Deep Flow Networks

Ozan Candogan Ayoub Foussoul

Booth School of Business, University of Chicago

Motivation: Predict-then-Optimize over Integers

Predict-then-optimize: learn a surrogate $\widehat{f}_\theta$ (parameters θ) from data, then optimize it to pick a decision

$$\min_{x \in X} \widehat{f}_\theta(x).$$

- In many settings the decisions are integer (e.g. unit commitment, vehicle routing, inventory planning), so $X \subseteq \mathbb{Z}^n$.
- The surrogate $\widehat{f}_\theta$ must be both accurate and easy to optimize over $\mathbb{Z}^n$.
- Expressive models such as neural networks fit well, but optimizing them over integers is hard and scales poorly.

We introduce *Deep Flow Networks* (DFNs): expressive surrogates that admit efficient integer optimization.

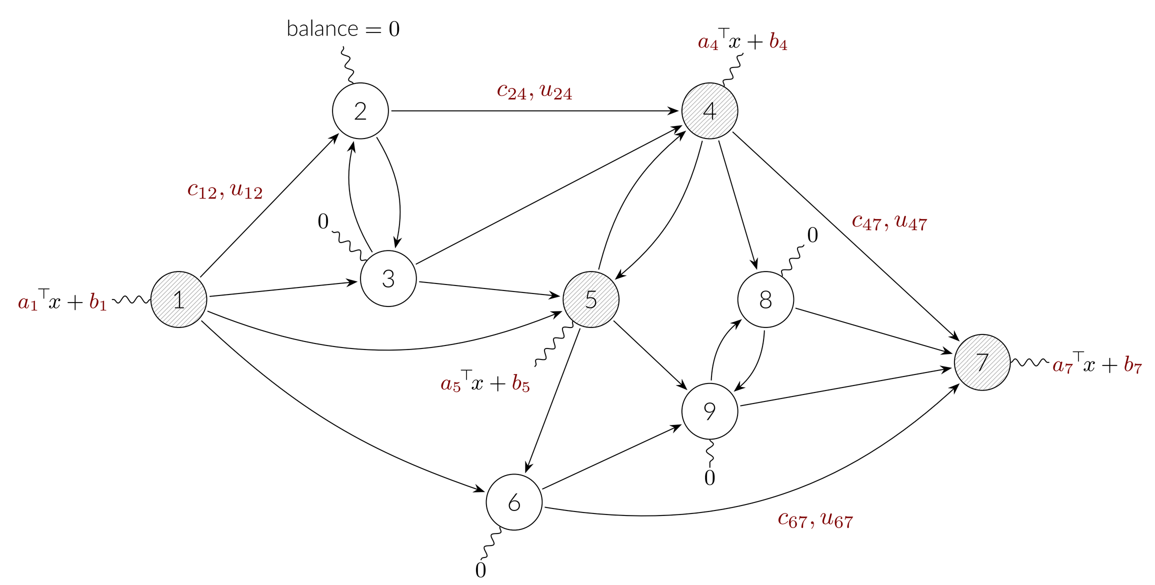
Related Literature

- The standard approach embeds a generic learned surrogate (typically a neural network) in the downstream optimization—e.g. for power systems (Kody et al. 2022), vehicle routing (Kaleem & Subramanyam 2024), and inventory control (Harsha et al. 2025)—yielding integer programs that are hard to solve.
- Tractable-by-design approximators are known only for *continuous* decisions—e.g. log-sum-exp networks (Calafiore et al. 2019) and max-affine regression (Ghosh et al. 2022), which exploit convexity.
- Convexity alone does not help over the integers—minimizing a convex function over $\mathbb{Z}^n$ is NP-hard, even for convex quadratics over $\{0,1\}^n$ (Gritzmann & Klee 1989). Tractability requires special structure—e.g. submodularity (Lovász 1983), L/M-convexity (Murota 2003), and integral convexity (Favati & Tardella 1990).

DFNs build on minimum-cost-flow value functions to give a surrogate rich enough to approximate complex data, yet tractable to optimize over $\mathbb{Z}^n$.

Deep Flow Networks

- fix a directed graph;
- each arc has a cost $c_{ij} \in \mathbb{Q}$ and capacity $u_{ij} \in \mathbb{Z}_{\geq 0}$;
- *special* nodes have a balance $a_i^\top x + b_i$ $\big(a_i \in \mathbb{Z}^n,\ b_i \in \mathbb{Z}\big)$; every other node has balance 0.



A *Deep Flow Network* function $\widehat{f}_\theta \colon \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ is the resulting minimum-cost-flow value

$$\begin{aligned} \widehat{f}_\theta(x) = \min_{\xi} \quad & c^\top \xi \\ \text{s.t.} \quad & \sum_{a \in \delta_i^+} \xi_a - \sum_{a \in \delta_i^-} \xi_a = a_i^\top x + b_i, \quad \forall (i) \text{ special}, \\ & \sum_{a \in \delta_i^+} \xi_a - \sum_{a \in \delta_i^-} \xi_a = 0, \quad \forall (i) \text{ not special}, \\ & 0 \leq \xi_a \leq u_a,\ \xi_a \in \mathbb{Z}, \quad \forall a \text{ arc}, \end{aligned}$$

where δ_i^+, δ_i^- are the arcs leaving/entering node i, and $\theta = \big(c,\ u,\ \{a_i\}_i,\ \{b_i\}_i\big)$.

Theory

Approximation power

DFNs can approximate *any* convex-extendible discrete function, e.g. submodular, M- and L-convex, and integrally convex.

Theorem (Universality)

For any convex $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, any finite $I \subset \operatorname{dom} g \cap \mathbb{Z}^n$, and any $\varepsilon > 0$, there exist parameters θ and scalars $\alpha, \beta \in \mathbb{R}$ such that

$$\max_{x \in I} \big| g(x) - (\alpha \widehat{f}_\theta(x) + \beta) \big| \leq \varepsilon.$$

Optimization tractability

Recall the classical notion of M-*convexity* (Murota 2003):

M-convexity

A function $f : \mathbb{Z}^d \to \mathbb{R} \cup \{+\infty\}$ is M-*convex* if it satisfies the following exchange property:

for all $y, y' \in \operatorname{dom} f$ and any i with $y_i > y'_i$, there is j with $y_j < y'_j$ such that

$$f(y) + f(y') \ \geq\ f(y - e_i + e_j) + f(y' + e_i - e_j)$$

As a consequence, an M-convex function can be minimized in time polynomial in the input.

Let $A \in \mathbb{Z}^{d \times n}$ be the matrix with rows a_i, $k = d - \operatorname{rank}(A)$, and

$$\Delta(A) = 2\,(2k)^{2k}\ \operatorname{pdet}(A).$$

Theorem (Optimization complexity)

DFN functions $\widehat{f}_\theta$ satisfy the following *generalized exchange* property:

for all $x, x' \in \operatorname{dom} \widehat{f}_\theta$ with $Ax \neq Ax'$, there is $\tilde{x} \in \mathbb{Z}^n$ with $0 < \|A\tilde{x}\|_1 \leq \Delta(A)$ and

$$\widehat{f}_\theta(x) + \widehat{f}_\theta(x') \ \geq\ \widehat{f}_\theta(x - \tilde{x}) + \widehat{f}_\theta(x' + \tilde{x})$$

As a consequence, a DFN function can be minimized in time polynomial in $d^{\Delta(A)}$ and the rest of the input.

Proof sketch (special case $k = 0$, i.e. A full rank).

- Take x, x' with $Ax \neq Ax'$. By the flow decomposition theorem, the balances $Ax + b$ can be transformed into $Ax' + b$ (and back) by routing unit flows along paths $i_\ell \to j_\ell$ between special nodes; these correspond to unit swaps $e_{i_\ell} - e_{j_\ell}$ of the balances.
- Applying any subset of these swaps gives two feasible flows of total cost $\leq \widehat{f}_\theta(x) + \widehat{f}_\theta(x')$.
- A subset T of these swaps corresponds to a move $\tilde{x}$ in x-space iff $\sum_{\ell \in T} (e_{i_\ell} - e_{j_\ell}) = A\tilde{x}$ (i.e. in $A\mathbb{Z}^n$).
- A vector $v \in A\mathbb{Z}^n$ if and only if $U^{-1}v \equiv 0$ in the group $\mathbb{Z}/d_1\mathbb{Z} \times \cdots \times \mathbb{Z}/d_d\mathbb{Z}$, where $A = U\operatorname{diag}(d_1, \ldots, d_d)V$ is the Smith normal form of A.
- This group has $d_1 \cdots d_d \leq \operatorname{pdet}(A)$ elements, so some subset of $\leq \operatorname{pdet}(A)$ swaps lands in $A\mathbb{Z}^n$.
- The resulting $\tilde{x}$ is the exchange, with $\|A\tilde{x}\|_1 = 2|T| \leq 2\operatorname{pdet}(A) = \Delta(A)$.

Implementation

A training recipe — *chosen by experimentation*

- *Graph.* Layered graph; special nodes sit on the boundary layers.
- *Forward/backward pass.* The forward pass evaluates $\widehat{f}_\theta(x)$ with a specialized min-cost-flow solver (LEMON); the primal/dual solutions give subgradients w.r.t. θ.
- *Learning u, A, b via STE.* These integer parameters are learned by straight-through estimation (Bengio et al. 2013): round in the forward pass, replace rounding with identity in the backward pass.
- *Keeping A near identity.* A's rows are initialized cycling through $e_1, \ldots, e_n$, so A starts near identity I — the M-convex regime where optimization is easiest.

Experiments

- *Data.* A supervised dataset of K pairs (x, y): inputs $x \in \mathbb{Z}^n$, labels $y = (x - x^\star)^\top Q\,(x - x^\star)$ from a convex quadratic ($Q \succ 0$).
- *Downstream optimization.* $\min_{x \in \mathbb{Z}^n} \widehat{f}_\theta(x)$ s.t. $x_{\min} \leq x \leq x_{\max}$, $\mathbf{1}^\top x = B$.
- *Models.* DFN → structured MILP; MLP (ReLU) → big-M MILP; LSET (LogSumExp) → mixed-integer convex. All with equal parameter counts, solved with Gurobi.
- *Results.* Four scales (n, K, #params): Small (8, 1k, 5k), Medium (16, 2k, 25k), Large (24, 4k, 50k), XLarge (32, 6k, 100k); mean ± s.e. over 3 seeds; solver capped at 1h.

	Optimization time (s)				*Test MSE* ($\times 10^{-3}$)			
	Small	Medium	Large	XLarge	Small	Medium	Large	XLarge
DFN	0.07±0.01	1.2±0.2	15.7±1.6	815±241	10.8±7.0	3.3±0.5	4.0±0.9	3.2±0.2
MLP	1.3±0.5	55±35	3563±37	3600±0	6.7±1.0	12.7±1.5	20.7±0.9	16.4±2.5
LSET	1560±1035	2657±944	3601±0.2	3165±435	36.6±4.9	86.2±8.1	141.7±7.0	126.5±14.3

- *Other ground truths.* We observe similar results on data generated from other ground-truth functions, including minimum-cost resource allocation and multiple-depot vehicle scheduling (MDVSP).
- *Evolution of A and optimization time.* As training proceeds, A drifts from its near-identity init. Re-solving the problem with A set to each checkpoint (other parameters fixed at their final values): the solve slows as $\log\operatorname{pdet}(A)$ grows — confirming our theoretical findings:

epoch	200	300	400	500	600	700	800	900	1000
$\log\operatorname{pdet}(A)$	15.0±1.3	15.0±1.3	20.2±1.0	33.4±1.6	41.3±1.0	45.4±0.8	48.0±0.7	49.9±0.5	51.4±0.5
Opt. time (s)	0.18±0.01	0.20±0.02	0.22±0.02	0.31±0.04	0.47±0.06	0.60±0.12	1.35±0.20	1.18±0.18	3.24±1.06

Takeaways

✓ **Theory:** *DFNs* are universal approximators of convex-extendible functions. Optimization complexity governed by $\Delta(A)$.

✓ **Practice:** trained end-to-end, they match or beat neural baselines while optimizing orders of magnitude faster. Solve time grows as A drifts from its near-identity initialization, confirming our theory.

References

Bengio, Y., Léonard, N., & Courville, A. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint* arXiv:1308.3432.

Calafiore, G. C., Gaubert, S., & Possieri, C. (2019). Log-sum-exp neural networks and posynomial models for convex and log-log-convex data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(3), 827–838.

Favati, P., & Tardella, F. (1990). Convexity in nonlinear integer programming. *Ricerca Operativa*, 53, 3–44.

Ghosh, A., Pananjady, A., Guntuboyina, A., & Ramchandran, K. (2022). Max-affine regression: Parameter estimation for Gaussian designs. *IEEE Transactions on Information Theory*, 68(3), 1851–1885.

Gritzmann, P., & Klee, V. (1989). On the 0–1-maximization of positive definite quadratic forms. In *Operations Research Proceedings 1988* (pp. 222–227). Springer.

Harsha, P., Jagmohan, A., Kalagnanam, J., Quanz, B., & Singhvi, D. (2025). Deep policy iteration with integer programming for inventory management. *Manufacturing & Service Operations Management*, 27(2), 369–388.

Kaleem, W., & Subramanyam, A. (2024). Neural embedded mixed-integer optimization for location-routing problems. *arXiv preprint* arXiv:2412.05665.

Kody, A., Chevalier, S., Chatzivasileiadis, S., & Molzahn, D. (2022). Modeling the AC power flow equations with optimally compact neural networks: Application to unit commitment. *Electric Power Systems Research*, 213, 108282.

Lovász, L. (1983). Submodular functions and convexity. In *Mathematical Programming: The State of the Art* (pp. 235–257). Springer.

Murota, K. (2003). *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics (SIAM).

Based on Candogan, O. & Foussoul, A. (2026). Deep Flow Networks. *International Conference on Machine Learning* (ICML 2026), Seoul, South Korea, July 6–11, 2026.