UNIVERSITY OF Southampton

Bilevel Optimisation with Applications into Hyper-parameter Tuning in Machine Learning

Shenglong Zhou work with **Alain Zemkoho**

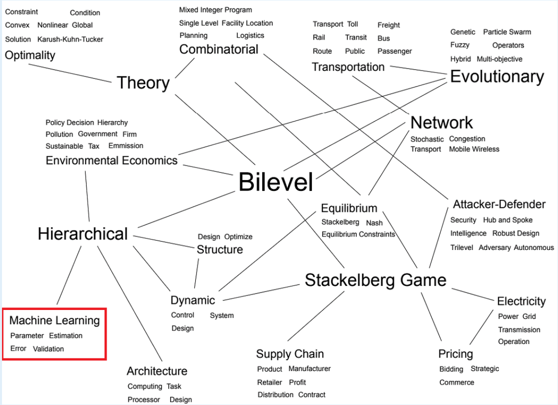
Mathematical Sciences, University of Southampton, United Kingdom

Turing Optimization Club, 11th Nov 2019

Outline

Bilevel Optimisation

Bilevel Optimisation



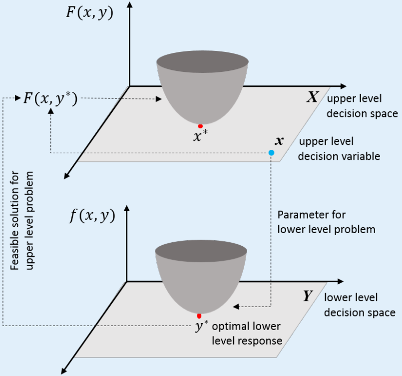
Bilevel network-map shows connections between various applications and theory since 1950s. Each connecting link represents either a topic connected with a subtopic, or an overlap between two subtopics.

Bilevel Optimisation

Bilevel optimisation (BO) problems are hierarchical. In many organizations, the realized outcome of any decision taken by the upper level leader to optimise their goals is affected by the response of lower level follower, who will seek to optimise their own outcomes. Mathematically,

$$\begin{aligned} \min_{x,y} \quad & F(x,y) \\ \text{s.t.} \quad & x \in X,\ y \in \arg\min_{z \in Y} f(x,z), \end{aligned}$$

where $F, f : X \times Y \to \mathbb{R}$.



Bilevel Optimisation

Particularly, we are interested in the bilevel optimisation model ,

$$\begin{aligned} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \leq 0,\ y \in \arg\min_{z} \ \{ f(x,z) :\ g(x,z) \leq 0 \}, \end{aligned}$$

where $F, f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $G : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$, and $g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q$. The equality constraints $H(x,y) = 0$ can be transferred into $H(x,y) \leq 0, -H(x,y) \leq 0$.

Hyper-parameter Tuning

Hyper-parameter Tuning
Regularized Regression

Regression problems frequently involve some parameters such as a tube parameter ϵ in the regression error or a trade-off parameter θ between the regression error and the regularisation:

$$\min_{\beta \in \mathbb{R}^n} \quad \underbrace{\sum_{i=1}^{m} \ell_\epsilon(h(\beta, a^i), b_i)}_{regression\ error} + \theta \underbrace{\phi(x)}_{regularisation},$$

where $(a^i, b_i) \in \mathbb{R}^n \times \mathbb{R}, i = 1, \cdots, m$ are sample data, ℓ is the loss function to quantify the regression error and ϕ is the regularisation favouring some structures of a solution.

Hyper-parameter Tuning
Applications from Machine Learning

Lasso	$\min_{\beta \in \mathbb{R}^n} \sum_{i=1}^{m} (\beta^\top a^i - b_i)^2 + \theta \|\beta\|_1,$
Ridge Lasso	$\min_{\beta \in \mathbb{R}^n} \sum_{i=1}^{m} (\beta^\top a^i - b_i)^2 + \theta_1 \|\beta\|_2^2, \text{ s.t. } \|\beta\|_1 \leq \theta_2,$
Logistic Regression	$\min_{\beta \in \mathbb{R}^n} \sum_{i=1}^{m} \left\{ \ln(1 + e^{\beta^\top a^i}) - b_i \beta^\top a^i \right\} + \theta \|\beta\|_2^2,$
Support Vector Regression	$\min_{\beta \in \mathbb{R}^n} \sum_{i=1}^{m} \max\{\lvert\beta^\top a^i - b_i\rvert - \epsilon, 0\} + \theta \|x\|_2^2,$
Support Vector Machine	$\min_{\beta \in \mathbb{R}^n, y \in \mathbb{R}} \sum_{i=1}^{m} \max\{1 - b_i(\beta^\top a^i - y), 0\} + \theta \|\beta\|_2^2.$

Hyper-parameter Tuning
K-fold Cross Validation

- Partition the set $T := \{1, \cdots, m\}$ into K parts $T_k, k = 1, \cdots, K$ with $T_k \cap T_j = \emptyset, k \neq j$ and $T = \cup_{k=1}^K T_k$.
- Partition the data $\mathcal{D} := \{(a^i, b_i)\}_{i=1}^m$ into K folders $\mathcal{D}_k := \{(a^i, b_i)\}_{i \in T_k}$.
- For a given parameter (e.g., $\mu := (\theta, \epsilon)$), for each k, solve the model to get $\beta^k(\mu)$ with training data $\mathcal{D} \setminus \mathcal{D}_k$, and then calculate the validation error on validation data $\mathcal{D}_k$, namely, $\sum_{i \in T_k} \ell_\epsilon(h(\beta^k(\mu), a^i), b_i)$. This gives us the average validation error

$$CV(\mu) := \frac{1}{K} \sum_{k=1}^{K} \sum_{i \in T_k} \ell_\epsilon(h(\beta^k(\mu), a^i), b_i).$$

- Do this for many values of μ and choose one making $CV(\mu)$ smallest.

Hyper-parameter Tuning

Bilevel Optimisation perspective

Based on the K-fold Cross Validation, for Regularized Regression, we could do

$$\begin{aligned}
\min_{\theta,\epsilon,\beta^1,\cdots,\beta^K} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{i\in T_k}\ell_\epsilon(h(\beta^k,a^i),b_i), \\
\text{s.t.} \quad & \text{for each } k=1,\cdots,K,\ \beta^k \in \operatorname*{argmin}_{z^k}\sum_{i\notin T_k}\ell_\epsilon(h(z^k,a^i),b_i)+\theta\phi(z^k),
\end{aligned}$$

Or we could solve

$$\begin{aligned}
\min_{\theta,\epsilon,\beta^1,\cdots,\beta^K} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{i\in T_k}\ell_\epsilon(h(\beta^k,a^i),b_i), \\
\text{s.t.} \quad & (\beta^1,\cdots,\beta^K) \in \operatorname*{argmin}_{(z^1,\cdots,z^K)}\sum_{k=1}^{K}\sum_{i\notin T_k}\left[\ell_\epsilon(h(z^k,a^i),b_i)+\theta\phi(z^k)\right],
\end{aligned}$$

Hyper-parameter Tuning
Advantage of Bilevel Optimisation

- Cross Validation becomes very expensive when the parameter is in high dimension. To choose the a parameter such the validation error $CV(\mu)$ smallest, the common way is to use the grid-searching. For example, if $\mu = (\mu_1, \mu_2, \mu_3)^\top \in \mathbb{R}^3$. Each μ_i has N values, then to choose a best μ, it needs repeat N^3 times to solve the model.
- CV does not give the value range of the parameter, which means we need to test many values to find a proper range and then do CV. While this can be overcome by Bilevel optimisation which calculates the parameter automatically.

Approaches to Solve Bilevel Optimisation

LLVF Reformulation Approach
Model Reformulation

Bilevel optimisation model

$$\begin{aligned} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \leq 0,\ y \in \arg\min_{z} \ \{ f(x,z) : \ g(x,z) \leq 0 \}. \end{aligned}$$

We reformulate the above two level problem into a single level model

$$\begin{aligned} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \leq 0,\ g(x,y) \leq 0,\ f(x,y) = \psi(x) \end{aligned}$$

by introduce the lower-level value function (LLVF),

$$\psi(x) := \min_{z\in\mathbb{R}^m} \ \{ f(x,z) : \ g(x,z) \leq 0 \},$$

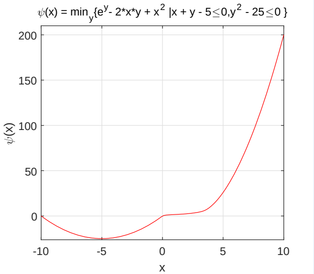
LLVF Reformulation Approach
Partial penalization

The single level model

$$\min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad F(x,y) \tag{1}$$
$$\text{s.t.} \quad G(x,y) \le 0,\ g(x,y) \le 0,\ f(x,y) = \psi(x).$$

To establish its necessary optimality condition, consider its the partial penalization, for $\lambda > 0$,

$$\min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad F(x,y) + \lambda(f(x,y) - \psi(x))$$
$$\text{s.t.} \quad G(x,y) \le 0,\ g(x,y) \le 0.$$



The equivalence is related to partial calmness.[1]

[1] Ye, J.J. and Zhu, D.L., 1995. Optimality conditions for bilevel programming problems. Optimization, 33(1), pp.9-27.

LLVF Reformulation Approach

Necessary optimality condition

Let (x, y) be a local optimal solution of (1). Under some assumptions, there exist $\lambda > 0$, $u \in \mathbb{R}^p$, $(v, w) \in \mathbb{R}^{2q}$, and $z \in \mathbb{R}^m$ such that we have

$$\begin{aligned}
\nabla_x F(x,y) + \nabla_x G(x,y)^\top u + \nabla_x g(x,y)^\top v + \lambda \nabla_x f(x,y) - \lambda \nabla_x \ell(x,z,w) &= 0,\\
\nabla_y F(x,y) + \nabla_y G(x,y)^\top u + \nabla_y g(x,y)^\top v + \lambda \nabla_y f(x,y) &= 0,\\
\nabla_z f(x,z) + \nabla_z g(x,z)^\top w &= 0,\\
u \geq 0,\ G(x,y) \leq 0,\ u^\top G(x,y) &= 0,\\
v \geq 0,\ g(x,y) \leq 0,\ v^\top g(x,y) &= 0,\\
w \geq 0,\ g(x,z) \leq 0,\ w^\top g(x,z) &= 0.
\end{aligned}$$

where $\ell(x, z, w)$ represents the lower-level Lagrangian function [2]

$$\ell(x,z,w) := f(x,z) + w^\top g(x,z)$$

[2] Gauvin, J. and Dubeau, F. (1982). Differential properties of the marginal function in mathematical programming, Mathematical Programming, pp. 101-119.

QVI Reformulation Approach
Model Reformulation

For the lower level problem

$$y \in \arg\min_{z\in\mathbb{R}^m} \; \{ f(x,z) : \; g(x,z) \leq 0 \},$$

if it is convex w.r.t. the second variable, then we could consider

$$\langle \nabla_y f(x,y), z - y \rangle \geq 0, \forall\, z \text{ such that } g(x,z) \leq 0,$$

These quasi-variational inequalities (QVI) are equivalent to

$$y^\top \nabla_y f(x,y) = \min_{z\in\mathbb{R}^m} \left\{ z^\top \nabla_y f(x,y) : \; g(x,z) \leq 0 \right\} =: \varphi(x,y),$$

which allows us to derive a single level reformulation as

$$\begin{aligned} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \leq 0, \; g(x,y) \leq 0, \; y^\top \nabla_y f(x,y) = \varphi(x,y). \end{aligned}$$

QVI Reformulation Approach
Partial penalization

The single level model

$$\begin{aligned} \min_{\zeta \in \mathbb{R}^{n+m}} \quad & F(\zeta) \\ \text{s.t.} \quad & G(\zeta) \le 0,\ g(\zeta) \le 0,\ h(\zeta) = \varphi(\zeta). \end{aligned} \tag{2}$$

Again we consider its the partial penalization

$$\begin{aligned} \min_{\zeta \in \mathbb{R}^{n+m}} \quad & F(\zeta) + \lambda(h(\zeta) - \varphi(\zeta)) \\ \text{s.t.} \quad & G(\zeta) \le 0,\ g(\zeta) \le 0, \end{aligned}$$

where $\zeta = (x; y)$ and $h(\zeta) = y^\top \nabla_y f(x, y)$.

QVI Reformulation Approach

Necessary optimality condition

Let ζ be a local optimal solution of (2). Under some assumptions, there exist $\lambda > 0$, $u \in \mathbb{R}^p$, $(v, w) \in \mathbb{R}^{2q}$, and $z \in \mathbb{R}^m$ such that we have

$$\begin{aligned}
\nabla F(\zeta) + \nabla G(\zeta)^\top u + \nabla g(\zeta)^\top v + \lambda \nabla h(\zeta) - \lambda \nabla_\zeta \ell(\zeta, z, w) = 0, \\
\nabla_{\zeta_2} f(\zeta) + \nabla_z g(\zeta_1, z)^\top w = 0, \\
u \geq 0,\ G(\zeta) \leq 0,\ u^\top G(\zeta) = 0, \\
v \geq 0,\ g(\zeta) \leq 0,\ v^\top g(\zeta) = 0, \\
w \geq 0,\ g(\zeta_1, z) \leq 0,\ w^\top g(\zeta_1, z) = 0.
\end{aligned}$$

where $\ell(\zeta, z, w)$ represents the lower-level Lagrangian function

$$\ell(\zeta, z, w) := z^\top \nabla_{\zeta_2} f(\zeta) + w^\top g(\zeta_1, z).$$

KKT Reformulation Approach
Model Reformulation

For the lower level problem

$$y \in \arg\min_{z \in \mathbb{R}^m} \{ f(x,z) : g(x,z) \leq 0 \},$$

replacing it by its Karush-Kuhn-Tucker (KKT) conditions derive a single level reformulation as

$$\begin{aligned} \min_{t \in \mathbb{R}^q, x \in \mathbb{R}^n, y \in \mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \leq 0, \\ & \nabla_y f(x,y) - \nabla_y g(x,y)^\top t = 0, \\ & g(x,y)^\top t = 0, \;\; g(x,y) \leq 0, \;\; t \leq 0. \end{aligned}$$

KKT Reformulation Approach
Partial penalization

The single level model

$$\begin{aligned} \min_{t\in\mathbb{R}^q, x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) \\ \text{s.t.} \quad & G(x,y) \le 0, \\ & \nabla_y f(x,y) - \nabla_y g(x,y)^\top t = 0, \\ & g(x,y)^\top t = 0, \;\; g(x,y) \le 0, \;\; t \le 0. \end{aligned} \tag{3}$$

Standard constraint qualifications fail to hold for (3) due to the complementary equations $g(x,y)^\top t = 0$. Consider its the partial penalization

$$\begin{aligned} \min_{t\in\mathbb{R}^q, x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & F(x,y) + \lambda g(x,y)^\top t \\ \text{s.t.} \quad & G(x,y) \le 0, g(x,y) \le 0, \;\; t \le 0, \\ & \nabla_y f(x,y) - \nabla_y g(x,y)^\top t = 0. \end{aligned}$$

KKT Reformulation Approach

Necessary optimality condition

Let (x, y, t) be a local optimal solution of (3). Under some assumptions, there exist $\lambda > 0$, $u \in \mathbb{R}^p$, $s \in \mathbb{R}^m$ and $(v, w) \in \mathbb{R}^{2q}$ such that we have

$$
\begin{aligned}
\nabla_x F(x,y) + \nabla_x G(x,y)^\top u + \nabla_x g(x,y)^\top (\lambda t + v) + \nabla_x h(x,y,t)^\top s = 0,\\
\nabla_y F(x,y) + \nabla_y G(x,y)^\top u + \nabla_y g(x,y)^\top (\lambda t + v) + \nabla_y h(x,y,t)^\top s = 0,\\
\lambda g(x,y) + w - \nabla_y g(x,y) s = 0,\\
h(x,y,t) = 0,\\
u \geq 0,\ G(x,y) \leq 0,\ u^\top G(x,y) = 0,\\
v \geq 0,\ g(x,y) \leq 0,\ v^\top g(x,y) = 0,\\
w \geq 0,\ t \leq 0,\ w^\top t = 0.
\end{aligned}
$$

where

$$h(x,y,t) := \nabla_y f(x,y) - \nabla_y g(x,y)^\top t.$$

Numerical Implementation

Semi-smooth Newton method

Fischer-Burmeister function

Three optimality conditions have nonlinear complementarity conditions, which are able to be rewritten as systems only containing equations through some NCP functions, (e.g. Fischer-Burmeister function[3]) defined by

$$\psi_{FB}(a,b) := \sqrt{a^2+b^2} - a - b. \tag{4}$$

For instance,

$$\begin{aligned} & u \geq 0,\ G(x,y) \leq 0,\ u^\top G(x,y) = 0 \\ \iff \quad & \psi_{FB}(-G(x,y),u) := \begin{bmatrix} \psi_{FB}(-G_1(x,y),u_1) \\ \vdots \\ \psi_{FB}(-G_p(x,y),u_p) \end{bmatrix} = 0. \end{aligned}$$

[3] Fischer, A. (1992). A special Newton-type optimization method. Optimization, 24(3-4), 269-284.

Semismooth Newton Method
Algorithmic framework

Semismooth Newton method[4] solves non-smooth equations $\Phi^\lambda(\chi) = 0$, with minimizing $\Psi^\lambda(\chi) := \frac{1}{2}\|\Phi^\lambda(\chi)\|^2$.

Step 0: Choose $\lambda, \epsilon, K > 0, \rho \in (0,1), \sigma \in (0, 1/2), \delta > 2$, χ^o and set $k := 0$.
Step 1: If $\|\Phi^\lambda(\chi^k)\| < \epsilon$ or $k \leq K$, then stop.
Step 2: Choose $W^k \in \partial_B \Phi^\lambda(\chi^k)$ and find the solution d^k of the system

$$W^k d^k = -\Phi^\lambda(\chi^k).$$

If the above system is not solvable of if the condition

$$\nabla \Psi^\lambda(\chi^k)^\top d^k \leq -\rho\|d^k\|^\delta$$

is not satisfied, set $d^k = -\nabla\Psi^\lambda(\chi^k)$.

Step 3: Find the smallest nonnegative integer s_k such that

$$\Psi^\lambda(\chi^k + \rho^{s_k} d^k) \leq \Psi^\lambda(\chi^k) + 2\sigma\rho^{s_k}\nabla\Psi^\lambda(\chi^k)^\top d^k.$$

Then set $\alpha_k := \rho^{s_k}$, $\chi^{k+1} := \chi^k + \alpha_k d^k$, $k := k+1$ and go to **Step 1**.

[4] De Luca, T., Facchinei, F. and Kanzow, C. (1996). A semismooth equation approach to the solution of nonlinear complementarity problems. Mathematical programming, 75(3), 407-439.

Bilevel Optimisation Toolbox

BiOpt toolbox[5], to help accelerate the development of numerical toolboxes for bilevel optimisation, aims at providing a platform on which users can test a wide range of bilevel optimization problems by using the provided solvers. The toolbox is made of

- Three bilevel optimisation solvers: SNLLVF, SNQVI and SNKKT based on three optimality conditions via Semismooth Newton method.
- BOLIB: Bilevel optimisation library of test problems containing 173 bilevel optimisation test examples.
- Derivatives calculator to calculate the first, second and third order derivatives of a single/set-valued function.
- Optimal-value function tools to operate an optimal-value function.

[5] Available at https://biopt.github.io/

Figure: Three solvers need calculate derivatives.

	F	G	f	g
`SNLLVF`	1st,2nd	1st,2nd	1st,2nd	1st,2nd
`SNQVI`	1st,2nd	1st,2nd	1st,2nd,3rd	1st,2nd
`SNKKT`	1st,2nd	1st,2nd	1st,2nd,3rd	1st,2nd,3rd

Numerical Experiments
Comparison of three solvers

Figure: Three solvers solve 124 bilevel optimisation examples.

λ		2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}
Number	SNLLVF	6	2	8	3	3	1	6
of	SNQVI	8	6	6	3	8	5	9
Failures	SNKKT	8	8	6	6	9	12	12
Average	SNLLVF	154.0	113.4	181.7	85.2	144.3	154.4	198.6
iterations	SNQVI	194.7	145.5	180.3	102.6	238.7	215.1	284.9
	SNKKT	166.1	173.7	157.9	170.9	212.4	233.2	308.9
Average	SNLLVF	0.17	0.10	0.16	0.07	0.15	0.14	0.21
time	SNQVI	0.55	0.39	0.44	0.35	1.98	1.94	1.62
	SNKKT	5.11	5.55	3.99	1.23	4.27	5.68	5.75

Numerical Experiments
Comparison of three solvers

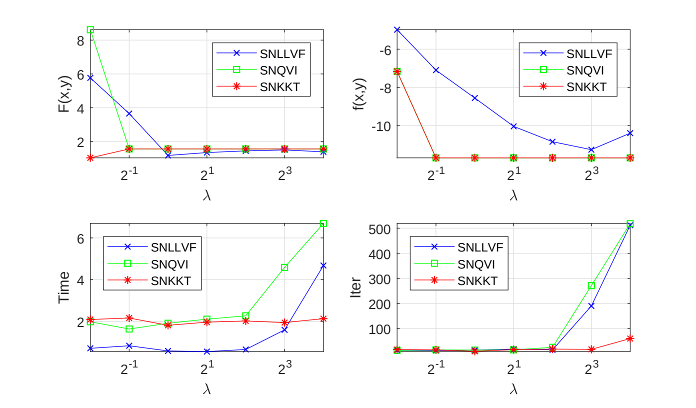


Figure: A simple example with $n = 1, m = 1, p = 1, q = 4$.

Numerical Experiments
Comparison of three solvers

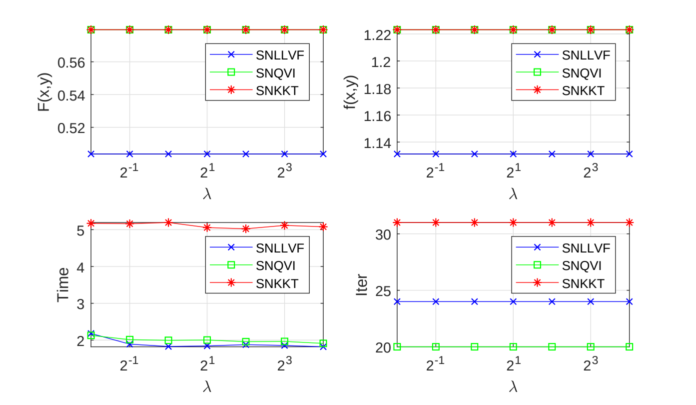


Figure: Optimal Control problem with $n = 2, m = 274, p = 3, q = 548$.

Hyper-parameter Tuning
Ridge Lasso

For Ridge Lasso problem:

$$\min_{\beta \in \mathbb{R}^n} \|A\beta - b\|^2 + \theta_1 \|\beta\|^2, \text{s.t. } \|\beta\|_1 \leq \theta_2.$$

Partition $T = \cup_{k=1}^{K} T_k$ with $T_k \cap T_j = \emptyset, k \neq j$ and let $A^k := A_{T_k}, \bar{A}^k := A_{T \setminus T_k}$. and $b^k := b_{T_k}, \bar{b}^k := b_{T \setminus T_k}$

$$\begin{aligned} \min_{\theta := (\theta_1, \theta_2), \mathbf{y} := (\beta^1, \cdots, \beta^K)} \quad & \frac{1}{K} \sum_{k=1}^{K} \|A^k \beta^k - b^k\|^2 + \|\theta\|^2, \\ \text{s.t.} \quad & \mathbf{y} \in \operatorname*{argmin}_{\|z^i\|_1 \leq \theta_2, i=1,\cdots,K} \sum_{k=1}^{K} \|\bar{A}^k z^k - \bar{b}^k\|^2 + \theta_1^2 \|z^k\|^2. \end{aligned}$$

Hyper-parameter Tuning
Ridge Lasso

Do $K = 5$-fold Cross Validation with $\theta_1 \in \{10^{-8}, 10^{-7.5}, \cdots, 10^{-1}\}$, $\theta_2 \in \{10^{-3}, 10^{-2.5}, \cdots, 10^{3}\}$. $CV(\theta)$ generated by Cross Validation and `SNQVI`.

$m = n$		$CV(\theta)$	θ_1^2	θ_2	Time(sec)
50	`CV`	0.0912	0.00	1.00	3.71
	`SNQVI`	0.0664	1.79	6.78	0.75
90	`CV`	0.2432	0.00	0.10	5.41
	`SNQVI`	0.1966	4.36	9.00	1.58
120	`CV`	0.4879	0.00	0.10	7.05
	`SNQVI`	0.3899	6.89	10.97	2.34
160	`CV`	0.3876	0.00	1.00	9.23
	`SNQVI`	0.3258	6.84	10.76	5.96

Hyper-parameter Tuning
Support Vector Regression

For SVR problem[6]

$$\min_{\beta\in\mathbb{R}^n}\sum_{i=1}^{m}\max\{|\beta^\top a^i-b_i|-\epsilon,0\}+\theta\|\beta\|_2^2,\ \ \text{s.t.}\ \ \underline{\beta}\le\beta\le\overline{\beta}.$$

Let $T=\{1,\cdots,m\}=\cup_{k=1}^K T_k$. The Bilevel optimisation model

$$\begin{aligned}
\min_{\substack{x:=(\underline{\beta},\overline{\beta},\epsilon,\theta)\\ y:=(\beta^1,\cdots,\beta^K)}} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{i\in T_k}|(\beta^k)^\top a^i-b_i| \\
\text{s.t.} \quad & \epsilon>0,\theta>0,\underline{x}\le\overline{\beta} \\
& y\in\underset{\underline{\beta}\le z^1,\cdots,z^K\le\bar{x}}{\operatorname{argmin}}\sum_{k=1}^{K}\sum_{i\in T\setminus T_k}\max\{|(z^k)^\top a^i-b_i|-\epsilon,0\}+\theta\|z^k\|_2^2,
\end{aligned}$$

[6] Bennett, K. P., Hu, J., Ji, X., Kunapuli, G. and Pang, J. S. (2006, July). Model selection via bilevel optimization. In The 2006 IEEE International Joint Conference on Neural Network Proceedings (pp. 1922-1929). IEEE.

Conclusion

- Bilevel optimisation is able to do better hyper-parameter tuning than cross validation idea.
- Three approaches are proposed to deal with bilevel optimisation. All of are then addressed by Semismooth Newton method which is limited to solve problems with large size.
- K-fold cross validation increases the problem size of the bilevel optimisation, how to design more efficient method ?

References

[1] A. Fischer, A.B. Zemkoho and **S.L. Zhou**, Semismooth Newton-type method for bilevel optimization: Global convergence and extensive numerical experiments, arXiv:1912.07079, 2019.

[2] **S.L. Zhou**, A.B. Zemkoho and S. Demphe, Value Function Approach to Qusai-Variational Inequalities with Applications to Bilevel Optimisation, Technique Report, 2019.

[3] A. Zemkoho and **S.L. Zhou**, Theoretical and numerical comparison of the Karush-Kuhn-Tucker and value function reformulations in bilevel optimization, arXiv:2004.10830, 2020.

[4] **S.L. Zhou** A.B. Zemkoho and A. Tin, BOLIB 2019 Test Examples Library Version 2, `https://biopt.github.io/`, 2019.

References

[5] K.P. Bennett, J. Hu, X. Ji, G. Kunapuli and J.S. Pang, Model selection via bilevel optimization, IJCNN, IEEE, 2006.

[6] G. Kunapuli, K.P. BeGnnett, J. Hu and J.S. Pang, Classification model selection via bilevel programming, Optimization Methods & Software 23.4: 475-489, 2008.

[7] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi and M. Pontil, Bilevel programming for hyperparameter optimization and meta-learning, PMLR, 80:1568-1577, 2018.

Thank you !