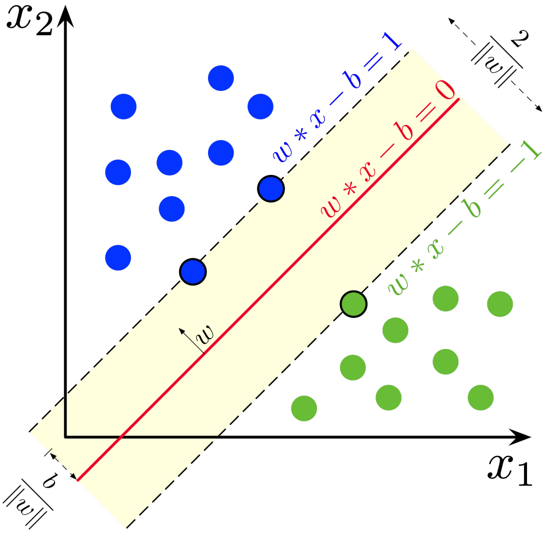
Nonparametric learning with matrix-valued predictors in high dimensions

Chanwoo Lee, Miaoyan Wang

Summary of research work

May 26, 2020

A successful story: Support vector machine (SVM)



Maximum-margin hyperplane for an SVM trained with 2-d vector predictors
(Picture source: Wiki)

SVM methods are powerful, however...

- How to efficiently classify high-dimensional matrices with limited sample size:

$$n \ll d_1 d_2 = \text{dimension of feature space ?}$$

- How to robustly predict the label probability when little is known about the function:

$$\mathbb{P}(\text{Label} = 1|\text{Matrix}) \overset{\text{def}}{=} f(\text{Matrix}) \text{ ?}$$

- How to effectively reduce the sufficient dimensions of the feature space without information lost:

$$\text{Label} \perp\!\!\!\perp \text{Matrix}|\phi(\text{Matrix}) \text{ ?}$$

- For the purpose of presentation, we focus on matrix predictors and binary outcomes.

Previous methods for nonparametric multivariate learning

Method	SDR*	Robust prediction	Binary outcome	Tensor predictor
Weighted large-margin (Wang et al, JCGS'19) (Zhang et al, JASA'10)	✗	✓	✓	✗
Deep Neural Network (...)	✗	✓	✓	✗
Principal SVM (Li et al, AOS'11) (Shin et al, Biometrika'17)	✓	✗	✗	✗
Nonparametric regression (Imaizumi et al ICML'16)	✗	✓	✗	✓
Our method	✓	✓	✓	✓

*SDR: sufficient dimension reduction. See later slides for definition.

Overview

1. Motivation and Problem
2. Linear Learning with low-rank kernels
    - Classification
    - Probability function estimation
    - Sufficient dimension reduction
    - Simulations
3. Nonlinear learning with low-rank kernels
    - Nonlinear matrix-valued kernels
    - Nonparametric learning with matrix predictors
    - Simulations

Problem

Let $\{(\boldsymbol{X}_i, y_i) \in \mathbb{R}^{d_1 \times d_2} \times \{-1, 1\} \colon i = 1, \dots, n\}$ denote an i.i.d. sample from distribution $\mathcal{X} \times \mathcal{Y}$.

- (Probability estimation.) Estimate the conditional probability function

$$\mathbb{P}(y^{\text{new}} = 1 | \boldsymbol{X}^{\text{new}}) = g(\boldsymbol{X}^{\text{new}}),$$

  where $f \colon \mathbb{R}^{d_1 \times d_2} \mapsto [0, 1]$ is the regression function of interest.

- (Sufficient dimension reduction, SDR.) Find a transformed predictor $\phi(\boldsymbol{X})$ such that

$$y \perp\!\!\!\perp \boldsymbol{X} | \phi(\boldsymbol{X}),$$

  where $\phi \colon \mathbb{R}^{d_1 \times d_2} \mapsto \mathbb{R}^{r_1 \times r_2}$ is the dimension reduction of interest.

- Probability estimation concerns conditional mean, whereas SDR concerns conditional independence (stronger than mean).

Key ingredient: low-rank support matrix machine (SMM)

- Classification is an easier task than probability estimation.
- First, we develop a large-margin classifier for matrix predictors in high dimensions.
- Consider the classifier, sign$\{\hat{f}(\boldsymbol{X})\}$, where $\hat{f}(\cdot)\colon \mathbb{R}^{d_1\times d_2} \mapsto \mathbb{R}$ is the solution to the following optimization over a function class $\mathcal{F}$:

$$\hat{f} = \operatorname*{argmin}_{f\in\mathcal{F}} \left\{ \sum_i [y_i f(\boldsymbol{X}_i)]_+ + \lambda \|f\|_F^2 \right\}. \tag{1}$$

  Here* $\mathcal{F} = \{f\colon f(\cdot) = \langle \cdot,\ \boldsymbol{B}\rangle$ where $\text{rank}(\boldsymbol{B}) \le r\}$ and $\|f\|_F^2 \overset{\text{def}}{=} \|\boldsymbol{B}\|_F^2$.
- Assume r is fixed and known (for now). The classifier (1) is referred to as low-rank support matrix machine (SMM).

*For presentation convenience, we omit the intercept b in the representation of f.

Key ingredient: low-rank SMM

- The proposed low-rank SMM finds the most separable hyperplane among all possible r-dimensional representations.
- The reason is that the low-rank SMM

$$\min_{\operatorname{rank}(\boldsymbol{B})\leq r} \left\{ \sum_i \left[y_i \left\langle \underbrace{\boldsymbol{X}_i}_{\text{ambient dimension } d_1 \times d_2}, \boldsymbol{B} \right\rangle \right]_+ + \lambda \|\boldsymbol{B}\|_F^2 \right\}.$$

is equivalent to

$$\min_{\substack{\boldsymbol{P}_r\boldsymbol{P}_r^T = \boldsymbol{P}_c\boldsymbol{P}_c^T = \boldsymbol{I}_r \\ \boldsymbol{C}\in\mathbb{R}^{r\times r}}} \left\{ \sum_i \left[y_i \left\langle \underbrace{\boldsymbol{P}_r\boldsymbol{X}_i\boldsymbol{P}_c^T}_{\text{intrinsic dimension } r \times r}, \boldsymbol{C} \right\rangle \right]_+ + \lambda \|\boldsymbol{C}\|_F^2 \right\},$$

where $\boldsymbol{P}_r \in \mathbb{R}^{r\times d_1}, \boldsymbol{P}_c \in \mathbb{R}^{r\times d_2}$ are row-wise and column-wise projection matrices, respectively.

Key ingredient: low-rank SMM

- The solution to the low-rank SMM is represented as

$$\hat{f}(\boldsymbol{X}) = \sum_i \hat{\alpha}_i y_i \langle \hat{\boldsymbol{P}}_r \boldsymbol{X}_i,\ \hat{\boldsymbol{P}}_r \boldsymbol{X} \rangle,$$

  where $\{\hat{\alpha}_i\}$ are estimated coefficients in the dual problem and $\hat{\boldsymbol{P}}_r$ is the estimated projection matrix.
- We develop an alternating minimization algorithm to jointly estimate $\boldsymbol{P}_r$ and $\{\alpha_i\}$. (The projection $\boldsymbol{P}_c$ is absorbed into $\{\alpha_i\}$.)
- We are particularly interested in the high-dimensional region when both n and $\min\{d_1, d_2\} \to \infty$ while $r = \mathcal{O}(1)$. *(An illustration of failures for classical SVMs in high dimensions; JMLR 18(45), 1-21, 2017).*
- The low-rankness in the model efficiently prevents overfitting in high dimensions.

Theory: Low-rank SMM in high dimensions

- Let $\langle \cdot,\ \cdot \rangle_P$ denote the rank-r kernel for matrix predictors:

$$\langle \boldsymbol{X}, \boldsymbol{Y} \rangle_{\boldsymbol{P}} \overset{\text{def}}{=} \langle \boldsymbol{P}\boldsymbol{X},\ \boldsymbol{P}\boldsymbol{Y} \rangle, \quad \text{for all } \boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{d_1 \times d_2},$$

  where $\boldsymbol{P} \in \mathbb{R}^{r \times d_1}$ is a (unknown) rank-r projection matrix.
- The low-rank SMM considers the decision function of the form:

$$f(\cdot) = \sum_{i=1}^{n} \alpha_i y_i \langle \boldsymbol{X}_i,\ \cdot \rangle_{\boldsymbol{P}}.$$

- The function $f \in \mathcal{F}_r$, where $\mathcal{F}_r$ is the reproducing kernel Hilbert Space (RKHS) induced by rank-r kernels $\{\langle \cdot,\ \cdot \rangle_{\boldsymbol{P}} \colon \text{projection } \boldsymbol{P} \in \mathbb{R}^{r \times d_1}\}$.
- Column-wise low-rank kernel can be similarly introduced $\Rightarrow$ same decision function. *Is this a coincidence or something fundamental?*

Theory: Low-rank SMM in high dimensions

Generalization bound (Lee and Wang, 2020+)

With probability at least $1-\delta$, the generalization error of low-rank SMM is

$$\mathbb{P}\{Y^{\text{new}} \neq \text{sign}[\hat{f}(\boldsymbol{X}^{\text{new}})]\} \leq \text{training error} + \mathbb{E}[\hat{R}_n(\mathcal{F}_r)] + \sqrt{\frac{\ln(\frac{1}{\delta})}{2n}},$$

where $\hat{R}_n(\mathcal{F}_r)$ denotes the Rademacher complexity of rank-r SMM classifiers. Roughly, in the case $d_1 \asymp d_2 = \mathcal{O}(d)$, we have

$$\mathbb{E}[\hat{R}_n(\mathcal{F}_r)] \leq 4\varepsilon + \frac{c_1 r}{\sqrt{n}} \int_{\varepsilon}^{\infty} \log^{1/2}\left(\frac{d/r}{c_2\varepsilon'}\right) d\varepsilon',$$

where the bound increases with d and r. Results highlights the role of r in preventing overfitting.

1. Check the proof more carefully; 2. consistency for relative risk.

See Varshney, K.R. and Willsky, A.S., IEEE Trans. Signal Process., 59(6), 2496-2512, 2011.

From classification to regression

Back to the probability estimation problem.

- Consider a piecewise-constant representation of the target probability function $g(\boldsymbol{X}) \overset{\text{def}}{=} \mathbb{P}(Y = 1|\boldsymbol{X})$:

$$g(\boldsymbol{X}) \approx \frac{1}{H} \sum_{h \in [H]} \mathbb{1}\left\{\boldsymbol{X} : g(\boldsymbol{X}) \leq \frac{h}{H}\right\},$$

  where $H \in \mathbb{N}_+ \to \infty$ is the smoothing parameter.

- The classification problem has provided candidate decision regions:

$$\mathbb{1}\left\{\boldsymbol{X} : \underbrace{\operatorname{sign}\left[\hat{f}_h(\boldsymbol{X})\right] = -1}_{\text{decision region from classification}}\right\} \xrightarrow{\text{in } p} \mathbb{1}\left\{\boldsymbol{X} : \underbrace{\mathbb{P}(Y = 1|\boldsymbol{X}) \leq \frac{h}{H}}_{\text{target sublevel set}}\right\},$$

  for any $h = 1, \ldots, H$.

- This suggests a non-parametric approach to estimating $g(\boldsymbol{X})$.

Algorithm

We develop the following algorithm to solve for the target function:

- Step 1. Choose a sequence of weights $\pi_h = \frac{h}{M}$, for $h = 1, \ldots, H$.
- Step 2. For each weight $\pi_h \in [0,1]$, solve the following weighted low-rank support matrix machine (SMM):

$$\hat{\boldsymbol{B}}_h = \underset{\text{rank}(\boldsymbol{B}) \leq r}{\text{argmin}} \left\{ \ell_h(\boldsymbol{B}) + \lambda \|\boldsymbol{B}\|_F^2 \right\}, \quad \text{where} \tag{2}$$

$$\ell_h(\boldsymbol{B}) = (1 - \pi_h) \sum_{y_i = 1} \left[1 - \langle \boldsymbol{X}_i, \boldsymbol{B} \rangle\right]_+ + \pi_h \sum_{y_i = -1} \left[1 + \langle \boldsymbol{X}_i, \boldsymbol{B} \rangle\right]_+ .$$

Algorithm (cont.)

- Step 3. Denote the sequence of solutions and decision regions

$$\hat{f}_h(\cdot) \overset{\text{def}}{=} \langle \cdot,\ \hat{\boldsymbol{B}}_h \rangle \quad \text{and} \quad \hat{\mathcal{D}}_h = \{\boldsymbol{X} \colon \text{sign}[\hat{f}_h(\boldsymbol{X})] = -1\},$$

  for all $h = 1, \ldots, H$.
- Step 4. Estimate the target probability function by

$$\hat{g}(\boldsymbol{X}) = \frac{1}{M} \sum_{h \in [H]} \mathbb{1}\left\{\boldsymbol{X} \in \hat{\mathcal{D}}_h\right\}. \tag{3}$$

The estimator (3) *is asymptotically equivalent to the original estimator in Wang et al [Biometrika'08]. I choose this form because of its good analytic properties.*

Theory: Probability prediction via low-rank kernels

High-dimensional consistency (Lee and Wang, 2020+)

Assume the true $g \in \mathcal{F}$, where $\mathcal{F}$ is the RKHS induced by $\langle \cdot, \ \cdot \rangle_{\text{rank-r}}$.

- Given any $\pi \in [0,1]$, the solution to (2) yields the Bayes rule:

$$\text{sign}[\hat{f}_\pi(\boldsymbol{X})] \xrightarrow{\text{in } p} \text{sign}[g(\boldsymbol{X}) - \pi], \text{ as } n, d \to \infty \text{ while } d/n \to 0.$$

- Our probability estimator (3) is consistent:

$$\hat{g}(\boldsymbol{X}) \xrightarrow{\text{in } p} g(\boldsymbol{X}), \quad \text{as } H, n, d \to \infty \text{ while } d/n \to 0.$$

- To the best of our knowledge, this is the first result for SVM-based prob. estimation in large dimension (d^2), large sample size (n) regime.
- *Detailed proofs, assumptions? Convergence in terms of smoothness of g, r, d, n, and H? Do we really need the assumption, $g \in \mathcal{F}$; i.e., $g(\cdot)$ linear in $\boldsymbol{X}$? Perhaps not... composition of monotonic + linear functions is also fine.. indictor functions are dense in $L[0,1]^2$...*

SDR via low-rank kernels

- A challenging problem is to identify the sufficient features with minimal modeling assumption in the prediction model.
- We develop a robust sufficient dimension reduction (SDR) method for matrix predictors in high dimensions.

Add two steps to Algorithm:

1. Pre-step 2 (Whitening): $\boldsymbol{X}_i \leftarrow \hat{\Sigma}_r^{-1/2} [\boldsymbol{X}_i - \text{Mean}(\boldsymbol{X})] \hat{\Sigma}_r^{-1/2}$, where $\hat{\Sigma}_r, \hat{\Sigma}_c$ are empirical row- and column-wise covariance matrices.
2. Post-step 4 (Assembling): Arrange outputs $\hat{\boldsymbol{B}}_h$ into an order-3 tensor:

$$\mathcal{B}(:\,,:\,,h) = \hat{\Sigma}_r^{1/2} \hat{\boldsymbol{B}}_h \hat{\Sigma}_r^{1/2}, \quad \text{for } h = 1, \ldots, (H-1).$$

Perform a rank-$(r_1, r_2, r_1 r_2)$ Tucker decomposition on $\mathcal{B}$. Let $\hat{\boldsymbol{P}}_c, \hat{\boldsymbol{P}}_r$ denote the estimated factor matrices at the first two modes.

Theory: SDR via low-rank kernels

Bilinear sufficient dimension reduction (SDR)

Let $(\boldsymbol{X}, Y) \in \mathbb{R}^{d_1 \times d_2} \times \{0,1\}$ be the pair of r.v.'s of interest. Bilinear SDR seeks a pair of matrices $\boldsymbol{P}_r \in \mathbb{R}^{r_1 \times d_1}, \boldsymbol{P}_c \in \mathbb{R}^{r_2 \times d_2}$ such that

$$Y \perp\!\!\!\perp \boldsymbol{X} \Big| \underbrace{\boldsymbol{X} \times_1 \boldsymbol{P}_r \times_2 \boldsymbol{P}_c}_{r_1\text{-by-}r_2}.$$

The minimum dimension of (r_1, r_2) is called the structure dimension.

Unbiasedness (Lee and Wang, 2020+)

Under suitable assumptions*, our proposed $\hat{\boldsymbol{P}}_r$ and $\hat{\boldsymbol{P}}_c$ are asymptotically unbiased estimators for $\boldsymbol{P}_r$ and $\boldsymbol{P}_c$, respectively; i.e.

$$\Theta(\hat{\boldsymbol{P}}_r, \boldsymbol{P}_r) \to 0, \quad \text{and } \Theta(\hat{\boldsymbol{P}}_c, \boldsymbol{P}_c) \to 0, \quad \text{as } n \to \infty.$$

Assume: (1) $\mathbb{E}(\boldsymbol{X}|\boldsymbol{X} \times_1 \boldsymbol{P}_r \times_2 \boldsymbol{P}_c)$ is a linear function of $\boldsymbol{X} \times_1 \boldsymbol{P}_r \times_2 \boldsymbol{P}_c$; (2) $\mathrm{Cov}(\mathrm{vec}(\boldsymbol{X}))=\Sigma_r \otimes \Sigma_c$.

Simulation: Classification

- Linearly separable model: $d_1 = 10, d_2 = 8, r = 5$.

  Error $= \|\boldsymbol{B} - \hat{\boldsymbol{B}}\|_F$, where both $\boldsymbol{B}, \hat{\boldsymbol{B}}$ are normalized to have F-norm 1.

	N = 100	N = 200	N = 300	N = 400
Error (SMM)	0.4827625	0.1903077	0.1020411	0.0663940
Error (SVM)	0.5405979	0.3149391	0.1361625	0.0734705

Table 1: The columns are the number of sample size and rows are the Frobenius norm error according to the two methods.

- Linearly separable model: $d_1 = 5, d_2 = 4, r = 3,\ N = 100$

	1st	2nd	3rd	4th	5th	average
SVM	0.90	0.95	0.9	0.75	0.9	0.88
SMM	0.95	0.95	1.0	0.75	0.9	0.91

Table 1: Miss classification rate in Simulation 1

- Linearly non-separable model: $d_1 = 5, d_2 = 4, r = 2,\ N = 100$

	1st	2nd	3rd	4th	5th	average
SVM	0.55	0.65	0.75	0.6	0.75	0.66
SMM	0.50	0.80	0.75	0.7	0.75	0.70

Table 3: Miss classification rate in Simulation 2

Details in "041720.pdf". Table caption: Miss classification $\Rightarrow$ 1-MCR?

Simulation: Probability function estimation

Ground truth $g(\boldsymbol{X}) = \mathbb{P}(y = 1|\boldsymbol{X})$: nonlinear in $\boldsymbol{X}$, but can be written as composition of monotonic + linear functions.

$$\boldsymbol{X}|y = -1 \sim_{i.i.d.} \text{MVN}(\boldsymbol{0}, \boldsymbol{I}_4),\ \boldsymbol{X}|y = 1 \sim_{i.i.d.} \text{MVN}(\boldsymbol{M}, \boldsymbol{I}_4),\ \text{rank}(\boldsymbol{M}) = 1$$

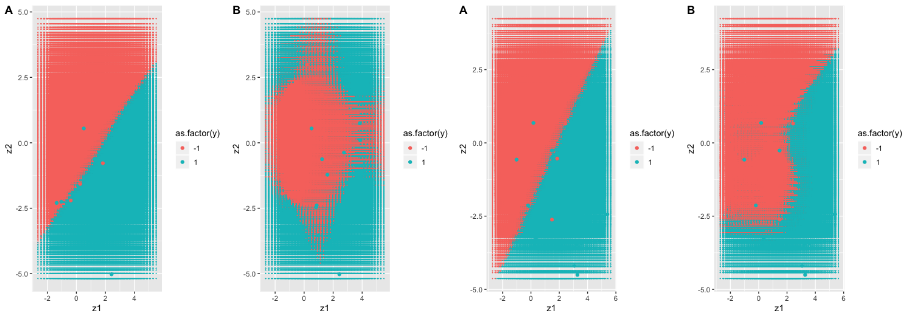


Figure A shows the classification rule of SMM with linear kernel and Figure B shows SMM with polynomial kernel.

Details in “042320.pdf”.

Simulation: Probability function estimation

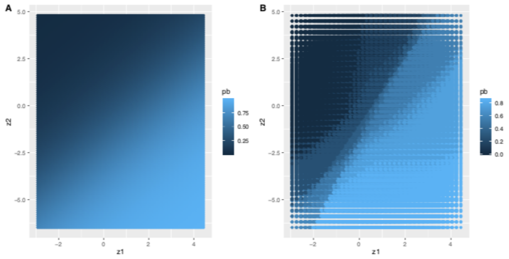


Figure 2: Figure A shows the true conditional probability of y given (Z_1, Z_2). Figure B shows the estimated probability using SMM weighted hinge loss approach.

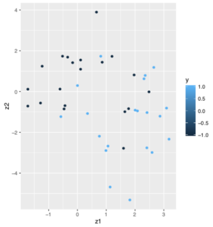


Figure 1: Visualization of the feature matrices. $Z1$ is the sum of the first column and $Z1$ is of the second one.

Details in "042020.pdf".

Nonlinear kernels for matrix predictors

- Let $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{d_1 \times d_2}$ be a pair of matrices. Given a projection matrix $\boldsymbol{P} \in \mathbb{R}^{r \times d_1}$, a linear low-rank kernel is defined as

$$\begin{aligned}\langle \boldsymbol{X}, \boldsymbol{Y} \rangle_{\boldsymbol{P}} &\overset{\text{def}}{=} \langle \boldsymbol{P}\boldsymbol{X}, \ \boldsymbol{P}\boldsymbol{Y} \rangle \\ &= \text{trace}\left[\underbrace{\boldsymbol{X}\boldsymbol{Y}^T}_{\text{bilinear map}} \quad \underbrace{\boldsymbol{P}^T\boldsymbol{P}}_{\text{low-rank projection}} \right].\end{aligned}$$

- Similarly, we propose a nonlinear low-rank kernel

$$\begin{aligned}\langle \boldsymbol{X}, \boldsymbol{Y} \rangle_{\boldsymbol{P},\text{h}} &\overset{\text{def}}{=} \langle \boldsymbol{P}h(\boldsymbol{X}), \ \boldsymbol{P}h(\boldsymbol{Y}) \rangle \\ &= \text{trace}\left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y})\boldsymbol{P}^T\boldsymbol{P} \right],\end{aligned}$$

  where $h\colon \mathbb{R}^{d_1 \times d_2} \mapsto \mathbb{R}^{d_1 \times d_2'}$ denotes a nonlinear map between two matrix spaces, and $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y}) \overset{\text{def}}{=} h(\boldsymbol{X})h^T(\boldsymbol{Y}) \in \mathbb{R}^{d_1 \times d_1}$ denotes the matrix product of mapped features.

Nonlinear kernels for matrix predictors

- Each entry of $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y})$ is the inner product between two vectors.
- We refer to $\boldsymbol{K}(\cdot, \ \cdot) \in \mathbb{R}^{d_1 \times d_1}$ as the *lifted (matrix-valued) kernel* induced by the nonlinear map h.
- Examples:
  - Linear kernel: $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{X}\boldsymbol{Y}^T$.
  - Polynomial kernel with degree m: $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y}) = (\boldsymbol{X}\boldsymbol{Y}^T + \lambda \boldsymbol{I})^{\circ m}$.
  - Gaussian kernel: the (i, j)-th entry of $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y})$ is

$$[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y})]_{(i,j)} = \exp\left\{ -\frac{1}{2\sigma^2} \|\boldsymbol{X}[i, :] - \boldsymbol{Y}[j, :]\|_2^2 \right\}$$

    for all $(i, j) \in [d_1] \times [d_1]$.
- Implementation: (1) symmetrization trick; (2) A single projection matrix is enough to guide the optimization. Update at every other step.
- *Perhaps use a different name for $\boldsymbol{K}$? Not really a kernel...*
  $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y}) \neq \boldsymbol{K}(\boldsymbol{Y}, \boldsymbol{X})$; $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Y}) \neq \boldsymbol{K}(\boldsymbol{X}^T, \boldsymbol{Y}^T)$.

Application of nonlinear kernels to matrix-based learning

- Nonparametric classifier: $\text{sign}[\hat{f}(\cdot)] \colon \mathbb{R}^{d_1 \times d_2} \mapsto \{-1, 1\}$, where

$$\hat{f}(\cdot) = \sum_i \hat{\alpha}_i y_i \text{tr}\left[\boldsymbol{K}(\cdot,\ \boldsymbol{X}_i)\hat{\boldsymbol{P}}^T \hat{\boldsymbol{P}}\right].$$

- Nonparametric regression:

$$\hat{\mathbb{P}}(Y = 1|\boldsymbol{X}^{\text{new}}) = \frac{1}{H} \sum_{h \in [H]} \mathbb{1}\left\{\boldsymbol{X}^{\text{new}} \colon \text{sign}[\hat{f}_h(\boldsymbol{X}^{\text{new}})] = 1\right\}$$

- Nonlinear SDR, $Y \perp\!\!\!\perp \boldsymbol{X} \Big| \phi(\boldsymbol{X})$, where

$$\phi(\boldsymbol{X}) = h(\boldsymbol{X}) \times_1 \hat{\boldsymbol{P}}_c \times_2 \hat{\boldsymbol{P}}_r (??)$$

Simulation: Linear vs. nonlinear classification

- linearly separable, homoscedastic case: $d_1 = 10, d_2 = 8, r = 3, N = ?$

	1st	2nd	3rd	4th	5th	average
SVM	0.85	0.70	0.75	0.70	0.50	0.70
SMM	0.90	0.90	0.75	0.80	0.85	0.84
SMM(polynomial)	0.75	0.55	0.85	0.55	0.75	0.69
SMM(gaussian)	0.65	0.50	0.85	0.70	0.80	0.70

Table 1: Miss Classification Rate (MCR) on 5 folded Cross validation(CV)

- Non-separable, heteroscedastic case: $d_1 = d_2 = 2, \mathrm{Cov}(\boldsymbol{X}|y = -1) = 4\mathrm{Cov}(\boldsymbol{X}|y = 1) = 4\boldsymbol{I}_2$.

	sim 2.1	sim 2.2	sim 2.3
SVM	0.76	0.735	0.860
SMM	0.75	0.740	0.865
SMM(polynomial)	0.80	0.750	0.785
SMM(gaussian)	0.71	0.745	0.830

Table 3: This table shows the averaged MCR of 5 folded CV according to different methods.

Details in "042720.pdf"

Simulation: Nonlinear learning

- Nonlinear classification

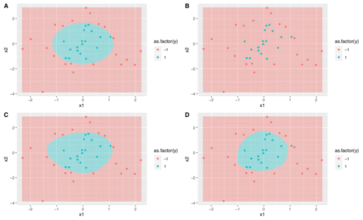


Figure 1: Subfigure A is true ellipsoid boundary. B is linear case boundary which assigns labels all 0. C and D show the boundary of polynomial and exponential kernel respectively.

- Nonlinear probability estimation

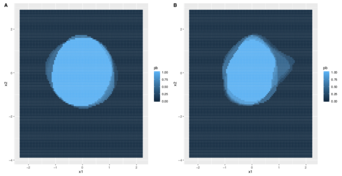


Figure 2: Figure A and B show the estimated probability with polynomial and exponential kernel respectively.

Details in "052120.pdf"

Future work

- [Regression] How does SVM-based learning compare to other nonparametric regression, such as local smoothing, K-NN, Neural Network?

  $\Rightarrow$ SVM-based learning is essentially a kernel smoothing method. Neighborhood is defined by support vectors (?). Intuition?

- [SDR] How does SVM-based SDR compared to other SDR methods?
- [Kernel] Connection between low-rank kernel vs. adaptive kernel, kernel learning, sum of rank-1 kernels?
- Extend to higher-order tensors. Data-driven choice of r.
- Real data analysis