A reproducibility study of "Supervised PCA: A Multiobjective Approach"

Bobo Bai[1] **Shanchen Liu**[2] **Peng Zhai**[1]

[1]Department of Earth and Environmental Sciences, University of Michigan, Ann Arbor
[2]Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor

1 Introduction and motivation

Supervised principal component analysis (SPCA) learns a low-dimensional data representation for principal component analysis (PCA) that is effective for supervised learning tasks. Despite extensive study, prior SPCA methods have primarily focused on minimizing prediction error, overlooking the importance of maximizing variance explained, which can lead to poor performance.

This project studies a multi-objective SPCA approach proposed in [1], which jointly optimizes variance explained and prediction accuracy, balancing these competing goals. We aim to evaluate whether this method offers a more interpretable, predictive low-dimensional representation for supervised learning tasks. To achieve this, we compare the novel SPCA approach with Barshan's method [2] as a baseline method, conducting experiments on six datasets across both regression and classification settings.

2 Problem statement

Let $\mathcal{X} \in \mathbb{R}^p$ and $\mathcal{Y} \in \mathbb{R}^q$ be jointly distributed random variables. The variable $\mathcal{Y}$ is assumed to be continuous for regression, and one-hot for classification. Consider n i.i.d. realizations $(\boldsymbol{x}_1, \boldsymbol{y}_1)$,..., $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ of $(\mathcal{X}, \mathcal{Y})$. Let $X := [\boldsymbol{x}_1, ..., \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times p}$ be the data matrix and $Y := [\boldsymbol{y}_1, ..., \boldsymbol{y}_n]^T \in \mathbb{R}^{n \times q}$ be the response matrix. Given X and Y, our objective is to find an orthogonal projection matrix $L \in \mathbb{R}^{p \times r}$ for $\mathcal{X}$, and a prediction function $f(L^T\mathcal{X}; \beta) : \mathbb{R}^r \to \mathbb{R}^q$ with a coefficient matrix $\beta \in \mathbb{R}^{r \times q}$, such that the extracted feature $L^T\mathcal{X}$ captures the variability in $\mathcal{X}$ while allowing f to accurately predict $\mathcal{Y}$.

3 Related work

The earliest SPCA approach, proposed by Bair [3], is similar to conventional PCA but includes a preliminary step of feature selection based on univariate standard regression coefficients. Specifically, only the features with the highest dependence on the label will be selected for PCA. Therefore, the learned principal components can be better applied for prediction purposes. However, Bair's method is restricted to univariate regression and binary prediction. A recent work [4] elaborates on Bair's method by iteratively applying PCA and recalculating feature selection based on supervised criteria. This iterative supervised principal component analysis (ISPCA) extended the application of Bair's method to multi-class classification. Another method, refered to as Barshan's method [2], incorporates the Hilbert-Schmidt Independence Criterion (HSIC) to measure the dependence between the data and target response in reproducing kernel Hilbert spaces (RKHS). Barshan's method aims to maximize an empirical measure of the HSIC, and similar to PCA, it uses a trace maximization formulation. Barshan's method has been adapted for sparse PCA [5].

4 Method

The method in [1] advances the state of the art in SPCA through a statistical formulation. It assumes the following generative model

$$\mathcal{X} \sim \mathcal{N}(0, \sigma_x^2 I_p + \alpha L L^T), \;\; \mathcal{Y}|\mathcal{X} \sim P_{\mathcal{Y}|\mathcal{X}} \tag{1}$$

where $\alpha > 0$ and $P_{\mathcal{Y}|\mathcal{X}}$ is parameterized by the orthogonal matrix L, the coefficient matrix β and some additional parameter θ. For liner regression, the conditional distribution is given by

$$\mathcal{Y}|\mathcal{X} = \boldsymbol{x} \sim \mathcal{N}(\beta^T L^T \boldsymbol{x}, \sigma_y^2 I_q) \tag{2}$$

For logistic regression, the conditional probability is

$$P(\mathcal{Y}|\mathcal{X} = \boldsymbol{x}) = \frac{\exp(\boldsymbol{x}^T L \boldsymbol{\beta}_{\mathcal{Y}})}{\sum_{j=1}^{q} \exp(\boldsymbol{x}^T L \boldsymbol{\beta}_j)} \tag{3}$$

where $\boldsymbol{\beta}_j$ is the j-th column of β and $\boldsymbol{\beta}_{\mathcal{Y}}$ is the column of β corresponding to the class given by $\mathcal{Y}$.

The negative log-likelihood (NLL) of the data (X, Y) is

$$G(L, \beta, \alpha, \sigma_x, \theta; X, Y) := -\sum_{i=1}^{n} \log P_{\mathcal{Y}|\mathcal{X}}(\boldsymbol{y}_i|\boldsymbol{x}_i; L, \beta, \theta) - \sum_{i=1}^{n} \log P_{\mathcal{X}}(\boldsymbol{x}_i; L, \alpha, \sigma_x) \tag{4}$$

For linear regression, ignoring additive constants and scaling the NLL by $2\sigma_y^2$, the scaled NLL becomes

$$G_{LS} := \|Y - XL\beta\|_F^2 + \lambda_{LS} \|X - \gamma X L L^T\|_F^2 \tag{5}$$

where $\lambda_{LS} = \frac{\sigma_y^2}{\sigma_x^2}$ and $\gamma = 1 - \sqrt{\frac{\sigma_x^2}{\sigma_x^2 + \alpha}}$.

For logistic regression, ignoring the additive constants, the NLL becomes

$$G_{LR} := -\sum_{i=1}^{n} \sum_{j=1}^{q} y_{ij} \log \frac{\exp(\boldsymbol{x}_i^T L \boldsymbol{\beta}_j)}{\sum_{j'=1}^{q} \exp(\boldsymbol{x}_i^T L \boldsymbol{\beta}_{j'})} + \lambda_{LR} \|X - \gamma X L L^T\|_F^2 \tag{6}$$

where y_{ij} is the j-th entry of $\boldsymbol{y}_i$ (or equivalently, the (i, j)-th entry of Y), $\lambda_{LR} = \frac{1}{2\sigma_x^2}$, and γ is defined as above.

In both G_{LS} and G_{LR}, the first term represents the the prediction loss (L_2 loss for G_{LS} and cross-entropy loss for G_{LR}), while the second term is a regularization term to increase the variability explained by XL. Thus, this method may be interpreted as a form of regularization that shinks the optimal L toward the PCA solution.

The overall optimization problem becomes

$$\min_{\substack{L \in \mathbb{R}^{p \times r},\ \beta \in \mathbb{R}^{r \times q}, \\ \lambda > 0, \gamma \in (0,1)}} \; G(L, \beta, \lambda, \gamma; X, Y) \;\; s.t. \;\; L^T L = I_r. \tag{7}$$

In the following sections, we will refer to the least square regression PCA as LRPCA and the logistic regression PCA as LRPCA. Following the convention in [1], we refer to λ and γ as nuisance parameters because, although they are not of primary interest, they must be accounted for to estimate L and β.

The Euclidean gradients of G with respect to L are

$$\nabla_L G_{LS} = -2X^T (Y - XL\beta) \beta^T - 2\lambda_{LS}\gamma(2 - \gamma) X^T X L \tag{8}$$

$$\nabla_L G_{LR} = -X^T (Y - M) \beta^T - 2\lambda_{LR}\gamma(2 - \gamma) X^T X L \tag{9}$$

where $M \in \mathbb{R}^{n \times q}$ is defined element-wise as:

$$M_{ij} = \frac{\exp(\boldsymbol{x}_i^T L \boldsymbol{\beta}_j)}{\sum_{j'=1}^{q} \exp(\boldsymbol{x}_i^T L \boldsymbol{\beta}_{j'})} \tag{10}$$

The detailed derivation of the Euclidean gradients can be found in the appendix.

In addition to the LSPCA and LRPCA methods, we will perform Barshan's method and its kernel version(kBarshan) as baseline methods for comparison. Due to space constraints, the illustration of Barhsan's method and the derivation of kBarshan's method are provided in the appendix.

5 Algorithm

We solve the optimization problems in LSPCA and LRPCA using an alternating optimization approach, where the method iteratively updates L, β, as shown in Algorithm 1.

A. Hyperparameter selection

There are three hyperparameters, r, λ, γ. The value r will either be set to 2 or determined by a 10-fold cross validation (CV). The nuisance parameters λ and γ will be updated iteratively by MLE (see Eq. (11)-(13) in [1]), or chosen by a 10-fold CV. From (13) and (14), we see that G_{LS} and G_{LR} depend on the nuisance parameters λ and γ only through the term $\lambda\gamma(2-\gamma)$. Therefore, performing cross-validation over (λ, γ) is equivalent to cross validating λ alone while fixing $\gamma = 1$.

B. The β subproblem

The β subproblem is convex and unconstrained for both squared error and logistic losses. For implementation, we use the backslash operator for LSPCA and built in logistic regression function for LRPCA.

C. The L subproblem

The L subproblem involves optimization problem over the Stiefel mannifold, i..e, $L^T L = I_r$. Since solving the consrained problem on the Riemannian manifold is beyond the scope of this lecture, we follow the original paper [1] and use the manifold conjugate gradient (MCG) methods in Manopt [6] to solve the L subproblem.

Algorithm 1 LSPCA/LRPCA Alternating Algorithm

Input: $X \in \mathbb{R}^{n\times p}$: data matrix, $Y \in \mathbb{R}^{n\times q}$: response matrix, $L_0 \in \mathbb{R}^{p\times r}$: initial orthogonal matrix, nuisance parameter λ (if doing CV), nuisance parameter $\gamma = 1$, $l_{CV} \in \{0, 1\}$: 0 if choose nuisance parameters by MLE; 1 if choose nuisance parameters by CV.
Output: $Z^* \in \mathbb{R}^{n\times r}$: reduced data matrix, $\beta^* \in \mathbb{R}^{r\times q}$: coefficient matrix, $L^* \in \mathbb{R}^{p\times r}$: optimal orthogonal matrix.

procedure ALTERNATING_ALGORITHM($X, Y, L_0, \lambda, \gamma, l_{CV}$)
  if LSPCA **then**
    $\beta_0 \leftarrow (XL_0)^+ Y$
  else if LRPCA **then**
    $\beta_0 \leftarrow$ SOLVELR(XL_0, Y) ▷ Solve logistic regression
  end if
  $k \leftarrow 0$
  repeat
    if $l_{CV} == 0$ **then** ▷ Use MLE to calculate λ, γ
      $\gamma, \lambda \leftarrow$ UPDATEPARAMS$(X, Y, L_{k-1}, \beta_{k-1}, \gamma)$
    end if
    $L_k \leftarrow$ MCG$(G(L, \beta_{k-1}, \lambda, \gamma; X, Y), L_{k-1})$ ▷ Solve L using MCG
    if LSPCA **then**
      $\beta_k \leftarrow (XL_k)^+ Y$ ▷ Solve linear regression
    else if LRPCA **then**
      $\beta_k \leftarrow$ SOLVELR(XL_k, Y) ▷ Solve logistic regression
    end if
    $k \leftarrow k + 1$
  until Convergence
  $Z \leftarrow XL_k$
  return Z, β_k, L_k
end procedure

6 Experiments

We performs all the methods (LSPCA, LRPCA, Barshan's method and kBarshan's method) on six datasets listed in Table 1. All datasets except for the Barshan A are taken from University of California, Irvine machine learning repository(UCI) or the Arizona State feature selection repository(ASU). The

Barshan A dataset is a synthetic dataset by the gernerative model in Eq(10) in [2]. For the Music dataset, we uniformly subsampled 100 observations for the experiments to obtain a regression dataset in the $n < p$ setting following the study of [1].

Table 1: Datasets

Name	Type	q	n	p	source	Name	Type	q	n	p	source
Residential	regr.	2	372	103	UCI	Ionosphere	class.	2	354	34	UCI
Barshan A	regr.	1	100	4	ref[2]	Colon	class.	2	208	60	ASU
Music	regr.	2	100(1059)	116	UCI	Arcene	class.	2	200	10000	ASU

For each dataset, 20% of the data was randomly selected as an independent test set. For methods requiring parameter tuning, excluding those using maximum likelihood updates, the remaining 80% of the data was utilized in a 10-fold cross-validation (CV) procedure. Subsequently, all methods were trained on the full 80% of the data using the parameter set that resulted in the smallest CV error. This entire process, including test set selection, was repeated 10 times to generate the results shown in Table 2.

10-fold CV for r is performed over the range 2 to 10 (except for the Barshan A, where the number of feature is 4). 10-fold CV for λ is conducted over the set $\{0.001, 0.01, 0.1, 1, 10\}$. For Barshan's method and kBarshan's method, when using a radial basis function (RBF) kernel, the bandwidth parameter is selected from $\{0.1, 1, 10, 100, 1000\}$ via 10-fold CV.

6.1 Prediction Performance

We evaluate all methods by fixing $r = 2$ and by choosing $r \geq 2$ via 10-fold CV. This process is repeated 10 times, and the prediction errors (MSE for linear regression and error rate for logistic regression) are then averaged to produce the entries in Table 2.

First, we compare our results with those reported in the original paper [1]. For linear regression, the average MSEs generally do not match the results reported in [1]. This could arise from (1) the differences in normalization procedures, or (2) differences in test data selections. For logistic regression, our classification error rate generally match the results in [1], indicating that we we successfully reproduce the classification experiments results.

Second, we compare the errors from different methods within this work. For linear regression, LSPCA(CV) typicallly outperforms other methods. For example, the average MSE from the LSPCA(CV)+r(CV) method for the Residential data is 0.0237, significantly smaller than those generated by other methods. Conversely, LSPCA(MLE) generally perform the worst among all methods. For classification, the Barshan's and kBarshan's method often achieve the lowest error rate. Besides, LRPCA(CV) often outperforms its MLE counterpart. For both regression and classification tasks, doing CV on r almost always perform better than fixing $r = 2$.

Table 2: Comparison of average MSE (regression) or error rate (classification) of competing methods with standard error (σ). Subspace dimension($r = 2$) was held fixed for results in the first column of each dataset. For results in the second column, subspace dimension was chosen by 10-fold CV. Kernel SPCA methods are listed in bold. For each experiment, the best linear method is shown in red. Blue color indicates the kBarshan method outperforms other linear methods for that dataset.

Regression	Residential		Barshan A		Music	
	$r = 2$	r(CV)	$r = 2$	r(CV)	$r = 2$	r(CV)
Barshan	0.2155 ± 0.0896	0.0875 ± 0.0266	0.4249 ± 0.4453	0.4270 ± 0.4477	0.8347 ± 0.2826	0.7933 ± 0.2907
LSPCA(MLE)	0.5376 ± 0.1779	0.1618 ± 0.0471	0.4214 ± 0.4181	0.4216 ± 0.4183	0.9031 ± 0.2683	0.8086 ± 0.2419
LSPCA(CV)	0.1033 ± 0.0433	0.0237 ± 0.0150	0.4214 ± 0.4195	0.4213 ± 0.4195	0.8757 ± 0.2901	0.8075 ± 0.2416
kBarshan	0.2151 ± 0.0894	0.0880 ± 0.0274	0.4235 ± 0.4582	0.4634 ± 0.7993	0.8686 ± 0.3031	0.8156 ± 0.3031
Classification	Ionosphere		Colon		Arcene	
	$r = 2$	r(CV)	$r = 2$	r(CV)	$r = 2$	r(CV)
Barshan	0.1900 ± 0.0357	0.1357 ± 0.0226	0.1917 ± 0.1472	0.1917 ± 0.1472	0.2975 ± 0.0777	0.2700 ± 0.0743
LRPCA(MLE)	0.3143 ± 0.0294	0.1157 ± 0.0333	0.6417 ± 0.1472	0.1833 ± 0.1405	0.3400 ± 0.0568	0.3075 ± 0.0746
LRPCA(CV)	0.1729 ± 0.1018	0.1114 ± 0.0315	0.4667 ± 0.1809	0.1583 ± 0.0998	0.3350 ± 0.0543	0.3050 ± 0.0789
kBarshan	0.1300 ± 0.0412	0.1014 ± 0.0557	0.2250 ± 0.1574	0.2750 ± 0.1419	0.3375 ± 0.0358	0.3000 ± 0.0565

6.2 Interpertability

We also compared the explained variance of all the methods (Fig. 1). Generally, there is a trade-off between maximizing the variance explained and minimizing the prediction error. The novel methods LSPCA(CV) or LRPCA(CV) often maintain a balance by increasing explained variance while maintaining a small prediction error. On the other hand, Barshan's and kBarshan's methods usually leads to lower prediction errors, especially in classification tasks, e.g., Ionosphere and Colon, but they may explain less variance compared to LSPCA or LRPCA.

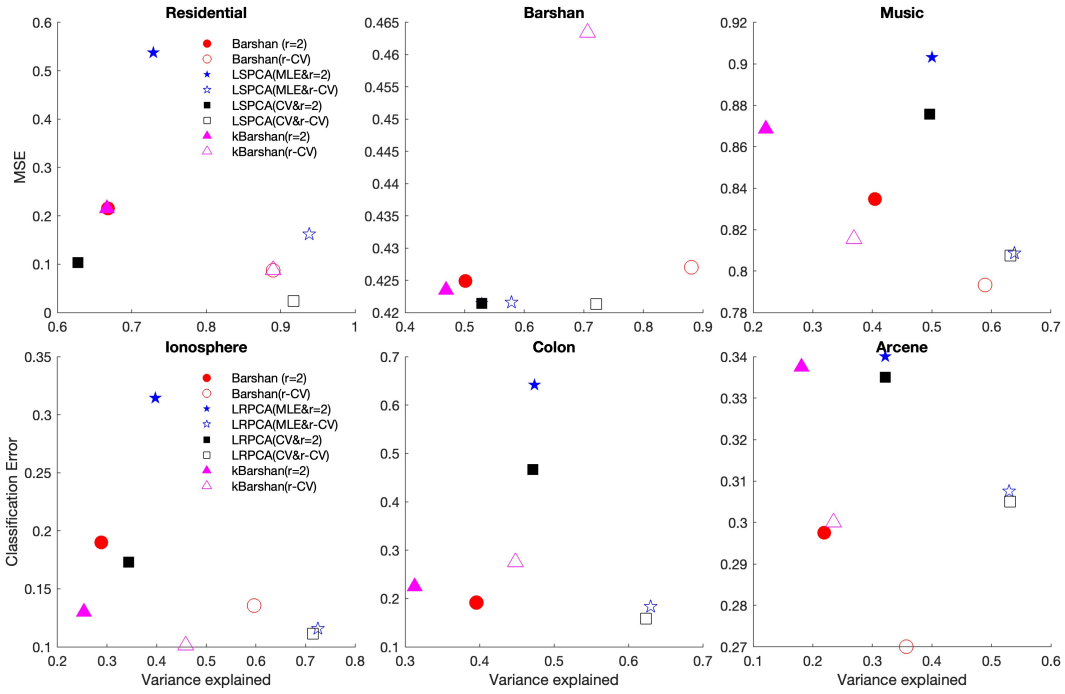

Figure 1: Comparison of Pareto optimality of competing methods in terms of prediction error and variation explained for various datasets

7 Conclusions

In this project, we successfully reproduced three competing methods: Barshan, kBarshan and LSPCA/LRPCA, and verified the superiority of the multi-objective approach in terms of variance explanation. The newly-developed multi-objective approach (LSPCA/LRPCA) minimizes the prediction error and maximizes the variance explained simultaneously. Compared with traditional SPCA methods such as Barshan's, LSPCA (or LRPCA) can achieve a higher data variation representation although with a slightly larger prediction error. Therefore, the multi-objective approach helps to avoid overfitting the data by maximizing the variance that can be explained. Moreover, we find that the r determined by cross-validation is usually larger than 2, resulting in better performance: a lower prediction error and a higher explained variance. We suggest that the choice of r significantly affects the performance of this multi-objective approach.

During this reproduction work, we encountered some difficulties. First, we had to calculate the Euclidean gradient of the objective function (see Appendix A) because Eqs (14) and (15) in [1] may not be correct. Secondly, the computational cost of performing cross-validation is significant. Using a high-performance computer (Greatlakes) with parallel computation, the implementation of the LSPCA(CV)+r(CV) method for the Arcene dataset still required about three days. While the LSPCA or LRPCA with CV leads to a superior performance, its high computational expense may make faster methods like Barshan's method more practical in real applications.

Contribution by Group Members

Bobo Bai: 1) Data pre-processing: Barshan A and Arcene. 2) Reproduce Barshan's method and kBarshan's method and apply them to six datasets. 3) Reproduce LSPCA(CV/MLE) and LR-PCA(CV/MLE). 4) Write the "problem statement","related work", "method", "appendix" sections of the final report. 5) Revise the final report.

Shanchen Liu: 1) Data pre-processing: Ionosphere and Colon. 2) Reproduce LRPCA(CV/MLE) method and apply it to three datasets: Ionosphere, Colon, and Arcene. 3) Write Introduction and motivation (Section 1) and Algorithm(Section 5) of the final report.

Peng Zhai: 1) Data pre-processing: Residential and Music. 2) Reproduce LSPCA(CV/MLE) method and apply it to three datasets: Residential, Barshan, Music, respectively. 3) Help running LRPCA code on Greatlakes. 4) Write draft of Experiments(Section 6) and Conclusion(Section 7) and create Table. 2 and Fig. 1 of the final write-up.

References

[1] A. Ritchie, L. Balzano, D. Kessler, C. S. Sripada, and C. Scott, "Supervised pca: A multiobjective approach," arXiv preprint arXiv:2011.05309, 2020.

[2] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," Pattern Recognition, vol. 44, no. 7, pp. 1357–1371, 2011.

[3] E. Bair, T. Hastie, D. Paul, and R. Tibshirani, "Prediction by supervised principal components," Journal of the American Statistical Association, vol. 101, no. 473, pp. 119–137, 2006.

[4] J. Piironen and A. Vehtari, "Iterative supervised principal components," in International Conference on Artificial Intelligence and Statistics, 2018, pp. 106–114.

[5] S. Sharifzadeh, A. Ghodsi, L. H. Clemmensen, and B. K. Ersbøll, "Sparse supervised principal component analysis (sspca) for dimension reduction and variable selection," Engineering Applications of Artificial Intelligence, vol. 65, pp. 168–177, 2017.

[6] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, "Manopt, a Matlab toolbox for optimization on manifolds," The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1455–1459, 2014.

Appendix

A. Derivation of Euclidean gradient of G with respect to L

To get the Euclidean gradient of G with respect to L, we first simply (5) and (6)

For the L_2 loss term in (5) , we note

$$\|Y - XL\beta\|_F^2 = \|Y\|_F^2 - 2tr(Y^T XL\beta) + \|XL\beta\|_F^2 \tag{11}$$

and for the regularization term in (5) and (6):

$$\|X - \gamma XLL^T\|_F^2 = \|X\|_F^2 - \gamma(2-\gamma)\|XL\|_F^2 \tag{12}$$

where (12) uses the property $L^T L = I_r$.

Ignoring the constant terms in (11) and (12), (5) and (6) reduce to

$$G_{LS} = -2tr(Y^T XL\beta) + \|XL\beta\|_F^2 - \lambda_{LS}\gamma(2-\gamma)\|XL\|_F^2 \tag{13}$$

$$\begin{aligned} G_{LR} &= -\sum_{i=1}^{n}\sum_{j=1}^{q} y_{ij}\boldsymbol{x}_i^T L\boldsymbol{\beta}_j + \sum_{i=1}^{n}\sum_{j=1}^{q} y_{ij}\log\sum_{j'=1}^{q}\exp(\boldsymbol{x}_i^T L\boldsymbol{\beta}_{j'}) - \lambda_{LR}\gamma(2-\gamma)\|XL\|_F^2 \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{q} y_{ij}\boldsymbol{x}_i^T L\boldsymbol{\beta}_j + \sum_{i=1}^{n}\log\sum_{j'=1}^{q}\exp(\boldsymbol{x}_i^T L\boldsymbol{\beta}_{j'}) - \lambda_{LR}\gamma(2-\gamma)\|XL\|_F^2 \end{aligned} \tag{14}$$

where the second equality in (14) uses the fact that $\sum_{j=1}^{q} y_{ij} = 1$, for $\forall i$, since $\boldsymbol{y}_i$ is one-hot for logistic regression.

Treating L as the variable in (13), we get the variation:

$$\begin{aligned}
\mathbf{d}G_{LS} &= -2tr(Y^T X \mathbf{d}L\beta) + 2tr(X\mathbf{d}L\beta\beta^T L^T X^T) - 2\lambda_{LS}\gamma(2-\gamma)tr(X\mathbf{d}LL^T X^T) \\
&= -2tr(\beta Y^T X \mathbf{d}L) + 2tr(\beta\beta^T L^T X^T X \mathbf{d}L) - 2\lambda_{LS}\gamma(2-\gamma)tr(L^T X^T X \mathbf{d}L) \\
&= -2 < X^T Y\beta^T, \mathbf{d}L > +2 < X^T XL\beta\beta^T, \mathbf{d}L > -2\lambda_{LS}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&=< -2X^T \left(Y - XL\beta\right)\beta^T - 2\lambda_{LS}\gamma(2-\gamma)X^T XL, \mathbf{d}L > \qquad (15)
\end{aligned}$$

where the second equality uses the commutativity property of trace operation and the last equality uses the linearity of inner product. From (15),we immediately obtain $\nabla_L G_{LS}$ as shown in (8).

For the NLL G_{LR}, the second term is the same as that in G_{LS}, thus we only need to get the gradient of the first term. Differentiating with respect to L, we have

$$\begin{aligned}
\mathbf{d}G_{LR} &= -\sum_{i=1}^{n}\sum_{j=1}^{q} y_{ij}\boldsymbol{x}_i^T \mathbf{d}L\boldsymbol{\beta}_j + \sum_{i=1}^{n}\sum_{j=1}^{q} M_{ij}\boldsymbol{x}_i^T \mathbf{d}L\boldsymbol{\beta}_j - 2\lambda_{LR}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{q} \left(y_{ij} - M_{ij}\right)\boldsymbol{x}_i^T \mathbf{d}L\boldsymbol{\beta}_j - 2\lambda_{LR}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{q} tr\left(\left(y_{ij} - M_{ij}\right)\boldsymbol{x}_i^T \mathbf{d}L\boldsymbol{\beta}_j\right) - 2\lambda_{LR}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{q} tr\left(\boldsymbol{\beta}_j\left(y_{ij} - M_{ij}\right)\boldsymbol{x}_i^T \mathbf{d}L\right) - 2\lambda_{LR}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{q} < \boldsymbol{x}_i\left(y_{ij} - M_{ij}\right)\boldsymbol{\beta}_j^T, \mathbf{d}L > -2\lambda_{LR}\gamma(2-\gamma) < X^T XL, \mathbf{d}L > \\
&=< -\sum_{i=1}^{n}\sum_{j=1}^{q} \boldsymbol{x}_i\left(y_{ij} - M_{ij}\right)\boldsymbol{\beta}_j^T - 2\lambda_{LR}\gamma(2-\gamma)X^T XL, \mathbf{d}L > \\
&=< -X^T(Y - M)\beta^T - 2\lambda_{LR}\gamma(2-\gamma)X^T XL, \mathbf{d}L > \qquad (16)
\end{aligned}$$

where the first the equality uses the definition of M_{ij} in (10), the third equality uses the fact that $\left(y_{ij} - M_{ij}\right)\boldsymbol{x}_i^T \mathbf{d}L\boldsymbol{\beta}_j$ is a scalar, the fourth equality uses the commutativity property of trace operation, and the last equality uses the fact that

$$\sum_{i=1}^{n}\sum_{j=1}^{q} \boldsymbol{x}_i\left(y_{ij} - M_{ij}\right)\boldsymbol{\beta}_j^T = X^T(Y - M)\beta^T \qquad (17)$$

From (16),we immediately obtain $\nabla_L G_{LR}$ as shown in (9).

B. Barshan's method

Barshan's method finds the orthogonal projection L by maximizing the dependency of between $L^T\mathcal{X}$ and $\mathcal{Y}$. The dependency is measured using HSIC. Given the data matrix X and the response matrix Y, an empirical estimate of HSIC between XL and Y is estimated as:

$$HSIC(XL, Y) = tr(L^T X^T HWHXL) \qquad (18)$$

where $H := I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ is the centering matrix, W is a kernel matrix of Y. In this study, a radial basis function (RBF) kernel is applied to Y for linear regression, and a delta kernal is applied to Y for logistic regression. The optimization problem is formulated as:

$$\max_{L\in\mathbb{R}^{p\times r}} \; tr(L^T X^T HWHXL) \;\; s.t. \;\; L^T L = I_r. \qquad (19)$$

According to the Generalized Rayleigh Quotient theorem, one solution is $L = [\boldsymbol{u}_1, ...\boldsymbol{u}_r]$, where $\boldsymbol{u}_1, ...\boldsymbol{u}_r$ are the eigenvectors of $X^T HWHX$ corresponding to the top r eigenvalues.

C. Kernel Barshan's method

Kernelizing Barshan's method is similar to kernelizing PCA. The key step is to show that the solution L to (19) has the form $X^T\omega$ for some $\omega \in \mathbb{R}^{n\times r}$.

Since W in (19) is a kernel matrix of Y, it is symmetric and positive semi-definite. Let $W^{1/2}$ be the symmetric matrix s.t. $W = W^{1/2}W^{1/2}$. A solution to (19) is given by $L = [\boldsymbol{u}_1, ...\boldsymbol{u}_r]$, where $\boldsymbol{u}_1, ...\boldsymbol{u}_r$ are the left singular vectors of $X^THW^{1/2}$ corresponding to the top r singular values.

Let s =rank$(X^THW^{1/2})$ and consider the singular value decomposition

$$X^THW^{1/2} = U\Sigma V^T \tag{20}$$

where $U \in \mathbb{R}^{p\times s}$ is orthogonal, $\Sigma \in \mathbb{R}^{s\times s}$ is diagonal, and $V \in \mathbb{R}^{n\times s}$ is orthogonal. Right multiplying both sides of (20) by $V\Sigma^{-1}$, we obtain

$$U = X^THW^{1/2}V\Sigma^{-1} \tag{21}$$

Thus, L can be expressed as:

$$\begin{aligned} L &= U\begin{bmatrix} I_r \\ \mathbf{0} \end{bmatrix} \\ &= X^THW^{1/2}V\Sigma^{-1}\begin{bmatrix} I_r \\ \mathbf{0} \end{bmatrix} \\ &= X^T\omega \end{aligned} \tag{22}$$

where $\omega = HW^{1/2}V\Sigma^{-1}\begin{bmatrix} I_r \\ \mathbf{0} \end{bmatrix}$.

Then the reduced data matrix is $XL = XX^Tw$, which only depends on the instances $\boldsymbol{x}_i$ through their inner products. Replacing XX^T with the data kernel matrix K, the reduced data matrix becomes $K\omega$.

To show that the algorithm is kernelizable, we replace L by X^Tw in (19) and get

$$\max_{\omega\in\mathbb{R}^{n\times r}} \; tr(\omega^TXX^THWHXX^T\omega) \;\; s.t. \;\; \omega^TXX^T\omega = I_r. \tag{23}$$

Further replacement of XX^T by the kernel matrix K leads to the kernel Barshan's optimization problem:

$$\max_{\omega\in\mathbb{R}^{n\times r}} \; tr(\omega^TKHWHK\omega) \;\; s.t. \;\; \omega^TK\omega = I_r. \tag{24}$$

One solution is the generalized eigenvectors of $(KHWHK, K)$ corresponding to the top r eigenvalues.