Parametric and Non-parametric Piecewise Linear Models and their Optimization

Xiaolin Huang

Department of Automation
School of Electronics, Information, and Electrical Engineering
Shanghai Jiao Tong Unviersity

SEIEE 2-517, Dongchuan Road 800, Shanghai, P.R. China,
xiaolinhuang@sjtu.edu.cn

2018-6-25

Outline

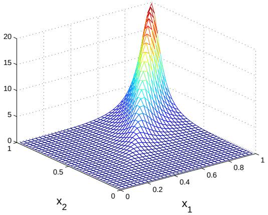
Outline

Continuous Piecewise Linear Function

- definition: $f(\mathbf{x}) : D \in \mathbb{R}^d \mapsto \mathbb{R}$ is a piecewise linear function, if there exist a finite number of affine/line functions $p_i(\mathbf{x})$:

$$f(\mathbf{x}) \in \{p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_M(\mathbf{x})\}.$$

moreover, if $f(\mathbf{x})$ is continuous, it is called continuous piecewise linear function.

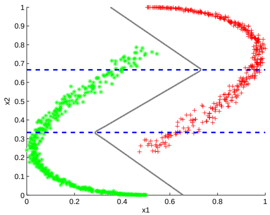


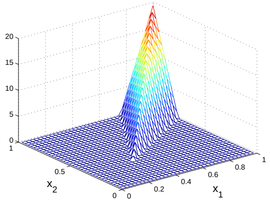
Continuous Piecewise Linear Function

- definition: $f(\mathbf{x}) : D \in \mathbb{R}^d \mapsto \mathbb{R}$ is a piecewise linear function, if there exist a finite number of affine/line functions $p_i(\mathbf{x})$:

$$f(\mathbf{x}) \in \{p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_M(\mathbf{x})\}.$$

moreover, if $f(\mathbf{x})$ is continuous, it is called continuous piecewise linear function.





Representations

- piecewise representation

$$f(\mathbf{x}) = p_i(\mathbf{x}), \forall \mathbf{x} \in \Omega_i.$$

with continuity condition

$$p_i(\mathbf{x}) = p_j(\mathbf{x}), \forall \mathbf{x} \in \Omega_i \bigcap \Omega_j, \forall i, j$$

- piecewise representations by boolean variables
- vertex representation

$$x(j) = \sum_{k=0}^{K} d_j^k \lambda_j^k, \forall j \quad f(\mathbf{x}) = \sum_{j=1}^{d} \sum_{k=0}^{K} f^j(d_j^k)\lambda_j^k,$$

where, d_j^k are breakpoints, satisfying: $\sum_{k=1}^{K} \lambda_j^k = 1$, $\lambda_j^k \geq 0$ and $\{\lambda_j^k\}$ is SOS2: special ordered set of type 2.

Compact Representations

- to represent a CPWL function as a sum/composition of basic PWL functions;

$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m B_m(\mathbf{x}).$$

  - composition of (finite) CPWL functions are still CPWL;
  - sum of (finite) CPWL functions are still CPWL.
- properties
  - capability to represent all CPWL functions;
  - capability to approach any continuous function;
  - continuity is naturally guaranteed;
  - machine learning is applicable;
  - optimized as a regular non-smooth function.

Outline

Canonical Representation and Hinging Hyperplanes

- canonical CPWL representation[1]

$$f(x) = \mathbf{a}_0^\top \mathbf{x} + b_0 + \sum_{m=1}^{M} w_m |\mathbf{a}_m^\top \mathbf{x} + b_m|.$$

- hinging hyperplanes[2]

$$f(x) = \mathbf{a}_0^\top \mathbf{x} + b_0 + \sum_{m=1}^{M} w_m \max\{0, \mathbf{a}_m^\top \mathbf{x} + b_m\}.$$

[1] Chua, Kang, Section-wise piecewise-linear functions: Canonical representation, properties, and applications, Proc. of IEEE, 1977.

[2] Breiman, Hinging hyperplanes for regression, classification and function approximation, IEEE-TIT, 1993.

Learning and Optimization

- learn **a** and **b** from samples $\{\mathbf{x}_i, y_i\}_{i=1}^N$:

$$\min_{\mathbf{a},\mathbf{b},\mathbf{w}} \sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}_0^\top \mathbf{x} + b_0 + \sum_{m=1}^{M} w_m \max\{0, \mathbf{a}_m^\top \mathbf{x}_i + b_m\} \right) \right)^2$$

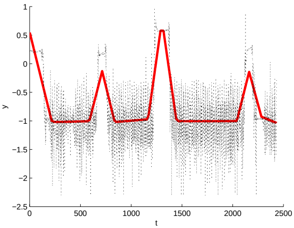
  - the function $f(\mathbf{x})$ is PWL w.r.t. $\mathbf{x}$ and parameters **a**, **b**;
  - the problem is piecewise quadratic to **a**, **b**, for squared error;
  - the problem is piecewise linear to **a**, **b**, for absolute error.
- Hinge Finding Algorithm[3]
  - for the m-th hinging hyperplane, select active set $\mathcal{I}_m = \{i : \mathbf{a}_m^\top \mathbf{x}_i + b_m > 0\}$;
  - least squares on $\mathbf{x}_i, i \in \mathcal{I}_m$ to update $\mathbf{a}_m$ and b_m.

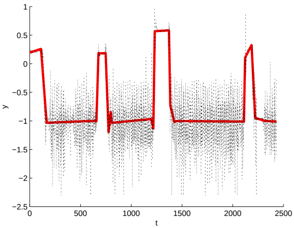
[3] Ernst, Hinging hyperplane trees for approximation and identification, IEEE-CDC, 1998.

Applications on Time-series Segmentation

- number of subregions are controlled by number of basis function[4]

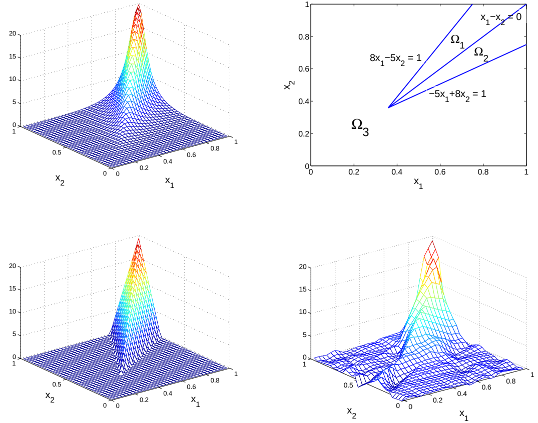
$$\begin{aligned} \min_{\mathbf{w},e} \quad & \frac{1}{2}\sum_{m=1}^{M} w_m^2 + \gamma\frac{1}{2}\sum_{i=1}^{N} e_i^2 + \sum_{m=1}^{M} \mu_m |w_m| \\ \text{s.t.} \quad & y(t_i) = e_i + w_0 + \sum_{m=1}^{M} w_m \phi_m(t_i), i = 1, 2, \ldots, N, \end{aligned}$$





[4] Huang, Matijás, Suykens, Hinging hyperplanes for time-series segmentation, IEEE-TNNLS, 2013

Limitation of Hinging Hyperplanes



Towards Full Representation Capability

- high level CPWL representation[5]
- generalized hinging hyperplane[6]

$$B_m(\mathbf{x}) = \max\left\{\mathbf{a}_{m0}^T\mathbf{x} + b_{m0}, \mathbf{a}_{m1}^T\mathbf{x} + b_{m1}, \ldots, \mathbf{a}_{md}^T\mathbf{x} + b_{md}\right\}$$

  - in deep neural networks, max pooling and maxout[7] share the theoretical discussion of GHH.
- adaptive hinging hyperplanes[8]
- irredundant lattice representation[9]
- smoothing hinging hyperplanes[10]

[5] Julián, Desages, Agamennoni, High-level canonical piecewise linear representation using a simplicial partition, IEEE-CS, 1999.

[6] Wang, Sun, Generalization of hinging hyperplanes, IEEE-TIT, 2005.

[7] Goodfellow, Warde-Farley, Mirza, Courville, Bengio, Maxout networks, 2013

[8] Xu, Huang, Wang, Adaptive hinging hyperplanes and its applications in dynamic system identification, Automatica, 2009

[9] Xu, van den Boom, De Schutter, Wang, Irredundant lattice representations of continuous piecewise affine functions, Automatica, 2016

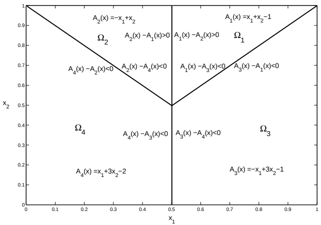
[10] Wang, Huang, Yeung, A neural network of smooth hinge functions, IEEE-TNN, 2010

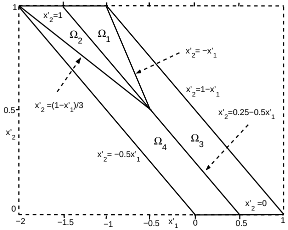
Compact Representation for Subregion

- linear subregion, Ω_i, where a CPWL function is linear:
  - Ω_i is a polyhedron;
  - Ω_i could be represented by upper/lower boundary function

$$\Omega_i = \{\mathbf{x}^{(d)}, \mid \mathcal{L}_i(\mathbf{x}^{(d-1)}) \leq x(1) \leq \mathcal{U}_i(\mathbf{x}^{(d-1)})\},$$

  - upper/lower boundaries are PWL functions in a lower space.





Domain Partition based Neural Networks

- a continuous piecewise linear function in $\mathbb{R}^d$ can be represented by the boundary functions[11],

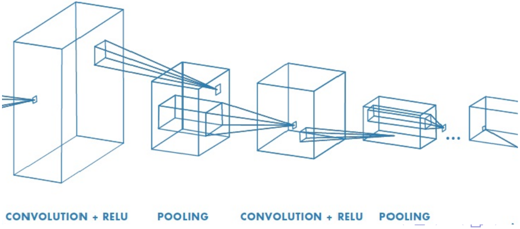
$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m \max\left\{0, \left\{\mathbf{x}(1) - \mathcal{L}_m(\mathbf{x}^{(d-1)}), \mathcal{U}_m(\mathbf{x}^{(d-1)}) - \mathcal{L}_m(\mathbf{x}^{(d-1)})\right\}\right\},$$

- recursive definition leads to deep structure;
- initialization and training by back propagation.

[11] Wang, Huang, Junaid, Configuration of continuous piecewise linear neural networks, IEEE-TNN, 2008

Deep PWL Neural Network

- linear modules in convolutional neural networks
  - convolutional operator: $\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} w_{ij}F_{ij}$
  - fully connected layer: $\sum_i\sum_j w_{ij}F_{ij}$
  - averaging pooling: $\frac{1}{K}\sum_i F_i$
- piecewise linear modules in convolutional neural networks
  - ReLu: $\max\{0, u\}$
  - LeakyReLu: $\max\{-\tau u, u\}$
  - max pooling: $\max\{F_1, F_2, \ldots, F_n\}$



Outline

Non-parametric Models

- support vector machine learns a discriminant function from training data $\{\mathbf{x}_i, y_i\}_{i=1}^N, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, +1\}$.

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|_{\ell_2}^2 + C\sum_{i=1}^N \max\left\{1 - y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right), 0\right\}.$$

- dual problem

$$\begin{aligned}\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N y_i\alpha_i\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\alpha_j y_j - \sum_{i=1}^N \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^N y_i\alpha_i = 0, \quad 0 \le \alpha_i \le C, \forall i.\end{aligned}$$

- kernel trick

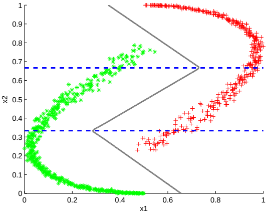
$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

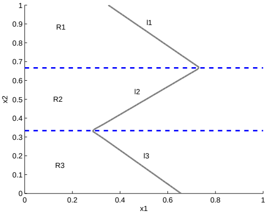
and

$$f(x) = \mathbf{w}^T\phi(\mathbf{x}_i) + b = \sum_{i=1}^N y_i\alpha_i\mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b.$$

Non-parametric Models

- sparsity and support vectors
  - only a part of samples, support vector, have $\alpha_i \neq 0$;
  - sparsity is beneficial for storage and computation;
  - if $\mathcal{K}(\mathbf{x}_i, \mathbf{x})$ is piecewise linear, then only $\alpha_i \neq 0$ provides non-convexity.





Piecewise Linear Kernels

- multiconlitron[12]: a separable model;
- intersection kernel[13] [14] $\mathcal{K}(\mathbf{u},\mathbf{v}) = \sum_{i=1}^{d} \min\{\mathbf{u}(i), \mathbf{v}(i)\}$
  - additive kernel;
  - subregion structure;
- truncated ℓ_1 kernel (TL1 kernel)[15]:

$$\mathcal{K}(\mathbf{u},\mathbf{v}) = \max\{0, \rho - \|\mathbf{u}-\mathbf{v}\|_{\ell_1}\}$$

  - non-separable functions
  - flexible subregion structure;
  - non-PSD (positive semi-definite) kernel.

[12] Li, Liu, Yang, Fu, Li, Multiconlitron:A general piecewise linear classifier, IEEE-TNN, 2011

[13] Maji, Berg, Malik, Classification using intersection kernel support vector machines is efficient, CVPR, 2008

[14] Maji, Berg, Malik, Efficient classification for additive kernel SVMs, IEEE-TPAMI, 2013

[15] Huang, Suykens, Wang, Hornegger, Maier, Classification with truncated l1 distance kernel, IEEE-TNNLS, 2018.

Indefinite Learning

- indefinite learning

$$\begin{aligned} \min_{\alpha_i} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i\alpha_i\mathcal{K}(\mathbf{x}_i,\mathbf{x}_j)\alpha_j y_j - \sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{N} y_i\alpha_i = 0, \quad 0 \le \alpha_i \le C, \forall i. \end{aligned}$$

  - there is no ϕ such that $\mathcal{K}(\mathbf{u},\mathbf{v}) = \phi(\mathbf{u})^\top\phi(\mathbf{v})$;
  - the kernel matrix $\mathcal{K} : \mathcal{K}_{ij} = \mathcal{K}(\mathbf{x}_i,\mathbf{x}_j)$ is non-PSD and the problem is non-convex;
- non-separable PWL kernels are likely to be indefinite.

Indefinite Learning

- reproducing kernel Hilbert space (RKHS) $\rightarrow$ reproducing kernel Kreĭn spaces (RKKS)[16]
    - feature space interpretation[17]
    - generalized representer theorem[18]
- convex problem $\rightarrow$ non-convex problem;
    - kernel generated model;
    - eigenvalue cutting[19]/flipping[20]/squaring[21];
    - finding the nearest PSD kernel[22], e.g., $\min_{\tilde{\mathcal{K}} \succeq 0} \|\tilde{\mathcal{K}} - \mathcal{K}\|_F$
    - non-convex optimization[23].

[16] G. Loosli, S. Canu, and C. S. Ong, Learning SVM in Kreĭn spaces, TPAMI, 2016.

[17] Y. Ying, C. Campbell, M. Girolami, Analysis of SVM with indefinite kernels, NIPS 2009

[18] C. S. Ong, X. Mary, S. Canu, A. J. Smola, Learning with non-positive kernels, ICML 2004

[19] E. Pekalska, et al., Kernel discriminant analysis for PSD/indefinite kernels, TPAMI, 2009.

[20] V. Roth, J. Laub, M. Kawanabe, J. M. Buhmann, Optimal cluster preserving embedding of nonmetric proximity data, TPAMI, 2003

[21] H. Sun et al., LS regression with indefinite kernels and coefficient regul., ACHA, 2011

[22] R. Luss, et al., SVM classification with indefinite kernels, NIPS 2008.

[23] F. Schleif, P. Tino, Indefinite proximity learning: A review, Neural Computation, 2015

LS-SVM

- primal problem:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\xi} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{M}\xi_i^2 \\ \text{s.t.} & y_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b) = 1-\xi_i,\ \forall i \in \{1,\ldots,m\} \end{array}$$

- dual problem:

$$\left[\begin{array}{cc} 0 & \mathbf{y}^T \\ \mathbf{y} & \mathbf{H}+\frac{1}{\gamma}\mathbf{I} \end{array}\right][b,\alpha_1,\ldots,\alpha_N]^T = \left[\begin{array}{c} 0 \\ \mathbf{1} \end{array}\right],$$

where $\mathbf{I}$ is an identity matrix, $\mathbf{1}$ is an all ones vector with the proper dimension, and $\mathbf{H}$ is given by

$$\mathbf{H}_{ij} = y_i y_j \mathbf{K}_{ij} = y_i y_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j).$$

Indefinite LS-SVM

- choose a non-PSD kernel $\mathbf{K}$, the dual problem of LS-SVM is still easy to solve, but it lacks of feature space interpretation.[24]

Theorem

The dual problem of

$$\begin{aligned} \min_{\mathbf{w}_+,\mathbf{w}_-,b,\xi} \quad & \frac{1}{2}(\mathbf{w}_+^T\mathbf{w}_+ - \mathbf{w}_-^T\mathbf{w}_-) + \frac{\gamma}{2}\sum_{i=1}^{N}\xi_i^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}_+^T\phi_+(\mathbf{x}_i) + \mathbf{w}_-^T\phi_-(\mathbf{x}_i) + b) = 1 - \xi_i, \quad \forall i \in \{1,2,\ldots,N\} \end{aligned}$$

is

$$\begin{bmatrix} 0 & \mathbf{y}^T \\ \mathbf{y} & \mathbf{H} + \frac{1}{\gamma}\mathbf{I} \end{bmatrix} [b, \alpha_1, \ldots, \alpha_M]^T = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}.$$

[24] Huang, Maier, Hornegger, Suykens, Indefinite kernels in least squares support vector machine and principal component analysis, Applied and Comput. Harmonic Analysis, 2017

Learning Performance of PWL Kernel

Table: Average Accuracy and Standard Deviation on Test Data

dataset	M	RBF kernel (σ by cross-validation)	TL1 kernel ($\rho = 0.7n$)
Qsar	528	$\underline{86.92} \pm 1.31\%$	$86.05 \pm 1.21\%$
Splice	1000	$89.83 \pm 0.09\%$	$\underline{92.74} \pm 0.02\%$
Guide3	1243	$84.15 \pm 3.45\%$	$\underline{97.56} \pm 0.00\%$
Madelon	2000	$58.83 \pm 0.00\%$	$\underline{61.33} \pm 0.00\%$
Spamb.	2300	$93.32 \pm 0.60\%$	$\underline{94.05} \pm 0.56\%$
ML-prove	3059	$72.48 \pm 0.32\%$	$\underline{79.08} \pm 0.00\%$
Guide1	3089	$96.84 \pm 0.16\%$	$\underline{97.12} \pm 0.04\%$
Wilt	4339	$85.80 \pm 0.74\%$	$\underline{86.80} \pm 0.44\%$
Phish.	5528	$\underline{95.92} \pm 0.30\%$	$93.83 \pm 0.48\%$
Magic	9510	$\underline{86.48} \pm 0.45\%$	$86.04 \pm 0.43\%$
RNA	59535	$\underline{96.66} \pm 0.20\%$	$95.74 \pm 0.22\%$

Indefinite kernel PCA

- primal problem:

$$\begin{aligned} \max_{\mathbf{w},\xi} \quad & \frac{\gamma}{2}\sum_{i=1}^{M}\xi_i^2 - \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{s.t.} \quad & \xi_i = \mathbf{w}^T(\phi(\mathbf{x}_i) - \hat{\mu}_\phi), \forall i \in \{1, \ldots, M\}, \end{aligned}$$

  where $\hat{\mu}_\phi$ is the centering term, i.e., $\hat{\mu}_\phi = \frac{1}{m}\sum_{i=1}^{M}\phi(\mathbf{x}_i)$.
- dual problem:

$$\Omega\alpha = \lambda\alpha,$$

  where the centered kernel matrix Ω is induced from $\mathcal{K}$:

$$\begin{aligned} \Omega_{ij} \quad = \quad & \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{M}\sum_{r=1}^{M}\mathcal{K}(x_i, x_r) \\ & -\frac{1}{M}\sum_{r=1}^{M}\mathcal{K}(x_j, x_r) + \frac{1}{M^2}\sum_{r=1}^{M}\sum_{s=1}^{M}\mathcal{K}(x_r, x_s). \end{aligned}$$

- a non-PSD kernel can also be directly used.

Indefinite kernel PCA with PWL kernel

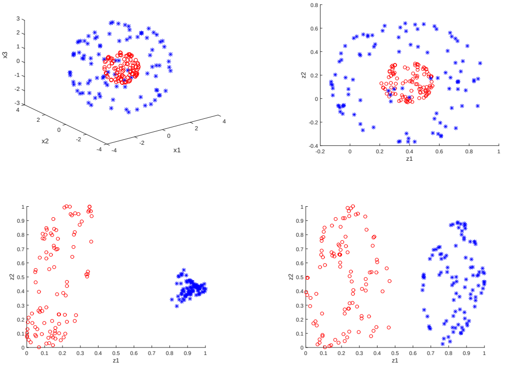
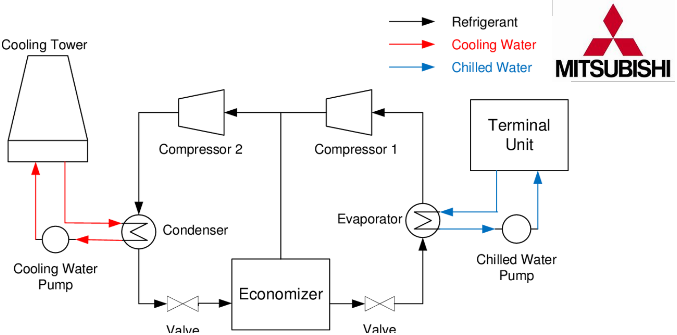


Figure: Reduce data of two classes in three dimensional space into two dimensional space

Outline

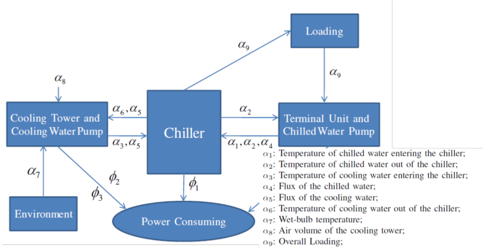
Example 1: Chiller Plants Optimization

- operation optimization for centrifugal chiller plants



Example 1: Chiller Plants Optimization

- model the input-output relationship by PWL functions;
- surrogate optimization via sub linear programmings;

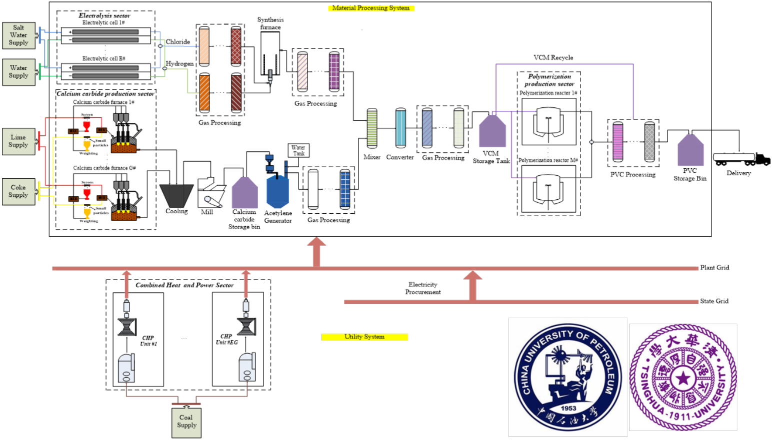


$$
\begin{aligned}
\min_{\alpha} \quad & f_0(\alpha) \\
\text{s.t.} \quad & f_i(\alpha) \leq 0 \\
& h_i(\alpha) = 0.
\end{aligned}
$$

	600 kW	1500 kW	2400 kW	3300 kW
10 °C	37.74%	30.69%	12.91%	19.85%
15 °C	14.05%	17.28%	17.68%	09.79%
20 °C	25.30%	02.11%	06.13%	16.92%
25 °C	24.51%	10.87%	10.04%	09.99%

Example 2: PVC Production Process Optimization

- PVC production process



Example 2: PVC Production Process Optimization

- equations: 13871
- optimization variables: 5064 (discrete), 8119 (continuous)
- optimization time (MILP): around 2000 s (MINLP: around 14000 s)

	Optimization Model	Current Model	improvement	
			absolute value (¥)	Relative value (%)
total cost	156,796,000	166,392,500	9,596,500	5.8↓
energy cost	68,424,300	79,406,633.5	10,982,334	13.8↓
coal cost	68,424,300	75,949,933.5	7,525,634	9.9↓
inventory cost	1,602,600	229,380	1,373,220	598.7↑
material cost	86,712,400	86,712,386.5	14	—
switching cost	56,700	44,100	12,600	28.6↑

PWL Optimization Problems

- optimization based on learned PWL models;
  - for unknown or complicated function $g(\mathbf{x})$, model a surrogate PWL function $f(\mathbf{x})$ and optimize $f(\mathbf{x})$;
  - discussion on specific PWL model, e.g., local optimality[25] and global heuristic;
- training for PWL models
  - piecewise linear penalty/loss $\rightarrow$ piecewise linear optimization
    - ℓ_1-norm regularization term, total variation, non-convex sparsity enhancer[26] [27] $p(\mathbf{u}) = \sum_{k=1}^{K} |u[k]|$
    - absolute loss, quantile loss (k-th maximum loss), hinge loss, ramp loss, ...
  - smooth penalties/loss $\rightarrow$ piecewise smooth optimization
    - ℓ_2-norm regularization term
    - squared loss, sigmoid loss, logarithmic loss, ...

[25] Huang, Xu, Wang, Exact penalty and optimality condition for nonseparable continuous piecewise linear programming, Journal of Optimization Theory and Applications, 2012

[26] Wang, Yin, Sparse signal recon. via iterative support detection, SIAM. Imag. Sci. 2010

[27] Huang, Van Huffel, Suykens, Two-level ℓ_1 minimization for CS., Signal Processing, 2015

Example 3: Ramp-LPSVM

- ℓ_1-norm penalty (sparsity) + ramp loss (robustness)[28];
- $p(\mathbf{u}) = \sum_{i=1}^{n} |u(i)|$, $l_{\text{ramp}}(u) = \max\{0, \min\{u, 1\}\}$;
- ramp loss linear programming SVM

$$\min_{\alpha} \quad \mu \sum_{i} |\alpha_i| + \frac{1}{N} \sum_{i=1}^{N} l_{\text{ramp}} \left(1 - y_i \left(\sum\nolimits_{j=1}^{N} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \right) \right)$$

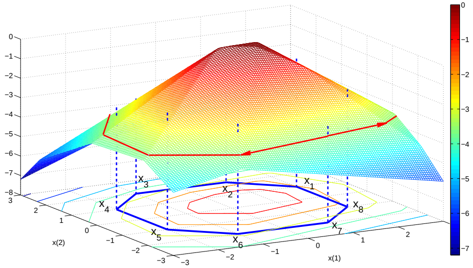
- difference of convex functions:

$$\begin{aligned} \min_{\alpha} \quad & \mu \sum_{i} |\alpha_i| + \frac{1}{N} \sum_{i=1}^{N} \max \left\{ 1 - y_i \left(\sum\nolimits_{j=1}^{N} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \right), 0 \right\} \\ & - \frac{1}{N} \sum_{i=1}^{N} \max \left\{ -y_i \left(\sum\nolimits_{j=1}^{N} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \right), 0 \right\} \end{aligned}$$

[28] Huang, Shi, Suykens, Ramp loss linear programming support vector machine, JMLR, 2014.

Example 3: Ramp-LPSVM

- solving ramp-LPSVM
  - linear programming for a local optimum;
  - hill detouring[29], i.e., search on contour lines of a concave PWL function;



[29] Huang, Xu, Mu, Wang, The hill detouring method for minimizing hinging hyperplanes functions, Computers & Operations Research, 2012.

Example 3: Ramp-LPSVM

Table: Accuracy on Test Data and Number of SV (10% outliers)

	Spect		Monk1		Monk2		Monk3		Breast	
C-SVM	81.42%	#79	76.22%	#51	72.41%	#99	80.05%	#57	89.69%	#34
ramp-LPSVM	87.88%	#34	79.33%	#51	81.57%	#70	83.43%	#39	93.35%	#24
	Pima		**Trans.**		**Haber.**		**Ionos.**			
C-SVM	61.66%	#61	70.33%	#73	70.65%	#42	85.79%	#78		
ramp-LPSVM	68.51%	#37	75.28%	#8	74.62%	#4	90.35%	#29		

- robustness to outliers is improved;
- sparsity is enhanced;
- algorithm is not applicable to large-scale problems.

Outline

- Conclusion
  - compact continuous piecewise linear models
    - parametric models and its link to neural networks: DP-CPLNN, AHH, SHH, ...
    - non-parametric models and indefinite learning: TL1 kernel, indefinite LS-SVM, and indefinite kPCA, ...
  - optimization based on compact piecewise linear models
    - surrogate optimization for chiller plants and PVC production process
    - machine learning based on piecewise linear models, e.g., ramp-LPSVM
- Outlook
  - learning behavior and interpretation
    - deep piecewise linear neural networks
    - piecewise linear indefinite kernels
  - piecewise linear optimization
    - fast local search and efficient global search
    - training piecewise linear neural networks
  - conversation among different piecewise linear models

Acknowledgement

- Prof. Shuning Wang,
  Tsinghua University, Beijing, P.R. China
- Prof. Johan A.K. Suykens,
  Katholieke Universiteit Leuven, Leuven, Belgium
- Prof. Andreas Maier,
  Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
- Prof. Jun Xu,
  Harbin Institute of Technology (Shenzhen Campus), Shenzhen, P.R. China




Thanks