A Dual–SPMA Framework for Convex MDPs

Shervin Khamooshian Ahmed Magd Pegah Aryadoost Danielle Nguyen

School of Computing Science, Simon Fraser University

Project Milestone Presentation

Claim (one sentence)

Fenchel dual ⇒ shaped-reward RL + a fast policy optimizer (SPMA) is a simple, competitive way to solve convex MDPs; we compare to NPG–PD.

Outline

1. Background
2. Policy player
3. Method
4. Evaluation

What is an MDP? (90s)

- **Loop:** see state s, take action a, env. moves, get reward.

Insert cartoon loop

state $\rightarrow$ action $\rightarrow$ next state

reward

What is an MDP? (90s)

- **Loop:** see state s, take action a, env. moves, get reward.
- **Policy:** a rule for picking actions in each state.

Insert cartoon loop

state $\rightarrow$ action $\rightarrow$ next state

reward

What is an MDP? (90s)

- **Loop:** see state s, take action a, env. moves, get reward.
- **Policy:** a rule for picking actions in each state.
- **Goal:** do well on average over time (discounted return).

Insert cartoon loop

state $\rightarrow$ action $\rightarrow$ next state

reward

RL's standard objective

- **English:** maximize the long-run (discounted) return.

RL's standard objective

- **English:** maximize the long-run (discounted) return.
- **Minimal math:** $J(\pi) = (1-\gamma)\,\mathbb{E}\left[\sum_{t\geq 0} \gamma^t\, r(s_t, a_t)\right]$

RL's standard objective

- **English:** maximize the long-run (discounted) return.
- **Minimal math:** $J(\pi) = (1-\gamma)\,\mathbb{E}\left[\sum_{t\geq 0} \gamma^t\, r(s_t, a_t)\right]$
- We'll re-express this using **where the policy spends time**.

Occupancy measure (with example)

- **Idea:** $d^\pi(s, a)$ = discounted time spent in (s, a).

Insert gridworld heatmap
"high" vs "low" d^π

Occupancy measure (with example)

- **Idea:** $d^\pi(s, a)$ = discounted time spent in (s, a).
- Think of a heatmap over grid cells and actions.

Insert gridworld heatmap
"high" vs "low" d^π

Occupancy measure (with example)

- **Idea:** $d^\pi(s, a)$ = discounted time spent in (s, a).
- Think of a heatmap over grid cells and actions.
- It's a probability distribution over (s, a).

Insert gridworld heatmap
"high" vs "low" d^π

RL is linear in occupancy

- **Key identity:** $J(\pi) = \langle r, d^{\pi} \rangle$

RL is linear in occupancy

- **Key identity:** $J(\pi) = \langle r, d^{\pi} \rangle$
- This inner product is the bridge to convex MDPs.

RL is linear in occupancy

- **Key identity:** $J(\pi) = \langle r, d^\pi \rangle$
- This inner product is the bridge to convex MDPs.
- But not all goals are linear in $d^\pi \Rightarrow$ need a more general objective.

Why convex MDPs? & the saddle idea

- Some goals: constraints, imitation, exploration $\Rightarrow$ minimize convex $f(d^\pi)$.

Insert 2-player cartoon:
cost player picks $y \Rightarrow$ policy learns under r_y

Why convex MDPs? & the saddle idea

- Some goals: constraints, imitation, exploration $\Rightarrow$ minimize convex $f(d^\pi)$.
- **Fenchel trick:** $f(d) = \max_y \{\langle y, d\rangle - f^*(y)\}$

Insert 2-player cartoon:
cost player picks $y \Rightarrow$ policy learns under r_y

Why convex MDPs? & the saddle idea

- Some goals: constraints, imitation, exploration $\Rightarrow$ minimize convex $f(d^\pi)$.
- **Fenchel trick:** $f(d) = \max_y \{\langle y, d\rangle - f^*(y)\}$
- **Saddle:** $\min_\pi \max_y \ \langle y, d^\pi\rangle - f^*(y)$

Insert 2-player cartoon:
cost player picks $y \Rightarrow$ policy learns under r_y

Why convex MDPs? & the saddle idea

- Some goals: constraints, imitation, exploration $\Rightarrow$ minimize convex $f(d^\pi)$.
- **Fenchel trick:** $f(d) = \max_y \{\langle y, d\rangle - f^*(y)\}$
- **Saddle:** $\min_\pi \max_y \langle y, d^\pi\rangle - f^*(y)$
- For fixed y, policy step is RL with **shaped reward** $r_y = -y$ (or $-\phi^\top y$).

Insert 2-player cartoon:
cost player picks $y \Rightarrow$ policy learns under r_y

SPMA in 90 seconds: what & why

- Mirror ascent in *logit* space with log-sum-exp mirror map.

Insert small panel:
update equation + "classification projection" box

SPMA in 90 seconds: what & why

- Mirror ascent in *logit* space with log-sum-exp mirror map.
- **Tabular update:** $\pi_{t+1}(a|s) = \pi_t(a|s)\big(1 + \eta\, A_{\pi_t}(s,a)\big)$

Insert small panel:
update equation + "classification projection" box

SPMA in 90 seconds: what & why

- Mirror ascent in *logit* space with log-sum-exp mirror map.
- **Tabular update:** $\pi_{t+1}(a|s) = \pi_t(a|s)\big(1 + \eta\, A_{\pi_t}(s,a)\big)$
- **Why SPMA?** Fast tabular convergence; no per-state renormalization; FA via convex softmax classification.

Insert small panel:
update equation + "classification projection" box

Our method: Dual–SPMA loop (overview)

- **Outer** (dual): OMD/FTL on y using $\hat{d}^{\pi}$ or $\widehat{\mathbb{E}}[\phi]$.

Insert Dual–SPMA loop diagram
(colored: cost vs policy)

Pseudocode (stub):

1. init π_1, y_1; **for** $k = 1..K$:
   (a) policy: SPMA on r_{y_k} for K_{in} steps $\to \pi_{k+1}$;
   (b) estimate $\hat{d}^{\pi_{k+1}}$ / $\widehat{\mathbb{E}}[\phi]$;
   (c) dual: $y_{k+1} \leftarrow \mathrm{OMD/FTL}(y_k, \hat{d}^{\pi_{k+1}} - \nabla f^*(y_k))$;

Our method: Dual–SPMA loop (overview)

- **Outer** (dual): OMD/FTL on y using $\hat{d}^{\pi}$ or $\widehat{\mathbb{E}}[\phi]$.
- **Inner** (policy): run K_{in} SPMA steps on r_y.

Insert Dual–SPMA loop diagram
(colored: cost vs policy)

Pseudocode (stub):

1. init π_1, y_1; **for** $k = 1..K$:
   (a) policy: SPMA on r_{y_k} for K_{in} steps $\to \pi_{k+1}$;
   (b) estimate $\hat{d}^{\pi_{k+1}}$ / $\widehat{\mathbb{E}}[\phi]$;
   (c) dual: $y_{k+1} \leftarrow \mathrm{OMD/FTL}(y_k, \hat{d}^{\pi_{k+1}} - \nabla f^*(y_k))$;

Our method: Dual–SPMA loop (overview)

- **Outer** (dual): OMD/FTL on y using $\hat{d}^{\pi}$ or $\widehat{\mathbb{E}}[\phi]$.
- **Inner** (policy): run K_{in} SPMA steps on r_y.
- **Estimator:** discounted occupancy or feature expectations.

Insert Dual–SPMA loop diagram
(colored: cost vs policy)

Pseudocode (stub):

1. init π_1, y_1; **for** $k = 1..K$:
   (a) policy: SPMA on r_{y_k} for K_{in} steps $\to \pi_{k+1}$;
   (b) estimate $\hat{d}^{\pi_{k+1}}$ / $\widehat{\mathbb{E}}[\phi]$;
   (c) dual: $y_{k+1} \leftarrow \mathrm{OMD/FTL}(y_k, \hat{d}^{\pi_{k+1}} - \nabla f^*(y_k))$;

Experiments: environments, metrics, ablations

Environments

- Tabular (grid/chain) with constraints or imitation

Metrics

- Saddle $L(\pi, y)$ (when f^* is known)

Baselines

Ablations: OMD vs FTL, inner SPMA steps K_{in}, step-size η, tabular vs FA.

Experiments: environments, metrics, ablations

Environments

- Tabular (grid/chain) with constraints or imitation
- Linear features (FA) versions

Baselines

Metrics

- Saddle $L(\pi, y)$ (when f^* is known)
- Constraint violation; return under r_y

Ablations: OMD vs FTL, inner SPMA steps K_{in}, step-size η, tabular vs FA.

Experiments: environments, metrics, ablations

Environments

- Tabular (grid/chain) with constraints or imitation
- Linear features (FA) versions

Baselines

- NPG–PD (CMDP)

Metrics

- Saddle $L(\pi, y)$ (when f^* is known)
- Constraint violation; return under r_y
- $\|d^\pi\|_1$ or $\|\mathbb{E}[\phi]\|$; wall-clock/sample efficiency

Ablations: OMD vs FTL, inner SPMA steps K_{in}, step-size η, tabular vs FA.

Results: convergence & constraint behavior

Insert plot: $L(\pi, y)$ vs iterations (Dual–SPMA vs NPG–PD)

Results: convergence & constraint behavior

Insert plot: $L(\pi, y)$ vs iterations (Dual–SPMA vs NPG–PD)

Insert plot: Constraint violation vs iterations

Baseline: NPG–PD in one slide

- Lagrangian $L(\pi, \lambda) = V_r^{\pi}(\rho) + \lambda(V_g^{\pi}(\rho) - b)$.

Baseline: NPG–PD in one slide

- Lagrangian $L(\pi, \lambda) = V_r^{\pi}(\rho) + \lambda(V_g^{\pi}(\rho) - b)$.
- Updates: natural PG for π; projected subgradient for λ.

Baseline: NPG–PD in one slide

- Lagrangian $L(\pi, \lambda) = V_r^\pi(\rho) + \lambda(V_g^\pi(\rho) - b)$.
- Updates: natural PG for π; projected subgradient for λ.
- Guarantees: $O(1/\sqrt{T})$ averaged gap & violation (dimension-free).

Dual–SPMA vs NPG–PD

Insert overlayed or side-by-side plot

Takeaways

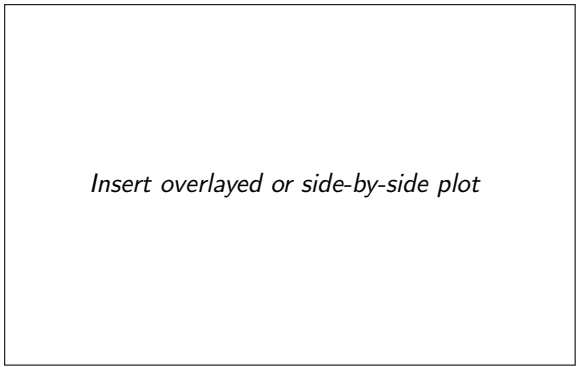
Dual–SPMA vs NPG–PD

Insert overlayed or side-by-side plot

Takeaways

- Convergence speed (where SPMA helps)

Dual–SPMA vs NPG–PD



Takeaways

- Convergence speed (where SPMA helps)
- Constraint control (where NPG–PD excels)

Dual–SPMA vs NPG–PD

Insert overlayed or side-by-side plot

Takeaways

- Convergence speed (where SPMA helps)
- Constraint control (where NPG–PD excels)
- Compute / simplicity

Takeaways & limitations

- **Recipe:** convex MDP $\Rightarrow$ shaped reward + SPMA.

Takeaways & limitations

- **Recipe:** convex MDP $\Rightarrow$ shaped reward + SPMA.
- **Competitive** with NPG–PD on our tasks.

Takeaways & limitations

- **Recipe:** convex MDP $\Rightarrow$ shaped reward + SPMA.
- **Competitive** with NPG–PD on our tasks.
- **Limits:** estimator variance, dual tuning, inner/outer coupling.

Q&A

Questions?

Occupancy polytope (flow constraints)

Discounted case

For all s: $\sum_a d(s,a) = (1-\gamma)\rho(s) + \gamma \sum_{s',a'} P(s|s',a')\, d(s',a'), \qquad d \geq 0.$

From convex objective to shaped reward (derivation)

$$\min_{d \in K} f(d) = \min_{d \in K} \max_{y} \{\langle y, d \rangle - f^*(y)\}$$
$$= \min_{\pi} \max_{y} \{\langle y, d^\pi \rangle - f^*(y)\}$$

Fixed y: policy step is standard RL with shaped reward $r_y = -y$ (or $-\phi^\top y$).

SPMA with features = convex softmax classification

Projection objective (per iteration)

$$\theta_{t+1} = \arg\min_{\theta} \sum_{s} d^{\pi_t}(s)\ \mathrm{KL}\big(\pi_{t+1/2}(\cdot|s) \,\|\, \pi_\theta(\cdot|s)\big)$$

Algorithm sketch (outer–inner)

1. Initialize y_1, policy π_1.
2. For $k = 1, \dots, K$:
   1. Policy step: run SPMA on r_{y_k} for K_{in} steps $\Rightarrow \pi_{k+1}$.
   2. Estimate $\hat{d}^{\pi_{k+1}}$ (or $\widehat{\mathbb{E}}[\phi]$).
   3. Dual step: $y_{k+1} \leftarrow \mathrm{OMD/FTL}\big(y_k, \hat{d}^{\pi_{k+1}} - \nabla f^*(y_k)\big)$.

References I