# Support Vector Machines (SVM): Recent Research

**Panos M. Pardalos**
**www.ise.ufl.edu/pardalos**
**https://nnov.hse.ru/en/latna/**

**Winter School on Data Analytics (Nov 20-22, 2020, HSE)**

# Classification and Clustering in Data Analysis

- Classification **(supervised learning)** uses **predefined classes** in which objects are assigned, while clustering (**unsupervised learning) identifies similarities between objects**, which it **groups** according to those characteristics in common and which differentiate them from other groups of objects. These groups are known as "**clusters**".

# Applications of Classification Algorithms

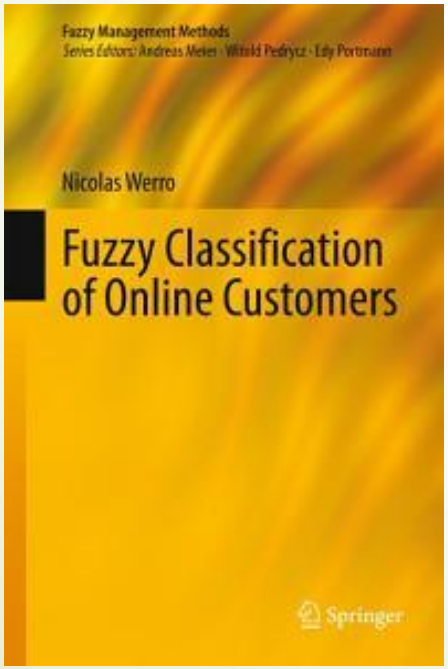
- Speech recognition
- Face recognition
- Handwriting recognition
- Biometric identification
- Document classification
- Fraud detection in finance
- Biomedicine

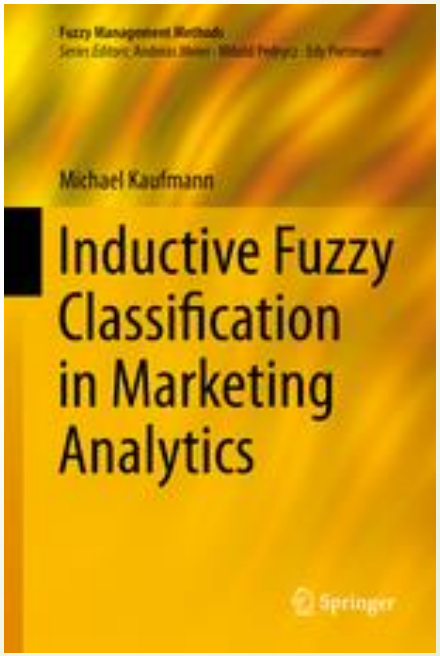
# Classification Algorithms

- Neural Networks
- Random Forest
- Decision Trees
- Nearest Neighbor
- Boosted Trees
- Linear Classifiers: Logistic Regression, Naïve Bayes Classifier
- **Support Vector Machines**

# Fuzzy approaches to classification

- Ducange, P., Fazzolari, M. & Marcelloni, F. An overview of recent distributed algorithms for learning fuzzy models in Big Data classification. J Big Data 7, 19 (2020). https://doi.org/10.1186/s40537-020-00298-6



# Quantum approaches to classification

- Is Quantum Machine Learning the next thing?

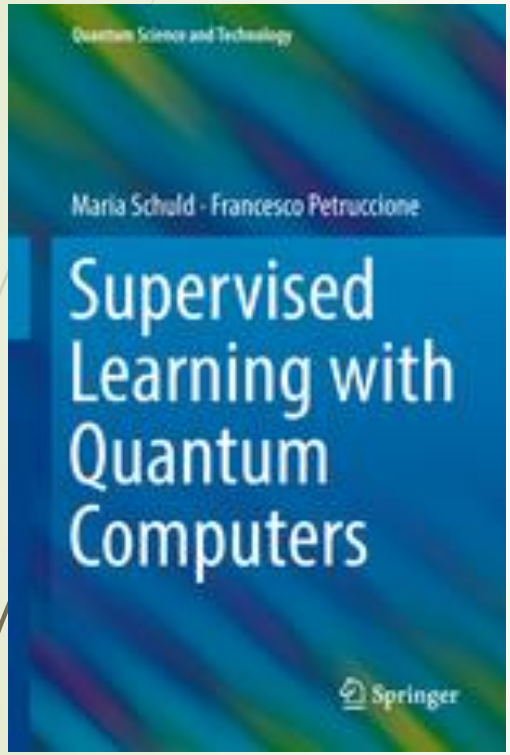
https://medium.com/illumination-curated/is-quantum-machine-learning-the-next-thing-6328b594f424

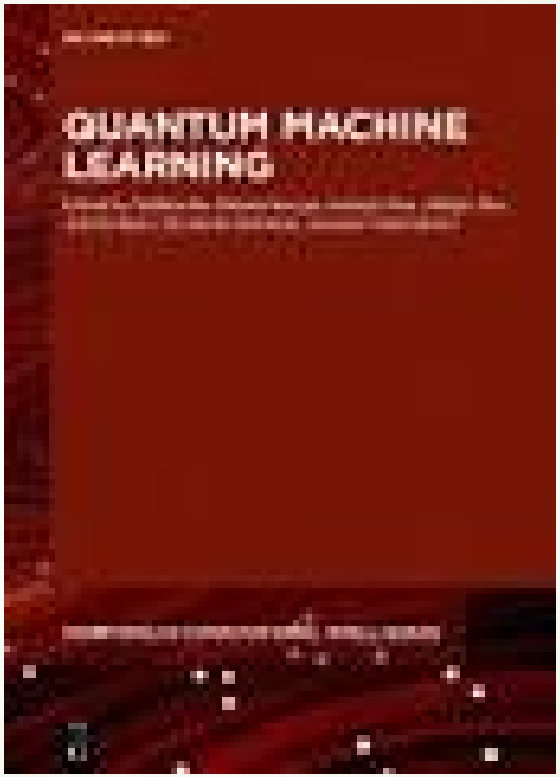
- Quantum Machine Learning Is The Next Big Thing

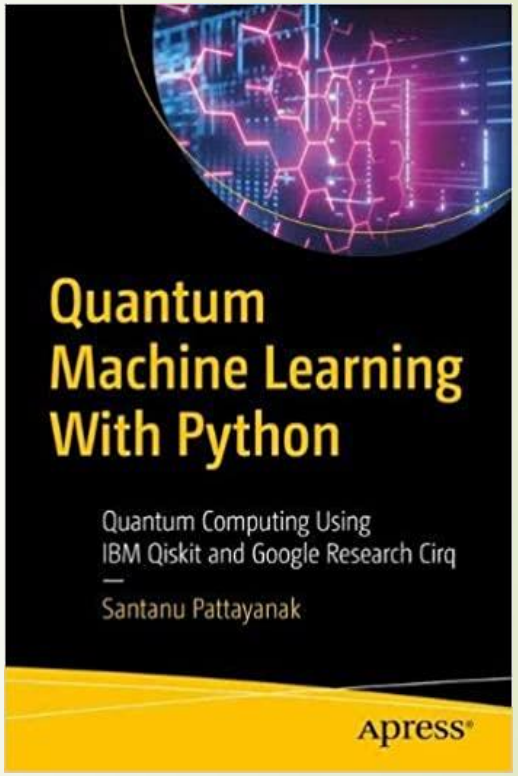
https://thequantumdaily.com/2020/05/28/quantum-machine-learning-is-the-next-big-thing/

- Daniel K. Park, Carsten Blank, Francesco Petruccione, **The theory of the quantum kernel-based binary classifier,** Physics Letters A, Volume 384, Issue 21, 2020, 126422

# Quantum approaches to classification



# Complexity of Classification

- Ana C. Lorena, Luís P. F. Garcia, Jens Lehmann, Marcilio C. P. Souto, and Tin Kam Ho. 2019. How Complex Is Your Classification Problem?: A Survey on Measuring Classification Complexity. ACM Comput. Surv. 52, 5, Article 107 (September 2019), 34 pages. https://doi.org/10.1145/3347711
- Each measure provides a distinct perspective on classification complexity, a combination of different measures is advised. Nonetheless, whether there is a subset of the complexity measures that can be considered core to stress the difficulty of problems from different application domains is still an **open issue.**

# What about clustering?

**A density-based statistical analysis of graph clustering algorithm performance**

Pierre Miasnikof, Alexander Y Shestopaloff, Anthony J Bonner, Yuri Lawryshyn, Panos M Pardalos

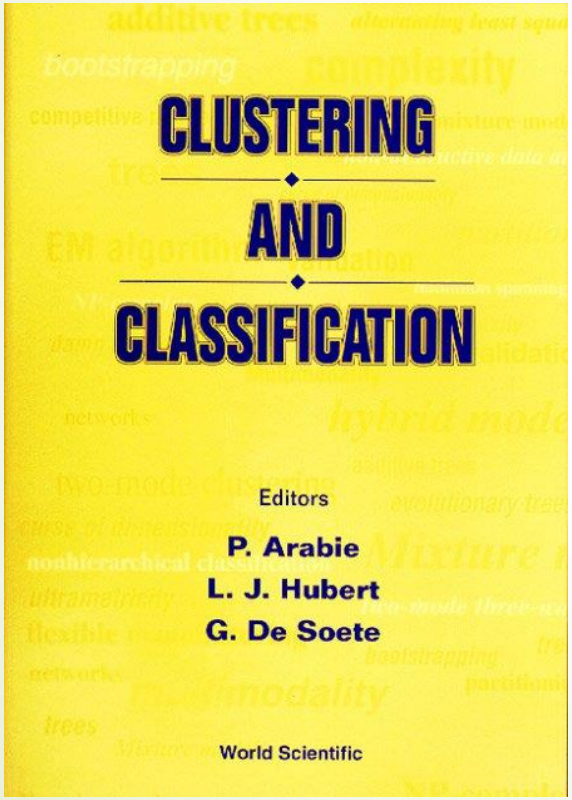
*Journal of Complex Networks*, Volume 8, Issue 3, June 2020, cnaa012, https://doi.org/10.1093/comnet/cnaa012

# Complexity measures

- (1) **Feature-based measures**, which characterize how informative the available features are to separate the classes;
- (2) **Linearity measures**, which try to quantify whether the classes can be linearly separated;
- (3) **Neighborhood measures**, which characterize the presence and density of same or different classes in local neighborhoods;
- (4) **Network measures**, which extract structural information from the dataset by modeling it as a graph;
- (5) **Dimensionality measures**, which evaluate data sparsity based on the number of samples relative to the data dimensionality;
- (6) **Class imbalance measures**, which consider the ratio of the numbers of examples between classes.

**Clustering and Classification** (P Arabie, L J Hubert, and G De Soete

https://doi.org/10.1142/1930 | January 1996)



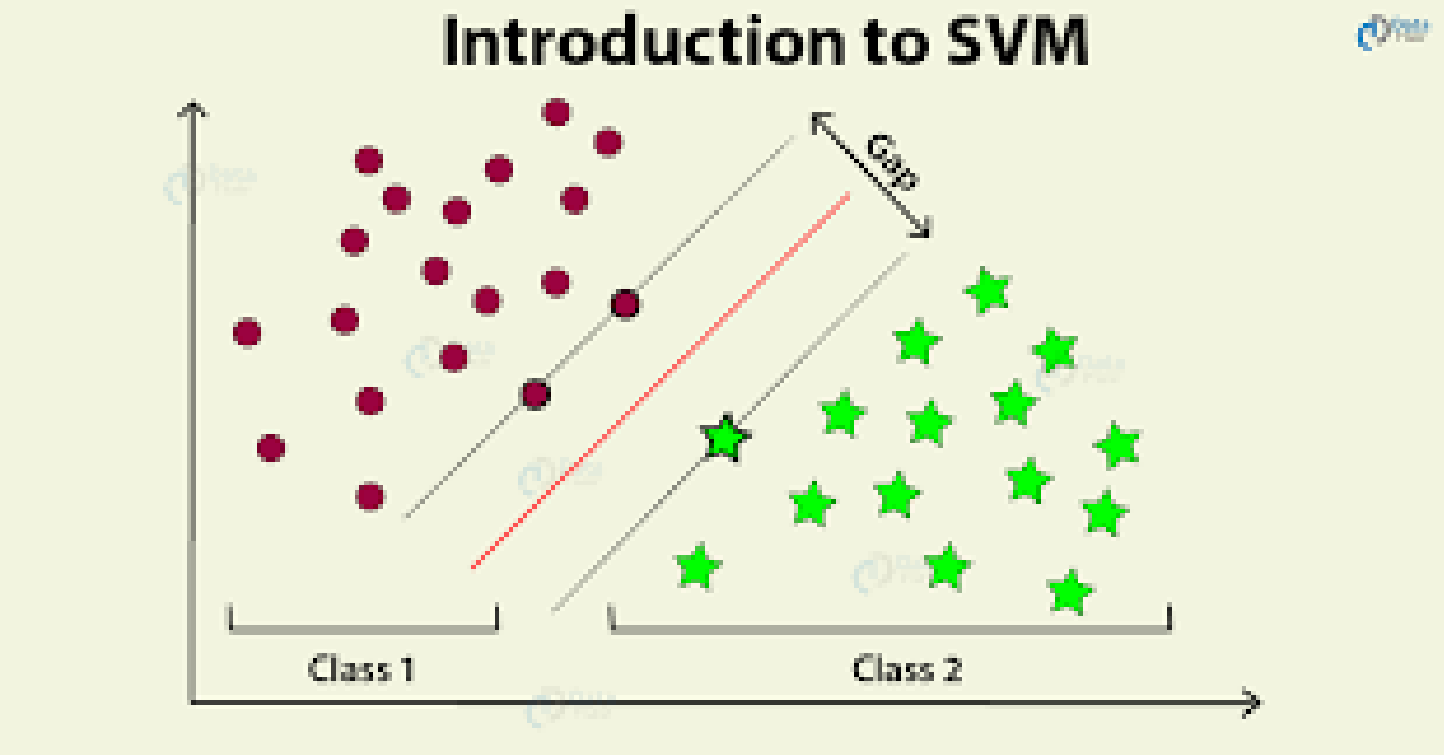
# Any issues with data analysis?

## Machine Learning Paradoxes

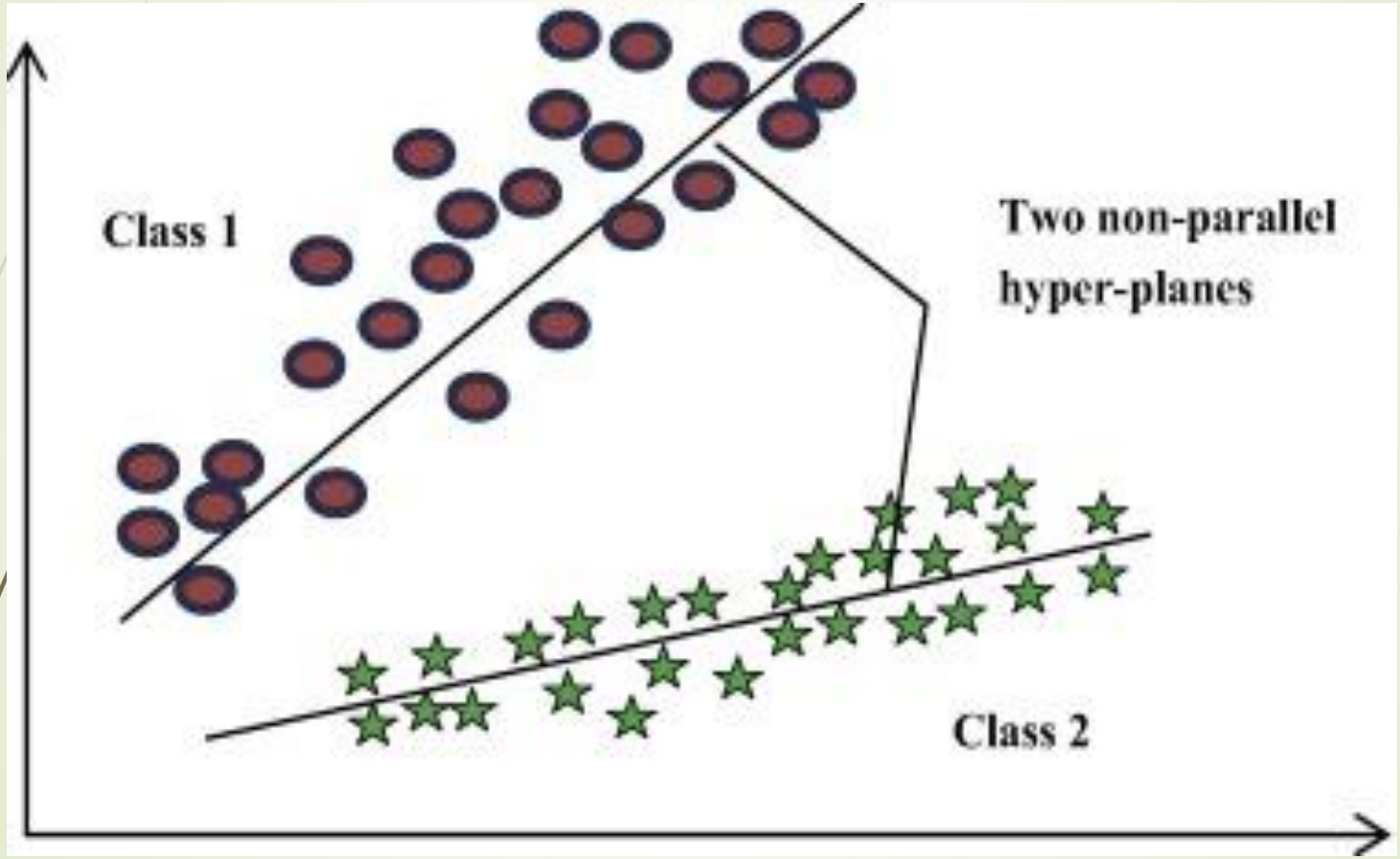
https://medium.com/dataseries/five-machine-learning-paradoxes-that-will-change-the-way-you-think-about-data-3b82513482b8


Five Machine Learning Paradoxes that will Change the Way You Think About Data

# Basic Support Vector Machines (SVM)

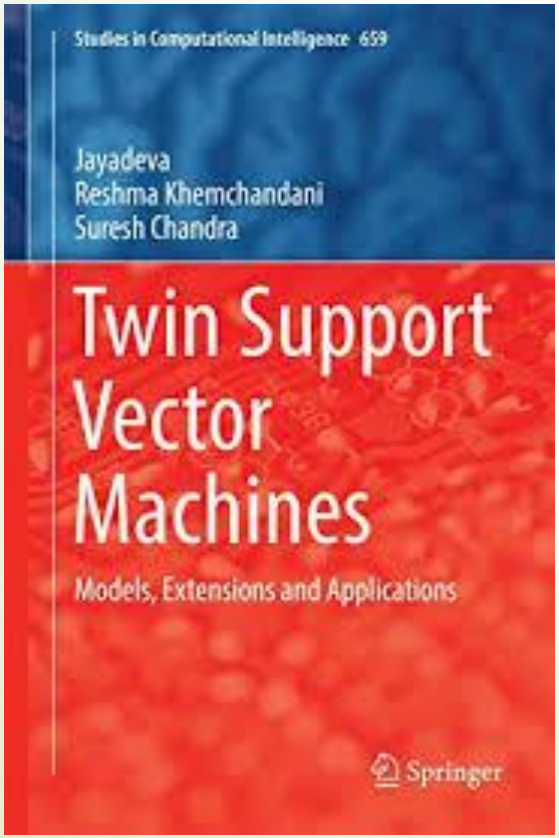


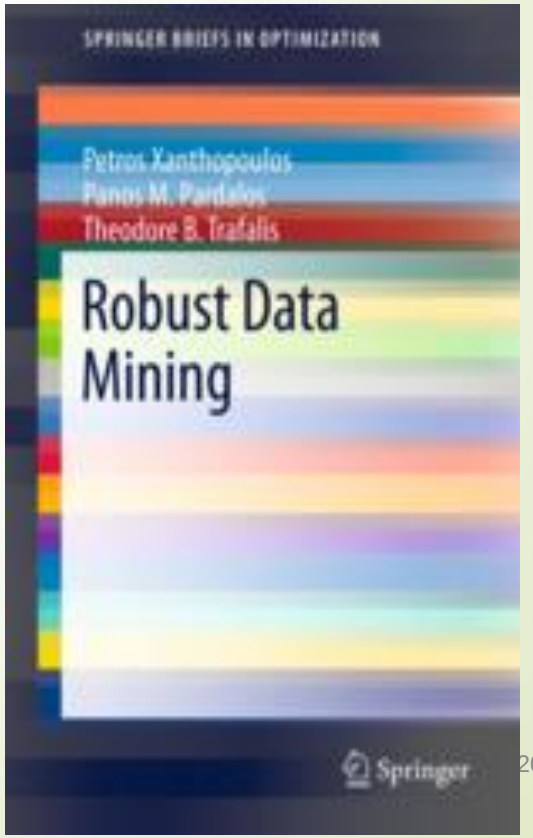
# Twin support vector machines



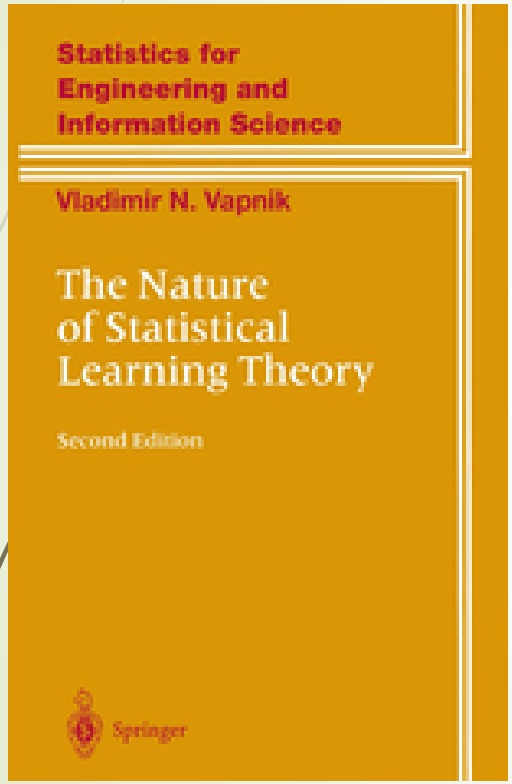
# Many Models of SVM

Wang, X., Pardalos, P.M. A Survey of Support Vector Machines with Uncertainties. *Ann. Data. Sci.* **1,** 293–309 (2014). https://doi.org/10.1007/s40745-014-0022-8

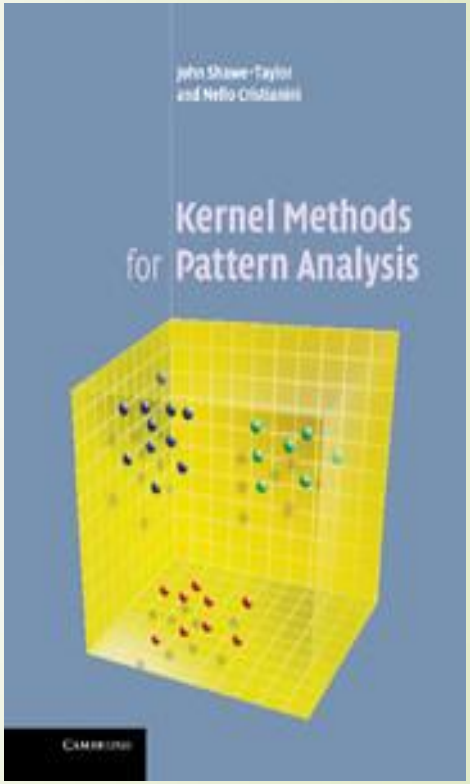


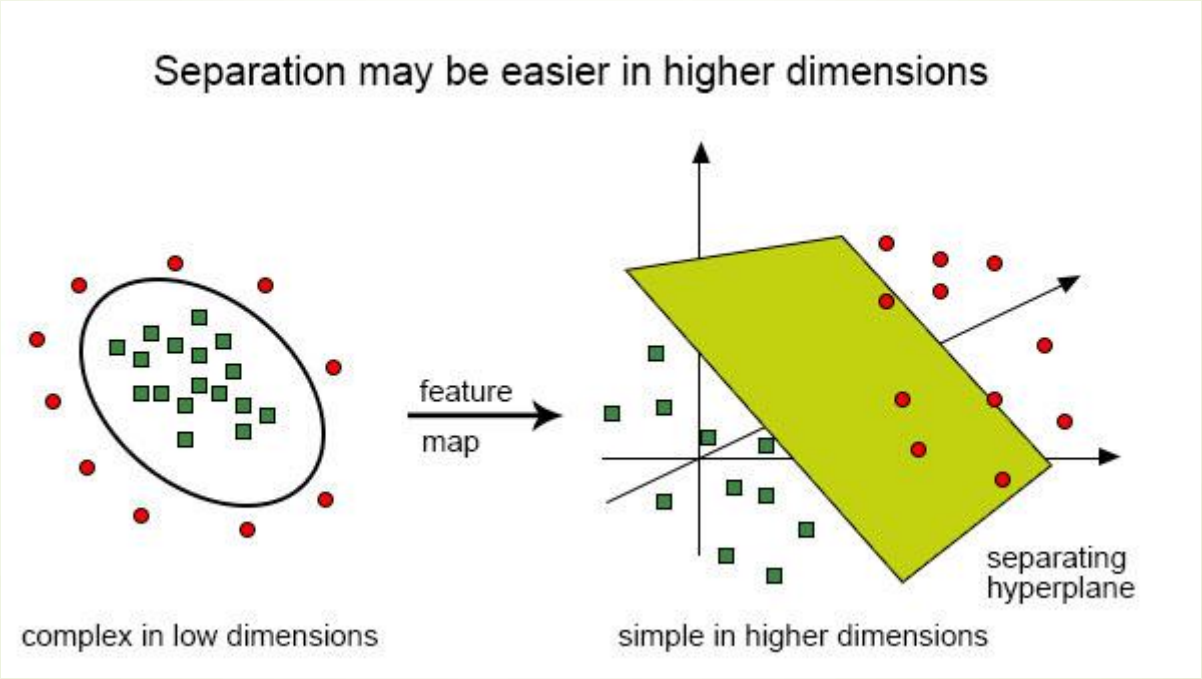
# Explosive research on svm







# Kernels - see e.g. https://www.educba.com/kernel-methods/

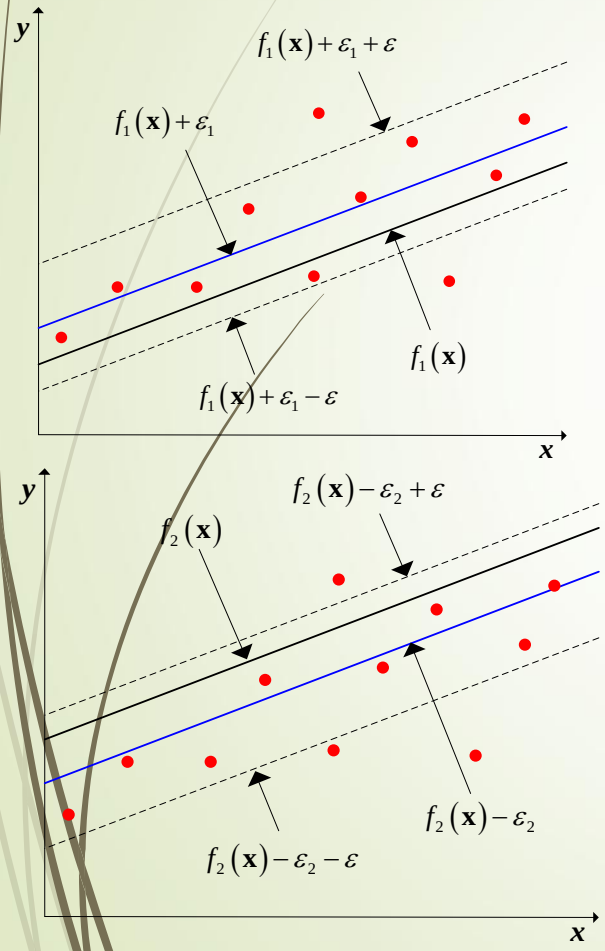


# Nonparallel support vector regression

- Structural risk minimization(SRM) principle. The SRM principle addresses overfitting by balancing the model's complexity against its success at fitting the training data. This principle was first set out in a 1974 paper by Vladimir Vapnik and Alexey Chervonenkis
- Sparsity of the model (number of support vectors). The decision functions constructed by support vector machines usually depend only on a subset of the training set—the so-called **support vectors**

# Nonparallel support vector regression

## Primal problem




$$\min_{\mathbf{w}_1,b_1,\boldsymbol{\eta}_1,\boldsymbol{\eta}_1^*,\boldsymbol{\xi}_1} \frac{1}{2}\mathbf{w}_1^T\mathbf{w}_1 + C_1(\|\boldsymbol{\eta}_1\|_1 + \|\boldsymbol{\eta}_1^*\|_1) + C_3\|\boldsymbol{\xi}_1\|_1$$

$$\text{s.t.} \quad \mathbf{y} - \mathbf{e}\varepsilon_1 - (\mathbf{A}\mathbf{w}_1 + \mathbf{e}b_1) \le \boldsymbol{\eta}_1 + \mathbf{e}\varepsilon \qquad \text{Lower bound}$$

$$-\mathbf{y} + \mathbf{e}\varepsilon_1 + (\mathbf{A}\mathbf{w}_1 + \mathbf{e}b_1) \le \boldsymbol{\eta}_1^* + \mathbf{e}\varepsilon \qquad \text{bound}$$

$$\mathbf{y} - (\mathbf{A}\mathbf{w}_1 + \mathbf{e}b_1) \ge \mathbf{e}\varepsilon_1 - \boldsymbol{\xi}_1$$

$$\boldsymbol{\eta}_1, \boldsymbol{\eta}_1^*, \boldsymbol{\xi}_1 \ge \mathbf{0}$$

$$\min_{\mathbf{w}_2,b_2,\boldsymbol{\eta}_2,\boldsymbol{\eta}_2^*,\boldsymbol{\xi}_2} \frac{1}{2}\mathbf{w}_2^T\mathbf{w}_2 + C_2(\|\boldsymbol{\eta}_2\|_1 + \|\boldsymbol{\eta}_2^*\|_1) + C_4\|\boldsymbol{\xi}_2\|_1$$

$$\text{s.t.} \quad \mathbf{y} + \mathbf{e}\varepsilon_2 - (\mathbf{A}\mathbf{w}_2 + \mathbf{e}b_2) \le \boldsymbol{\eta}_2 + \mathbf{e}\varepsilon \qquad \text{Upper bound}$$

$$-\mathbf{y} - \mathbf{e}\varepsilon_2 + (\mathbf{A}\mathbf{w}_2 + \mathbf{e}b_2) \le \boldsymbol{\eta}_2^* + \mathbf{e}\varepsilon \qquad \text{bound}$$

$$(\mathbf{A}\mathbf{w}_2 + \mathbf{e}b_2) - \mathbf{y} \ge \mathbf{e}\varepsilon_2 - \boldsymbol{\xi}_2$$

$$\boldsymbol{\eta}_2, \boldsymbol{\eta}_2^*, \boldsymbol{\xi}_2 \ge \mathbf{0}$$

# NPSVR

## Advantages of NPSVR

- Equivalent sparseness to the standard SVR;
- Does not involve computing inverse matrix;
- Same formulation as the standard SVR. An SMO-type solver can be developed to accelerate the training process;

**Algorithm1:** Sub-optimization

**Instruction:** $k = 1$ for problem (38); $k = 2$ for problem (39)

**Step1.** For each $i \in U_k'$: if $(\partial W_k/\partial t_{ki}^-)|_{t_{kU_k'}=0}(\partial W_k/\partial t_{ki}^+)|_{t_{kU_k'}=0} \le 0$, remove $i$ from the work set $U_k$; otherwise, determine $\mathbf{z}_{kU_k'}$, where $z_{ki} = \left(\partial g_k(\beta_{ki}+t_{ki}) + \partial g_k(\beta_{ks}+t_{ks})\right)/\partial t_{ki}$ can be calculated according to Eq. (81) or (82);

**Step2.** Obtain $\mathbf{t}_{kU_k'}^{\min}$ and $t_{ks}^{\min}$ according to Eq. (83) and (76);

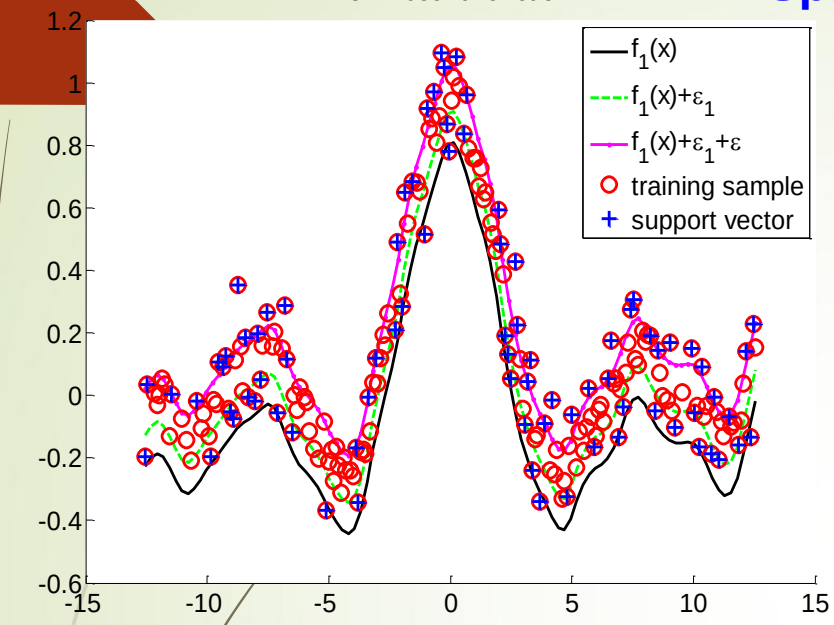
**Step3.** For each $i \in U_k$: Obtain $t_{ki}'^{\min}$;

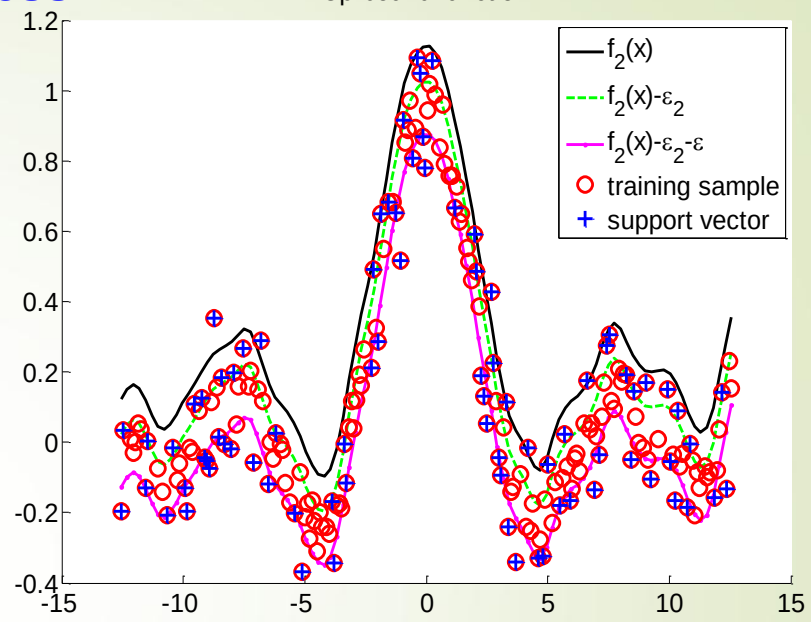
**Step4.** Renew $\boldsymbol{\beta}_{kU_k}^{new} = \boldsymbol{\beta}_{kU_k} + r_k \mathbf{t}_{kU_k}^{\min}$, where $r_k = \min_{i \in U_k}\left(t_{ki}'^{\min}/t_{ki}^{\min}\right)$;

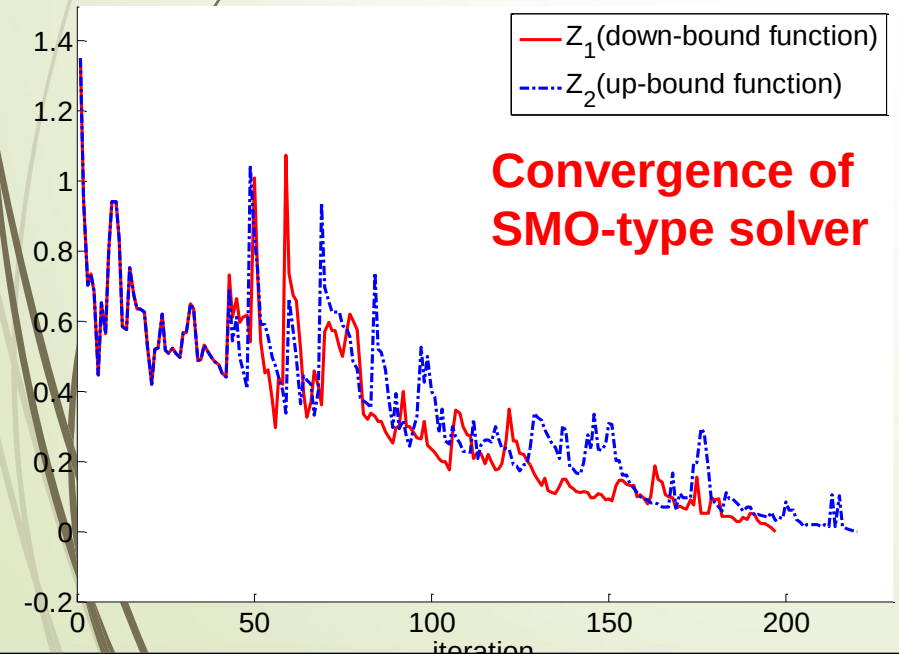
**Algorithm2:** SMO-type solver
**Instruction:** $k = 1$ for problem (38); $k = 2$ for problem (39)
ExamAll=1;
*iter* =0
Initialize $\boldsymbol{\beta}_k = \mathbf{0}$;
**While** ExamAll
    ExamAll=0;
    *iter* = *iter*+1;
    **If** condition (94) is not achieved
        Determine the pair of working variables according to Eq. (93);
        **If** *iter*-1
            Select loop variables from previous working sets of preceding $\min(iter-1, I_c)$ iterations;
        **End**
        Obtain the working set $U_k$;
        Execute **Algorithm1**;
        ExamAll=1;
    **Else if** condition (69) or (71) is not achieved
        Determine the pair of working variables according to Eq. (92);
        Obtain the working set $U_k$;
        Execute **Algorithm1**;
        ExamAll=1;
    **End**
**End**

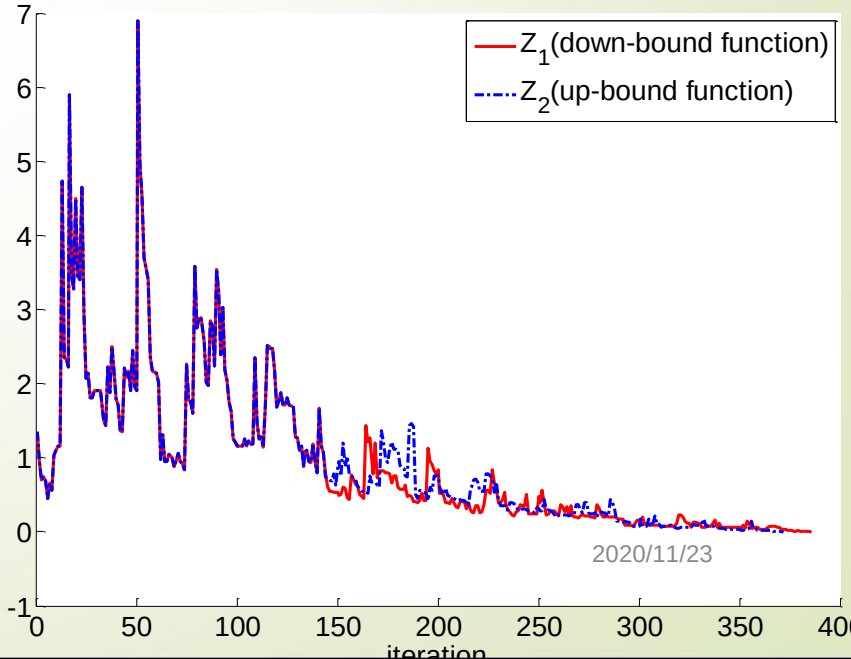
# NPSVR

## Sparseness



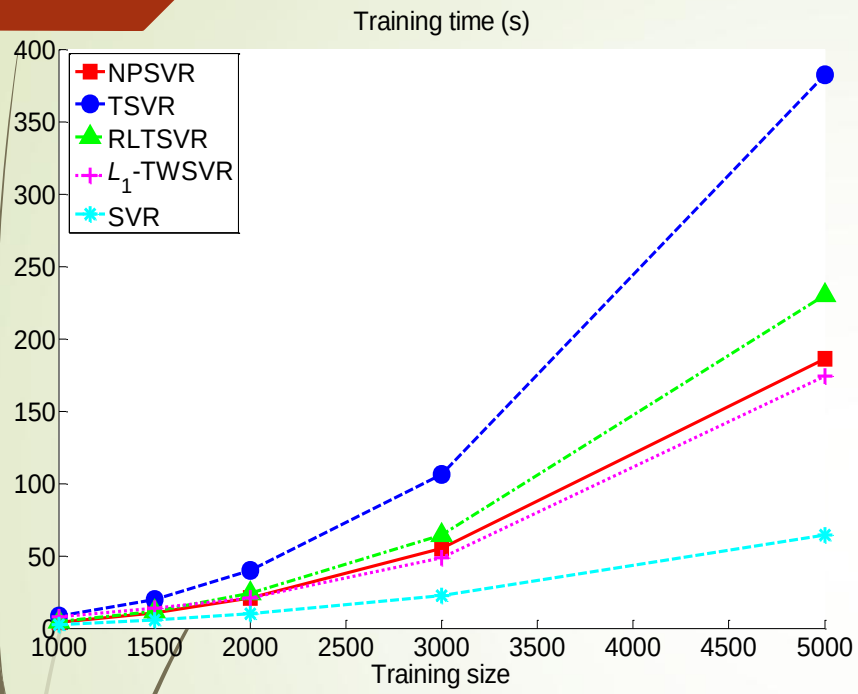



# NPSVR






**Training speed test of large-scale data sets**

## Accuracy test of UCI data sets

| Dataset training test | Algorithm | RMSE | MAE | SSE/SST | SSR/SST | SMAPE | MASE |
|---|---|---|---|---|---|---|---|
| AM (7 × 200) (7 × 150) | NPSVR | **0.0554** | **0.0414** | **0.1078** | 0.9278 | **0.1652** | **0.3546** |
| | TSVR | 0.0570 | 0.0447 | 0.1144 | 0.9459 | 0.1783 | 0.3825 |
| | RLTSVR | 0.0581 | 0.0436 | 0.1190 | 0.9617 | 0.1731 | 0.3729 |
| | $L_1$-TWSVR | 0.0603 | 0.0445 | 0.1281 | **0.8992** | 0.1724 | 0.3807 |
| | SVR | 0.0568 | 0.0439 | 0.1136 | 0.9289 | 0.1761 | 0.3759 |
| BH (13 × 200) (13 × 306) | NPSVR | **0.0737** | 0.0522 | **0.1306** | 0.9514 | **0.1718** | 0.4616 |
| | TSVR | 0.0777 | 0.0571 | 0.1449 | 0.9635 | 0.1958 | 0.5047 |
| | RLTSVR | 0.0776 | 0.0571 | 0.1445 | 0.9446 | 0.1821 | 0.5045 |
| | $L_1$-TWSVR | 0.1098 | 0.0699 | 0.2894 | **0.6469** | 0.2063 | 0.6182 |
| | SVR | 0.0768 | **0.0517** | 0.1418 | 0.8705 | 0.1737 | **0.4566** |
| CH (6 × 100) (6 × 109) | NPSVR | **0.0762** | 0.0354 | **0.3128** | **0.3813** | 0.5336 | 0.5065 |
| | TSVR | 0.0828 | **0.0346** | 0.3687 | 0.4183 | **0.4713** | **0.4953** |
| | RLTSVR | 0.0835 | 0.0367 | 0.3752 | 0.3922 | 0.5405 | 0.5244 |
| | $L_1$-TWSVR | 0.0820 | 0.0352 | 0.3624 | 0.4537 | 0.6594 | 0.5028 |
| | SVR | 0.0778 | 0.0359 | 0.3258 | 0.3901 | 0.5318 | 0.5127 |
| CCS (8 × 500) (8 × 530) | NPSVR | **0.0811** | **0.0590** | **0.1509** | 0.9089 | **0.1766** | **0.3502** |
| | TSVR | 0.0836 | 0.0616 | 0.1603 | 0.9550 | 0.1855 | 0.3653 |
| | RLTSVR | 0.0876 | 0.0657 | 0.1763 | 0.9226 | 0.2047 | 0.3896 |
| | $L_1$-TWSVR | 0.1101 | 0.0852 | 0.2783 | **0.6005** | 0.2546 | 0.5051 |
| | SVR | 0.0815 | 0.0602 | 0.1523 | 0.9102 | 0.1782 | 0.3569 |
| EE1 (8 × 300) (8 × 468) | NPSVR | **0.0234** | **0.0151** | **0.0074** | 1.0239 | **0.0571** | **0.2114** |
| | TSVR | 0.0310 | 0.0218 | 0.0130 | 0.9921 | 0.1126 | 0.3059 |
| | RLTSVR | 0.0552 | 0.0399 | 0.0411 | **0.9521** | 0.1242 | 0.5602 |
| | $L_1$-TWSVR | 0.1570 | 0.1380 | 0.3323 | 1.2435 | 0.3730 | 1.9372 |
| | SVR | 0.0239 | 0.0189 | 0.0077 | 0.9768 | 0.0895 | 0.2657 |
| EE2 (8 × 300) (8 × 468) | NPSVR | **0.0441** | **0.0353** | **0.0319** | 0.9666 | 0.1590 | **0.4081** |
| | TSVR | 0.0513 | 0.0424 | 0.0432 | 0.9397 | 0.2012 | 0.4909 |
| | RLTSVR | 0.0750 | 0.0553 | 0.0923 | **0.9214** | 0.2229 | 0.6400 |
| | $L_1$-TWSVR | 0.0793 | 0.0563 | 0.1033 | 0.9528 | 0.2110 | 0.6515 |
| | SVR | 0.0480 | 0.0358 | 0.0379 | 0.9706 | **0.1551** | 0.4144 |
| YH (6 × 150) (6 × 158) | NPSVR | **0.0098** | **0.0075** | **0.0015** | 0.9924 | 0.3698 | **0.0426** |
| | TSVR | 0.0115 | 0.0089 | 0.0021 | 0.9839 | **0.3322** | 0.0510 |
| | RLTSVR | 0.0446 | 0.0303 | 0.0316 | 0.8420 | 0.5875 | 0.1728 |
| | $L_1$-TWSVR | 0.1313 | 0.1052 | 0.2733 | **0.6528** | 0.9550 | 0.6001 |
| | SVR | 0.0117 | 0.0088 | 0.0022 | 0.9947 | 0.4096 | 0.0504 |
| CCPP (4 × 300) (4 × 300) | NPSVR | 0.0561 | 0.0443 | 0.0660 | 1.0010 | **0.1285** | 0.1775 |
| | TSVR | 0.0563 | 0.0447 | 0.0664 | 0.9975 | 0.1305 | 0.1790 |
| | RLTSVR | **0.0547** | **0.0441** | **0.0628** | **0.9878** | 0.1319 | **0.1769** |
| | $L_1$-TWSVR | 0.0583 | 0.0472 | 0.0713 | 0.9973 | 0.1430 | 0.1890 |
| | SVR | 0.0559 | 0.0442 | 0.0656 | 1.0057 | 0.1287 | 0.1773 |
| PT1 (16 × 500) (16 × 500) | NPSVR | **0.2090** | **0.1684** | **0.7895** | 0.3917 | 0.4097 | 8.6203 |
| | TSVR | 0.2263 | 0.1895 | 0.9257 | **0.1086** | 0.4512 | 9.6972 |
| | RLTSVR | 0.2128 | 0.1747 | 0.8180 | 0.3202 | 0.4213 | 8.9402 |
| | $L_1$-TWSVR | 0.2157 | 0.1767 | 0.8409 | 0.2379 | 0.4230 | 9.0440 |
| | SVR | 0.2112 | **0.1684** | 0.8063 | 0.3830 | **0.4078** | **8.6175** |
| PT2 (16 × 500) (16 × 500) | NPSVR | 0.2065 | 0.1626 | 0.8547 | 0.1853 | 0.3924 | 9.3476 |
| | TSVR | 0.2189 | 0.1755 | 0.9606 | **0.1152** | 0.4227 | 10.0886 |
| | RLTSVR | **0.2026** | 0.1622 | **0.8228** | 0.2236 | 0.3930 | 9.3270 |
| | $L_1$-TWSVR | 0.2092 | 0.1657 | 0.8769 | 0.1771 | 0.4016 | 9.5284 |
| | SVR | 0.2064 | **0.1609** | 0.8539 | 0.2000 | **0.3901** | **9.2482** |

**Tang Long, Tian Yingjie*, Yang Chunyan. Nonparallel support vector regression and its SMO-type solver. *Neural networks*, 2018, 105: 431-446.**

# Ramp loss function based nonparallel support vector regression (RL-NPSVR)

- A Ramp $\varepsilon$ -insensitive loss function is constructed to compel as many training samples as possible to locate the down (up) bound hyperplane within a $2\varepsilon$ -wide band
- A Ramp loss function is constructed to keep as many training samples as possible above (below) the down (up) bound hyperplane
- A regularized term is added into each primal problem by rigidly following the SRM principle
- Trading Convexity for Scalability

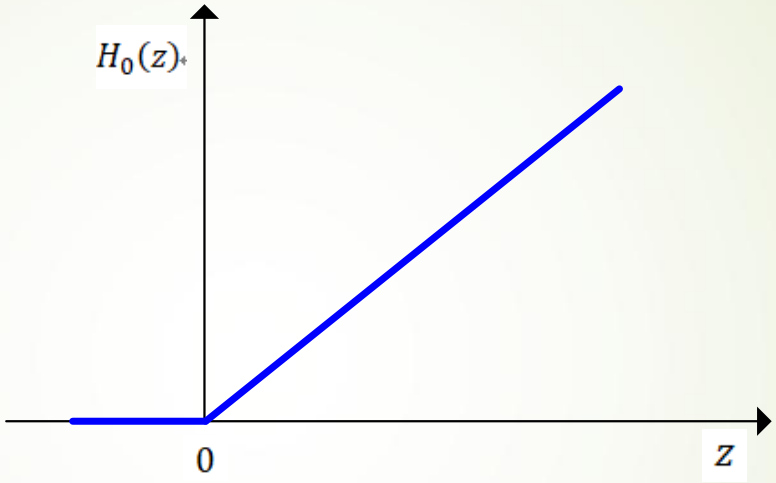
# Ramp-loss NPSVR

Compared to the existing TSVRs, our proposed RL-NPSVR has the following merits:

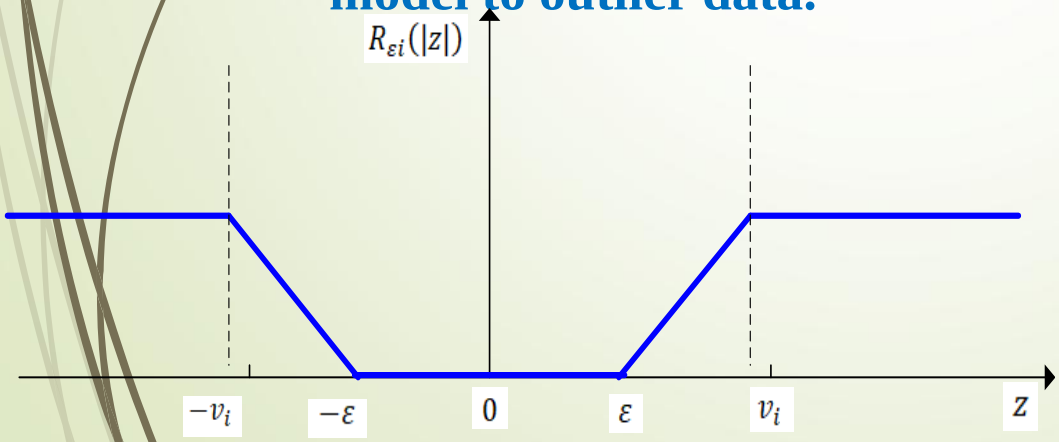
- (1) It can explicitly filter noise and outlier suppression in the training process
- (2) RL-NPSVR has inherent sparseness as the standard SVR, and the adopted Ramp-type loss functions make it sparser
- (3) The dual of each reconstructed convex optimization problem has the same formulation as that of the standard SVR, so computing inverse matrix is avoided and the kernel trick can be directly applied to the nonlinear case
- (4) Available SMO-type fast algorithm exists to solve this problem

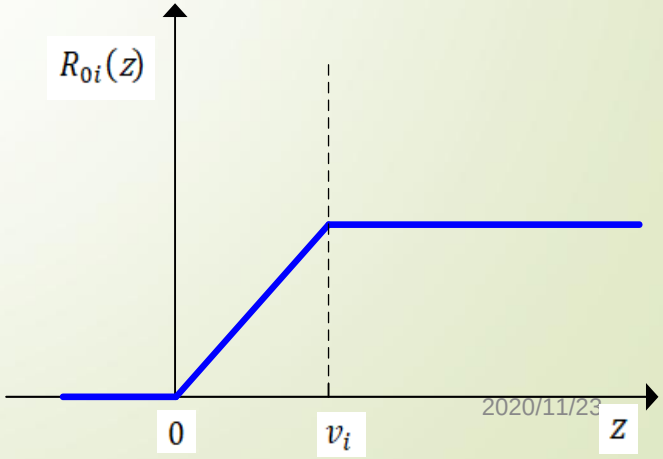
# Ramp-loss NPSVR

- **Original loss function is sensitive to the outlier data, limiting the generalization ability.**




- **Ramp-loss is adopted to improve the robustness of the model to outlier data.**



# Ramp-loss NPSVR

## Dual problem

$$\min_{\tilde{\boldsymbol{\alpha}}_1,\bar{\boldsymbol{\alpha}}_1,\bar{\boldsymbol{\beta}}_1} \quad \frac{1}{2}\left(\tilde{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\beta}}_1\right)^T \mathbf{A}\mathbf{A}^T\left(\tilde{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\beta}}_1\right)$$

$$-\left(\tilde{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\beta}}_1\right)^T \mathbf{y}+\left(\tilde{\boldsymbol{\alpha}}_1+\bar{\boldsymbol{\alpha}}_1\right)^T \mathbf{e}\varepsilon$$

$$\text{s.t.}$$

$$\left(\tilde{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\alpha}}_1-\bar{\boldsymbol{\beta}}_1\right)^T \mathbf{e}=0$$

$$-\tilde{\boldsymbol{\theta}}_1^t \le \tilde{\boldsymbol{\alpha}}_1 \le C_1\mathbf{e}-\tilde{\boldsymbol{\theta}}_1^t$$

$$\bar{\boldsymbol{\theta}}_1^t \le \bar{\boldsymbol{\alpha}}_1 \le C_1\mathbf{e}+\bar{\boldsymbol{\theta}}_1^t$$

$$\boldsymbol{\delta}_1^t \le \bar{\boldsymbol{\beta}}_1 \le C_3\mathbf{e}+\boldsymbol{\delta}_1^t$$

**Non-convexity：CCCP (concave–convex programming)**

**Algorithm:** RL-NPSVR
**Instruction:** $k=1$ for down-bound hyperplane; $k=2$ for up-bound hyperplane
**Input:** Training set $S$ (1)
Parameters $\varepsilon_1, \varepsilon,\ C_1, C_3, C_2, C_4$
Kernel function $K(\mathbf{x},\ \mathbf{x}')$ ($(\mathbf{x}\cdot\mathbf{x}')$ for linear case)
$\tau > \varepsilon$
**Step.1** Initialize $\tilde{\theta}_k^0=\mathbf{0}$, $\bar{\theta}_k^0=\mathbf{0}$ and $\delta_k^0=0$; preprocess the training set according to Eqs. (67) and (68); take the Ramp loss parameters $v_i=\tau y_i'$; set $t=0$;
**Step.2** Solve the following QPP:

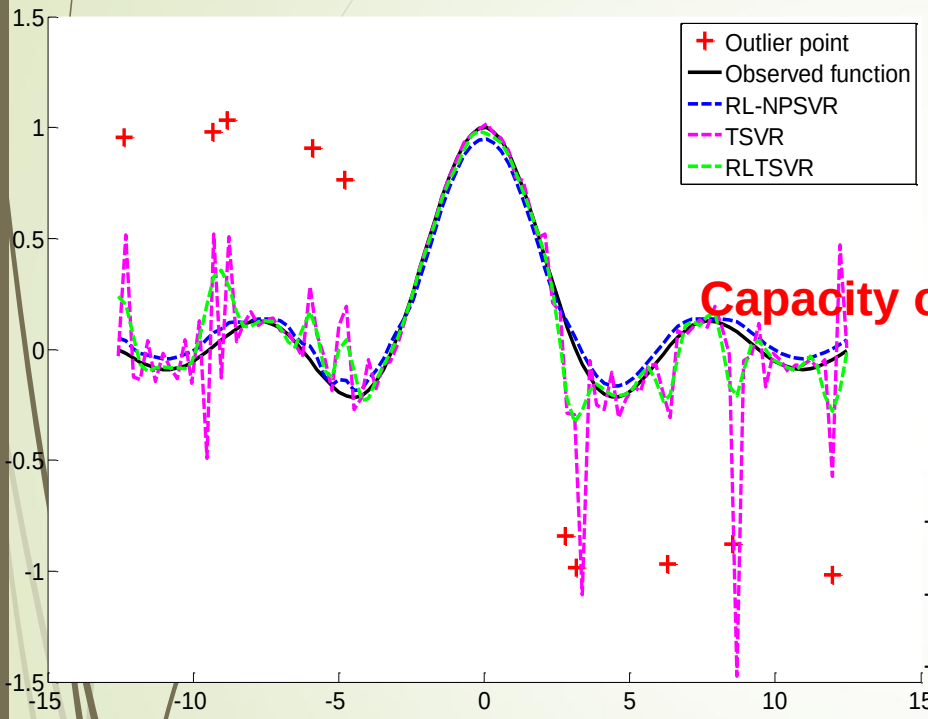
$$\min_{\tilde{\alpha}_k^t,\bar{\alpha}_k^t,\bar{\beta}_k^t} \ \frac{1}{2}(\tilde{\alpha}_k^t-\bar{\alpha}_k^t+(-1)^k\bar{\beta}_k^t)^T K(\mathbf{A},\mathbf{A}^T)(\tilde{\alpha}_k^t-\bar{\alpha}_k^t+(-1)^k\bar{\beta}_k^t)$$
$$-(\tilde{\alpha}_k^t-\bar{\alpha}_k^t+(-1)^k\bar{\beta}_k^t)^T\mathbf{y}'+(\tilde{\alpha}_k^t+\bar{\alpha}_k^t)^T\mathbf{e}\varepsilon$$
$$\text{s.t.} \quad (\tilde{\alpha}_k^t-\bar{\alpha}_k^t+(-1)^k\bar{\beta}_k^t)^T\mathbf{e}=\mathbf{0} \qquad (70)$$
$$-\tilde{\theta}_k^t \le \tilde{\alpha}_k^t \le C_k\mathbf{e}-\tilde{\theta}_k^t$$
$$\bar{\theta}_k^t \le \bar{\alpha}_k^t \le C_k\mathbf{e}+\bar{\theta}_k^t$$
$$-(-1)^k\delta_k^t \le \bar{\beta}_k^t \le C_{k+2}\mathbf{e}-(-1)^k\delta_k^t$$

**Step.3** Obtain $f_k^t(\mathbf{x})=K(\mathbf{A}_i,\mathbf{A}^T)(\tilde{\alpha}_k^t-\bar{\alpha}_k^t+(-1)^k\bar{\beta}_k^t)+b_k^t$;
**Step.4** Compute $\tilde{\theta}_k^{t+1}, \bar{\theta}_k^{t+1}$ and $\delta_k^{t+1}$ according to Eqs. (21)–(23) or (27)–(29);
**Step.5** If $(\tilde{\theta}_k^{t+1}, \bar{\theta}_k^{t+1}, \delta_k^{t+1})=(\tilde{\theta}_k^t, \bar{\theta}_k^t, \delta_k^t)$, converge; else set $t=t+1$, go back to **Step.2**;
**Output:** Decision hyperplane $f_k(\mathbf{x})=f_k^t(\mathbf{x})+y_0-1$;

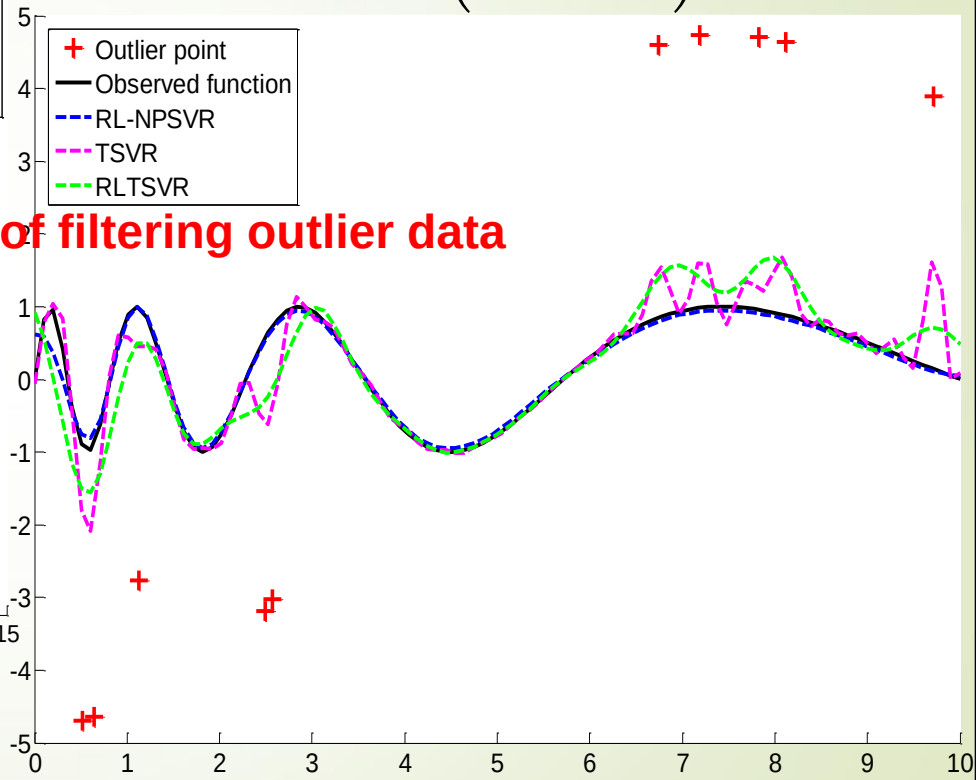
**SMO-type solver of NPSVR can be used to solve each sub-optimization.**

# Ramp-loss NPSVR

$$f(x) = \frac{\sin x}{x}, x \in [-4\pi, 4\pi]\backslash\{0\}$$




$$f(x) = \sin\left(\frac{9\pi}{0.35x + 1}\right), x \in [0,10]$$




**Capacity of filtering outlier data**

- **Stochastically generate 200 training points , in which 5% of them are set to outlier points。**

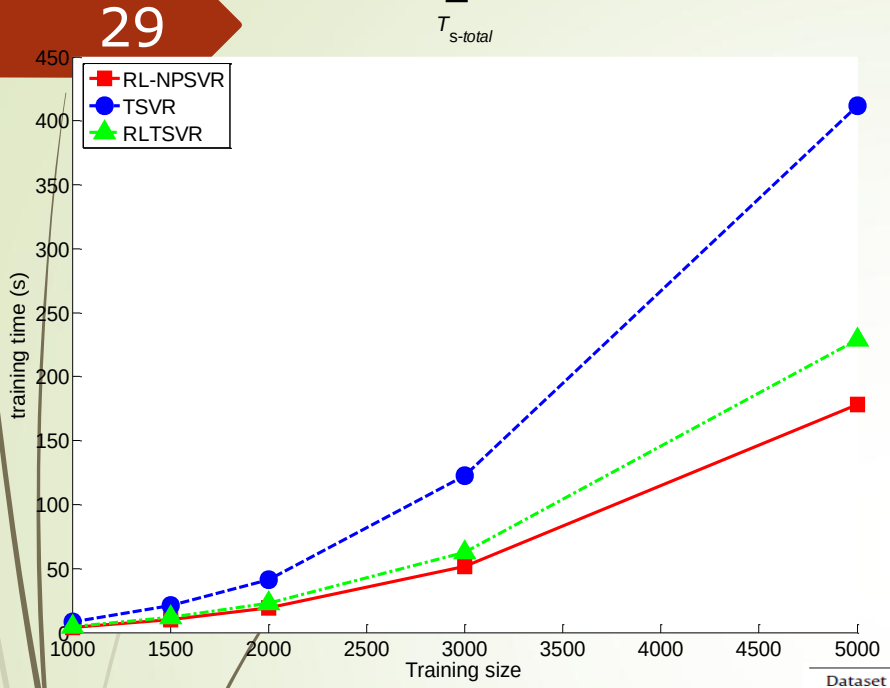
# Ramp-loss NPSVR

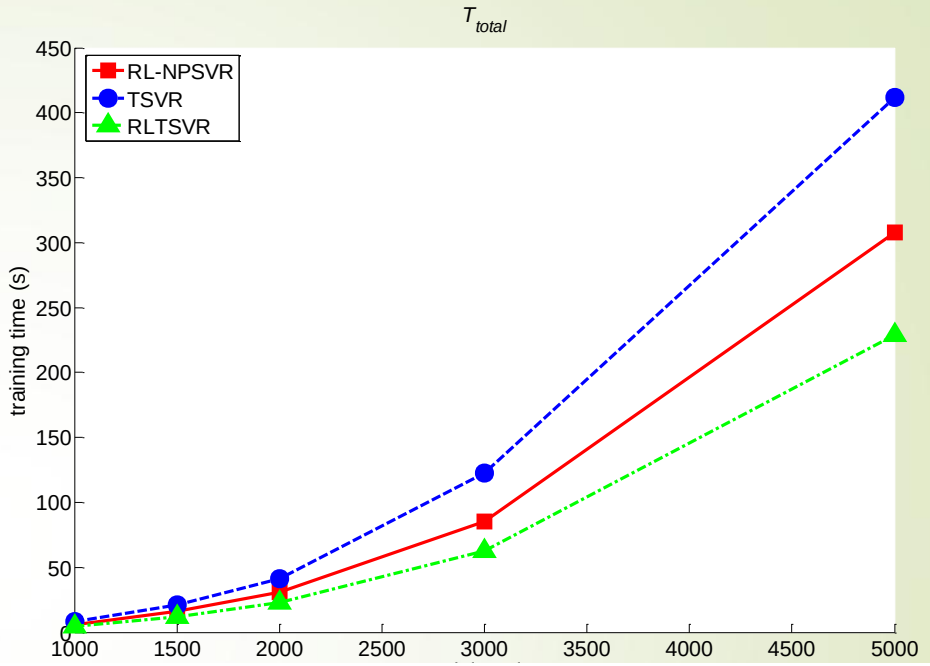
## Accuracy test of UCI data sets

| Dataset Training Test | Algorithm | RMSE | MAE | SSE/SST | SSR/SST | SMAPE | MASE | POSV (%) $f_1 + f_2$ |
|---|---|---|---|---|---|---|---|---|
| AM | RL-NPSVR | **0.0777** | **0.0529** | **0.1439** | **0.9532** | **0.1655** | **0.4386** | **95.5 + 92.5** |
| (7 × 200) | TSVR | 0.0815 | 0.0558 | 0.1583 | 0.9550 | 0.1790 | 0.4627 | 100 + 100 |
| (7 × 192) | RLTSVR | 0.0805 | 0.0548 | 0.1543 | 0.9760 | 0.1703 | 0.4549 | 100 + 100 |
| BH | RL-NPSVR | 0.0758 | **0.0524** | 0.1382 | 0.9706 | **0.1734** | **0.4635** | **93 + 97** |
| (13 × 200) | TSVR | **0.0748** | 0.0537 | **0.1345** | 0.9631 | 0.1837 | 0.4745 | 100 + 100 |
| (13 × 306) | RLTSVR | 0.0776 | 0.0571 | 0.1445 | **0.9446** | 0.1821 | 0.5045 | 100 + 100 |
| CH | RL-NPSVR | **0.0789** | **0.0319** | **0.3351** | 0.4182 | **0.4551** | **0.4563** | **82 + 77** |
| (6 × 100) | TSVR | 0.0845 | 0.0378 | 0.3844 | 0.4005 | 0.5593 | 0.5401 | 100 + 100 |
| (6 × 109) | RLTSVR | 0.0835 | 0.0367 | 0.3752 | **0.3922** | 0.5405 | 0.5244 | 100 + 100 |
| CCS | RL-NPSVR | **0.0814** | **0.0584** | **0.1520** | **0.9153** | **0.1741** | **0.3465** | **93 + 93** |
| (8 × 500) | TSVR | 0.0827 | 0.0612 | 0.1570 | 0.9582 | 0.1845 | 0.3629 | 100 + 100 |
| (8 × 530) | RLTSVR | 0.0876 | 0.0657 | 0.1763 | 0.9225 | 0.2047 | 0.3896 | 100 + 100 |
| EE1 | RL-NPSVR | **0.0234** | **0.0151** | **0.0074** | 1.0239 | **0.0571** | **0.2114** | **69.7 + 69** |
| (8 × 300) | TSVR | 0.0310 | 0.0218 | 0.0130 | 0.9924 | 0.1123 | 0.3056 | 100 + 100 |
| (8 × 468) | RLTSVR | 0.0552 | 0.0399 | 0.0411 | **0.9521** | 0.1242 | 0.5602 | 100 + 100 |
| EE2 | RL-NPSVR | **0.0445** | **0.0346** | **0.0324** | 0.9742 | **0.1527** | **0.4005** | **90.3 + 88.7** |
| (8 × 300) | TSVR | 0.0513 | 0.0424 | 0.0432 | 0.9397 | 0.2011 | 0.4908 | 100 + 100 |
| (8 × 468) | RLTSVR | 0.0750 | 0.0553 | 0.0923 | **0.9214** | 0.2229 | 0.6400 | 100 + 100 |
| YH | RL-NPSVR | **0.0098** | **0.0075** | **0.0015** | 0.9924 | 0.3698 | **0.0426** | **61.3 + 59.3** |
| (6 × 150) | TSVR | 0.0115 | 0.0090 | 0.0021 | 0.9841 | **0.3292** | 0.0511 | 100 + 100 |
| (6 × 158) | RLTSVR | 0.0446 | 0.0303 | 0.0316 | **0.8420** | 0.5884 | 0.1728 | 100 + 100 |
| CCPP | RL-NPSVR | 0.0563 | 0.0446 | 0.0665 | 1.0161 | **0.1290** | 0.1786 | **88 + 92.7** |
| (4 × 300) | TSVR | 0.0575 | 0.0453 | 0.0692 | 1.0049 | 0.1323 | 0.1817 | 100 + 100 |
| (4 × 300) | RLTSVR | **0.0547** | **0.0441** | **0.0628** | **0.9876** | 0.1319 | **0.1769** | 100 + 100 |
| PT1 | RL-NPSVR | **0.2090** | **0.1711** | **0.8368** | 0.3723 | **0.4004** | **0.6525** | **94.2 + 82.4** |
| (16 × 500) | TSVR | 0.2171 | 0.1853 | 0.9024 | **0.1385** | 0.4297 | 0.7064 | 100 + 100 |
| (16 × 500) | RLTSVR | 0.2127 | 0.1765 | 0.8667 | 0.2686 | 0.4129 | 0.6729 | 100 + 100 |
| PT2 | RL-NPSVR | 0.2100 | **0.1621** | 0.9130 | 0.2103 | **0.3806** | **0.6382** | **97 + 92.8** |
| (16 × 500) | TSVR | 0.2112 | 0.1716 | 0.9238 | **0.1786** | 0.4053 | 0.6754 | 100 + 100 |
| (16 × 500) | RLTSVR | **0.2021** | 0.1626 | **0.8455** | 0.1955 | 0.3842 | 0.6402 | 100 + 100 |

# Ramp-loss NPSVR

## Training speed test of large-scale data sets




| Dataset Training Test | Algorithm | RMSE | MAE | SSE/SST | SSR/SST | SMAPE | MASE | POSV (%) $f_1 + f_2$ |
|---|---|---|---|---|---|---|---|---|
| AM | RL-NPSVR | **0.0777** | **0.0529** | **0.1439** | **0.9532** | **0.1655** | **0.4386** | **95.5 + 92.5** |
| (7 × 200) | TSVR | 0.0815 | 0.0558 | 0.1583 | 0.9550 | 0.1790 | 0.4627 | 100 + 100 |
| (7 × 192) | RLTSVR | 0.0805 | 0.0548 | 0.1543 | 0.9760 | 0.1703 | 0.4549 | 100 + 100 |
| BH | RL-NPSVR | 0.0758 | **0.0524** | 0.1382 | 0.9706 | **0.1734** | **0.4635** | **93 + 97** |
| (13 × 200) | TSVR | **0.0748** | 0.0537 | **0.1345** | 0.9631 | 0.1837 | 0.4745 | 100 + 100 |
| (13 × 306) | RLTSVR | 0.0776 | 0.0571 | 0.1445 | **0.9446** | 0.1821 | 0.5045 | 100 + 100 |
| CH | RL-NPSVR | **0.0789** | **0.0319** | **0.3351** | 0.4182 | **0.4551** | **0.4563** | **82 + 77** |
| (6 × 100) | TSVR | 0.0845 | 0.0378 | 0.3844 | 0.4005 | 0.5593 | 0.5401 | 100 + 100 |
| (6 × 109) | RLTSVR | 0.0835 | 0.0367 | 0.3752 | **0.3922** | 0.5405 | 0.5244 | 100 + 100 |
| CCS | RL-NPSVR | **0.0814** | **0.0584** | **0.1520** | **0.9153** | **0.1741** | **0.3465** | **93 + 93** |
| (8 × 500) | TSVR | 0.0827 | 0.0612 | 0.1570 | 0.9582 | 0.1845 | 0.3629 | 100 + 100 |
| (8 × 530) | RLTSVR | 0.0876 | 0.0657 | 0.1763 | 0.9225 | 0.2047 | 0.3896 | 100 + 100 |
| EE1 | RL-NPSVR | **0.0234** | **0.0151** | **0.0074** | 1.0239 | **0.0571** | **0.2114** | **69.7 + 69** |
| (8 × 300) | TSVR | 0.0310 | 0.0218 | 0.0130 | 0.9924 | 0.1123 | 0.3056 | 100 + 100 |
| (8 × 468) | RLTSVR | 0.0552 | 0.0399 | 0.0411 | **0.9521** | 0.1242 | 0.5602 | 100 + 100 |
| EE2 | RL-NPSVR | **0.0445** | **0.0346** | **0.0324** | 0.9742 | **0.1527** | **0.4005** | **90.3 + 88.7** |
| (8 × 300) | TSVR | 0.0513 | 0.0424 | 0.0432 | 0.9397 | 0.2011 | 0.4908 | 100 + 100 |
| (8 × 468) | RLTSVR | 0.0750 | 0.0553 | 0.0923 | **0.9214** | 0.2229 | 0.6400 | 100 + 100 |
| YH | RL-NPSVR | **0.0098** | **0.0075** | **0.0015** | 0.9924 | 0.3698 | **0.0426** | **61.3 + 59.3** |
| (6 × 150) | TSVR | 0.0115 | 0.0090 | 0.0021 | 0.9841 | **0.3292** | 0.0511 | 100 + 100 |
| (6 × 158) | RLTSVR | 0.0446 | 0.0303 | 0.0316 | **0.8420** | 0.5884 | 0.1728 | 100 + 100 |
| CCPP | RL-NPSVR | 0.0563 | 0.0446 | 0.0665 | 1.0161 | **0.1290** | 0.1786 | **88 + 92.7** |
| (4 × 300) | TSVR | 0.0575 | 0.0453 | 0.0692 | 1.0049 | 0.1323 | 0.1817 | 100 + 100 |
| (4 × 300) | RLTSVR | **0.0547** | **0.0441** | **0.0628** | **0.9876** | 0.1319 | **0.1769** | 100 + 100 |
| PT1 | RL-NPSVR | **0.2090** | **0.1711** | **0.8368** | 0.3723 | **0.4004** | **0.6525** | **94.2 + 82.4** |
| (16 × 500) | TSVR | 0.2171 | 0.1853 | 0.9024 | **0.1385** | 0.4297 | 0.7064 | 100 + 100 |
| (16 × 500) | RLTSVR | 0.2127 | 0.1765 | 0.8667 | 0.2686 | 0.4129 | 0.6729 | 100 + 100 |
| PT2 | RL-NPSVR | 0.2100 | **0.1621** | 0.9130 | 0.2103 | **0.3806** | **0.6382** | **97 + 92.8** |
| (16 × 500) | TSVR | 0.2112 | 0.1716 | 0.9238 | **0.1786** | 0.4053 | 0.6754 | 100 + 100 |
| (16 × 500) | RLTSVR | **0.2021** | 0.1626 | **0.8455** | 0.1955 | 0.3842 | 0.6402 | 100 + 100 |

**Tang Long, Tian Yingjie, Pardalos P. M*, Yang Chunyan. Ramp-loss nonparallel support vector regression: robust, sparse and scalable approximation. *Knowledge-based systems*, 2018, 147: 55-67**

# Regular simplex support vector machine (RSSVM) for the K -class classification

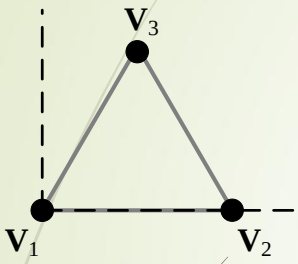
- RSSVM maps the K classes to K vertices of a (K−1) -dimensional regular simplex so that the K-class classification becomes a (K-1)-output learning task
- We measure the training loss by comparing the square of the distance between the output point of each sample and its vertices
- Adding an appropriate regularized term to the primal problem, makes the dual problem a quadratic programming problem, and we developed an exclusive sequential minimization optimization–type solver to accelerate our ability to solve it

# Regular simplex SVM for multi-classification

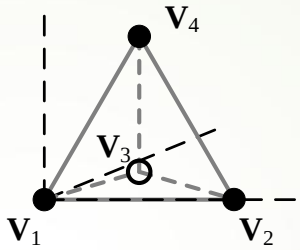
- Limitations of traditional Partitioning *one-versus-one* (1-v-1), *one-versus-rest* (1-v-r) strategies
- Establish multiple sub-binary classifiers, limiting the sparseness of the model
- Lack of definite classifying boundaries
- Individual classifier can hardly use complete information of training samples

# RSSVM

## Primal problem




$\mathbf{V}_1 \longleftrightarrow (0,0)^T$
$\mathbf{V}_2 \longleftrightarrow (1,0)^T$
$\mathbf{V}_3 \longleftrightarrow (0.5,0.866)^T$




$\mathbf{V}_1 \longleftrightarrow (0,0,0)^T$
$\mathbf{V}_2 \longleftrightarrow (1,0,0)^T$
$\mathbf{V}_3 \longleftrightarrow (0.5,0.866,0)^T$
$\mathbf{V}_4 \longleftrightarrow (0.5,0.2887,0.8165)^T$

$$\min_{\mathbf{w},\mathbf{b}} \sum_{j=1}^{K-1} \frac{1}{2}\left(\mathbf{w}_j^T\mathbf{w}_j + b_j^2\right) + C\sum_{i=1}^{N}\sum_{k\neq c_i} \xi_{i,k}$$

$$\text{s.t. } \sum_{j=1}^{K-1}\left(2\left(V_{c_i,j} - V_{k,j}\right)\left(\mathbf{w}_j^T\mathbf{x}_i + b_j\right) + V_{k,j}^2 - V_{c_i,j}^2\right) \geq \varepsilon - \xi_{i,k}, (i = 1,2,\cdots N)$$

$$\xi_{i,k} \geq 0, (i = 1,2,\cdots N)$$

**The classes are mapped to different vertices of a regular simplex, and square distance is used to measure the loss.**

# RSSVM



- **Dual problem**

$$\min_{\hat{\boldsymbol{\alpha}}} \frac{1}{2}\hat{\boldsymbol{\alpha}}^T\left(\sum_{j=1}^{K-1}\hat{\mathbf{E}}_j(\mathbf{A}\mathbf{A}^T+\mathbf{e}\mathbf{e}^T)\hat{\mathbf{E}}_j^T\right)\hat{\boldsymbol{\alpha}}-\hat{\boldsymbol{\alpha}}^T\sum_{j=1}^{K-1}(\hat{\mathbf{F}}_j+\varepsilon\mathbf{e})$$

$$\text{s.t.} \quad \mathbf{0}\le\hat{\boldsymbol{\alpha}}\le C\mathbf{e}$$

- **Advantages of RSSVM**
- **The primal includes only a single optimization problem.**
- **The adapted loss function preserves equivalent sparseness of the original SVM in the RSSVM.**
- **Matched SMO-type solver can be developed for training.**

**Algorithm 2:** SMO-type solver for RSSVM

**Input:** Training set $S=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\dots,(\mathbf{x}_N,y_N)\}$

Parameter $C>0,\mu>0,\varepsilon>0,I_c,\eta>0$

Matrix $(\mathbf{H})_{N\times(K-1),N\times(K-1)}$

Vector $(\mathbf{B})_{N\times(K-1),1}$

**Output:** Vector $(\hat{\boldsymbol{\alpha}})_{N\times(K-1),1}$

Set iterative step $s$=0;

Initialize $\hat{\boldsymbol{\alpha}}=\mathbf{0}$;

**While** $\Gamma_1\cup\Gamma_2\neq\emptyset$

Compute $G_l(\hat{\boldsymbol{\alpha}}), l\in\{1,2,\dots N\times(K-1)\}$;

$s=s+1$;

Find the working variable $I=\arg_l\left(\max_{\hat{\alpha}_l\in\Gamma_1\cup\Gamma_2}|G_l(\hat{\boldsymbol{\alpha}})|\right)$

**If** $s-1$ and $0<\hat{\alpha}_l<C$

Select loop variables from previous working sets of preceding $\min(s-1,I_c)$ iterations;

**End**

**If** the loop variables are more than $2I_c$

Select $2I_c$ from them randomly;

**End**

Obtain the working set $U$ by combining the working variable and loop variables and removing the repeated ones.

**For** each $i\in U$

**If** $(\mathbf{H}_{i,:})\hat{\boldsymbol{\alpha}}-\mathbf{B}_i=0$

Remove $i$ from $U$;

**End**

**End**

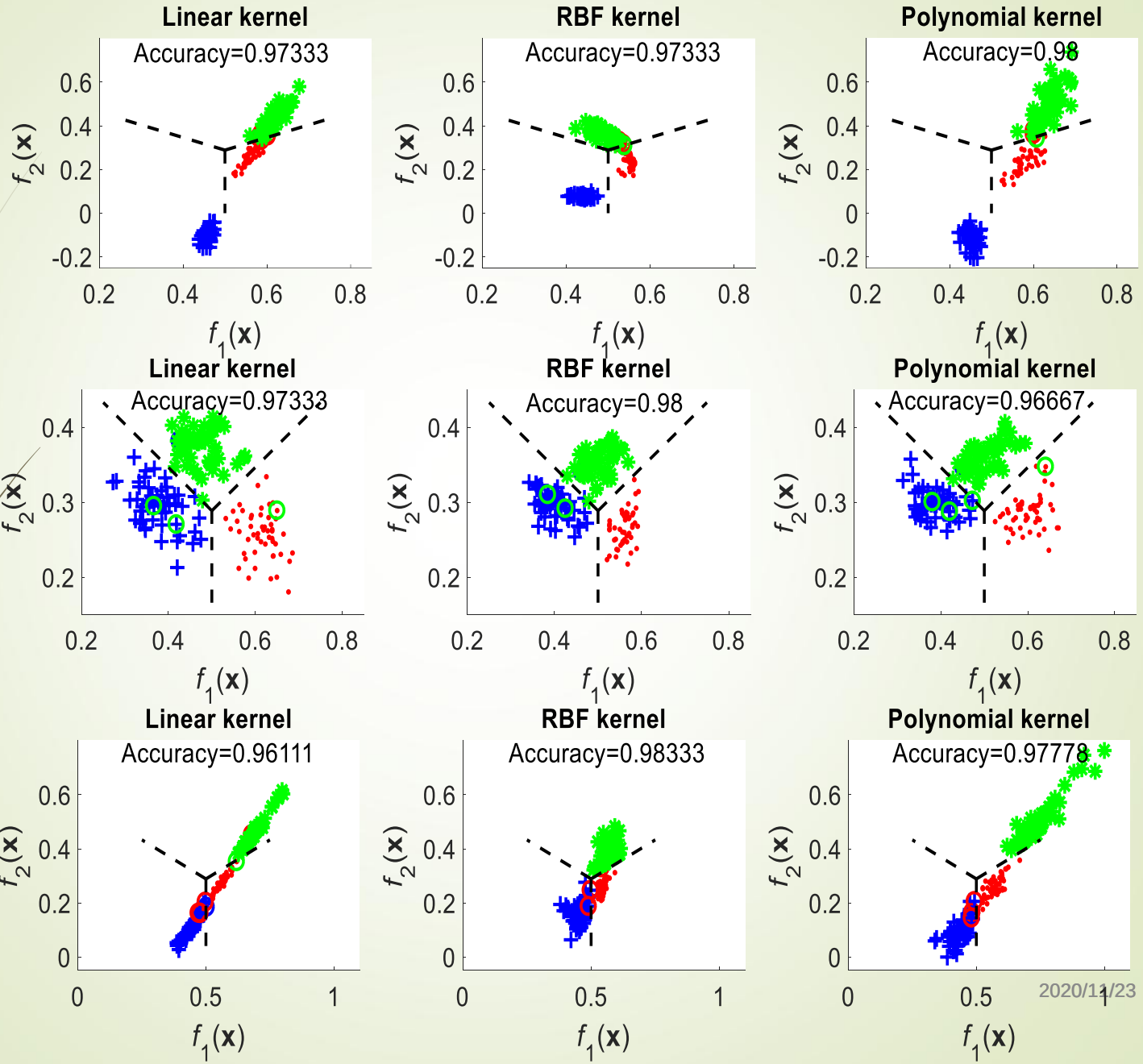
Obtain $\mathbf{t}_U^{\min}=(\mathbf{H}_{U,U})^{-1}(\mathbf{B}_U-(\mathbf{H}_{U,:})\hat{\boldsymbol{\alpha}})$;

Calculate scaled $\mathbf{t}_U^{\min}$ and coefficient $r=\min_{i\in U}\frac{t_i^{\min}}{t_i^{\min}}$;

Renew $\hat{\boldsymbol{\alpha}}_U=\hat{\boldsymbol{\alpha}}_U+r\mathbf{t}_U^{\min}$;

**End**

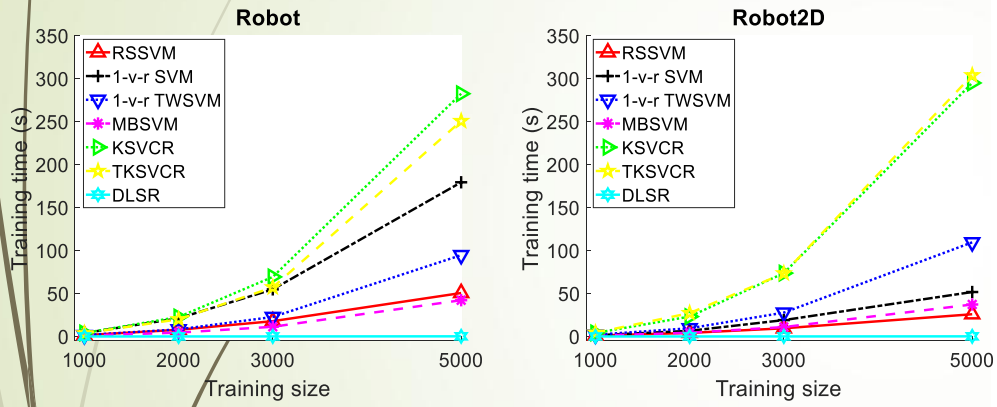
## Classifying mode



# RSSVM



**The developed SMO-type solver has excellent scalability.**




**Training speed test of large-scale data sets**

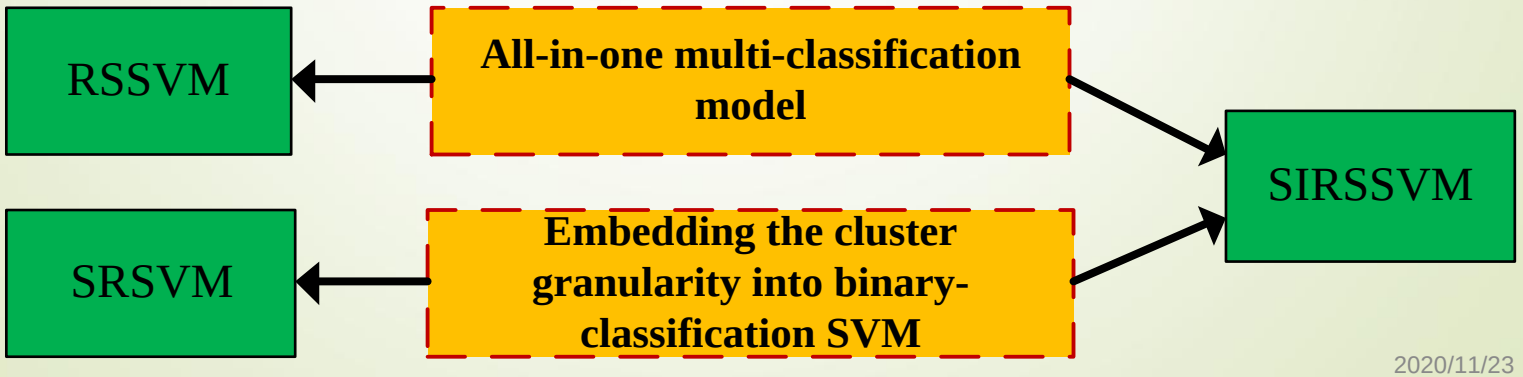
**Tang Long, Tian Yingjie, Pardalos P. M*. A novel perspective on multiclass classification: regular simplex support vector machine. *Information sciences*, 2019, 480: 324-338.**

## Accuracy test of UCI data sets

| Data Set, $(n \times N \times K)$ | Method | Accuracy (%) | Training Time (s) | PSV (%) | Test Time (s) | $C(c_1)$, $(\lambda)$ | $c_2$ | $\mu$ |
|---|---|---|---|---|---|---|---|---|
| Iris, (4×150×3) | RSSVM | **97.33±3.44** | 0.0257 | **23.48** | 0.0011 | 1 | - | $2^{-4}$ |
| | 1-v-r SVM | **97.33±4.66** | 0.0431 | 29.41 | 0.0017 | $2^{7}$ | - | $2^{-3}$ |
| | 1-v-r TWSVM | 96.67±3.51 | 0.0780 | 100 | 0.0056 | $2^{-5}$ | - | 1 |
| | MBSVM | 96.00±4.66 | 0.0386 | 100 | 0.0052 | $2^{-7}$ | - | 2 |
| | KSVCR | 96.00±4.66 | 0.0412 | 81.04 | 0.0032 | $2^{5}$ | $2^{3}$ | $2^{-2}$ |
| | TKSVCR | 94.67±6.13 | 0.0699 | 100 | 0.0060 | $2^{-1}$ | $2^{-7}$ | $2^{-1}$ |
| | DLSR | 92.00±7.57 | **0.0015** | 100 | **0.0002** | $10^{-4}$ | - | - |
| DUMDHTK, (5×180×3) | RSSVM | **96.11±3.75** | 0.0340 | **22.65** | 0.0014 | $2^{2}$ | - | $2^{-4}$ |
| | 1-v-r SVM | 93.89±6.65 | 0.0597 | 36.98 | 0.0019 | $2^{7}$ | - | $2^{-1}$ |
| | 1-v-r TWSVM | 95.56±4.38 | 0.0506 | 100 | 0.0071 | $2^{-7}$ | - | $2^{-1}$ |
| | MBSVM | 88.33±5.52 | 0.0481 | 100 | 0.0055 | $2^{-1}$ | - | 1 |
| | KSVCR | 94.44±5.24 | 0.0474 | 95.80 | 0.0045 | $2^{6}$ | 2 | $2^{-3}$ |
| | TKSVCR | 91.11±7.50 | 0.0737 | 100 | 0.0062 | $2^{-4}$ | $2^{-3}$ | $2^{-1}$ |
| | DLSR | 92.22±5.97 | **0.0012** | 100 | **0.0002** | $10^{-4}$ | - | - |
| ecoli, (7×150×3) | RSSVM | 94.67±5.26 | 0.0235 | **16.96** | 0.0008 | $2^{6}$ | - | $2^{-7}$ |
| | 1-v-r SVM | 95.33±4.50 | 0.0411 | 26.07 | 0.0013 | $2^{7}$ | - | $2^{-2}$ |
| | 1-v-r TWSVM | 92.67±8.58 | 0.0434 | 100 | 0.0043 | $2^{-7}$ | - | $2^{-1}$ |
| | MBSVM | 92.67±4.92 | 0.0457 | 100 | 0.0046 | $2^{-7}$ | - | $2^{2}$ |
| | KSVCR | **96.00±4.66** | 0.0363 | 89.04 | 0.0033 | $2^{2}$ | $2^{-2}$ | 1 |
| | TKSVCR | 87.33±10.2 | 0.0825 | 100 | 0.0045 | $2^{-1}$ | $2^{-4}$ | 1 |
| | DLSR | 95.33±4.50 | **0.0011** | 100 | **0.0001** | $10^{-1}$ | - | - |
| Robot, (24×600×4) | RSSVM | **87.17±4.58** | 0.4608 | **47.89** | 0.0259 | $2^{-3}$ | - | $2^{-2}$ |
| | 1-v-r SVM | 84.67±5.92 | 0.8794 | 61.94 | 0.0324 | $2^{5}$ | - | $2^{-3}$ |
| | 1-v-r TWSVM | 81.33±7.15 | 0.4428 | 100 | 0.0522 | $2^{-7}$ | - | $2^{-2}$ |
| | MBSVM | 82.50±4.92 | 0.3016 | 100 | 0.0522 | $2^{-7}$ | - | $2^{-2}$ |
| | KSVCR | 86.17±4.91 | 0.8664 | 100 | 0.0440 | $2^{4}$ | $2^{-7}$ | $2^{-2}$ |
| | TKSVCR | 83.83±5.03 | 0.9445 | 100 | 0.0522 | $2^{2}$ | $2^{-2}$ | $2^{-7}$ |
| | DLSR | 68.83±3.34 | **0.0048** | 100 | **0.0004** | $10^{-4}$ | - | - |
| Robot2D, (2×600×4) | RSSVM | 98.50±1.66 | 0.2596 | **5.24** | 0.0040 | $2^{7}$ | - | $2^{5}$ |
| | 1-v-r SVM | 96.17±3.05 | 0.3118 | 15.80 | 0.0095 | $2^{7}$ | - | $2^{7}$ |
| | 1-v-r TWSVM | 97.33±2.25 | 0.5054 | 100 | 0.0474 | $2^{-6}$ | - | $2^{7}$ |
| | MBSVM | 95.83±2.52 | 0.2897 | 100 | 0.0511 | $2^{-7}$ | - | $2^{7}$ |
| | KSVCR | 96.33±2.46 | 1.0604 | 72.52 | 0.0316 | $2^{7}$ | $2^{2}$ | $2^{7}$ |
| | TKSVCR | **99.00±1.17** | 1.0481 | 100 | 0.0455 | $2^{3}$ | $2^{-7}$ | $2^{2}$ |
| | DLSR | 79.67±6.02 | **0.0035** | 100 | **0.0003** | $10^{-4}$ | - | - |

# Structural improved RSSVM

- **Shortcomings of directly combining the partitioning (1-v-1, 1-v-r) strategies and RSSVM.**
- **Repeatedly computing the clustering information matrices under different partitions increases the training time.**
- **Individual classifier can hardly use complete information of training samples.**



# Structural improved RSSVM



**Algorithm 1:** Extracting structural information

**Input:** Kernel matrix $\mathbf{AA}^T$ (linear kernel) or $K(\mathbf{A},\mathbf{A}^T)$ (nonlinear kernel), where $\mathbf{A}=[\mathbf{x}_1,\mathbf{x}_2,\cdots\mathbf{x}_N]^T$ denotes inputs of the training samples.

**Output:** The overall covariance matrix $\boldsymbol{\Sigma}$

**For** $j=1:K$

Use $\widetilde{\mathbf{K}}^j_{N^{j,0}\times N^{j,0}}$ to denote the sub-matrix of $\mathbf{AA}^T$ ($K(\mathbf{A},\mathbf{A}^T)$) corresponding to the samples of class $j$, where $N^{j,0}$ is the number of the samples of class $j$;

Regard each sample as an individual cluster;

Calculate the initial matrix $\mathbf{W}^{j,0}$ of distance between each two clusters:

$W^{j,0}_{s,t}=\frac{\|\phi(\mathbf{x}_s)-\phi(\mathbf{x}_t)\|^2}{2}=\frac{\hat{K}^j_{s,s}+\hat{K}^j_{t,t}-2\hat{K}^j_{st}}{2}$;

Set $\mathbf{P}^{j0}=\mathbf{I}_{N^{j,0}\times N^{j,0}}$;

Set the iteration number $i=1$;

Use $N^{j,i}$ to denote the number of the clusters at the current iteration, namely $N^{j,i}=N^{j,0}$;

**While** $N^{j,i}>1$

$\mathbf{W}^{j,i}=\mathbf{W}^{j,i-1}$;

$\mathbf{P}^{j,i}=\mathbf{P}^{j,i-1}$;

Find the closet pair of clusters: find current minimum positive element of the matrix distance $\mathbf{W}^{j,i}$: $W^{j,i}_{s^i,t^i}$ $(s^i<t^i)$;

Merge clusters $s^i$ and $t^i$;

Renew the matrix $\mathbf{W}^{j,i}$ and $\mathbf{P}^{j,i}$:

$$1)\rightarrow W^{j,i}_{s^i,q}=\frac{\left(\sum_k P^{j,i}_{k,s^i}+\sum_k P^{j,i}_{k,q}\right)W^{j,i}_{s^i,q}+\left(\sum_k P^{j,i}_{k,t^i}+\sum_k P^{j,i}_{k,q}\right)W^{j,i}_{t^i,q}-\left(\sum_k P^{j,i}_{k,q}\right)W^{j,i}_{s^i,t^i}}{\sum_k P^{j,i}_{k,s^i}+\sum_k P^{j,i}_{k,t^i}+\sum_k P^{j,i}_{k,q}};$$

$$2)\rightarrow W^{j,i}_{t^i,q}=W^{j,i}_{s^i,t^i}=0;$$

$$3)\rightarrow \mathbf{P}^{j,i}_{:s^i}=\mathbf{P}^{j,i}_{:s^i}+\mathbf{P}^{j,i}_{:t^i};$$

$$4)\rightarrow \mathbf{P}^{j,i}_{:t^i}=\mathbf{P}^{j,i}_{:t^i}-\mathbf{P}^{j,i}_{:t^i}=\mathbf{0};$$

$N^{j,i+1}=N^{j,i}-1$;

$i=i+1$;

**End**

Find the optimal number of clusters among $\left(N^{j,1},W^{j,1}_{s^1,t^1}\right),\left(N^{j,2},W^{j,2}_{s^2,t^2}\right),\ldots\left(N^{j,i-1},W^{j,i-1}_{s^{i-1},t^{i-1}}\right)$:

set $N'^{j,k}=\frac{N^{j,k}-\min(N^{j,k})}{\max(N^{j,k})-\min(N^{j,k})},(1\le k\le i-1)$

$W'^{j,k}_{s^k,t^k}=\frac{W^{j,k}_{s^k,t^k}-\min\left(W^{j,k}_{s^k,t^k}\right)}{\max\left(W^{j,k}_{s^k,t^k}\right)-\min\left(W^{j,k}_{s^k,t^k}\right)},(1\le k\le i-1)$;

and we have $k^j_0=\arg\left(\min\left(\left(W'^{j,k}_{s^k,t^k}-\min\left(W'^{j,k}_{s^k,t^k}\right)\right)^2+\left(N'^{j,k}-\min(N'^{j,k})\right)^2\right)\right)$;

so the optimal number of clusters is $N^{j,k^j_0}$.

Construct $S^j=\left\{q:\mathbf{P}^{j,k^j_0}_{:q}\neq\mathbf{0}\right\}$, $|S^j|=N^{j,k^j_0}$;

Thus, for $\forall q\in S^j$, $S^j_q=\left\{\mathbf{x}_k:P^{j,k^j_0}_{k,q}\neq\mathbf{0}\right\}$ is a cluster of class $j$;

Use $C^i\left(i=1,2,\cdots N_t,N_t=\sum_{j=1}^K N^{j,k^j_0}\right)$ to denote the clusters without differentiating the classes, and use $\boldsymbol{\Sigma}^i$ to denote their covariance matrices;

$$\boldsymbol{\Sigma}=\sum_{i=1}^{N_t}\boldsymbol{\Sigma}^i=\sum_{i=1}^{N_t}\frac{1}{|C^i|}\sum_{k\in C^i}\left(\phi(\mathbf{x}_k)-\frac{1}{|C^i|}\sum_{l\in C^i}\phi(\mathbf{x}_l)\right)\left(\phi(\mathbf{x}_k)-\frac{1}{|C^i|}\sum_{l\in C^i}\phi(\mathbf{x}_l)\right)^T$$

**End**

Compute complete cluster information matrix

- **Primal problem**

$$\min_{\mathbf{w},\mathbf{b}}\sum_{j=1}^{K-1}\frac{1}{2}\left(\mathbf{w}_j^T\mathbf{w}_j+b_j^2\right)+d_1\sum_{i=1}^{N}(K-1)\xi_i+\sum_{j=1}^{K-1}\frac{d_2}{2}\mathbf{w}_j^T\boldsymbol{\Sigma}\mathbf{w}_j$$

$$\text{s.t. }\sum_{j=1}^{K-1}\left(2\left(V_{c_i,j}-V_{k,j}\right)\left(\mathbf{w}_j^T\mathbf{x}_i+b_j\right)+V_{k,j}^2-V_{c_i,j}^2\right)\ge\varepsilon-\xi_{i,k},(i=1,2,\cdots N)$$

$$\xi_{i,k}\ge 0,(i=1,2,\cdots N)$$

**Algorithm 2:** SMO-type algorithm for SIRSSVM

**Input:** Training set $S=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\ldots,(\mathbf{x}_N,y_N)\}$

Model parameter $d_1>0,d_2>0,\mu>0,\varepsilon>0$

Algorithm parameter $I_c,\eta>0$

Matrix $(\mathbf{H})_{N\times(K-1),N\times(K-1)}$

Vector $(\mathbf{B})_{N\times(K-1),1}$

**Output:** Vector $(\hat{\boldsymbol{\alpha}}^*)_{N\times(K-1),1}$

Improved SMO-type solver

Set the iteration counter $s=0$;

Initialize $\hat{\boldsymbol{\alpha}}=\mathbf{0}$;

**While** $\Gamma_1\cup\Gamma_2\neq\emptyset$

Compute $G_l(\hat{\boldsymbol{\alpha}}),l\in\{1,2,\ldots N\times(K-1)\}$;

$s=s+1$;

Find the current set of working variables $U^s_W=\{\hat{\alpha}_l:p(l_0)=p(l),\hat{\alpha}_l>0\}\cup\{\hat{\alpha}_{l_0}\}$;

where $\hat{\alpha}_{l_0}=\arg\left(\max_{\hat{\alpha}_l\in\Gamma_1\cup\Gamma_2}|G_l(\hat{\boldsymbol{\alpha}})|\right)$;

**If** $s-1$

**If** $\hat{\alpha}_{l_0}>0$ and $U^s_W\cap\left(U^{s-1}\cup U^{s-2},\cdots U^{s-\min(s-1,I_c)}\right)\neq\emptyset$

Determine the current set of all qualified loop variables:

$U^s_{l0}=\left\{\hat{\alpha}_l|\hat{\alpha}_l\in U^{s-1}\cup U^{s-2},\cdots U^{s-\min(s-1,I_c)},\hat{\alpha}_l>0,\mathbf{\theta}_{p(l),:}\hat{\boldsymbol{\alpha}}<d_1(K-1)\right\}$;

**End**

**If** $|U^s_{l0}|>2I_c$

Randomly generate a subset $U^s_l$ of $U^s_{l0}$, satisfying $|U^s_l|=2I_c$;

**End**

**End**

Obtain the current working set $U^s=U^s_W\cup U^s_l$;

**For** each $i\in U^s$

**If** $\left(\mathbf{H}_{i,:}\right)\hat{\boldsymbol{\alpha}}-\mathbf{B}_i=0$

Remove $i$ from $U^s$;

**End**

**End**

We have $\mathbf{t}^{\min}_{U^s}=\left(\mathbf{H}_{U^s,U^s}+\sigma\mathbf{I}\right)^{-1}\left(\mathbf{B}_{U^s}-\left(\mathbf{H}_{U^s,:}\right)\hat{\boldsymbol{\alpha}}\right)$;

Calculate scaling coefficients $\eta_1=\min\left(\eta,\frac{\left(C_1(K-1)-\mathbf{\theta}_{p(l),U^s}\hat{\boldsymbol{\alpha}}_{U^s}\right)}{\mathbf{\theta}_{p(l),U^s}\mathbf{t}_{U^s}}\right)$ and $r_0=\min(\eta_1)$;

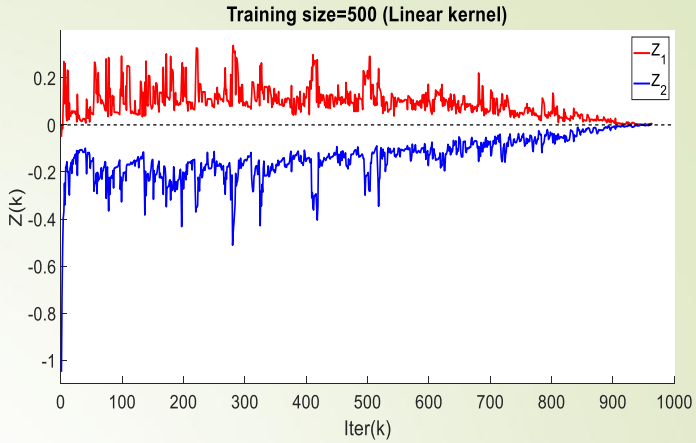
Renew $\hat{\alpha}_{U^s}=\hat{\alpha}_{U^s}+r_0\mathbf{t}^{\min}_{U^s}$;

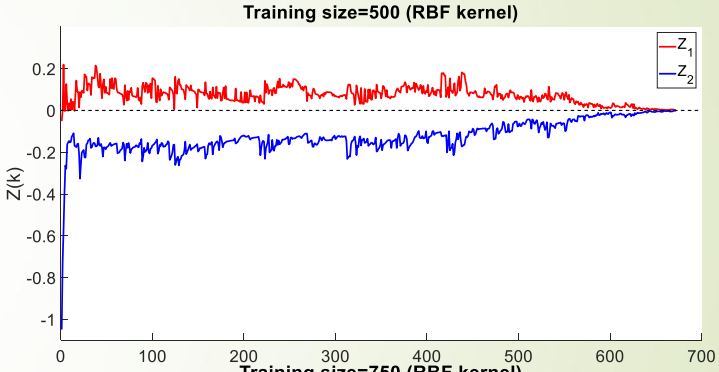
**End**

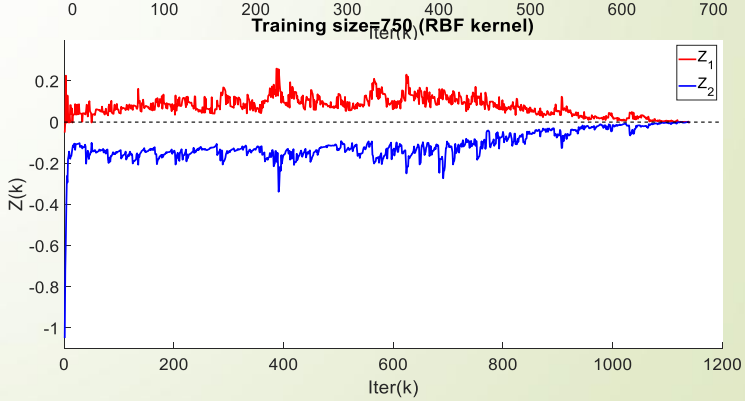
Obtain $\hat{\boldsymbol{\alpha}}^*=\hat{\boldsymbol{\alpha}}$

# Accuracy test

| Data set. | RSSVM | IRSSVM | KSVCR | SIRSSVM | 1-v-r SNPSVM | 1-v-r SRSVM |
|---|---|---|---|---|---|---|
| *Digits*-mfeat-fou_1-5 | 90.00±3.16 | **90.40±3.58** | **90.40±5.18** | **90.40±3.58** | **90.40±3.58** | 84.80±4.60 |
| *Digits*-mfeat-fou_6-10 | 80.80±5.22 | 82.00±2.83 | **82.80±3.03** | **82.80±4.60** | 80.40±2.61 | 77.20±3.35 |
| *Digits*-mfeat-fac_1-5 | 99.60±0.89 | 99.60±0.89 | 98.80±1.10 | 99.60±0.89 | **100.00±0.00** | 99.20±1.10 |
| *Digits*-mfeat-fac_6-10 | 98.00±2.00 | 98.00±2.00 | 97.60±2.61 | **98.40±2.61** | **98.40±2.19** | 98.00±2.00 |
| *Digits*-mfeat-kar_1-5 | 98.00±1.41 | 98.00±1.41 | 98.00±1.41 | **98.40±0.89** | **98.40±0.89** | 95.20±2.28 |
| *Digits*-mfeat-kar_6-10 | 96.40±2.61 | 96.40±3.29 | **97.20±3.03** | 96.80±3.90 | 96.80±4.15 | 96.00±2.83 |
| *Digits*-mfeat-pix_1-5 | 98.00±2.45 | 98.00±2.45 | 97.20±3.03 | 98.40±1.67 | **98.80±1.79** | 95.60±1.67 |
| *Digits*-mfeat-pix_6-10 | 96.40±1.67 | 96.80±2.68 | 96.80±2.68 | 97.20±1.79 | **97.60±1.67** | 96.40±2.61 |
| *Digits*-mfeat-zer_1-5 | 96.80±2.28 | 96.40±1.67 | 96.80±2.28 | **97.20±2.28** | 96.80±2.68 | 95.20±1.79 |
| *Digits*-mfeat-zer_6-10 | **77.60±6.07** | 76.40±5.73 | 76.40±3.85 | **77.60±6.54** | 76.80±4.60 | 74.40±5.18 |
| *Digits*-mfeat-mor_1-5 | 86.80±3.63 | 86.80±4.38 | 84.00±4.69 | **87.60±3.29** | 84.80±5.02 | 80.80±2.28 |
| *Digits*-mfeat-mor_6-10 | 78.80±4.82 | 80.00±3.46 | 78.00±2.00 | **80.80±3.03** | **80.80±3.90** | 78.40±2.61 |
| *Corel*-class_1-3 | 77.00±7.85 | 78.33±2.04 | 77.00±3.21 | **79.67±3.42** | 78.67±4.62 | 74.33±4.94 |
| *Corel*-class_4-6 | 75.00±5.65 | 75.00±5.65 | 73.33±5.14 | 75.00±5.65 | **76.00±4.50** | 73.00±6.28 |
| *Corel*-class_7-9 | 78.33±5.27 | 77.33±8.13 | **80.33±6.28** | 78.67±6.39 | 79.33±7.96 | 76.67±7.73 |
| *Corel*-class_10-12 | 72.33±3.65 | 72.33±5.48 | 73.67±3.21 | 73.33±3.12 | 73.33±3.12 | **74.33±3.25** |
| *Corel*-class_13-15 | 69.67±7.40 | 69.67±3.80 | 68.67±7.67 | **70.00±5.40** | 69.67±4.77 | **70.00±4.41** |
| *Corel*-class_16-18 | 74.67±11.93 | 73.67±9.60 | 73.67±9.89 | 75.33±10.17 | **76.00±8.05** | 71.67±7.55 |
| *Corel*-class_19-21 | 88.00±6.39 | 88.00±4.62 | **88.67±4.62** | 88.33±4.71 | 88.00±6.60 | 86.33±3.61 |
| *Corel*-class_22-24 | **69.33±7.42** | 67.00±8.93 | 69.00±6.41 | **69.33±7.32** | 68.33±8.16 | 66.67±3.54 |
| *Corel*-class_25-27 | 77.00±2.17 | 77.00±5.06 | 75.67±5.35 | **78.00±6.17** | 76.67±5.14 | 76.00±1.90 |
| *Corel*-class_28-30 | 94.00±5.35 | 94.00±5.35 | 94.00±6.73 | 94.00±5.35 | **95.00±5.53** | 94.00±4.50 |
| *Corel*-class_31-33 | 92.67±1.90 | 92.67±1.90 | 92.33±3.03 | **93.33±1.18** | 92.00±2.47 | 92.33±2.24 |
| *Corel*-class_34-36 | 78.00±4.15 | 76.67±3.12 | 77.67±3.25 | **78.67±5.19** | **78.67±5.45** | 75.33±2.98 |
| *Corel*-class_37-39 | 86.00±7.32 | 86.00±3.46 | 85.67±4.94 | **86.67±6.35** | 86.00±4.50 | 85.00±3.54 |
| *Corel*-class_40-42 | 79.67±3.21 | 80.00±3.33 | 78.67±9.01 | **80.33±2.98** | 79.67±4.62 | 76.00±6.30 |
| *Corel*-class_43-45 | 79.67±12.55 | 80.67±12.05 | 79.67±12.99 | **81.00±11.70** | 79.00±9.90 | 80.00±12.30 |
| *Corel*-class_46-48 | 91.33±2.74 | 91.00±2.53 | 90.67±2.53 | 91.67±3.54 | **92.33±3.25** | 90.33±4.47 |
| *Corel*-class_49-51 | **72.00±3.61** | **72.00±3.61** | 71.00±2.24 | **72.00±3.61** | 71.33±1.39 | 69.67±3.98 |
| *Corel*-class_52-54 | 80.33±6.17 | 80.33±6.17 | 79.33±5.48 | 80.33±6.60 | **81.33±6.71** | 78.67±6.50 |
| *Corel*-class_55-57 | 87.67±9.17 | 88.67±7.01 | 87.00±8.28 | **89.33±7.69** | 89.00±9.90 | 87.33±10.04 |
| *Corel*-class_58-60 | 68.33±9.86 | 68.67±10.17 | 66.00±10.38 | **69.00±10.18** | 68.67±12.88 | 65.00±8.25 |
| *Corel*-class_61-63 | 97.00±1.83 | 97.00±2.17 | 97.33±1.49 | **98.33±1.18** | **98.33±1.18** | 98.00±2.17 |
| *Corel*-class_64-66 | 86.00±6.52 | 87.00±5.58 | 84.33±4.50 | **89.00±3.46** | 88.33±3.54 | 85.33±5.45 |
| *Corel*-class_67-69 | 64.00±6.52 | 65.00±5.89 | 63.33±3.54 | **65.67±8.55** | 65.00±7.73 | 63.33±5.89 |
| *Corel*-class_70-72 | 89.33±3.03 | **90.00±3.91** | 89.33±1.90 | **90.00±3.91** | 89.67±2.98 | 88.67±3.42 |
| *Corel*-class_73-75 | 75.00±7.73 | 76.33±9.75 | 73.33±8.98 | 76.33±9.82 | **76.67±8.74** | 71.67±11.79 |
| *Corel*-class_76-78 | 93.67±3.21 | 93.67±3.21 | **94.00±3.84** | 93.67±2.98 | **94.00±3.03** | 93.67±3.61 |

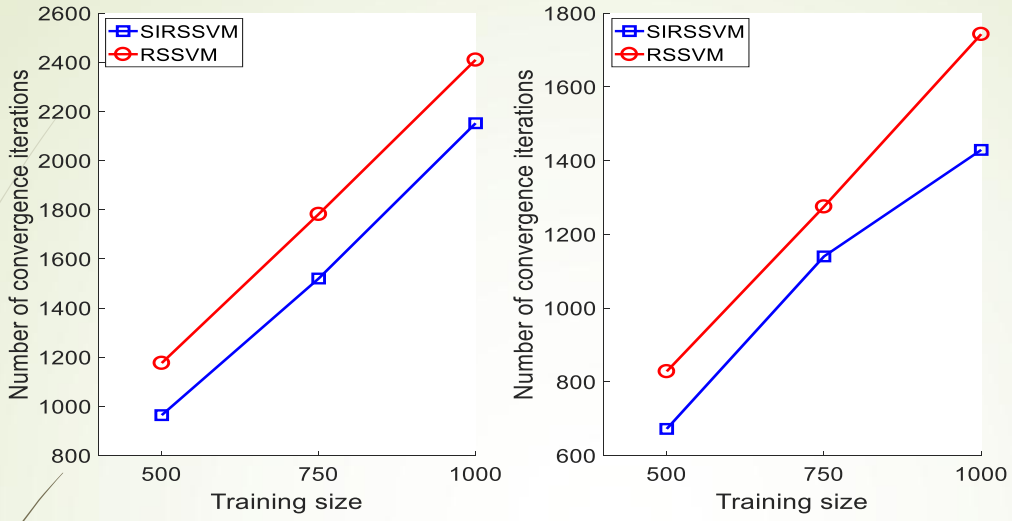


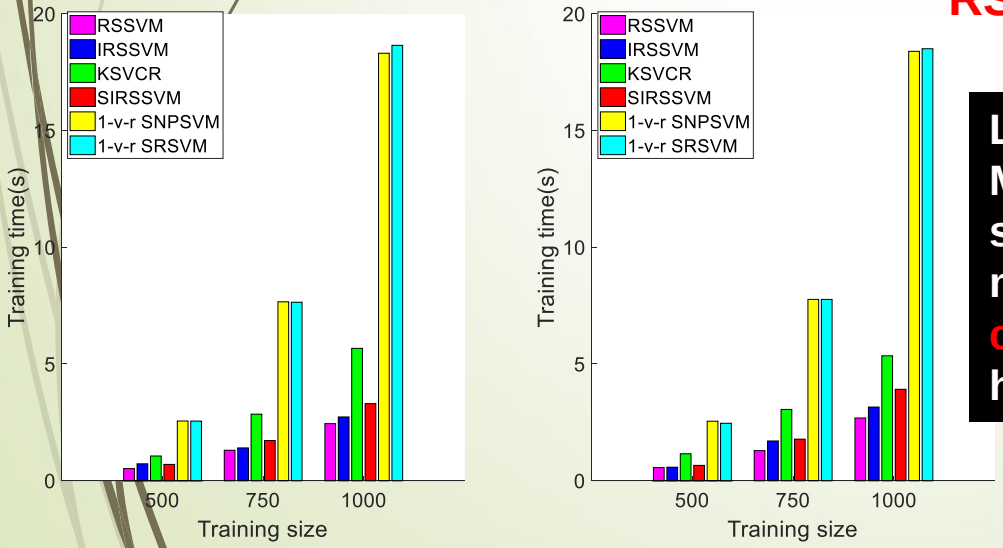




## Convergence process

# Structural improved RSSVM




**Comparison of training speed**




**SIRSSVM has better convergence than RSSVM.**

**Long Tang; Yingjie Tian; Wenjun Li; Panos M. Pardalos*; Structural improved regular simplex support vector machine for multiclass classification, Applied soft computing, 2020, 91, https://doi.org/10.1016/j.asoc.2020.106235.**

# Challenging issues with SVM

- Unbalanced data
- Structural data sets
- Multi-label classification
- Semi-supervised learning
- Massive data sets

Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, Asdrubal Lopez, **A comprehensive survey on support vector machine classification: Applications, challenges and trends**, Neurocomputing, Volume 408, 2020, Pages 189-21
https://www.sciencedirect.com/science/article/pii/S0925231220307153

# Thank you!