Dictionary Construction for Accurate and Low-Cost Subspace Learning in Unsupervised Spike Sorting

Majid Zamani, Salinna Abdullah, Andreas Demosthenous

Department of Electronic and Electrical Engineering
University College London, UK.
E-mail: m.zamani; salinna.abdullah.13; a.demosthenous@ucl.ac.uk

***Abstract* - This paper details the construction of highly reliable and power efficient dictionaries as the main block in unsupervised feature learning from evolving subspaces. Three types of dictionary considered, are the Hadamard $\varphi_{H_h(k)}$, the equiangular tight frame $\varphi_{ETF(k)}$ and the random Bernoulli $\varphi_{Bern(k)}$. The dictionaries are utilized in an unsupervised feature learning algorithm, and the classification results are investigated using a library-based neural spike simulator providing a range of noise levels and 300 different average spike shapes. The proposed dictionaries obtain high performance with a classification error of around 7% over 100 windows of generated data for 3 to 6 clusters and noise levels σ_N between 0.05 and 0.3. The combination of dictionaries and subspace learning present a new class of implantable feature extractors robust to wide signal variations and is well-suited for hardware implementation.**

Keywords - Dictionary construction, low-cost design, spike sorting, subspcae learning.

I. INTRODUCTION

The latest generation of therapeutic devices for bioelectronic medicines provide efficient and reliable treatment with fewer side effects [1]. Such devices decipher neural spike interactions to enable investigation and model application specific processing for biological functionality regularization between the brain and organs in the body. A broad range of applications are suited to using this class of treatment [2]-[6] (e.g. hand prosthesis control). This encourages designers to embed efficacy in alternative treatments by integrating highly reliable on-chip neural spike processors to allow real-time interpretation of multi-channel recording. Computationally efficient feature extraction algorithms which shape their characteristics with the acquired neural data are required [7]-[10].

This paper discusses the construction and implementation of a family of dictionaries exploited in a dictionary-based feature extractor characterised by unsupervised subspace learning introduced in [11]. Selection of a dictionary uses the following principles: 1) It must reduce implementation costs to provide compatibility with strict constraints imposed in implantable devices such as the chip area and power budget by reducing the number of multiplications (or divisions); 2) A dictionary should contain extensive information in the construction phase to cover a variety of linear and non-linear subspace configurations. This ensures a high probability to capture the most informative subspace; and 3) It must be capable of maximizing the subspace quantization efficiency. For example, there are three elements {-1,0 *or* 1} in the proposed dictionaries which give flexible distribution in each of their columns. The principles in selecting the dictionaries help to explore the most informative subspace at different levels of difficulty (e.g. highly similar and noisy subspaces).

Three dictionaries found to satisfy these principles are the Hadamard $\varphi_{H_h(k)}$, the equiangular tight frame $\varphi_{ETF(k)}$ and the random Bernoulli $\varphi_{Bern(k)}$. The construction process and the classification performance will be discussed. The rest of the paper is organized as follows: Section II briefly explains the unsupervised learning algorithm. It continues with dictionary construction and explains the neural data simulator used for examining the classification performance. Section III compares the performance of dictionary-based feature extractors and concluding remarks are drawn in Section IV.

II. FEATURE EXTRACTION CORE DESIGN

A. Unsupervised Subspace Learning Algorithm [11]

The neural spike data $X(i)$ is defined as $X(i) = S(i) + n(i)$ where $S(i)$ contains the signal information that lies in a low-dimensional subspace which evolves over time and $n(i)$ accounts for noise. Referring to Fig. 1 the generated data $X(i)$ is fed to a data partitioning block which selects a segment of a generated neural signal ($X = [x1, x2, \ldots, xn] \in R^N$) and passes it to the feature extraction core. For the k^{th} segment, there are W detected and aligned spike waveforms $x(k)$ which are used in the feature extraction process. R columns of the initially generated dictionaries (e.g. $\varphi_{ETF(k)}$) are chosen and transposed to form the feature extraction matrix $\varphi(k)$ where R depicts the desired number of features. The constituent elements of all the chosen dictionaries ($\varphi_{H_h(k)}$, $\varphi_{ETF(k)}$ or $\varphi_{Bern(k)}$) are {-1,0 *or* 1} which results in efficient realization of the algorithm in hardware.

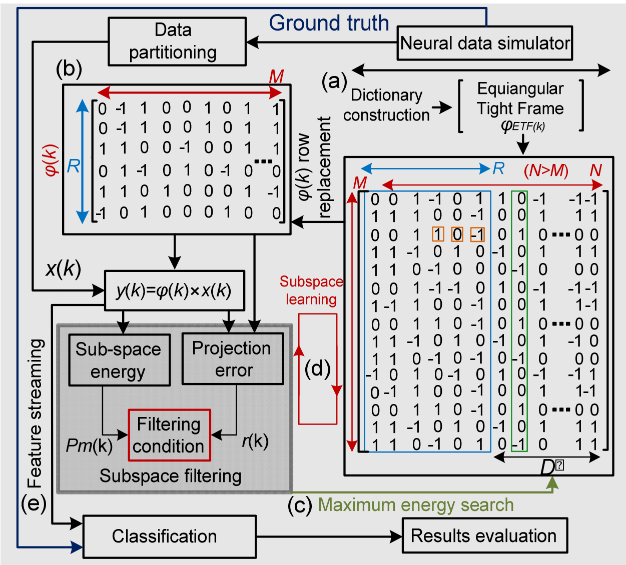

Fig. 1. Dictionary-based feature extraction. (a) $\varphi_{ETF(k)}$ construction. (b) R columns of constructed dictionary are extracted and transposed to form feature extraction dictionary $\varphi(k)$ based on desired number of features. (c) Exploring the index of non-utilized columns D' offering maximum energy and updating $\varphi(k)$ based on it. (d) Subspace learning through iterative subspace synthesis using $P_m(k)$ and $r(k)$. (e) Streaming phase.

The dictionaries contain *M x N* columns and rows *(N > M)*. Each column offers a unique distribution for capturing different subspace configurations. A typical column identified by the green rectangle in Fig. 1(a) consists of three elements $\{-1, 0 \text{ and } 1\}$ to maximize subspace quantization efficiency. The learning is initiated by projection of the first aligned spike waveform by $y(k) = x(k)\times\varphi(k)$. Once the projection is performed, the detectors in the filtering unit namely projection energy $P_m(k)$ and projection error $r(k)$ which is defined as the difference between the original $x(k)$ and the recovered $x\hat{}(k)$ spike waveforms, are activated to quantify the correlation likelihood between $\varphi(k)$ arrays {-1,0 *or* 1} and the optimal subspace. If $r(k) < P_m(k)$ is satisfied, the feature extraction process retains much of the energy in the original subspace which makes optimal separation of the spike waveforms possible. $r(k) \geq P_m(k)$ indicates misalignment of $\varphi(k)$ arrays to the informative subspace. Since the main aim of feature extraction is to maximize the projection energy whilst minimizing the residual, the row in $\varphi(k)$ representing the least feature energy after projection is replaced using a row closer to the residue direction. The replacement process begins by configuring the unused columns (D') in the initially generated dictionary and exploring the column offering the highest projection energy:

$$(l = \arg\max |D'(k)\times x(k)|)\ .$$

The identified column D'_l is used to update $\varphi(k)$ and has good alignment to the optimal subspace. The $\varphi(k)$ updating scheme over a number of learning steps $\left\{e.g.\ \varphi(k)\big|_{t=t_1}, \cdots, \varphi(k)\big|_{t=t_6}\right\}$ intermittently guarantees the optimal separation between the neuron spikes for different noise levels and varying templates in the recording channel.

B. Dictionary Construction

1) Hadamard Matrix

For a matrix that only contains low-cost elements, the Hadamard matrix [12] is the most popular. Its simple structure is an eligible candidate to store optimal weighting for applying the subspace changes on the data stream over time. If H_h is the Hadamard matrix of order h, the following properties of H_h are:

1) $H_h H_h^T = I_h$, where I_h is an identity matrix of order h.
2) $H_h H_h^T = H_h^T H_h$.
3) H_h has orthogonal vectors, it can be changed into another Hadamard matrix by permuting rows and columns and by multiplying rows and columns by -1.
4) The order of Hadamard matrix (h) is 1,2, 4n where n n is a positive integer.

To make the directions of the feature vectors more abundant, an identity matrix is added into the dictionary together with the Hadamard matrix. The identity matrix has the diagonal elements equal to 1 and all others equal to 0; it is a set of orthonormal basis of unit vectors. Therefore, the construction of the first dictionary is:

$$\varphi_{H_h(k)} = \begin{bmatrix} I & H_h \end{bmatrix}. \tag{1}$$

The chosen size of $\varphi_{H_h(k)}$ in simulations is 64 (M = spike length) × (N =128) consists of $\{-1, 0 \text{ and } 1\}$ elements.

2) Equiangular Tight Frame

The construction procedures proposed by Fickus et al. [13] are utilized to generate the equiangular tight frame (ETF) dictionary $\varphi_{ETF(k)}$. If an incidence matrix (A) belongs to a $\left(2,\breve{K},v\right)$-Steiner system [14], where $\breve{K}$ and v are positive integers $\left(\breve{K}<v\right)$, the transpose of it, A^T, has the following properties:

1) The size is: $\dfrac{v(v-1)}{\breve{K}(\breve{K}-1)}\times v$.
2) There are $\breve{K}$ ones in each row.
3) There are $\dfrac{v(v-1)}{\breve{K}(\breve{K}-1)}$ ones in each column.
4) Any two of its columns have an inner product of one.

Algorithm 1. Constructing ETF Dictionary $\varphi_{ETF(k)}$

1. Based on the given M and N calculate $\breve{K}$, v, the known relationship is:

$M=\dfrac{v(v-1)}{\breve{K}(\breve{K}-1)}$ (Denotes the dimension, should equal to original number of features)

$N=v\times(1+\dfrac{v(v-1)}{(\breve{K}-1)})$ (denotes number of vectors in the dictionary)

2. Form an incidence matrix A of size: $\dfrac{v(v-1)}{\breve{K}(\breve{K}-1)}\times v$, which is block design of Steiner system

3. For $j=1,2,\cdots,v$, let H_{h_j} be any $(1+\dfrac{v(v-1)}{(\breve{K}-1)})\times(1+\dfrac{v(v-1)}{(\breve{K}-1)})$ matrix that has orthogonal rows and unimodular entries. A typical matrix of H_h is a Hadamard matrix.

4. Let $D_j(j=1,2\cdots,v)$ be the j^{th} column of the incidence matrix formed in step 2, and replaced every entry that is equal to 1 by a random row in H_{h_j}, and replaced every zero entry by a row of zeros.

5. Combine all the columns together and form the dictionary $\varphi_{ETF(k)}=[D_1\cdots D_v]$.

Algorithm 1 shows the construction procedure to form an ETF dictionary $\varphi_{ETF(k)}$ of M by N. To realize $\varphi_{ETF(k)}$, the conditions need to be satisfied in a way that the generated Hadamard matrix of size $1+\dfrac{(v-1)}{(\breve{K}-1)}$ used in step 3 of Algorithm1 to be real. Typically for a M < 100, M = 6, 7, 28, 35, 66, 99 satisfies the conditions in step 3. As the number of rows in the dictionary should be the same as the number of original spike waveform features, M is set to 66. Therefore, the parameters forming $\varphi_{ETF(k)}$ for the size of 66 × 144 are M = 66, N = 144, $\breve{K}$ = 2 and v = 12. As the value of M = 66 should satisfy the ETF conditions, two dummy rows containing zero values are added to the tail of the $\varphi_{ETF(k)}$ matrix to change the number of original features from 64 to 66, for ETF dictionary analysis.

3) Random Bernoulli Matrix

The third dictionary for realization of $\varphi(k)$ is the Random Bernoulli Matrix (RBM) [15]. It consists only {0 *and* 1}. The following steps generate the matrix:

1) Preset a possibility p.
2) Initialize $\varphi_{Bern(k)}$ of M × 2 N zero matrix.
3) Loop for i = 1 to N and j = 1 to N times, generate a random number between 0 and 1, if this number is larger than p, then assign 1 to $\varphi_{Bern(k)}$, otherwise remain 0.

To conform $\varphi_{Bern(k)}$ with the other dictionaries, the size of $\varphi_{Bern(k)}$ is set to (M =64 × N =128). Examples of the constructed dictionaries using three matrices are shown in Fig. 2.

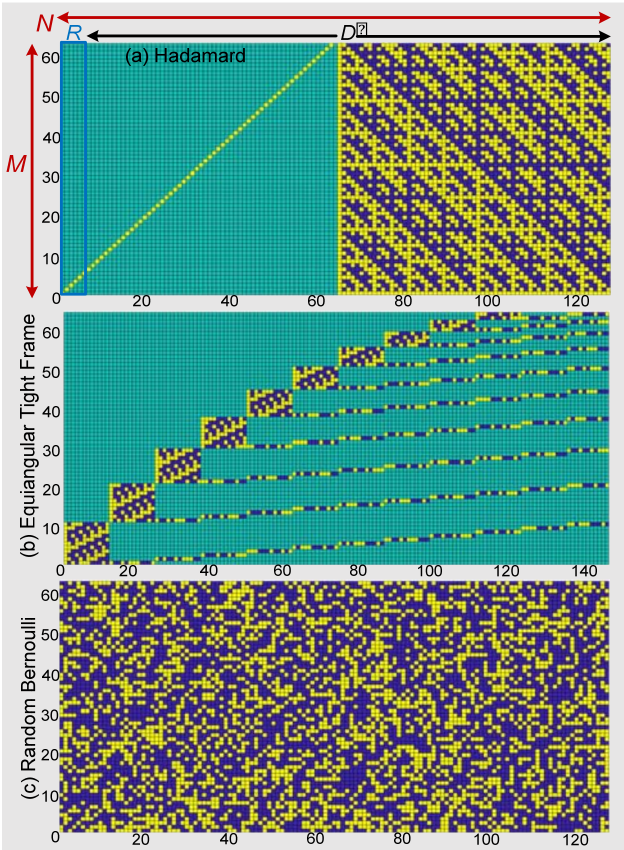

Fig. 2. Constructed dictionaries using (a) Hadamard, (b) ETF and (c) random Bernoulli algorithms. The green, yellow and dark blue dots represent 0, 1 and -1 elements in Hadamard and ETF respectively. In (c), yellow and dark blue dots represent 1 and 0.

C. Neural Data Simulator

A library-based neural spike simulator was developed to emulate extracellular recordings with realistic background noise and a known "ground truth" for evaluation of the algorithms for spike waveform classification. To generate neural spikes, a database of synthetic spike waveforms containing 300 different average spike shapes was constructed. Fig. 3 shows the procedure used for neural data generation and examination of the feature extraction algorithms. The neural spike waveforms are randomly selected and placed in a data stream from the spike library using a defined number of clusters (NOC) and spike firing rates (SFR). The data stream is then corrupted by additive noise with varying standard deviations (e.g. σ_N = 0.01) at random times. The extracted spike templates (Template 1…Template W) form a matrix; it is the function of the simulator setting over each segment (e.g. $F(\sigma_N, SFR, NOC|_{k_1})$ where k_1 shows the first data segment. The changes over each segment embed four elements $(\sigma_N, SFR, NOC \text{ and } time = (k_1 \cdots k_n) \times W)$ to emulate a real recording channel and changing subspace for in-depth analysis of the feature extraction methods at different conditions.

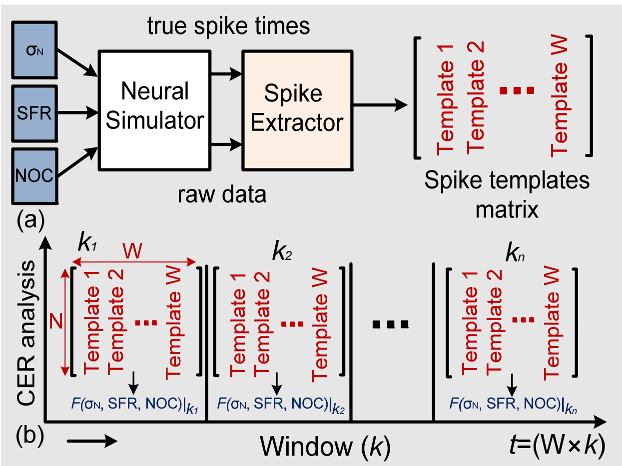

Fig. 3. (a) Signal-processing chain used to evaluate feature extraction algorithms classification performance. Spike extractor uses true spike times to identify the spike waveforms and it is considered as an ideal detector. The output of spike extractor is spike templates matrix with W templates. (b) The spike template matrix changes over the data segments $(k_1 \cdots k_n)$. The matrix generation is the function of input factors to the simulator (e.g. $F(\sigma_N, SFR, NOC|_{k_1})$) where SFR and NOC are spike firing rate a and number of clusters respectively.

III RESULTS AND DISCUSSION

This section quantifies the classification error (CER) for k-means clustering [16] in the feature extraction methods as a result of the increase of the noise level (σ_N), varying the NOC and the similarity between the spike mean waveforms. Fig. 4 illustrates the CER variations of the dictionary-based feature extractors $\{SL\varphi_{H_h(k)}, SL\varphi_{ETF(k)} \text{ and } SL\varphi_{Bern(k)}\}$ when the data stream specifications are set to $k = \{1 \cdots 100\}$ and $W = \{105\}$. CER is defined as:

$$CER = (1 - CA_{CC}) \, . \tag{2}$$

Where CA_{CC} is the number of truly assigned feature vectors over the total number of feature vectors. Each window considers random spike templates selection and noise distribution generation using the defined σ_N and NOC NOC to emulate the recording channel variations over time $(time = (k_1 \cdots k_n) \times W)$.

In Fig. 4, for $NOC = 6$, $SL\varphi_{Bern(k)}$ shows higher noise and spike template similarity sensitivity compared with the $SL\varphi_{ETF(k)}$ and $SL\varphi_{H_h(k)}$. This is because the constructed dictionary based on RBM has only 1 and 0 elements used in subspace synthesis which makes it less efficient compared to Hadamard and ETF. For instance, at $\{20 < k < 40\}$ for $\sigma_N = 0.2$ in Fig. 4(d), $SL\varphi_{Bern(k)}$ is not capable of projecting the randomly selected spike waveforms into low dimensional feature space which results in higher CER.

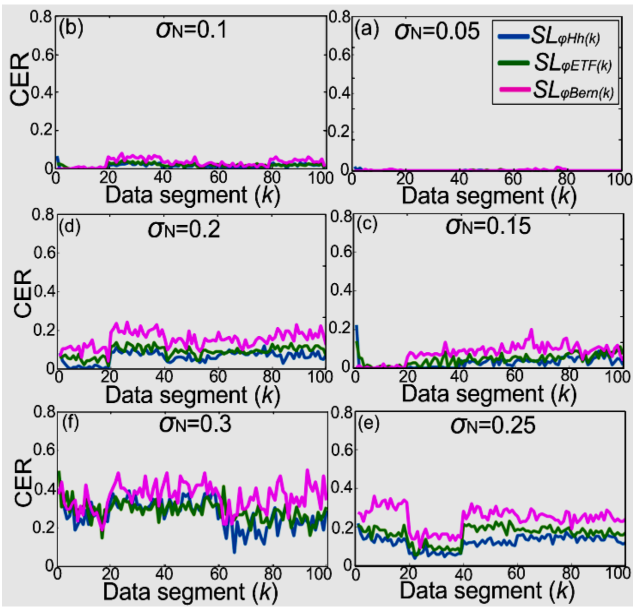

Fig. 4. Comparison of classification error (CER) between the feature extraction methods as a function of noise level $\sigma_N = \{0.05, \cdots, 0.3\}$ when NOC is set to 6. The CER of each method is the average of five runs across the data segments $k = (1 \cdots 100)$. Six spike waveforms with different degrees of similarity are randomly chosen from the designed spike library in each segment and fed to the sorting chain for proper examination of the feature extraction methods. R = 6 is chosen for CER analysis.

The $SL\varphi_{ETF(k)}$ and $SL\varphi_{H_h(k)}$ show more robust behavior when they are faced with wide range of variations generated in the data stream, for noise $\sigma_N = \{0.05, 0.1, 0.15\ and\ 0.2\}$ and similarity between chosen spike waveforms. The similarity indices (*SIs*) between the spike waveforms are defined as a range in the neural simulator and in this experiment is set to $(0.5 < SIs < 0.85)$. The results in Fig. 4 reveal that dictionary-based feature extractors provide robust classification at $NOC = (6)$ and $\sigma_N = \{0.05, \cdots, 0.3\}$. In addition to quantifying CER variations, classification can be displayed where two features with the most deviation from normality (Feature I and Feature II) are chosen for 2D projection of clusters as shown in Fig. 5.

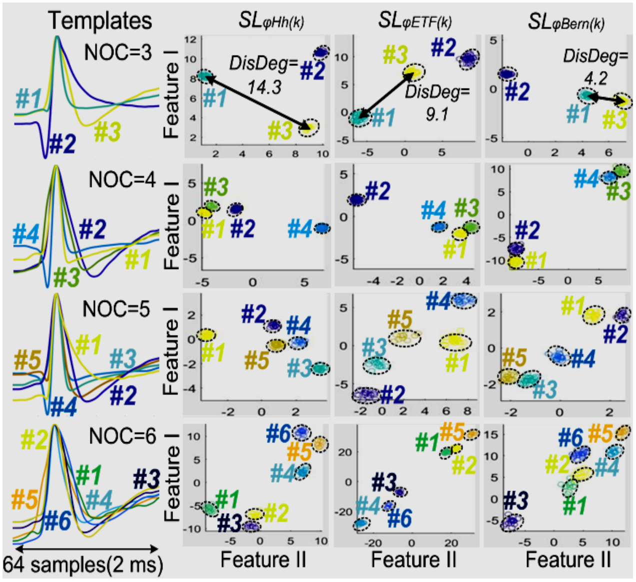

Fig. 5. 2D projection of spike waveforms (64 samples extracted per spike) using the two features with the most deviation from normality (Feature I and Feature II). The test set-up in this figure includes two elements in data generation, $NOC = \{3, 4, 5\ and\ 6\}$ and $\sigma_N = 0.15$. The columns show 2D projection of the clusters using $SL\varphi_{H_h(k)}$, $SL\varphi_{ETF(k)}$ and $SL\varphi_{Bern(k)}$. The projected clusters are colour-coded and numbered according to the simulator ground truth.

The display is designed to show the separation quality when recording channel parameters including $NOC = \{3, 4, 5\ and\ 6\}$ and similarity index between the mean waveforms at $\sigma_N = 0.15$. Mean spike waveforms are shown in the first column of Fig. 5 and the rest of columns from second to the fourth show the 2D projection of clusters of $SL\varphi_{H_h(k)}$, $SL\varphi_{ETF(k)}$ and $SL\varphi_{Bern(k)}$ respectively. The *AvgDisDeg* which is the average 'discrimination degree *DisDeg*' between all possible cluster combinations demonstrates that the dictionary-based feature extractors provide good separation strength over the range $NOC = \{3, 4, 5\ and\ 6\}$. *DisDeg* is the ratio of intercluster distance to intracluster distance, defined as *DisDeg=(inter/intra)*. The *inter* is the distance between two clusters and *intra* is the radius of the cluster. The constructed dictionaries are low-cost and capable of handling the wide range of variations generated in the data stream. For example, ETF has an overall classification error of almost 0.06 (0r 6%), over ten runs consists of 100 data segments $k = (1 \cdots 100)$ which is 2.49% higher than a conventional such as uPCA [17], but it has 327 times lower computational complexity estimated based on $Complexity = N_{add/sub} + 10 \times N_{mul/div}$ [7], where $N_{add/sub}$ is the number of additions (or subtractions) and $N_{mul/div}$ is the number of multiplications (or divisions). In uPCA the projection matrix is updated according to the subspace changes. The projection matrix $\varphi(k)$ is updated at every segment k using the standard PCA algorithm.

IV. CONCLUSION

Construction of three types of dictionary, the Hadamard, ETF and RB were explained for implementing an adaptive subspace learning algorithm. These dictionaries used in subspace learning are compatible with implantable deigns hardware restrictions, they only use $\{-1, 0\ or\ 1\}$ arrays in subspace synthesis and avoid multiplications in learning and streaming phases. The potential size of the arrays generated in the dictionary construction offer great potential in learning optimal (linear or non-linear) subspaces (e.g. 64 (N = spike length) × 144 in ETF). Constructed dictionaries were used in tracking and learning true subspaces of neural spike data. Using the K-means algorithm they delivered superior classification performance of spike waveforms. The constructed dictionaries provide flexibility in the subspace learning process and offer reliable classification within the ranges $NOC = \{3, 4, 5, 6\}$ and $\sigma_N = \{0.05, 0.15, \cdots, 0.3\}$.

REFERENCES

[1] Ponce F. A., Asaad W. F., Foote K. D., Anderson W. S., Rees Cosgrove G., Baltuch G. H., et al. "Bilateral deep brain stimulation of the fornix for Alzheimer's disease: surgical safety in the advance trial. J. Neurosurg. 125, pp. 75–84, 2016.

[2] A. Mohammed, M. Zamani, R. Bayford, and A. Demosthenous, "Toward on-demand deep brain stimulation using Online Parkinson's disease prediction driven by dynamic detection," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 25, no. 12, pp. 2441–2452, Dec. 2017.

[3] J. D. Weiland sand M. S. Humayun, "Intraocular retinal prosthesis," IEEE Eng. Med. Biol. Mag., vol. 25, no. 5, pp. 60–66, Sep./Oct. 2006.

[4] T. W. Berger *et al.*, "Restoring lost cognitive function: Hippocampalcortical neural prostheses," IEEE Eng. Med. Biol. Mag., vol. 24, no. 5, pp. 30–44, Sep./Oct. 2005.

[5] M. Capogrosso et al., “A brain-spine interface alleviating gait deficits after spinal cord injury in primates,” Nature, vol. 539, pp. 284–288, Nov. 2016.

[6] S. Micera, J. Carpaneto, and S. Raspopovic, “Control of hand prostheses using peripheral information,” IEEE Rev. Biomed. Eng., vol. 3, pp. 48–68, Jan. 2010.

[7] M. Zamani and A. Demosthenous, “Feature extraction using extrema sampling of discrete derivatives for spike sorting in implantable upper limb neural prostheses,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 22, no. 4, pp. 716–726, Jul. 2014.

[8] M. Zamani, D. Jiang, and A. Demosthenous, “An adaptive neural spike processor with embedded active learning for improved unsupervised sorting accuracy,” IEEE Trans. Biomed. Circuits Syst., vol. 12, no. 3, pp. 665–676, Jun. 2018.

[9] M. Zamani, and A. Demosthenous, “Dimensionality reduction using asynchronous sampling of first derivative features for real-time and computationally efficient neural spike sorting,” Proc. ICECS, Abu Dhabi, United Arab Emirates, Dec. 2013, pp. 237–240.

[10] J. Sokolić, M. Zamani, A. Demosthenous and M. R. D. Rodrigues, "A feature design framework for hardware efficient neural spike sorting," 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, 2015, pp. 1516-1519.

[11] M. Zamani, J. Sokolic, D. Jiang, F. Renna, M. Rodrigues and A. Demosthenous, “Accurate, very low computational complexity spike sorting using unsupervised matched subspace learning,” IEEE Trans Biomed. Circuits Syst., doi: 10.1109/TBCAS.2020.2969910

[12] J. Seberry, B. JWysocki, and T. A. Wysocki, “On some applications of Hadamard matrices,” Metrika, vol. 62, nos. 2–3, pp. 221–239, 2005.

[13] M. Fickus, D. G. Mixon, J. D. Peterson and J. Jasper, “Steiner equiangular tight frames redux,” Int. Conf. Sampling Theory and Applications (SampTA), 2015 , Washington, DC, 2015, pp. 347-351.

[14] C. J. Colbourn and R. Mathon, “Steiner systems,” in CRC Handbook of Combinatorial Designs, C. J. Colbourn and J. H. Dinitz, Eds. Boca Raton, FL, USA: CRC Press, 2007, pp. 102–110.

[15] V. Rojkova and M. Kantardzic, “Feature extraction using random matrix theory approach,” in Machine Learning and Applications, 2007. ICMLA 2007. Sixth International Conference on, Dec 2007, pp. 410–416.

[16] D. J. Bora and A. K. Gupta, “Effect of different distance measures on the performance of K-Means algorithm: An experimental study in Matlab,” Int. J. Computer Science and Information Technologies, vol. 5, no. 2, pp. 2501-2506, 2014.

[17] H. Abdi and L. J. Williams, “Principal component analysis,” Wiley Interdiscip. Rev. Comput. Stat., vol. 2, pp. 433–459, 2010.

.