Exploiting Non-sequence Data in Dynamic Model Learning

Tzu-Kuo Huang TZUKUOH@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Abstract

Virtually all methods of learning dynamic models from data start from the same basic assumption: that the learning algorithm will be provided with a single or multiple sequences of data generated from the dynamic model. However, in quite a few modern time series modeling tasks, the collection of reliable time series turns out to be a major challenge, either due to the slow progression of the dynamic process of interest, or inaccessibility of repetitive measurements of the same dynamic process over time. In those situations, we observe that it is often easier to collect a large amount of non-sequence samples, or snapshots of the dynamic process of interest.

This thesis aims to exploit such non-sequence samples in dynamic model learning. We first consider the case where the only data available are snapshots taken from multiple instantiations of a dynamic process at unknown times, and the dynamic process falls in the class of fully observable, discrete-time, first-order linear or non-linear dynamic models. We pointed out several issues in model identifiability when learning from non-sequence data, and developed several learning algorithms based on optimizing approximate posterior distributions. When applied to a number of synthetic data sets and a gene expression time series data set, the proposed algorithms are able to learn moderately to highly accurate dynamic models, but at times suffer severely from the model ambiguity inherent in this setting.

We thus consider a second scenario: a small amount of sequence data and a lot of non-sequence data are available. The class of dynamic models we have studied so far are first-order discrete-time stable vector auto-regressive (VAR) models, and the non-sequence data are assumed to be independent samples drawn from the stationary distribution of the VAR model. We developed learning algorithms that minimize a novel penalized least square objective, which incorporates non-sequence data in the penalization term through violation of the Lyapunov equation of the stable VAR model. Experimental results on synthetic data and a video sequence demonstrate that the proposed method improves significantly over standard algorithms when there is only little sequence data.

There are some interesting topics we plan to explore. A major direction is to investigate ways of exploiting non-sequence data to learn state space models, which are a common tool for studying high-dimensional data. We also plan to work on real-world problems that will truly benefit from the proposed methods. Possible targets include learning dynamic models for cell cycles from cell images, where the destructive nature of the measuring procedure makes it difficult and costly to obtain successive observations, and learning dynamic models of certain biological processes, such as disease progression, from gene expressions, where static data are much more abundant than sequence data.

1. Introduction

Learning dynamic models from data is the traditional topic of system identification (Ljung, 1999) in control theory and many algorithms have been proposed. In the machine learning literature, the learning of temporal graphical models, such as dynamic Bayesian networks (Ghahramani, 1998a; Murphy, 2002), and the learning of various types of Markov models (e.g., Rabiner (1989); Ghahramani (1998b); Beal et al. (2002); Abbeel and Ng (2005); Hsu et al. (2009); Song et al. (2010)) have been extensively studied. Virtually all methods of learning dynamic models from data start from the same basic assumption: that the learning algorithm will be provided with a single or multiple sequences of data generated from the dynamic model. However, in quite a few modern dynamic modelling tasks, a major difficulty turns out to be the collection of reliable time series data. In some of these tasks, such as learning dynamic models of galaxy or star evolution, the dynamics of the processes of interest are far too slow for researchers to collect successive data points showing any meaningful changes. At more modest time scales, the same problem arises in the understanding of slow-evolving human diseases such as Alzheimer's or Parkinson's, which may progress over a decade or more. In other situations, the dynamic process of interest may not be able to undergo repetitive measurements, so researchers have to measure multiple instances of the same process while maintaining synchronization among these instances. One such example is gene expression time series. In their study, Tu et al. (2005) measured expression profiles of yeast genes along consecutive metabolic cycles. Due to the destructive nature of the measurement technique, they collected expression data from multiple yeast cells. In order to obtain reliable time series data, they spent a lot of effort developing a stable environment to synchronize the cells during the metabolic cycles. Yet, they point out in their discussion that such a synchronization scheme may not work for other species, e.g., certain bacteria and fungi, as effectively as for yeast.

While obtaining reliable time series can be difficult, it is often easier to collect non-sequence samples, or snapshots of the dynamic process of interest[1]. For example, the Sloan Digital Sky Survey (SDSS)[2] have collected images of millions of celestial objects, each of which may be in a different phase of its life cycle. In medical sciences, a scientist studying Alzheimer's or Parkinson's can collect samples from his or her current pool of patients, each of whom may be in a different stage of the disease. Or in gene expression analysis, current technology already enables large-scale collection of static gene expression data.

We propose in this thesis ways to exploit such non-sequence samples in dynamic model learning. After giving a brief survey of related work in Section 2, we first consider in Section 3 the case where the only data available are snapshots taken from multiple instantiations of a dynamic process at unknown times, and the dynamic process falls in the class of fully observable, discrete-time, first-order linear or non-linear dynamic models. We pointed out several issues in model identifiability when learning from non-sequence data, and developed several learning algorithms based on optimizing approximate posterior distributions. These learning algorithms solve non-convex optimization problems, and thus require good initializations. This leads us to study a different but related problem, that of reconstructing a

1. In several disciplines, such as social and medical sciences, the former is usually referred to as a *longitudinal study*, while the latter is similar to what is called a *cross-sectional study*.
2. http://www.sdss.org/

temporal sequence from out-of-order data points. To solve this problem, we proposed a convex program that optimizes a measure of temporal smoothness, along with efficient solution algorithms. When applied to a number of synthetic data sets and a gene expression time series data set, the proposed algorithms are able to learn moderately to highly accurate dynamic models, but at times suffer severely from the model ambiguity inherent in this setting.

We thus in Section 4 consider a second scenario: a small amount of sequence data and a lot of non-sequence data are available. The class of dynamic models studied here is restricted to first-order discrete-time stable vector auto-regressive (VAR) models, and the non-sequence data are assumed to be independent samples drawn from the stationary distribution of the VAR model. The latter assumption is valid when, for example, snapshots are taken from multiple trajectories of a VAR process after they have reached stationarity. We developed learning algorithms that minimize a novel penalized least square objective, which incorporates non-sequence data in the penalization term through violation of the Lyapunov equation relating the autoregressive model to the covariance of its stationary distribution. In a number of experiments on synthetic data, we observe that when the amount of sequence data is small, our proposed method of exploiting non-sequence data significantly improves over standard learning algorithms, which use only the sequence data, though the improvement gradually diminishes as the amount of sequence data grows. Experimental results on a video sequence demonstrate the same kind of improvement.

Then in Section 5 we list a few interesting topics we plan to explore. In modern time series applications, it is often the case that the vector of measured or observed variables is high-dimensional while the dynamics resides in a lower-dimensional latent state space. Common dynamic models for this setting include linear dynamical systems, Kalman filters, and hidden Markov models. We would like to investigate ways of exploiting non-sequence data in learning these models. In some other domains, prior knowledge or belief may suggest that the many observed variables can be partitioned into several unknown disjoint groups, and each group of variables evolves as an independent dynamic process. In such applications, identifying groupings of the variables will often greatly enhance understanding of the dynamic phenomenon, and may also reveal subsets of variables worth further studying. Moreover, knowing the group structure simplifies learning since each smaller model can be learnt independently. With only limited sequence data, it can be quite difficult to learn a partition of the variables. Under some model assumption, however, we observe that it is possible to exploit non-sequence data to help recover the group structure, and we plan to investigate this possibility in more depth. Finally, we plan to work on more real-world problems that will truly benefit from the proposed methods. We have already started analyzing a Parkinson's disease (PD) data set (Tsanas et al., 2010), which consists of features extracted from weekly voice recordings of 42 subjects over a period of six months. The full span of PD is usually on the scale of years, so being able to learn dynamics of PD's progression using non-sequence snapshots will greatly accelerate related research. Another problem is learning dynamic models for cell cycles by analyzing cell images. Since the photons emitted by the image-taking device are toxic to cells, it is difficult and costly to obtain successive images of the same cells. Our proposed methods thus have the potential to significantly reduce the effort for image collection while retaining much of the model accuracy. We have access to both sequence and non-sequence cell images, and some extracted features that

are proven useful in various cell cycle related recognition/classification tasks (Buck et al., 2009). By the end of this thesis research, we hope to demonstrate concrete progress by the proposed methods in at least these two application domains.

2. Related work

The most relevant work is perhaps the analysis of cell images by Buck et al. (2009). They were interested in the dependence of protein subcellular localization on the cell cycle, but instead of relying on time-series cell images as in most existing studies, they aim to utilize static, asynchronous snapshots taken from multiple cells at various phases of the cell cycle because such images are easier to obtain on a large scale than time-series images. To do so, they proposed to find a one-dimensional surrogate of cell cycle time from static cell image features by manifold learning techniques, and verified on real data that such a surrogate is well correlated with the cell cycle. However, it is not clear how to use or augment their approach for predictive analysis, which can be important in understanding cell dynamics. With our proposed methods, we hope to learn from such static snapshots of cells dynamic models that predict well.

A closely related problem studied in a number of disciplines is that of ordering a set of objects. Depending on the domain of interest, an ordering can be interpreted as progression of time, some coherent sequential structure or monotonic property. In natural language processing, the task of multi-document summarization requires ordering of sentences selected from different documents, and automatic title generation techniques construct a headline by selecting and ordering words from the input text (Barzilay and Elhadad, 2002; Deshpande et al., 2007). In multimedia analysis and retrieval, automatic generation of video or slideshow from photos involves laying down a coherent and smoothly transitioning sequence of scenes (Hua et al., 2004; Chen et al., 2006). Some of the techniques developed for these tasks are tailored to a specific problem domain, and most of them have access to some external knowledge about orderings of objects, such as time stamps of photos or grammatical rules for sentence compositions. In contrast, we consider a more general problem setting which relies on no or little domain specific knowledge, though our proposed methods make more explicit model assumptions.

The computational biology community has also studied the problem of ordering objects, in the context of finding a temporal ordering of static, asynchronous microarray measurement data (Magwene et al., 2003; Gupta and Bar-Joseph, 2008). The proposed methods therein, however, may be conceived as being less domain dependent and fall in a large family of algorithms for solving the *curve reconstruction problem*, which has been studied in various fields such as computational geometry (e.g., Giesen (1999)), statistics (Hastie and Stuetzle, 1989), and machine learning (Smola et al., 2001). More specifically, Magwene et al. (2003) proposed to reconstruct the temporal ordering of microarray samples through finding the minimum spanning tree on the graph formed by the sample points, while Gupta and Bar-Joseph (2008) proposed to solve an instance of the traveling salesman problem (TSP) and proved that under certain conditions on the dynamics generating the samples the optimal TSP path accurately reconstructs the true ordering. A key assumption behind these two methods is that temporally close sample points should also be spatially close. Both of these methods are unable to choose an overall direction of time, a limitation due to

their objective functions being invariant under both time directions. In fact, Peters et al. (2009) showed that under some linear dynamic models the true direction in time is not identifiable. Our problem setting differs from the aforementioned in that we consider snapshots from *multiple trajectories* of some dynamic process rather than out-of-order samples from a *single sequence*. Moreover, we focus more on learning a model for the underlying dynamics than simply ordering the data points. Under our assumption about the data, which will be formalized in subsequent sections, one can imagine the existence of an ordering of the data points based on their unobserved time stamps, but that ordering may not be very useful to existing dynamic model learning methods since they need as input a single or multiple sequences describing the status of *the same instances* over time. Nevertheless, ordering objects is still a useful component in our proposed methods in Section 3, but the objects being ordered, instead of raw data points, are some representative points discovered by clustering algorithms.

We also find two specific problems relevant to our proposal. One is collective inference on Markov models (Sheldon et al., 2008), which finds the most likely collection of paths on a trellis graph given observations on the collective behavior of a group of dynamic objects. Their motivation was to trace out trajectories of individual birds from aggregate statistics of an entire species of migrating birds. The other is connecting the dots between news articles (Shahaf and Guestrin, 2010), which aims to build a chronological *and* coherent story line of news that connects a given pair of starting and end articles, thereby providing readers a detailed description of the causal relationship between two events. A common feature in both problems is the need of identifying structures of sequentially matched objects from partially ordered data. A similar situation arises in one component of our methods, where the data points are put into ordered clusters for further processing (Section 3.3). But instead of finding hard matchings between data points in adjacent clusters, we take a soft-matching type of approach, updating the soft matching and the dynamic model alternatingly.

Finally, we briefly mention where our proposed work lies in the vast space of research on dynamical systems conducted in physics and mathematics. Most dynamical theories are concerned with the asymptotic behavior of some dynamical system, under various assumptions on the phase or state space of the system and the short-time evolution law (Katok and Hasselblatt, 1996). But our proposed work studies in some sense the reverse problem, that is, given observations that reflect the global status of a dynamical system, we try to develop methods that figure out the short-time or local evolution law.

3. Learning Vector Auto-regressive Models from Non-sequence Data

We start with first-order, discrete-time, fully observable linear dynamic models described as by the following transition function:

$$\mathbf{x}^{(t+1)} = A\mathbf{x}^{(t)} + \boldsymbol{\epsilon}^{(t+1)}, \tag{1}$$

where $\mathbf{x}^{(t)} \in \mathbb{R}^{p\times 1}$ is the state vector at time t, $A \in \mathbb{R}^{p\times p}$ is the state transition matrix, and $\boldsymbol{\epsilon}^{(t)}$ is the noise vector at time t. Such a model is also known as a first-order vector auto-regressive model (VAR) in the time series literature. For simplicity, we assume hereafter that $\forall t,\ \boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\cdot|\mathbf{0}, \sigma^2 I)$, a Gaussian distribution with zero mean and covariance $\sigma^2 I$, where I is the identity matrix. But the proposed methods in later sections all can be

Algorithm 1 Sampling from multiple trajectories

Input: transition matrix A, σ^2, $\mathbf{x}^{(0)}$, $T_{\max}$, and n
for $i = 1$ **to** n **do**
  Pick a random time stamp t_i from $\{1, \ldots, T_{\max}\}$.
  for $t = 1$ **to** t_i **do**
    $\mathbf{x}^{(t)} \leftarrow A\mathbf{x}^{(t-1)} + \boldsymbol{\epsilon}^{(t)}$, $\boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\cdot|\mathbf{0}, \sigma^2 I)$.
  end for
  Set $\mathbf{x}_i = \mathbf{x}^{(t_i)}$.
end for
Output: A sample $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$.

extended to handle general covariance matrices. The dynamical system also has a start state, which we denote as $\mathbf{x}^{(0)}$. Thus, the linear dynamic models we consider are fully characterized by $\Theta = \{A, \sigma^2, \mathbf{x}^{(0)}\}$.

When sequenced observations are available, a basic learning method is least-square linear regression of the observations at time t on the observations at time $t-1$, whose properties have been studied extensively (see e.g., Hamilton (1994)). The problem without observed state sequences is much more difficult. We assume that n executions of the dynamic model (1) have taken place, and from each execution we have observed a single data point drawn at random from the sequence of states generated in that execution. The result is n data points, $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, each from a different trajectory and having occurred at an unknown point in time. To avoid confusion in indices, hereafter we use parenthesized super-script, e.g., $\mathbf{x}^{(t)}$, to denote the time index, but sub-script, e.g., $\mathbf{x}_i$, to denote the data index. A precise description of this generative process is given in Algorithm 1.

We focus on estimating A and σ^2, and treat the start state $\mathbf{x}^{(0)}$ as a nuisance parameter. If the time index t_i of $\mathbf{x}_i$ is known, then by the properties of the Gaussian distribution we obtain the likelihood

$$L(\mathbf{x}_i|\boldsymbol{\theta}, t_i) \;=\; \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}^{(t_i)}, \Sigma^{(t_i)}) \;=\; \int \frac{\exp(-\frac{\|\mathbf{x}_i - A\mathbf{x}\|^2}{2\sigma^2})}{(2\pi\sigma^2)^{\frac{p}{2}}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i-1)}, \Sigma^{(t_i-1)}) d\mathbf{x}, \qquad (2)$$

where $\|\cdot\|$ is the vector two-norm, and

$$\boldsymbol{\mu}^{(t)} \;:=\; A^t \mathbf{x}^{(0)}, \qquad \Sigma^{(t)} \;:=\; \begin{cases} \sigma^2 \sum_{i=0}^{t-1} A^i \left(A^i\right)^\top, & t \geq 1, \\ \mathbf{0}, & t = 0. \end{cases} \qquad (3)$$

Since the n data points are drawn independently, we can factorize the likelihood of the sample points as

$$L(\mathbf{x}_1, \ldots, \mathbf{x}_n|\boldsymbol{\theta}, t_1, \ldots, t_n) = \prod_{i=1}^{n} L(\mathbf{x}_i|\boldsymbol{\theta}, t_i). \qquad (4)$$

The maximization of (4) is a challenging task because, as suggested by (3), the transition matrix A appears in (4) as polynomials whose degrees depend on the missing time indices t_i's.

Before presenting our proposed methods, we discuss several possibly non-identifiable properties of the model when the true temporal information is missing. Consider a simple linear dynamic model with the following transition matrix and initial point:

$$A = \begin{bmatrix} \cos\left(\frac{2\pi}{T}\right) & -\sin\left(\frac{2\pi}{T}\right) \\ \sin\left(\frac{2\pi}{T}\right) & \cos\left(\frac{2\pi}{T}\right) \end{bmatrix}, \qquad \mathbf{x}^{(0)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The ideal trajectory rolled out by this simple dynamic model lies on the unit circle in the two-dimensional Euclidean space. Suppose we observe a set of points from the ideal trajectory, but do not know their time indices. It is easy to see that all of the following dynamic models:

$$A(t) = \begin{bmatrix} \cos\left(\frac{2\pi t}{T}\right) & -\sin\left(\frac{2\pi t}{T}\right) \\ \sin\left(\frac{2\pi t}{T}\right) & \cos\left(\frac{2\pi t}{T}\right) \end{bmatrix}, \qquad t \in \{\pm 1, \pm 2, \ldots, \pm(T-1)\}$$

would explain the data equally well under any reasonable measure of goodness of fit. In the presence of process noise, some of these models may become less likely, but it would still be hard to uniquely determine the true dynamic model.

The above example suggests two possibly non-identifiable properties of the model: the overall direction in time and the speed of the underlying dynamics. In fact, Peters et al. (2009) showed that under some linear dynamic models the true direction in time is not identifiable. The methods proposed in subsequent sections thus do not resolve these ambiguities; the learnt model may follow either of the two directions in time, but usually corresponds to the slowest dynamics.

Another perhaps more intriguing example is depicted in Table 1, which presents a non-sequenced and noiseless data set in the left column and two possible dynamic models in the right column. On the one hand, according our assumption of a single fixed start state as in Algorithm 1, Model 1 should be favored over Model 2 under any reasonable measure of goodness of fit that incorporates such an assumption. On the other hand, under a certain level of noise and/or some non-uniform sampling rate in the temporal domain, the data generated from Model 1 may be more similar to a cylinder than to a spiral, making Model 2 equally or even more likely to have generated the data. There are more examples of this type, such as a torus of points where rotations around the short and the long circumferences can hardly be distinguished from each other in the absence of any temporal information. Further theoretical investigation is necessary to understand these issues and find out conditions under which these ambiguities can or cannot be resolved.

Next we present two of our methods for estimating A and σ^2, which are based on optimizing approximate posteriors by an Expectation Maximization (EM) type of algorithms. Then we demonstrate an extension for learning nonlinear dynamic models through reproducing kernels. For brevity, we present the main results and leave out most of the technical derivations, which are in our papers (Huang and Schneider, 2009; Huang et al., 2010).

3.1 Unordered and Partially-ordered Posteriors

We first remove the problem of unknown time indices by marginalizing out the missing t_i's. According to the generative process in Algorithm 1, the distributions of t_i's are independent

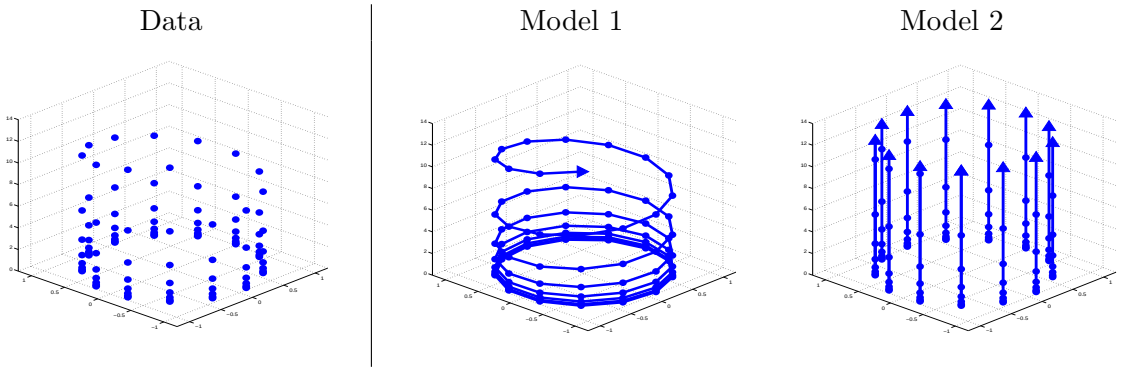


Table 1: An example of two possible dynamic models (right column) for a non-sequenced data set (left column).

from A and σ^2, and also mutually independent. Let $P(t_i)$ denote the probability mass function of $t_i \in \{1, \ldots, T_{\max}\}$. We then write

$$\begin{aligned} L(\mathbf{x}_1, \ldots, \mathbf{x}_n|\boldsymbol{\theta}) &:= \sum_{t_1=1}^{T_{\max}} \cdots \sum_{t_n=1}^{T_{\max}} L(\mathbf{x}_1, \ldots, \mathbf{x}_n, t_1, \ldots, t_n|\boldsymbol{\theta}) \\ &= \sum_{t_1=1}^{T_{\max}} \cdots \sum_{t_n=1}^{T_{\max}} \left(\prod_{i=1}^{n} L(\mathbf{x}_i|\boldsymbol{\theta}, t_i) P(t_i) \right) \\ &= \prod_{i=1}^{n} \sum_{t_i=1}^{T_{\max}} L(\mathbf{x}_i|\boldsymbol{\theta}, t_i) P(t_i). \end{aligned}$$

Plugging in the conditional likelihood (2), we obtain

$$\begin{aligned} L(\mathbf{x}_1, \ldots, \mathbf{x}_n|\boldsymbol{\theta}) &= \prod_{i=1}^{n} \sum_{t_i=1}^{T_{\max}} \left(\int \frac{\exp(-\frac{\|\mathbf{x}_i - A\mathbf{x}\|^2}{2\sigma^2})}{(2\pi\sigma^2)^{\frac{p}{2}}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i-1)}, \Sigma^{(t_i-1)}) d\mathbf{x} \right) P(t_i) \\ &= \prod_{i=1}^{n} \left(\int \frac{\exp(-\frac{\|\mathbf{x}_i - A\mathbf{x}\|^2}{2\sigma^2})}{(2\pi\sigma^2)^{\frac{p}{2}}} \left(\sum_{t_i=1}^{T_{\max}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i-1)}, \Sigma^{(t_i-1)}) P(t_i) \right) d\mathbf{x} \right). \quad (5) \end{aligned}$$

In the case of $P(t_i) = 1/T_{\max}$, i.e. t_i's are uniformly distributed, we have

$$\begin{aligned} \sum_{t_i=1}^{T_{\max}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i-1)}, \Sigma^{(t_i-1)}) P(t_i) &= \sum_{t_i=1}^{T_{\max}} \frac{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i-1)}, \Sigma^{(t_i-1)})}{T_{\max}} \\ &\approx \sum_{t_i=1}^{T_{\max}} \frac{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(t_i)}, \Sigma^{(t_i)})}{T_{\max}}, \quad (6) \end{aligned}$$

where the last term is the density that the data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ are generated from. This gives another view of the generative process: a random sample point is drawn (approximately) from (6) and an observation is created by applying (1) to it. However, (6) still

depends on the unknown t_i's. To remove this dependency, we replace (6) with its empirical estimate given by the sample points we have. This together with (5) lead to the following approximate likelihood:

$$\widehat{L}(\mathbf{x}_1, \ldots, \mathbf{x}_n|\boldsymbol{\theta}) \;:=\; \prod_{i=1}^{n} \left(\sum_{j \neq i} \frac{\exp(-\frac{\|\mathbf{x}_i - A\mathbf{x}_j\|^2}{2\sigma^2})}{(n-1)(2\pi\sigma^2)^{\frac{p}{2}}} \right). \tag{7}$$

We exclude the case that $\mathbf{x}_i$ generates itself to avoid the degenerate estimate $A = I$. Moreover, we impose a zero-mean Gaussian prior on A with precision λI and an inverse Gamma prior on σ^2 with shape and scale parameters α and β, leading to the approximate log-posterior:

$$\log \widehat{P}_{\mathrm{UM}}(\boldsymbol{\theta}|\mathbf{x}_1, \ldots, \mathbf{x}_n) \;:=\; \sum_{i=1}^{n} \log \left(\sum_{j \neq i} \frac{\exp(-\frac{\|\mathbf{x}_i - A\mathbf{x}_j\|^2}{2\sigma^2})}{(n-1)(2\pi\sigma^2)^{\frac{p}{2}}} \right) - \frac{\lambda}{2}\|A\|_F^2 - (\alpha+1)\log\sigma^2 - \frac{\beta}{\sigma^2}. \tag{8}$$

This is the objective our first method aims to maximize. Since there is no notion of ordering involved in (8), we refer to it as the Unordered Model (UM).

One can immediately see that (8) considers the data points as if each $\mathbf{x}_i$ were generated from some other $\mathbf{x}_j$ in the sample by (1). However, according to Algorithm 1 no $\mathbf{x}_i$ was generated from any other $\mathbf{x}_j$ in the sample. Such a discrepancy is due to our replacing (6) with its empirical estimate, and an immediate consequence is that σ^2 in (8) now accounts for not only the noise $\boldsymbol{\epsilon}$ in the dynamic model (1), but also the approximation error introduced by replacing (6) with the empirical density.

A second observation is that the approximate likelihood (7) is a product of summations of Gaussian densities, a structure similar to the likelihood of Gaussian Mixture Models (GMM), for which Expectation Maximization (EM) algorithms are a common estimation procedure. Although (8) is not a GMM, its similar structure allows us to derive an EM procedure with analytical update rules as summarized below.

E step:

$$\widetilde{Z}_{ij} \;=\; \begin{cases} \frac{\exp\left(-\frac{\|\mathbf{x}_i - A\mathbf{x}_j\|^2}{2\sigma^2}\right)}{\sum_{s \neq i} \exp\left(-\frac{\|\mathbf{x}_i - A\mathbf{x}_s\|^2}{2\sigma^2}\right)}, & i \neq j, \\ 0, & i = j. \end{cases} \tag{9}$$

M-step:

$$A \;=\; \left(\sum_{i=1}^{n}\sum_{j=1}^{n} \widetilde{Z}_{ij}\mathbf{x}_i\mathbf{x}_j^\top \right) \left(\sum_{i=1}^{n}\sum_{j=1}^{n} \widetilde{Z}_{ij}\mathbf{x}_j\mathbf{x}_j^\top + \lambda\sigma^2 I \right)^{-1}, \tag{10}$$

$$\sigma^2 \;=\; \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \widetilde{Z}_{ij}\|\mathbf{x}_i - A\mathbf{x}_j\|^2 + 2\beta}{pn + 2(\alpha+1)}, \tag{11}$$

Note that (11) can be easily generalized to handle general covariance structures.

It turns out that the EM algorithm outlined above is a version of the iteratively re-weighted least square (IRLS) procedure. Although it is simple and often computationally

efficient, one may worry that optimizing (8) without any directional consistency constraints may lead to degenerate dynamic models, especially when the sample size is limited. In fact, this happens in our experiments on simulated data. We thus proposed several variants of (8) that incorporate additional partial-order constraints. In one of them, the estimation procedure turns out to be very similar to the above EM algorithm, the only difference being the E-step replaced by maximum spanning tree on a weighted directed graph (Tarjan, 1977; Camerini et al., 1979). We refer to this partially-ordered model based method as PM.

3.2 Nonlinear Extension via Kernel Regression

To learn nonlinear dynamic models, we extend the aforementioned methods through kernel regression. We use UM as an example, but all other variants can similarly extended. We consider nonlinear dynamic models of the following form:

$$\mathbf{x}^{(t+1)} = B\phi(\mathbf{x}^{(t)}) + \boldsymbol{\epsilon}^{(t)},\ \boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\cdot|\mathbf{0}, \sigma^2 I). \tag{12}$$

where $\phi(\cdot)$ maps a point in $\mathbb{R}^p$ into a Reproducing Kernel Hilbert Space (RKHS) endowed with a kernel function $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y})$, and B is a linear mapping from the RKHS to $\mathbb{R}^p$. Replacing $A\mathbf{x}_j$ in (8) by $B\phi(\mathbf{x}_j)$ then leads to a nonlinear extension of the approximate log posterior. To extend the EM algorithm in Section 3.1 for learning B and σ^2, we first replace $A\mathbf{x}_j$ in (9) by $B\phi(\mathbf{x}_j)$ to obtain the E-step. Then for the M-step, we solve a weighted kernel least square problem, leading to the following update rules:

$$\begin{aligned} B &= \left(\sum_{i=1}^n \sum_{j=1}^n \widetilde{Z}_{ij} \mathbf{x}_i \phi(\mathbf{x}_j)^\top\right) \left(\sum_{i=1}^n \sum_{j=1}^n \widetilde{Z}_{ij} \phi(\mathbf{x}_j)\phi(\mathbf{x}_j)^\top + \lambda\sigma^2 I\right)^{-1} \\ &= X\widetilde{Z}\phi(X)^\top \left(\phi(X)\Lambda_{\widetilde{Z}}\phi(X)^\top + \lambda\sigma^2 I\right)^{-1} = X\widetilde{Z}\left(K\Lambda_{\widetilde{Z}} + \lambda\sigma^2 I\right)^{-1}\phi(X)^\top \end{aligned} \tag{13}$$

and

$$\sigma^2 = \frac{\sum_{i=1}^n \sum_{j=1}^n \widetilde{Z}_{ij}\|\mathbf{x}_i - B\phi(\mathbf{x}_j)\|^2 + 2\beta}{pn + 2(\alpha+1)}, \tag{14}$$

where $X := [\mathbf{x}_1\ \cdots\ \mathbf{x}_n]$ collects the data into a p-by-n matrix, $\phi(X) := [\phi(\mathbf{x}_1)\ \cdots\ \phi(\mathbf{x}_n)]$ is the RKHS mapping of the entire data set, $\Lambda_{\widetilde{Z}}$ is a diagonal matrix with $(\Lambda_{\widetilde{Z}})_{ii} := \sum_{j=1}^n \widetilde{Z}_{ji}$, and $K := \phi(X)^\top\phi(X)$ is the kernel matrix. We obtain (13) by using the Matrix Inversion lemma.

One issue with the above extension is that we cannot compute B when the mapping $\phi(\cdot)$ is of infinite dimension. However, we observe that the EM procedures only make use of $B\phi(X)$, and according to (13)

$$\begin{aligned} B\phi(X) &= X\widetilde{Z}\left(K\Lambda_{\widetilde{Z}} + \lambda\sigma^2\right)^{-1}\phi(X)^\top\phi(X) \\ &= X\widetilde{Z}\left(K\Lambda_{\widetilde{Z}} + \lambda\sigma^2\right)^{-1} K. \end{aligned}$$

Therefore, instead of B we maintain and update a p-by-n matrix $M := X\widetilde{Z}\left(K\Lambda_{\widetilde{Z}} + \lambda\sigma^2\right)^{-1}$ in the EM iterations. To predict the next state for a new observation $\mathbf{x}$, we compute $M\phi(X)^\top\phi(\mathbf{x})$, which also only requires kernel evaluations. Alternatively, we may compute

a finite-dimensional approximation to $\phi(X)$ by doing a low-rank factorization of the kernel matrix $K \approx \widetilde{\phi}(X)^\top \widetilde{\phi}(X)$, and replace $\phi(X)$ in the EM procedure with $\widetilde{\phi}(X) \in \mathbb{R}^{m \times n}, m < n$. Then we can maintain and update $B \in \mathbb{R}^{p \times m}$ explicitly. To do prediction on a set of new data points, we project them onto the basis found by factorizing the training kernel matrix, thereby computing their finite-dimensional approximation $\widetilde{\phi}$, and then apply the estimated B to the mapped points.

3.3 Initializing EM by Temporal Smoothing

All of the proposed methods are solving non-convex optimization problems, and avoiding local optima is a critical issue. A common practice in applying EM methods is to run the algorithm multiple times, each with a randomly initialized model, and then choose the best local optimum as the final estimate. We follow this practice in our experiments on simulated data in Section 3.4, but observe that the number of random restarts needed to obtain a good model is usually large, meaning that a lot of random initializations lead to undesirable local optima. Moreover, our simulated data are low dimensional, but the problem caused by local optima will only become worse in a higher dimension, which is common with real data. We thus investigate an alternative way of initialization.

We begin by observing that in the case of a linear dynamic model, the samples generated by Algorithm 1 can be viewed as i.i.d. samples drawn from the following mixture of Gaussians:

$$\mathbf{x} \sim \sum_{t=1}^{T_{\max}} \pi^{(t)} \mathcal{N}(\cdot | \boldsymbol{\mu}^{(t)}, \Sigma^{(t)}), \tag{15}$$

where $\boldsymbol{\mu}^{(t)}$ and $\Sigma^{(t)}$ are defined in (3) and $\pi^{(t)} \geq 0$ is the probability that $\mathbf{x}$ is drawn at time t. Based on this view, we devise a heuristic to initialize the model:

1. Estimate $\boldsymbol{\mu}^{(t)}$'s by fitting a GMM to the data
2. Estimate the true temporal order of $\boldsymbol{\mu}^{(t)}$'s based on their estimates from Step 1
3. Learn a dynamic model from the estimated sequence of $\boldsymbol{\mu}^{(t)}$'s by existing dynamic model learning methods

For Step 1 we can use the standard EM algorithm for learning GMMs, or simply the k-means algorithm since subsequent steps only need estimates of the means. Step 2 in its own right is a challenging problem. If we believe temporally close $\boldsymbol{\mu}^{(t)}$'s should be similar, we can compute pairwise distances between estimates of $\boldsymbol{\mu}^{(t)}$'s and solve a traveling salesman problem (TSP). Then we need to decide the direction of time on the TSP path, which is often impossible without prior or expert knowledge. In our experiments in Section 3.4 we simply try both directions and report the one that performs better. In high dimensions, Euclidean distances suffer from the curse of dimensionality and are vulnerable to noise. We thus propose an alternative way to recover the true temporal order, which is based on the idea of temporal smoothing.

Unlike methods proposed in previous sections, the method we present in the following does not make any assumptions about the functional form of the underlying dynamic model. It only assumes the underlying dynamics to be *smooth*, i.e., the curvature of the trajectory

rolled out by the dynamic model is small. More precisely, we quantify smoothness by the second order differences of temporally adjacent points generated by the dynamic model:

$$S = \sum_{t=2}^{T_{\max}-1} \|(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}) - (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})\|^2, \tag{16}$$

where $T_{\max}$ is the maximum time. Small values of S correspond to smooth trajectories. Such a smoothness measure has been used as the regularization term in the Hodrick-Prescott filter (Hodrick and Prescott, 1997; Lesser, 1961), a common tool in macroeconomics for obtaining a smooth and nonlinear representation of a time series.

The quantity (16) cannot be computed on our data since the true time indices of the data points are missing. Nevertheless, it can be succinctly expressed using the Laplacian L of the *temporal adjacency graph* obtained by connecting temporally adjacent pairs of data points. More specifically, we let $X := [\mathbf{x}_1 \ \cdots \ \mathbf{x}_n]$ be the p-by-n data matrix as before, and Z be a directed *temporal* adjacency matrix such that $Z_{ij} = 1$ if $\mathbf{x}_j$ precedes $\mathbf{x}_i$ immediately in time, and 0 otherwise. Then, we define $\overline{Z} := Z + Z^\top$ to represent the undirected, symmetric temporal adjacency of the data points. If the data points were sorted according to their true temporal order, the matrix $\overline{Z}$ would consist of ones in the upper-first and lower-first off-diagonals and zeros elsewhere. The graph Laplacian based on the adjacency matrix $\overline{Z}$ is then $L = \mathsf{diag}(\overline{Z}\mathbf{e}) - \overline{Z}$, where $\mathbf{e}$ is a vector of ones and $\mathsf{diag}(\overline{Z}\mathbf{e})$ denotes the diagonal matrix with the vector $\overline{Z}\mathbf{e}$ in the main diagonal. Simple algebraic manipulation shows that the smoothness S can be expressed in terms of L (hence Z) as follows:

$$S(Z) \ = \ \|XL\|_F^2 \ = \ \mathrm{tr}((\mathsf{diag}(\overline{Z}\mathbf{e}) - \overline{Z})^\top X^\top X(\mathsf{diag}(\overline{Z}\mathbf{e}) - \overline{Z})), \tag{17}$$

which is a quadratic and convex function of $\overline{Z}$ and hence Z. Since we assume the true dynamics to be smooth, a natural way to reconstruct a temporal ordering would be to solve the following problem:

$$\begin{aligned} Z^* &= \arg\min_{Z} \ S(Z) \\ &\text{s.t. } Z \text{ represents a directed Hamiltonian path through the data points.} \end{aligned} \tag{18}$$

However, this problem is essentially a quadratic version of TSP, and to the best of our knowledge, no efficient solver exists for such problems. We thus consider the following two-step heuristics. In the first step, we minimize $S(Z)$ under a modified set of constraints:

$$\begin{aligned} \widehat{Z} \ &= \ \arg\min_{Z} \quad S(Z) \\ \text{s.t.} \quad & Z\mathbf{e} = \mathbf{e}, \quad Z^\top\mathbf{e} = \mathbf{e}, \quad Z_{ij} \geq 0, \quad Z_{ii} = 0. \end{aligned} \tag{19}$$

The constraints in (19) are not a proper relaxation of (18) because Z must have one zero row and one zero column to represent a Hamiltonian path. Nevertheless, we can interpret solving (19) as learning a pairwise similarity $\widehat{Z}$ whose (i,j)-th entry reflects how likely $\mathbf{x}_j$ is to precede $\mathbf{x}_i$ temporally. Then in the second step, we solve an instance of TSP with $1 - (\widehat{Z} + \widehat{Z}^\top)/2$ as the distance, and obtain an ordering from the optimal TSP path.

The optimization problem (19) is essentially convex quadratic programming (QP) under linear and bound constraints. However, the number of variables is *quadratic* in the number

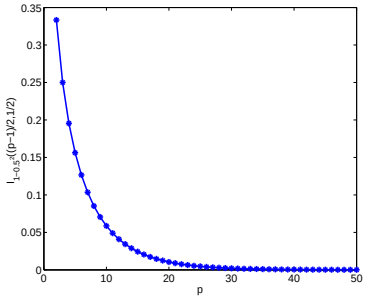


Figure 1: The probability (21) for $s = 0.5$ as a function of the dimension p

of data points, and as the data size increases, directly applying a general-purpose QP or nonlinear programming solver may become inefficient or even infeasible. We thus devise a simple and efficient *projected gradient method*, Algorithm 3 in (Huang et al., 2010), that iteratively updates the rows and the columns of Z.

3.4 Some Experimental Results

Here we present a selective set of our experimental results. More results can be found in our papers (Huang and Schneider, 2010, 2009; Huang et al., 2010).

Our evaluation scheme is as follows. Let $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(T)}\}$ be a test sequence of state vectors. After we learn a dynamic model from a training data set, we predict for each $\mathbf{x}^{(t)}, t = 1, \ldots, T-1$ the next state $\widehat{\mathbf{x}}^{(t+1)}$. To evaluate the predictions, we use the following two measures:

Cosine Score:

$$\frac{1}{T-1}\left|\sum_{t=1}^{T-1}\frac{(\mathbf{x}^{(t+1)}-\mathbf{x}^{(t)})^\top(\widehat{\mathbf{x}}^{(t+1)}-\mathbf{x}^{(t)})}{\|\mathbf{x}^{(t+1)}-\mathbf{x}^{(t)}\|\|\widehat{\mathbf{x}}^{(t+1)}-\mathbf{x}^{(t)}\|}\right|, \tag{20}$$

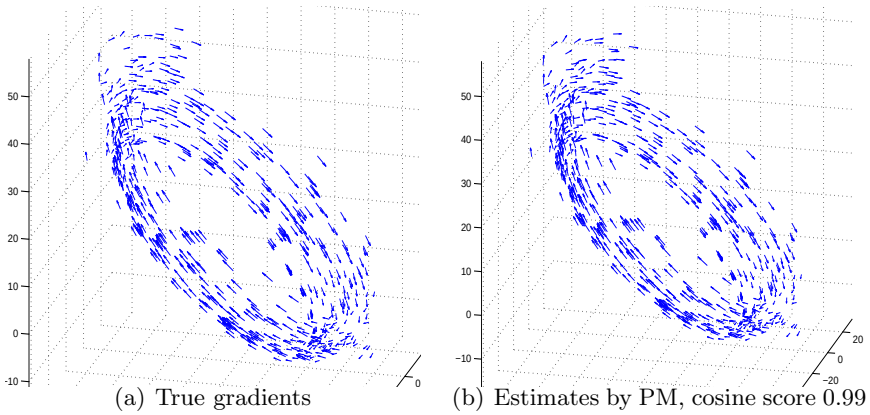
A higher score means a better prediction. To interpret the cosine score, let us consider the probability a random prediction achieves some cosine score s. Since we normalize the movements, it suffices to consider the probability that the angle between a random p-dimensional unit vector and some fixed unit vector is no more than $\cos^{-1}(s)$. This probability is equivalent to the ratio of the surface area of a cap with height $1-s$ on a unit p-sphere to the unit p-sphere surface area. Li (2011) gave a closed-form formula of such a ratio:

$$\frac{1}{2}\mathcal{I}_{1-s^2}\left(\frac{p-1}{2}, \frac{1}{2}\right), \tag{21}$$

where $\mathcal{I}_x(a, b)$ is the regularized incomplete beta function. Figure 1 shows this ratio for $s = 0.5$ as a function of p. We can see that the probability decreases quickly as p increases, and when $p = 50$ the probability for a random prediction to achieve a cosine score of 0.5 is less than 10^{-4}.

Normalized Error:

$$\frac{1}{T-1}\sum_{t=1}^{T}\frac{\|\mathbf{x}^{(t+1)}-\widehat{\mathbf{x}}^{(t+1)}\|}{\|\mathbf{x}^{(t+1)}-\mathbf{x}^{(t)}\|}, \tag{22}$$



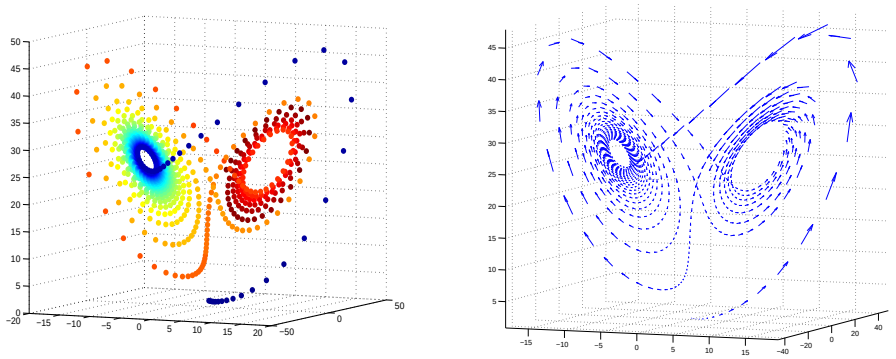
(a) True gradients (b) Estimates by PM, cosine score 0.99

Figure 2: Results on a three-dimensional linear system

which measures how close the predictions are to the true next state vectors. A smaller normalized error means a better prediction, and a constant prediction ($\widehat{\mathbf{x}}^{(t+1)} = \mathbf{x}^{(t)}$) gives a normalized error of one.

In Figures 2 and 3 we give qualitative results on two synthetic data sets respectively, one generated with Algorithm 1 from a three-dimensional linear system, and the other sampled from the noiseless trajectory of the Lorenz attractor (Lorenz, 1963). We did not use Algorithm 1 with the Lorenz attractor because it is a chaotic system and even little noise can disturb the system trajectory significantly. Both figures demonstrate that the estimated gradient vectors $\widehat{A}\mathbf{x}^{(t)} - \mathbf{x}^{(t)}$ by our proposed methods are very close to the true gradient vectors $\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}$, achieving very good cosine scores of 0.99 and 0.98, respectively. However, these results were obtained with tens to hundreds of random initializations, and a lot of the random initializations led to degenerate or undesirable local optima.

Figure 4 shows results on two real data sets: a video sequence of a swinging pendulum (Siddiqi et al., 2010) and gene expression time series of yeast (Tu et al., 2005). The pendulum data, after the raw pixels being pre-processed, consist of 500 sequential points in 20-dimension, the first 400 used as training data and the last 100 as testing data. The yeast data are expression profiles of 3,552 genes along three consecutive metabolic cycles, each containing 12 measurements. Gene expressions in the first two cycles (24 points) were used as training data, the remaining as testing data. We applied the temporal smoothing heuristics in Section 3.3, with k-means as the clustering method, to initialize our proposed UM and PM algorithms. Since temporal sequences in both cases are available, we compared our proposed methods with the usual penalized least square estimator, shown as a blue-dashed line in Figure 4. On both data sets, the proposed methods were able to perform quite as well as the sequential learning method, though on the yeast data the performance gap is larger and the numerical values of the two measures are worse than on the pendulum data. This is reasonable because the yeast data are very high dimensional and contain few data points. Yet, a cosine score of 0.66 in 3,552 dimensions is very significant in light of the interpretation in (21). Moreover, we also see that the proposed methods improved modestly over the initial model.



(a) Ideal trajectory. Time forwards as points turn from blue to red.

(b) Estimated gradients by kernelized UM, cosine score 0.98

Figure 3: Results on Lorenz Attractor

4. Combining Sequence and Non-sequence Data

Here we aim to combine these two types of data to improve learning of dynamic models. We assume that a small amount of sequence samples and a large amount of non-sequence samples are available. Our aim is to rely on the few sequence samples to obtain a rough estimate of the model, while refining this rough estimate using the non-sequence samples. As in previous sections we consider p-dimensional vector auto-regressive models, but treat the state variables as a row vector instead of a column vector:

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} A + \boldsymbol{\epsilon}^{(t+1)}. \tag{23}$$

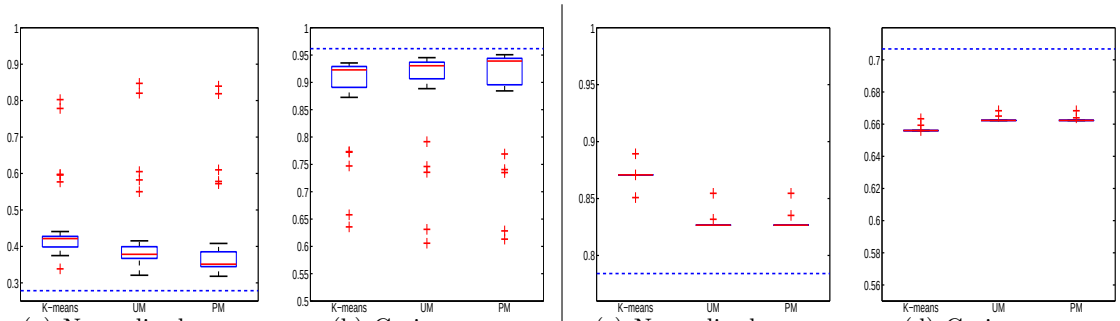
In addition, we assume that the process (23) is stable, i.e., the eigenvalues of A have modulus less than one. As a result, the process (23) has a stationary distribution, whose covariance Q is determined by the following discrete-time Lyapunov equation:

$$A^\top Q A + \sigma^2 I = Q. \tag{24}$$

Linear quadratic Lyapunov theory (see e.g., Antsaklis and Michel (2005)) gives that Q is *uniquely* determined if and only if $\lambda_i(A)\lambda_j(A) \neq 1$ for $1 \leq i, j \leq p$, where $\lambda_i(A)$ is the i-th eigenvalue of A. If the noise process $\boldsymbol{\epsilon}^t$ follows a normal distribution, the stationary distribution also follows a normal distribution, with covariance Q determined as above. Since our goal is to estimate A, a more relevant perspective is viewing (24) as a system of constraints on A. What motivates this work is that the estimation of Q requires only samples drawn from the stationary distribution rather than sequence data. However, even if we have the true Q and σ^2, we still cannot uniquely determine A because (24) is an under-determined system[3] of A. We thus rely on the few sequence samples to resolve the ambiguity.

Let $\{\mathbf{x}^{(i)}\}_{i=1}^T$ be a sequence of observations generated by the process (23). The standard least-square estimator for the transition matrix A is the solution to the following

3. If we further require A to be symmetric, (24) would be a simplified *Continuous-time Algebraic Riccati Equation*, which has a unique solution under some conditions (c.f. Antsaklis and Michel (2005)).



(a) Normalized error (b) Cosine score (c) Normalized error (d) Cosine score

Swinging pendulum | Yeast gene expressions

Figure 4: Results on real data. Blue-dashed lines resulted from using known sequences.

minimization problem:

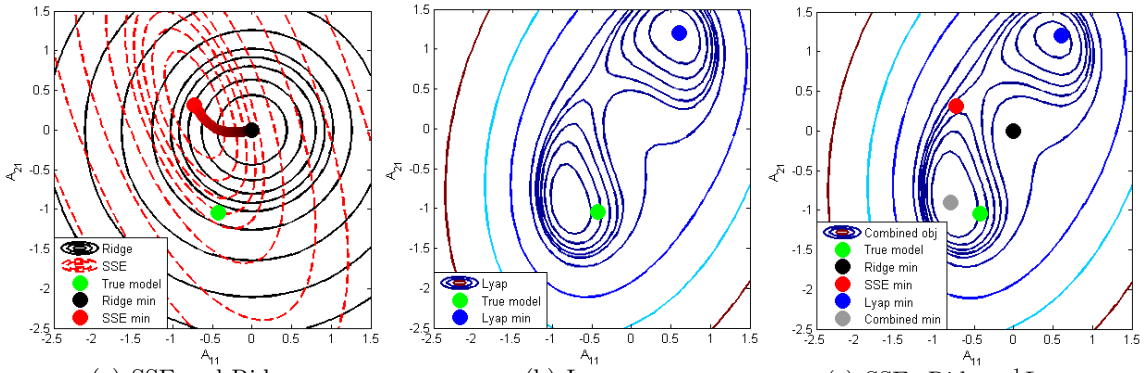
$$\min_{A} \quad \|Y - XA\|_F^2, \tag{25}$$

where $Y^\top := [(\mathbf{x}^{(2)})^\top \ (\mathbf{x}^{(3)})^\top \cdots (\mathbf{x}^{(T)})^\top]$, $X^\top := [(\mathbf{x}^{(1)})^\top \ (\mathbf{x}^{(2)})^\top \cdots (\mathbf{x}^{(T-1)})^\top]$, and $\|\cdot\|_F$ denotes the matrix Frobenius norm. When $p > T$, which is often the case in modern time series modeling tasks, the least square problem (25) has multiple solutions all achieving zero squared error, and the resulting estimator overfitts the data. A common remedy is adding a penalty term on A to (25) and minimizing the resulting regularized sum of squared errors. Usual penalty terms include the ridge penalty $\|A\|_F^2$ and the sparse penalty $\|A\|_1 := \sum_{i,j} |A_{ij}|$.

Now suppose we also have a set of non-sequence observations $\{\mathbf{z}_i\}_{i=1}^n$ drawn independently from the stationary distribution of (23). Note that we use superscripts for time indices and subscripts for data indices. As described in Section 1, the size n of the non-sequence sample can usually be much larger than the size T of the sequence data. To incorporate the non-sequence observations into the estimation procedure, we first obtain a covariance estimate $\widehat{Q}$ of the stationary distribution from the non-sequence sample, and then turn the Lyapunov equation (24) into a regularization term on A. More precisely, in addition to the usual ridge or sparse penalty terms, we also consider the following regularization:

$$\|A^\top \widehat{Q} A + \sigma^2 I - \widehat{Q}\|_F^2, \tag{26}$$

which we refer to as the *Lyapunov penalty*. To compare (26) with the ridge penalty and the sparse penalty, we consider (25) as a multiple-response regression problem and view the i-th column of A as the regression coefficient vector for the i-th output dimension. From this viewpoint, we immediately see that both the ridge and the sparse penalizations treat the p regression problems as unrelated. On the contrary, the Lyapunov penalty incorporates relations between pairs of columns of A by using a covariance estimate $\widehat{Q}$. In other words, although the non-sequence sample does not provide direct information about the individual regression problems, it does reveal how the regression problems are related to one another. To illustrate how the Lyapunov penalty may help to improve learning, we give an example



(a) SSE and Ridge (b) Lyap (c) SSE+Ridge+$\frac{1}{2}$Lyap

Figure 5: Level sets of different functions in a bivariate AR example

in Figure 5. The true transition matrix is

$$A = \begin{bmatrix} -0.4280 & 0.5723 \\ -1.0428 & -0.7144 \end{bmatrix} \tag{27}$$

and $\boldsymbol{\epsilon}^t \sim \mathcal{N}(\mathbf{0}, I)$. We generate a sequence of 4 points, draw a non-sequence sample of 20 points independently from the stationary distribution and obtain the sample covariance $\widehat{Q}$. We fix the second column of A but vary the first, and plot in Figure 5(a) the resulting level sets of the sum of squared errors on the sequence (SSE) and the ridge penalty (Ridge), and in Figure 5(b) the level sets of the Lyapunov penalty (Lyap). We also give coordinates of the true $[A_{11}\ A_{21}]^\top$, the minima of SSE, Ridge, and Lyap, respectively. To see the behavior of the ridge regression, we trace out a path of the ridge regression solution by varying the penalization parameter, as indicated by the red-to-black curve in Figure 5(a). This path is pretty far from the true model, due to insufficient sequence data. For the Lyapunov penalty, we observe that it has two local minima, one of which is very close to the true model, while the other, also the global minimum, is very far. Thus, neither ridge regression nor the Lyapunov penalty can be used on its own to estimate the true model well. But as shown in Figure 5(c), the combined objective, SSE+Ridge+$\frac{1}{2}$Lyap, has its global minimum very close to the true model. This demonstrates how the ridge regression and the Lyapunov penalty may complement each other: the former by itself gives an inaccurate estimation of the true model, but is just enough to identify a good model from the many candidate local minima provided by the latter.

In the following we describe our proposed methods for incorporating the Lyapunov penalty (26) into ridge and sparse least-square estimation. We also discuss robust estimation for the covariance Q. Most of these results are in (Huang and Schneider, 2011).

4.1 Ridge and Lyapunov penalty

Here we estimate A by solving the following problem:

$$\min_A \quad \frac{1}{2}\|Y - XA\|_F^2 + \frac{\lambda_1}{2}\|A\|_F^2 + \frac{\lambda_2}{4}\|A^\top \widehat{Q} A + \sigma^2 I - \widehat{Q}\|_F^2, \tag{28}$$

where $\widehat{Q}$ is a covariance estimate obtained from the non-sequence sample. We treat λ_1, λ_2 and σ^2 as hyperparameters and determine their values on a validation set. Given these hyperparameters, we solve (28) by gradient descent with back-tracking line search for the step size. The gradient of the objective function is given by

$$-X^\top Y + X^\top X A + \lambda_1 A + \lambda_2 \widehat{Q} A (A^\top \widehat{Q} A + \sigma^2 I - \widehat{Q}). \tag{29}$$

As mentioned before, (28) is a non-convex problem and thus requires good initialization. We use the following two initial estimates of A:

$$\widehat{A}^{lsq} := (X^\top X)^\dagger X^\top Y \quad \text{and} \quad \widehat{A}^{ridge} := (X^\top X + \lambda_1 I)^{-1} X^\top Y, \tag{30}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo inverse of a matrix, making $\widehat{A}^{lsq}$ the minimum-norm solution to the least square problem (25). We run the gradient descent algorithm with these two initial estimates, and choose the estimated A that gives a smaller objective.

4.2 Sparse and Lyapunov penalty

Sparse learning for vector auto-regressive models has become a useful tool in many modern time series modeling tasks, where the number p of states in the system is usually larger than the length T of the time series. For example, an important problem in computational biology is to understand the progression of certain biological processes from some measurements, such as temporal gene expression data.

Using an idea similar to (28), we estimate A by

$$\begin{aligned} \min_A \quad & \frac{1}{2}\|Y - XA\|_F^2 + \frac{\lambda_2}{4}\|A^\top \widehat{Q} A + \sigma^2 I - \widehat{Q}\|_F^2, \\ \text{s.t.} \quad & \|A\|_1 \le \lambda_1. \end{aligned} \tag{31}$$

Instead of adding a sparse penalty on A to the objective function, we impose a constraint on the ℓ_1 norm of A. Both the penalty and the constraint formulations have been considered in the sparse learning literature, and shown to be equivalent in the case of a convex objective. Here we choose the constraint formulation because it can be solved by a simple projected gradient descent method. On the contrary, the penalty formulation leads to a non-smooth and non-convex optimization problem, which is difficult to solve with standard methods for sparse learning. In particular, the soft-thresholding-based coordinate descent method for LASSO does not apply due to the Lyapunov regularization term. Moreover, most of the common methods for non-smooth optimization, such as bundle methods, solve convex problems and need non-trivial modification in order to handle non-convex problems (Noll et al., 2008).

Let $J(A)$ denote the objective function in (31) and $A^{(k)}$ denote the intermediate solution at the k-th iteration. Our projected gradient method updates $A^{(k)}$ to $A^{(k+1)}$ by the following rule:

$$A^{(k+1)} \leftarrow \Pi(A^{(k)} - \eta^{(k)} \nabla J(A^{(k)})), \tag{32}$$

where $\eta^{(k)} > 0$ denotes a proper step size, $\nabla J(A^{(k)})$ denotes the gradient of $J(\cdot)$ at $A^{(k)}$, and $\Pi(\cdot)$ denotes the projection onto the feasible region $\|A\|_1 \le \lambda_1$. More precisely, for any p-by-p real matrix V we define

$$\Pi(V) := \arg\min_{\|A\|_1 \le \lambda_1} \|A - V\|_F^2. \tag{33}$$

Algorithm 2 Armijo's rule along the projection arc

Input: $A^{(k)}, \nabla J(A^{(k)}), 0 < \beta < 1, 0 < c < 1$
Output: $A^{(k+1)}$

1: Find $\eta^{(k)} = \max\{\beta^{r_k} | r_k \in \{0, 1, \ldots\}\}$ such that $A^{(k+1)} := \Pi(A^{(k)} - \eta^{(k)} \nabla J(A^{(k)}))$ satisfies

$$J(A^{(k+1)}) - J(A^{(k)}) \leq c \operatorname{tr}\left(\nabla J(A^{(k)})^\top (A^{(k+1)} - A^{(k)})\right) \tag{35}$$

To compute the projection, we use the efficient ℓ_1 projection technique given in Figure 2 of Duchi et al. (2008), whose expected running time is linear in the size of V.

For choosing a proper step size $\eta^{(k)}$, we consider the simple and effective *Armijo rule along the projection arc* described by Bertsekas (1999). This procedure is given in Algorithm 4.2, and the main idea is to ensure a sufficient decrease in the objective value per iteration (35). Bertsekas (1999) proved that there always exists $\eta^{(k)} = \beta^{r_k} > 0$ satisfying (35), and every limit point of $\{A^{(k)}\}_{k=0}^{\infty}$ is a stationary point of (31). In our experiments we set $c = 0.01$ and $\beta = 0.1$, both of which are typical values used in gradient descent. As in the previous section, we need good initializations for the projected gradient descent method. Here we use these two initial estimates:

$$\widehat{A}^{lsq'} := \arg\min_{\|A\| \leq \lambda_1} \|A - \widehat{A}^{lsq}\|_F^2 \quad \text{and} \quad \widehat{A}^{sp} := \arg\min_{\|A\| \leq \lambda_1} \frac{1}{2}\|Y - XA\|_F^2, \tag{34}$$

where $\widehat{A}^{lsq}$ is defined in (30), and then choose the one leading to a smaller objective value.

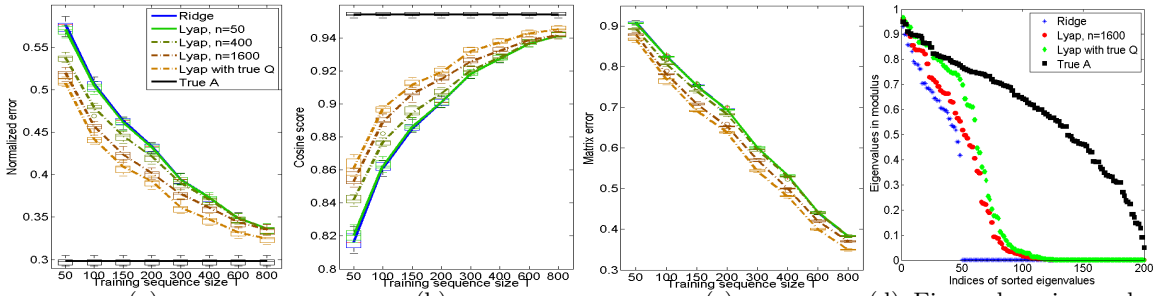
4.3 Robust estimation of covariance matrices

To obtain a good estimator for A using the proposed methods, we need a good estimator for the covariance of the stationary distribution of (23). Given an independent sample $\{\mathbf{z}_i\}_{i=1}^n$ drawn from the stationary distribution, the sample covariance is defined as

$$S := \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{z}_i - \bar{\mathbf{z}})^\top (\mathbf{z}_i - \bar{\mathbf{z}}), \quad \text{where } \bar{\mathbf{z}} := \frac{\sum_{i=1}^n \mathbf{z}_i}{n}. \tag{36}$$

Although unbiased, the sample covariance is known to be vulnerable to outliers, and ill-conditioned when the number of sample points n is smaller than the dimension p. Both issues arise in many real world problems, and the latter is particularly common in gene expression analysis. Therefore, researchers in many fields, such as statistics (Stein, 1975; Yang and Berger, 1994; Ledoit and Wolf, 2004), finance (Ledoit and Wolf, 2003), signal processing (Chen et al., 2010a,b), and recently computational biology (Schäfer and Strimmer, 2005), have investigated robust estimators of covariances. Most of these results originate from the idea of *shrinkage estimators*, which shrink the covariance matrix towards some target covariance with a simple structure, such as a diagonal matrix. It has been shown in, e.g., (Stein, 1975; Ledoit and Wolf, 2003) that shrinking the sample covariance can achieve a smaller mean-squared error (MSE). More specifically, Ledoit and Wolf (2003) considers the following linear shrinkage:

$$\widehat{Q} = (1 - \alpha)S + \alpha F \tag{37}$$



(a) (b) (c) (d) Eigenvalues in modulus

Figure 6: Testing performances and eigenvalues in modulus for the dense model

for $0 < \alpha < 1$ and some target covariance F, and derive a formula for the optimal α that minimizes the mean-squared error:

$$\alpha^* := \arg\min_{0 \le \alpha \le 1} \mathbb{E}(\|\widehat{Q} - Q\|_F^2), \tag{38}$$

which involves unknown quantities such as true covariances of S. Schäfer and Strimmer (2005) proposed to estimate α^* by replacing all the population quantities appearing in α^* by their unbiased empirical estimates, and derived the resulting estimator $\widehat{\alpha}^*$ for several types of target F. For the preliminary experiments here we use the estimator proposed by Schäfer and Strimmer (2005) with the following F:

$$F_{ij} = \begin{cases} S_{ij}, & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases} \quad 1 \le i, j \le p. \tag{39}$$

4.4 Experiments

To evaluate the proposed methods, we conduct experiments on synthetic and video data. For brevity we skip some details, which can all be found in (Huang and Schneider, 2011).

In all our experiments, we have a training sequence, a testing sequence, and a non-sequence sample. We split the training sequence into two halves and use the second half as a validation sequence for a grid search over the hyper-parameters λ_1, λ_2 and σ^2, In most of our experiments, we find that the proposed methods are much less sensitive to σ^2 than to λ_1 and λ_2.

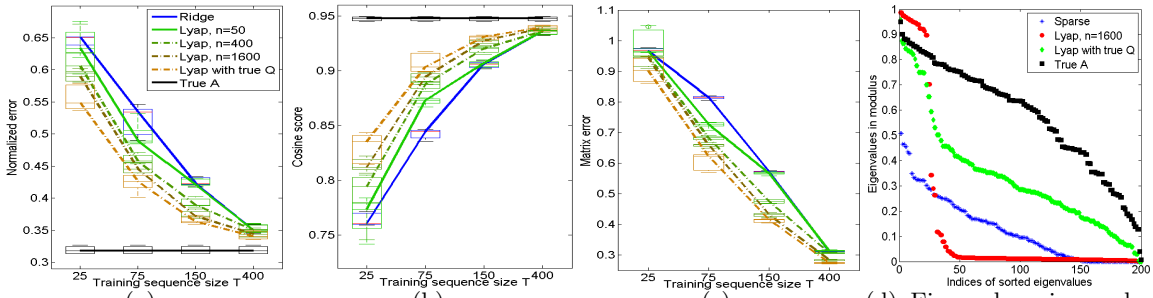
4.4.1 Synthetic Data

We consider the following two VAR models with Gaussian noise $\boldsymbol{\epsilon}^t \sim \mathcal{N}(\mathbf{0}, I)$.

$$\text{Dense Model:} \quad A = \frac{0.95M}{\max(|\lambda_i(M)|)}, M_{ij} \sim \mathcal{N}(0,1), 1 \le i, j \le 200.$$

$$\text{Sparse Model:} \quad A = \frac{0.95(M \odot B)}{\max(|\lambda_i(M \odot B)|)}, M_{ij} \sim \mathcal{N}(0,1), B_{ij} \sim \text{ Bern } (1/8), 1 \le i, j \le 200,$$

where Bern(h) is the Bernoulli distribution with success probability h, and $\odot$ denotes the entrywise product of two matrices. Both models are stable. We set the length of the



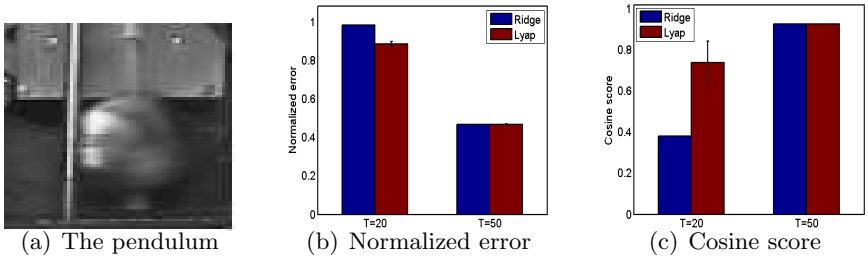
(a) (b) (c) (d) Eigenvalues in modulus

Figure 7: Testing performances and eigenvalues in modulus for the sparse model

testing sequence to be 800, and vary the training sequence length T and the non-sequence sample size n. Under each combination of T and n, we compare the proposed Lyapunov penalization method with the baseline approach of penalized least square, which uses only the sequence data. To investigate the limit of the proposed methods, we also use the true Q for the Lyapunov penalization. We run 10 such experiments for the dense model and 5 for the sparse model, and report the overall performances in terms of the normalized error (22), the cosine score (20), and the matrix error $\|A - \widehat{A}\|_F^2$.

Experimental results for the dense model are in Figures 6(a) to 6(c) for the three performance measures. The ridge regression approach and the proposed Lyapunov penalization method (28) are abbreviated as Ridge and Lyap, respectively. One main observation is that Lyap improves over Ridge more significantly when the training sequence length T is small (≤ 200) and the non-sequence sample size n is large (≥ 400). When T is large, Ridge already performs quite well and Lyap does not improve the performance much. But with the true stationary covariance Q, Lyap outperforms Ridge significantly for all T. When n is small, the covariance estimate $\widehat{Q}$ is far from the true Q and the Lyapunov penalty does not provide useful information about A. In this case, the value of λ_2 determined by the validation performance is usually quite small (0.5 or 1) compared to λ_1 (256), so the two methods perform similarly on testing sequences. A precise statement on how the estimation error in Q affects $\widehat{A}$ is worth studying in the future. We also plot in Figure 6(d) the sorted eigenvalues in modulus of the true A and the $\widehat{A}$'s when $T = 50$ and $n = 1600$. Both Ridge and Lyap severely under-estimate the eigenvalues in modulus, but Lyap preserves the spectrum much better than Ridge.

Results for the sparse model are in Figures 7(a) to 7(c), and the eigenvalues in modulus are in Figure 7(d). The sparse least-square method and the proposed method (31) are abbreviated as Sparse and Lyap, respectively. We observe the same type of improvement as in the dense model: Lyap improves over Sparse more significantly when T is small and n is large. A major difference lies in the impact of the Lyapunov penalization on the spectrum of $\widehat{A}$, as revealed in Figure 7(d). When T is as small as 25, the sparse least-square method shrinks all the eigenvalues but still keep most of them non-zero, while Lyap with a non-sequence sample of size 1600 over-estimates the first few largest eigenvalues in modulus but shrink the rest to have very small modulus. In contrast, Lyap with the true Q preserves



(a) The pendulum (b) Normalized error (c) Cosine score

Figure 8: Results on the pendulum video data

the spectrum much better. We may thus need an even better covariance estimate for the sparse model.

4.4.2 Video Data

We test our methods on the video sequence of a periodically swinging pendulum used in Section 3.4. The period is about 23 frames, and one frame is given in Figure 8(a). We take the second-level Gaussian pyramids of the images to reduce the size to 9-by-11, then treat each reduced image as a 99-dimensional vector.

We compare the proposed method (28) with the ridge regression for two lengths of the training sequence: $T \in \{20, 50\}$, and treat the last 50 frames as the testing sequence. For both methods, we split the training sequence into two halves and use the second half as a validation sequence. For the proposed method, we simulate a non-sequence sample by randomly choosing 300 frames from between the $(T+1)$-st frame and the 450-th frame without replacement. We repeat this 10 times.

The testing normalized errors and cosine scores of both methods are given in Figures 8(b) and 8(c). For the proposed method, we report the mean performance measures over the 10 simulated non-sequence samples with standard deviation. When $T = 20$, which is less than but close to the period, the proposed method outperforms ridge regression very significantly. However, when we slightly increase T to 50, the difference between the two methods vanishes, even though there is still much room for improvement as indicated by the result of our model sanity check before. This may be due to our use of dependent data as the non-sequence sample, or simply insufficient non-sequence data.

5. Future Directions

We list below some interesting topics we plan to pursue.

5.1 Learning Dynamic Models with Hidden States

As mentioned earlier, a common motivation for learning dynamic models with hidden states is to deal with high-dimensional observations resulting from low-dimensional dynamics. As another strong motivation, it arises in quite a few applications that a history of observations may be compressed into more succinct representations because of temporal continuity or regularity of the underlying dynamics, an assumption made by many approaches for

learning dynamic models. A well-known example is the use of the *Hankel matrix* in system identification (Ljung, 1999), where observations from a short segment of history are stacked together to form a matrix, whose most significant linear subspace then serves to estimate the hidden state space. Other examples are the Kalman filter and smoother (Kalman, 1960; Ghahramani and Hinton, 1996), which maintain a probabilistic representation of the hidden states and update each hidden state representation based on observations and temporally adjacent hidden states.

To exploit non-sequence data for this task, we plan to investigate two possibilities. One is to incorporate into our EM-based methods (Section 3.1) components for learning hidden states, and the resulting algorithm may take the form of a combination of our EM algorithm and the parameter learning algorithm presented by Ghahramani and Hinton (1996). The other is to combine the temporal smoothing idea (Section 3.3) with manifold learning algorithms, most of which try to find a nonlinear, dimension-reducing mapping that, in addition to maximizing some pre-defined notion of good mappings, preserves the *spatial distances* in the original space. In dynamic model learning, however, we may prefer a low-dimensional mapping that preserves foremost temporal adjacency to one based simply on spatial proximity. We can thus take the output by our temporal smoothing algorithm as a temporal proximity measure, and feed it to existing proximity based manifold learning algorithms, such as Locally Linear Embedding (Roweis and Saul, 2000), Maximum Variance Unfolding (Weinberger et al., 2004), Isomap (Tenenbaum et al., 2000), Laplacian Eigenmap Belkin and Niyogi (2003), etc.

5.2 Learning Block-structured VAR Models

In some applications involving high-dimensional dynamic models, it might be sensible to assume that variables can be partitioned into disjoint subsets, and each subset of variables evolve independently of other subsets. In some cases the partition is available, but in many more it is not. Moreover, sometimes it might be preferable to learn such a structure from the data rather than relying on prior knowledge or belief.

Under the assumption of a VAR model, learning a partition of the variables is the same as learning a block-structured transition matrix with an unknown block structure. To the best of our knowledge, this problem, even in the presence of sequence data, has not been studied before. When only non-sequence data are available and we assume they are independent samples drawn from the stationary distribution of some block-structured VAR model, the Lyapunov equation (24) implies that, when the conditions for a unique solution are satisfied, the covariance Q of the stationary distribution and the transition matrix A share the same block-diagonal structure, suggesting that we may look for the block structure in a covariance estimate $\widehat{Q}$.

We consider two ways of approaching this problem. One the one hand, Marlin et al. (2009) proposed a prior and a variational Bayes procedure for learning an inverse covariance matrix along with its block structure under Gaussian distributions. Their method is readily applicable to our task here if we assume Gaussian noise for the VAR model. In addition, we are able to modify their method for learning a block-structured transition matrix when sequence data are available. However, these variational Bayes methods cannot guarantee the quality of the block structure they find.

One the other hand, a simple two-stage procedure is as follows: first obtain some covariance estimate from the samples, and then cluster the variables using the covariance estimate as pairwise similarities. Among the many clustering algorithms, we find Spectral Clustering particularly appealing because it has been proved recently by Balakrishnan et al. (2011) as being able to, with high probability, perfectly recover underlying clusters from an input similarity matrix that satisfies certain structural and noise assumptions. Building on their analysis, we hope to show rigorous guarantees on such a two-stage procedure, particularly in terms of structural estimation consistency and sample complexity.

5.3 Analyzing and Extending the Lyapunov Regularization Approach

As mentioned in Section 4.4.1, we are not clear about the impact of the covariance estimation quality on the Lyapunov regularized estimator. We hope to rigorously study their relationship, possibly through some type of stability or perturbation analysis. We also like to explore the possibility of learning state space models using similar regularization, such as the Riccati equations combined with non-sequence data.

5.4 Temporally-smooth Ordering through Integer Linear Programming

In Section 3.3 we apply relaxation and heuristics to the temporal smoothing problem, mainly due to lack of tools for dealing with the quadratic objective (17) under the combinatorial constraint (18). However, we observe that the smoothness measure (16) can actually be written as a linear function of parameters involving *triples* of data points. More precisely, let $Z \in \{0,1\}^{n\times n\times n}$ now denote a three-dimensional array of indicators such that $Z_{ijk} = Z_{kji}\ \forall i \neq j \neq k$ and $Z_{iii} = Z_{iij} = Z_{jii} = 0\ \forall i \neq j$. We interpret Z as follows: $Z_{ijk} = 1$ if and only if the two edges (i,j) and (j,k) are present in the graph. Let $W_{ijk} = W_{kji}$ denote the weight on the triple (i,j,k). Then the temporal smoothing problem is equivalent as

$$\begin{aligned} \min_{Z} \quad & \sum_{i,j,k} Z_{ijk} W_{ijk} \\ \text{s.t.} \quad & Z \text{ represents a Hamiltonian path on the data graph,} \\ & W_{ijk} = \|\mathbf{x}_i - 2\mathbf{x}_j + \mathbf{x}_k\|^2. \end{aligned} \tag{40}$$

It is easy to reduce the TSP to the above problem by setting $W_{ijk} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 + \|\mathbf{x}_j - \mathbf{x}_k\|^2$, so (40) is NP hard. However, the promising practical performance of Concorde, an exact TSP solver by Applegate et al., gives some hope that (40) could be solved within a reasonable running time in practice. As a standard paradigm in integer linear programming, we may start from identifying good linear relaxations of the Hamiltonian-path constraint, and then apply branch-and-bound or cutting-plane algorithms.

5.5 Analyzing Real Data

Our first application is learning dynamic models for cell cycle from cell images. We plan to work on the two data sets analyzed by Buck et al. (2009). The first is a single time-series images of HeLa cells originally collected by Zhou et al. (2009). There are a total of 149 sequences of cell nuclei, and 53 of them span all of the 100 time frames while the others are born at later time frames as a result of cell division. Each cell nucleus

is represented by a 49-dimensional feature vector extracted from the raw image by Buck et al. (2009). One experiment setting we have been deploying is to leave out one sequence at a time as testing data, treat all or part of the other data as non-sequence training data for our proposed methods. The available temporal information allow us to compare the proposed methods against standard methods in terms of predictive performance, and preliminary results suggest that the proposed methods of exploiting non-sequence data are very competitive with standard methods. The second data set consists of un-synchronized NIH 3T3 cell images collected by García Osuna et al. (2007). Details of the image processing pipeline, including segmentation and feature extraction, are as described by Buck et al. (2009), and the final data we analyze are 62-dimensional feature vectors of more than 60,000 cell nuclei. No temporal information are available in this data set, so we are still thinking about ways to evaluate the proposed methods. One possibility is to organize the data into some ordered structure based on learnt models, and ask domain experts to evaluate that structure.

Our second application is learning dynamic models for voice deterioration of people with Parkinson's Disease (PD). Tsanas et al. (2010) collected weekly voice recordings of 42 subjects over a six-months period, and extracted features that measure seriousness of dysphonia. In addition to the voice features, they also recorded demographic information about the subjects, such as age and gender. In our experiments, we will again include the leave-one-out setting, separating data from one subject at a time as testing data, but will also consider grouping subjects based on demographic characteristics and analyzing each group separately to avoid possible confounding factors.

5.6 Theoretical Analysis of Learning Dynamics from Non-sequence Data

In Section 3.1 we point out some properties of dynamic models that cannot be learnt accurately from non-sequence data, though some of them, such as the direction of time and the speed, can sometimes be made certain with domain knowledge. More serious ambiguities arise in learning a stable VAR model from only non-sequence samples drawn from its stationary distribution, since a whole lot of different VAR models, as suggested by the Lyapunov equation (24), explain the data equally well. We would like to investigate more formally such issues, characterizing how much and under what conditions on dynamic models and/or data-generating mechanisms we can possibly learn from non-sequence data.

6. Tentative Schedule

Our tentative schedule is as follows.

- Fall 2011
  Analyzing real data
  Learning block-structured VAR models
- Spring 2012
  Analyzing real data
  Learning block-structured VAR models
  Learning dynamic models with hidden states

- Fall 2012
  Analyzing and extending the Lyapunov regularization approach
  Temporally-smooth ordering through ILP

- Spring 2013
  Wrapping up and thesis writing

We will be pursuing theoretical analysis of learning dynamics from non-sequence data through out the rest of our thesis work since it is related to all other topics.

References

Pieter Abbeel and Andrew Y. Ng. Learning first order Markov models for control. In *Advances in Neural Information Processing Systems 17*, 2005.

P.J. Antsaklis and A.N. Michel. *Linear systems*. Birkhauser, 2005.

David Applegate, Ribert Bixby, Vašek Chvátal, and William Cook. Concorde TSP solver. URL `http://www.tsp.gatech.edu/concorde/index.html`.

Sivaraman Balakrishnan, Min Xu, Akshay Krishnamurthy, and Aarti Singh. Noise thresholds for spectral clustering. In *Advances in Neural Information Processing Systems 25*. To appear, 2011.

R. Barzilay and N. Elhadad. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55, 2002.

Matthew J. Beal, Zoubin Ghahramani, and Carl Edward Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*, pages 577–584, 2002.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA 02178-9998, second edition, 1999.

T.E. Buck, A. Rao, L.P. Coelho, M.H. Fuhrman, J.W. Jarvik, P.B. Berget, and R.F. Murphy. Cell cycle dependence of protein subcellular location inferred from static, asynchronous images. In *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1016–1019. IEEE, 2009.

P. M. Camerini, L. Fratta, and F. Maffioli. A note on finding optimum branchings. *Networks*, 9:309–312, 1979.

J.C. Chen, W.T. Chu, J.H. Kuo, C.Y. Weng, and J.L. Wu. Tiling slideshow. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 25–34. ACM, 2006.

Yilun Chen, Ami Wiesel, Yonina C. Eldar, and Alfred O. Hero. Shrinkage algorithms for mmse covariance estimation. *IEEE Transactions on Signal Processing*, 58:5016–5029, 2010a.

Yilun Chen, Ami Wiesel, and Alfred O. Hero. Robust shrinkage estimation of high-dimensional covariance matrices. Technical report, arXiv:1009.5331v1 [stat.ME], September 2010b.

P. Deshpande, R. Barzilay, and D.R. Karger. Randomized decoding for selection-and-ordering problems. In *Proceedings of NAACL HLT*, pages 444–451, 2007.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, pages 272–279, 2008.

E. García Osuna, J. Hua, N.W. Bateman, T. Zhao, P.B. Berget, and R.F. Murphy. Large-scale automated analysis of location patterns in randomly tagged 3t3 cells. *Annals of biomedical engineering*, 35(6):1081–1087, 2007.

Z. Ghahramani and G.E. Hinton. Parameter estimation for linear dynamical systems. *University of Toronto technical report CRG-TR-96-2*, 6, 1996.

Zoubin Ghahramani. Learning dynamic bayesian networks. *Lecture Notes in Computer Science*, 1387:168–197, 1998a.

Zoubin Ghahramani. Variational learning for switching state-space models. *Neural Computation*, 12:963–996, 1998b.

Joachim Giesen. Curve reconstruction in arbitrary dimension and the traveling salesman problem. *Proceedings of the 8th International Conference on Discrete Geometry for Computational Imagery, Lecture Notes in Computer Science*, 1568:164–176, 1999.

Anupam Gupta and Ziv Bar-Joseph. Extracting dynamics from static cancer expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5:172–182, 2008.

J.D. Hamilton. *Time series analysis*. Princeton Univ Pr, 1994.

Trevor Hastie and Werner Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84:502–516, 1989.

Robert Hodrick and Edward C. Prescott. Postwar U.S. business cycles: An empircal investigation. *Journal of Money, Credit, and Banking*, 29:1–16, 1997.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden Markov models. In *Proceedings of the Twenty-Second Annual Conference on Learning Theory*, 2009.

X.S. Hua, L. Lu, and H.J. Zhang. Automatically converting photographic series into video. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 708–715. ACM, 2004.

Tzu-Kuo Huang and Jeff Schneider. Learning linear dynamical systems without sequence information. In *Proceedings of the 26th International Conference on Machine Learning*, pages 425–432, 2009.

Tzu-Kuo Huang and Jeff Schneider. Learning auto-regressive models from sequence and non-sequence data. In *Advances in Neural Information Processing Systems 25*. To appear, 2011.

Tzu-Kuo Huang and Jeff Schneider. Learning dynamic models from non-sequenced data. Data Analysis Project Report, Machine Learning Department, Carnegie Mellon University, 2010. URL `http://www.ml.cmu.edu/research/dap-papers/dap_huang.pdf`.

Tzu-Kuo Huang, Le Song, and Jeff Schneider. Learning nonlinear dynamic models from non-sequenced data. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

A. Katok and B. Hasselblatt. *Introduction to the modern theory of dynamical systems*, volume 54. Cambridge Univ Pr, 1996.

Olivier Ledoit and Michael Wolf. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of Empirical Finance*, 10: 603–621, 2003.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88:365–411, 2004.

C. E. V. Lesser. A simple method of trend construction. *Journal of the Royal Statistical Society, Series B*, 23:91–107, 1961.

S. Li. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics and Statistics*, 4(1):66–70, 2011.

Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, New Jersey, second edition, 1999.

Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, 1963.

Paul M. Magwene, Paul Lizardi, and Junhyong Kim. Reconstructing the temporal ordering of biological samples using microarray data. *Bioinformatics*, 19:842–850, 2003.

Benjamin M. Marlin, Mark Schmidt, and Kevin P. Murphy. Group sparse priors for covariance estimation. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI 2009)*, Montreal, Canada, 2009.

Kevin Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California at Berkeley, 2002.

D. Noll, O. Prot, and A. Rondepierre. A proximity control algorithm to minimize nonsmooth and nonconvex functions. *Pacific Journal of Optimization*, 4(3):569–602, 2008.

Jonas Peters, Dominik Janzing, Arthur Gretton, and Bernhard Schölkopf. Detecting the direction of causal time series. In *Proceedings of the 26th International Conference on Machine Learning*, pages 801–808, 2009.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323, 2000.

Juliane Schäfer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4, 2005.

D. Shahaf and C. Guestrin. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632. ACM, 2010.

Daniel Sheldon, M.A. Saleh Elmohamed, and Dexter Kozen. Collective inference on markov models for modeling bird migration. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1321–1328. MIT Press, Cambridge, MA, 2008.

Sajid M. Siddiqi, Byron Boots, and Geoffrey J. Gordon. Reduced-rank hidden Markov models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

Alex J. Smola, Sebastian Mika, Bernhard Schölkopf, and Robert C. Williamson. Regularized principal manifolds. *Journal of Machine Learning Research*, 1:179–209, 2001.

Le Song, Byron Boots, Sajid Siddiqi, Geoffrey Gordon, and Alex Smola. Hilbert space embeddings of hidden Markov models. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning*, 2010.

Charles Stein. Estimation of a covariance matrix. In *Rietz Lecture, 39th Annual Meeting, Atlanta, GA*, 1975.

Robert E. Tarjan. Finding optimum branchings. *Networks*, 7:25–35, 1977.

J.B. Tenenbaum, V. Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319, 2000.

A. Tsanas, M.A. Little, P.E. McSharry, and L.O. Ramig. Accurate telemonitoring of parkinson's disease progression by non-invasive speech tests. *IEE Transactions on Biomedical Engineering*, 57(4):884–893, 2010.

Benjamin P. Tu, Andrzej Kudlicki, Maga Rowicka, and Steven L. McKnight. Logic of the yeast metabolic cycle: Temporal compartmentalization of cellular processes. *Science*, 310 (5751):1152–1158, 2005.

Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the 21st International Conference on Machine Learning*, pages 839–846, 2004.

Ruoyong Yang and James O. Berger. Estimation of a covariance matrix using the reference prior. *Annals of Statistics*, 22:1195–1211, 1994.

X. Zhou, F. Li, J. Yan, and S.T.C. Wong. A novel cell segmentation method and cell phase identification using markov model. *Information Technology in Biomedicine, IEEE Transactions on*, 13(2):152–157, 2009.