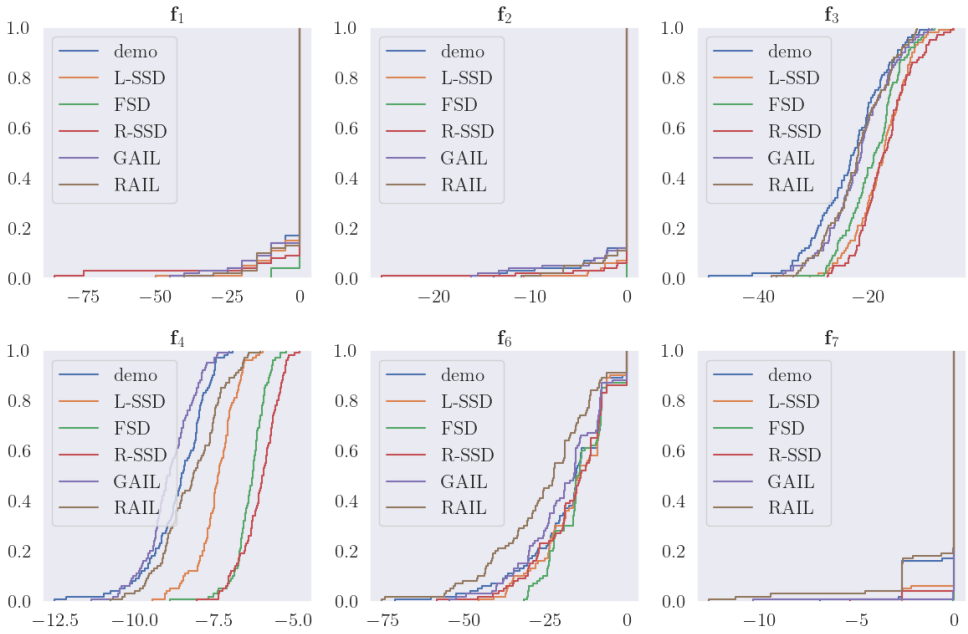

Figure 5: CDFs of different features in demonstrations and under different policies for `DrivingSim`.

The demonstrations are provided by an RL demonstrator. We use a neural network policy model, and use the target probability signal from the LPs with Policy Gradient (Williams, 1992) to train for FSD, L-SSD, and R-SSD. Figure 5 shows the resulting CDFs of each feature, except one that is not shown because the output was constant and same for all policies. As

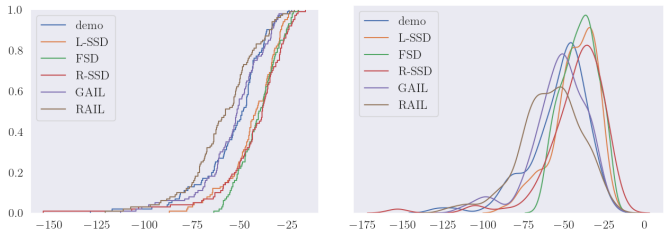

Figure 6: For `DrivingSim`, under different policies the CDFs of true empirical returns (left), and Kernel Density Estimation (right) of population PDFs.

another point of evaluation, we
also compute the true returns using the true reward function the environment uses and plot CDFs in Figure 6, and also the Kernel Density Estimation Plots. Empirical estimates of different risk measures for the true reward are given in Table 2. Note though, that our guarantee is broad and not limited to the environment's reward function so we report per-feature tail measures corresponding to the results in Figure 5 in Appendix C.

Table 2: Mean returns and different tail measures for different policies for `DrivingSim`. (Mean±SD over 20 runs)

Policy	**Mean**	**LCVaR**$_{0.01}$	**LCVaR**$_{0.05}$	**LCVaR**$_{0.1}$	**LCVaR**$_{0.25}$	**RCVaR**$_{0.25}$	**RCVaR**$_{0.1}$	**RCVaR**$_{0.05}$	**RCVaR**$_{0.01}$
Demos	-53.33	-128.56	-103.52	-93.20	-78.58	-34.93	-29.96	-27.54	-24.39
L-SSD	-46.83±1.23	-109.91±21.81	-87.03±8.74	-77.55±5.43	-66.51±3.05	-30.15±1.25	-25.45±1.49	-22.98±1.78	-19.57±2.30
FSD	**-43.37**±1.38	**-89.16**±22.11	**-73.00**±6.36	**-67.59**±4.17	**-59.78**±2.44	-28.87±1.42	-24.70±1.46	-22.41±1.38	-18.51±2.30
R-SSD	-45.51±2.01	-121.94±25.42	-96.79±13.68	-84.89±10.35	-69.20±6.19	**-27.24**±1.64	**-22.67**±1.84	**-20.50**±1.71	**-17.62**±2.05
GAIL	-51.28±1.79	-123.45±21.16	-96.27±9.38	-85.51±6.51	-72.84±3.74	-33.97±1.45	-29.31±1.76	-26.76±1.62	-23.93±1.93
RAIL	-57.05±2.32	-135.15±19.57	-111.69±11.75	-99.63±8.22	-83.73±4.70	-36.06±1.94	-30.34±2.24	-27.48±2.17	-24.04±2.51

Our results reflect some of the inherent trade-offs between right-sided SSD, left-sided SSD, and FSD optimization. R-SSD provides much better right-sided CVaR values than any of the other imitation methods, but much worse left-sided CVaR values as well. This is reflected by the left-skewed distribution estimate in Figure 6. For L-SSD, the performance trade-off is in the other direction, with much better left-sided CVaR values. While L-SSD over the demonstrations is relatively easy to achieve, FSD is more challenging and pushes the FSD policy to also achieve better LCVaR performance than L-SSD. The other baseline methods, while somewhat outperforming demonstrations, are much worse than the stochastic dominance approaches across metrics.

4.4 POINTBOT

For continuous control, we employ `PointBot` (Javed et al., 2021) where continuous 2D velocity actions drive a point-mass agent for navigation towards a fixed goal position. Both the starting position and environment transitions are augmented with noise, while the agent faces resistance in movement due to air friction. Gray unsafe regions cover most of the maze, the agent is expected to reach the goal in as few steps as possible and also avoid those regions.

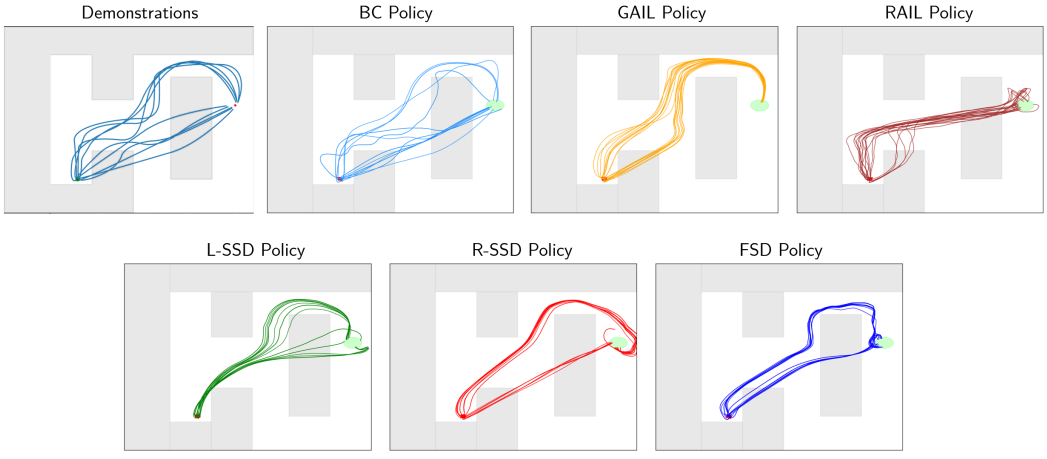

Figure 7: `PointBot` trajectories from the set of demonstrations and those obtained via sampled rollouts of the policies trained using our stochastic dominance-guided policy optimization viz. Left-sided SSD, Right-sided SSD, and FSD, and of the baselines considered for evaluation.

We consider two sets of noisy demonstrations for training: one where the agent is oblivious to gray regions while reaching the goal, and one where it tries to avoid all gray regions. A three-dimensional reward feature vector defined as: $(is_gray,\ not_gray,\ distance_to_goal)^T$, is accumulated across time-steps to characterize trajectory quality.

We then use REINFORCE (Williams, 1992) to optimize a diagonal Gaussian multilayer Perceptron (MLP) policy model, using our slack-augmented SSD verification LP on rollout and demonstration feature vectors to obtain surrogate returns for both demonstration and rollout transition samples.

For evaluation, we choose reward weight vectors whose optimal policies correspond closely to the two demonstration modes, and compute the mean conditional value-at-risk for the return distribution under these weights induced by policies trained using our stochastically dominant imitation methods (L-SSD, R-SSD, and FSD), vs. policies trained with GAIL (Ho and Ermon, 2016), RAIL (Santara et al., 2018), and Behavior Cloning (Pomerleau, 1988), at various risk levels α. Table 3 provides the complete results.

Table 3: Average `PointBot` results on demonstration reward feature weights using diagonal Gaussian MLP policy with hidden layer dimensions (64, 64, 64, 64)

Policy	**Mean**	**LC$\bar{\text{V}}$aR$_{0.01}$**	**LC$\bar{\text{V}}$aR$_{0.05}$**	**LC$\bar{\text{V}}$aR$_{0.1}$**	**LC$\bar{\text{V}}$aR$_{0.25}$**	**RC$\bar{\text{V}}$aR$_{0.25}$**	**RC$\bar{\text{V}}$aR$_{0.1}$**	**RC$\bar{\text{V}}$aR$_{0.05}$**	**RC$\bar{\text{V}}$aR$_{0.01}$**
L-SSD	**1.417**	1.358	**1.368**	**1.375**	**1.385**	1.446	1.451	1.454	1.457
R-SSD	1.415	1.355	1.357	1.358	1.360	**1.460**	**1.463**	**1.464**	**1.465**
FSD	1.413	**1.361**	1.362	1.364	1.369	1.454	1.457	1.459	1.461
RAIL	1.361	1.277	1.296	1.306	1.219	1.436	1.398	1.402	1.407
GAIL	1.389	1.331	1.347	1.350	1.318	1.391	1.427	1.430	1.452
BC	1.385	1.294	1.306	1.319	1.358	1.421	1.455	1.456	1.456
Demos	1.370	1.239	1.239	1.239	1.282	1.442	1.452	1.452	1.452

Though all methods outperform the demonstration distribution in many metrics, we note that L-SSD provides the best performance for almost all left-sided CVaR measures and R-SSD provides the best performance for all right-sided CVaR measures. As shown in Figure 7, R-SSD produces (high velocity) trajectories that sometimes "miss" the absorbing goal state of the environment. In contrast, more reliably reaching the goal is crucial for the left-sided guarantees of L-SSD. FSD provides a more balanced performance across left-sided and right-sided CVaR that lies between the performances of L-SSD and R-SSD.

5 Conclusions, Limitations, and Future Work

In this paper, we have developed algorithms to use multivariate second-order stochastic dominance as a (partial) ordering mechanism for guiding imitation learning to produce behavior that is better than the distribution of demonstrations. This provides important theoretical guarantees enabling imitation learning under the assumption of concave and non-decreasing utility functions. Our algorithm employs distribution-matching and resembles behavioral cloning when the imitator policy is worse than the demonstrator distribution, while seeking to outperform the demonstration as the imitator policy improves. Our experiments show the effectiveness of our approach in realizing its intended objective and how these difference in focus of L-SSD and R-SSD translate to significant differences on distributional performance measures.

There are some limitations of our presented approach: our policy models for continuous domains (conditional Gaussian) may be insufficient for producing multi-modal behavior; our approach relies on large amounts of on-policy trajectory samples; and our policies may be under-optimized if the demonstrations are fairly easy to stochastically dominate. Natural directions for future work are to address these limitations. Incorporating policy diffusion models (Wang et al., 2022) with better multi-modality for continuous domains is of great practical interest. Adding feature-based margins to our optimization is one way to avoid under-optimizing the policy, instead pushing it towards the Pareto frontier of feature trade-offs. Exploring other modifications for improvement beyond domination of the demonstrations is important future work. Additionally, formulations of our optimization for off-policy trajectories are also of significant practical interest so that replay buffers could potentially be leveraged. Next, extensions to higher-order stochastic dominance (e.g., third-order (Post and Kopa, 2017)), which remain stricter than dominance in terms of expected utility, are also of interest. Finally, though it is not a computational bottleneck for our experiments in this paper, exploring more efficient methods to compute or approximate the linear program solution of our approach (Definition 2) is also of interest.

REFERENCES

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pages 1–8, 2004.

Benjamin Armbruster and James Luedtke. Models and formulations for multivariate dominance-constrained stochastic programs. *IIE Transactions*, 47(1):1–14, 2015.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL `http://arxiv.org/abs/1606.01540`.

Yaswanth Chittepu, Ativ Joshi, Rajarshi Bhattacharjee, and Scott Niekum. Safe rlhf beyond expectation: Stochastic dominance for universal spectral risk control. *arXiv preprint arXiv:2603.10938*, 2026.

Hanjun Dai, Yuan Xue, Niao He, Yixin Wang, Na Li, Dale Schuurmans, and Bo Dai. Learning to optimize with stochastic dominance constraints. In *International Conference on Artificial Intelligence and Statistics*, pages 8991–9009. PMLR, 2023.

Darinka Dentcheva and Andrzej Ruszczyński. Portfolio optimization with stochastic dominance constraints. *Journal of Banking & Finance*, 30(2):433–451, 2006.

Darinka Dentcheva and Eli Wolfhagen. Optimization with multivariate stochastic dominance constraints. *SIAM Journal on Optimization*, 25(1):564–588, 2015.

Ali Farajzadeh, Danyal Saeed, Syed M Abbas, Rushit N. Shah, Aadirupa Saha, and Brian D Ziebart. Imitation beyond expectation using pluralistic stochastic dominance. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Ido Greenberg, Yinlam Chow, Mohammad Ghavamzadeh, and Shie Mannor. Efficient risk-averse reinforcement learning. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, 2022.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in Neural Information Processing Systems*, 29, 2016.

Zaynah Javed, Daniel S Brown, Satvik Sharma, Jerry Zhu, Ashwin Balakrishna, Marek Petrik, Anca Dragan, and Ken Goldberg. Policy gradient Bayesian robust optimization for imitation learning. In *International Conference on Machine Learning*, pages 4785–4796. PMLR, 2021.

Milos Kopa and Barbora Petrová. Strong and weak multivariate first-order stochastic dominance. *Available at SSRN 3144058*, 2018.

Timo Kuosmanen. Efficient diversification according to stochastic dominance criteria. *Management Science*, 50(10):1390–1406, 2004.

Jonathan Lacotte, Mohammad Ghavamzadeh, Yinlam Chow, and Marco Pavone. Risk-sensitive generative adversarial imitation learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2154–2163, 2019.

Haim Levy. Stochastic dominance and expected utility: Survey and analysis. *Management science*, 38(4):555–593, 1992.

Anirudha Majumdar, Sumeet Singh, Ajay Mandlekar, and Marco Pavone. Risk-sensitive inverse reinforcement learning via coherent risk models. In *Robotics*, Robotics: Science and Systems. MIT Press Journals, 2017.

Alfred Müller and Dietrich Stoyan. *Comparison methods for stochastic models and risks*. Wiley, 2002.

Andrew Y Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, volume 1, page 2, 2000.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

Thierry Post and Miloš Kopa. Portfolio choice based on third-degree stochastic dominance. *Management Science*, 63(10):3381–3392, 2017.

Gabriel Rioux, Apoorva Nitsure, Mattia Rigotti, Kristjan Greenewald, and Youssef Mroueh. Multivariate stochastic dominance via optimal transport and applications to models benchmarking. *Advances in Neural Information Processing Systems*, 37:39190–39223, 2024.

Anirban Santara, Abhishek Naik, Balaraman Ravindran, Dipankar Das, Dheevatsa Mudigere, Sasikanth Avancha, and Bharat Kaul. RAIL: Risk-Averse Imitation Learning. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2062–2063, 2018.

Moshe Shaked and J George Shanthikumar. *Stochastic orders*. Springer, 2007.

Amita Sharma and Aparna Mehra. Financial analysis based sectoral portfolio optimization under second order stochastic dominance. *Annals of Operations Research*, 256(1):171–197, 2017.

Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, 2007.

Umar Syed, Michael H. Bowling, and Robert E. Schapire. Apprenticeship learning using linear programming. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, ACM International Conference Proceeding Series, pages 1032–1039. ACM, 2008. doi: 10.1145/1390156.1390286. URL https://doi.org/10.1145/1390156.1390286.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Brian D. Ziebart. Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy. 7 2018. doi: 10.1184/R1/6720692.v1. URL https://kilthub.cmu.edu/articles/thesis/Modeling_Purposeful_Adaptive_Behavior_with_the_Principle_of_Maximum_Causal_Entropy/6720692.

Brian D Ziebart, Andrew Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, volume 3, pages 1433–1438, 2008.

A Baseline Implementation Details

First-order stochastic dominance: Since FSD implies SSD equation 8, FSD-based imitation learning (Farajzadeh et al., 2025) is one of our primary baselines for comparison. We implement this as similarly as possible to our SSD implementation using a slack-augmented linear program adapted for FSD (Definition 5).

Definition 5. The slack-augmented first-order stochastic dominance linear program is:

$$\min_{\epsilon \succeq \mathbf{0}, \gamma \succeq \mathbf{0}} \sum_j \epsilon_j \text{ such that: } \gamma_{i,j} \tilde{\mathbf{f}}_j \succeq \gamma_{i,j} \mathbf{f}_i \; \forall (i,j) \in ([N],[M]); \epsilon_j + \sum_{i=1}^{N} \gamma_{i,j} = \frac{1}{M} \; \forall j \in [M]. \quad (19)$$

We similarly define a target distribution using equation 14 and apply the same policy optimization procedure that we describe for SSD in Algorithm 1.

Maximum entropy inverse reinforcement learning (MaxEntIRL): We use the default model-based specification of the algorithm with β set at 10, a learning rate of 0.5, and L2-ball-projected gradient descent for updating the reward weight vector. Soft Value Iteration Ziebart (2018) was used on the obtained reward weights to get a stochastic policy. The entropy temperature parameter was set to 0.1 for this, as well as all other GridWorld experiments.

Linear Programming Apprenticeship Learning (LPAL): We implemented Algorithm 2 of LPAL Syed et al. (2008) directly as a linear program, setting the discount factor $\gamma = 0.99$ for LP feasibility, and using our demonstration trajectory feature averages as empirical "reward-basis" values of the expert. The algorithm returns a state-action occupancy measure which is normalized to obtain the stochastic policy.

Generative Adversarial Imitation Learning (GAIL): For GAIL (Ho and Ermon, 2016) we used the same architecture of policy models as our learners for each environment, and used the implementation from PG-BROIL repository[1] with minor compatibility modifications. GAIL models were trained for 5000 epochs.

Risk-Averse Imitation Learning (RAIL): We use the RAIL (Santara et al., 2018) implementation from the same repository[1], used the same policy models as our learners and as RAIL requires specifying a tail size for which it computes CVaR in the training procedure, we kept $\alpha = 0.1$ for all RAIL experiments. RAIL models were also trained for 5000 epochs.

B Code and Computational Requirements

All code along with required datasets and hyperparameter configurations to reproduce results is provided in an anonymized git repository[2]. All code, datasets, and models will be made publicly available upon publication.

B.1 DrivingSim

The policy model is a neural network with one hidden layer of 32 units. All experiments are initialized with Behavior Cloning for 200 epochs, and then run for a maximum of 2500 epochs with models being evaluated via the LP objective. The demonstration set has 100 trajectories. The environment is based on CeSoR repository[3], with minor modifications (compatibility, memory, etc.) that do not affect the behavior of the environment. All models for `DrivingSim` were trained on a consumer computer with 14-core 2.4GHz maximum clock speed processor, 32GB RAM. GPU acceleration was not used. Wall clock times for training our models was $\sim$3.5 hours for each run. Training RAIL and GAIL took $\sim$12 hours. Demonstrator trajectories were provided by an RL demonstrator which took $\sim$4 hours to train.

[1] `https://github.com/zaynahjaved/pg-broil`

[2] `https://anonymous.4open.science/r/ssdil-iclr-2027/`

[3] `https://github.com/ido90/CeSoR`

B.2 POINTBOT

Experiments for `PointBot` were conducted remotely on a Linux workstation computer with AMD EPYC 9555 CPU (64 cores, up to 4.4 GHz), in a Python 3.14 environment. Multi-layer perceptron policy models with hidden layer dimensions (32, 32), (64, 64), and (64, 64, 64, 64) were trained, with (64, 64, 64, 64) employed for the final results. Message-passing Interface (MPI) was employed to train 8 policy models in parallel, using code from OpenAI Gym (Brockman et al., 2016). The policy with the best training evaluation performance among all processes was ultimately chosen for evaluation on test data. The best performing models were trained for 2000 epochs per worker (or, 16000 epochs in total), requiring an execution time of $\approx$ 1 hour for the (64, 64, 64, 64)-dimensional policy model on the remote server. No GPU compute was employed for `PointBot` experiments.

Baseline experiments involving GAIL (Ho and Ermon, 2016) and RAIL (Santara et al., 2018) were run on the exact same setup, with similarly sized neural networks for policy, value, and discriminator representations. The final model for GAIL was obtained after $\approx$ 2 hours of training, while the model for RAIL took $\approx$ 5 hours.

C ADDITIONAL RESULTS

Table 4: Mean values and different tail measures for each return feature under different policies for DrivingSim. (Mean±SD over 20 runs)

Policy	Feature	Mean	$\text{LCVaR}_{0.01}$	$\text{LCVaR}_{0.05}$	$\text{LCVaR}_{0.1}$	$\text{LCVaR}_{0.25}$	$\text{RCVaR}_{0.25}$	$\text{RCVaR}_{0.1}$	$\text{RCVaR}_{0.05}$	$\text{RCVaR}_{0.01}$
Demo	f_1	-2.60	-50.00	-30.00	-17.50	-2.60	0.00	0.00	0.00	0.00
	f_2	-0.75	-13.83	-10.77	-7.06	-0.75	0.00	0.00	0.00	0.00
	f_3	-23.22	-49.19	-38.96	-36.09	-32.33	-14.86	-12.29	-11.00	-8.93
	f_4	-8.65	-12.55	-11.41	-10.75	-9.99	-7.62	-7.36	-7.22	-6.93
	f_6	-17.63	-71.26	-55.59	-47.82	-36.54	-4.07	0.00	0.00	0.00
	f_7	-0.48	-6.90	-3.56	-3.12	-1.92	0.00	0.00	0.00	0.00
L-SSD	f_1	-2.04 ±0.78	-48.00 ±21.53	-24.58 ±8.44	-14.96 ±9.20	-2.04 ±0.78	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_2	-0.41 ±0.27	-14.91 ±12.28	-6.65 ±4.92	-1.67 ±2.49	-0.41 ±0.27	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_3	-19.03 ±0.54	-33.23 ±3.04	-30.44 ±1.34	-28.66 ±1.09	-25.93 ±0.84	-12.43 ±0.64	-10.11 ±0.87	-8.79 ±1.00	-6.42 ±1.71
	f_4	-7.38 ±0.07	-9.79 ±0.66	-9.13 ±0.28	-8.81 ±0.19	-8.35 ±0.11	-6.52 ±0.10	-6.24 ±0.15	-6.06 ±0.21	-5.74 ±0.30
	f_6	-17.73 ±0.99	-54.53 ±6.13	-45.22 ±3.39	-40.47 ±2.82	-33.38 ±1.99	-4.38 ±1.09	-0.67 ±1.14	**0.00** ±0.00	0.00 ±0.00
	f_7	-0.24 ±0.10	-7.43 ±2.78	-4.08 ±1.46	-2.35 ±0.98	-0.96 ±0.40	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
FSD	f_1	**-0.56** ±0.48	**-35.25** ±28.79	**-6.02** ±10.63	**-0.56** ±0.48	**-0.56** ±0.48	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_2	**-0.07** ±0.10	**-6.22** ±8.61	**-0.07** ±0.10	**-0.07** ±0.10	**-0.07** ±0.10	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_3	-20.59 ±0.53	-35.18 ±2.45	-31.95 ±1.28	-30.21 ±1.05	-27.49 ±0.83	-13.89 ±0.65	-11.53 ±0.80	-10.05 ±1.12	-7.15 ±2.05
	f_4	-6.35 ±0.05	**-7.69** ±0.26	-7.43 ±0.14	-7.25 ±0.12	-6.97 ±0.09	-5.78 ±0.04	-5.58 ±0.05	-5.46 ±0.07	-5.29 ±0.15
	f_6	**-15.77** ±1.14	**-46.23** ±7.07	**-39.00** ±3.62	**-35.27** ±2.59	**-29.47** ±1.81	**-3.77** ±1.25	**-0.25** ±0.55	**0.00** ±0.00	0.00 ±0.00
	f_7	**-0.02** ±0.03	**-1.64** ±2.42	**-0.38** ±0.51	**-0.20** ±0.25	**-0.08** ±0.10	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
R-SSD	f_1	-2.51 ±1.28	-63.00 ±27.63	-35.70 ±18.01	-17.72 ±14.53	-2.51 ±1.28	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_2	-0.42 ±0.28	-15.80 ±10.84	-7.24 ±4.90	-1.60 ±3.18	-0.42 ±0.28	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_3	**-17.27** ±0.50	**-30.45** ±2.22	**-28.14** ±1.75	**-26.50** ±1.36	**-23.95** ±0.96	**-10.60** ±0.75	**-8.15** ±1.01	**-6.71** ±1.23	**-4.79** ±1.44
	f_4	**-5.99** ±0.04	-7.72 ±0.43	**-7.28** ±0.18	**-7.06** ±0.13	**-6.72** ±0.08	**-5.36** ±0.06	**-5.16** ±0.09	**-5.05** ±0.10	**-4.87** ±0.16
	f_6	-18.81 ±1.10	-64.47 ±9.41	-51.91 ±5.98	-45.39 ±4.73	-36.32 ±2.98	-4.22 ±1.22	-0.43 ±1.08	-0.06 ±0.28	0.00 ±0.00
	f_7	-0.51 ±0.17	-10.26 ±2.68	-7.19 ±1.73	-4.80 ±1.49	-2.04 ±0.68	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
GAIL	f_1	-2.71 ±0.65	-54.00 ±22.00	-28.75 ±7.59	-19.42 ±5.64	-2.71 ±0.65	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_2	-0.64 ±0.32	-17.72 ±8.38	-9.81 ±4.29	-4.74 ±4.03	-0.64 ±0.32	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_3	-21.65 ±0.50	-36.47 ±1.96	-33.79 ±1.46	-32.04 ±1.24	-29.09 ±0.85	-14.41 ±0.67	-11.79 ±0.77	-10.13 ±0.84	-7.62 ±1.86
	f_4	-8.91 ±0.09	-11.38 ±0.47	-10.83 ±0.30	-10.50 ±0.24	-9.98 ±0.17	-7.96 ±0.10	-7.66 ±0.14	-7.49 ±0.16	-7.21 ±0.27
	f_6	-17.20 ±1.06	-58.81 ±10.62	-46.71 ±4.00	-41.10 ±2.94	-33.28 ±2.38	-4.11 ±1.08	-0.38 ±0.72	**0.00** ±0.00	0.00 ±0.00
	f_7	-0.17 ±0.10	-9.63 ±5.94	-3.21 ±2.00	-1.66 ±1.02	-0.68 ±0.42	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
RAIL	f_1	-2.34 ±0.61	-47.00 ±20.02	-26.26 ±6.65	-18.68 ±6.10	-2.34 ±0.61	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_2	-0.68 ±0.29	-19.19 ±7.77	-10.85 ±4.41	-4.70 ±4.29	-0.68 ±0.29	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
	f_3	-21.94 ±0.50	-37.27 ±4.12	-33.96 ±2.16	-32.08 ±1.41	-29.23 ±0.87	-14.85 ±0.79	-12.31 ±0.98	-10.75 ±1.22	-7.97 ±2.22
	f_4	-8.24 ±0.10	-11.24 ±0.91	-10.35 ±0.35	-10.01 ±0.22	-9.46 ±0.14	-7.11 ±0.15	-6.71 ±0.23	-6.45 ±0.34	-5.76 ±1.13
	f_6	-22.99 ±1.51	-78.02 ±11.12	-62.92 ±5.41	-55.01 ±4.14	-44.12 ±2.90	-5.81 ±1.52	-1.15 ±1.44	-0.19 ±0.59	0.00 ±0.00
	f_7	-0.85 ±0.24	-15.65 ±4.79	-9.11 ±2.08	-6.45 ±1.38	-3.35 ±0.91	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00

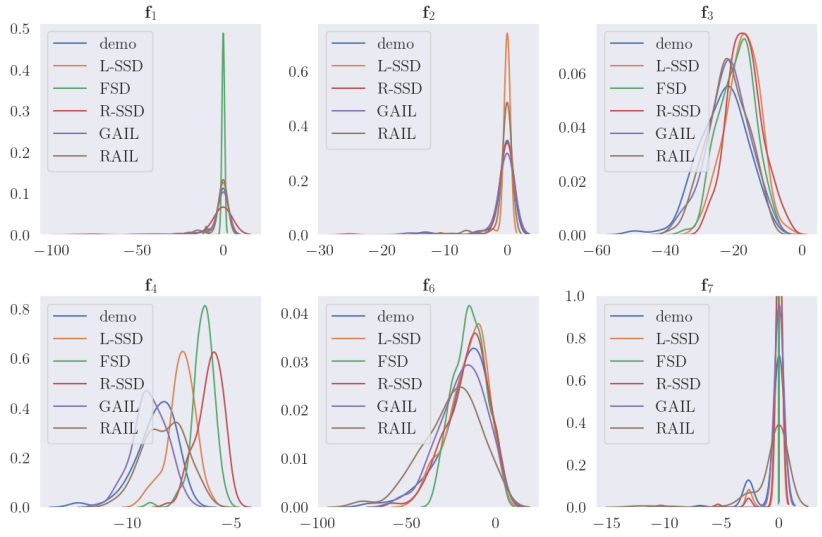

Figure 8: Estimated PDFs of different features in demonstrations and under different policies for `DrivingSim`. For $\mathbf{f}_2$ the FSD policy had no variance in our test run hence the PDF could not be estimated, but can be considered to be an impulse at 0.

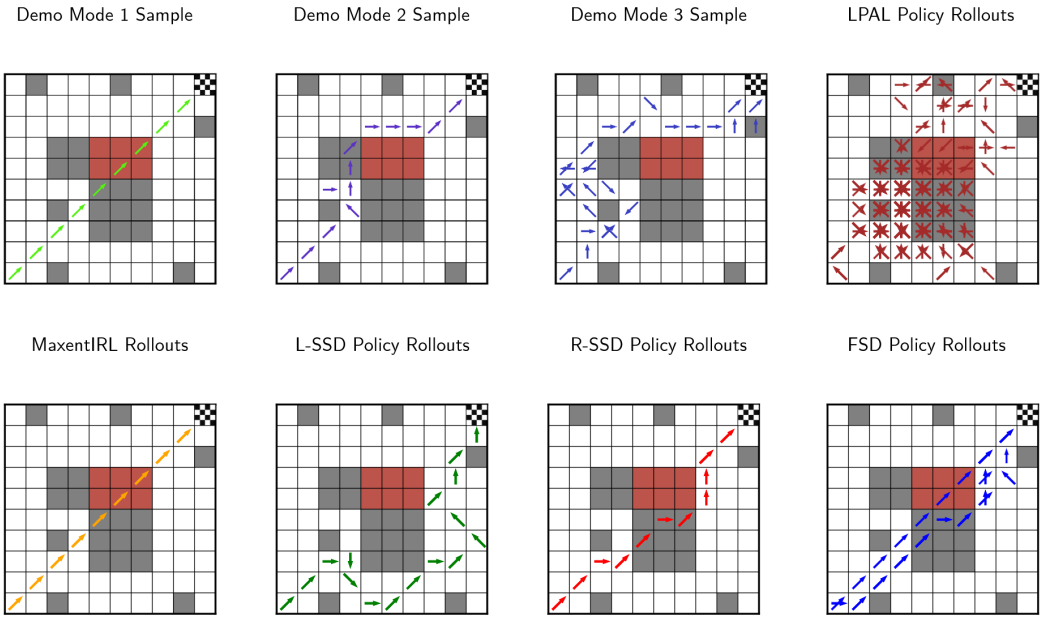

Figure 9: Sample trajectories in the GridWorld environment depicting the three demonstration modes and rollouts of stochastic policies trained to satisfy trajectory-level stochastic dominance-based conditions: L-SSD, R-SSD, and FSD, along with rollouts of other baselines methods considered. As mentioned in the main text, we consider demonstrations that span across three modes of behavior while trying to reach the goal: fully risk-seeking (mode 1) where the agent is unfazed by any hazardous red or gray cells, medium risk-averse (mode 2) where the agent avoids only the red cells, and fully risk-averse (mode 3) where the agent avoids both red and white cells.

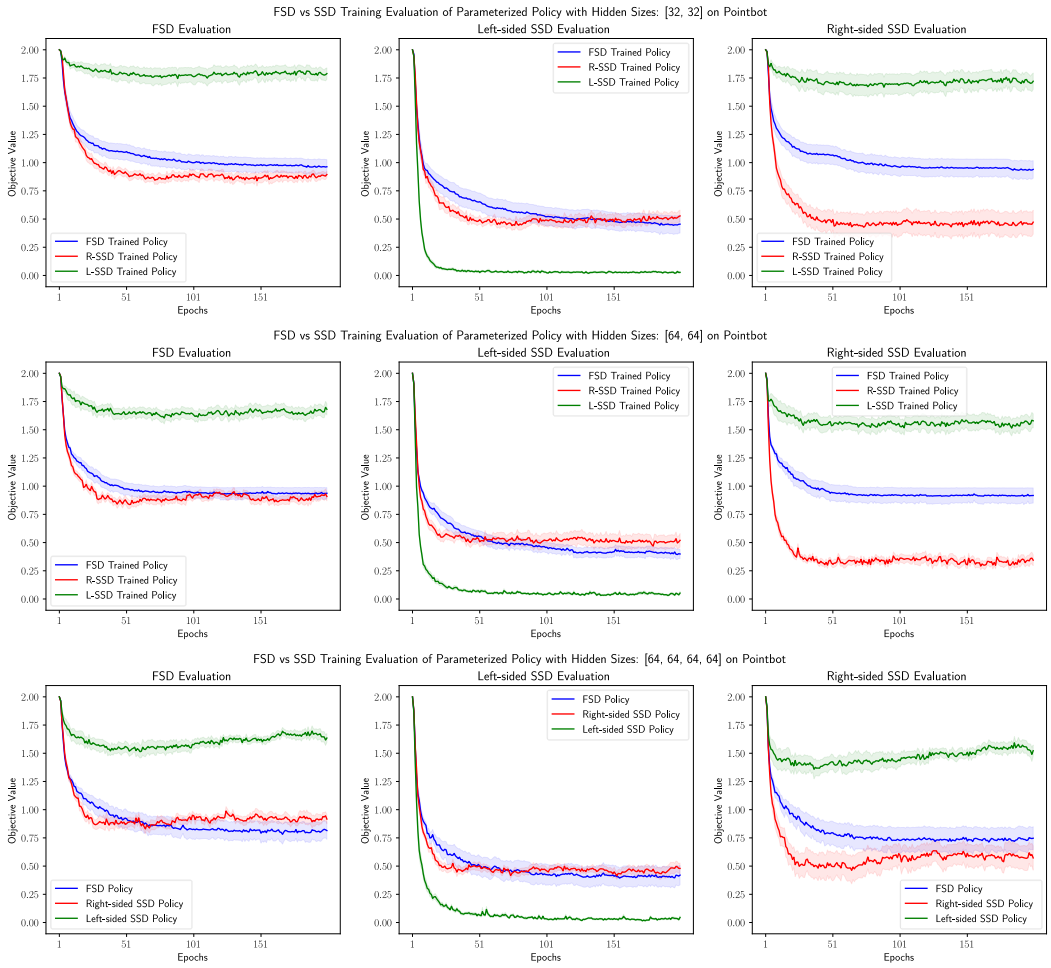

Figure 10: Training curves for L-SSD, R-SSD, and FSD policy optimization methods in the `PointBot` environment evaluated for L-SSD, R-SSD, and FSD violation against the training set for three different policy hidden layer sizes: (32, 32), (64, 64), and (64, 64, 64, 64). As policy model capacity decreases, i.e. the policy network gets shallower, it is apparent that the more relaxed distributional guarantee of second-order stochastic dominance remains achievable for robust imitation beyond expectation matching.

D SSD Linear Program Running time Experimental Analysis

In this section we evaluate how the SSD linear program scales with respect to number of rollout trajectories (N), demonstration trajectories (M) and features (K). All reported times are wall clock times. We conduct oour experiments using CVXPY with a MOSEK interior-point solver.

Tables 5 and 6 give left- and right-sided SSD linear program running times for different rollout, demonstration and feature set sizes in GridWorld. All GridWorld experiments were run on a server with AMD Epyc 9555 CPU.

Tables 7 and 8 give right-sided SSD training and evaluation linear program running times in the DrivingSim environment. All DrivingSim experiments were run on an Intel Core i9-13900H CPU. As the environment has 7 features, for $K > 7$ random features are added in addition to the real features to test LP performance. And as we had a total of 360 demonstrations available, for M beyond 360, random demonstrations are added. Figure 11 plots these.

Our LP is based on work by Armbruster and Luedtke (2015) with NM variables and $M + NK$ constraints. Results in Table 1 of their paper shows near-linear scaling with K. LP theory would suggest $O((NM)^3)$ or $O((NM)^{3.5})$ based on the number of variables, but their results (with $M = N$) show closer to the square root of this rate. The trends in our results are very similar to those reported by them.

Table 5: Stochastic dominance optimization program runtimes (Mean $\pm$ SD) in GridWorld environment for a sample run with varying counts of random policy rollouts N against demonstrations M, with fixed feature dimension $K = 3$.

N	M	K	L-SSD LP Runtime (s)	R-SSD LP Runtime (s)
10	10	3	$0.001 \pm 9 \times 10^{-5}$	$0.001 \pm 9 \times 10^{-5}$
50	50	3	$0.003 \pm 5 \times 10^{-5}$	$0.003 \pm 4 \times 10^{-5}$
100	100	3	$0.010 \pm 2 \times 10^{-4}$	$0.010 \pm 3 \times 10^{-4}$
500	500	3	0.349 ± 0.017	0.323 ± 0.010
1000	1000	3	1.953 ± 0.120	2.160 ± 0.190
1500	1500	3	5.940 ± 0.400	5.480 ± 0.490

Table 6: Stochastic dominance optimization program runtimes (Mean $\pm$ SD) for a sample run with 100 random policy rollouts against 100 demonstrations ($N = M = 100$) and varying feature dimensions K for GridWorld.

N	M	K	L-SSD LP Runtime (s)	R-SSD LP Runtime (s)
100	100	10	$0.030 \pm 3 \times 10^{-4}$	$0.025 \pm 3 \times 10^{-4}$
100	100	50	$0.098 \pm 6 \times 10^{-4}$	0.096 ± 0.001
100	100	100	0.179 ± 0.008	0.193 ± 0.002
100	100	500	1.181 ± 0.013	1.360 ± 0.012
100	100	1000	2.725 ± 0.232	2.850 ± 0.019
100	100	1500	4.610 ± 0.044	4.440 ± 0.044

Table 7: LP runtimes for DrivingSim environment for different number of features K (with random features added beyond 7), for fixed number of demonstrations and rollouts. (Mean $\pm$ SD for 30 runs)

N	M	K	R-SSD train LP runtime (s)	R-SSD eval LP runtime (s)
100	100	7	0.06 ± 0.01	0.13 ± 0.02
100	100	12	0.11 ± 0.02	0.16 ± 0.02
100	100	17	0.12 ± 0.01	0.15 ± 0.02
100	100	32	0.19 ± 0.05	0.22 ± 0.04
100	100	107	0.48 ± 0.15	0.51 ± 0.04
100	100	307	1.30 ± 0.17	1.26 ± 0.05
100	100	507	2.17 ± 0.08	2.19 ± 0.05
100	100	707	3.10 ± 0.08	3.10 ± 0.12

Table 8: The effect of number of demonstrations and rollouts for fixed feature set size ($K = 7$) on LP runtime for the DrivingSim environment. (Mean $\pm$ SD for 30 runs)

N	M	K	R-SSD train LP runtime (s)	R-SSD eval LP runtime (s)
50	50	7	0.01 ± 0.00	0.03 ± 0.00
100	100	7	0.06 ± 0.01	0.13 ± 0.02
230	230	7	0.29 ± 0.03	0.78 ± 0.10
360	360	7	0.84 ± 0.19	2.55 ± 0.45
500	500	7	2.85 ± 0.70	6.77 ± 1.18
700	700	7	5.29 ± 0.46	13.51 ± 3.46
900	900	7	10.94 ± 1.08	26.28 ± 4.98

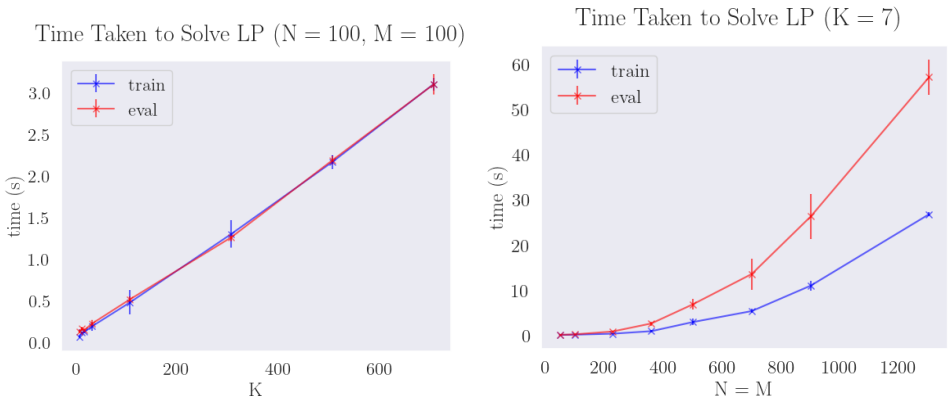

Figure 11: LP runtimes for DrivingSim environment for different K (left) and different N and M (right).