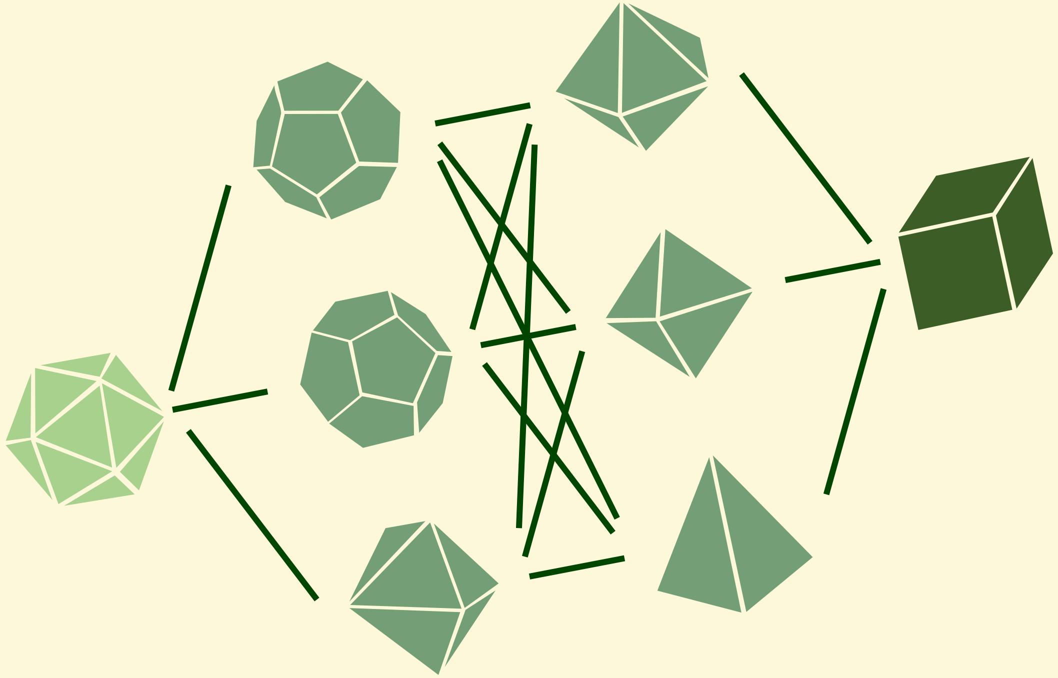
当机器学习PyTorch遇到运筹学COPT：PyEPO与端到端预测后优化



主讲人：唐博

2023年11月29日

Mechanical & Industrial Engineering
UNIVERSITY OF TORONTO

个人介绍

唐博

多伦多大学机械与工业工程系
第四年博士生

Elias B. Khalil

多伦多大学机械与工业工程系
助理教授

现代供应链的数据驱动算法
SCALE AI研究主席

问题描述

$$\min_{\underbrace{\boldsymbol{w}}_{\text{决策变量}}}\{\overbrace{\boldsymbol{c}^\top\boldsymbol{w}}^{\text{线性目标函数}}:\underbrace{\boldsymbol{w}\in\boldsymbol{S}}_{\text{可行域}}\}$$

使用适当的算法（如线性规划、二次约束规划、约束规划、贪心算法……）获取**最优解**$\boldsymbol{w}^*(\boldsymbol{c})$

COPT Cardinal Optimizer

问题描述

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_1{}^{\top}\boldsymbol{w}:\boldsymbol{w}\in\boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_2{}^{\top}\boldsymbol{w}:\boldsymbol{w}\in\boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_3{}^{\top}\boldsymbol{w}:\boldsymbol{w}\in\boldsymbol{S}\}$$

$$\ddots$$

COPT Cardinal Optimizer

问题描述

未知成本向量

$$\min_{\boldsymbol{w}} \{\boldsymbol{c}_1{}^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}} \{\boldsymbol{c}_2{}^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}} \{\boldsymbol{c}_3{}^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

∵ 相同的约束条件

COPT Cardinal Optimizer

问题描述

未知成本向量

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_1{}^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_2{}^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_3{}^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

⋱ 相同的约束条件

观测到的特征向量

$\boldsymbol{x}_1$

$\boldsymbol{x}_2$

$\boldsymbol{x}_3$

⋮

问题描述

未知成本向量

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_1^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_2^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_3^\top \boldsymbol{w}: \boldsymbol{w} \in \boldsymbol{S}\}$$

$\ddots$ 相同的约束条件

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

预测

观测到的特征向量

$\boldsymbol{x}_1$

$\boldsymbol{x}_2$

$\boldsymbol{x}_3$

$\vdots$

问题描述

未知成本向量

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_1^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_2^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

$$\min_{\boldsymbol{w}}\{\boldsymbol{c}_3^\top \boldsymbol{w} : \boldsymbol{w} \in \boldsymbol{S}\}$$

$\because$ 相同的约束条件

成本向量预测值

模型参数

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

预测

观测到的特征向量

$\boldsymbol{x}_1$

$\boldsymbol{x}_2$

$\boldsymbol{x}_3$

$\vdots$

例子

❖车辆路线规划

❖能源调度

❖投资组合

例子

❖车辆路线规划

❖能源调度

❖投资组合

未知成本： 旅行时间、电价、资产回报率

例子

❖车辆路线规划

❖能源调度

❖投资组合

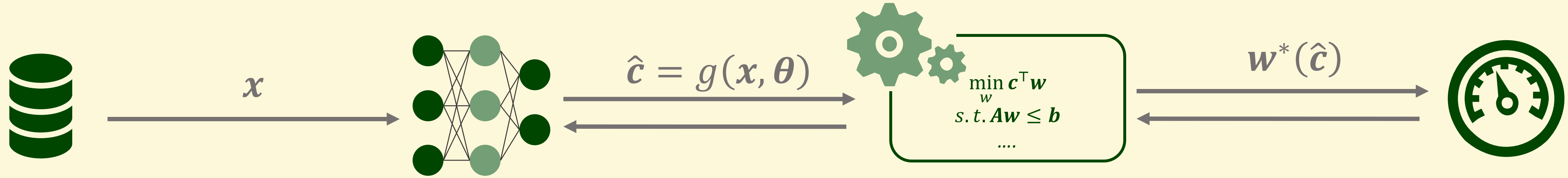
未知成本： 旅行时间、电价、资产回报率

⬆

已知数据： 距离、时间、天气、金融因子…

COPT Cardinal Optimizer

端到端预测后优化



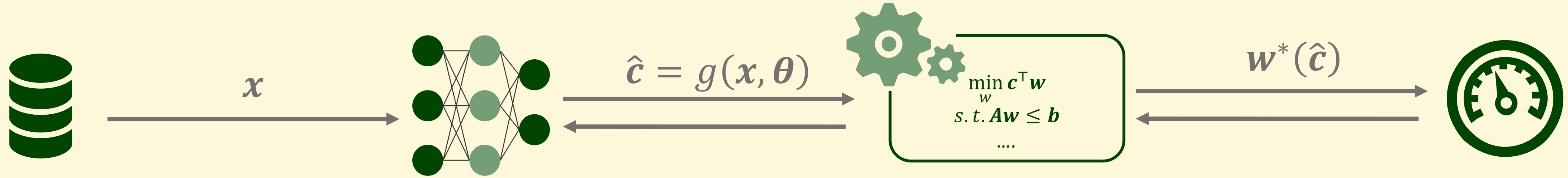
数据集 $\mathcal{D}$
有数据对 $(\boldsymbol{x}, \boldsymbol{c})$

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

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

损失函数$l(\cdot)$
衡量决策损失

端到端预测后优化



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

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

优化求解
$\boldsymbol{w}^*(\hat{\boldsymbol{c}}) = \operatorname*{argmin}_{\boldsymbol{w} \in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$

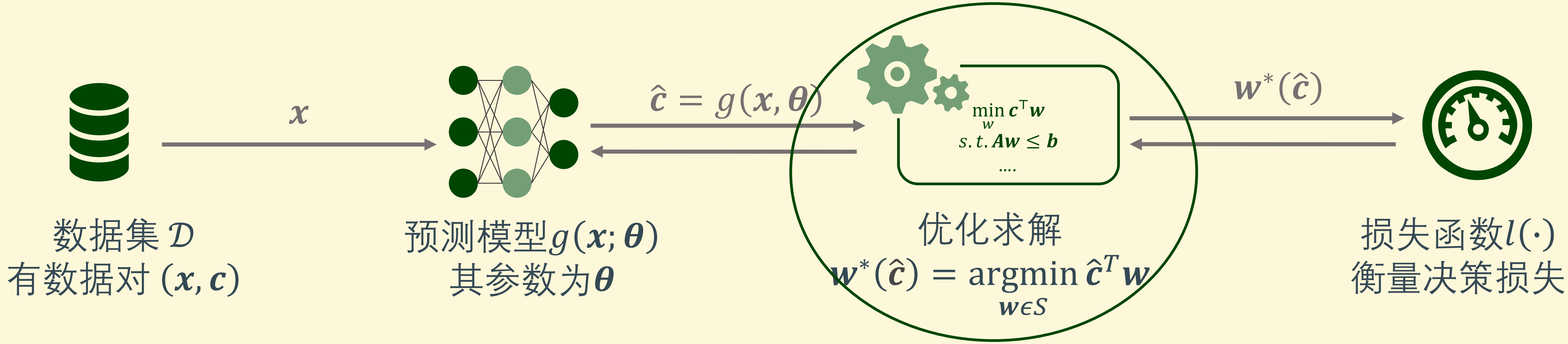
损失函数$l(\cdot)$
衡量决策损失

e.g.

$$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \overbrace{\boldsymbol{c}^{\top}\boldsymbol{w}^*(\hat{\boldsymbol{c}})}^{\text{给定预测成本}\hat{c}\text{的最优解}} - \overbrace{\boldsymbol{c}^{\top}\boldsymbol{w}^*(\boldsymbol{c})}^{\text{真实成本}c\text{下的最优目标值}}$$

$$l_{\text{Square}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \frac{1}{2}\|\boldsymbol{w}^*(\boldsymbol{c}) - \boldsymbol{w}^*(\hat{\boldsymbol{c}})\|_2^2$$

端到端预测后优化

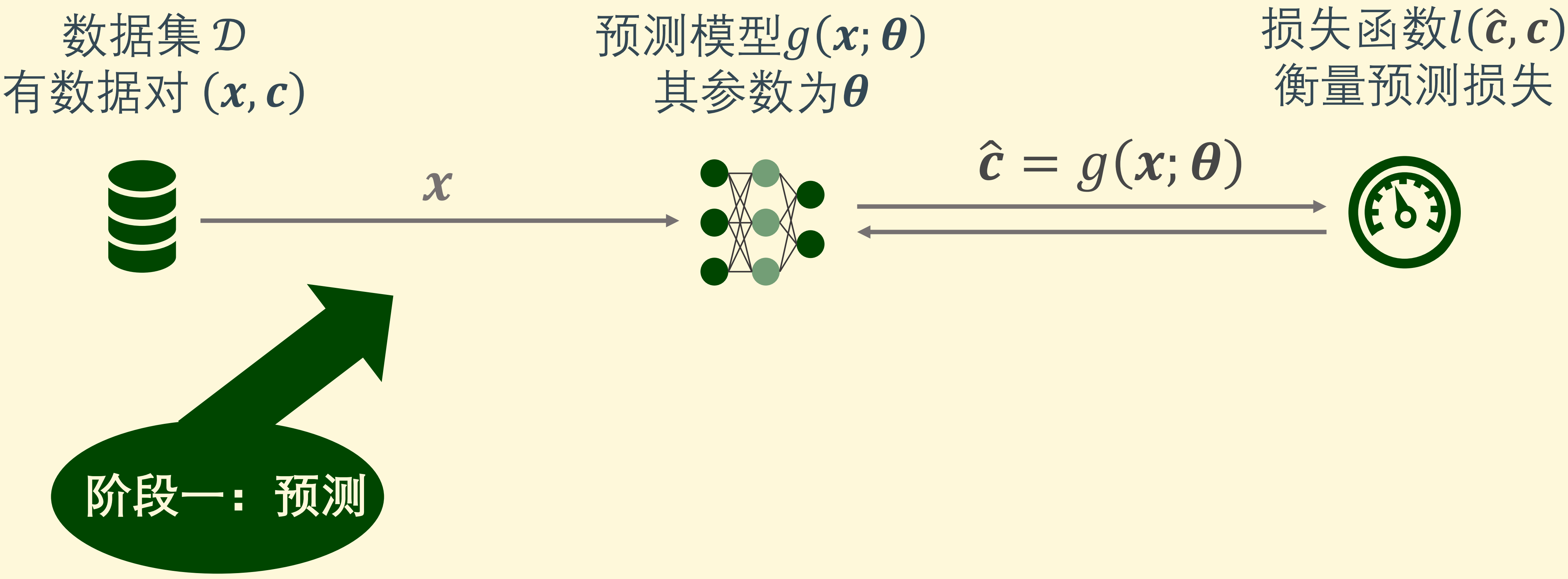


Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$

Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x}; \boldsymbol{\theta})$
for epochs **do**
  for each batch of training data $(\boldsymbol{x}, \boldsymbol{c})$ **do**
    Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
    Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x}; \boldsymbol{\theta})$
    Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \operatorname{argmin}_{\boldsymbol{w} \in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$
    Forward pass to compute decision loss $l(\cdot)$
    Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
  end for
end for

两阶段预测后优化

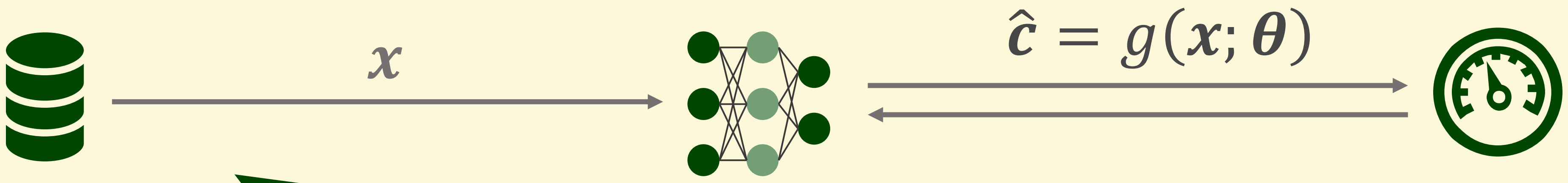

数据集 $\mathcal{D}$
有数据对 $(\boldsymbol{x}, \boldsymbol{c})$
预测模型$g(\boldsymbol{x}; \boldsymbol{\theta})$
其参数为$\boldsymbol{\theta}$
损失函数$l(\hat{\boldsymbol{c}}, \boldsymbol{c})$
衡量预测损失
$\boldsymbol{x}$
$\hat{\boldsymbol{c}} = g(\boldsymbol{x}; \boldsymbol{\theta})$
阶段一：预测

两阶段预测后优化

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

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

损失函数$l(\hat{\boldsymbol{c}}, \boldsymbol{c})$
衡量预测损失



阶段一：预测

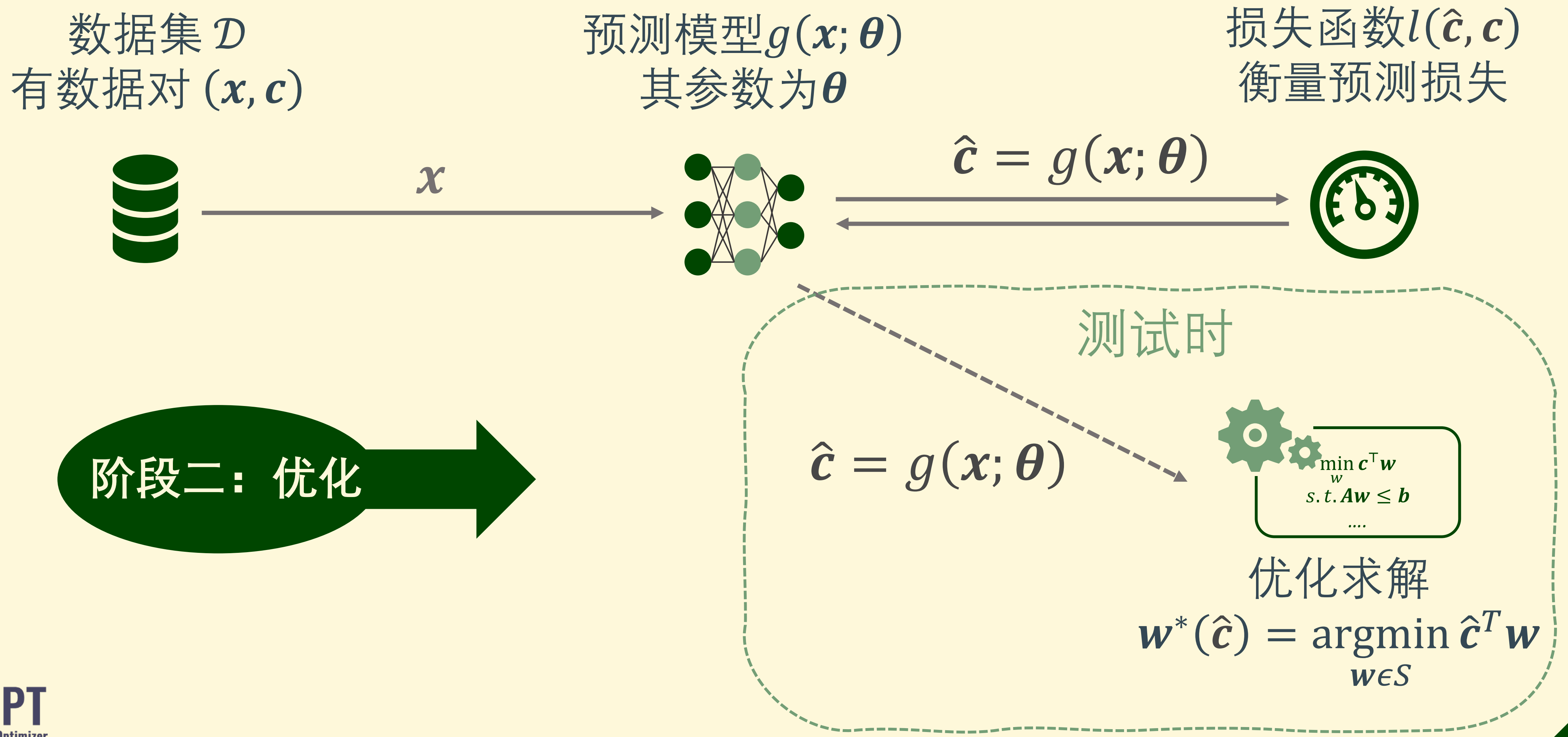
e.g.

$$l_{\mathrm{MSE}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \frac{1}{2} \|\boldsymbol{c} - \hat{\boldsymbol{c}}\|_2^2$$

$$l_{\mathrm{MAE}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \|\boldsymbol{c} - \hat{\boldsymbol{c}}\|_1$$

COPT
Cardinal Optimizer

两阶段预测后优化



COPT Cardinal Optimizer

预测和决策误差不匹配

例子

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

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

预测和决策误差不匹配

例子

$$\max_{w_1,w_2} c_1w_1 + c_2w_2$$
$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$

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$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 1$
- 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$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 2$

COPT Cardinal Optimizer

预测和决策误差不匹配

例子

$$\max_{w_1,w_2} c_1w_1 + c_2w_2$$
$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$

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$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 1$
- 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$l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = 2$

像 $l_{MSE}(\hat{c}, c)$ 这样的预测误差，不能准确地衡量决策的质量。

COPT Cardinal Optimizer

预测和决策误差不匹配

例子

$\max c$
w
$s.t$

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

- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (1,0)$，其最优解
- 当我们将成本向量预测为$\hat{\boldsymbol{c}} = (0,3)$，其最优解

像 $l_{MSE}(\hat{\boldsymbol{c}}, \boldsymbol{c})$ 这样的预测误差
策的质量。

模仿学习和可微分优化

直接预测：$\hat{\boldsymbol{w}}^* = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$

令$\hat{\boldsymbol{w}}^*$与真实最优解$\boldsymbol{w}^*$接近　**模仿学习**

令目标函数$f(\hat{\boldsymbol{w}}^*)$降低　**可微分优化**

COPT
Cardinal Optimizer

模仿学习和可微分优化

直接预测：$\hat{\boldsymbol{w}}^* = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta})$

- 令$\hat{\boldsymbol{w}}^*$与真实最优解$\boldsymbol{w}^*$接近 **模仿学习**
- 令目标函数$f(\hat{\boldsymbol{w}}^*)$降低 **可微分优化**

计算效率

规避了计算效率的主要瓶颈：优化求解

模仿学习和可微分优化

直接预测： $\hat{w}^* = g(x, \theta)$

令$\hat{w}^*$与真实最优解w^*接近　**模仿学习**

令目标函数$f(\hat{w}^*)$降低　**可微分优化**

计算效率

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

COPT
Cardinal Optimizer

链式法则求导

Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$

1: Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x}; \boldsymbol{\theta})$
2: **for** epochs **do**
3:     **for** each batch of training data $(\boldsymbol{x}, \boldsymbol{c})$ **do**
4:         Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
5:         Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x}; \boldsymbol{\theta})$
6:         Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \text{argmin}_{\boldsymbol{w} \in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$
7:         Forward pass to compute decision loss $l(\cdot)$
8:         Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
9:     **end for**
10: **end for**

梯度:

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

链式法则求导

Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$

Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x}; \boldsymbol{\theta})$
for epochs **do**
  for each batch of training data $(\boldsymbol{x}, \boldsymbol{c})$ **do**
    Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
    Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x}; \boldsymbol{\theta})$
    Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \text{argmin}_{\boldsymbol{w} \in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$
    Forward pass to compute decision loss $l(\cdot)$
    Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
  end for
end for

梯度:

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

链式法则求导

Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$

1. Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x}; \boldsymbol{\theta})$
2. **for** epochs **do**
3.   **for** each batch of training data $(\boldsymbol{x}, \boldsymbol{c})$ **do**
4.   Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
5.   Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x}; \boldsymbol{\theta})$
6.   Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \text{argmin}_{\boldsymbol{w} \in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$
7.   Forward pass to compute decision loss $l(\cdot)$
8.   Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
9.   **end for**
10. **end for**

梯度:

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

简单:

预测模型参数的梯度

链式法则求导

Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$

1: Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x};\boldsymbol{\theta})$
2: **for** epochs **do**
3:     **for** each batch of training data $(\boldsymbol{x},\boldsymbol{c})$ **do**
4:         Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
5:         Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x};\boldsymbol{\theta})$
6:         Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \text{argmin}_{\boldsymbol{w}\in S}\hat{\boldsymbol{c}}^T\boldsymbol{w}$
7:         Forward pass to compute decision loss $l(\cdot)$
8:         Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
9:     **end for**
10: **end for**

梯度：

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

困难：

决策损失随预测成本的变化而变化

链式法则求导

Algorithm 1 End-to-end Learning

Require: coefficient matrix $\boldsymbol{A}$, right-hand side $\boldsymbol{b}$, data $\mathcal{D}$
1: Initialize predictor parameters $\boldsymbol{\theta}$ for predictor $g(\boldsymbol{x};\boldsymbol{\theta})$
2: **for** epochs **do**
3:     **for** each batch of training data $(\boldsymbol{x}, \boldsymbol{c})$ **do**
4:         Sample batch of the cost vectors $\boldsymbol{c}$ with the corresponding features $\boldsymbol{x}$
5:         Predict cost using predictor $\hat{\boldsymbol{c}} := g(\boldsymbol{x};\boldsymbol{\theta})$
6:         Forward pass to compute optimal solution $\boldsymbol{w}^*_{\hat{\boldsymbol{c}}} := \operatorname{argmin}_{\boldsymbol{w}\in S} \hat{\boldsymbol{c}}^T \boldsymbol{w}$
7:         Forward pass to compute decision loss $l(\cdot)$
8:         Backward pass from loss $l(\cdot)$ to update parameters $\boldsymbol{\theta}$ with gradient
9:     **end for**
10: **end for**

梯度:

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

困难：

决策损失随预测成本的变化而变化

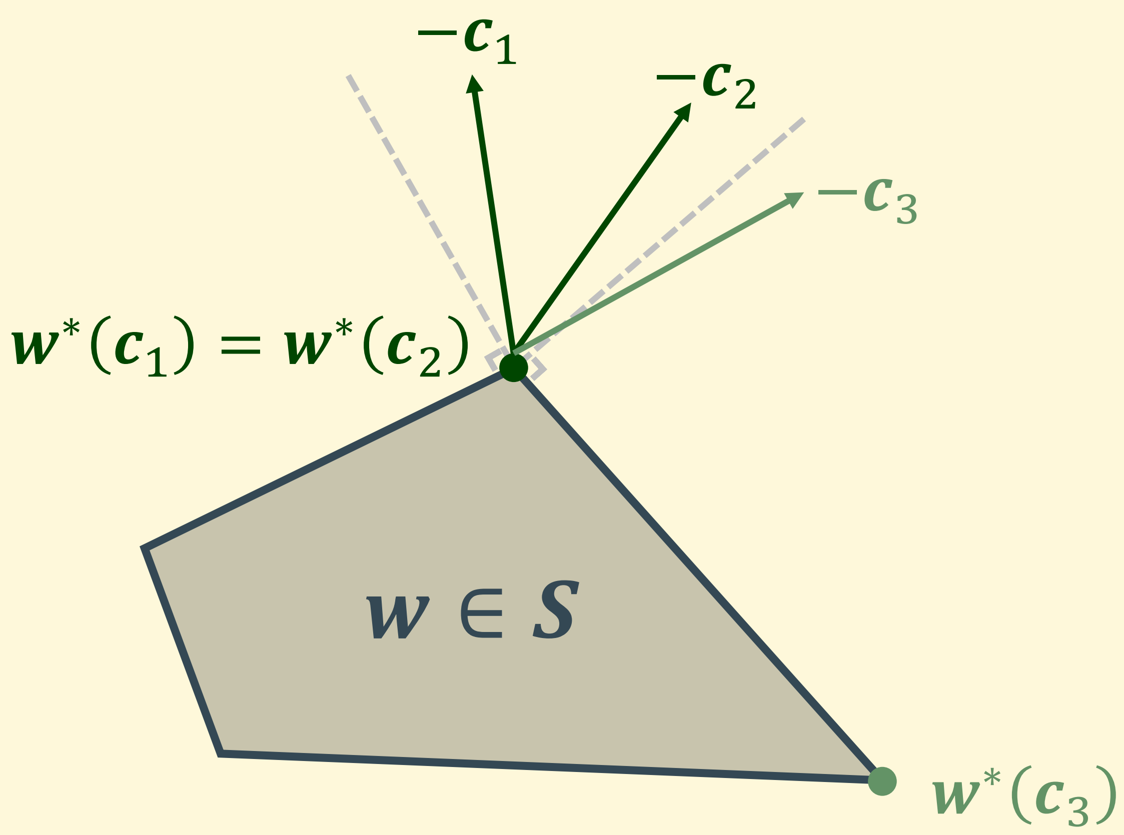
$$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^T \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^\top \boldsymbol{w}^*(\boldsymbol{c})$$

$$l_{\text{Square}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \frac{1}{2} \|\boldsymbol{w}^*(\boldsymbol{c}) - \boldsymbol{w}^*(\hat{\boldsymbol{c}})\|_2^2$$

与最优解$\boldsymbol{w}^*(\hat{\boldsymbol{c}})$有关!

COPT Cardinal Optimizer

分片常数函数



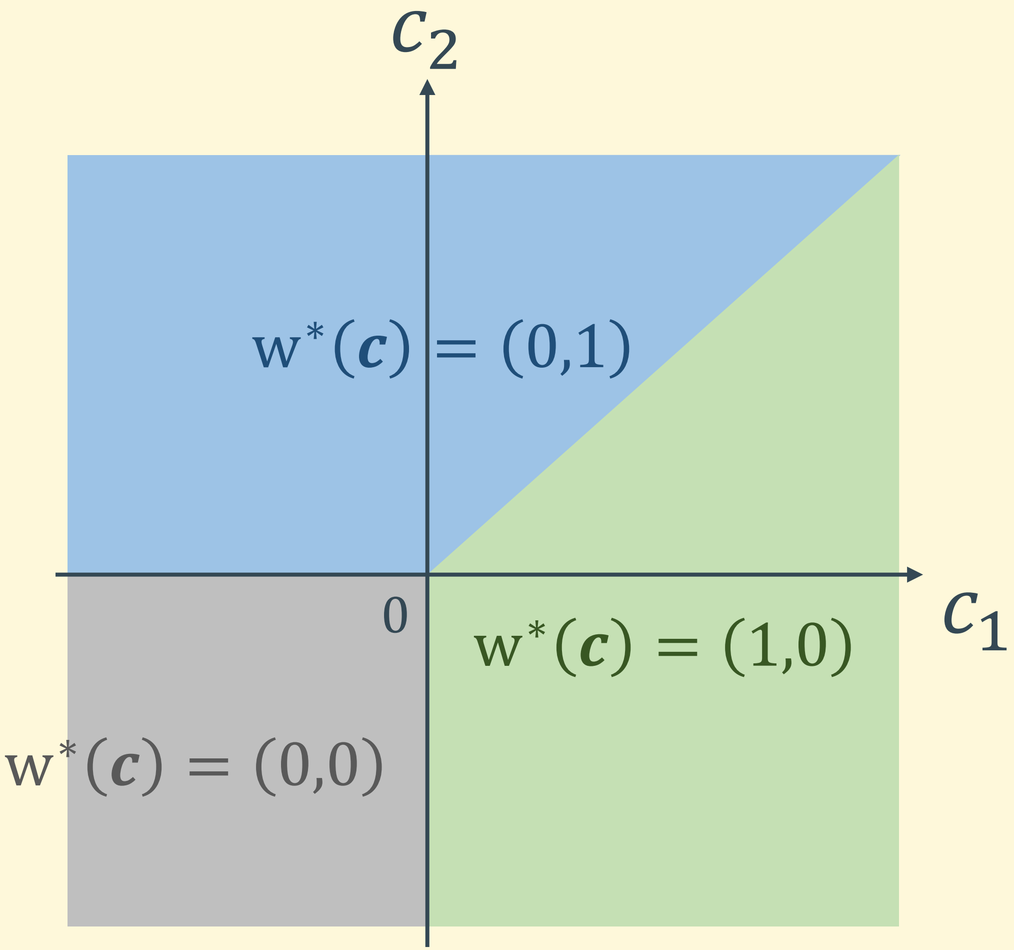
线性规划最优解$w^*(c)$作为成本参数c的函数，是一个分片常数函数！

COPT Cardinal Optimizer

分片常数函数

例子

$$\max_{w_1,w_2} c_1w_1 + c_2w_2$$
$$s.t. \quad w_1 + w_2 \leq 1$$
$$w_1, w_2 \geq 0$$



COPT Cardinal Optimizer

分段常数函数

谢谢观看，散了吧

分段常数函数

~~谢谢观看，散了吧~~

替代损失/梯度！

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获得梯度

Karush-Kuhn-Tucker conditions

Given general problem

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

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)
- $h_i(x) \leq 0, \ \ell_j(x) = 0$ for all i, j (primal feasibility)
- $u_i \geq 0$ 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).

COPT
Cardinal Optimizer

Smart “predict, then optimize”

$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^\top \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^\top \boldsymbol{w}^*(\boldsymbol{c})$的凸上界:

COPT Cardinal Optimizer

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

Smart “predict, then optimize”

$$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^{\top}\boldsymbol{w}^{*}(\hat{\boldsymbol{c}}) - \boldsymbol{c}^{\top}\boldsymbol{w}^{*}(\boldsymbol{c})\text{的凸上界:}$$

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

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

Smart “predict, then optimize”

$$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^\top \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^\top \boldsymbol{w}^*(\boldsymbol{c})$$的凸上界:

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

计算开销：每一个迭代都需要求解一个优化问题 $\min_{\boldsymbol{w} \in \boldsymbol{W}} (2\hat{\boldsymbol{c}} - \boldsymbol{c})^\top \boldsymbol{w}$.

COPT Cardinal Optimizer

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

Smart "predict, then optimize"

$l_{\text{Regret}}(\hat{\boldsymbol{c}}, \boldsymbol{c}) = \boldsymbol{c}^\top \boldsymbol{w}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^\top \boldsymbol{w}^*(\boldsymbol{c})$的凸上界：

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

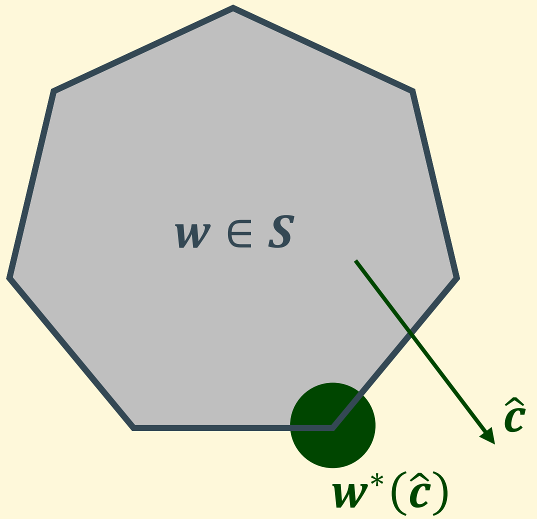
可计算$l_{\text{SPO+}}(\hat{\boldsymbol{c}}, \boldsymbol{c})$的次梯度

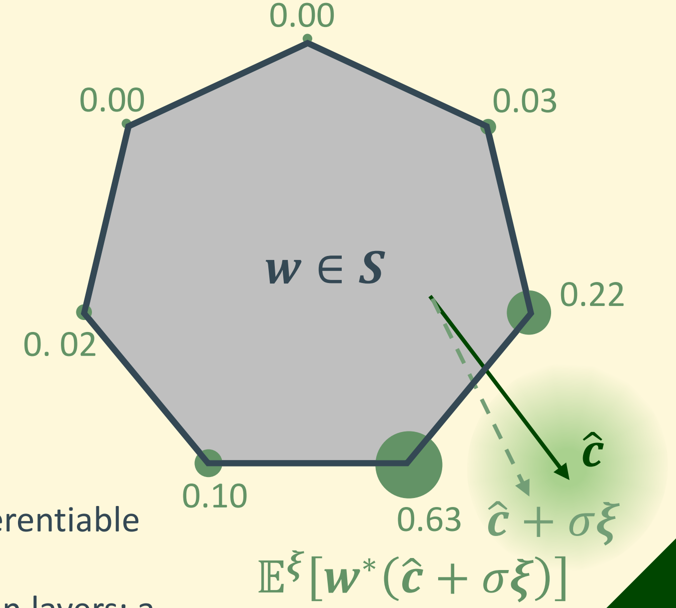
$$2\boldsymbol{w}^*(\boldsymbol{c}) - 2\boldsymbol{w}^*(2\hat{\boldsymbol{c}} - \boldsymbol{c}) \in \frac{\partial l_{\text{SPO+}}(\hat{\boldsymbol{c}}, \boldsymbol{c})}{\partial \hat{\boldsymbol{c}}}$$

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

扰动方法

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





- 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.c

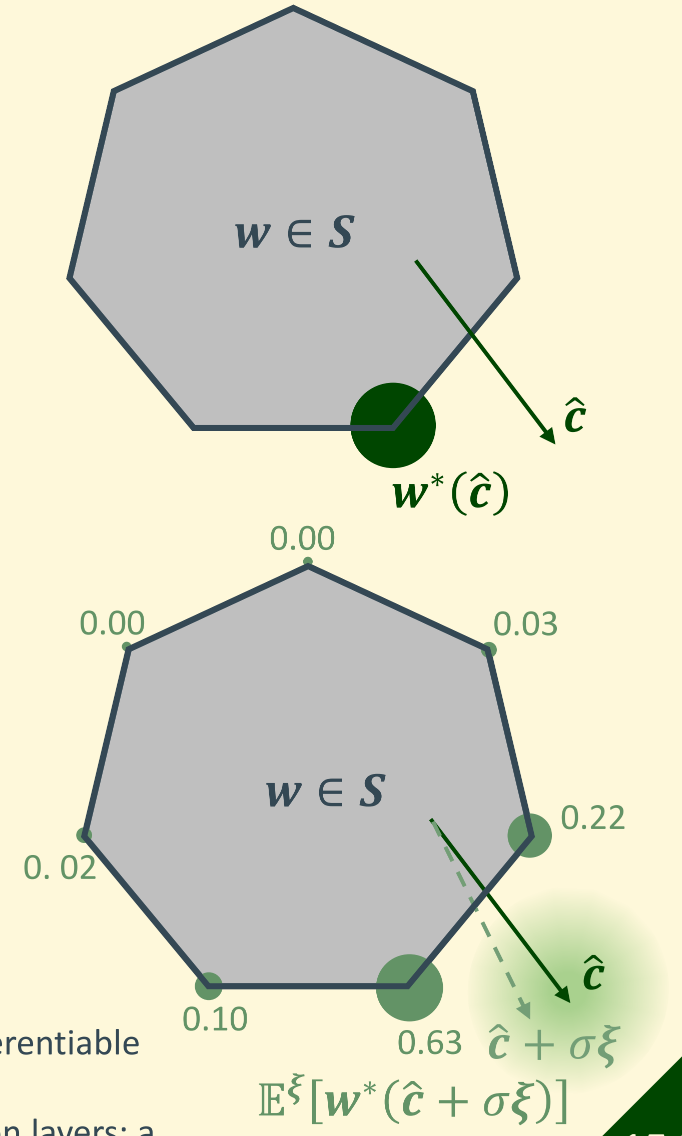
COPT Cardinal Optimizer

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）。

对预测成本向量$\hat{\boldsymbol{c}}$进行随机扰动：$\boldsymbol{\xi} \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{1})$



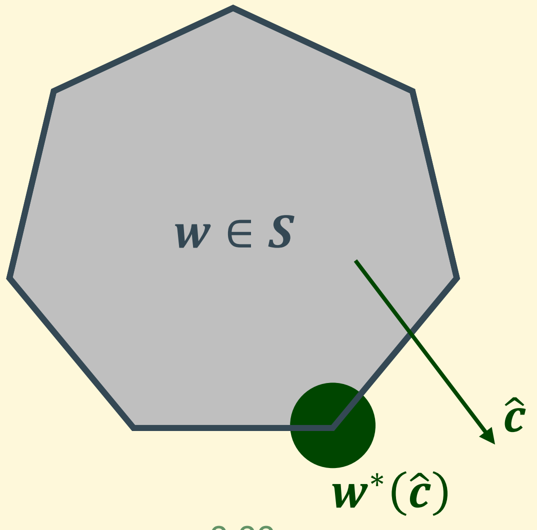
- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization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

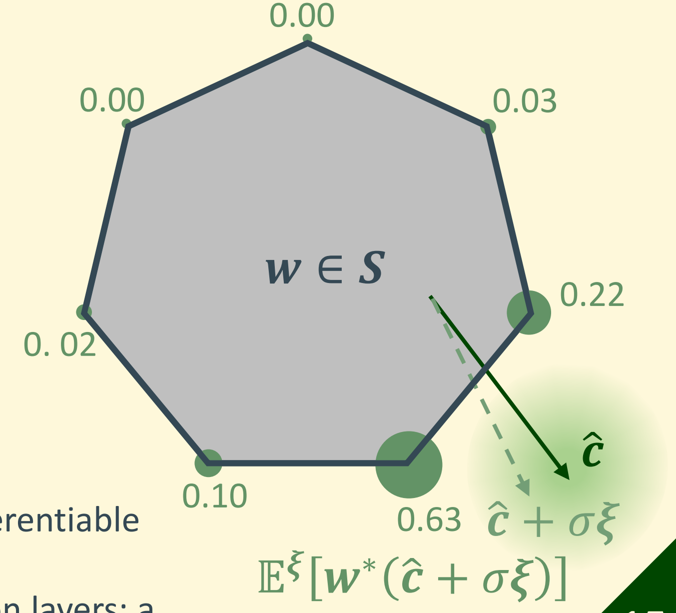
扰动方法

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）。

如何求期望？





- 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.c

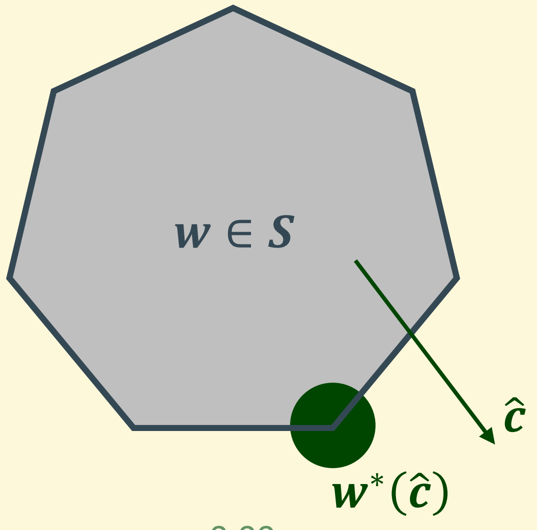
扰动方法

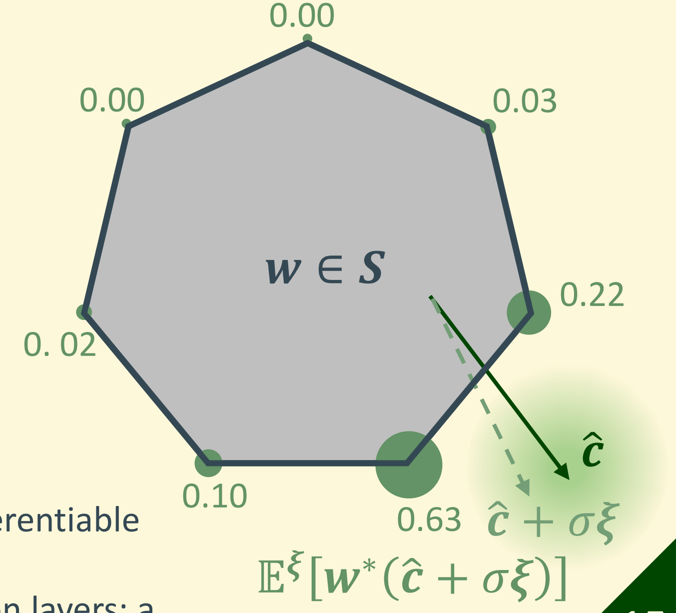
随机扰动来处理成本向量的预测值$\hat{\boldsymbol{c}}$。

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）。

如何求期望?

$$\mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})] \approx \underbrace{\frac{1}{\mathrm{K}}\sum_{\kappa}^{\mathrm{K}} \boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi}_{\boldsymbol{\kappa}})}_{\text{蒙特卡洛采样：需求解}K\text{个优化问题}}$$



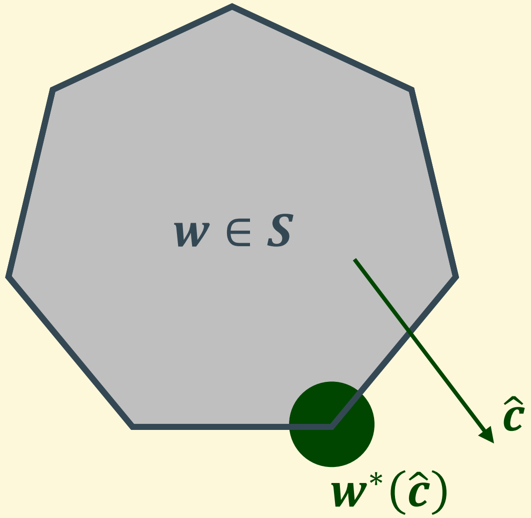


- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization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

扰动方法

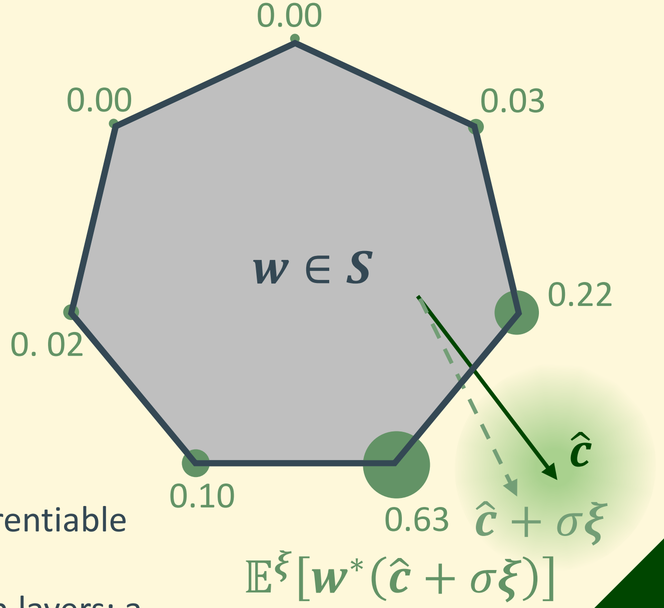
随机扰动来处理成本向量的预测值$\hat{\boldsymbol{c}}$。

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）。



如何求期望?

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



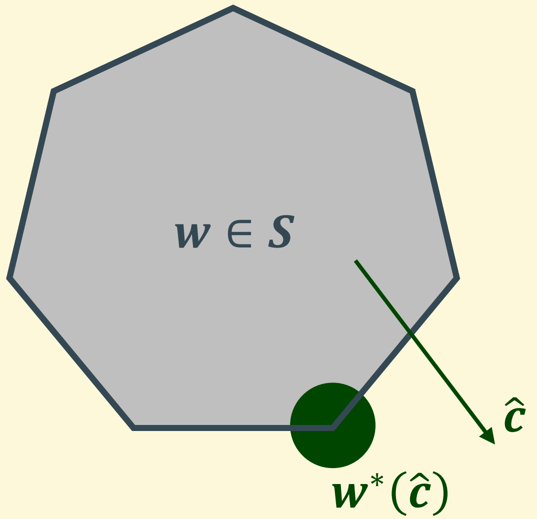
对成本向量$\hat{\boldsymbol{c}}$有非负性的要求：乘法扰动

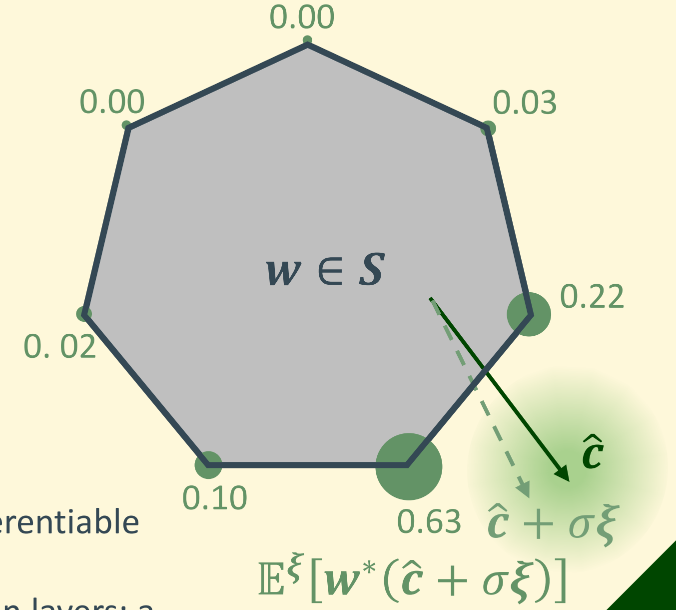
- Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J. P., & Bach, F. (2020). Learning with differentiable pertubed optimizers. Advances in neural information processing systems, 33, 9508-9519.
- Dalle, G., Baty, L., Bouvier, L., & Parmentier, A. (2022). Learning with combinatorial optimization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

COPT Cardinal Optimizer

扰动方法

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





- 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.c

COPT
Cardinal Optimizer

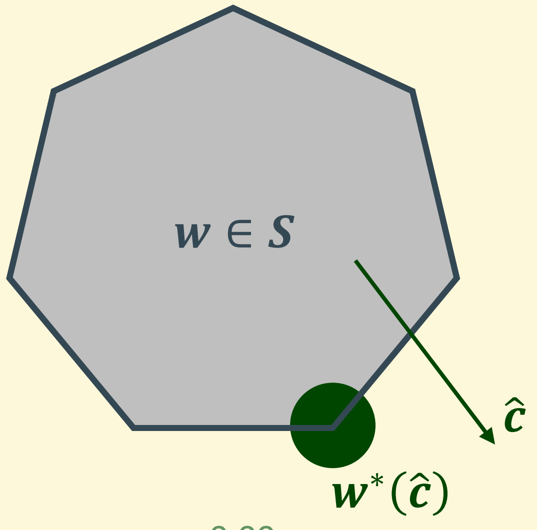
扰动方法

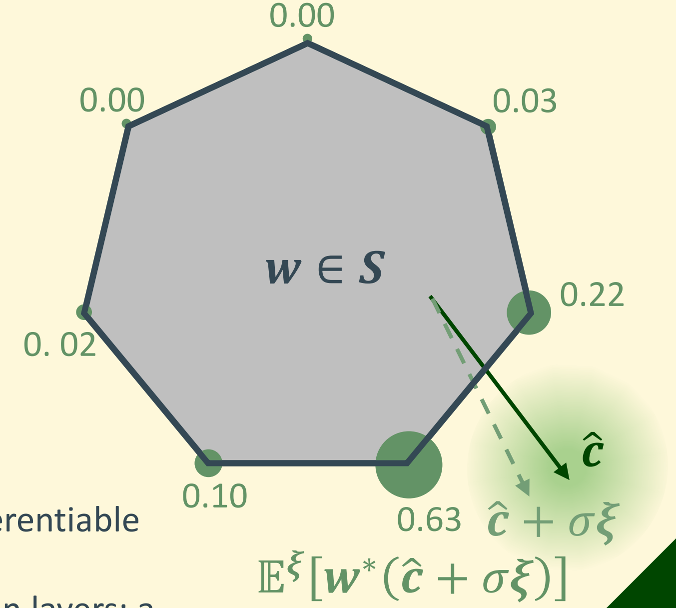
有期望目标函数$F^{\xi}(\boldsymbol{c}) = \mathbb{E}^{\xi}\left[\min_{\boldsymbol{w}\in \boldsymbol{W}}(\boldsymbol{c}+\sigma\boldsymbol{\xi})^{T}\boldsymbol{w}\right]$

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

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

减小对偶间隙





- 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.c

COPT Cardinal Optimizer

扰动方法

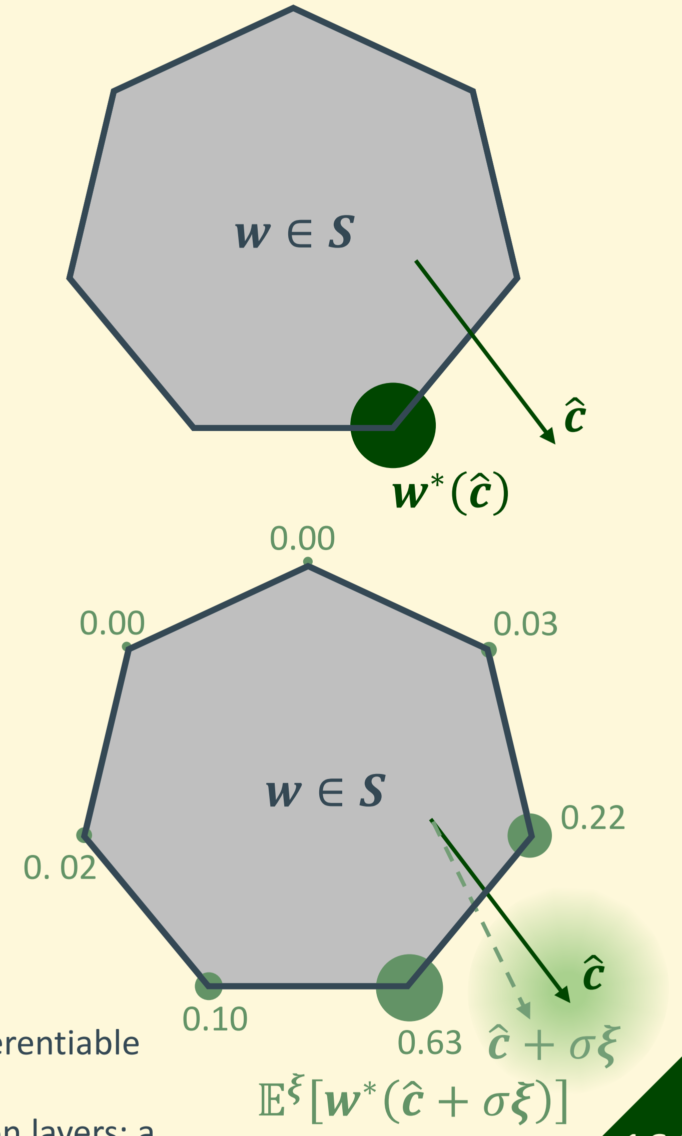
有期望目标函数$F^{\xi}(\boldsymbol{c}) = \mathbb{E}^{\xi}\left[\min_{\boldsymbol{w}\in\boldsymbol{W}}(\boldsymbol{c}+\sigma\boldsymbol{\xi})^T\boldsymbol{w}\right]$

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

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

对于$\hat{\boldsymbol{c}}$是常数

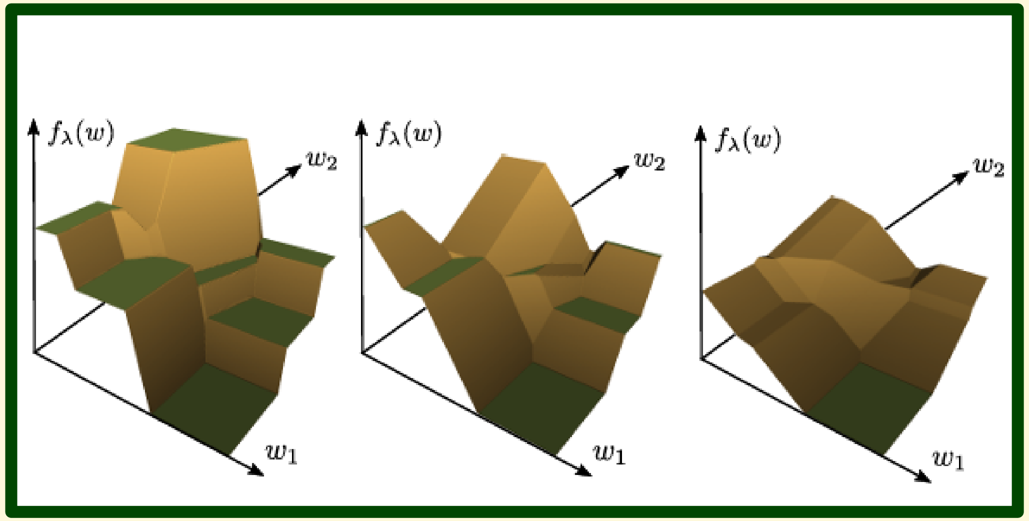
$$\frac{\partial l_{\text{PFY}}(\hat{\boldsymbol{c}},\boldsymbol{w}^*(\boldsymbol{c}))}{\partial\hat{\boldsymbol{c}}} = \boldsymbol{w}^*(\boldsymbol{c}) - \mathbb{E}^{\xi}[\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi})]$$
$$\approx \boldsymbol{w}^*(\boldsymbol{c}) - \frac{1}{\mathrm{K}}\sum_{\kappa}^{\mathrm{K}}\boldsymbol{w}^*(\hat{\boldsymbol{c}}+\sigma\boldsymbol{\xi}_{\kappa})$$



- 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.c

COPT
Cardinal Optimizer

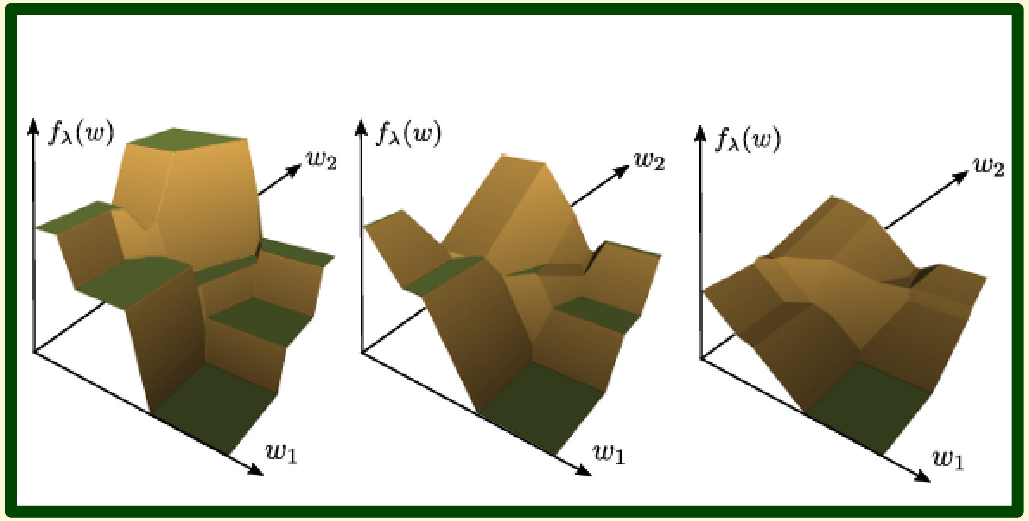
黑箱方法



- “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.

黑箱方法



- “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.

COPT
Cardinal Optimizer

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.

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：**

将次优解的子集$\Gamma \backslash \boldsymbol{w}^*(\boldsymbol{c})$作为负样本，让最优解和次优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.

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：**

将次优解的子集$\Gamma\backslash \boldsymbol{w}^*(\boldsymbol{c})$作为负样本，让最优解和次优解之间的的差值尽可能大

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

- **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Γ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.

开源软件

GitHub

扫码查看PyEPO GitHub Repo:
https://github.com/khalil-research/PyEPO

开源软件

数据集

- Shortest Path
- Knapsack
- Traveling Salesperson

运筹优化

COPT Cardinal Optimizer

PYOMO

其他求解器及算法

机器学习

PyTorch

scikit learn

端到端算法

- SPO+
- DBB
- NID
- DPO
- PFYL
- NCE
- LTR
- ...

代码实战

```python
from coptpy import COPT
from coptpy import Envr
from pyepo.model.copt import optCoptModel

class myOptModel(optCoptModel):
    def _getModel(self):
        # ceate a model
        m = Envr().createModel()
        # varibles
        x = m.addVars(5, nameprefix='x', vtype=COPT.BINARY)
        # sense
        m.setObjSense(COPT.MAXIMIZE)
        # constraints
        m.addConstr(3*x[0]+4*x[1]+3*x[2]+6*x[3]+4*x[4]<=12)
        m.addConstr(4*x[0]+5*x[1]+2*x[2]+3*x[3]+5*x[4]<=10)
        m.addConstr(5*x[0]+4*x[1]+6*x[2]+2*x[3]+3*x[4]<=15)
        return m, x

optmodel = myOptModel()
```

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

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

演示代码：

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

COPT Cardinal Optimizer

自动求导函数细节

端到端算法

- SPO+
- DBB
- NID
- DPO
- PFYL
- NCE
- LTR
- …

`pyepo.func.perturbedFenchelYoung` allows us to set a Fenchel-Young loss for training, which requires parameters:

- `optmodel` : a PyEPO optimization model
- `n_samples` : number of Monte-Carlo samples
- `sigma` : the amplitude of the perturbation for costs
- `processes` : number of processors for multi-thread, 1 for single-core, 0 for all of the cores
- `seed` : random state seed for perturbations

```
import pyepo

# init SPO+ loss
spop = pyepo.func.SPOPlus(optmodel, processes=2)
# init PFY loss
pfy = pyepo.func.perturbedFenchelYoung(optmodel, n_samples=3, sigma=1.0, processes=2)
# init NCE loss
nce = pyepo.func.NCE(optmodel, processes=2, solve_ratio=0.05, dataset=dataset_train)
```

感谢指导

GitHub

扫码查看PyEPO GitHub Repo

COPT

扫码加入COPT QQ技术交流群
申请试用COPT www.shanshu.ai/copt

Mechanical & Industrial Engineering
UNIVERSITY OF TORONTO

主讲人：唐博

2023年11月29日