5.9 DC 规划及 CCCP 算法

SMaLL

[1] 中国石油大学（华东）
SMaLL 课题组
small.sem.upc.edu.cn
liangxijunsd@163.com

2023

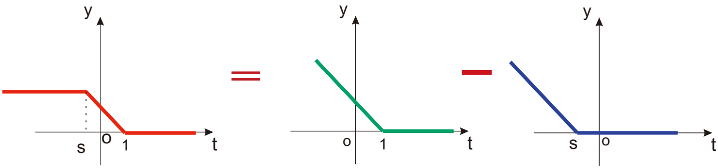
DC 函数

定义 1

设 f 是 $\mathbb{R}^n$ 到 $\mathbb{R}$ 的实值函数，如果存在凸函数 g，h：$\mathbb{R}^n \to \mathbb{R}$，使 f 可以被分解为 g 和 h 的差值，那么 f 就称为 DC 函数，即

$$f(\mathbf{x}) = g(\mathbf{x}) - h(\mathbf{x}), \qquad \forall \mathbf{x} \in \mathbb{R}^n.$$

E.g.1 ramp 损失函数



$$\text{ramp}_s(t) = h_1(t) - h_s(t)$$

其中 $\text{ramp}_s(t) = \min(1-s, \max(0, 1-t))$， $s < 1$ 为参数.
分解得到两个凸函数分别为：

$$h_s(t) = \max(0, s-t), h_1(t) = \max(0, 1-t).$$

E.g.2 指数平方损失函数

考虑指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

E.g.2 指数平方损失函数

考虑指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

E.g.2 指数平方损失函数

考虑指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

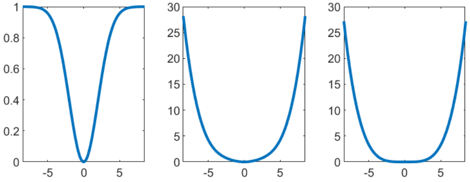


图: 左: 损失平方函数 $\phi_\gamma(t)$; 中: $\phi_\gamma(t) + v(t)$; 右: $v(t)$

DC 函数

Proposition 1 (命题)

令 f_i，$i = 1, \dots m.$ 为 *DC* 函数，那么下列形式函数都属于 *DC* 函数。

(1) $\sum_i \lambda_i f_i(x)$, *for* $\lambda_i \in \mathbb{R}$

(2) $\max_i f_i(x)$

(3) $\min_i f_i(x)$

(4) $\prod_i f_i(x)$

(5) 连续二阶可微函数 f

(6) 如果函数 f 是 *DC* 函数，且函数 g 是凸函数，那么复合函数 $(g \circ f)$ 也是 *DC* 函数.

(7) 凸集合上的每个连续函数都可以表示为一致收敛的 *DC* 函数的极限.

DC 规划

DC 规划

- 无约束 DC 规划:

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \quad f_0(\mathbf{x})$$

$f_0 : \mathbb{R}^n \to \mathbb{R}$ 是一个 DC 函数.

- 有约束 DC 规划:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0, i = 1, \ldots, m. \end{aligned}$$

其中 $f_i : \mathbb{R}^n \to \mathbb{R}$ 是 DC 函数, $i = 0, \ldots, m$.

基本 DC 规划算法 (DCA)

初始化: 令 $x^0 \in \operatorname{dom}\partial h, k=0.$

遍历 $k=0,1,\ldots$ 直到 $\{x^k\}$ 收敛;

步 1: 计算 $y^k \in \partial h(x^k)$;

步 2: 计算

$$x^{k+1} \in \operatorname{argmin}\{g(x)-h_k(x): x\in\mathbb{R}^n\} \qquad (P_k)$$

- $h_k(x)=h(x^k)+\langle y^k, x-x^k\rangle$;
- $x\in\mathbb{R}^n \to x\in C$

CCCP 算法

CCCP 是 DCA 在光滑优化中的一个示例.

- 假设目标函数 f 是一个二阶可微函数
  $\Rightarrow f$ 是 DC 函数, $f(x) = f_{\text{vex}}(x) + f_{\text{cav}}(x)$
- 假设可行集是凸集 C;
- CCCP 算法每次迭代都用切线逼近凹函数，并极小化得到的凸函数:

$$x^{k+1} \in \arg\min \left\{ f_{vex}(x) + \left\langle x, \nabla f_{cav}(x^k) \right\rangle : x \in C \right\}$$

CCCP (Convex-Concave Procedure)

$$\min_{x} \quad f_{\text{vex}}(x) + f_{\text{cav}}(x)$$

SMaLL
Statistical Machine Learning Lyceum

CCCP (Convex-Concave Procedure)

$$\min_{x} \quad f_{\text{vex}}(x) + f_{\text{cav}}(x)$$

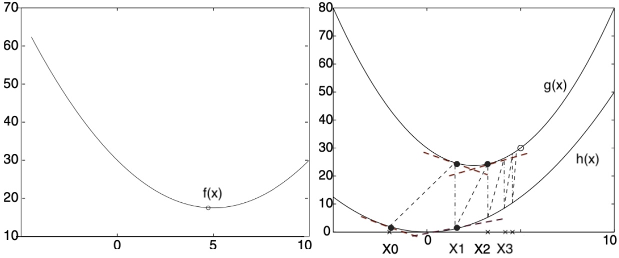
算法 Concave-Convex procedure CCCP 对于求解 x

1: **初始化:** 令 $x^0 \in C$,$k = 0$
2: **遍历** $k = 0, 1, \ldots$
3: $\quad x^{k+1} = \operatorname{argmin}_x f_{\text{vex}}(x) + \langle \nabla f_{\text{cav}}(x^k), x \rangle$
4: **直到** $\{x^k\}$ 收敛.

CCCP (Convex-Concave Procedure)

对于凸函数 g, h:

$$\min_x f(x) = g(x) - h(x)$$

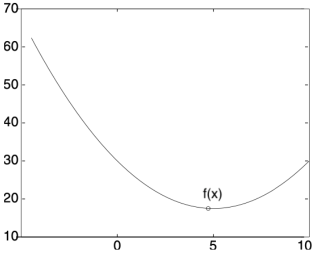


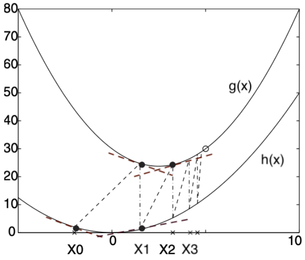
- 当 $g, h \in \mathcal{C}^1$ 时 $\Rightarrow f \in \mathcal{C}^1$;

CCCP (Convex-Concave Procedure)

对于凸函数 g, h:

$$\min_x f(x) = g(x) - h(x)$$



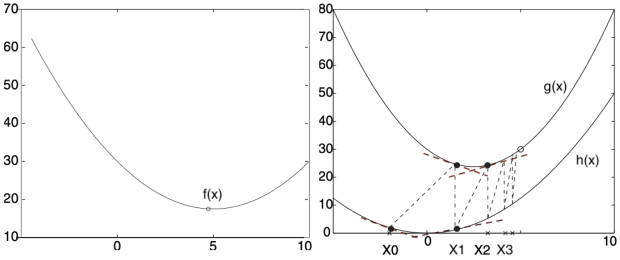


- 当 $g, h \in \mathcal{C}^1$ 时 $\Rightarrow f \in \mathcal{C}^1$;
- $x^* \in \operatorname{argmin}_x f \Rightarrow \nabla f(x^*) = \nabla g(x^*) - \nabla h(x^*) = 0$;

CCCP (Convex-Concave Procedure)

对于凸函数 g, h:

$$\min_x f(x) = g(x) - h(x)$$

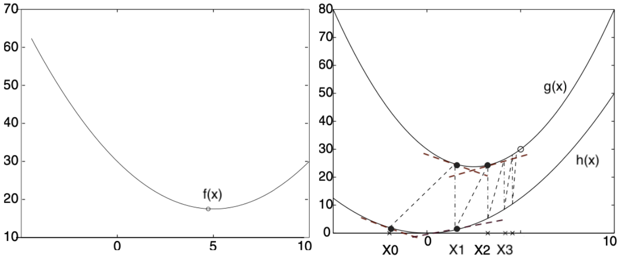


- 存在 $x^{k+1} \in \operatorname{argmin}_x [g(x) - h_k(x)]$

CCCP (Convex-Concave Procedure)

对于凸函数 g, h:

$$\min_x f(x) = g(x) - h(x)$$



- 存在 $x^{k+1} \in \text{argmin}_x[g(x) - h_k(x)]$
- $\Rightarrow$ 寻找 x^{k+1} 使得: $\nabla g(x^{k+1}) - \nabla h(x^k) = 0$.

DCA 的性质

初始化: 令 $x^0 \in \operatorname{dom} \partial h, k = 0.$

遍历 $k = 0, 1, \ldots$ 直到 $\{x^k\}$ 收敛:

step1: 计算 $y^k \in \partial h(x^k)$;

step2: 计算 $x^{k+1} \in \operatorname{argmin}\{g(x) - h_k(x) : x \in \mathbb{R}^n\} \quad (P_k).$

Statistical Machine Learning Lyceum

DCA 的性质

初始化: 令 $x^0 \in \operatorname{dom} \partial h, k = 0.$

遍历 $k = 0, 1, \ldots$ 直到 $\{x^k\}$ 收敛:

step1: 计算 $y^k \in \partial h\,(x^k)$;

step2: 计算 $x^{k+1} \in \operatorname{argmin}\{g(x) - h_k(x) : x \in \mathbb{R}^n\} \quad (P_k).$

- 最优解集为 (P_k): $\partial g^*\,(y^k)$
- DCA 方法可以用另一种形式表示:

$$\text{遍历 } k = 0, 1, \ldots$$

$$\text{令} \quad y^k \in \partial h\,(x^k), \qquad x^{k+1} \in \partial g^*\,(y^k).$$

DCA 的性质

DCA 是一种无需线搜索的下降法，但具有全局收敛性.

- 如果问题的最优值　是有限的且无限序列 $\{x^k\}$ 是有界的，则该序列的每一个极限点 x^* 都是 $g-h$ 的一个临界点

DCA 的性质

DCA 是一种无需线搜索的下降法，但具有全局收敛性.

- 如果问题的最优值　是有限的且无限序列 $\{x^k\}$ 是有界的，则该序列的每一个极限点 x^* 都是 $g-h$ 的一个临界点
- 对一般的 DC 规划，DCA 具有线性收敛性.

DCA 的性质

DCA 是一种无需线搜索的下降法，但具有全局收敛性.

- 如果问题的最优值　是有限的且无限序列 $\{x^k\}$ 是有界的，则该序列的每一个极限点 x^* 都是 $g-h$ 的一个临界点
- 对一般的 DC 规划，DCA 具有线性收敛性.
- 在 多面体 DC 规划中 $\Rightarrow$ 序列 $\{x^k\}$ 包含有限多个元素，经有限多次迭代后，DCA 收敛到临界点 x^*.

近似 DCA

- 基本的 DCA 方法: $y^k \in \partial h(x^k)$, $x^{k+1} \in \partial g^*(y^k)$.
- 难点. 这些计算不一定都是准确的
- 解决方法. 使用近似 DCA 方法:

$$y^k \in \partial_{\varepsilon_k} h\left(x^k\right); \quad x^{k+1} \in \partial_{\varepsilon_k} g^*\left(y^k\right).$$

- 已经被证明:approximate DCA 方法仍然收敛到一个临界点为 $\varepsilon_k \downarrow 0$.

DC 规划

Statistical Machine Learning Lyceum

Computational Statistics and Data Analysis 155 (2021) 107094

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

journal homepage: www.elsevier.com/locate/csda

ELSEVIER

Robust variable selection with exponential squared loss for the spatial autoregressive model[☆]

Yunquan Song [a,*], Xijun Liang [a,*], Yanji Zhu [a], Lu Lin [b]

[a] College of Science, China University of Petroleum, Qingdao 266580, PR China
[b] Zhongtai Securities Institute for Financial Studies, Shandong University, Jinan 250014, PR China

ARTICLE INFO

ABSTRACT

Sec. 3 应用

考虑如下优化模型

$$\min_{\beta \in R^p, \rho \in [0,1]} \quad L(\beta, \rho) = \frac{1}{n} \sum_{i=1}^{n} \phi_\gamma(Y_i - \rho \tilde{Y}_i - X_i \beta) + \lambda \sum_{j=1}^{p} P(|\beta_j|) \qquad (1)$$

其中 $\lambda > 0$, $\tilde{Y} = WY$, $\sum_{j=1}^{p} P(|\beta_j|)$ 惩罚项, $\phi_\gamma(\cdot)$ 是 指数平方损失函数:

$$\phi_\gamma(t) = 1 - \exp(-t^2/\gamma).$$

Algorithm 1 The block coordinate descent (BCD) algorithm

1: 令初始值 $\beta^0 \in \mathbb{R}^p$ 且 $\rho^0 \in (0,1)$;

2: 遍历 $k = 0, 1, 2, \cdots$

3: 使用初始点 ρ 对 ρ^k 进行迭代:

$$\rho^{k+1} \leftarrow \min_{\rho \in [0,1]} L(\beta^k, \rho); \tag{2}$$

4: 进一步更新 β^k, 通过

$$\min_{\beta \in R^p} L(\beta, \rho^{k+1}) \tag{3}$$

可得到 β^{k+1}, 需保证 $L(\beta^k, \rho^{k+1}) - L(\beta^{k+1}, \rho^{k+1}) \leq 0$, 且 β^{k+1} 在 $L(\beta, \rho^{k+1})$ 上是一个稳定点.

5: 令 $k \leftarrow k + 1$

6: 直到收敛.

E.g.2 指数平方损失函数

指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

E.g.2 指数平方损失函数

指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

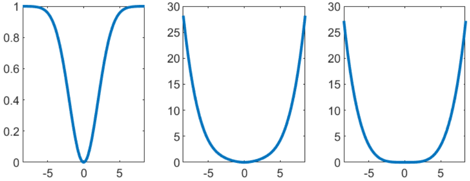


图: 左: 指数损失函数 $\phi_\gamma(t)$; 中: $\phi_\gamma(t) + v(t)$, 右: $v(t)$

E.g.2 指数平方损失函数

指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

$e^{\frac{t^2}{\gamma}}$ 可能会引起计算困难: 当 $\gamma = 1, t = 10$,

$e^{\frac{t^2}{\gamma}} = e^{100} \approx 2.688 \times 10^{43}$, 值过大 $\rightarrow$ 计算缺陷.

E.g.2 指数平方损失函数

指数平方损失函数:

$$\phi_\gamma(t) = 1 - e^{-\frac{t^2}{\gamma}}.$$

DC 分解:

$$\phi_\gamma(t) = [\phi_\gamma(t) + v(t)] - v(t),$$

其中 $\phi_\gamma(t) + v(t)$ 也是凸函数. 尝试可得函数 $v(t) = e^{\frac{t^2}{\gamma}}$ 是满足条件的函数之一.

Proposition 2

[另一种分解]

$$\phi_\gamma(t) := [\phi_\gamma(t) + v(t)] - v(t) := u(t) - v(t), \tag{4}$$

$v(t) = \frac{1}{3\gamma^2} t^4$, $u(t) = \phi_\gamma(t) + v(t)$.

目标函数的 DC 分解

记：

$$
\begin{aligned}
J_{vex}(\beta) &= \frac{1}{n}\sum_{i=1}^{n} u(Y_i - \rho^k\langle w_i, Y\rangle - X_i\beta) + \lambda\sum_{j=1}^{p} P(|\beta_j|), \\
J_{cav}(\beta) &= \frac{1}{n}\sum_{i=1}^{n} v(Y_i - \rho^k\langle w_i, Y\rangle - X_i\beta),
\end{aligned} \tag{5}
$$

其中 w_i 是权重矩阵 W 的第 i 行, $\sum_{j=1}^{p} P(|\beta_j|)$ 一个关于 β 的凸惩罚项.

求解函数 (3)

$$
\min_{\beta\in R^p} \quad L(\beta, \rho^k) = J_{vex}(\beta) + J_{cav}(\beta),
$$

$\longrightarrow$ 凹凸过程算法（CCCP）

Algorithm 2 The Concave-Convex Procedure (CCCP)

1. 初始化 β^0. 令 $k = 0$.
2. **repeat**
3. $$\beta^{k+1} = \mathrm{argmin}_\beta \quad J_{\mathrm{vex}}(\beta) + J'_{\mathrm{cav}}(\beta^k) \cdot \beta \tag{6}$$
4. **until** 收敛至 β^k.

求解 CCCP 问题 I I

- $J'_{\text{cav}}(\beta^k) \cdot \beta$ 关于 β 是线性的,
- $J_{\text{vex}}(\beta) + J'_{\text{cav}}(\beta^k) \cdot \beta \to$

$$\min_{\beta \in R^p} \quad \psi(\beta) + \lambda \sum_{i=1}^{p} P(|\beta_i|), \tag{7}$$

其中 $\psi(\beta) \in \mathcal{C}^1$ 是凸的, $\sum_{i=1}^{p} P(|\beta_i|)$ 是 Lasso 惩罚项, 或者自适应 Lasso 惩罚项, $\sum_{i=1}^{p} \eta_i |\beta_i|$, $\eta_i \geq 0$, $i = 1, \cdots, p$.

- [Beck and Teboulle(2009)]: ISTA 和 FISTA 算法, 使用结构 (7) 的 Lasso 惩罚项求解了该模型.
- $\to$ (泛化): 使用自适应 Lasso 惩罚项求解该模型.

求解 CCCP 问题 II

- ISTA 近似 $F(\beta) = \psi(\beta) + \lambda \sum_{i=1}^{p} \eta_i |\beta_i|$ at $\beta = \xi$ as:

$$Q_L(\beta, \xi) = \psi(\xi) + \langle \beta - \xi, \nabla\psi(\xi) \rangle + \frac{L}{2} \|\beta - \xi\|^2 + \lambda \sum_{i=1}^{p} \eta_i |\beta_i|.$$

- 这个函数拥有最小值点

$$\begin{aligned} \Theta_L(\xi) &= \operatorname{argmin}_{\beta \in R^p} \; Q_L(\beta, \xi) \\ &= \operatorname{argmin}_{\beta \in R^p} \; \{\lambda \sum_{i=1}^{p} \eta_i |\beta_i| + \frac{L}{2} \|\beta - (\xi - \frac{1}{L} \nabla\psi(\xi)\|^2\} \quad (8) \\ &= \mathcal{S}_{\lambda\eta/L}(\xi - \frac{1}{L} \nabla\psi(\xi)), \end{aligned}$$

求解 CCCP 问题 III

其中 $\eta = [\eta_1, \cdots, \eta_p] \in R^p$, 且有 $\nu = \lambda\eta/L \in R_+^p$, $\mathcal{S}_\alpha : \mathbb{R}^p \to \mathbb{R}^p$ 软阈值算子

$$\mathcal{S}_\nu(\beta) = \bar{\beta}, \quad \bar{\beta}_i = (|\beta_i| - \nu_i)_+ \text{sign}(\beta_i), \; i = 1, \ldots, p.$$

- ISTA 求解 (7):

$$\beta^k = \Theta_L(\beta^{k-1}).$$

- 提升 ISTA 速度的变形 $\to$ FISTA

求解 ρ

- $$\rho^{k+1} \leftarrow \min_{\rho\in[0,1]} L(\beta^k, \rho); \tag{9}$$
  最小化一个单变量函数在区间 [0,1] 上 $\longrightarrow$ 经典的黄金分割搜索算法
- 比较模型相对应到平方损失:
  $$\min_{\rho\in(0,1)} L(\rho, \beta^k) := \frac{1}{n}\|y - X\beta^k - \rho Wy\|_2^2 + \lambda \sum_{i=1}^{p} P(|\beta_i|). \tag{10}$$
  最优解:
  $$\rho^* = \mathrm{Proj}_{[0,1]} \frac{\langle y - X\beta^k, Wy\rangle}{\|Wy\|_2^2} \tag{11}$$
  其中 $\mathrm{Proj}_{[0,1]}(t) = \begin{cases} t, & t\in[0,1], \\ 1, & t>1, \\ 0, & t<0. \end{cases}$

实验

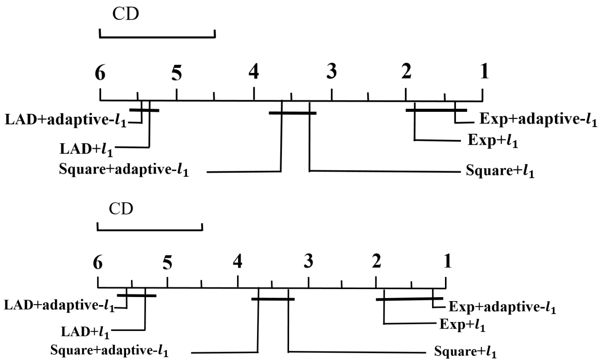


图: Nemenyi test with regularizers with outliers in y, 置信水平: $\alpha = 0.05$. Top: for "Correct" , Bottom: for "MedSE"

Amir Beck and Marc Teboulle.
A fast iterative shrinkage-thresholding algorithm for linear inverse problems.
SIAM Journal on Imaging Sciences, 2(1):183–202, 2009.