52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy

Low-rank Representation of Neural Activity and Detection of Submovements

Young Hwan Chang, Mo Chen, Suraj Gowda, Simon A. Overduin, Jose M. Carmena and Claire Tomlin

Abstract— **In this study, Robust Principal Component Analysis (RPCA) is applied to neural spike datasets to extract neural signatures that signify the onset of submovements, a type of motor primitive. Given neural activity recorded from rhesus macaques during a set of reaches between targets in a horizontal plane, we aim to identify common event-related neural features and validate submovement-based motor primitives inferred from the hand velocity profiles. Such features represent common dynamic patterns across many experimental trials and may be used as a signature to detect discrete events such as submovement onset. We present RPCA, a method well suited for extracting data matrices' low-rank component and this method allows (1) removal of task-irrelevant signal from data, (2) identification of task-related dynamic patterns, and (3) detection of submovements. We also explored using the Random Projection (RP) technique and applying RP to data prior to applying RPCA improved the submovement prediction performance by de-sparsifying neural data while preserving certain statistical characteristics of aggregate neural activity.**

I. INTRODUCTION

Many brain-machine interface (BMI) researchers have demonstrated the feasibility of using input-output models for reconstructing hand trajectories given recorded neural activity. For example, some recent research [1][2][3] has focused on the relationship between kinematic parameters such as reach direction and speed, and measured activity of motor cortex neurons. In this view, neural activity encodes movement parameters in a straightforward manner; for instance, a neuron's spiking activity may be spatially tuned to reach direction [4][5]. In order to design a decoder, model parameters are adjusted to minimize the error between model output and actual hand movements based on a statistical criterion such as mean square error.

Until recently most studies have focused on the spatial, rather than temporal, structure of neural activity. Several recent studies [6] [7] [8] focus on the temporal complexity and heterogeneity of single-neuron activity in the premotor and motor cortices. In [6], Churchland *et al.* showed that neural activity patterns in the primary motor cortex and premotor cortex associated with nearly identical velocity profiles can be very different. Yu *et al.* [8] considered the problem of extracting smooth, low-dimensional neural trajectories that summarize the activity recorded simultaneously from many neurons during individual experimental trials. They proposed a novel method for extracting neural trajectories - Gaussian-process factor analysis - which unifies smoothing and dimensionality-reduction operations in a common probabilistic framework.

Much of the motor control literature has dealt with the possibility that movements may be programmed via combinations of discrete motor primitives [9][10][11][12][13]. Several studies suggested forms of motor primitives which are united in providing the nervous system with a putative mechanism for reducing both the number of degrees of freedom it has to control, and the frequency at which this control needs to be executed. One example of a motor primitive is a submovement, which is the focus of this paper and will be described in the next section.

Our goal is to present a new spatiotemporal perspective on analyzing neural activity data, which consists of the spike rates of a group of neurons as a function of time. The spike rate of neuron, or unit, is obtained by counting its action potentials within a time bin and dividing by the duration of that bin. The data are collected from a monkey instructed to repeatedly perform a simple task. Robust principal component analysis (RPCA) [14] is applied to the data matrix to extract temporal characteristics of neural activity by decomposing the data matrix into a low-rank component and a sparse component. This allows us to identify the similarities in spatial and temporal structures when the monkey performs similar tasks at different times.

In the context of neural activity, the low-rank matrix corresponds to the common features of neural activity across similar motor primitives, which are submovements in this paper; the sparse matrix represents the uncommon component across submovements, and is interpreted as noise or as signal unrelated to submovement onset. Using RPCA, we test how well neural activity reflects the onset of submovements across different task conditions.

II. CHARACTERIZING TASK-RELEVANT PATTERNS

Neural activity is typically studied by averaging noisy spiking activity across multiple experimental trials to obtain an approximate neural firing rate that varies smoothly over time. However, if neural activity is more a reflection of internal neural dynamics rather than response to external stimulus, the time series of neural activity may differ even when an animal is performing nominally identical tasks

Y.H. Chang, M. Chen, S. Gowda, S.A. Overduin are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA {yhchang, mochen72, surajgowda, overduin}@berkeley.edu

Prof. J.M Carmena with the Department of Electrical Engineering and Computer Sciences, Helen Wills Neuroscience Institute, University of California, Berkeley, and UCB/UCSF Graduate Program in Bioengineering carmena@eecs.berkeley.edu

Prof. Claire Tomlin is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley and is Faculty Scientist, Life Sciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720 USA tomlin@eecs.berkeley.edu

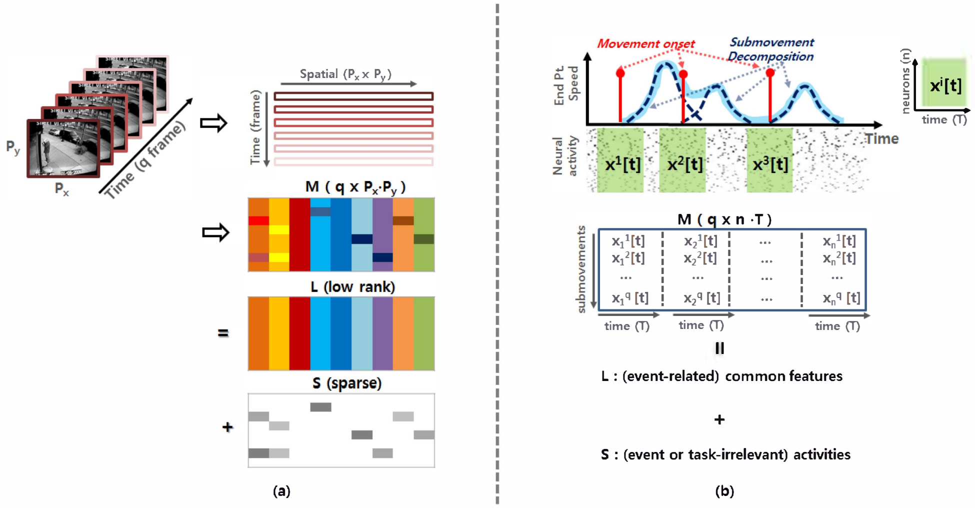

Fig. 1. (a) RPCA applied to computer vision. A typical example of video surveillance where the low-rank component represents the unchanging background and the sparse component represents the movements in the foreground. (b) RPCA applied to neural systems. The low-rank component putatively represents (submovement relevant) neural signatures and the sparse component represents neural activity unrelated to submovement onset.

[8]. This is particularly true of behavioral tasks involving perception, decision making, attention, or motor planning. In these settings, it is critical not to average the neural data across trials, but to analyze it on a trial-by-trial basis [7]. Moreover, stimulus representations in some sensory systems are characterized by the precise spike timing of a small number of neurons [15][16][17], suggesting that the details of operations in the brain are embedded not only in the overall neural spike rate, but also the timings of spikes.

Neural models of supervised learning are usually concerned with processing static spatial patterns of intensities without regard to temporal information [7]. In this paper, we will show that the low-rank component of the data, which is related to the dynamical features in the population activity, can be extracted by using RPCA. Then, we will use the low-rank neural features to detect and predict the onset of submovements.

A. Submovement Decomposition

According to Gordon *et al.* [18] and Fishbach *et al.* [19], the hand speed profile as a function of time resulting from arm movements can be represented by a sum of bell-shaped functions, each of which is called a “submovement”. One biological interpretation of submovement decompositions is that instead of applying continuous control to the arm, the brain controls arm movements by initiating discrete submovements. The velocity profile of a more complex movement looks like a sum of several bell-shapes, where submovements following the first can be interpreted as corrections to the initial submovement.

Rohrer and Hogan outline various types of roughly bell-shaped functions representing submovements and present algorithms for fitting sums of bell-shaped functions to kinematic data [20][21]. The types of bell-shaped functions include the Gaussian curve, support-bounded log-normal curve, and the minimum jerk curve. In this paper, the planar velocity of the monkey’s hand is decomposed into minimum jerk curves, similar methods to the ones applied by [21]. In the subsequent RPCA analysis, submovements with small amplitude and/or long duration are ignored to avoid artifacts of overfitting.

B. Robust Principal Component Analysis (RPCA)

Suppose we are given a large data matrix $\mathbf{M}$, which has common features in the low-rank component and may contain some anomaly in the sparse component as shown in Figure 1 (a). It is natural to model the common variations as approximately the low-rank component $\mathbf{L}$, and the anomaly as the sparse component $\mathbf{S}$. For example, in video surveillance, we need to identify activities that stand out from the background given a sequence of video frames [14]. Figure 1 (a) shows that if we stack the video frames as rows of a matrix $\mathbf{M} \in \mathbb{R}^{q \times P_x \cdot P_y}$ where q is the number of frames, and P_x and P_y represent the number of pixels of 2-D images respectively, then the low-rank component $\mathbf{L}$ corresponds to the stationary background and the sparse component $\mathbf{S}$ captures the moving objects in the foreground. Using RPCA, we can decompose $\mathbf{M} = \mathbf{L} + \mathbf{S}$. We can formulate this as follows:

$$\min_{L,S} ||\mathbf{L}||_* + \lambda ||\mathbf{S}||_1 \text{ s.t. } \mathbf{M} = \mathbf{L} + \mathbf{S} \tag{1}$$

where $||\mathbf{L}||_*$ denotes the nuclear norm of the matrix $\mathbf{L}$, i.e. the sum of the singular value of $\mathbf{L}$, and $||\mathbf{S}||_1 = \sum_{ij} |\mathbf{S}_{ij}|$ represents l_1-norm of $\mathbf{S}$. Choosing the tuning parameter λ to be $\lambda = 1/\sqrt{\max(n_1, n_2)}$, works well for incoherent matrices where n_1, n_2 represent the dimension of matrix $\mathbf{M}$ [14]. For practical problems, however, it is often possible to improve performance by choosing λ according to prior knowledge about the solution.

C. Neural population dynamics

The motor and premotor cortex have been extensively studied but their basic response properties are poorly understood [7]. Also, there is debate about whether neural activity relates to muscles or to abstract movement features. We define the motor cortical activity, which represents movement parameters as per equation (2), and the dynamical system that generates movements as per equation (3) [7]:

$$\begin{aligned} x_i(t) &= f_i(\text{param}_1(t), \text{param}_2(t), \text{param}_3(t), ...) \quad (2) \\ \dot{\mathbf{x}}(t) &= g(\mathbf{x}(t)) + \mathbf{u}(t) \quad (3) \end{aligned}$$

where $x_i(t)$ is the firing rate of neuron i at time t, f_i is its tuning function, and each param_j may represent a movement parameter such as hand velocity, target position or direction. In (3), $\mathbf{x} \in \mathbb{R}^n$ is a vector describing the firing rate of all neurons where n is the number of neurons, $\dot{\mathbf{x}}$ is its derivative, g is an unknown function, and $\mathbf{u}$ is an external input. In (3), neural activity is governed by underlying dynamics, $g(\cdot)$, so dynamical features should be present in the population activity.

Although neural population dynamics is typically much more complex than a simple linear model, here we simply consider a linear model because piecewise linear modeling is the process of developing a series of locally linear models which approximates a nonlinear system such as (3). This example can be used to illustrate how we construct our input matrix $\mathbf{M}$ for analyzing neural activity data and in this setting, the low-rank component represents the characteristics of dynamical system.

Consider a simple linear model: $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{u}(t)$ where $\mathbf{A} \in \mathbb{R}^{n \times n}$ represents the system or plant matrix and $\mathbf{u} \in \mathbb{R}^n$ represents an external input such as sensory input or task cue which affects the neural activity. We will align neural activity with the times that a monkey is engaged in a repeated task, so for simplicity, we assume no external input ($\mathbf{u} = \mathbf{0}$). Then, if the initial vector $\mathbf{x}_0 = \mathbf{x}(t = 0)$ is aligned with eigenvector $\mathbf{r}_k$ of the matrix $\mathbf{A}$, the dynamics is simple, $\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{r}_k = \lambda_k \mathbf{r}_k$ where λ_k is the corresponding eigenvalue; the solution of this equation is $\mathbf{x}(t) = \mathbf{r}_k e^{\lambda_k t}$. If $\mathbf{A}$ is diagonalizable, then any vector in an n-dimensional space can be represented by a linear combination of the right and left eigenvectors (denoted $\mathbf{l}_k$) of the matrix $\mathbf{A}$:

$$\mathbf{x}_0 = \sum_{k=1}^{n} (\mathbf{l}_k \cdot \mathbf{x}_0)\mathbf{r}_k \implies \mathbf{x}(t) = \sum_{k=1}^{n} \eta_k \mathbf{r}_k e^{\lambda_k t} \quad (4)$$

where $\eta_k \triangleq \mathbf{l}_k \cdot \mathbf{x}_0 \in \mathbb{R}$ and $\mathbf{x}(t) = \left[x_1(t), x_2(t), ..., x_n(t)\right]^\top$.

Consider high dimensional time series data $\mathbf{x}^j[t] \triangleq \left[\mathbf{x}^j(1) \quad \mathbf{x}^j(2) \quad ... \quad \mathbf{x}^j(T)\right] \in \mathbb{R}^{n \times T}$ where the superscript represents jth submovement ($j = 1, ..., q$), $(\cdot)$ represents the sample time points, n is the number of neurons and T is the number of time points. By (4), we can write $\mathbf{x}^j[t]$ as

$$\mathbf{x}^j[t] = \left[\sum_{k=1}^{n} \eta_k^j \mathbf{r}_k e^{\lambda_k 1 \cdot \Delta T} \quad ... \quad \sum_{k=1}^{n} \eta_k^j \mathbf{r}_k e^{\lambda_k T \cdot \Delta T}\right] \quad (5)$$

where η_k^j represents the initial condition for jth submovement and ΔT represents a sampled time or time bin (in this study, we choose $\Delta T = 50$ ms). Therefore, neural activity $\mathbf{x}^j[t]$ reflects underlying dynamics related to eigenvalues (λ_k) and eigenvectors ($\mathbf{r}_k$), even though the initial condition may be different across submovements (η_k^j). If we stack the neural activities $\mathbf{x}^j[t]$ across all the submovements and represent these data as a matrix $\mathbf{M} \in \mathbb{R}^{q \times n \cdot T}$ as shown in (6) and Figure 1 (b), then we can extract common dynamical features.

$$\mathbf{M} = \begin{bmatrix} x_1^1[t] & x_2^1[t] & ... & x_n^1[t] \\ x_1^2[t] & x_2^2[t] & ... & x_n^2[t] \\ ... & ... & ... & ... \\ x_1^q[t] & x_2^q[t] & ... & x_n^q[t] \end{bmatrix} = \begin{bmatrix} \mathcal{X}_1 & \mathcal{X}_2 \; ... \; \mathcal{X}_n \end{bmatrix} \triangleq \mathbb{X} \quad (6)$$

where $\mathcal{X}_i \triangleq \left[\mathbf{e}_i^\top \mathbf{x}^1[t]; \mathbf{e}_i^\top \mathbf{x}^2[t]; ...; \mathbf{e}_i^\top \mathbf{x}^q[t]\right] \in \mathbb{R}^{q \times T}$ represents the temporal neural activity of the ith neuron, $\mathbf{e}_i \in \mathbb{R}^n$ is a unit vector, and q is the number of submovements across all data. Note that we aligned each time series data $\mathbf{x}^j[t]$ to the same temporal condition (submovement onset) as shown in Figure 1 (b) but we do not separate different types of submovement. For example, submovements with different reach directions, or with different ordinal positions in an overlapped series of submovements, are combined in our input matrix $\mathbb{X}$. Even though the activity of each neuron may vary significantly across submovements, some portion of the variability may reflect common features.

III. Disentanglement

In the previous section, we showed how to construct our input matrix $\mathbb{X}$. In [14], Candès *et al.* discuss the identifiability issue. For example, suppose the matrix $\mathbf{M}$ is equal to $\mathbf{e}_1\mathbf{e}_1^*$. Then, we cannot decide whether it is low-rank or sparse since $\mathbf{M}$ is both sparse and low-rank. To make the problem meaningful, the low-rank component $\mathbf{L}$ must not be sparse. In other words, the singular vectors of the low-rank matrix $\mathbf{L}$ should be reasonably spread out. Another identifiability issue arises if the sparse matrix has low-rank. For example, this will occur if all the nonzero entries of $\mathbf{S}$ occur in a few columns or in a few rows. In many applications in image and video analysis, practical low-rank and sparse separation gives visually appealing solutions.

However, only a small subset of the whole ensemble of neurons is active at any moment, as seen in the dataset shown in Figure 3 (left). This implies that the singular vectors of the low-rank component may not be reasonably spread out due to the sparsity of the input matrix $\mathbf{M}$. This observation about the neural data makes the solution to the separation problem seem ambiguous. However, random projection (RP) can both succinctly summarize our sparse data in a lower-dimensional space, while at the same time de-sparsify the data so that they can be reliably separated into a low-rank and a sparse component.

A. Random Projection (RP)

Recent theoretical work has identified random projection as a promising dimensionality reduction technique [22]. Projecting the data onto a random lower-dimensional subspace preserves the similarity of different data vectors, for example, the distances between the points are approximately preserved. Also, RP can reduce the dimension of data while keeping clusters of data points well-separated [22]. Moreover, using RP is substantially less expensive to compute than using techniques such as PCA because RP is data-independent.

The idea of RP is that a small number of random linear projections can preserve key information. Theoretical work [22][25][26][27] guarantees that with high probability, all pairwise Euclidean and geodesic distances between points on a low-dimensional manifold are well-preserved under the mapping $\Psi : \mathbb{R}^n \rightarrow \mathbb{R}^m, m < n$. Consider a linear signal model

$$\mathbf{y}(t) = \mathbf{\Psi}\mathbf{x}(t) = \sum_{i=1}^{n} x_i(t)\psi_i \in \mathbb{R}^m \tag{7}$$

where $\Psi = \begin{bmatrix} \psi_1 & \psi_2 & \ldots & \psi_n \end{bmatrix}$ is an $m \times n$ projection matrix whose elements are drawn randomly from independent identical distributions. First, note that the dimensionality of the data $\mathbf{x}$ is reduced since $m < n$. Also, if we define $\mathcal{Y}_i \triangleq \left[\bar{\mathbf{e}}_i^\top \mathbf{y}^1[t]; \bar{\mathbf{e}}_i^\top \mathbf{y}^2[t]; \ldots; \bar{\mathbf{e}}_i^\top \mathbf{y}^q[t]\right] \in \mathbb{R}^{q \times T}$ where $\bar{\mathbf{e}}_i \in \mathbb{R}^m$ and $\mathbb{Y} \triangleq \begin{bmatrix} \mathcal{Y}_1 & \mathcal{Y}_2 & \ldots & \mathcal{Y}_m \end{bmatrix}$, then $\mathbb{Y}^\top = (\mathbf{\Psi} \otimes \mathbf{I}_T)\mathbb{X}^\top$ or $\mathbb{Y} = \mathbb{X}(\mathbf{\Psi}^\top \otimes \mathbf{I}_T)$ where $\otimes$ represents the Kronecker product and $\mathbf{I}_T \in \mathbb{R}^{T \times T}$ is an identity matrix.

B. Identifiability, dimensionality reduction and benefits

By using RP, we can handle the identifiability issue because our input matrix $\mathbb{Y} \in \mathbb{R}^{q \times m \cdot T}$ is denser than the original input matrix $\mathbb{X} \in \mathbb{R}^{q \times n \cdot T}$. RP also preserves the similarity of the data vectors well and reduces the dimension. Furthermore, because our data are sparse, no information is lost from performing RP on $\mathbb{X}$.

1) Identifiability: Suppose our input $\mathbb{X}$ in equation (6) can be decomposed as $\mathbb{X} = \mathbf{L} + \mathbf{S} = \sum_{i=1}^{d_L} \sigma_i \mathbf{u}_i \mathbf{v}_i^* + \sum_{i=1}^{d_S} \lambda_i \mathbf{a}_i \mathbf{b}_i^*$ where σ_i are the positive singular values, $\mathbf{u}_i \in \mathbb{R}^{q \times 1}, \mathbf{v}_i^* \in \mathbb{R}^{1 \times n \cdot T}$ are the left- and right-singular vectors of $\mathbf{L}$, and d_L represents the rank of the matrix $\mathbf{L}$. d_S is the number of sparse components in $\mathbf{S}$, and $\mathbf{a}_i \in \mathbb{R}^{q \times 1}, \mathbf{b}_i \in \mathbb{R}^{1 \times n \cdot T}$ are sparse with only one nonzero entry respectively. By using RP, we have for $\mathbb{Y}$,

$$\begin{aligned} \mathbb{Y} &= \mathbb{X}(\mathbf{\Psi}^\top \otimes \mathbf{I}_T) \triangleq \mathbb{X}\mathbf{R} = \mathbf{L}\mathbf{R} + \mathbf{S}\mathbf{R} \\ &= \sum_{i=1}^{d_L} \sigma_i \mathbf{u}_i (\mathbf{R}^\top \mathbf{v}_i)^* + \sum_{i=1}^{d_S} \lambda_i \mathbf{a}_i (\mathbf{R}^\top \mathbf{b}_i)^* \\ &= \sum_{i=1}^{d_L} \sigma_i \mathbf{u}_i \tilde{\mathbf{v}}_i^* + \sum_{i=1}^{d_S} \lambda_i \mathbf{a}_i \tilde{\mathbf{b}}_i^* \end{aligned} \tag{8}$$

where we denote $(\mathbf{\Psi}^\top \otimes \mathbf{I}_T)$ by $\mathbf{R}$. As we mentioned above, our input $\mathbb{X}$ is sparse, so the singular vectors of the low-rank matrix $\mathbf{L}$ might not be reasonably spread out. However, by using RP (multiplying by $\mathbf{R}$), the singular vectors $\tilde{\mathbf{v}}_i$ of the resulting matrix become reasonably spread out.

2) Compressive Sensing (CS) and recovery of the exact signal: CS theory states that with high probability, every K-sparse signal $z \in \mathbb{R}^N$ (i.e., every signal z having only K nonzero components) can be recovered from just $M(\geq cK\log(N/K))$ linear measurements $b = Az$. Here, A represents an $M \times N$ measurement (or encoding) matrix drawn randomly from an acceptable distribution for satisfying the incoherence condition and c is a small constant [23]. CS decoding involves recovering the signal $z \in \mathbb{R}^N$ from its measurements $b \in \mathbb{R}^M$ where $M < N$ in general. Although such inverse problems are generally ill-posed, CS recovery algorithms exploit the additional assumption of sparsity in z with the basis A to identify the correct signal z from an uncountable number of possibilities [23].

We can consider $(\mathbf{\Psi} \otimes \mathbf{I}_T)$ as a measurement matrix A. Then, since we choose an incoherent matrix $\mathbf{\Psi}$ drawn randomly with i.i.d and the coherence of $\mathbf{\Psi}$ is exactly the same as that of $\mathbf{\Psi} \otimes \mathbf{I}_T$ (i.e., $\mu(\mathbf{\Psi}) = \mu(\mathbf{\Psi} \otimes \mathbf{I}_T)$), we can recover the exact signal $\mathbb{X}$ from the randomly projected signal $\mathbb{Y}$. Recall that the jth columns of $\mathbb{X}^\top$ and $\mathbb{Y}^\top$ represent the high dimensional time series data and projected time series data of the jth submovement, respectively. Since the measurement matrix $\mathbf{\Psi} \otimes \mathbf{I}_T$ satisfies the coherence condition and the jth column of $\mathbb{X}^\top$, $\mathbb{X}^\top_{(:,j)}$, is known to be sparse, we can recover the original data from $\mathbb{Y}^\top_{(:,j)}$ by solving the l_1-minimization [23] [24]:

$$\min_z ||z||_1 \text{ s.t. } \mathbb{Y}^\top_{(:,j)} = (\mathbf{\Psi} \otimes \mathbf{I}_T)z \tag{9}$$

Therefore, RP does not lose any information even though dimensionality is reduced.

3) Dimension Reduction and Eccentric Distribution: The dimension of the original input $\mathbb{X}$ is $q \times (n \cdot T)$, where q is the number of submovements, n is the number of neurons, and T is the number of time points. The dimension of the new input $\mathbb{Y}$ after application of RP to $\mathbb{X}$ is $q \times (m \cdot T)$ where $m < n$. Thus, the dimension of the projected input $\mathbb{Y}$ is reduced.

In [22], Dasgupta showed that even if the original distribution of data samples is highly skewed (having an ellipsoidal contour of high eccentricity), its projected counterparts will be more spherical. For example, neural activity data vectors often form very eccentric clusters. As shown by the different amplitude for different neurons in Figure 3 (left), some neurons are highly activated (30-40 spikes/sec) but others typically have only a few spikes per second. Since it is conceptually much easier to design algorithms for spherical clusters than ellipsoidal ones, this feature of random projection can simplify the separation into the low-rank and sparse components. Therefore, we can reduce the computational complexity of the non-smooth convex optimization, in particular l_1 and nuclear norms minimization, used in RPCA[1].

C. Simple Examples

To illustrate the issue of identifiability and how RP can alleviate the issue, we consider a simple example: we generate a sparse low-rank input matrix $\mathbb{X} \in \mathbb{R}^{50 \times 2 \cdot 10}$ ($q = 50$, $n = 2$,

[1] Many speedup methods were developed in optimization by avoiding large-scale SVD. In [28], Mu *et al.* demonstrated the power of projected matrix nuclear norm by reformulating RPCA and in [29], Zhou *et al.* demonstrated the effectiveness and the efficiency of Bilateral Random Projections. However, both methods consider a dense matrix $\mathbb{X}$ while in this paper we consider the case when the input matrix $\mathbb{X}$ is sparse.

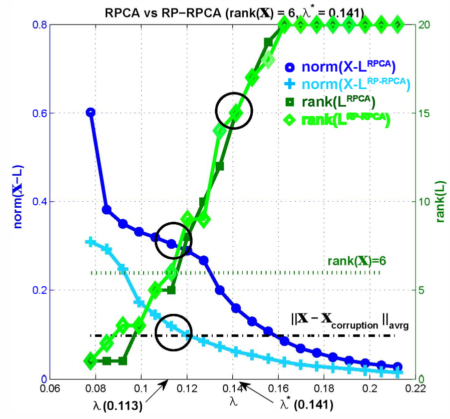

Fig. 2. Results: we run RPCA for $\mathbb{X}_{corruption}, \mathbb{Y}_{corruption}$ (We added sparse corruption to $\mathbb{X}$). Left y-axis represents the norm of $\mathbb{X} - \mathbf{L}$ and the right y-axis shows the rank of $\mathbf{L}$.

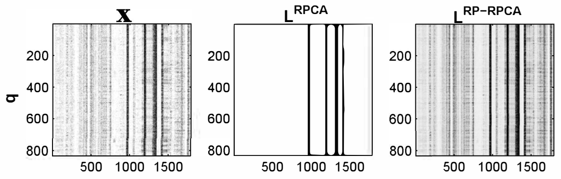

Fig. 3. The low-rank matrices from both RPCA and RP-RPCA where $\mathbb{X}$ are input matrices and we choose $m = n = 64$ for the comparison.

$T = 10$) where the rank of $\mathbb{X}$ is 6. Note that in this example we chose the same dimension for the input $\mathbb{X}$ and $\mathbb{Y}$ ($m = n$). This is done so that $\mathbf{\Psi} \in \mathbb{R}^{m \times n}$ in equation (7) is invertible, allowing us to compare the outputs of RPCA and RP-RPCA directly, as described below. In general, choosing $m < n$ makes $\mathbb{Y}$ much denser because information is compressed by RP.

To evaluate the performance of separation into a low-rank and a sparse component, we add sparse corruption for $\mathbb{X}$: $\mathbb{X}_{corruption} = \mathbb{X} + \mathbf{S}_{corruption}$ and $\mathbb{Y}_{corruption} = \mathbb{X}_{corruption}\mathbf{R} = \mathbb{X}\mathbf{R} + \mathbf{S}_{corruption}\mathbf{R}$ where $\mathbf{R} = (\mathbf{\Psi}^{\top} \otimes \mathbf{I}_T)$ is the projection so $\mathbb{Y}_{corruption}$ is the projected corrupted input matrix $\mathbb{X}_{corruption}$. To compare the performance of RP-RPCA with RPCA, we first decompose $\mathbb{Y}_{corruption}$ into its low-rank and sparse components. Then, we invert the projection:

$$\begin{aligned} \mathbb{X}_{corruption} &= \mathbf{L}^{rpca} + \mathbf{S}^{rpca} \quad \text{(orginal RPCA)} \\ &= \mathbb{Y}_{corruption}\mathbf{R}^{-1} = (\mathbf{L}^{rpca}_{\mathbb{Y}} + \mathbf{S}^{rpca}_{\mathbb{Y}})\mathbf{R}^{-1} \\ &\triangleq \bar{\mathbf{L}}^{rpca} + \bar{\mathbf{S}}^{rpca} \quad \text{(RP-RPCA)} \end{aligned}$$

where we define $\bar{\mathbf{L}}^{rpca} \triangleq \mathbf{L}^{rpca}_{\mathbb{Y}}\mathbf{R}^{-1}$ and $\bar{\mathbf{S}}^{rpca} \triangleq \mathbf{S}^{rpca}_{\mathbb{Y}}\mathbf{R}^{-1}$. In this example, inverting $\mathbf{R}$ is possible because we choose $\mathbf{\Psi} \in \mathbb{R}^{m \times n}$ where $m = n$. If $m < n$, then we have to solve l_1-minimization in equation (9) to obtain $\bar{\mathbf{L}}^{rpca}$ and $\bar{\mathbf{S}}^{rpca}$ instead. Figure 2 shows the statistics of both RPCA and RP-RPCA (in which RPCA is applied to the matrix $\mathbb{Y}$) as a function of the tuning parameter λ in equation (1). In this example, $\lambda^* = 1/\sqrt{\max(q, n \cdot T)} = 1/\sqrt{50}$. Since our input is still sparse in this example, the rank of both $\mathbf{L}^{rpca}, \bar{\mathbf{L}}^{rpca}$ is 15 for $\lambda^* = 0.141$ (rank($\mathbb{X}$) = 6). If we choose $\lambda = 0.113$ (discounting the penalty for sparse component), the ranks of $\mathbf{L}^{rpca}, \bar{\mathbf{L}}^{rpca}$ are approximately 6, which is the same as the rank of the original input $\mathbb{X}$. With this choice of λ, for RPCA we find that $||\mathbb{X} - \mathbf{L}^{rpca}||$ is much bigger than the original corruption signal $||\mathbb{X} - \mathbb{X}_{corruption}|| = ||\mathbf{S}_{corruption}||$. On the other hand, for the RP-RPCA, we have $||\mathbb{X} - \bar{\mathbf{L}}^{rpca}|| \approx ||\mathbf{S}_{corruption}||$. Therefore, for RP-RPCA, the separation of the low-rank component and sparse component is close to the true solution but for original RPCA, we have misidentification in both the low-rank and sparse components.

IV. Application to Neural Data

A. Experimental Setup

The experiment involved an adult male rhesus macaque instructed to make visually-guided planar reaches with its right hand[2]. Hand velocity data (sampled at 100 Hz) were decomposed into a sum of minimum-jerk basis functions. Figure 3 (left) shows the actual neural activities aligned with movement onset. The aligned neural activity shows that the ratios between units' mean firing rates are fairly constant from the salient vertical striations in the plots and temporal patterns exists across all the submovements. Also, as mentioned previously, the neural population activities are sparsely active (white color represents 0 spikes/sec) and show eccentric behavior; for example, some neurons have a much higher spiking rate than others.

B. Prediction

In order to extract the common features, we divide our dataset into training (70%) and test (30%) datasets. First, we run RPCA and RP-RPCA to extract the low-rank components, and then use these components as signatures or templates to detect submovement onset. Here, we simply use a correlation function as our metric, $\gamma_k = \frac{\langle s_k, \bar{\mathbf{T}} \rangle}{||s_k|| ||\bar{\mathbf{T}}||}$ where $\bar{\mathbf{T}}$ represents the extracted temporal characteristics or template (in this case, an average of the low-rank component across submovements). The neural activity signal can be denoted as $s_k = \begin{bmatrix} x_1^k[t] & \dots & x_n^k[t] \end{bmatrix} \in \mathbb{R}^{1 \times n \cdot T}$ (for RPCA) or $s_k = \begin{bmatrix} y_1^k[t] & \dots & y_m^k[t] \end{bmatrix} \in \mathbb{R}^{1 \times m \cdot T}$ (for RP-RPCA). For practical purposes, we can choose a correlation threshold and if the correlation is over the chosen threshold, we label a submovement onset as detected.

C. Results

Figure 3 (middle) (right) show the low-rank matrix from both RPCA and RP-RPCA respectively (for simple comparison, we choose $m = n$). Since $\mathbb{X}$ is sparse and has an eccentric distribution, the singular vectors may not be reasonably spread out. Applying RPCA directly to $\mathbb{X}$ would result in the low-rank component being composed of only

[2] All procedures were conducted in compliance with the National Institute of Health Guide for Care and Use of Laboratory Animals and were approved by the University of California, Berkeley Institutional Animal Care and Use Committee. Neural units from primary motor and dorsal premotor cortex were recorded from chronically-implanted arrays.

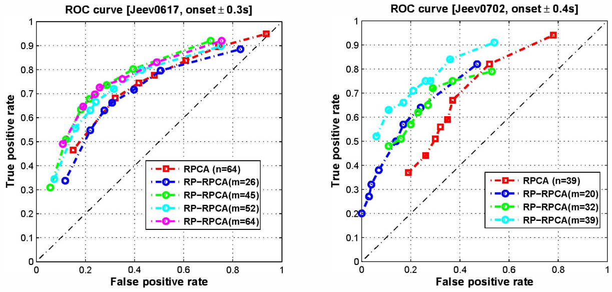

Fig. 4. Receiver Operating Characteristic (ROC) curve of different thresholds: (a) point to point task (b) target jumps task.

highly modulated neural activity (middle). On the other hand, RP-RPCA can extract a low-rank component from a more distributed set of neural dimensions than RPCA alone can.

To accurately predict submovement onset times found by submovement decomposition, the correlation function should peak around the movement onset time. For RP-RPCA, we compare the performance for different projection dimensions. Figure 4 represents the receiver operating characteristics (ROC) curve based on our prediction. We vary thresholds for correlation score and we can see that the overall prediction performance based on RP-RPCA is better than the performance based on RPCA (increasing true positive rate and reducing false positive rate).

V. Conclusion

In this study, we develop a new method, the combination of random projection and RPCA for neural data which are naturally sparse. This is the first application known to us of RPCA to neuronal data and we validate the method via extracting a low-rank elements of motor cortical activity. Such features represent common dynamic structures in the neural activity and can be used as a signature to detect motor primitives such as submovements. We explored the benefits of using RP and showed the improvement of the overall prediction performance.

Acknowledgments

This work is supported by the NIGMS and by the NSF under grant EFRI 1137267.

References

[1] Carmena J.M., Lebedev M.A., Crist R.E., O'Doherty J.E., Santucci D.M., Dimitrov D.F., Patil P.G., Henriquez C.S. and Nicolelis M.A.L., "Learning to control a brain-machine interface for reaching and grasping by primates," *PLoS Biology*, 1(2), 2003, pp.193-208.

[2] Dangi S., Gowda S., Heliot R. and Carmena J.M., "Adaptive Kalman filtering for closed-loop brain-machine interface systems," *Proc. 5th International IEEE EMBS Conference on Neural Engineering*, 2011, pp. 609-612.

[3] Orsborn A.L., Dangi S., Moorman H.G. and Carmena J.M., "Closed-loop decoder adaptation on intermediate time-scales facilitates rapid BMI performance improvements independent of decoder initialization conditions," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 20(4), 2012, pp. 468-477.

[4] Georgopoulos A.P., Kalaska J.F., Caminiti R., Massey J.T., "On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex," *The Journal of Neuroscience*, 2(11), 1982, pp. 152-1537.

[5] Schwartz A.B., Kettner R.E., Georgopoulos A.P., "Primate motor cortex and free arm movements to visual targets in three-dimensional space. I. Relations between single cell discharge and direction of movement," *The Journal of Neuroscience*, 8(8), 1988, pp. 2913-2927.

[6] Churchland M.M. and Shenoy K.V., "Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex," *Journal of Neurophysiology*, 97(6), 2007, pp. 4235-4257.

[7] Churchland M.M., Cunningham J.P., Kaufman M.T., Foster J.D., Nuyujukian P., Ryu S.I. and Shenoy K.V., "Neural population dynamics during reaching," *Nature*, 487(7405), 2012, pp. 51-56.

[8] Yu B.M., Cunningham J.P., Santhanam G., Ryu S.I., Shenoy K.V. and Sahani M., "Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity," *Journal of Neurophysiology*, 102(1), 2009, pp. 614-635.

[9] Keele SW., "Movement control in skilled motor performance," *Psychological Bulletin*, 70, 1968, pp. 387-403.

[10] Brooks VB., "Motor programs revisited," *In: Talbott, R.E., Humphrey DR (eds), Posture and Movement*, New York, 1979, pp. 13-49.

[11] Polit A., Bizzi E., "Characteristics of motor programs underlying arm movements in monkeys," *Journal of Neurophysiology*, 42(1), 1979, pp. 183-94.

[12] Georgopoulos AP, Kalaska JF, Caminiti R, Massey JT, "Interruption of motor cortical discharge subserving aimed arm movements," *Experimental Brain Research*, 49(3), 1983, pp. 327-340.

[13] Overduin S.A., d'Avella A., Roh J., Bizzi E., "Modulation of muscle synergy recruitment in primate grasping," *The Journal of Neuroscience*, 28(4), pp. 880-892.

[14] Candès E.J., Li X., Ma Y. and Wright J., "Robust principal component analysis?", *Journal of ACM*, 58(1), pp. 1-37.

[15] Gerstner W, Kempter R., van Hemmen J.L., Wagner H., "A neuronal learning rule for sub-millisecond temporal coding," *Nature*, 383, 1996, pp. 76-78.

[16] Song S, Miller K.D., Abbott L.F., "Competitive Hebbian learning through spike-timing-dependent synaptic plasticity," *Nature neuroscience*, 3(9), 2000, pp. 919-26.

[17] Long M.A., Jin D.Z., and Fee M.S., "Support for a synaptic chain model of neuronal sequence generation," *Nature*, 468, 2010, pp. 394-399.

[18] Gordon J, Ghilardi MF , and Ghez C., "Accuracy of planar reaching movements. I. Independence of direction and extent variability," *Experimental Brain Research*, 99(1), 1994, pp. 97-111.

[19] Fishbach A., Roy S.A., Bastianen C., Miller L.E., Houk J.C., "Kinematic properties of on-line error corrections in the monkey," *Experimental Brain Research*, 164(4), pp. 442-457.

[20] Rohrer B., Hogan N., "Avoiding spurious submovement decompositions: A globally optimal algorithm," *Biological Cybernetics*, 89(3), 2003, pp. 190-199.

[21] Rohrer B., Hogan N., "Avoiding spurious submovement decompositions II: A scattershot algorithm," *Biological Cybernetics*, 94(5), 2006, pp. 409-14.

[22] Dasgupta S., "Experiments with random projection," *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pp. 143-151.

[23] E. J. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Problems*, 23, pp. 969-985.

[24] E. J. Candès and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, 2008, pp. 21-30.

[25] Bingham E., Mannila H., "Random projection in dimensionality reduction: applications to image and text data," *Proceeding KDD '01 Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 245-250.

[26] Deegalla S. and Bostrom H., "Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification," *5th International Conference on Machine Learning and Applications (ICMLA)*, 2006, pp. 245-250.

[27] Baraniuk R.G. and Wakin M.B., "Random projections of smooth manifolds," *Journal of Foundations of Computational Mathematics*, 9(1), 2009, pp. 51-77.

[28] Mu Y., Dong J., Yuan X. and Yan S., "Accelerated low-rank visual recovery by random projection," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 2609-2616.

[29] Zhou T., Tao D., "Bilateral random projections," *arXiv:1112.5215*, 2011.