# Applications of Linear Dimension Reduction

## Three Real Examples and Their Caveats

Dr. Gabriel Odom

# Overview of Topics

- Introduction
- Poorly-Posed Heteroscedastic Classification
- High-Dimensional Classification
- Multi-State Multivariate Process Monitoring
- Conclusion
- Future work
- References

# Introduction

# Introduction

- In multivariate contexts, the number of parameters to estimate increases quadratically with dimension.
- That is, we must estimate $p + \frac{p}{2}(p+1) \in \mathbb{O}(p^2)$ parameters if we have $p$ features.
- Linear dimension reduction (LDR) allows us to simplify complex data structures to more simple data structures via linear combinations of features.
- LDR is built around eigen-decomposition / principal component analysis (PCA) or the singular value decomposition (SVD).

# LDR Benefits

Proper application of LDR can:

- increase model parsimony,
- reduce computational complexity and costs,
- reduce the effect of the *curse of dimensionality*,
- reduce storage costs, and
- reduce feature redundancy.

# Poorly-Posed Heteroscedastic Classification

# The Problem Defined

- We want to classify observations from different elliptical distributions (distributions with negligible higher-order moments).
- Covariance matrices for each class are sufficiently unequal, implying that pooling these estimates will lead to classifier degredation.
- Real examples are often poorly posed ($n_i < p^2/2$), thus class precision matrix estimates are unstable.
- We introduce a shrinkage estimator to class covariance matrices which uses the bias-variance tradeoff to stabilize these etimates.
- We employ dimension reduction to increase classifier speed and accuracy.

# Notation

Consider a data matrix $\mathbf{X}$ containing $n$ observations from $K$ distinct $p$-dimensional elliptical distributions, with class means $\boldsymbol{\mu}_k$ and class covariances $\boldsymbol{\Sigma}_k \in \mathbb{R}_p^{>}$, for $k \in 1, \ldots, K$. Furthermore, define

- $\bar{\mathbf{x}}_k$ is the sample mean vector for the $k^{th}$ class
- $\mathbf{S}_k$ is the sample covariance matrix for the $k^{th}$ class
- $\alpha_k$ is the *a priori* probability of class membership for the $k^{th}$ class
- $\bar{\mathbf{x}} := \sum_{k=1}^{K} \alpha_k \bar{\mathbf{x}}_k$ is the grand mean
- $\mathbf{S}_W := \sum_{k=1}^{K} \alpha_k \mathbf{S}_k$ is the sample within-class covariance
- $\mathbf{S}_B := \sum_{k=1}^{K} \left(\bar{\mathbf{x}}_k - \bar{\mathbf{x}}\right)\left(\bar{\mathbf{x}}_k - \bar{\mathbf{x}}\right)^T$ is the sample between-class covariance
- $\widehat{\boldsymbol{\Gamma}} := \mathbf{S}_B + \mathbf{S}_W$ is the estimated marginal covariance of the data matrix

# Current Approaches

The following are all different methods which calculate a *data sufficiency matrix*, a matrix that contains functions of all the sufficient statistics of the data.

- SY from Ounpraseuth et al. (2015)
- LD from Loog and Duin (2004)
- Sliced Average Variance Estimation (SAVE) from Cook and Weisberg (1991)
- Sliced Inverse Regression (SIR) from K.-C. Li (1991)

# SYS: A New Sufficiency Matrix

Replace the MLE estimators for class covariances with the Haff Shrinkage Estimator in the data sufficiency matrix. That is

$$\widehat{\mathbf{M}}_{SYS} := \Big[\tilde{\mathbf{S}}_2^{-1}\bar{\mathbf{x}}_2 - \tilde{\mathbf{S}}_1^{-1}\bar{\mathbf{x}}_1 \vdots \cdots \vdots \tilde{\mathbf{S}}_K^{-1}\bar{\mathbf{x}}_K - \tilde{\mathbf{S}}_1^{-1}\bar{\mathbf{x}}_1 \vdots \mathbf{S}_2 - \mathbf{S}_1 \vdots \cdots \vdots \mathbf{S}_K -$$

The precision shrinkage estimator is

$$\tilde{\mathbf{S}}_k^{-1} := \left(1 - t(U_k)\right)\left(n_k - p - 2\right)\mathbf{S}_k^{-1} + \frac{t(U_k)\left(pn_k - p - 2\right)}{\mathrm{tr}\left(\mathbf{S}_k\right)}\mathbf{I}_p,$$

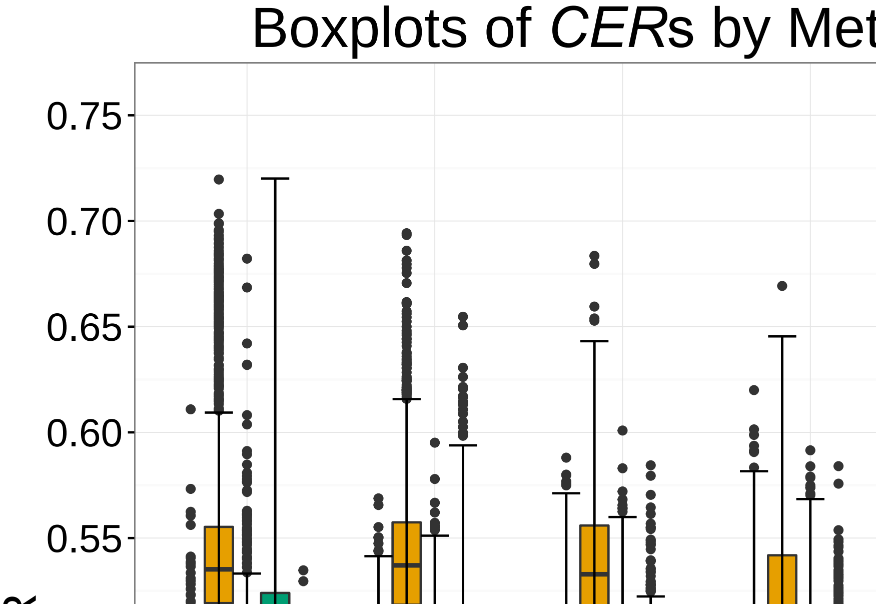
where

$$t(U_k) := \min\left\{\frac{4\left(p^2 - 1\right)}{\left(n_k - p - 2\right)p^2}, 1\right\} U_k^{1/p}, \text{ and } U_k := \frac{p\left|\mathbf{S}_k\right|^{1/p}}{\mathrm{tr}\left(\mathbf{S}_k\right)}.$$

# Simulation Setup

- The synthetic data has three classes.
- We draw 5,000 observations from three $p = 10$-dimensional normal distributions.
- The true data sufficency matrix $\mathbf{M}$ given the known parameters has rank 2.
- We chose $n_i = 15$ as the training sample size to induce a poorly-posed scenario.
- Using the five competing methods, we reduced the data dimension from 10 to $1, \ldots, 9$.
- For each dimension within each method, we built the quadratic classifier and classified the remaining 4,985 observation in the test data set.
- We repeated this process 2,500 times to effectively remove simulation error.

# Simulation Results



# Simulation Results Discussion

- The vertical axis (abscissa) is the conditional error rate (CER). This is the proportion of the nearly-15,000 testing observations which were incorrectly classified.
- The horizontal axis (ordinate) is the reduced feature dimension.
- The coloured boxplots represent the empirical distribution of the 2,500 CERs for each data sufficiency matrix.
- The CER without dimension reduction is the red line at 0.49, which is still better than random guessing for three classes.
- Overall minimum Median CER is 32%, achieved by the SYS method in reduced dimension 2.
- When we increase the ratio of $n$ to $p$, SYS approaches SY (not shown).

# Ionosphere Data

- The radar data from Sigillito et al. (1989) was collected by a signal collection system in Goose Bay, Labrador.
- These observations are measurements on radar signals broadcasted into the ionosphere.
- Based on atmospheric conditions, these signals passed through the ionosphere (bad) or were suitable for further analysis on the ground (good).
- There is a high cost associated with analysing “bad” observations, so we aim to classify an observation before analysis.

# Cross-Validation Setup

- The observations have $p = 32$ recorded continuous features.
- The data has 225 “good” ($n_g$) and 126 “bad” ($n_b$) observations for a total of 351 observations.
- We trained on 40%, 45%, . . . , 90% of the observations.
- We present the results from training with 80% (280) of the observations. This training percent minimised overall CER.
- We repeated non-parametric bootstrap draws 5,000 times.

# Ionosphere Results

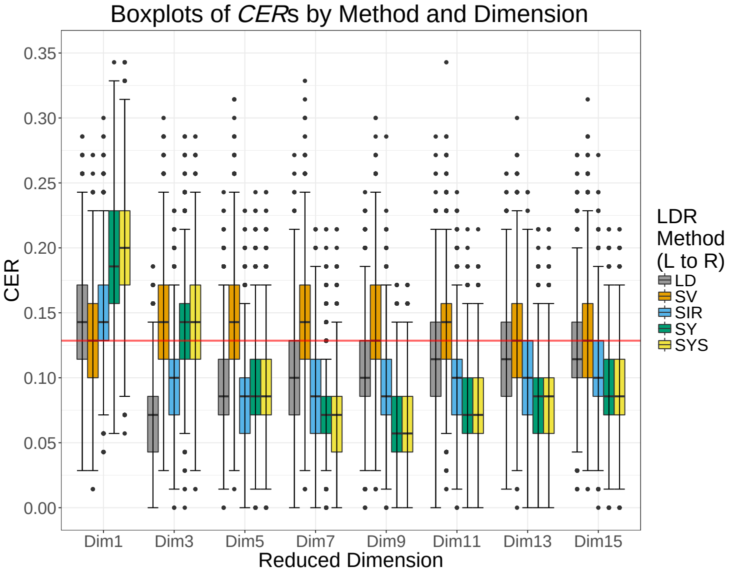


Figure 1

# Results Discussion

- Because the ratio of $n$ to $p$ is larger, SYS and SY perform similarly.
- The Median CER without dimension reduction is 13%.
- The overall minimum Median CER is 5.6% in dimensions 8 and 9, and is attained by the SY and SYS methods.
- With LDR, we reduce the number of parameters to estimate from 1120 to 88, thereby drastically increasing the value of each observation.

# Package `covEst`

- `http://covest.bearstatistics.com/`
- This is our package for generating and estimating matrices.
- Use it to generate Wishart or Singular Wishart covariance matrices for simulation and estimator testing.
- Given a data matrix or data sufficent statistics, we can use it to calculate different covariance matrix estimators.
- The `covEst` package is a “helper” package for `slidR`.

# Package slidR

- http://slidr.bearstatistics.com/
- This package contains a collection of heteroscedastic linear dimension reduction methods useful for supervised learning.
- We designed it from the ground up to utilize pipe operators and other aspects of Hadley's tidyverse
- Minimum working example:

```r
library(tidyverse)
library(modelr)
library(slidR)

df <- iris %>% group_by(Species) %>% crossv_mc(100)
classifier <- df$train %>% map(SYS, targetDim = 1) %>% map(
predicted <- map2(classifier, df$test, predict)
```

# High-Dimensional Classification

# The Problem Defined

- Micro-array and -omics data are often ill-conditioned data sets ($p >> n$).
- Some classification methods are severely degraded in ill-posed scenarios.
- Other classification methods fail entirely.
- Use LDR to reduce the feature space of the data and improve classification opportunities.

# Current Approaches

- BagBoosting; Dettling (2004)
- Random Forests (RF); Breiman (2001)
- Support Vector Machines (SVM); Cristianini and Shawe-Taylor (2000)
- k Nearest Neighbours (kNN); Cunningham and Delany (2007)
- Diagonal LDA and QDA; Pang, Tong, and Zhao (2009)
- Boosting; Schapire (1990)
- Nearest Shrunken Centroids (PAM); Choi, Bair, and Lee (2017)
- Sparse LDA and QDA; Q. Li and Shao (2015)
- Friedman's Regularized Discriminant Analysis (RDA); Guo, Hastie, and Tibshirani (2007)
- High-Dimensional RDA (HDRDA); Ramey et al. (2016)

# A New LDR Algorithm

Split the observations into testing and training sets with sample sizes $n_{\text{Test}} + n_{\text{Train}} = N$.

Calculate the Singular Value Decomposition on the $p$-dimensional training data, and create the lossless $p \times n_{\text{Train}}$ linear projection matrix. As shown in T. Hastie, Tibshirani, and Friedman (2001) (Section 18.3.5), this projection preserves *all* of the information in the training observations.

Project the training data to $n_{\text{Train}}$, and calculate a data sufficiency matrix.

Calculate the Singular Value Decomposition on the $n_{\text{Train}}$-dimensional training data, then create a $n_{\text{Train}} \times q$ linear projection matrix, where $q < n_{\text{Train}}$.

Reduce the dimension of the training and test data sets, and use these data sets to construct and test your favorite classifier.

# Sparse Correlation LDR

- After LDR, calculate class covariance matrices.
- Decompose these class covariances into their respective correlation matrices.
- Truncate all non-significant correlations based on the standard error under the null hypothesis from Fisher's transformation of the Pearson coefficient.
- Recompose the newly-sparsed covariance matrix.
- Perform LDR via sparse SVD routines to reduce the number of parameters to estimate and decrease computational cost.
- Classify the test observations in the lower-dimensional space.

# Real Data

Alon et al. (1999) present a data set on colon cancer. It has 2,000 gene expression measurements on 62 patients, 22 normal patients and 40 cases. Currently, the best mean CER of 0.1040 has been attained by Q. Li and Shao (2015) with their Sparse Quadratic Discriminant Analysis method. The study we compare to uses these parameters:

- Randomly select a training set with 13 normal observations and 29 cases.
- Hold aside the other 9 normal observations and 11 cases as a test data set.
- Train a classifer and test it, recording the CER for that permutation.
- Repeat this sampling and classification 50 times.

We matched their design in every way, but increased the replications from 50 to 1000, reducing the simulation standard deviation to less than 0.005. Further, their classification was in the original feature space, which is far more computationally expensive.

## Conditonal Error Rates Table

The CER of our sparse correlation method has the following distribution (in three-dimensional feature space):

| | Min | Q1 | Median | Mean | Q3 | Max |
|---|---|---|---|---|---|---|
| Error | 0.09054 | 0.09950 | 0.10188 | 0.10229 | 0.10457 | 0.11591 |

Other methods have the following mean CERs, as given by Dettling (2004):

| | BagBoost | RF | SVM | kNN | DLDA | Boosting |
|---|---|---|---|---|---|---|
| Mean ER | 0.1610 | 0.1486 | 0.1505 | 0.1638 | 0.1286 | 0.1914 |

# Results Discussion

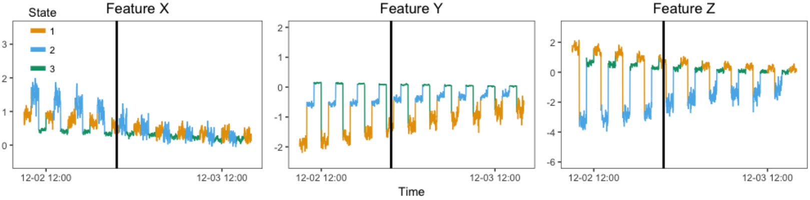
- Our method is at minimum non-inferior—and possibly superior—to the current world's best CER for this colon cancer data.
- Our method uses simple, pre-existing code to perform classification. Specifically, we have used both Friedman's RDA and QDA with squelched-correlation sparse covariance estimation as our classifiers.
- Q. Li and Shao (2015) presented a complex method without published code or packages.
- In contrast, our method is computationally simple and quick, taking less than 3 seconds for classification in a specified dimension for Friedman's RDA (which requires internal cross-validation).

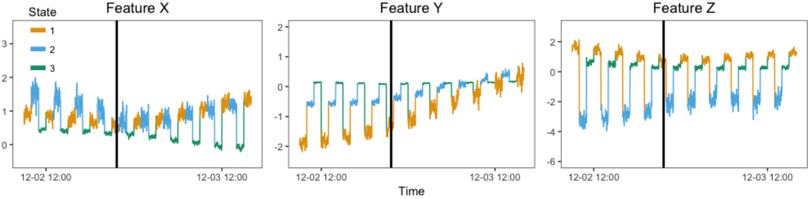
# Multi-State Multivariate Process Monitoring

# The Problem Defined

- Many factories or other closed production systems use real-time online process monitoring.
- We aim to detect potential problems within the system before they cause damage or cause the process to shut down.
- The process has many sensors (a time series for each).
- These sensor readings are not independent across time, and their values may have daily trends / perturbations (autocorrelated and non-stationary).
- The sensors themselves are not independent (cross-correlation of the multiple time series).
- Compute the Squared Prediction Error (SPE) and Hotelling's $T^2$ process monitoring statistics regularly to check divergence from Normal Operating Conditions (NOC).

# Example Process Graphs: NOC vs. Fault





# Current Approaches

PCA fails because of the autocorrelated and non-linearity / non-stationarity of the data. These are a few of the methods currently employed:

- Adaptive-Dynamic PCA (AD-PCA) of Kazor et al. (2016)
- Kernel PCA (kPCA) of Ge, Yang, and Song (2009)
- Adaptive kPCA (AkPCA) of Chouaib, Mohamed-Faouzi, and Messaoud (2013)
- Local Linear Embedding (LLE) Miao et al. (2013)
- Multi-dimensional scaling and isometric mapping (IsoMap) of Tenenbaum, Silva, and Langford (2000)
- Semidefinite Embedding (SDE) / Maximimum Variance Unfolding (MVU) of Weinberger and Saul (2006)

# MSAD-PCA: Our Contribution

We choose to work with AD-PCA because it is simple in idea and computation and has non-inferior to superior results to more complicated methods. We followed this train of thought:

- As the process-dimension increases, computation of process statistics increase cubically. We must reduce the data dimension (PCA).
- Feature distributions change over time and are serially correlated (Adaptive-Dynamic PCA).
- Some processes have multiple states.
- Examples include brain waves during different parts of the sleep cycle or chemical concentrations in a tank during different cleaning steps.
- These states are highly discrete and can cause data matrix instability (near-0 variance).
- Feature distributions change with different known process states, so block on them (Multi-State ADPCA).

# Synthetic Data Fault Detection Time

We present the distribution of time in minutes after synthetic fault induction until the first alarm by each linear projection method. We also record the censoring percentatge for each method (OOB%), which states what percentage of the time the specified method failed to detect a fault within 24 hours. Finally, we also include the expected number of false alarms per day by method.

| | 0.05 | Mean | 0.95 | OOB% | False Alarm % | False Alar |
|---|---:|---:|---:|---:|---|---|
| MSAD SPE | 264 | 370 | 621 | 0% | 0.2% | 2.88 |
| MSAD T2 | 364 | 493 | 533 | 0% | 0.0% | 0 |
| AD SPE | Inf | Inf | Inf | 100% | 0.0% | 0 |
| AD T2 | 35 | 1114 | 1406 | 2.3% | 1.5% | 21.6 |

# Real Data

- This work is motivated by our partnership with the Colorado School of Mines on the ReNUWit water preservation grant.
- Our team manages a decentralised wastewater treatment plant in Golden, CO.
- We measure 40 features and their lagged values (80 features).
- The continuous features are aggregated to the minute-level.
- We aim to develop a monitoring process capable of detecting system faults before human operators.
- We have choices for the blocking variables:
  - Blower operation: controls aeration of the mixture
  - Sequencing Batch Bioreactor Phase: fill, mix, steep, or release
  - Membrane Bioreactor Mode: mixing, cleaning, etc

# Package `mvMonitoring`

- No website: our package is currently in private beta-testing at the facility.
- So far, engineers have been pleased with what we've developed so far, and they are working with us closely to polish the package.
- `mspProcessData()` generates random draws from a serially autocorrelated and nonstationary multi-state (or single-state) multivariate process.
- `mspTrain()` trains the projection matrix and fault detection components.
- `mspMonitor()` assesses incoming process observations and classifies them as normal or abnormal.
- `mspWarning()` keeps a running tally of abnormal observations and raises an alarm if necessary.
- Alarms can be sent via email or SMS from the remote facility to the operators or to the central facility.
- Our next phase goal is integrating an iPhone / Android applet for real-time mobile process updates.

# Summary and References

# Summary

- Described three applications of linear dimension reduction
- Presented three accompanying real data sets
- Discussed three code packages

# Future Work

- Savvy Feature Filtering: use the SVD “backwards” to select features which have strong influence on a classifier.
- Optimal Principal Component Selection: rather than choosing features by the largest eigenvalues, choose features by their contribution to a classifier.

# References

Alon, U., N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. 1999. "Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays." *Proc. Natl. Acad. Sci. U.S.A.* 96 (12): 6745–50.

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32. doi:10.1023/A:1010933404324.

Choi, Byeong Yeob, Eric Bair, and Jae Won Lee. 2017. "Nearest Shrunken Centroids via Alternative Genewise Shrinkages." *PLOS ONE* 12 (2): e0171068. doi:10.1371/journal.pone.0171068.

Chouaib, C., H. Mohamed-Faouzi, and D. Messaoud. 2013. "Adaptive Kernel Principal Component Analysis for Nonlinear Dynamic Process Monitoring." In *Control Conference (ASCC), 2013 9th Asian*, 1–6. doi:10.1109/ASCC.2013.6606291.

Cook, R. Dennis, and Sanford Weisberg. 1991. "Sliced Inverse Regression for Dimension Reduction: Comment." *J. Am. Stat. Assoc.* 86 (414): 328–32. doi:10.2307/2290564.