Lecture Notes: Singular Value Decomposition, Principal Component Analysis, and Applications

Alice Tang* **and Shonushka Sawant†**
Team 5-5
CS 189/289 Final Project T

Contents

1 Singular Value Decomposition

Blha blah blah

1.1 subsection 1

blah balh

1.1.1 subsubsection 1

blah blah

*alicetang@berkeley.edu
†shonushka.sen@berkeley.edu

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

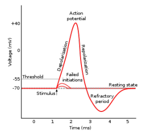


Figure 1: SVD in pseudoinverse.

2 Principal Component Analysis

blah blah

3 Minimum Norm

Consider solving a linear system of equations in the form $Xw = y$, where $X \in \mathbb{R}^{nxd}$ is your data vector with n data points and d dimensions, w is your feature weights or coefficients, and y are your observations/labels that you wish to predict or estimate.

With training data X_{train} and y_{train}, you want to be able to come up with a solution for an estimate for w such that the error or residual $\|Xw - y\|_2^2$ is minimized.

3.1 4 scenerios

There are 4 scenarios in which we can determine a solution for w:

1. X is square ($n = d$) and full rank
2. X is tall or over determined ($n > d$) and full rank.
3. X is wide or under determined ($n < d$) and full rank.
4. X is rectangular ($n \neq d$) and not full rank.

For (1.), to find w we can simply invert X to get our solution:

$$w = X^{-1}y$$

Note that $X^{-1}X = XX^{-1} = I_{nxn}$.

For (2.), we want to solve a linear regression problem where our estimate or prediction is of the form $\hat{y} = Xw$, where for each point we have $\hat{y}_i = x_i^T w$. We can try to find an approximate solution for w that will minimize the error or residual between $\hat{y}$ and y: $\min_w \|Xw - y\|_2^2$. This will be our least squares solution.

$$w = (X^T X)^{-1} X^T y$$

Note that $(X^T X)^{-1} X^T X = I_{nxn}$

For (3.), our problem $Xw = y$ has infinite solutions. In such a scenario, we are interested in the minimum-norm solution, so we can reformulate our problem:

$$\min_w \|w\| \text{ s.t. } Xw = y$$

Generally, these type of problems are useful for constrained optimization and control problems, which you may have encountered in other engineering courses.

All of our solutions to this problem will have the form $\{w | Xw = y\} = \{w_r + w_n | w_r \in R(X), w_n \in N(X)\}$, where $R(X)$ is the range of the columns of X and $N(X)$ is the null space of the columns of X.

The least norm solution will be:

$$w = X^T (XX^T)^{-1} y$$

To see that this is the least norm solution, let us consider another solution z, so we have $y = Xz = Xw$. Then $X(z - w) = 0$. Then we see

$$(z - w)^T w = (z - w)^T X^T (XX^T)^{-1} y = (X(z - w))^T (XX^T)^{-1} y = 0$$

So

$$\|z\|^2 = \|z + w - w\|^2 = \|w\|^2 + \|z - w\|^2 \geq \|w\|^2$$

Note that cross terms cancel. From this, we see that our least norm solution has the smallest norm.

3.2 SVD Perspective

We will now see that we can solve the systems of linear equations using an SVD perspective to give us a way to create a form of an inverse for our solution. This inverse is called the Moore Penrose Pseudo-Inverse.

3.2.1 Moore Penrose Pseudo-Inverse

When solving linear least squares problems in the form $Xw = y$ (X data matrix, w coefficients, y observations to predict). Often, X is not square or directly invertible.

The Moore-Penrose pseudoinverse X^+ gives us a generalization of the inverse matrix that can help us solve linear least squares solutions, $w = X^+ y$.

A computationally simple and accurate of computing the pseudoinverse involves using the singular value decomposition. If $X = USV^T$, then the pseudo-inverse would be

$$X^+ = VS^+U^T$$

where S^+ of a diagonal matrix of singular values S is obtained by taking the reciprocal of the nonzero diagonal entries, then transposing that matrix $(S^{-1})^T$.

3.2.2 Solving linear equation with the Pseudoinverse

The pseudo-inverse works for the overdetermined and underdermined system of equations case. In Figure 2, we can see how we can use SVD to derive the Moore-Penrose Pseudoinverse to solve systems of linear equations

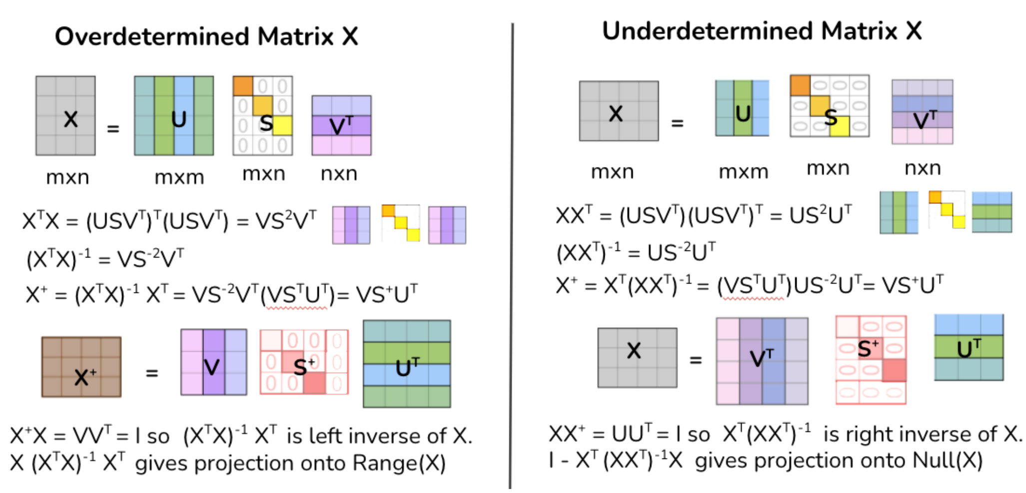


Figure 2: SVD form of the pseudoinverse for tall and wide matrices.

The following table summarizes the block form of the pseudo-inverse for all 4 scenarios listed above.

	row m, col n, rank r	$\boldsymbol{X}^{+}$	=	$\boldsymbol{V}$	$\boldsymbol{S}^{+}$	$\boldsymbol{U}$
1.	$m = n = r$	$\boldsymbol{X}^{-1}$	=	$\begin{bmatrix} \boldsymbol{V}_R \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{S}^{-1} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{U}_R^T \end{bmatrix}$
2.	$m > n, n = r$	$\boldsymbol{X}^{+}$ $\boldsymbol{X}^{-L}$	=	$\begin{bmatrix} \boldsymbol{V}_R \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{S}^{-1} & 0 \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{U}_R^T \\ \boldsymbol{U}_N^T \end{bmatrix}$
3.	$m = r, m < n$	$\boldsymbol{X}^{+}$ $\boldsymbol{X}^{-R}$	=	$\begin{bmatrix} \boldsymbol{V}_R & \boldsymbol{V}_N \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{S}^{-1} \\ 0 \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{U}_R^T \end{bmatrix}$
4.	$m \neq n \neq r$	$\boldsymbol{X}^{+}$	=	$\begin{bmatrix} \boldsymbol{V}_R & \boldsymbol{V}_N \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{S}^{-1} & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{U}_R^T \\ \boldsymbol{U}_N^T \end{bmatrix}$

3.2.3 Deriving Minimum Norm Solution Using Langrangian Multipliers

Let us now device the minimum norm solution for underdetermined equations using Lagrange Multipliers.

Our problem is formulated as:

$$\min_{w} \|w\|_2$$
$$\text{s.t.} Xw = y$$

Let us introduce Lagrange Multipliers:

$$L(w, \lambda) = w^T w + \lambda^T (Xw - y)$$

Taking derivatives:

$$\nabla_w L = 2w + X^T \lambda = 0 \Rightarrow w = -\frac{X^T \lambda}{2}$$
$$\nabla_\lambda L = Xw - y = 0$$

Substituting w from the first equation into the 2nd equation, we get:

$$Xw - y = -X \frac{X^T \lambda}{2} - y = 0 \Rightarrow \lambda = -2(XX^T)^{-1} y$$

And therefore we plug back into the first equation to get:

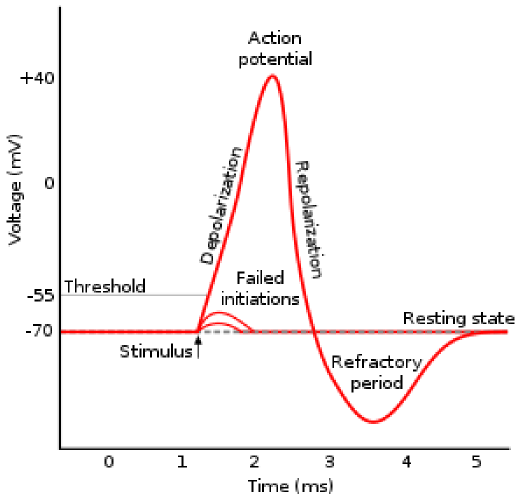
$$w = \frac{X^T \lambda}{2} = (X^T)(XX^T)^{-1} y$$

which is our minimum norm solution.

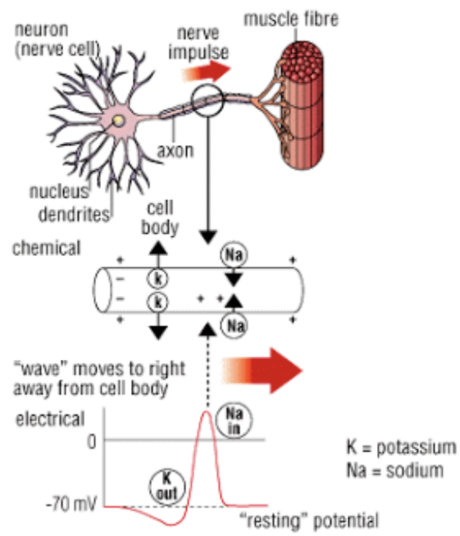
3.2.4 Extension: General Norm Optimization

4 Introduction to Brain Computer Interface

Brain Computer Interface (BCI) refers to the idea of connecting the human brain with an external device (e.g. computer, robot). Successful integration can be applied for various purposes such as research, augmenting human abilities, controlling devices through thought, or treating neurological disorders.



(a) The form of the change in membrane voltage during an action potential.



(b) Propogation of an action potential down an axon.

Figure 3: Neuron Action Potentials.

Generally in the process of implementing a BCI, a device will be used to record electrical activity from the brain, which can be placed superficially on the head or implanted in the brain through surgery. These signals will need to be processed and understood in order to extract meaningful information. We will give a basic overview of basic neural signal processing workflow, and how dimensionality reduction can be useful.

4.1 Electrophysiology Basics

Neurons conduct electrical signals in the brain via action potentials (AP). These occur when the electrochemical gradient across the cell membranes reaches a level that causes ion movement, generating a 'current' as the neuron propogates this signal across axons. Neurotransmitters, a type of chemical signal, will then be released to neighboring neurons. With enough neurotransmitter release, ion flow across neighboring neurons will activate another action potential and pass on the signal around the brain.

These electrical impulses can be picked up with sensors. Figure 3a shows us the form of an action potential, and Figure 3b shows how these action potentials propogate through neurons.

4.2 Neuronal Recordings and Applications

Electrical impulses from neurons can be picked up by electrodes, although the type and quality of the signal will depend upon many factors. At a small scale, it is possible to record from a single neuron. Usually in humans, recordings are done via extracellular recordings, where the electrodes are placed directly on the brain (called ECoG, or electrocorticography). These recordings usually pick up an ensemble of neuron signals, but some single neuron recordings can possibly be extracted. At an even higher and non-invasive level, electrical signals can be picked up by electrocorticography (EEG), which involved recording from the skin. These signals are usually noisy and give average electrical activity in a large region of the brain. Usually, there can be high levels features that can be extracted, such as time, frequency, space, and phase.

For recordings at a level where single neuron spikes can be picked up, one technique widely used is spike detection and sorting. From ensemble waveforms, spikes can be extracted, and then classified based upon the waveform characteristics of the spike to be associated with individual neurons or independent spiking events. Knowing what class a spike comes from is useful for various applications by then looking at spike frequency. By classifying and bringing meaning to the neuronal signals picked up by the sensors, we can then use that to extract meaningful commands to help us control a computer or device! This is the appeal of brain computer interface!

4.3 Spike Sorting

With extracellular recordings, usually the resolution is at a level where we can extract signals from nearby individual neurons. Let us walk through the step of processing this signal:

1. Filtering: The first step after obtaining a recording is to perform filtering in order to extract the frequencies of interest, and remove noise. Removing noise is an important step to extract meaningful information from our signal. There are various techniques for filtering, but frequencies of interest should also be done into consideration of likely frequencies associated with the task of interest.
2. Spike Detection: With a cleaned signal, we want to extract parts of the signal may be associated with a single neuron. Usually this signals are louder than the surrounding noise. With a strong clean signal, simple peak detection methods work fine. For more complicated signals, algorithms such as nonlinear energy operator (NEO) attempts to identify peaks using the energy of the signal.
3. Waveform Extraction: With peaks identified, the waveforms contributing to those peaks can then be identified to represent the shape of the neuronal spike at that time. Usually a timeframe before and after the highest peak point are used as features to represent the waveform.
4. Waveform Classification Using Dimensionality Reduction: With the waveform extracted, it is not an unsupervised clustering problem to identify what waveforms belong to which neurons. Waveforms can be projected into a lower dimensional space using PCA as a dimensionality reduction algorithm. Then various methods can be used to identify clusters. With new signals and new waveforms, the waveform can be projected into the PCA space and classified based upon the most likely cluster that the waveform belongs to. This is the basis of spike sorting.

Figure 4 gives a pictoral overview of the steps outlines above.

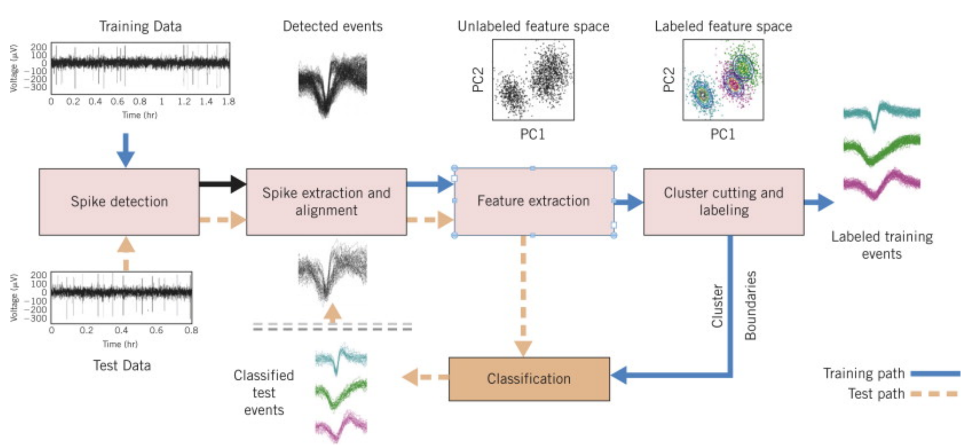


Figure 4: General Spike Sorting Pipeline.