Supervised Manifold Learning for Wide Data

Joshua T. Vogelstein, Mauro Maggioni

Abstract

Supervised learning is a fundamental problem in statistics and pattern recognition. In the modern era, it is routine to collect high-dimensional data, without obtaining a large number of samples. In such situations, some form of regularization is required to achieve high-quality performance. Current approaches either do not perform competitively, or are unfortunately computationally intensive. Using simple geometric arguments, we discover that combining the means and covariances in an intuitive fashion yields a low-dimensional projection that is nearly optimal in a wide variety of simulated and real data scenarios. By utilizing off-the-shelf numerical libraries, our method, called “Linear Optimal Low-rank” (LOL) is extremely computationally efficient as well. Theoretical analyses confirm our intuition. We therefore position LOL as a potential new and principled “default” algorithm for any supervised learning problem with wide data. To enable reproducability and extendability, we provide open-source code in Matlab, R, and Python.

Supervised learning—the art and science of discovering, from data, statistical relationships between multiple different measurement types—is one of the most useful tools in the scientific toolbox. SL has been enabled a wide variety of basic and applied findings, ranging from discovering biomarkers in omics data, to object recognition from images. A special case of SL is classification; a classifier can predict the ’class’ of a novel observation via training on a set of paired observations and class labels (for example, predicting male vs. female from MRI scans). In such problems, the goal is to find the optimal discriminant boundary, which partitions the space of observations into the different classes. In the data age, the ambient (or observed) dimensionality of the observations is quickly ballooning, and with it, the dimensionality of the discriminant boundary. While historical data may have consisted of only a few dimensions (e.g., height and weight), modern scientific datasets often consist of hundreds, thousands, or even millions of dimensions (e.g. genetics, neuroscience, omics). Regardless of the dimensionality, when a scientist or analyst obtains a new dataset consisting of some observations and labels, she must decide which of the myriad available tools to use. Reference algorithms for datasets with low dimensionality include linear and quadratic discriminant analysis, support vector machines, and random forests (**?**).

Classical methods, however, often rely on very restrictive assumptions. In particular, the theoretical guarantees upon which many classical methods rest require that the number of samples is much larger than the dimensionality of the problem ($n \gg p$). In scientific contexts, while the dimensionality of datasets is booming, the sample is not witnessing a concomitant increase (see, for example, Table 2 of (**?**) in connectomics). When the number of dimensions is orders of magnitude larger than the sample size, as is now typical, this $n \gg p$ assumption is woefully inadequate. This inadequacy is not a mere theoretical footnote; rather, the implementation of the algorithm itself sometimes fails or crashes if this assumption is not met. Worse, often times the algorithm will run to completion, but the answer will be essentially random, with little or no predictive power or accuracy (e.g., (**?**)).

To combat these issues, the fields of statistics and machine learning have developed a large collection of new methods that relax these assumptions, and exhibit significantly improved performance characteristics. Each such approach makes some “structural” assumptions about the data (sometimes in the form of priors), therefore potentially adding some bias, while reducing variance. The best approach, for a given problem, is the approach that wins this bias/variance trade-off; that is, the approach whose assumptions best correspond to the properties of the data (minimizing bias), and yields estimates have that low error under those assumptions (minimizing variance). These approaches complement “feature engineering” approaches, where the practitioner designs new features based on prior domain specific knowledge.

One of the earliest approaches for discovering low-dimensional representations in supervised learning problems is regularization or shrinkage (**? ? ?**). Many shrinkage methods mitigate the dimensionality problem by smoothing (**?**), for example, by regressing parameters to the mean. More recently, a special case of regularization has risen to prominence, called *sparsity* (**?**), in which it is assumed that a small number of dimensions can encode the discriminant boundary (**?**). This assumption, when accurate, can lead to substantial improvements in accuracy with relatively moderate increase in computational cost (**?**). This

framework includes methods such as Lasso (**?**), higher criticism thresholding (**?**), and sparse variants of linear discriminant analysis (**? ? ? ? ? ?**).
However, for a wide class of problems, such as image classification, sparisty in the ambient space is an overly restrictive, and therefore, bias-inducing assumption (see Figure 1 for an example on the classic MNIST dataset). A generalization of sparsity is "low-rank", in which a small number of linear combinations of the ambient dimensions characterize the data. Unsupervised low-rank methods date back over a century, including multidimensional scaling (**? ?**) and principal components analysis (**? ?**). More recent nonlinear versions of unsupervised dimensionality reduction, or manifold learning, include developments from neural network theory such as self-organizing maps (**?**), generative topographic mapping (**?**). In this century, manifold learning became more popular, including isomap (**?**), local linear embedding (**?**), Laplacian eigenmaps (**?**), local tangent space alignment (**?**), diffusion maps (**?**), and geometric multi-resolution analysis (**?**). All these approaches can be used as pre-processing steps, to reduce the dimensionality of the data prior to solving the supervised learning problem (**?**).
However, such manifold learning methods, while exhibiting both strong theoretical (1**? ?**) and empirical performance, are fully unsupervised. Thus, in classification problems, they discover a low-dimensional representation of the data, ignoring the labels. This can be highly problematic when the discriminant dimensions and the directions of maximal variance are not aligned (see Figure 1 for an example). Supervised dimensionality reduction techniques, therefore, combine the best of both worlds, search for low-dimensional discriminant boundaries. A set of methods from the statistics community is collectively referred to as "sufficient dimensionality reduction" (SIR) or "first two moments" (F2M) methods (**? ? ? ? ?**). These methods are theoretically elegant, but typically require the sample size to be larger than the number of observed dimensions (although see (**?**) for some promising work). Other approaches formulate an optimization problem, such as projection pursuit (**?**), empirical risk minimization (**?**), or supervised dictionary learning (**?**). These methods are limited because they are prone to fall into local minima, they require costly iterative algorithms, and lack any theoretical guarantees (**?**). Thus, there remains a gap in the literature: a supervised learning method with theoretical convergence guarantees appropriate when the dimensionality is orders of magnitude larger than the sample size.
The challenge lies is posing the problem in such a way that efficient numerical algorithms can be brought to bear, without costly iterations or tuning parameters. Our approach, which we call "Linear Optimal Low-rank" (Lol) embedding (see Figure 1), utilizes the first two moments, as do SIR, spectral decompositions, and high-dimensional discriminant analysis methods (**?**), but does not require iterative algorithms and therefore is vastly more computationally efficient. The motivation for Lol comes from a simple geometric intuition (Figure 2). Indeed, we provide both theoretical insight explaining why our method is more general than previous approaches (low-bias), as well as finite sample guarantees (low-variance). A variety of simulations provide further evidence that Lol efficiently finds a better low-dimensional representation than competing methods, not just under the provable model assumptions, but also under much more general contexts (Figure 3). Moreover, we demonstrate that Lol achieves better performance, in less time, as compared to several reference high-dimensional classifiers, on several benchmark datasets, including genomics, connectomics, and image processing problems (Figure 5). Finally, Lol can also be used to improve high-dimensional regression and testing (Figure 6). Based on the above, we suggest that Lol be considered as one of the reference method for supervised manifold learning for wide data. For reproducibility and extensibility, MATLAB code to run all numerical experiments and reproduce all figures is available from our github repository available here: `http://jovo.me/lol`.

Results

An Illustrative Real Data Example of Supervised Linear Manifold Learning

Pseudocode of any method that embeds high-dimensional data as part of classification proceeds as schematized in Figure 1: (A) obtain/select n training samples of the data, (B) learn a low dimensional projection, (C) project n' testing samples onto the lower dimensional space, (D) classify the embedded testing samples using some classifier. We consider three different linear dimensionality reduction methods—Lasso, Pca, and Lol—each of which we compose with a classifier to form high-dimensional classifiers.[1]

[1] Although Lasso is not a 2-step method (where embedding is learned first, and then a classifier is applied), adaptive lasso (2) and its variants improve on lasso's theoretical and empirical properties, so we consider such an approach here.

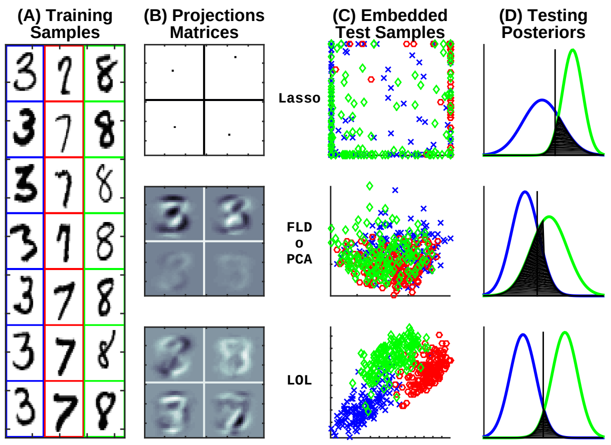

Figure 1: Illustrating three different classifiers—Lasso (top), Fld∘Pca (middle), and Lol (bottom)—for embedding images of the digits 3, 7, and 8 (from MNIST), each of which is 28 $\times$ 28 = 784 dimensional. **(A)**: Exemplars, boundary colors are only for visualization purposes. **(B)**: The first four projection matrices learned by the three different approaches on 300 training samples. Note that Lasso is sparse and supervised, Pca is dense and unsupervised, and Lol is dense and supervised. **(C)**: Embedding 500 test samples into the top 2 dimensions using each approach. Digits color coded as in (A). **(D)**: The estimated posterior distribution of test samples after 5-dimensional projection learned via each method. We show only 3 vs. 8 for simplicity. The vertical line shows the classification threshold. The filled area is the estimated error rate: the goal of any classification algorithm is to minimize that area. Clearly, Lol exhibits the best separation after embedding, which results in the best classification performance.

To demonstrate the utility of Lol, we first consider one of the most popular benchmark datasets ever, the MNIST dataset (3). This dataset consists of many thousands of examples of images of the digits 0 through 9. Each such image is represented by a 28$\times$28 matrix, which means that the observed (or ambient) dimensionality of the data is p =784. Because we are motivated by the $n \ll p$ scenario, we subsample the data to select n=300 examples of the numbers 3, 7, and 8. We then apply all three approaches to this subsample of the MNIST dataset, learning a projection, and embedding n'=500 testing samples, and classifying the resulting embedded data.
Lasso, by virtue of being a sparse method, finds the pixels that most discriminate the 3 classes. The resulting embeddings mostly live along the boundaries, because these images are close to binary, and therefore, images either have or do not have a particular pixel. Indeed, although the images themselves are nearly sparse (over 80% of the pixels in the dataset have intensity ≤ 0.05), a low-dimensional discriminant boundary does not seem to be so. Pca, on the other hand, finds the linear combinations of training samples that maximize the variance. This unsupervised linear manifold learning method results in projection matrices that indeed look like linear combinations of the three different digits. The goal here, however, is separating classes, not maximizing variability. The resulting embeddings are not particularly well separated, suggesting the the directions of discriminability are not the same as the directions of maximum variance. Lol is our newly proposed supervised linear manifold learning method (see below for details). The projection matrices it learns look qualitatively much like those of Pca. This is not surprising, as both are linear combinations of the training examples. The resulting embeddings however, look quite different. The three different classes are very clearly separated by even the first two dimensions. The result of these embeddings yields classifiers whose performance is obvious from looking at the embeddings: Lol achieves significantly smaller error than the other two approaches. This numerical experiment justifies the use of supervised linear manifold learning, we next investigate the performance of these methods in simpler simulated examples, to better illustrate when we can expect Lol to outperform other methods, and perhaps more importantly, when we expect this “vanilla” variant of Lol to fail.

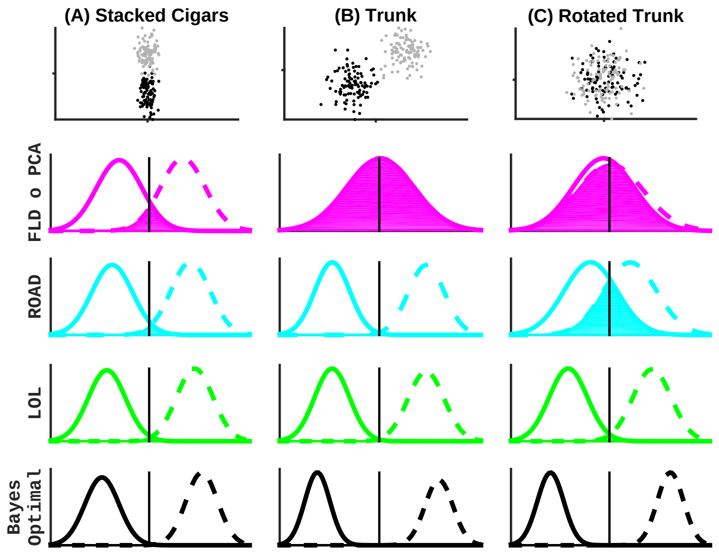

Figure 2: Lol achieves near optimal performance for a wide variety of distributions. Each point is sampled from a multivariate Gaussian; the three columns correspond to different simulation parameters (see Methods for details). In each of 3 simulations, we sample n=100 points in p=1000 dimensions. And for each approach, we embed into the top 20 dimensions. Note that we use the sample estimates, rather than the true population values of the parameters. In this setting, the results are similar. **(A)**: The mean difference vector is aligned with the direction of maximal variance, maxing it ideal for both Pca to discover the discriminant direction and a sparse solution. **(B)**: The mean difference vector is orthogonal to the direction of maximal variance, making Pca fail, but sparse methods can still recover the correct dimensions. **(C)**: Same as (B), but the data are rotated. **Row 1**: A scatter plot of the first two dimensions of the sampled points, with class 0 and 1 as black and gray dots, respectively. **Row 2**: Fld ∘ Pca. **Row 3**: Road, a sparse method designed specifically for this model (**?**). **Row 4**: Lol, our newly proposed method. **Row 5**: the Bayes optimal classifier, which is what all classifiers strive to achieve. Note that Lol is closest to Bayes optimal in all three settings.

Linear Gaussian Intuition

The above real data example suggests the geometric intuition for when Lol outperforms its sparse and unsupervised counterparts. To further investigate, both theoretically and numerically, we consider the simplest setting that illustrates the relevant geometry. In particular, we consider a two-class classification problem, where both classes are distributed according to a multivariate normal distribution, the class priors are equal, and the joint distribution is centered, so that the only difference between the classes is their means (we call this the Linear Discriminant Analysis (LDA) model; see Methods for details).

To motivate Lol, and the following simulations, lets consider what the optimal projection would be in this scenario. The optimal low-dimensional projection is analytically available as the dot product of the difference of means and the inverse covariance matrix, $\boldsymbol{A}_* = \boldsymbol{\delta}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}$ (**?**) (see Methods for details). Pca, the dominant unsupervised manifold learning method, utilizes only the covariance structure of the data, and ignores the difference between the means. In particular, Pca would project the data on the top d eigenvectors of the covariance matrix. **The key insight of our work is the following: we can use both the difference of the means and the covariance matrix, rather than just the covariance matrix, to find a low dimensional projection.** Naiv̈ely, this should typically improve performance, because in this stylized scenario, both are important. Formally, we implement this idea by simply concatenating the difference of the means with the top d eigenvectors of the covariance. This is equivalent to first projecting onto the difference of the means vector, and then projecting the residuals onto the first d principle components. Thus, it requires almost no additional computational time or complexity, rather, merely estimates the difference of the means. In this sense, Lol can be thought of as a very simply “supervised Pca”.

Figure 2 shows three different examples of data sampled from the LDA model to geometrically illustration this intuition. In each, we sample n=100 training samples in p=1000 dimensional space, so $n \ll p$. Figure 2(A) shows an example we call "stacked cigars". In this example and the next, the covariance matrix is diagonal, so all ambient dimensions are independent of one another. Moreover, the difference between the means and diagonal are both large along the same dimensions (they are highly correlated with one another). This is an idealized setting for Pca, because Pca finds the direction of maximal variance, which happens to correspond to the direction of maximal separation. However, Pca does not weight the discriminant directions sufficiently, and therefore performs only moderately well.[2] Because all dimensions are independent, this is a good scenario for sparse methods. Indeed, Road, a sparse classifier designed for precisely this scenario, does an excellent job finding the most useful ambient dimensions. Lol does the best of all three approaches, by using both the difference of the means and the covariance.
Figure 2(B) shows an example which is a worst case scenario for using Pca to find the optimal projection for classification. In particular, the variance is getting larger for subsequent dimensions, $\sigma_1 < \sigma_2 < \cdots < \sigma_p$, while the magnitudes of the difference between the means are decreasing with dimension, $\delta_1 > \delta_2 < \cdots > \delta_p$. Thus, for any truncation level, Pca finds exactly the *wrong* directions. Road is not hampered by this problem, it is also able to find the directions of maximal discrimination, rather than those of maximal variance. Again, Lol, by using both parameters, does extremely well.
Figure 2(C) is exactly the same as (B), except the data have been randomly rotated in all 1000 dimensions. This means that none of the original coordinates have much information, rather, linear combinations of them do. This is evidenced by observing the scatter plot, which shows that two dimensions clearly fail to disambiguate the two classes. Pca, being rotationally invariant, fails in this scenario as it did in (B). Now, there is no small number of ambient dimensions that separate the data well, so Road also fails. However, Lol, by virtue of being rotationally invariant, is unperturbed by this rotation. In particular, it is able to "unrotate" the data, to find dimensions that optimally separate the two classes.

Theoretical Confirmation

The above numerical experiments provide the intuition to guide our theoretical developments.

Theorem 1. *Under the LDA model,* Lol *is better than* Pca*.*

In words, it is better to incorporate the mean difference vector into the projection matrix. The degree of improvement is a function of the embedding dimension d, the ambient dimensionality p, and the parameters (see Methods for details and proof).

How many dimensions to keep?

In the above numerical and theoretical investigations, we fixed d, the number of dimensions to embed into. Much unsupervised manifold learning theory typically focuses on finding the "true" intrinsic dimensionality of the data. The analogous question for supervised manifold learning would be to find the true intrinsic dimensionality of the discriminant boundary. However, in real data problems, typically, their is no perfect low dimensional representation.
Thus, in all the following simulations, the true ambient dimensionality of the data is equal to the dimensionality of the optimal discriminant boundary (given infinite data). In other words, there does not exist a discriminant space that is lower dimensional than the ambient space, so we cannot find the "intrinsic dimension" of the data or the discriminant boundary. Rather, we face a trade-off: keeping more dimensions reduces bias, but increases variance. The optimal bias/variance trade-off depends on the distribution of the data, as well as the sample size (**?**). We formalize this notion for the LpA model and proof the following:

Theorem 2. *Under the LDA model, estimated* Lol *is better than* Pca*.*

Note that the degree of improvement is a function of the number of samples n, in addition to the embedding dimension d, the ambient dimensionality p, and the parameters (see Methods for details and proof).
Consider again the rotated trunk example as well as a "Toeplitz" example, as depicted in Figures 3(A) and (B). In both cases, the data are sampled from the LDA model, and in both cases, the optimal dimensionality

[2] When having to estimate the eigenvector from the data, Pca performs even worse. This is because when $n \ll p$, Pca is an inconsistent estimator with large variance (**? ?**)

depends on the particular approach, but is never the true dimensionality. Moreover, LOL dominates the other approaches, regardless of the number of dimensions used. Figure 3(C) shows a sparse example with "fat tails" to mirror real data settings better. The qualitative results are consistent with those of (A) and (B). Indeed, we can generalize Theorem 2 to include "sub-Gaussian" data, rather than just Gaussian:

Theorem 3. *Under a sub-Gaussian generalization of the LDA model,* LOL *is still better than* PCA.

Multiple Classes

LOL can trivially be extended to > 2 class situations. Naiv̈ely it may seem like we would need to keep all pairwise differences between means. However, given k classes, the set of all k^2 differences is only rank $k-1$. In other words, we can equivalently find the class which has the maximum number of samples (breaking ties randomly), and subtract its mean from all other class means. Figure 3(D) shows a 3-class generalization of (A). While LOL uses the additional class naturally, many previously proposed high-dimensional FLD variants, such as ROAD, natively only work for 2-classes.

Generalizations of LOL

The simple geometric intuition which led to the development of LOL suggests that we can easily generalize LOL to be more appropriate for more complicated settings. We consider three additional scenarios:

QDA Sometimes, it makes more sense to model each class as having a unique covariance matrix, rather than a shared covariance matrix. Assuming everything is Gaussian, the optimal classifier in this scenario is called Quadratic Discriminant Analysis (QDA) (**?**). Intuitively then, we can modify LOL to compute the eigenvectors separately for each class, and concatenate them (sorting them according to their singular values). Moreover, rather than classifying the projected data with LDA, we can then classify the projected data with QDA. Indeed, simulating data according to such a model (Figure 3(E), LOL performs slightly above chance, regardless of the number of dimensions we use to project, whereas QOQ (which denotes we estimate eigenvectors separately and then use QDA on the projected data) performs significantly better regardless of how many dimensions it keeps.

Outliers Outliers persist in many real data sets. Finding outliers, especially in high-dimensional data, is both tedious and difficult. Therefore, it is often advantageous to have estimators that are robust to certain kinds of outliers (4**? ?**). PCA and eigenvector computation are particularly sensitive to outliers (5). Because LOL is so simple and modular, we can replace typical eigenvector computation with a robust variant thereof, such as (**?**). Figure 3(F) shows an example where we generated $n/2$ training samples according to the simple LDA model, but then added another $n/2$ training samples from a noise model. LRL (our robust variant of LOL that simply replaces the fragile eigenvector computation with a robust version), performs better than LOL regardless of the number of dimensions we keep.

XOR XOR is perhaps the simplest nonlinear problem, the problem that led to the demise of the perceptron, prior to its resurgence after the development of multi-layer perceptrons (**?**). Thus, in our opinion, it is warranted to check whether any new classification method can perform well in this scenario. The classical (two-dimensional) XOR problem is quite simple: the output of a classifier is zero if both inputs are the same (00 or 11), and the output is one if the inputs differ (01 or 10). Figure 3(G) shows a high dimensional and stochastic variant of XOR. This simulation was designed such that standard classifiers, such as support vector machines and random forests, achieve chance levels (not shown). LOL, performs moderately better than chance, and QOQ performs significantly better than chance, regardless of the chosen dimensionality. This demonstrates that our classifiers developed herein, though quite simple and intuition, can perform well even in settings where the data are badly modeled by our underlying assumptions. This mirrors previous findings where the so-called "idiots's Bayes" classifier outperforms more sophisticated classifiers (**?**). In fact, we think of our work as finding intermediate points between idiot's Bayes (or naiv̈e Bayes) and FLD, by enabling degrees of regularization by changing the dimensionality used.

Computational Efficiency

In many applications, the main quantifiable consideration in whether to use a particular method, other than accuracy, is numerical efficiency. Because implementing LOL requires only highly optimized linear algebraic routines—including computing moments and singular value decomposition—rather than the costly iterative

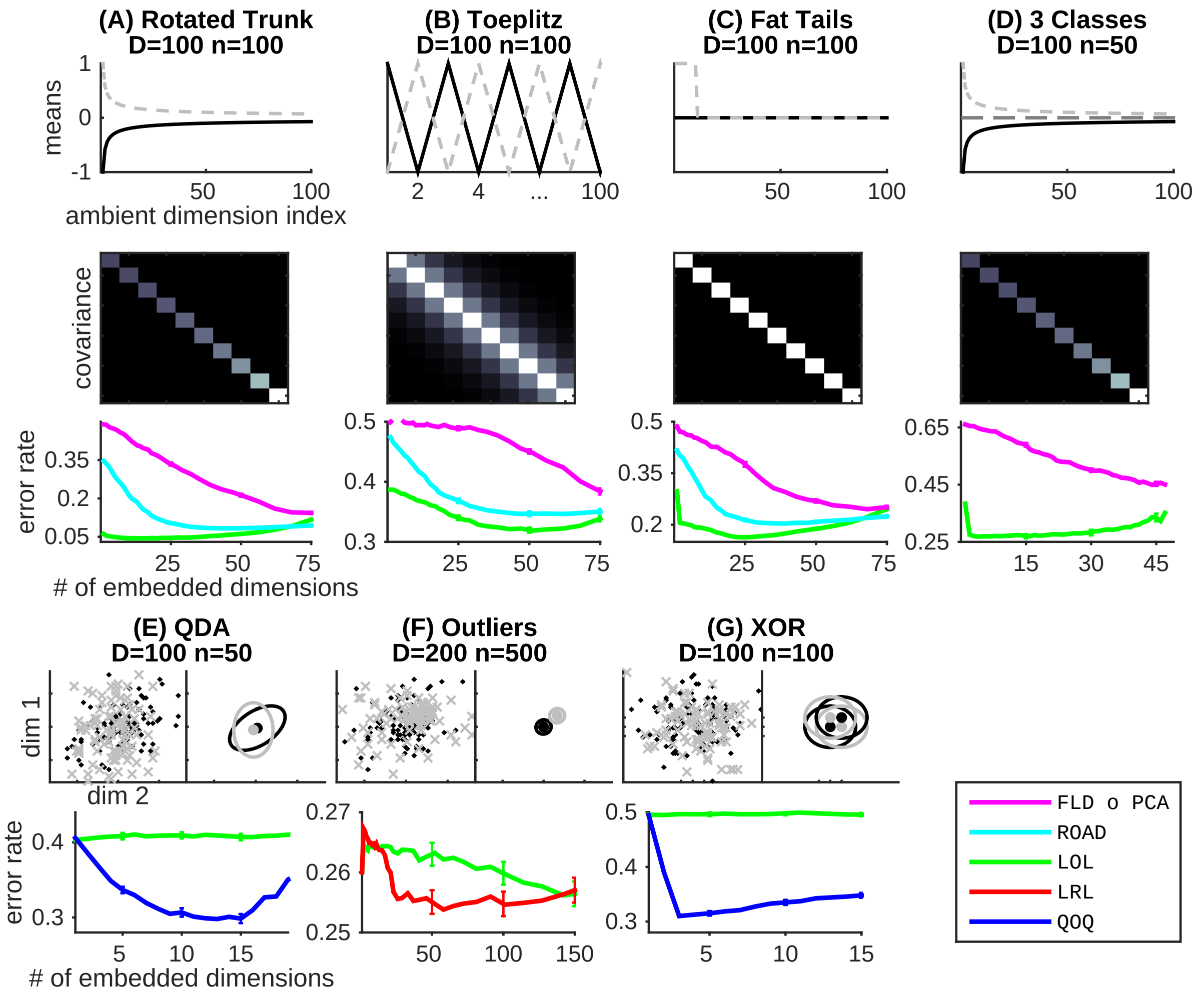

Figure 3: Seven simulations demonstrating that even when the true discriminant boundary is high-dimensional, LOL can find a low-dimensional projection that wins the bias-variance trade-off against competing methods. For the first four, the top panels depict the means (top), the shared covariance matrix (middle). For the next three, the top panels depict a 2D scatter plot (left), mean and level set of one standard deviation of covariance matrix (right). For all seven simulations, the bottom panel shows misclassification rate as a function of the number of embedded dimensions, for several different classifiers. The simulations settings are as follows: **(A)** Rotated Trunk: same as Figure 2(C). **(B)** Toeplitz: another setting where mean difference is not well correlated with any eigenvector, and no ambient coordinate is particularly useful on its own. **(C)** Fat Tails: a common phenomenon in real data; we have theory to support this generalization of the LDA model. **(D)** 3 Classes: LOL naturally adapts to multiple classes. **(E)** QDA: QOQ, a variant of LOL when each class has a unique covariance, outperforms LOL, as expected. **(F)** Outliers: adding high-dimensional outliers degrades performance of standard eigensolvers, but those can easily be replaced in LOL for a robust variants (called LRL). **(F)** XOR: a high-dimensional stochastic generalization of XOR, demonstrating the LOL and QOQ work even in scenarios that are quite distinct from the original motivating problems. In all 7 cases, LOL, or the appropriate generalization thereof, outperforms unsupervised, sparse, or other methods. Moreover, the optimal embedding dimension is never the true discriminant dimension, but rather, a smaller number jointly determined by parameter settings and sample size.

programming techniques currently required for sparse or dictionary learning type problems. To quantify the computational efficiency of LOL and its variants, Figure 4 shows the wall time it takes to run each method on the stacked cigars problem, varying the ambient dimensionality, embedded dimensionality, and sample size. Note that for completeness, we include two additional variants of LOL: LAL and LFL. LFL (short for fast LOL) replaces the standard SVD algorithm with a randomized variant, which can be much faster in certain situations (**?**). LAL goes even one step further, replacing SVD with random projections (**?**). This variant of LOL is the fastest, its runtime is least sensitive to (p,d,n), and its accuracy is often commensurate (or better) than other variants of LOL. We will explore RaLOL in future work. Note that the runtime of all the variants of LOL are quite similar to FLD $\circ$ PCA. Given, given LOL's improved accuracy, and nearly identical simplicity, it seems there is very little reason to not use LOL instead of FLD $\circ$ PCA.

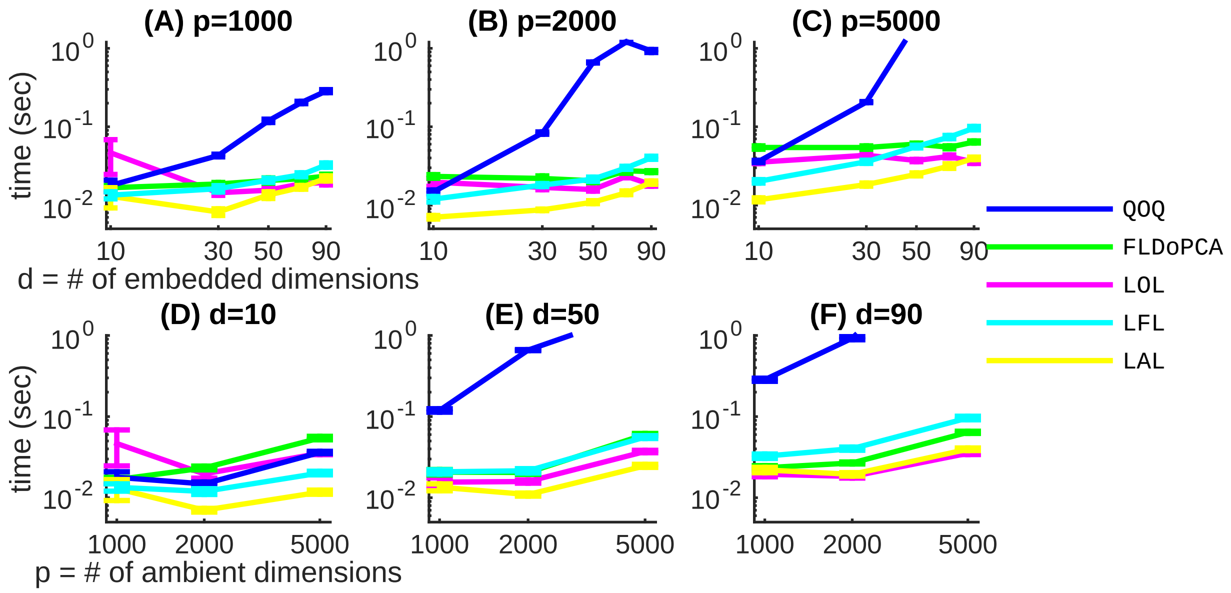

Figure 4: Computational efficiency of various low-dimensional projection methods. In all cases, n=100, and we used the "stacked cigars" simulation parameters. We compare PCA with the projection steps from LOL, QOQ, LRL, LFL, and LAL, for different values of (p,d). The addition of the mean difference vector is essentially negligible. Moreover, for small d, the LFL is advantageous. LAL is always fastest, and its performance is often comparable to other methods (not shown).

Benchmark Real Data Applications

To more comprehensively understand the relative advantages and disadvantages of LOL with respect to other high-dimensional classification approaches, in addition to evaluating its performance in theory, and in a variety of numerical simulations, it is important to evaluate it also on benchmark datasets. For these purposes, we have selected four commonly used high-dimensional datasets (see Methods for details). For each, we compare LOL to (i) support vector machines, (ii) ROAD, (iii) lasso, (iv) and random forest (RF). Because in practice all these approaches have "hyperparameters" to tune, we consider several possible values for SVM, lasso, and LOL (but not RF, as its runtime was too high). Figure 5 shows the results for all four datasets.

Qualitatively, the results are similar across datasets: LOL achieves high accuracy and computational efficiency as compared to the other methodologies. Considering Figure 5(A) and (B), two popular sparse settings, we find that LOL can find very low dimensional projections with very good accuracy. For the prostate data, with a sufficiently non-sparse solution for ROAD, it slightly outperforms LOL, but at substantial computational cost, in particular, ROAD takes about 100 times longer to run on this dataset. Figure 5(C) and (D) are 10-class problems, so ROAD is no longer possible. Here, SVM can again slightly outperform LOL, but again, requiring 100 fold additional computational time. In all cases, the beloved random forest classifier performs subpar.

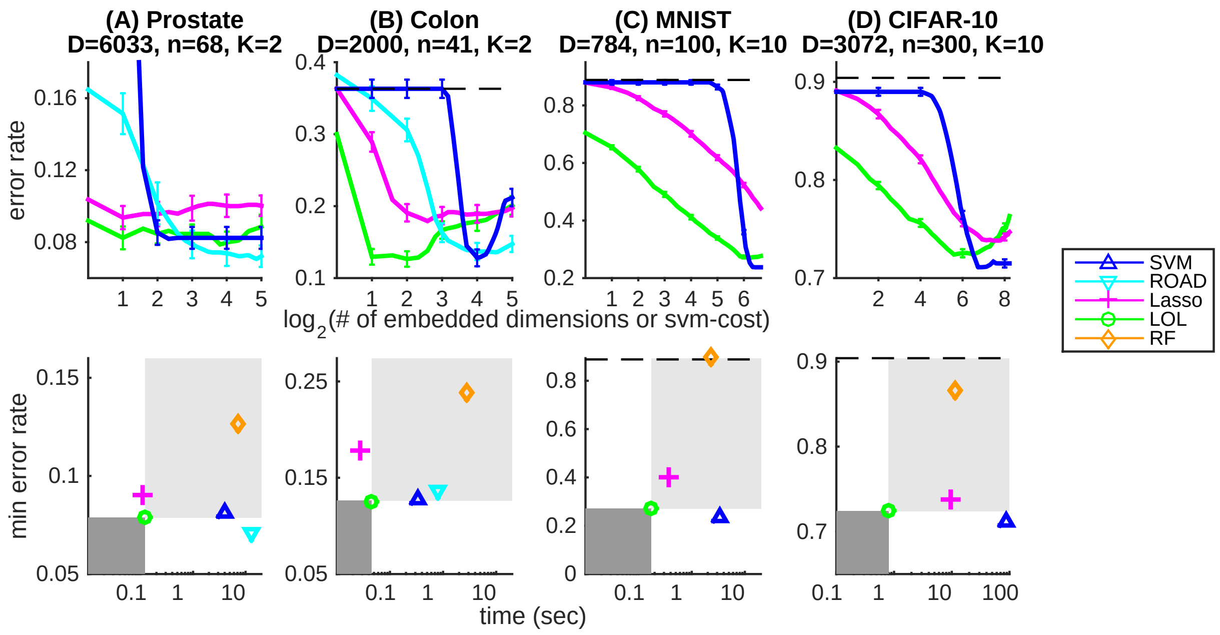

Figure 5: For four standard datasets, we benchmark LOL (green circles) versus standard classification methods, including support vector machines (blue up triangles), ROAD (cyan down triangles), LASSO (magenta pluses), and random forest (orange diamonds). Top panels show error rate as a function of log$_2$ number of embedded dimensions (for LOL, ROAD, and LASSO) or cost (for SVM). Bottom panels show the minimum error rate achieved by each of the five algorithms versus time. The lower left dark gray (upper right light gray) rectangle is the area in which any algorithm is *better* (worse) than LOL in terms of both accuracy and efficiency. **(A)** Prostate: a standard sparse dataset. 1-dimensional LOL does very well, although keeping 2^5 ambient coordinates slightly improves performance, at a significant cost of compute time (two orders of magnitude), with minimal additional interpretability. **(B)** Colon: another standard sparse dataset. Here, 2-4 dimensions of LOL outperforms all other approaches considered. **(C)** MNIST: 10 image categories here, so ROAD is not possible. LOL does very well regardless of the number of dimensions kept. SVN marginally improves on LOL accuracy, at a significant cost in computation (two orders of magnitude). **(D)** CIFAR-10: a higher dimensional and newer 10 category image classification problem. Results are qualitatively similar to (C). Note that, for none of the problems is there an algorithm ever performing better and faster than LOL; rather, most algorithms typically perform worse and slower (though some are more accurate and much more computationally expensive. This suggests that regardless of how one subjectively weights computational efficiency versus accuracy, LOL is the best default algorithm in a variety of real data settings.

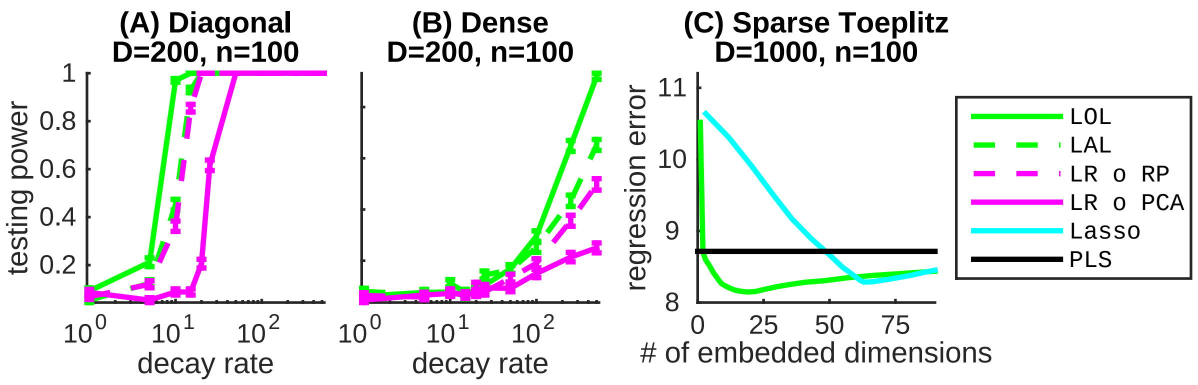

Figure 6: The intuition of including the mean difference vector is equally useful for other supervised manifold learning problems, including testing and regression. (A) and (B) show two different high-dimensional testing settings, as described in Methods. Power is plotted against the decay rate of the spectrum, which approximates the effective number of dimensions. LOL composed with Hotelling's test outperforms the random projections variants described in (**?**), as well as several other variants. (C) shows a high-dimensional regression settings, as described in Methods. Log_{10} mean squared error is plotted against the number of embedded dimensions. Regression LOL composed with linear regression outperforms LASSO (cyan), the classic sparse regression method, as well as partial least squares (PLS; black). In the legend, 'A' denote either 'linear regression' (in (C)), or 'Hotelling' (in (A) and (B)). These three simulation settings therefore demonstrate the generality of this technique.

Extensions to Other Supervised Learning Problems

The utility of incorporating the mean difference vector into supervised machine learning for wide data extends beyond merely classification. In particular, hypothesis testing can be considered as a special case of classification, with a particular loss function. Therefore we apply the same idea to a hypothesis testing scenario. The multivariate generalization of the t-test, called Hotelling's Test, suffers from the same problem as does the classification problem; namely, it requires inverting an estimate of the covariance matrix. To mitigate this issue in the hypothesis testing scenario, authors have applied similar tricks as they have done in the classification setting. One particularly nice and related example is that of Lopes et al. (**?**), who addresses this dilemma by using random projections to obtain a low-dimensional representation, following by applying Hotelling's Test in the lower dimensional subspace. Figure 6(A) and (B) shows the power of their test alongside the power of the same approach, but using the LOL projection rather than random projections. The two different simulations include the simulated settings considered in their manuscript (see Methods for details). The results make it clear that the LOL test has higher power for essentially all scenarios. Moreover, it is not merely the replacing random projections with PCA, nor simply incorporating the mean difference vector, but rather, it appears that LOL for testing uses both modifications to improve performance. High-dimensional linear regression is another supervised learning method that can utilize this idea. Linear regression, like classification and Hotelling's Test, requires inverting a singular matrix as well. By projecting the data only a lower dimensional subspace first, followed by linear regression on the low-dimensional data, we can mitigate the curse of high-dimensions. To choose the projection matrix, we partition the data into K partitions, based on the percentile of the target variable, we obtain a K class classification problem. Then, we can apply LOL to learn the embedding. Figure 6(C) shows an example of this approach, contrasted with lasso and partial least squares, in a sparse simulation setting (see Methods for details). LOL is able to find a better low-dimensional projection than lasso, and performs significantly better than PLS, for essentially all choices of number of dimensions to embed into.

Discussion

We have introduced a very simple, yet new, device to improve performance on supervised learning problems with wide data. In particular, we have proposed a supervised manifold learning procedure, the utilizes both the difference of the means, and the covariance matrices. This is in stark contrast to previous approaches, which only utilize the covariance matrices (or kernel variants thereof), or solve a difficult optimization theoretic problem. In addition to demonstrating the accuracy and numerical efficiency of LOL on simulated and real classification problems, we also demonstrate how the same idea can also be used for other kinds of supervised learning problems, including regression and hypothesis testing.

One of the first publications to compose FLD with an unsupervised learning method was the celebrated Fisherfaces paper (**?**). The authors showed via a sequence of numerical experiments the utility of embedding with PCA prior to classifying with FLD. We extend this work by adding a supervised component to the initial embedding. Moreover, we provide the geometric intuition for why and when this is advantageous, as well as show numerous examples demonstrating its superiority. Finally, we have matrix concentration inequalities proving the advantages of LOL over Fisherfaces.

The LOL idea, appending the mean difference vector to convert unsupervised manifold learning to supervised manifold learning, has many potential applications. We have presented the first few. Incorporating additional nonlinearities via kernel methods (**?**), ensemble methods such as random forests (6), multiscale methods (**?**), and more scalable implementations (**?**), are all of immediate interest.

ALGORITHMS	p < n	2-Class LDA MODELS, p > n			GENERALIZED MODELS, p > n					OTHER TASKS, p > n		ALGORITHM PROPERTIES							ALGORITHMS	REF	NOTES
	LDA model	delta & cov aligned	& cov misaligned	delta & cov misaligned rotated	Fat Tails	>2 Class	QDA Model	Outlier Model	Non-linear	Testing	Regress	Fast	Simple	Theory	Open Source	Dense	2-step/ DL	Super-vised			
kNN	X	X	X	X	X	X		X	X		X		X	X	X	X		X	kNN	1	not considered
HCT	X	X	X									X	X	X			X	X	HCT	14	
LDA o Trace-Ratio	X	X	X	X								X	X	X		X	X	X	LDA o Trace-Ratio	4	
F2M / SDR	X			X		X								X	X	X	X	X	F2M / SDR	10	needs n>p
Deep Learning	X				X	X	X	X	X		X				X	X	X	X	Deep Learning	11	best when n>>p
Vowpal Wabbit	X				X	X		X			X	X			X	X	X	X	Vowpal Wabbit	5	best when n>>p
Ridge Regression	X											X	X	X	X	X		X	Ridge Regression	1	regression
Partial Least Squares											X	X						X	Partial Least Squares	1	regression
Naive Bayes	X	X	X			X						X	X	X	X	X		X	Naive Bayes	1,12	
HDDA	X	X	X	X		X									X	X		X	HDDA	6	
ROAD	X	X											X	X	X			X	ROAD	7,8,9	
LASSO	X	X				X					X	X	X	X	X		X	X	LASSO	3	
kSVM	X	X	X	X	X	X	X	X	X					X	X	X		X	kSVM	1	
Random Forest	X													X	X	X		X	Random Forest	1, 15	
LDA	X											X	X	X	X			X	LDA	1	
LDA o PCA	X	X				X					X	X	X	X	X	X	X	1/2	LDA o PCA	2	
LOL	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	LOL	*	
FIGURE		2A	2B, 3A	2C, 3B	3C	3D	3E	3F	3G	5A, 5D	5C, 5D	6									

Figure 7: Table of algorithms and their properties for high-dimensional data. Gray elements indicate that results are demonstrated in the Figure labeled in the bottom row. 'X' denotes relatively good performance for a given setting, or has the particular property.

A Theory

I.A The Classification Problem

Let $(\boldsymbol{X}, Y)$ be a pair of random variables, jointly sampled from $F := F_{\boldsymbol{X},Y} = F_{\boldsymbol{X}|Y} F_Y$. Let $\boldsymbol{X}$ be a multivariate vector-valued random variable, such that its realizations live in p dimensional Euclidean space, $\boldsymbol{x} \in \mathbb{R}^p$. Let Y be a categorical random variable, whose realizations are discrete, $y \in \{0, 1, \ldots C\}$. The goal of a classification problem is to find a function $g(\boldsymbol{x})$ such that its output tends to be the true class label y:

$$g^*(\boldsymbol{x}) := \operatorname*{argmax}_{g \in \mathcal{G}} \mathbb{P}[g(\boldsymbol{x}) = y].$$

When the joint distribution of the data is known, then the Bayes optimal solution is:

$$g^*(\boldsymbol{x}) := \operatorname*{argmax}_{y} f_{y|\boldsymbol{x}} = \operatorname*{argmax}_{y} f_{\boldsymbol{x}|y} f_y = \operatorname*{argmax}_{y} \{\log f_{\boldsymbol{x}|y} + \log f_y\} \tag{1}$$

Denote expected misclassification rate of classifier g for a given joint distribution F,

$$L_g^F := \mathbb{E}[g(\boldsymbol{x}) \neq y] := \int \mathbb{P}[g(\boldsymbol{x}) \neq y] f_{\boldsymbol{x},y} d\boldsymbol{x} dy,$$

where $\mathbb{E}$ is the expectation, which in this case, is with respect to F_{XY}. For brevity, we often simply write L_g, and we define $L_* := L_{g^*}$.

I.B Linear Discriminant Analysis (LDA) Model

A statistical model is a family of distributions indexed by a parameter $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\mathcal{F}_{\boldsymbol{\theta}} = \{F_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$. Consider the special case of the above where $F_{\boldsymbol{X}|Y=y}$ is a multivariate Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma})$ and F_Y is a categorical distribution $\mathcal{C}(\boldsymbol{\pi})$, where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_C)$ and $\mathbb{P}[Y = y]$ is π_y. In this scenario, the different classes have different means, but a shared covariance matrix. We refer to this model as the Linear Discriminant Analysis (LDA) model. Let $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and let $\boldsymbol{\Theta}_{LDA} = (\triangle_C, \mathbb{R}^{p \times C}, \mathbb{R}^{p \times p}_{\succ 0})$, where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C)$, $\triangle_C$ is the C dimensional simplex, that is $\triangle_C = \{\boldsymbol{x} : x_i \geq 0 \forall i, \sum_i x_i = 1\}$, and $\mathbb{R}^{p \times p}_{\succ 0}$ is the set of positive definite $p \times p$ matrices. Denote $\mathcal{F}_{LDA} = \{F_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{LDA}\}$.
Define

$$g_{LDA}(\boldsymbol{x}) = \operatorname*{argmin}_{y} \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_0) + \mathbb{I}\{Y = y\} \log \pi_y,$$

where $\mathbb{I}\{\cdot\}$ is one when its argument is true, and zero otherwise. Let L_{LDA} be the misclassification rate of the above classifier.

Lemma 1. *For any $F \in \mathcal{F}_{LDA}$, $L_{LDA} = L_*$.*

Proof. Under the LDA model, the Bayes optimal classifier is available by plugging the explicit distributions into Eq. (1). □

Under the two-class model, with equal class prior and centered means, $\pi_0 = \pi_1$ and $(\boldsymbol{\mu}_0 + \boldsymbol{\mu}1)/2 = 0$, re-arranging a bit, we obtain

$$g_{LDA}(\boldsymbol{x}) := \operatorname*{argmin}_{y} \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_y = \mathbb{I}\{\boldsymbol{x}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta} > 0\},$$

where $\boldsymbol{\delta} = \boldsymbol{\mu}_0 - \boldsymbol{\mu}_1$. In words, the Bayes optimal classifier, under the LDA model, takes the input vector $\boldsymbol{x}$, and projects it onto the real number line with projection matrix $\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$. If the resulting projection is greater than 0, then $\hat{y} = 1$, otherwise, $\hat{y} = 0$. Note that the equal class prior and centered means assumptions merely changes the threshold constant from 0 to something else.

I.C Projection Based Classifiers

Let $\boldsymbol{A} \in \mathbb{R}^{d\times p}$ be an orthonormal matrix, that is, a matrix that projects p dimensional data into a d dimensional subspace, where $\boldsymbol{A}\boldsymbol{A}^\mathsf{T}$ is the $d \times d$ identity matrix, and $\boldsymbol{A}^\mathsf{T}\boldsymbol{A}$ is symmetric $p \times p$ matrix with rank d. The question that motivated this work is: what is the best projection matrix that we can estimate, to use to "pre-process" the data prior to applying LDA. Projecting the data $\boldsymbol{x}$ onto a low-dimensional subspace, and the classifying via LDA in that subspace is equivalent to redefining the parameters in the low-dimensional subspace, $\boldsymbol{\Sigma}_A = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\mathsf{T} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{\delta}_A = \boldsymbol{A}\boldsymbol{\delta} \in \mathbb{R}^d$, and then performing

$$g_A(x) := \mathbb{I}\{(\boldsymbol{A}\boldsymbol{x})^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A > 0\}. \tag{2}$$

Let $L_A := \int \mathbb{P}[g_A(\boldsymbol{x}) = y] f_{\boldsymbol{x},y} d\boldsymbol{x} dy$. Our goal therefore is to be able to choose A for a given parameter setting $\boldsymbol{\theta} = (\boldsymbol{\delta}, \boldsymbol{\Sigma})$, such that L_A is as small as possible (note that L_A will never be smaller than L_*).
Formally, we seek to solve the following optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{A}}{\text{minimize}} && \mathbb{E}[\mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A > 0\} \neq y] \\ &\text{subject to} && \boldsymbol{A} \in \mathbb{R}^{p\times d}, \quad \boldsymbol{A}\boldsymbol{A}^\mathsf{T} = \boldsymbol{I}_{d\times d}, \end{aligned} \tag{3}$$

where $\boldsymbol{I}_{u\times v}$ is the $u \times v$ identity matrix identity, that is, $\boldsymbol{I}(i,j) = 1$ for all $i = j \leq \min(u,v)$, and zero otherwise. In our opinion, Eq. (3) is the simplest supervised manifold learning problem there is: a two-class classification problem, where the data are multivariate Gaussians with shared covariances, the manifold is linear, and the classification is done via LDA. Nonetheless, solving Eq. (3) is difficult, because we do not know how to evaluate the integral analytically, and we do not know any algorithms that are guaranteed to find the global optimum in finite time. This has led to previous work using a surrogate function (**?**). We proceed by studying a few natural choices for $\boldsymbol{A}$.

I.C(1) Bayes Optimal Projection

Lemma 2. *For any* $F \in \mathcal{F}_{LDA}$, $L_{\boldsymbol{\delta}^\mathsf{T}\boldsymbol{\Sigma}^{-1}} = L_*$

Proof. Let $\boldsymbol{B} = (\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta})^\mathsf{T} = \boldsymbol{\delta}^\mathsf{T}(\boldsymbol{\Sigma}^{-1})^\mathsf{T} = \boldsymbol{\delta}^\mathsf{T}\boldsymbol{\Sigma}^{-1}$, so that $\boldsymbol{B}^\mathsf{T} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$, and plugging this in to Eq. (2), we obtain

$$\begin{aligned} g_B(x) &= \mathbb{I}\{\boldsymbol{x}\boldsymbol{B}^\mathsf{T}\boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B > 0\} \\ &= \mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta} \times (\boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B) > 0\} \\ &= \mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}k > 0\}, \end{aligned}$$

where the third equality follows from the fact that $(\boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B)$ is just a positive constant $k > 0$. In other words, letting $\boldsymbol{B}$ be the Bayes optimal projection recovers the Bayes classifier, as it should. □

I.C(2) Principle Components Analysis (PCA) Projection

Principle Components Analysis (PCA) finds the directions of maximal variance in a dataset. PCA is closely related to eigendecompositions and singular value decompositions (SVD). In particular, the top principle component of a matrix $\boldsymbol{X} \in \mathbb{R}^{p\times n}$, whose columns are centered, is the eigenvector with the largest corresponding eigenvalue of the centered covariance matrix $\boldsymbol{X}\boldsymbol{X}^\mathsf{T}$. SVD enables one to estimate this eigenvector without ever forming the outer product matrix, because SVD factorizes a matrix $\boldsymbol{X}$ into $\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\mathsf{T}$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal $p \times n$ matrices, and $\boldsymbol{S}$ is a diagonal matrix, whose diagonal values are decreasing, $s_1 \geq s_2 \geq \cdots > s_n$. Defining $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n]$, where each $\boldsymbol{u}_i \in \mathbb{R}^p$, then $\boldsymbol{u}_i$ is the i^{th} eigenvector, and s_i is the square root of the i^{th} eigenvalue of $\boldsymbol{X}\boldsymbol{X}^\mathsf{T}$. Let $\boldsymbol{A}_d^{PCA} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]$ be the truncated PCA orthonormal matrix.
The PCA matrix is perhaps the most obvious choice of a orthonormal matrix for several reasons. First, truncated PCA minimizes the squared error loss between the original data matrix and all possible rank d representations:

$$\underset{A \in \mathbb{R}^{d\times p}: \boldsymbol{A}\boldsymbol{A}^\mathsf{T} = \boldsymbol{I}_{d\times d}}{\operatorname{argmin}} \left\| \boldsymbol{X} - \boldsymbol{A}^T\boldsymbol{A} \right\|_F^2 .$$

Second, the ubiquity of PCA has led to a large number of highly optimized numerical libraries for computing PCA (for example, LAPACK (**?**)).
Moreover, let $\boldsymbol{U}_d = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d] \in \mathbb{R}^{p \times d}$, and note that $\boldsymbol{U}_d^\mathsf{T}\boldsymbol{U}_d = \boldsymbol{I}_{d\times p}$ and $\boldsymbol{U}_d^\mathsf{T}\boldsymbol{U}_d = \boldsymbol{I}_{p\times d}$. Similarly, let $\boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\mathsf{T} = \boldsymbol{\Sigma}$, and $\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\mathsf{T} = \boldsymbol{\Sigma}^{-1}$. Let $\boldsymbol{S}_d$ be the matrix whose diagonal entries are the eigenvalues, up to the d^{th} one, that is $\boldsymbol{S}_d(i,j) = s_i$ for $i = j \leq d$ and zero otherwise. Similarly, $\boldsymbol{\Sigma}_d = \boldsymbol{U}\boldsymbol{S}_d\boldsymbol{U}^\mathsf{T} = \boldsymbol{U}_d\boldsymbol{S}_d\boldsymbol{U}_d^\mathsf{T}$.
Let $g_{PCA_d} := g_{A_d^{PCA}}$, and let $L_{PCA_d} := L_{A_d^{PCA}}$. And let $g_{LDA_d} := \mathbb{I}\{x\boldsymbol{\Sigma}_d^{-1}\boldsymbol{\delta} > 0\}$ be the regularized LDA classifier, that is, the LDA classifier, but sets the bottom $p - d$ eigenvalues to zero.

Lemma 3. $L_{A_d^{PCA}} = L_{LDA_d}$.

Proof. Plugging $\boldsymbol{U}_d$ into Eq. (2) for $\boldsymbol{A}$, and considering only the left side of the operand, we have

$$\begin{aligned}(\boldsymbol{A}\boldsymbol{x})^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A &= \boldsymbol{x}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{A}\boldsymbol{\Sigma}^{-1}\boldsymbol{A}^\mathsf{T}\boldsymbol{A}\boldsymbol{\delta},\\ &= \boldsymbol{x}^\mathsf{T}\boldsymbol{U}_d\boldsymbol{U}_d^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}_d\boldsymbol{U}_d^\mathsf{T}\boldsymbol{\delta},\\ &= \boldsymbol{x}^\mathsf{T}\boldsymbol{U}_d\boldsymbol{U}_d^\mathsf{T}\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}\boldsymbol{U}_d\boldsymbol{U}_d^\mathsf{T}\boldsymbol{\delta},\\ &= \boldsymbol{x}^\mathsf{T}\boldsymbol{U}_d\boldsymbol{I}_{d\times p}\boldsymbol{S}^{-1}\boldsymbol{I}_{p\times d}\boldsymbol{U}_d^\mathsf{T}\boldsymbol{\delta},\\ &= \boldsymbol{x}^\mathsf{T}\boldsymbol{U}_d\boldsymbol{S}_d^{-1}\boldsymbol{U}_d^\mathsf{T}\boldsymbol{\delta},\\ &= \boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}_d^{-1}\boldsymbol{\delta}.\end{aligned}$$

□

The implication of this lemma is that if one desires to implement Fisherfaces, rather than first learning the eigenvectors and then learning FLD, one can instead directly implement regularized FLD by setting the bottom $p - d$ eigenvalues to zero.

I.C(3) Linear Optimal Low-Rank (LOL) Projection

The basic idea of LOL is to let $\boldsymbol{A}_d^{LOL} = [\boldsymbol{\delta}, \boldsymbol{A}_{d-1}^{PCA}]$. In other words, we simply concatenate the mean difference vector with the top $d - 1$ principal components. Technically, to maintain orthonormality, we must orthonormalize, $\boldsymbol{A}_d^{LOL} = \text{ORTH}([\boldsymbol{\delta}, \boldsymbol{A}_{d-1}^{PCA}])$. Below, we show that this orthonormalization does not matter very much.

I.C(4) FLD is rotationally invariant

For certain classification tasks, the ambient coordinates have intrinsic value, for example, when simple interpretability is desired. However, in many other contexts, interpretability is less important (**?**). When the exploitation task at hand is invariant to rotations, then we have no reason to restrict our search space to be sparse in the ambient coordinates, rather, for example, we can consider sparsity in the eigenvector basis. Fisherfaces is one example of a rotationally invariant classifier, under certain model assumptions. Let $\boldsymbol{W}$ be a rotation matrix, that is $\boldsymbol{W} \in \mathcal{W} = \{\boldsymbol{W} : \boldsymbol{W}^\mathsf{T} = \boldsymbol{W}^{-1}$ and $\det(\boldsymbol{W}) = 1\}$. Moreover, let $\boldsymbol{W} \circ F$ denote the distribution F after rotation by $\boldsymbol{W}$. For example, if $F = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ then $\boldsymbol{W} \circ F = \mathcal{N}(\boldsymbol{W}\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\mathsf{T})$.

Definition 1. *A rotationally invariant classifier has the following property:*

$$L_g^F = L_g^{W\circ F}, \qquad F \in \mathcal{F}.$$

In words, the Bayes risk of using classifier g on distribution F is unchanged if F is first rotated, for any $F \in \mathcal{F}$.

Now, we can state the main lemma of this subsection: FLD is rotationally invariant.

Lemma 4. $L_{Fld}^F = L_{Fld}^{W\circ F}$, *for any $F \in \mathcal{F}$.*

Proof. FLD simply becomes thresholding $\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$. Thus, we can demonstrate rotational invariance by demonstrating that $\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$ is rotationally invariant.

$$
\begin{aligned}
(\boldsymbol{W}\boldsymbol{x})^\mathsf{T}(\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\mathsf{T})^{-1}\boldsymbol{W}\boldsymbol{\delta} &= \boldsymbol{x}^\mathsf{T}\boldsymbol{W}^\mathsf{T}(\boldsymbol{W}\boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\mathsf{T}\boldsymbol{W}^\mathsf{T})^{-1}\boldsymbol{W}\boldsymbol{\delta} && \text{by substituting } \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\mathsf{T} \text{ for } \boldsymbol{\Sigma}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{W}^\mathsf{T}(\widetilde{\boldsymbol{U}}\boldsymbol{S}\widetilde{\boldsymbol{U}}^\mathsf{T})^{-1}\boldsymbol{W}\boldsymbol{\delta} && \text{by letting } \widetilde{\boldsymbol{U}} = \boldsymbol{W}\boldsymbol{U}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{W}^\mathsf{T}(\widetilde{\boldsymbol{U}}\boldsymbol{S}^{-1}\widetilde{\boldsymbol{U}}^\mathsf{T})\boldsymbol{W}\boldsymbol{\delta} && \text{by the laws of matrix inverse}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{W}^\mathsf{T}\boldsymbol{W}\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\mathsf{T}\boldsymbol{W}^\mathsf{T}\boldsymbol{W}\boldsymbol{\delta} && \text{by un-substituting } \boldsymbol{W}\boldsymbol{U} = \widetilde{\boldsymbol{U}}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\mathsf{T}\boldsymbol{\delta} && \text{because } \boldsymbol{W}^\mathsf{T}\boldsymbol{W} = \boldsymbol{I}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta} && \text{by un-substituting } \boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\mathsf{T} = \boldsymbol{\Sigma}
\end{aligned}
$$

□

One implication of this lemma is that we can reparameterize without loss of generality. Specifically, defining $\boldsymbol{W} := \boldsymbol{U}^\mathsf{T}$ yields a change of variables: $\boldsymbol{\Sigma} \mapsto \boldsymbol{D}$ and $\boldsymbol{\delta} \mapsto \boldsymbol{U}^\mathsf{T}\boldsymbol{\delta} := \widetilde{\boldsymbol{\delta}}$, where $\boldsymbol{D}$ is a diagonal covariance matrix. Moreover, let $\boldsymbol{d} = (\sigma_1, \ldots, \sigma_D)^\mathsf{T}$ be the vector of eignevalues, then $\boldsymbol{D}^{-1}\widetilde{\boldsymbol{\delta}} = \boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}}$, where $\odot$ is the Hadamard (entrywise) product. The FLD classifier may therefore be encoded by a unit vector, $\widetilde{\boldsymbol{d}} := \frac{1}{m}\boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}}$, and its magnitude, $m := \left\|\boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}}\right\|$.

I.C(5) Rotation of Projection Based Linear Classifiers g_A

By a similar arguement as above, one can easily show that:

$$
\begin{aligned}
(\boldsymbol{A}\boldsymbol{W}\boldsymbol{x})^\mathsf{T}(\boldsymbol{A}\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\mathsf{T}\boldsymbol{A}^\mathsf{T})^{-1}\boldsymbol{A}\boldsymbol{W}\boldsymbol{\delta} &= \boldsymbol{x}^\mathsf{T}(\boldsymbol{W}^\mathsf{T}\boldsymbol{A}^\mathsf{T})(\boldsymbol{A}\boldsymbol{W})\boldsymbol{\Sigma}^{-1}(\boldsymbol{W}^\mathsf{T}\boldsymbol{A}^\mathsf{T})(\boldsymbol{A}\boldsymbol{W})\boldsymbol{\delta}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{Y}^\mathsf{T}\boldsymbol{Y}\boldsymbol{\Sigma}^{-1}\boldsymbol{Y}^\mathsf{T}\boldsymbol{Y}\boldsymbol{\delta}\\
&= \boldsymbol{x}^\mathsf{T}\boldsymbol{Z}\boldsymbol{\Sigma}^{-1}\boldsymbol{Z}^\mathsf{T}\boldsymbol{\delta}\\
&= \boldsymbol{x}^\mathsf{T}(\boldsymbol{Z}\boldsymbol{\Sigma}\boldsymbol{Z}^\mathsf{T})^{-1}\boldsymbol{\delta} = \boldsymbol{x}^\mathsf{T}\widetilde{\boldsymbol{\Sigma}}_d^{-1}\boldsymbol{\delta},
\end{aligned}
$$

where $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{W} \in \mathbb{R}^{d\times p}$ so that $\boldsymbol{Z} = \boldsymbol{Y}^\mathsf{T}\boldsymbol{Y}$ is a symmetric $p \times p$ matrix of rank d. In other words, rotating and then projecting is equivalent to a change of basis. The implications of the above is:

Lemma 5. *g_A is rotationally invariant if and only if span(A)=span($\boldsymbol{\Sigma}_d$). In other words,* PCA *is the only rotationally invariant projection.*

I.C(6) Simplifying the Objective Function

Recalling Eq. (2), a projection based classifier is effectively thresholding the dot product of x with the linear projection operator $\boldsymbol{P}_A := \boldsymbol{A}^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A \in \mathbb{R}^p$. Unfortunately, the nonlinearity in in Eq. (3) makes analysis difficult. However, because of the linear nature of the classifier and projection matrix operator, an objective function that is simpler to evaluate is available:

$$
\begin{aligned}
\underset{\boldsymbol{A}}{\text{minimize}} \quad & -\frac{\boldsymbol{P}_A^\mathsf{T}\boldsymbol{P}_*}{||\boldsymbol{P}_A||_2||\boldsymbol{P}_*||_2},\\
\text{subject to} \quad & \boldsymbol{A} \in \mathbb{R}^{p\times d}, \quad \boldsymbol{A}\boldsymbol{A}^\mathsf{T} = \boldsymbol{I}_{d\times d}.
\end{aligned} \tag{4}
$$

Lemma 6. *The solution to Eq. (4) is also the solution to Eq. (3) for any given d.*

Proof. The minimum of Eq. (4) is clearly $\boldsymbol{A} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$, which is also the minimum of Eq. (3). □

Define $\angle(\boldsymbol{P}, \boldsymbol{P}') = \frac{\boldsymbol{P}^\mathsf{T}\boldsymbol{P}'}{||\boldsymbol{P}||_2||\boldsymbol{P}'||_2} \in (0, 1)$. Let $\boldsymbol{P}_* = \boldsymbol{P}_{A_*} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$, and $\alpha_A^* = \angle(\boldsymbol{P}_*, \boldsymbol{P}_A)$. A corollary to the above is:

Corollary 7. $\angle(\boldsymbol{P}_A, \boldsymbol{P}_*) \leq \angle(\boldsymbol{P}_{A'}, \boldsymbol{P}_*) \implies L_A \leq L_{A'}$.

Proof. i'm not sure this is true. □

Note that Corollary 7 is a stronger statement than Lemma 6, and in particular, Corollary 7 implies Lemma 6. Given the above, we can evaluate various choices of $\boldsymbol{A}$ in terms of their induced projection operator $\boldsymbol{P}_A$ and the angle between said projection operators and the Bayes optimal projection operator.

I.C(7) Evaluating Different Projections using Eq. (4)

Based on the above, rather than operating on the non-convex L_A, we can instead analyze the properties of different $\boldsymbol{A}$ matrices, and their resulting projection matrices, $\boldsymbol{P}_A$, and their angles with respect to $\boldsymbol{P}_*$. More specifically, we would like to prove something like:

Theorem 4.

$$\angle(\boldsymbol{P}_{\texttt{Pca}}, \boldsymbol{P}_*) \leq \angle(\boldsymbol{P}_{\texttt{Lol}}, \boldsymbol{P}_*) \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \text{ where } \boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \boldsymbol{A}). \tag{5}$$

This would mean that for any $\boldsymbol{\theta}$, Lol would yield a projection closer than Pca to the optimal projection. Recall some basic probability theory which we will use in the sequel. The distribution $\boldsymbol{X}$ is actually mixture of Gaussians: $\boldsymbol{X} \sim \sum_j \pi_j \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma})$. Assume that $\boldsymbol{X}$ is mean centered without loss of generality, to simplify notation. Further assume that we only have two classes, so the number of mixture components is two, and $\mu_0 = -\mu_1$ and $\boldsymbol{\delta} = 2\boldsymbol{\mu}_0 = 2\boldsymbol{\mu}$. When $\boldsymbol{\Sigma}$ is diagonal, this factorizes: $\boldsymbol{X} \sim \prod_{i \in [p]} \sum_j \pi_j \mathcal{N}(\mu_{ij}, \sigma_i^2)$. Consider each X_i separately therefore, we have: $X_i \sim \sum_j \pi_j \mathcal{N}(\mu_{ij}, \sigma_i^2)$, where mean($X_i$)= $\boldsymbol{0}$ by assumption and var(X_i)= $\sum_j \pi_j \mu_{i,j}^2 \sigma_i^2 = \sum_j \pi_j \mu_i^2 \sigma_i^2$, where the second equality follows from the centered mean assumption. To build up to being able to prove Theorem 4, we start very simply.

Lemma 8. *When $d = 1$, so $\boldsymbol{A} \in \mathbb{R}^p$ is a p-dimensional vector, and $\boldsymbol{\Sigma} = \boldsymbol{I}$ and $\pi_0 = \pi_1$. Then (i) $\angle(\boldsymbol{P}_{\texttt{Lol}}, \boldsymbol{P}_*) = 1$ and (ii) $\angle(\boldsymbol{P}_{\texttt{Pca}}, \boldsymbol{P}_*) \leq 1$.*

Proof. First note that when $\boldsymbol{\Sigma} = c\boldsymbol{I}$, that $\boldsymbol{A}_* = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta} = \boldsymbol{\delta}$. Moreover, note that by definition, the first projection vector of $\boldsymbol{A}_{\texttt{Lol}}$ is $\boldsymbol{\delta}$. Therefore, (i) the lemma is immediate.
Now, to obtain the first principle component, consider the general definition of variance, and let each $\sigma_i = 1$, such that the variance of the i^{th} dimension is var(X_i)=$\sum_j \pi_j \mu_i^2 = \mu_i^2$. In such a scenario, the eigenvector with the largest eigenvalue (the direction of maximal variance) will be $\boldsymbol{u}_1 = (\mu_1^2, \ldots, \mu_p^2)$. Let $\widetilde{\boldsymbol{u}}_1 = \boldsymbol{u}_1 / \left\|\boldsymbol{u}_1\right\|$ and $\widetilde{\boldsymbol{\delta}} = \boldsymbol{\delta} / \left\|\boldsymbol{\delta}\right\|$. Then, $\widetilde{\boldsymbol{u}}_1^{\mathsf{T}} \widetilde{\boldsymbol{\delta}} = 1$ if and only if $\mu_i \in \{\mu, 0\}$, for some $\mu \in \mathbb{R}$. □

Lemma 9. *$\boldsymbol{A}_{\texttt{Pca}_1} = \boldsymbol{A}_{\texttt{Lol}_1} = \boldsymbol{A}_* = \widetilde{\boldsymbol{d}}$ when $\boldsymbol{\Sigma} = \boldsymbol{D}$, $\pi_0 = \pi_1$, and $\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1 = \boldsymbol{0}$.*

Proof. Recall that $\widetilde{\boldsymbol{d}} = \boldsymbol{d} \odot \boldsymbol{\delta}$, where $\boldsymbol{d}$ is the diagonal of $\boldsymbol{D}$, and $\odot$ is the Hadamard (elementwise) product. By definition, the first dimension of Lol is $\widetilde{\boldsymbol{d}}$, which proves the second equality.
The variance of the i^{th} dimension is var(X_i)=var($\pi_0 \mathcal{N}(\mu_{i,0}, \sigma_i^2) + \pi_1 \mathcal{N}(\mu_{i,1}, \sigma_i^2)$)= $\sum_j \pi_j \mu_{i,j}^2 \sigma_i^2$. □

Another good thing to prove would be

Theorem 5.

$$\boldsymbol{P}_{\texttt{Pca}}^{\mathsf{T}} \boldsymbol{P}_* / \left\|\boldsymbol{P}_{\texttt{Pca}}\right\| \left\|\boldsymbol{P}_*\right\| - \boldsymbol{P}_{\texttt{Lol}}^{\mathsf{T}} \boldsymbol{P}_* / \left\|\boldsymbol{P}_{\texttt{Lol}}\right\| \left\|\boldsymbol{P}_*\right\| < t \text{ whenever } \boldsymbol{\delta} \in \mathcal{X}, \boldsymbol{\Sigma} \in \mathcal{Y}, \boldsymbol{A} \in \mathcal{Z},$$

for suitable $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$.

This would mean that Lol is better than Pca as a projection.

I.C(8) Probabilistic Extensions of the above

Theorem 6.

$$\mathbb{P}[\boldsymbol{P}_{\texttt{Pca}}^{\mathsf{T}} \boldsymbol{P}_* - \boldsymbol{P}_{\texttt{Lol}}^{\mathsf{T}} \boldsymbol{P}_* > t \left\|\boldsymbol{P}_A\right\| \left\|\boldsymbol{P}_*\right\|] < f(t, p, d),$$

which would state that Lol is better than Pca, again, under suitable assumptions.
In terms of distributiosn of the above, it seems that perhaps we could start simple. Assume for the moment that $\boldsymbol{\delta}, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_p \overset{iid}{\sim} \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$, and let $\boldsymbol{\Lambda} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p)^{\mathsf{T}}$, and $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{\mathsf{T}} \boldsymbol{\Lambda}$.

The reason the above is probabilistic is because it is under certain assumptiosn on the *distributions* of $bdel$, $\boldsymbol{\Sigma}$, and $\boldsymbol{A}$.
Perhaps even simpler is to start with specific assumptions about $\boldsymbol{\delta}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{A}$. Because FLD is rotationally invariant, I believe that we can assert, without loss of generality, that $\boldsymbol{\Sigma} = \boldsymbol{D}$, where $\boldsymbol{D}$ is a diagonal matrix with diagonal entries $\sigma_1, \ldots, \sigma_p$, where all $\sigma_j > 0$. Now, the optimal projection $\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$ is just a simple dot product, $\boldsymbol{d}^{\mathsf{T}}\boldsymbol{\delta}$, where $\boldsymbol{d}$ =diag($\boldsymbol{D}$)$\in \mathbb{R}^p$.
For example, letting $\boldsymbol{A} = \boldsymbol{U}_d$, and letting $\boldsymbol{U}_i = e_i$ be the unit vector, with zeros everywhere except a one in the i^{th} position, we have

$$\boldsymbol{P}_A^{\mathsf{T}}\boldsymbol{P}_* = \boldsymbol{\delta}^{\mathsf{T}}\boldsymbol{U}_d^{\mathsf{T}}\boldsymbol{U}_d\boldsymbol{\Sigma}^{-1}\boldsymbol{U}_d^{\mathsf{T}}\boldsymbol{U}_d\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}\boldsymbol{\delta}^{\mathsf{T}}\boldsymbol{\Sigma}_d\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_d\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta} = \boldsymbol{\delta}^{\mathsf{T}}\boldsymbol{\Sigma}^{-2}\boldsymbol{\delta}.$$

So, we want to understand the probability that α_{PCA} is small under different parameter settings, $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

I.D Learning Classifiers from Data

In real data problems, however, the true joint distribution is typically not provided. Instead, what is provided is a set of training data. We therefore assume the existence of n training samples, each of which has been sampled identically and independently from the same distribution, $(\boldsymbol{X}_i, Y_i) \overset{iid}{\sim} P_{\boldsymbol{X},Y}$, for $i = 1, 2, \ldots, n$. We can use these training samples to then estimate $f_{x|y}$ and f_y. Plugging these estimates in to Eq. (**??**), we obtain the Bayes plugin classifier:

$$\hat{g}_n^*(\boldsymbol{x}) := \operatorname*{argmax}_y \hat{p}_{\boldsymbol{x}|y}\hat{p}_y. \tag{6}$$

Under suitable conditions, it is easy to show that this Bayes plugin classifiers performance is asymptotically optimal. Formally, we know that: $L_{\hat{g}_n^*} \to L_{g^*}$.
When the parameters, $\boldsymbol{\Sigma}$ and $\boldsymbol{\delta}$ are unknown, as in real data scenarios, we can use the training samples to estimate them, and plug them in, as in Eq. (6):

$$\hat{g}_n^*(\boldsymbol{x}) := \mathbb{I}\{\boldsymbol{x}^{\mathsf{T}}\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\delta}} > 0\}. \tag{7}$$

This Bayes plugin classifier is called Fisher's Linear Discriminant (FLD; in contrast to LDA, which uses the true—not estimated—parameters). Unfortunately, when $p \gg n$, the estimate of the covariance matrix $\boldsymbol{\Sigma}$ will be low-rank, and therefore, not invertible (because an infinite number of solutions all fit equally well). In such scenarios, we seek alternative methods, even in the LDA model.

B Bibliography

[1] V. de Silva and J. B. Tenenbaum, “Global Versus Local Methods in Nonlinear Dimensionality Reduction,” in *Neural Information Processing Systems*, 2003, pp. 721–728. 2

[2] H. Zou, “The Adaptive Lasso and Its Oracle Properties,” pp. 1418–1429, 2006. 2

[3] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database.” [Online]. Available: http://yann.lecun.com/exdb/mnist/ 3

[4] Y. Qin and C. E. Priebe, “Maximum L q -Likelihood Estimation via the Expectation-Maximization Algorithm: A Robust Estimation of Mixture Models,” *Journal of the American Statistical Association*, vol. 108, no. 503, pp. 914–928, Sep. 2013. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/01621459.2013.787933#.VYPMzFxVikohttp://www.tandfonline.com.proxy3.library.jhu.edu/doi/abs/10.1080/01621459.2013.787933#.UuJtQmQo7og 6

[5] E. J. Candès and B. Recht, “Exact Matrix Completion via Convex Optimization,” *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, Apr. 2009. [Online]. Available: http://www.springerlink.com/index/10.1007/s10208-009-9045-5 6

[6] L. Breiman, “Random Forests,” Tech. Rep., 2001. [Online]. Available: http://oz.berkeley.edu/users/breiman/RandomForests/cc_papers.htm 11