Canonical Kernel Dimension Reduction

Chenyang Tao[a,b], Jianfeng Feng[a,b,c,*]

[a] *Centre for Computational Systems Biology and School of Mathematical Sciences, Fudan University, Shanghai, 200433, PR China*
[b] *Department of Computer Science, Warwick University, Coventry, UK*
[c] *School of Life Science and the Collaborative Innovation Center for Brain Science, Fudan University, Shanghai, 200433, PR China*

Abstract

A new kernel dimension reduction (KDR) method based on the gradient space of canonical functions is proposed for sufficient dimension reduction (SDR). Similar to existing KDR methods, this new method achieves SDR for arbitrary distributions, but with more flexibility and improved computational efficiency. The choice of loss function in cross-validation is discussed, and a two-stage screening procedure is proposed. Empirical evidence shows that the new method yields favorable performance, both in terms of accuracy and scalability, especially for large and more challenging datasets compared with other distribution-free SDR methods.

Keywords: Canonical correlation analysis, Canonical functions, Kernel dimension reduction, Krylov subspace, Sufficient dimension reduction, Reproducing kernel Hilbert space

1. Introduction

In the era of big data, supervised dimension reduction serves as an invaluable tool to make the best use of the high-dimensional datasets by casting them onto some lower dimensional manifolds with minimum loss of relevant information. The task is to seek a low-dimensional embedding $Z \in \mathbb{R}^d$ of some high-dimensional vector $X \in \mathbb{R}^p$ using information from some auxiliary variable

*Corresponding author
Email address: jianfeng64@gmail.com (Jianfeng Feng)

Y, which in most cases is a $\mathbb{R}^q$ vector but can also be more abstract objects such as graphs, texts, *etc.* Popular methods to achieve this task include canonical correlation analysis, partial least square, and LASSO, among others.

One particular research direction is the so-called sufficient dimension reduction (SDR), where a low-dimension representation Z of X that fully captures the conditional distribution of Y given X, i.e., $\mathbb{P}(Y|Z) = \mathbb{P}(Y|X)$, is identified. For computational reasons, Z is usually restricted to linear combinations of X, while not prohibiting other forms (Wang et al., 2014). Since the seminal paper of sliced inverse regression (SIR) (Li, 1991), SDR has been extensively studied (Cook and Ni, 2005; Li and Dong, 2009; Ma and Zhu, 2013). In current studies, SDR is approached in three ways: inverse regression, forward regression and joint approach. Inverse regression focuses on the distribution of X given Y, and popular methods in this category include SIR (Li, 1991), sliced average variance estimator (Cook and Weisberg, 1991) and principal Hessian direction (Li, 1992). While these methods are computationally cheap, they depend on such strong assumptions as elliptical distribution of X. Average derivative estimation (Härdle and Stoker, 1989; Samarov, 1993), minimum average variance estimation (Xia et al., 2002) and sliced regression (Wang and Xia, 2008) are examples of forward regression, which focuses on the distribution of Y, given X. They are free of restrictive probability assumptions, yet suffer from heavy computational burden as a result of the nonparametric estimation procedures involved. The joint approach, including methods such as those based on Kullback-Leibler divergence (Yin and Cook, 2005; Yin et al., 2008), mutual information (MI) (Suzuki and Sugiyama, 2013; Tangkaratt et al., 2015), Fourier analysis (Zhu and Zeng, 2006), integral transforms (Zeng and Zhu, 2010), or canonical dependency (Fung et al., 2002; Karasuyama and Sugiyama, 2012), all focus on exploiting the joint distribution of (X, Y).

The pioneering works of Fukumizu have produced kernel dimension reduction (KDR) techniques, such as trace-based kernel dimension reduction (tKDR) (Fukumizu et al., 2004, 2009) and gradient-based kernel dimension reduction (gKDR) (Fukumizu and Leng, 2012). Among other joint approaches, these

techniques present solutions to the problem of SDR by embedding probability distributions in the reproducing kernel Hilbert space (RKHS) and exploiting the cross-covariance operators between RKHSs. These methods are also characterized as distribution-free. Apart from its theoretical grounding, KDR also showed very competitive empirical performance. Still, its applications is limited by the heavy computational burden involved, especially for tKDR. Although gKDR is much more efficient than tKDR, it suffers from degenerated accuracy on many benchmark problems when compared to tKDR.

In this work, we describe a novel kernel dimension reduction method that improves upon the accuracy of tKDR, while, at the same time, consuming less computational resources than that of gKDR. Our approach is based on kernel canonical-correlation analysis, and, as such, it is termed as ccaKDR. We prove that the central space is equivalent to the space spanned by the derivative of the canonical functions with nonvanishing eigenvalues in RKHS under mild conditions, and a more scalable linear scaling approximation algorithm is presented. We also present a two-stage screening procedure and discuss the choice of loss function, both topics of pragmatic importance. Empirical evidence reveals that better accuracy and scalability can be expected from ccaKDR compared with other distribution-free alternatives.

The paper is organized as follows. In Section 2, we briefly review the technical tools required, propose ccaKDR and present its theoretical justifications, followed by a discussion of relevant issues. In Section 3, we conduct numerical experiments on both synthetic and real-world data to substantiate the paper. Concluding remarks are given in Section 4. MATLAB code for the algorithms and sample data can be found on the authors website.

2. CCA-based kernel dimension reduction

2.1. Background

In this section, we briefly review the mathematical tools needed to derive and compute the proposed ccaKDR. We use capital letters $X, Y, \cdots$ to denote

random variables, bold font capital letters $\boldsymbol{A}, \boldsymbol{B}, \cdots$ to denote matrices, and use notation $[n]$ for the set $\{1, \cdots, n\}$.

Reproducing kernel Hilbert space (RKHS) has been established as a versatile tool in machine learning, especially for nonlinear problems, with the most prominent examples including support vector machines in classification and regression. We briefly review the basic concepts here. If we denote Ω of some set, then we call a real-valued symmetric function $\kappa(\cdot, \cdot)$ defined on $\Omega \times \Omega$ a positive definite kernel if it satisfies $\sum_{i,j=1}^{n} c_i c_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$ for any $\{c_i\}_{i=1}^{n} \in \mathbb{R}$ and $\{\boldsymbol{x}_i\}_{i=1}^{n} \in \Omega$ with any $n \geq 0$, and we will hereinafter simply refer to it as a kernel. For such a kernel on Ω, Aronszajn (1950) established that there is a unique Hilbert space $\mathcal{H}$, with its inner product $\langle \cdot, \cdot \rangle$ induced by κ, consisting of functions on Ω such that (i) $\kappa(\cdot, \boldsymbol{x}) \in \mathcal{H}$, (ii) the linear hull of $\{\kappa(\cdot, \boldsymbol{x}) | \boldsymbol{x} \in \Omega\}$ is dense in $\mathcal{H}$, and (iii) for any $\boldsymbol{x} \in \Omega$ and $f \in \mathcal{H}$, $\langle f, \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}} = f(\boldsymbol{x})$. We note that (iii) is the famous reproducing property and, thus the name reproducing kernel Hilbert space. The representer theorem (Kimeldorf and Wahba, 1970) serves as the foundation of almost all kernel methods, and it basically states that the minimizer of functions in $\mathcal{H}$ of some empirical risk function plus regularization admits the form of a linear combination of $\kappa(\cdot, \boldsymbol{x}_i)$ based on empirical samples $\{\boldsymbol{x}_i\}_i^n$. This equates the optimization on an infinite dimensional search space $\mathcal{H}$ to a finite dimensional search space $\mathbb{R}^n$.

Kernel embedding and cross-covariance operators are theoretical tools developed in recent years for kernel techniques involved with distributions for many statistical problems. Let $(\mathcal{X}, \mathcal{B}_{\mathcal{X}}, \mu_X)$ be the probability measure space for random variable X defined on $\mathcal{X}$ and $(\kappa_{\mathcal{X}}, \mathcal{H}_{\mathcal{X}})$ the measurable kernel and associated RKHS, respectively. A *kernel embedding* of $\mu_{\mathcal{X}}$ with respect to κ_X is defined as $\mathbb{E}_{\mu_X}[\kappa(\cdot, X)] \in \mathcal{H}_{\mathcal{X}}$, and if such embedding map from the space of all probability distributions defined on $\mathcal{X}$ to $\mathcal{H}_{\mathcal{X}}$ is injective, then we call the kernel *characteristic*. That is to say for characteristic kernels $\mathbb{E}_{\mu}[\kappa(\cdot, X)] = \mathbb{E}_{\nu}[\kappa(\cdot, X)]$ implies $\mu = \nu$. This is a generalization of the characteristic functions on probability measures, as defined on Euclidean spaces, and popular examples of characteristic kernels include Gaussian kernel and Laplace kernel. Let $(\mathcal{X} \times$

$\mathcal{Y}, \mathcal{B}_{\mathcal{X}\times\mathcal{Y}}, \mu_{XY})$ be the probability measure space for random variable (X, Y) defined on $\mathcal{X} \times \mathcal{Y}$, and let $\kappa_{\mathcal{X}}$ and $\kappa_{\mathcal{Y}}$ be the measurable kernels on $\mathcal{X}$ and $\mathcal{Y}$, respectively, with the corresponding associated RKHS $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$. Further assume that $\mathbb{E}[\kappa(X, X)]$ and $\mathbb{E}[\kappa(Y, Y)]$ are bounded. The *cross-covariance operator* $C_{XY} : \mathcal{H}_{\mathcal{Y}} \to \mathcal{H}_{\mathcal{X}}$ is defined as the operator that

$$\langle f, C_{XY} g \rangle = \mathrm{Cov}[f(X), g(Y)]$$

holds for all $f \in \mathcal{H}_{\mathcal{X}}$ and $g \in \mathcal{H}_{\mathcal{Y}}$. We can define the *covariance operators* C_{XX} and C_{YY} in a similar manner, and further define $V_{XY} \coloneqq C_{XX}^{-1/2} C_{XY} C_{YY}^{-1/2}$ as the *normalized cross-covariance operator* (NOCCO). For notational simplicity, we suppress the dependence on kernels for the cross-covariance operators in notation. To avoid clutter, we always assume the kernel function κ is centralized with respect to the distribution, *i.e.* $\mathbb{E}[\kappa(X, \boldsymbol{x})] = 0$ for all $\boldsymbol{x}$, and therefore all $f \in \mathcal{H}$ are mean zero (see Gretton et al. (2005); Fukumizu et al. (2007); Sriperumbudur et al. (2010) for more details).

Kernel canonical correlation analysis (KCCA) (Akaho, 2001; Shawe-Taylor and Cristianini, 2004; Hardoon et al., 2004; Fukumizu et al., 2007) solves the following correlation maximization problem

$$\max_{f \in \mathcal{H}_{\mathcal{X}}, g \in \mathcal{H}_{\mathcal{Y}}} \frac{\mathrm{Cov}[f(X), g(Y)]}{\sqrt{\mathrm{Var}[f(X)]\mathrm{Var}[g(Y)]}}. \tag{1}$$

More generally, we call (f_h, g_h) the h-th pair of canonical functions where $f_h \in \mathcal{H}_{\mathcal{X}}$, $g_h \in \mathcal{H}_{\mathcal{Y}}$, (f_h, g_h) are the maximizers of (1) subject to the constraints $\mathbb{E}[f_k f_j] = \mathbb{E}[g_k g_j] = 0$ for all $j < k$. The h-th maximized correlation is denoted as ρ_h and referred to as h-th canonical correlation. Usually (f_h, g_h) are normalized to satisfy $\mathbb{E}[f_h^2] = \mathbb{E}[g_h^2] = 1$, and we call $\{(f_h, g_h, \rho_h)\}_{h=1}^{\infty}$ the solution of KCCA, and refer to $\{(f_h, g_h)\}$ as the canonical function pairs of C_{XY}.

Kernel matrix approximation is a technique often used for kernel applications with large sample size n to facilitate computation. For large n, it is prohibitive to store the full kernel matrix in memory and manipulate it directly. Luckily, many theoretical results justify the use of low-rank approximations

for large-scale kernel matrices (Widom, 1963, 1964; Bach and Jordan, 2003; Bach, 2012). The low-rank approximation usually factorizes the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ into the form $\boldsymbol{K} \approx \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{n\times r}, \boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}$ and $r \ll n$. Extensive researches have been given to the low rank approximation of kernel matrices and popular choices include incomplete Cholesky decomposition (ICL) (Golub and Van Loan, 2012; Bach and Jordan, 2003) and various Nyström method variants (Drineas and Mahoney, 2005; Kumar et al., 2012).

Ritz approximation is an iterative method that extracts Ritz pairs $(\widetilde{\rho}_h, \widetilde{\xi}_h)$ that approximate leading eigenpairs to the eigenvalue problem $\boldsymbol{A}\boldsymbol{\xi} = \rho\boldsymbol{\xi}$, where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. Solving the full eigenvalue problem is inefficient for applications where n is large and only a few leading eigenpairs are needed. Ritz approximation approaches the solution based on Krylov subspace projection. For some $\boldsymbol{b} \in \mathbb{R}^n$, the m-order Krylov subspace is generated by $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b}) = \mathrm{Span}\{\boldsymbol{A}^{m-1}\boldsymbol{b}, \cdots, \boldsymbol{b}\}$. Let $\boldsymbol{V} \in \mathbb{R}^{n\times m}$ be an orthonormal basis for $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b})$; we compute $\boldsymbol{H} \coloneqq \boldsymbol{V}^\top\boldsymbol{A}\boldsymbol{V} \in \mathbb{R}^{m\times m}$ and its eigenpairs $(\widetilde{\rho}_h, \widetilde{\boldsymbol{\zeta}}_h)$. We denote $\widetilde{\boldsymbol{\xi}}_h \coloneqq \boldsymbol{V}\widetilde{\boldsymbol{\zeta}}_h$ and call $(\widetilde{\rho}_h, \widetilde{\boldsymbol{\xi}}_h)$ a Ritz pair, which gives an approximation of the h-th leading eigenpair of the original problem (for details refer to Saad (1992)).

Sufficient dimension reduction (SDR) refers to the construction of a lower-dimensional embedding of X such that the conditional independency between X and Y is captured. More precisely, for a random variable $X \in \mathbb{R}^p$, we want to find a projection matrix $\boldsymbol{B} \in \mathbb{R}^{p\times d}$, such that

$$\mathbb{P}(Y|X) = \mathbb{P}(Y|\boldsymbol{B}^\top X) \text{ or equivalently } Y \perp\!\!\!\perp X|\boldsymbol{B}^\top X,$$

where $d < p$ and $\perp\!\!\!\perp$ represents independence. The space $\boldsymbol{B}^\top X$ is referred to as sufficient dimension reduction space or effective dimension reduction (EDR) space in the literature (Li, 1991). Under some mild conditions, the intersection of all the sufficient dimension reduction spaces exists and is called the *central subspace* (CS) (Cook, 1994; Yin et al., 2008). Another space of practical interest is the so-called *central mean subspace* (CMS) denoted by $\mathcal{S}_{\mathbb{E}[Y|X]}$ where only the conditional independence of expectation, i.e.

$$Y \perp\!\!\!\perp \mathbb{E}[Y|X]|\boldsymbol{B}^\top X,$$

is considered (Cook and Li, 2002). This can give more compact representation of data when only the expectation rather than variance needs to be characterized, which suffices for many prediction tasks.

2.2. ccaKDR

In this section, we detail the proposed ccaKDR and prove some theoretical results. Let $\{(f_h, g_h, \rho_h)\}$ be the solution of KCCA and denote $\mathcal{I} \coloneqq \{h|\rho_h > 0\}$ the set of indices corresponding to nonzero ρ_h. Intuitively, $\{(f_h, g_h)\}_{h\in\mathcal{I}}$ characterizes all modes of dependency between X and Y provided $\mathcal{H}_X$ and $\mathcal{H}_Y$ are sufficiently large, and to recover the CS, it should only be necessary to look at $\{(f_h, g_h)\}_{h\in\mathcal{I}}$. Assuming that a differentiable function f on $\mathcal{X}$ only depends on $\boldsymbol{z} = \boldsymbol{B}^\top\boldsymbol{x}$, *i.e.* $f(\boldsymbol{x}) = f(\boldsymbol{B}^\top\boldsymbol{x}) = f(\boldsymbol{z})$, then by taking derivative *w.r.t.* $\boldsymbol{x}$ yields $\nabla_{\boldsymbol{x}} f = \boldsymbol{B}\nabla_{\boldsymbol{z}} f$ by the chain rule. Thus, to estimate the CS heuristically, one can first estimate the canonical pairs $\{(f_h, g_h)\}_{h\in\mathcal{I}}$ and then use the space spanned by their derivatives to approximate CS. This intuition is confirmed by Theorem 1 below, which uses different assumptions compared to gKDR.

Theorem 1. *In addition to the assumptions* $(i \sim v)$ *in Appendix, assume that the kernel* $\kappa_{\mathcal{Y}}$ *is characteristic. Denote* $\{(f_h, g_h)\}_{h=1}^{\infty}$ *the canonical function pairs of* C_{XY} *and* $\{\rho_h\}_{h=1}^{\infty}$ *the corresponding canonical correlations. If for all* f_h *with nonzero* ρ_h *it holds that* ∇f_h *is contained in Span*$(\boldsymbol{B})$ *almost surely, then* Y *and* X *are conditionally independent given* $\boldsymbol{B}^\top X$.

The theorem above motivates the naïve ccaKDR algorithm described in Algorithm 2 below, which involves an empirical estimate of the KCCA solution. From the proof of the theorem, as shown in the Appendix, we know that $\nabla f_h(\boldsymbol{x}) \in \mathrm{Span}(\boldsymbol{B})$ for all $\boldsymbol{x} \in \mathcal{X}$ and h with $\rho_h > 0$ for any SDR projection matrix $\boldsymbol{B}$. Given that the first l eigenvalues are nonvanishing and that $\boldsymbol{B}$ is a rank-d projection, we want to estimate $\widehat{\boldsymbol{B}}$, given empirical sample $(\boldsymbol{X}, \boldsymbol{Y}) = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$. Denote the kernel matrices for $\boldsymbol{X}, \boldsymbol{Y}$ with their

respective kernel functions $\kappa_{\mathcal{X}}, \kappa_{\mathcal{Y}}$ as $\boldsymbol{K}_x, \boldsymbol{K}_y$, and $\epsilon_x, \epsilon_y > 0$ the regularization parameters. The empirical estimator of canonical function $\hat{f}_h$ is given by $\hat{f}_h = \sum_i \boldsymbol{\xi}_i^h \kappa_{\mathcal{X}}(\cdot, \boldsymbol{x}_i)$ where $\boldsymbol{\xi}^h$ is the h-th eigenvector of the following eigenvalue problem (Bach and Jordan, 2003)

$$\begin{pmatrix} \boldsymbol{0} & \boldsymbol{K}_x\boldsymbol{K}_y \\ \boldsymbol{K}_y\boldsymbol{K}_x & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\zeta} \end{pmatrix} = \rho \begin{pmatrix} (\boldsymbol{K}_x + \epsilon_x \boldsymbol{I})^2 & \boldsymbol{0} \\ \boldsymbol{0} & (\boldsymbol{K}_y + \epsilon_y \boldsymbol{I})^2 \end{pmatrix} \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\zeta} \end{pmatrix}, \quad (2)$$

here $\boldsymbol{I}$ is the identity matrix. Then $\widehat{\boldsymbol{B}}$ is given by the d-leading principal directions of $\{\nabla \hat{f}_h(\boldsymbol{x}_i) |\ i \in [n], h \in [l]\}$.

Algorithm 1: Naïve ccaKDR

Input: Data $(\boldsymbol{X}, \boldsymbol{Y})$, Kernel functions $(\kappa_{\mathcal{X}}, \kappa_{\mathcal{Y}})$, Regularization parameter ϵ,
Number of canonical functions l, Target dimensionality d.

Output: Projection matrix $\boldsymbol{B} \in \mathbb{R}^{p \times d}$

Compute $\boldsymbol{K}_x, \boldsymbol{K}_y$ from $(\kappa_{\mathcal{X}}, \boldsymbol{X})$, $(\kappa_{\mathcal{Y}}, \boldsymbol{Y})$

Centralizing kernel matrices $\boldsymbol{K} := \boldsymbol{H}\boldsymbol{K}\boldsymbol{H}$, where $\boldsymbol{H} = \boldsymbol{I} - n^{-1}\boldsymbol{1}\boldsymbol{1}'$

Set $\epsilon_x = \epsilon_y = \epsilon$, solve (2) for $\{\widehat{\boldsymbol{\xi}}_h\}_{h=1}^l$

Stack $\{\sum_{i=1}^n \widehat{\xi}_{h,i} \nabla \kappa_{\mathcal{X}}(\boldsymbol{x}_j, \boldsymbol{x}_i) \in \mathbb{R}^p | j \in [n], h \in [l]\}$ into matrix $\boldsymbol{N} \in \mathbb{R}^{p \times nl}$

Set $\boldsymbol{B}$ to the left singular vectors of $\boldsymbol{N}$ corresponding to the d largest singular values

Denote (f, g) the first canonical function pair of C_{XY}, and (ϕ, ψ) the unit eigenfunctions of V_{XY} corresponding to the largest singular value satisfying

$$\langle \phi, V_{XY}\psi \rangle_{\mathcal{H}_X} = \max_{\substack{f \in \mathcal{H}_X, g \in \mathcal{H}_Y, \\ \|f\|_{\mathcal{H}_X} = \|g\|_{\mathcal{H}_Y} = 1}} \langle f, V_{XY} g \rangle_{\mathcal{H}_X}.$$

(f, g) and (ϕ, ψ) are related by

$$f = C_{XX}^{-1/2}\phi, \quad g = C_{YY}^{-1/2}\psi.$$

Let $\{\epsilon_n\}_{n=1}^{\infty}$ be a sequence of positive numbers satisfying:

$$\lim_{n\to\infty} \epsilon_n = 0, \quad \lim_{n\to\infty} \frac{n^{-1/3}}{\epsilon_n} = 0, \quad (3)$$

and denote $(\widehat{f}_n, \widehat{g}_n)$ as the n-sample empirical estimator of (f, g) by solving (2) with the regularization parameters set to ϵ_n. The following theorem from Fukumizu et al. (2007) establishes the convergence of empirical estimator of the first canonical function pair, and the result also holds for ensuing canonical functions.

Theorem 2 (Fukumizu et al. (2007), Theorem 2)**.** *Let $\{\epsilon_n\}_{n=1}^{\infty}$ be a sequence of positive numbers satisfying (3). Assume ϕ and ψ are included in $\mathcal{R}(C_{XX})$ and $\mathcal{R}(C_{YY})$, respectively, and that V_{YX} is compact. Then,*

$$\|(\widehat{f}_n - \mathbb{E}_X[\widehat{f}_n(X)]) - (f - \mathbb{E}_X[f(X)])\|_{\mathcal{L}^2(P_X)} \xrightarrow{P} 0 \tag{4}$$

and

$$\|(\widehat{g}_n - \mathbb{E}_Y[\widehat{g}_n(Y)]) - (g - \mathbb{E}_X[g(Y)])\|_{\mathcal{L}^2(P_Y)} \xrightarrow{P} 0. \tag{5}$$

With further assumptions, the consistency of the derivative estimator can also be established. See supplementary material. We note however the convergence rate of the estimator for particular random variable (X, Y) depends on the choice of kernel functions and the regularization sequence. Therefore, in actual practice, it is essential to optimize such choices to achieve optimal sensitivity via cross-validation. The parameter configuration which gives the minimum loss in the cross-validation will then be used to give an estimation with the full sample.

2.3. Efficient computation

This section provides details on how to construct an efficient algorithm for ccaKDR, pseudo-code can be found in the Appendix.

2.3.1. Fast computation of KCCA

For large sample size n, the $\mathcal{O}(n^3)$ naïve implementation is computationally overwhelming, and we therefore use the low-rank approximation to speedup the computation. Let $\boldsymbol{L}_x \boldsymbol{L}_x^\top$ and $\boldsymbol{L}_y \boldsymbol{L}_y^\top$ be the respective low-rank approximations

for $\boldsymbol{K}_x$ and $\boldsymbol{K}_y$, where $\boldsymbol{L}_x \in \mathbb{R}^{n\times r_x}$ and $\boldsymbol{L}_y \in \mathbb{R}^{n\times r_y}$. We adopt the idea from Bach and Jordan (2003) to reformulate the original $2n \times 2n$ eigenvalue problem into a $(r_x + r_y) \times (r_x + r_y)$ eigenvalue problem via singular value decomposition of $\boldsymbol{L}_x$ and $\boldsymbol{L}_y$, and then Ritz approximation is used to efficiently approximate leading eigenvectors. For completeness we present the fast KCCA solution below.

By adding a term on both sides of (2) we can solve the following equivalent problem

$$\begin{pmatrix} \boldsymbol{K}_{XX}^{\epsilon} & \boldsymbol{K}_{XY} \\ \boldsymbol{K}_{YX} & \boldsymbol{K}_{YY}^{\epsilon} \end{pmatrix}\begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\zeta} \end{pmatrix} = (1+\rho)\begin{pmatrix} \boldsymbol{K}_{XX}^{\epsilon} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{K}_{YY}^{\epsilon} \end{pmatrix}\begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\zeta} \end{pmatrix}. \tag{6}$$

where $\boldsymbol{K}_{XX}^{\epsilon} = \boldsymbol{K}_x + \epsilon_x \boldsymbol{I}, \boldsymbol{K}_{YY}^{\epsilon} = \boldsymbol{K}_y + \epsilon_y \boldsymbol{I}, \boldsymbol{K}_{XY} = \boldsymbol{K}_x \boldsymbol{K}_y$. Taking $\widetilde{\boldsymbol{\xi}} = \boldsymbol{K}_x^{\epsilon}\boldsymbol{\xi}$, $\widetilde{\boldsymbol{\zeta}} = \boldsymbol{K}_y^{\epsilon}\boldsymbol{\zeta}$ and eliminating the block diagonal matrix in (6), we have

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{K}_x^r \boldsymbol{K}_y^r \\ \boldsymbol{K}_y^r \boldsymbol{K}_x^r & \boldsymbol{I} \end{pmatrix}\begin{pmatrix} \widetilde{\boldsymbol{\xi}} \\ \widetilde{\boldsymbol{\zeta}} \end{pmatrix} = (1+\rho)\begin{pmatrix} \widetilde{\boldsymbol{\xi}} \\ \widetilde{\boldsymbol{\zeta}} \end{pmatrix}, \tag{7}$$

where $\boldsymbol{K}_x^r \coloneqq \left(\boldsymbol{K}_x^{\epsilon}\right)^{-1}\boldsymbol{K}_x$ and $\boldsymbol{K}_y^r \coloneqq \left(\boldsymbol{K}_y^{\epsilon}\right)^{-1}\boldsymbol{K}_y$. Denote the low rank factorization $\boldsymbol{K}_x \approx \boldsymbol{L}_x\boldsymbol{L}_x^{\top}$ and $\boldsymbol{K}_y \approx \boldsymbol{L}_y\boldsymbol{L}_y^{\top}$. We perform SVD on $\boldsymbol{L}_x = \boldsymbol{U}_x\boldsymbol{S}_x\boldsymbol{V}_x$ and $\boldsymbol{L}_y = \boldsymbol{U}_y\boldsymbol{S}_y\boldsymbol{V}_y$, so $\boldsymbol{K}_x \approx \boldsymbol{U}_x\boldsymbol{\Lambda}_x\boldsymbol{U}_x^{\top}$ and $\boldsymbol{K}_y = \boldsymbol{U}_y\boldsymbol{\Lambda}_y\boldsymbol{U}_y^{\top}$, where $\boldsymbol{\Lambda}_x = \boldsymbol{S}_x^2$ and $\boldsymbol{\Lambda}_y = \boldsymbol{S}_y^2$. Notice

$$\begin{aligned} &\boldsymbol{K}_x + \epsilon_x\boldsymbol{I} = [\boldsymbol{U}_x, \boldsymbol{U}_x^{\perp}](\widetilde{\boldsymbol{\Lambda}}_x + \epsilon_x\boldsymbol{I})[\boldsymbol{U}_x, \boldsymbol{U}_x^{\perp}]^{\top} \\ &(\boldsymbol{K}_x + \epsilon_x\boldsymbol{I})^{-1} = [\boldsymbol{U}_x, \boldsymbol{U}_x^{\perp}]\left(\widetilde{\boldsymbol{\Lambda}}_x + \epsilon_x\boldsymbol{I}\right)^{-1}[\boldsymbol{U}_x, \boldsymbol{U}_x^{\perp}]^{\top} \\ &[\boldsymbol{U}_x, \boldsymbol{U}_x^{\perp}]^{\top}\boldsymbol{U}_x = [\boldsymbol{I}_{r_x}; \boldsymbol{0}] \end{aligned}$$

where $\widetilde{\boldsymbol{\Lambda}}_x = \mathrm{Diag}(\boldsymbol{\Lambda}_x, \boldsymbol{0})$ and $\boldsymbol{U}_x^{\perp}$ is the orthonormal complement of $\boldsymbol{U}_x$. Plug these into the definition of $\boldsymbol{K}_x^r$ gives us

$$\boldsymbol{K}_x^r \approx \boldsymbol{U}_x\boldsymbol{R}_x\boldsymbol{U}_x^{\top},$$

where $\boldsymbol{R}_x$ is applying the function $x \mapsto \frac{x}{x+\epsilon_x}$ to $\boldsymbol{\Lambda}_x$'s elements. And similarly we have

$$\boldsymbol{K}_y^r \approx \boldsymbol{U}_y \boldsymbol{R}_y \boldsymbol{U}_y^\top.$$

This gives the following factorization of l.h.s matrix in (7)

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{K}_x^r \boldsymbol{K}_y^r \\ \boldsymbol{K}_y^r \boldsymbol{K}_x^r & \boldsymbol{I} \end{pmatrix} = [\mathbf{U}, \mathbf{V}] \mathrm{Diag}(\mathbf{R}, \boldsymbol{I}) [\mathbf{U}, \mathbf{V}]^\top$$

where

$$\mathbf{U} \coloneqq \mathrm{Diag}(\boldsymbol{U}_x, \boldsymbol{U}_y), \mathbf{V} \coloneqq \mathrm{Diag}(\boldsymbol{U}_x^\perp, \boldsymbol{U}_y^\perp)$$

and

$$\mathbf{R} \coloneqq \begin{pmatrix} \boldsymbol{I} & \boldsymbol{R}_x \boldsymbol{U}_x^\top \boldsymbol{U}_y \boldsymbol{R}_y \\ \boldsymbol{R}_y \boldsymbol{U}_y^\top \boldsymbol{U}_x \boldsymbol{R}_x & \boldsymbol{I} \end{pmatrix}.$$

This means it is equivalent to solve the eigenvalue problem of $\mathbf{R}$, which is a $(r_x + r_y) \times (r_x + r_y)$ matrix. Denote $(\check{\boldsymbol{\xi}}^\top, \check{\boldsymbol{\zeta}}^\top)^\top$ is the eigenvector estimated for $\mathbf{R}$, then the eigenvector for the original problem is $(\widetilde{\boldsymbol{\xi}}^\top, \widetilde{\boldsymbol{\zeta}}^\top)^\top = \mathbf{U}(\check{\boldsymbol{\xi}}^\top, \check{\boldsymbol{\zeta}}^\top)^\top$, and

$$\begin{aligned} \boldsymbol{\xi} &= (\boldsymbol{K}_x^\epsilon)^{-1} \widetilde{\boldsymbol{\xi}} \\ &= [\boldsymbol{U}_x, \boldsymbol{U}_x^\perp] \left(\widetilde{\boldsymbol{\Lambda}}_x + \epsilon_x \boldsymbol{I} \right)^{-1} [\boldsymbol{U}_x, \boldsymbol{U}_x^\perp]^\top \boldsymbol{U}_x \check{\boldsymbol{\xi}} \\ &= \boldsymbol{U}_x \left(\boldsymbol{\Lambda}_x + \epsilon_x \boldsymbol{I} \right)^{-1} \check{\boldsymbol{\xi}}. \end{aligned}$$

2.3.2. Low-rank approximation of the derivatives

For commonly used Gaussian kernel $\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2/(2\sigma^2))$, the gradient of canonical function $\nabla f_h(\boldsymbol{x})$ can be efficiently computed using a trick similar to that of Fukumizu and Leng (2012). We first construct an auxiliary tensor $\widetilde{\boldsymbol{L}} \in \mathbb{R}^{n \times r_x \times p}$ defined as $\widetilde{\boldsymbol{L}}_{i,j,a} \coloneqq \sigma^{-2} \boldsymbol{X}_{i,a} \boldsymbol{L}_{i,j}$, where $\boldsymbol{X}_{i,a}$ is the a-th dimension of i-th sample and $\boldsymbol{L}_{i,j}$ is the (i,j)-th element of low-rank approximation matrix $\boldsymbol{L}_x$. Given a weight vector $\boldsymbol{\xi} \in \mathbb{R}^n$, the corresponding derivative of function $f_\xi = \sum_i \xi_i \kappa(\boldsymbol{x}_{i'}, \boldsymbol{x}_i)$ *w.r.t.* dimension a at location $\boldsymbol{x}_{i'}$ can

be approximated as

$$\begin{aligned}
\sum_i \xi_i \partial_a \kappa(\boldsymbol{x}_{i'}, \boldsymbol{x}_i) &\approx \sigma^{-2} \sum_i \xi_i (\boldsymbol{X}_{i',a} - \boldsymbol{X}_{i,a}) \kappa(\boldsymbol{x}_{i'}, \boldsymbol{x}_i) \\
&\approx \sigma^{-2} \sum_{j=1}^{r} \sum_{i=1}^{n} (\boldsymbol{X}_{i',a} - \boldsymbol{X}_{i,a}) L(i', j) L(i, j) \xi_i \\
&= \sum_{j=1}^{r} \sum_{i=1}^{n} \widetilde{\boldsymbol{L}}_{i',j,a} \boldsymbol{L}_{i,j} \xi_i - \widetilde{\boldsymbol{L}}_{i,j,a} \boldsymbol{L}_{i',j} \xi_i \\
&= \sum_{j=1}^{r} \widetilde{\boldsymbol{L}}_{i',j,a} \left(\sum_{i=1}^{n} \xi_i \boldsymbol{L}_{i,j} \right) - \sum_{j=1}^{r} \boldsymbol{L}_{i',j} \left(\sum_{i=1}^{n} \widetilde{\boldsymbol{L}}_{i,j,a} \xi_i \right).
\end{aligned}$$

and therefore the matrix $\boldsymbol{N}$ in Algorithm 2 only costs $\mathcal{O}(n)$ operations to approximate instead of $\mathcal{O}(n^2)$ to be computed exactly. This trick can be extended to other Gaussian-like kernels such as Laplacian kernel.

2.4. Complexity analysis

In this section, we present a time complexity analysis of the linear scaling ccaKDR. For notational simplicity, we use $r = \max\{r_x, r_y\}$ to denote the rank of approximated kernel matrix and assume $l < r < n$. We further assume that p is small; therefore, the computational cost involving higher order terms of p would be negligible. $\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y$ and $\boldsymbol{\epsilon}$ denote the set of cross-validation parameters, and $|\cdot|$ denotes the cardinality of a set. We denote n_c as the number of cross-validation folds, $n_\sigma \coloneqq \max\{|\boldsymbol{\sigma}_x|, |\boldsymbol{\sigma}_y|\}$, $n_\epsilon = |\boldsymbol{\epsilon}|$, and c_n for the cost of the loss function. Low-rank factorization is $\mathcal{O}(nr^2)$. Estimating the derivatives for first l canonical functions at all sample points takes $\mathcal{O}(nrpl)$. Thus, all together, the time complexity for ccaKDR is given by $\mathcal{O}(n_\sigma^2 nr^2 + n_\sigma^2 n_\epsilon n_c (nrpl + c_n))$. For comparison, the original gKDR is $\mathcal{O}(n_\sigma^2 nr^2 + n_\sigma^2 n_\epsilon n_c (nr^2 p + c_n))$, which takes $\mathcal{O}(n_\sigma^2 n_\epsilon n_c nrp(r - l))$ more computations. See the discussion in the Appendix section for details and special treatment when p is large.

2.5. Further discussion

In this part, we discuss loss function and cross-validation, which affect the performance of KDR methods, both in terms of accuracy and efficiency.

2.5.1. Loss function

Loss function is a key issue in many statistical learning problems. For KDR, the optimal parameter configuration is determined by choosing the one with minimum loss using cross-validation. In Fukumizu and Leng (2012), loss function is conveniently chosen as the ℓ_2 distance between K-nearest neighbor (KNN) prediction and observation, i.e., $\ell_{2,\mathrm{knn}} = \sum_i \|\widehat{\boldsymbol{y}}_i - \boldsymbol{y}_i\|_2^2$, where $\widehat{\boldsymbol{y}}_i$ is the average of k corresponding $\boldsymbol{y}$ of nearest neighbors of $\boldsymbol{x}_i$ under the estimated projection. Minimizing ℓ_2 is the maximum likelihood estimation under Gaussian assumption, which is notorious for its poor robustness against deviation from Gaussian distribution. Also, in many situations, the ℓ_2 is simply the convenient choice for computational reasons such as convexity and smoothness, whereas in SDR, such properties are irrelevant. For general purpose SDR, strong assumptions on the underlying probability distribution should be avoided, and the more distribution-robust loss functions should be considered. KNN cannot be extended to more general cases within a KDR framework, e.g., $(\mathcal{Y}, \kappa_{\mathcal{Y}})$ as graphs, trees or even distributions with their corresponding kernels, where neither the metric nor averaging operation is defined. In addition, KNN is not economical since $\mathcal{O}(n\log(n))$ time complexity makes it asymptotically the dominant term of KDR if it is used.

Luckily for ccaKDR, the l-leading canonical function pairs $\{(\hat{f}_h, \hat{g}_h)\}_{h=1}^{l}$ can be exploited to give a loss function that circumvents these issues. Because $(\hat{f}_h, \hat{g}_h)$ converges to ρ_h as the sample size grows, provided it has been properly regularized, the empirical correlation estimator $\hat{\rho}_h^t := 1/n' \sum_i \hat{f}_h(\boldsymbol{x}_i^t)\hat{g}_h(\boldsymbol{y}_i^t)$ using the validation set $(\boldsymbol{X}_t, \boldsymbol{Y}_t)$ [1] is well suited to examine if the model is estimated correctly. In our work we simply use $\sum_{h=1}^{l} \hat{\rho}_h^t$ as the loss function, as defined by eigenfunctions, and denote it as ℓ_ρ. This choice makes no further assumptions on the domain of Y and has the same linear scaling as ccaKDR.

[1] Validation set $(\boldsymbol{X}_t, \boldsymbol{Y}_t)$ is independent of the training set $(\boldsymbol{X}_e, \boldsymbol{Y}_e)$ from which $\hat{f}$ and $\hat{g}$ are estimated.

2.5.2. Cross-validation

While ccaKDR and gKDR algorithms are, by themselves, efficient, their performances rely on the cross-validation of parameters, and exploration of parameter space is the major source of computational burden. This could be solved naturally by narrowing down parameter configurations to a smaller candidate set. The idea of screening has recently been proven useful, both theoretically and practically, when processing (ultra-) high dimensional data (Fan and Lv, 2008). In this light, we propose a two-stage screening procedure to be applied first on the uninformative parameter space $\boldsymbol{\Theta}$ with the goal of shrinking it to a more manageable candidate configuration set in the first stage. In the second stage, a cross-validation procedure is run only on this candidate set, thus avoiding wasteful computation on unlikely configurations. An ideal screening operator should provide economical, but still sensitive, computation able enough to sift the parameter space. Preferably, such screening operator should be different from the ensuing cross-validation criteria to avoid bias. The ℓ_ρ introduced above is a natural candidate for the job. In our work we simply run through the full parameter space using ℓ_ρ on part of the cross-validation batches and keep only the fraction of parameters with minimum loss for full cross-validation.

Algorithm 2: ccaKDR-CV

Input: Data $(\boldsymbol{X}, \boldsymbol{Y})$, Kernel function and regularization parameter set $\{(\kappa_{\mathcal{X}}^{\lambda}, \kappa_{\mathcal{Y}}^{\lambda}, \epsilon^{\lambda}) | \lambda \in \Lambda\}$,
Number of canonical functions l, Target dimensionality d.

Output: Projection matrix $\boldsymbol{B} \in \mathbb{R}^{p \times d}$

Divide sample into training set $(\boldsymbol{X}_e, \boldsymbol{Y}_e)$ and validation set $(\boldsymbol{X}_t, \boldsymbol{Y}_t)$

for $\lambda \in \Lambda$ **do**

  Estimate the canonical functions $(\hat{f}_1^{\lambda}, \hat{g}_1^{\lambda})$ with $(\boldsymbol{X}_e, \boldsymbol{Y}_e, \kappa_{\mathcal{X}}^{\lambda}, \kappa_{\mathcal{Y}}^{\lambda}, \epsilon^{\lambda})$

  $\rho_\lambda = \operatorname{corr}(\hat{f}_1^{\lambda}(\boldsymbol{X}_t), \hat{g}_1^{\lambda}(\boldsymbol{Y}_t))$

end for

$\lambda^* = \arg\max \rho_\lambda$, $\boldsymbol{B} = \text{ccaKDR}(\boldsymbol{X}, \boldsymbol{Y}, \kappa_{\mathcal{X}}^{\lambda^*}, \kappa_{\mathcal{Y}}^{\lambda^*}, \epsilon^{\lambda^*}, l, d)$

2.6. Comparison with related alternatives

2.6.1. gKDR

Recall that gKDR uses the eigen-decomposition of

$$\boldsymbol{M}_g(x) = \langle C_{YX}C_{XX}^{-1}\nabla\kappa_{\mathcal{X}}, C_{YX}C_{XX}^{-1}\nabla\kappa_{\mathcal{X}}\rangle_{\mathcal{Y}}$$

to reconstruct the EDR subspace. A more detailed decomposition of the empirical $\boldsymbol{M}_g$ is, with slight abuse of notation, written as

$$\overbrace{\underbrace{C_{YX}}_{\frac{1}{n}\boldsymbol{K}_y\boldsymbol{K}_x}\underbrace{C_{XX}^{-1}}_{n\boldsymbol{K}_x^{-1}}}^{\text{RKHS operators}}\overbrace{\underbrace{\mathcal{P}\frac{\partial}{\partial x_a}\kappa_{\mathcal{X}}(\cdot,\boldsymbol{x}_j)}_{(\boldsymbol{K}_x+\epsilon\boldsymbol{I})^{-1}\frac{\partial}{\partial x_a}\kappa_{\mathcal{X}}(:,X_j)}}^{\text{Projection of derivative}}.$$

The kernel gradient projection step $\mathcal{P}$ is not explicitly stated in Fukumizu and Leng (2012). We note the following distinctions between gKDR and ccaKDR: 1) **Source of error:** The error in gKDR comes from the approximation error $\|\partial\kappa - \mathcal{P}\partial\kappa\|$ and the projection error $\|\mathcal{P}\kappa - \mathcal{P}_\epsilon\kappa\|$ resulting from regularization term ϵ while for ccaKDR the error comes from the approximation error of canonical function $\|f_h - \hat{f}_h\|$. 2) **Use of sample:** In gKDR, all samples $\{\boldsymbol{x}_j\}_{j=1}^n$ are used as an interpolation point to get the projection of $\partial_a\kappa(\boldsymbol{x},\boldsymbol{x}_i)$ for some i, followed by forming the sample-specific matrix M_i by taking the inner product $M_i(a,b) \coloneqq \langle\partial_a\kappa(\boldsymbol{x},\boldsymbol{x}_i), \partial_b\kappa(\boldsymbol{x},\boldsymbol{x}_i)\rangle_{\mathcal{H}_X}$, and finally average over M_i to get the estimate of projection. In ccaKDR, samples are weighted to form an estimate of the canonical functions, and the derivatives of the estimated canonical functions are evaluated on the full sample to estimate the projection.

In gKDR, variants, such as gKDR-i or gKDR-v, can also be similarly used on ccaKDR. And our method, by its formulation, allows for extra flexibility compared with gKDR. For example, an optimal number of canonical pairs l_{opt} can be dynamically determined based on data and the regularization pair (ϵ_x, ϵ_y) can be decoupled to give better control over model complexity of the canonical function pairs estimated.

2.6.2. SMI-DR

Square-loss mutual information based dimension reduction (SMI-DR) is a family of dimension reduction methods which achieves dimension reduction by employing *squared-loss mutual information* (SMI), which can be defined as the a special case of the f-divergence between $p(\boldsymbol{x})p(\boldsymbol{y})$ and $p(\boldsymbol{x}, \boldsymbol{y})$:

$$\mathrm{SMI} := \frac{1}{2} \int \left(\frac{p(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})} - 1 \right)^2 p(\boldsymbol{x})p(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y}.$$

SMI-DR utilizes an analytic approximator of SMI based on density ratio estimation (Sugiyama et al., 2012), which has been shown to possess suitable convergence properties, thus being more tractable than those dimension reduction methods directly based on estimating MI (Cover and Thomas, 2012; Darbellay et al., 1999; Torkkola, 2003; Kraskov et al., 2004; Van Hulle, 2005; Suzuki et al., 2008). Specifically, SMI-DR formulates the problem by directly modeling the projection $\boldsymbol{B}_x, \boldsymbol{B}_y$ and then maximizing $\mathrm{SMI}(\boldsymbol{B}_x^\top X, \boldsymbol{B}_y^\top Y)$ *w.r.t.* $\boldsymbol{B}_x, \boldsymbol{B}_y$, which generalizes canonical correlation from an information theoretical perspective (Karasuyama and Sugiyama, 2012). Since $\boldsymbol{B}_x, \boldsymbol{B}_y$ are orthonormal matrices lying on the *Stiefel manifold*, optimization is achieved via following the *natural gradient* (Amari, 1998). Basically, SMI-DR leverages RKHS nonparametric estimation to approximate the density ratio function thereby characterizing the dependency, while ccaKDR is based on the theory of RKHS cross-covariance operator. While this explicit formulation is very effective for smaller datasets, it does suffer from a few drawbacks. For example, SMI-DR does not assign priorities to the recovered projection, meaning that it is equivalent under orthogonal transformations. Therefore, to move from a higher dimensional projection estimate to a lower dimensional projection estimate, one has to run the algorithm again. Also, the algorithm solves a nonlinear optimization problem that can be trapped in local minima and does not scale well to larger sample size and higher dimensionality.

2.6.3. CANCOR

CANCOR (Fung et al., 2002) achieves dimension reduction via linear canonical correlation analysis. It builds a B-spline basis Φ for the target variable Y, and finds the linear canonical correlation between $\Phi(Y)$ and X, then directly uses the linear weights for X as the projection. Although computationally simple, this method is based on strong assumptions and derived for the classical small p large n case; as such, it does not generalize well for many modern datasets of interest.

3. Experiments

In this section we perform extensive experiments on the proposed ccaKDR. Since a comprehensive study has been conducted in Fukumizu and Leng (2012) to compare the performance of gKDR and many other SDR methods, we focus on benchmarking ccaKDR against gKDR. We also carry out a comparison with SMI-DR since both gKDR and SMI-DR were published at the same time, and such comparison is missing in the literature. For numerical examples with known projections, we use the *trace correlation error* (TCE) defined as

$$\mathrm{TCE}(\boldsymbol{B}, \widehat{\boldsymbol{B}}) := \sqrt{\mathbf{Tr}(\boldsymbol{B}\boldsymbol{B}^\top(\boldsymbol{I}_p - \widehat{\boldsymbol{B}}\widehat{\boldsymbol{B}}^\top))/\mathbf{Tr}(\boldsymbol{B}^\top\boldsymbol{B})}$$

to measure the discrepancy between the ground truth projection B and estimated projection $\widehat{B}$. For real-world datasets with unknown ground truth projections, we compare the prediction accuracy between gKDR and ccaKDR on holdout data using the estimated projections. We use our own implementation of the linear scaling gKDR in MATLAB by modifying the original gKDR code[2] and tune it for maximum efficiency. As in this study we focus on scalability and convergence for KDR on large datasets. Therefore, only the linear scaling implementation is used in the experiments, while similar results can be obtained for exact implementations (result not shown). The Gaussian kernel

[2] `http://www.ism.ac.jp/~fukumizu/software.html`

$\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2/(2\sigma^2))$ is used for all the experiments, and we use settings for the cross-validation parameters similar to those used in Fukumizu and Leng (2012): regularization parameter $\epsilon \in \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}$, kernel bandwidth scaling factor $c \in \{0.5, 0.75, 1, 2, 5, 10\}$, KNN neighbor set to $k = 5$ and cross-validation fold set to $n_c = 5$. Kernel bandwidth is set to $\sigma_c = c\sigma_{\text{med}}$, where σ_{med} is the median Euclidean distance in the input space. For ccaKDR, we use ℓ_ρ to eliminate 90% of the parameter settings (negative) using 3 out of 5 cross-validation batches and run full cross-validation on the remaining 10% parameter configurations (positive) with ℓ_1 as the loss function. All experiments were run on a PC with Intel i5-4590 CPU and 16G memory, unless otherwise specified.

3.1. Synthetic examples

The following four synthetic examples from Fukumizu and Leng (2012) are used to compare the empirical performance of gKDR and ccaKDR.

$$(a)\begin{cases} Y_1 = Z\sin(Z) + W, & Z = \frac{1}{\sqrt{5}}(X_1 + 2X_2), \\ X_j \sim \mathcal{U}[-1,1], & W \sim \mathcal{N}(0, 10^{-2}). \end{cases} \qquad (b)\begin{cases} Y_1 = (Z_1^3 + Z_2)(Z_1 - Z_2^3) + W, \\ Z_1 = \frac{1}{\sqrt{2}}(X_1 + X_2), Z_2 = \frac{1}{\sqrt{2}}(X_1 - X_2), \\ X_j \sim \mathcal{U}[-1,1], W \sim \text{Gamma}(1,2). \end{cases}$$

$$(c)\begin{cases} Y_1 = (X_1 - a)^4 E, \\ X_j \sim \mathcal{N}(0,1/4) * I_{[-1,1]}, E \sim \mathcal{N}(0,1), \\ (\text{with } a = 0). \end{cases} \qquad (d)\begin{cases} Y_1 = \sum_{j=1}^{5}(Z_{2j-1}^3 + Z_{2j})(Z_{2j-1} - Z_{2j}^3) + W, \\ Z_{2j-1} = \frac{1}{\sqrt{2}}(X_{2j-1} + X_{2j}), \\ Z_{2j} = \frac{1}{\sqrt{2}}(X_{2j-1} - X_{2j}), \\ X_j \sim \mathcal{U}[-1,1], W \sim \text{Laplace}(0,2). \end{cases}$$

$j \in [10]$ for $(a) - (c)$ and $j \in [50]$ for (d).

Here $\mathcal{U}[a,b]$ denotes random variable uniformly distributed on $[a,b]$, $\mathcal{N}(\mu,\sigma^2)$ is the normal distribution with mean μ and variance σ^2, $I_{\mathcal{A}}$ is the indicator function for set $\mathcal{A}$, and Gamma(a,b) and Laplace(a,b) denote the Gamma and Laplace distribution with respective parameters.

3.1.1. Convergence and scalability

In this part, we examine error convergence and algorithmic scalability with respect to the kernel matrix rank and sample size for gKDR and ccaKDR using the synthetic data generated from model (a)-(d). For the rank experiments, the sample size is fixed at $n = 1,000$, and the maximum allowed rank r is tested for $\{40, 60, 80, 120, 160, 200, 300\}$. ICL is used for the low rank approximation of the kernel matrices. For the sample size experiments, we fix the maximum allowed rank at $r = 100$ and test for sample size $n \in \{250, 500, 1000, 2000\}$. We copula transform Y to normalize the target space variable. We dynamically choose the number of canonical functions with its maximum set to $d + 1$. All experiments are repeated for $M = 100$ times to ensure stability. The results are presented in Figure 1 and Figure 2 for the rank and sample size experiment, respectively.

For the synthetic problem (a)-(c), it can be seen that ccaKDR consistently outperforms gKDR as r and n grow larger, while gKDR shows more robustness against poor approximation of the kernel matrix. ccaKDR gives poor reconstruction in test problem (d) because the leading canonical functions are not well approximated, and the association is distributed across the entire spectrum. The error becomes smaller as finer approximation is used for the kernel matrix. Better scalability is observed for ccaKDR in the sample size experiment. To gain further insights, we plot the error distribution in Figure 3 for test problem (b) at $n = 1,000$ and $r \in \{40, 80, 100, 300\}$. While ccaKDR generally gives a much better performance for problem (b) when r is large, it should be noted that it occasionally generates a projection of large discrepancy, although the occurrence of such event vanishes with increasing sample size.

To explain the improved performance alongside the degenerated solution, the scatter plot of empirical loss and estimation error, as shown in Figure 4, can be consulted for a test problem (b) experiment with $n = 1,000$ and $r = 100$. For ccaKDR, the proposed screening criteria successfully filters out most poorly performing parameter settings, although some of the good candidates have also been excluded. The performance gain for ccaKDR over gKDR seems

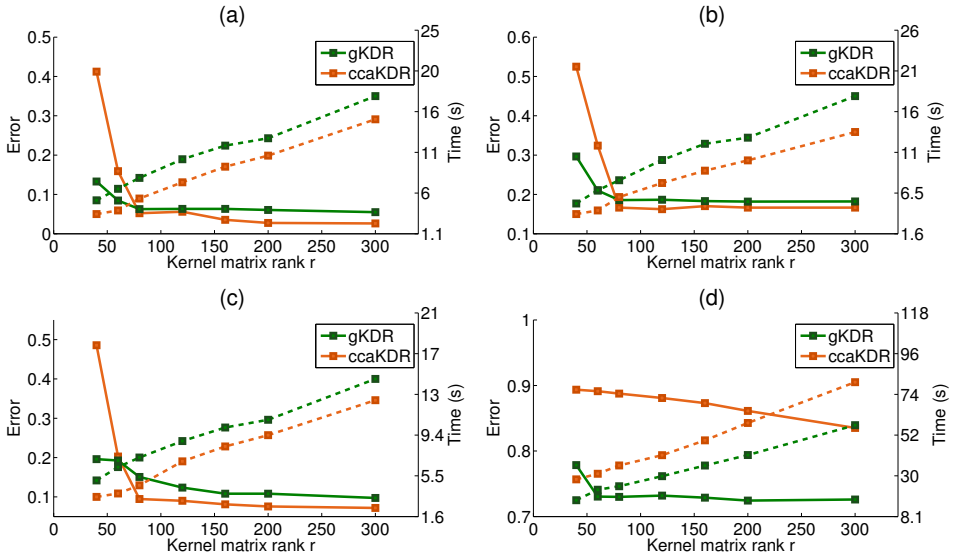

Figure 1: Mean trace correlation error (solid) convergence and computation time (dashed) scaling with respect to kernel matrix rank. green: gKDR, orange: ccaKDR. (lower is better)

to come from the fact that for many problems, 1) the lower bound for the error is better than that of gKDR, 2) more parameter configurations correspondingly give relatively better performance. That is to say, ccaKDR is more insensitive to the parameters compared with gKDR.

3.1.2. Loss function and target space transformation

To further study the influence of using different loss functions and applying transformations on the target space variables, we test the performance of ℓ_2, ℓ_1, ℓ_{rank}, ℓ_{true} and ℓ_ρ with $n = 1,000$ and $r = 100$ for both original and transformed Y. Here ℓ_{true} denotes the log-likelihood of ground truth probability density function of the additive error when applicable, and for ccaKDR, we fix $l = d$. We summarize the results in Table 1. We exclude the discussion of ccaKDR with problem (d) as it does not work with the specified rank. Consistent with previous experiments, ccaKDR yields the best performance on problem (a)-(c). All loss functions benefit from using the copula transformed kernel $\boldsymbol{K}_y$ except

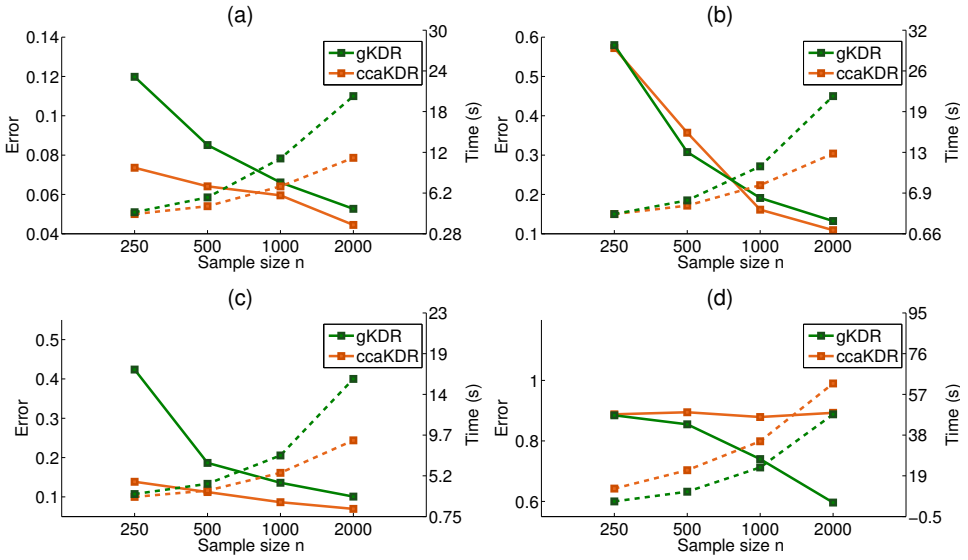

Figure 2: Mean trace correlation error (solid) convergence and computation time (dashed) scaling with respect to sample size. green: gKDR, orange: ccaKDR. (lower error is better)

for the ℓ_2 loss in problem (c). The ℓ_ρ performs surprisingly well for (b) and (c) in that it reduces the error by a large margin. This is quite understandable because for (b), and especially, for (c), all other loss functions are mis-specified. While ℓ_{true} works fine for (a), it does not give satisfactory results for (b), suggesting that prior knowledge of the error distribution might not help. Using more robust loss functions such as ℓ_1 and ℓ_{rank} do improve the results in problem (b) and (c) for both gKDR and ccaKDR, but with a larger gain for ccaKDR.

3.1.3. Comparison with SMI-DR

We further compare the performance of gKDR and ccaKDR with the following two SMI-DR methods with corresponding MATLAB codes obtained from the website of Sugiyama's Lab[3]:

LSDR (Suzuki and Sugiyama, 2013) Projection $\boldsymbol{B}_x$ is obtained via maximizing

[3] `http://www.ms.k.u-tokyo.ac.jp/software.html`

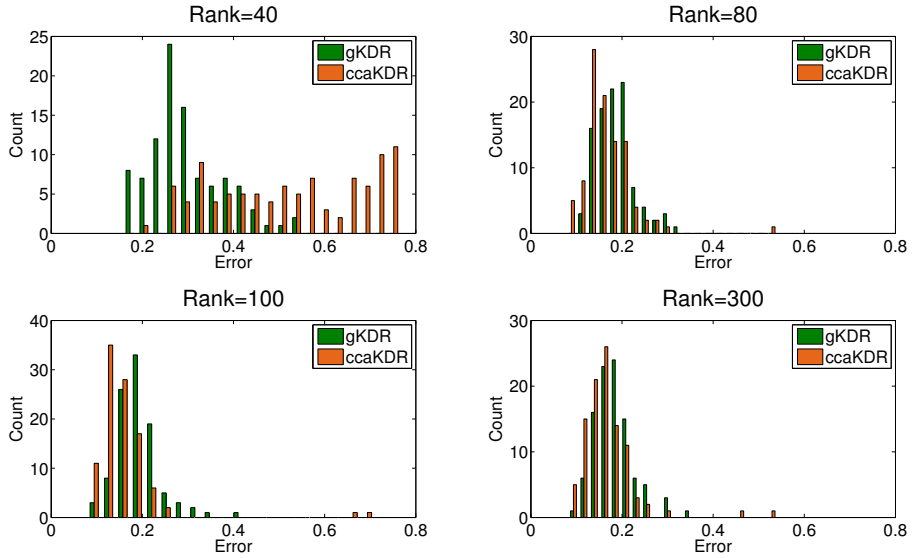

Figure 3: Mean trace correlation error histogram for gKDRb at $n = 1,000$ for $r = \{40, 80, 100, 300\}$. green: gKDR, orange: ccaKDR. (lower error is better)

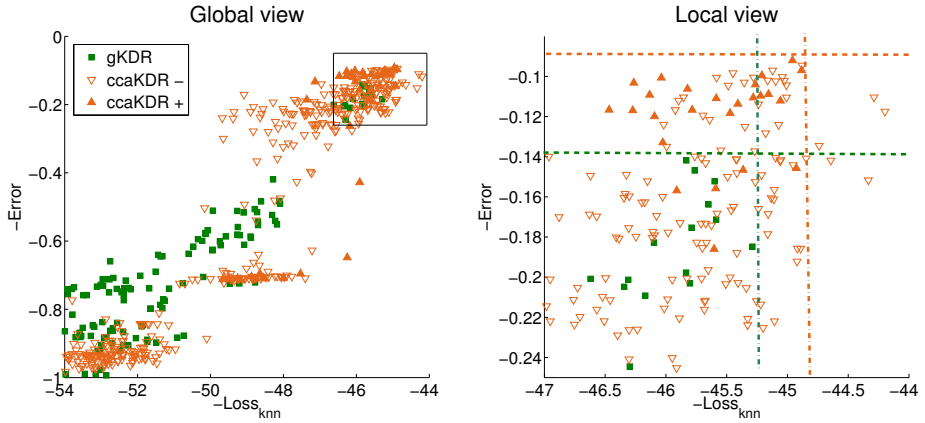

Figure 4: Empirical loss function evaluation vs trace correlation error for all parameter configurations. green: gKDR, solid orange: ccaKDR + (positive parameter configurations selected by the screening criteria), hollow orange: ccaKDR – (negative parameter settings excluded by the screening criteria), horizontal line: best possible error, vertical line, achieved error. (Left) All parameter configurations. (Right) Parameter configurations within the bounding box on the left figure.

Table 1: Comparison of loss functions and transformation on Y

Problem	Method	Original Y					Copula transformed Y				
		ℓ_1	ℓ_2	ℓ_{rank}	ℓ_{true}	ℓ_ρ	ℓ_1	ℓ_2	ℓ_{rank}	ℓ_{true}	ℓ_ρ
(a)	gKDR	0.0810	0.0792	0.0813	0.0792	-	0.0649	0.0648	0.0658	0.0648	-
	ccaKDR	0.0487	0.0476	0.0497	0.0476	0.0866	0.0471	**0.0459**	0.0476	**0.0459**	0.0870
(b)*	gKDR	0.1962	0.2063	0.1844	0.3711	-	0.1907	0.1984	0.1830	0.3777	-
	ccaKDR	0.2317	0.2352	0.1988	0.3545	0.2472	0.1734	0.1773	0.1656	0.2397	**0.1431**
(c)	gKDR	0.1002	0.1862	0.1195	-	-	0.0881	0.2350	0.1159	-	-
	ccaKDR	0.0808	0.1832	0.1030	-	0.0666	0.0730	0.1663	0.0699	-	**0.0582**
(d)	gKDR	0.7475	0.7489	0.7488	0.7475	-	0.7368	0.7357	**0.7343**	0.7368	-
	ccaKDR	0.8851	0.8841	0.8855	0.8851	0.8859	0.8876	0.8883	0.8875	0.8876	0.8871

Mean TCE, lower is better.

* The ℓ_{true} for (b) is demeaned and regularized to avoid degeneracy.

SMI between Y and $\boldsymbol{B}_x^\top X$.

LSCDA (Karasuyama and Sugiyama, 2012) Projection $\boldsymbol{B}_x$ is obtained via maximizing SMI between $\boldsymbol{B}_y^\top Y$ and $\boldsymbol{B}_x^\top X$, where $\boldsymbol{B}_y$ is optimized along with $\boldsymbol{B}_x$.

These two methods represent the counterpart of gKDR and ccaKDR respectively in the SMI-DR genre, as one uses all information contained in Y while the other tries to project Y onto some canonical space together with X. For LSCDA, the projection dimension of Y must be specified and we set that to the true value, which is 1. To further test the capability of the canonical formulation, in addition to the original test problem we set up a more challenging problem than the original one: augmenting Y to 10 dimensions via filling the 9 new dimensions with shuffled Y_1. Default parameter setting is used for both LSDR and LSCDA as we found no improvement using other settings. To establish a fair comparison, we restrict the trial and maximum iteration number of SMI-DR so that similar time is used for KDR and SMI-DR. $n = 1,000$ samples are drawn, and we repeat the experiments for $M = 100$ times. Note that we exclude ccaKDR and LSDR from problem (d), as the former has already been shown to have bad performance under the parameter setup, and the latter does not terminate within 5 minutes. The results are summarized in Table 2.

Table 2: Comparison with SMI-DR

Dim(Y)	Method	Problem			
		(a)	(b)	(c)	(d)
1	ccaKDR	0.0560(4.74)	**0.1500(5.59)**	0.0608(4.86)	-
	gKDR	0.0636(7.17)	0.1850(7.38)	0.1362(7.24)	**0.7279(25.62)**
	LSDR	**0.0337(6.49)**	0.2567(13.29)	**0.0469(5.99)**	-
	LSCDA	0.3554(17.64)	0.6667(11.53)	0.3510(13.04)	0.8162(31.71)
10	ccaKDR	**0.0751(5.98)**	**0.2982(6.76)**	**0.0771(6.09)**	-
	gKDR	0.0806(11.32)	0.4920(11.66)	0.9173(11.40)	0.8909(42.37)
	LSDR	0.2372(6.48)	0.8178(6.05)	0.1538(9.36)	-
	LSCDA	0.6805(11.99)	0.8419(14.81)	0.9015(9.29)	**0.8879(30.42)**
	LSDR†	0.1074(51.97)	0.7935(40.42)	0.1379(52.05)	-
	LSCDA†	0.0248(501.59)	0.1708(765.56)	0.3661(559.76)	time out

Mean TCE (mean computation time), lower is better.

† Using default setting without the time constraint, run on a two-way Intel Xeon 2660 v3 machine.

In the original case, LSDR achieves either optimal or near optimal result for problem (a) and (c), but does not perform so well for problem (b), even though a longer runtime was allocated. Our ccaKDR gives the best performance for problem (b), while a suboptimal result is obtained for problem (a) and (c). gKDR gives the best performance for problem (d) and reasonable performance for (a) and (b). LSCDA gives the worst performance, even though a result similar to that of LSDR should be expected. For the augmented case, ccaKDR clearly shows more robustness, as only slight degeneration is suffered for problem (a) and (c), and it still outperforms three other methods for problem (b). gKDR shows reasonably good performance for problem (a) but suffers substantial loss for problem (b) and (c). For problem (d), no algorithm gives a reasonable result. LSDR loses ground in its performance because the augmented dimensions make it harder to optimize the SMI. LSCDA again gives the worst performance in the augmented case. Given enough time, SMI-DR methods, particularly LSCDA, tend to outperform KDR methods on some problems (see last two rows in Table 2 for the augmented case). However, their advantage disappears as the dimension p, q increases, or the structural dimensionality is misspecified. On the other hand, while inferior to SMI-DR in certain cases, KDR methods are much more efficient and can be used to provide a good initialization for SMI-SDR to speed up their convergence.

3.2. Real-world datasets

In this section, real-world datasets are used to benchmark the proposed ccaKDR together with gKDR and LSCDA. We exclude LSDR since it scales so badly to p and fails to converge within the time limit[4] on most datasets. We use multitarget regression data curated on the Mulan projects site[5] (Tsoumakas et al., 2011) and UCI data repository[6] (Lichman, 2013). The datasets are briefly summarized in Table 3 with more details found in the supplementary material.

[4]Time limit is set to 600s for $n < 500$ and $1500s$ otherwise.

[5]`http://mulan.sourceforge.net`

[6]`https://archive.ics.uci.edu/ml/index.html`

Although proposed as distribution-free, KDR and SMI-DR have mostly been benchmarked on single-target regression or single-label prediction tasks in previous studies. To fill this gap, we specifically chose several multitarget regression and multilabel classification tasks. The experimental setup is to estimate $1 \sim 8$ dimensional projections and we use 5-neighbor KNN regression to evaluate the result. Except for the binary labels for the target entries, all data entries are normalized to zero mean and unit variance. No further transformations are attempted on target space. We use the same parameter space as that used in synthetic problems for cross-validation. For datasets with predefined training/testing set, we use only samples from the training set to train the model and all samples from the testing set for validation. For datasets with unspecified training/testing, we randomly draw a number of samples for training and use the rest for testing. To study the convergence, we sample a progressively larger number of instances as the training set. To ensure the stability of the result, we repeat each experiment 50 times for regression tasks[7] and 10 times for classification tasks. The mean errors from the experiments are reported in Table 4 and Table 5 for regression and classification tasks respectively. For KDR methods, ℓ_1 loss is used for cross-validation and ICL[8] is used to restrict the rank of the approximated kernel matrix to a maximum of 100. For LSCDA, we use d-dimensional projection estimated plus one random direction as the initial guess for the $(d+1)$-dimensional projection to 'hot-start' the algorithm.

As shown in Table 4, for *WQ* and *RF2*, all three methods have similar prediction accuracy. However, for *ATP1d* and *SCM1d*, which are characterized by high feature space dimension and more complex target space, ccaKDR gives better predictions compared with the other two competitors. gKDR consistently gives the worst performance on these two datasets, while LSCDA lies

[7] For LSCDA with SCM1d, only 10 repetitions are attempted because it took too long.

[8] Although some earlier studies reported that ICL gave inferior performance amongst a range of kernel matrix low rank approximation schemes, we observed that for most of the datasets used in our study, ICL gives better or similar approximation compared with the Nyström scheme, with or without adaptive sampling.

Table 3: Dataset summary

Name	Type	Instances	Features	Targets
ATP1d	MTR	337	411	6
WQ	MTR	1060	16	14
RF2	MTR	4108/5017	576	8
SCM1d	MTR	8145/1658	280	16
Ionosphere	SLC	351	33	2
Wdbc	SLC	569	30	2
Usps	SLC	2007	256	10
Isolet	SLC	6238/1559	617	26
Yeast	MLC	1500/917	103	14

MTR: multitarget regression, SLC: single-label classification, MLC: multilabel classification

somewhere in the middle. Our ccaKDR prevails for all four datasets in terms of computational efficiency and scalability. For *SCM1d* and *RF2*, we note the occurrence of overlearning from the training sample, which can be attributed to some unknown shift in the training and testing domain. This suggests that pooling all available samples to train the model may not be a good practice in field applications, especially when data are known to be collected from different domains or distant periods.

For binary label prediction datasets *Ionosphere* and *Wdbc*, also featuring small sample size and low dimensional feature space, LSCDA achieves the lowest prediction error, but with substantially more time. ccaKDR demonstrates slightly inferior accuracy, but with significantly less runtime, while gKDR gives the worst prediction for lower dimensional embeddings, although its performance slowly catches up as the projection dimension increases. For *Usps* and *Isolet*, with more labels and higher dimensional feature space, LSCDA still gives the best prediction, except for the *Isolet* with the smallest sample size, where

ccaKDR did better. However, significantly more computational resources were used by LSCDA since it took $6 \sim 9$ times more time than that used by ccaKDR when the sample size was over 500. To see what may benefit if extra computational time is granted for KDR, we additionally list the results for *Usps* and *Isolet* with $d = \{9, \cdots, 16\}$ in Table 6 for gKDR and ccaKDR. A much better accuracy is achieved by ccaKDR, using a fraction of the time required for LSCDA. Very slow convergence is observed for gKDR for those two datasets. Finally, for the more challenging multilabel prediction Yeast dataset, ccaKDR gives the best prediction accuracy using a minimum amount of time. Bad scaling makes it impractical for LSCDA to be used directly in those big data applications.

4. Conclusion

In this paper, we extend the works of (Fukumizu et al., 2004, 2009; Fukumizu and Leng, 2012) by presenting a new kernel dimension reduction method called ccaKDR. The proposed ccaKDR exploits the gradient of the estimated leading canonical functions to find the directions that achieve sufficient dimension reduction, which generalizes tKDR and gKDR. It inherits the favorable properties of KDR by being distribution free, and the SDR property is theoretically assured. Meanwhile the new formulation has made ccaKDR more scalable than its predecessors.

Practical issues were discussed with particular focus on the choice of loss function. Empirical experiments not only confirmed the advantage of ccaKDR in scalability, but also showed improvement in accuracy over gKDR on a wide range of problems. Comparisons with SMI-DR methods were also presented. Empirical evidence shows that ccaKDR gives better performance on those more challenging tasks in practice. While the proposed ccaKDR presents an interesting alternative in SDR especially for large and high dimensional datasets, many aspects still need to be improved in the future. First, for problems with dependency structure not effectively captured by the few leading canonical functions, ccaKDR suffered degenerated performance. Second, the parameter space

Table 4: Target prediction error (mean absolute error)

N	Method	1	2	3	4	5	6	7	8	Time
					ATP1d					
	gKDR	0.3841	0.3827	0.3523	0.3342	0.3163	0.3067	0.2976	0.2873	105.72
250	ccaKDR	0.3560	0.3007	0.2810	0.2727	0.2615	0.2562	**0.2525**†	0.2527	**58.05**
	LSCDA	0.3702	0.3041	0.2842	0.2742	0.2680	0.2636	0.2618	0.2601	136.68
					WQ					
	gKDR	0.7830	0.7518	0.7300	0.7115	0.6993	0.6908	0.6850	0.6792	17.89
250	ccaKDR	0.7579	0.7233	0.7031	0.6901	0.6829	0.6805	0.6775	**0.6759**	**3.41**
	LSCDA	0.7736	0.7278	0.7064	0.6915	0.6843	0.6805	0.6780	0.6773	18.85
	gKDR	0.7831	0.7507	0.7214	0.7003	0.6836	0.6730	0.6670	0.6624	24.63
500	ccaKDR	0.7527	0.7195	0.6974	0.6808	0.6726	0.6697	0.6656	0.6635	**4.76**
	LSCDA	0.7653	0.7209	0.6977	0.6796	0.6703	0.6655	0.6623	**0.6613**	71.68
	gKDR	0.7876	0.7490	0.7164	0.6924	0.6744	0.6636	0.6555	0.6505	30.98
750	ccaKDR	0.7499	0.7157	0.6904	0.6721	0.6651	0.6581	0.6542	**0.6502**†	**5.81**
	LSCDA	0.7611	0.7161	0.6894	0.6712	0.6619	0.6562	0.6522	0.6509	84.47
					RF2					
	gKDR	0.9552	0.8401	0.7487	0.6996	0.6734	0.6768	0.6606	0.6455	15.41
250	ccaKDR	0.8726	0.7457	0.7487	0.7260	0.6952	0.6728	0.6551	**0.6455**†	**2.64**
	LSCDA	0.8948	0.6899	0.6639	0.6572	0.6527	0.6507	0.6478	0.6465	14.05
	gKDR	0.9703	0.8165	0.7300	0.7013	0.6834	0.6856	0.6600	0.6493	18.74
500	ccaKDR	0.8670	0.7854	0.7365	0.7148	0.6804	0.6775	0.6583	**0.6492**	**3.65**
	LSCDA	0.8939	0.7178	0.6778	0.6646	0.6584	0.6530	0.6517	0.6500	56.08
	gKDR	0.9938	0.8301	0.7086	0.6954	0.6765	0.6884	0.6644	**0.6521**	29.26
1,000	ccaKDR	0.8749	0.7657	0.7433	0.7083	0.6776	0.6717	0.6532	0.6522	**5.62**
	LSCDA	0.9202	0.6996	0.6728	0.6636	0.6577	0.6540	0.6533	0.6529	84.77
					SCM1d					
	gKDR	0.6282	0.7560	0.6944	0.6442	0.6116	0.5879	0.5612	0.5331	87.81
250	ccaKDR	0.6349	0.5349	0.4906	0.4446	0.4245	0.4164	**0.4129**	0.4179	**38.20**
	LSCDA	0.6795	0.6054	0.5615	0.5386	0.5112	0.4847	0.4784	0.4715	106.36
	gKDR	0.6288	0.7309	0.6675	0.6278	0.5901	0.5567	0.5311	0.5124	142.18
500	ccaKDR	0.6297	0.5314	0.4790	0.4430	0.4129	0.4076	**0.4055**†	0.4065	**57.35**
	LSCDA	0.6666	0.5919	0.5545	0.5117	0.4767	0.4533	0.4387	0.4348	354.44
	gKDR	0.6307	0.7251	0.6716	0.6249	0.5786	0.5301	0.5131	0.5020	268.82
1,000	ccaKDR	0.6298	0.5522	0.4991	0.4606	0.4291	**0.4119**	0.4128	0.4183	**93.54**
	LSCDA	0.6462	0.5806	0.5319	0.5000	0.4783	0.4473	0.4314	0.4234	584.78
	gKDR	0.6282	0.7196	0.6695	0.6049	0.5647	0.5344	0.5139	0.5035	522.43
2,000	ccaKDR	0.6264	0.5435	0.5104	0.4733	0.4399	0.4277	**0.4226**	0.4250	**144.72**
	LSCDA	0.6599	0.5790	0.5374	0.4975	0.4665	0.4584	0.4360	0.4243	1054.73

Table 5: Label prediction error

N	Method	1	2	3	4	5	6	7	8	Time
					Ionosphere					
	gKDR	0.1899	0.2364	0.1824	0.1521	0.1440	0.1364	0.1392	0.1380	18.81
250	ccaKDR	0.3309	0.1416	0.1289	0.1226	0.1232	0.1283	0.1309	0.1360	**5.81**
	LSCDA	0.2523	0.1543	0.1255	**0.1184**†	0.1190	0.1166	0.1204	0.1236	24.76
					Wdbc					
	gKDR	0.1121	0.2078	0.1310	0.0902	0.0685	0.0562	0.0408	0.0381	18.07
250	ccaKDR	0.1520	0.0322	0.0305	0.0322	0.0308	0.0345	0.0366	0.0362	**5.60**
	LSCDA	0.0449	0.0303	0.0312	0.0298	0.0316	**0.0296**†	0.0313	0.0353	25.23
					Usps					
	gKDR	0.7401	0.6573	0.5694	0.4992	0.4433	0.4102	0.3614	0.3267	41.74
250	ccaKDR	0.8002	0.5243	0.4265	0.3308	0.2703	0.2310	0.1983	0.1813	**34.55**
	LSCDA	0.6533	0.5038	0.3365	0.2758	0.2179	0.1858	0.1767	**0.1689**	96.10
	gKDR	0.7339	0.6393	0.5342	0.4490	0.3914	0.3353	0.2902	0.2480	78.27
500	ccaKDR	0.8035	0.5325	0.4233	0.3559	0.2514	0.1966	0.1618	0.1447	**49.03**
	LSCDA	0.6686	0.4589	0.2818	0.2137	0.1664	0.1457	0.1331	**0.1270**	410.25
	gKDR	0.7154	0.6436	0.5350	0.4415	0.3657	0.3075	0.2642	0.2152	145.09
1000	ccaKDR	0.7975	0.5447	0.4362	0.3349	0.2486	0.1886	0.1442	0.1289	**75.49**
	LSCDA	0.6500	0.4147	0.2497	0.1786	0.1433	0.1246	0.1178	**0.1119**†	630.38
					Isolet					
	gKDR	0.8838	0.8764	0.8217	0.7810	0.7282	0.6811	0.6470	0.6120	117.77
250	ccaKDR	0.9309	0.7326	0.5431	0.4413	0.3797	0.3448	0.3245	**0.2852**	**72.95**
	LSCDA	0.8753	0.7069	0.5712	0.4474	0.4022	0.3729	0.3405	0.3153	217.08
	gKDR	0.8751	0.8358	0.7724	0.7026	0.6455	0.5824	0.5267	0.4904	215.62
500	ccaKDR	0.9201	0.7072	0.5336	0.4123	0.3389	0.2882	0.2554	0.2410	**105.40**
	LSCDA	0.8666	0.6639	0.4857	0.3391	0.2736	0.2407	0.2131	**0.1975**	793.90
	gKDR	0.8711	0.8190	0.7182	0.6361	0.5661	0.4975	0.4448	0.3963	421.81
1,000	ccaKDR	0.9244	0.6985	0.5199	0.3845	0.2967	0.2534	0.2308	0.2072	**176.79**
	LSCDA	0.8468	0.6175	0.4097	0.2929	0.2276	0.1924	0.1725	**0.1606**†	1185.11
					Yeast					
	gKDR	0.2812	0.2699	0.2597	0.2532	0.2486	0.2459	0.2446	0.2428	41.57
250	ccaKDR	0.2972	0.2602	0.2431	0.2385	0.2335	0.2336	0.2319	**0.2302**	**31.68**
	LSCDA	0.2909	0.2669	0.2531	0.2460	0.2429	0.2400	0.2387	0.2373	52.89
	gKDR	0.2787	0.2688	0.2564	0.2494	0.2452	0.2424	0.2421	0.2399	60.45
500	ccaKDR	0.2959	0.2534	0.2413	0.2317	0.2284	0.2265	0.2251	**0.2247**	**35.03**
	LSCDA	0.2867	0.2592	0.2421	0.2393	0.2364	0.2339	0.2294	0.2295	178.94
	gKDR	0.2775	0.2662	0.2557	0.2477	0.2426	0.2377	0.2353	0.2324	106.07
1,000	ccaKDR	0.2962	0.2556	0.2398	0.2282	0.2253	0.2220	0.2197	**0.2198**†	**44.89**
	LSCDA	0.2841	0.2565	0.2383	0.2332	0.2298	0.2255	0.2242	0.2237	253.36

Table 6: Label prediction error (extended)

N	Method	9	10	11	12	13	14	15	16	Time
		Usps								
250	gKDR	0.2930	0.2650	0.2476	0.2232	0.2143	0.2070	0.1949	0.1847	**59.03**
	ccaKDR	0.1626	0.1524	0.1541	0.1538	0.1554	0.1565	0.1538	**0.1529**	84.90
500	gKDR	0.2273	0.2042	0.1882	0.1691	0.1573	0.1499	0.1382	0.1308	**94.22**
	ccaKDR	0.1321	0.1248	0.1256	0.1205	0.1244	0.1197	0.1162	**0.1144**	101.39
1,000	gKDR	0.1814	0.1635	0.1481	0.1352	0.1261	0.1151	0.1098	0.1074	166.96
	ccaKDR	0.1155	0.1094	0.1101	0.1082	0.1049	0.0983	0.0985	**0.0964**†	**139.84**
		Isolet								
250	gKDR	0.5784	0.5436	0.5147	0.4861	0.4495	0.4266	0.4009	0.3799	**141.96**
	ccaKDR	0.2750	0.2551	0.2490	0.2501	0.2362	0.2262	0.2246	**0.2179**	173.68
500	gKDR	0.4579	0.4167	0.3806	0.3492	0.3151	0.2893	0.2635	0.2380	259.01
	ccaKDR	0.2248	0.2125	0.2021	0.1876	0.1833	0.1627	0.1595	**0.1567**	**225.67**
1,000	gKDR	0.3628	0.3210	0.2872	0.2591	0.2339	0.2030	0.1790	0.1633	503.59
	ccaKDR	0.1898	0.1728	0.1654	0.1620	0.1477	0.1396	0.1271	**0.1232**	**343.43**
2,000	gKDR	0.3119	0.2713	0.2376	0.2088	0.1890	0.1662	0.1538	0.1378	1007.27
	ccaKDR	0.1721	0.1586	0.1445	0.1408	0.1316	0.1319	0.1263	**0.1158**†	**569.24**

explored by ccaKDR is still confined by the heuristics to avoid overexploration; thus, devising a better strategy will be of practical significances.

Acknowledgements

Authors report no conflict of interest. The authors would like to thank the editor and reviewers for their insightful comments, which substantially improved the paper. CY Tao is supported by the China Scholarship Council (CSC) and National Natural Science Foundation of China (No. 11101429 and No. 11471081). JF Feng is also partially supported by the National High Technology Research and Development Program of China (No. 2015AA020507) and the Key Project of Shanghai Science & Technology Innovation Plan (No. 15JC1400101). The research was partially supported by the National Centre for Mathematics and Interdisciplinary Sciences (NCMIS) of the Chinese Academy of Sciences and Key Program of the National Natural Science Foundation of China (No. 91230201). The authors would also like to thank Prof. D Waxman for his advices and both Prof. CL Leng, Dr. L Zhao for fruitful discussions.

Appendix A. Assumptions

(i) $\kappa_{\mathcal{X}}$ and $\kappa_{\mathcal{Y}}$ are measurable, and $\mathbb{E}[\kappa_{\mathcal{X}}(X,X)] < \infty$, $\mathbb{E}[\kappa_{\mathcal{Y}}(Y,Y)] < \infty$.

(ii) $\mathbb{E}[g|X = \cdot] \in \mathcal{H}_{\mathcal{X}}$ for any $g \in \mathcal{H}_Y$.

(iii) $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$ are separable.

(iv) $f_h(\boldsymbol{z})$ is is differentiable *w.r.t.* $\boldsymbol{z}$.

(v) C_{XX} is injective.

Appendix B. Proof of Theorem 1

Lemma 1 below tells us the conditional expectation of $g \in \mathcal{H}_Y$ given X, which is needed to establish Theorem 1.

Lemma 1 (Fukumizu et al. (2004), Theorem 1)**.** *Let* $(\mathcal{H}_\mathcal{X}, \kappa_\mathcal{X})$ *and* $(\mathcal{H}_\mathcal{Y}, \kappa_\mathcal{Y})$ *be RKHS on measurable spaces* Ω_1 *and* Ω_2*, respectively, and* (X, Y) *be a random vector on* $\Omega_\mathcal{X} \times \Omega_\mathcal{Y}$*. Under the assumptions* (i, ii) *in the Appendix, for all* $g \in \mathcal{H}_\mathcal{Y}$ *we have*

$$C_{XX}\mathbb{E}_{Y|X}[g(Y)|X = \cdot] = C_{XY}g. \tag{B.1}$$

If C_{XX} is injective, for example when κ_X is a continuous kernel on topological space $\mathcal{X}$ and $\mathbb{P}_X$ is a Borel probability measure such that $\mathbb{P}(U) > 0$ for any open set U in $\mathcal{X}$, then the result above can be expressed as

$$\mathbb{E}_{Y|X}[g(Y)|X = \cdot] = C_{XX}^{-1}C_{XY}g. \tag{B.2}$$

The assumption $\mathbb{E}_{Y|X}[\kappa(\cdot, \boldsymbol{y})|X = \cdot] \in \mathcal{H}_X$, however, may not hold in general. We remark that $C_{XX}^{-1}C_{XY}g$ can still be understood in the sense of approximation. Let $(\widehat{C}_{XX}^{(n)} + \epsilon_n \boldsymbol{I}_n)^{-1} = (\boldsymbol{K}_{XX} + \epsilon_n \boldsymbol{I}_n)^{-1}$ and $C_{XY}^{(n)} = \boldsymbol{K}_{XY}$ be the regularized empirical estimator of C_{XX}^{-1} and C_{XY} respectively, then

$$(\widehat{C}_{XX}^{(n)} + \epsilon_n \boldsymbol{I}_n)^{-1}C_{XY}^{(n)}g$$

is an consistent estimator of $\mathbb{E}[g(Y)|X = \cdot]$ in $\mathcal{L}^2(X)$ given an appropriate sequence of $\{\epsilon_n\}$, even if $\mathbb{E}[g(Y)|X = \cdot] \notin \mathcal{H}_X$ (Fukumizu and Leng, 2012). This is to say although $\mathbb{E}[g(Y)|X = \cdot] \notin \mathcal{H}_X$, it can be approximated by a sequence of functions in $\mathcal{H}_X$.

We now present the proof of Theorem 1 as follows.

Proof. Similar to the multivariate canonical correlation analysis, the following facts can be established:

$$\begin{aligned}
\langle f_i, C_{XY}g_j\rangle_{\mathcal{H}_\mathcal{X}} &= \mathbb{E}f_i(X)g_j(Y) = \rho_i\delta_{ij},\\
\langle f_i, C_{XX}f_j\rangle_{\mathcal{H}_\mathcal{X}} &= \mathbb{E}f_i(X)f_j(X) = \delta_{ij},\\
\langle C_{XY}g_i, f\rangle_{\mathcal{H}_\mathcal{X}} &= \rho_i\langle C_{XX}f_i, f\rangle_{\mathcal{H}_\mathcal{X}}.
\end{aligned}$$

For any $g \in \mathcal{H}_\mathcal{Y}$, let $\alpha_k = \langle C_{YY} g, g_k \rangle_{\mathcal{H}_\mathcal{Y}}$ and $g_\perp = g - \sum_k \alpha_k g_k$, we first prove $C_{XY} g = C_{XX}(\sum_{h \in \mathcal{I}} \alpha_h \rho_k f_h)$. Let $\Delta g = C_{XY} g - C_{XX}(\sum_{h \in \mathcal{I}} \alpha_h \rho_h f_h) \in \mathcal{H}_X$, we know

$$\langle \Delta g, f \rangle_{\mathcal{H}_X} = \langle C_{XY} g_\perp, f \rangle_{\mathcal{H}_X} = 0$$

holds for all $f \in \mathcal{H}_X$. This implies

$$\|\Delta g\|^2_{\mathcal{H}_X} = \langle \Delta g, \Delta g \rangle_{\mathcal{H}_X} = 0,$$

i.e. Δg is the null element in $\mathcal{H}_X$. This proves $C_{XY} g = C_{XX}(\sum_{k \in \mathcal{I}} \alpha_h \rho_h f_h)$. Plugin this into (B.2) gives

$$\begin{aligned}
\nabla \mathbb{E}[g|\boldsymbol{x}] &= \nabla \left\langle C_{XX}^{-1} C_{XY} g, \kappa_\mathcal{X}(\cdot, \boldsymbol{x}) \right\rangle_{\mathcal{H}_\mathcal{X}} \\
&= \nabla \langle C_{XX}^{-1} C_{XX} (\sum_{h \in \mathcal{I}} \alpha_h \rho_h f_h), \kappa_\mathcal{X}(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}_\mathcal{X}} \\
&= \sum_{h \in \mathcal{I}} \alpha_h \rho_h \nabla \left\langle f_h, \kappa_\mathcal{X}(\cdot, \boldsymbol{x}) \right\rangle_{\mathcal{H}_\mathcal{X}} \\
&= \sum_{h \in \mathcal{I}} \alpha_h \rho_h \nabla f_h(\boldsymbol{x}).
\end{aligned}$$

This implies $\nabla_{\boldsymbol{v}} \mathbb{E}[g|\boldsymbol{x}] = \mathbf{0}$ almost surely where $\boldsymbol{v} = \boldsymbol{B}_\perp^\top \boldsymbol{x}$, here $\boldsymbol{B}_\perp$ is the orthonormal bases orthogonal to $\boldsymbol{B}$. Denote $\mu_{Y|X=\boldsymbol{x}}$ the kernel embedding of conditional $Y|X = \boldsymbol{x}$ with the characteristic kernel $\kappa_\mathcal{Y}$, then $\mathbb{E}[g|\boldsymbol{x}] = \langle g, \mu_{Y|X=\boldsymbol{x}} \rangle_{\mathcal{H}_Y}$. The result above suggests $\mu_{Y|X=\boldsymbol{x}}$ only depends on $\boldsymbol{B}^\top X$, thus proving the conditional independency. □

We remark that our proof differs from previous works, where the conditional embedding $\mu_{Y|X=\boldsymbol{x}}$ was explicitly expressed as $C_{YX} C_{XX}^{-1} \kappa_X(\cdot, \boldsymbol{x})$. The existence of element $C_{XX}^{-1} \kappa_X(\cdot, \boldsymbol{x})$, however, can not be guaranteed (Song et al., 2009; Fukumizu et al., 2013). Our proof only conerns the conditional expectation and does not require the analytical expression of $\mu_{Y|X=\boldsymbol{x}}$, therefore circumventing this technical difficulty.

Appendix C. Linear scaling algorithms

Algorithm 3 and Algorithm 4 present pseudo code for the linear scaling ccaKDR and the Ritz approximation it uses. Here $\mathcal{S}_d(\mathbb{R}^p)$ denotes the (p,d)-Stiefel manifold, *i.e.* $\boldsymbol{B} \in \mathcal{S}_d(\mathbb{R}^p)$ if $\boldsymbol{B} \in \mathbb{R}^{p\times d}$ and $\boldsymbol{B}^\top \boldsymbol{B} = \boldsymbol{I}_d$.

Algorithm 3: Linear-scaling ccaKDR

Data: $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, $\boldsymbol{Y} \in \mathbb{R}^{n\times q}$, $\kappa_{\mathcal{X}}$, $\kappa_{\mathcal{Y}}$

Preprocessing: Compute $\boldsymbol{L}_t$ such that $\boldsymbol{K}_t \approx \boldsymbol{L}_t' \boldsymbol{L}_t$ for $t \in \{x, y\}$

Input: $\boldsymbol{L}_x \in \mathbb{R}^{n\times r_x}, \boldsymbol{L}_y \in \mathbb{R}^{n\times r_y}, \epsilon_x, \epsilon_y, l, d$

Output: $\boldsymbol{B}^{l,d} \in \mathcal{S}_d(\mathbb{R}^p)$

Set $\boldsymbol{L}_t^c \coloneqq \boldsymbol{L}_t - \frac{1}{N}\mathbf{1}\mathbf{1}'\boldsymbol{L}_t$ for $t \in \{x, y\}$

$[\boldsymbol{U}_t, \boldsymbol{S}_t] \coloneqq \mathrm{SVD}(\boldsymbol{L}_t^c)$, $\boldsymbol{\Lambda}_t \coloneqq \boldsymbol{S}_t^2$, $\boldsymbol{R}_t \coloneqq (\boldsymbol{\Lambda}_t + \epsilon_t \boldsymbol{I})^{-1}\boldsymbol{\Lambda}_t$

$\boldsymbol{R}_{xy} \coloneqq \boldsymbol{R}_x \boldsymbol{U}_x^\top \boldsymbol{U}_y \boldsymbol{R}_y$, $\mathbf{R} \coloneqq [\boldsymbol{I}_{r_x}, \boldsymbol{R}_{xy}; \boldsymbol{R}_{xy}^\top, \boldsymbol{I}_{r_y}]$

Solve for l leading canonical pairs $\{(\rho_h, \boldsymbol{\eta}_h)\}_{h=1}^l$ for $\boldsymbol{R}$

$\check{\boldsymbol{\xi}}_h \coloneqq \boldsymbol{\eta}_h(1 : r_x)$, $\check{\boldsymbol{\zeta}}_h \coloneqq \boldsymbol{\eta}_h(r_x + 1 : \mathrm{end})$

$\boldsymbol{\xi}_h = \boldsymbol{U}_x(\boldsymbol{\Lambda}_x + \epsilon_x \boldsymbol{I})^{-1}\check{\boldsymbol{\xi}}_h$, $\boldsymbol{\zeta}_h = \boldsymbol{U}_y(\boldsymbol{\Lambda}_y + \epsilon_y \boldsymbol{I})^{-1}\check{\boldsymbol{\zeta}}_h$

Set $\boldsymbol{W}_x \coloneqq [\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^l] \in \mathbb{R}^{n\times l}$

Compute $\boldsymbol{F} \in \mathbb{R}^{n\times r_x \times p}$ defined as $\boldsymbol{F}_{i,j,a} \coloneqq \boldsymbol{X}_{i,a}\boldsymbol{L}_{x,i,j}$

Set $\widetilde{\boldsymbol{L}}_x \coloneqq \boldsymbol{L}_x' \boldsymbol{W}_x \in \mathbb{R}^{r_x \times l}$

Compute $\boldsymbol{J} \in \mathbb{R}^{r_x\times p\times l}$ defined as $\boldsymbol{J}_{j,a,l'} \coloneqq \sum_{i=1}^n \boldsymbol{F}_{i,j,a}\boldsymbol{W}_{x,i,l'}$

Compute $\boldsymbol{G} \in \mathbb{R}^{n\times p\times l}$ defined as $\boldsymbol{G}_{i,a,l'} \coloneqq \sum_{j=1}^{r_x} \boldsymbol{L}_{x,i,j}\boldsymbol{J}_{j,a,l'}$

Compute $\boldsymbol{H} \in \mathbb{R}^{n\times p\times l}$ defined as $\boldsymbol{H}_{i,a,l'} \coloneqq \sum_{j=1}^{r_x} \boldsymbol{F}_{i,j,a}\widetilde{\boldsymbol{L}}_x(j, l')$

Set $\boldsymbol{T} \coloneqq \boldsymbol{G} - \boldsymbol{H}$

Compute $\boldsymbol{M}^{(l)} \in \mathbb{R}^{p\times p}$ defined as $\boldsymbol{M}^{(l)}(a, b) \coloneqq \sum_{l'=1}^l \sum_{i=1}^n \boldsymbol{T}_{i,a,l'}\boldsymbol{T}_{i,b,l'}$

Set $\boldsymbol{B}^{l,d}$ to the d leading eigenvectors of $\boldsymbol{M}^{(l)}$

Force $B^{l,d}$ to $\mathcal{S}_d(\mathbb{R}^p)$ if a complex projection is found

Appendix D. Complexity analysis

To simplify the discussion we fix the regularization parameter to be equal. The space complexity for storing the factorization L is $\mathcal{O}(nr)$. Calculate $\nabla_a f_{\boldsymbol{w}}(\boldsymbol{x}_{i'})$

Algorithm 4: Ritz approximation of leading eigenpairs

Input: $\boldsymbol{A} \in \mathbb{R}^{n\times n}$,$l$

Output: $\{(\rho_h, \boldsymbol{v}_h)\}_{h=1}^{l}$

Set $k \coloneqq \min(2l, l+10)$, randomly initialize $\boldsymbol{s}_1 \in \mathbb{R}^N$

for $\underline{k' = 2, \cdots, k}$ **do**
  $\boldsymbol{s}_{k'} \coloneqq \boldsymbol{A}\boldsymbol{s}_{k'-1}$

end for

Set $\boldsymbol{S} \coloneqq [\boldsymbol{s}_1, \cdots, \boldsymbol{s}_k]$ and $[\boldsymbol{Q}, \boldsymbol{R}] \coloneqq \mathrm{SVD}(\boldsymbol{S})$

Solve $\{(\rho_h, \widetilde{\boldsymbol{v}}_h)\}_{h=1}^{l}$ first l eigenpairs for $\boldsymbol{H} \coloneqq \boldsymbol{Q}'\boldsymbol{A}\boldsymbol{Q}$

Set $\boldsymbol{v}_h \coloneqq \boldsymbol{Q}\widetilde{\boldsymbol{v}}_h$

for $(i', a) \in \{1, \cdots, n\} \times \{1, \cdots, p\}$ is an $\mathcal{O}(nrp)$ operation. For l such weights $\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_l\}$, we denote the tensor $\widetilde{\Gamma} \in \mathbb{R}^{n\times p\times l}$ defined as $\widetilde{\Gamma}(i, a, s) \coloneqq \nabla_a f_{\boldsymbol{w}_s}(\boldsymbol{x}_i)$. Constructing $\widetilde{\Gamma}$ takes $\mathcal{O}(nrpl)$ operations. By Theorem 2, $\widetilde{\Gamma}$ contains all the information for the reconstruction of central space given $\{\boldsymbol{w}_s\}_{s=1}^{l}$ are the canonical vectors and $\lambda_{l+1} = 0$. Now let us look at the computation of eigenpairs $\{(\lambda_s, \boldsymbol{w}_s)\}_{s=1}^{l}$. Computing the Krylov space takes involves taking kernel matrix inversion, by utilizing Woodbury matrix inversion identity, it is an $\mathcal{O}(nr^2 + r^3) + \mathcal{O}(nr + r^2)$ operation. (The former being constructing $L^\top L$ and the later taking an inverse of r^2 matrix.) The $\mathcal{O}(nr^2)$ can be done as a preprocessing step, and for the computation of $\mathcal{O}(l)$-dimension Krylov space $\mathcal{O}(nrl)$ operations are performed. The QR step takes further $\mathcal{O}(nl^2)$ operations. Then $\mathcal{O}(nrl + nl^2)$ operations for the construction of Ritz matrix and $\mathcal{O}(l^3)$ for its eigen-decomposition. Then $\mathcal{O}(nl^2)$ operations are taken to approximate the original eigencomponent $\{\boldsymbol{w}_s\}$. For the computation of EDR space, note Fukumizu and Leng (2012) propose to first construct $M = \Gamma^\top\Gamma$, which takes $\mathcal{O}(nrp^2)$ operations and then decompose it with $\mathcal{O}(p^3)$ operations. For large p (e.g. $p \gg n$) it is not very economic to do so. We could similarly use a Ritz-approximation to further curtail the computational burden. For $d \ll p$, we only need $\mathcal{O}(nrpd)$ operations to construct the Krylov space, then $\mathcal{O}(pd^2)$ for QR, $\mathcal{O}(d^3)$ for the eigen-decomposition and $\mathcal{O}(pd^2)$ to get the projection in original space. Summing

together, it's $\mathcal{O}(nr^2)$ for the preprocessing and $\mathcal{O}(nrpl)$ for the CV loop, this gives $\mathcal{O}(n_\theta^2 nr^2 + n_\theta^3 n_c(nrp(l+d)+c))$ for the overall complexity, where c denote the computation cost for evaluating the loss function.

For the original gKDR, the overall time complexity is $\mathcal{O}(n_\theta^2 nr^2 + n_\theta^3 n_c(nr^2 p + nrp^2 + c))$. With the modified gKDR, the overall time complexity is $\mathcal{O}(n_\theta^2 nr^2 + n_\theta^3 n_c(nr^2 p + nrpd + c))$. Thus the proposed will significantly outperform the original gKDR implementation in terms of computation complexity with large p and small d. Even for the modified gKDR, ccaKDR still achieves considerable reduction in computation for more complex Y (with larger r_y) and smaller l.

References

Akaho, S., 2001. A kernel method for canonical correlation analysis. In: Proceedings of the International Meeting of the Psychometric Society (IMPS2001).

Amari, S.-I., 1998. Natural gradient works efficiently in learning. Neural Computation 10 (2), 251–276.

Aronszajn, N., 1950. Theory of reproducing kernels. Transactions of the American mathematical society, 337–404.

Bach, F., 2012. Sharp analysis of low-rank kernel matrix approximations. arXiv preprint arXiv:1208.2015/ IN proceedings, COLT 2013.

Bach, F. R., Jordan, M. I., 2003. Kernel independent component analysis. The Journal of Machine Learning Research 3, 1–48.

Cook, R. D., 1994. On the interpretation of regression plots. Journal of the American Statistical Association 89 (425), 177–189.

Cook, R. D., Li, B., 2002. Dimension reduction for conditional mean in regression. Annals of Statistics, 455–474.

Cook, R. D., Ni, L., 2005. Sufficient dimension reduction via inverse regression. Journal of the American Statistical Association 100 (470).

Cook, R. D., Weisberg, S., 1991. Comment. Journal of the American Statistical Association 86 (414), 328–332.

Cover, T. M., Thomas, J. A., 2012. Elements of information theory. John Wiley & Sons.

Darbellay, G. A., Vajda, I., et al., 1999. Estimation of the information by an adaptive partitioning of the observation space. IEEE Transactions on Information Theory 45 (4), 1315–1321.

Drineas, P., Mahoney, M. W., 2005. On the Nyström method for approximating a gram matrix for improved kernel-based learning. The Journal of Machine Learning Research 6, 2153–2175.

Fan, J., Lv, J., 2008. Sure independence screening for ultrahigh dimensional feature space. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70 (5), 849–911.

Fukumizu, K., Bach, F. R., Gretton, A., 2007. Statistical consistency of kernel canonical correlation analysis. The Journal of Machine Learning Research 8, 361–383.

Fukumizu, K., Bach, F. R., Jordan, M. I., 2004. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. The Journal of Machine Learning Research 5, 73–99.

Fukumizu, K., Bach, F. R., Jordan, M. I., 2009. Kernel dimension reduction in regression. The Annals of Statistics, 1871–1905.

Fukumizu, K., Leng, C., 2012. Gradient-based kernel method for feature extraction and variable selection. In: Advances in Neural Information Processing Systems. pp. 2114–2122.

Fukumizu, K., Song, L., Gretton, A., 2013. Kernel Bayes' rule: Bayesian inference with positive definite kernels. Journal of Machine Learning Research 14 (1), 3753–3783.

Fung, W. K., He, X., Liu, L., Shi, P., 2002. Dimension reduction based on canonical correlation. Statistica Sinica 12 (4), 1093–1113.

Golub, G. H., Van Loan, C. F., 2012. Matrix computations. Vol. 3. JHU Press.

Gretton, A., Herbrich, R., Smola, A., Bousquet, O., Schölkopf, B., 2005. Kernel methods for measuring independence. The Journal of Machine Learning Research 6, 2075–2129.

Härdle, W., Stoker, T. M., 1989. Investigating smooth multiple regression by the method of average derivatives. Journal of the American Statistical Association 84 (408), 986–995.

Hardoon, D. R., Szedmak, S., Shawe-Taylor, J., 2004. Canonical correlation analysis: An overview with application to learning methods. Neural Computation 16 (12), 2639–2664.

Karasuyama, M., Sugiyama, M., 2012. Canonical dependency analysis based on squared-loss mutual information. Neural networks 34, 46–55.

Kimeldorf, G. S., Wahba, G., 1970. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. Annals of Mathematical Statistics 41 (2), 495–502.

Kraskov, A., Stögbauer, H., Grassberger, P., 2004. Estimating mutual information. Physical review E 69 (6), 066138.

Kumar, S., Mohri, M., Talwalkar, A., 2012. Sampling methods for the Nyström method. The Journal of Machine Learning Research 13 (1), 981–1006.

Li, B., Dong, Y., 2009. Dimension reduction for nonelliptically distributed predictors. The Annals of Statistics, 1272–1298.

Li, K.-C., 1991. Sliced inverse regression for dimension reduction. Journal of the American Statistical Association 86 (414), 316–327.

Li, K.-C., 1992. On principal hessian directions for data visualization and dimension reduction: another application of stein's lemma. Journal of the American Statistical Association 87 (420), 1025–1039.

Lichman, M., 2013. UCI machine learning repository.
URL http://archive.ics.uci.edu/ml

Ma, Y., Zhu, L., 2013. Efficient estimation in sufficient dimension reduction. Annals of statistics 41 (1), 250.

Saad, Y., 1992. Numerical methods for large eigenvalue problems. Vol. 158. SIAM.

Samarov, A. M., 1993. Exploring regression structure using nonparametric functional estimation. Journal of the American Statistical Association 88 (423), 836–847.

Shawe-Taylor, J., Cristianini, N., 2004. Kernel methods for pattern analysis. Cambridge university press.

Song, L., Huang, J., Smola, A., Fukumizu, K., 2009. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In: International Conference on Machine Learning. pp. 961–968.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., Lanckriet, G. R., 2010. Hilbert space embeddings and metrics on probability measures. The Journal of Machine Learning Research 11, 1517–1561.

Sugiyama, M., Suzuki, T., Kanamori, T., 2012. Density ratio estimation in machine learning. Cambridge University Press.

Suzuki, T., Sugiyama, M., 2013. Sufficient dimension reduction via squared-loss mutual information estimation. Neural Computation 25 (3), 725–758.

Suzuki, T., Sugiyama, M., Sese, J., Kanamori, T., 2008. Approximating mutual information by maximum likelihood density ratio estimation. FSDM 4, 5–20.

Tangkaratt, V., Sasaki, H., Sugiyama, M., 2015. Direct estimation of the derivative of quadratic mutual information with application in supervised dimension reduction. arXiv preprint arXiv:1508.01019.

Torkkola, K., 2003. Feature extraction by non parametric mutual information maximization. The Journal of Machine Learning Research 3, 1415–1438.

Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I., 2011. Mulan: A java library for multi-label learning. Journal of Machine Learning Research 12, 2411–2414.

Van Hulle, M. M., 2005. Multivariate edgeworth-based entropy estimation. Machine Learning for Signal Processing 2005, 311–316.

Wang, H., Xia, Y., 2008. Sliced regression for dimension reduction. Journal of the American Statistical Association 103 (482), 811–821.

Wang, T., Guo, X., Zhu, L., Xu, P., 2014. Transformed sufficient dimension reduction. Biometrika 101 (4), 815–829.
URL http://biomet.oxfordjournals.org/content/101/4/815.abstract

Widom, H., 1963. Asymptotic behavior of the eigenvalues of certain integral equations. Transactions of the American Mathematical Society, 278–295.

Widom, H., 1964. Asymptotic behavior of the eigenvalues of certain integral equations. ii. Archive for Rational Mechanics and Analysis 17 (3), 215–229.

Xia, Y., Tong, H., Li, W., Zhu, L.-X., 2002. An adaptive estimation of dimension reduction space. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64 (3), 363–410.

Yin, X., Cook, R. D., 2005. Direction estimation in single-index regressions. Biometrika 92 (2), 371–384.

Yin, X., Li, B., Cook, R. D., 2008. Successive direction extraction for estimating the central subspace in a multiple-index regression. Journal of Multivariate Analysis 99 (8), 1733–1757.

Zeng, P., Zhu, Y., 2010. An integral transform method for estimating the central mean and central subspaces. Journal of Multivariate Analysis 101 (1), 271–290.

Zhu, Y., Zeng, P., 2006. Fourier methods for estimating the central subspace and the central mean subspace in regression. Journal of the American Statistical Association 101 (476), 1638–1651.