ONLINE SUFFICIENT DIMENSIONALITY REDUCTION FOR SEQUENTIAL HIGH-DIMENSIONAL TIME-SERIES

A Thesis
Presented to
The Academic Faculty

by

Qingbin Li

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computational Science and Engineering in the
H. Milton Stewart School of Industrial & Systems Engineering (ISyE)

Georgia Institute of Technology
May 2015

ONLINE SUFFICIENT DIMENSIONALITY REDUCTION FOR SEQUENTIAL HIGH-DIMENSIONAL TIME-SERIES

Approved by:

Dr. Yao Xie, Committee Chair
H. Milton Stewart School of Industrial & Systems Engineering (ISyE)
Georgia Institute of Technology

Dr. Le Song
School of Computational Science and Engineering
Georgia Institute of Technology

Dr. Enlu Zhou
H. Milton Stewart School of Industrial & Systems Engineering (ISyE)
Georgia Institute of Technology

Date Approved: Jan 7th 2015

To my parents and all my friends

ACKNOWLEDGEMENTS

I would like to thank the people who have provided support and encouragement throughout my time in the Master program at Georgia Tech.

First and foremost, I would like to thank my advisor, Professor Yao Xie, for her guideline, encouragement and support in the past two years. Thanks for her patiently teaching me when I was lack of background. She directs me toward interesting research topics and helps me a lot not only in research but also in my life and my career. I hope I could live up to her expectation in my future life. I would also like to thank the members of my advisory committee, Dr. Le Song and Dr. Enlu Zhou, for taking the time to review my work and expressing interest in this topic.

Last but not least, I would like to thank my parents, Jiwen and Shuhua, who have given me all the love and support. Thanks for everything they did for me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SUMMARY

In this thesis, we present Online Sufficient Dimensionality Reduction (OSDR) algorithm for real-time high-dimensional sequential data analysis. Real-time high-dimensional sequential data presents several challenges for data analysis. (a) Traditional data analysis may not be applied to high-dimensional data directly, for the consideration of time complexity or model accuracy. (b) It's almost infeasible to use a batch method to process and analyze real-time high-dimensional data. (c) Due to the constrain of memory and data storage, data compression is usually needed for storing and processing real-time high-dimensional data. To deal with challenge (a), dimensionality reduction techniques can be used as a pre-processing step to learn a low-dimensional representation of the high-dimensional data and then apply models or algorithms on the low-dimensional representation of the original high-dimensional data. For solving challenge (b), researchers focus on developing new online learning algorithms or adapting the existing algorithms to the online setting. Moreover, dimensionality reduction is a popular choice for dealing with problem (c).

OSDR solves the challenges by the following characteristics. (a) As an online learning algorithm, OSDR doesn't need to process all the data in order to train the model. (b) As a dimensionality reduction algorithm, by finding the low-dimensional intrinsic subspace of the original space, OSDR projects the high-dimensional data onto the estimated low-dimensional subspace and then regresses on the low-dimensional representations. (c) When the intrinsic subspace dynamically evolves over time, OSDR is able to detect and track the evolution of the subspace. (d) Current online subspace tracking algorithm ignores the label information. However, inspired by the idea of sufficient statistic, OSDR is along the same line with sufficient dimensionality reduction, which enables OSDR to preserve the information provided by the label of the data. In this way, the accuracy of OSDR is improved. (e) OSDR is able to handle missing and noise elements in the feature vectors.

By simulation and real-data experiments, we demonstrate the correctness and effectiveness of OSDR.

CHAPTER I

INTRODUCTION

1.1 Big Data Analysis

In the past decades [24], with the tremendous revolution in technology, big data becomes increasingly common in many fields, including finance, social network, genomics, complex physics simulations, etc. As reported [15][16][17], as of 2012, 2.5 exabytes (2.5×10^{18}) of data were created every day; as of 2014, 2.3 zettabytes (2.3×10^{21}) of data were created every day. Big data is difficult to work with using most traditional data analysis methods and tools. It can be described as the following characteristics [14]:

- **Volume** - The quantity of data that is generated everyday is explosive. For instance, Facebook generates 130 terabytes in data each day, just in user logs. Google processes 25 petabytes each day.
- **Variety** - The data isn't limited to number. People need to process, store and analyze a wide variety of data format, like text, video, audio.
- **Velocity** - The term 'velocity' refer to the speed of generating data and how fast the data is processed and analyzed to meet user's demand.
- **Veracity** - The quality of the data being captured can vary greatly. Accuracy of analysis depends on the veracity of the source data.

Challenges come with big data. On one hand, new data storage and data management tools are needed for processing, storing and managing big data. The most widely used relational database management systems cannot work well with big data. Massive parallel data software running on hundreds, or thousands servers becomes the trend in big data era. On the other hand, traditional data analysis methods might not be applied to big data analysis directly. Efficient and accurate big data analysis algorithms are in urgent need nowadays.

One of the main concern in big data analysis is the dimensionality of the data. The dimension of the data is the number of variables on each observation. Variables is the term mostly used in statistics, while "attribute" and "feature" are commonly used in machine learning and data mining. There are three main challenges presented by high-dimensional data. Firstly, the absolute number of dimensions is high. Instead of dealing with observation with a few tens variables, observations with hundreds, or even thousands of features become a common situation. Secondly, more and more datasets have much more features than observations. For instance, DNA microarray datasets are composed by thousands of variables and a few tens to hundreds of samples [28]. Thirdly, missing data and sparse data often come with high-dimensional data. High-dimensional data analysis present challenges as well as opportunities. To deal with such high-dimensional data, there are two main options. One is to adapt the existing methods to solve high-dimensional data problem directly. Another is to reduce the dimension of the observations and then applied statistical or machine learning methods. The core topic in the second option is dimensionality reduction.

1.2 Dimensionality Reduction

Due to the well known *curse of dimensionality* [26], which refers to the decrease of a learning algorithms with the increase of the number of features, most machine learning and data mining techniques may not be effective for high-dimensional data. Accuracy and efficiency degrade rapidly as the dimension increase. Thus, many researchers improve the efficiency and accuracy by seeking efficient way to reduce the dimension of the data. Dimensionality reduction can be divided into feature selection and feature extraction [8]. *Feature selection* is to select a subset of the most relevant features from the original data set. Feature selection is based on the assumption that the data contains many redundant or irrelevant features. Filter method [27] and wrapper method [20] are the most popular feature selection methods. *Feature extraction* is to transform the high-dimensional data into a meaningful representation of reduced dimensionality without losing information. Ideally, the reduced representation should catch the intrinsic characteristic of the original data. When we talk about dimensionality reduction in the following part, it refers to feature extraction. Generally, dimensionality reduction is used as a data pre-processing step to simplify the data model. By identifying a proper low-dimensional representation of the high-dimensional original data, classification and clustering tasks can be applied to the low-dimensional representation, which can not only often yield more accuracy results but also reduce the computational cost significantly. The motivation of dimensionality reduction can be summarized as follows [8]:

- Identifying a feature set which is most relevant to predictive variables is meaningful from a knowledge discovery perspective.

- Storage cost and computational cost will increase directly with the increase of the number of features.

- Noisy or irrelevant features has the same impact on the predictive variables as relevant features, which will have negative impact on model accuracy.

- Missing values also has negative impact on model.

- High dimensional data is almost impossible to visualize.

1.3 Contributions

In this thesis, we present a dimensionality reduction algorithm, called Online Sufficient Dimensionality Reduction (OSDR), for real-time high-dimensional data. OSDR combines the idea of online learning and sufficient dimensionality reduction. Specifically, in the first step of OSDR, it catches the intrinsic low-dimensional structure of the high-dimensional original data. In the second step of OSDR, it regresses on the low-dimensional linear subsets and learn the regression model. OSDR is able to handle nose and missing elements in the feature vectors. Besides, in many cases, the low dimensional structure dynamically evolves overtime. OSDR will be able to detect and track that evolvement.

We will demonstrate the performance of OSDR by testing it on simulation data and real-world data. The numerical experiments are designed to compare the performance of ODR and OSDR and show the improvement and benefit in efficiency and accuracy of OSDR for high-dimensional sequential data analysis. USPS digits recognition example example shows the performance of OSDR in solving real-world problem.

1.4 Outline

In Chapter 1, we first talk about the challenges and opportunities in big data analysis. Then we introduce the dimensionality reduction technique and discuss its motivation and potential application in big data analysis. After that, the contribution and outline of this thesis are described.

In Chapter 2, we give a preliminary review of techniques related to this thesis, including dimensionality reduction, sufficient dimensionality reduction and online subspace tracking. By comparing the difference between unsupervised dimensionality reduction and supervised dimensionality reduction, prove the advantage of supervised dimensionality reduction over unsupervised dimensionality reduction. Then, give a review of sufficient dimensionality reduction technique, which combines the idea of dimensionality reduction and the concept of sufficient statistic. Finally, we investigate two online algorithms, GROUSE and MOUSSE-LOGIT, which provide theoritical foundation and motivation for our novel algorithm.

In Chapter 3, we analyze the limitation of MOUSSE-LOGIT. Inspired by the idea of sufficient dimensionality reduction, we propose a novel dimensionality reduction algorithm. First, formulate the problem of real-time high-dimensional data analysis. Second, propose an online dimensionality reduction which is derived from GROUSE directly. Third, incorporate the idea of sufficient dimensionality reduction, propose the novel dimensionality reduction algorithm.

In Chapter 4, we present example to study the performance of OSDR. Both simulation data and real-world data are examined and tested. Compared with online dimension reduction algorithm, OSDR shows lower misclassification rate and higher accuracy rate compared with online dimensionality reduction algorithm.

The conclusions and future work are given in Chapter 5.

CHAPTER II

PRELIMINARY

2.1 Dimensionality Reduction

In Chapter 1, we bring the concept of dimensionality reduction. The motivation of dimensionality reduction is also studied. In this part, detailed content of dimensionality reduction will be went over. The problem of dimensionality reduction can be formulated as follows.

Given a D-dimensional variable $x = (x_1, ..., x_D)^T$, assume it has intrinsic low-dimensional structure with dimensionality d (where $d \ll D$). The intrinsic low-dimensional structure implies that the data in or near a manifold with dimensionality d is embedded into the high-dimensional space with dimensionality D. The target of dimensionality reduction is to transform the data x with dimensionality D into a new low-dimensional representation β with dimensionality d ($d \ll D$), while retaining the geometry of the data as much as possible. Generally speaking, dimensionality reduction problem can only be solved with assumption of the property of the data, since neither the intrinsic low-dimensional d nor the geometry of the data manifold are known. [29]

2.1.1 Unsupervised Dimensionality Reduction

Based on whether the learning process is supervised or unsupervised, dimensionality reduction can be divided into unsupervised dimensionality reduction and supervised dimensionality reduction. Traditional, unsupervised dimensionality reduction methods are well studied by researchers in statistics, machine learning. Principal component analysis (PCA) [18] is the most popular (unsupervised) dimensionality reduction technique. PCA is a second-order method. It performs dimensionality reduction by embedding the data into a linear subspace of lower dimensionality. In different fields, PCA is called the Kosambi Karhunen Love theorem [12] and the singular value decomposition (SVD) [34].

2.1.2 Supervised Dimensionality Reduction

While unsupervised dimensionality reduction is a well studied problem, dimensionality reduction technique has not been well explored for the supervised case. When labels of data are available, e.g., in the classification task, unsupervised dimensionality reduction techniques are not able to incorporate this information. It will be helpful and meaningful to incorporate the label information into the dimensionality reduction process and derive a supervised dimensionality reduction for input data. In the past ten years, a lot of supervised dimensionality reduction techniques are proposed.

Instead of finding global discriminants in the data, a framework for supervised subspace sampling was proposed in order to create a reduced representation of the data for classification application [1]. SPPCA extended the probabilistic PCA model to incorporate labels of data into projection and propose a supervised PCA model [33]. In QUADRO model, it analyzed the problem of Rayleigh quotient optimization under nonlinear setting and proposed a novel Rayleigh quotient based sparse quadratic dimension reduction method [10]. Without making assumptions on the distribution of the data classes, which can lead to ad-hoc and sub-optimal implementation, an approach was proposed to take an information-geometric approach by maximizing the between class information distances [4].

2.2 Sufficient Dimensionality Reduction

2.2.1 Introduction

Sufficient dimensionality reduction (SDR) [22] techniques have drawn considerable attention in the analysis of high-dimensional data. It combines the idea of dimensionality reduction with the concept of sufficient statistic [19]. SDR can effectively reduce the dimensional the feature vectors, while retaining full regression information and imposing no parametric models.

Earlier research on SDR, like *sliced inverse regression* [22], *principal Hessian direction* [23], and *sliced average variance estimation* [5] replies on the distribution assumption of the data, which might not be achieved in real-world problem. Among these algorithm, *sliced inverse regression*(SIR) is the most commonly used SDR method, and there are many studies that elaborate on SIR [6].

Recently, kernel-based dimensionality reduction has been a hot topic. *Kernel dimension reduction* (KDR) is one of the most influenced work. It employs a kernel-based dependence measure, which is distribution-free. However, the performance of KDR depends heavily on the choice of kernel functions and the regularization parameters, which is the key limitation of KDR. Besides, the gradient based optimization is computationally demanding which makes KDR scale poorly onto massive datasets.

Most previous work of SDR has only been applied to static data. However, in practice, sequential time-series data is common in many areas, like signal processing [21], computer vision [3], etc. Limited work has been done in this part. *Sequence kernel dimension reduction approach* (S-KDR) [25] combines spatial, temporal and periodic information in a principled manner, and learns an optimal embedding without assuming any distribution of the data. The new designed kernels can capture dynamics, periodic motions and multi-class classification. The advantages of sufficient dimensionality reduction and the lack of research on sufficient dimensionality reduction for modeling time-series data is one of the motivation of this thesis.

2.2.2 Problem Formulation

In the following, we will give a brief problem formulation of sufficient dimensionality reduction [25].

Let x be the feature vector, with $x \in \mathbb{R}^D$ and let y be the label of x. In supervised learning, the goal of sufficient dimensionality reduction is to estimate a low-dimensional representation β, with $\beta \in \mathbb{R}^d$, that is sufficient for the prediction task, where

$$\beta = Ux. \tag{2.2.1}$$

U is the projection matrix to a d-dimensional subspace, with $d \ll D$. The criterion of SDR is formulated as given β, the remaining features of x are conditionally independent of the label y. In this sense, we say x is sufficient for estimating y.

Compared with other supervised dimensionality reduction algorithm or unsupervised dimensionality reduction algorithm, sufficient dimensionality reduction makes no assumption on the distribution of x.

2.3 Online Subspace Tracking

Subspace tracking is a terminology mostly used in computer vision for dimensionality reduction. It has been extensively studied in computer vision with application in background subtraction [30], object tracking [7], etc. Online subspace tracking is a recently proposed topic in this domain. In online subspace tracking, observations are presented sequentially in the form of an unknown mixture of primary subspaces plus a residual component. The target of online subspace tracking is to keep the estimated subspace updating as the observations continually present themselves.

2.3.1 GROUSE

Grassmannian Rank-One Update Subspace Estimation (GROUSE) [2], is an efficient online algorithm for identifying and tracking subspaces from highly incomplete observations. It's derived by analyzing incremental gradient descent and can be easily used for online matrix completion.

Suppose we want to track a D-dimensional space with intrinsic d-dimension subspace evolve over time. At each time t, we only observe feature vector x_t on a subset of indices Ω_t. Let $\mathcal{P}_{\Omega_t}$ denotes a $|\Omega_t| \times D$ matrix which select the coordinate axes of $\mathbb{R}^D$ indexed by Ω_t. At time t, we only observe

$$x_{\Omega_t} \triangleq \mathcal{P}_{\Omega_t} x_t \tag{2.3.1}$$

The error of the our subspace is defined as

$$F(U;t) = \min_w \|U_{\Omega_t} w - x_{\Omega_t}\|_2^2, \tag{2.3.2}$$

where U_{Ω_t} denotes the submatrix of U, the basis of space, consisting the rows indexed by Ω_t.

Grassmannian [11], a compact Riemannian manifold, is a space which parameterizes all linear subspaces of a vector space D of given dimension d. Derived from an application of incremental gradient descent on the Grassmannian manifold, GROUSE computes a gradient of the cost function F and then follow the gradient along a short geodesic curve in Grassmannian [2]. The derivative of F is given by

$$\frac{\partial F}{\partial U} = -2\gamma\omega^T, \tag{2.3.3}$$

where γ is residual vector and is equal to $(x_t - U\omega)$ on Ω_t. ω is the corresponding least square solution of equation (2.3.3). The gradient with respect to the Grassmannian is

$$\nabla F = -2(I - UU^T)\gamma\omega^T. \tag{2.3.4}$$

To ensure that U remains within the Grassmannian manifold, we need to follow the geodesics along the direction of the gradient. The geodesics on Grassmann manifolds is given by Equation (2.65) in [9]. For a step size of length $\eta > 0$, the update is given by

$$U_{new} = U + ((\cos(2\|\gamma\|\|\omega\|\eta) - 1)\frac{p}{\|p\|} + \sin(2\|\gamma\|\|\omega\|\eta)\frac{\gamma}{\|\gamma\|})\frac{\omega^T}{\|\omega\|}, \tag{2.3.5}$$

where $p = U\omega$ is the predicted vector of the project of vector x on the current estimated subspace. The update rule consists of a rank-one modification of the current subspace basis U.

In sum, GROUSE provides an online algorithm for tracking subspaces from incomplete observations based on incremental gradient descent iterations on the Grassmannian manifold of subspace.

2.3.2 MOUSSE-LOGIT

MOUSSE-LOGIT, proposed by Xie [32], is a technique for online logisitc regression when the feature vectors lie close to a dynamic low-dimensional manifold and when observations of the feature vectors may be noisy or have missing elements. The problem is formulated as follows.

Given training data $x_{\Omega_t}, y_t, \Omega_t,\ t = 1, 2, ..$, the feature vector $x_t \in \mathbb{R}^D$ is a noisy realization of true intrinsic vector β_t, which lies on submanifold with dimension d. The noise is a Gaussian noise with zero mean and variance σ^2. At each time t, only x_{Ω_t} on a subset of indices Ω_t is observed. The label $y_t \in 0, 1$ is Bernoulli with probability p of being 1 that depends on β

$$\mathbb{P}(y_t = 1) = h(\beta; \theta, b), \tag{2.3.6}$$

where $h(\beta; \theta, b)$ denotes the logistic model.

$$h(\beta; \theta, b) = \frac{1}{1 + e^{-\theta^T \beta - b}}, \tag{2.3.7}$$

where β is feature vector, θ is weight vector and b is offset or intercept. The goal is to design a classifier for online logistic regression based on streaming data $\{x_{\Omega_t}, y_t, \Omega_t\}$ and exploit the dynamic evolving submanifold of feature vector x_T.

To solve the above problem, MOUSSE-LOGIT combines Multiscale Online Union of Subspace Estimation (MOUSSE) algorithm [31] and stochastic gradient descent method. It approximates the manifold by union of linear subsets, which is organized in a tree structure. The model training process can be summarized as three main steps. (a) Project feature vector x_t onto the current estimated submanifold. (b)

Using MOUSSE to update the submanifold estimation and tree structure. (c) Using stochastic gradient descent to update the logistic model.

Numerical and real-data experiments show that when feature vector has an intrinsic manifold structure, MOUSE-LOGIT have the following benefits. (a) Reduce computational cost by regressing on low-dimensional space. (b) Achieve better performance compared with regressing on the ambient space. (c) Achieve better performance compared with regressing on one single subspace.

From the view of learning process, MOUSSE-Logit can be regarded as unsupervised dimensionality reduction. As discussed before, unsupervised dimensionality reduction can not incorporate data label into learning process, which might result in the loss of information and degrade in model accuracy. To overcome the drawback of MOUSSE-LOGIT and gain a better model performance, a novel supervised dimensionality reduction is needed for this problem. This is one of the motivation of this thesis.

CHAPTER III

ONLINE SUFFICIENT DIMENSIONALITY REDUCTION

3.1 Motivation

In Chapter 2, we review a method, called MOUSSE-LOGIT, for online logistic regression when the high-dimensional feature vectors are sparse and with intrinsic low-dimensional structure. MOUSSE-LOGIT takes the unsupervised learning paradigm, which means the label of the feature vectors isn't considered as a factor when doing dimensionality reduction. Thus, in some cases, the information of the label might be lost and the low-dimensional subspace isn't well represented. The accuracy of dimensionality reduction is not convincing.

To overcome this limitation and improve the accuracy of online dimensionality reduction, we take the label of the feature vectors into consideration. Inspired by the idea of sufficient dimensionality reduction, we proposed a new algorithms called online sufficient dimensionality reduction, which is a paradigm for analyzing real-time high-dimensional sparse data with low-dimensional intrinsic structure that combining the ideas of online dimensionality reduction and the concept of sufficient statistic .

3.2 Logistic Regression

At every time t, the data has feature vector $x_t \in \mathbb{R}^D$, with D denotes the ambient dimension, and a label $y_t \in \{0, 1\}$. Assume the intrinsic characteristic of the feature vector x_t lies on a manifold $\subset \mathbb{R}^D$. The dimension of the manifold is d, where $d \ll D$. That is, feature vector x_t is high-dimensional vector with dimensionality D and has low-dimensional intrinsic structure with dimensionality d. We use $U \in \mathbb{R}^{D\times d}$ denotes the orthogonal basis of the manifold where $U^{\mathrm{T}}U = I \in \mathbb{R}^{d\times d}$. The notation $^{\mathrm{T}}$ denotes the transpose of a matrix or vector. Thus, the feature vector x can be denoted as $x = U\beta$, where β is the coefficient $\in \mathbb{R}^d$ and $\beta = U^{\mathrm{T}}x$. In the online setting, assume the manifold may evolve over time, we use U_t to denote the basis of the manifold at time t. Thus

$$x_t = U_t\beta_t, \tag{3.2.1}$$

where $x_t \in \mathbb{R}^D$, $U_t \in \mathbb{R}^{D\times d}$ and $\beta_t \in \mathbb{R}^d$.

At each time t, we only observe feature vector x_t on a subset of indices Ω_t. Let $\mathcal{P}_{\Omega_t}$ denotes a $|\Omega_t| \times D$ matrix which select the coordinate axes of $\mathbb{R}^D$ indexed by Ω_t. At time t, we observe

$$x_{\Omega_t} \triangleq \mathcal{P}_{\Omega_t}x_t \tag{3.2.2}$$

Define the logistic function as

$$h(x;\theta,b) = \frac{1}{1+e^{-\theta^{\mathrm{T}}x-b}}. \tag{3.2.3}$$

Assume the label y_t is related to feature vector x_t via logistic model

$$y_t \sim \mathrm{Bernoulli}(h(x_t;\theta_t,b_t)), \tag{3.2.4}$$

where $\theta \in \mathbb{R}^D$ denotes the weight vector, $b \in \mathbb{R}$ denotes the intercept or offset.

In sum, the logistic regression problem is that at every time t, given feature vector x_t and its label y_t, assume the feature vectors has intrinsic low-dimensional manifold and the manifold evolves overtime. Besides, the label is related to feature vectors by logistic model. The primary goal is to find an efficient and accurate algorithm for finding and tracking the basis of the manifold U overtime and at the same time learn the corresponding logistic model.

3.3 Linear Regression

The set up of linear regression is similar to the previous logistic regression problem. At every time t, the data has feature vector $x_t \in \mathbb{R}^D$, with D denote the ambient dimension, and a label $y_t \in (-\infty, +\infty)$. Assume the intrinsic characteristic of the feature vector x_t lies on a manifold $\in \mathbb{R}^D$. The dimension of the manifold is d, where $d \ll D$. $U \in \mathbb{R}^{D\times d}$ denotes the orthogonal basis of the manifold where $U^{\mathrm{T}}U = I \in \mathbb{R}^{d\times d}$. The notation $^{\mathrm{T}}$ denotes the transpose of a matrix or vector. Thus, the feature vector x can be denoted as $x = U\beta$, where β is the coefficient $\in \mathbb{R}^d$ and $\beta = U^{\mathrm{T}}x$. In the online setting, assume the manifold may evolve over time, we use U_t to denote the basis of the manifold at time t. Thus

$$x_t = U_t\beta_t, \tag{3.3.1}$$

where $x_t \in \mathbb{R}^D$, $U_t \in \mathbb{R}^{D\times d}$ and $\beta_t \in \mathbb{R}^d$.

At each time t, we only observe feature vector x_t on a subset of indices Ω_t. Let $\mathcal{P}_{\Omega_t}$ denotes a $|\Omega_t| \times D$ matrix which select the coordinate axes of $\mathbb{R}^D$ indexed by Ω_t. At time t, we observe

$$x_{\Omega_t} \triangleq \mathcal{P}_{\Omega_t}x_t \tag{3.3.2}$$

Define the linear regression function as

$$h(x; \theta, b) = \theta^T x + b. \tag{3.3.3}$$

Assume the label y_t is related to feature vector x_t via linear regression function

$$y_t = h(x_t; \theta_t, b_t), \tag{3.3.4}$$

where $\theta \in \mathbb{R}^D$ denotes the weight vector, $b \in \mathbb{R}$ denotes the intercept or offset.

In sum, the linear regression problem is that at every time t, given feature vector x_t and its label y_t, assume the feature vectors has intrinsic low-dimensional manifold and the manifold evolves overtime. Besides, the label is related to feature vectors by linear regression function. The primary goal is to find an efficient and accurate algorithm for finding and tracking the basis of the manifold U overtime and at the same time learn the corresponding linear regression model.

3.4 Online Dimensionality Reduction

In Chapter 2, we review an online algorithm for subspace tracking, called GROUSE. It can be easily adjusted for solving the problem we present above. In the following part, we derive a new online algorithm, called online dimensionality reduction (ODR), based on GROUSE and stochastic gradient descent.

ODR uses GROUSE for online subspace tracking and uses stochastic gradient descent for learning the parameters of regression model. The parameters used in ODR at time t are

$$\{U_t, \eta_t, \theta_t, b_t, \mu\}. \tag{3.4.1}$$

U_t is the basis of the manifold at time t. η_t is the step-sizes in updating manifold. θ_t and b_t, the slope and intercept of the regression model at time t. μ is the step-sizes of updating regression model by using stochastic gradient descent.

3.4.1 Parameters Initialization

The parameters in (3.4.1) are initialized as follows. Randomly generate an orthogonal matrix $\in \mathbb{R}^{D\times d}$ and assigned it to U_0. Choose a constant C as η_0 and η_t is computed as $\eta_t = C/t$. Randomly generate a vector with dimensionality D as the initialization of θ. b_0 and μ is a randomly generated number.

3.4.2 Parameters Update

When a new data sample (x_t, y_t) is available, ODR updates the parameters using three main steps. (a) Given feature vector x_t, compute the projection coefficient β_t in the current manifold estimation with basis U_t. (b) Update the manifold by using GROUSE. (c) Update (θ_t, b_t) by using stochastic gradient descent.

At each time t, we observe a feature vector x_t at locations $\Omega_t \subset \{1, ..., n\}$. Given the manifold with parameters U_t, let Δ_{Ω_t} be a $D \times D$ diagonal matrix which has 1 in the j^{th} diagonal entry if $j \in \Omega_t$ and has 0 otherwise. The projection coefficient β_t of x_t is computed as

$$\beta_t = \arg\min_v \|\Delta_{\Omega_t}(x_t - U_t v)\|^2. \tag{3.4.2}$$

In this way, the algorithm handles the missing elements problem in feature vectors. Then use GROUSE [2] to update the manifold.

3.4.2.1 Logistic Regression

In the following, we focus on updating the parameters of logistic model by maximizing the log-likelihood function. In the online setting, assume the parameters of the logistic model at time $t-1$ is (θ_{t-1}, b_{t-1}). At time t, when we get the feature vector x_t without its label, compute the label using

$$\hat{y}_t = \begin{cases} 1 & \text{if } h(\beta_t;\, \theta_{t-1},\, b_{t-1}) \geq 1/2 \\ 0 & \text{otherwise} \end{cases} \tag{3.4.3}$$

When both feature vector x_t and its label y_t are available, update parameters of logistic model using stochastic gradient descent by maximizing the log-likelihood function.

$$l(\theta, b) = y_t \text{log} h(\beta_t; \theta_{t-1}, b_{t-1}) + (1 - y_t)\text{log}[1 - h(\beta_t; \theta_{t-1}, b_{t-1})]. \tag{3.4.4}$$

Use $[w]_k$ denotes the k-th element of vector w and use $\hat{y}_t$ denotes $h(\beta_t; \theta_{t-1}, b_{t-1})$. The gradient of (3.4.4) is given by

$$\begin{aligned} \frac{\partial l(\theta, b)}{\partial [\theta]_k} &= (y_t \frac{1}{\hat{y}_t} - (1 - y_t)\frac{1}{1 - \hat{y}_t})\frac{\partial}{\partial [\theta]_k}\hat{y}_t \\ &= (y_t \frac{1}{\hat{y}_t} - (1 - y_t)\frac{1}{1 - \hat{y}_t})\hat{y}_t(1 - \hat{y}_t)[\beta_t]_k \\ &= (y_t(1 - \hat{y}_t) - (1 - y_t)\hat{y}_t)[\beta_t]_k \\ &= (y_t - \hat{y}_t)[\beta_t]_k \end{aligned} \tag{3.4.5}$$

$$\begin{aligned} \frac{\partial l(\theta, b)}{\partial b} &= (y_t \frac{1}{\hat{y}_t} - (1 - y_t)\frac{1}{1 - \hat{y}_t})\frac{\partial}{\partial b}\hat{y}_t \\ &= (y_t \frac{1}{\hat{y}_t} - (1 - y_t)\frac{1}{1 - \hat{y}_t})\hat{y}_t(1 - \hat{y}_t) \\ &= (y_t(1 - \hat{y}_t) - (1 - y_t)\hat{y}_t) \\ &= (y_t - \hat{y}_t) \end{aligned} \tag{3.4.6}$$

We can update the parameters $(\theta_{t-1},\ b_{t-1})$ by

$$\begin{aligned} \theta_{\mathrm{t}} &= \theta_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t]\beta_t \\ b_{\mathrm{t}} &= b_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t] \end{aligned} \tag{3.4.7}$$

3.4.2.2 Linear Regression

In this section, we focus on updating the parameters of linear regression model by minimizing the cost function. In the online setting, assume the parameters of the linear regression model at time $t-1$ is $(\theta_{t-1},\ b_{t-1})$. At time t, when we get the feature vector x_t without its label, compute the label using

$$\hat{y}_t = \theta_{t-1}^T \beta_t + b_{t-1} \tag{3.4.8}$$

When both feature vector x_t and its label y_t are available, update parameters of linear regression model using stochastic gradient descent by minimizing the cost function.

$$l(\theta, b) = \frac{1}{2}(h(\beta_t; \theta_{t-1}, b_{t-1}) - y_t)^2 \tag{3.4.9}$$

Use $[w]_k$ denotes the k-th element of vector w and use $\hat{y}_t$ denotes $h(\beta_t; \theta_{t-1}, b_{t-1})$. The gradient of (3.4.9) is given by

$$\begin{aligned} \frac{\partial}{\partial[\theta]_k} l(\theta, b) &= (\hat{y}_t - y_t)\frac{\partial}{\partial[\theta]_k}\hat{y}_t \\ &= (\hat{y}_t - y_t)[\beta_t]_k \end{aligned} \tag{3.4.10}$$

$$\begin{aligned} \frac{\partial}{\partial b} l(\theta, b) &= (\hat{y}_t - y_t)\frac{\partial}{\partial b}\hat{y}_t \\ &= (\hat{y}_t - y_t) \end{aligned} \tag{3.4.11}$$

We can update the parameters $(\theta_{t-1},\ b_{t-1})$ by

$$\begin{aligned} \theta_{\mathrm{t}} &= \theta_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t]\beta_t \\ b_{\mathrm{t}} &= b_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t] \end{aligned} \tag{3.4.12}$$

3.4.3 Conclusion

In sum, the online dimensionality reduction algorithm can be described as follows.

Algorithm 1 Online Dimensionality Reduction

```
Input: An $D \times d$ orthogonal matrix $U_0$. A sequence of vectors $x_t$, each observed in
    entries $\Omega_t$, with its label $y_t$. A set of step-sizes $\eta_t$ for manifold updating. Slope
    vector and offset of logistic model $\theta_0$ and $b_0$. Step-size $\mu$ for stochastic gradient
    descent.
for t = 1, ..., T do
    Estimate projection coefficients: $\beta_t = \arg\min_v \|\Delta_{\Omega_t}(x_t - U_t v)\|^2$
    Compute residual: $\gamma = \Delta_{\Omega_t}(x_t - U_t\beta_t)$
    Compute parameter: $\sigma = \|\gamma\|\|U_t\beta_t\|$
    Update subspace: $U_{t+1} = U_t + ((\cos(\sigma\eta_t) - 1)\frac{U_t\beta_t}{\|U_t\beta_t\|} + \sin(\sigma\eta_t)\frac{\gamma}{\|\gamma\|})\frac{\beta_t^T}{\|\beta_t\|}$
    Predict vector label: $\hat{y}_t = h(\beta_t; \theta_{t-1}, b_{t-1})$
    Update slope vector: $\theta_\mathrm{t} = \theta_\mathrm{t-1} + \mu[y_t - \hat{y}_t]\beta_t$.
    Update residual: $b_\mathrm{t} = b_\mathrm{t-1} + \mu[y_t - \hat{y}_t]$
end for
```

The difference of ODR in solving the linear regression problem and the logistic regression problem lies in the regression function $h(\beta_t; \theta_{t-1}, b_{t-1})$. In linear regression problem,

$$h(\beta_t; \theta_{t-1}, b_{t-1}) = \theta_{t-1}^T \beta_t + b_{t-1}.$$

However, in logistic regression problem,

$$h(\beta_t; \theta_{t-1}, b_{t-1}) = \frac{1}{1 + e^{-\theta_{t-1}^\mathrm{T} \beta_t - b_{t-1}}}.$$

3.5 Online Sufficient Dimensionality Reduction

In many cases, the label y is closely correlated to the input feature x. However, in the existing dimensionality reduction methods, including ODR, they don't take the label y into consideration and use the unsupervised learning paradigm to do the dimensionality reduction. To overcome this limitation, as we have reviewed in Chapter 2, people proposed the sufficient dimensionality reduction algorithm. However, limited research has been done in the online setting about this topic. Thus, to make our approximation of the feature vector to have the best predictive power to the label, instead of using the unsupervised dimensionality reduction method, we use the label y as a factor when doing dimensionality reduction and form a new algorithm, called online sufficient dimensionality reduction (OSDR).

3.5.1 Main Steps

The parameters used in OSDR and the parameters initialization are the same with ODR. When a new data sample $(x_t,\ y_t)$ is available, it takes three main steps: (a) Given feature vector x_t, compute the projection coefficient β_t in the current manifold estimation with basis U_t. (b) Update the manifold by the following update method. (c) Update $(\theta_t,\ b_t)$ by using stochastic gradient descent.

3.5.2 Online Manifold Tracking

Given the manifold with parameters U_t, the projection coefficient β_t of feature vector x_t can be computed by (3.4.2). Then, we focus on developing a new online manifold tracking and updating method, which is the core part of OSDR. To focus on method development, we simply the derivative process by excluding the time stamp t in the following.

3.5.2.1 Logistic Regression

Instead of tracking U by minimizing the distance between the real input feature vector and the approximate vector, we find the basis U by maximizing the log-likelihood ratio, which is along the same line as sufficient dimensionality reduction. The log-likelihood function is defined as

$$f(U, \theta) = y \log h(U\beta; \theta, b) + (1 - y) \log[1 - h(U\beta; \theta, b)]. \tag{3.5.1}$$

where

$$h(U\beta; \theta, b) = \frac{1}{1 + e^{-\theta^{\mathrm{T}} U\beta - b}}.$$

The derivative of $f(U, \theta)$ with respect to U is given by

$$\frac{df}{dU} = [y - h(U\beta; \theta, b)]\theta\beta^{\mathrm{T}}. \tag{3.5.2}$$

The gradient on the Grassmannian is given by

$$\nabla f = [y - h(U\beta; \theta, b)](I - UU^{\mathrm{T}})\theta\beta^{\mathrm{T}}. \tag{3.5.3}$$

Let $r = [y - h(U\beta; \theta, b)](I - UU^{\mathrm{T}})\theta$. Thus,

$$\nabla f = r\beta^{\mathrm{T}}. \tag{3.5.4}$$

For the gradient is rank-one, we can write

$$\nabla f = \begin{bmatrix} r/\|r\| & v_2 & \dots & v_d \end{bmatrix} \mathrm{diag}(\sigma) \begin{bmatrix} \beta/\|\beta\| & z_2 & \dots & z_d \end{bmatrix}^{\mathrm{T}}, \tag{3.5.5}$$

where

$$\sigma = -[y - h(U\beta; \theta, b)]\|r\|\|\beta\|. \tag{3.5.6}$$

$v_2, \dots, v_d$ are an orthonormal set orthogonal to r and $z_2, \dots, z_d$ are an orthonormal set orthogonal to β.

Update of U can be efficiently calculated as

$$\begin{aligned} U_{\text{new}} &= U + \frac{(\cos(\sigma\eta) - 1)}{\|\beta\|^2} U\beta\beta^{\mathrm{T}} + \sin(\sigma\eta)\frac{r}{\|r\|}\frac{\beta^{\mathrm{T}}}{\|\beta\|} \\ &= U + ((\cos(\sigma\eta) - 1)\frac{U\beta}{\|U\beta\|} + \sin(\sigma\eta)\frac{\gamma}{\|\gamma\|})\frac{\beta^{\mathrm{T}}}{\|\beta\|} \end{aligned} \tag{3.5.7}$$

where $\eta > 0$ is a step-size.

3.5.2.2 Linear Regression

In the linear regression problem, we find the basis U by minimizing the cost function. The cost function is defined as

$$f(U, \theta) = \|y - h(U\beta; \theta, b)\|^2. \tag{3.5.8}$$

The derivative of $f(U, \theta)$ with respect to U is given by

$$\frac{df}{dU} = -2[y - h(U\beta; \theta, b)]\theta\beta^{\mathrm{T}}. \tag{3.5.9}$$

where

$$h(U\beta;\theta,b) = \theta^T U\beta + b$$

The gradient on the Grassmannian is given by

$$\nabla f = -2[y - h(U\beta;\theta,b)](I - UU^{\mathrm{T}})\theta\beta^{\mathrm{T}}. \tag{3.5.10}$$

Let $r = [y - h(U\beta;\theta,b)](I - UU^{\mathrm{T}})\theta$. Thus,

$$\nabla f = -2r\beta^{\mathrm{T}}. \tag{3.5.11}$$

For the gradient is rank-one, we can write

$$\nabla f = \begin{bmatrix} -r/\|r\| & v_2 & \dots & v_d \end{bmatrix} \mathrm{diag}(\sigma) \begin{bmatrix} \beta/\|\beta\| & z_2 & \dots & z_d \end{bmatrix}^{\mathrm{T}}, \tag{3.5.12}$$

where

$$\sigma = -[y - h(U\beta;\theta,b)]\|r\|\|\beta\|. \tag{3.5.13}$$

$v_2,\dots,v_d$ are an orthonormal set orthogonal to r and $z_2,\dots,z_d$ are an orthonormal set orthogonal to β.

Update of U can be efficiently calculated as

$$\begin{aligned} U_{\text{new}} &= U + \frac{(\cos(\sigma\eta)-1)}{\|\beta\|^2}U\beta\beta^{\mathrm{T}} + \sin(\sigma\eta)\frac{r}{\|r\|}\frac{\beta^{\mathrm{T}}}{\|\beta\|} \\ &= U + ((\cos(\sigma\eta)-1)\frac{U\beta}{\|U\beta\|} + \sin(\sigma\eta)\frac{\gamma}{\|\gamma\|})\frac{\beta^{\mathrm{T}}}{\|\beta\|} \end{aligned} \tag{3.5.14}$$

where $\eta > 0$ is a step-size.

3.5.3 Parameters Update

Similar to what we do in ODR, at time t, when we get the feature vector x_t without its label, compute the predict label using (3.4.3). When both feature vector x_t and its label y_t are available, update the parameters of logistic model using (3.4.7) or update the parameters of linear regression model using (3.4.12).

3.5.4 Conclusion

In sum, the online sufficient dimensionality reduction algorithm can be described as follows.

Algorithm 2 Online Sufficient Dimensionality Reduction

```
Input: An $D \times d$ orthogonal matrix $U_0$. A sequence of vectors $x_t$ with its label $y_t$.
    A set of step-sizes $\eta_t$ for manifold updating. Slope vector and offset of logistic
    model $\theta_0$ and $b_0$. Step-size $\mu$ for stochastic gradient descent.
for t = 1, ..., T do
    Estimate weights: $\beta_t = \arg\min_v \|\Delta_{\Omega_t}(x_t - U_t v)\|^2$
    Predict vector label: $\hat{y}_t = h(\beta_t; \theta_{t-1}, b_{t-1})$
    Compute parameters $r$: $r = [y_t - \hat{y}_t](I - U_t U_t^{\mathrm{T}})\theta_{t-1}$
    Compute parameters $\sigma$: $\sigma = -[y_t - \hat{y}_t]\|r\|\|\beta_t\|$
    Update subspace: $U_{t+1} = U_t + ((\cos(\sigma\eta_t) - 1)\frac{U_t\beta_t}{\|U_t\beta_t\|} + \sin(\sigma\eta_t)\frac{\gamma}{\|\gamma\|})\frac{\beta_t^{\mathrm{T}}}{\|\beta_t\|}$
    Update weight vector: $\theta_{\mathrm{t}} = \theta_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t]\beta_t$.
    Update residual: $b_{\mathrm{t}} = b_{\mathrm{t-1}} + \mu[y_t - \hat{y}_t]$
end for
```

The difference of OSDR in solving the linear regression problem and the logistic regression problem lies in the regression function $h(\beta_t; \theta_{t-1}, b_{t-1})$. In linear regression problem,

$$h(\beta_t; \theta_{t-1}, b_{t-1}) = \theta_{t-1}^T \beta_t + b_{t-1}.$$

However, in logistic regression problem,

$$h(\beta_t; \theta_{t-1}, b_{t-1}) = \frac{1}{1 + e^{-\theta_{t-1}^{\mathrm{T}}\beta_t - b_{t-1}}}.$$

CHAPTER IV

NUMERICAL AND EXPERIMENTAL EXAMPLES

4.1 Numerical Examples

We will first use the following example to discuss the motivation and main advantages of OSDR over ODR. Let's consider two extreme cases, (a) $\theta = [5, 0]$. (b) $\theta = [0, 5]$. Set the parameters $a = 10$, $b = 0.1$, $c = 5$. Visualize the data in Figure 1.

In the unsupervised learning paradigm, assume the data is distributed as in Figure 1(a). Based on the knowledge of principle components analysis [18], in Figure 1(a), it's easy to see that the first principal component lies in the major axis of the ellipse. In this case, the first principal component is x_1 axis. Data vary a lot in the first principal component compared with the second principal component. Thus, when we reduce the dimension from $D = 2$ to $d = 1$, the unsupervised dimensionality reduction techniques will project the data onto the direction of the first principal component.

Figure 1(b) and in Figure 1(c) show the data distribution in the supervised learning paradigm. In case Figure 1(b), the unsupervised dimensionality reduction techniques are expected to work well since the variance of the data lies in the first principal component and it will be captured. Data is well represented and separated in the reduced dimension and we can learn the logistic model on the new reduced data.

However, when it comes to case in Figure 1(c), if the dimensionality reduction algorithms takes the unsupervised learning paradigm, it will still reduce the dimension and project the data onto the direction of the first principal component. But, since in in Figure 1(c), data vary a lot in the second principal component rather than the first principal component. The information of the label will be lost and data with different label would be mixed together in the reduced dimension. We can't learn the logistic model since the data in the reduced dimension isn't well separated. To solve this problem, the idea of sufficient dimensionality reduction should be included. Thus, based on the theory of sufficient dimensionality reduction, we know that the information of label won't be lost.

Let's go back to the comparison of ODR and OSDR. Since ODR takes the unsupervised learning paradigm, it will work well on in Figure 1(b) but encounter the problem when working on in Figure 1(c). In contrast to ODR, OSDR is along the same line as sufficient dimensionality reduction. It's expected to find the right direction, where data vary a lot and is well separated.

4.1.1 Static Subspace I

We first consider the problem of tracking a static manifold in $\mathbb{R}^2$, with $D = 2$ and $d = 1$. Points on the manifold $x_t \triangleq [x_{t,1}, x_{t,2}]^\mathrm{T}$ obey the following rules:

$$x_t = x^* + w_t, \tag{4.1.1}$$

w_t is a $D \times 1$ vector whose entries are i.i.d $N(0, \sigma^2)$. Entries of x_t^* are i.i.d $N(0, 1)$ and obey

$$\frac{(x_{t,1}^*)^2}{a^2} + \frac{(x_{t,2}^*)^2}{c^2} \leq 1. \tag{4.1.2}$$

In this way, the entries of x^* lie in the ellipse-shape space. The label y_t is related to x_t via logistic model

$$y_t \sim \text{Bernoulli}(h(x^*; \theta, b)), \tag{4.1.3}$$

where $\theta = [\theta_1, \theta_2]$ is a $D \times 1$ vector.

Generate $N = 6000$ data points using the method above. Divide the data set into two parts, the first 3000 data as the training set and the remaining as the testing set. For the training and testing procedure, we first use ODR and OSDR to predict label $\hat{y}_t$, then reveal the true label and use (x_t, y_t) to update the classifier. The performance metric is given by the average misclassification error on the test sample:

$$P_e = \frac{1}{3000} \sum_{t=3000}^{6000} \mathbb{I}\{\hat{y}_t \neq y_t\}. \tag{4.1.4}$$

In the following, we average P_e over 100 independent trials. We run the test with different value of θ. Figure 2 demonstrate the ordered misclassification error, as well as the ordered number of measurements calculated from the formulas, for different a/c rates, respectively. Note that the misclassification error of ODR is higher than OSDR. Although in some cases, ODR has almost the same performance with OSDR. In most situation, OSDR is better than ODR.

4.1.2 Static Subspace II

In the above simulation experiment, we demonstrate the advantage of sufficient dimensionality reduction and demonstrate the effectiveness of OSDR. In the following experiment, we will test the performance of OSDR on real-time high-dimensional data. Two-stage experiments will be done in this section. In the first stage, we embed a static low-dimensional space with dimension d into the high-dimensional space with dimension D. In the second stage, we formulate a dynamically evolving low-dimensional space with dimension d as the intrinsic space and embed it into the high-dimensional space with dimension D.

Suppose $\beta_t \in \mathbb{R}^d$ is the true feature vector which lies in the intrinsic low-dimensional subspace, assume $d = 2$ and $\beta_t \triangleq [\beta_{t,1}, \beta_{t,2}]^{\mathrm{T}}$. Entries of β are i.i.d $N(0,1)$ and obey

$$\frac{\beta_{t,1}^2}{a^2} + \frac{\beta_{t,2}^2}{c^2} \leq 1. \tag{4.1.5}$$

The observed feature vector $x_t \in \mathbb{R}^D$, where $d \ll D$, is the high-dimensional noisy realization of β_t.

$$x_t = U\beta_t + w_t, \tag{4.1.6}$$

w_t is a Gaussian noise with zero mean and variance σ^2. Here d denotes the intrinsic dimension and D denotes the ambient dimension. $U \in \mathbb{R}^{D\times d}$ is the basis of the manifold.

The label y_t is related to x_t via logistic model

$$y_t \sim \text{Bernoulli}(h(\beta_t; \theta, b)). \tag{4.1.7}$$

Generate N = 6000 data using the method above and divide the data set into two parts, the first 3000 data as the training set and the remaining 3000 data as the testing set. We will first use ODR and OSDR to predict label $\hat{y}_t$ and then reveal the true label and use (x_t, y_t) to update the classifier. The performance is also given by (4.1.4). Figure 3 demonstrates the performance of OSDR on real-time high-dimensional data with static intrinsic low dimensional subspace. The value of d has effect on the misclassification rate. Smaller d results in higher misclassification rate. Let D/d denote the data compression ratio, when the value of D is fixed, smaller d means higher data compression ratio, which always comes with loosing more information. It consists with our experimental results. By comparing the results among Figure 3(a) to Figure 3(e), we find that the shape of the intrinsic low-dimensional subspace has little effect on the performance of OSDR.

4.1.3 Rotating Subspace

Suppose $\beta_t^* \in \mathbb{R}^d$ is the true feature vector which lies in a dynamically evolving intrinsic low-dimensional subspace, assume $d = 2$ and $\beta^* = R_t\beta_t$. R_t is the rotation matrix, where

$$R = \begin{bmatrix} \cos(\alpha_t) & -\sin(\alpha_t) \\ \sin(\alpha_t) & \cos(\alpha_t) \end{bmatrix} \tag{4.1.8}$$

α_t is the rotation rate. $\beta_t \triangleq [\beta_{t,1}, \beta_{t,2}]^{\mathrm{T}}$. Entries of β are i.i.d $N(0, 1)$ and obey

$$\frac{\beta_{t,1}^2}{a^2} + \frac{\beta_{t,2}^2}{c^2} \le 1. \tag{4.1.9}$$

The observed feature vector $x_t \in \mathbb{R}^D$, where $d \ll D$, is the high-dimensional noisy realization of β_t^*.

$$x_t = U\beta_t^* + w_t \tag{4.1.10}$$

w_t is a Gaussian noise with zero mean and variance σ^2. Here d denotes the intrinsic dimension and D denotes the ambient dimension. $U \in \mathbb{R}^{D\times d}$ is the basis of the manifold.

The label y_t is related to x_t via logistic model

$$y_t \sim \text{Bernoulli}(h(\beta_t^*; \theta, b)). \tag{4.1.11}$$

Generate N = 6000 data using the method above and divide the data set into two parts, the first 3000 data as the training set and the remaining 3000 data as the testing set. The value of rotation rate α_t follows the rule:

$$\alpha_t = \begin{cases} 0 & \text{if } t \leq 500 \\ \frac{2\pi}{\tau} \cdot \frac{t-500}{6000-500} & \text{if } 500 < t \leq 6000 \end{cases} \tag{4.1.12}$$

τ is called rotation ratio. We will first use ODR and OSDR to predict label $\hat{y}_t$ and then reveal the true label and use (x_t, y_t) to update the classifier. The performance is also given by (4.1.4).

Figure 4, Figure 5 and Figure 6 shows the performance of OSDR in real-time high-dimensional data with rotating low-dimensional intrinsic subspace. Different shapes of intrinsic subspaces are examined. Different rotation ratios under same data compression ratio are also tested. OSDR demonstrates its effectiveness and robustness in tracking real-time high-dimensional data with rotating low-dimensional intrinsic subspace. Even though the intrinsic subspace rotates with a high rate, OSDR can still capture this evolution without loosing too much accuracy. The effect of rotation ratio τ on misclassification rate is demonstrated in Figure 7. When the value of rotation ratio is more than a threshold, it doesn't affect the performance of OSDR. When the value of rotation ratio is small, to improve the accuracy of OSDR, smaller data compression ratio may be considered.

4.1.4 Linear Regression

In the above experiments, we test the performance of OSDR in solving the problem with the assumption that the label is related to feature vectors via logistic model. In this section, we will demonstrate its performance on the problem whose assumption is that the label is related to feature vectors via linear regression model.

Let's consider the problem of tracking a static manifold in $\mathbb{R}^2$, with $D = 2$ and $d = 1$. Points on the manifold $x_t \triangleq [x_{t,1}, x_{t,2}]^{\mathrm{T}}$ obey the following rules:

$$x_t = x^* + w_t, \tag{4.1.13}$$

w_t is a $D \times 1$ vector whose entries are i.i.d $N(0, \sigma^2)$. Entries of x_t^* are i.i.d $N(0, 1)$ and obey

$$\frac{(x_{t,1}^*)^2}{a^2} + \frac{(x_{t,2}^*)^2}{c^2} \leq 1. \tag{4.1.14}$$

In this way, the entries of x^* lie in the ellipse-shape space. The label y_t is related to x_t via linear regression model

$$y_t = \theta^T x^* + b, \tag{4.1.15}$$

where $\theta = [\theta_1, \theta_2]$ is a $D \times 1$ vector.

Generate $N = 6000$ data points using the method above. Divide the data set into two parts, the first 3000 data as the training set and the remaining as the testing set. For the training and testing procedure, we first use ODR and OSDR to predict label $\hat{y}_t$, then reveal the true label and use (x_t, y_t) to update the classifier. The performance metric is given by root mean squared error on the test sample:

$$P_e = \sqrt{E((\hat{y} - y)^2)} = \sqrt{\frac{\sum_{t=3000}^{6000} (\hat{y}_t - y_t)^2}{3000}}. \tag{4.1.16}$$

In the following, we average P_e over 100 independent trials. We run the test with different value of θ. Figure 8 demonstrates the performance of OSDR in solving the problem that label is related to feature vector via linear regression model. OSDR has better performance than ODR. Figure 8(b) tells the case when the feature vectors are in ellipse-shape space and the weight of major axis is smaller than that of the minor axis. The larger the difference between the weight is, the worse the performance of ODR is. However, the performance is OSDR isn't affected by the difference, since it's a sufficient method. This is consist with the analysis in the previous experiment.

4.2 Real-data Experiments

4.2.1 Data Description

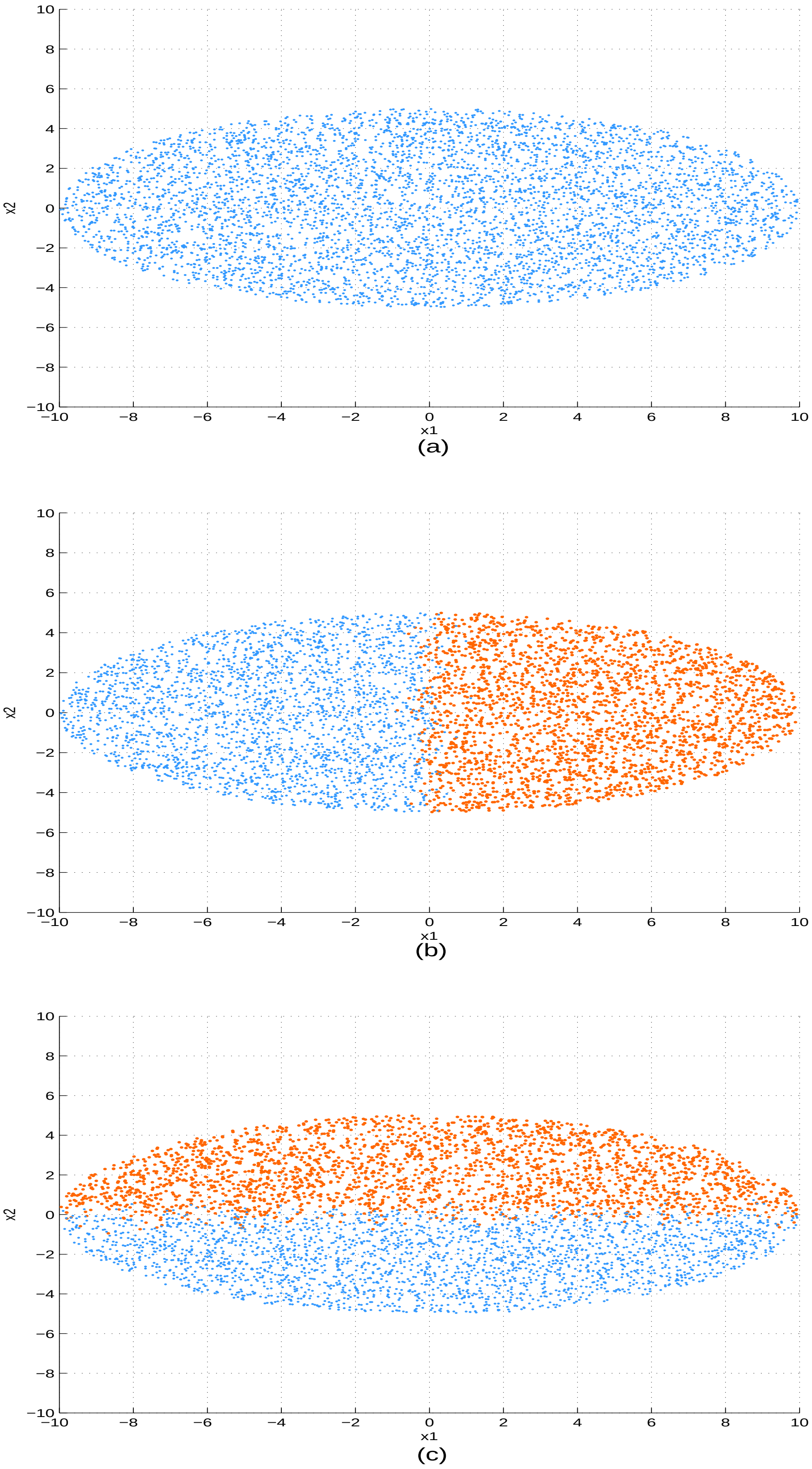
We use a well-known handwritten data sets, USPS digits recognition data sets, to test the performance of OSDR on online handwritten digits recognition.

The USPS handwritten digits dataset [13] is used from a project for recognizing handwritten digits on envelopes. The training set has 7291 observations, and the test set has 2007 observations, distributed as in Table 9. The digits were downscaled to 16×16 pixels and scaled without distortion.

To enable the online setting, we read one digit each time. We first process feature vector x_T of the digit and use ODR and OSDR to predict label $\hat{y}_t$. Then we reveal the true label and use (x_t, y_t) to update the classifier. In this experiment, we only consider a binary classification problem. For instance, whether a digit is 0 or not. The performance of OSDR and ODR is given by averaging their performance among all the digits.

4.2.2 Experimental Results

The performance of OSDR in USPS digits recognition data sets is shown in Table 10. Again, Figure 9 demonstrates that OSDR has good performance compared to ODR.



(a)

(b)

(c)

Figure 1: (a) Distribution of data without label. (b) Distribution of data when $\theta = [5, 0]$. (c) Distribution of data when $\theta = [0, 5]$

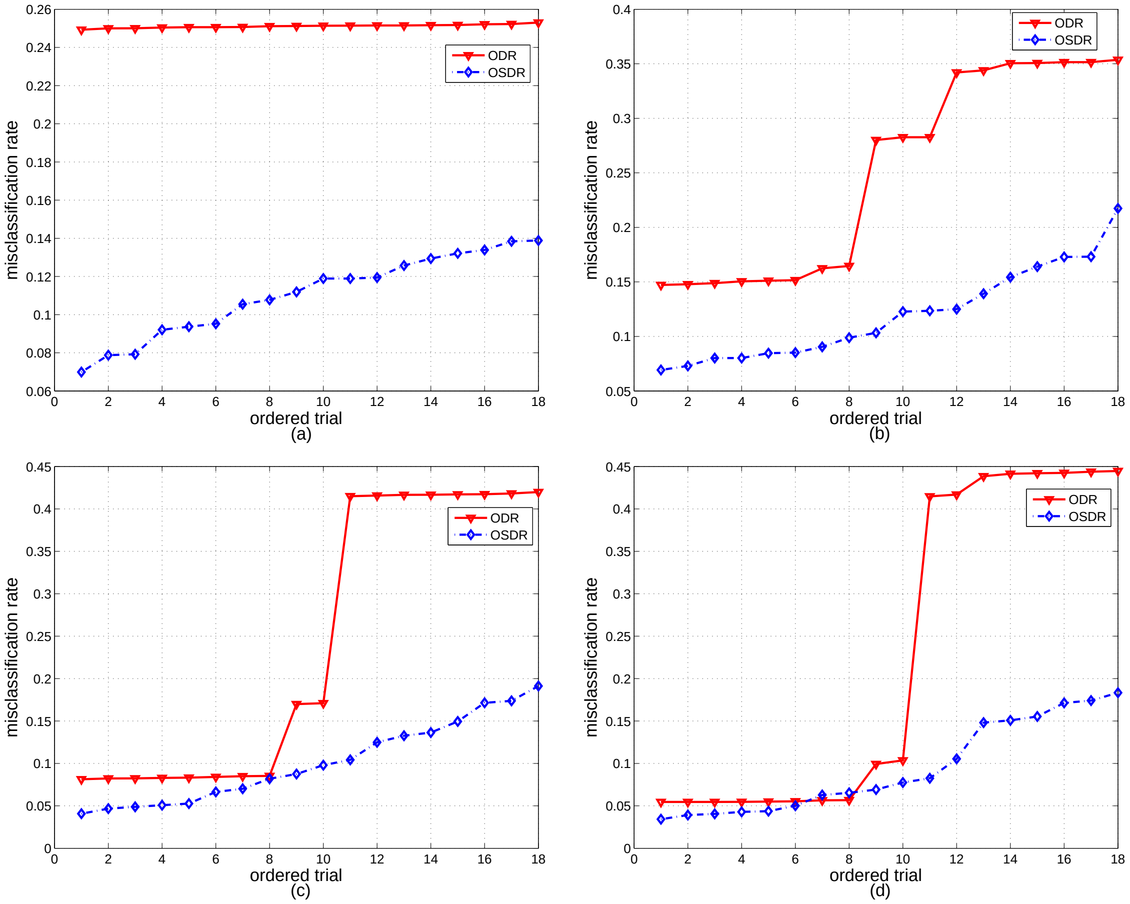


Figure 2: Static subspace I: (a) Misclassification error when $a/c = 1$. (b) Misclassification error when $a/c = 3$. (c) Misclassification error when $a/c = 6$. (d) Misclassification error when $a/c = 10$.

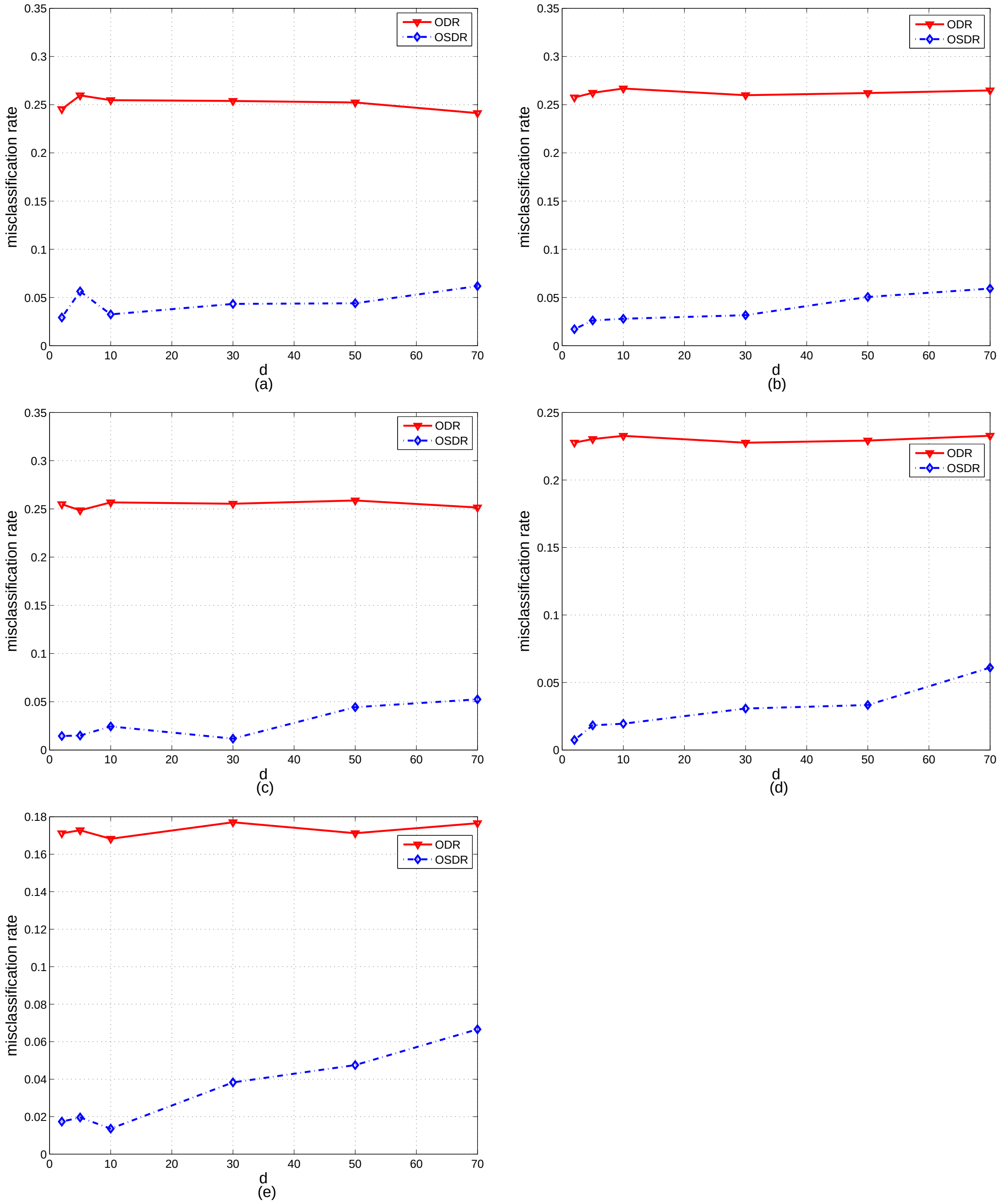


Figure 3: Static subspace II: (a) d v.s. misclassification error when $a/c = 1$. (b) d v.s. misclassification error when $a/c = 3$. (c) d v.s. misclassification error when $a/c = 5$. (d) d v.s. misclassification error when $a/c = 7$. (e) d v.s. misclassification error when $a/c = 10$.

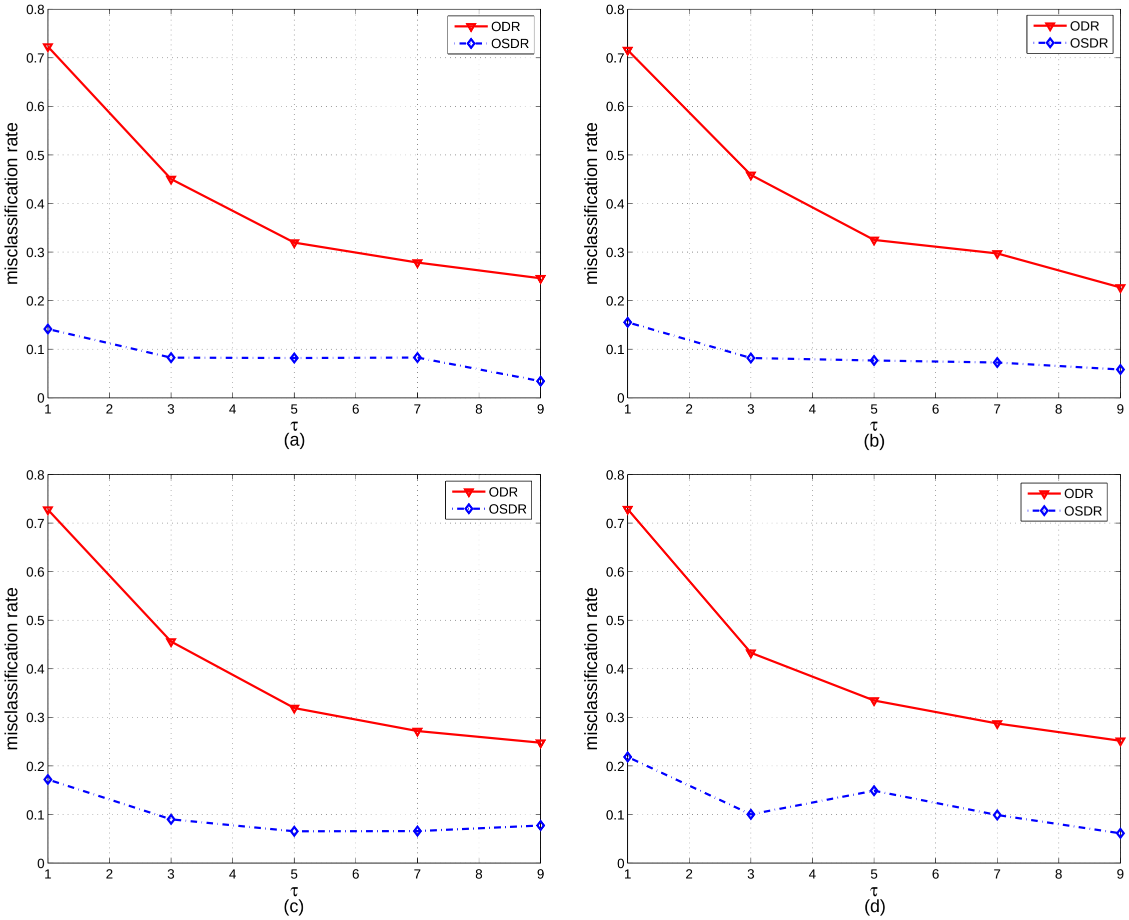


Figure 4: Rotating subspace: $a/c = 10$ (a) rotation ratio τ v.s. misclassification rate when $d = 30$ (b) rotation ratio τ v.s. misclassification rate when $d = 10$ (c) rotation ratio τ v.s. misclassification rate when $d = 5$ (d) rotation ratio τ v.s. misclassification rate when $d = 2$

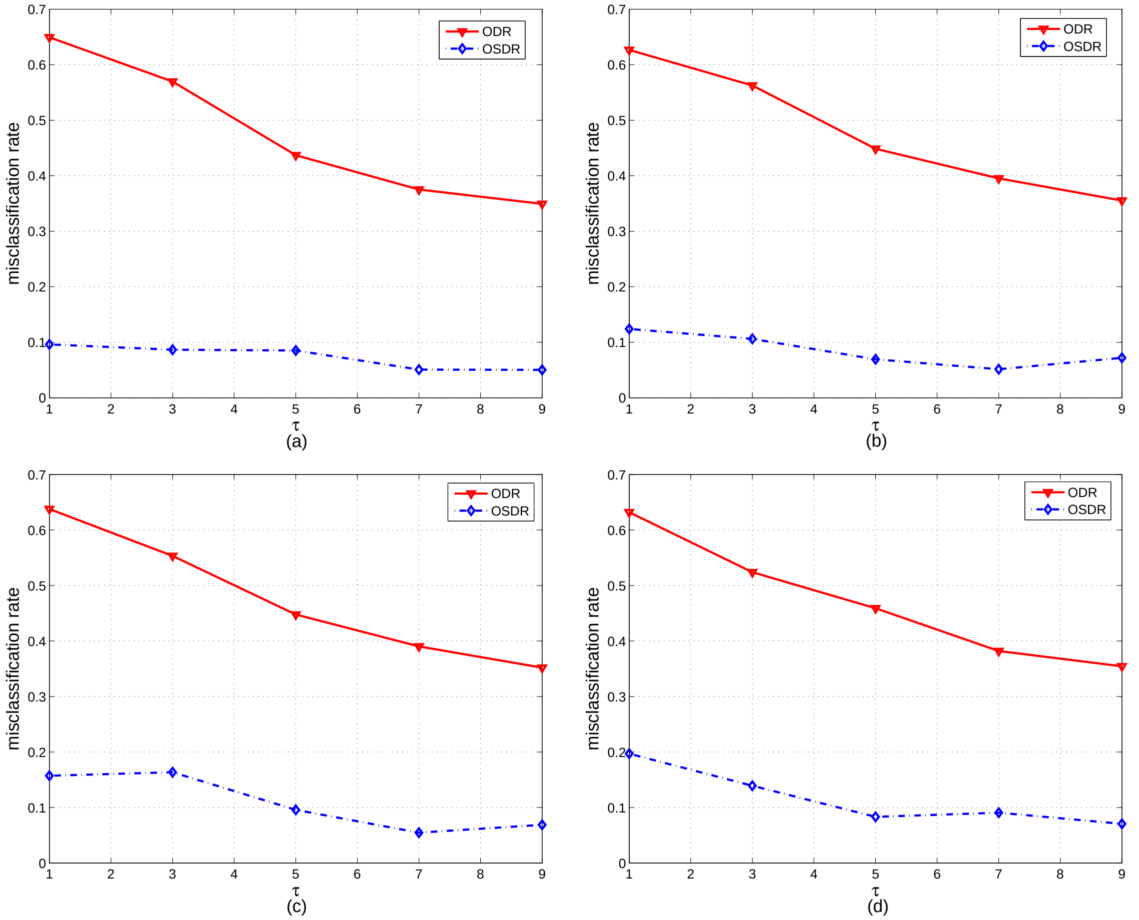


Figure 5: Rotating subspace: $a/c = 6$ (a) rotation ratio τ v.s. misclassification rate when $d = 30$ (b) rotation ratio τ v.s. misclassification rate when $d = 10$ (c) rotation ratio τ v.s. misclassification rate when $d = 5$ (d) rotation ratio τ v.s. misclassification rate when $d = 2$

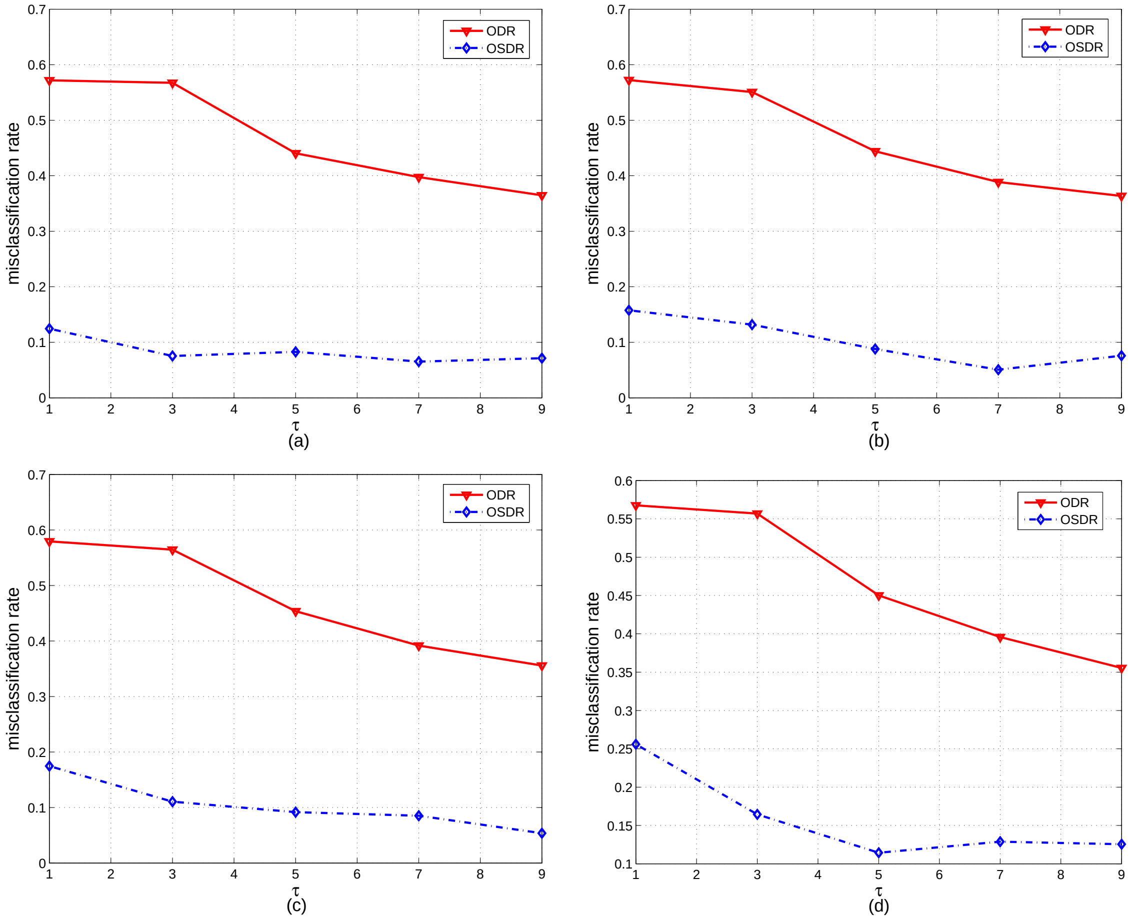
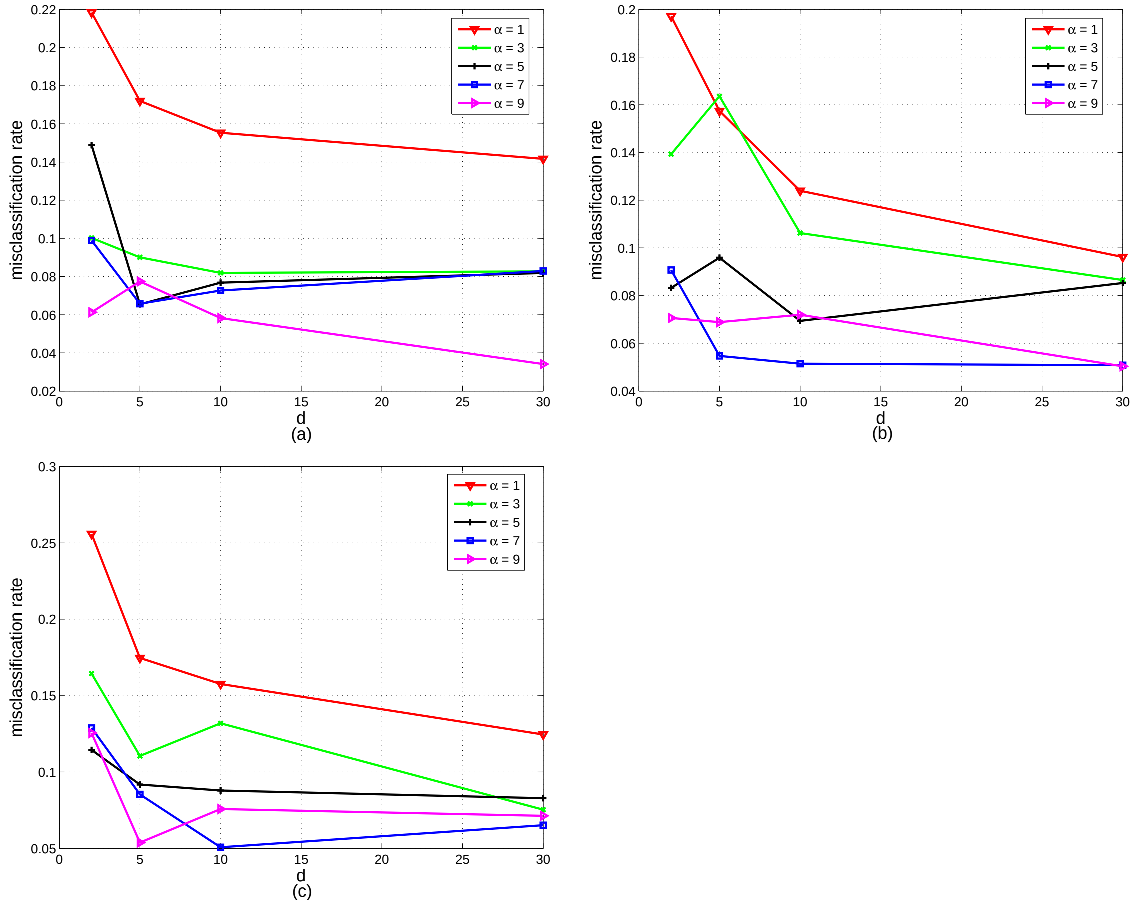


Figure 6: Rotating subspace: $a/c = 3$ (a) rotation ratio τ v.s. misclassification rate when $d = 30$ (b) rotation ratio τ v.s. misclassification rate when $d = 10$ (c) rotation ratio τ v.s. misclassification rate when $d = 5$ (d) rotation ratio τ v.s. misclassification rate when $d = 2$

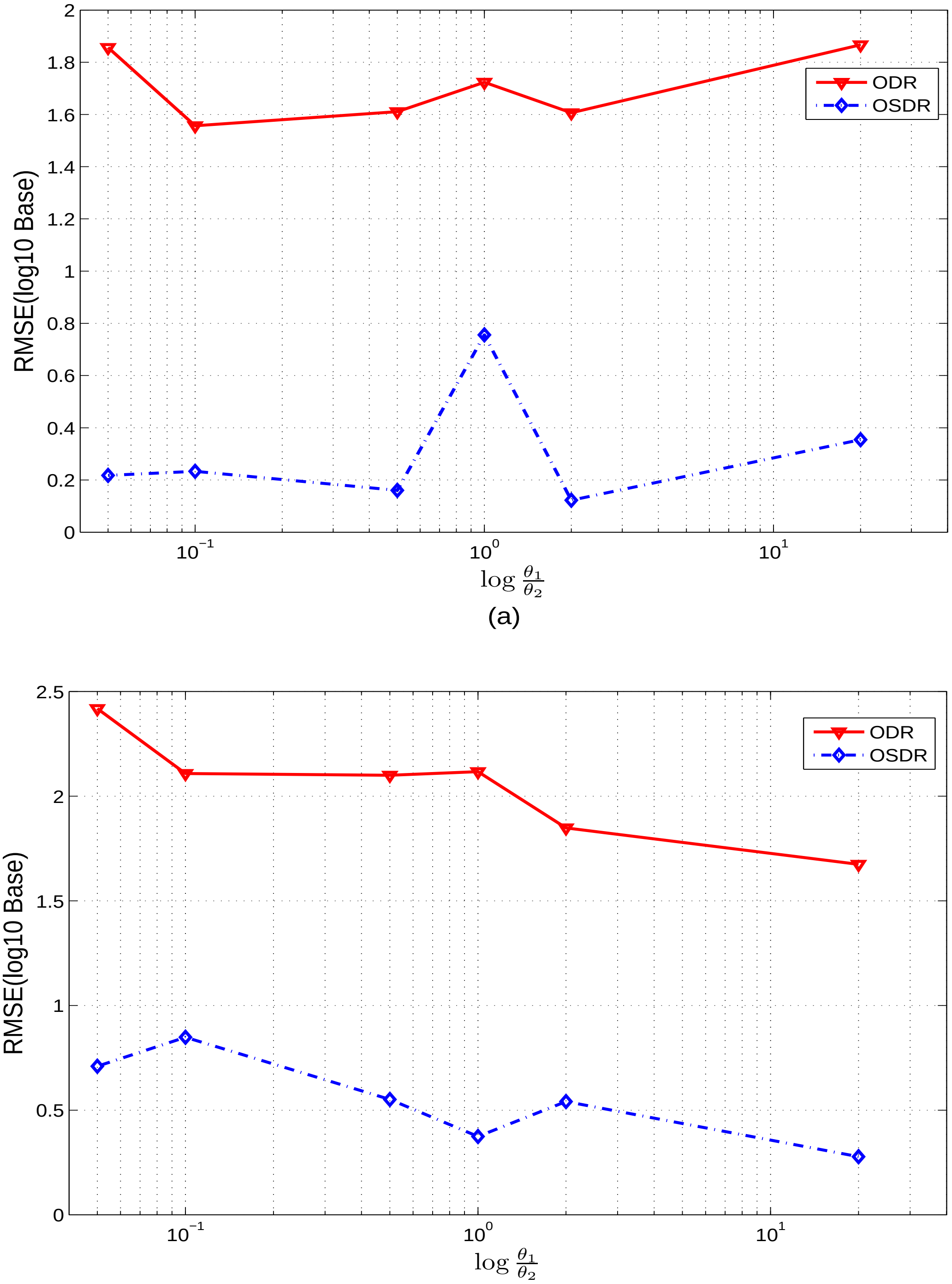


(a)

(b)

(c)

Figure 7: Rotating subspace: (a) d v.s. misclassification rate when $a/c = 10$ (b) d v.s. misclassification rate when $a/c = 6$ (c) d v.s. misclassification rate when $a/c = 3$



(a)

(b)

Figure 8: Linear regression: (a) $log(\theta_1/\theta_2)$ v.s. log(RMSE) when $a/c = 1$ (b) $log(\theta_1/\theta_2)$ v.s. log(RMSE) when $a/c = 2$

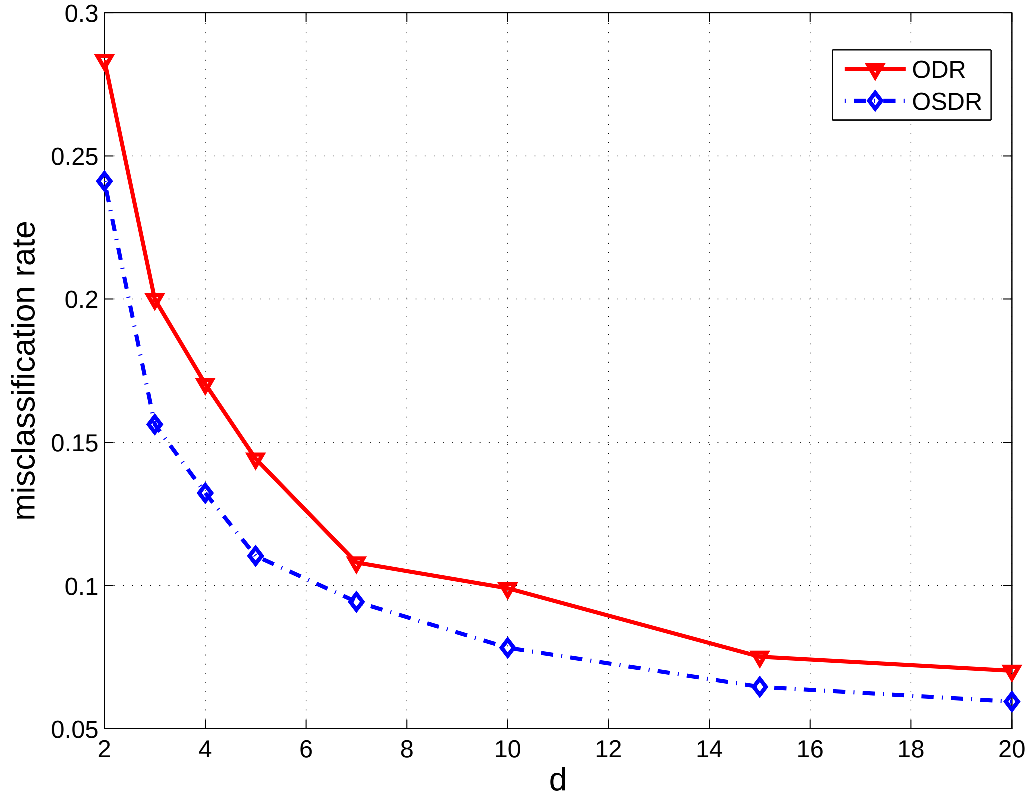


Figure 9: Real-data: USPS digits recognition. d v.s. misclassification rate

CHAPTER V

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

In this thesis, we propose a novel dimensionality reduction algorithm, online sufficient dimensionality reduction algorithm(OSDR) for real-time high-dimensional sequential data. OSDR consists of three steps: In the first step, given feature vector x_t, compute its projection onto the current estimated manifold. In the second step, update the estimated manifold using the new proposed tracking method. In the final step, update the regression model using stochastic gradient descent. It has the following characteristics. (a) As an online algorithm, OSDR doesn't need to process all the observations in order to do dimensionality reduction, which enable its qualification for real-time high-dimensional data processing and analyzing. (b) Inspired by the idea of sufficient dimensionality reduction, OSDR avoids losing information and improves performance by incorporating the label of observation into consideration. (c) OSDR can be applied to tracking the evolution of subspace (d) OSDR has low complexity and computational efficiency. (e) OSDR is able to handle noise and missing elements in feature vectors.

Numerical experiments demonstrate the accuracy and effectiveness of OSDR. Also the real-data experiments prove that OSDR is not only feasible in simulation but also can be applied in solving real-world problem.

5.2 Future Work

In the future, we would like to investigate how to adapt step-size in OSDR automatically to varying data. In the above experiments, we have seen that there are substantial performance gains when the step-size is optimized. It might be possible to automatically tune the step size based on the current error residuals by incorporating Least Squares Estimation [2].

Another concern of future work lies in the way of setting up an initial basis U. Right now, a lot of random restarts might be needed for finding a good local optimum, which will increase the running time and reduce the reliability of OSDR.

In OSDR, the label is assumed to related to feature vector via logistic model and linear regression model. In the future work, more general assumption could be made. We may consider solving the problem with the assumption of generalized linear model.

CHAPTER VI

EXTRA NOTES

Table 1: Misclassification Rate of ODR v.s OSDR when $a/c = 1$.

Methods $\frac{\theta_1}{\theta_2}$	ODS	OSDR
20	0.25175(0.00005)	0.11890(0.02072)
15	0.24927(0.00003)	0.11192(0.02220)
10	0.25072(0.00004)	0.07879(0.01400)
5	0.25041(0.00004)	0.09370(0.01822)
1	0.25041(0.00004)	0.09370(0.01822)
$\frac{1}{5}$	0.25224(0.00004)	0.07923(0.01241)
$\frac{1}{10}$	0.25165(0.00004)	0.10774(0.02708)
$\frac{1}{15}$	0.25148(0.00006)	0.06992(0.01008)
$\frac{1}{20}$	0.25140(0.00004)	0.11893(0.02849)
$-\frac{1}{20}$	0.24998(0.00004)	0.13385(0.02644)
$-\frac{1}{15}$	0.25002(0.00005)	0.09207(0.01794)
$-\frac{1}{10}$	0.25060(0.00004)	0.12937(0.02452)
$-\frac{1}{5}$	0.25214(0.00003)	0.10545(0.02243)
-1	0.25111(0.00005)	0.13884(0.02493)
-5	0.25060(0.00006)	0.13842(0.03027)
-10	0.25135(0.00003)	0.11944(0.02763)
-15	0.25304(0.00003)	0.12571(0.02643)
-20	0.25123(0.00003)	0.13213(0.02695)

Table 2: Misclassification Rate of ODR v.s OSDR when $a/c = 3$.

Methods $\frac{\theta_1}{\theta_2}$	ODS	OSDR
20	0.14719(0.00002)	0.12351(0.2105)
15	0.15109(0.00005)	0.08513(0.01171)
10	0.15158(0.00003)	0.09046(0.01150)
5	0.16459(0.00003)	0.06929(0.00728)
1	0.28269(0.00012)	0.08023(0.01307)
$\frac{1}{5}$	0.34209(0.00006)	0.13919(0.02792)
$\frac{1}{10}$	0.28269(0.00012)	0.08023(0.01307)
$\frac{1}{15}$	0.35048(0.00006)	0.15434(0.03664)
$\frac{1}{20}$	0.35149(0.00004)	0.12281(0.02349)
$-\frac{1}{20}$	0.25153(0.00004)	0.17291(0.03693)
$-\frac{1}{15}$	0.35070(0.00007)	0.17314(0.03624)
$-\frac{1}{10}$	0.34392(0.00004)	0.21737(0.03811)
$-\frac{1}{5}$	0.2807(0.00008)	0.12503(0.02234)
-1	0.35367(0.00004)	0.10328(0.02491)
-5	0.16247(0.00003)	0.08466(0.00853)
-10	0.15055(0.00003)	0.16401(0.02382)
-15	0.14891(0.00003)	0.07301(0.00454)
-20	0.14784(0.00003)	0.09883(0.01479)

Table 3: Misclassification Rate of ODR v.s OSDR when $a/c = 6$.

Methods $\frac{\theta_1}{\theta_2}$	ODS	OSDR
20	0.08237(0.00002)	0.05075(0.00479)
15	0.08224(0.00002)	0.06646(0.00563)
10	0.08407(0.00003)	0.04076(0.00200)
5	0.08495(0.00002)	0.04675(0.00305)
1	0.16997(0.00015)	0.07016(0.0932)
$\frac{1}{5}$	0.41659(0.00007)	0.13641(0.01737)
$\frac{1}{10}$	0.41675(0.00003)	0.13269(0.02316)
$\frac{1}{15}$	0.41714(0.00004)	0.08747(0.01390)
$\frac{1}{20}$	0.41734(0.00004)	0.14947(0.03030)
$-\frac{1}{20}$	0.41575(0.00004)	0.17381(0.03298)
$-\frac{1}{15}$	0.41818(0.00005)	0.17145(0.03363)
$-\frac{1}{10}$	0.41500(0.00009)	0.19120(0.03361)
$-\frac{1}{5}$	0.41990(0.00068)	0.12483(0.01689)
-1	0.17095(0.00009)	0.10419(0.00936)
-5	0.08527(0.00001)	0.05271(0.00525)
-10	0.08319(0.00002)	0.9792(0.02008)
-15	0.08293(0.00003)	0.04881(0.00178)
-20	0.08135(0.00002)	0.08199(0.00772)

Table 4: Misclassification Rate of ODR v.s OSDR when $a/c = 10$.

Methods $\frac{\theta_1}{\theta_2}$	ODS	OSDR
20	0.05466(0.00001)	0.04376(0.00206)
15	0.05463(0.00002)	0.04048(0.00113)
10	0.05536(0.00001)	0.03917(0.00185)
5	0.05670(0.00001)	0.04284(0.00246)
1	0.09930(0.00014)	0.03425(0.0150)
$\frac{1}{5}$	0.41693(0.00007)	0.06918(0.00412)
$\frac{1}{10}$	0.43857(0.00007)	0.10541(0.01273)
$\frac{1}{15}$	0.44206(0.00005)	0.15076(0.02254)
$\frac{1}{20}$	0.44264(0.00005)	0.15539(0.02690)
$-\frac{1}{20}$	0.44390(0.00005)	0.14807(0.02543)
$-\frac{1}{15}$	0.44144(0.00007)	0.18340(0.03024)
$-\frac{1}{10}$	0.44478(0.00043)	0.17131(0.01922)
$-\frac{1}{5}$	0.41482(0.00007)	0.17405(0.01752)
-1	0.10357(0.00001)	0.08252(0.00780)
-5	0.05647(0.00001)	0.07746(0.01183)
-10	0.05462(0.00001)	0.06532(0.01047)
-15	0.05457(0.00001)	0.05015(0.00714)
-20	0.05496(0.00002)	0.06278(0.01127)

Table 5: Misclassification Rate of ODR v.s. OSDR in Static subpsace II

	d	ODR	OSDR
$\frac{a}{b} = 1$	70	0.24541(0.00031)	0.02927(0.01)
	50	0.25497(0.0002)	0.05635(0.01784)
	30	0.25469(0.00457)	0.3249(0.0101)
	10	0.25385(0.00333)	0.04339(0.01213)
	5	0.25445(0.00522)	0.04399(0.00904)
	2	0.24131(0.00026)	0.06183(0.00618)
$\frac{a}{b} = 3$	70	0.25752(0.00025)	0.01718(0.00182)
	50	0.2623(0.00015)	0.02624(0.00509)
	30	0.26575(0.00013)	0.02793(0.00332)
	10	0.25984(0.00019)	0.03165(0.00288)
	5	0.26203(0.00015)	0.05055(0.00822)
	2	0.26476(0.00024)	0.05925(0.00545)
$\frac{a}{b} = 5$	70	0.25486(0.00035)	0.01445(0.00073)
	50	0.24858(0.00016)	0.01499(0.00071)
	30	0.25647(0.0003)	0.02439(0.003)
	10	0.25533(0.00025)	0.01173(0.00011)
	5	0.25875(0.00026)	0.04437(0.00786)
	2	0.25144(0.00023)	0.05251(0.00448)
$\frac{a}{b} = 7$	70	0.22765(0.0004)	0.00737(0.00003)
	50	0.2303(0.00027)	0.01825(0.00062)
	30	0.2327(0.00033)	0.01941(0.00104)
	10	0.22761(0.00046)	0.03069(0.00173)
	5	0.22921(0.00033)	0.03331(0.0018)
	2	0.23275(0.00043)	0.06095(0.00414)
$\frac{a}{b} = 10$	70	0.17115(0.00112)	0.01734(0.00051)
	50	0.17276(0.00111)	0.01957(0.00273)
	30	0.1682(0.00061)	0.01361(0.00021)
	10	0.17704(0.000144)	0.03829(0.00188)
	5	0.17119(0.00105)	0.04751(0.00314)
	2	0.17651(0.0007)	0.06663(0.0026)

Table 6: Misclassification Rate of ODR v.s. OSDR when $a/c = 10$ in rotating subspace

Low Dimension	Rate	ODR	OSDR
$d = 30$	1	0.72295(0.00064)	0.14161(0.00338)
	3	0.45043(0.00593)	0.08272(0.00369)
	5	0.31948(0.00425)	0.08185(0.00709)
	7	0.27835(0.0043)	0.08287(0.00465)
	9	0.24609(0.00158)	0.03412(0.00057)
$d = 10$	1	0.7158(0.00087)	0.15532(0.00445)
	3	0.45891(0.0069)	0.08191(0.00449)
	5	0.325(0.00325)	0.07676(0.00582)
	7	0.29716(0.00193)	0.07264(0.00468)
	9	0.22717(0.0028)	0.05821(0.00401)
$d = 5$	1	0.72765(0.00046)	0.17196(0.00237)
	3	0.45604(0.00384)	0.09004(0.00597)
	5	0.31909(0.0042)	0.0654(0.00301)
	7	0.27175(0.00217)	0.06575(0.00529)
	9	0.24768(0.00226)	0.0774(0.0057)
$d = 2$	1	0.72845(0.00063)	0.21832(0.00529)
	3	0.43291(0.00877)	0.1002(0.0082)
	5	0.33468(0.00685)	0.14884(0.01861)
	7	0.28727(0.00348)	0.09889(0.01129)
	9	0.25168(0.00223)	0.06127(0.00276)

Table 7: Misclassification Rate of ODR v.s. OSDR when $a/c = 6$ in rotating subspace

Low Dimension	Rate	ODR	OSDR
$d = 30$	1	0.64932(0.00256)	0.09619(0.00253)
	3	0.56964(0.00113)	0.08652(0.00692)
	5	0.43693(0.00131)	0.08528(0.007)
	7	0.37511(0.00153)	0.05081(0.00443)
	9	0.34928(0.00062)	0.05036(0.00471)
$d = 10$	1	0.62667(0.00149)	0.12393(0.00356)
	3	0.56253(0.00172)	0.1062(0.00492)
	5	0.44853(0.00091)	0.06941(0.00781)
	7	0.39529(0.00106)	0.05147(0.00552)
	9	0.35529(0.00059)	0.072(0.00539)
$d = 5$	1	0.63783(0.00259)	0.15737(0.00264)
	3	0.55328(0.00236)	0.16356(0.01231)
	5	0.44769(0.00073)	0.09588(0.00958)
	7	0.39021(0.00032)	0.05473(0.00188)
	9	0.35207(0.00072)	0.06887(0.01006)
$d = 2$	1	0.63207(0.0027)	0.19708(0.00354)
	3	0.52392(0.00463)	0.13927(0.01334)
	5	0.45896(0.00072)	0.8324(0.00354)
	7	0.38201(0.00071)	0.09072(0.09072)
	9	0.3544(0.00039)	0.0706(0.00512)

Table 8: Misclassification Rate of ODR v.s. OSDR when $a/c = 3$ in rotating subspace

Low Dimension	Rate	ODR	OSDR
$d = 30$	1	0.57187(0.00111)	0.12452(0.00944)
	3	0.5674(0.00051)	0.07531(0.01172)
	5	0.44044(0.00032)	0.08279(0.01087)
	7	0.39751(0.00032)	0.06512(0.00562)
	9	0.36469(0.00015)	0.07125(0.0127)
$d = 10$	1	0.57228(0.0013)	0.1576(0.00904)
	3	0.55076(0.00118)	0.13188(0.01316)
	5	0.44408(0.00029)	0.08784(0.0125)
	7	0.38857(0.00032)	0.05069(0.00351)
	9	0.36357(0.00037)	0.07572(0.01228)
$d = 5$	1	0.57951(0.00093)	0.17467(0.00631)
	3	0.56457(0.00061)	0.11052(0.00989)
	5	0.45368(0.00017)	0.09173(0.00757)
	7	0.39179(0.00015)	0.08529(0.01082)
	9	0.35573(0.00049)	0.05375(0.00751)
$d = 2$	1	0.56763(0.00094)	0.25587(0.00689)
	3	0.55684(0.00068)	0.16439(0.01494)
	5	0.45016(0.00035)	0.1144(0.01074)
	7	0.39577(0.00042)	0.12876(0.01598)
	9	0.35548(0.00021)	0.12548(0.01055)

Table 9: Distribution of digits in USPS dataset.

	0	1	2	3	4	5	6	7	8	9	Total
Train	1194	1005	731	658	652	556	664	645	542	644	7291
Test	359	264	198	166	200	160	170	147	166	177	2007

Table 10: d v.s. misclassification rate in USPS digits recognition.

	2	3	4	5	7	10	15	20
ODR	0.2834	0.2000	0.1704	0.1443	0.1080	0.0990	0.0751	0.0702
OSDR	0.2411	0.1563	0.1323	0.1103	0.0943	0.0783	0.0646	0.0594

REFERENCES

[1] Aggarwal, C. C., "A framework for local supervised dimensionality reduction of high dimensional data," in *SDM'06*, pp. –1–1, 2006.

[2] Balzano, L., Nowak, R., and Recht, B., "Online identification and tracking of subspaces from highly incomplete information," *CoRR*, pp. –1–1, 2010.

[3] Brendel, W. and Todorovic, S., "Activities as time series of human postures," in *ECCV (2)'10*, pp. 721–734, 2010.

[4] Carter, K. M., Raich, R., and Hero, A. O., "An information geometric approach to supervised dimensionality reduction," in *ICASSP'09*, pp. 1829–1832, 2009.

[5] Cook, R. D., "Save: a method for dimension reduction and graphics in regression," *Theory and Methods*, pp. pp. 2109–2121, 2000.

[6] Cook, R. D. and Ni, L., "Sufficient dimension reduction via inverse regression: A minimum discrepancy approach," *Journal of the American Statistical Association*, vol. Vol.100, 2005.

[7] Costeira, J. P. and Kanade, T., "A multibody factorization method for independently moving objects," *International Journal of Computer Vision*, p-p. 159–179, 1998.

[8] Cunningham, P., "Dimension reduction," 2007.

[9] Edelman, A., Arias, T. A., and Smith, S. T., "The geometry of algorithms with orthogonality constrints.," *SIAM JOURNAL on Matrix Analysis and Applications 20*, vol. Vol.20, pp. pp. 303–353, 1998.

[10] Fan, J., Ke, T., Liu, H., and Xia, L., "Quadro: A supervised dimension reduction method via rayleigh quotient optimization," 2013.

[11] Harris, J., *Grassmannians and Related Varieties.* Springer, 1992.

[12] Henry, S. and W., W. J., *Probability, Random Processes, and Estimation Theory for Engineers.* Prentice-Hall, Inc., 1986.

[13] Hull, J. J., "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 550–554, 1994.

[14] IBM, "IBM: The Four V's of Big Data ." `http://www.ibmbigdatahub.com/infographic/four-vs-big-data`, 2013. [Online].

[15] IBM, "IBM What is big data? — Bringing big data to the enterprise." `http://www.ibm.com/big-data/us/en/`, 2013. [Online].

[16] IBM, "Big Data at the Speed of Business - What is big data?." `http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html`, 2014. [Online].

[17] IBM, "IBM What is big data? — Bringing big data to the enterprise." `http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html`, 2014. [Online].

[18] I.T., J., *Principal Component Analysis.* NY: Springer, 2nd ed ed., 2002.

[19] Kholevo, A., "Sufficient statistic," 2001.

[20] Kohavi, R. and John, G. H., "Wrappers for feature subset selection," *Artif. Intell.*, pp. pp 273–324, 1997.

[21] Kuo, C.-F., Kuo, T.-W., and Chang, C., "Real-time digital signal processing of phased array radars," *IEEE Trans. Parallel Distrib. Syst.*, pp. 433–446, 2003.

[22] Li, K.-C., "Sliced inverse regression for dimension reduction," *Journal of the American Statistical Association*, pp. pp. 316 – 327, 1991.

[23] Li, K.-C., "On principal hessian directions for data visualization and dimension reduction: Another application of steins lemma," *Journal of the American Statistical Association*, pp. pp. 1025 – 1034, 1992.

[24] Lütkebohle, I., "BWorld Robot Control Software." `http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/`, 2008. [Online; accessed 19-July-2008].

[25] Nilsson, J., Sha, F., and Jordan, M. I., "Regression on manifolds using kernel dimension reduction," in *ICML'07*, pp. 697–704, 2007.

[26] R.Bellman, *Adaptive Control Processes: A Guided Tour.* Princeton University Press, 1961.

[27] Snchez-Maroo, N., Alonso-Betanzos, A., and Tombilla-Sanromn, M., *Filter Methods for Feature Selection - A Comparative Review*, pp. pp 178 – 187. 2007.

[28] Sorzano, C. O. S., Vargas, J., and Pascual-Montano, A. D., "A survey of dimensionality reduction techniques," *CoRR*, pp. –1–1, 2014.

[29] van der Maaten, L., Postma, E., and van den Herik, H., "Dimensionality reduction: A comparative review," *Technical Report TiCC TR 2009-005*, 2009.

[30] Wang, L., Wang, L., Wen, M., Zhuo, Q., and Wang, W., "Background subtraction using incremental subspace learning," in *ICIP (5)'07*, pp. 45–48, 2007.

[31] Xie, Y., Huang, J., and Willett, R., “Multiscale online tracking of manifolds,” pp. 620–623, 2012.

[32] Xie, Y. and Willett, R., “Online logistic regression on manifolds,” in *ICASSP’13*, pp. 3367–3371, 2013.

[33] Yu, S., Yu, K., Tresp, V., Kriegel, H.-P., and Wu, M., “Supervised probabilistic principal component analysis,” in *KDD’06*, pp. 464–473, 2006.

[34] Zhang, Y. and Xu, G., “Singular value decomposition,” in *Encyclopedia of Database Systems*, pp. 2657–2658, 2009.