Learning Decision-Sufficient Representations for Linear Optimization

Yuhan Ye
Joint work with Saurabh Amin and Asuman Ozdaglar

Massachusetts Institute of Technology

39th Annual Conference on Learning Theory
COLT 2026

Problem Formulation

- Decision-task: $\min_{x \in \mathcal{X}} c^\top x$, where $\mathcal{X} \subset \mathbb{R}^d$ is a bounded polyhedron.
- Prior information: The cost vector c is unknown. Prior information is modeled through an uncertainty set $c \in \mathcal{C} \subset \mathbb{R}^d$.
- Dataset: Consists of a set of queries $\mathcal{D} = \{q_1, \ldots, q_N\} \subset \mathbb{R}^d$, which provides the observations $c^\top q_1, \ldots, c^\top q_N$.

- Observing or predicting the full cost c can be unnecessary: optimal decisions may depend only on a few decision-relevant directions.

Key Questions.

- Which and how many observations suffice to identify optimal decisions?
- Can we learn such a dataset (queries) from historical data with provable guarantees in polynomial time?

Problem Formulation

- Decision-task: $\min_{x \in \mathcal{X}} c^\top x$, where $\mathcal{X} \subset \mathbb{R}^d$ is a bounded polyhedron.
- Prior information: The cost vector c is unknown. Prior information is modeled through an uncertainty set $c \in \mathcal{C} \subset \mathbb{R}^d$.
- Dataset: Consists of a set of queries $\mathcal{D} = \{q_1, \ldots, q_N\} \subset \mathbb{R}^d$, which provides the observations $c^\top q_1, \ldots, c^\top q_N$.

- Observing or predicting the full cost c can be unnecessary: optimal decisions may depend only on a few decision-relevant directions.

Key Questions.

- Which and how many observations suffice to identify optimal decisions?
- Can we learn such a dataset (queries) from historical data with provable guarantees in polynomial time?

Problem Formulation

- Decision-task: $\min_{x \in \mathcal{X}} c^\top x$, where $\mathcal{X} \subset \mathbb{R}^d$ is a bounded polyhedron.
- Prior information: The cost vector c is unknown. Prior information is modeled through an uncertainty set $c \in \mathcal{C} \subset \mathbb{R}^d$.
- Dataset: Consists of a set of queries $\mathcal{D} = \{q_1, \ldots, q_N\} \subset \mathbb{R}^d$, which provides the observations $c^\top q_1, \ldots, c^\top q_N$.

- Observing or predicting the full cost c can be unnecessary: optimal decisions may depend only on a few decision-relevant directions.

Key Questions.

- Which and how many observations suffice to identify optimal decisions?
- Can we learn such a dataset (queries) from historical data with provable guarantees in polynomial time?

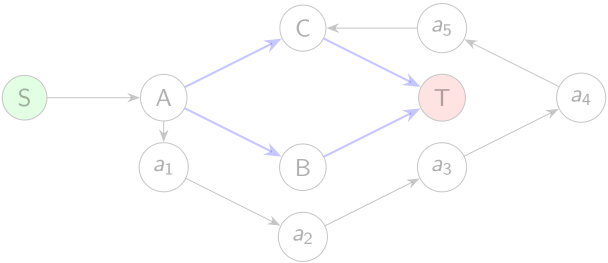
Problem Formulation: Global Sufficient Datasets

Definition (Global Sufficient Decision Dataset (Global SDD))

A set $\mathcal{D} := \{q_1, \ldots, q_N\}$ is a global sufficient decision dataset for uncertainty set $\mathcal{C}$ and decision set $\mathcal{X}$ if there exists a mapping $\hat{x} : \mathbb{R}^N \longrightarrow \mathcal{X}$ such that

$$\forall c \in \mathcal{C}, \quad \hat{x}\left(c^\top q_1, \ldots, c^\top q_N\right) \in \underset{x \in \mathcal{X}}{\arg\min}\ c^\top x.$$

$\underbrace{\{\delta_{AC} + \delta_{CT}, \delta_{AB} + \delta_{BT}\}}_{\hat{\mathcal{D}}}$ is a global SDD for $\mathcal{C} = [1, 5]^{11}$ in the example below.



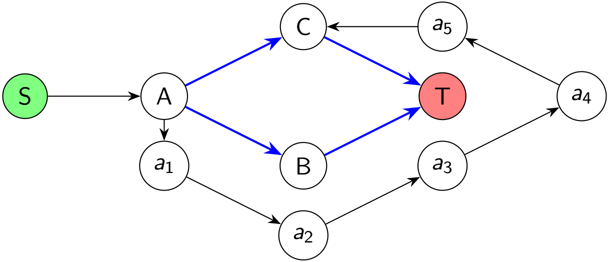
Problem Formulation: Global Sufficient Datasets

Definition (Global Sufficient Decision Dataset (Global SDD))

A set $\mathcal{D} := \{q_1, \ldots, q_N\}$ is a global sufficient decision dataset for uncertainty set $\mathcal{C}$ and decision set $\mathcal{X}$ if there exists a mapping $\hat{x} : \mathbb{R}^N \longrightarrow \mathcal{X}$ such that

$$\forall c \in \mathcal{C}, \quad \hat{x}\left(c^\top q_1, \ldots, c^\top q_N\right) \in \arg\min_{x \in \mathcal{X}} c^\top x.$$

$\underbrace{\{\delta_{AC} + \delta_{CT}, \delta_{AB} + \delta_{BT}\}}_{\hat{\mathcal{D}}}$ is a global SDD for $\mathcal{C} = [1, 5]^{11}$ in the example below.



Linear Programming Basics

- For a cost vector c, the set of optimal solutions is

$$\mathcal{X}^\star(c) := \operatorname*{argmin}_{x \in \mathcal{X}} c^\top x.$$

- For an extreme point $x^\star \in \mathcal{X}^\angle$, its optimality cone is

$$\Lambda(x^\star) := \{c \in \mathbb{R}^d : \ x^\star \in \mathcal{X}^\star(c)\}.$$

- For an uncertainty set $\mathcal{C}$, the reachable optimal solutions are

$$\mathcal{X}^\star(\mathcal{C}) := \bigcup_{c \in \mathcal{C}} \mathcal{X}^\star(c),$$

- The decision-relevant subspace is

$$W_\star := \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) = \mathrm{span}\{x - x' : x, x' \in \mathcal{X}^\star(\mathcal{C})\}.$$

Characterization of Global SDDs

- A global SDD must preserve all directions along which the optimal decision can change over $\mathcal{C}$.
- For open convex $\mathcal{C}$, the geometric characterization gives

$$\mathcal{D} \text{ is globally sufficient} \quad \Longleftrightarrow \quad W_\star \subseteq \operatorname{span}(\mathcal{D}).$$

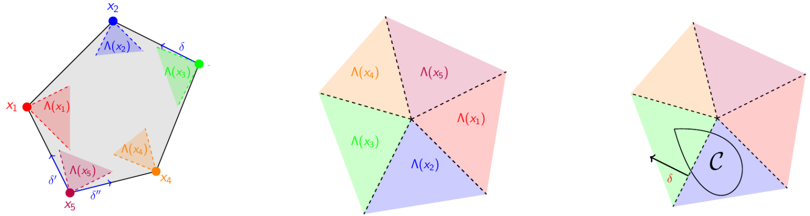
- Hence the minimum number of independent measurements is

$$d^\star = \dim(W_\star),$$

which may be much smaller than the ambient cost dimension d.

[Bennouna et al. 25]

A Geometric Perspective in a Toy Example



- In this example, $\mathcal{C}$ intersects only the two relevant optimality cones $\Lambda(x_2)$ and $\Lambda(x_3)$; hence $\mathcal{X}^\star(\mathcal{C}) = \text{conv}\{x_2, x_3\}$.
- The decision-relevant subspace is one-dimensional:
$$W_\star := \text{dir}(\mathcal{X}^\star(\mathcal{C})) = \text{span}\{x_3 - x_2\} = \text{span}\{\delta\}.$$
- Therefore, by the characterization,
$$\mathcal{D} \text{ is globally sufficient} \quad \Longleftrightarrow \quad \delta \in \text{span}(\mathcal{D}).$$

MIP-Based SDD Construction

When $\mathcal{C}$ is polyhedral and valid bounds are available, the data-selection algorithm of [Bennouna et al. 25] provides an MIP-based implementation in which each search call is a mixed-integer program with

$$\begin{array}{ll} O(d + \#\mathrm{constr}(\mathcal{X})) & \text{variables,} \\ O(d + \#\mathrm{constr}(\mathcal{X}) + \#\mathrm{constr}(\mathcal{C})) & \text{constraints.} \end{array}$$

[Bennouna et al. 25]

Question: Is finding a minimum-size global SDD intrinsically hard?

Finding Minimum-Size Global SDDs Is Hard

- We focus on the polyhedral setting: $\mathcal{C}$ is a polyhedron.
- Recall the exact characterization for open convex $\mathcal{C}$:

$$\mathcal{D} \text{ is globally sufficient} \iff \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) \subseteq \mathrm{span}\, \mathcal{D}, \qquad d^\star = \dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})).$$

- **Worst-case barrier:** Computing $d^\star = \dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C}))$ is NP-hard. [This work]

Proof Idea: Reduction from PISPP-W+, which is NP-complete. [Ley–Merkert 25]

> **PISPP-W+** = *partial inverse shortest path with weight increases*: given a DAG, lengths d, modification costs κ, budget B, and one arc r, decide whether
>
> $$\exists w \geq 0, \quad \kappa^\top w \leq B, \quad \text{s.t. some shortest } s\text{–}t \text{ path under } d + w \text{ uses } r.$$

- **The PISPP-W+ answer becomes a dimension comparison.** Deciding PISPP-W+ can be done by two queries to a "dimension oracle": checking whether

$$\dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) > \dim \mathrm{dir}(\mathcal{X}_r^\star(\mathcal{C})).$$

- Computing $d^\star$ is NP-hard under polynomial-time Turing reductions.

Finding Minimum-Size Global SDDs Is Hard

- We focus on the polyhedral setting: $\mathcal{C}$ is a polyhedron.
- Recall the exact characterization for open convex $\mathcal{C}$:

$$\mathcal{D} \text{ is globally sufficient} \iff \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) \subseteq \mathrm{span}\, \mathcal{D}, \qquad d^\star = \dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})).$$

- **Worst-case barrier:** Computing the dimension $d^\star$ is NP-hard. [This work]

Proof Idea: Reduction from PISPP-W+, which is NP-complete. [Ley–Merkert 25]

PISPP-W+ = *partial inverse shortest path with weight increases*: given a DAG, lengths d, modification costs κ, budget B, and one arc r, decide whether

$$\exists w \geq 0, \quad \kappa^\top w \leq B, \quad \text{s.t. some shortest } s\text{–}t \text{ path under } d + w \text{ uses } r.$$

- **The PISPP-W+ answer becomes a dimension comparison.** Deciding PISPP-W+ can be done by two queries to a "dimension oracle": checking whether

$$\dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) > \dim \mathrm{dir}(\mathcal{X}_r^\star(\mathcal{C})).$$

- Computing $d^\star$ is NP-hard under polynomial-time Turing reductions.

Finding Minimum-Size Global SDDs Is Hard

- We focus on the polyhedral setting: $\mathcal{C}$ is a polyhedron.
- Recall the exact characterization for open convex $\mathcal{C}$:

$$\mathcal{D} \text{ is globally sufficient} \iff \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) \subseteq \operatorname{span} \mathcal{D}, \qquad d^\star = \dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})).$$

- **Worst-case barrier:** Computing the dimension $d^\star$ is NP-hard. [This work]

Proof Idea: Reduction from PISPP-W+, which is NP-complete. [Ley–Merkert 25]

> **PISPP-W+** = *partial inverse shortest path with weight increases*: given a DAG, lengths d, modification costs κ, budget B, and one arc r, decide whether
>
> $$\exists w \geq 0, \quad \kappa^\top w \leq B, \quad \text{s.t. some shortest } s\text{–}t \text{ path under } d + w \text{ uses } r.$$

- **The PISPP-W+ answer becomes a dimension comparison.** Deciding PISPP-W+ can be done by two queries to a "dimension oracle": checking whether

$$\dim \mathrm{dir}(\mathcal{X}^\star(\mathcal{C})) > \dim \mathrm{dir}(\mathcal{X}_r^\star(\mathcal{C})).$$

- Computing $d^\star$ is NP-hard under polynomial-time Turing reductions.

Beyond Worst-Case: Learning from Distributional Data

- The same LP structure is solved repeatedly, while costs vary across instances:

$$c \sim P_c, \qquad c_1, \dots, c_n \overset{\text{i.i.d.}}{\sim} P_c, \qquad \mathrm{supp}(P_c) \subseteq \mathcal{C}.$$

- Goal: learn a small dataset $\widehat{\mathcal{D}}$ that is sufficient for a fresh draw with high probability, rather than for every worst-case $c \in \mathcal{C}$.
- **Traffic example.** Realized travel times change with time, weather, incidents, and congestion. Historical samples may reveal a few critical road observations that determine the optimal decision for most future instances.
- This is a beyond-worst-case, distributional viewpoint: hard regions of $\mathcal{C}$ may be negligible under P_c. [Roughgarden 19/21; Balcan 21]

Pointwise Sufficiency

- We first define a per-instance version of sufficiency.
- Fix a realized cost c with prior information $c \in \mathcal{C}$.
- A query dataset $\mathcal{D}$ gives the measurement observation

$$s_{\mathcal{D}}(c) := (q^\top c)_{q \in \mathcal{D}}.$$

- **Pointwise sufficiency** asks whether these measurements are enough to find and recover some optimal decision $x^\star \in \mathcal{X}^\star(c)$.

$\mathcal{D}$ is a **pointwise sufficient dataset (pointwise SDD) at c** if

$$\exists x^\star \in \mathcal{X}^\star(c) \quad \text{s.t.} \quad F_{\mathcal{D}}(c) \subseteq \Lambda(x^\star).$$

Here $F_{\mathcal{D}}(c) := \{c' \in \mathcal{C} : \; s_{\mathcal{D}}(c') = s_{\mathcal{D}}(c)\}$ is the data-consistent fiber.

Pointwise Sufficiency

- We first define a per-instance version of sufficiency.
- Fix a realized cost c with prior information $c \in \mathcal{C}$.
- A query dataset $\mathcal{D}$ gives the measurement observation

$$s_{\mathcal{D}}(c) := (q^\top c)_{q \in \mathcal{D}}.$$

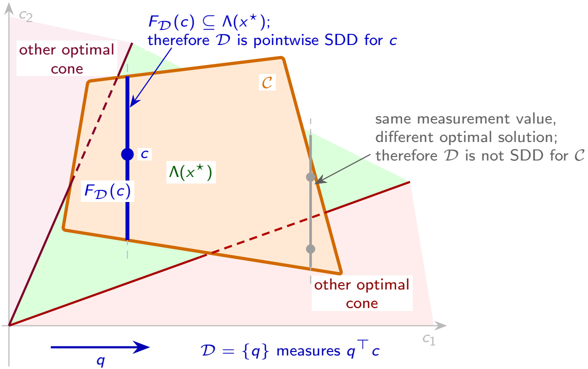
- **Pointwise sufficiency** asks whether these measurements are enough to find and recover some optimal decision $x^\star \in \mathcal{X}^\star(c)$.

$\mathcal{D}$ is a **pointwise sufficient dataset (pointwise SDD) at c** if

$$\exists x^\star \in \mathcal{X}^\star(c) \quad \text{s.t.} \quad F_{\mathcal{D}}(c) \subseteq \Lambda(x^\star).$$

Here $F_{\mathcal{D}}(c) := \{c' \in \mathcal{C} : \ s_{\mathcal{D}}(c') = s_{\mathcal{D}}(c)\}$ is the data-consistent fiber.

Pointwise SDD vs. Global SDD



In the figure, $\mathcal{D} = \{q\}$ is **not** a global SDD for $\mathcal{C}$.

But $\mathcal{D} = \{q\}$ is **pointwise sufficient at** c, because

$$F_{\mathcal{D}}(c) \subseteq \Lambda(x^\star).$$

$$\underbrace{s_{\mathcal{D}}(c) := (q^\top c)_{q\in\mathcal{D}}}_{\text{measurement observation}}, \qquad \underbrace{F_{\mathcal{D}}(c) := \{c' \in \mathcal{C} : \ s_{\mathcal{D}}(c') = s_{\mathcal{D}}(c)\}}_{\text{data-consistent fiber}}.$$

Pointwise Verification

- Verifying pointwise sufficiency at c is equivalent to the cone-containment test

$$\exists x^\star \in \mathcal{X}^\star(c) \quad \text{s.t.} \quad F_{\mathcal{D}}(c) \subseteq \Lambda(x^\star). \tag{CC}$$

- In general, this verification problem is coNP-hard. [This work]
- **Reduction from H-in-V polytope containment.**[†] [Freund–Orlin 85]
- With nondegeneracy, (CC) can be checked by polynomially many convex programs.

[†]An H-polytope is a bounded intersection of finitely many halfspaces, e.g., $P = \{z : Hz \leq h\}$. A V polytope is a convex hull of finite set of points.

Pointwise Verification

- Verifying pointwise sufficiency at c is equivalent to the cone-containment test

$$\exists x^\star \in \mathcal{X}^\star(c) \quad \text{s.t.} \quad F_{\mathcal{D}}(c) \subseteq \Lambda(x^\star). \tag{CC}$$

- In general, this verification problem is coNP-hard. [This work]
- **Reduction from H-in-V polytope containment.**[†] [Freund–Orlin 85]
- With nondegeneracy, (CC) can be checked by polynomially many convex programs.

[†] An H-polytope is a bounded intersection of finitely many halfspaces, e.g., $P = \{z : Hz \leq h\}$. A V polytope is a convex hull of finite set of points.

Constructing Pointwise SDDs

Algorithm 1: Cutting planes for pointwise SDDs. Input: LP data, prior $\mathcal{C}$, realized cost c, initial dataset $\mathcal{D}_{\text{init}}$. **Initialize:** $k = 0$, $\mathcal{D}_0 \leftarrow \mathcal{D}_{\text{init}}$, observe $s_{\mathcal{D}_0}(c)$, and solve once at c to obtain an optimal basis B and decision $x(B)$.

1. **Form the current data-consistent fiber.**

$$F_k := \{c' \in \mathcal{C} : \ q^\top c' = q^\top c \ \forall q \in \mathcal{D}_k\}.$$

2. **Test containment in the fixed optimality cone.** The optimality cone for c is $\Lambda(c) = \{c' : (c')^\top \delta(B, j) \geq 0 \ \forall j \in N\}$. For each $j \in N$, solve the face-intersection subproblem

$$m_j := \min_{c' \in F_k} (c')^\top \delta(B, j). \quad (\mathrm{FI}_j)$$

3. **Stop if certified.** If $m_j \geq 0$ for all $j \in N$, then $F_k \subseteq \Lambda(B)$; return $\mathcal{D}_k$ and $x(B)$.

4. **Otherwise cut and repeat.** Choose a minimizer $c^{\text{out}} \in F_k$ from a violated test; let $j^\star$ be the first facet of $\Lambda(B)$ crossed by $[c, c^{\text{out}}]$. Query $q_{k+1} := \delta(B, j^\star)$, observe $q_{k+1}^\top c$, update $\mathcal{D}_{k+1} := \mathcal{D}_k \cup \{q_{k+1}\}$, set $k \leftarrow k + 1$, and return to Step 1.

Constructing Pointwise SDDs

Algorithm 1: Cutting planes for pointwise SDDs. **Input:** LP data, prior $\mathcal{C}$, realized cost c, initial dataset $\mathcal{D}_{\text{init}}$. **Initialize:** $k = 0$, $\mathcal{D}_0 \leftarrow \mathcal{D}_{\text{init}}$, observe $s_{\mathcal{D}_0}(c)$, and solve once at c to obtain an optimal basis B and decision $x(B)$.

1. **Form the current data-consistent fiber.** $F_k := \{c' \in \mathcal{C} : \; q^\top c' = q^\top c \; \forall q \in \mathcal{D}_k\}$.
2. **Test containment in the fixed optimality cone.** The optimality cone for c is $\Lambda(c) = \{c' : (c')^\top \delta(B, j) \geq 0 \; \forall j \in N\}$.
   For each $j \in N$, solve the face-intersection subproblem
   $$m_j := \min_{c' \in F_k} (c')^\top \delta(B, j). \tag{FI$_j$}$$
3. **Stop if certified.** If $m_j \geq 0$ for all $j \in N$, then $F_k \subseteq \Lambda(B)$; return $\mathcal{D}_k$ and $x(B)$.
4. **Otherwise cut and repeat.** Choose a minimizer $c^{\text{out}} \in F_k$ from a violated test. Let $j^\star$ be the first facet of $\Lambda(B)$ crossed by $[c, c^{\text{out}}]$. Query $q_{k+1} := \delta(B, j^\star)$, observe $q_{k+1}^\top c$, update $\mathcal{D}_{k+1} := \mathcal{D}_k \cup \{q_{k+1}\}$, set $k \leftarrow k+1$, and return to Step 1.

Constructing Pointwise SDDs

Algorithm 1: Cutting planes for pointwise SDDs. **Input:** LP data, prior $\mathcal{C}$, realized cost c, initial dataset $\mathcal{D}_{\mathrm{init}}$. **Initialize:** $k = 0$, $\mathcal{D}_0 \leftarrow \mathcal{D}_{\mathrm{init}}$, observe $s_{\mathcal{D}_0}(c)$, and solve once at c to obtain an optimal basis B and decision $x(B)$.

1. **Form the current data-consistent fiber.** $F_k := \{c' \in \mathcal{C} : \ q^\top c' = q^\top c \ \forall q \in \mathcal{D}_k\}$.
2. **Test containment in the fixed optimality cone.** The optimality cone for c is $\Lambda(c) = \{c' : (c')^\top \delta(B, j) \geq 0 \ \forall j \in N\}$.
   For each $j \in N$, solve the face-intersection subproblem
   $$m_j := \min_{c' \in F_k} (c')^\top \delta(B, j). \tag{FI$_j$}$$
3. **Stop if certified.** If $m_j \geq 0$ for all $j \in N$, then $F_k \subseteq \Lambda(B)$; return $\mathcal{D}_k$ and $x(B)$.
4. **Otherwise cut and repeat.** Choose a minimizer $c^{\mathrm{out}} \in F_k$ from a violated test. Let $j^\star$ be the first facet of $\Lambda(B)$ crossed by $[c, c^{\mathrm{out}}]$. Query $q_{k+1} := \delta(B, j^\star)$, observe $q_{k+1}^\top c$, update $\mathcal{D}_{k+1} := \mathcal{D}_k \cup \{q_{k+1}\}$, set $k \leftarrow k + 1$, and return to Step 1.

Constructing Pointwise SDDs

Algorithm 1: Cutting planes for pointwise SDDs. **Input:** LP data, prior $\mathcal{C}$, realized cost c, initial dataset $\mathcal{D}_{\mathrm{init}}$. **Initialize:** $k = 0$, $\mathcal{D}_0 \leftarrow \mathcal{D}_{\mathrm{init}}$, observe $s_{\mathcal{D}_0}(c)$, and solve once at c to obtain an optimal basis B and decision $x(B)$.

1. **Form the current data-consistent fiber.** $F_k := \{c' \in \mathcal{C} : \; q^\top c' = q^\top c \; \forall q \in \mathcal{D}_k\}$.
2. **Test containment in the fixed optimality cone.** The optimality cone for c is $\Lambda(c) = \{c' : (c')^\top \delta(B, j) \geq 0 \; \forall j \in N\}$.

   For each $j \in N$, solve the face-intersection subproblem

   $$m_j := \min_{c' \in F_k} (c')^\top \delta(B, j). \tag{FI$_j$}$$

3. **Stop if certified.** If $m_j \geq 0$ for all $j \in N$, then $F_k \subseteq \Lambda(B)$; return $\mathcal{D}_k$ and $x(B)$.
4. **Otherwise cut and repeat.** Choose a minimizer $c^{\mathrm{out}} \in F_k$ from a violated test. Let $j^\star$ be the first facet of $\Lambda(B)$ crossed by $[c, c^{\mathrm{out}}]$. Query $q_{k+1} := \delta(B, j^\star)$, observe $q_{k+1}^\top c$, update $\mathcal{D}_{k+1} := \mathcal{D}_k \cup \{q_{k+1}\}$, set $k \leftarrow k + 1$, and return to Step 1.

Guarantee. Under LP nondegeneracy, if $\mathcal{C}$ is an H-polytope or ellipsoid, Algorithm 1 runs in polynomial time and adds at most $d^\star$ queries that form a pointwise SDD at c.

Learning SDDs over Samples

Define the 0–1 sufficiency loss and risk:

$$\ell(\mathcal{D}, c) := \mathbf{1}\{\mathcal{D} \text{ fails at } c\}, \qquad R(\mathcal{D}) := \Pr_{c \sim P_c}[\ell(\mathcal{D}, c) = 1].$$

Algorithm 2: Learning sufficient decision datasets over samples.
Input: prior $\mathcal{C}$, LP data, samples $c_1, \ldots, c_n$.
Initialize: $\mathcal{D}_0 = \emptyset$, hard set $T = \emptyset$.

For $i = 1, \ldots, n$**:**

1. Run Algorithm 1 on c_i, warm-started with $\mathcal{D}_{i-1}$.
2. Let the returned dataset be $\mathcal{D}_i$.
3. If $\mathcal{D}_i \neq \mathcal{D}_{i-1}$, mark i hard: $T \leftarrow T \cup \{i\}$.

Return: $\mathcal{D}_n$ and the hard subsequence indexed by T.

- Only **hard samples** (elements of T) add new decision-relevant directions.

Stable Sample-Compression Scheme

Algorithm 2 induces a realizable stable compression scheme. [Hanneke–Kontorovich 21]

Three key properties

- **Realizability:** $\ell(\mathcal{D}_n, c_i) = 0$ for every training sample $i = 1, \ldots, n$.
- **Stability:** $\mathcal{D}_n$ is fully determined by the hard subsequence $(c_i)_{i \in T}$.
- **Small compression size:** $|T| \leq |\mathcal{D}_n| \leq d^\star$.

Theorem (Certificate via stable compression)

For any $\delta \in (0,1)$, with probability at least $1-\delta$ over $c_1, \ldots, c_n$, Algorithm 2 returns $(\mathcal{D}_n, T)$ satisfying

$$R(\mathcal{D}_n) \leq \frac{4}{n}\left(6|T| + \ln\frac{e}{\delta}\right) \leq \frac{4}{n}\left(6d^\star + \ln\frac{e}{\delta}\right).$$

[This work]

Distribution-Free Learning Certificate

Theorem (Certificate via stable compression)

For any $\delta \in (0,1)$, with probability at least $1-\delta$ over $c_1, \ldots, c_n$, Algorithm 2 returns $(\mathcal{D}_n, T)$ satisfying

$$R(\mathcal{D}_n) \leq \frac{4}{n}\left(6|T| + \ln\frac{e}{\delta}\right) \leq \frac{4}{n}\left(6d^\star + \ln\frac{e}{\delta}\right).$$

[Hanneke–Kontorovich 21; This work]

- Without realizability, a $\widetilde{O}(1/\sqrt{n})$ bound is the standard deviation scale.

Data-driven algorithm design. Prof. Nina Balcan and her group

[COLT 17; ICML 18; FOCS 18; STOC 21/JACM 24; NeurIPS 22]

- Data-driven projection for LPs. [Sakaue–Oki 24]
- This line gives uniform-convergence bounds for the expected performance gap on future instances, typically of order $\widetilde{O}(1/\sqrt{n})$.

Distribution-Free Learning Certificate

Theorem (Certificate via stable compression)

For any $\delta \in (0,1)$, with probability at least $1-\delta$ over $c_1, \dots, c_n$, Algorithm 2 returns $(\mathcal{D}_n, T)$ satisfying

$$R(\mathcal{D}_n) \le \frac{4}{n}\left(6|T| + \ln\frac{e}{\delta}\right) \le \frac{4}{n}\left(6d^\star + \ln\frac{e}{\delta}\right).$$

Proof Intuition: [Hanneke–Kontorovich 21; This work]

- Draw $S^+ = (z_1, \dots, z_{n+1})$ and hold out a uniformly random index J. Let $T(S^+)$ be the compression set (hard samples) for S^+, with $|T(S^+)| \le k$. If $J \notin T(S^+)$, stability says removing z_J does not change the learned hypothesis, and the full run has zero loss on z_J. Hence

$$\Pr\{\text{test fails}\} \lesssim \frac{|T(S^+)|}{n+1} \le \frac{k}{n+1}.$$

For our SDD learner, $k = d^\star$, so $R(\mathcal{D}_n) = \widetilde{O}(d^\star/n)$.

Application: Contextual Linear Optimization

- Data are drawn as $(\xi, c) \sim P$: context $\xi \in \mathbb{R}^p$ predicts the downstream LP cost $c \in \mathcal{C} \subseteq \mathbb{R}^d$.
- The **SPO loss** measures downstream regret:

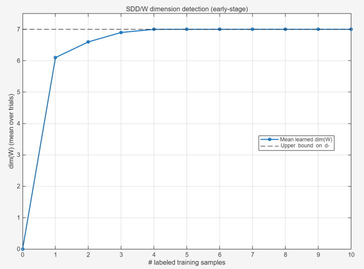
$$\ell_{\mathrm{SPO}}(\hat{c}, c) := c^\top x^\star(\hat{c}) - c^\top x^\star(c).$$

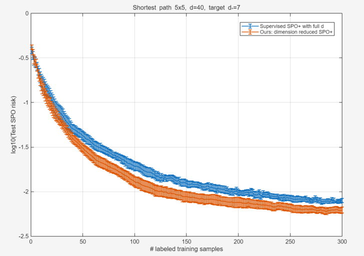
- Learn a $d^\star$-dimensional decision-relevant subspace with Algorithm 2, then train an SPO predictor in $\mathbb{R}^{d^\star}$ and lift back to $\mathbb{R}^d$.

Compressed bound.

$$\widetilde{O}\left(\sqrt{\frac{d^\star p}{n}}\right) \quad \text{instead of} \quad \widetilde{O}\left(\sqrt{\frac{dp}{n}}\right)$$

[El Balghiti et al. 23; This work]





- 5-by-5 shortest-path grid: ambient cost dimension $d = 40$, context dimension $p = 5$, intrinsic dimension $d^\star = 7$.

Open Directions

- **Hardness beyond degeneracy, approximation algorithm.** Do the hardness results persist under nondegeneracy assumptions on $\mathcal{X}$? Can we compute a global SDD whose cardinality is within a constant factor of the minimum?
- **Extension to noisy and robust settings.** With noisy observations, one may need margin conditions controlling how often c lies near cone boundaries. Can robust sample-compression schemes handle corrupted samples or adversarial outliers (e.g., an ε-fraction of arbitrary corruptions)?
- **SDD geometry beyond LPs.** Extend decision-sufficient representations to mixed-integer programs, quadratic programs, or LPs with varying constraints.

Thank you.