FROM APPROXIMATE TO VERTICES: SET DISTRIBUTION MATCHING FOR LINEAR PROGRAMMING CROSSOVER

Anonymous authors
Paper under double-blind review

ABSTRACT

First-order solvers such as PDLP substantially accelerate large-scale linear programming (LP). However, crossover procedures developed for interior-point solutions perform poorly on PDLP outputs, leaving costly vertex and basis recovery for industrial applications such as simplex reoptimization and tableau-based cut generation. We propose Basis-SDM to learn basis recovery from continuous solver states. We identify three challenges overlooked by variable-label supervision: a basis is an unordered set, its columns are interdependent, and degeneracy and multiple optima permit many correct bases. Basis-SDM addresses these challenges through a distribution over complete bases. It assigns order-invariant probabilities to sets; models column interactions through a determinant factor that excludes singular sets; and uses optimality certificates to define target distributions over compatible optimal bases. Shared updates move an initial distribution toward a target under exact KL supervision, followed by conditional-greedy basis selection for simplex cleanup. On 111 unseen LPs, Basis-SDM reduces cleanup iteration sGM from 367.99 to 43.66 relative to native COPT crossover with primal–dual hybrid gradient inputs, and from 518.52 to 4.74 with interior-point inputs. The KKT-adapted LP-GNN baseline obtains 274.40 on the latter, while Basis-SDM requires no cleanup on 64 instances. Controlled ablations demonstrate the benefit of supervising optimal-basis families. On 20 large MILPs, replacing native crossover in a hybrid GPU PDLP solver reduces mean complete solving time by 11.0%. Partial integer-assignment prediction further demonstrates the applicability of distributional supervision to other optimization decisions.

1 INTRODUCTION

PDLP and related first-order solvers accelerate large-scale LP using parallel matrix–vector operations instead of expensive factorizations (Applegate et al., 2021; 2026). Many industrial applications also require a vertex solution and its basis. A basis supports simplex reoptimization of related LPs, including node relaxations in mixed-integer optimization, and tableau-based Gomory mixed-integer cuts (Bixby & Saltzman, 1994; Ben Amor et al., 2006; Gurobi Optimization, 2021; Eifler & Gleixner, 2024). Crossover recovers this structure from continuous LP solutions.

Efficient basis recovery is therefore essential to realizing the benefit of faster continuous optimization. Traditional crossover procedures use active-set identification, primal–dual scaling, and numerical recovery developed for interior-point method (IPM) solutions (Tapia & Zhang, 1991; Cartis & Yan, 2016; Schork & Gondzio, 2020; Ge et al., 2025). They perform poorly on the PDHG outputs in our experiments (Table 3). These projected iterates can be sparse and need not follow the central path (Liu & Lu, 2026). Even accurate continuous solutions can require substantial simplex work to recover a suitable optimal basis.

Learning from solved LPs can improve basis selection. Existing graph-based approaches predict variable statuses and then assemble a basis (Fan et al., 2023; Lu & Yan, 2025). We identify three difficulties with this formulation (Figure 1). First, a basis is an unordered set, so its probability should not depend on selection order. Second, column choices are interdependent: two highly rated columns may be nearly parallel. Third, degeneracy and multiple optima admit several correct bases,

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.

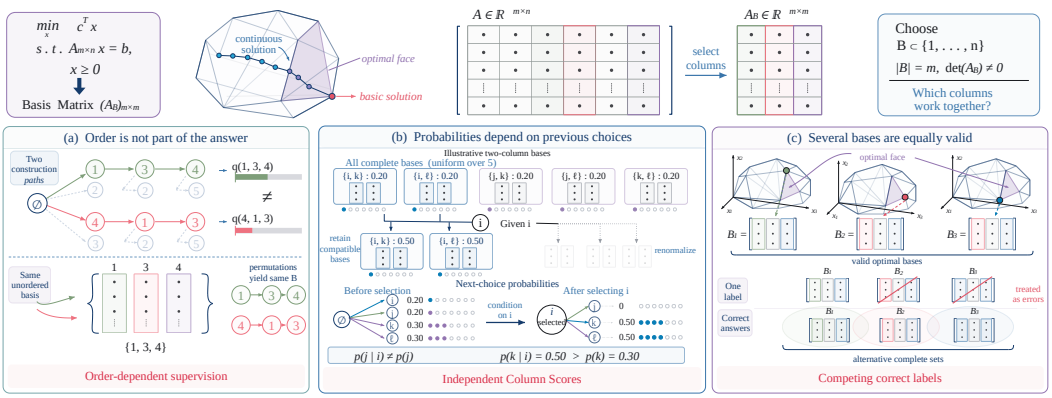

Figure 1: Three structural requirements for learning a basis: invariance to selection order, dependence among column choices, and multiple valid optimal bases. In the five-basis example in (b), p denotes the probability of the next selection.

whereas a single basis label penalizes valid alternatives. Graph representations capture variable–constraint relationships, but variable-status supervision does not directly model a joint distribution over these alternative complete bases.

We propose **Basis-SDM** to learn distributions over complete bases. A determinant factor models column interactions and excludes singular sets. Optimality certificates define target distributions over compatible optimal bases, preserving multiple correct answers. A Global KKT encoder guides shared distribution updates trained with exact KL supervision; terminal decoding supplies a basis for simplex cleanup.

Our main contributions are as follows:

1. We formulate learned crossover as prediction over complete, unordered bases, jointly modeling column choices while enforcing cardinality and rank.
2. We turn optimality certificates into analytical targets over compatible basis families. Exact normalizers and inclusion probabilities give the joint KL and gradient $\nabla_\theta D_{\mathrm{KL}}(q^* \| q_\theta) = 2(\pi(\theta) - \pi^*)$ (Eq. 11), making full-family supervision computable without basis enumeration, repeated sampling, or sampling-induced gradient variance.
3. Experiments on 111 unseen LPs demonstrate reduced cleanup. Ablations test target and output representations, while complete MILP solves, direct prediction, perturbations, and assignment-set generation establish the broader utility of the approach.

2 Related Work

Crossover and numerical basis recovery. Classical crossover combines active-set identification and numerical basis construction (Tapia & Zhang, 1991; Bixby & Saltzman, 1994; Cartis & Yan, 2016). Diagonal weighting uses primal–dual scales (Al-Jeiroudi et al., 2008); scaled CPQR uses column-pivoted QR (Businger & Golub, 1965); weighted exchange-20 and scaled maxvol-20 use maximum-volume exchanges (Schork & Gondzio, 2020; Goreinov et al., 2010). Smart Crossover combines face identification and perturbations (Ge et al., 2025); our Smart → HiGHS baseline uses its recovered basis. HiGHS and COPT provide native crossover (Schork & Gondzio, 2020; Huangfu & Hall, 2018; HiGHS Development Team, 2026; Ge et al., 2022; Cardinal Operations, 2026). For first-order solutions, Liu & Lu (2026) exploit PDHG's spiral dynamics to recover a vertex. We compare against these methods to assess the benefit of learned basis recovery.

Learning to initialize LP solvers. Fan et al. (2023) predict variable statuses with a GNN and adjust them to obtain a basis; we add KKT features for crossover. Lu & Yan (2025) introduce a duality-inspired tripartite graph and basicness and feasibility supervision; their official implementation remains unreleased. Basis-SDM learns complete-basis distributions with optimal-family targets, representing both column dependencies and multiple correct answers.

Set representations and subset distributions. Deep Sets and Set Transformers learn representations of unordered collections, while direct set prediction addresses unordered outputs (Zaheer et al., 2017; Lee et al., 2019; Zhang et al., 2019). DPPs provide tractable subset probabilities, conditioning, and sampling (Lyons, 2003; Kulesza & Taskar, 2012; Gautier et al., 2017). Learning DPPs from observed subsets is well established, including in document summarization (Dupuy & Bach, 2018). Here, the LP matrix defines the determinant factor, and optimality certificates analytically define target distributions over compatible bases. We use DPP tractability to match these derived targets, supervising entire optimal-basis families instead of observed basis labels.

3 Basis-SDM: Set Distribution Matching for Crossover

Given an IPM or PDHG state, Basis-SDM learns a distribution over complete bases and decodes a basis for simplex cleanup.

3.1 From basis prediction to a set space

Consider a feasible, bounded LP

$$\min_{x\geq 0} c^\top x \quad \text{s.t.} \quad Ax = b, \qquad A \in \mathbb{R}^{m\times n}, \quad \operatorname{rank}(A) = m < n. \tag{1}$$

The set of structural bases is

$$\mathcal{B}(A) = \{B \subseteq [n] : |B| = m,\ \det(A_B) \neq 0\}. \tag{2}$$

For a basis B, the associated basic solution satisfies $x_B^B = A_B^{-1}b$ and $x_{\bar{B}}^B = 0$, and its reduced costs are $r^B = c - A^\top A_B^{-\top} c_B$. If both are nonnegative, the basis is optimal. Given the LP and any available solver state, collected in H, we seek a distribution that favors complete bases requiring little additional simplex work to reach optimality.

A distribution over $\mathcal{B}(A)$ represents alternative optimal bases and conditional choices after partial selection.

3.2 An explicit distribution on complete bases

For finite column potentials $\theta \in \mathbb{R}^n$, define

$$q_\theta(B) = \frac{e^{2\sum_{i\in B}\theta_i}\det(A_B)^2}{\mathcal{Z}(\theta)}, \qquad \mathcal{Z}(\theta) = \det\big(A\operatorname{diag}(e^{2\theta})A^\top\big). \tag{3}$$

We define the basis distribution through a determinantal point process (DPP) (Kulesza & Taskar, 2012). Potentials express column preferences, and the determinant evaluates their joint compatibility: full-rank m-column sets have positive probability, while singular sets have zero probability. Thus cardinality and rank are guaranteed; optimization-derived targets guide feasibility. Cauchy–Binet gives the normalizer (Appendix A.2); factor-based evaluation and conditional decoding are described in Appendix A.1.

For IPM states, we initialize $\theta_i^{(0)} = \frac{1}{2}(\log x_i - \log s_i)$. PDHG uses positive proxies for primal values and reduced costs, while direct prediction uses $\theta^{(0)} = 0$ (Appendix C). A common shift of all potentials leaves the distribution unchanged. The inclusion probabilities satisfy

$$\pi_i(\theta) = \mathbb{P}_{q_\theta}(i \in B), \qquad \nabla_\theta \log \mathcal{Z}(\theta) = 2\pi(\theta), \qquad \sum_i \pi_i(\theta) = m. \tag{4}$$

3.3 Obtaining the target distribution

During training, an available primal–dual optimum (x_c^*, y_c^*, s_c^*) defines mandatory and admissible columns:

$$P_c = \{i : x_{c,i}^* > 0\}, \qquad Z_c = \{i : s_{c,i}^* = 0\}, \qquad \mathcal{E}_c = \{B \in \mathcal{B}(A) : P_c \subseteq B \subseteq Z_c\}. \tag{5}$$

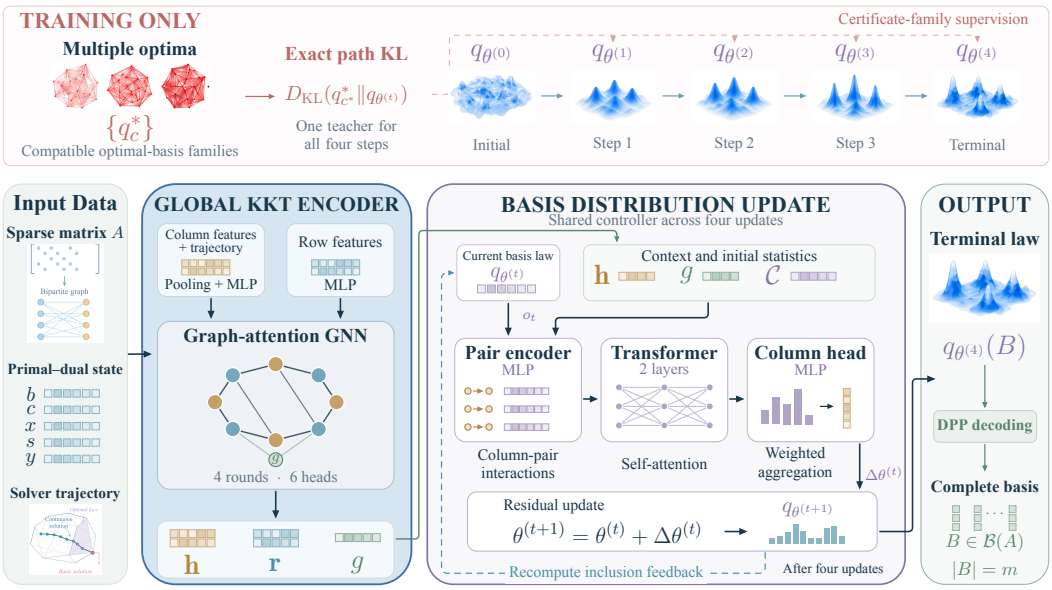

Figure 2: Overview of Basis-SDM. Optimality certificates define target distributions for exact KL supervision (top). A Global KKT encoder represents the solver state, and a shared update module combines this context with distribution feedback to update the column potentials four times (bottom). Conditional-greedy decoding returns a complete basis. The landscapes illustrate distributions schematically; gold, blue, and green denote column, row, and global representations.

The event $\mathcal{E}_c$ describes all bases compatible with the certificate. Including P_c preserves the primal solution, and choosing columns only from Z_c preserves dual feasibility. Thus every basis in this family is optimal. For a nonempty family, we define the target by conditioning the initial distribution:

$$q_c^*(B) = q_{\theta^{(0)}}(B \mid B \in \mathcal{E}_c), \qquad \pi_c^* = \mathbb{E}_{q_c^*} \mathbf{1}_B. \tag{6}$$

The target preserves all compatible completions and their relative initial probabilities. A degenerate solution can supervise several bases, and different optima supply further families. Exact conditioning computes the targets without enumeration; certificates are used only during training.

3.4 Learning the Distribution Transformation

We learn to transform the initial distribution toward one of these target families through four shared updates:

$$q_{\theta^{(0)}} \longmapsto q_{\theta^{(1)}} \longmapsto q_{\theta^{(2)}} \longmapsto q_{\theta^{(3)}} \longmapsto q_{\theta^{(4)}}. \tag{7}$$

Potential updates redistribute probability among complete bases while preserving normalization and support on $\mathcal{B}(A)$ (Figure 2).

Global KKT encoding. An MLP combines column features and trajectory embeddings, and a separate MLP embeds rows. Sparse attention passes information from columns to rows and back along nonzeros of A, followed by global pooling. Four rounds produce column representations $\mathbf{h}$, row representations $\mathbf{r}$, and global context g.

Distribution updates. An MLP combines encoder context and distribution feedback into column-pair tokens. A Transformer models their interactions, and a column-wise MLP predicts potential increments. Initial statistics $\mathcal{C}$ and dynamic feedback o_t are defined in Appendices B.2 and B.4. All four updates share parameters:

$$\Delta\theta^{(t)} = \mathrm{Update}_\phi(\theta^{(t)}, \mathbf{h}, g, \mathcal{C}, o_t, t), \qquad \theta^{(t+1)} = \theta^{(t)} + \Delta\theta^{(t)}. \tag{8}$$

Centering and bounding stabilize the potentials. Row information enters through $\mathbf{h}$ and g; architecture details appear in Appendix B.

Algorithm 1 Basis-SDM training

Require: State H, certified optimal families $\{\mathcal{E}_c\}$, parameters ϕ
Ensure: Loss $\mathcal{L}$ and updated parameters ϕ
1: $(\theta^{(0)}, \mathcal{C}) \leftarrow \mathrm{Prepare}(H)$; $(\mathbf{h}, \mathbf{r}, g) \leftarrow \mathrm{GlobalKKT}_\phi(H)$
2: Form teachers $q_c^* = q_{\theta^{(0)}}(\,\cdot \mid \mathcal{E}_c)$ and their exact statistics
3: **for** $t = 0, 1, 2, 3$ **do**
4: $\quad o_t \leftarrow \mathrm{Observe}(\theta^{(t)}, \mathcal{C})$
5: $\quad \Delta\theta^{(t)} \leftarrow \mathrm{Update}_\phi(\theta^{(t)}, \mathbf{h}, g, \mathcal{C}, o_t, t)$
6: $\quad \theta^{(t+1)} \leftarrow \theta^{(t)} + \Delta\theta^{(t)}$
7: **end for**
8: Evaluate exact KL at $\theta^{(1:4)}$ and select one teacher c^* using (9)
9: Backpropagate $\mathcal{L}$ through the four-step recurrence and update ϕ

3.5 Exact KL supervision along the path

When several target families are available, we select one teacher for the entire path and supervise every intermediate distribution against it:

$$c^* = \arg\min_c \sum_{t=1}^{4} \omega_t D_{\mathrm{KL}}(q_c^* \| q_{\theta^{(t)}}), \qquad \mathcal{L} = \frac{1}{\sqrt{m}} \sum_{t=1}^{4} \omega_t D_{\mathrm{KL}}(q_{c^*}^* \| q_{\theta^{(t)}}), \quad \omega_t = \frac{t}{10}. \tag{9}$$

A common teacher provides a consistent objective while retaining alternative optimal families. Increasing weights emphasize the terminal distribution.

The model and teacher share the same determinant factor, which cancels in their log-density ratio. Let $\mathcal{Z}_c(\theta^{(0)})$ denote the unnormalized initial weight of $\mathcal{E}_c$. The joint KL and its gradient then become

$$D_{\mathrm{KL}}(q_c^* \| q_\theta) = \log \mathcal{Z}(\theta) - \log \mathcal{Z}_c(\theta^{(0)}) + 2(\theta^{(0)} - \theta)^\top \pi_c^*, \tag{10}$$

$$\nabla_\theta D_{\mathrm{KL}}(q_c^* \| q_\theta) = 2\{\pi(\theta) - \pi_c^*\}. \tag{11}$$

The normalizers and inclusion probabilities are computed by matrix factorizations (Appendices A.1 and A.5), reducing supervision over exponentially many bases to matrix operations. This avoids enumerating bases or repeatedly sampling them to estimate the loss and gradient. Unlike sample-based score-function estimation (Tokui & Sato, 2017), the gradient introduces no Monte Carlo variance from basis sampling. Calibrated controller feedback is separate from these exact training moments.

3.6 Generating a complete basis

The terminal distribution $q = q_{\theta^{(4)}}$ is converted into a basis by deterministic conditional-greedy selection. Let $M_q(S) = \mathbb{P}_{B\sim q}(S \subseteq B)$. Starting from the empty set, we select the column with the largest conditional inclusion probability:

$$j^* \in \arg\max_{j \notin S} \mathbb{P}(j \in B \mid S \subseteq B) = \arg\max_{j \notin S} \frac{M_q(S \cup \{j\})}{M_q(S)}.$$

We condition on this choice and repeat until $|S| = m$. Since the probabilities are updated after each selection, every decision accounts for the columns already chosen. The procedure preserves a full-rank completion (Appendix A.6); its steps are detailed in Appendix B.5. Section 4 also establishes exact sampling properties of the distribution.

Algorithm 1 summarizes training. At inference, the model uses only H and returns a basis without access to optimality certificates.

4 Properties of the Set Model and Supervision

We establish valid basis generation, order-independent sampling, and a bound relating training loss to optimal-family mass. These exact-arithmetic results are proved in Appendix A.

Theorem 1 (Valid support and compatible targets). *For full-row-rank A and finite θ, equation (3) is a normalized distribution supported exactly on $\mathcal{B}(A)$. Its probability is invariant to the enumeration of a basis, and the conditional maximization rule (22) returns a member of $\mathcal{B}(A)$. For a nonempty certified family $\mathcal{E}_c$, every supported basis of q_c^* is optimal for the LP.*

Proof. See Appendices A.2, A.3, and A.6.

Proposition 2 (Exact sampling and order independence). *For a size-m basis law q, start with $S = \varnothing$ and sample a new column according to*

$$\mathbb{P}(J = j \mid S) = \frac{M_q(S \cup \{j\})}{(m - |S|)M_q(S)}, \qquad j \notin S. \tag{12}$$

After m steps, the unordered output has law q. Every one of the $m!$ enumerations of a given basis B has probability $q(B)/m!$.

Proof. See Appendix A.7.

Proposition 3 (KL controls optimal-family mass). *For the selected teacher c^* and any $t = 1, \ldots, 4$,*

$$q_{\theta^{(t)}}(\mathcal{E}_{c^*}) \geq \exp\left(-\frac{\sqrt{m}}{\omega_t}\mathcal{L}\right). \tag{13}$$

Moreover, q_c^* belongs to the closure of the unrestricted basis distribution family.

Proof. See Appendices A.8 and A.9.

5 Experiments

Our experiments examine the quality of generated bases, their effect on solving time, and the role of distributional supervision. We evaluate IPM and PDHG crossover, direct prediction, coefficient perturbations, and complete MILP solves, then extend the set formulation to integer assignments.

5.1 Evaluation protocol

We train on 125 original LPs from Netlib and the MISC/STOCHLP subsets of the Mészáros collection, together with augmented and generated instances. Crossover and direct prediction are evaluated on the same 111 previously unseen LPs: 37 small ($n < 1000$), 58 medium ($1000 \leq n \leq 8000$), and 16 large ($n > 8000$). Each input setting uses separately trained parameters (Appendix B.7).

Each constructed basis initializes HiGHS simplex with presolve disabled and feasibility tolerances 10^{-7}. Remaining iterations are summarized by the shifted geometric mean

$$\mathrm{sGM}(k) = \exp\left(\frac{1}{N}\sum_{h=1}^{N}\log(1 + k_h)\right) - 1. \tag{14}$$

Failures receive $10{,}000$ iterations and are also counted separately. Zero cleanup means termination without a further simplex iteration. Imported states, baseline definitions, and native HiGHS/COPT counters appear in Appendix C.

5.2 IPM crossover

Basis-SDM requires no cleanup on 64 of the 111 LPs and succeeds on every instance. Its cleanup sGM is 4.74, compared with 518.52 for native COPT and 274.40 for KKT-adapted LP-GNN (Table 1). Relative to the latter, which follows the variable-status prediction approach of Fan et al. (2023), Basis-SDM reduces sGM by 98.3%. Under the same HiGHS interface, it reduces Smart basis recovery's sGM from 37.06 to 4.74 (87.2%) and achieves the lowest cleanup in every size group.

On the largest 16 LPs, Basis-SDM reduces sGM from Smart's 330.25 to 165.34.

The reduction in simplex work also lowers total crossover time, including prediction and decoding: 0.0359 and 0.3075 seconds for medium and large LPs, respectively, compared with Smart's 0.3387 and 2.0097 (Table 2). Appendix C.3 defines the timing intervals and reports their components.

Table 1: Simplex iterations after crossover from IPM solutions. Results are sGM over 111 LPs. HiGHS excludes crossover pushes; COPT uses its native simplex counter.

Method	Simplex iterations (sGM) ↓				Failures
	Small (37)	Medium (58)	Large (16)	All (111)	
Diagonal weighting	62.54	514.48	5228.91	357.26	14
Weighted exchange-20	23.87	358.52	3895.62	207.08	10
Scaled CPQR	17.00	233.65	3110.51	143.71	0
Scaled maxvol-20	17.14	233.18	3186.87	144.46	0
Smart basis → HiGHS	7.27	54.49	330.25	37.06	0
KKT-adapted LP-GNN	23.13	559.07	5857.35	274.40	20
HiGHS native crossover	100.16	850.20	8015.52	577.20	1
COPT native crossover	92.99	810.37	5380.01	518.52	0
Basis-SDM	**0.07**	**5.62**	**165.34**	**4.74**	0

Baseline definitions and sources: Appendix C.2.

Table 2: Time to recover and optimize a basis from IPM solutions. Geometric means in seconds.

Method	Medium (58)		Large (16)	
	Total	Solver	Total	Solver
Diagonal weighting†	1.5303	0.0249	31.0814	0.1493
Weighted exchange-20†	2.4226	0.0243	60.0921	0.1878
Scaled CPQR	0.5779	0.0281	9.0041	0.4544
Scaled maxvol-20	1.3274	0.0275	31.8963	0.4624
Smart → HiGHS	0.3387	0.0166	2.0097	0.1309
KKT-adapted LP-GNN†	0.3688	0.0181	8.6005	0.1499
HiGHS native crossover†	0.0694	0.0694	0.8895	0.8895
COPT native crossover	0.0438	0.0438	0.5016	0.5016
Basis-SDM	0.0359	0.0129	0.3075	0.1024

Table 3: Simplex iterations after PDHG crossover. sGM and failures on 111 LPs.

Method	All (111)	
	sGM	Fail.
Diagonal weighting	343.12	15
Weighted exchange-20	262.96	15
Scaled CPQR	221.55	0
Scaled maxvol-20	228.53	0
Smart → HiGHS	1102.78	65
Liu–Lu → HiGHS	293.12	17
HiGHS native crossover	496.14	0
COPT native crossover	367.99	0
Basis-SDM	**43.66**	0

Methods as in Table 1. Total includes construction; solver includes setup and execution. †Includes failed attempts. Full PDHG groups: Table 10.

5.3 PDHG CROSSOVER

We next evaluate crossover from PDHG states at different residual accuracies. Every method receives the same primal–dual endpoints, selected as described in Appendix C.4.

Basis-SDM achieves cleanup sGM 43.66, compared with 293.12 for the PDHG-specific method of Liu & Lu (2026), 367.99 for native COPT, and 221.55 for scaled CPQR (Table 3). All 111 runs succeed, including 26 without cleanup. The Liu–Lu comparison uses the same PDHG endpoints and HiGHS cleanup interface, demonstrating the benefit of learned basis recovery for these first-order solutions. The full results by problem size appear in Table 10.

5.4 DIRECT BASIS PREDICTION

To evaluate basis prediction without a solver trajectory, the direct model observes only (A, b, c) and starts from zero potentials. We pass the same predicted column set to HiGHS and COPT. Within each solver, native simplex and predicted-basis initialization use matched settings, and iteration counts include feasibility restoration.

Our model reduces HiGHS iteration sGM from 1123.61 to 217.76 (80.6%) and COPT's from 958.77 to 224.36 (76.6%), with all 111 runs successful in both solvers (Table 4). Solver optimization time falls from 0.0281 to 0.0124 seconds in HiGHS and from 0.0467 to 0.0097 seconds in COPT.

Table 4: Simplex performance with predicted initial bases. Iteration sGM and geometric-mean solving time (seconds) on 111 LPs. Failure penalties are defined in Appendix C.5.

Method	Small	Medium	Large	All	Fail.	Time (s)
HiGHS						
Native simplex	262.66	1647.95	8039.88	1123.61	0	0.0281
Scaled CPQR	138.49	1172.08	9175.71	774.96	1	0.0281
Scaled maxvol-20	142.96	1110.90	8552.25	753.82	1	0.0273
Fan et al. (2023)	181.72	1212.51	7510.80	838.61	11	0.0355
Basis-SDM	**42.55**	**374.58**	**1287.45**	**217.76**	0	**0.0124**
COPT						
Native simplex	250.40	1490.25	4302.29	958.77	0	0.0467
Basis-SDM	**43.53**	**398.29**	**1203.77**	**224.36**	0	**0.0097**

Table 5: Effect of coefficient perturbations on crossover. Cleanup sGM on 13 matched LPs; unperturbed values are 72.77 for Basis-SDM and 833.06 for native HiGHS.

Perturbed data	1%		10%		30%	
	Basis-SDM	HiGHS	**Basis-SDM**	HiGHS	**Basis-SDM**	HiGHS
A	**94.48**	833.49	**74.98**	656.72	**142.81**	797.46
b	**82.11**	781.46	**99.34**	677.65	**113.84**	696.79
c	**111.24**	789.96	**108.91**	714.27	**118.07**	965.62
A, b, c	**98.54**	837.44	**80.38**	874.77	**116.52**	803.53

Table 6: Effect of supervision and output representation on basis recovery. Cleanup sGM on 111 LPs; single-basis supervision and Basis-SDM use IPM states, and independent scores use (A, b, c). Settings appear in Appendix C.7.

Method	Epoch	Small	Medium	Large	All	Fail.
Single-basis supervision	5	8.18	158.97	1288.54	82.36	0
Single-basis supervision	20	8.78	168.89	1839.81	91.48	0
Independent scores + rank repair	20	101.98	892.39	4162.94	541.78	20
Basis-SDM	20	**0.07**	**5.62**	**165.34**	**4.74**	0

5.5 Numerical perturbations

We assess robustness by perturbing A, b, and c separately and jointly at 1%, 10%, and 30%. The trained IPM model remains fixed, and each perturbed LP receives a newly computed IPM trajectory.

Table 5 compares the same 13 LPs with valid inputs under every condition. Basis-SDM reduces cleanup under all twelve perturbations. Across the full set of 315 valid inputs from 30 source LPs, it succeeds on every instance with sGM 77.07, compared with 570.13 for native HiGHS. These results show that the learned basis preferences remain effective when numerical coefficients change; perturbation details are given in Appendix C.6.

5.6 Supervision and output ablations

The target distribution is central to the improvement. With the encoder, updates, and decoder fixed and both models trained for 20 epochs, supervising compatible optimal-basis families instead of one optimal basis reduces cleanup sGM from 91.48 to 4.74 (94.8%; Table 6). Preserving multiple correct answers therefore improves the basis learned under otherwise identical conditions.

With static (A, b, c) inputs, independent scores trained by binary cross-entropy and followed by rank repair yield sGM 541.78 with 20 failures; direct Basis-SDM yields 217.76 with none (Table 4). Appendix C.7 gives the variant definitions and training settings.

Table 7: MILP solving performance with simplex and hybrid PDLP. Target attainment requires gap $\leq 1\%$ within 480 seconds. Complete time and final gap are arithmetic means over all 20 instances.

Method	Target reached ↑	Mean time (s) ↓	Mean gap ↓
CPU simplex	8/20	425.33	4.70%
Hybrid GPU PDLP + native crossover	15/20	371.08	1.37%
Hybrid GPU PDLP + Basis-SDM	**17/20**	**330.25**	**1.21%**

5.7 Crossover inside large-scale MILP solving

We evaluate complete solves of 20 large MILPs with 8,130–72,747 variables, including 12 with more than 30,000 variables. Within SCIP, we compare CPU simplex with two hybrid GPU PDLP methods. The hybrid methods use the same node scheduling policy and return a basis to HiGHS simplex through either native crossover or Basis-SDM. Complete solving time includes LP solution, basis construction, and MILP search (Appendix C.8).

Replacing native crossover with Basis-SDM reduces mean complete solving time from 371.08 to 330.25 seconds (11.0%) and increases target attainment from 15 to 17 instances (Table 7). CPU simplex reaches the target on eight instances, with a mean time of 425.33 seconds. Thus the improvement in basis quality translates into faster mixed-integer search.

6 Beyond Crossover: Predicting Sets in MILP

Mathematical optimization often requires choosing a set of interdependent decisions. Predicting each element separately does not directly capture which choices work together or which alternative sets are valid. We learn a joint distribution over complete sets: optimization objectives and certificates define target distributions over compatible families, and shared updates move an initial distribution toward a target under exact KL supervision. LP basis prediction is one instance of this broader modeling approach. We extend it here to *partial integer assignments* in MILP, using a layered decision graph to represent the distribution and completing predicted sets into feasible solutions for solver initialization.

The model selects a nonempty set of integer assignments from an LP-relaxation solution, using overlapping blocks for compatibility. A verified integer-feasible witness, chosen by objective quality, defines a uniform target over all nonempty subsets of agreeing assignments. Each can be completed by the witness. Related relaxation-based neighborhood heuristics include RINS and RENS (Danna et al., 2005; Berthold, 2014); here, the compatible family provides distributional supervision. A short sub-MILP completes the predicted set for SCIP initialization (Appendix D).

We evaluate eight MIPLIB validation problems (Gleixner et al., 2021), two seeds, two time budgets, and three methods: 96 paired runs. At ten seconds, family supervision produces verified candidates in 75.0% of runs, versus 43.8% for quality-weighted targets and 37.5% for the LP rule (Table 11). At two seconds, the rates are 37.5%, 25.0%, and 12.5%. Rates include preparation failures. The improvement at both budgets supports supervision over completable sets; further solver results appear in Appendix D.5.

7 Conclusion

Basis-SDM treats a complete basis as the unit of prediction and learns from families of compatible optimal bases. Exact KL supervision makes this joint distribution directly trainable. The resulting bases eliminate IPM cleanup on 64 of 111 unseen LPs and reduce simplex work across first-order inputs, direct prediction, and perturbations. Controlled ablations establish the benefit of family supervision, and complete MILP solves connect improved basis recovery to lower solving time. The assignment-set extension shows how optimization-derived families can supervise other coupled decisions. Limitations are discussed in Appendix E.

AI USE STATEMENT

In preparing this paper, we used generative AI tools to polish the writing, retrieve and identify relevant literature, and draft sections from author-provided notes and results. Writing assistance also included figure layout and LaTeX formatting. AI-assisted drafts and literature suggestions were checked and revised during manuscript preparation. The authors take full responsibility for the final text, references, technical claims, and experimental results.

REFERENCES

Ghussoun Al-Jeiroudi, Jacek Gondzio, and Julian A. J. Hall. Preconditioning indefinite systems in interior point methods for large scale linear optimisation. *Optimization Methods and Software*, 23(3):345–363, 2008. doi: 10.1080/10556780701535910.

David Applegate, Mateo Díaz, Oliver Hinder, Haihao Lu, Miles Lubin, Brendan O'Donoghue, and Warren Schudy. Practical large-scale linear programming using primal-dual hybrid gradient. In *Advances in Neural Information Processing Systems*, volume 34, pp. 20243–20257, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/a8fbbd3b11424ce032ba813493d95ad7-Abstract.html.

David Applegate, Mateo Díaz, Oliver Hinder, Haihao Lu, Miles Lubin, Brendan O'Donoghue, and Warren Schudy. PDLP: A practical first-order method for large-scale linear programming. arXiv:2501.07018, version 2, 2026. URL https://arxiv.org/abs/2501.07018v2.

Hatem Ben Amor, Jacques Desrosiers, and François Soumis. Recovering an optimal LP basis from an optimal dual solution. *Operations Research Letters*, 34(5):569–576, 2006. doi: 10.1016/j.orl.2005.10.001.

Timo Berthold. RENS: The optimal rounding. *Mathematical Programming Computation*, 6(1):33–54, 2014. doi: 10.1007/s12532-013-0060-9.

Robert E. Bixby and Matthew J. Saltzman. Recovering an optimal LP basis from an interior point solution. *Operations Research Letters*, 15(4):169–178, 1994. doi: 10.1016/0167-6377(94)90074-4.

Peter Businger and Gene H. Golub. Linear least squares solutions by Householder transformations. *Numerische Mathematik*, 7(3):269–276, 1965. doi: 10.1007/BF01436084.

Cardinal Operations. Cardinal optimizer (COPT) user guide: Parameters, 2026. URL https://guide.coap.online/copt/en-doc/parameter.html. Crossover interface and parameters; accessed September 19, 2026.

Coralia Cartis and Yiming Yan. Active-set prediction for interior point methods using controlled perturbations. *Computational Optimization and Applications*, 63(3):639–684, 2016. doi: 10.1007/s10589-015-9791-z.

Emilie Danna, Edward Rothberg, and Claude Le Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming*, 102(1):71–90, 2005. doi: 10.1007/s10107-004-0518-7.

Christophe Dupuy and Francis Bach. Learning determinantal point processes in sublinear time. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pp. 244–257, 2018. URL https://proceedings.mlr.press/v84/dupuy18a.html.

Leon Eifler and Ambros Gleixner. Safe and verified Gomory mixed-integer cuts in a rational mixed-integer program framework. *SIAM Journal on Optimization*, 34(1):742–763, 2024. doi: 10.1137/23M156046X.

Zhenan Fan, Xinglu Wang, Oleksandr Yakovenko, Abdullah Ali Sivas, Owen Ren, Yong Zhang, and Zirui Zhou. Smart initial basis selection for linear programs. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 9650–9664, 2023. URL https://proceedings.mlr.press/v202/fan23d.html.

Guillaume Gautier, Rémi Bardenet, and Michal Valko. Zonotope hit-and-run for efficient sampling from projection DPPs. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1223–1232, 2017. URL https://proceedings.mlr.press/v70/gautier17a.html.

David M. Gay. Electronic mail distribution of linear programming test problems. *Mathematical Programming Society COAL Newsletter*, 13:10–12, 1985. URL https://www.netlib.org/bibnet/authors/g/gay-david-m.html.

Dongdong Ge, Qi Huangfu, Zizhuo Wang, Jian Wu, and Yinyu Ye. Cardinal optimizer (COPT) user guide. arXiv:2208.14314, version 1, 2022. URL https://arxiv.org/abs/2208.14314v1.

Dongdong Ge, Chengwenjian Wang, Zikai Xiong, and Yinyu Ye. From an interior point to a corner point: Smart crossover. *INFORMS Journal on Computing*, 37(6):1670–1688, 2025. doi: 10.1287/ijoc.2022.0291.

Ambros Gleixner, Gregor Hendel, Gerald Gamrath, Tobias Achterberg, Michael Bastubbe, Timo Berthold, Philipp Christophel, Kati Jarck, Thorsten Koch, Jeff Linderoth, Marco Lübbecke, Hans D. Mittelmann, Derya Ozyurt, Ted K. Ralphs, Domenico Salvagnin, and Yuji Shinano. MIPLIB 2017: Data-driven compilation of the 6th mixed-integer programming library. *Mathematical Programming Computation*, 13(3): 443–490, 2021. doi: 10.1007/s12532-020-00194-3. URL https://miplib.zib.de/.

S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. How to find a good submatrix. In Vadim Olshevsky and Eugene Tyrtyshnikov (eds.), *Matrix Methods: Theory, Algorithms and Applications*, pp. 247–256. World Scientific, 2010. doi: 10.1142/9789812836021_0015.

Gurobi Optimization. How does NodeMethod work? Gurobi Help Center, 2021. URL https://support.gurobi.com/hc/en-us/community/posts/4413676791441-How-does-NodeMethod-work. Accessed September 19, 2026.

HiGHS Development Team. HiGHS documentation, 2026. URL https://ergo-code.github.io/HiGHS/stable/. Accessed September 19, 2026.

Qi Huangfu and Julian A. J. Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018. doi: 10.1007/s12532-017-0130-5.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012. doi: 10.1561/2200000044.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Tianhao Liu and Haihao Lu. A new crossover algorithm for LP inspired by the spiral dynamic of PDHG. Software and data archive, INFORMS Journal on Computing, 2025. URL https://github.com/INFORMSJoC/2024.0996. Repository DOI: 10.1287/ijoc.2024.0996.cd.

Tianhao Liu and Haihao Lu. A new crossover algorithm for LP inspired by the spiral dynamic of PDHG. *INFORMS Journal on Computing*, 2026. doi: 10.1287/ijoc.2024.0996. Published online April 6, 2026.

Anqi Lu and Junchi Yan. Learning initial basis selection for linear programming via duality-inspired tripartite graph representation and comprehensive supervision. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 40831–40847, 2025. URL https://proceedings.mlr.press/v267/lu25q.html.

Russell Lyons. Determinantal probability measures. *Publications Mathématiques de l'IHÉS*, 98:167–212, 2003. doi: 10.1007/s10240-003-0016-0.

Csaba Mészáros. Linear programming test problems. MISC and STOCHLP collections, n.d. URL https://sparse.tamu.edu/Meszaros. Collection description and original repository links; accessed September 19, 2026.

Lukas Schork and Jacek Gondzio. Implementation of an interior point method with basis preconditioning. *Mathematical Programming Computation*, 12(4):603–635, 2020. doi: 10.1007/s12532-020-00181-8.

Richard A. Tapia and Yin Zhang. An optimal-basis identification technique for interior-point linear programming algorithms. *Linear Algebra and its Applications*, 152:343–363, 1991. doi: 10.1016/0024-3795(91)90281-Z.

Seiya Tokui and Issei Sato. Evaluating the variance of likelihood-ratio gradient estimators. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3414–3423, 2017. URL https://proceedings.mlr.press/v70/tokui17a.html.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Deep set prediction networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/6e79ed05baec2754e25b4eac73a332d2-Abstract.html.

A Proofs and Exact Computation

Throughout this section, $A \in \mathbb{R}^{m\times n}$ has full row rank, θ is finite, and a certified family $\mathcal{E}_c$ is assumed nonempty whenever its teacher is used. All probability identities are in exact arithmetic.

We use $[n] = \{1, \dots, n\}$, $\bar{B} = [n] \setminus B$, and $a_i = A_{:i}$. The vector $\mathbf{1}_B \in \{0,1\}^n$ records membership in B, while $\mathbf{1}_{\{i\in B\}}$ is its ith coordinate. All sums over bases use unordered column sets. We first describe how to compute the distribution, then prove normalization, target optimality, exact training, and decoding properties.

A.1 Computing the basis distribution

Although an LP may have many possible bases, training only requires the normalizing constant and column inclusion probabilities. Both can be obtained from a thin QR factorization. The same factor also supports conditional-greedy selection, so neither training nor decoding needs to enumerate bases or store an $n \times n$ kernel.

Normalization and moments. Let $\alpha = \max_i \theta_i$ and $X = A\,\mathrm{diag}(e^{\theta-\alpha\mathbf{1}})$. The common shift leaves the distribution unchanged and keeps the column weights at most one. Write a thin QR factorization as

$$X^\top = QR, \qquad Q \in \mathbb{R}^{n\times m}, \quad Q^\top Q = I_m,$$

where $R \in \mathbb{R}^{m\times m}$ is upper triangular. Since $XX^\top = R^\top R$, the normalizer and the probability of a complete basis are

$$\begin{aligned} \log \mathcal{Z}(\theta) &= 2\sum_{r=1}^{m} \log|R_{rr}| + 2m\alpha, \\ q_\theta(B) &= \det(Q_{B,:})^2. \end{aligned} \tag{15}$$

For $v_i = Q_{i,:}^\top \in \mathbb{R}^m$, the projection kernel is represented implicitly by

$$K_{ij} = v_i^\top v_j, \qquad \pi_i = \|v_i\|_2^2. \tag{16}$$

Thus all marginal inclusions are row norms, and any required pair statistic uses one inner product. Logarithms of the triangular diagonal evaluate the normalizer without multiplying determinants or explicitly forming a matrix inverse. Appendix A.2 proves these identities.

Target distributions and exact supervision. For a certificate family, mandatory columns are fixed first, forbidden columns are removed, and the remaining admissible columns are projected orthogonally to the mandatory span. Applying the same factorization to this reduced matrix yields the target normalizer and inclusion probabilities (Appendix A.4). These quantities give the joint KL and its gradient through Eqs. (10)–(11); no individual basis labels or sampled loss estimates are needed.

Conditional-greedy decoding. Use the terminal factor Q and initialize $S_0 = \varnothing$ and $z_i^{(0)} = v_i$. After k selections, $z_i^{(k)}$ is the component of v_i orthogonal to the selected vectors. Its squared norm is exactly the conditional inclusion probability:

$$\begin{aligned} p_i^{(k)} &= \mathbb{P}(i \in B \mid S_k \subseteq B) = \|z_i^{(k)}\|_2^2, \qquad i \notin S_k, \\ j_k &\in \arg\max_{i\notin S_k} p_i^{(k)}, \qquad S_{k+1} = S_k \cup \{j_k\}. \end{aligned} \tag{17}$$

Conditioning on j_k removes one direction from every remaining vector:

$$u_k = \frac{z_{j_k}^{(k)}}{\|z_{j_k}^{(k)}\|_2}, \qquad z_i^{(k+1)} = z_i^{(k)} - u_k\big(u_k^\top z_i^{(k)}\big). \tag{18}$$

This is the orthogonal-projection form of DPP conditioning (Kulesza & Taskar, 2012). The scores sum to $m-k$, so a positive-probability extension exists until m columns have been selected. The resulting set is a complete basis, as proved in Appendix A.6. In dense arithmetic, both the factorization and the complete decoding sequence cost $O(nm^2)$ operations, with $O(nm + m^2)$ storage. Computation therefore depends on the matrix dimensions, not on the number of candidate bases.

A.2 Normalization, support, and inclusion probabilities

We first use Cauchy–Binet to obtain the normalizer and identify the sets with positive probability. Differentiating the normalizer then gives the inclusion probabilities.

Normalization and support. Let $D_\theta = \operatorname{diag}(e^{\theta_1}, \dots, e^{\theta_n})$ and $X = AD_\theta$. For any m-column set B, column scaling gives $\det(X_B) = \det(A_B) \prod_{i \in B} e^{\theta_i}$. Cauchy–Binet expresses $\det(XX^\top)$ as the sum of the squared determinants of these square submatrices:

$$\begin{aligned} \mathcal{Z}(\theta) &= \sum_{\substack{B \subseteq [n] \\ |B|=m}} e^{2\sum_{i \in B} \theta_i} \det(A_B)^2 \\ &= \sum_{\substack{B \subseteq [n] \\ |B|=m}} \det(X_B)^2 = \det(XX^\top) = \det\big(AD_\theta^2 A^\top\big)\,. \end{aligned} \tag{19}$$

Since D_θ is invertible and A has full row rank, $XX^\top$ is positive definite. Thus the sum is finite and strictly positive. Each summand is positive exactly when A_B is invertible, proving both normalization and support in Theorem 1. A different enumeration of the same set multiplies $\det(A_B)$ by the sign of a permutation; its square and the sum of potentials are unchanged. Hence its probability depends on the set, not its enumeration.

Inclusion probabilities. To compute its moments, write $G = AD_\theta^2 A^\top$ and

$$K_\theta = D_\theta A^\top G^{-1} A D_\theta. \tag{20}$$

Differentiating the sum of weights gives the expected membership indicator; differentiating the log determinant gives a matrix expression for the same quantity:

$$\begin{aligned} \partial_{\theta_i} \log \mathcal{Z}(\theta) &= 2 \sum_{B \ni i} q_\theta(B) = 2\pi_i(\theta), \\ \partial_{\theta_i} \log\det G &= \operatorname{tr}\big(G^{-1}\, 2e^{2\theta_i} a_i a_i^\top\big) = 2e^{2\theta_i} a_i^\top G^{-1} a_i. \end{aligned}$$

where a_i is column i of A. Therefore $\pi_i(\theta) = (K_\theta)_{ii}$ and $\nabla \log \mathcal{Z} = 2\pi$. A second differentiation gives

$$\frac{\partial^2 \log \mathcal{Z}}{\partial\theta_i \partial\theta_j} = 4\left\{\mathbb{E}[\mathbf{1}_{\{i \in B\}} \mathbf{1}_{\{j \in B\}}] - \pi_i \pi_j\right\}.$$

The Hessian is four times a covariance matrix and is positive semidefinite. Moreover, $K_\theta^2 = K_\theta$, $\operatorname{tr} K_\theta = m$, and $\sum_i \pi_i = m$. Adding $a\mathbf{1}$ to θ multiplies every size-m weight, and hence the normalizer, by e^{2am}. Thus $q_{\theta + a\mathbf{1}} = q_\theta$, which justifies centering the potential updates.

Joint inclusion probabilities. For indeterminates $t_1, \dots, t_n$, Cauchy–Binet and Sylvester's determinant identity give

$$\begin{aligned} \mathbb{E} \prod_{i \in B} (1 + t_i) &= \frac{\det\big(X \operatorname{diag}(1+t) X^\top\big)}{\det G} \\ &= \det(I_n + \operatorname{diag}(t) K_\theta)\,. \end{aligned}$$

Comparing the coefficient of $\prod_{i \in S} t_i$ on both sides proves $\mathbb{P}(S \subseteq B) = \det(K_{\theta,SS})$.

A.3 Optimality of a compatible basis

The certificate imposes two simple requirements: retain every column whose primal variable is positive, and select only columns with zero dual slack. The first preserves the primal solution; the second preserves the dual solution. An invertible basis satisfying both is therefore optimal.

Recall that $P_c = \{i : x^*_{c,i} > 0\}$ and $Z_c = \{i : s^*_{c,i} = 0\}$; a compatible basis satisfies $P_c \subseteq B \subseteq Z_c$ and $\det(A_B) \neq 0$.

Let (x_c^*, y_c^*, s_c^*) be primal–dual optimal, so that

$$Ax_c^* = b, \quad x_c^* \geq 0,$$
$$A^\top y_c^* + s_c^* = c, \quad s_c^* \geq 0,$$
$$(x_c^*)^\top s_c^* = 0.$$

Complementarity implies $P_c \subseteq Z_c$. For $B \in \mathcal{E}_c$, $P_c \subseteq B$ contains the support of x_c^*. Consequently $A_B x_{c,B}^* = b$, and invertibility of A_B gives

$$x_B^B = A_B^{-1} b = x_{c,B}^*, \qquad x_{\bar{B}}^B = 0 = x_{c,\bar{B}}^*.$$

Thus $x^B = x_c^*$ is primal feasible. Since $B \subseteq Z_c$, $s_{c,B}^* = 0$, so $A_B^\top y_c^* = c_B$. The basis dual solution is therefore $A_B^{-\top} c_B = y_c^*$, and its reduced costs satisfy

$$r^B = c - A^\top A_B^{-\top} c_B = s_c^* \geq 0.$$

The primal and dual objectives agree, proving optimality. Finally, nonempty $\mathcal{E}_c$ means that P_c lies inside at least one independent set of size m; hence A_{P_c} is independent and $|P_c| \leq m$. Because the initial law is strictly positive on every structural basis, a nonempty family has positive initial mass and its conditional teacher is well defined. These are the target claims in Theorem 1.

A.4 The optimal-family partition function

The target family fixes mandatory columns and allows the remaining columns to vary. To sum its weights, we first remove the directions already spanned by the mandatory columns. The remaining completion problem has lower dimension, so the same Cauchy–Binet argument applies.

Separate mandatory and free columns. Write $P = P_c$, $F = Z_c \setminus P$, and $p = |P|$. Choose an orthonormal basis Q_1 for the span of A_P, write $A_P = Q_1 R_P$, and let Q_2 span its orthogonal complement. $Q_1 \in \mathbb{R}^{m \times p}$, $Q_2 \in \mathbb{R}^{m \times (m-p)}$, and $R_P \in \mathbb{R}^{p \times p}$; $[Q_1, Q_2]$ is orthogonal. For any $D \subseteq F$ of size $m - p$,

$$\begin{bmatrix} Q_1^\top \\ Q_2^\top \end{bmatrix} [A_P, A_D] = \begin{bmatrix} R_P & Q_1^\top A_D \\ 0 & Q_2^\top A_D \end{bmatrix}.$$

The left multiplier is orthogonal. Taking squared determinants, including zero determinants for dependent completions, yields

$$\det(A_{P \cup D})^2 = \det(A_P^\top A_P) \det(Q_2^\top A_D)^2.$$

Define $\bar{A}_c = Q_2^\top A_F$. Nonempty $\mathcal{E}_c$ ensures that $\bar{A}_c$ has row rank $m - p$. Summing over D and applying Cauchy–Binet gives

$$\mathcal{Z}_c(\theta) = e^{2\sum_{i \in P_c} \theta_i} \det(A_{P_c}^\top A_{P_c}) \det\left(\bar{A}_c \operatorname{diag}(e^{2\theta_F}) \bar{A}_c^\top\right). \tag{21}$$

This is the sum of unnormalized weights over $\mathcal{E}_c$, so $q_\theta(\mathcal{E}_c) = \mathcal{Z}_c(\theta)/\mathcal{Z}(\theta)$. When $p = 0$, take $Q_2 = I$; when $p = m$, the free determinant is the empty determinant, equal to one. These conventions include both endpoint cases.

Conditional inclusion probabilities. Let $G_c = \bar{A}_c \operatorname{diag}(e^{2\theta_F}) \bar{A}_c^\top$ and let $\bar{a}_i$ be column i of $\bar{A}_c$, indexed by F. Differentiation gives the conditional inclusion vector

$$\frac{1}{2} \frac{\partial \log \mathcal{Z}_c(\theta)}{\partial \theta_i} = \begin{cases} 1, & i \in P_c, \\ e^{2\theta_i} \bar{a}_i^\top G_c^{-1} \bar{a}_i, & i \in F, \\ 0, & i \notin Z_c. \end{cases}$$

Evaluating at $\theta^{(0)}$ gives π_c^*. When $p = m$, the middle case is zero because every free column is excluded by the fixed cardinality. Thus both the teacher partition and its moments are available without enumerating bases.

A.5 EXACT KL AND ITS GRADIENT

The model and teacher assign weights using the same squared determinant. This common factor cancels in their probability ratio. Consequently, the joint KL depends only on their normalizers, potentials, and teacher inclusion probabilities.

The joint KL. On $\mathcal{E}_c$, the teacher density is

$$q_c^*(B) = \frac{e^{2\sum_{i\in B}\theta_i^{(0)}}\det(A_B)^2}{\mathcal{Z}_c(\theta^{(0)})}.$$

For each such B, its log ratio with q_θ is

$$\log\frac{q_c^*(B)}{q_\theta(B)} = \log\mathcal{Z}(\theta) - \log\mathcal{Z}_c(\theta^{(0)}) + 2(\theta^{(0)}-\theta)^\top\mathbf{1}_B.$$

Taking expectation under q_c^* replaces $\mathbf{1}_B$ by π_c^* and gives Eq. (10):

$$D_{\mathrm{KL}}(q_c^*\|q_\theta) = \log\mathcal{Z}(\theta) - \log\mathcal{Z}_c(\theta^{(0)}) + 2(\theta^{(0)}-\theta)^\top\pi_c^*.$$

The gradient. The input, teacher, and initial potential are held fixed when differentiating the predicted potential. Applying Appendix A.2 then gives

$$\nabla_\theta D_{\mathrm{KL}}(q_c^*\|q_\theta) = \nabla\log\mathcal{Z}(\theta) - 2\pi_c^* = 2\{\pi(\theta)-\pi_c^*\},$$

which proves (11). For a unique minimizing teacher in (9), differentiation of the selected branch gives the gradient of the minimum objective. At a tie, one minimizing teacher is selected for the entire path; the derivative above remains the exact derivative of that teacher's branch.

A.6 CONDITIONAL DECODING PRESERVES A COMPLETE BASIS

To prove that decoding returns a basis, it suffices to show that each partial selection still has a valid completion. We track the total probability of all bases containing the selected columns and show that a positive-probability extension always exists.

For a partial selection S, define $M_q(S) = \sum_{B\supseteq S} q(B)$, the probability that a sampled basis contains all columns in S. Conditional-greedy decoding selects the largest conditional inclusion probability:

$$j^* \in \arg\max_{j\notin S}\frac{M_q(S\cup\{j\})}{M_q(S)}, \qquad S\leftarrow S\cup\{j^*\}, \quad S_0=\varnothing. \tag{22}$$

The following argument applies to any probability law q supported on size-m structural bases. For a partial set S with $|S|<m$ and $M_q(S)>0$,

$$\sum_{j\notin S} M_q(S\cup\{j\}) = \sum_{B\supseteq S} q(B)\sum_{j\notin S}\mathbf{1}_{\{j\in B\}} = (m-|S|)M_q(S) > 0. \tag{23}$$

The equality follows because every basis containing S has exactly $m-|S|$ elements outside S. At least one extension consequently has positive mass, and a maximum of the conditional scores $M_q(S\cup\{j\})/M_q(S)$ is positive. Any maximizing choice preserves $M_q(S\cup\{j\})>0$.

Initially $M_q(\varnothing)=1$. Induction shows that all m choices are well defined and preserve positive completion mass. The process never repeats a column, so its terminal set has size m. At this size, the only size-m set containing it is itself, and $M_q(S)=q(S)>0$. The returned set is therefore a structural basis, proving the decoder claim of Theorem 1. The guarantee follows from preserving positive completion mass at each selection.

A.7 EXACT SAMPLING AND ORDER INDEPENDENCE

A sequential sampler must recover the probability of an unordered set, accounting for all its selection orders. The key is that the conditional factors telescope: every ordering of a basis receives exactly the same fraction of its probability.

By (23), the probabilities in (12) sum to one whenever $M_q(S) > 0$ and $|S| < m$. An extension with zero completion mass has zero selection probability. Starting from the empty set, the sampler therefore encounters only positive-mass prefixes and has no zero denominator.

Fix a supported basis B and any permutation $(b_{\sigma(1)}, \dots, b_{\sigma(m)})$ of its elements. Set $S_r = \{b_{\sigma(1)}, \dots, b_{\sigma(r)}\}$. Since $M_q(S_r) \geq q(B) > 0$, every factor along this ordering is defined, and the probability of the ordered path is

$$\begin{aligned}\mathbb{P}(J_1 = b_{\sigma(1)}, \dots, J_m = b_{\sigma(m)}) &= \prod_{r=1}^{m} \frac{M_q(S_r)}{(m-r+1)M_q(S_{r-1})} \\ &= \frac{M_q(B)}{m! M_q(\varnothing)} = \frac{q(B)}{m!}. \end{aligned} \tag{24}$$

A terminal set with $q(B) = 0$ cannot be reached, because the final extension would have zero probability. There are $m!$ distinct orderings of each supported B; their events are disjoint, so summing (24) gives exactly $q(B)$. Conditional on the returned set, its enumeration is uniform. This proves Proposition 2: sequential sampling realizes the prescribed distribution over unordered bases.

A.8 KL CONTROLS OPTIMAL-FAMILY MASS

A small KL loss must place substantial probability on the teacher family. We make this precise by separating two effects: how much total mass reaches that family, and how that mass is distributed within it.

Fix the teacher c^* selected in (9), and write $p = q^*_{c^*}$, $E = \mathcal{E}_{c^*}$, and $q_t = q_{\theta^{(t)}}$. Since E is nonempty and each potential is finite, $q_t(E) > 0$. On E, define $q_{t,E}(B) = q_t(B)/q_t(E)$. The support of p lies in E, hence

$$\begin{aligned} D_{\mathrm{KL}}(p\|q_t) &= \sum_{B \in E} p(B) \log \frac{p(B)}{q_{t,E}(B) q_t(E)} \\ &= D_{\mathrm{KL}}(p\|q_{t,E}) - \log q_t(E) \geq -\log q_t(E). \end{aligned} \tag{25}$$

Here the inequality is nonnegativity of KL between two normalized laws on E. Since every $\omega_t > 0$ and every term of the path loss is nonnegative,

$$\omega_t D_{\mathrm{KL}}(p\|q_t) \leq \sum_{u=1}^{4} \omega_u D_{\mathrm{KL}}(p\|q_u) = \sqrt{m}\mathcal{L}.$$

$$-\log q_t(E) \leq D_{\mathrm{KL}}(p\|q_t) \leq \frac{\sqrt{m}}{\omega_t}\mathcal{L}, \qquad q_t(E) \geq \exp\left(-\frac{\sqrt{m}}{\omega_t}\mathcal{L}\right).$$

Combining this inequality with (25) and exponentiating proves (13). Thus the training objective controls the mass assigned to the selected optimal family at every supervised state. The result quantifies the learned distribution, while Theorem 1 establishes validity of conditional-greedy decoding.

A.9 APPROXIMATING THE TARGET DISTRIBUTION

To approach a target family, increase the potentials of mandatory columns and decrease those of forbidden columns. Compatible bases retain their relative weights, while every incompatible basis is increasingly penalized. The normalized distribution therefore converges to the conditional teacher.

Let $w_0(B) = e^{2\sum_{i \in B} \theta_i^{(0)}} \det(A_B)^2$ and, for $\lambda > 0$, define

$$\theta_{c,i}(\lambda) = \theta_i^{(0)} + \lambda \mathbf{1}_{\{i \in P_c\}} - \lambda \mathbf{1}_{\{i \notin Z_c\}}.$$

Relative to $w_0(B)$, the new weight is multiplied by $e^{2\lambda(|B\cap P_c|-|B\setminus Z_c|)}$. Divide every weight by $e^{2\lambda|P_c|}$, which does not change the law. The resulting weight is

$$w_0(B)e^{-2\lambda d_c(B)}, \qquad d_c(B) = |P_c \setminus B| + |B \setminus Z_c|.$$

Exactly the sets in $\mathcal{E}_c$ have $d_c(B) = 0$; every other structural basis has $d_c(B) \geq 1$. Set $Z_E = \sum_{B\in\mathcal{E}_c} w_0(B) > 0$, $Z_{\text{out}} = \sum_{B\notin\mathcal{E}_c} w_0(B)$, and

$$R_\lambda = \sum_{B\notin\mathcal{E}_c} w_0(B)e^{-2\lambda d_c(B)} \leq e^{-2\lambda} Z_{\text{out}}.$$

Inside $\mathcal{E}_c$, the normalized law is $w_0(B)/(Z_E + R_\lambda)$, whereas the teacher is $w_0(B)/Z_E$. The total excess mass outside $\mathcal{E}_c$ equals the total missing mass inside it. Using $\|p - q\|_{\text{TV}} = \frac{1}{2}\sum_B |p(B) - q(B)|$, their total variation distance is

$$\|q_{\theta_c(\lambda)} - q_c^*\|_{\text{TV}} = \frac{R_\lambda}{Z_E + R_\lambda} \leq \frac{e^{-2\lambda} Z_{\text{out}}}{Z_E} \longrightarrow 0.$$

If $Z_{\text{out}} = 0$, the two laws already agree. Otherwise convergence is to a boundary distribution of the finite-potential family. This proves the closure claim in Proposition 3. Centering $\theta_c(\lambda)$ leaves each law unchanged, so the same limit holds for centered potentials.

B Architecture and Training

Basis-SDM has two learned components. The Global KKT encoder represents the LP and its solver state; a shared update module transforms the basis distribution using this context and distribution feedback. The same update module is applied four times, as summarized in Algorithm 1.

B.1 Global KKT encoder

The Global KKT encoder represents the LP as a bipartite graph: columns are variable nodes, rows are constraint nodes, and nonzero coefficients of A are edges. Its inputs describe both the LP structure and the current solver state. Column features include objective coefficients, column norms and degrees, slack-variable indicators, initial potentials, complementarity, residuals, and the empirical ranks of primal values and dual slacks. Row features describe right-hand sides, row norms and degrees, dual values, constraint activities, and primal residuals. Edge features encode coefficient signs and magnitudes.

Up to 12 recorded states provide trajectory information. For IPM, changes in $\log x_i$ and $\log s_i$ are measured against $-\log \mu_{\text{IPM}}$. Their local slopes and relative observation times describe how each variable approaches optimality. Attention pooling summarizes these observations into a trajectory embedding. A column MLP combines this embedding with the current column features; a separate MLP embeds the row features.

Four rounds of graph attention pass information from columns to rows and back to columns. Global attention pooling summarizes the whole LP. The hidden width is 192, with six attention heads per round and separate parameters across rounds. The outputs are column representations $\mathbf{h}$, row representations $\mathbf{r}$, and a global representation g. Permuting variables or constraints permutes the corresponding node representations and leaves g unchanged.

B.2 Initial basis distribution

For IPM inputs, the initial potential is $\theta_i^{(0)} = \frac{1}{2}\log(x_i/s_i)$. It favors columns with large primal values relative to their dual slacks. Direct prediction has no solver state and starts from $\theta^{(0)} = 0$.

A PDHG state may contain zero primal values or negative reduced costs. To define the same logarithmic ratio, we use

$$\widehat{x}_i = [x_i]_+ + \delta_x, \qquad \widehat{s}_i = |s_i| + \delta_s, \qquad \theta_i^{(0)} = \tfrac{1}{2}\log(\widehat{x}_i/\widehat{s}_i).$$

Here $[a]_+ = \max(a, 0)$, and the small positive offsets scale with the nonzero coordinates. The negative part $[-s_i]_+$ remains an input feature, so the encoder retains information about dual infeasibility.

The initial distribution also supplies fixed statistics $\mathcal{C}$ for the update module. Using the factorization in Appendix A.1, let $K_0 = Q_0 Q_0^\top$ and $\pi_i^{(0)} = K_{0,ii}$. For distinct columns i, j, define the two probabilities

$$\begin{aligned} p_{\mathrm{src}} &= \mathbb{P}(i \in B,\ j \notin B) = \pi_i^{(0)}(1 - \pi_j^{(0)}) + K_{0,ij}^2, \\ p_{\mathrm{dst}} &= \mathbb{P}(j \in B,\ i \notin B) = \pi_j^{(0)}(1 - \pi_i^{(0)}) + K_{0,ij}^2. \end{aligned} \tag{26}$$

These follow from $\mathbb{P}(i, j \in B) = \pi_i^{(0)}\pi_j^{(0)} - K_{0,ij}^2$. The initial inclusions and the log ratio $\log(p_{\mathrm{dst}}/p_{\mathrm{src}})$ form $\mathcal{C}$: they indicate which columns are favored and which member of a pair the distribution prefers to retain.

B.3 Shared basis-distribution updates

At each step, the update module combines the fixed encoder output with feedback from the current distribution. A pair MLP embeds two column representations together with the global context, initial statistics, step index, and feedback. A two-layer Transformer then models interactions among the pair embeddings. It uses six attention heads, feed-forward width 384, GELU activations, and layer normalization before attention and feed-forward layers.

Weighted aggregation maps the pair embeddings back to columns and produces a shared summary. A column MLP combines these representations with the current potentials, their changes from initialization, and the initial and calibrated inclusion scores. Its output is the increment in

$$\theta^{(t+1)} = \theta^{(t)} + \Delta\theta^{(t)}. \tag{27}$$

The increment includes centering and bounding of the potentials. All four steps use the same network parameters. Each step therefore updates one distribution over complete bases, while the feedback tells the next step how column preferences have changed.

B.4 Distribution feedback

The feedback o_t describes how column preferences evolve relative to the initial distribution. Let $d^{(t)} = \theta^{(t)} - \theta^{(0)}$. Calibrated inclusion scores are

$$\begin{aligned} \widetilde{\pi}_i^{(t)} &= \sigma\Big(\operatorname{logit} \pi_i^{(0)} + 2d_i^{(t)} - \lambda_t\Big), \\ \sum_{i=1}^{n} \widetilde{\pi}_i^{(t)} &= m. \end{aligned} \tag{28}$$

Here σ is the logistic function and $\operatorname{logit}(p) = \log[p/(1-p)]$. The scalar λ_t makes the scores sum to the required basis size m; initial probabilities of zero or one remain fixed. The corresponding feedback for a column pair is

$$\widetilde{\ell}_{i \to j}^{(t)} = \log \frac{p_{\mathrm{dst}}}{p_{\mathrm{src}}} + 2\big(d_j^{(t)} - d_i^{(t)}\big). \tag{29}$$

Increasing θ_j relative to θ_i strengthens the preference for selecting j in place of i. These scores summarize the changing distribution for the neural update. The KL loss and its gradient use the exact inclusion probabilities from Appendix A.1, rather than these calibrated feedback scores.

B.5 Conditional-greedy basis selection

Starting from the empty set, the decoder repeatedly selects the largest conditional inclusion probability under the terminal law, with a fixed column order resolving ties. The orthogonal updates in Appendix A.1 account for all previously selected columns. After m choices, the resulting set is a full-rank basis by Appendix A.6.

B.6 Joint reweighting of complete bases

A potential increment changes the relative probabilities of complete bases according to

$$\begin{aligned} &\log \frac{q_{\theta+\Delta\theta}(B_1)}{q_{\theta+\Delta\theta}(B_2)} - \log \frac{q_\theta(B_1)}{q_\theta(B_2)} \\ &\qquad = 2\Delta\theta^\top(\mathbf{1}_{B_1} - \mathbf{1}_{B_2}). \end{aligned} \tag{30}$$

Table 8: Composition of the LP training and validation sets. Counts refer to numerical LP instances, excluding repeated solver observations and certificates. Testing uses a separate set of 111 unseen LPs.

Data category	Training	Validation
Original real LPs	125	6
Local augmentation	1,849	355
Grouped-structure instances	2,048	128
KKT-generated instances	7,680	128
Total	11,702	617

Thus column-wise potentials control a joint set distribution: the increment reallocates mass among bases, while the determinant preserves their structural compatibility.

B.7 Training and data

The encoder and shared update module are trained jointly with AdamW, learning rate 3×10^{-5}, and zero weight decay. The four path weights are $0.1/0.2/0.3/0.4$, with normalization $1/\sqrt{m}$. IPM, PDHG, and direct prediction use the same architecture with separately trained parameters. Their targets are conditioned on the corresponding initial law; direct prediction observes only (A, b, c) and starts from zero potentials.

Data. The training set contains 125 original LPs from Netlib (Gay, 1985) and the MISC/STOCHLP subsets of the Mészáros collection (Mészáros, n.d.), together with augmented and generated problems (Table 8). Six additional original LPs form the real-instance validation set. Model selection uses training and validation data, while the same 111 unseen LPs are reserved for testing under the three input settings.

Certificate-family supervision. Each training LP has primal–dual witnesses satisfying feasibility and complementarity. A witness with nonempty compatible family $\mathcal{E}_c$ defines the conditional teacher in Eq. (6); additional witnesses supply alternative families. One certificate therefore supervises all compatible full-rank completions. Training uses the exact KL between these teachers and the predicted laws, and inference requires no certificates.

Certificate-preserving augmentation. For a fixed source matrix A, nonnegative complementary vectors $\widetilde{x}, \widetilde{s}$ and a dual vector $\widetilde{y}$ define

$$\widetilde{b} = A\widetilde{x}, \qquad \widetilde{c} = A^\top \widetilde{y} + \widetilde{s}, \qquad \widetilde{x}^\top \widetilde{s} = 0.$$

These identities certify optimality for the augmented LP. Varying the witnesses within a support pattern or toward its boundary produces coefficient variations with corresponding target families. Each augmented problem supplies its own solver observations and conditional teachers; all observations of the same LP retain the same split.

C Experimental Settings

C.1 Evaluation metrics

The LP experiments use 111 unseen problems: 37 small, 58 medium, and 16 large, grouped by the number of structural columns. Within each comparison, all methods use the same LP and optimization state. We report the shifted geometric mean of simplex iterations defined in Eq. (14). Successful solves retain their observed iteration counts; failures count as 10,000 iterations. For native crossover, we use the solver's reported simplex count. HiGHS does not include its internal crossover pushes in this count.

Time results are geometric means over problems. COPT native IPM crossover is run three times per problem, and its mean time enters the cross-problem summary. Failed IPM attempts retain their measured time; the timing rule for direct-prediction failures is given in Appendix C.5.

C.2 IPM INPUTS AND COMPARISON METHODS

For each LP, we retain the last recorded IPM state before relative complementarity falls below 10^{-4}. This state supplies the common input for basis recovery.

Basis-selection baselines. Diagonal weighting selects linearly independent columns in descending primal–dual scale order (Al-Jeiroudi et al., 2008). Scaled CPQR applies column-pivoted QR to AD, where $D = \mathrm{diag}(e^{\theta})$ (Businger & Golub, 1965). Weighted exchange-20 and scaled maxvol-20 improve the diagonal-weighting and CPQR bases, respectively, using maximum-volume exchanges (Schork & Gondzio, 2020; Goreinov et al., 2010). Each exchange maximizes the increase in $\log|\det(A_B D_B)|$, with at most 20 accepted exchanges.

Smart crossover identifies a candidate optimal face from the input state and recovers a basis by solving a perturbed subproblem (Ge et al., 2025). KKT-adapted LP-GNN follows the variable-status prediction approach of Fan et al. (2023): a bipartite GNN learns from the statuses of one optimal basis, and a rank-aware selection procedure assembles the predicted columns into a basis. We supply optimization-state features for crossover; the static-input version is evaluated separately in Table 4. All constructed bases initialize the same HiGHS simplex solver.

For native crossover, we pass the state through each solver's supported interface: HiGHS receives the primal solution, while COPT also receives the dual solution and reduced costs $c - A^{\top}y$.

C.3 IPM CROSSOVER TIME

Table 2 measures the complete crossover process, from a given IPM state to the end of simplex cleanup. Each problem is processed individually and timed in one continuous execution. Table 9 divides this interval into basis construction and subsequent solving. Construction includes preparing the supplied inputs, neural prediction, and decoding; solving includes basis validation, solver setup, basis import, and simplex optimization.

The interval starts after the IPM solve and model loading. For native methods, it includes solver setup, native crossover, and subsequent optimization. Each table entry is aggregated from its per-problem times, including the continuously measured total. Thus the total need not equal the sum of the two stage-wise geometric means.

Table 9: Components of Basis-SDM crossover time from IPM states. Geometric means in seconds; total time is measured over the complete crossover interval.

Stage	Medium (58)	Large (16)
Basis construction	0.02068	0.13260
Solver stage	0.01287	0.10239
Total crossover	0.03589	0.30748

C.4 CROSSOVER FROM PDHG SOLUTIONS

We evaluate whether crossover can recover useful bases from first-order solutions at different accuracies. PDHG runs for up to 8,000 iterations on each test LP. We use the first recorded solution with residual $\rho \leq 10^{-4}$; if this accuracy is not reached, we use the recorded solution with the smallest residual. Of the 111 problems, 93 reach the threshold; the remaining 18 use their lowest-residual solution. The model also receives up to 11 earlier states to describe the solver trajectory.

All crossover methods start from these same solutions. The Liu–Lu baseline uses the authors' released implementation with its default settings (Liu & Lu, 2026; 2025). Its recovered basis, like those of the other basis-construction methods, initializes HiGHS for simplex cleanup. Native HiGHS and COPT perform crossover through their own interfaces. Smart's reduced-face subproblem is infeasible on 65 problems; these attempts count as failures under the common evaluation rule. Table 10 gives the results by problem size.

Table 10: Simplex iterations after PDHG crossover across problem sizes. Results use the same 111 LPs and failure penalties as Table 3.

Method	Small	Medium	Large	All	Fail.
Diagonal weighting	63.36	535.42	3322.24	343.12	15
Weighted exchange-20	38.93	453.31	2905.99	262.96	15
Scaled CPQR	38.49	350.22	2320.54	221.55	0
Scaled maxvol-20	37.36	354.54	2939.78	228.53	0
Smart (PDHG adapter)	252.83	1641.35	7823.42	1102.78	65
Liu–Lu → HiGHS	50.78	413.33	4714.63	293.12	17
HiGHS native crossover	97.63	711.03	5692.58	496.14	0
COPT native crossover	69.14	625.21	2521.27	367.99	0
Basis-SDM	**2.15**	**107.43**	**822.84**	**43.66**	0

C.5 Direct basis prediction

Here the input consists only of (A, b, c). Basis-SDM starts from zero potentials. Scaled CPQR and maxvol use column-norm weights $d_i = 1/\max(\|A_{:i}\|_2, \varepsilon)$, with at most 20 maxvol exchanges. The learned initializer of Fan et al. (2023) is trained for 30 epochs to predict column and row statuses, which determine an augmented basis of structural columns and signed identity columns.

Each predicted basis is evaluated in both HiGHS and COPT. Comparisons within a solver use the same LP, disabled presolve, and feasibility tolerance 10^{-7}; COPT also disables dualization. Reported iterations include feasibility restoration, and the times in Table 4 measure optimization after basis import.

All 111 problems enter the summaries. To account for failures in the timing comparison, the longest successful solve across methods in the same solver and size group serves as a lower bound on failure time. A failed attempt receives this value or its measured solve time, whichever is larger; an attempt that does not reach the solver receives the former.

C.6 Numerical perturbations

We perturb 30 source LPs to test sensitivity to changes in their coefficients. Each source contributes an unperturbed problem and 12 variants: perturbations of A, b, c, or all three, at strengths of 1%, 10%, and 30%. Perturbations are multiplicative and preserve sparsity. For each coefficient block, the same perturbation direction is used at all three strengths.

Of the resulting 390 problems, 315 provide valid evaluation inputs; 68 are infeasible, three are unbounded, and four cannot be prepared. Table 5 uses the 13 sources valid under every perturbation condition (three small, six medium, and four large). This gives a matched comparison against the same unperturbed problems at every strength.

A fresh IPM run supplies each perturbed input, using the first state with $\mu/\mu_0 \le 10^{-5}$. Both crossover methods receive that state, and Basis-SDM is applied without retraining. Across all 315 valid inputs, its cleanup sGM is 77.07 with no failures, compared with 570.13 and seven failures for native HiGHS.

C.7 Supervision and output ablations

The supervision ablation replaces a certificate-family target with one optimal basis B^* from that family. The target becomes $q^* = \delta_{B^*}$, and the loss is

$$-\frac{1}{\sqrt{m}} \sum_{t=1}^{4} \omega_t \log q_{\theta^{(t)}}(B^*).$$

The encoder, solver-state inputs, four shared updates, optimizer, and conditional-greedy decoder remain the same. Both models train for 20 epochs; Table 6 also shows the single-basis model after five epochs.

The independent-score variant instead predicts a score for each column. It uses static (A, b, c) inputs with the same encoder and update architecture, and trains for 20 epochs by binary cross-entropy against certificate-derived inclusion labels. One certificate family supervises the whole path. At prediction time, columns are considered in descending score order and retained when they increase rank. Table 4 provides the Basis-SDM results with the same static inputs.

C.8 Large-scale MILP crossover

The 20 MILPs contain 8,130–72,747 original variables, with more than 30,000 in 12 problems. We compare complete SCIP solves with a HiGHS LP interface. The CPU method uses simplex throughout. Both hybrid methods use GPU cuPDLP with the same node scheduling policy; they differ in whether native crossover or Basis-SDM supplies a basis for subsequent HiGHS simplex solves.

The primal–dual gap measures the relative difference between the incumbent objective and the solver's dual bound. A run meets the target if this gap falls to 1% or below within 480 seconds. Complete time includes initialization, LP solves, basis recovery, and mixed-integer search, with model initialization apportioned across problems. Table 7 reports the arithmetic mean of complete time and final gap over all 20 runs, including those that reach the time limit.

D Set Distributions for Partial Integer Assignments

We extend set-distribution supervision from LP bases to partial integer assignments. The model selects variables to fix, then a sub-MILP completes the remaining variables to provide a feasible starting solution. These choices are coupled: individually plausible assignments may be incompatible when imposed together.

We model the entire assignment set with a layered decision graph. A known integer-feasible solution defines a family of compatible target sets, and an exact KL objective supervises their joint distribution. The following sections describe the representation, probability computations, supervision, and solver experiment.

D.1 Representing a distribution over assignment sets

Consider a MILP with linear constraints $\ell_r \le A_r w \le u_r$, variable bounds $\ell_j \le w_j \le u_j$, and integer-variable indices $\mathcal{I}$. A partial assignment is an unordered set $S = \{(j, v_j) : j \text{ is fixed}\}$. Imposing S fixes the listed variables and leaves all other variables in their original domains.

Let $\bar{w}$ be a feasible solution of the continuous relaxation. We consider integer variables whose relaxation values are already integral and lie in their candidate domains, and denote their index set by F. These domains are formed from rounded LP values, bounds, and neighboring integers. For each $j \in F$, the model chooses either to fix $w_j = \bar{w}_j$ or to leave w_j free, denoted by $\perp$. Thus, it learns which integral LP values to retain together.

We represent these choices in blocks of at most three variables. Adjacent blocks share one variable and must agree on its assignment. A local configuration is excluded if the unfixed variables, even within their full bounds, cannot satisfy an original linear constraint. The complete assignment must be nonempty, and shared variables are counted only once. These conditions define the allowed collection $\mathcal{S}_{\mathrm{LP}}(H)$ for the problem and relaxation state H.

A layered decision graph represents the collection: each layer chooses a block configuration, and transitions enforce agreement between adjacent blocks. Every allowed set corresponds to exactly one path. Giving configuration c in block g a potential $\theta_{g,c}$ defines

$$q_\theta(S \mid H) = \frac{\mathbf{1}\{S \in \mathcal{S}_{\mathrm{LP}}(H)\}}{Z_{\mathrm{LP}}(\theta)} \exp\left(\sum_g \theta_{g,c_g(S)}\right), \tag{31}$$

where $c_g(S)$ is the configuration induced by S and Z_{LP} normalizes the distribution. The graph rules ensure consistent assignments and remove local conflicts. Whether all remaining integer variables can be assigned feasibly is established by a known feasible solution during training or by solving the resulting sub-MILP during evaluation.

D.2 Computing the distribution and choosing an assignment

Dynamic programming computes the normalizer and configuration probabilities exactly. Let u_0 be the initial state and θ_e the configuration potential on edge $e : u \to v$. The log total weights of paths reaching and leaving a state are denoted by α and β, respectively. They satisfy

$$\begin{aligned} \alpha(v) &= \mathrm{LSE}_{e:u\to v}\{\alpha(u) + \theta_e\}, \\ \beta(u) &= \mathrm{LSE}_{e:u\to v}\{\theta_e + \beta(v)\}, \\ \log Z_{\mathrm{LP}} &= \beta(u_0), \end{aligned}$$

where LSE denotes log-sum-exp, $\alpha(u_0) = 0$, and $\beta = 0$ at valid terminal states. States with no valid path have value $-\infty$. An edge is selected with marginal probability

$$\mu_e = \exp\{\alpha(u) + \theta_e + \beta(v) - \log Z_{\mathrm{LP}}\}.$$

Summing these probabilities over edges carrying the same configuration gives $\mu_{g,c}$. In particular, $\mu_{g,c} = \partial \log Z_{\mathrm{LP}}/\partial\theta_{g,c}$, the identity used to compute the training gradient.

These marginals also give the entropy $\log Z_{\mathrm{LP}} - \sum_{g,c} \mu_{g,c}\theta_{g,c}$ and expected assignment count $\sum_{g,c} \mu_{g,c} n_{g,c}$, where $n_{g,c}$ counts variables assigned for the first time in block g. Both summarize the current distribution for the learned updates.

For the solver experiments, we choose the most probable assignment set (MAP). Replacing log-sum-exp by maximization gives the highest-weight path and hence the maximizing set. Exact sampling is also available: at state u, choose edge $e : u \to v$ with probability $\exp\{\theta_e + \beta(v) - \beta(u)\}$. The intermediate terms cancel along the path, recovering Eq. (31). Both selection rules take time linear in the number of graph edges.

D.3 Supervising all assignments compatible with a feasible solution

A feasible solution provides a whole family of correct assignment labels: fixing any subset of its variable values leaves that solution available as a completion.

Among verified integer-feasible solutions, choose $w^\star$ with the best original objective value subject to a nonempty agreement set $J = \{i \in F : w^\star_i = \bar{w}_i\}$. The target is uniform over all nonempty subsets of these agreeing assignments:

$$\begin{aligned} \mathcal{C}_J &= \{\varnothing \neq S \subseteq \{(i, \bar{w}_i) : i \in J\}\}, \\ q^*(S) &= \frac{\mathbf{1}\{S \in \mathcal{C}_J\}}{2^{|J|} - 1}. \end{aligned} \tag{32}$$

Every target set admits the same feasible completion $w^\star$. Supervision thus retains the freedom to choose among compatible sets. The solution need only be feasible, not globally optimal; the objective determines which available solution defines the family. These solutions and their objective values are used only during training.

This use of agreement is related to established MILP heuristics. RINS fixes variables on which a relaxation solution and an incumbent agree (Danna et al., 2005); RENS fixes integral relaxation values and restricts the remaining integer variables to adjacent integers (Berthold, 2014). Here, agreement defines a distribution over assignment sets for learning. Let $\mu^*_{g,c}$ denote the probability of block configuration c under this target. The joint KL and its gradient then have a closed form.

Proposition 4 (Supervision by a feasible-solution family)**.** *Assume $w^\star$ is integer feasible, $J \neq \varnothing$, and $\mathcal{S}_{\mathrm{LP}}(H)$ contains every set in $\mathcal{C}_J$. Every target set admits the completion $w^\star$, and*

$$\begin{aligned} D_{\mathrm{KL}}(q^* \| q_\theta) &= \log Z_{\mathrm{LP}}(\theta) - \sum_{g,c} \theta_{g,c}\mu^*_{g,c} - \log(2^{|J|} - 1), \\ \nabla_\theta D_{\mathrm{KL}}(q^* \| q_\theta) &= \mu(\theta) - \mu^*. \end{aligned} \tag{33}$$

Proof. First, every assignment in $S \in \mathcal{C}_J$ agrees with $w^\star$. Fixing these variables therefore leaves $w^\star$ feasible. There are $2^{|J|} - 1$ nonempty subsets, each represented once, so the target is normalized and its entropy is $\log(2^{|J|} - 1)$.

Next, on the target support, Eq. (31) gives

$$\log \frac{q^*(S)}{q_\theta(S)} = \log Z_{\mathrm{LP}}(\theta) - \sum_g \theta_{g,c_g(S)} - \log(2^{|J|} - 1).$$

Taking the expectation under q^* replaces each configuration indicator by $\mu^*_{g,c}$, yielding the KL expression. Finally, differentiating $\log Z_{\mathrm{LP}}$ gives the model marginals $\mu(\theta)$, so the gradient is their difference from the target marginals. The support assumption ensures that all terms are finite. □

D.4 Model and supervision comparison

The model combines a sparse variable–constraint encoder with four shared distribution updates. The encoder represents all linear constraints and both continuous and integer variables using two rounds of attention, four heads, and hidden width 64. Column representations, global context, and numerical features are combined into block embeddings. An LP-distance prior and a learned head initialize the configuration potentials. Each update uses the current potentials, configuration marginals, entropy, and expected assignment count. Centering the potentials within each block leaves the distribution unchanged, since every set selects one configuration per block.

We compare family supervision with a quality-weighted target over a finite collection of assignment sets. For the latter, up to 24 distinct sets per training problem are obtained from feasible solutions, rules, and joint samples. A set enters the target if a 0.75-second sub-MILP solve finds a verified feasible completion. Its weight favors a better objective and a shorter completion time:

$$\ell_a = \frac{\max\{0, s(z_a - z_{\mathrm{ref}})\}}{\max(1, |z_{\mathrm{ref}}|)} + 0.005 \min\{t_a/0.75, 2\},$$
$$p_a \propto \exp(-\ell_a/0.02).$$

Here z_a is the completed solution's objective, z_{ref} is the published reference objective, and t_a is completion time. The sign is $s = 1$ for minimization and $s = -1$ for maximization. These target sets are prepared offline.

Both learned models use the same inputs, allowed sets, four-update architecture, and MAP selection. Their supervision differs: quality weighting uses a finite set of successful assignments, whereas Eq. (32) covers all nonempty subsets agreeing with the selected feasible solution. The quality-weighted model is trained for 30 epochs with AdamW, learning rate 3×10^{-4}, and weight decay 10^{-5}; validation KL selects epoch 14. Family supervision starts from the same parameters and optimizes Eq. (33) for three epochs with Adam at learning rate 10^{-4}. Its loss is normalized by the number of eligible integer variables.

D.5 Generating initial solutions for MILP solving

We evaluate whether predicted assignments produce feasible starting solutions and how SCIP performs after receiving them. The LP-integer rule fixes all eligible integral relaxation values; the quality-weighted and family-supervised models select a set from the same allowed collection.

Problems and evaluation setting. We use 64 MIPLIB problems, split by instance and MIPLIB group into 40 training, eight validation, and 16 reserved test problems. Each has integer variables, a finite published reference objective, and at most 8,000 columns, 20,000 rows, and 100,000 nonzeros. The collection covers binary and general-integer variables, with and without continuous variables. Target families are available for 31 training and seven validation problems. The results below use all eight validation problems, including the one without a training target. They are `control30-3-2-3`, `n9-3`, `neos-2652786-brda`, `neos2`, `ns4-pr6`, `reblock354`, `supportcase14`, and `wachplan`.

Each method is evaluated with two solver seeds under total time limits of 2 and 10 seconds. This gives 16 runs per method and time limit, or 96 runs in total. All methods use the same compatibility conditions and one solver thread.

From predicted assignments to a starting solution. The selected variables are fixed in a sub-MILP, with all other domains and integrality constraints preserved. Up to 20% of the total time limit

is used to find a feasible completion. Any resulting complete solution is passed to SCIP as an initial incumbent for solving the original MILP in the remaining time. The total includes input and graph preparation, prediction, completion, and SCIP search; model loading and warm-up are completed beforehand.

Table 11 reports the number of feasible starting solutions. Family supervision succeeds in 12 of 16 runs at 10 seconds, compared with seven for quality-weighted supervision and six for the LP-integer rule; it also leads at 2 seconds.

Initial and final solution quality. Table 12 distinguishes the starting solution obtained by completion from the final solution returned by SCIP. To compare objectives across problems, let

$$g(z) = 100\frac{\max\{0, s(z - z_{\mathrm{ref}})\}}{\max(1, |z_{\mathrm{ref}}|)}$$

be the objective deviation from the published reference, with the same sign convention as above. The starting-solution score is $u = 100g/(100 + g)$; the final-solution score is $G = \min(100, g)$. Both range from 0 to 100 and are lower when the objective is closer to the reference. A score of zero means the reference objective is reached or improved upon. If no feasible solution is available within the corresponding stage, its score is 100. Each column reports the mean over all 16 runs, including such failures; the two stages use different transformations and are compared separately across methods.

Family supervision has the highest feasible-start rate and lowest starting-solution score at both time limits. Quality-weighted supervision achieves the lowest final SCIP score, tied with the LP-integer rule at 2 seconds. The benefit here is in generating feasible initial solutions.

Table 11: Feasible starting solutions obtained from predicted assignments. Each entry gives successful runs out of 16 and the corresponding success rate. The time limits cover the complete procedure, including subsequent SCIP search.

Method	2 s total limit	10 s total limit
LP-integer rule	2/16 (12.5%)	6/16 (37.5%)
Quality-weighted supervision	4/16 (25.0%)	7/16 (43.8%)
Family supervision (ours)	**6/16 (37.5%)**	**12/16 (75.0%)**

Table 12: Quality of the initial and final MILP solutions. Scores range from 0 to 100 (lower is better), with 100 assigned when no feasible solution is found. Each score uses the stage-specific definition above and is averaged over all 16 runs. Total time includes preparation, prediction, completion, and SCIP search.

Total limit	Method	Initial solution score ↓	Final solution score ↓	Total time (s)
2 s	LP-integer rule	94.17	**67.40**	1.839
	Quality-weighted supervision	95.55	**67.40**	1.850
	Family supervision (ours)	**80.28**	69.65	1.862
10 s	LP-integer rule	73.55	56.99	8.047
	Quality-weighted supervision	72.05	**56.27**	7.992
	Family supervision (ours)	**49.93**	59.30	8.009

E LIMITATIONS

Training requires available optimization certificates, and extending the approach to other set decisions requires target distributions whose normalizers and statistics can be computed tractably. The partial integer-assignment experiment provides an initial validation of this broader idea; developing it into a mature solver component requires further study. Exact KL supervision avoids basis enumeration and sampling-based estimation, but matrix factorizations and conditional-greedy decoding still incur computational and memory costs. With the current implementation and experimental hardware, instances with approximately 80,000 constraint rows approach the GPU memory limit. Scaling to larger problems therefore requires further reductions in computational and memory costs.