5-2016

Consensus Model of Families of Images using Tensor-based Fourier Analysis

Joel A. Shelton
East Tennessee State Universtiy

Consensus Model of Families of Images using Tensor-based Fourier Analysis

---

A thesis

presented to

the faculty of the Department of Mathematics

East Tennessee State University

In partial fulfillment

of the requirements for the degree

Master of Science in Mathematical Sciences

---

by

Joel Shelton

May 2016

Jeff Knisley, Ph.D.

Michele Joyner, Ph.D.

Robert Gardner, Ph.D.

Keywords: Consensus Model, Signal Processing, Tensors, Fourier Analysis

ABSTRACT

Consensus Model of Families of Images

using Tensor-based Fourier Analysis

by

Joel Shelton

A consensus model is a statistical approach that uses a family of signals or in our case, a family of images to generate a predictive model. In this thesis, we consider a family of images that are represented as tensors. In particular, our images are $(2, 0)$-tensors. The consensus model is produced by utilizing the quantum Fourier transform of a family of images as tensors to transform images to images. We write a quantum Fourier transform in the numerical computation library for Python, known as Theano to produce the consensus spectrum. From the consensus spectrum, we produce the consensus model via the inverse quantum Fourier transform. Our method seeks to improve upon the phase reconstruction problem when transforming images to images under a 2-dimensional consensus model by considering images as $(2, 0)$-tensors.

ACKNOWLEDGMENTS

I would like to thank my parents for their love and support throughout my college career. This will all pay off one day, I promise. I'm honored to have such great friends that have encouraged me to continue moving forward with my education, while offering enjoyable philosophical conversations along the way. I would also like to thank Dr. Jeff R. Knisley for his insight, wit and kindness. He has provided me with the "big picture" motivation that I've personally needed to understand advanced math concepts. My committee members, Dr. Michele Joyner and Dr. Robert Garnder, have been of great help and inspiration throughout the duration of my thesis and during my time here at the wonderful Mathematics and Statistics Department at East Tennessee State University. This department offers an excellent learning experience for students with their one on one help and dedication to the success of their students. I've truly enjoyed it.

TABLE OF CONTENTS

ABSTRACT 2

ACKNOWLEDGMENTS 4

LIST OF FIGURES 7

1 INTRODUCTION AND BACKGROUND 8

2 MATHEMATICAL BACKGROUND 10

2.1 A Brief Review of Linear Algebra 10

2.2 Multilinear Algebra and Tensors 11

2.3 Singular Value Decomposition 18

2.4 The Phase Problem 19

2.5 Fourier Transform 20

3 IMAGE PROCESSING AND CONSENSUS MODELING 24

3.1 Fundamentals of Image Processing 24

3.2 The Quantum Fourier Transform 27

3.3 Foundations and Goal of Consensus Modeling 33

3.4 Andrew Young's Consensus Model 34

4 IMPLEMENTATION OF THE QUANTUM FOURIER TRANSFORM FOR A CONSENSUS MODEL OF FAMILIES OF IMAGES 36

4.1 Implementation of Our Model 36

4.2 Results from Our Model 38

5 POSSIBLE FUTURE WORK 41

BIBLIOGRAPHY 42

APPENDIX: Python Code 45

VITA 49

LIST OF FIGURES

1 Input Image of Old Man 20

2 Reconstructed Image of Distored Old Man under FFT2 20

3 Pumpkin Man 25

4 Our main "Pumpkin Man" image 38

5 Differing phase of our family of "Pumpkin Man" images 39

6 Image coefficients in the orthonormal basis 39

7 Reconstruction of "Pumpkin Man" 40

1 INTRODUCTION AND BACKGROUND

In a previous thesis, Andrew Young used signal processing techniques to obtain a consensus model for a family of one dimensional EEG signals [18]. In this thesis, we extend some of Mr. Young's techniques to families of two dimensional images. Young's model works well for one-dimensional signals. However, his model would fail – no longer be a predictive model – for two or higher-dimensional models. In other words, his model would not be able to reconstruct the phase from the original signal. Phase is always an issue for dimension $D \geq 2$ models. What if we could improve phase related issues by considering another approach?

In our approach, we redefine the problem to allow for a tensorial structure. By using this tensorial approach, we seek to produce a predictive consensus model for two-dimensional images as tensors under the quantum Fourier transform (QFT). Furthermore, when using the inverse quantum Fourier transform (iQFT), one should see an improvement in the phase between the original image and the reconstructed image. Finally, the improvement is seen in our consensus model.

The phase reconstruction problem or image distortion [3] is always an issue when dealing with two-dimensional or higher-dimensional image processing, because the phase is not always well-defined under transformations, processing or any model dealing with images of these sort. In this thesis, we seek to improve upon the distortion problems in two-dimensional image processing by considering images as tensors. Representing images as tensors allow for the phase to be well-defined for two-dimensional images. This method will be seen throughout this thesis. We first present an overview. We note that we use concepts from several fields of study such as multilinear algebra,

image processing, bioinformatics and Fourier analysis.

The QFT is special because its properties allow us to unitarily transform a family of images $\mathcal{X}$ to a family of images $\mathcal{X}$. That is, $QFT : \mathcal{X} \to \mathcal{X}$ in a way that preserves their structure and geometry. In our case, the family of images is represented as $(2, 0)$-tensors and defined in a way to reduce the image distortion under the QFT. Furthermore, the inverse quantum Fourier transform iQFT will also take a family images to a family of images, $iQFT : \mathcal{X} \to \mathcal{X}$. This is implemented as a two-dimensional QFT. We use this two-dimensional QFT to produce what is known as a *consensus model* for tensor representations of images.

However, before moving into developing this useful QFT to generate our consensus model, we require a strong mathematical introduction to the concepts needed for developing this particular model for image processing. This thesis is structured in the following way: Chapter 2 consists of the mathematical background needed for understanding tensors and Fourier analysis. Chapter 3 is an elementary introduction to image processing and consensus modeling. Chapter 4 is our implementation process in Python. Finally, chapter 5 is the potential future development of our problem and expansion of our consensus model in image processing.

2 MATHEMATICAL BACKGROUND

We begin with a review of some important concepts from linear algebra, which is useful for our discussion of *multilinear algebra*. It should be noted that we are assuming the reader has a rudimentary understanding of linear algebra. We review some basic definitions from linear algebra, because of how often they are referenced in this thesis.

2.1 A Brief Review of Linear Algebra

We begin our review with some notation, before looking at formal definitions. Let V be a real vector space. Suppose $\{b_1, \ldots, b_n\}$ is a subset of elements or *vectors* of V. Also, consider $\{\alpha_1, \ldots, \alpha_n\}$ as the scalars in $\mathbb{R}$. We use this notation, along with additional notation, once we begin defining new objects.

Definition 2.1 *The vectors $\{b_1, \ldots, b_n\}$ in V are linearly independent if*

$$\alpha_1 b_1 + \cdots + \alpha_n b_n = 0$$

implies that $\alpha_1 = 0, \ldots, \alpha_n = 0$.

Definition 2.2 *The vectors $\{b_1, \ldots, b_n\}$ in V form a basis of V if the following are true [9]:*

1. *The vectors are linearly independent.*

2. *The vectors span V, which is to say that every vector in V can be written as a linear combination of $\{b_1, ..., b_n\}$.*

Example 2.3 *Let $V = \mathbb{R}^n$. Then $\delta_1 = (1, 0, 0, \ldots, 0), \delta_2 = (0, 1, 0, \ldots, 0), \ldots, \delta_n = (0, 0, 0, \ldots, 0, 1)$ is a basis for $\mathbb{R}^n$. In fact, this is called the standard basis for $\mathbb{R}^n$. We note that this basis is used extensively in this paper.*

Recall, that a *linear transformation*, $f : V \to V$ is a transformation which satisfies the property

$$f(\alpha v + \beta w) = \alpha f(v) + \beta f(w),$$

where $\alpha, \beta \in \mathbb{R}$ and $v, w \in V$ [6]. It's common to say that f is linear or that f is a linear map, which just means that f is a linear transformation.

2.2 Multilinear Algebra and Tensors

Tensors were first introduced at the end of the nineteenth century [4]. Since, the arrival of tensors, they have proven useful in physics, engineering, differential calculus, algebra, geometry, and computer science. In this thesis, we consider images to be tensors. We begin by discussing what tensors are and how tensors will be used in the duration of this thesis. First, we must introduce multilinear algebra.

Definition 2.4 *Let V be a real vector space. A linear functional on V is a map $L : V \to \mathbb{R}$ if for every $\alpha, \beta \in \mathbb{R}$ and for every $v, w \in V$, we get the following:*

$$L(\alpha v + \beta w) = \alpha L(v) + \beta L(w).$$

Example 2.5 *Let $x = [x_1, \ldots x_n]^T$ be a set of vectors in $\mathbb{R}^n$, which is represented as column vectors. Then any linear functional say L, there exists a row vector $[a_1 \ \ldots, \ a_n]$ such that*

$$L(x) = [a_1 \ \ldots, \ a_n] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = a_1 x_1 + \cdots + a_n x_n$$

We also define $V^* = \{L : V \to \mathbb{R} \mid L(\alpha v + \beta w) = \alpha L(v) + \beta L(w)\}$ to be the *dual space* of V. The components of a linear functional L are said to be *covariant*. That implies that the components will change by L if the basis for a given vector also changes by L. A linear functional is known as a *covariant tensor* or a $(1,0)$-*tensor*, denoted by T_1^0. This is also called a *rank*-1 *tensor*. In the above definition, we think of the elements of V as columns and the elements of V^* as rows. Once we look at tensor spaces, this will become more clear. Let us now expand our knowledge of functionals to include both *bilinear* and *multilinear* functionals.

Definition 2.6 *A bilinear functional on V is a map $G : V \times V \to \mathbb{R}$ such that*

$$G(u, \tau v + \mu w) = \tau G(u, v) + \mu G(u, w)$$

$$G(\tau u + \mu w, u) = \tau G(v, u) + \mu G(w, u)$$

for every $\tau, \mu \in \mathbb{R}$ and for every $u, v, w \in V$.

The definition of a bilinear functional extends naturally to that of multi-linear functions.

Definition 2.7 *A multilinear functional [2] is a map $T : V_1 \times V_2 \times \cdots \times V_n \to \mathbb{R}$ which is linear with respect to each variable, where $V_j, j = 1, \ldots, n$ are real vector spaces.*

We say that the multilinear functional T is a covariant $(0,n)$-*tensor*, denoted by T_n^0. Using the definition of multilinear functional, we begin developing our understanding of tensors to (m,n)-*tensors* and the rank structure of tensors.

Definition 2.8 *A* tensor *of type* (m,n) *is a multilinear mapping of the form* $L_m^n : V \times \ldots \times V \times V^* \times \ldots \times V^* \to \mathbb{R}$, *where there are* n *copies of* V *and* m *copies of* V^*.

Definition 2.9 *The rank of an arbitary* (m,n)*-tensor* T, *denoted* $R = rank(T)$, *is the minimal number of rank-*1 *tensors that yield* T *in a linear combination [5].*

Let $V = \mathbb{R}^k$ and denote $v \in V$ with respect to the standard basis by $v = \langle v^1, v^2, ..., v^k \rangle$. Similarly, $v^* \in V^*$ is denoted $\langle v_1^*, \ldots, v_k^* \rangle$. An elementary *Riesz representation theorem* [1] says that if $T_m^n \in T_m^n(V)$, then there exists coefficients $T_{j_1,\ldots,j_n}^{i_1,\ldots,i_n} \in \mathbb{R}$ such that

$$T_m^n(v_1, \ldots, v_n, v_1^*, \ldots, v_m^*) = \sum_{i_1,\ldots,i_n,j_1,\ldots,j_m=1}^{k} T_{j_1,\ldots,j_m}^{i_1,\ldots,i_n} v^{j_1} \ldots v^{j_m} v_{i_1}^* \ldots v_{i_n}^*.$$

It follows that $T_1(V) = T_1^0(V) = V^{**} \cong V$. Consequently, it also follows that $T^1(V) = T_0^1(V) = V^*$. We study monochrome images in this thesis which are type $(2,0)$ tensors T_2. Color images are type $(3,0)$ tensors, but neither $(2,0)$ or $(3,0)$ tensors are "matrices" in the usual sense. Indeed, consider the $(1,1)$ tensor

$$\begin{aligned} L(v, v^*) &= \sum_{i,j=1}^{k} T_j^i v^j v_i^* \\ &= [v_1^*, \ldots, v_k^*] \begin{bmatrix} \sum T_j^i v^j \\ \vdots \\ \sum T_j^i v^j \\ \sum T_j^i v^j \end{bmatrix}. \end{aligned}$$

Then T^i_j maps $v \in V \to w \in V$, which is a linear transformation. One could say that linear algebra is the study of T^1_1 tensors, but these are not $(2,0)$ tensors. This will be important later when we use column vectors of tensors to model families of images. Since we have introduced tensors and multilinear algebras, it is time to look at the tensor product.

The *tensor product* of two real vector spaces V and W is the vector space of bilinear mappings of the form $L : V^* \times W^* \to \mathbb{R}$. The set of all (m, n) tensors over a vector space V is denoted $T^n_m(V)$, and are said to have rank $m + n$. The tensor product of vector spaces V and W is denoted $V \otimes W$ [6]. However, we also want to consider the formal definition and existence of the tensor product, as it will allow us to define the tensor product of two vectors.

Definition 2.10 *The tensor product $V \otimes W$ is defined in the following way. Suppose $S = V \times W$ is a set [16]. Then consider E to be a vector space on S. This means an element of E is represented by the linear combination,*

$$\sum_{i=1}^{m} \alpha_i(v_i, w_i),$$

where $w_i \in W, v_i \in V$ and $\alpha_i \in \mathbb{R}$. If $F \subset E$ is the smallest subspace containing all linear combinations, then it is either of the following forms:

$$(\alpha v_1 + v_2, w) - \alpha(v_1, w) - (v_2, w)$$

or

$$(v_1 \alpha w_1 + w_2) - \alpha(v, w_1) - (v, w_2).$$

We define $V \otimes W = E/F$. We write $v \otimes w$ for $Im(v, w) \in V \otimes W$. Then elements of

$V \otimes W$ are formally expressed as

$$\sum_{i=1}^{m} \alpha_i (v_i \otimes w_i).$$

Let us now introduce a highly important concept, namely, *the Kronecker delta function.*

Definition 2.11 *The j^{th} Kronecker delta function is a function of the form*

$$\delta_j(k) = \begin{cases} 1 & \text{if } k = j \\ \\ 0 & \text{if } k \neq j. \end{cases}$$

Example 2.12 *Let $V = \mathbb{R}^k$. Consider the following:*

$$x = [\delta_1 | \delta_2 | \dots | \delta_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

It follows that x takes the form:

$$\begin{aligned} x &= \langle x_1, x_2, \dots, x_n \rangle \\ &= x_1\delta_1 + x_2\delta_2 + \cdots + x_n\delta_n. \end{aligned}$$

Theorem 2.13 *If $\{e_i\}$ is a basis for V and $\{f_j\}$ is a basis for W, then $\{e_i \otimes w_j\}$ is a basis for $V \otimes W$.*

Let us look at an idea which is fundamental for our understanding of images. This idea is known as a *tensor space* or an *image space.*

Definition 2.14 *The tensor space $T_m^n(v)$ is the vector space of all real-valued, multilinear functionals on the Cartesian product of n copies of V with m copies of V^*.*

In general, we obtain linear transformations on a tensor spaces from tensor products on the tensor factors of that space. Let's look at an example.

Example 2.15 *Suppose $A : \mathbb{R}^m \to \mathbb{R}^m$ and $B : \mathbb{R}^n \to \mathbb{R}^n$ are linear transformations given coefficient wise by $A = [a_{ij}]$ and $B = [b_{kl}]$. The action of A on $\mathbb{R}^m$ is given by*

$$Ax = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \Big[\sum a_{ij} x_j \Big]_{i=1}^{n}.$$

Furthermore, $[x_1, ..., x_n]^T$ implies that $x = x_1\delta_1 + \cdots + x_n\delta_n$. Hence, $x_1 = \langle x_1, \delta_1 \rangle, \dots, x_n = \langle x_n, \delta_n \rangle$. The result is

$$Ax = \Big[\sum a_{ij} x_j \Big]_{i=1}^{n} = \Big(\sum a_{1j} x_j \Big)\delta_1 + \cdots + \Big(\sum a_{nj} x_j \Big)\delta_n$$

More compactly,

$$Ax = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_j \delta_i.$$

Here, we can say the dyadic product of A and B is defined

$$\begin{aligned} (A \otimes B)(x \otimes y) &= Ax \otimes By \\ &= \sum_{i,j=1}^{n} \sum_{k,l=1}^{m} a_{ij} x_j b_{kl} y_l \delta_i \otimes \delta_k \\ &= \sum_{i,k=1}^{n} \sum_{j,l=1}^{m} a_{ij} b_{kl} x_j y_l \delta_{ik} \end{aligned}$$

where $\delta_{ij} = \delta_i \otimes \delta_j$ is 1 at index i, j and 0 elsewhere, and $y = [y_l]_{j=1}^{m}$. In general, if $t \in T_2^0$,

$$(A \otimes B)t = \sum_{i,j=1}^{n} \sum_{k,l=1}^{m} a_{ij} b_{kl} t_{jl} \delta_{ik}.$$

Now, let $B(V)$ be the set of bounded linear transformations, $B : V \to V$. Suppose $\Gamma \in B(\mathbb{R}^m \otimes \mathbb{R}^n)$. Then

$$\Gamma t = \Big[\sum_{ij}^{m} \sum_{kl}^{n} \gamma_{ijkl} t_{jl} \Big].$$

If $e_{kl} = b_k \otimes c_l$, then

$$\begin{aligned} e_{kl} = b_k \otimes c_l &= \sum_s \sum_t b_{ks} \delta_s \otimes c_{tl} \delta_l \\ &= \sum_s \sum_t b_{ks} c_t \delta_s \otimes \delta_l \\ &= \sum_s \sum_t b_{ks} c_{lt} \delta_{st}. \end{aligned}$$

We write the inner product of $\langle e_{ij}, e_{kl} \rangle$ as

$$\begin{aligned} \Big\langle \sum_k \sum_l b_{ik} c_{jl} \delta_{kl}, \sum_s \sum_t b_{ks} c_{lt} \delta_{st} \Big\rangle &= \sum_{kl} \sum_{st} b_{ik} c_{jl} b_{ks} c_{lt} \langle \delta_{kl}, \delta_{st} \rangle \\ &= \sum_{kl} \sum_{st} b_{ik} c_{jl} b_{ks} c_{lt}. \end{aligned}$$

Let $s \in T_0^2$. Then $s = [s_{ij}]_{i,j=1}^{m,n}$ with respect to the standard basis

$$s = \sum_{i,j}^{m,n} s_{ij} \delta_{ij}.$$

Let $t \in T_0^2$. Then $t = [t_{kl}]_{k,l=1}^{m,n}$ with respect to the standard basis

$$t = \sum_{k,l}^{m,n} t_{kl} \delta_{kl}.$$

Now,

$$\begin{aligned} \langle s, t \rangle &= \Big\langle \sum_{i,j}^{m,n} s_{ij} \delta_{ij}, \sum_{k,l}^{m,n} t_{kl} \delta_{kl} \Big\rangle \\ &= \sum_{i,j}^{m,n} \sum_{k,l}^{m,n} s_{ij} t_{kl} \langle \delta_{ij}, \delta_{kl} \rangle \end{aligned}$$

Since, $\langle \delta_{ij}, \delta_{kl} \rangle$ is an orthonormal basis, it follows that $\langle \delta_{ij}, \delta_{kl} \rangle = 0$ when $i \neq j$ and $k \neq l$. Hence,

$$\langle s, t \rangle = \sum_{ij} s_{ij} t_{ij}.$$

Dyads are of the form $\delta_{ij} = \delta_i \otimes \delta_j$.

2.3 Singular Value Decomposition

Since we are utilizing images, it is useful to understand what a *singular value decomposition* is and how it is implemented. The singular value decomposition of a matrix is commonly abbreviated by SVD.

Consider $A = [a_{ij}]$ to be an invertible matrix. We wish to find vectors $\{v_1, \ldots, v_i\}$ in the row space $\mathbb{R}^m$, the vectors $\{w_1, \ldots, w_i\}$ in the column space $\mathbb{R}^n$ and the positive-valued scalars $\{\alpha_1, \ldots, \alpha_i\}$ so that the v_i's and w_i's are orthonormal for which the following is true:

$$Av_i = \alpha_i w_i.$$

Since A is square, we want to find orthonormal matrices say V and U, where U and V are square. Also, we want a square diagonal matrix Σ so that

$$AV = U\Sigma.$$

Since, V is orthonormal, it follows that $V^{-1} = V^T$. Hence, $A = U\Sigma V^T$. Consequently,

$$A^T = V\Sigma^T U^T.$$

From here,

$$\begin{aligned} A^TA &= V\Sigma U^{-1}U\Sigma V^T \\ &= V\Sigma^2 V^T \\ &= V\begin{bmatrix} \alpha_1 & 0 & 0 & \cdots & 0 \\ 0 & \alpha_2 & 0 & \cdots & 0 \\ 0 & 0 & \alpha_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & & \alpha_n \end{bmatrix} V^T. \end{aligned}$$

We now find V through the diagonalization of our symmetric, positive-definite matrix, A^TA, where the columns of V are the eigenvectors of A^TA and the eigenvalues of A^TA are the α_i^2'-values. To find U, we use the same process with AA^T [15].

2.4 The Phase Problem

An important topic in image processing is the phase reconstruction problem, which deals with the significance of phase and magnitude when studying a particular image [13]. The reason is that the phase and magnitude of an image are both needed to reconstruct the original image from its Fourier transform. It should be noted that neither the magnitude nor the phase of an image are sufficient for the reconstruction of an image. It is possible to reconstruct an image based on phase, but there must be a sufficient amount of zeroes in the signal [11]. In both cases, however, the original image is difficult to recognize or is of poor quality.

For the purpose of this thesis, we look at a FFT applied to images (later, as tensors). When dealing with a 1-signal (Andrew Young's model) [18] under the FFT, the phase is well-defined. This means that we reconstruct an image from its Fourier transform. However, when dealing with a 2-dimensional images under the FFT2, the

phase is not well-defined. As expected, this means the image becomes difficult to recognize or in the below example, it becomes distorted:

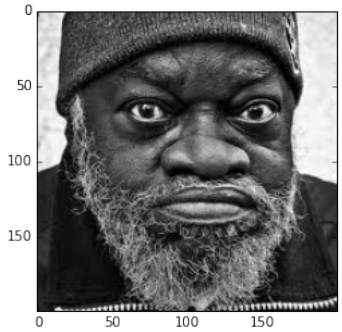

Figure 1: Input Image of Old Man

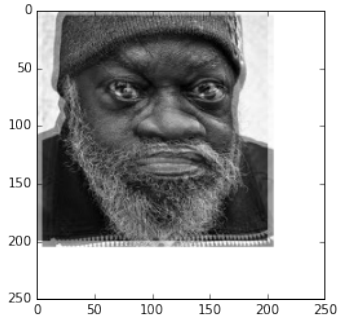

Figure 2: Reconstructed Image of Distored Old Man under FFT2

This phase distortion issue can be improved by looking at images as tensors.

2.5 Fourier Transform

The Fourier transform FT has many important applications in various fields of study. Fourier transforms are commonly used in analyzing differential equations, spectroscopy, quantum mechanics, and image processing. We utilize the FT and

its properties to the study of image processing. In particular, we apply the Fourier transform to a tensor structure, in the form of the Quantum Fourier transform QFT. Before, let us formally define the Fourier transform. We look at the Fourier transform on $\mathbb{R}^n$. Recall, that periodic functions $f \in L^1([-1,1])$ can be written using a Fourier series:

$$f(x) = \frac{a_0}{2} + \sum_{m=1}^{\infty} a_m \cos(m\pi x) + \sum_{m=1}^{\infty} b_m \sin(m\pi x).$$

Definition 2.16 *Suppose f and h are continuous signals. Let $f : \mathbb{R}^n \to \mathbb{C}$. Then the Fourier transform of f is defined as*

$$\mathscr{F}\{f\}(u) = \int_{\mathbb{R}^n} f(x) e^{-2\pi i \langle x,u \rangle} dx$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ if it exists.

There's also an *inverse Fourier transform*, which is given by the Fourier inversion theorem. We formally define it below.

Definition 2.17 *Suppose f and h are continuous signals. Let $f : \mathbb{R}^n \to \mathbb{C}$. Then the Inverse Fourier transform iFT of f is defined as*

$$\mathscr{F}^{-1}\{f\}(x) = \int_{\mathbb{R}^n} f(u) e^{2\pi i \langle x,u \rangle} du$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ if it exists.

Also, let us note the following concepts from functional analysis, which come from the existence of the Fourier transform. If $f \in L^1(\mathbb{R}^n)$, then $\mathscr{F}(f)$ exists. Also, $\mathscr{F}$ maps $L^1(\mathbb{R})$ onto $L^\infty(\mathbb{R})$. If $f \in L^1(\mathbb{R}^n) \cap L^2(\mathbb{R}^n)$, then $\mathscr{F}(f)$ and $\mathscr{F}^{-1}(f)$ exist. Furthermore, $\mathscr{F}^{-1}\mathscr{F}(f) = f$. There are cases in which there exists $f \in L^1(\mathbb{R})$ and $g \in L^2(\mathbb{R})$ for which the above do not hold. Let us consider one of these cases.

Example 2.18 *Define, for* $\frac{1}{2} < \alpha < 1$ *the function,*

$$h(x) = \begin{cases} 0 & \text{if } x < 1 \\ x^{-\alpha} & \text{if } x \geq 1 \end{cases}$$

Then $h(x) \in L^2(\mathbb{R})$ *but* $\mathscr{F}(h)$ *does not exist, because*

$$\mathscr{F}(h) = \int_1^\infty x^{-\alpha} e^{-iux} dx$$

we see that $\mathscr{F}(h)$ *does not converge absolutely.*

Now, we look at a particular property which is of great importance to the study of image processing. This is the property of the FT known as *convolution.*

Definition 2.19 *Suppose* f *and* h *are functions in* $L^1[-\infty, \infty]$. *Then the convolution of* f *and* h *is the function* $f * h$ *defined by*

$$(f * h) = \int_{-\infty}^{\infty} f(t - x)h(x)dx.$$

We also have the discrete Fourier transform DFT. This is referenced often throughout this thesis. Especially, when we introduce the Quantum Fourier transform QFT. We consider the DFT in 2-dimensions.

Definition 2.20 *Suppose* f *is a discrete signal. Let* $x = x_1 e_1 + x_2 e_2$. *Define* $f : \{0, ..., N_1 - 1\} \times \{0, ..., N_2 - 1\} \to \mathbb{C}^{n \times n}$. *The Discrete Fourier transform in* $2D$ *is defined as*

$$DFT\{f\}(u, v) = \sum_{i=0}^{N_1 - 1} \sum_{j=0}^{N_2 - 1} f(x) e^{-2\pi(iu + jv)/N}.$$

We also have an inverse discrete Fourier transform iDFT.

Definition 2.21 *Suppose f is a discrete signal. Let $x = x_1e_1 + x_2e_2$. Define $f : \{0, ..., N_1 - 1\} \times \{0, ..., N_2 - 1\} \to \mathbb{C}^{n\times n}$. The Inverse Discrete Fourier transform in $2D$ is defined as*

$$DFT^{-1}\{f\}(x) = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} f(u)e^{2\pi(iu+jv)/N}.$$

Fourier transforms are successful in the study of image processing for a number reasons. One of those reasons is the existence of a fast algorithm approach known as the *fast Fourier transform* FFT. This algorithm computes the DFT, but reduces the computational complexity of the DFT from $\mathcal{O}(n^2) \to \mathcal{O}(n \log n)$. The way this works is that the original algorithm assumed that the dimensions of the images are of the form $N = 2^k$, where $k \in \mathbb{Z}$. Then the algorithm recursively divided the even and odd elements. Hence, reducing the complexity and making the FFT incredibly useful to dealing with images. Modern FFTs allow for a dimension of any size. This is of particular interest for our implementation process in Python. Let us look at an example in $\mathbb{R}^4$.

Example 2.22 *Consider $V = \mathbb{R}^4$. The (FFT) in $\mathbb{R}^4$ is*

$$\mathscr{F} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

3 IMAGE PROCESSING AND CONSENSUS MODELING

3.1 Fundamentals of Image Processing

The development of modern technology has allowed us to manipulate a variety of *multi-dimensional* signals, which have a system of signals that range from basic circuits to high-performance computers. This makes the study of image processing of fundamental importance for the future of technological advancement. One of the ways we work with signal manipulation is through *image processing*. In image processing, we receive a signal called an image as an input and eventually, we receive another signal called an image, but this time as an output.

More advanced aspects of image processing include that of image analysis, which studies the informational properties of an image. This means that an image comes in the form of an input, while a measurement is released as an output. Both image processing and image analysis is used extensively in this thesis. First, we must define an image.

Definition 3.1 *A digital image in* $2D$ *discrete space, denoted* $a[m, n]$*, is derived from an analog image* $a(x, y)$ *in* $2D$ *continuous space by a sampling process known as digitalization.*

More specifically, the $2D$ analog image $a(x, y)$ has N-rows and M-columns [19]. When we take the intersection of the rows and columns, we obtain what is known as a *pixel*. For a discrete digital image $a[m, n]$, we have the following for the N-rows and M-columns, $m = \{0, 1, 2, \ldots, M-1\}$ and $n = \{0, 1, 2, \ldots, N-1\}$. Consider the image in Figure 3.

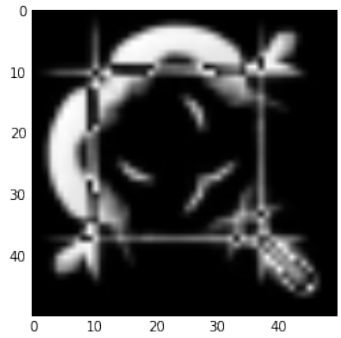

Figure 3: Pumpkin Man

A given value that is assigned to a pixel is defined as the average brightness in the pixel so that it is rounded to the nearest positive integer. In addition to the rows and columns of an image, we also have what are known as the *grey levels* of an image, denoted by L-grey levels. We often find common values for the parameters of a digital image. For N-rows, we often see the values $\{256, 512, 525, 625, 1024, 1080\}$. For M-columns, we see $\{256, 512, 768, 1024, 1920\}$ and for L-grey levels, we see $\{2, 64, 256, 1024, 4096, 16384\}$. It is very common to see that case $M = N = 2$, where $K = \{8, 9, 10, 11, 12\}$, because this case is used in fast computer algorithms such as the fast Fourier transform (FFT).

There are many tools for processing digital images. One of those is a mathematical tool known as *convolution*. We have already introduced the notion of convolution with the FT. Here, we look at the important properties of this concept for the general case, which is helpful in our understanding of image processing.

Definition 3.2 *Let a, b, c, d be arbitrary images. Then convolution is commutative if*

$$c = a \otimes b = b \otimes a.$$

Convolution is associative if

$$c = a \otimes (c \otimes d) = (a \otimes c) \otimes d = a \otimes c \otimes d.$$

Convolution is distributive if

$$c = a \otimes (b + d) = (a \otimes b) + (a \otimes d).$$

The above convolution properties hold for both discrete and continuous digital images [8]. Let's look at some examples of images in a mathematical context.

If $x \in \mathbb{R}^k \otimes \mathbb{R}^k$, then $\exists x_{ij} \in \mathbb{R} \ni$

$$x = \sum_{i=1}^{k} \sum_{j=1}^{k} x_{ij} \delta_{ij}$$

Note that, x is an array or an image, not a matrix, though it can also be written as a rectangular array as

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1k} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2k} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{k1} & x_{k2} & x_{k3} & \dots & x_{kk} \end{bmatrix}$$

Furthermore, x is a vector in $\mathbb{R}^k \otimes \mathbb{R}^k$. Thus, images are vectors in $\mathbb{R}^k \otimes \mathbb{R}^k$. Tensors have great value in the study of images, because the tensor structure allows us to obtain an orthonormal basis for arrays. In other words, we obtain an orthonormal basis for $\mathbb{R}^m \otimes \mathbb{R}^n$ as follows:

1. $b_1, \dots, b_m$ being an orthonormal basis for $\mathbb{R}^m$.

2. $c_1, \dots, c_n$ being an orthonormal basis for $\mathbb{R}^n$.

3. $e_{ij} = b_i \otimes b_j$ being an orthonormal basis for $\mathbb{R}^m \otimes \mathbb{R}^n$.

Also,

$$b_i = b_{i1}\delta_1 + \cdots + b_{im}\delta_m = \sum_k b_{ik}\delta_k.$$

and

$$c_j = \sum_l c_{jl}\delta_l.$$

Hence,

$$\begin{aligned} e_{ij} &= b_i \otimes c_j \\ &= \sum_k \sum_l b_{ik}\delta_k \otimes c_{jl}\delta_l \\ &= \sum_k \sum_l b_{ik}c_{jl}\delta_k \otimes \delta_l \\ &= \sum_k \sum_l b_{ik}c_{jl}\delta_{kl}. \end{aligned}$$

Hence, $\begin{bmatrix} b_{i1}c_{j1} & b_{i1}c_{j2} & \dots & b_{i1}c_{jn} \\ \vdots & \vdots & \vdots & \vdots \\ b_{im}c_{j1} & b_{im}c_{j2} & \dots & b_{im}c_{jn} \end{bmatrix} \in \mathbb{R}^2 \otimes \mathbb{R}^2.$

3.2 The Quantum Fourier Transform

Before we define the Quantum Fourier Transform QFT, we introduce some concepts in quantum information. These concepts will help motivate and clarify some of the subtly involved with the QFT. Quantum computing and quantum information are a developing field of interest for computer science, physics and applied mathematics. It was started by a plenary talk and subsequent paper by Richard Feyman in 1982 [7]. It was originally motivated as a means of testing Quantum Field Theory Models. Quantum computing uses principle of superposition to solve (some) deterministic non-polynomial time problems in polynomial time. Before defining the QFT, let's look at some basic concepts in quantum computing. An understanding of the

mathematical framework behind quantum information is required for understanding the QFT. This all starts with *Hilbert spaces*. Hilbert spaces are of great significance in quantum information.

Let our vector space be $V = \mathbb{C}^n$, where the n-tubles are complex numbers $(z_1, \ldots, z_n)$. In this case, the n-tubles are the vectors for V. Now, physicists use a convenient and accessible notation for an inner product. This is known as *bra-ket notation*, denoted by $\langle \cdot |$ (called a 'bra') and $| \cdot \rangle$ (called a 'ket'). Mathematically speaking, these help indicate whether an object belongs to either a row space or a column space, respectively. For instance, the inner product $|u\rangle$ and $|v\rangle$ is denoted by $\langle u|v\rangle$.

Definition 3.3 *Let u, v, w be vectors and α be a scalar in a vector space V. Then an inner product satisfies the following properties:*

1. $\langle u|v\rangle = (\langle v|u\rangle)^*$ *(Conjugate symmetry)*
2. $\langle u|v + w\rangle = \langle u|v\rangle + \langle u|w\rangle$ *(Linearity)*
3. $\langle u|v\rangle \geq 0$. *(Positive definite)*

Example 3.4 *Consider $\mathbb{C}^n$. This vector space has an inner product, defined by,*

$$\langle u|v\rangle = \sum_{j=1}^{n} u_j^* v_j = (u_1^*, \ldots, u_n^*)(v_1, \ldots, v_n)'.$$

Also, an inner product induces a *norm*, $||u|| = \sqrt{\langle u|u\rangle}$. This particular example is a Hilbert space. This brings us to formally defining a Hilbert Space.

Definition 3.5 *A Hilbert space is a vector space $\mathcal{H}$ with an inner product $\langle u|v\rangle$ such that the norm is defined by*

$$||u|| = \sqrt{\langle u|u\rangle}$$

and the space is complete in the norm.

We have already seen one example of a Hilbert space. Now, let us look at some more examples of Hilbert spaces to help illustrate exactly what spaces are Hilbert spaces.

Example 3.6 *Consider* $\mathbb{R}^n$ *with the inner product,*

$$\langle x|y\rangle = \sum_{k=1}^{n} x_k y_k.$$

This is a Hilbert space over $\mathbb{R}$*.*

Example 3.7 *An important example from analysis is that of* $L^2(\mathbb{R})$ *with the inner product,*

$$\langle f|g\rangle = \int f\overline{g}.$$

This is a Hilbert space induced by the L^2 *norm over* $\mathbb{R}$*.*

From the above definition of a Hilbert space, we also get some important definitions which utilize the concept of a Hilbert space to develop and construct other mathematical objects.

Definition 3.8 *An operator* L *on* $\mathcal{H}$

$$L : \mathcal{H} \to \mathcal{H}, \text{ is linear if}$$

the following is true:

$$L(a\,|u\rangle + b\,|v\rangle) = aL(|u\rangle) + bL(|v\rangle),$$

where $|u\rangle$*,* $|v\rangle \in \mathcal{H}$ *and* $a, b \in \mathbb{C}$*.*

Definition 3.9 *An operator L on $\mathcal{H}$ is a self-adjoint operator if $L = L^*$. In physics, this is known as a Hermitian operator.*

We note that an operator $\mathcal{U}$ is a *unitary operator* if $\mathcal{U}\mathcal{U}^* = \mathcal{U}^*\mathcal{U} = I$. These are some key definitions that follow from introducing Hilbert spaces. These definitions will be of use in later chapters.

In classical computing, computers use transistors to compute ones or zeroes, individually. However, in quantum computing, quantum computers would be able to compute ones and zeroes, simultaneously. This is known as a *superposition* of quantum states. A superposition of a quantum state is a state of information in which both ones and zeroes are computed at the same time via a quantum bit [14]. Quantum bits are very similar to the concept of a bit in classical computing. For a classical bit, we have a state of either 0 or 1. Whereas in quantum computing, we have the notation of a qubit, which has the states $|0\rangle$ and $|1\rangle$. More specifically, a qubit could also take the superposition states,

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,$$

where $\alpha, \beta \in \mathbb{C}$ and have the property

$$|\alpha|^2 + |\beta|^2 = 1$$

called *amplitudes*. That means the states of a given qubit are vectors of length 1 (called unit vectors), which belong to a complex vector space $V = \mathbb{C}$. Furthermore, the states $|0\rangle$ and $|1\rangle$ are an orthonormal basis for the space V. We call this a *computational basis state*.

In mathematics or other related fields, it is often necessary to solve problems by transforming a given problem into another type of problem so that the solution is thought to exist there or is defined in a manner to help us understand it better. This is why the study of transformations are so important to both mathematics and computer science. The field of quantum computing has led to the discovery that some transformations allow for faster computations on a quantum computer. This discovery has created an interest in the construction of fast algorithms in mathematics.

This brings us directly to the quantum Fourier transform QFT. As seen in a previous chapter, we defined important transformations e.g., the Fourier transform FT and the discrete Fourier transform DFT. These are useful for our definition of the QFT, which is a special type of DFT. In particular, the QFT is a DFT that occurs on an orthonormal basis of quantum states, $|0\rangle, \ldots, |N-1\rangle$. From here, we define the QFT as a linear operator L with an action on the basis states. We formally write the QFT as,

$$|j\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-2\pi ijk/N} |k\rangle .$$

We could also write this in bra-ket notation as,

$$\sum_{j=0}^{N-1} x_j |j\rangle \to \sum_{k=0}^{N-1} y_k |k\rangle .$$

The result is known as the *quantum Fourier transform* [10]. Continuing this idea, we define the *inverse quantum Fourier transform* (iQFT) by

$$|k\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |j\rangle .$$

In bra-ket notation as,

$$\sum_{k=0}^{N-1} y_k |k\rangle \to \sum_{j=0}^{N-1} x_j |j\rangle .$$

From our earlier look at the Kronecker delta function, we see how to construct the QFT and its inverse by using this function, and other needed objects from linear algebra, such as, the notation of a basis $\mathcal{B}$.

Consider the following example:

$$\begin{aligned} x &= [\delta_1|\delta_2|\delta_3|\delta_4]\mathcal{B}\mathcal{B}^{-1}x \\ &= [b_1|b_2|b_3|b_4]\mathcal{B}^{-1}x. \end{aligned}$$

Thus,

$$\mathcal{B}^{-1}x = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix}$$

This allows us to define the following, $x = \beta_1 b_1 + \beta_2 b_2 + \beta_3 b_3 + \beta_4 b_4$. We look at the Quantum Fourier Transform QFT. Recall, $\mathcal{F}X$ = Fourier Transform. For the QFT,

$$\mathcal{F}\begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix} = \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \hat{\beta}_3 \\ \hat{\beta}_4 \end{bmatrix}$$

All of this implies that, $\text{QFT}(x) = \hat{\beta}_1 b_1 + \hat{\beta}_2 b_2 + \hat{\beta}_3 b_3 + \hat{\beta}_4 b_4$. Also, there's an inverse QFT:

$$\mathcal{F}^{-1}\begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \hat{\beta}_3 \\ \hat{\beta}_4 \end{bmatrix} = \begin{bmatrix} \hat{\hat{\beta}}_1 \\ \hat{\hat{\beta}}_2 \\ \hat{\hat{\beta}}_3 \\ \hat{\hat{\beta}}_4 \end{bmatrix}$$

Therefore, $\text{iQFT}(x) = \hat{\hat{\beta}}_1 b_1 + \hat{\hat{\beta}}_2 b_2 + \hat{\hat{\beta}}_3 b_3 + \hat{\hat{\beta}}_4 b_4$. Now, that we have looked the definition of the Quantum Fourier transform and at how one is constructed, it is necessary to see how the QFT is used in this paper.

3.3 Foundations and Goal of Consensus Modeling

We look at a statistical approach from *bio-informatics* known as a *consensus model*. A consensus model utilizes a family of signals for a predictive model. A consensus model uses what is known as a *homological family*.

Definition 3.10 *A homological family is a family of signals (family of sequences, family of residue chains, etc) in which each signal represents an individual instantiation of an overall process.*

Some examples include, a protein expressed by a common gene that occurs in several species, mutations of a particular protein family and an (EEG) recording of the same physiological phenomena. In particular, a family of signals is homologous if each signal is sampled from the same phenomenon. Now, a consensus model uses the concept of a homological family.

Definition 3.11 *A consensus model of a homological family is a sequence (signal) of random variables (stochastic process) whose expected value at any given "time" corresponds to the "most likely occurring" value in the given homological family at a given time.*

In general, a consensus model combines statistical methods along with signal processing concepts to build a better representation of a homologous family of related signals. A consensus model is derived from a consensus spectrum, which is a spectral representation i.e., frequency or periodicity of the homological family rather than directly from the family itself. Consensus modeling is a newly developing approach

with some beneficial applications, such as, the reference Genome for a species of organisms, an (EEG), and for studying sequences of amino acids [12].

The consensus spectrum has some useful applications such as identifying certain hot spots in proteins [17]. The hot spots are found by generating a consensus spectrum from the signal of proteins. From here, the consensus model determines the "peeks" (hot spots) from the consensus spectrum. For this thesis, our signals are from a homological family of images as tensors.

3.4 Andrew Young's Consensus Model

A former graduate student at East Tennessee State University (ETSU), Andrew Young, also had a consensus model approach in his thesis "A Consensus Model for Electroencephalogram Data Via the S-Transform" [18]. In his approach, 32 signals were taken from an Electroencephalogram (EEG) recording session. This data acts as the family of signals used in his consensus approach to help produce the consensus model. He used an S-transform, which is just a special type of Fourier transform, to produce the consensus. After the consensus was produced, he used a bootstrapping process to generate the consensus spectrum in MATLAB.

Finally, a *consensus model* was produced via an inverse S-transform of the consensus spectrum. Earlier in the paper, we saw the important of phase, in particular, when looking at the phase reconstruction problem and how it is difficult to reconstruct higher dimensional images. Furthermore, this is highly important because a consensus model depends on phase. Andrew Young's model worked well for one-dimensional signals i.e., Young's model was a one-dimensional consensus model. However, his model

runs into phase issues with two-dimensional models under the FFT.

His model fails, because it is no longer a predictive model. In other words, his model is unable to reconstruct the phase from the original signal. Our approach seeks to redefine the problem, in such a way, to produce a predictive consensus model for two-dimensional images as tensors under the quantum Fourier transform QFT. Also, when we use the inverse quantum Fourier transform iQFT, we are able to see an improvement in the phase between the original image and the reconstructed image. We note that we are attempting to merely improve upon the phase related issues with higher dimensional models.

4 IMPLEMENTATION OF THE QUANTUM FOURIER TRANSFORM FOR A CONSENSUS MODEL OF FAMILIES OF IMAGES

Our implementation of the quantum Fourier transform for a consensus model of families of images seeks to improve upon the phase reconstruction issues with 2-dimensional images. In particular, this implementation process represents square images as $(2,0)$-tensors. From this point, we transform those images under the QFT to produce a consensus model. Below, we explain our implementation process and results in a strong theoretical manner. In the appendix section, our Python-based code can be viewed and used for future related consensus models.

4.1 Implementation of Our Model

We consider a collection of grayscale "Pumpkin Man" images $\mathcal{A}_1, \ldots, \mathcal{A}_m$ as $n \times n$ shaped T_2^0 tensors differing only in phase. Define $\mathcal{A} = [\mathcal{A}_1, \ldots, \mathcal{A}_m]^t$ to be a new matrix. We note that A is the direct sum $\oplus$ of m-copies of $T_2^0(n)$. This is a column vector in m-copies of $V \oplus \cdots \oplus V$.

If $W = V \oplus V^*$, then column representation is not a good model of a family of images. We require another method to build our model, which better represents our family of images. Recall linear transformations are T_1^1 tensors. This means they are a mix of V and V^*. Hence, we must treat images as $(2,0)$-tensors, because image tensors are pure tensors. We only consider an image $\mathcal{X}$ to be an $n \times n$ picture i.e., a square image. Our image $\mathcal{X}$ is represented in the SVD as $\mathcal{X} = \mathcal{U}\Sigma\mathcal{V}^T$. However, this approach is an "ad hoc" tensor approach. We want to formally represent $\mathcal{X}$ as a tensor. We want to transform the SVD approach into a tensor decomposition

approach. This is conducted in the following way. Let $\mathcal{U} = [u_1|, \ldots, |u_n]$ and $\mathcal{V} = [v_1|, \ldots, |v_n]$. Then $e_{ij} = u_i \otimes v_j$ is a basis for $\mathcal{U} \otimes \mathcal{V}$. The SVD loos like, $\mathcal{X} = \sum_i s_i u_i \otimes v_i$, where s_i are the singular values. Furthermore, $\mathcal{X} = \sum_{i,j} c_{ij} e_{ij}$, where

$$c_{ij} = \begin{cases} s_i & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

This becomes our formal representation of $\mathcal{X}$ in the e_{ij} basis i.e., we have taken the SVD approach into the tensor decomposition approach.

In matrix algebra, we have an "ad hoc" SVD decomposition of $\mathcal{A}$,

$$\begin{aligned} C = \mathcal{A}^*\mathcal{A} &= \mathcal{A}_1^*\mathcal{A}_1 + \cdots + \mathcal{A}_m^*\mathcal{A}_m \\ &= \mathcal{V}\Sigma^2\mathcal{V}^*, \end{aligned}$$

where diagonalization of C is given by $C = \mathcal{V}\Sigma^2\mathcal{V}^*$ and $\mathcal{V} = [v_1| \ldots |v_n]$ is unitary i.e., $\mathcal{V}^* = \mathcal{V}^{-1}$. Also, $\Sigma = \text{diag}(s_1 \ldots s_n)$ are the singular values of $\mathcal{A}$. Furthermore, define $u_j = \frac{1}{s_j}\mathcal{A}v_i$ for $j = 1, \ldots, r$. Then $\{u_1, \ldots, u_r\}$ is an orthonormal basis for range($\mathcal{A}$).

Let us consider our family of images to take the form $\mathcal{A}_1 \approx \mathcal{A}_2 \approx \cdots \approx \mathcal{A}_m$ differing only in phase. By the SVD decomposition,

$$\mathcal{A} = [u_1| \ldots |u_m][\Sigma_1, \ldots, \Sigma_m]^t V^t.$$

Also, our tensor coefficients with respect to $\mathcal{V}$ are of the form $\sum_j \mathcal{V}^t$, where $\mathcal{A} = u_i \sum_j \mathcal{V}^t$. Since we have our family of images (differing in phase) represented in the tensor decomposition, we apply the QFT. Hence,

$$QFT\Big(\sum_j \mathcal{V}^t\Big) = \hat{t}_j \implies t_j = QFT\big(\hat{t}_j\big).$$

Thus, A consensus spectrum $\hat{t}_c$ is obtained via bootstrapping of geometric mean of $\hat{t}_1, \ldots, \hat{t}_m$. We used a single geometric mean to get $\hat{t}_c$. Therefore, we obtain a Consensus model in the following form $\mathcal{A}_c = \text{QFT}^*(\hat{t}_c)$ and $\mathcal{A}_j = u_j\mathcal{A}_c = u_j \text{QFT}^*(\hat{t}_c)$. This explains how our Consensus model was developed. We show in the next section the results of our model through a series of images that were generated by importing our family of images in Python and constructing our consensus model via the quantum Fourier transform.

4.2 Results from Our Model

We assessed our method by considering a homologous family of "Pumpkin Man" images that differ only in phase. Notice in Figure 4 that the "Pumpkin Man" image is symmetric.

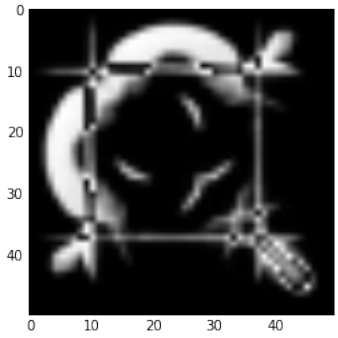

Figure 4: Our main "Pumpkin Man" image

We begin by importing the "Pumpkin Man" in Python and creating versions of it with differing phases. An example is shown in Figure 5.

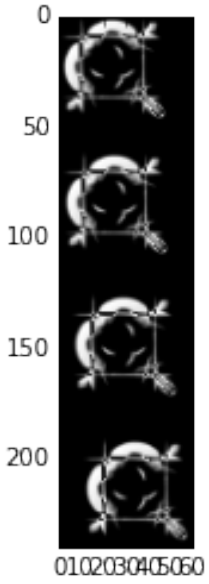

Figure 5: Differing phase of our family of "Pumpkin Man" images

Figure 6 is our family of images with respect to the tensor basis.

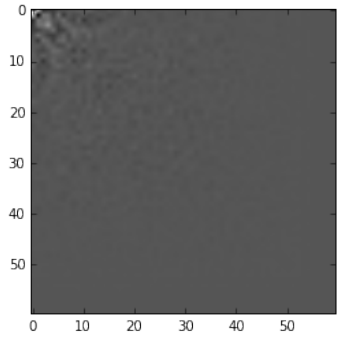

Figure 6: Image coefficients in the orthonormal basis

In Figure 7, we see the reconstruction of "Pumpkin Man." These images were produced from our implementation in Python, which can be seen in the appendices.

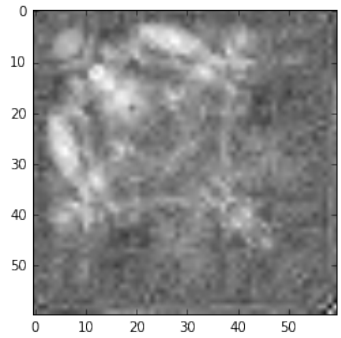

Figure 7: Reconstruction of “Pumpkin Man”

5 POSSIBLE FUTURE WORK

There are many different directions in which the ideas from this thesis could be taken. One of those is to consider the case where a family of images are not merely approximate, but to develop a consensus model for all images being the same i.e., $\mathcal{A}_1 = \mathcal{A}_2, \ldots, = \mathcal{A}_m$. This allows us to consider the model, $\mathcal{A} = \mathcal{B}\mathcal{U}\Sigma\mathcal{V}^t$ with an orthonormal basis for T_2^0 from $\mathcal{V} = [v_1|, \ldots, |v_n]$. Thus, the SVD of any $\mathcal{A}_j$ takes the form, $\mathcal{A}_j = \mathcal{U}\Sigma\mathcal{V}^t$, where $V \in \mathcal{A}^*\mathcal{A}$. Therefore, developing a Consensus model for the case of $\mathcal{A}_1 = \mathcal{A}_2, \ldots, \mathcal{A}_m$ leads to $\mathcal{A}_c = \mathcal{U}\Sigma\mathcal{V}^t$.

One could also consider non-symmetric images, non-square images or color images. However, all of these require an implementation on a cluster. The reason is that tensor based image structures consume computer memory at an exponential rate, due to their immense size. This is why we used a family of symmetric square images, because of limited computational time.

BIBLIOGRAPHY

[1] Ralph Abraham, Jerrold E Marsden, and Tudor Ratiu. *Manifolds, tensor analysis, and applications*, volume 75. Springer Science & Business Media, 2012.

[2] Ray M Bowen and Chao-Cheng Wang. *Introduction to vectors and tensors*, volume 2. Courier Corporation, 2008.

[3] Ronald Bracewell. *Fourier analysis and imaging*. Springer Science & Business Media, 2004.

[4] Pierre Comon. Tensor: a partial survey. *Signal Processing Magazine*, page 11, 2014.

[5] Lieven De Lathauwer and Bart De Moor. From matrix to tensor: Multilinear algebra and signal processing. In *Institute of Mathematics and Its Applications Conference Series*, volume 67, pages 1–16. Citeseer, 1998.

[6] John M Erdman. Elements of linear and multilinear algebra.

[7] Richard Phillips Feynman, JG Hey, and Robin W Allen. *Feynman lectures on computation*. Addison-Wesley Longman Publishing Co., Inc., 1998.

[8] Mario Hlawitschka, Julia Ebling, and Gerik Scheuermann. Convolution and fourier transform of second order tensor fields. In *Proc. Fourth IASTED Intl Conf. Visualization, Imaging, and Image Processing (VIIP04)*, pages 78–83, 2004.

[9] Ron Larson. *Elementary linear algebra*. Nelson Education, 2015.

[10] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[11] Eliyahu Osherovich, Michael Zibulevsky, and Irad Yavneh. Signal reconstruction from the modulus of its fourier transform. Technical report, Technical Report CS-2009-09, Technion (December 2008), 2008.

[12] Thomas D Schneider. Consensus sequence zen. *Applied bioinformatics*, 1(3):111, 2002.

[13] Kedarnath P Vilankar, Logesh Vasu, and Damon M Chandler. On the perception of band-limited phase distortion in natural scenes. In *IS&T/SPIE Electronic Imaging*, pages 78650C–78650C. International Society for Optics and Photonics, 2011.

[14] Yazhen Wang et al. Quantum computation and quantum information. *Statistical Science*, 27(3):373–394, 2012.

[15] David S Watkins. *Fundamentals of matrix computations*, volume 64. John Wiley & Sons, 2004.

[16] Greub Werner. Multilinear algebra, 1978.

[17] Yashpal Yadav and Sulochana Wadhwani. Determination of characteristic frequency for identification of hot spots in proteins. *International Journal of Electrical and Electronics Engineering (IJEEE)*, 1(1), 2011.

[18] Andrew Coady Young. A consensus model for electroencephalogram data via the s-transform, 2012.

[19] Ian T Young, Jan J Gerbrands, and Lucas J Van Vliet. *Fundamentals of image processing*. Delft University of Technology Delft, The Netherlands, 1998.

APPENDIX: Python Code

```
%pylab inline

import numpy
import numpy as np

np.set_printoptions(precision=5, suppress=True)


#import theano
#import theano.tensor as T
#from theano.tensor.nnet import conv

from scipy.linalg import svd, diagsvd

img = imread('Batman.bmp')[:,:,0]  #subject to change, depending on image.
img = (img + img.T) /2

m,n = img.shape; m,n

r,s = img.shape; r,s

gray()
imshow( img )

A0 = zeros((60,60), dtype=uint8)
A0[0:50,0:50] = img

A1 = zeros((60,60), dtype=uint8)
A1[1:51,1:51] = img

A2 = zeros((60,60), dtype=uint8)
A2[5:55,5:55] = img

A3 = zeros((60,60), dtype=uint8)
A3[10:,10:] = img

A = vstack( [A0,A1,A2,A3])
A.shape

U,Sigma,Vt = svd(A, full_matrices = 0)
```

```
U

Up = U[:250,:250]

matrix(U)*Up.T

matrix(Up)*diagsvd(Sigma,60,60)*Vt

matrix(Up)*Up.T

matrix(Up).I

def TensorBasis(U,V):
    m,r = U.shape
    n,s = V.shape
    assert m == r
    assert n == s

    E = np.zeros( (m,n,m,n) )
    for i in range(m):
        for j in range(n):
            eij = np.zeros((m,n))
            for k in range(m):
                for l in range(n):
                    eij[k,l] = U[k,i]*V[l,j]
            E[i,j,:,:] = eij
    return E

Onb = TensorBasis(Vt.T,Vt.T)

tij = Onb[0,0,:,:]  # = e00
sij = Onb[1,1,:,:]  # = e11

tij

def TensorDot(a,b):
    m,n = a.shape
    r,k = b.shape
    assert m==r
    assert n==k
```

```
    acc = 0
    for i in range(m):
        for j in range(n):
            acc +=  a[i,j]*b[i,j]
    return acc

TensorDot(tij,sij)

acc = np.zeros(tij.shape)

for i in range(len(Sigma)):
    acc += Sigma[i]*Onb[i,i,:,:]

ImCoeffOnb = zeros( A0.shape  )

for i in range(60):
    for j in range(60):
        ImCoeffOnb[i,j] = TensorDot(A0,Onb[i,j,:,:])

imshow(ImCoeffOnb)

QFT = fft2(ImCoeffOnb)
for i in range (10):
    for j in range(10):
        QFT += ImCoeffOnb[i,j] * Onb[i,j,:,:]

QFT

ImReCon = zeros( A0.shape )

for i in range(60):
    for j in range(60):
        ImReCon +=  ImCoeffOnb[i,j] * Onb[i,j,:,:]

imshow(ImReCon)

Comp0 = fft2(ImCoeffOnb)

imshow( abs(Comp0))

ImCoeffOnb1 = zeros( A1.shape  )
```

```
for i in range(60):
    for j in range(60):
        ImCoeffOnb1[i,j] = TensorDot(A1,Onb[i,j,:,:])

ImReCon1 = zeros( A1.shape )

for i in range(60):
    for j in range(60):
        ImReCon1 +=  ImCoeffOnb1[i,j] * Onb[i,j,:,:]

imshow(ImReCon1)

Comp1 = fft2(ImCoeffOnb1)

imshow( abs( Comp1 ))

imshow( abs(Comp0))

Comp1*Comp0

imshow(abs(_))

csm = ifft2(sqrt(Comp1*Comp0))
csm.real

ImReConC = zeros( A1.shape )

csmr = csm.real

for i in range(60):
    for j in range(60):
        ImReConC +=  csmr[i,j] * Onb[i,j,:,:]

imshow(ImReConC)
```

VITA

JOEL SHELTON

Education:	B.S. Mathematics, East Tennessee State University, Johnson City, Tennessee 2013 M.S. Mathematical Sciences, East Tennessee State University, Johnson City, Tennessee 2016
Professional Experience:	Astronomer, Bays Mountain Park and Planetarium Kingsport, Tennessee, 2011–2014 Graduate Assistant, East Tennessee State University Johnson City, Tennessee, 2014–2016 Mathematics and ACT Instructor, Sylvan Learning Center Johnson City, Tennessee, 2014–present