THE LOSS IS ONLY HALF OF THE STEP: RETHINKING DECISION LOSSES IN DECISION-FOCUSED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Decision-focused learning replaces the squared error of a cost predictor with a decision loss and treats the choice of this loss as the main design decision. We find that the squared error already moves each prediction in a safe direction. Moving a prediction toward its realized cost never increases the regret of that sample when the objective is affine in the cost, and every task in our comparison has such an objective. Regret can rise only through the parameter update, which moves all predictions together and can carry a prediction off its path to the realized cost. Accordingly, we split a training step into the displacement that the loss assigns to each prediction and the transport by which one parameter update moves all predictions. Decision losses change only the displacement, so they correct a direction that is already safe. We introduce RIFT (regression in function space with trust regions), which fixes the transport as a damped Gauss-Newton step and accepts any displacement. RIFT with the regression displacement is not significantly worse than any of 22 methods on five tasks of PredictiveCO-Benchmark, and it trains 7 to 27 times faster than the strongest decision loss. Replacing the regression targets by the rescaled direction of the SPO+ surrogate or of a perturbed Fenchel-Young loss raises regret by 1.11 to 3.04 points on two knapsack tasks. Replacing the transport by gradient steps with a validated step size changes regret by at most 0.07 points, and both transports reach a regret 1.49 points below the regression baseline of the benchmark on energy prices. On these tasks, regression gains from a better transport and not from a decision loss.

1 INTRODUCTION

Many decisions in scheduling, routing and resource allocation are combinatorial programs whose costs are unknown when the decision is made. Predictive combinatorial optimization learns a model h_θ that predicts the costs from observed features x, and a solver returns the decision that is optimal for the predicted costs (Elmachtoub & Grigas, 2022; Mandi et al., 2024). The model is judged by the regret of this decision under the realized costs. Because the solver maps predictions to decisions through a piecewise constant function, a small prediction error can change the decision, while a large error may leave it unchanged.

A squared error ignores this structure and treats an error that changes the decision like one that leaves it unchanged. Decision-focused learning replaces it by decision losses, whose gradients carry information from the solver or from a surrogate of its regret. Examples include the SPO+ surrogate (Elmachtoub & Grigas, 2022), perturbed Fenchel-Young (PFY) losses (Berthet et al., 2020), the differentiation of blackbox solvers (Vlastelica Pogančić et al., 2020), ranking and contrastive losses (Mulamba et al., 2021; Mandi et al., 2022) and learned local losses (Shah et al., 2022). More recent variants build on targets smoothed over nearest neighbors (kNN targets) (Schutte et al., 2024), normal cone alignment (Tang & Khalil, 2024a), directional gradients (Huang & Gupta, 2024) and tangent space projections of the prediction error (Lee et al., 2026). A benchmark of both families reports decision losses ahead of regression on seven of its eight problems, with no design choice that wins consistently (Geng et al., 2024).

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.

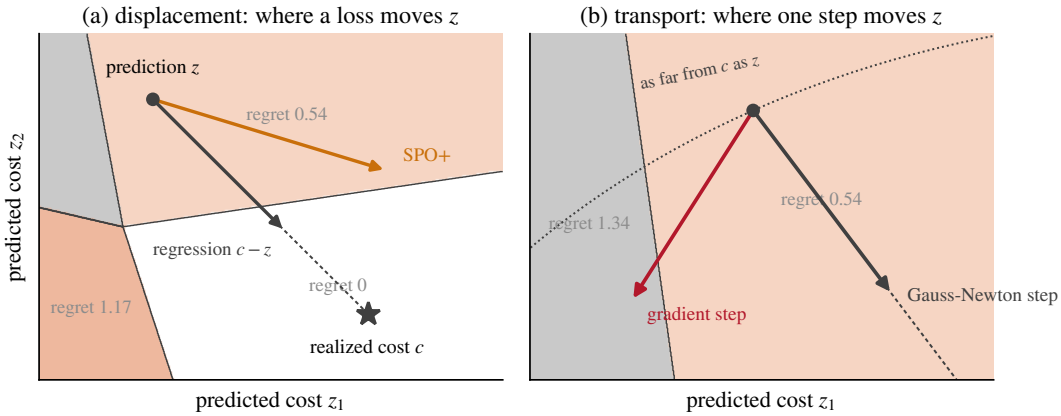

Figure 1: Displacement and transport for one prediction z of a small program with four decisions. The shading gives the regret under the realized cost c. (a) The regression displacement $c - z$ points along a segment on which the regret never increases (Proposition 1). SPO+ rotates this direction toward a difference of two decisions, and a decision loss changes only this part of the step. (b) Two steps of the same length, both computed from $c - z$. A gradient step with an anisotropic Jacobian moves z closer to c (inside the dotted arc) but into a decision with regret 1.34. A Gauss-Newton step follows the segment and keeps the regret at 0.54. RIFT changes this part of the step.

The squared error, however, already moves each prediction in a safe direction. Moving a prediction along the segment toward its realized cost never increases the regret of that sample when the objective is affine in the cost (Figure 1a and Proposition 1). Hence the regret of a sample can rise only when its prediction leaves the segment or moves past its realized cost. A gradient step can do both, because one update of the shared parameters moves all predictions together. The update rotates the move of each prediction and mixes it with the moves of other samples. As a result, a gradient step can carry a prediction into a worse decision even while it reduces the prediction error (Figure 1b).

This harm comes from the parameter update, which a benchmark comparison changes together with the loss and the model selection. A gain in such a comparison need not come from the loss, and we isolate the loss by splitting a first order training step into two parts. The *displacement* of a sample is the negative gradient of the loss at its prediction and gives the direction in which the loss moves this prediction. One update of the shared parameters combines the displacements of all samples, and the *transport* describes where this update actually moves each prediction. A decision loss trained by gradient descent changes only the displacement, so the loss sets only half of the step. *It corrects a displacement that is already safe for its own sample and keeps the transport of regression.* Such a correction can still lower regret, for example when the corrected direction crosses a decision boundary that the regression direction reaches only after a longer step (Remark A.2). Consequently, we measure the effect of each part while the other stays fixed.

This measurement requires a training method whose transport does not depend on the displacement. We introduce RIFT (regression in function space with trust regions) for this purpose. RIFT turns any displacement into targets and moves the predictor toward them with a damped Gauss-Newton step, which follows the targets as closely as the predictor allows within a trust region. With the regression displacement, the targets are the realized costs. The parameter update then calls no solver, and the solver only selects the stage with the lowest validation regret.

RIFT with the regression displacement is not significantly worse than any of 22 methods on five tasks of PredictiveCO-Benchmark (Geng et al., 2024). No method is both faster and significantly better on any task, and PFY with kNN targets, the strongest decision loss in this comparison, needs 7 to 27 times the training time of RIFT. Our contributions are the following.

- Moving a prediction toward its realized cost never increases its regret when the objective is affine in the cost, so the squared error moves every prediction in a safe direction. Regret can rise only through the transport of a training step, and decision losses change only the displacement (Section 3).
- RIFT transports any displacement with a damped Gauss-Newton step. With the regression displacement, it calls no solver in its update and needs no search over step sizes (Section 4).

- Controlled exchanges on two knapsack tasks show that regression gains from a better transport and not from a decision loss. Replacing the regression targets by a decision loss direction raises regret, whereas replacing the Gauss-Newton step by validated gradient steps changes regret by at most 0.07 points. Both transports reach a lower regret than the regression baseline of the benchmark on energy prices, and neither wider predictors nor more exact solves let a correction lower regret (Section 5).

2 PROBLEM SETTING

Each instance has features $x \in \mathcal{X}$ and costs $c \in \mathbb{R}^d$. The decision w belongs to a feasible set $\mathcal{W}$, and the objective $F(c, w) = g(w) + c^\top \phi(w)$ is affine in the cost. Linear programs have $g = 0$ and $\phi(w) = w$. The energy scheduling task of Section 5 also has this form, because its cost is linear in a schedule derived from the decision. The portfolio task adds a risk term that does not depend on the cost, and g absorbs it. Every task of our comparison has an objective of this form. For a predicted cost z, the solver returns $w^\star(z) \in \arg\min_{w \in \mathcal{W}} F(z, w)$ under a fixed tie breaking rule, and the regret under the realized cost is

$$\mathrm{R}(z, c) = F\big(c, w^\star(z)\big) - \min_{w \in \mathcal{W}} F(c, w) \ \geq 0. \tag{1}$$

Maximization problems are handled by a change of sign. A predictor h_θ with p parameters produces $z_i = h_\theta(x_i)$ for the training pairs (x_i, c_i), $i = 1, \ldots, N$. We report the relative regret in percent, which divides the mean regret over the test instances by the mean absolute optimal objective. Appendix A.1 collects the notation.

3 REGRESSION CAN RAISE REGRET ONLY THROUGH THE TRANSPORT

3.1 MOVING TOWARD THE REALIZED COST IS SAFE

Proposition 1 (Monotone objective along the segment)**.** *Let $F(c, w) = g(w) + c^\top \phi(w)$ and $z_t = (1 - t)z + tc$ for $t \in [0, 1]$. For $0 \leq s < t \leq 1$ and any minimizers $w_s \in \arg\min_{w \in \mathcal{W}} F(z_s, w)$ and $w_t \in \arg\min_{w \in \mathcal{W}} F(z_t, w)$, the true objectives satisfy $F(c, w_t) \leq F(c, w_s)$. In particular, $t \mapsto \mathrm{R}(z_t, c)$ is nonincreasing on $[0, 1]$ for every tie breaking rule.*

Adding the optimality conditions of w_s and w_t gives $(t - s)\,(c - z)^\top (\phi(w_s) - \phi(w_t)) \geq 0$, and the proposition follows from this inequality (Appendix A.2). Consequently, every decision boundary that the segment crosses leads to a decision that is at least as good under the realized cost. The argument needs neither convexity, nor a finite feasible set, nor a unique minimizer. The guarantee ends at $t = 1$, because a prediction that overshoots its realized cost can reach a worse decision (Remark A.1). A numerical check found no violation on 100,000 random problems with finite feasible sets and on 20,000 problems with quadratic and nonlinear g, and the segments of the finite problems crossed 104,842 decision boundaries.

3.2 A TRAINING STEP MOVES ALL PREDICTIONS TOGETHER

Proposition 1 concerns a single prediction, whereas the parameters of the predictor are shared by all samples. For a loss $L(z_1, \ldots, z_N)$ on the predictions, the displacement of sample i is $u_i = -\partial L / \partial z_i$. A gradient step changes the parameters by $\Delta = \eta \sum_i J_i^\top u_i$ with $J_i = \partial h_\theta(x_i)/\partial \theta$, and to first order the predictions then move by

$$z_i \ \mapsto \ z_i + \eta \sum_j K_{ij} u_j, \qquad K_{ij} = J_i J_j^\top. \tag{2}$$

The loss appears in equation 2 only through the displacements, and the kernel blocks K_{ij} determine how the displacements are transported. The stacked predictions move by $\mathcal{T}u$, where $\mathcal{T} = \eta K$ is the transport of a gradient step. Regression uses the displacement $u_i = c_i - z_i$, which is safe for each sample by Proposition 1. Any increase of regret under regression must then come from the transport. The block K_{ii} rotates the displacement of sample i unless the block is a multiple of the identity, and the blocks K_{ij} with $j \neq i$ add the displacements of other samples. A gradient step on the squared error increased regret in 1,036 of 20,000 random instances with two costs, four candidate decisions

and a linear predictor, and 982 of these steps also reduced the prediction error (Appendix A.6 shows one of them).

Decision losses change the same step only through their displacements, which have closed forms when the objective has no cost independent term. SPO+ has the displacement $u_i = 2\big(\phi(w^\star(2z_i - c_i)) - \phi(w^\star(c_i))\big)$, and a PFY loss has the displacement $u_i = \mathbb{E}_\varepsilon\big[\phi(w^\star(z_i + \sigma\varepsilon))\big] - \phi(w^\star(c_i))$ (Lemma A.1). Both losses keep the gradient transport and add the correction $u_i - (c_i - z_i)$ to the regression displacement. The SPO+ displacement never forms an obtuse angle with $c_i - z_i$ (Lemma A.2), so its correction rotates the regression direction toward a difference of two decisions by at most 90 degrees. Because the regression direction is already safe for the sample itself, this rotation has no harmful direction to repair, and we measure its effect on regret in Section 5.3.

3.3 A DAMPED GAUSS-NEWTON TRANSPORT

A training method can instead change the transport so that each prediction moves closer to its own target $y_i = z_i + u_i$.

Proposition 2 (Output motion of a damped Gauss-Newton step)**.** *Stack the displacements into $u \in \mathbb{R}^{Nd}$ and the Jacobians into $J \in \mathbb{R}^{Nd\times p}$. The damped Gauss-Newton step $\Delta_\lambda = \arg\min_\Delta \|J\Delta - u\|^2 + \lambda\|\Delta\|^2$ with $\lambda > 0$ produces the linearized output motion $J\Delta_\lambda = JJ^\top(JJ^\top + \lambda I)^{-1}u$. If J has full row rank, which requires $p \geq Nd$, this motion converges to u as $\lambda \to 0$, so every prediction reaches its target to first order. If $\operatorname{rank} J < Nd$, the limit is the orthogonal projection of u onto the range of J, and the component of the motion of a sample along its own segment is not controlled.*

The motion in Proposition 2 corresponds to the transport $\mathcal{T} = K(K + \lambda I)^{-1}$. The linearized motion reaches the targets under full row rank and vanishing damping, so targets on the segments carry Proposition 1 over to every sample. The Gauss-Newton step in Figure 1b stays on the segment for this reason. A damped and truncated step reaches its targets only approximately, but the new prediction still selects the decision of its target when it stays inside the decision margin of that target (Lemma A.3). The benchmark predictors are far from the exact regime, because 852 or 321 parameters must move 6,400 to 26,400 predicted costs in one step, and we widen them in Section 5.5 to test whether this gap matters. In this rank deficient regime, gradient and Gauss-Newton transports are closer than their formulas suggest.

Proposition 3 (Transports as spectral filters)**.** *Let $K = \sum_k \mu_k v_k v_k^\top$ with $\mu_k \geq 0$, and hold u fixed while the outputs move to first order. The damped Gauss-Newton step produces the motion $\sum_k f_\lambda(\mu_k)(v_k^\top u)\, v_k$ with $f_\lambda(\mu) = \mu/(\mu + \lambda)$, and T full batch gradient steps of size η on $\frac{1}{2}\|u - J\Delta\|^2$, started at $\Delta = 0$, produce the motion $\sum_k g_{\eta,T}(\mu_k)(v_k^\top u)\, v_k$ with $g_{\eta,T}(\mu) = 1 - (1 - \eta\mu)^T$. If $\eta\mu_k \leq 1$ for all k, both filters take values in $[0, 1]$, vanish at $\mu = 0$ and are nondecreasing in μ, with $f_\lambda(\lambda) = 1/2$ and $g_{\eta,T}(1/(\eta T)) \geq 1 - e^{-1}$.*

By Proposition 3, both transports pass the directions of K with large eigenvalues and suppress the others. The two filters differ mainly in the shape of the cutoff, and the damping λ plays the role of $1/(\eta T)$. A similar filter need not give a similar regret, because a decision changes only when a prediction crosses a boundary. We compare the two transports directly in Section 5.4. A transport that follows noisy targets closely also reproduces their noise, because the expected distance of the new predictions from the noise free costs contains a term that grows with the trace of the squared filter (Remark A.4). The deeper solve of Section 5.5 tests this effect.

4 REGRESSION IN FUNCTION SPACE WITH TRUST REGIONS (RIFT)

RIFT uses the step of Proposition 2 as a transport that accepts any displacement, and it trains in stages (Algorithm 1). Each stage sets the targets $y_i = z_i + u_i$ from the chosen displacement, computes the damped Gauss-Newton step with at most ten conjugate gradient iterations, and evaluates the validation regret of the new parameters. The conjugate gradient iterations use Jacobian vector products and never form J. The damping $\lambda = \lambda_0\, \kappa\, \operatorname{tr}(J^\top \Omega J)/p$ scales with the Jacobian, where $\lambda_0 = 10^{-3}$, Ω holds the sample weights and four Rademacher probes estimate the trace. Because Proposition 2 describes the linearized motion, a Levenberg-Marquardt factor $\kappa \geq 1$ (Levenberg, 1944; Marquardt, 1963) keeps the step in the region where the linearization holds. The step is accepted when the ratio γ between the actual and the predicted reduction of the weighted squared

Algorithm 1 One RIFT stage. With the regression displacement, only the validation step calls the solver.

1: $z_i \leftarrow h_\theta(x_i)$ and $y_i \leftarrow z_i + u_i$ for every training sample ▷ displacement, $u_i = c_i - z_i$ by default
2: $\omega_i \leftarrow 1$ or the Huber weights of equation 3, and $\Omega \leftarrow \mathrm{diag}(\omega_i I_d)$
3: **for** attempt $a = 0, 1, 2, 3$ **do**
4: $\quad\Delta \leftarrow$ at most ten conjugate gradient iterations on $(J^\top \Omega J + \lambda I)\Delta = J^\top \Omega u$
5: $\quad\theta \leftarrow \theta + \Delta$, and $\gamma \leftarrow$ actual over predicted reduction of $\sum_i \omega_i \|y_i - h_\theta(x_i)\|^2$
6: $\quad$**if** $\gamma \geq 0.25$ or $a = 3$ **then** leave the loop **else** undo the step and set $\kappa \leftarrow 4\kappa$
7: **if** $\gamma > 0.75$ **then** $\kappa \leftarrow \max(\kappa/2, 1)$
8: compute the validation regret of h_θ with the solver, and keep θ if it is the lowest so far

residual is at least 0.25. Otherwise, κ grows by a factor of four and the step is solved again. A stage makes at most four attempts and keeps the last one. Training stops after 30 stages or after 10 stages without a lower validation regret, and it returns the stage with the lowest validation regret. These settings are the same for every task (Appendix B.3).

Displacements. The default displacement $u_i = c_i - z_i$ makes RIFT a regression method. The same transport also accepts the SPO+ and PFY displacements of Section 3.2, which we rescale to the length of the segment from z_i to c_i so that they differ from the regression displacement in direction and not in length. Four further corrections add decision structure to the targets. *Partial moves* place the target at the first point of the segment whose decision lowers the current regret by at least a quarter. *Normal cone targets* require the prediction to lie inside the region of the target decision with a margin. *Conditional mean targets* replace c_i by an estimate of $\mathbb{E}[c \mid x_i]$, because the risk neutral decision depends on the costs only through this mean (Hu et al., 2022). *Decision quotients* remove every direction of u_i that no pair of feasible decisions can distinguish. Appendix B.4 gives the details, including a factorial design that combines the corrections.

Huber weights. The squared residual lets a sample influence the step in proportion to its distance from the target. The pseudo Huber penalty $\psi_\delta(r) = \delta^2\big(\sqrt{1 + (r/\delta)^2} - 1\big)$ bounds this influence (Huber, 1964; Charbonnier et al., 1994), and RIFT approximates it with three rounds of iteratively reweighted least squares. Each round solves the damped system with the weights

$$\omega_i = \big(1 + \|r_i\|^2/\delta^2\big)^{-1/2}, \qquad \delta = \delta_0 \cdot \mathrm{median}_j \, \|r_j\|, \tag{3}$$

where r_i is the residual that the previous round leaves for sample i. The first round starts from $r_i = y_i - z_i$, so with $\delta_0 = 1$ the samples whose required motion exceeds the median receive the smallest weights.

Computation per stage. A stage needs one forward pass, on average 33 Jacobian vector and vector Jacobian products, and one solver call per validation instance. The parameter update calls no solver when RIFT uses the regression displacement.

5 Experiments

5.1 Setup

We use the data, splits, predictors, solvers and regret of PredictiveCO-Benchmark (Geng et al., 2024) on five of its tasks. *Knapsack with generated costs* has 20 items and 320/80/200 training, validation and test instances. *Knapsack with energy prices* has 48 items, one per half hour of a day, and the value of each item is the electricity price in its half hour. *Energy scheduling* assigns jobs to machines over the same 48 slots. *Bipartite matching* predicts the edges of citation graphs, and *portfolio* selects stocks under a quadratic risk term that does not depend on the costs. All predictors are the perceptrons of the benchmark with one hidden layer of 32 units, and Appendix B.1 gives the task sizes.

We compare RIFT with the eleven methods of the benchmark, which include its regression baseline MSE. PFY and the kNN targets for SPO+ and for PFY extend this set, and eight further methods run on generated knapsack only (Table 1 and Appendix B.2). Every baseline keeps the training loop of the benchmark, which runs Adam at learning rate 0.01 for at most 300 epochs and stops after 40

Table 1: Relative regret on test in percent (mean and standard deviation over seeds). Shading marks the leading group of each task, the methods whose regret is not significantly higher than that of any other method, and ↓ marks a significantly higher regret than RIFT (paired t tests, $p < 0.05$). No method has a significantly lower regret than RIFT on any task. A superscript number gives the number of seeds of a cell that differs from its column. The mark ^{b} denotes BCE, which the benchmark uses as the regression loss for matching. A slash marks a loss whose definition assumes an objective $c^\top w$, which the risk term of portfolio rules out. A dash marks a combination that was not evaluated (Appendix B.2).

Method	Knapsack, generated 5 seeds	Knapsack, energy prices 5 seeds	Energy scheduling 5 seeds	Bipartite matching 3 seeds	Portfolio 3 seeds
RIFT (ours)	5.98 ±0.19	7.60 ±0.27	1.89 ±0.17	92.41 ±1.25	648 ±85
MSE	6.08 ±0.85	9.09 ±0.44$^{\downarrow}$	2.08 ±0.42	93.54 ±0.85^{b}	660 ±79
SPO+ (Elmachtoub & Grigas, 2022)	5.88 ±0.50	8.69 ±0.38$^{\downarrow}$	1.84 ±0.18	92.41 ±2.63	616 ±55
PFY (Berthet et al., 2020)	6.04 ±0.33	8.57 ±1.10	–	–	/
SPO+, kNN targets (Schutte et al., 2024)	5.73 ±0.67	9.47 ±0.43$^{\downarrow}$	1.90 ±0.13	–	/
PFY, kNN targets (Schutte et al., 2024)	5.60 ±0.56	7.93 ±0.57	1.69 ±0.18^{4}	–	/
DBB (Vlastelica Pogančić et al., 2020)	28.25 ±3.77$^{\downarrow}$	28.32 ±16.18$^{\downarrow}$	11.92 ±7.99	92.58 ±0.76	749 ±23
Identity gradient (Sahoo et al., 2023)	35.39 ±5.23$^{\downarrow}$	30.40 ±16.17$^{\downarrow}$	12.39 ±6.74$^{\downarrow}$	92.75 ±0.48	814 ±19
DFL (Wilder et al., 2019)	26.43 ±4.15$^{\downarrow}$	15.16 ±11.97	10.36 ±6.17$^{\downarrow}$	93.08 ±0.22	752 ±19
cvxpylayers (Agrawal et al., 2019)	28.80 ±4.96$^{\downarrow}$	28.54 ±16.30$^{\downarrow}$	–	93.14 ±1.02	779 ±30
NCE (Mulamba et al., 2021)	12.21 ±1.55$^{\downarrow}$	11.70 ±1.49$^{\downarrow}$	1.96 ±0.14	91.52 ±0.69	991 ±78$^{\downarrow}$
Pointwise LTR (Mandi et al., 2022)	8.66 ±2.74	15.24 ±8.84	5.40 ±2.48$^{\downarrow}$	92.85 ±0.94	642 ±53
Pairwise LTR (Mandi et al., 2022)	7.58 ±0.83$^{\downarrow}$	8.69 ±0.25$^{\downarrow}$	1.86 ±0.26	92.64 ±2.20	639 ±59^{2}
Listwise LTR (Mandi et al., 2022)	5.91 ±0.31	8.00 ±0.49	2.00 ±0.32	94.01 ±0.84	681 ±11
LODL (Shah et al., 2022)	26.85 ±3.18$^{\downarrow}$	11.96 ±1.61$^{\downarrow}$	3.68 ±0.86$^{\downarrow}$	92.23 ±0.57	–

epochs without a lower validation regret. RIFT uses one setting for all tasks, so no method is tuned per task. Every method is evaluated once on test at the epoch or stage with the lowest validation regret. The first three tasks use five seeds and the last two use three. We compare methods by the paired t test over seeds at $p < 0.05$, and Appendix B.6 adds the Holm correction for multiple comparisons. All runs use one thread of the same machine, an AMD EPYC 7R13 processor with 16 virtual cores and 128 GB of memory, and no run uses a GPU.

5.2 RIFT REACHES THE REGRET OF THE STRONGEST DECISION LOSSES AT A FRACTION OF THEIR TRAINING TIME

Table 1 compares RIFT with the regression displacement against every method evaluated on the five tasks. Several methods have a lower mean regret than RIFT on some task, but the regret varies across seeds by as much as these gaps. No method has a significantly lower regret than RIFT on any task. PFY with kNN targets is the strongest baseline on the three tasks with five seeds. The mean paired difference of RIFT from this method over their common seeds is $+0.38$ points on generated knapsack $[-0.16, +0.92]$, -0.34 on knapsack with energy prices $[-1.24, +0.57]$ and $+0.18$ on energy scheduling $[-0.14, +0.50]$, with a positive value for a higher regret of RIFT. RIFT has a smaller spread across seeds than this method on all three tasks and the smallest spread of all methods on generated knapsack. Conversely, RIFT is significantly better than 14 of the 22 methods on generated knapsack, 9 of 14 on energy prices and 4 of 12 on energy scheduling. Most of these differences remain significant after the Holm correction (Appendix C.1).

Single means do not separate the leading methods, so we summarize each task by its leading group. This group holds every method whose regret is not significantly higher than that of any other method, and Table 1 shades it. RIFT is the only method in the leading group of all five tasks, and PFY with kNN targets is in it on the three tasks where it runs. With three seeds, the leading groups of bipartite matching and portfolio hold 10 of 12 and 6 of 11 methods. RIFT also has the smallest worst-case gap to the lowest mean of a task among the 10 methods evaluated on all five tasks (Figure 2i). Its regret lies at most 11.6% above that mean, against 14.3% for SPO+ and 18.3% for listwise LTR.

Figure 2a–e adds the training time on all five tasks. No method is both faster than RIFT and significantly better on any task, and RIFT is the fastest method of the leading group on generated knapsack, energy prices and bipartite matching. PFY with kNN targets needs 27, 17 and 7 times the training

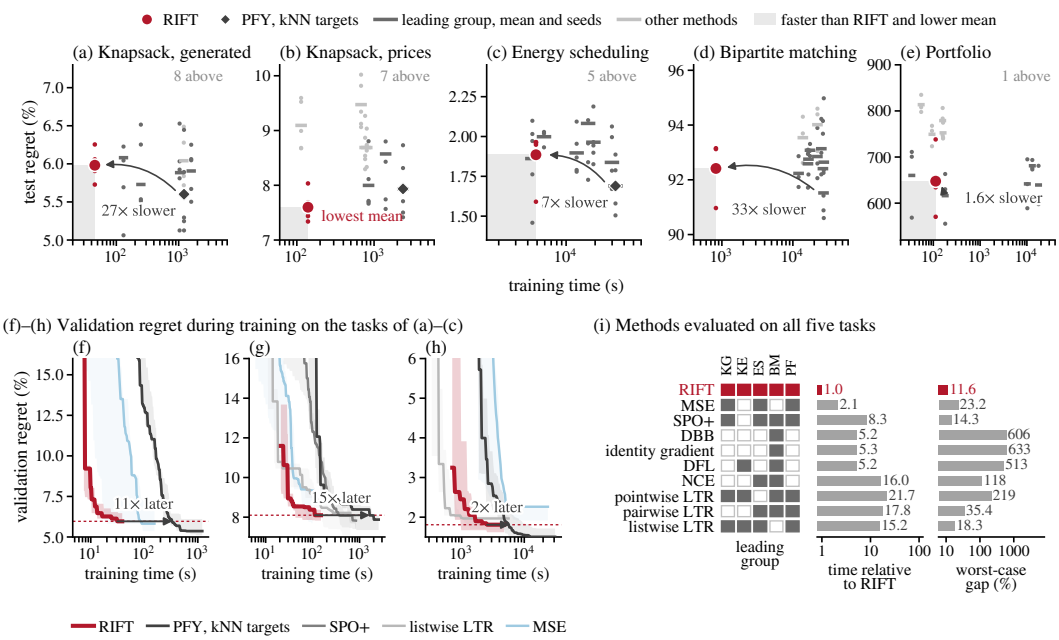

Figure 2: Regret and training time on the five tasks. (a)–(e) Test regret against training time, with the seeds of a method as dots and their mean as a bar. Dark gray marks the leading group (Table 1), and an arrow runs from the lowest mean regret of a task to RIFT. (f)–(h) Median validation regret of the best model so far, with the interquartile range over seeds. (i) Leading groups, geometric mean of the training time relative to RIFT, and largest gap to the lowest mean of a task.

time of RIFT, and no compared method trains faster than RIFT on generated knapsack or bipartite matching (Appendix C.2). The gap holds throughout training (Figure 2f–h) and comes from the parameter update. RIFT with the regression displacement solves no training instance in this update, whereas PFY with kNN targets solves perturbed problems for every training instance in every epoch.

5.3 Replacing the regression targets raises regret

Table 1 compares methods that differ in every part of their training step. Figure 3a instead changes one part of the RIFT step and keeps the other, and its three blocks change the displacement, the transport or both. The rows without an entry for energy scheduling come from exchanges on the two knapsack tasks, the fastest of the three tasks with five seeds, where a RIFT run takes 45 and 139 seconds against 4,866 seconds on energy scheduling.

The first two rows keep the Gauss-Newton transport and replace the regression targets by the rescaled SPO+ or PFY direction. The SPO+ direction raises regret by 1.11 points on generated knapsack and by 1.81 points on energy prices, and the PFY direction raises it by 3.04 and 2.57 points. All four increases are significant. Each rotated target leaves the segment and gives up the guarantee of Proposition 1, which the regression targets keep. SPO+ keeps its displacement within 90 degrees of the regression direction (Lemma A.2), and even this moderate rotation toward a difference of decisions raises regret on both tasks. PFY satisfies no such bound, and its displacement can point away from the realized cost for samples whose decision is already correct (Remark A.3).

Partial moves and normal cone targets, in contrast, keep the targets on or near the segment. Their effects on regret are smaller, but neither lowers the mean regret significantly. Normal cone targets even raise it on energy prices. The remaining rows of the block apply the decision quotient or replace c_i by an estimate of $\mathbb{E}[c \mid x_i]$, fully or in part. None of these four variants changes regret significantly on any of the three tasks, and Appendix C.3 reports all 28 variants of the factorial design.

5.4 Validated gradient steps match the Gauss-Newton transport

The second block of Figure 3a keeps the regression targets and changes the step. Gradient steps within the same stages filter the same kernel as the Gauss-Newton step, and a validated damping or

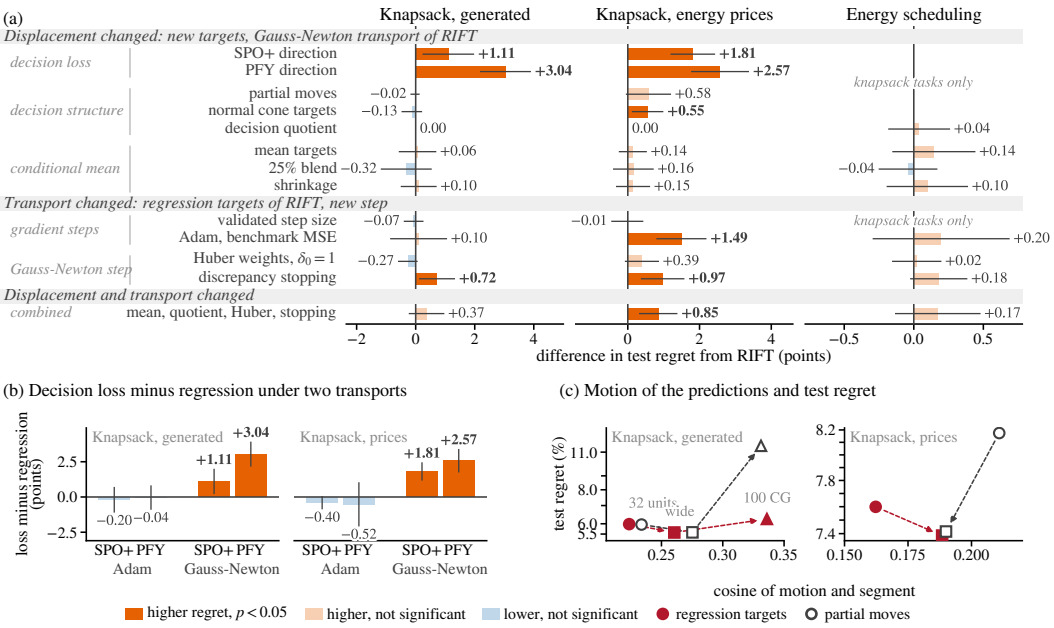

Figure 3: Changes of one part of the RIFT step. (a) Mean paired difference in test regret from RIFT over five seeds, with 95% t intervals, for the rows that Appendix B.4 defines. Orange marks a higher and blue a lower regret, and a light bar is not significant (paired t test, $p < 0.05$). (b) Regret of SPO+ and PFY minus that of regression, under the Adam training of the benchmark (Table 1) and under the Gauss-Newton transport of (a). (c) Test regret against the mean cosine between the motion of a prediction and its segment.

step size sets the cutoff of either filter (Proposition 3). Validation regret selects the step size and the number of passes per stage from 7 settings (Appendix C.4). The selected settings end within 0.07 and 0.01 points of the Gauss-Newton step on the two knapsack tasks ($p = 0.57$ and $p = 0.93$). The Gauss-Newton step reaches this regret without a step size search, whereas 3 of the 7 gradient settings diverge on energy prices. Huber weights and discrepancy stopping change the Gauss-Newton step itself. Huber weights lower the mean regret on generated knapsack, where validation regret also selects them. They raise it on energy prices. Discrepancy stopping ends the conjugate gradient solve at an estimated noise level and raises regret on both knapsack tasks. No change of the step, alone or combined with new targets, lowers regret significantly on any task.

The Adam row compares RIFT with MSE, which uses the same displacement but trains with Adam at a fixed learning rate. RIFT reaches a regret 1.49 points below MSE on energy prices ($p = 0.004$), and the validated gradient steps reach nearly the same regret. No correction of the displacement lowers regret by as much, and no decision loss improves on MSE by as much on this task. Figure 3b compares SPO+ and PFY with regression under each transport. Trained with Adam as in the benchmark, both losses have a lower mean regret than MSE on the two knapsack tasks, and no difference is significant. Their directions have a significantly higher regret than the regression targets under the Gauss-Newton transport. Consequently, the advantage of decision losses over the regression baseline in benchmark comparisons can come from how this baseline is trained.

5.5 Following the targets more closely does not lower regret

The corrections of Section 5.3 may also fail because the predictor cannot follow them. The benchmark predictors have far fewer parameters than outputs, so a step produces only a projection of the displacement (Proposition 2). The motion of a prediction has an average cosine of only 0.16 to 0.22 with its segment (Figure 3c), and 25% to 41% of the decision changes within a stage raise the training regret (Table 13). To test this explanation, we widen the hidden layer to 256 units on generated knapsack and to 2560 units on energy prices, which gives more parameters than outputs. The wider predictors lower the regret of RIFT to 5.57% and 7.39%, but partial moves still bring no gain ($p = 0.95$ over 15 seeds). RIFT also stays within one point of MSE and of PFY with kNN targets at these widths, and no difference is significant. The cosine rises only from 0.22 to 0.26, so the extra parameters move the predictions along their segments much less than their number suggests.

RIFT still computes each step of the wide predictors with at most 10 conjugate gradient iterations. Raising this limit to 100 increases the cosine from 0.26 to 0.34, and the mean regret rises by 0.33 points with the regression displacement ($p = 0.072$) and by 5.69 points with partial moves ($p < 0.001$). This increase is consistent with Remark A.4, because a filter that passes more of a noisy target also passes more of its noise. Consequently, a deeper solve gives decision structure no advantage, and the damped and truncated solve of RIFT keeps the lower regret.

6 Related work

Decision losses differ in the training signal they differentiate. SPO+ is a convex upper bound of regret (Elmachtoub & Grigas, 2022), Fenchel-Young and perturbation methods differentiate a smoothed solver (Berthet et al., 2020; Niepert et al., 2021; Minervini et al., 2023), DBB and the identity gradient replace the Jacobian of the solver by surrogate Jacobians (Vlastelica Pogančić et al., 2020; Sahoo et al., 2023), and contrastive, ranking and learned losses are defined over a cache of decisions (Mulamba et al., 2021; Mandi et al., 2022; Shah et al., 2022). Later work reduces solver calls or gradient variance through target smoothing (Schutte et al., 2024), normal cone alignment (Tang & Khalil, 2024a), directional gradients (Huang & Gupta, 2024), tangent space projections of the prediction error (Lee et al., 2026) or losses that need no solver during training (Wan & Liu, 2026). All of these losses change only the displacement in equation 2 and keep the gradient transport. Hu et al. (2022) show that estimating the costs and then optimizing can attain faster regret rates than integrated approaches in contextual linear optimization. Geng et al. (2024) evaluate both families with shared data, predictors and solvers and report decision losses ahead of regression on seven of eight problems, and Tang & Khalil (2024b) provide implementations for such comparisons. On the five tasks studied here, the regression displacement under a damped Gauss-Newton step closes this gap. Gauss-Newton and Levenberg-Marquardt steps computed from Jacobian vector products are standard in neural network optimization (Schraudolph, 2002; Martens, 2010). The kernel $JJ^\top$ of equation 2 is the empirical neural tangent kernel (Jacot et al., 2018), and damped least squares, gradient iterations and truncated conjugate gradients are classical regularizing filters for ill posed problems (Engl et al., 1996). RIFT uses these tools to hold the transport fixed while the displacement changes.

7 Discussion and conclusion

The squared error already moves each prediction in a safe direction when the objective is affine in the cost, and regret can rise only through the transport of a training step. A decision loss changes only the displacement, so the loss sets only half of the step. Replacing the regression targets by a decision loss direction raised regret on both knapsack tasks, whereas replacing the Gauss-Newton step by validated gradient steps changed regret by at most 0.07 points. RIFT transports the regression displacement with a damped Gauss-Newton step. It stays in the leading group of every task, and no method is both faster and significantly better on any task.

A benchmark comparison of a decision loss with regression changes the displacement, the transport and the training schedule at once. Crediting a gain to the loss requires a regression baseline with the same model selection and a validated or damped step. On the knapsack and scheduling tasks, the transport moved predictions far from their segments, and no correction lowered regret significantly.

The guarantee for the regression displacement requires only an objective that is affine in the cost, and RIFT transports the displacement of any loss. Hence the same exchange applies to every decision loss on this class of problems. It measures what a new loss adds to regression under the same transport.

AI use statement

We used generative AI tools to write and debug experiment and analysis code under the authors' direction and to edit the text. We reviewed all AI-assisted work and take full responsibility for the content of this paper.

Ethics statement

This work studies training methods on public benchmark data about knapsack selection, energy scheduling, matching and portfolio selection. It involves no human subjects or personal data, and we see no direct risk of harm.

Reproducibility statement

The supplementary material contains the settings and the per-seed test results of every experiment, the training curves and diagnostics behind the figures, the results of the numerical checks, and a script that recomputes the statistical comparisons of Section 5 from these results. Appendix A contains the proofs and the numerical checks of the theory, and Appendix B gives all settings, including the baseline implementations and the statistics. The code of RIFT will be released with the camera-ready version.

References

Akshay Agrawal, Brandon Amos, Shane T. Barratt, Stephen P. Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, volume 32, pp. 9558–9570, 2019.

Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis R. Bach. Learning with differentiable pertubed optimizers. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems*, volume 3, 2021.

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of the IEEE International Conference on Image Processing*, volume 2, pp. 168–172, 1994.

John M. Danskin. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664, 1966.

Adam N. Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68 (1):9–26, 2022.

Heinz W. Engl, Martin Hanke, and Andreas Neubauer. *Regularization of Inverse Problems*, volume 375 of *Mathematics and Its Applications*. Kluwer Academic Publishers, 1996.

Haoyu Geng, Hang Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. Benchmarking PtO and PnO methods in the predictive combinatorial optimization regime. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 3207–3214, 2018.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Yichun Hu, Nathan Kallus, and Xiaojie Mao. Fast rates for contextual linear optimization. *Management Science*, 68(6):4236–4245, 2022.

Michael Huang and Vishal Gupta. Decision-focused learning with directional gradients. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 8580–8589, 2018.

Junhyeong Lee, Sangjin Jin, and Yongjae Lee. Decision-focused learning via tangent-space projection of prediction error. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

Jayanta Mandi, Víctor Bucarey, Maxime Mulamba Ke Tchomba, and Tias Guns. Decision-focused learning: Through the lens of learning to rank. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 14935–14947, 2022.

Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *Journal of Artificial Intelligence Research*, 80:1623–1701, 2024.

Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

James Martens. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 735–742, 2010.

Pasquale Minervini, Luca Franceschi, and Mathias Niepert. Adaptive perturbation-based gradient estimation for discrete latent variable models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9200–9208, 2023.

V. A. Morozov. On the solution of functional equations by the method of regularization. *Soviet Mathematics Doklady*, 7:414–417, 1966.

Maxime Mulamba, Jayanta Mandi, Michelangelo Diligenti, Michele Lombardi, Victor Bucarey, and Tias Guns. Contrastive losses and solution caching for predict-and-optimize. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 2833–2840, 2021.

Mathias Niepert, Pasquale Minervini, and Luca Franceschi. Implicit MLE: Backpropagating through discrete exponential family distributions. In *Advances in Neural Information Processing Systems*, volume 34, pp. 14567–14579, 2021.

Subham Sekhar Sahoo, Anselm Paulus, Marin Vlastelica, Vít Musil, Volodymyr Kuleshov, and Georg Martius. Backpropagation through combinatorial algorithms: Identity with projection works. In *International Conference on Learning Representations*, 2023.

Nicol N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural Computation*, 14(7):1723–1738, 2002.

Noah Schutte, Krzysztof Postek, and Neil Yorke-Smith. Robust losses for decision-focused learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 4868–4875, 2024.

Sanket Shah, Kai Wang, Bryan Wilder, Andrew Perrault, and Milind Tambe. Decision-focused learning without decision-making: Learning locally optimized decision losses. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Bo Tang and Elias B. Khalil. CaVE: A cone-aligned approach for fast predict-then-optimize with binary linear programs. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)*, volume 14743 of *Lecture Notes in Computer Science*, pp. 193–210, 2024a.

Bo Tang and Elias B. Khalil. PyEPO: A PyTorch-based end-to-end predict-then-optimize library for linear and integer programming. *Mathematical Programming Computation*, 16(3):297–335, 2024b.

Marin Vlastelica Pogančić, Anselm Paulus, Vít Musil, Georg Martius, and Michal Rolínek. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2020.

Beichen Wan and Mo Liu. A solver-free training method for predict-then-optimize. *arXiv preprint arXiv:2606.19587*, 2026.

Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1658–1665, 2019.

A Theory

A.1 Notation

All statements are written for a minimization objective. A maximization problem with objective F is treated as the minimization of $-F$, which reverses the sign of every displacement derived from the objective (Lemma A.1).

Table 2: Notation.

Symbol	meaning
$x, c \in \mathbb{R}^d$	features and realized costs of an instance
$h_\theta, \theta \in \mathbb{R}^p$, $z = h_\theta(x)$	predictor, its parameters and the predicted costs
$\mathcal{W}, w, \phi(w) \in \mathbb{R}^d$, $g(w)$	feasible set, decision, cost coefficients of a decision and cost independent part
$F(c, w) = g(w) + c^\top \phi(w)$	objective, affine in the cost
$w^\star(z)$	decision returned by the solver for the cost z under a fixed tie breaking rule
$\mathrm{R}(z, c)$	regret of $w^\star(z)$ under c in equation 1
$z_t = (1-t)z + tc$	point of the segment from z to c
$u_i \in \mathbb{R}^d$, $u \in \mathbb{R}^{Nd}$	displacement of sample i and the stacked displacements
$y_i = z_i + u_i$	target of sample i
$J_i \in \mathbb{R}^{d \times p}$, $J \in \mathbb{R}^{Nd \times p}$	Jacobian of z_i with respect to θ and the stacked Jacobians
$K = JJ^\top$, $K_{ij} = J_i J_j^\top$	output kernel and its blocks, with eigenpairs (μ_k, v_k)
$\mathcal{T}$	transport, the linear map from the stacked displacements to the first order motion of the predictions, ηK for one gradient step and $K(K + \lambda I)^{-1}$ for a damped Gauss-Newton step
$u_i - (c_i - z_i)$	correction of the regression displacement by a decision loss
η, T	step size and number of gradient steps
$\lambda = \lambda_0 \kappa \operatorname{tr}(J^\top \Omega J)/p$	damping of the Gauss-Newton step, with Levenberg-Marquardt factor κ
$\Omega = \operatorname{diag}(\omega_i I_d)$	sample weights
γ	ratio of the actual to the predicted reduction of the weighted squared residual
ψ_δ, $\delta = \delta_0 \operatorname{median}_j \lVert r_j \rVert$	pseudo Huber penalty and its scale, over the residuals r_j of the current reweighting round
$m(y)$	decision margin of a cost y with respect to a pool of decisions
$\bar{d}$	mean paired difference over seeds between two methods or variants

A.2 A single sample on the segment

Proof of Proposition 1. Fix $0 \le s < t \le 1$ and minimizers w_s of $F(z_s, \cdot)$ and w_t of $F(z_t, \cdot)$. Write $G = g(w_s) - g(w_t)$, $D = \phi(w_s) - \phi(w_t)$, $a = z^\top D$ and $b = c^\top D$, so that $z_s^\top D = (1-s)a + sb$ and $z_t^\top D = (1-t)a + tb$. Because w_s minimizes $F(z_s, \cdot)$, comparing it with w_t gives $F(z_s, w_s) \le F(z_s, w_t)$, that is,

$$G + (1-s)a + sb \le 0. \tag{4}$$

In the same way, the optimality of w_t at z_t gives $F(z_t, w_t) \le F(z_t, w_s)$, that is,

$$G + (1-t)a + tb \ge 0. \tag{5}$$

Subtracting equation 4 from equation 5 yields $(t-s)(b-a) \ge 0$, and since $t > s$ this means $b \ge a$. The difference of the true objectives is $F(c, w_s) - F(c, w_t) = G + b$, and inserting the lower bound

on G from equation 5 gives

$$F(c, w_s) - F(c, w_t) \;\geq\; -(1-t)a - tb + b \;=\; (1-t)(b-a) \;\geq\; 0,$$

where the last inequality uses $t \leq 1$ and $b \geq a$. The argument uses only the two optimality inequalities, so it holds for every feasible set on which the minimizers exist and for every choice among tied minimizers. □

Corollary A.1. *Under the assumptions of Proposition 1,* $\mathrm{R}(z_t, c)$ *is nonincreasing in* $t \in [0, 1]$*, and every decision boundary crossed by the segment leads from a decision to one whose true objective is at least as good.*

Proof. The term $\min_{w \in \mathcal{W}} F(c, w)$ in equation 1 does not depend on t, and $w^\star(z_t)$ is one of the minimizers w_t of Proposition 1. A decision boundary crossed between z_s and z_t separates two such minimizers, whose true objectives the proposition compares. □

Remark A.1 (Overshooting the realized cost). The factor $1 - t$ in the last display becomes negative for $t > 1$, and the guarantee can then fail. Take $\mathcal{W} = \{e_1, e_2\} \subset \mathbb{R}^2$, $g = 0$, $\phi(w) = w$, $c = (1, 2)$ and $z = (1, 3)$. Both z and c select e_1, so the regret is zero, but at $t = 2.5$ the point $z_t = (1, 0.5)$ selects e_2 with regret $F(c, e_2) - F(c, e_1) = 1$. Consequently, a training step that carries a prediction beyond its realized cost loses the protection of Proposition 1, even though the step points in the direction of the regression displacement.

Remark A.2 (A corrected direction can cross a boundary earlier). Proposition 1 compares points of one segment, and it does not compare directions at a fixed step length. Take $\mathcal{W} = \{e_1, e_2\}$, $g = 0$, $\phi(w) = w$, $c = (0, 1)$ and $z = (2, 0)$, and let one step move the prediction by ηu with $\eta = 0.65$ and an identity kernel, so that the transport neither rotates nor mixes the displacements. The regression displacement $u = c - z$ gives $z + \eta u = (0.7, 0.65)$, which still selects e_2 and keeps the regret at 1, because the segment crosses the decision boundary only at $t = 2/3$. The SPO+ displacement of Lemma A.1, rescaled to $\|c - z\|$, gives $z + \eta u = (0.972, 1.028)$, which selects e_1 and has regret 0. Hence a correction can lower regret without offsetting any harm of the transport.

A.3 Displacements of decision losses

The closed forms quoted in Section 3.2 follow from the subgradients of SPO+ and of the perturbed Fenchel-Young loss with respect to the prediction. Both losses are defined for an objective without a cost independent term, so this section takes $g = 0$ except where stated otherwise.

Lemma A.1 (Displacements of SPO+ and of a perturbed Fenchel-Young loss)**.** *Let* $F(c, w) = c^\top \phi(w)$ *be minimized. The SPO+ loss* $\ell(z, c) = \max_{w \in \mathcal{W}} (c - 2z)^\top \phi(w) + 2z^\top \phi(w^\star(c)) - c^\top \phi(w^\star(c))$ *has the subgradient* $2\big(\phi(w^\star(c)) - \phi(w^\star(2z - c))\big)$ *with respect to* z*, so its displacement is*

$$u^{\mathrm{SPO+}} = 2\big(\phi(w^\star(2z - c)) - \phi(w^\star(c))\big).$$

For a perturbation ε *with a symmetric density and* $\sigma > 0$*, the perturbed Fenchel-Young loss with target decision* $w^\star(c)$ *has the gradient* $\phi(w^\star(c)) - \mathbb{E}_\varepsilon[\phi(w^\star(z + \sigma\varepsilon))]$ *with respect to* z*, so its displacement is*

$$u^{\mathrm{FY}} = \mathbb{E}_\varepsilon\big[\phi(w^\star(z + \sigma\varepsilon))\big] - \phi(w^\star(c)).$$

For a maximization objective both displacements change sign, with $w^\star$ *denoting the maximizer.*

Proof. In the SPO+ loss, the maximum of $(c - 2z)^\top \phi(w)$ over $\mathcal{W}$ is attained where $(2z - c)^\top \phi(w)$ is minimal, that is, at $w^\star(2z - c)$. By Danskin's theorem (Danskin, 1966), $-2\phi(w^\star(2z - c))$ is a subgradient of this maximum with respect to z, and the remaining terms contribute $2\phi(w^\star(c))$, which gives the stated subgradient and, after a change of sign, the displacement. For the perturbed loss, Berthet et al. (2020) consider the maximization form $F_\sigma(\vartheta) = \mathbb{E}_\varepsilon[\max_w (\vartheta + \sigma\varepsilon)^\top \phi(w)]$, show that $\nabla F_\sigma(\vartheta) = \mathbb{E}_\varepsilon[\phi(w_{\max}(\vartheta + \sigma\varepsilon))]$, and obtain the gradient $\nabla F_\sigma(\vartheta) - \phi(w^\star(c))$ of the Fenchel-Young loss with target $\phi(w^\star(c))$. A minimization with cost z is the maximization with $\vartheta = -z$, and $w_{\max}(-z + \sigma\varepsilon) = w^\star(z - \sigma\varepsilon)$. The chain rule multiplies the gradient by -1, and the symmetry of ε replaces $z - \sigma\varepsilon$ by $z + \sigma\varepsilon$ inside the expectation. Finally, a maximization objective equals the minimization with costs $-z$ and $-c$, and substituting these costs into the two expressions reverses their signs. □

Both displacements vanish when the decisions they compare coincide. The SPO+ displacement is zero whenever the reflected prediction $2z - c$ selects the same decision as c, and a prediction that selects the optimal decision need not satisfy this condition. The perturbed displacement, in contrast, is zero when the perturbed predictions reproduce $\phi(w^\star(c))$ on average, which does not require them to select $w^\star(c)$. Lemma A.2 below relates the direction of the SPO+ displacement to the direction of the regression displacement.

Lemma A.2 (The SPO+ displacement is not obtuse to the regression displacement)**.** *For $F(c, w) = g(w) + c^\top \phi(w)$ minimized over $\mathcal{W}$ and any z and c, the displacement $u = 2\big(\phi(w^\star(2z - c)) - \phi(w^\star(c))\big)$ satisfies $(c - z)^\top u \geq 0$.*

Proof. Let $a = w^\star(2z - c)$, $b = w^\star(c)$, $D = \phi(a) - \phi(b)$ and $G = g(a) - g(b)$. The optimality of a at the cost $2z - c$ gives $(2z - c)^\top D + G \leq 0$, and the optimality of b at c gives $c^\top D + G \geq 0$. The first inequality is equivalent to $2z^\top D \leq c^\top D - G$, and the second to $-G \leq c^\top D$. Combining them yields $2z^\top D \leq 2c^\top D$, that is $(c - z)^\top D \geq 0$, and $u = 2D$ proves the claim. □

By Lemma A.2, the SPO+ displacement never points against the regression displacement. It rotates this displacement onto a difference of two decisions and sets it to zero for many samples, and this rotation increases regret on both knapsack tasks in Section 5.3. The perturbed displacement satisfies no such inequality, as the counterexample of Remark A.3 shows.

Remark A.3 (The perturbed displacement can point away from the realized cost). Take $\mathcal{W} = \{e_1, e_2\}$, $g = 0$, $\phi(w) = w$, $c = (0, 1)$ and $z = (0, 5)$, so that $w^\star(z) = w^\star(c) = e_1$ and the regret is zero. With Gaussian ε and $\sigma > 0$, the perturbed decision equals e_1 with some probability $q \in (0, 1)$, hence $u^{\mathrm{FY}} = (q, 1 - q) - (1, 0) = (1 - q)(-1, 1)$ and $(c - z)^\top u^{\mathrm{FY}} = -4(1 - q) < 0$. For this sample the perturbed displacement enlarges the margin of a correct decision by moving the prediction away from its realized cost, whereas $u^{\mathrm{SPO+}} = 0$ because $w^\star(2z - c) = w^\star((0, 9)) = e_1$.

A.4 Transports as spectral filters

Proof of Proposition 2. The objective $\|J\Delta - u\|^2 + \lambda\|\Delta\|^2$ is strictly convex for $\lambda > 0$, and setting its gradient to zero gives the unique minimizer $\Delta_\lambda = (J^\top J + \lambda I)^{-1} J^\top u$. Multiplying the identity $(J^\top J + \lambda I)J^\top = J^\top (JJ^\top + \lambda I)$ from the left by $(J^\top J + \lambda I)^{-1}$ and from the right by $(JJ^\top + \lambda I)^{-1}$ yields $(J^\top J + \lambda I)^{-1} J^\top = J^\top (JJ^\top + \lambda I)^{-1}$, and hence $J\Delta_\lambda = K(K + \lambda I)^{-1} u$ with $K = JJ^\top$. With the eigendecomposition $K = \sum_k \mu_k v_k v_k^\top$, $\mu_k \geq 0$, this motion equals $\sum_k \frac{\mu_k}{\mu_k + \lambda} (v_k^\top u)\, v_k$. As $\lambda \to 0$, each factor tends to one if $\mu_k > 0$ and to zero if $\mu_k = 0$, so the motion converges to the orthogonal projection of u onto the span of the eigenvectors with $\mu_k > 0$, which is the range of K and thus the range of J. If J has full row rank, all μ_k are positive, the projection is the identity, and $J\Delta_\lambda \to u$. □

The weighted system that RIFT solves in Algorithm 1 has the same structure. With $A = \Omega^{1/2} J$, the identity above gives $\Omega^{1/2} J\Delta_\lambda = AA^\top (AA^\top + \lambda I)^{-1} \Omega^{1/2} u$, so all statements of Proposition 2 hold for $\Omega^{1/2} u$ and the kernel $\Omega^{1/2} K \Omega^{1/2}$. In particular, the limiting motion is the Ω-orthogonal projection of u onto the range of J, and Huber weights change the projection but not its range.

Proposition 3 in Section 3.3 compares the two transports in the linearized regime. There, an update Δ moves the outputs to $z + J\Delta$, and the displacement u stays fixed during the updates.

Proof of Proposition 3. The first statement is part of the proof of Proposition 2. For the gradient steps, let Δ_s denote the parameters after s steps and $r_s = u - J\Delta_s$ the remaining displacement. The update $\Delta_{s+1} = \Delta_s + \eta J^\top r_s$ gives $r_{s+1} = r_s - \eta J J^\top r_s = (I - \eta K) r_s$, hence $r_T = (I - \eta K)^T u$ and $J\Delta_T = u - r_T = \big(I - (I - \eta K)^T\big) u$, whose eigenvalues are $g_{\eta,T}(\mu_k)$. For $0 \leq \eta\mu \leq 1$ the base $1 - \eta\mu$ lies in $[0, 1]$ and decreases in μ, so $g_{\eta,T}$ lies in $[0, 1]$, vanishes at $\mu = 0$ and increases in μ. The inequality $1 - x \leq e^{-x}$ gives $(1 - \eta\mu)^T \leq e^{-\eta T \mu}$, and $(1 - \eta\mu)^T \to e^{-\tau\mu}$ as $\eta \to 0$ with $\eta T = \tau$ fixed. The properties of f_λ follow directly from its formula. □

The two filters of Proposition 3 differ in shape, but both pass the displacement along eigendirections of K with large eigenvalues and suppress it along eigendirections with small eigenvalues. The

damping sets the cutoff of the Gauss-Newton step, and $1/(\eta T)$ sets that of gradient steps. Consequently, selecting λ, η or T on validation data tunes comparable filters. The agreement of the two transports in Section 5.4 is consistent with this reading. The gradient transport of our experiments keeps the targets fixed within a stage and uses minibatches, and the proposition describes its full batch version. A conjugate gradient solve truncated after k iterations also acts as a filter. It returns an update in the Krylov space generated by $J^\top \Omega u$ and $J^\top \Omega J + \lambda I$, which corresponds to a polynomial filter of degree at most k in the kernel. A small k suppresses the small eigenvalues in a similar way, and this regularizing effect of damping, gradient iterations and truncated conjugate gradients is classical for ill posed least squares problems (Engl et al., 1996).

Remark A.4 (Exact fitting reproduces label noise). Suppose that the targets are $y = \mu + \xi$, where $\mu \in \mathbb{R}^{Nd}$ is the noise free signal and the noise ξ has zero mean and covariance $\sigma_\xi^2 I$ given the predictions z. A transport with a symmetric filter $F = \sum_k f_k v_k v_k^\top$ moves the outputs from z to $z + F(y - z)$, and since $z + F(y - z) - \mu = (I - F)(z - \mu) + F\xi$ with a cross term of zero mean, the expected squared distance of the new outputs from the signal is

$$\mathbb{E}\,\|z + F(y - z) - \mu\|^2 = \|(I - F)(z - \mu)\|^2 + \sigma_\xi^2 \operatorname{tr}(F^2).$$

A correlated noise replaces the last term by $\operatorname{tr}(F\Sigma F^\top)$. As all filter factors approach one, the first term vanishes while the second approaches its maximum $\sigma_\xi^2 Nd$, so an exact fit of noisy targets reproduces the noise in the training outputs. A deeper conjugate gradient solve moves the filter factors toward one and trades the first term for the second, and the increase of test regret in Section 5.5 is consistent with this trade.

A.5 Decision margin

Partial moves place each target at the point of largest decision margin within the interval of its decision. For a cost y, a linear objective and a finite pool $\mathcal{P}$ of decisions observed during training, the margin is

$$m(y) = \min_{w \in \mathcal{P},\, w \neq w^\star(y)} \frac{y^\top \big(\phi(w) - \phi(w^\star(y))\big)}{\|\phi(w) - \phi(w^\star(y))\|}.$$

Lemma A.3 (Margin certificate)**.** *Every perturbation e with $\|e\| < m(y)$ leaves $w^\star(y)$ optimal within $\mathcal{P}$ for the cost $y + e$, where the minimum runs over the competitors with $\phi(w) \neq \phi(w^\star(y))$ and $m(y) = \infty$ when no such competitor exists.*

Proof. For a competitor $w \in \mathcal{P}$ with $w \neq w^\star(y)$, let $D_w = \phi(w) - \phi(w^\star(y))$. The Cauchy-Schwarz inequality and the definition of $m(y)$ give $(y + e)^\top D_w \geq y^\top D_w - \|e\|\,\|D_w\| \geq \big(m(y) - \|e\|\big)\|D_w\| > 0$, so every competitor has a strictly larger objective than $w^\star(y)$ at the cost $y + e$. □

RIFT computes m over the pool of decisions observed during training, so the certificate covers every decision of this pool.

A.6 A recorded example of the random check

The random check of Section 3.2 found 982 instances in which a gradient step on the squared error increased regret while lowering the prediction error. Figure 4 shows the first of them, and Table 3 lists its data. The predictor is linear, $z = J\theta$, with one sample, two costs, four candidate decisions $w_1, \ldots, w_4$ and step size 0.2. The decision at the initial parameters is optimal, with regret 0.00. One gradient step on $\frac{1}{2}\|J\theta - c\|^2$ moves the prediction along $JJ^\top(c - z)$, which reduces the distance to c but crosses into the region of a decision with regret 0.32. Because the 2×2 Jacobian has full row rank, the Gauss-Newton step to the point $z + 0.2\,(c - z)$ of the segment is attained exactly and keeps the regret at 0.00, in agreement with Proposition 1.

A.7 Numerical checks

Numerical checks on random problems confirm the statements of this appendix. Proposition 1 held on 100,000 random problems with finite feasible sets, both optimization senses and three tie breaking rules, sampled at 64 points per segment with 104,842 decision changes. It also held on 20,000

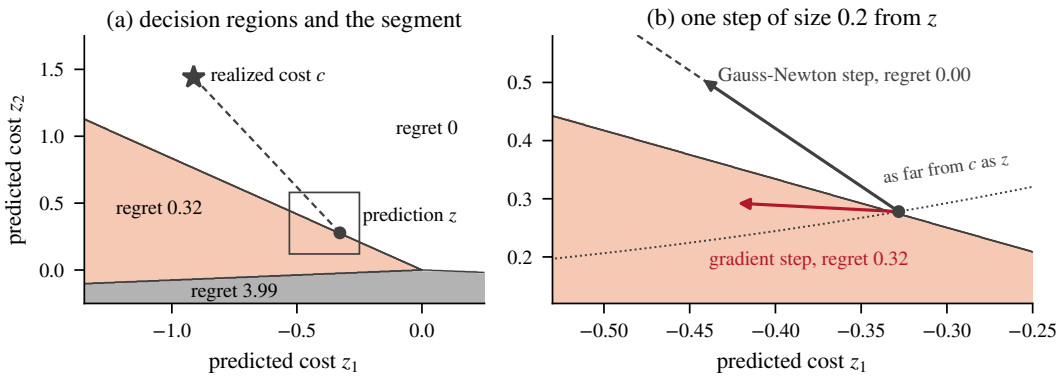

Figure 4: One instance of the random check in the space of predicted costs. For each prediction, the shading gives the regret of the solver's decision under the realized cost c. (a) The segment from the prediction z to c lies inside the region of the optimal decision. (b) A gradient step on the squared error moves the prediction closer to c, inside the dotted arc, but across a decision boundary into a region with regret 0.32. A Gauss-Newton step of the same size stays on the segment, where the regret remains zero.

Table 3: Data of the instance in Figure 4.

Jacobian rows of J	$(-0.585,\ 0.656), (-0.141,\ -0.171)$
parameters θ	$(-0.656,\ -1.084)$
realized cost c	$(-0.913,\ 1.441)$
$\phi(w_1)$	$(-1.201,\ -0.993)$
$\phi(w_2)$	$(-0.986,\ 1.910)$
$\phi(w_3)$	$(-0.941,\ 0.931)$
$\phi(w_4)$	$(-0.802,\ -0.515)$

problems with linear, convex quadratic and nonlinear g and 18,782 decision changes. All 500 perturbations inside the margin of each of 2,000 targets kept the decision in agreement with Lemma A.3, and a perturbation just outside the margin changed the decision for all 2,000 targets. The conjugate gradient step of a small network agrees with the dense Gauss-Newton solution to a relative difference below 10^{-6}, and Jacobian vector products agree with finite differences to 3.4×10^{-11}. A last layer with full row rank fits its targets to a relative error of 3.5×10^{-15}. The decision quotient preserved the decision in all 4,000 checks on top-k, simplex and matching problems. The derivative of the pseudo Huber penalty stayed below its bound, with a largest value of 0.697 for $\delta = 0.7$, and discrepancy stopping ended after 11 instead of 200 iterations with a signal error of 0.403 instead of 0.449.

B Experimental setup

B.1 Tasks and predictors

All tasks come from PredictiveCO-Benchmark (Geng et al., 2024) at its public release. Generated knapsack instances have 5 features, a polynomial cost map of degree 4 with noise half width 0.5, 20 items and capacity 30. Knapsack with energy prices uses the electricity prices of the 48 half hours of a day with 8 features per slot, and a single predictor is shared across slots. Energy scheduling uses the same price data and assigns jobs to machines over the 48 slots. Gurobi solves these three tasks with one thread. Bipartite matching predicts the edges between two sets of 50 nodes of the Cora citation graph and solves the matching with cvxpy. Portfolio selects among 50 stocks with risk aversion 0.1, and its risk term depends on the decision alone, so it belongs to g in Section 2. The number of outputs per step in Table 4 counts the predicted costs that one update moves. It equals the number of training instances times the number of costs per instance, and its ratio to the number of parameters determines which case of Proposition 2 applies. This study does not include the remaining tasks of the benchmark, which are shortest paths on Warcraft maps, the traveling salesman problem, budget allocation and advertising. Cubic top-k is also outside the comparison, because its released interface declares a minimization objective for a solver that selects the largest predicted values, and the losses of the benchmark branch on the declared sense.

Table 4: Tasks with five seeds, their splits and predictor sizes. With 32 hidden units every task lies in the rank deficient case of Proposition 2.

Task	d	train	validation	test	parameters	outputs per step
Knapsack, generated	20	320	80	200	852	6,400
Knapsack, energy prices	48	442	110	237	321	21,216
Energy scheduling	48	550	100	139	321	26,400

B.2 BASELINES

Every baseline uses the training loop of the benchmark. It trains with Adam at learning rate 0.01 for at most 300 epochs, stops early after 40 epochs without a lower validation regret, and evaluates on test once, at the epoch with the lowest validation regret. The batch sizes follow the benchmark release. They are 256 for MSE, DBB, the identity gradient and cvxpylayers, 10,000 for DFL, and 1 for all other losses. PEAR has no setting in the benchmark and uses batch size 32. LODL learns a quadratic surrogate from two samples per instance on generated knapsack and energy scheduling and a weighted squared error from 5,000 samples per instance on knapsack with energy prices. Both are settings of the benchmark, and the quadratic surrogate gives 12.08% instead of 11.96% on knapsack with energy prices. The loop omits the regret on the full training set after every epoch, which neither training nor model selection uses. Every baseline keeps the settings of the benchmark release, and RIFT uses the settings of Table 6 on every task, so no method is tuned per task.

The eleven methods of the benchmark use its released code. PyEPO 2.2.7 (Tang & Khalil, 2024b) provides PFY (Berthet et al., 2020), DPO (Berthet et al., 2020), I-MLE (Niepert et al., 2021), AI-MLE (Minervini et al., 2023), NID (Sahoo et al., 2023), PG (Huang & Gupta, 2024), CMAP (Mulamba et al., 2021), a second implementation of NCE (Mulamba et al., 2021) and the kNN targets of Schutte et al. (2024). These losses receive the predictions, costs and solver of the benchmark, so that data, splits, predictors, solvers and regret remain those of the benchmark. CaVE (Tang & Khalil, 2024a) needs the binding constraints of a Gurobi model of the task, and PEAR (Lee et al., 2026) uses the code of its authors. For both methods, we rebuild the knapsack model and verify on sampled instances before training that it returns the decisions of the benchmark solver.

Not every method runs on every task, and the tables mark the empty cells in two ways. A slash marks a loss that is not defined for the task. The PyEPO losses, including PFY and the kNN targets, are defined for an objective $c^\top w$ without a cost independent term. The quadratic risk term of portfolio falls outside this form. CaVE and PEAR also need an explicit knapsack constraint. A dash marks a combination that was not evaluated. Generated knapsack is the fastest task, and the eight methods DPO, I-MLE, AI-MLE, NID, PG, CMAP, CaVE and PEAR and the second implementation of NCE were evaluated on this task only. Here, 7 of them have a significantly higher regret than RIFT and none has a lower one (Table 10). PFY was evaluated on generated knapsack and on knapsack with energy prices. On energy scheduling, one training epoch of a loss that solves every training instance takes several minutes, and PFY and cvxpylayers were not evaluated there. The kNN targets and PFY were not evaluated on bipartite matching. LODL has no result on portfolio, because the benchmark release lacks its settings for this task.

Table 5: Two implementations of the same loss on generated knapsack, one from the benchmark and one from PyEPO. Mean relative regret over five seeds, $\bar{d}$ is PyEPO minus the benchmark, and p is the paired t test. The two SPO+ implementations agree on every seed.

Loss	benchmark	PyEPO	$\bar{d}$	p
SPO+	5.88	5.88	+0.00	tie
NCE	12.21	12.28	+0.07	0.84
identity gradient and NID	35.39	35.58	+0.19	0.61
DBB and DPO	28.25	27.35	−0.90	0.50

For SPO+, the implementation of PyEPO gives the same regret as that of the benchmark on every seed (Table 5). The two NCE implementations differ in their training details, and so do the pairs of identity gradient and NID and of DBB and DPO. The main tables keep the benchmark version of each pair. CaVE lies far above RIFT for every learning rate, and the rates 0.001, 0.01, 0.05, 0.1 and 0.5 give 32.93%, 18.08%, 15.68%, 19.37% and 18.74% on one seed.

The benchmark code needs small changes to run with current versions of PyTorch and NumPy, and two of these changes affect results. CMAP and NCE of PyEPO initialize their solution pool from the training split, which the changes provide in the data format of PyEPO. LODL on bipartite matching fits its local losses by a binary cross entropy on decision objectives outside $[0, 1]$. PyTorch 1.12 accepts such targets and later versions reject them, so the changes compute these losses as version 1.12 does. The regularized Frank-Wolfe loss of PyEPO and the perturbed optimizer of the benchmark give no result. The first completes only 3 to 5 epochs in 8 to 19 hours on generated knapsack, and the second fails at its first step.

B.3 RIFT

Table 6 lists the settings shared by all RIFT experiments. A stage computes the targets, makes at most four attempts at the damped Gauss-Newton step and evaluates the validation regret once. Each round of the reweighting in equation 3 uses the residual that the previous round leaves, and the trace that sets the damping is estimated once per stage, before the first reweighting. The fourth attempt is kept even when the ratio of the actual to the predicted reduction stays below 0.25, which happened in 16 of the 345 stages of RIFT on the three tasks with five seeds. RIFT reports the parameter vector of the stage with the lowest validation regret. It also records two diagnostics in every stage, which Figure 3c and Table 13 report. The cosine of the actual motion is $\cos \angle(z_i^{\mathrm{new}} - z_i,\, c_i - z_i)$ for each training sample after the accepted step, averaged over samples, stages and seeds. The share of harmful decision changes counts the training samples whose regret increases by more than 10^{-6} after a stage and divides this count by the number of samples whose decision changes. Both counts are summed over stages.

Table 6: Settings shared by all RIFT experiments.

Setting	value
stages	at most 30, stop after 10 stages without a lower validation regret
conjugate gradients	at most 10 iterations, relative tolerance 10^{-3}, 4 Rademacher probes for the trace
damping	$\lambda_0 = 10^{-3}$ times $\kappa \operatorname{tr}(J^\top \Omega J)/p$, no weight decay
acceptance	accept at $\gamma \geq 0.25$, otherwise multiply κ by 4 and retry, keep the fourth attempt, halve κ after $\gamma > 0.75$
predictor	the perceptron of the benchmark, 32 hidden units unless stated otherwise

B.4 DISPLACEMENTS

Table 7 defines every displacement under the short name used in the tables, and the regression displacement appears there as realized costs. Each row changes the listed setting relative to this displacement, and all other settings follow Table 6. Two further settings apply to rows of Figure 3a. For the SPO+ and PFY directions, a sample whose decision loss displacement is zero keeps its current prediction as its target in that stage. Normal cone targets select the target decision from the pool of observed decisions.

Besides the control with realized costs, the factorial design has 28 variants in two rounds. We fixed the 17 variants of the first round before the experiments. The 11 variants of the second round, which all tables mark with a dagger, followed the first test results. They are the Huber scales $\delta_0 = 0.25$ and 0.125, the three Huber weights scaled by the margin, the 10% and 35% blends with the mean, three combinations of Huber weights with blends, and discrepancy stopping with Huber weights.

B.5 TRANSPORT, WIDTH AND SOLVE DEPTH

The gradient transport of Section 5.4 replaces the conjugate gradient solve by stochastic gradient steps on the same weighted squared residual, with minibatches of 32. It keeps the stages and the validation selection of RIFT and recomputes the targets at every stage. No acceptance test is applied, so every step is kept. The grid contains the step sizes 0.2, 0.05, 0.01 and 0.002 with one pass over the training set per stage and the step sizes 0.05, 0.01 and 0.002 with ten passes per stage. We report the setting with the lowest mean validation regret over seeds. The wide predictors change only the hidden layer, which has 256 units and 6,676 parameters on generated knapsack and 2560 units and 25,601 parameters on energy prices. Baselines at these widths follow the settings of Section B.2.

Table 7: Displacements and weights, with the short names used in the tables.

Short name	definition
realized costs	$u_i = c_i - z_i$, the regression displacement
SPO+ direction	$u^{\mathrm{SPO+}}$ of Lemma A.1, with one extra solve per sample and stage at the cost $2z_i - c_i$, rescaled to $\|c_i - z_i\|$
perturbed Fenchel-Young direction	u^{FY} of Lemma A.1 with five Gaussian probes and $\sigma = 0.1\,\|z_i\|/\sqrt{d}$, rescaled to $\|c_i - z_i\|$
partial moves	first point of an 8 point grid on the segment whose decision lowers the current regret by 25%, moved to the largest margin m within the interval of that decision, with one retry that gives 4 times the weight to samples whose training regret rose; a sample whose decision is already optimal keeps the realized cost as its target
partial moves on all samples	partial moves, except that a sample whose decision is already optimal takes the point of largest margin m of that decision on its segment as its target
partial moves on all samples (no retry)	partial moves on all samples with the acceptance test of Table 6 only
normal cone targets	squared hinge on the margin constraints of the target decision, margin at least 4% of the mean training cost norm, weight 0.1 on $\|y_i - z_i\|^2$, weight decay 0.1
mean target	$y_i = \hat{\mu}_i$, predicted by a perceptron with 32 hidden units trained for 300 full batch epochs of Adam at learning rate 0.01 on the four folds of a five fold split that exclude sample i
blend with the mean	$y_i = \beta\hat{\mu}_i + (1-\beta)c_i$ with $\beta \in \{0.1, 0.25, 0.35, 0.5\}$
shrinkage	blend whose weight on c_i is the variance of $\hat{\mu}_i$ across folds divided by the sum of this variance and $\|c_i - \hat{\mu}_i\|^2$
analytic quotient	removes from u_i the directions in which all feasible decisions have the same objective, which is no direction on both knapsack tasks and the constant direction on energy scheduling
quotient estimated from decisions	removes the directions orthogonal to the span of observed decision differences
principal directions (control)	keeps the leading principal directions of the training costs, with the rank of the estimated quotient
random subspace (control)	keeps a random subspace of the same rank
Huber weights	weights equation 3 with $\delta_0 \in \{0.125, 0.25, 0.5, 1, 2\}$
Huber weights, scaled by margin	$\delta_i = \delta_0\, m(y_i)$ in place of the median scale
discrepancy stopping	stops the conjugate gradient iterations once the residual reaches an estimated noise level of the targets (Morozov, 1966)

For the deeper solve, we raise the limit of conjugate gradient iterations from 10 to 100 with the same tolerance. We evaluate the wide predictor on generated knapsack with 15 seeds.

B.6 Statistics, hardware and timing

For a method m and a reference r we form the paired differences $d_s = \mathrm{R}_m(s) - \mathrm{R}_r(s)$ over their common seeds s and report their mean $\bar{d}$. We use paired tests over seeds because variation across seeds can reverse the ranking of methods (Henderson et al., 2018; Bouthillier et al., 2021). The paired t test uses $\bar{d}/(\mathrm{sd}(d)/\sqrt{n})$ with $n-1$ degrees of freedom, and the 95% interval is $\bar{d} \pm t_{0.975,n-1}\,\mathrm{sd}(d)/\sqrt{n}$. We use the t test because the smallest attainable p value of the exact two-sided sign test with five seeds is $2 \cdot 2^{-5} = 0.0625$. We treat two methods as tied without a test when their regret agrees on every seed up to floating point noise below 10^{-6} points.

The main text compares RIFT with every other method of a task, which forms a family of up to 22 tests. After the correction of Holm (1979) over the methods of each task, RIFT remains significantly better than 12 methods on generated knapsack, 4 on energy prices and 0 on energy scheduling, and no difference reaches significance on the two tasks with three seeds. No method is significantly better than RIFT on any task, before or after the correction. Table 9 gives every paired difference with both tests.

The leading group of a task in Section 5.2 contains every method whose regret is not significantly higher than that of any other method of the task in the paired t test over their common seeds. We form these groups without and with the Holm correction over the pairs of methods of each task. The

worst-case gap of a method is its largest relative difference from the lowest mean of a task over the five tasks, and we report it for the 10 methods of Table 1 that run on all five tasks.

All experiments ran on the CPU of one machine with an AMD EPYC 7R13 processor (16 virtual cores) and 128 GB of memory under Ubuntu. The software was Python 3.13, PyTorch 2.13, NumPy 2.3, PyEPO 2.2.7 and Gurobi 13.0.3, and every run used one thread for both PyTorch and Gurobi. Every method ran under the same shared load of several concurrent runs, and timings under this load vary by a few tens of percent. Times are wall clock from the start to the end of training, measured by RIFT itself and, for the baselines, from the benchmark log. The times of RIFT also include two solves of every training instance per stage that only compute diagnostics, so they are upper bounds on its training time.

C Additional results

C.1 No method has a significantly lower regret than RIFT

Table 9 reports the paired difference of every method of Table 1 from RIFT on all five tasks. A positive difference means a higher regret than RIFT. Negative differences larger than one point occur only on bipartite matching and portfolio, and none of them is significant with three seeds. Table 10 lists the eight methods evaluated only on generated knapsack. Among them, PEAR comes closest to RIFT, and the intervals of all others lie above zero.

With the Holm correction over the pairs of methods of each task, the leading groups grow, and RIFT, MSE, SPO+, pointwise LTR and listwise LTR stay in all five of them. PFY and PFY with kNN targets stay in the groups of the tasks where they run. Table 8 separates these methods by their gap to the lowest mean of each task. The worst case of RIFT is 11.6%, and that of every other method evaluated on all five tasks is larger.

Table 8: Gap of the mean relative regret to the lowest mean of each task, in percent of that mean, for the methods of Table 1 that run on all five tasks. The last column gives the largest gap over the tasks.

Method	Knapsack, generated	Knapsack, energy prices	Energy scheduling	Bipartite matching	Portfolio	worst case
RIFT (ours)	6.8	0.0	11.6	1.0	5.1	11.6
MSE	8.6	19.7	23.2	2.2	7.1	23.2
SPO+ (Elmachtoub & Grigas, 2022)	5.1	14.3	8.7	1.0	0.0	14.3
DBB (Vlastelica Pogančić et al., 2020)	404.4	272.7	605.7	1.2	21.6	605.7
Identity gradient (Sahoo et al., 2023)	531.9	300.0	633.2	1.3	32.0	633.2
DFL (Wilder et al., 2019)	371.9	99.6	513.4	1.7	22.1	513.4
NCE (Mulamba et al., 2021)	117.9	53.9	16.2	0.0	60.9	117.9
Pointwise LTR (Mandi et al., 2022)	54.5	100.5	219.4	1.5	4.1	219.4
Pairwise LTR (Mandi et al., 2022)	35.4	14.3	10.2	1.2	3.8	35.4
Listwise LTR (Mandi et al., 2022)	5.4	5.2	18.3	2.7	10.5	18.3

C.2 RIFT trains in a fraction of the time of the strongest decision loss

Table 11 gives the mean training time of every method, and Figure 2 plots these times against regret for all five tasks. No compared method trains faster than RIFT on generated knapsack, and only MSE trains faster on energy prices, at a significantly higher regret. On energy scheduling, LODL trains faster at a significantly higher regret, and pairwise LTR has a similar regret and training time (1.86% in 4,466 seconds against 1.89% in 4,866 seconds). The time of SPO+ depends on the implementation. The PyEPO implementation reaches the regret of the benchmark implementation on generated knapsack in 226 instead of 1,035 seconds. Figure 2f–h plots the validation regret of the best model so far against training time. The median curve of PFY with kNN targets reaches the final median of RIFT after 2 to 15 times the training time that RIFT needs for it.

C.3 Variants of the regression step

Figure 5 shows one representative of each family of variants, and Table 12 lists all 28 variants against the control of the factorial design. This control is RIFT with realized costs and reproduces

Table 9: Paired difference in relative regret from RIFT (method minus RIFT, in points) on all five tasks. A bullet marks $p < 0.05$ in the paired t test, and bold marks a difference that remains significant after the correction of Holm (1979) over the methods of the task. Slashes and dashes follow Table 1.

Method	Knapsack, generated	Knapsack, energy prices	Energy scheduling	Bipartite matching	Portfolio
MSE	+0.10	**+1.49**$^\bullet$	+0.20	+1.12	+12.45
SPO+ (Elmachtoub & Grigas, 2022)	−0.10	+1.09$^\bullet$	−0.05	−0.01	−31.43
PFY (Berthet et al., 2020)	+0.06	+0.97	–	–	/
SPO+, kNN targets (Schutte et al., 2024)	−0.25	**+1.87**$^\bullet$	+0.01	–	/
PFY, kNN targets (Schutte et al., 2024)	−0.38	+0.34	−0.18	–	/
DBB (Vlastelica Pogančić et al., 2020)	**+22.27**$^\bullet$	+20.72$^\bullet$	+10.04	+0.17	+102
Identity gradient (Sahoo et al., 2023)	**+29.41**$^\bullet$	+22.80$^\bullet$	+10.50$^\bullet$	+0.34	+166
DFL (Wilder et al., 2019)	**+20.45**$^\bullet$	+7.57	+8.48$^\bullet$	+0.67	+105
cvxpylayers (Agrawal et al., 2019)	**+22.82**$^\bullet$	+20.94$^\bullet$	–	+0.72	+131
NCE (Mulamba et al., 2021)	**+6.23**$^\bullet$	**+4.10**$^\bullet$	+0.08	−0.89	+344$^\bullet$
Pointwise LTR (Mandi et al., 2022)	+2.68	+7.64	+3.51$^\bullet$	+0.44	−5.87
Pairwise LTR (Mandi et al., 2022)	+1.60$^\bullet$	**+1.09**$^\bullet$	−0.02	+0.23	+36.99
Listwise LTR (Mandi et al., 2022)	−0.08	+0.40	+0.11	+1.60	+33.44
LODL (Shah et al., 2022)	**+20.87**$^\bullet$	+4.36$^\bullet$	+1.80$^\bullet$	−0.18	–

Table 10: Methods evaluated only on generated knapsack, in the order of Appendix B.2, with mean relative regret over five seeds. $\bar{d}$ is the method minus RIFT with its 95% t interval.

Method	n	regret (%)	$\bar{d}$	95% interval of $\bar{d}$	p	p (Holm)
RIFT (ours)	5	5.98 ±0.19	ref.			
DPO (Berthet et al., 2020)	5	27.35 ±3.58	+21.37	[+17.13, +25.62]	<0.001	0.003
I-MLE (Niepert et al., 2021)	5	29.04 ±3.73	+23.06	[+18.50, +27.62]	<0.001	0.003
AI-MLE (Minervini et al., 2023)	5	26.03 ±4.53	+20.05	[+14.61, +25.49]	<0.001	0.007
NID (Sahoo et al., 2023)	5	35.58 ±4.77	+29.60	[+23.81, +35.39]	<0.001	0.003
PG (Huang & Gupta, 2024)	5	15.37 ±4.25	+9.38	[+4.09, +14.67]	0.008	0.071
CMAP (Mulamba et al., 2021)	5	7.91 ±0.57	+1.93	[+1.31, +2.55]	<0.001	0.011
CaVE (Tang & Khalil, 2024a)	5	15.50 ±2.17	+9.52	[+6.95, +12.08]	<0.001	0.007
PEAR (Lee et al., 2026)	5	6.11 ±0.42	+0.13	[−0.35, +0.61]	0.50	1.00

the results of RIFT in Table 1 on every seed. No variant of the first round lowers regret significantly (Section 5.3). Table 13 compares normal cone targets and partial moves on all samples with RIFT on the same three tasks. Partial moves on all samples differ from the partial moves of Figure 3 only in the targets of samples whose decision is already optimal. Both kinds of targets raise regret significantly on energy prices, and neither lowers it significantly on any task.

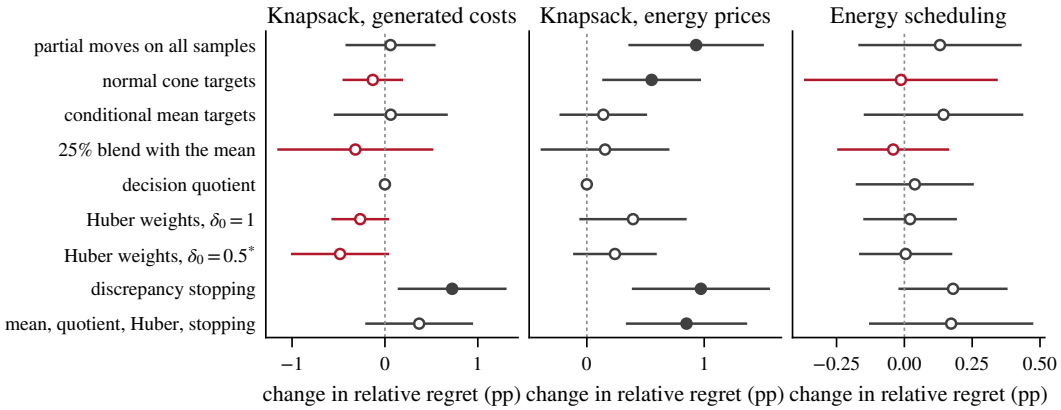

Figure 5: Change in relative regret from RIFT with realized costs for one variant of each family, with 95% t intervals over five paired seeds. Filled markers are significant at $p < 0.05$, and red marks a lower mean. The analytic quotient is the identity on both knapsack tasks, so its change is exactly zero there. The asterisk marks $\delta_0 = 0.5$, the Huber scale with the lowest mean test regret on generated knapsack, which represents its family.

Table 11: Mean training time in seconds over the seeds of Table 1, including validation after every epoch or stage.

Method	Knapsack, generated	Knapsack, energy prices	Energy scheduling	Bipartite matching	Portfolio
RIFT (ours)	45	139	4,866	824	112
MSE	130	114	16,914	13,817	35
SPO+ (Elmachtoub & Grigas, 2022)	1,035	766	29,876	26,514	179
PFY (Berthet et al., 2020)	1,209	1,412	–	–	/
SPO+, kNN targets (Schutte et al., 2024)	248	679	12,863	–	/
PFY, kNN targets (Schutte et al., 2024)	1,218	2,371	32,518	–	/
DBB (Vlastelica Pogančić et al., 2020)	588	313	38,523	16,320	94
Identity gradient (Sahoo et al., 2023)	1,495	324	27,339	16,576	55
DFL (Wilder et al., 2019)	246	445	33,513	17,439	161
cvxpylayers (Agrawal et al., 2019)	1,298	352	–	25,169	158
NCE (Mulamba et al., 2021)	1,101	454	19,191	27,025	11,573
Pointwise LTR (Mandi et al., 2022)	3,239	1,160	15,869	22,156	10,220
Pairwise LTR (Mandi et al., 2022)	2,917	804	4,466	27,258	17,732
Listwise LTR (Mandi et al., 2022)	1,589	849	5,898	22,201	13,054
LODL (Shah et al., 2022)	2,205	6,563	2,377	12,036	–

Conditional mean targets change the decision of 86% to 97% of the training samples and lie about 30% of the cost norm away from the realized costs (Table 14). Their residual on the held out folds has 26 to 34 times the variance of the disagreement between fold models. Because $\mathbb{E}[c \mid x]$ is also the population minimizer of regression toward c, replacing c by an estimate of this mean removes label noise but adds the error of the estimate. A residual that is large compared with the disagreement between folds can come from either source. Blends with the mean lower the mean regret on generated knapsack by up to 0.36 points without significance and do not lower it on the price tasks.

The analytic quotient is the identity on both knapsack tasks, because the differences between feasible knapsack decisions span the cost space. On energy scheduling, it changes 0.2% of the objective values without a significant effect on regret. The quotient estimated from decisions and the two controls change between 16% and 77% of the objective values, yet none of them changes regret significantly on any task. Discrepancy stopping increases regret significantly on both knapsack tasks, by 0.72 points on generated knapsack ($p = 0.027$). This increase is consistent with an early end of the solve at the estimated noise level, before the realized costs are fitted.

Huber weights keep every target at the realized cost and change only how the Gauss-Newton step weighs the samples (Appendix A.4), so they act on the transport and not on the displacement. They lower the mean regret for all 11 variants on generated knapsack, by 0.10 to 0.48 points. The generated costs carry independent label noise, and a large required motion there often signals a noisy label. On price data, a large required motion often reflects a price spike that matters for the decision, and no variant with Huber weights is significantly better than the control. The 2 significant reductions on generated knapsack come from the second round, and none of them survives the Holm correction over the variants of the task. The combination of mean targets, analytic quotient, Huber weights and discrepancy stopping ranks at position 24 of 29, 21 of 24 and 19 of 21 variants, counted from the best.

C.4 Gradient transport across step sizes

Table 15 lists the complete grid behind the comparison of Section 5.4. Gradient steps are sensitive to the step size. On generated knapsack, the step size 0.002 with one pass per stage ends at 23.08%, which suggests that 30 short stages move the predictor too little. The gradient steps diverge on energy prices for 3 of the 7 settings. The validation rule nevertheless selects the step size 0.2 with 1 pass on generated knapsack and 0.002 with 10 passes on energy prices. Their test regret lies within a few hundredths of a point of the best setting in the grid, which reaches 5.87% and 7.59%. The Gauss-Newton step reaches the same regret without a step size to select, which is its practical advantage over gradient steps.

Table 12: All variants of the factorial design, grouped by family. $\bar{d}$ is the variant minus the control over five paired seeds. A bullet marks $p < 0.05$, a dagger marks a variant of the second round, and a dash marks a variant that was not evaluated on a task. Table 7 defines the variants.

	Knapsack, generated			Knapsack, energy prices			Energy scheduling		
Variant	mean	$\bar{d}$	p	mean	$\bar{d}$	p	mean	$\bar{d}$	p
control, realized costs	5.98	ref.		7.60	ref.		1.89	ref.	
Conditional mean targets									
mean target	6.04	+0.06	0.79	7.74	+0.14	0.36	2.03	+0.14	0.24
shrinkage by variance ratio	6.08	+0.10	0.65	7.75	+0.15	0.42	1.98	+0.10	0.40
10% blend with the mean†	5.83	−0.15	0.36	–			1.93	+0.04	0.47
25% blend with the mean	5.66	−0.32	0.35	7.75	+0.16	0.48	1.84	−0.04	0.61
35% blend with the mean†	5.62	−0.36	0.21	–			1.92	+0.03	0.77
50% blend with the mean	5.87	−0.11	0.44	7.80	+0.20	0.49	1.93	+0.05	0.58
Decision quotients									
analytic quotient	5.98	+0.00		7.60	+0.00		1.92	+0.04	0.65
estimated from decisions	6.51	+0.53	0.057	11.00	+3.40	0.28	1.85	−0.04	0.71
principal directions (control)	6.39	+0.40	0.34	7.81	+0.21	0.16	1.89	+0.00	0.92
random subspace (control)	8.60	+2.62	0.12	7.89	+0.29	0.39	2.00	+0.11	0.087
Huber weights									
$\delta_0 = 2$	5.57	−0.41	0.085	7.90	+0.30	0.15	1.88	−0.01	0.79
$\delta_0 = 1$	5.72	−0.27	0.076	7.99	+0.39	0.075	1.91	+0.02	0.75
$\delta_0 = 0.5$	5.50	−0.48	0.064	7.84	+0.24	0.14	1.89	+0.00	0.94
$\delta_0 = 0.25^\dagger$	5.54	−0.44	0.055	8.19	+0.59	0.19	–		
$\delta_0 = 0.125^\dagger$	5.55	−0.43$^\bullet$	0.036	–			–		
$\delta_0 = 2$, scaled by margin†	5.88	−0.10	0.52	7.89	+0.29	0.17	–		
$\delta_0 = 1$, scaled by margin†	5.86	−0.12	0.24	7.92	+0.33	0.14	–		
$\delta_0 = 0.5$, scaled by margin†	5.85	−0.14	0.25	7.96	+0.36	0.10	–		
$\delta_0 = 0.5$, 25% blend†	5.55	−0.43	0.059	7.58	−0.02	0.89	–		
$\delta_0 = 0.5$, 35% blend†	5.61	−0.37	0.15	7.75	+0.15	0.42	–		
$\delta_0 = 1$, 35% blend†	5.69	−0.29$^\bullet$	0.008	–			–		
Discrepancy stopping									
realized costs	6.71	+0.72$^\bullet$	0.027	8.57	+0.97$^\bullet$	0.010	2.06	+0.18	0.068
mean target	6.95	+0.97$^\bullet$	0.047	8.51	+0.91$^\bullet$	0.018	2.00	+0.12	0.44
Huber weights†	6.14	+0.16	0.34	–			1.93	+0.04	0.73
Combinations									
mean target, quotient	6.04	+0.06	0.79	7.74	+0.14	0.36	1.86	−0.02	0.77
mean target, Huber weights	5.90	−0.08	0.70	7.88	+0.28	0.28	2.03	+0.14	0.22
quotient, Huber weights	5.72	−0.27	0.076	7.99	+0.39	0.075	2.17	+0.28	0.22
mean, quotient, Huber, stopping	6.35	+0.37	0.15	8.45	+0.85$^\bullet$	0.010	2.06	+0.17	0.19

C.5 Width and solve depth

Figure 3c plots test regret against the mean cosine between the motion of a prediction and its segment for the settings of Section 5.5, and Table 16 reports all variants with the wide predictors. If a small cosine kept decision structure from helping, predictors with a larger cosine would favor partial moves. The cosine, however, stays far below one in every setting. Even with more parameters than outputs, the damped and truncated solve produces only a small part of the displacement along the segments, with a cosine of 0.26 for realized costs and 0.28 for partial moves on generated knapsack. Wider predictors lower regret for both displacements, and at the wide widths the difference between partial moves and realized costs is close to zero on both tasks. The deeper solve raises the cosine on generated knapsack to 0.34 and increases regret for both displacements, so a larger cosine does not favor partial moves either. This pattern is consistent with Remark A.4, by which a filter that passes more of a noisy displacement also passes more of its noise. MSE, SPO+ and PFY with kNN targets stay within one point of RIFT at the wide widths on both tasks, and no difference is significant. Consequently, the comparison of Section 5.2 also holds for the larger predictors.

Table 13: Partial moves on all samples and normal cone targets against realized costs over five paired seeds, with the motion diagnostics of Section 5.5 averaged over seeds. A bullet marks $p < 0.05$.

	Knapsack, generated			Knapsack, energy prices			Energy scheduling		
Displacement	mean	$\bar{d}$	p	mean	$\bar{d}$	p	mean	$\bar{d}$	p
realized costs (RIFT)	5.98	ref.		7.60	ref.		1.89	ref.	
partial moves on all samples	6.04	+0.06	0.75	8.53	+0.93$^\bullet$	0.011	2.02	+0.13	0.29
partial moves on all samples (no retry)	5.99	+0.01	0.97	8.14	+0.54$^\bullet$	0.040	2.01	+0.12	0.18
normal cone targets	5.85	−0.13	0.33	8.15	+0.55$^\bullet$	0.022	1.87	−0.01	0.93
Diagnostics of the joint motion, mean over seeds									
parameters over outputs per step (%)		13.3			1.5			1.2	
cosine of the motion and the segment		0.22			0.16			0.20	
harmful decision changes (%)									
realized costs		25.2			34.9			41.1	
partial moves on all samples		23.7			32.5			40.1	
training time (s)									
realized costs		45			139			4,866	
partial moves on all samples		79			259			9,415	

Table 14: Diagnostics of the quotients and of the conditional mean targets. For each quotient, the table gives the share of probe costs, in percent, whose decision (dec.) or objective value (obj.) changes after the projection, and the mean relative regret of the variant.

	Knapsack, generated			Knapsack, energy prices			Energy scheduling		
Quotient	dec.	obj.	regret	dec.	obj.	regret	dec.	obj.	regret
decision quotient	0.0	0.0	5.98	0.0	0.0	7.60	6.1	0.2	1.92
estimated from decisions	38.3	30.8	6.51	50.0	46.4	11.00	61.3	58.1	1.85
principal directions (control)	40.5	25.0	6.39	21.4	16.4	7.81	27.3	20.8	1.89
random subspace (control)	53.0	39.5	8.60	65.3	63.0	7.89	77.8	77.2	2.00
Conditional mean targets									
decisions changed by the target (%)		86.2			90.0			96.6	
distance to the realized cost (% of its norm)		30.0			29.6			30.5	
residual variance over variance across folds		25.9			30.0			33.6	

C.6 Selecting a variant with validation data

The Spearman correlation between validation and test regret across the factorial design lies between +0.55 and +0.63 for every seed on generated knapsack. It varies more across seeds on the price tasks (Figure 6c), so validation regret ranks the variants of RIFT reliably only on generated knapsack. Table 17 shows the consequences for three selection rules that use only validation regret. The first rule picks the candidate with the lowest mean validation regret over seeds, the second additionally requires this mean to be at least 5% below that of the control, and the third requires a lower validation regret than the control on every seed. We apply the rules to three candidate sets. The first set contains realized costs and Huber weights with $\delta_0 = 1$, the second contains the control and the 17 variants of the first round, and the third adds the 11 variants of the second round. The last row of each block gives the best test regret in the set, which no rule can claim. All rules select Huber weights with $\delta_0 = 1$ from the first round on generated knapsack, with a test regret of 5.72%. The lower test regret of $\delta_0 = 0.5$ cannot be reached without test data. On the price tasks, the rules select variants whose test regret stays close to that of the control.

C.7 RIFT variants on bipartite matching and portfolio

Table 18 adds the variants of RIFT to the methods of Table 1 on the two tasks with three seeds. All 15 methods reach between 91.5% and 94.0% relative regret on bipartite matching. BCE is the two stage baseline of the benchmark for matching and fits the edge labels with a binary cross entropy. On portfolio, no method or RIFT variant has a relative regret below 591%. This value exceeds 100% because the optimal objectives of the test instances are small relative to their regret.

Table 15: Gradient transport with the regression displacement, with mean validation and test regret over five seeds for every setting of the grid. The triangle marks the setting selected by mean validation regret, and the last row gives the Gauss-Newton step of RIFT. Validation regret is not normalized, so its scale differs between the tasks.

		Knapsack, generated		Knapsack, energy prices	
step size	passes per stage	validation	test	validation	test
0.2	1	2.10$^{\triangleright}$	5.91	diverged	
0.05	1	2.34	6.22	diverged	
0.01	1	4.43	11.57	67.80	8.07
0.002	1	8.91	23.08	67.75	8.20
0.05	10	2.22	5.87	diverged	
0.01	10	2.31	5.98	67.86	7.91
0.002	10	2.93	7.74	67.39$^{\triangleright}$	7.59
Gauss-Newton step		2.37	5.98	68.89	7.60

Table 16: All variants with the wide predictors, against RIFT with realized costs at the same width. n is the number of seeds.

	Knapsack, generated (256 units)				Knapsack, energy prices (2560 units)			
Variant	n	mean	$\bar{d}$	p	n	mean	$\bar{d}$	p
RIFT, realized costs	15	5.57	ref.		5	7.39	ref.	
RIFT, partial moves	15	5.58	+0.01	0.95	5	7.42	+0.03	0.89
RIFT, normal cone targets	5	6.35	+0.43	0.067	5	8.18	+0.79	0.045
RIFT, Huber weights	5	5.62	−0.30	0.35	5	7.60	+0.21	0.35
MSE	5	5.45	−0.47	0.083	5	8.05	+0.66	0.17
SPO+ (Elmachtoub & Grigas, 2022)	5	5.59	−0.33	0.22	5	8.20	+0.82	0.11
PFY, kNN targets (Schutte et al., 2024)	5	5.85	−0.07	0.86	5	7.41	+0.02	0.90

D EXPERIMENT LOGS

The supplementary material contains the settings and the test result of every experiment, and its script recomputes the statistical comparisons of the paper from these results. Repeating nine factorial experiments and thirteen further experiments reproduced their logged test regret exactly.

Baseline runs have a time limit of 10 hours. The first baseline runs on energy scheduling had no limit, and 8 of them took between 11.5 and 18.5 hours, whereas no RIFT experiment on this task took longer than 3.8 hours. The limit ended two runs of PFY with kNN targets on energy scheduling. Seed 2024 reached it at epoch 66 with its best validation epoch at 28, and its test result evaluates the best checkpoint that the benchmark saved. The patience of 40 epochs would have ended this run at epoch 68. Hence its result equals that of a completed run unless one of the last two epochs had improved. Seed 2027 was still improving at epoch 68, with its best epoch at 49. We exclude this seed, which leaves four seeds. The limit also ended the regularized Frank-Wolfe loss of PyEPO on the three tasks with five seeds and one seed of pairwise LTR on portfolio. All 28 factorial variants were evaluated on generated knapsack. Knapsack with energy prices lacks 5 of them and energy scheduling lacks 8, and every evaluated variant has five seeds. The gradient transport diverged in 3 settings on energy prices.

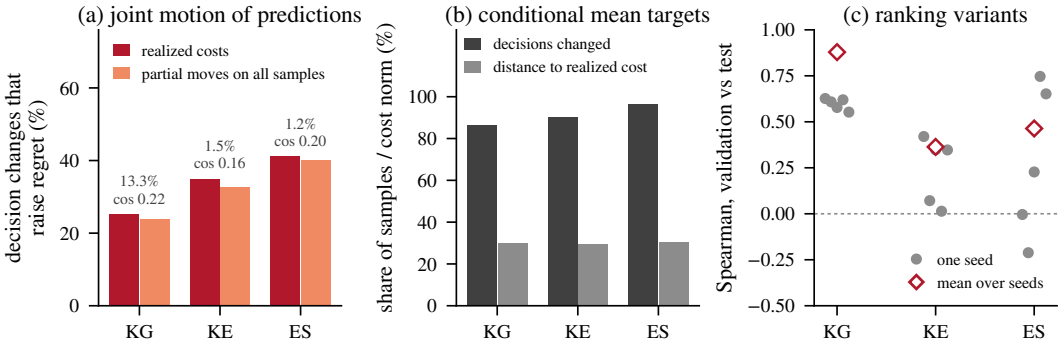

Figure 6: Diagnostics of the displacements. (a) Share of decision changes per stage that increase the training regret, annotated with the ratio of parameters to predicted outputs per step and the mean cosine between the motion of a prediction and its segment. (b) Share of training samples whose decision changes under conditional mean targets, and distance of these targets from the realized costs relative to the cost norm. (c) Spearman correlation between validation and test regret across the variants of the factorial design, per seed and for the means over seeds. KG, KE and ES denote generated knapsack, knapsack with energy prices and energy scheduling.

Table 17: Variants selected by validation regret and their test regret.

Candidate set	rule	Knapsack, generated		Knapsack, energy prices		Energy scheduling	
realized costs or $\delta_0 = 1$	lowest validation	$\delta_0 = 1$	5.72	realized costs	7.60	$\delta_0 = 1$	1.91
	5% margin	$\delta_0 = 1$	5.72	realized costs	7.60	$\delta_0 = 1$	1.91
	every seed	$\delta_0 = 1$	5.72	realized costs	7.60	realized costs	1.89
	best on test	$\delta_0 = 1$	5.72	realized costs	7.60	realized costs	1.89
first round (17 variants)	lowest validation	$\delta_0 = 1$	5.72	50% blend	7.80	estimated quotient	1.85
	5% margin	$\delta_0 = 1$	5.72	realized costs	7.60	estimated quotient	1.85
	every seed	$\delta_0 = 1$	5.72	realized costs	7.60	25% blend	1.84
	best on test	$\delta_0 = 0.5$	5.50	realized costs	7.60	25% blend	1.84
both rounds (28 variants)	lowest validation	$\delta_0 = 0.25$	5.54	50% blend	7.80	estimated quotient	1.85
	5% margin	$\delta_0 = 0.25$	5.54	realized costs	7.60	estimated quotient	1.85
	every seed	$\delta_0 = 1$	5.72	realized costs	7.60	25% blend	1.84
	best on test	$\delta_0 = 0.5$	5.50	$\delta_0 = 0.5$, 25% blend	7.58	25% blend	1.84

Table 18: Relative regret (mean and standard deviation over three seeds) on bipartite matching and portfolio, including the RIFT variants.

Bipartite matching			Portfolio		
method	mean	std	method	mean	std
NCE (Mulamba et al., 2021)	91.52	0.69	SPO+	616.22	55.24
RIFT, partial moves on all samples	92.12	1.26	Pairwise LTR	639.38	58.82
LODL (Shah et al., 2022)	92.23	0.57	Pointwise LTR	641.78	53.03
SPO+ (Elmachtoub & Grigas, 2022)	92.41	2.63	MSE	660.11	79.48
RIFT	92.41	1.25	Listwise LTR	681.09	10.93
DBB (Vlastelica Pogančić et al., 2020)	92.58	0.76	DBB	749.32	23.11
Pairwise LTR (Mandi et al., 2022)	92.64	2.20	DFL	752.41	19.17
RIFT, partial moves on all samples (no retry)	92.70	0.22	cvxpylayers	779.00	29.83
Identity gradient (Sahoo et al., 2023)	92.75	0.48	Identity gradient	813.55	19.16
Pointwise LTR (Mandi et al., 2022)	92.85	0.94	NCE	991.43	78.25
RIFT, normal cone targets	92.94	3.31			
DFL (Wilder et al., 2019)	93.08	0.22			
cvxpylayers (Agrawal et al., 2019)	93.14	1.02			
BCE	93.54	0.85			
Listwise LTR (Mandi et al., 2022)	94.01	0.84			