Time Series Clustering: Complex is Simpler!

Lei Li LEILI@CS.CMU.EDU
B. Aditya Prakash BADITYAP@CS.CMU.EDU
Computer Science Department, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213 USA

Abstract

Given a motion capture sequence, how to identify the category of the motion? Classifying human motions is a critical task in motion editing and synthesizing, for which manual labeling is clearly inefficient for large databases. Here we study the general problem of time series clustering. We propose a novel method of clustering time series that can (a) learn joint temporal dynamics in the data; (b) handle time lags; and (c) produce interpretable features. We achieve this by developing complex-valued linear dynamical systems (CLDS), which include real-valued Kalman filters as a special case; our advantage is that the transition matrix is simpler (just diagonal), and the transmission one easier to interpret. We then present ComplexFit, a novel EM algorithm to learn the parameters for the general model and its special case for clustering. Our approach produces significant improvement in clustering quality, 1.5 to 5 times better than well-known competitors on real motion capture sequences.

1. Introduction

Motion capture is a useful technology for generating realistic human motions, and is used extensively in computer games, movies and quality of life research (Lee & Shin, 1999; Safonova et al., 2003; Kagami et al., 2003). However, automatically analyzing (e.g. segmentation and labeling) such a large set of motion sequences is a challenging task. This paper is motivated by the application of clustering motion capture sequences (corresponding to different marker positions), an important step towards understanding human motion, but our proposed method is a general one and applies to other time series as well.

Clustering algorithms often rely on effective features extracted from data. Some most popular approaches include using the dynamic time warping (DTW) distance among sequences (Gunopulos & Das, 2001), using Principal Component Analysis (PCA) (Ding & He, 2004) and using Discrete Fourier Transform (DFT) coefficients. But unfortunately, directly applying traditional clustering algorithms to the features may not lead to appealing results. This is largely due to *two* distinct characteristics of time series data, (a) temporal dynamics; and (b) time shifts (lags). Differing from the conventional view of data as points in high dimensional space, time sequences encode temporal dynamics along the time ticks. Such dynamics often imply the grouping of those sequences in many real cases. For example, walking, running, dancing, and jumping motions are characterized by particular movements of human body, which result in different dynamics among the sequences. Hence by identifying the evolving temporal components, we can find the clusters of sequences with similar dynamics. As mentioned above, another often overlooked characteristic is time shift. For example, two walking motions may start from different footsteps, resulting in a lag among the sequences. Traditional methods like k-means with PCA features can not handle such lags in sequences, yielding poor clustering results. On the other hand, DTW, while handling lags, misses joint dynamics - thus sequences having the same underlying process but slightly different parameters (e.g. walking veering left vs. walking veering right) will have large DTW distances.

Hence we want the following main properties in any clustering algorithm for time series:

P1 It should be able to identify joint dynamics across the sequences;
P2 It should be able to eliminate lags (time shifts) across sequences;
P3 The features generated should be interpretable.

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

As we show later, our proposed method achieves all of the above characteristics, while other traditional methods miss out on one or more of these (more in Section 5). The main idea is to use *complex-valued* linear dynamical system (CLDS), which leads to several advantages: we can afford to have a diagonal transition matrix, which is simpler and faster to estimate; the resulting hidden variables are easy to interpret; and we meet all the design goals, including lag-invariance.

Specifically, the contributions of this paper are:

1. *Design of CLDS*: We develop complex-valued linear dynamical systems (CLDS), which includes traditional real-valued Kalman Filters as special cases. We then provide a novel complex valued EM algorithm, Complex-Fit, to learn the model parameters from the data.
2. *Application to Clustering*: We also use a special formulation of CLDS for time series clustering by imposing a restricted form of the transition dynamics corresponding to frequencies, without losing any expressiveness. Such an approach enhances the interpretability as well. Our clustering method then uses the participation weight (energy) of the hidden variables as features, thus eliminating lags. Hence it satisfies P1, P2 and P3 mentioned before.
3. *Validation*: Finally, we evaluate our algorithm on real motion capture data. Our proposed method is able to achieve best clustering results, comparing against several other popular time series clustering methods.

In addition, our proposed CLDS includes as special cases several popular, powerful methods like PCA, DFT and AR.

In the following sections, we will first present several pieces of the related models and techniques in time series clustering. We will also briefly introduce complex normal distributions and a few useful properties, and then present our CLDS and its learning algorithm, along with its application in time series clustering.

2. Background and Related Work

This section briefly introduces the background and related work for time-series clustering. Many algorithms have been proposed for time series classification, including decision trees (Rodriguez & Alonso, 2004), neural networks (Nanopoulos et al., 2001), Bayesian classifiers, SVM (Wu & Chang, 2004), etc. Among the most popular features for sequential data are DTW, PCA, LDS and DFT; we briefly describe these next. We will elaborate on the shortcomings and relationship of these methods to our proposed method later in Section 5.

DTW The typical distance function used for clustering is the time warping distance, also known as Dynamic Time Warping (DTW) (e.g., see the tutorial (Gunopulos & Das, 2001)). The linear-time constrained versions of DTW (Itakura parallelogram, Sakoe-Chiba band) have been studied in (Keogh, 2002; Fu et al., 2005). In spite of great progress on speeding up DTW, it is still expensive to compute (Xi et al., 2006), its plain version being quadratic on the length of sequences, and typically can not handle slight variations in the underlying generative dynamics.

PCA Principal Component Analysis (PCA) is the textbook method for dimensionality reduction, by spotting redundancies and (linear) correlations among the given sequences. Technically, it gives the optimal low rank approximation for the data matrix $\boldsymbol{X}$. Singular value decomposition (SVD) is the typical method to compute PCA. For a data matrix $\boldsymbol{X}$ (assume $\boldsymbol{X}$ is zero-centered), SVD computes the decomposition $\boldsymbol{X} = \boldsymbol{U} \cdot \boldsymbol{S} \cdot \boldsymbol{V}^T$ where both $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices, $\boldsymbol{S}$ is a diagonal matrix with singular values on the diagonal, and $\boldsymbol{U} \cdot \boldsymbol{S}$ can serve as features. PCA is particularly effective for Gaussian distributed data (Tipping & Bishop, 1999). However, often the low dimensional projections are hard to interpret. Moreover, PCA can not capture time-evolving and time-shifted patterns (since it is designed to *not* care about the ordering of the rows or the columns).

DFT The T-point Discrete Fourier Transform (DFT) of sequence $(x_0, \ldots, x_{T-1})$ is a set of T complex numbers c_k, given by the formula $c_k = \sum_{t=0}^{N-1} x_t e^{-\frac{2\pi i}{N} kt}$ $(k = 0, \ldots, N-1)$ where $i = \sqrt{-1}$ is imaginary unit. The c_k numbers are also referred to as the *spectrum* of the input sequence. DFT, as well as wavelet transforms, is powerful in spotting periodicities in a single sequence, with numerous uses in signal, voice, and image processing. However, by missing out the dynamics and having a fixed basis of frequencies, it can not find arbitrary and near-by frequencies.

LDS and Augmented Kalman Filters Linear Dynamical Systems (LDS), also known as Kalman filters, have been used previously to model multi-dimensional continuous valued time series. There exist methods based on Kalman filters for clustering time series data (Buzan et al., 2004; Li et al., 2010). The classical Kalman filter assumes the observed data sequences $(\boldsymbol{x}_n)$ are generated from the a series of hidden variables $(\boldsymbol{z}_n)$ with a linear projection matrix $\boldsymbol{C}$, and the hidden variables are evolving over time with linear tran-

sition matrix $\boldsymbol{A}$, so that next time tick only depends on the previous time tick as in Markov chains. All noises ($\boldsymbol{\omega}$'s and $\boldsymbol{\epsilon}$'s) arising from the process are modeled as independent Gaussian noises with covariances $\boldsymbol{Q}_0$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ respectively. Given the observation series, there exist algorithms for estimating hidden variables (Kalman, 1960) and EM algorithms for learning the model parameters (Shumway & Stoffer, 1982; Ghahramani & Hinton, 1996). Apart from hard-to-interpret model parameters, LDS can not handle time lags well (see discussion Section 5). Our approach is also remotely related to several variations of complex Kalman filters for signal processing (**??**). These approaches are based on the widely-linear filters, which explicitly regress on the conjugate of the state variables in addition to the traditional Kalman filters. The common, main difference is that they all focus on forecasting while our goal is time series clustering. Therefore we design the model very carefully to achieve good clustering (e.g. the use of diagonal transition matrix).

3. Preliminary: Complex Linear Gaussian Distributions

This section introduces the basic notations of complex valued normal distribution and its related extension, linear Gaussian distributions (or linear normal distributions), which are building blocks of our proposed method. We will give a concise summary of the joint, the marginal and the posterior distributions as well. For a full description of the normal distributions of complex variables, we refer readers to (Goodman, 1963; Andersen et al., 1995).

Definition 1 (Multivariate Complex Normal Distribution) *Let $\boldsymbol{x}$ be a vector of complex random variables, with dimensionality of m. $\boldsymbol{x}$ follows a multivariate complex normal distribution, denoted as $\boldsymbol{x} \sim \mathcal{CN}(\mu, H)$, if its p.d.f is*

$$p(\boldsymbol{x}) = \pi^{-m}|H|^{-1}\exp(-(\boldsymbol{x}-\mu)^* H^{-1}(\boldsymbol{x}-\mu))$$

where H is a positive semi-definite and hermitian matrix (Andersen et al., 1995). The mean and variance are given by $\mathbb{E}[\boldsymbol{x}] = \mu$ and $Var(\boldsymbol{x}) = H$.

All the following lemmas are heavily used in our derivation to obtain the EM algorithm for CLDS.

Lemma 1 (Linear Gaussian distributions) *If $\boldsymbol{x}$ and $\boldsymbol{y}$ random vectors from the distributions $\boldsymbol{x} \sim \mathcal{CN}(\boldsymbol{\mu}, \boldsymbol{H})$ and $\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{CN}(\boldsymbol{A}\cdot\boldsymbol{b}, \boldsymbol{V})$, it follows that $\boldsymbol{z} = \begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{pmatrix}$ will follow a complex normal distribution with the mean and covariance given by:*

$$\mathbb{E}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{pmatrix} = \begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{A}\cdot\boldsymbol{\mu}+\boldsymbol{b}\end{pmatrix}$$
$$\mathrm{Var}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{pmatrix} = \begin{pmatrix}\boldsymbol{H} & \boldsymbol{H}\cdot\boldsymbol{A}^*\\ \boldsymbol{A}\cdot\boldsymbol{H} & \boldsymbol{V}+\boldsymbol{A}\cdot\boldsymbol{H}\cdot\boldsymbol{A}^*\end{pmatrix}$$

Lemma 2 (Marginal distribution) *Under the same assumption as Lemma 1, it follows that*

$$\boldsymbol{y} \sim \mathcal{CN}(\boldsymbol{A}\cdot\boldsymbol{\mu}+\boldsymbol{b}, \boldsymbol{V}+\boldsymbol{A}\cdot\boldsymbol{H}\cdot\boldsymbol{A}^*)$$

Lemma 3 (Posterior distribution) *Under the same assumption as Lemma 1, the posterior distribution of $\boldsymbol{y}|\boldsymbol{x}$ is complex normal, and its mean $\boldsymbol{\mu}_{\boldsymbol{y}|\boldsymbol{x}}$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}}$ given by,*

$$\boldsymbol{\mu}_{\boldsymbol{y}|\boldsymbol{x}} = \boldsymbol{\mu} + \boldsymbol{K}\cdot(\boldsymbol{y}-\boldsymbol{b}-\boldsymbol{A}\cdot\boldsymbol{\mu})$$
$$\boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}} = (I-\boldsymbol{K}\cdot\boldsymbol{A})\cdot\boldsymbol{H}$$

where the "gain" matrix $\boldsymbol{K} = \boldsymbol{H}\cdot\boldsymbol{A}^*\cdot(\boldsymbol{V}+\boldsymbol{A}\cdot\boldsymbol{H}\cdot\boldsymbol{A}^*)^{-1}$.

A nice property of complex linear Gaussian distribution is "rotation invariance". In the simplest form, the marginal will remain the same for a family of linear transformation, i.e. $y = ax \sim P(0, |a|^2)$ iff $x \sim \mathcal{CN}(0,1)$. In this case, ax and $|a|x$ have the same distribution.

4. Complex Linear Dynamical Systems

In this section we describe the formulation of complex-valued linear dynamical systems and its special case for clustering.

The complex linear dynamical systems (CLDS) is defined with the following equations.

$$\boldsymbol{z}_1 = \boldsymbol{\mu}_0 + \boldsymbol{w}_1$$
$$\boldsymbol{z}_{n+1} = \boldsymbol{A}\cdot\boldsymbol{z}_n + \boldsymbol{w}_{n+1}$$
$$\boldsymbol{x}_n = \boldsymbol{C}\cdot\boldsymbol{z}_n + \boldsymbol{v}_n$$

where the noise vectors follow complex normal distribution. $\boldsymbol{w}_1 \sim \mathcal{CN}(0, \boldsymbol{Q}_0)$, $\boldsymbol{w}_i \sim \mathcal{CN}(0, \boldsymbol{Q})$, and $\boldsymbol{v}_j \sim \mathcal{CN}(0, \boldsymbol{R})$. Note that unlike Kalman filters, CLDS allows complex values in the parameters, with the restriction that $\boldsymbol{Q}_0$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ should be Hermitian and positive definite. Figure 1 shows the graphical model. It can be viewed as consecutive linear Gaussian distributions on the hidden variable $\boldsymbol{z}$'s and observation $\boldsymbol{x}$.

The problem of learning is to estimate the best fit parameters $\theta = \{\mu_0, \boldsymbol{Q}_0, \boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{C}, \boldsymbol{R}\}$, giving the observation sequence $\boldsymbol{x}_1 \ldots \boldsymbol{x}_N$. We develop Complex-Fit,

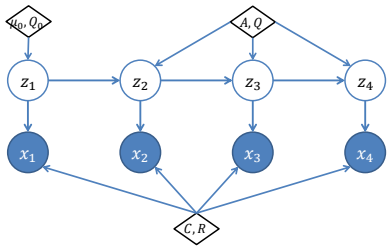


Figure 1. Graphical Model for CLDS. $\boldsymbol{x}$ are real valued observations and $\boldsymbol{z}$ are complex hidden variables. Arrows denote linear Gaussian distributions.

a novel complex valued expectation-maximization algorithm towards a maximum likelihood fitting.

The expected negative-loglikelihood of the model is

$$\begin{aligned}\mathcal{L}(\theta) &= \mathbb{E}_{\boldsymbol{Z}|\boldsymbol{X}}[-\log P(\boldsymbol{X},\boldsymbol{Z}|\theta)] \\ &= \log|\boldsymbol{Q}_0| + \mathbb{E}[(\boldsymbol{z}_1-\boldsymbol{\mu}_0)^*\boldsymbol{Q}_0^{-1}(\boldsymbol{z}_1-\boldsymbol{\mu}_0)] \\ &+ \mathbb{E}[\sum_{n=1}^{N-1}(\boldsymbol{z}-\boldsymbol{A}\cdot\boldsymbol{z}_n)^*\cdot Q^{-1}\cdot(\boldsymbol{z}_{n+1}-\boldsymbol{A}\cdot\boldsymbol{z}_n)] \\ &+ \mathbb{E}[\sum_{n=1}^{N}(\boldsymbol{x}_n-\boldsymbol{C}\cdot\boldsymbol{z}_n)^*\cdot R^{-1}\cdot(\boldsymbol{x}_n-\boldsymbol{C}\cdot\boldsymbol{z}_n)] \\ &+ (N-1)\log|Q| + N\log|R| \end{aligned} \tag{1}$$

where the expectation $\mathbb{E}[]$ is over the posterior distribution of $\boldsymbol{Z}$ given $\boldsymbol{X}$.

Unlike traditional Kalman filters, the objective here is function of complex values, requiring nonstandard optimization in complex domain. We will first describe the M-step here. In the negative-loglikelihood, there were two sets of unknowns, the parameters and the posterior distribution. The overall idea of the Complex-Fit algorithm is to optimize over the parameter set θ as if we know the posterior and then estimate the posterior with current parameters. It then takes turns to obtain the optimal solution.

Complex-Fit M-step The M-step is derived by taking complex derivatives of the objective function and equating them to zeros. Unlike the real valued version, taking derivatives of complex functions should take extra care, since they are not always analytic or holomorphic. The above function (5) is not differentiable in classical setting since it does not satisfy Cauchy-Riemann condition(Mathews & Howell, 2006). However, if x and $\bar{x}$ are treated independently, we could obtain their generalized partial derivatives, as defined in (Brandwood, 1983; Hjorungnes & Gesbert, 2007). The optimal of the function $f(x)$ can be achieved when both partial derivatives of $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial \overline{x}}$ equal zero.

The solution minimizing $\mathcal{L}$ is given by

$$\frac{\partial}{\partial \boldsymbol{\mu}_0}\mathcal{L}=0 \qquad \frac{\partial}{\partial \overline{\boldsymbol{\mu}_0}}\mathcal{L}=0 \qquad \frac{\partial}{\partial \boldsymbol{Q}_0}\mathcal{L}=0 \qquad \frac{\partial}{\partial \overline{\boldsymbol{Q}_0}}\mathcal{L}=0$$

where

$$\frac{\partial}{\partial \boldsymbol{\mu}_0}\mathcal{L} = -(\mathbb{E}[z]_1-\boldsymbol{\mu}_0)^*\cdot\boldsymbol{Q}_0^{-1} \tag{2}$$

$$\frac{\partial}{\partial \overline{\boldsymbol{\mu}_0}}\mathcal{L} = -(\mathbb{E}[z]_1-\boldsymbol{\mu}_0)^T\cdot(\boldsymbol{Q}_0^{-1})^T \tag{3}$$

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{Q}_0}\mathcal{L} =& (\boldsymbol{Q}_0^T)^{-1} - (\boldsymbol{Q}_0^T)^{-1}\cdot\mathbb{E}[\overline{(\boldsymbol{z}_1-\boldsymbol{\mu}_0)}\cdot(\boldsymbol{z}_1-\boldsymbol{\mu}_0)^T] \\ &\cdot(\boldsymbol{Q}_0^T)^{-1}\end{aligned} \tag{4}$$

It follows that $\boldsymbol{\mu}_0=\mathbb{E}[\boldsymbol{z}_1]$ and $\boldsymbol{Q}_0=\mathbb{E}[\boldsymbol{z}_1\cdot\boldsymbol{z}_1^*]-\boldsymbol{\mu}_0\cdot\boldsymbol{\mu}_0^*$. Similarly, we obtain update rules for $\boldsymbol{A}$, $\boldsymbol{Q}$, $\boldsymbol{C}$ and $\boldsymbol{R}$, by taking partial derivatives, $\frac{\partial}{\partial \boldsymbol{A}}\mathcal{L}$, $\frac{\partial}{\partial \overline{\boldsymbol{A}}}\mathcal{L}$, $\frac{\partial}{\partial \boldsymbol{Q}}\mathcal{L}$, $\frac{\partial}{\partial \overline{\boldsymbol{Q}}}\mathcal{L}$, $\frac{\partial}{\partial \boldsymbol{C}}\mathcal{L}$, $\frac{\partial}{\partial \overline{\boldsymbol{C}}}\mathcal{L}$, $\frac{\partial}{\partial \boldsymbol{R}}\mathcal{L}$, and $\frac{\partial}{\partial \overline{\boldsymbol{R}}}\mathcal{L}$, and equating them to zeros. Here is the final update rules for each of these parameters.

$$\boldsymbol{A} = (\sum_{n=1}^{N-1}\mathbb{E}[\boldsymbol{z}_{n+1}\cdot\boldsymbol{z}_n^*])\cdot(\sum_{n=1}^{N-1}\mathbb{E}[\boldsymbol{z}_n\cdot\boldsymbol{z}_n^*])^{-1} \tag{5}$$

$$\begin{aligned}\boldsymbol{Q} =& \frac{1}{N-1}\sum_{n=1}^{N-1}\Big(\mathbb{E}[\boldsymbol{z}_{n+1}\cdot\boldsymbol{z}_{n+1}^*] - \mathbb{E}[\boldsymbol{z}_{n+1}\cdot\boldsymbol{z}_n^*]\cdot\boldsymbol{A}^* \\ &- \boldsymbol{A}\cdot\mathbb{E}[\boldsymbol{z}_n\cdot\boldsymbol{z}_{n+1}^*] + \boldsymbol{A}\cdot\mathbb{E}[\boldsymbol{z}_n\cdot\boldsymbol{z}_n^*]\cdot\boldsymbol{A}^*\Big)\end{aligned} \tag{6}$$

$$\boldsymbol{C} = (\sum_{n=1}^{N}\boldsymbol{x}_n\cdot\mathbb{E}[\boldsymbol{z}_n^*])\cdot(\sum_{n=1}^{N}\mathbb{E}[\boldsymbol{z}_n\cdot\boldsymbol{z}_n^*])^{-1} \tag{7}$$

$$\begin{aligned}\boldsymbol{R} =& \frac{1}{N}\sum_{n=1}^{N}\big(\boldsymbol{x}_n\cdot\boldsymbol{x}_n^* - \boldsymbol{x}_n\cdot\mathbb{E}[\boldsymbol{z}_n^*]\cdot\boldsymbol{C}^* - \boldsymbol{C}\cdot\mathbb{E}[\boldsymbol{z}_n]\cdot\boldsymbol{x}_n^* \\ &+ \boldsymbol{C}\cdot\mathbb{E}[\boldsymbol{z}_n\cdot\boldsymbol{z}_n^*]\cdot\boldsymbol{C}^*\big)\end{aligned} \tag{8}$$

Complex-Fit E-step The above M-step requires computation of the sufficient statistics on the posterior distribution of hidden variables $\boldsymbol{z}$. During the E-step, we will compute mean and covariance of the marginal and joint posterior distributions $P(\boldsymbol{z}_n|\boldsymbol{X})$ and $P(\boldsymbol{z}_n,\boldsymbol{z}_{n+1}|\boldsymbol{X})$. The E-step computes the posteriors in with the forward-backward sub steps (corresponding to Kalman filtering and smoothing in the traditional LDS). The forward step computes the partial posterior $\boldsymbol{z}_n|\boldsymbol{x}_1\ldots\boldsymbol{x}_n$, and the backward pass computes the full posterior distributions. We can show by induction that all these posteriors are complex normal distributions and the transition between them satisfying the condition of linear Gaussian distribution. Such facts will help us derive an algorithm to find the means and variances of those posterior distributions.

The forward step computes the partial posterior $\boldsymbol{z}_n|\boldsymbol{x}_1 \dots \boldsymbol{x}_n$ from the beginning $\boldsymbol{z}_1$ to the tail of the chain $\boldsymbol{z}_N$. By exploiting Markov properties and applying Lemma 1, Lemma 2 and Lemma 3 on posteriors $\boldsymbol{z}_n|\boldsymbol{x}_1 \dots \boldsymbol{x}_n$, we can show that $\boldsymbol{z}_n|\boldsymbol{x}_1 \dots \boldsymbol{x}_n \sim \mathcal{CN}(\boldsymbol{u}_n, \boldsymbol{U}_n)$, with following equations for computing $\boldsymbol{u}_n$ and $\boldsymbol{U}_n$ recursively,

$$\boldsymbol{u}_{n+1} = \boldsymbol{A} \cdot \boldsymbol{u}_n + \boldsymbol{K}_{n+1} \cdot (\boldsymbol{x}_{n+1} - \boldsymbol{C} \cdot \boldsymbol{A} \cdot \boldsymbol{u}_n) \quad (9)$$

$$\boldsymbol{U}_{n+1} = (I - \boldsymbol{K}_{n+1} \cdot \boldsymbol{C}) \cdot \boldsymbol{P}_{n+1} \quad (10)$$

and we define,

$$\boldsymbol{P}_{n+1} = \boldsymbol{A} \cdot \boldsymbol{U}_n \cdot \boldsymbol{A}^* + \boldsymbol{Q} \quad (11)$$

$$\boldsymbol{K}_{n+1} = \boldsymbol{P}_{n+1} \cdot \boldsymbol{C}^* \cdot (\boldsymbol{R} + \boldsymbol{C} \cdot \boldsymbol{P}_{n+1} \cdot \boldsymbol{C}^*)^{-1} \quad (12)$$

The initial step is given by $\boldsymbol{u}_1 = \boldsymbol{\mu}_0 + \boldsymbol{K}_1 \cdot (\boldsymbol{x}_1 - \boldsymbol{C} \cdot \boldsymbol{\mu}_0)$ and $\boldsymbol{U}_1 = (I - \boldsymbol{K}_1 \cdot \boldsymbol{C}) \cdot \boldsymbol{Q}_0$. $\boldsymbol{K}_1$ is the complex-valued "Kalman gain" matrix, $\boldsymbol{K}_1 = \boldsymbol{Q}_0 \cdot \boldsymbol{C}^* \cdot (\boldsymbol{R} + \boldsymbol{C} \cdot \boldsymbol{Q}_0 \cdot \boldsymbol{C}^*)^{-1}$.

The backward step computes the posterior $\boldsymbol{z}_n|\boldsymbol{x}_1 \dots \boldsymbol{x}_N$ from the tail $\boldsymbol{z}_N$ to the head of the chain $\boldsymbol{z}_1$. Again using the lemmas of complex linear Gaussian distributions, we can show $\boldsymbol{z}_n|\boldsymbol{x}_1 \dots \boldsymbol{x}_N \sim \mathcal{CN}(\boldsymbol{v}_n, \boldsymbol{V}_n)$, and compute the posterior means and variances through the following equations.

$$\boldsymbol{v}_n = \boldsymbol{u}_n + \boldsymbol{J}_{n+1} \cdot (\boldsymbol{v}_{n+1} - \boldsymbol{A} \cdot \boldsymbol{u}_n) \quad (13)$$

$$\boldsymbol{V}_n = \boldsymbol{U}_n + \boldsymbol{J}_{n+1} \cdot (\boldsymbol{V}_{n+1} - \boldsymbol{P}_{n+1}) \cdot \boldsymbol{J}_{n+1}^* \quad (14)$$

where $\boldsymbol{J}_{n+1} = \boldsymbol{U}_n \cdot \boldsymbol{A}^* \cdot (\boldsymbol{A} \cdot \boldsymbol{U}_n \cdot \boldsymbol{A}^* + \boldsymbol{Q})^{-1} = \boldsymbol{U}_n \cdot \boldsymbol{A}^* \cdot \boldsymbol{P}_{n+1}^{-1}$ Obviously, $\boldsymbol{v}_N = \boldsymbol{u}_N$ and $\boldsymbol{V}_N = \boldsymbol{U}_N$.

With a similar induction, from Lemma 1 we can compute the following sufficient statistics,

$$\mathbb{E}[\boldsymbol{z}_n \cdot \boldsymbol{z}_n^*] = \boldsymbol{V}_n + \boldsymbol{v}_n \cdot \boldsymbol{v}_n^* \quad (15)$$

$$\mathbb{E}[\boldsymbol{z}_n \cdot \boldsymbol{z}_{n+1}^*] = \boldsymbol{J}_{n+1} \cdot \boldsymbol{V}_{n+1} + \boldsymbol{v}_n \cdot \boldsymbol{v}_{n+1}^* \quad (16)$$

Special Case and CLDS Clustering In addition to the full model as described above, we consider a special case with diagonal transition matrix $\boldsymbol{A}$. The diagonal elements of $\boldsymbol{A}$ correspond to its eigenvalues, denoted as $\boldsymbol{a}$. The eigenvalues of the matrix will be similar to the frequencies in Fourier analysis. The justification of using diagonal matrix lies in the observation of the rotation invariance property in linear Gaussian distributions (Lemma 4). In simplest case, such rotation invariant matrix is diagonal.

Lemma 4 (Rotation invariance) *Assume* $\boldsymbol{x} \sim \mathcal{CN}(0, \boldsymbol{I})$ *and* $\boldsymbol{B} = \boldsymbol{A} \cdot \boldsymbol{V}$ *with unitary* $\boldsymbol{V}$.[1]*, it follows that* $\boldsymbol{A} \cdot \boldsymbol{x}$ *and* $\boldsymbol{B}\boldsymbol{x}$ *have exactly the same distribution. By abusing the definition of* $\sim$ *slightly, it can be written as* $\boldsymbol{A} \cdot \boldsymbol{x} \sim \boldsymbol{B}\boldsymbol{x} \sim \mathcal{CN}(0, \boldsymbol{A} \cdot \boldsymbol{A}^*)$.

To get the optimal solution in Eq. (5) with such a diagonal $\boldsymbol{A}$, we will use the definition of Hadamard product[2] and its related results. Let $\boldsymbol{a}$ be the diagonal elements of $\boldsymbol{A}$. Since $\boldsymbol{A}$ is diagonal, the difference will result in a rather different solution in partial derivatives. The conditions of optimal solutions are given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{a}} = \sum_{n=1}^{N-1} \mathbb{E}[(\boldsymbol{Q}^{-1} \cdot (\boldsymbol{z}_{n+1} - \boldsymbol{a} \circ \boldsymbol{z}_n)) \circ \overline{\boldsymbol{z}_n}]^* = 0$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{Q}} = (N-1)(\boldsymbol{Q}^T)^{-1} - (\boldsymbol{Q}^T)^{-1} \cdot (\sum_{n=1}^{N-1} \mathbb{E}[(\overline{\boldsymbol{z}_{n+1}} - \overline{\boldsymbol{a}} \circ \overline{\boldsymbol{z}_n}) \cdot (\boldsymbol{z}_{n+1} - \boldsymbol{a} \circ \boldsymbol{z}_n)^T]) \cdot (\boldsymbol{Q}^T)^{-1} = 0$$

To solve the above equations, we use the following iterative update rules.

$$\boldsymbol{a} = (\boldsymbol{Q}^{-1} \circ (\sum_{n=1}^{N-1} \mathbb{E}[\boldsymbol{z}_n \cdot \boldsymbol{z}_n^*])^T)^{-1} \cdot (\boldsymbol{Q}^{-1} \circ (\sum_{n=1}^{N-1} \mathbb{E}[\boldsymbol{z}_{n+1} \cdot \boldsymbol{z}_n^*])^T) \cdot \boldsymbol{1} \quad (17)$$

$$\boldsymbol{Q} = \frac{1}{N-1} \sum_{n=1}^{N-1} \Big(\mathbb{E}[\boldsymbol{z}_{n+1} \cdot \boldsymbol{z}_{n+1}^*] - \mathbb{E}[\boldsymbol{z}_{n+1} \cdot (\boldsymbol{a} \circ \boldsymbol{z}_n)^*] - \mathbb{E}[(\boldsymbol{a} \circ \boldsymbol{z}_n) \cdot \boldsymbol{z}_{n+1}^*] + \mathbb{E}[(\boldsymbol{a} \circ \boldsymbol{z}_n) \cdot (\boldsymbol{a} \circ \boldsymbol{z}_n)^*] \Big) \quad (18)$$

Once we have the best estimate of such parameters using Complex-Fit(with diagonal transition matrix), the overall idea of CLDS clustering is essentially using the output matrix in CLDS as features, and then applying any off-the-shelf clustering algorithm (e.g. k-means clustering). In more detail, we take only the magnitude of $\boldsymbol{C}$ to eliminate the lags in the data, since its magnitude represents the energy or weight of participation of the *learned* hidden variables in the observations. In this sense, our method can also be used as a feature extraction tool in other applications such as signal compression.

5. Discussion

Relationship to (real-valued) Linear Dynamical Systems The graphical representation of our CLDS is similar to the real-valued linear dynamical systems (LDS, also known as Kalman filters), except

[1] A matrix $\boldsymbol{V}$ is unitary if $\boldsymbol{V} \cdot \boldsymbol{V}^* = \boldsymbol{V}^* \cdot \boldsymbol{V} = \boldsymbol{I}$

[2] $(\boldsymbol{A} \circ \boldsymbol{B})_{i,j} = \boldsymbol{A}_{i,j} \cdot \boldsymbol{B}_{i,j}$

that the conditional distribution changes to complex-valued normal distribution.

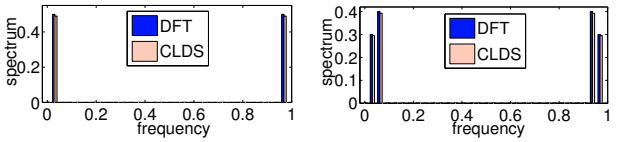
But due to this, there is a significant difference in the space of the optimal solutions. In LDS, such a space contains many essentially equivalent solutions. Consider a set of estimated parameters for LDS: it will yield equivalent parameters simply by exchanging the order in hidden variables and initial state (and correspondingly columns of $\boldsymbol{A}$ and $\boldsymbol{C}$). A generalization of this would be a proper "rotation" of the hidden space, by applying a linear transformation with a orthogonal matrix. Our approach actually tries to find a representative for such an equivalent family of solutions. In traditional Kalman filters, it is not always possible to get the most compact solution with real valued transition matrix, while in our model with the diagonal transition matrix, the solution is invariant in a proper sense.

Furthermore, LDS does not have a explicit notion of time shifts in its model, while in our method, it is already encoded in the phase of initial states and the output matrix $\boldsymbol{C}$. This is also confirmed by our experiments: LDS does not generate features helpful in clustering, while CLDS significantly improves that.

Relationship to Discrete Fourier Transform CLDS is closely related to Fourier analysis, since the eigenvalues of the transition matrix $\boldsymbol{A}$ essentially encode a set of base frequencies. In the special restricted case (used for clustering), the diagonal elements of $\boldsymbol{A}$ directly tell those frequencies. Hence, with proper construction, CLDS includes Discrete Fourier transform (DFT) as a special instance.

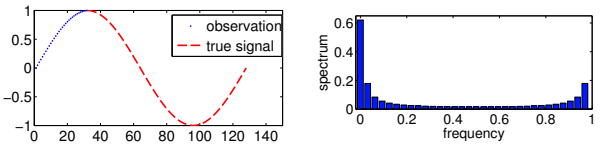
Consider one dimensional sequence $x_{1,\cdots,N}$: we can build a probabilistic version of DFT by fixing $\boldsymbol{\mu}_0 = \mathbf{1}$, and $\boldsymbol{A} = \text{diag}(\exp(\frac{2\pi i}{N}k)), k = 1, \ldots, N$. We conjecture that if we train such a model on the data, the estimated output matrix $\boldsymbol{C}$ will be equivalent to the Fourier coefficients from DFT. This is also confirmed by our experiments on synthetic signals. Figure 2 exhibits the spectrum of coefficients from DFT and the output matrix $\boldsymbol{C}$ from CLDSfor two signals. They almost perfectly match each other.

Compared to DFT, our proposed method clearly enjoys four benefits: (a) it allows dynamics corresponding to arbitrary frequency components, contrary to a fixed set of base frequencies as in DFT; (b) being an explicit probabilistic model allows a rich family of extension to other non Gaussian noises; (c) it has direct control over the model complexity and sparsity with the number of hidden variables, i.e. choosing a small number will result in forcing the approximation of the harmonics or frequencies in the data; (d) it can estimate harmonic components jointly present in multiple signals (but with small noise), while it is not straightforward to extend DFT to multiple sequences. For e.g., Figure 3 showcases the limitation of DFT on signals only observed for partial cycles: it fails to recognize the exact frequency component in the signal (non-integer multiple of the base frequency), while CLDS can almost perfectly identify the frequency components with two hidden variables.



(a) $x = \sin(\frac{2\pi t}{32})$ (b) $x = 0.6\sin(\frac{2\pi t}{32}) + 0.8\sin(\frac{2\pi t}{16})$

Figure 2. Spectrum of synthetic signals. Note CLDS can learn spectrums very close to DFT's, by fixing diagonal transition matrices corresponding to base frequencies.



(a) $x = \sin(\frac{2\pi t}{128})$ (b) Spectrum of DFT

Figure 3. Limitation of DFT (right) on a partially observed synthetic signal (left). Note DFT can not recover exact frequencies, while by setting hidden dimension to be two, CLDS's estimates are $\boldsymbol{a} = \{0.9991 \pm 0.0494i\}$, equivalent to frequencies of $\pm 1/127.19$, close to true signal.

Other related models Autoregression (AR) is another popular model for time series used for forecasting. CLDS also includes AR as a special case, which can be obtained by setting the output matrix $\boldsymbol{C}$ to be the identity matrix. Principal component analysis (PCA) can also be viewed as a special case of CLDS. By setting the transition matrix to be zeros, CLDS degenerates to Probabilistic PCA (Tipping & Bishop, 1999).

6. Experiments

We used two datasets (`MOCAPPOS` and `MOCAPANG`) from a public human motion capture database[3]. `MOCAPPOS` includes 49 motion sequences of marker positions in body local coordinates, each motion is labeled with ei-

[3]http://mocap.cs.cmu.edu/ subject #16 and #35

ther walking or running as annotated in the database. On the other hand, MOCAPANG includes 33 sequences of joint angles, 10 being walking motions and the rest running. While the original motion sequences have different lengths, we trim them with equal duration. Since there are multiple markers used in the motion capture, we only choose the one (e.g. right foot marker) that is most significant in telling human motions apart, suggested by domain experts. Alternatively, this can also be achieved through an additional feature selection process, which is not the focus of our paper.

We compare our method against several baselines:

PCA: As we mentioned in background, Principal component analysis is a textbook method to extract features from high dimensional data[4]. In this method, we follow a standard pipeline of clustering high dimensional data (Ding & He, 2004): first performing a dimensionality reduction on the data matrix by keeping k (=2) principal components, and then clustering on the PCA scores using k-means.

DFT: The second baseline is the Fourier method. This method first computes Fourier coefficients for each motion sequences using Discrete Fourier Transform. It then uses PCA to extract two features from the Fourier coefficients, and finally finds clusters again through k-means clustering on top of the DFT-PCA features.

DTW: The third method, dynamic time warping (DTW), is a popular method to calculate the minimal distance between pairs of sequences by allowing flexible shift in alignment (thus it would be fair competitor on time series with time lags). In this method, we compute all pairwise DTW distances and again use the k-means on top of them to find clusters.

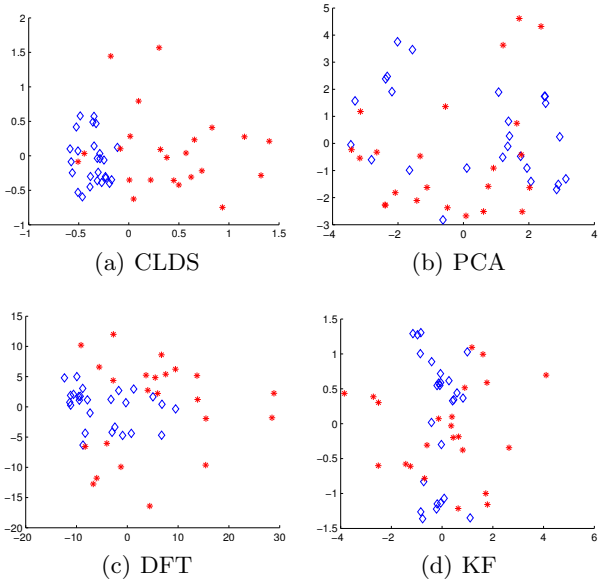
KF: Another baseline method is learning a Kalman filter or linear dynamical systems (LDS) from the data and using its output matrix as features in k-mean clustering. In this experiment, we tried a few values for the number of hidden variables and chose the one with best clustering performance (=8).

To evaluate the quality, we use the conditional entropy S of the true labeling with respect to the prediction, defined by the confusion matrix M: $S(M) = \sum_{i,j} \frac{M_{i,j}}{\sum_{k,l} M_{k,l}} \log \frac{\sum_k M_{i,k}}{M_{i,j}}$. The element $M_{i,j}$ corresponds to the number of sequences with true label j in cluster i. Intuitively, the conditional entropy S tells difference between the prediction and the actual, therefore a lower score indicates a better prediction and the best case is $S = 0$. In information theory, the conditional entropy corresponds to the additional information of the actual labels based on the prediction.

[4]Note: the dimensionality in PCA corresponds to the duration in time series. The dimensionality in time series usually refers to the number of sequences.

Table 1. Conditional entropies (S) of clustering methods on both datasets. Note a lower score corresponds to a better clustering, and in both cases our proposed method CLDS achieves the lowest scores 1.5 to 5 times better than others, yielding clusters most close to the true labels.

methods	MOCAPPOS S	MOCAPANG S
CLDS	**0.3786**	**0.1015**
PCA	0.6818	0.3635
DFT	0.6143	0.2538
DTW	0.5707	0.4229
KF	0.6749	0.5239



(a) CLDS (b) PCA (c) DFT (d) KF

Figure 4. Typical scatter plots: Top two features extracted by different methods on MOCAPPOS. Note that CLDS produces a clear separated grouping of walking motions (blue ◇) and running motion (red ⋆).

Table 1 lists the conditional entropies of each method on the task of clustering MOCAPPOS and MOCAPANG datasets. Note that our method CLDS achieves the best performance with the lowest entropy. It is also confirmed in the scatter plot of top two features using CLDS (Figure 4).

7. Conclusion

Motivated by clustering human motion-capture time sequences, in this paper we developed a novel method of clustering time series data, that can learn joint tem-

poral dynamics in the data (Property P1), handle time lags (Property P2) and produces interpretable features (Property P3). Specifically, our contributions are:

1. *Design of CLDS*: We developed CLDS, complex-valued linear dynamical systems. We then provided Complex-Fit, a novel complex valued EM algorithm for learning the model parameters from observation sequences.
2. *Application to Clustering*: We used a special case of CLDS for time series clustering by enforcing a diagonal transition matrix, corresponding to frequencies. Our clustering method then uses the participation weight (energy) of the hidden variables as features, thus eliminating lags. Such an approach yields all three desired properties.
3. *Validation*: Our approach produces significant improvement in clustering quality (1.5 to 5 times better than several popular time series clustering methods) when evaluated on real motion capture sequences.

CLDS is insensitive to the rotations in the hidden variables due to properties of the complex normal distributions. Moreover we showed that CLDS includes PCA, DFT and AR as special cases.

Acknowledgments The authors would like to thank Christos Faloutsos for discussions. This material is based on work supported by the NSF under Grants No. CNS-0721736, IIS-1017415 and under the auspices of the U.S. DoE by LLNL under contract No. DE-AC52-07NA27344, subcontract B594252.

References

Andersen, Heidi H., Hojbjerre, Malene, Sorensen, Dorte, and Eriksen, Poul Svante. *Linear and graphical models for the multivariate complex normal distribution*. Springer-Verlag, 1995.

Brandwood, D.H. A complex gradient operator and its application in adaptive array theory. *Communications, Radar and Signal Processing, IEE Proceedings F*, 130(1): 11 –16, 1983.

Buzan, D., Sclaroff, S., and Kollios, G. Extraction and clustering of motion trajectories in video. In *ICPR*, volume 2, 2004.

Ding, Chris and He, Xiaofeng. K-means clustering via principal component analysis. In *ICML*, pp. 29, New York, NY, USA, 2004. ACM.

Fu, Ada Wai-Chee, Keogh, Eamonn J., Lau, Leo Yung Hang, and Ratanamahatana, Chotirat (Ann). Scaling and time warping in time series querying. In *VLDB*, pp. 649–660, 2005.

Ghahramani, Zoubin and Hinton, Geoffrey E. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2, February 1996.

Goodman, N. R. Statistical analysis based on a certain multivariate complex gaussian distribution (an introduction). *The Annals of Mathematical Statistics*, 34(1):pp. 152–177, 1963.

Gunopulos, Dimitrios and Das, Gautam. Time series similarity measures and time series indexing. In *SIGMOD Conference*, Santa Barbara, CA, 2001. Tutorial.

Hjorungnes, A. and Gesbert, D. Complex-valued matrix differentiation: Techniques and key results. *IEEE Transactions on Signal Processing*, 55(6):2740 –2746, 2007.

Kagami, S., Mochimaru, M., Ehara, Y., Miyata, N., Nishiwaki, K., Kanade, T., and Inoue, H. Measurement and comparison of human and humanoid walking. In *CIRA*, volume 2, pp. 918 – 922 vol.2, 2003.

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

Keogh, Eamonn J. Exact indexing of dynamic time warping. In *VLDB*, pp. 406–417, 2002.

Lee, Jehee and Shin, Sung Yong. A hierarchical approach to interactive motion editing for human-like figures. In *SIGGRAPH*, pp. 39–48, 1999.

Li, Lei, Prakash, B. Aditya, and Faloutsos, Christos. Parsimonious linear fingerprinting for time series. In *PVLDB*, volume 3, pp. 385–396, 2010.

Mathews, John H. and Howell, Russell W. *Complex Analysis for Mathematics and Engineering*. Jones & Bartlett Pub, 5 edition, January 2006.

Nanopoulos, A., Alcock, R., and Manolopoulos, Y. Feature-based classification of time-series data. In *Intl. Journal of Computer Research*, 2001.

Rodriguez, J. J. and Alonso, C. J. Interval and dynamic time warping-based decision trees. In *ACM Symposium on Applied Computing (SAC)*, 2004.

Safonova, Alla, Pollard, Nancy, and Hodgins, Jessica K. Optimizing human motion for the control of a humanoid robot. In *AMAM2003*, March 2003.

Shumway, R. H. and Stoffer, D. S. An approach to time series smoothing and forecasting using the em algorithm. *Journal of Time Series Analysis*, 3:253–264, 1982.

Tipping, Michael E. and Bishop, Chris M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61:611–622, 1999.

Wu, Y. and Chang, E. Y. Distance-function design and fusion for sequence data. In *CIKM*, 2004.

Xi, Xiaopeng, Keogh, E., Shelton, C., Wei, L., and Ratanamahatana, C. A. Fast time series classification using numerosity reduction. In *ICML*, 2006.