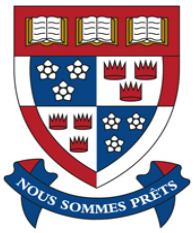

Simon Fraser University
Faculty of Statistics & Actuarial Science
Burnaby, British Columbia

Functional Independent Component Analysis

Written By:
Barinder Thind

Supervised By:
Dr. Cao

Contents

Thank you to Dr. Cao for being a great supervisor.

1 Introduction

This write-up was done as a pre-cursor to my masters of science thesis. It was done under the supervision of Dr. Cao - a Professor in the Department of Statistics at Simon Fraser University. The thesis will be built upon the ideas presented here and will likely echo many of the presented sections. This document will also be used as evidence of my work done for the KEY Big Data USRA.

2 Independent Component Analysis

Much of what is written here is borrowed from *Aapo Hyvarinen*. All the plots have been reproduced in **R** for the purposes of understanding. The language has been altered for the purposes of approachability.

2.1 Motivation

Imagine that are are trying to discern what a single individual is saying from a microphone recording. The issue is that there are 2 microphones recording and 2 people speaking, simultaneously. How would you go about separating what each individual is saying from the other? Independent Component Analysis is an approach that seems to be highly effective at solving such a problem.

2.2 The Basics of the Model

A latent variable approach is taken; that is to say, we will consider the independent components (e.g. the speakers in the example above) to be random variables that have values which are not directly observed. Let these random variables be defined by the vector $\mathbf{s}$[1].

The observed information will be defined by the vector $\mathbf{x}$. For clarity, this would be like the information obtained from the microphones in the motivation example. Lastly, we have what is often referred to as the "Mixing Matrix" - defined as $\mathbf{A}$. This matrix contains values on various factors that distort the original independent components. For example, the above problem, this could be distance or quality of microphone which leads to some proportion of the independent components making up an observed mixture. This matrix, $\mathbf{A}$, is also unknown.

The observed mixtures, $\mathbf{x}$, are defined to be linear combinations of all the independent components. For example, the first mixture would be defined as:

$$x_1 = a_{11}s_1 + a_{12}s_2 + ... + a_{1n}s_n$$

For clarity, the mixtures in the given problem would be random samples at various times, t, of the random variable X.

Alternatively, we can use vector notation to state the model as follows[2]:

$$\mathbf{x} = \mathbf{As}$$

The ultimate goal of ICA is to estimate $\mathbf{A}$ and $\mathbf{s}$ using only the information given in $\mathbf{x}$. If we had known $\mathbf{A}$, this would be a trivial problem as $\mathbf{Wx}$ would give us $\mathbf{s}$ where $\mathbf{W}$ is the inverse of $\mathbf{A}$. One final note: we are going to assume 0 mean for the observed values. If this is not the case, subtracting the sample mean will give us this result if desired.

[1] Which is to say, s_1 = the data on what one of the individuals was saying if we had that information

[2] Note that $\mathbf{s}$ refers to a column vector whereas $\mathbf{s}^T$ would refer to a row vector

2.3 Assumptions & Notes

Some important assumptions need to be made before we can proceed and are listed as follows:

- The independent components, **s**, must have non-Gaussian distributions. A demonstration of why this must be the case will be available in section 2.5.
- The mixing matrix must be square.

We may also add a noise term to the model to represent problems that could arise in the observed mixtures. For simplicity sake, they will be omitted. It is also fruitful to note that if we knew the underlying distributions, other approaches could be taken to solve this problem such as the ones presented by [placeholder].

A couple of ambiguities of note are:

1. It is not possible to determine the variance of the independent components. Since both **s** and **A** are unknown, a scalar multiplication of **A** could account for changes in magnitude of **s**. It is useful then, to fix variance at 1.
2. We cannot determine the order of the independent components. This can be demonstrated using a permutation matrix as follows:

$$\begin{aligned}\mathbf{x} &= \mathbf{A}\mathbf{s}, \\ &= \mathbf{A}\mathbf{P}\mathbf{P}^{-1}\mathbf{s} \\ &= \underbrace{\mathbf{A}\mathbf{P}}_{\text{New Mixing Matrix}} \overbrace{\mathbf{P}^{-1}\mathbf{s}}^{\text{Reordered Independent Components}}\end{aligned}$$

For ambiguity two, it is apparent that we could just rearrange the order of the independent components and just identify a new mixing matrix to get the same separation that we would without the permutation. Lastly, we cannot determine the sign of the extracted independent component.

2.4 ICA Illustration

In order to showcase an example of ICA, consider independent components with the following uniform distributions:

$$f(s_1) = \begin{cases} \frac{1}{2\sqrt{3}} & |s_1| \leq \sqrt{3} \\ 0 & \textit{Otherwise} \end{cases}$$

$$f(s_2) = \begin{cases} \frac{1}{2\sqrt{3}} & |s_2| \leq \sqrt{3} \\ 0 & \textit{Otherwise} \end{cases}$$

Since one of the key components of ICA is that these latent random variables must be independent, the joint distribution will be bivariate uniform and geometrically, will be a cube formed on the support of s_1 and s_2. This can be seen in figure 1.

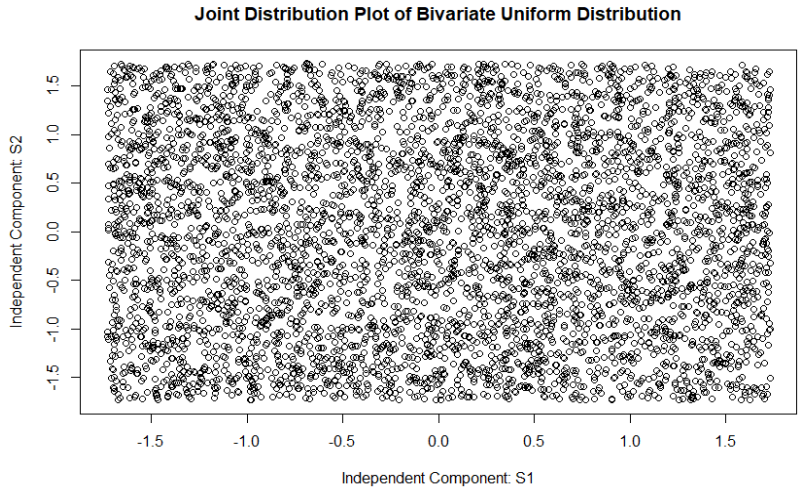

Figure 1: Surface of the Joint Density of s_1 and s_2

For example's sake, consider the following mixing matrix:

$$\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$$

Using this mixing matrix, we can obtain the mixtures, x_1 and x_2[3]. Now, looking at the joint distribution of both of these random variables (x_1, x_2), we can see that a parallelogram is formed as seen in in figure 2:

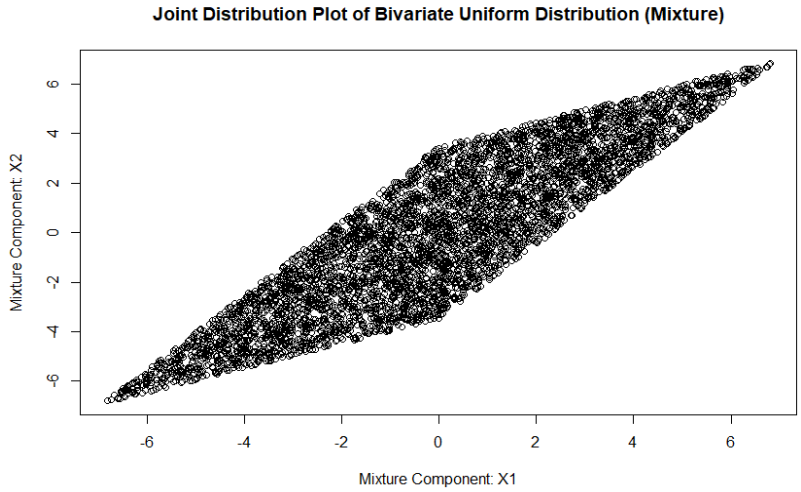

Figure 2: Surface of the Joint Density of x_1 and x_2

What is interesting to note here is that the mixture is no longer independent. In fact, taking the maximum of x_1, we observe that x_1 can only be one unique value. This was not the case for the original components, $\mathbf{s}$. This implies that we could find a solution to the ICA problem (figuring out $\mathbf{A}$) by figuring out the edges of the joint distribution of x_1 and x_2. However, note that this is only true for the bivariate uniform distribution thus far - more general approach would be needed to estimate for the ICA model.

[3] In a practical situation, we would observe these

2.5 Notes On Independence

Often, it can be useful to constrain the ICA model such that the independent components are uncorrelated. This reduces the amount of free parameters. Also, considering the example given in section 2.4, we can now look at why the Gaussian distribution poses problems. In the above example, it was clear that a solution could be found from the joint distribution of x_1 and x_2 however, let us consider the following orthogonal matrix:

$$\mathbf{A} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

It can be seen that the determinant is 1 and that the set of row vectors and the set of column vectors have a dot product of 0, respectively. Now, consider two guassian independent random variables: s_1 and s_2. Their joint distribution is defined as:

$$f(s_1, s_2) = \begin{cases} \frac{e^{(\frac{-s_1^2 - s_2^2}{2})}}{2\pi} & \forall \ s_1, s_2 \\ 0 & Otherwise \end{cases}$$

Using the orthogonal matrix, we can obtain our mixtures, x_1 and x_2. In figure 3, we can see that, because the underlying components are Gaussian, there seems to be no solution available in this situation (at least one similar to the one discerned from the bivariate uniform case).

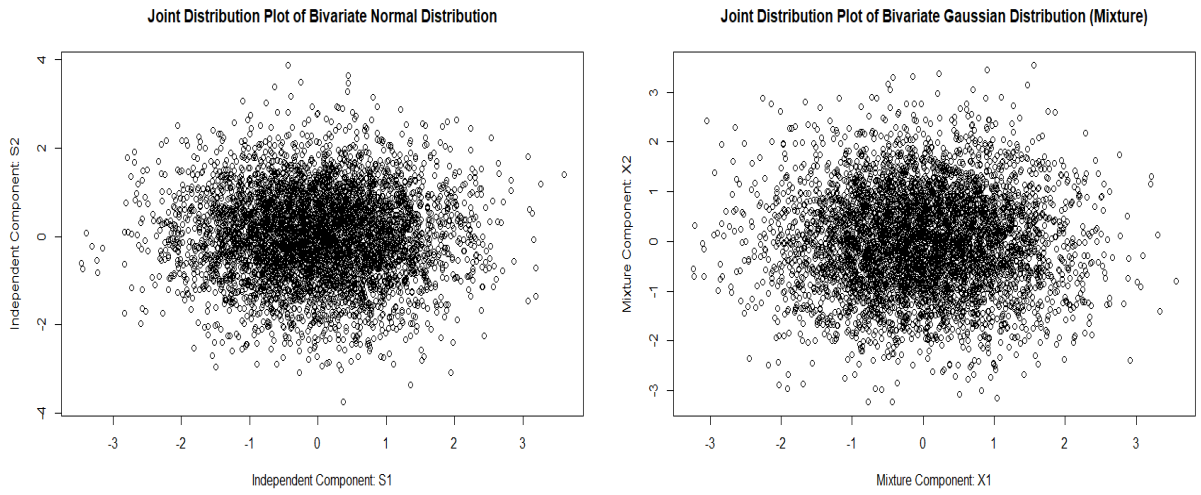

Figure 3: Surfaces of the joint Gaussian distributions

There are no edges here that allow for the previously proposed solution. Hence, solutions of this case seem only plausible up until the orthogonal mixing matrix case.

2.6 Principles of ICA

Before going into more details, we need to consider the Central Limit Theorem (CLT). In short, the CLT asserts that the sum of independent random variables is more Gaussian than the individual random variables themselves. This is more clearly true for random variables which are not normal to begin with.

Let us assume that the mixture is distributed according to the ICA model: it is a linear combination of independent components. Let us also assume that **x** has the same distribution. Then, if we wanted to find one of the independent components, say y[4], then we could write the equation to be solved as:

$$s_i = y = w^T x = \sum_i w_i x_i$$

In this example, you could think of w^T as one row of the mixing matrix, **A**. In order to proceed, we will need to use a change of variables: $\mathbf{z} = \mathbf{A}^T\mathbf{w}$. Now, going back to our previous equation, and substituting for **x**, we get:

$$\begin{aligned}\mathbf{y} &= \mathbf{w}^T\mathbf{x}, \\ &= \mathbf{w}^T\mathbf{A}\mathbf{s} \\ &= \mathbf{z}^T\mathbf{s}\end{aligned}$$

So, we observe that y is a linear combination of the independent components, **s**[5], i.e. $y = s_1z_1 + s_2z_2 + ... + s_nz_n$. Thus, using CLT, we can conclude that $\mathbf{z}^T\mathbf{s}$ is more Gaussian than any single component belonging to **s**. This also allows to infer that non-gaussianity is maximized when all z are 0 except one. This is because, as stated earlier, CLT would imply that the linear combination of the independent components is more Gaussian.

Now, in order to make this happen, we need to let **w** be the vector that maximizes non-gaussianity. Therefore, **w** must correspond to a **z** that has only one non-zero component. That is: $\mathbf{w}^T$x = $\mathbf{z}^T$s = y = one of the independent components if **w** is defined in the correct way.

In order to maximize the non-gaussianity of: $\mathbf{w}^T\mathbf{x}$, we need consider the optimization landscape for **w**. It will have $2n$ local maxima - 2 for each independent component. The criterion can be simplified since we can restrict the space such that each maximum is uncorrelated with any other - an attribute or consequence of the independence of the components. *NEED TO LEARN MORE ABOUT THIS - PLACEHOLDER -*

2.7 Measures of Non-Gaussianity: Kurtosis

One measure of non-gaussianity is kurtosis, defined as:

$$kurt(y) = E[y^4] - 3(E[y^2]^2)$$

And, due to the assumption of unit variance, we get $kurt(y) = E[y^4] - 3$. This comes from the fact that var(y) = E(y^2) - E(y)^2 $\Rightarrow$ var(y) - E(y)^2 = 1. Kurtosis is 0 for a normal random variable[6]. Since we are assuming that the independent components are non-normal, they have a kurtosis $\neq 0$.

Using the previous change of variables, we get: $kurt(y) = kurt(z_1s_1) + kurt(z_2s_2) \Rightarrow z_1^4 kurt(s_1) + z_2^4 kurt(s_2)$. Then, using the fact that var(y) = 1, we get the following:

$$\begin{aligned}1 = E(y^2) &= E((z_1s_1 + z_2s_2)^2) \\ &= E(z_1^2s_1^2 + 2z_1z_2s_1s_2 + z_2^2s_2^2) \\ &= z_1^2 + z_2^2\end{aligned}$$

The last line coming from the fact that $E(s_1) = 0$. - PLACEHOLDER - NOT SURE ABOUT THIS -. This equation sets the optimization landscape to be a unit circle and thus, there are two maxima

[4] That is to say $s_i = y$

[5] Note that, y itself is an independent component

[6] A kurtosis > 0 is defined as super-gaussian and < 0 is sub-gaussian

achieved. This comes from the fact that kurtosis is an absolute measure, so a maximum is achieved at 1 or -1.

However, this method has a few drawbacks. Namely, it is not a robust method, it is sensitive to outliers, and depends on a few tail observations.

2.8 Measures of Non-Gaussianity: Negentropy

Another measure of non-gaussianity is rooted in information theory. First, *entropy* is defined as the degree of information that the observation of some variable could give. More technically, it is defined as:

$$\mathbf{H(y)} = -\int f(y) log f(y) dy$$

Where y is the observed random variables and $f(y)$ is the density of the random variable. An important outcome from information theory is that the Gaussian random variable has the highest entropy among all random variables with equal variance. This implies that entropy could be used as a measure of non-gaussianity.

Another measure closely related to entropy is negentropy; this is defined as: $\mathbf{J(y)} = \mathbf{H}(\mathbf{y}_{gauss})$ - $\mathbf{H(y)}$[7]. Then, we see that negentropy has the property of being non0negative and it is only 0 when the random variable y is Gaussian.

- PLACEHOLDER - NEGENTROPY IS INVARIANT FOR INVERTIBLE LINEAR TRANSFORMATION

However, measuring negentropy can be very computationally taxing (since densities may need to be approximated). Hence, we need approximations of negentropy. One important approximation is as follows:

$$\mathbf{J(y)} \approx \sum_{i=1}^{p} k_i [E[G_i(y)] - E[G_i(v)]]$$

Where k_i is some positive constant, v is a Gaussian random variable with mean 0 and unit variance, y is the random variable in question with zero-mean and unit variance, and G_i is a non-quadratic function[8].

Some empirically good choices for G are:

$$G_1(u) = \frac{1}{a_1} log(cosh a_i u)$$

$$G_2(u) = -exp(\frac{-u^2}{2})$$

- PLACEHOOLDER - SO DO WE HAVE APPROXIMATIONS OF THE FUNCTION Y USING OUR INFORMATION FROM THE MIXTURES X? I think it is one of the linear combinations of independent components.

2.9 Measures of Non-Gaussianity: Mutual Information

Mutual Information (MI) is defined as:

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y})$$

[7] The variable y_{gauss} is a Gaussian random variable such that it has the same covariance matrix as y

[8] -PLACEHOLDER- NEED TO FIGURE OUT WHY NON-QUADRATIC

MI is a natural measure of the dependence between random variables. Essentially, the $y_i's$ are those random variables in this case (the independent components) and **y** is the joint density of all independent components[9].

The mutual information is 0 if and only if the components are independent. If $\mathbf{y} = \mathbf{Wx}$, then we get:

$$I(y_1, y_2, ..., y_n) = \sum_i H(y_i) - H(x) - log|detW|$$

Now, if we constraint the y_i to be uncorrelated and of unit variance, we get that the determinant of W (the inverse of A) must be a constant. Also, for y_i of unit variance, we get that entropy and negentropy differ only by a constant, and the sign, so therefore:

$$I(y_1, ..., y_n) = C - \sum_i J(y_i)$$

Where C is a constant that does not depend on **W**. The above equations highlights the relationship between negentropy and mutual information.

2.10 Maximum Liklihood Estimation

2.11 Projection Pursuit

Projection pursuit is a technique developed in statistics to find better projections or transformations of data with respect to visualization. ICA can be seen as a variant of projection pursuit.

2.12 Pre-Processing

Before we can implement our algorithms in order to separate the mixed components, some pre-processing can be done to make the process less computationally taxing and an overall easier process. The first step, as discussed earlier, is to center the data. This would make the mixtures have zero-mean which, in turn, implies that the independent components also have a mean of 0.

Next, it helps to "whiten" the mixtures. This is a process in which we do a linear transformation of the mixtures, **x**, such that the new components are uncorrelated and their variances are 1. This process eliminates the need for some of the parameters to be estimated because we use an Eigen-Value Decomposition to get a new mixture matrix that is orthogonal. This means that the number of parameters that need to be estimated will about half as much as would be required for any other arbitrary matrix. - PLACEHOLDER: FIGURE OUT WHY-

In the example given (in Section 2.4) regarding the uniform distribution, this would mean that only 1 parameter needs to be estimated (instead of n^2) since there were two random variables. In this particular case, this would be a problem of calculating an angle. This can be seen in Figure 4 which looks like a rotation on the original independent components.

2.13 FastICA Algorithm

In practice, we need an algorithm that maximizes a given contrast function (for example, maximizes a function that is a measure of negentropy). The FastICA algorithm is one such approach.

We begin with a single independent component approach (one-unit) and then notice that this extends to the multiple unit case. Let **w** be a weight vector that is associated with a particular "artificial neuron". This vector will be updated by a learning rule as defined: find a direction or a unit vector **w** such

[9] PLACEHOLDER - NEED TO CONFIRM THIS

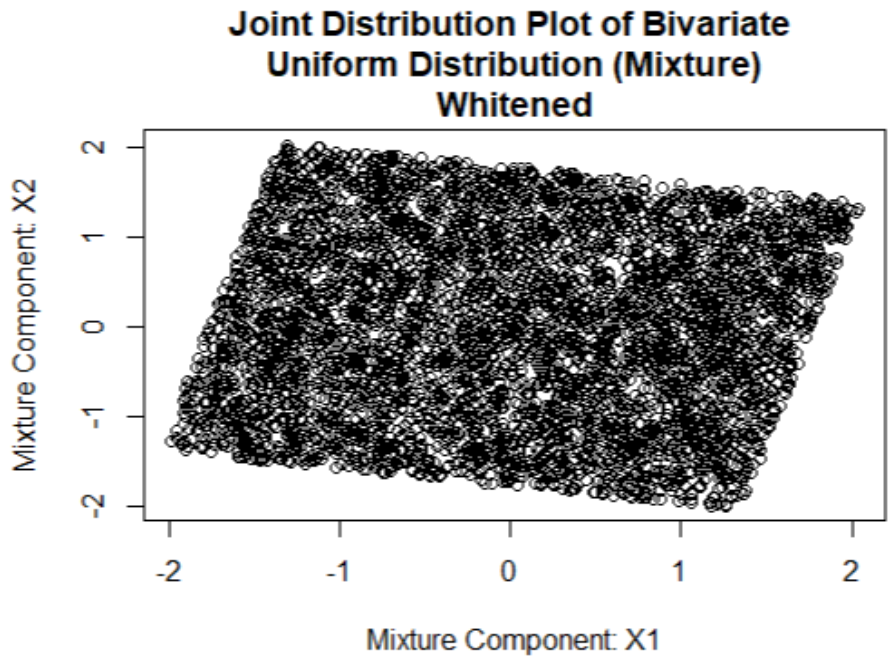

Figure 4: Surface of the Joint Density of x_1 and x_2 after Whitening

that $\mathbf{w}^T\mathbf{x}$ maximizes non-gaussianity as measured by negentropy ($J(w^T x)$). Note that, for whitened data[10], this is equivalent to constraining the norm of $\mathbf{w}$ to be 1[11].

The algorithm is then outlined as:

Algorithm 1 FastICA for One Unit

1: Choose a random $\mathbf{w}$
2: **procedure** While Not Converged
3: Define $\mathbf{w}^+ = \mathrm{E}\{xg(w^t x)\} - E\{g'(w^t x)\}w$
4: Let $\mathbf{w} = \frac{w^+}{\|w^+\|}$
5: Check Convergence: $\mathbf{w}^T\mathbf{w}^+ = \mathbf{1}$
6: Check 2.

The g is defined to be the derivative of the chosen function for the negetropy approximation defined in Section 2.8. The reason $\mathbf{w}^+$ is defined as is due to the fact that a previous - PLACEHOLDER -.

This notion can be extended for multiple units (or independent components) by making sure that the set of $\mathbf{w}$'s are decorrelated after each iteration. This is to make sure that they don't converge to the same values. One way to do this is to do each vector one-by-one but making sure that precautions are taken at each turn - PLACEHOLDER -.

The FastICA algorithm has several favourable properties. One is that the convergence is cubic rather than linear. This is helpful computationally. There is also no step-size parameter which is normally the case for most algorithms. No distribution needs to be estimated either which can be computationally taxing and serves as another element of inaccuracy. The function G can be chosen such that it is optimal for the given context. Lastly, due to the iterative nature of the component estimation, computational taxation is further relaxed.

[10] And all of this will be whitened due to the pre-processing
[11] $\|\mathbf{w}\| = 1$

2.14 Summary

In this section, I give a quick summary of all the steps outlined in this section:

1. We have a mixture and want the independent components back
2. Model is defined as: **x = As** where **x** is the mixture, **A** is the mixing matrix and **s** is vector of independent components
3. **x** is a vector containing values which are linear combinations of the independent components
4. Two basic assumptions: s's are independent and s's have non-gaussian distribution
5. We want: **s = Wx**
6. Ambiguities: a) Order b) Variance of the s's c) Sign
7. Note that the gaussian distribution only works up until an orthogonal transformation
8. The way we estimate **W** is by using some criterion that minimizes non-gaussianity
9. Three measures: Kurtosis, Negentropy, Mutual information
10. We use the FastICA algorithm to maximize non-gaussianity via a maximization of negentropy using information theory
11. The result is a set of vectors **w**, that separate out the independent components when multiplied with the observed data in **x**

3 Functional Data Analysis

3.1 Motivation

Often, we have repeated measurements of some number of individuals over a time interval, **T**. Normally, this data would be presented as discrete values however, a lot more can be gleaned from information available if the discrete data for each subject, is treated as a single *functional* observation rather than multivariate data. In particular, you may be able to take derivatives to infer how the underlying function changes over time. For example, imagine that you had height data over time for a number of children. Then, finding an underlying function could allow you to understand how growth *accelerates* or slows down at a particular age.

3.2 Statistical Properties & Definitions

In functional data analysis, the mean and variance are functions (or a vector of points) rather that single values. In fact, they are defined as:

$$\bar{x}(t) = \frac{1}{N}\sum_i x_i(t)$$

$$var_x(t) = \frac{1}{N-1}\sum_{i=1}^{N}(x_i(t) - \bar{x}(t))^2$$

These values are calculated for each particular t for which the set of observations is available. Similarly, covariances are compared between a set of two time points. In order to demonstrate this, consider the plot in figure 4 of temperatures of various Canadian cities:

Now, using the above functions, we can see that a functional mean would have a value for each of the months presented above[12]. This can be seen in the following plots:

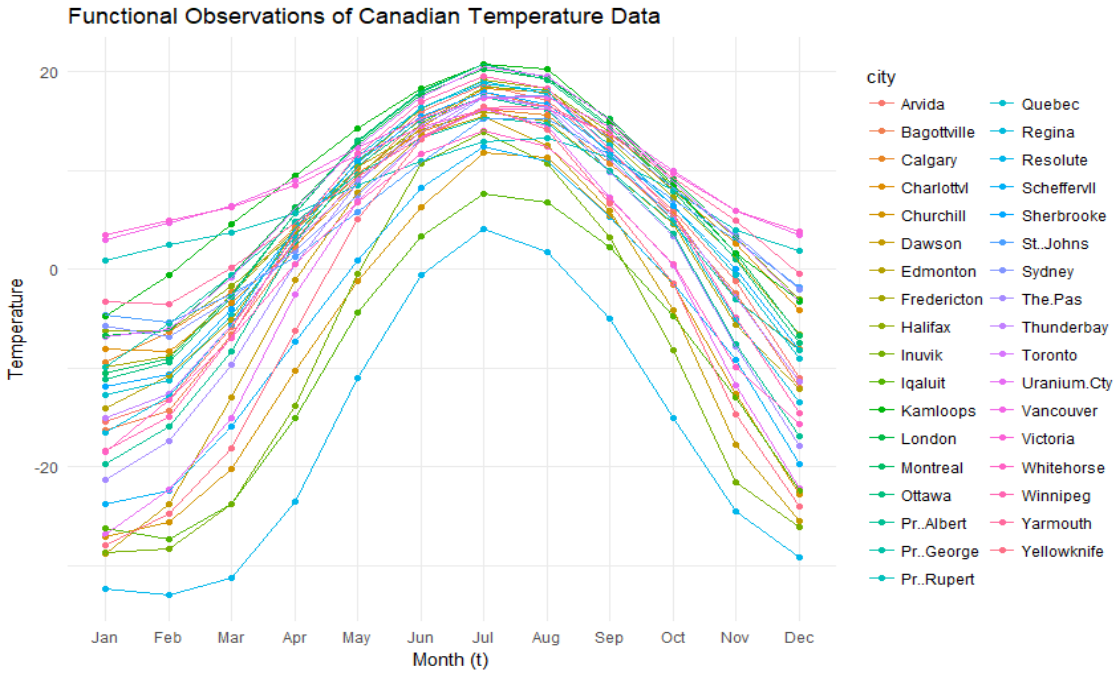

Figure 5: Temperatures for Various Canadian Cities

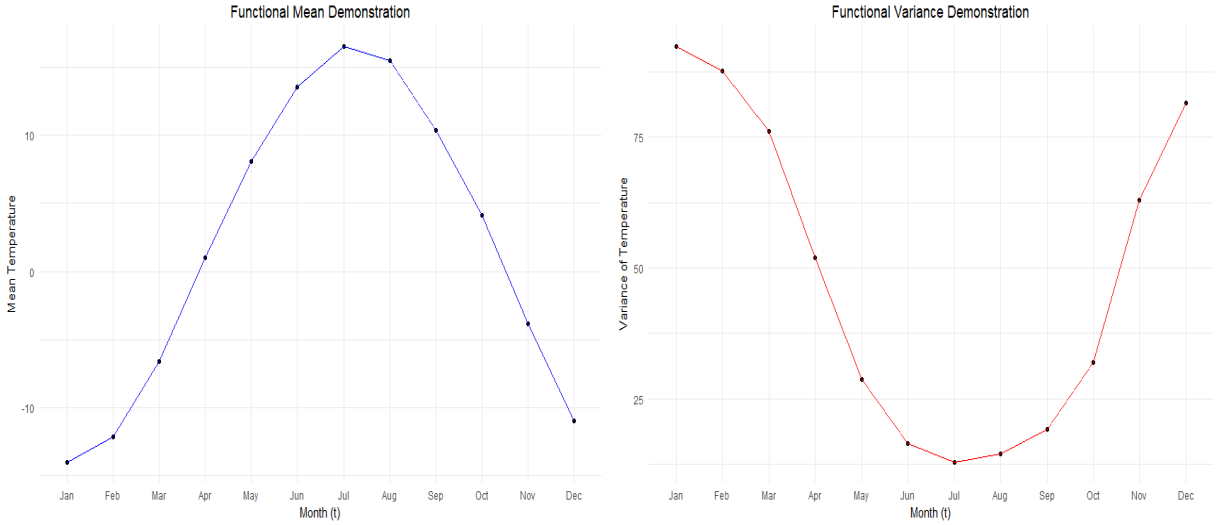

Figure 6: Mean and Variance for Functional Data

Here are a few more useful definitions when discussing functional data:

- Level - This refers to an important function value
- Crossing - This refers to a set of important function values[13]
- Amplitude - This refers to the length of *size* of a particular level
- Peak - A particular point that is distinguished because it has a specific set of attributes
- Location - This refers to the corresponding time point of a particular level
- Resolution - This refers to our ability to pin down specific events[14]

[12] As will the variance
[13] For example, all the *levels* that have the maximum value for each city
[14] That is to say, a high resolution data set would imply a higher propensity to pin down small events

Of note is the dimensionality of a function. This is defined as the amount of information required to to define it. Theoretically, a function or curve would require an infinite amount of data to define exactly (one for each t in T) however, it can often be sufficient enough to estimate the curve using *infinitely* less data than that. For example, for growth data, 12-16 pieces of information is often enough.

3.3 Turning Discrete Data into Functional Data

Basis functions are used to represent functions. Think of observed discrete data points as single entities. Functional is a reference to the intrinsic nature of the data. Functional data is also observed and recorded as a pair: (t_j, y_j) - this can be seen as a snapshot of the function at some time t[15].

DΔTΔ SENSE

Let x be the functional observation to be estimated. We assume that x has reason to exist in the given context. We also must get x to be smooth as this is what differentiates functional data from multivariate data. It also indicates that derivatives exist. For example, we can estimate the various dynamics: position, velocity, and acceleration if we had functional observations for such data. There is also another important term: signal-to-noise ratio. This is the amount of measurement error to actual usable data. In this section, we explore ways to esimate x using basis systems.

In general, we are concerned with a collection or sample of functional data, rather than one x. For example, here is what a set of functional observations might look like in matrix form:

$$x_i = (t_{ij}, y_{ij}) = \begin{bmatrix} (t_{11}, y_{11}) & (t_{12}, y_{12}) & \dots & (t_{1n}, y_{1n}) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ (t_{m1}, y_{m1}) & \dots & \dots & (t_{mn}, y_{mn}) \end{bmatrix}$$

Where each row can be thought of as a separate functional observation to be estimated. Note that sometimes the t values might be the same across all discrete data points, hence the notation can be simplified. For the rest of this section, we usually assume that we are dealing with a single x estimation[16].

There are two basic subsets in which the time component of functional data is distributed: periodically and non-periodically. Examples of periodic data might be the calendar year where t loops around after December. Non-period data might be information collected pertaining to hand movements while someone uses a pen. Also, functional data itself could be distributed over multiple domains.

Since our function fit will not be exact to the discrete data points, the assumption is made that the difference between the function fit and the discrete value is a matter of random error (corresponding to the "roughness" of the data). That is:

$$y_j = x(t_j) + \epsilon_j$$

Alternatively, we can write this in matrix notation as: $\mathbf{y = x + e}$. We can assume the function values to be fixed effects and therefore, they have 0 variance. This implies that the variance-covariance matrix of $\mathbf{y}$ is equivalent to the variance-covariance matrix of the error term denoted as $\sum_e$. The variance of the residuals varies over the argument, t[17]. Consideration also needs to be given to the correlation between neighboring residuals - normally known as the autocorrelation.

[15] Note that time is not the only domain for which functional data can exist, for example, it could also be spatial
[16] However, if the signal-to-noise ratio is low, then using information from other curves might be pertinent
[17] For example, error in height measurements changes with age

The resolving power of data is the density of the argument values, t_j relative to the amount of curvature. This is in contrast to just using n as a measure of sparsity in the dataset. Essentially, if curvature is high, then more points are required and if curvature is low, then less points are required for accuracy.

3.3.1 Basis Functions

A basis function system is a set of known functions, ϕ_k, such that the functions are independent of one another. The main property of such a system is that the set of K functions in the basis is a linear combination that can approximate any function arbitrarily well. For example, the monomial basis system includes the functions: $\{1, a, a^2, a^3, ...\}$. Functions are represented as:

$$\begin{aligned} x(t) = &= \sum_{k=1}^{K} c_k \phi_k(t) \\ &= c_1\phi_1(t) + c_2\phi_2(t) + ... + c_K\phi_K(t) \\ &= y_j - \epsilon_j \\ &= c^T\phi \end{aligned}$$

The dimension of the expansion is K. We are essentially representing the infinite-dimensional $x(t)$ in a finite space as this basis. When $K = n$, we get an exact interpolation[18]. With a smaller K and a better choice in basis, we are given the privilege of more degrees of freedom and less computational time. The behavior of the derivative of the estimated fit can be a good measure of what a good basis choice is; a good basis for x might not be a good basis for Dx.

3.3.2 Fourier Basis for Periodic Data

The Fourier basis is classic for period data. The basis is defined as:

$$\phi_0(t) = 1, \phi_{2r-1} = sin(rwt), \phi_{2r} = cos(rwt)$$

Where w is related to the period as, $p = \frac{2\phi}{w}$. Also, if t is equally spaced, then the basis is orthogonal (i.e. $\Phi^T\Phi$ = diagonal and will equal the identity matrix via a scalar multiplication). The Fast Fourier Transform (FFT) algorithm helps make this basis very computationally fast[19]. Note that this basis is not particularly well-suited for data involving potential discontinuities.

3.3.3 Spline Basis System

T

3.4 Smoothing Functional Data by Least Squares

In the previous section, we discussed how to select a basis depending on the type of data you have however, remember that a functional data object is defined as:

$$y_j = \textstyle\sum_k c_k\phi_k(t_j)$$

So how then, do we calculate the coefficient c_k? One method to do so is to use the classic ordinary least squares estimation. If we defined Φ to be the n by K matrix containing our K basis functions each evaluated at the values of t, then the estimate for c_k is:

$$\hat{c} = (\Phi^T\Phi)^{-1}\Phi^T y$$

[18] That is: $x(t_j) = y_j$

[19] With a big-O run time of $O(nlog(n))$

And therefore, $\hat{y} = \Phi(\Phi^T\Phi)^{-1}\Phi^T y$. This is called the least squares smoother in the context of functional data analysis. However, the classic model may not always be realistic for the data we have due to autocorrelation and non-stationarity of the residuals. A weighted least squares method may be used alternatively in that case. The estimate then becomes:

$$\hat{c} = (\Phi^T\Phi)^{-1}\Phi^T W y$$

Where W is a symmetric positive definite matrix. If $\sum_e$ is known then, W = $\sum_e^-$ 1. Note that the coefficient matrix is often called the projection matrix[20] and has the property of being idempotent (that is, if the matrix was defined as S, then $S = SS$).

Some commentary on least squares degrees of freedom: In most books, the degrees of freedom of a fit is the number of parameters estimated from the data that are required to define the model however, for a smooth fit, the degrees of freedom is defined as the trace of the project matrix, S. Turns out that this value is equal to K in the ordinary least squares case.

3.4.1 Bias-Variance Trade-Off

The higher the number of basis functions chosen, the closer the fit is to the discrete data (implying a lower bias) however, you run the risk of fitting to noise in this case. On the hand, the lower the value of K is, the less accurate your predictions will be and thus, you can expect your fit to be less relevant to the given context (so, lower variance but higher bias).

We want to minimize the mean squared error which is well defined as:

$$MSE[\hat{x}(t)] = bias^2[\hat{x}(t)] + var[\hat{x}(t)]$$

test

3.5 Smoothing Functional Data with a Roughness Penalty

Basis functions are good approximations if the functions picked have similar characteristics as the underlying data generating process. However, the methods discussed in the previous section have clumsy discontinuity issues when it comes to smoothing. The smoothness used with a roughness penalty makes it explicit in the minimization criterion, what it is that is being minimized.

Imagine you have a function **x(t)** to be estimated which is non-periodic. Then, remember there are two competing goals: to have a good fit and to reduce "wigglyness" or roughness. How does one measure roughness? One way is to look at the square of the second derivative at some time, t, defined as the curvature. This is a good measure because a straight line is considered to be perfectly smooth and the second derivative of such a curve is 0. Curvature is defined as:

$$PEN_m(x) = \int [D^m x(s)]^2 ds$$

Incorporating this curvature measure into our least squares minimization criteria yields:

$$PENSSE_\lambda(x|y) = [y - x(t)]^T W[y - x(t)]^2 + \lambda PEN_2(x)$$

Here, λ is a smoothing parameter that measures the trade-off between a good fit and the curvature (or variance) of the data. The larger λ is, the more smooth our fit because roughness is being penalized heavier. On the other hand, the smaller λ is the, the more closer the fit is (lower bias) to the original data points. When $\lambda = 0$, we have the weighted least squares case.

A theorem states that PENSSE is minimized with knots at the data points and using a cubic spline! Since we will have $n + 2$ basis functions in this case (edge cases), when $lambda = 0$, we will have a perfect fit. The following is the minimization of the above function:

[20] Which means the image of y has residuals orthogonal to the fit oof $\hat{y}$

$$\hat{c} = ($$

Where **R** = .

4 An Analogue: Functional Principal Component Analysis

4.1 Motivation

4.2 Placeholder

4.3 Placeholder

5 Functional Independent Component Analysis

5.1 Motivation

5.2 Placeholder

5.3 Placeholder

6 Conclusion & Future Considerations

7 References

8 Appendix

8.1 ICA Illustration in R

Using **R**, a demonstration of ICA is presented using the fastICA algorithm[21]. The *fastICA* **R** package is used. First, the independent components and mixing matrix are generated as follows[22]:

```
# Libraries
library(tidyverse)
library(fastICA)

# Source matrix
S2 <- cbind(log((1:1000)/20), cos((1:1000)/20))

# Mixing matrix
A2 <- matrix(c(0.8321, runif(1, 0, 1), rnorm(1), 0.5572), 2, 2)
```

The independent components are displayed in Figure 6. The base functions used for these independent signals were the logarithmic and cosine ones.

After some matrix algebra, we get the following mixtures:

```
# Mixed two signals
X2 <- S2 %*% A2
```

[21] http://rpubs.com/skydome20/93614

[22] Note that, there are 2 independent components for which 1000 observations have been recorded each in the form of the mixtures (i.e. the microphones in the classic example)

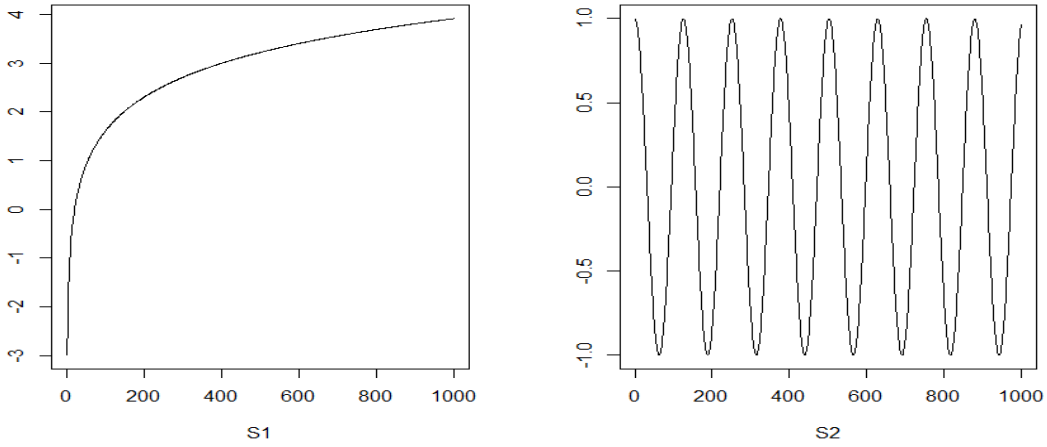

Figure 7: The Independent Components

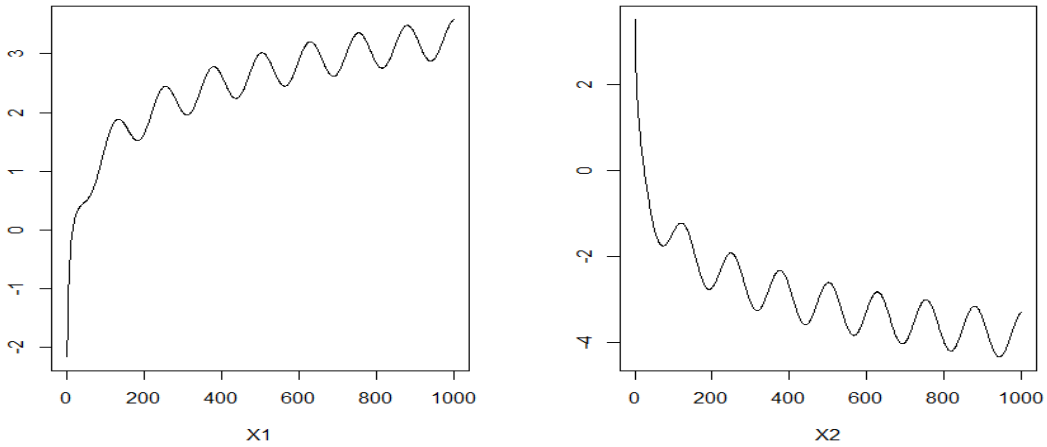

Figure 8: The Mixed Components

Next, the *fastICA* algorithm is applied and the following result is achieved after the attempted separation.

```
# ICA for extracting independent sources from mixed signals
a2 <- fastICA(X2, 2, alg.typ = "parallel", fun = "logcosh", alpha = 1,
              method = "R", row.norm = FALSE, maxit = 200,
              tol = 0.0001, verbose = TRUE)
```

An interesting note here is that the two of the ambiguities mentioned in Section 2 have manifested themselves! The order in which the components got extracted seems to have switched (as evident by the plot display order). Also, for the independent component corresponding to the logarithmic function, the sign seems to have been flipped.

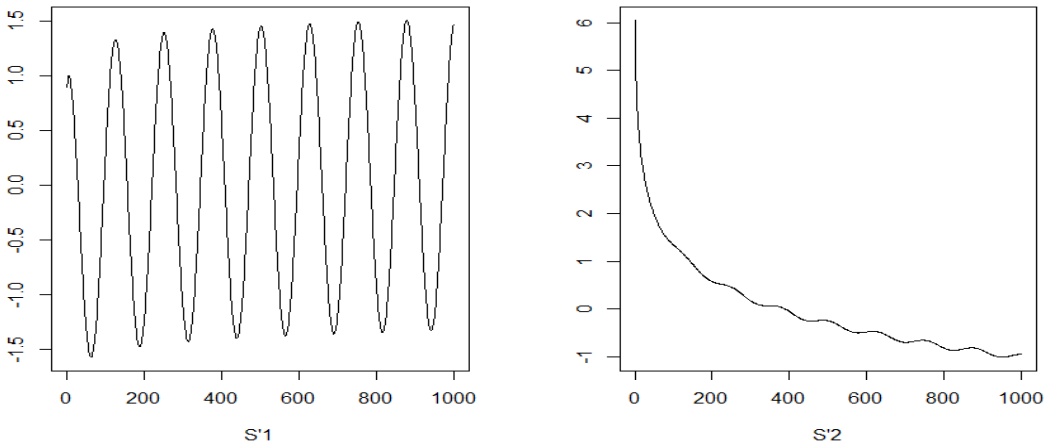

Figure 9: Separated Independent Components using *fastICA*