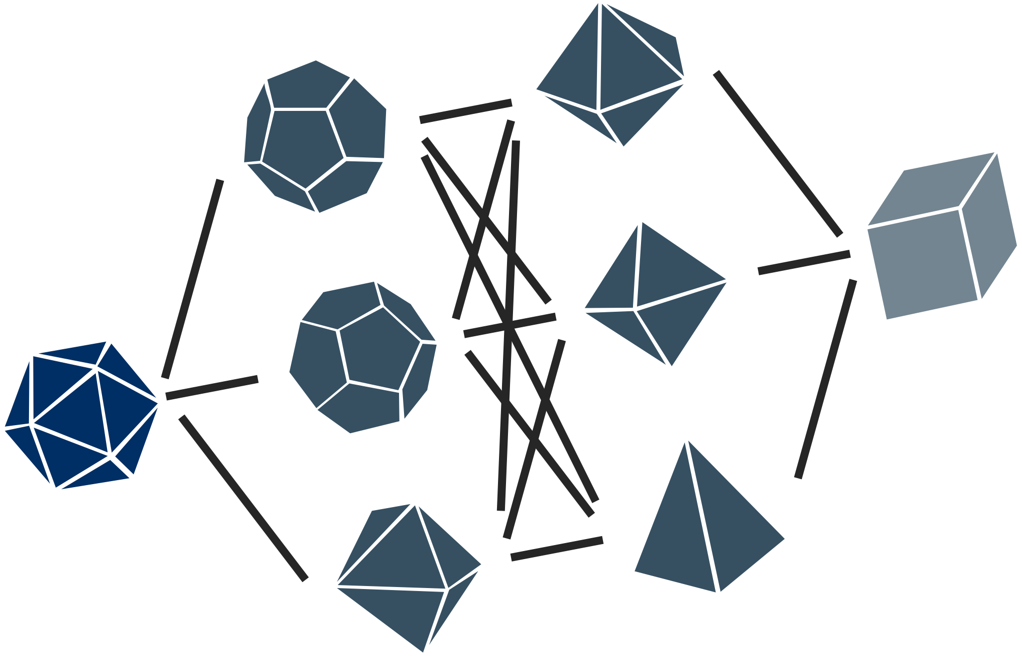
当机器学习遇上运筹学：PyEPO与端对端预测后优化



唐博

2023年07月22日

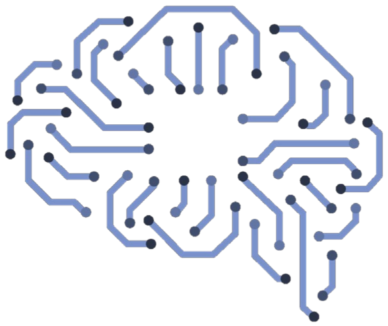
个人介绍

唐博

多伦多大学机械与工业工程系博士生，西北太平洋国家实验室数据科学实习生。

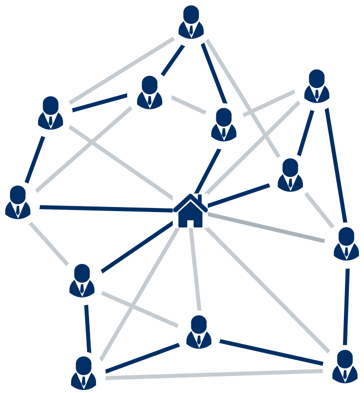
研究方向包括整数规划和数据驱动优化方法。

引言



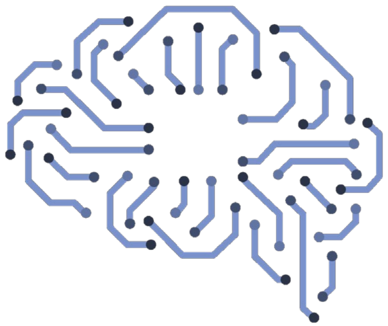
机器学习

+



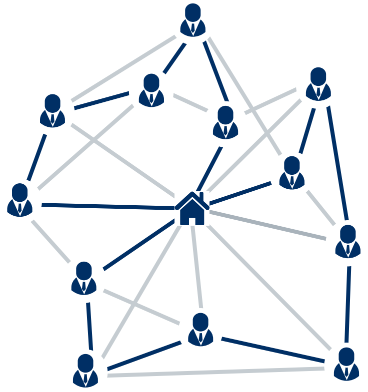
优化方法

引言



机器学习

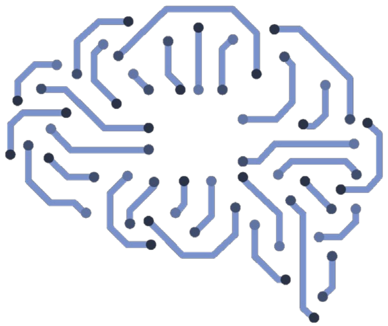
+



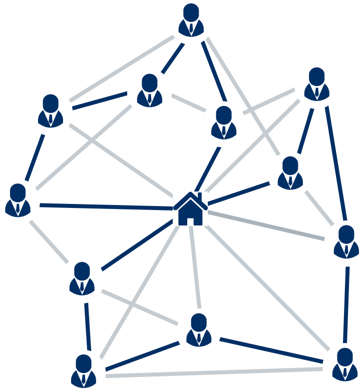
优化方法

❖ 车辆路径规划

引言



＋

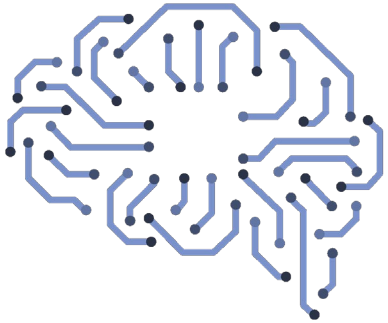


机器学习　　优化方法

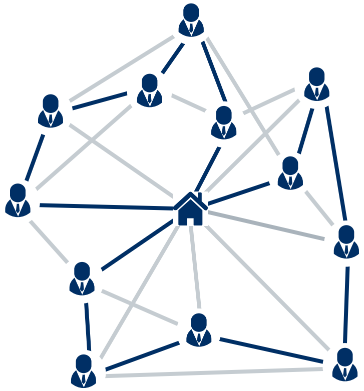
❖ 车辆路径规划

❖ 电网调度

引言



机器学习



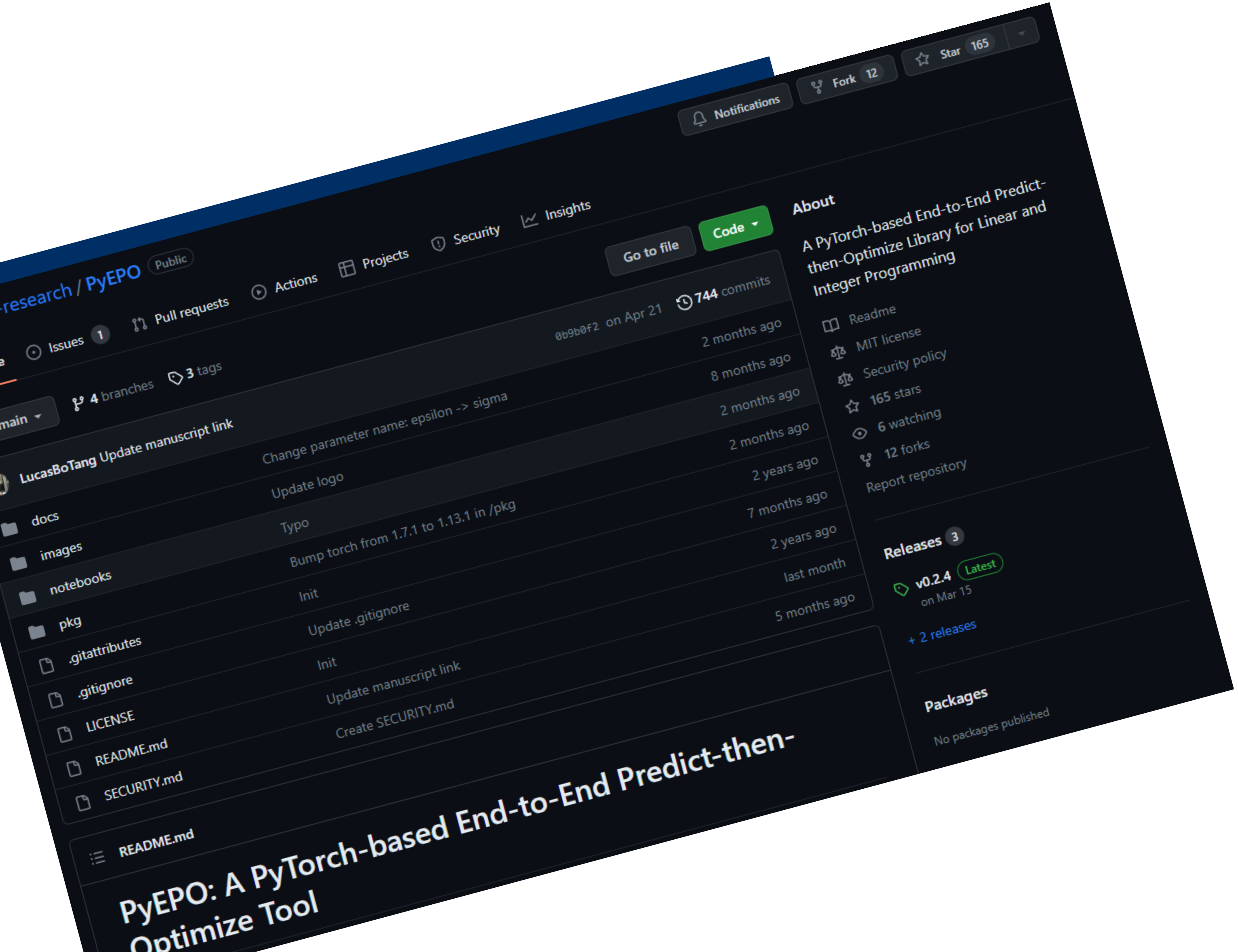
优化方法

❖ 车辆路径规划

❖ 电网调度

❖ 投资组合

引言


-research / PyEPO Public
Notifications
Fork 12
Star 165
Issues 1
Pull requests
Actions
Projects
Security
Insights
main
4 branches
3 tags
Go to file
Code
LucasBoTang Update manuscript link
0b9b0f2 on Apr 21
744 commits
docs
Change parameter name: epsilon -> sigma
2 months ago
images
Update logo
8 months ago
notebooks
Typo
2 months ago
pkg
Bump torch from 1.7.1 to 1.13.1 in /pkg
2 months ago
.gitattributes
Init
2 years ago
.gitignore
Update .gitignore
7 months ago
LICENSE
Init
2 years ago
README.md
Update manuscript link
last month
SECURITY.md
Create SECURITY.md
5 months ago
About
A PyTorch-based End-to-End Predict-then-Optimize Library for Linear and Integer Programming
Readme
MIT license
Security policy
165 stars
6 watching
12 forks
Report repository
Releases 3
v0.2.4 Latest
on Mar 15
+ 2 releases
Packages
No packages published
README.md
PyEPO: A PyTorch-based End-to-End Predict-then-Optimize Tool


PyEPO

GitHub

问题描述

优化求解

$$
\begin{aligned}
\max_{w_1, w_2} \quad & c_1 w_1 + c_2 w_2 \\
s.t. \quad & w_1 + w_2 \leq 1 \\
& w_1, w_2 \geq 0
\end{aligned}
$$

问题描述

优化求解

成本向量$\boldsymbol{c}$ 决策变量$\boldsymbol{w}$

$$\max_{w_1,w_2} c_1 w_1 + c_2 w_2$$

$$s.t. \quad w_1 + w_2 \leq 1$$

$$w_1, w_2 \geq 0$$

可行域$\boldsymbol{W}$

问题描述

优化求解

成本向量$\boldsymbol{c}$ 决策变量$\boldsymbol{w}$

$$\max_{w_1,w_2} c_1 w_1 + c_2 w_2$$

$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$

可行域$\boldsymbol{W}$

预测模型

$$\hat{\boldsymbol{c}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$$

问题描述

优化求解

成本向量$\boldsymbol{c}$ 决策变量$\boldsymbol{w}$

$$\max_{w_1,w_2} c_1 w_1 + c_2 w_2$$

$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$

可行域$\boldsymbol{W}$

预测模型

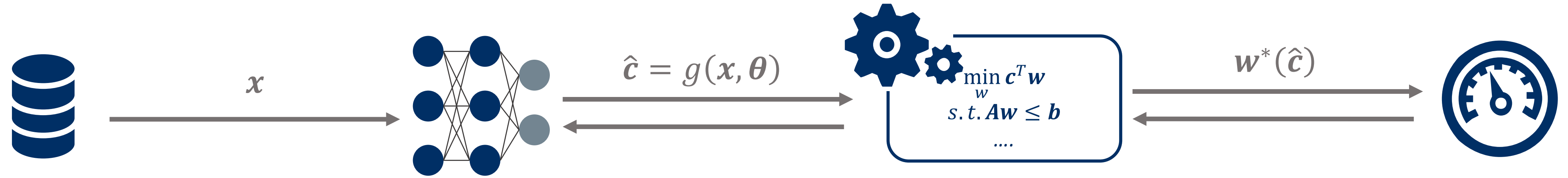
成本向量预测值

模型参数

$$\hat{\boldsymbol{c}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$$

特征数据

什么是端对端预测后优化



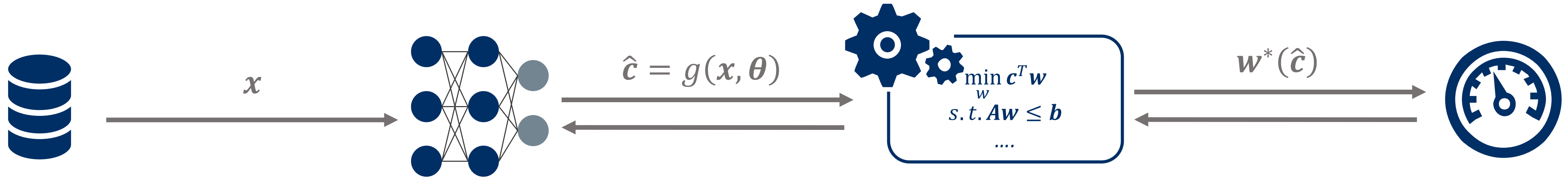
数据集 $\mathcal{D}$ 有数据对 $(\boldsymbol{x}, \boldsymbol{c})$ or $(\boldsymbol{x}, \boldsymbol{w}^*(\boldsymbol{c}))$

预测模型$g(\boldsymbol{x}, \boldsymbol{\theta})$
其参数为$\boldsymbol{\theta}$

优化求解器
$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = \underset{\boldsymbol{w} \epsilon \boldsymbol{W}}{\operatorname{argmin}} \hat{\boldsymbol{c}}^T \boldsymbol{w}$

损失函数$l(\cdot)$
定义决策误差

什么是端对端预测后优化



数据集 $\mathcal{D}$ 有数据对 $(\boldsymbol{x}, \boldsymbol{c})$ or $(\boldsymbol{x}, \boldsymbol{w}^*(\boldsymbol{c}))$

预测模型$g(\boldsymbol{x}, \boldsymbol{\theta})$ 其参数为$\boldsymbol{\theta}$

优化求解器 $\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = \underset{\boldsymbol{w} \in \boldsymbol{W}}{\operatorname{argmin}}\, \hat{\boldsymbol{c}}^T \boldsymbol{w}$

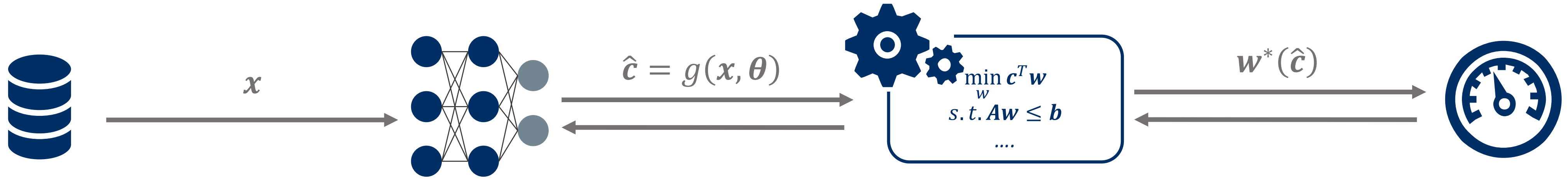
损失函数$l(\cdot)$ 定义决策误差

链式法则求梯度:

$$\frac{\partial l(\cdot)}{\partial \boldsymbol{\theta}} = \frac{\partial l(\cdot)}{\partial \hat{\boldsymbol{c}}} \frac{\partial \hat{\boldsymbol{c}}}{\partial \boldsymbol{\theta}}$$

$$= \frac{\partial l(\cdot)}{\partial \boldsymbol{w}^*_{\hat{\boldsymbol{c}}}} \frac{\partial \boldsymbol{w}^*(\boldsymbol{c})}{\partial \hat{\boldsymbol{c}}} \frac{\partial \hat{\boldsymbol{c}}}{\partial \boldsymbol{\theta}}$$

什么是端对端预测后优化



数据集 $\mathcal{D}$ 有数据对
$(\boldsymbol{x}, \boldsymbol{c})$ or $(\boldsymbol{x}, \boldsymbol{w}^*(\boldsymbol{c}))$

预测模型$g(\boldsymbol{x}, \boldsymbol{\theta})$
其参数为$\boldsymbol{\theta}$

优化求解器
$$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = \underset{\boldsymbol{w} \epsilon \boldsymbol{W}}{\operatorname{argmin}}\ \hat{\boldsymbol{c}}^T \boldsymbol{w}$$

损失函数$l(\cdot)$
定义决策误差

链式法则求梯度:

$$\frac{\partial l(\cdot)}{\partial \boldsymbol{\theta}} = \frac{\partial l(\cdot)}{\partial \hat{\boldsymbol{c}}} \frac{\partial \hat{\boldsymbol{c}}}{\partial \boldsymbol{\theta}}$$
$$= \frac{\partial l(\cdot)}{\partial \boldsymbol{w}^*_{\hat{\boldsymbol{c}}}} \frac{\partial \boldsymbol{w}^*(\hat{\boldsymbol{c}})}{\partial \hat{\boldsymbol{c}}} \frac{\partial \hat{\boldsymbol{c}}}{\partial \boldsymbol{\theta}}$$

需要计算求解过程的梯度

为什么要使用端对端预测后优化

两阶段的预测后优化

为什么要使用端对端预测后优化

两阶段的预测后优化 ➡ 像$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c})$这样的预测误差，不能准确地衡量决策的质量。

为什么要使用端对端预测后优化

两阶段的预测后优化 ➡ 像$l_{MSE}(\hat{\boldsymbol{c}},\boldsymbol{c})$这样的预测误差，不能准确地衡量决策的质量。

$$
\begin{aligned}
\max_{w_1,w_2}\ & c_1w_1 + c_2w_2 \\
s.t.\ \ & w_1 + w_2 \le 1 \\
& w_1, w_2 \ge 0
\end{aligned}
$$

假设实际成本向量为$\boldsymbol{c} = (0,1)$，最优解为$\boldsymbol{w} = (0,1)$

- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (1,0)$，其最优解为$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = (1,0)$，预测的均方误差$l_{MSE}(\hat{\boldsymbol{c}},\boldsymbol{c}) = 2$
- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (0,3)$，其最优解为$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = (0,1)$，预测的均方误差$l_{MSE}(\hat{\boldsymbol{c}},\boldsymbol{c}) = 4$

为什么要使用端对端预测后优化

两阶段的预测后优化 ➡ 像$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c})$这样的预测误差，不能准确地衡量决策的质量。

$$\max_{w_1, w_2} c_1 w_1 + c_2 w_2$$
$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$

假设实际成本向量为$\boldsymbol{c} = (0,1)$，最优解为$\boldsymbol{w} = (0,1)$

- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (1,0)$，其最优解为$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = (1,0)$，预测的均方误差$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 2$
- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (0,3)$，其最优解为$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = (0,1)$，预测的均方误差$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 4$

为什么要使用端对端预测后优化

模仿学习：预测最优解$\hat{\boldsymbol{w}}^* = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$

为什么要使用端对端预测后优化

模仿学习：预测最优解$\hat{\boldsymbol{w}}^* = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$

计算效率

它规避了计算效率的主要瓶颈：优化求解

为什么要使用端对端预测后优化

模仿学习：预测最优解$\hat{\boldsymbol{w}}^* = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$

计算效率

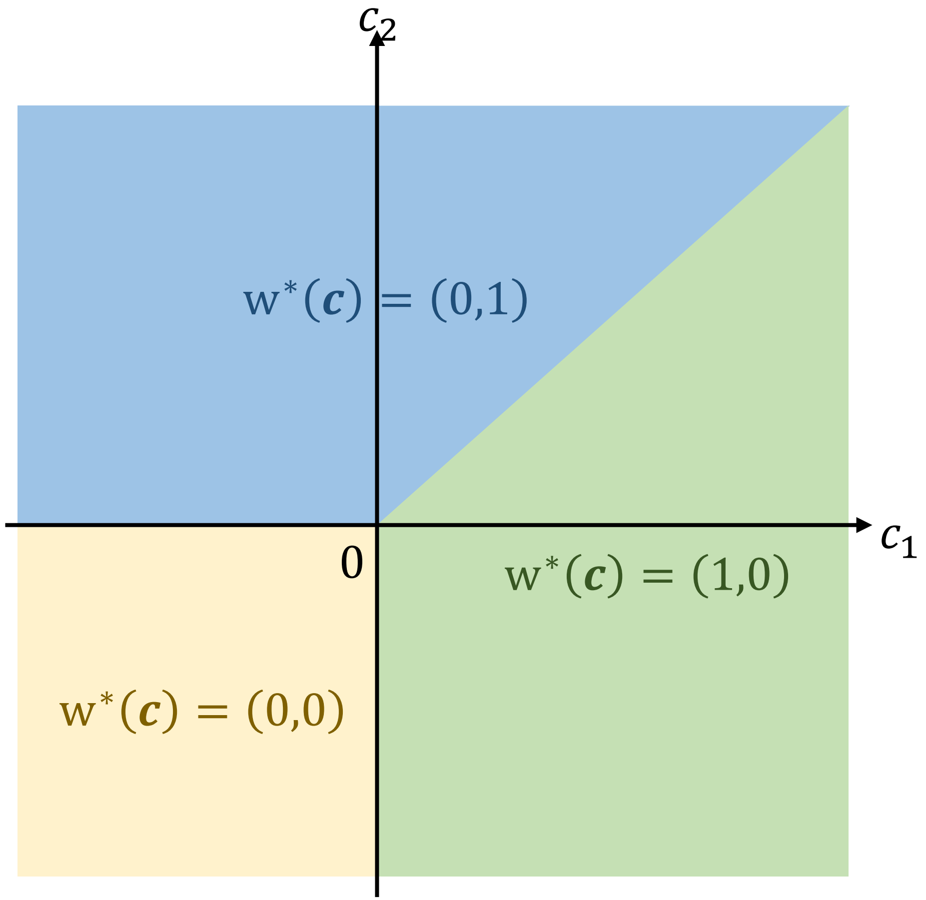
它规避了计算效率的主要瓶颈：优化求解

可行性

预测结果常常面临可行性问题

如何进行线性问题的端对端预测后优化

$$\max_{w_1,w_2} c_1 w_1 + c_2 w_2$$
$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$



线性规划最优解$\boldsymbol{w}^*(\boldsymbol{c})$作为成本参数$\boldsymbol{c}$的函数，是一个分片常数函数

如何进行线性问题的端对端预测后优化

基于KKT条件的隐函数求导

OptNet:

求解KKT条件的偏微分矩阵线性方程组来计算求解器反向传播的梯度

线性目标函数中加二次项

Karush-Kuhn-Tucker conditions

Given general problem

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & h_i(x) \le 0, \;\; i = 1, \dots m \\ & \ell_j(x) = 0, \;\; j = 1, \dots r \end{aligned}$$

The **Karush-Kuhn-Tucker conditions** or **KKT conditions** are:

- $0 \in \partial f(x) + \sum_{i=1}^{m} u_i \partial h_i(x) + \sum_{j=1}^{r} v_j \partial \ell_j(x)$ (stationarity)
- $u_i \cdot h_i(x) = 0$ for all i (complementary slackness)
- $h_i(x) \le 0, \; \ell_j(x) = 0$ for all i, j (primal feasibility)
- $u_i \ge 0$ for all i (dual feasibility)

- Amos, B., & Kolter, J. Z. (2017, July). Optnet: Differentiable optimization as a layer in neural networks. In International Conference on Machine Learning (pp. 136-145). PMLR.
- Wilder, B., Dilkina, B., & Tambe, M. (2019, July). Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 1658-1665).

如何进行线性问题的端对端预测后优化

SPO+

定义决策误差$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^T \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^T \boldsymbol{w}^*(\boldsymbol{c})$，没有非$\mathbf{0}$导数

• Elmachtoub, A. N., & Grigas, P. (2021). Smart “predict, then optimize”. Management Science.

如何进行线性问题的端对端预测后优化

SPO+

定义决策误差$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{w}^*(\boldsymbol{c})$，没有非$\mathbf{0}$导数

有$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c})$的凸上界：

$$l_{\text{SPO+}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = -\min_{\boldsymbol{w} \in \boldsymbol{W}}(2\hat{\boldsymbol{c}} - \boldsymbol{c})^{\boldsymbol{T}}\boldsymbol{w} + 2\hat{\boldsymbol{c}}^{\boldsymbol{T}}\boldsymbol{w}^*(\boldsymbol{c}) - \boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{w}^*(\boldsymbol{c})$$

- Elmachtoub, A. N., & Grigas, P. (2021). Smart “predict, then optimize”. Management Science.

如何进行线性问题的端对端预测后优化

SPO+

定义决策误差$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^T \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^T \boldsymbol{w}^*(\boldsymbol{c})$，没有非$\mathbf{0}$导数

有$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c})$的凸上界：

$$l_{\text{SPO+}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = -\min_{\boldsymbol{w} \in \boldsymbol{W}} (2\hat{\boldsymbol{c}} - \boldsymbol{c})^T \boldsymbol{w} + 2\hat{\boldsymbol{c}}^T \boldsymbol{w}^*(\boldsymbol{c}) - \boldsymbol{c}^T \boldsymbol{w}^*(\boldsymbol{c})$$

对于损失函数$l_{SPO+}(\hat{\boldsymbol{c}}, \boldsymbol{c})$，有次梯度：

$$2\boldsymbol{w}^*(\boldsymbol{c}) - 2\boldsymbol{w}^*(2\hat{\boldsymbol{c}} - \boldsymbol{c}) \in \frac{l_{\text{SPO+}}(\hat{\boldsymbol{c}}, \boldsymbol{c})}{\partial \hat{\boldsymbol{c}}}$$

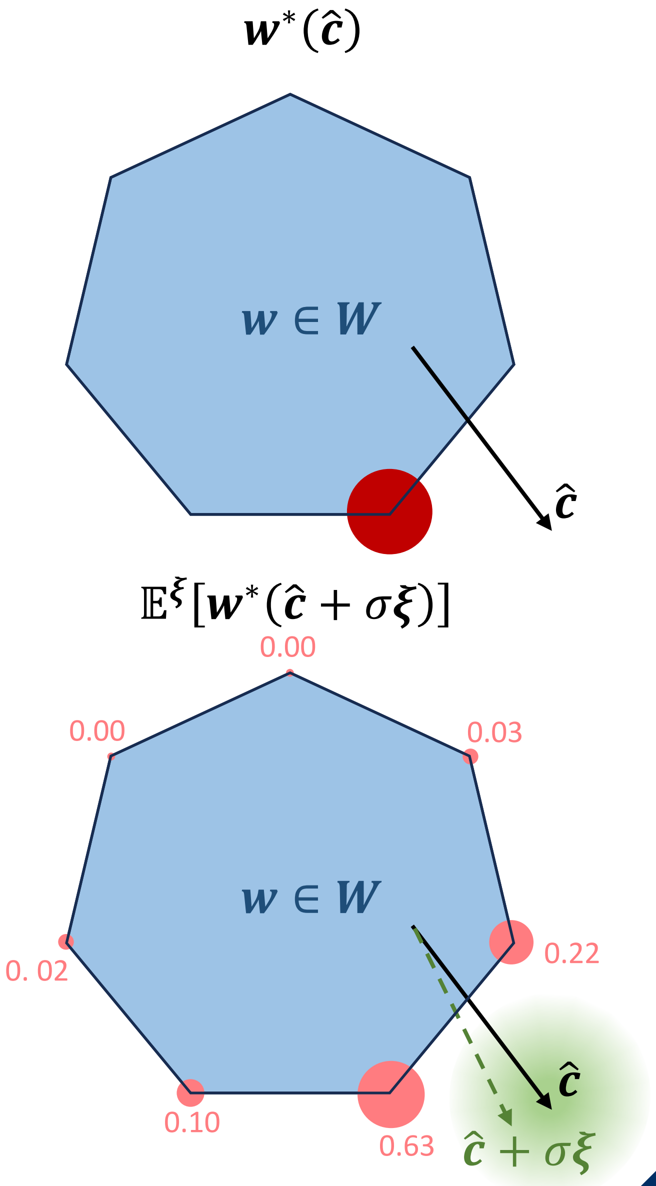
- Elmachtoub, A. N., & Grigas, P. (2021). Smart “predict, then optimize”. Management Science.

如何进行线性问题的端对端预测后优化

扰动方法

随机扰动来处理成本向量的预测值$\hat{\boldsymbol{c}}$

最优决策的期望值$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})]$代替$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$，即可行域极点的加权平均（凸组合）



- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization layers: a probabilistic approach. arXiv preprint arXiv:2207.13513.

如何进行线性问题的端对端预测后优化

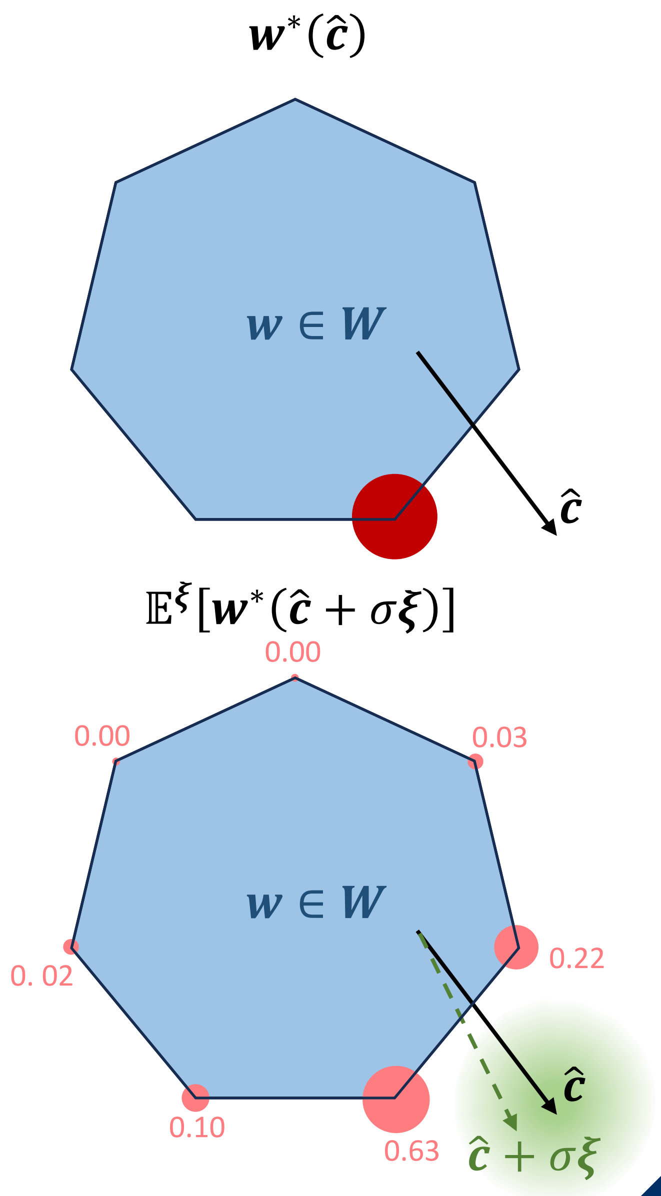
扰动方法

随机扰动来处理成本向量的预测值$\hat{\boldsymbol{c}}$

最优决策的期望值$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})]$代替$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$，即可行域极点的加权平均（凸组合）

用样本量为K的蒙特卡洛采样来近似期望：

$$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})] \approx \frac{1}{\mathrm{K}}\sum_{\kappa}^{\mathrm{K}} \boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi}_{\boldsymbol{\kappa}})$$



- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization layers: a probabilistic approach. arXiv preprint arXiv:2207.13513.

如何进行线性问题的端对端预测后优化

扰动方法

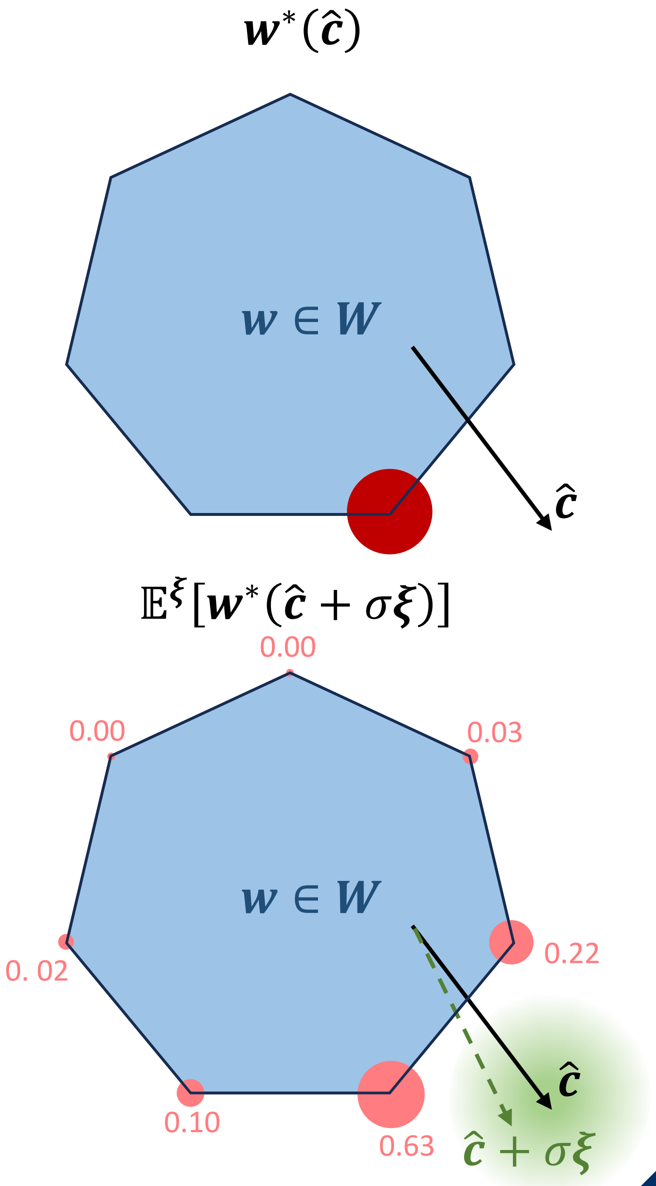
随机扰动来处理成本向量的预测值$\hat{\boldsymbol{c}}$

最优决策的期望值$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})]$代替$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$，即可行域极点的加权平均（凸组合）

用样本量为K的蒙特卡洛采样来近似期望：

$$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})] \approx \frac{1}{\mathrm{K}}\sum_{\kappa}^{\mathrm{K}} \boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi}_{\boldsymbol{\kappa}})$$

对成本向量$\hat{\boldsymbol{c}}$有非负性的要求：乘法扰动



- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization layers: a probabilistic approach. arXiv preprint arXiv:2207.13513.

如何进行线性问题的端对端预测后优化

扰动方法

有期望目标函数$F^{\xi}(\boldsymbol{c}) = \mathbb{E}^{\xi}\left[\min_{\boldsymbol{w}\in\boldsymbol{W}}(\boldsymbol{c}+\sigma\boldsymbol{\xi})^{\boldsymbol{T}}\boldsymbol{w}\right]$

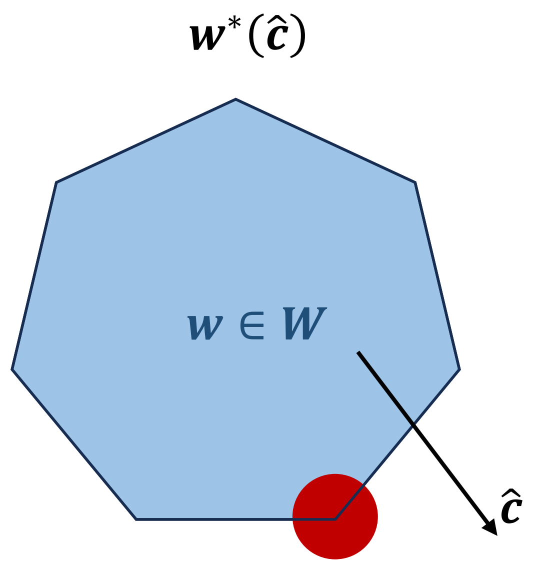
Fenchel-Young对偶:

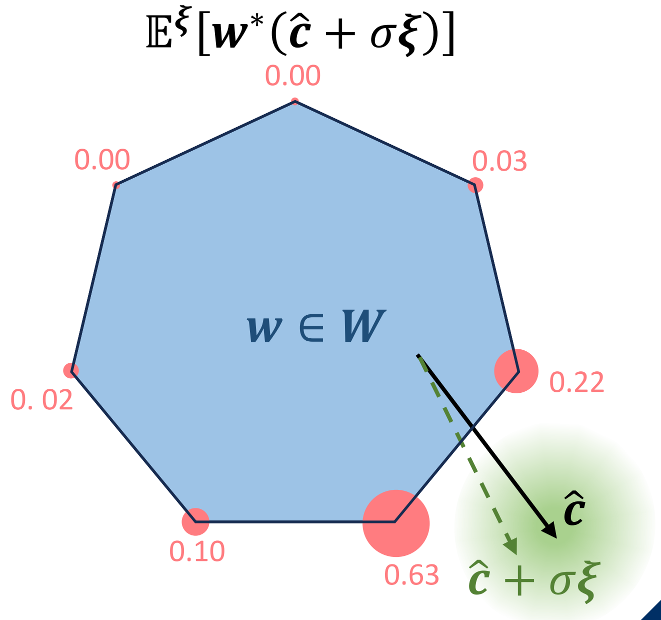
令$\Omega\big(\boldsymbol{w}^*(\boldsymbol{c})\big)$为$F^{\xi}(\boldsymbol{c})$的对偶，则有:

$$l_{\text{PFY}}\big(\hat{\boldsymbol{c}}, \boldsymbol{w}^*(\boldsymbol{c})\big) = \hat{\boldsymbol{c}}^T\boldsymbol{w}^*(\boldsymbol{c}) - F^{\xi}(\hat{\boldsymbol{c}}) - \Omega\big(\boldsymbol{w}^*(\boldsymbol{c})\big)$$

求导:

$$\frac{l_{\text{PFY}}\big(\hat{\boldsymbol{c}}, \boldsymbol{w}^*(\boldsymbol{c})\big)}{\partial\hat{\boldsymbol{c}}} = \boldsymbol{w}^*(\boldsymbol{c}) - \mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})]$$



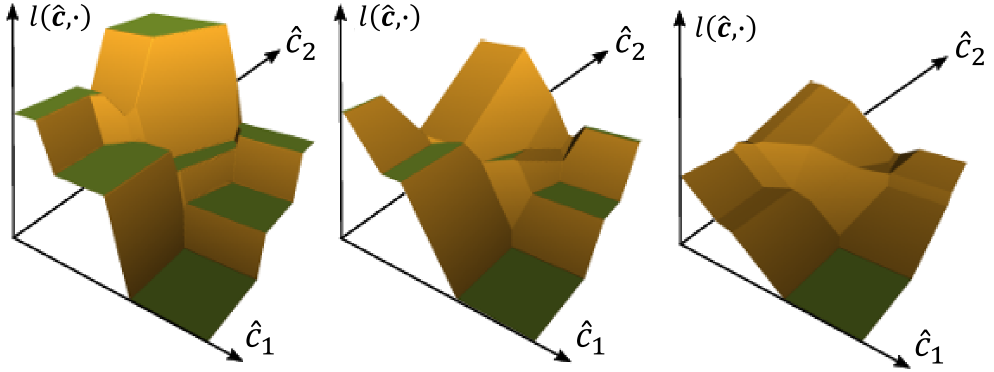


- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.

如何进行线性问题的端对端预测后优化

黑盒方法

- “Differentiable Black-box”方法：对分片常数损失函数进行连续插值，从而将其转化为分片线性函数

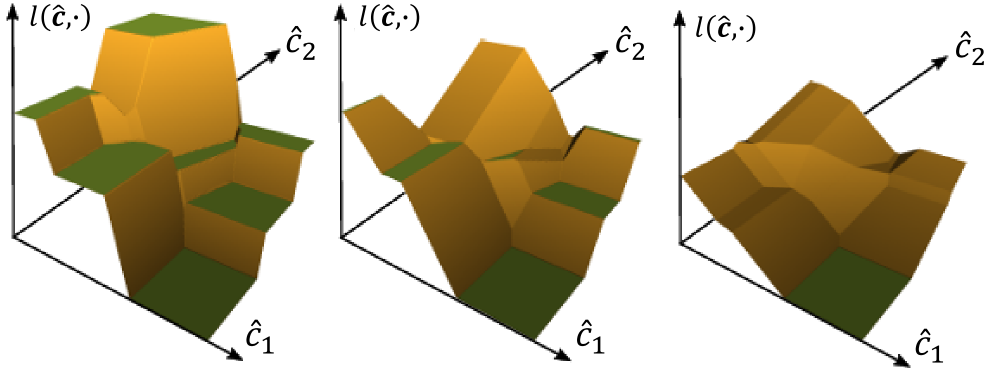


- Pogančić, M. V., Paulus, A., Musil, V., Martius, G., & Rolinek, M. (2019, September). Differentiation of blackbox combinatorial solvers. In International Conference on Learning Representations.
- Sahoo, S. S., Paulus, A., Vlastelica, M., Musil, V., Kuleshov, V., & Martius, G. (2022). Backpropagation through combinatorial algorithms: Identity with projection works. arXiv preprint arXiv:2205.15213.

如何进行线性问题的端对端预测后优化

黑盒方法

- “Differentiable Black-box”方法：对分片常数损失函数进行连续插值，从而将其转化为分片线性函数
- “Negative Identity”方法：用负单位矩阵$-\boldsymbol{I}$替代求解器梯度$\frac{\partial \boldsymbol{w}^*(\hat{\boldsymbol{c}})}{\partial \hat{\boldsymbol{c}}}$。更新成本参数的预测值$\hat{\boldsymbol{c}}$：沿着$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$上升的方向减少，沿着$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$下降的方向增加。让$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$接近$\boldsymbol{w}^*(\boldsymbol{c})$。



- Pogančić, M. V., Paulus, A., Musil, V., Martius, G., & Rolinek, M. (2019, September). Differentiation of blackbox combinatorial solvers. In International Conference on Learning Representations.
- Sahoo, S. S., Paulus, A., Vlastelica, M., Musil, V., Kuleshov, V., & Martius, G. (2022). Backpropagation through combinatorial algorithms: Identity with projection works. arXiv preprint arXiv:2205.15213.

如何进行线性问题的端对端预测后优化

对比、排序方法

在训练集以及训练、求解过程中，我们可以自然地收集到大量的可行解，形成一个解集合Γ。

- Mulamba, M., Mandi, J., Diligenti, M., Lombardi, M., Bucarey, V., & Guns, T. (2021). Contrastive losses and solution caching for predict-and-optimize. Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence.
- Mandi, J., Bucarey, V., Mulamba, M., & Guns, T. (2022). Decision-focused learning: through the lens of learning to rank. Proceedings of the 39th International Conference on Machine Learning.

如何进行线性问题的端对端预测后优化

对比、排序方法

在训练集以及训练、求解过程中，我们可以自然地收集到大量的可行解，形成一个解集合Γ。

- 对比方法：

将非最优可行解的子集$\Gamma\backslash \boldsymbol{w}^*(\boldsymbol{c})$作为负样本，让最优解和其他解之间的的差值尽可能大

$$l_{NCE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \frac{1}{|\Gamma| - 1} \sum_{\Gamma\backslash \boldsymbol{w}^*(\boldsymbol{c})}^{\boldsymbol{w}^\gamma} (\hat{\boldsymbol{c}}^T \boldsymbol{w}^*(\boldsymbol{c}) - \hat{\boldsymbol{c}}^T \boldsymbol{w}^\gamma)$$

- Mulamba, M., Mandi, J., Diligenti, M., Lombardi, M., Bucarey, V., & Guns, T. (2021). Contrastive losses and solution caching for predict-and-optimize. Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence.
- Mandi, J., Bucarey, V., Mulamba, M., & Guns, T. (2022). Decision-focused learning: through the lens of learning to rank. Proceedings of the 39th International Conference on Machine Learning.

如何进行线性问题的端对端预测后优化

对比、排序方法

在训练集以及训练、求解过程中，我们可以自然地收集到大量的可行解，形成一个解集合Γ。

- 对比方法：

将非最优可行解的子集$\Gamma\backslash \boldsymbol{w}^*(\boldsymbol{c})$作为负样本，让最优解和其他解之间的的差值尽可能大

$$l_{NCE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \frac{1}{|\Gamma|-1}\sum_{\Gamma\backslash \boldsymbol{w}^*(\boldsymbol{c})}^{\boldsymbol{w}^\gamma} (\hat{\boldsymbol{c}}^T \boldsymbol{w}^*(\boldsymbol{c}) - \hat{\boldsymbol{c}}^T \boldsymbol{w}^\gamma)$$

- 排序方法：

将端对端预测后优化任务转化为一个排序学习(Learning to rank)，其目标是学习一个目标函数（如$\hat{\boldsymbol{c}}^T \boldsymbol{w}$）作为排序得分，以便对可行解的子集$\Gamma$进行正确排序。
有：单文档方法、文档对方法、以及文档列表方法

- Mulamba, M., Mandi, J., Diligenti, M., Lombardi, M., Bucarey, V., & Guns, T. (2021). Contrastive losses and solution caching for predict-and-optimize. Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence.
- Mandi, J., Bucarey, V., Mulamba, M., & Guns, T. (2022). Decision-focused learning: through the lens of learning to rank. Proceedings of the 39th International Conference on Machine Learning.

使用PyEPO进行端对端预测后优化

$$\max_{\boldsymbol{w}} \sum_{i=0}^{4} c_i w_i$$

$$\begin{aligned} s.t.\quad & 3w_0 + 4w_1 + 3w_2 + 6w_3 + 4w_4 \leq 12 \\ & 4w_0 + 5w_1 + 2w_2 + 3w_3 + 5w_4 \leq 10 \\ & 5w_0 + 4w_1 + 6w_2 + 2w_3 + 3w_4 \leq 10 \\ & w_0, w_1, w_2, w_3, w_4 \in \{0,1\} \end{aligned}$$

Colab Tutorial:
https://colab.research.google.com/github/LucasBoTang/PyEPO-PredOpt-Chinese-Tutorial/blob/main/Example.ipynb

感谢
Q&A

唐博
2023年07月22日