ON THE CHARACTERISTICS OF A DATA-DRIVEN MULTI-SCALE FRAME CONVERGENCE ALGORITHM

A Dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

by

BEVERLY K. GRUNDEN
B.A., University of Delaware, 1970
M.A., University of Delaware, 1972
M.S., Wright State University, 1998

2021
Wright State University

Wright State University
Graduate School

April 26, 2021

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Beverly K. Grunden ENTITLED On the Characteristics of a Data-Driven Multi-scale Frame Convergence Algorithm BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy.

Major Travis J. Bemrose, Ph.D.
Dissertation Co-Director

Jason Deibel, Ph.D.
Dissertation Co-Director

Ivan Medvedev, Ph.D.
Director, Interdisciplinary Applied Science
and Mathematics Ph.D. Program

Barry Milligan, Ph.D.
Vice Provost for Academic Affairs
Dean of the Graduate School

Committee on
Final Examination

Co-Advisor, Major Travis J. Bemrose, Ph.D.

Co-Advisor, Jason Deibel, Ph.D.

Qingbo Huang, Ph.D.

Steen Pedersen, Ph.D.

ABSTRACT

Grunden, Beverly K. Ph.D., Interdisciplinary Applied Science and Math Ph.D. Program, Wright State University, 2021. *On the Characteristics of a Data-Driven Multi-scale Frame Convergence Algorithm.*

In recent years, data-driven representation methods have been introduced to improve compressed sensing image reconstruction. This research explores a recently proposed algorithm that utilizes a data-driven multi-scale Parseval frame for image compression. Because a sensing matrix by itself may be insufficient to obtain a sparse representation for an image, a frame is combined with the compressed sensing matrix to increase flexibility in obtaining a sparse representation. The two-step algorithm optimizes the representation by alternating between adjusting a sparse coefficient vector and tuning a small filterbank which determines the frame. The structure of the frame and its relationship with the underlying filterbank were examined. Numerical experiments to characterize the algorithm include a search for the appropriate regularization parameters that control emphasis between the two terms of the objective function, examination of the effect of image size, a parameter sweep of the relaxation factor of the Weak Matching Pursuit function in the first step of the algorithm, and the relaxation of the Parseval constraint in the second step. Performance metrics used to assess the numerical results include execution time and number of loops to reach convergence, sparsity of the representation, and two image quality measures – peak signal to noise ratio (PSNR) and Structural Similarity (SSIM). The experiments indicated the algorithm takes a very long time to reach convergence, even for images of moderate size, and that reconstructions will result in greater accuracy on image patches with a small number of pixels (fewer than 100). It was also found that algorithm performance varies depending on the image format used to specify image brightness of the pixels. Finally, the Parseval constraint could be removed from the algorithm with improvement in execution time and sparsity, but without loss of accuracy.

Table of Contents

List of Figures

List of Tables

Acknowledgment

I would like to take this opportunity to extend my deepest gratitude to Dr. Travis Bemrose, my thesis director and co-advisor, for his endless patience, knowledge, motivation, and infusions of humor. His guidance helped me through every part of the research and provided support in writing this dissertation.

My sincere thanks go to Dr. K.T. Arasu and Dr. Jason Deibel, both of whom gave me the encouragement I needed at the beginning, and the continued support throughout this journey, and to Dr. Steen Pedersen and Dr. Qingbo Huang for their insightful input into my research and writing.

Generous support by the National Science Foundation (NSF) (MRI Award No. 1531923) is greatly appreciated. I am thankful to the NSF for the computational resources that were used for this research and especially to Dr. Amit Sharma for the time spent giving assistance in accessing the GLUON HPC cluster at Wright State University.

Professional help from the Statistical Consulting Center at Wright State University was invaluable. I am especially grateful to Michael Bottomley for the time he spent generating the results and explaining the methodology of the statistical analyses.

I would especially like to thank someone close to me for his great patience and offerings of grace when I needed it most - my husband, Rev. Dr. Larry Grunden. I could not have done such a great thing at this time in my life without his expressions of confidence, support, and love, while stepping in to relieve me of certain household and financial responsibilities. I would also include thanks to Dr. Emily Stelzer and Dr. Joshua Whiting, who contributed thoughtful editing suggestions, and who, along with Dr. Peter Muthard, Lara Grunden Whiting, and Rev. Larry Grunden, Jr., provided moral support.

Dedicated to

My husband, Rev. Dr. Larry A. Grunden, Sr.

Chapter 1

Introduction

Digital imaging has had an impact on all parts of our lives from everyday activities to scientific problems in mathematics, physics, engineering, chemistry, and psychology. Acquisition of these digital images comes from different kinds of sensors that convert the information into digital signals, which are then processed by a computer. The result is ultimately used by the consumer.

Handheld digital cameras and those mounted on drones create photographs for professionals and amateur enthusiasts, but other kinds of sensors allow diverse types of imaging to be generated. The medical field uses X-rays for imaging, such as digital radiography, fluoroscopy, and CT scans; and gamma rays for digital imaging, such as scintigraphy, SPECT, and PET scans. Sensors employing sound are used in medical ultrasonography and sonar; radio waves are utilized in military installations to handle radar. In geology, electromagnetic and sonographic sensors are used to detect seismic reflection and refraction. Physicists use terahertz radiation for material characterization. Chemists use fluorescence spectroscopy and hyperspectral analysis to detect the presence of certain gases. Analysis by InfoTrends indicates that people now take more than 1.4 trillion photos per year, and there are at least 7.4 trillion digital photos stored as of 2020 [1].

Images might be stored on computer hard drives, flash drives or in the cloud. The

Internet is a place for sharing, editing, and viewing digital images – not only for personal use, but also for educational, law enforcement, scientific and medical uses. It is well-known that a single uncompressed digital image can take up a great amount of memory in terms of megabytes, or even gigabytes for some purposes, depending on the image resolution. Therefore, being able to store and transmit these digital images is hampered by the potentially large file size. Keeping them in their original file sizes will preserve the image quality, but it is usually necessary to reduce the file size to allow for quicker transmission and faster downloads. The difficulty is how this can be accomplished without giving up too much quality and allowing reconstruction with fidelity. This research explores one algorithmic method of representation and examines the parameters used in this algorithm to determine where improvements could be made.

In image processing, tight frames have become more frequently applied in handling both natural and cartoon-type images for almost two decades [2–4]. Fixed, redundant-signal representation systems, which were described in the early part of this time period, performed most efficiently for cartoon-type images but did not do nearly so well for images with complex texture patterns or those without self-recursive image structures, such as fingerprint and medical images [5]. They rely on functional assumptions that may be invalid for particular types of images, e.g. isolated objects with C^2 singularities assumed by some tight frame methods. Such an assumption is applicable to cartoon-type images but not to textured images [6].

More recently, adaptive methods were developed to improve the efficiency of image processing for images that lack self-recursive structures. K-SVD[1] and other dictionary learning methods are examples of these adaptive approaches [7–9]. However, overcomplete dictionaries present at least two problems for image reconstruction. First, they lack the *perfect reconstruction property*, which ensures the given signal can be perfectly represented by its canonical expansion in a manner similar to orthonormal bases; secondly, the

[1] A dictionary learning algorithm that generalizes the K-Means Clustering process using Singular Value Decomposition

overcomplete system often results in a severely ill-posed problem [6].

The next iteration of image processing methods began to take advantage of the information stored in the image to generate data-driven representations [5, 6, 10–12]. While early methods used a fixed tight frame, newer data-driven image processing methods were developed to use the information stored in the image to derive a dynamically tailored tight frame system. These image processing methods were extended to more fields of research, including image denoising, geophysics, medicine, and seismic data analysis. Some of the early methods using data-driven tight frames were developed in the field of medicine to reconstruct computed tomography (CT) and magnetic resonance images (MRI) [13, 14] and in harmonic analysis for image denoising [6]. Research using data-driven methods has continued to progress in areas such as the analysis of seismic data by interpolation and denoising [15], redundant transforms based on Parseval frames [16], adaptive wavelet tight frame construction for accelerating MRI reconstruction [17], and limited angle CT reconstruction based on total variation (TV) and data-driven tight frames [18].

In the past several years, wavelet tight frames have been used to create sparse representations of images and signals. These sprang from a generalization of orthonormal wavelet bases. Because it permits redundancy, a frame provides more flexibility in obtaining sparsity than orthonormal bases can [19, 20]. By introducing a multi-scale wavelet, frames can efficiently capture the complex texture patterns that appear in natural images and have the property of perfect reconstruction. In addition, the construction of a multi-scale tight frame is more likely to result in a highly sparse approximation of the input image than previous methods, and it has been shown that the minimization problems arising in the construction of these tight frames are better conditioned than those of generic overcomplete dictionaries [6].

The multi-scale wavelet frame is versatile in its ability to decompose images. The low-pass portion of the multi-scale wavelet filters generates coefficients used to express the global scales of the image, while the high-pass portion produces coefficients used to

represent the local scales of the image [21]. By combining the multi-scale nature of this frame with a small number of compressed sensing measurements, this algorithm aims to produce sparse representations for a wide variety of images.

Compressed sensing is a field of study that was introduced in 2006 [22] and has developed into an important tool in many fields including applied mathematics, computer science, and electrical engineering. The key concept is that the signal itself is sparse (i.e., has a small number of non-zero entries), or has a sparse representation. Data acquisition condenses the signal directly into the compressed representation, without taking a large number of samples, most of which would have otherwise been thrown away when converted to a lossy compression format, such as jpeg. Reducing the file size allows more images to be stored in a given amount of memory space. Efficient algorithms exist to recover a high-dimensional signal from a small number of measurements when that signal has a sparse representation against some suitable basis or frame [23]. During the last fifteen years, applications that use compressed sensing have grown to encompass photography, holography, facial recognition, magnetic resonance imaging, network tomography, seismic data analysis, and transmission electron microscopy.

A model combining a sensing matrix with a data-driven multi-scale wavelet frame to increase the flexibility in obtaining a sparse representation was presented by Cao and Gao [10]. They proposed an iterative, two-step algorithm to obtain the image representation. The purpose of this research is to characterize that algorithm, including the tuning of optimization parameters and aspects that affect the structure of the multi-scale frame, and to discuss additional refinements regarding the implementation of the algorithm.

1.1 Organization of Dissertation Chapters

The dissertation is organized as follows. Chapter **??** describes background information that lays the foundation needed for the algorithm being researched. Section 1.2.1 presents

the compressed sensing model, and Section 1.2.2 introduces frame theory. These two methods are combined to allow more flexibility in obtaining a sparse representation for the original image. Section 1.3.1 details the calculation of the tight frame matrix D, Section 1.3.2 discusses the model that leads to the two-step representation algorithm shown in Section 1.3.3.

In Chapter **??**, the specifics of the implementation of the algorithm are explained. Section 2.1 points out how the MATLAB® solvers were chosen. Section 2.2 lays out specific details for collecting data, such as the filterbanks used to initialize the frame D, the generation of the sensing matrix A, the dilation of filters, and the performance metrics used to measure algorithm results. Two image formats are described in Section 2.4.

Chapter **??** contains the results of searches for parameters that will optimize algorithm execution and results. In Section 3.1 the parameters needed for MATLAB solver *fminsearch* are narrowed down to a feasible set from which a user may choose to obtain optimal results, and in Section 3.2, regularization parameters are chosen for three different scenarios that appear in the next chapter.

Chapter **??** contains the results of the research questions laid out near the beginning of this study. In Section 4.1 the effects of image size and two maximum dilation levels on the metrics is described. Section 4.2 explores the results of all possible maximum dilation levels for a limited set of image sizes. The effect of the relaxation factor of the Weak Matching Pursuit function used in Step 1 of the algorithm is explored in Section 4.3. In Section 4.4, the results of applying two different filterbanks is reported, while in Section 4.5, two different image formats are explored. And finally, Section 4.6 contains the results of interacting three parameters, i.e. filterbank size, maximum dilation level, and image format.

In Chapter 5, three possible functions used to relax the Parseval constraint are outlined in Section 5.1, and the results of the relaxation are given in Section 5.3.

1.2 Background Information

In recent years, wavelet tight frames have been used to create sparse representations of images and signals. These applications sprang from a generalization of orthonormal wavelet bases. The construction of these tight frames does not work well on all types of images, so an alternate approach is to utilize a data-driven method to generate the representation system. Two important ideas, compressed sensing and frame theory, come together in the model used and analyzed in this research to generate a representation of the original measured image.

1.2.1 Compressed Sensing

Classes of high dimensional signals are typically concentrated in or near lower dimensional subspaces, thus allowing the signal to be represented by a sparse linear combination of vectors from some basis [19]. Compressed sensing models an original signal $\boldsymbol{f} \in \mathbb{R}^N$ by collecting $m \ll N$ observed linear measurements $\boldsymbol{g} \in \mathbb{R}^m$ such that

$$\boldsymbol{g} = \boldsymbol{A}^* \boldsymbol{f} + \boldsymbol{e} \tag{1.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{N \times m}$ is a sensing matrix whose columns represent sensor elements, and $\boldsymbol{e} \in \mathbb{R}^m$ is an error term that represents noise. The goal is to reconstruct the signal $\boldsymbol{f}$ based on the observed measurements $\boldsymbol{g}$ and the known sensing matrix $\boldsymbol{A}$. By convex programming, one finds the solution to

$$\underset{\tilde{\boldsymbol{f}} \in \mathbb{R}^N}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{f}} \right\|_1 \quad s.t. \quad \left\| \boldsymbol{A}^* \tilde{\boldsymbol{f}} - \boldsymbol{g} \right\|_2 \leq \varepsilon, \tag{1.2}$$

where $\|\cdot\|_2$ denotes the standard Euclidean norm, $\|f\|_1 = \sum |f_i|$ is the ℓ_1-norm and ε is an upper bound on the noise [24].

In the earliest compressed sensing literature, the unknown signal $\boldsymbol{f}$ was assumed to be sparse or compressible in an orthonormal basis [22]. In this research, the signals are

assumed to be sparse in a redundant frame $\boldsymbol{D} \in \mathbb{R}^{N\times r}$ where $N < r$. Then $\boldsymbol{f} = \boldsymbol{D}\boldsymbol{v}$, where $\boldsymbol{v}$ is a sparse coordinate representation for $\boldsymbol{f}$, that is, $\boldsymbol{f}$ is sparse or compressible in the frame $\boldsymbol{D}$ and $\boldsymbol{f}$ can be found by solving Equation (1.3) for $\boldsymbol{v}$ and using it to compute $\boldsymbol{f}$.

$$\hat{\boldsymbol{v}} = \underset{\boldsymbol{v}\in\mathbb{R}^r}{\operatorname{argmin}} \|\boldsymbol{v}\|_1 \quad s.t. \quad \|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2 \leq \varepsilon. \tag{1.3}$$

1.2.2 Frame Theory

This section presents preliminary information on frames, tight frames, and Parseval frames in a Hilbert space $\mathcal{H} \cong \mathcal{F}^m$ for some field $\mathcal{F}$. Let $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ denote a preferred inner product and associated norm for $\mathcal{H}$, respectively. For an index set $\mathcal{I} \subset \mathbb{Z}$, a sequence of vectors $\{\boldsymbol{\phi}_i\}_{i\in\mathcal{I}} \subset \mathcal{H}$ is a frame for $\mathcal{H}$ if there exist $0 < A \leq B < \infty$ such that for all $\boldsymbol{f} \in \mathcal{H}$,

$$A\|\boldsymbol{f}\|_2^2 \leq \sum_{i\in\mathcal{I}} |\langle\boldsymbol{\phi}_i, \boldsymbol{f}\rangle|^2 \leq B\|\boldsymbol{f}\|_2^2. \tag{1.4}$$

The constants A and B are called, respectively, the lower and upper frame bounds for the frame. If it is possible that $A = B$, then $\{\boldsymbol{\phi}_i\}$ is called a tight frame. If it is possible that $A = B = 1$, then $\{\boldsymbol{\phi}_i\}$ is called a Parseval tight frame.

There are three important operators associated with a given frame $\{\boldsymbol{\phi}_i\}_{i\in\mathcal{I}}$. One is the synthesis operator, defined as $\Phi : \ell^2(\mathcal{I}) \to \mathcal{H}$ where

$$\Phi(\boldsymbol{c}) := \boldsymbol{\Phi}\boldsymbol{c} = \sum_{i\in\mathcal{I}} \boldsymbol{\phi}_i c_i,$$

for any sequence $\{c_i\} = c \in \ell^2(\mathcal{I})$. When represented against a basis that is orthonormal with respect to the given inner product, the matrix that represents the synthesis operator has columns which are coordinate vectors for the frame vectors: $\boldsymbol{\Phi} = [\ldots \quad \boldsymbol{\phi}_i \quad \ldots]$.

Another is the analysis operator defined as $\Phi^* : \mathcal{H} \to \ell^2(\mathcal{I})$, where

$$\Phi^*(\boldsymbol{f}) := \{\langle \phi_i, \boldsymbol{f} \rangle\}_{i \in \mathcal{I}} .$$

Note that this notation differs from frame theory literature that uses traditional inner product notation, which is conjugate-linear in the right, and this uses the more modern inner product notation which is conjugate-linear in the left hand side. The modern notation ensures that when vectors are represented against an orthonormal basis, the inner product corresponds with the dot product: $\langle x, y \rangle = x^* y$.

Finally, the frame operator is defined as $S = \Phi\Phi^* : \mathcal{H} \to \mathcal{H}$. It might be considered the most important operator associated with a frame, because it encodes crucial properties of the frame [19]. The property of most importance for this research is that Φ is Parseval if and only if $S = I$. In addition, the condition $0 < A \leq B < \infty$ in the definition for a frame ensures that S is bounded and invertible.

For a thorough introduction to the basics of frame theory, it is recommended that the reader consult Chapter 1 in the book by Casazza and Kutyniok [19].

1.3 Tight Frame D, Model, and Representation Algorithm

The two-step representation algorithm being studied [10] is summarized here to improve clarity. Individual filters are dilated to various levels and used to calculate the multi-scale frame $\boldsymbol{D}$. The model provides the foundation of the two-step algorithm, and is followed by the motivation for the two-step format.

1.3.1 Multi-scale Frame $\boldsymbol{D}$

To create the multi-scale frame $\boldsymbol{D}$ used in this research, a set of k filters $\{h_i\}_{i=0}^{k-1}$, each of length b, is selected as a starting point [10, 25]. To create the multi-scale features of the frame, the filters are dilated to different scales, or levels, before being combined in the final matrix $\boldsymbol{D}$. At level l, each filter is dilated by inserting $2^{(l-1)} - 1$ zeros between consecutive filter elements. Level $l = 1$ corresponds to the original undilated filter, and level $l = L$ to the maximum dilation level allowed by the size of the image or perhaps restricted by the user. For a given filter h_i at any dilation level $l \in \{1, 2, \ldots, L\}$, the dilated sequence $h = h^{(l)}$ is created as shown in Equation (1.5).

$$h^{(l)} := \left\{ h_i(1), \underbrace{0, \ldots, 0}_{2^{(l-1)}-1}, h_i(2), 0, \ldots \quad \ldots, 0, h_i(b-1), \underbrace{0, \ldots, 0}_{2^{(l-1)}-1} h_i(b) \right\} \tag{1.5}$$

It is somewhat more convenient to index this dilated sequence as $h = \{h(j)\}_{j=-J}^{J}$, so that $h(0)$ is in the middle. Note that if b is the length of each filter, h_i, then the number of elements in the dilated sequence, $h^{(l)}$, will be:

$$2J + 1 = b + (b-1)(2^{(l-1)} - 1). \tag{1.6}$$

The matrix $\boldsymbol{S}(h)$, a convolution operator under the Neumann (symmetric) boundary condition and shown in definition (1.7) below, is the sum of a Toeplitz matrix plus a Hankel matrix. It is noted here that the convolution operator $\boldsymbol{S}(h)$ is following the convention of previous papers on this topic and should not be confused with S typically used as the frame operator in frame theory. Regardless of the dilation level of h, each of these matrices will be $N \times N$, where N is the total number of pixels in the original image. Zeros fill out the Toeplitz matrix in the outer diagonal rows; for the Hankel matrix, zeros fill in the skew diagonals from the center to meet the entries from h in the upper left and lower right corners.

$$\boldsymbol{S}(h) :=$$

$$\begin{bmatrix} h(0) & h(-1) & \dots & h(-J) & & 0 \\ h(1) & \ddots & \ddots & & \ddots & \\ \vdots & \ddots & \ddots & \ddots & & h(-J) \\ h(J) & & \ddots & \ddots & \ddots & \vdots \\ & \ddots & & \ddots & \ddots & h(-1) \\ 0 & & h(J) & \dots & h(1) & h(0) \end{bmatrix} + \begin{bmatrix} h(1) & h(2) & \dots & h(J) & & 0 \\ h(2) & & \iddots & & \iddots & \\ \vdots & \iddots & & \iddots & & h(-J) \\ h(J) & & \iddots & & \iddots & \vdots \\ & \iddots & & \iddots & & h(-2) \\ 0 & & h(-J) & \dots & h(-2) & h(-1) \end{bmatrix} \tag{1.7}$$

To simplify the notation for $\boldsymbol{D}$, define $\boldsymbol{H}_i^{(l)} := \boldsymbol{S}(h_i^{(l)})$. These $N \times N$ matrices at different dilation levels are then multiplied together to create blocks in the multi-scale synthesis operator $\boldsymbol{D}$ as shown in Equation (1.8).

$$\begin{aligned} \boldsymbol{D} := \Bigg[\prod_{l=1}^{L} \boldsymbol{H}_0^{(l)} \Bigg| \left(\prod_{l=1}^{L-1} \boldsymbol{H}_0^{(l)} \right) \boldsymbol{H}_1^{(L)} \Bigg| \dots \Bigg| \left(\prod_{l=1}^{L-1} \boldsymbol{H}_0^{(l)} \right) \boldsymbol{H}_{k-1}^{(L)} \\ \left| \left(\prod_{l=1}^{L-2} \boldsymbol{H}_0^{(l)} \right) \boldsymbol{H}_1^{(L-1)} \right| \dots \left| \left(\prod_{l=1}^{L-2} \boldsymbol{H}_0^{(l)} \right) \boldsymbol{H}_{k-1}^{(L-1)} \right. \\ \vdots \\ \left| \quad \boldsymbol{H}_1^{(1)} \quad \right| \dots \left| \quad \boldsymbol{H}_{k-1}^{(1)} \quad \right. \Bigg] \end{aligned} \tag{1.8}$$

To clarify the composition of the factors of $\boldsymbol{D}$, observe that although its structure is

actually

$$\boldsymbol{D} = [\ \boldsymbol{A} \mid \boldsymbol{B}_{11} \mid \ \ldots\ \mid \boldsymbol{B}_{1c} \mid \boldsymbol{B}_{21} \mid \ \ldots\ \mid \boldsymbol{B}_{rc}\ ], \tag{1.9}$$

in order to highlight the repetition in the block structure in Equation (1.8), it has been formatted as shown below.

$$\begin{aligned} \boldsymbol{D} = [\ \boldsymbol{A} \mid \boldsymbol{B}_{11} \mid \ldots \mid \boldsymbol{B}_{1c} \\ \vdots \\ \mid \boldsymbol{B}_{r1} \mid \ldots \mid \boldsymbol{B}_{rc}\ ] \end{aligned}$$

The frame $\boldsymbol{D}$ consists of $1 + (k-1)L$ blocks which are each $N \times N$, so $\boldsymbol{D}$ has N rows and $(1 + (k-1)L)N$ columns. Thus the choice of L has two related effects on $\boldsymbol{D}$, both of which make it more flexible for representing data. First, each increase in L adds $k-1$ more blocks to $\boldsymbol{D}$ providing more vectors (columns) to choose from. Second, the matrix products in those new blocks have an additional term $\boldsymbol{H}_i^{(L)}$ created from a more dilated filter than in other blocks, so that the resulting columns of $\boldsymbol{D}$ in those new blocks have more non-zero entries, or larger support.

1.3.2 Model

Consider the compressed sensing model in Equation (1.1), where if it can be assumed that $\boldsymbol{f}$ is sparse, the recovery algorithm could take the form

$$\min_{\boldsymbol{f}} \|\boldsymbol{f}\|_0, \quad s.t. \quad \boldsymbol{g} = \boldsymbol{A}^*\boldsymbol{f}, \tag{1.10}$$

where $\|\boldsymbol{f}\|_0$ represents the number of non-zero entries in $\boldsymbol{f}$. Solving this problem exactly is NP-hard. Replacing the ℓ_0-pseudo-norm with the ℓ_1-norm is a convex relaxation of Equation (1.10) and yields a problem that is more easily solved. This formulation of the problem is known as Basis Pursuit [26–28] and is defined as

$$\min_{\boldsymbol{f}} \|\boldsymbol{f}\|_1, \quad s.t. \quad \boldsymbol{g} = \boldsymbol{A}^*\boldsymbol{f}. \tag{1.11}$$

To recover $\boldsymbol{f}$ from the small number of measurements in $\boldsymbol{g}$, compressed sensing algorithms assume the fixed matrix $\boldsymbol{A}$ is represented against an appropriate basis such that $\boldsymbol{f}$ is sparse in that basis. In practice, $\boldsymbol{f}$ may belong to a class of signals which is not well understood or one may want to work with multiple classes, so it's not clear which measurement matrix will lead to a sparse $\boldsymbol{f}$. Instead of assuming $\boldsymbol{f}$ is sparse in the relevant basis, it can represented against a frame $\boldsymbol{D}$ as $\boldsymbol{f} = \boldsymbol{D}\boldsymbol{v}$, where $\boldsymbol{v}$ is a coefficient or coordinate vector for $\boldsymbol{f}$ in terms of $\boldsymbol{D}$. By tuning the filter-bank scalars that determine $\boldsymbol{D}$, the sparsity of $\boldsymbol{v}$ can be improved. As frames can have more flexibility than a basis, in general there's a better chance for $\boldsymbol{v}$ and a data-driven, multi-scale frame $\boldsymbol{D}$ taken together to provide a sparse representation for $\boldsymbol{f}$. The usual compressed sensing model in Equation (1.1) is now replaced by Equation (1.12).

$$\boldsymbol{g} = \boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v} + \boldsymbol{e} \tag{1.12}$$

The minimization problem in Equation (1.11) now becomes the one presented in [10] and shown in Equation (1.13),

$$\min_{\boldsymbol{v},\{h_i\}_{i=0}^{k-1}} \left(\mu\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2 + \|\boldsymbol{v}\|_1\right) \quad s.t. \quad \boldsymbol{D}\boldsymbol{D}^* = I, \tag{1.13}$$

where $\mu > 0$ is some regularization parameter, $\boldsymbol{A} \in \mathbb{R}^{N\times m}$ is a compressed sensing matrix, $\boldsymbol{D}$ is the multi-scale frame in Equation (1.8), $\boldsymbol{v}$ is the sparse representation of the vectorized image, as described above, and $\boldsymbol{g} \in \mathbb{R}^m$ is the vector of observed measurements of the original image. The constraint $\boldsymbol{D}\boldsymbol{D}^* = I$ ensures $\boldsymbol{D}$ is a Parseval frame, that is, tight with frame bounds equal to 1.

1.3.3 Representation Algorithm

The optimization problem given in Equation (1.13) is minimized over two different types of variables. Rather than trying to optimize it all at once, splitting it into alternating steps

that address $\boldsymbol{D}$ and $\boldsymbol{v}$ independently permits the use of the most efficient algorithm for each purpose. Starting with an initial filterbank so that $\boldsymbol{D}$ is nonzero, Step 1, shown in Equation (1.14), holds the filterbank fixed while minimizing the objective function with respect to $\boldsymbol{v}$. The first term in the objective function ensures the accuracy of the representation, while the ℓ_1-norm term encourages the sparsity of $\boldsymbol{v}$.

$$\text{Step 1:} \quad \boldsymbol{v} := \underset{\boldsymbol{v}}{\operatorname{argmin}} \left(\mu_1 \|\boldsymbol{A}^* \boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2 + \|\boldsymbol{v}\|_1 \right) \tag{1.14}$$

A regularization parameter μ_1 is used to control the emphasis placed on the two terms of the objective function. More on the regularization parameter will be explained in Section 3.2.

In Step 2, shown in Equation (1.15), the vector $\boldsymbol{v}$ found in Step 1 is held fixed, while minimizing the error term with respect to the filterbank $\{h_i\}_{i=0}^{k-1}$, subject to the Parseval constraint.

$$\text{Step 2:} \quad \{h_i\}_{i=0}^{k-1} := \underset{\{h_i\}_{i=0}^{k-1}}{\operatorname{argmin}} \left(\|\boldsymbol{A}^* \boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2 \right) \quad s.t. \quad \boldsymbol{D}\boldsymbol{D}^* = I \tag{1.15}$$

After Step 2, $\boldsymbol{D}$ is updated using the filterbank $\{h_i\}_{i=0}^{k-1}$ and control returns to Step 1.

Chapter 2

Implementation

The representation algorithm studied in this research was originally presented by Cao and Gao [10]. Because the manner used by the authors to implement the algorithm was not exactly clear, methods of how this might be accomplished were investigated. This was followed by questions regarding how the algorithm parameters and conditions might be varied, with a goal of improving accuracy and execution time.

To implement the algorithm in Section 1.3.3, it was necessary to choose suitable solvers (i.e., MATLAB functions) to find the coefficient vector $\boldsymbol{v}$ in Step 1 and the updated filterbank in Step 2. It should be noted that the speed of execution was a significant limiting factor and motivated many of our choices early on in the process. It was taking days to run 1000 pixel image patches, and an estimate of the time to complete the full images with $65,000$ pixels (thought to have been used by Cao and Gao) could have taken months or years. The solvers considered for Steps 1 and 2 are reviewed in Section 2.1. Details of the implementation, such as fixed parameters and filterbanks are included in Section 2.2. Alternate forms considered for the error term are discussed briefly in Section 2.3. In Section 2.5, a verification of the correct direction of multiplication was performed, and in Section 2.4, reviews of two image formats used in this research were completed.

2.1 Solvers

To implement the two-step algorithm, several MATLAB solvers were considered, including *fminunc*, *quadprog*, *fseminf*, *fmincon*, *fminsearch* and variations of OMP (Orthogonal Matching Pursuit). Several of the solvers were eliminated because they did not correctly apply to our objective function, but those that were appropriate for the settings in Step 1 and Step 2 and resulted in reasonable execution times were retained. The final set of possible MATLAB solvers explored for Step 1 were *quadprog*, Orthogonal Matching Pursuit (OMP), and Weak Matching Pursuit (WMP), and for Step 2 were *fmincon* and *fminsearch*.

For Step 1, time tests were performed on the solvers *quadprog*, OMP, and WMP. *Quadprog* required reformulating the objective function in a specific quadratic programming format. This included requiring the entries of v to be positive, accomplished by separating v into the sum of its positive and negative entries. This added complexity to the objective function and may be part of the reason *quadprog* did not perform as well as OMP and WMP, taking much longer to execute and resulting in far less accuracy. OMP was explored next, having the goal of constructing a target support by adding vectors one at a time from a frame by iteratively finding the vector that best matches the remaining unrepresented portion of the target vector [29–31]. WMP, a variation of OMP, specifies a factor $w \in (0, 1]$ and allows for a suboptimal choice of the next vector to be included such that it is a factor w away from the optimal choice. When $w = 1$, WMP is identical to OMP. By stopping when it finds a vector that is good enough (as defined above), WMP executes faster than OMP. The function WMP was selected for Step 1, because of the need to decrease run-time and because in early experimentation it was found that the representation algorithm produced similar accuracy results with WMP as it did with OMP.

The choice of WMP presented a concern: neither of MATLAB's built in OMP/WMP functions could be used for Step 1, because they wouldn't minimize the complete objective function shown in Equation (1.14), but just the error term, $\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2$. Minimizing only the error term is inadequate, because the standard OMP and WMP functions would stop

execution when the norm of the unrepresented residual was smaller than some threshold. That isn't sufficient in this case, because the minimum of Equation (1.14) is not expected to approach zero. As more non-zero entries are placed in $\boldsymbol{v}$, the error term decreases monotonically and the 1-norm term increases monotonically.

A version of the WMP function was written to include $\|\boldsymbol{v}\|_1$ and to be used specifically for Step 1 in the representation algorithm. It was observed that the error term and 1-norm terms did not change smoothly, rather changing in spurts such that Equation (1.14) tended to have local minima near the global minimum and the WMP function would exit when it encountered the first local minimum. This was accepted, because experimentation showed it was nearly as good as the global minimum. It took less time than guaranteeing the global minimum had been found, and it emphasized the sparsity of $\boldsymbol{v}$. For all numerical results, the revised WMP function was used with MATLAB's default value $w = 0.6$ except for those reported in Section 4.3, which explored the choice of w.

For Step 2, the MATLAB solver *fmincon* (a general constrained minimization solver) was tried on the formulation shown in Equation (1.15), but for speed purposes was switched out in favor of using *fminsearch* which required the unconstrained formulation in Equation (2.1).

$$\{h_i\}_{i=0}^{k-1} := \underset{\{h_i\}_{i=0}^{k-1}}{\operatorname{argmin}} \left(\|\boldsymbol{A}^* \boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2 + \mu_2 \|\boldsymbol{D}\boldsymbol{D}^* - I\|_F^2 \right) \tag{2.1}$$

All results presented used WMP for Step 1 and *fminsearch* with the form shown in Equation (2.1) for Step 2.

2.2 Details

In this section additional important information relevant to our implementation is described. Two-dimensional images were vectorized into one-dimensional signals. This is a linear

transformation that converts the two-dimensional image matrix into a column vector by stacking columns, beginning with column 1, then column 2, etc. For all numerical experiments, the sensing matrix $\boldsymbol{A}$ was a random i.i.d. Gaussian matrix, scaled so that it satisfied the Restricted Isometry Property (RIP) with high probability [32]. The sampling rate for $\boldsymbol{A}$ was $p = 1/2$, such that $\boldsymbol{A} \in \mathbb{R}^{N\times(N/2)}$.

It was determined that computing the full frame $\boldsymbol{D}$ all at once was of concern, because of the amount of execution time and the computer memory required. Even when using sparse data structures, the memory required was significant, but further, $\boldsymbol{D}$ consists of several block matrices multiplied together after which it is used to compute the product $\boldsymbol{Dv}$. When faced with a matrix-matrix-vector product, it is much faster to do the matrix-vector product first, and so it's best to break $\boldsymbol{v}$ into corresponding blocks and multiply them by each Toeplitz-plus-Hankel matrix, $\boldsymbol{H}_i^{(l)}$. This can be done in such a way that each block $\boldsymbol{H}_i^{(l)}$ is only needed once and can then be discarded before constructing the next. For Step 1, the entire matrix $\boldsymbol{A}^*\boldsymbol{D}$ is needed in memory all at once to construct $\boldsymbol{v}$, but similarly this can be built faster and with less memory by combining $\boldsymbol{A}^*$ with the appropriate blocks, rather than by constructing $\boldsymbol{D}$ first. Finally, when testing the Parseval condition, it is better to construct $\boldsymbol{DD}^*$ block-wise rather than by computing $\boldsymbol{D}$ first and multiplying $\boldsymbol{DD}^*$. MATLAB functions were written to facilitate these multiplication strategies.

The numerical experiments and parameter searches were run under MATLAB R2019a (64 bit) on a high performance cluster with 2x Intel Ten Core Xeon processors (2.2 GHz) each having 128 GB of memory, or under MATLAB R2020a (64 bit) on a PC workstation with an 8 core INTEL CPU (4.2 GHz) and 32 GB of memory.

The stopping rule for the representation algorithm was a test of convergence. At the end of each step of the representation algorithm, the maximum changes in the individual elements of $\boldsymbol{v}$ and the filterbank h were calculated. If both changes were less than a specified tolerance the algorithm would finish. The tolerance value $tol = \sqrt{eps} \cong 1.49^{-8}$ was used

for this research[1]. The filterbank obtained in the last iteration of Step 2 was used to find the data-driven frame $\boldsymbol{D}$. The vector $\boldsymbol{v}$ found in the last iteration of Step 1 was used along with $\boldsymbol{D}$ to generate the reconstructed image by calculating $\boldsymbol{f} = \boldsymbol{D}\boldsymbol{v}$.

2.2.1 B-Spline Wavelet Tight Frames

The blocks $\boldsymbol{H}_i^{(l)}$ used to build the synthesis operator $\boldsymbol{D}$, described in Section 1.3.1, require a set of filters, called a filterbank. These filters may be considered a set of 'complementary' wavelets, which will decompose data without gaps or overlap so that the decomposition process is mathematically reversible. It is well-known that once the one-dimensional frame filterbank $\{a_i\}_{i=1}^{k}$ for generating a tight frame is given, a two-dimensional tight frame could be constructed by the tensor product $\{a_i \otimes a_j\}_{i,j=1}^{k}$ of the one-dimensional frame filters. To minimize execution time and computer memory requirements, for this research, the two-dimensional image with N pixels was viewed as a vector in $\mathbb{R}^N$ by stacking the columns, beginning with the first column on top of the stack, followed by the second column, etc.

For this research, two versions of the B-spline framelet filter were used to initialize $\boldsymbol{D}$, both described by Shen et al [33]. For most of the research, $\boldsymbol{D}$ was initialized with the piecewise linear B-spline framelet shown in Equation (2.2), which consists of filters $\{h_i\}_{i=0}^{2}$ of length $b = 3$.

$$\begin{bmatrix} h_0 & h_1 & h_2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & \sqrt{2} & -1 \\ 2 & 0 & 2 \\ 1 & -\sqrt{2} & -1 \end{bmatrix} \tag{2.2}$$

The results of using this set of filters was contrasted with those obtained by using a larger set of filters $\{j_i\}_{i=0}^{4}$ of length $b = 5$. For this comparison, the cubic B-spline filter shown in Equation (2.3) was chosen.

[1] In MATLAB, eps gives the machine precision, the difference between 1.0 and the nearest distinguishable floating point number.

$$\begin{bmatrix} h_0 & h_1 & h_2 & h_3 & h_4 \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 1 & 2 & -\sqrt{6} & -2 & 1 \\ 4 & 4 & 0 & 4 & -4 \\ 6 & 0 & 2\sqrt{6} & 0 & 6 \\ 4 & -4 & 0 & -4 & -4 \\ 1 & -2 & -\sqrt{6} & 2 & 1 \end{bmatrix} \tag{2.3}$$

The possible levels of dilation that can be applied to a filter are constrained by b, the length of the filter, and N, the number of pixels in the image. Referring back to Equation (1.5) and Equation (1.7), the number of elements in the filter at any level $l \in [1, L]$ must fit into the upper left hand corner of the $N \times N$ Toeplitz matrix:

$$b + (b-1)(2^{(l-1)} - 1) \leq 2N - 1 \tag{2.4}$$

$MaxL$ represents the maximum possible dilation level, and it can be obtained by solving Equation (2.4) for l, then taking the floor of the upper bound to get the largest valid integer.

$$l \leq L \leq MaxL = \text{floor}\left(1 + \log_2 \frac{2(N-1)}{b-1}\right) \tag{2.5}$$

It is possible the user may choose to restrict the value of L to something smaller than its upper bound. Reasons for preferring $L < MaxL$ are usually due to limited computing resources or a need for faster run time, but may also be due to its effect on accuracy, as will be seen in Section 4.2. Unless specified differently in the description of the numerical experiments, the maximum dilation level parameter was held fixed at $L = 3$.

2.2.2 Performance Metrics

Six metrics were used to compare the performance of the representation algorithm in all the scenarios explored in this research. $Time$ was a platform-dependent measure (in seconds) and $Loops$ was a platform-independent measure of how long it would take to reach convergence. (Note that in order to detect convergence, a final 'static' loop with no change is always needed.) Two metrics describe the vector $\boldsymbol{v}$; they are the number of nonzero entries in $\boldsymbol{v}$ represented by v_0, and the 1-norm of $\boldsymbol{v}$ denoted by v_1.

Finally, two metrics measure the accuracy of reconstruction. The first accuracy measure is Peak Signal to Noise Ratio ($PSNR$), whose values lie in the interval $[0, \infty)$. $PSNR$ is a commonly used measure that is easy to calculate. The formula is shown in Equation (2.6)

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}, \tag{2.6}$$

where $MSE = \frac{1}{M*N} \sum_{i=1}^{M} \sum_{j=1}^{N} (f(i,j) - g(i,j))^2$, and $f(i,j)$, $g(i,j)$ are the pixel values of the original image and estimate of the image, respectively, and the dimensions of both images are $M \times N$. Note that the value of 255 in the numerator corresponds to the maximum brightness level of a pixel in the image when using the integer format. Section 2.4 contains a more detailed discussion of this topic.

The second accuracy measure is the Structural Similarity measure ($SSIM$), and is based on a product of three factors that are indicators of luminance, contrast, and structure. Hore and Ziou [34] explain this more complicated formula shown in Equation (2.7)

$$SSIM(f,g) = l(f,g) * c(f,g) * s(f,g), \tag{2.7}$$

Table 2.1: Performance metrics used in this research.

Metric	Description
$Time$	Time to complete representation algorithm (in seconds)
$Loops$	Number of loops to complete representation algorithm
v_0	$\|v\|_0$ - number of nonzero entries (pseudo-norm notation)
v_1	$\|v\|_1$ - the 1-norm of $\boldsymbol{v}$
$PSNR$	Peak Signal to Noise Ratio - measure of accuracy
$SSIM$	Structural Similarity - measure of accuracy

where

$$
\begin{aligned}
l(f,g) &= \frac{2\mu_f\mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1} \\
c(f,g) &= \frac{2\sigma_f\sigma_g + C_2}{\sigma_f^2 + \sigma_g^2 + C_2} \\
s(f,g) &= \frac{\sigma_{fg} + C_3}{\sigma_f\sigma_g + C_3}
\end{aligned}
\tag{2.8}
$$

The first definition in Equation (2.8) is the luminance comparison function and uses means μ_f and μ_g. This factor is maximized and equal to 1 only if luminance is the same for both f and g. The second definition in Equation (2.8) is the contrast comparison function and uses standard deviations σ_f and σ_g. This factor is maximized and equal to 1 only if contrast is the same for both f and g. The third definition is the structure comparison function which measures the correlation coefficient between f and g using the covariance σ_{fg}. The positive constants C_1, C_2, and C_3 are used to avoid a null denominator.

Formally, $SSIM$ can be negative (for example, by comparing a photograph and its film negative), but comparing two random and completely unrelated images will yield an $SSIM$ of 0 on average, and for our purposes of trying to match an original image with a reconstruction, $SSIM$ should lie in the interval $[0, 1]$. Note that larger values for either measure indicate greater accuracy in the representation and reconstruction. Interestingly, it has been shown there is an analytical relation between $PSNR$ and $SSIM$ [34].

Ideally, one would like the largest possible values for $PSNR$ and $SSIM$, but the

smallest possible values for $Loops$, $Time$, v_0 and v_1. These will be the goals when choosing parameters in the research that follows.

2.3 Alternate Forms Considered

As part of the exploration of this algorithm, alternate forms of the objective function were considered in Step 1. An early attempt to improve upon the representation algorithm collapsed the two-step algorithm into a single step method. This updated both the coefficient vector v and the filterbank $\{h_i\}_{i=0}^{k-1}$ in the same step, as shown in Equation (2.9)

$$\min_{\boldsymbol{v},\{h_i\}_{i=0}^{k-1}} \left(\mu\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v}-\boldsymbol{g}\|_2^2+\|\boldsymbol{v}\|_1\right) \quad s.t. \quad \boldsymbol{D}\boldsymbol{D}^*=\boldsymbol{I} \tag{2.9}$$

Both MATLAB solvers, *fmincon* and *fminsearch* were applied in time tests. However, this single step was inefficient, regardless of the solver used. Because of the time and computer resources required to complete the single step, it was abandoned.

Another early attempt explored variations on the norms used in the objective function in Step 1 of the representation algorithm. Forms (2.10) and (2.11) are variations of Form (2.12), used by the authors of the algorithm. These were considered in the hope that something faster would be found - fast enough to handle the $65,000$ pixel images the authors [10] appeared to use for their reported results. They were abandoned, because both resulted in larger reconstruction errors, took longer to execute, and showed no obvious advantages.

$$\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v}-\boldsymbol{g}\|_2+\|\boldsymbol{v}\|_1 \tag{2.10}$$

$$\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v}-\boldsymbol{g}\|_2^2+\|\boldsymbol{v}\|_1^2 \tag{2.11}$$

$$\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v}-\boldsymbol{g}\|_2^2+\|\boldsymbol{v}\|_1 \tag{2.12}$$

2.4 Test Images and Image Formats

Test images were utilized to explore the efficiency and accuracy of the representation algorithm, which is the focus of this research. Those shown in Figure 2.1 are commonly used in image processing research, because they demonstrate a range of textures and brightness values. Each of these were stored as 256×256 grayscale images, resulting in a total of $N = 2^{16}$ pixels.

Figure 2.1: Test images: 256×256 grayscale pixels.

The test images were stored in integer format, where each pixel value has an intensity or brightness value equal to an integer in the interval $[0, 255]$. The value '0' represents black, '255' represents white, and values in between represent shades of gray. Figure 2.2 displays the range of brightness values for this format [35].

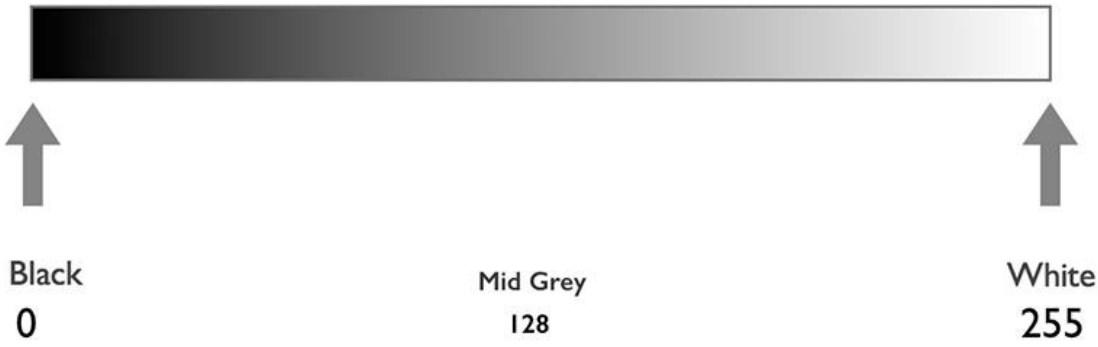


Figure 2.2: In integer format, grayscale images are stored with 256 different integer values.

Sometimes images are stored using floating point values, instead of integer values. The range of values for floating point images have intensity or brightness values in the interval

$[0, 1]$. The value '0' still represents black, but '1' represents white, and decimal values in between '0' and '1' represent shades of gray. Figure 2.3 displays the range of shades of gray for this format [35].

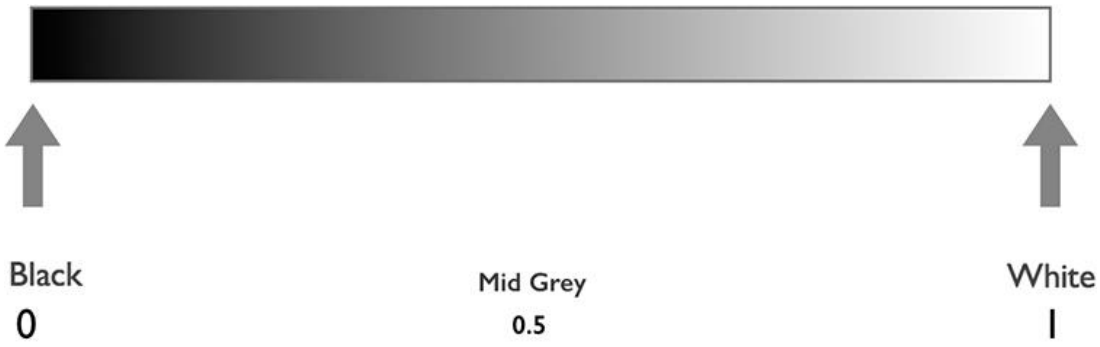


Figure 2.3: In floating point format, grayscale images are stored with decimal values in $[0, 1]$.

The image format might have an effect on the execution of the representation algorithm and the metrics used to characterize it. Section 4.5 explores these effects and provides details about them using descriptive statistics. To get the floating point format from the integer format, each pixel value was divided by 255 to obtain a value in $[0, 1]$.

2.5 Determining the Correct Order of Multiplication of the Blocks in $\boldsymbol{D}$

It was not clear whether to interpret the matrix multiplication in Equation (1.8) as the usual left to right multiplication or as in operator composition, resulting in multiplication from right to left, especially since some sources conflated whether the expressions in Equation (1.8) were producing the analysis or synthesis operator for the frame, which are transposes of each other. To be absolutely certain, the construction of $\boldsymbol{D}$ was performed in two different ways: first by multiplying the matrix components $\boldsymbol{H}_i^{(l)}$ of all blocks as

indicated in Equation (1.8), i.e., the forward direction, and second by multiplying them together in the reverse order. For example, when $L = 3$ the first block of $\boldsymbol{D}$ would look like $\boldsymbol{H}_0^{(1)}\boldsymbol{H}_0^{(2)}\boldsymbol{H}_0^{(3)}$ in the forward direction and $\boldsymbol{H}_0^{(3)}\boldsymbol{H}_0^{(2)}\boldsymbol{H}_0^{(1)}$ in the reverse. These results are briefly included here as a guide to readers.

For each of the $n = 105$ observations, the algorithm was run constructing $\boldsymbol{D}$ in both forward and reverse directions. Means and standard deviations were calculated for each of the performance metrics in both directions. Paired t-tests were used to compare means and Kolmogorov-Smirnov (KS) tests to compare distributions. Those results are shown in Table 2.2. More information about the statistical tests used here can be found in Chapters 7.2 and 13.7 in the book by Wayne Daniels [36].

Table 2.2: Paired t-tests compare means and K-S tests compare distributions of forward and reverse directions of block multiplications for constructing the analysis operator $\boldsymbol{D}$.

	Forward Mean ± SD	Reverse Mean ± SD	Paired t-test: p-value	K-S test: p-value
$Time(s)$	22.1 ± 4.9	118.2 ± 26.2	1.6×10^{-59}	1.6×10^{-45}
$Loops$	16.6 ± 4.2	42.7 ± 13.4	1.8×10^{-34}	5.3×10^{-42}
$PSNR$	22.8 ± 4.6	2.4 ± 4.5	1.4×10^{-61}	7.2×10^{-43}
$SSIM$	0.46 ± .15	0.01 ± .02	2.1×10^{-53}	1.6×10^{-45}
v_0	117.1 ± 10.6	128.0 ± 0.1	3.0×10^{-17}	1.9×10^{-39}
v_1	43,818 ± 11,520	65,043 ± 44,612	1.5×10^{-06}	2.8×10^{-07}

Histograms of the data observations are shown in Figure 2.4. Smaller values for $Loops$, $Time$, v_0, and v_1 are desirable, and the data represented by the blue bars for the forward direction in the histograms are obviously smaller than the data represented by the red bars used for the reverse direction. Larger values are preferred for $PSNR$ and $SSIM$. For these two metrics the forward data is clearly larger than for the reverse data. The histograms certainly indicate the forward direction appears to be significantly better than the reverse direction for all the metrics. The p-values for the K-S and paired t-tests in Table 2.2 strongly

support this conclusion, even after taking into consideration the number of tests that were performed.

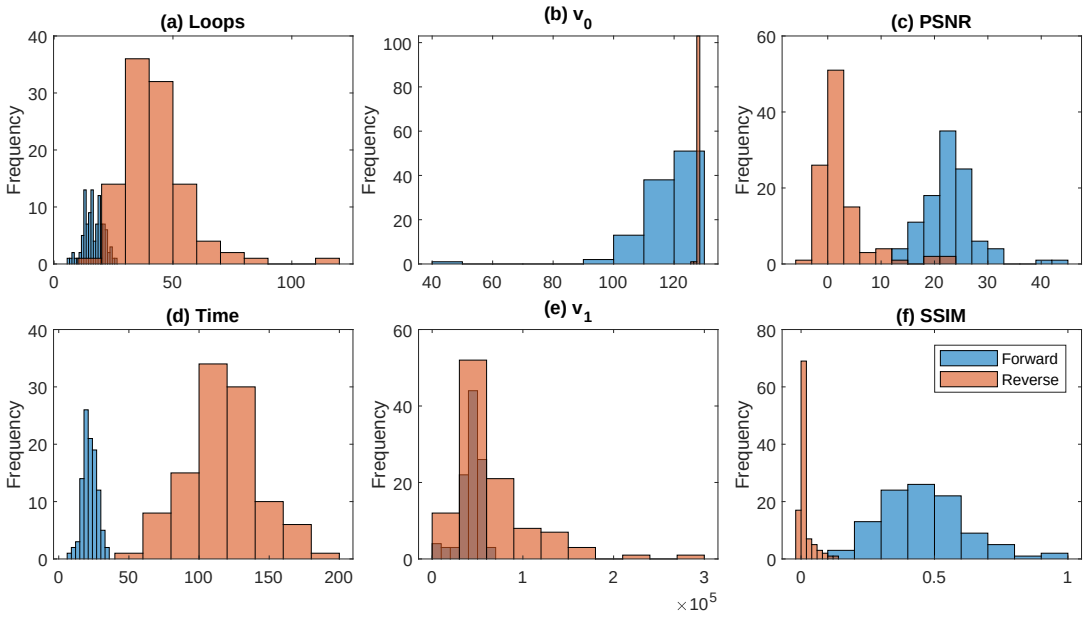
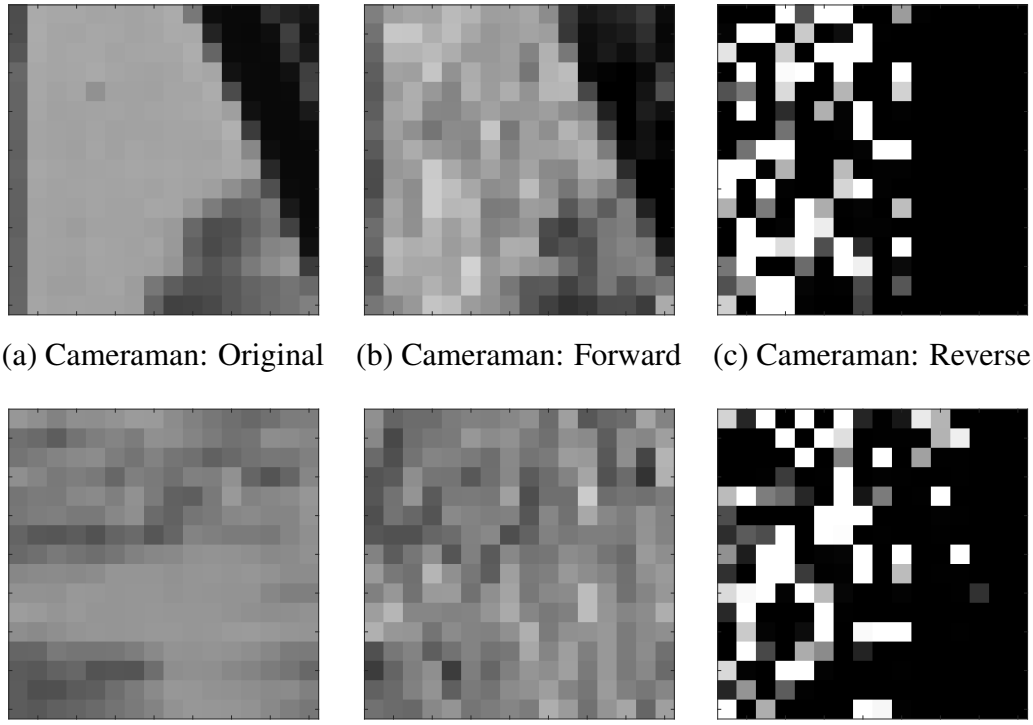


Figure 2.4: Histograms display distributions of individual observations in forward and reverse directions of multiplication in the construction of the analysis operator $\boldsymbol{D}$. Forward direction distributions display more desirable values, i.e., smaller for $Loops$, $Time$, v_0, and v_1, and higher for $PSNR$ and $SSIM$.

Visual examination of two randomly chosen image patch reconstructions verified the extreme inaccuracy indicated by the reverse order, to the point that one would not even consider those to be representations of the original. Figure 2.5 shows the forward and reverse reconstruction results of two randomly chosen image patches.

The extremity of the results provides a clear indication that the forward direction is the correct order of multiplication, and this direction was used for all experiments described in this research.



(a) Cameraman: Original (b) Cameraman: Forward (c) Cameraman: Reverse

(d) Boat: Original (e) Boat: Forward (f) Boat: Reverse

Figure 2.5: Originals of two randomly chosen image patches from the Cameraman and Boat test images and reconstructions using forward and reverse directions of multiplication of D. The reconstructions for the reverse directions are clearly inferior to the forward direction results.

2.6 Chapter Summary

This chapter reported many of the details that were important in the implementation of the representation algorithm being characterized. Time tests were used to determine that Weak Matching Pursuit (WMP) would be best suited for Step 1 of the algorithm, while the MATLAB function *fminsearch* would be best for Step 2. It was found, however, that the MATLAB function for WMP did not minimize the entire objective function, only the error term $\|\boldsymbol{A}^*\boldsymbol{D}\boldsymbol{v} - \boldsymbol{g}\|_2^2$. A version of the WMP function was written specifically to include $\|\boldsymbol{v}\|_1$.

Additional details that were outlined include two different filterbanks to initialize the frame matrix D, metrics to measure performance under different scenarios, the test images that were used to examine the proposed parameter variations, and a comparison of two image formats, i.e., methods of representing the brightness of pixels in the test images. Alternate forms of the error term in the objective function were considered, and a verification of the multiplication direction of the blocks making up the frame matrix D was shown.

Chapter 3

Parameters

Some of the parameters used in the implementation of the algorithm were related to settings for a MATLAB solver, while others were part of the objective function that is minimized. This chapter explores choices of tolerance parameters for MATLAB solver *fminsearch* and regularization parameters that control emphasis of the terms of the objective function minimized in the two-step algorithm studied here.

3.1 Tolerance Parameters for fminsearch

The MATLAB solver $fminsearch$ has two tolerance parameters that govern its termination, and these were examined with the goals of improving execution duration and sparsity, while maximizing reconstruction accuracy. The first, $TolX$, sets a tolerance for the step size in the independent variable (i.e., the filterbank h) and the second, $TolFun$, sets a tolerance for the change in the output of the objective function (i.e., Equation (2.1)), and the algorithm stops when both changes are smaller than the specified tolerances. Matlab sets the default values of these parameters as $TolX = 10^{-4}$ and $TolFun = 10^{-4}$.

For this experiment, ranges of values were explored for the parameters $TolX$ and

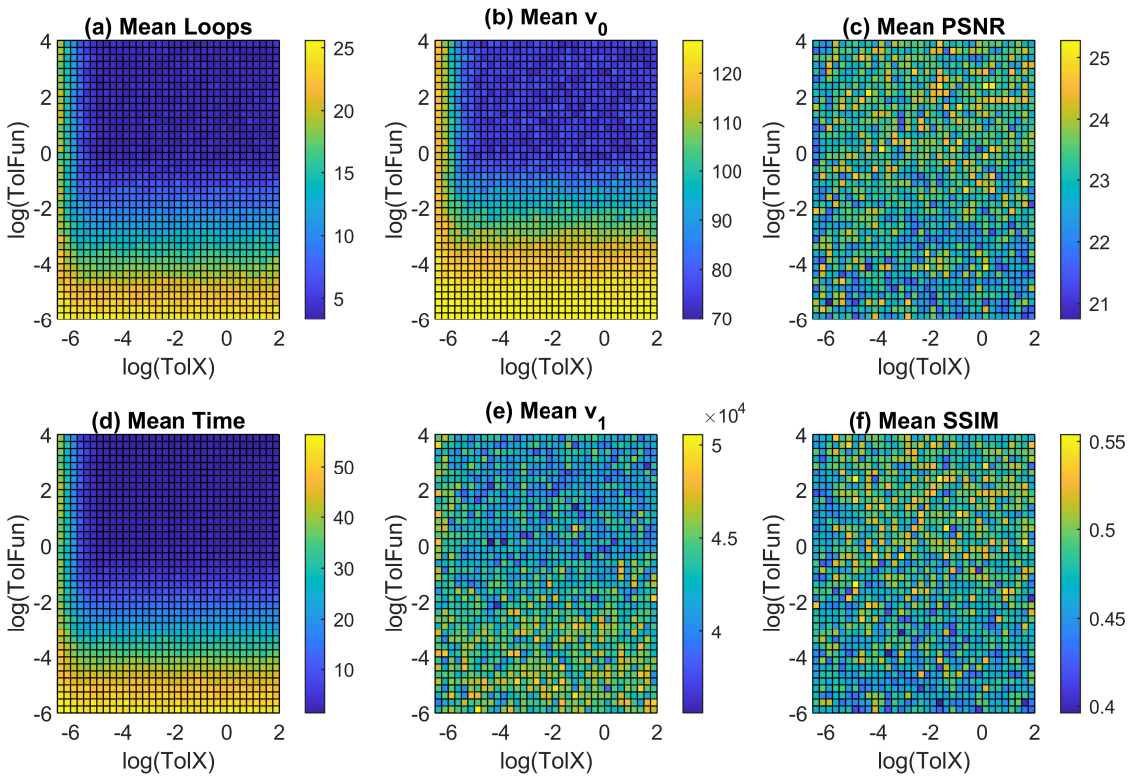


Figure 3.1: Heat maps displaying means of the metrics by termination tolerance parameters $TolFun$ and $TolX$, used in MATLAB solver *fminsearch*. Log scales are used for both axes.

$TolFun$. They are $TolX = 10^j$, where $j \in \{-6.5, -6.25, \ldots, 1.75, 2\}$, and $TolFun = 10^i$, where $i \in \{-6, -5.75, \ldots, 3.75, 4\}$. There were $n = 50$ observations collected for each $(TolX, TolFun)$ combination. The ranges for each of the tolerance parameters include the default values, but they are extended in the direction that preliminary exploration suggested the optimal values might exist for this application. Heatmaps displaying means of the six metrics are shown in Figure 3.1.

Except at the edge where $TolX \leq 10^{-6}$, the means for $Loops$, $Time$, and v_0 clearly display movement toward optimal values as $TolFun$ increases. For $TolX > 10^{-6}$ the means change only when $TolFun$ changes, indicating $TolX$ has no effect on these three metrics in this interval. The plots also indicate that using the MATLAB default value $TolFun = 10^{-4}$ makes the algorithm work harder and brings in more non-zeros entries of $\boldsymbol{v}$, which in turn, accesses more columns of $\boldsymbol{D}$. Based on only these three plots, choosing $TolX > 10^{-6}$ and

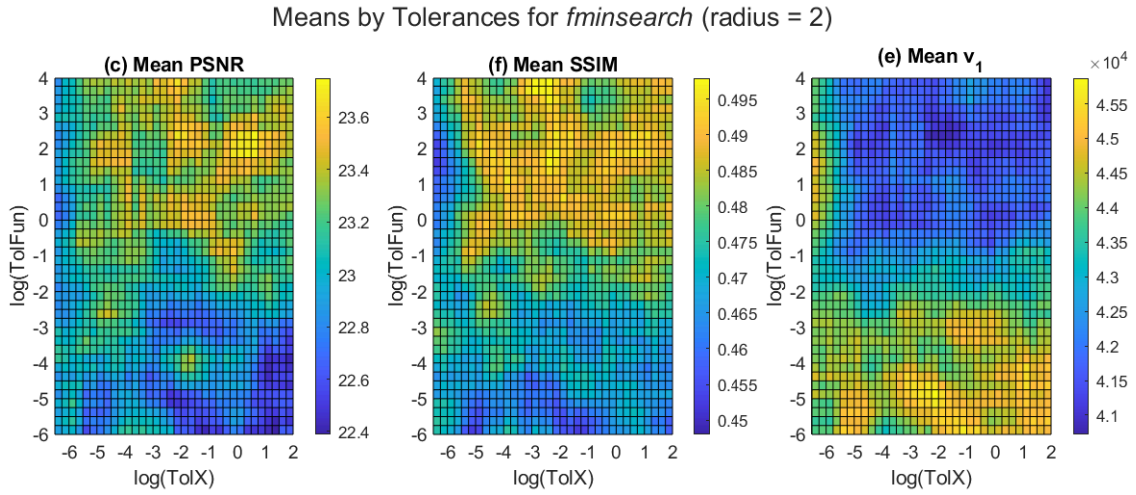


Figure 3.2: Heat maps displaying blurred versions of plots in Figure 3.1 for $PSNR$, $SSIM$, v_1. Each point is an average of the data within 2 grid points (horizontally and vertically) around each $(TolX, TolFun)$ combination.

$TolFun > 10^{-1}$ would yield more optimal values for $Loops$, $Time$, and v_0.

In contrast, for the mean plots of $PSNR$, $SSIM$, and v_1 in Figure 3.1, it is unclear whether the high and low values are concentrated in certain regions or distributed across the space. It appears that the upper half of each of these plots shows more mean values in the optimal direction than the lower half, but to explore this further, means were calculated using all data within 2 grid points (horizontally and vertically) around each ($TolX$, $TolFun$) combination. In Figure 3.2, each point is an average of k grid points where $k \in \{25, 20, 16, 15, 12, 9\}$, depending on how close the point is to the boundaries of the parameter space. Now it is clear from these blurred versions of the plots that the upper half of each of these will contain optimal values for $TolFun$. The results observed for $Loops$, $Time$, and v_0 combined with those from the blurred plots for $PSNR$, $SSIM$, and v_1 suggest that $TolX \in [10^{-5}, 10^{2}]$ and $TolFun \in [10^{-1}, 10^{4}]$ would be good values for the use of *fminsearch* in Step 2 of this representation algorithm.

3.2 Regularization Parameters

When minimizing an objective function, it is common to place a large (positive) regularization parameter, or weight, on the term that needs more emphasis (rather than placing a small scalar on the term that needs less emphasis). These values must be chosen, but there is nothing inherent to this problem that would indicate good values, nor is it clear what the authors of the algorithm used. They did include a regularization parameter for Step 1, but for Step 2 it seems likely they used the Lagrange factor method [37] in order to enforce the Parseval constraint shown in Equation (1.15). The Lagrange factor method was tested, but the run-time took far too long and limited the feasible number of contexts that could be explored. This led to a reformulation of Equation (1.15) as Equation (2.1). For Step 1, the regularization parameter, μ_1, used by the authors was retained. Its purpose is to apply emphasis on driving the error term to be zero, while the 1-norm term keeps the coefficient vector $\boldsymbol{v}$ as sparse as possible. For Step 2, Equation (2.1) was developed by converting the Parseval constraint in Equation (1.15) to a new term in the objective function. The inclusion of the regularization parameter, μ_2, enforces that constraint by pushing the frame operator $\boldsymbol{D}\boldsymbol{D}^*$ toward the identity matrix and $\boldsymbol{D}$ toward being Parseval.

The process of selection of the values of μ_1 and μ_2 and the motivation behind it is described in the sections below. The following articles are a small sample of the still growing field of research on regularization for discrete ill-posed problems [38–41].

It is notable that a large emphasis is highly desired on terms which should be zero, but starting with a value that is too large can make the optimization sensitive to the initial conditions, and it may get locked into an undesirable local minimum early on. One way to avoid this is to start with a smaller value and increase it over the execution of the algorithm. This permits more flexibility to move across the search space, while still ending the optimization with a large emphasis on the desired terms. To incorporate this paradigm, constant and increasing values for both μ_1 and μ_2 were explored.

A sensitivity analysis was performed for each of the four scenarios in which these

parameters were needed. The following sections lay out the method and specific details used to determine the regularization parameters included in the algorithm. It was not assumed that one pair of parameters would be appropriate in all settings, and the general method was repeated for the cases shown below. The goal was to prioritize accuracy of the reconstruction, followed by execution speed.

The results of the sensitivity analyses in searching for optimal regularization parameters for three of the scenarios appear in the sections below. However, the fourth scenario required more information about Parseval Relaxation and is included in Chapter **??**.

3.2.1 Regularization Parameters: Integer Format and 3×3 Filterbank

To do a thorough sensitivity test, both constant and increasing functions of the loop index were tested for each of the regularization parameters. After a bit of preliminary exploration, the first set of test functions were chosen for μ_1 and μ_2. The initial sets of functions spanned a broad parameter space, but without much resolution. The constant functions tested were 10^x, where $x \in \{5, 10, 15, \ldots, 40\}$. The increasing functions tested were B^t, where t was the loop index of the representation algorithm, and $B \in \{\sqrt{2}, 2, 5, 10\}$. All possible combinations of μ_1 and μ_2 were examined for $n = 50$ data points, and because the parameter space was large, smaller 8×8 image patches were utilized. It is generally desirable to seek parameters that will minimize values of $Loops$, $Time$, and v_0, but maximize values of $PSNR$ and $SSIM$.

The results for this first stage of testing are summarized as descriptive statistics shown in Table 3.1. This was a first attempt to convey the results concisely, and the data tables from which these statistics were calculated can be seen in Appendix A.1. First, it was noted that regardless of μ_2, a constant function for μ_1 was not optimal. When μ_1 is constant, the number of $Loops$ required to reach convergence was very small and showed little variability as measured by the standard deviations (SD = 0.5 when μ_2 is constant and SD = 0.0 when μ_2 is increasing). For most of these (μ_1, μ_2) combinations, the algorithm reached convergence

Table 3.1: Descriptive statistics based on means for round 1 tests of constant and increasing μ_1 and μ_2. Data collected utilizing integer image format and 3×3 filterbank.

		Constant μ_2				Increasing μ_2			
		Loops	Time (s)	PSNR	SSIM	Loops	Time (s)	PSNR	SSIM
	Max	3.0	2.6	22.8	0.51	2.0	0.2	22.8	0.51
Constant	Mean	2.4	1.3	22.8	0.51	2.0	0.2	22.8	0.51
μ_1	Mode	2.0	2.6	22.8	0.51	2.0	0.2	22.8	0.51
	Min	2.0	0.2	22.8	0.51	2.0	0.2	22.8	0.51
	SD[a]	0.5	1.1	0.0	0.00	0.0	0.0	0.0	0.00
	Max	23.1	8.4	23.4	0.54	42.5	18.4	23.2	0.53
Increasing	Mean	6.5	3.5	23.0	0.52	31.3	11.6	22.9	0.52
μ_1	Mode[b]	3.5	3.2	23.4	0.53	—	—	22.9	0.52
	Min	3.3	0.9	22.8	0.51	18.2	4.0	22.6	0.50
	SD	4.6	1.8	0.2	0.01	6.8	4.6	0.2	0.01

[a] Standard Deviation
[b] No mode exists: —

after only 2 iterations. Because the algorithm detects convergence after a final loop where both $\boldsymbol{v}$ and $\boldsymbol{D}$ are essentially unchanged, converging when $Loops = 2$ means each step was executed just once in the first loop. One small region of the parameter space having constant μ_1 showed that 3 loops were executed, however, neither of these $Loops$ values indicates a robust execution of the two-step algorithm. Because $Time$, $PSNR$, and $SSIM$ also showed little variability in this part of the parameter space, all combinations where μ_1 was a constant function were rejected.

In the next part of the parameter space to be examined both μ_1 and μ_2 were increasing functions. Both $PSNR$ and $SSIM$ have reasonably good results with Max, Min, and Mean all close to $PSNR = 23$ and $SSIM = 0.52$, but $Loops$ and $Time$ display unwanted, large values. This observation suggested the combination of increasing μ_1 and increasing μ_2 should be avoided. Finally, for the last part of the parameter space, μ_1 was increasing and μ_2 was constant. In this region, the values of all four metrics were within reasonable ranges and

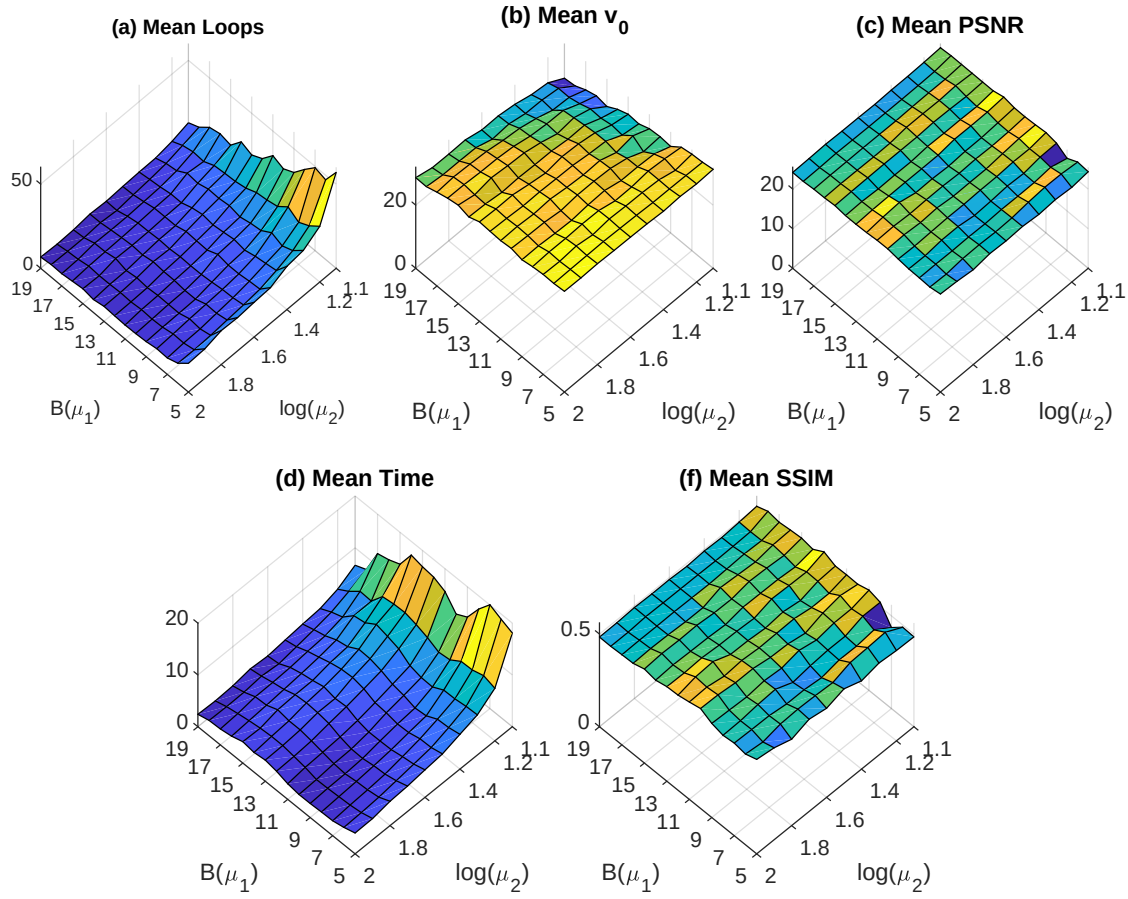


Figure 3.3: Surface plots based on means for round 2 tests of increasing μ_1 and constant μ_2. Data collected utilizing integer image format and 3×3 filterbank. The base B of μ_1 is shown on the left axis and the exponent of μ_2 is shown on the right axis in reverse order.

indicate the algorithm is executing as expected. By process of elimination, the conclusion was to use μ_1 as an increasing function of the loop index t and μ_2 as a constant function.

Based on the values observed and the conclusions noted in the paragraph above, the second, more refined parameter search was focused on the area with better results and included increased resolution. The increasing functions tested in the second round were $\mu_1 = B^t$, where $B \in \{1.1, 1.2, \ldots, 2.0\}$. The constant functions tested were $\mu_2 = 10^x$, where $x \in \{5, 6, \ldots, 19\}$.

Results of the second set of tests with refinements as described above are shown in Figure 3.3 as surface plots of the metric means. In Figure 3.3(a) the means of *Loops* decrease as both $B(\mu_1)$ and $\log(\mu_2)$ increase. In Figure 3.3(d) the *Time* plot shows similar

behavior. The plot for v_0 shown in Figure 3.3(b) shows a decrease (i.e. increasing sparsity) as $B(\mu_1)$ increases and as $\log(\mu_2)$ decreases. Finally, $PSNR$ and $SSIM$ do not seem to exhibit any trends except, perhaps, a tendency to decrease when $\log(\mu_2)$ gets larger. After examining all these plots more carefully, it was determined that for future analyses that utilize the integer format and 3×3 filterbank, the regularization parameters employed would be $\mu_1 = 1.4^t$ and $\mu_2 = 10^{13}$.

3.2.2 Regularization Parameters: Floating Point Format

The effect on this algorithm of using a floating point format to store the image pixel values was explored. It was necessary to select a potentially different pair of regularization parameters to be used when floating point image formats were utilized. The method used to choose appropriate regularization parameter values followed along the same lines as that used for the default integer format with 3×3 filterbank shown in Section 3.2.1.

In Round 1 of testing, both constant and increasing functions were included in the search for μ_1 and μ_2. The constant functions tested were 10^x, where $x \in \{5, 10, 15, \ldots, 40\}$. The increasing functions tested were B^t, where t was the loop index of the representation algorithm, and $B \in \{\sqrt{2}, 2, 5, 10\}$. As was noted for the first regularization parameter search, only the combinations that included increasing μ_1 and constant μ_2 resulted in appropriate values for all performance metrics. The tables containing Round 1 means, based on $n = 20$ observations for all combinations and for all metrics are shown Appendix A.2.

The round 2 search restricted the parameter space to increasing $\mu_1 = B^t$ where $B \in \{1.3, 1.5, 1.7, \ldots, 3.9, 4.1\}$, and constant $\mu_2 = 10^x$, where $x \in \{5, 7, \ldots, 19\}$. Results for the means for all the metrics are shown as surface plots in Figure 3.4. The metrics $Loops$, $Time$, and v_0 show the most desirable values where $\log(\mu_2)$, is smallest. Unfortunately, this part of the parameter space is also where the accuracy measures, $PSNR$ and $SSIM$, are smallest and least desirable.

Because a good pair of choices for μ_1 and μ_2 was not evident in the Round 2 results,

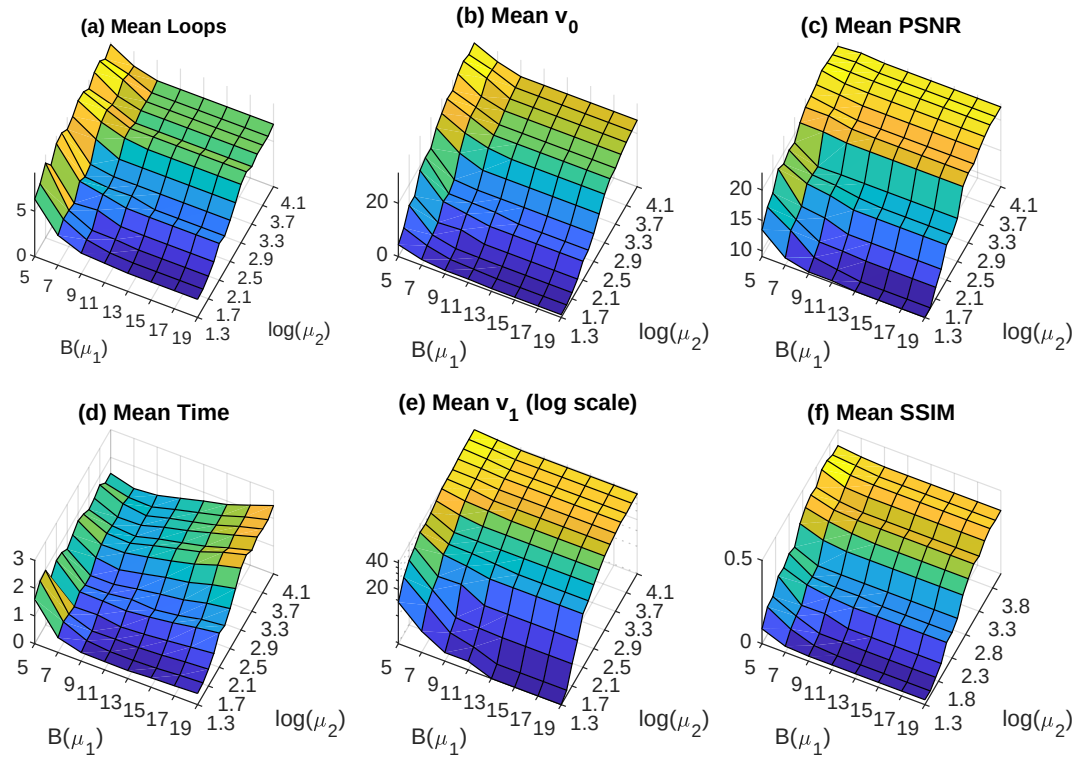


Figure 3.4: Surface plots based on means for round 2 tests of increasing μ_1 and constant μ_2. Data collected utilizing floating point image format and 3×3 filterbank. The base B of μ_1 is shown on the horizontal axis and the exponent of μ_2 is shown on the vertical axis.

a third search was performed with an expanded search area. The values were extended to test increasing $\mu_1 = B^t$ where $B \in \{4.0, 4.1, , \ldots, 4.4, 4.5, 5, 6, \ldots, 14, 15\}$, and constant $\mu_2 = 10^x$, where $x \in \{3, 4, 5, 6, 7\}$. The tables with conditional formatting are shown in Figures 3.5, 3.6, and 3.7. Note that more desirable values are formatted in green, while less desirable values are formatted in red.

Figure 3.5 indicates that larger values of $B(\mu_1)$, labeled as $Mu1$, will yield faster execution time and fewer loops for convergence of the algorithm. Figure 3.6 points to two pairs of values within that area of the parameter space that appear likely to provide larger accuracy. Although smaller values of $B(\mu_1)$ might have reduced sparsity slightly, it was determined that $\mu_1 = 14^t$ and $\mu_2 = 10^5$ would best for floating point image format data.

Mean Loops	Mu2 / Constant	1E+03	1E+04	1E+05	1E+06	FPR3 1E+07
Mu1	4.0	9.90	9.30	8.80	8.95	8.10
	4.1	9.65	9.35	8.75	8.50	8.20
	4.2	9.95	9.55	8.85	8.30	7.95
	4.3	9.95	9.45	8.60	8.55	8.15
Increasing	4.4	10.10	9.55	8.60	8.25	8.25
	4.5	9.10	9.40	8.45	8.35	8.25
	5.0	9.20	8.65	7.95	7.65	7.40
	6.0	8.40	8.25	7.55	7.40	7.35
	7.0	7.70	7.85	6.60	6.60	6.25
	8.0	7.65	7.60	6.85	6.15	6.35
	9.0	7.60	7.00	6.90	6.25	6.30
	10.0	7.60	7.20	6.85	6.15	6.05
	11.0	6.80	7.20	6.50	5.75	5.95
	12.0	7.25	6.85	6.40	5.85	5.70
	13.0	6.90	6.40	6.40	5.90	5.80
	14.0	7.10	6.60	6.20	5.85	5.75
	15.0	6.45	6.40	6.15	5.55	5.45

Mean Time	Mu2 / Constant	1E+03	1E+04	1E+05	1E+06	FPR3 1E+07
Mu1	4.0	3.25	2.28	1.64	1.60	1.42
	4.1	3.23	2.30	1.59	1.50	1.44
	4.2	3.27	2.34	1.56	1.48	1.38
	4.3	3.18	2.30	1.54	1.52	1.43
Increasing	4.4	3.06	2.27	1.49	1.51	1.44
	4.5	2.87	2.19	1.47	1.54	1.46
	5.0	2.70	2.08	1.36	1.36	1.28
	6.0	2.46	2.00	1.24	1.35	1.24
	7.0	2.09	1.77	1.10	1.05	1.01
	8.0	1.86	1.57	0.98	0.93	1.03
	9.0	1.87	1.43	0.99	0.96	1.00
	10.0	1.83	1.38	0.94	0.95	1.00
	11.0	1.63	1.37	0.85	0.80	0.97
	12.0	1.73	1.29	0.86	0.83	0.92
	13.0	1.67	1.09	0.91	0.85	0.90
	14.0	1.64	1.07	0.86	0.83	0.88
	15.0	1.51	1.06	0.84	0.78	0.83

Figure 3.5: Conditionally formatted tables displaying round 3 means for *Loops* and *Time* in search for regularization parameters of floating point data with 3×3 filterbank. Mu1 is the base of $\mu_1 = B^t$ and Mu2 is μ_2. More desirable values are formatted in green, less desirable values are formatted in red.

Mean PSNR	Mu2 / Constant	1E+03	1E+04	1E+05	1E+06	FPR3 1E+07
Mu1	4.0	21.96	22.41	23.72	24.86	24.47
	4.1	22.24	22.45	23.89	24.83	24.70
	4.2	22.36	22.49	24.02	25.00	24.72
	4.3	22.22	22.26	23.89	24.80	24.84
Increasing	4.4	22.09	22.99	24.21	24.72	24.84
	4.5	21.60	23.27	24.32	24.73	24.83
	5.0	22.67	23.48	24.29	24.46	24.94
	6.0	22.86	22.70	24.94	24.83	24.96
	7.0	23.16	23.61	24.79	24.84	24.58
	8.0	23.39	23.59	24.86	24.85	24.85
	9.0	23.17	23.25	24.82	25.08	24.97
	10.0	23.38	24.09	24.51	24.94	25.01
	11.0	23.47	23.21	24.50	24.76	24.97
	12.0	22.79	24.56	24.71	24.86	24.96
	13.0	23.38	24.62	25.39	24.89	24.84
	14.0	23.74	24.77	25.36	24.89	24.84
	15.0	24.27	24.71	25.10	25.00	24.84

Mean SSIM	Mu2 / Constant	1E+03	1E+04	1E+05	1E+06	FPR3 1E+07
Mu1	4.0	0.362	0.371	0.432	0.470	0.453
	4.1	0.372	0.387	0.428	0.474	0.462
	4.2	0.375	0.396	0.430	0.473	0.456
	4.3	0.380	0.385	0.422	0.472	0.469
Increasing	4.4	0.368	0.414	0.437	0.467	0.469
	4.5	0.350	0.430	0.445	0.467	0.468
	5.0	0.387	0.445	0.450	0.461	0.472
	6.0	0.385	0.405	0.476	0.480	0.479
	7.0	0.401	0.423	0.481	0.474	0.467
	8.0	0.410	0.430	0.484	0.478	0.478
	9.0	0.408	0.424	0.469	0.496	0.495
	10.0	0.424	0.457	0.462	0.488	0.497
	11.0	0.427	0.411	0.458	0.481	0.494
	12.0	0.390	0.478	0.472	0.488	0.494
	13.0	0.415	0.472	0.516	0.489	0.485
	14.0	0.430	0.479	0.516	0.489	0.485
	15.0	0.458	0.480	0.503	0.491	0.485

Figure 3.6: Conditionally formatted tables displaying round 3 means for *PSNR* and *SSIM* in search for regularization parameters of floating point data with 3×3 filterbank. Mu1 is the base of $\mu_1 = B^t$ and Mu2 is μ_2. More desirable values are formatted in green, less desirable values are formatted in red.

Mean V0	Mu2	Constant			FPR3
	1E+03	1E+04	1E+05	1E+06	1E+07
Mu1 4.0	31.60	31.60	30.55	30.70	29.35
4.1	31.10	31.70	30.60	30.60	30.35
4.2	31.35	31.70	31.60	30.05	30.00
4.3	31.40	31.80	31.55	31.50	31.20
4.4	31.70	31.75	31.65	30.60	31.35
4.5	31.55	31.80	31.75	30.60	31.50
5.0	31.75	31.45	31.70	30.50	30.40
6.0	31.75	31.80	31.55	31.70	31.80
7.0	31.75	31.75	30.55	29.70	28.65
8.0	31.65	31.70	30.75	29.65	29.70
9.0	31.70	31.80	31.60	31.90	31.90
10.0	31.75	31.80	31.85	31.85	31.85
11.0	31.75	31.70	31.90	31.65	31.90
12.0	31.85	31.90	31.80	31.65	31.85
13.0	31.95	31.90	31.85	31.85	31.90
14.0	31.95	31.90	31.90	31.90	31.90
15.0	31.95	31.75	31.90	31.90	31.90

Increasing

Figure 3.7: Conditionally formatted tables displaying round 3 means for v_0 in search for regularization parameters of floating point data with 3×3 filterbank. Mu1 is the base of $\mu_1 = B^t$ and Mu2 is μ_2. More desirable values are formatted in green, less desirable values are formatted in red.

3.2.3 Regularization Parameters: Integer Format and 3×3 Filterbank

This research included the use of a larger 5×5 filterbank, shown in Equation (2.3) and described in [33]. Before using this filterbank, a sensitivity analysis was performed to determine if the regularization parameters used in the research with the 3×3 filterbank should be adjusted to values more appropriate for the larger 5×5 filterbank.

A small number of observations showed that, as in the testing for 3×3 filterbanks and for floating point data, only the (μ_1, μ_2) combinations for which μ_1 was increasing and μ_2 was constant produced results that were considered reasonable. The search was focused on the area with better results and included increased resolution. The increasing functions tested were $\mu_1 = B^t$, where $B \in \{1.1, 1.2, \ldots, 1.7\}$. The constant functions tested were $\mu_2 = 10^x$, where $x \in \{12, 13, \ldots, 18\}$.

The results for the performance metrics in this round of tests are shown as surface plots in Figure 3.8. Examination of the plots indicated that only a slight change from the

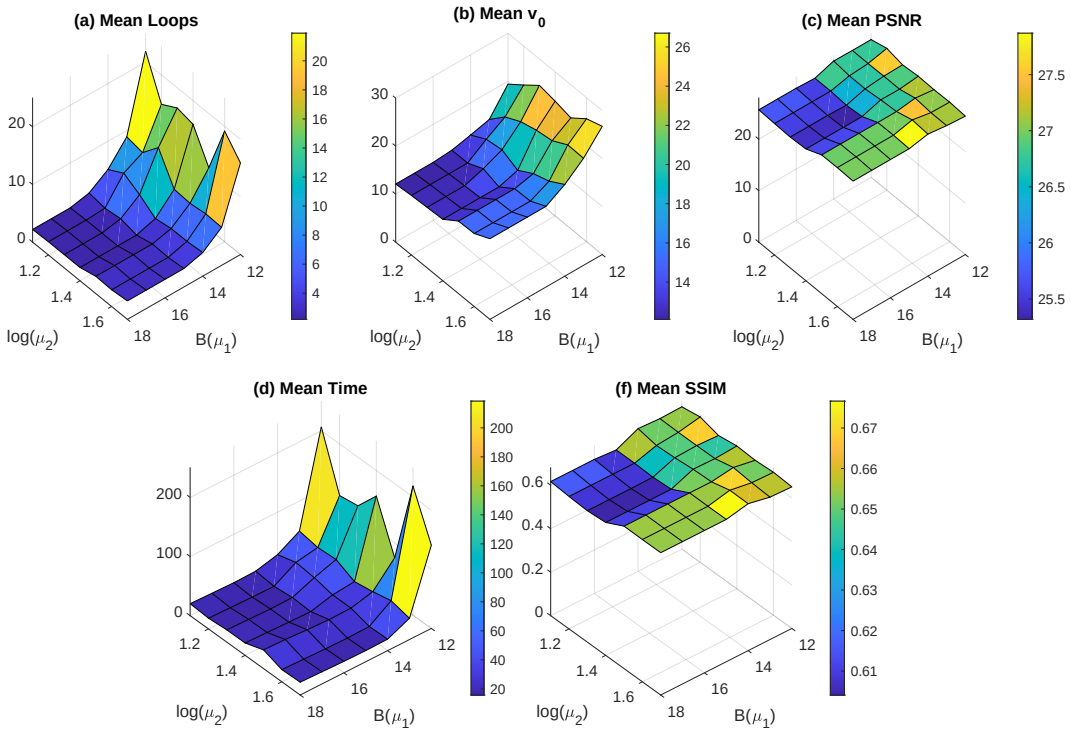


Figure 3.8: Surface plots based on means for round 2 tests of increasing μ_1 and constant μ_2. Data collected utilizing integer image format and 5×5 filterbank. The base B of μ_1 is shown on the horizontal axis and the exponent of μ_2 is shown on the vertical axis.

regularization parameters used for the 3×3 filterbank would result in better results for all five metrics whose means were observed. The final choice of $\mu_1 = 1.6^t$ and $\mu_2 = 10^{14}$ yielded better accuracy without increasing execution time and sparsity too much.

3.3 Chapter Summary

Two groups of parameters were examined in this chapter. The first was the pair of parameters needed for the MATLAB function *fminsearch*. *TolX* sets the tolerance for the stepsize of the independent variable, the filterbank h, and *TolFun*, sets the tolerance for the change in the objective function used in Step 2. The algorithm exits when both changes are smaller than the specified tolerances. It was found that tolerance values resulting in more desirable values for all metrics are $TolX \in [10^{-5}, 10^{2}]$ and $TolFun \in [10^{-1}, 10^{4}]$.

The second group to be explored included the pairs of regularization parameters μ_1 in Step 1 and μ_2 in Step 2. It was found that different pairs of regularization parameters were appropriate for different settings. Three of these settings were : (1) integer format and 3×3 filterbank, (2) floating point format and 3×3 filterbank, and (3) integer format and 5×5 filterbank. The regularization parameters are shown in Table 3.2. A fourth setting required information not yet developed, and is shown in Chapter **??**.

Table 3.2: Regularization parameters for first three settings, where *t* is the loop index of the representation algorithm.

Image Format	FB	μ_1	μ_2
Integer	3×3	1.4^t	10^{13}
Floating Point	3×3	14^t	10^5
Integer	5×5	1.6^t	10^{14}

Chapter 4

Numerical Experiments

Numerical experiments were performed to determine the effect that different variables had on the effectiveness of the representation algorithm, as measured by the performance metrics. This chapter describes these explorations and draws conclusions, where possible. Section 4.1 examines how the size of the image patch would affect the performance metrics. The results were expected to depend on the maximum dilation level L, and results were compared using $L = 3$, the default maximum dilation level used by the authors, and $L = MaxL$, the maximum possible dilation level based on image and filter sizes. Section 4.2 examines the results when $3 \leq L \leq MaxL$ for image patches having number of pixels $N = 2^j$ where $j \in \{3, 4, \ldots, 10\}$. The image patch sizes were chosen because these N showed important results in Section 4.1. Section 4.3 explores the choice of relaxation parameter, w, specified in the WMP function used in Step 1 of the representation algorithm. This exploration was expanded to include the interaction of the choice of w with the sampling rate p, which governed the number of columns in the sensing matrix A. Section 4.4 compares the performance metrics between usage of filterbanks of size 3×3 and 5×5, and Section 4.5 makes a comparison between the two image formats described in Section 2.4. The last section in this chapter pulls together parameters in a three-way analysis to find interactions among filterbank sizes, image formats and maximum dilation levels.

4.1 Algorithm Behavior for Number of Pixels Between Powers of 2

The initial goal for this experiment was to characterize the best performance of the representation algorithm in terms of accuracy. Authors of the original article used $L = 3$, but it seemed likely that a larger maximum dilation level might yield better results by providing more frame vectors to choose from, so $L = MaxL$ was included. Data for this exploration was generated using different image patch sizes, and for each N, used the largest dilation level possible. Because the filters have length $b = 3$, Equation (2.5) becomes the simplified relationship between N and $MaxL$ shown here.

$$MaxL = \text{floor}(1 + \log_2(N - 1))$$
$$N \in \left(2^{j-1}, 2^j\right] \quad \Longleftrightarrow \quad MaxL = j$$

Image patch dimensions were chosen so that N is the product of height (HT) and width (WD), where HT and WD are integers and the aspect ratio was restricted to the interval $\left[\frac{1}{2}, 2\right]$. If more than one pair of dimensions produced the same value of N, the pair with aspect ratio closest to 1 was used. There were $n = 112$ observations collected for each of the 178 possible values of $N \in [6, 512]$. For each of the patch sizes, because they were vectorized vertically, the observations were evenly split between tall and wide patches, to avoid the orientation becoming a confounding factor.

In the results, a pattern was noted in the behavior of some performance metrics within each level of $MaxL$. To highlight this pattern, in Figure 4.1 each interval of $N \in \left(2^{j-1}, 2^j\right] = (0.5, 1] \cdot 2^j$ was scaled to the interval $(0.5, 1]$. Figure 4.1 shows a peak at the middle of each scaled interval for $Loops$, $Time$, and v_1, becoming more pronounced as N and $MaxL$ increase. $Loops$ and v_1 can have a significant drop after the middle of the Scaled N axis. $Time$ also appears to have a dip, but a much smaller one. The final observation to make

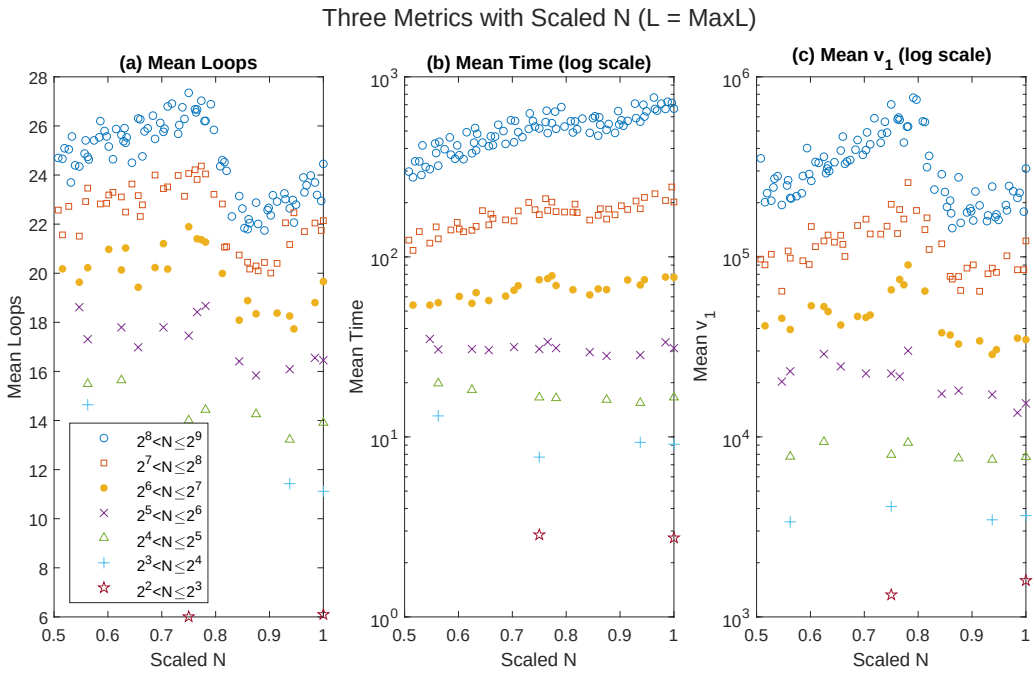


Figure 4.1: Means for $Loops$, $Time$, v_1 when $L = MaxL$. Each set of N values, having the same value of $MaxL$, has been scaled to $(0.5, 1]$ allowing a comparison of behavior among different values of $MaxL$.

from Figure 4.1 is to note that each of these metrics exhibits an upward step as N transitions from one interval to the next as $MaxL$ increments.

To see if this pattern held for smaller values of L, the plots for $Loops$, $Time$, and v_1 are shown in Figure 4.2, for $L = 3$. The same pattern between powers of 2 is not observed at this level of L, nor is there a stepped increase in these metrics as N transitions through powers of 2. It was expected these behaviors would become gradually present and more pronounced as L approached $MaxL$ (rather than due to some special phase shift that only occurs, for example, when L goes from $MaxL - 1$ to $MaxL$).

It seemed likely the 'swoosh' pattern in $Loops$ and v_1, and to a lesser extent in $Time$, must develop as L increases from $L = 3$ to $MaxL$. To explore that expectation, another set of plots examine the transition from $L = 3$ to $L = MaxL$ for the set of N such that $2^7 < N \leq 2^8$. These values of N have $MaxL = 8$, and the plots include sets of points for $L \in \{3, 4, 5, 6, 7, 8\}$. For the given set of N, it seems that the 'swoosh' develops more and

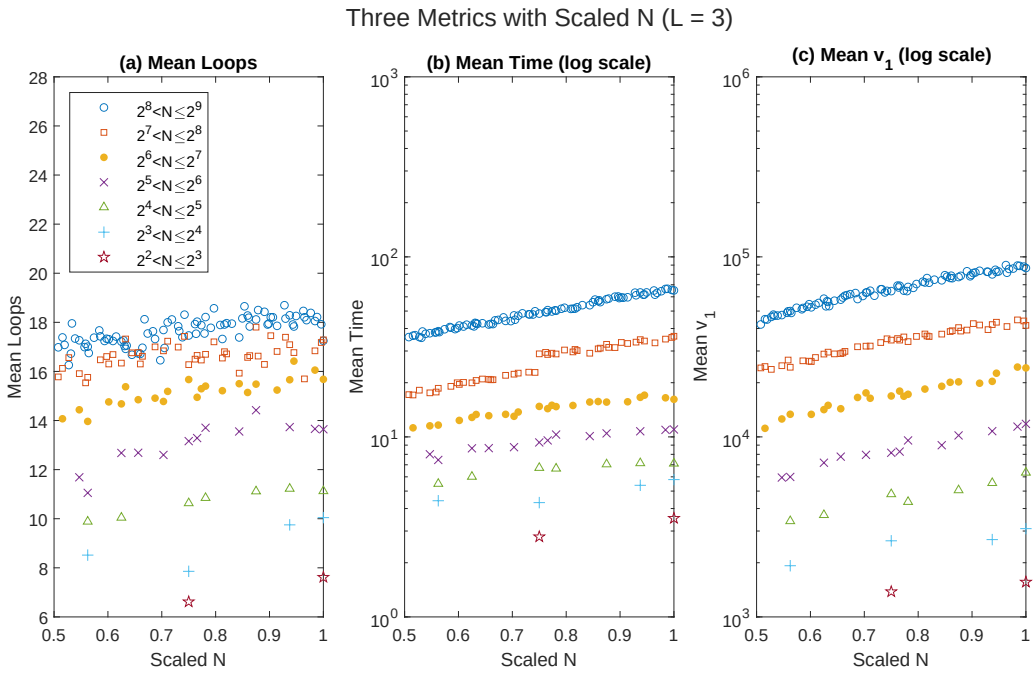


Figure 4.2: Means for $Loops$, $Time$, v_1 when $L = 3$. Each set of N values, having the same value of $MaxL$, has been scaled to $(0.5, 1]$ allowing a comparison of behavior among different values of $MaxL$. The jump in $Time$ occurring between $N = 190$ and 192 (shown in red squares) is likely due to machine architecture.

more as L increases, becoming obvious when $L = 7$. However, the stepped increase as N transitions through powers of 2 as seen in the $L = MaxL$ plots does not appear in the plots of Figure 4.3.

It was noted the $L = 3$ results exhibit an odd jump in $Time$ when N goes from 190 to 192. The jump is also seen in Figure 4.3 for $L = 4$, but begins to fade out for $L = 6$. This jump was observed on two different sets of data generated on the cluster, but the jump was not present there when run on the PC workstation. $Time$ is expected to vary from machine to machine, but the other metrics in the desktop results were consistent with the results from the cluster, so it is likely the observed gap is due to machine specific architecture of the cluster.

The other three metrics, v_0, $PSNR$, and $SSIM$, show no such sensitivity to $MaxL$ and exhibit smoother behavior as N increases, as shown in Figure 4.4 and Figure 4.5. It is worth noting here that the number of measurements $m = pN$ forms an upper bound for

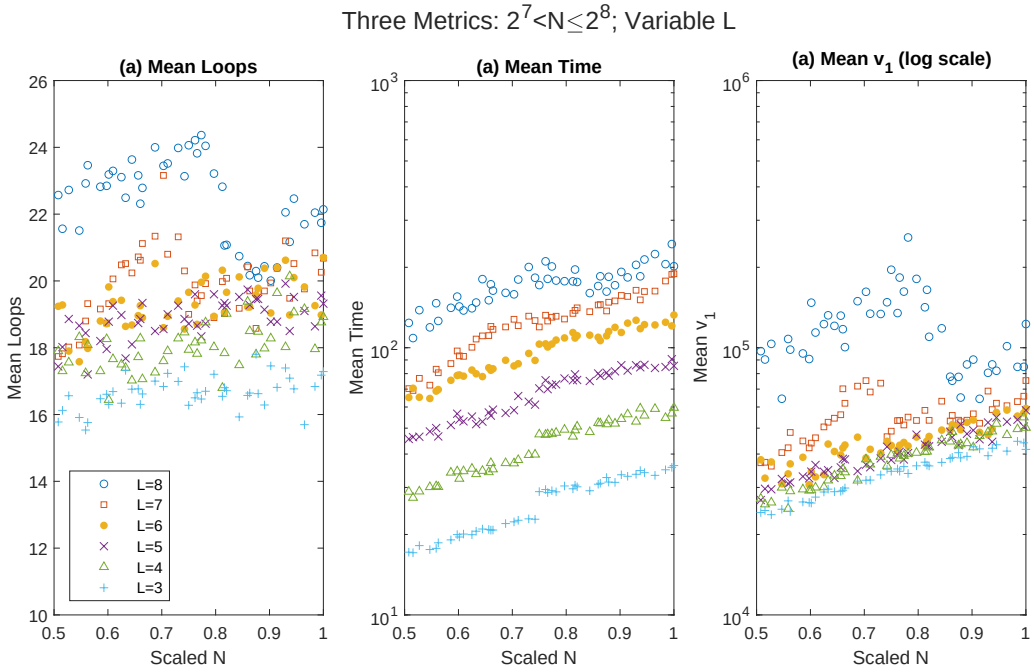


Figure 4.3: Means for *Loops*, *Time*, v_1 for $2^7 < N \leq 2^8$ and maximum dilation levels $L \in \{3, 4, 5, 6, 7, 8\}$. N has been scaled from $(2^7, 2^8]$ to $(0.5, 1]$.

v_0 (where p is the sampling rate). This occurs because the WMP function used in Step 1 sequentially selects linearly independent columns of $\boldsymbol{A}^*\boldsymbol{D}$ to represent $\boldsymbol{g}$ in $\mathbb{R}^m$ and stops when the residual is small. By the time m columns are selected, WMP has constructed a basis for $\mathbb{R}^m$ (so that the residual is the zero vector). These results were produced using $p = 0.5$, and a linear fit through the means of v_0 in Figure 4.4(a) has slope 0.436 (0.454 for Figure 4.4(b)), indicating that on average this algorithm will produce a vector $\boldsymbol{v}$ with 87% as many non-zero entries (91% for $L = 3$) as the number of measurements made.

Counterintuitively, an examination of Figures 4.5, (a) and (b), shows that after an initial increase for extremely small N, both *PSNR* and *SSIM* indicate a minor decline in accuracy as the sizes of image patches increased. More research is required to determine why this occurred. It should be noted that there is no relationship between the limits in the vertical axes for Figure 4.5. While *SSIM* is restricted to $[0, 1]$ (for our purposes), 40 was chosen as a nice round upper value for the *PSNR* axis, because it made the axes and data easy to compare. For $L = 3$, *PSNR* is slightly smaller but *SSIM* decreases more, when compared

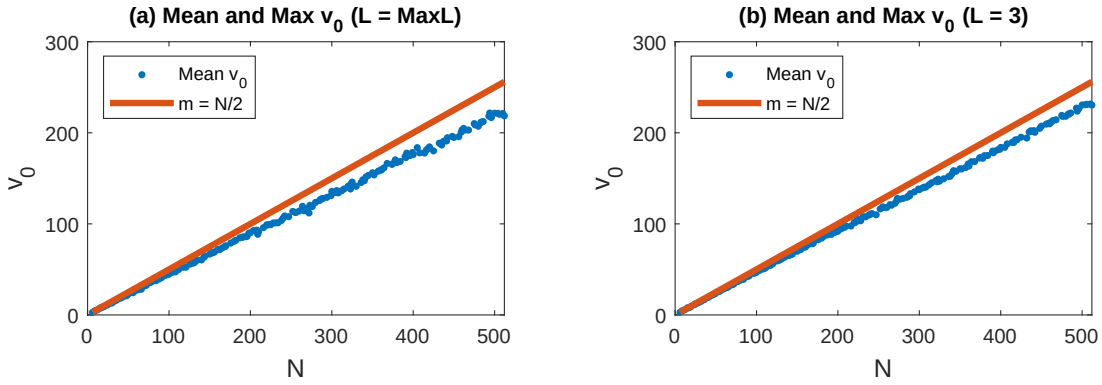


Figure 4.4: Means and maximum possible values of v_0. The red line is $m = pN = N/2$.

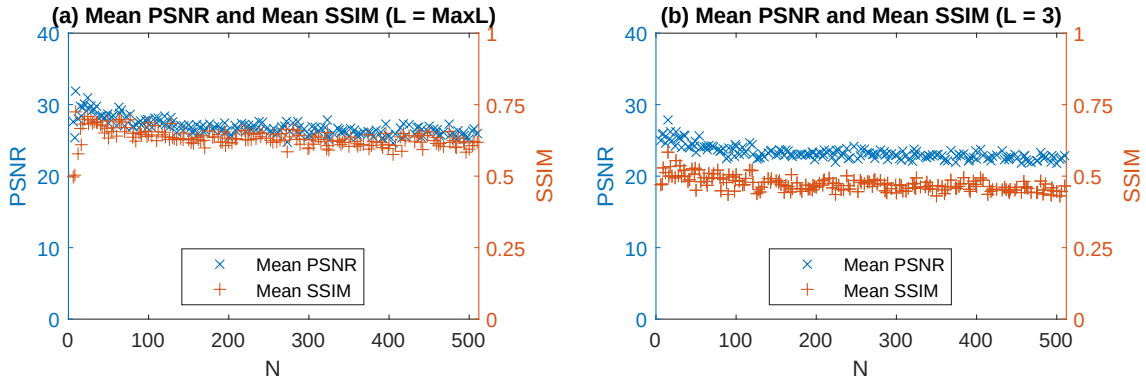


Figure 4.5: Mean $PSNR$ and Mean $SSIM$ for $L = MaxL$ and $L = 3$.

to these metrics for $L = MaxL$. These results are consistent with the findings in [34], and all values in Figure 4.5 are consistent with a curve just above the one highlighted in red in Figure 2 of that paper, corresponding with a covariance value slightly below $\sigma = 10^2$.

4.2 Setting Dilation Level: $L < MaxL$ vs $L = MaxL$

In the calculation of the Parseval frame $\boldsymbol{D}$, there are several things to consider when choosing between letting L take on its maximum possible value, or fixing it at some smaller value. The choice depends on parameters such as the length of filters in the filterbank, the number of pixels in the image, and the various costs of additional execution time as the value of L increases. This experiment attempts to determine what dilation level might be most appropriate for different image patch sizes and constraints on resources.

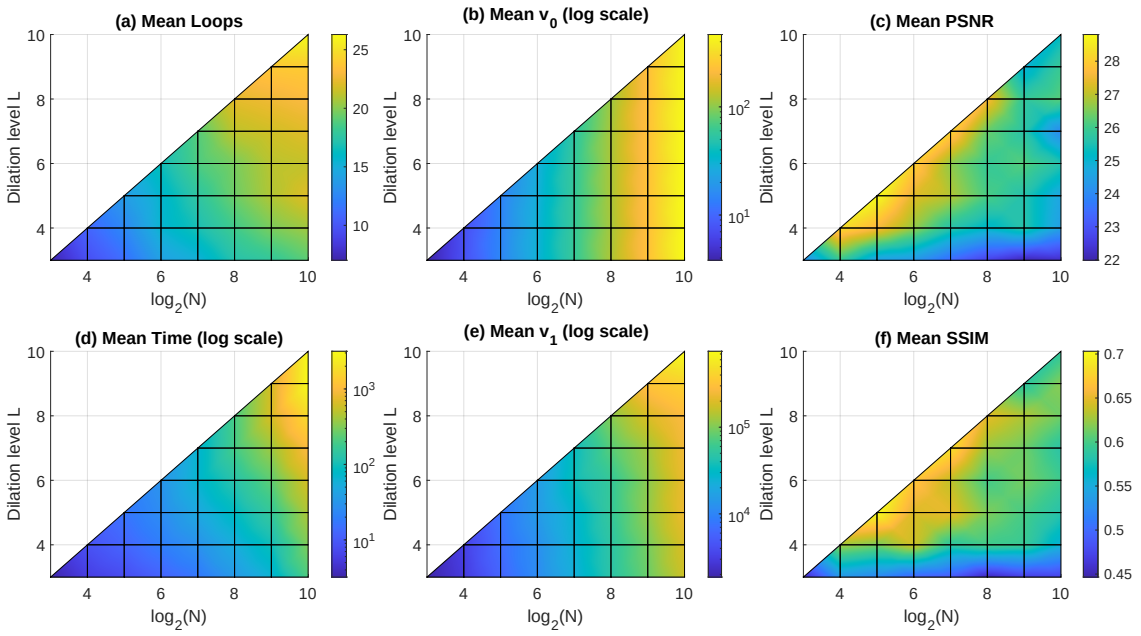


Figure 4.6: Heat maps of means for metrics with image sizes $N = 2^j$ for $j \in \{3, 4, \ldots, 10\}$ and dilation levels $L \in \{3, 4, \ldots, MaxL\}$. Means of $Time$, v_0 and v_1 are plotted on a log scale.

Figure 4.6 shows heatmaps of the means for the six performance metrics. Because of the jump in execution time going from $N = 2^j$ to $N = 2^j + 1$ shown in Figure 4.1(b), the data for this numerical experiment were generated using image patch sizes $N = 2^j$ for $j \in \{3, 4, \ldots, 10\}$. For each value of N the dilation levels were set to all possible values of $L \in \{3, 4, \ldots, MaxL\}$. Because $MaxL = j$ for $N = 2^j$, this produced the triangular-shaped parameter space shown in Figure 4.6. Here, vertical lines correspond with fixed N, horizontal lines correspond with fixed L, and the grid points on the hypotenuse correspond with $L = MaxL$. There are 36 possible (N, L) ordered pairs in this parameter space, and $n = 100$ observations were collected for each. For this section, the data for the 36 points correspond with the vertices in the figure, and the heatmap color is interpolated between those grid points. The metrics $Time$, v_0, and v_1 are displayed with a log scale because extreme values in the upper-right corner had oversaturated the heatmaps, making it impossible to interpret the variation in the rest of the space.

The purpose of Section 4.1 was to examine the algorithm performance as N increases,

ranging between powers of 2. The purpose of this Section 4.2 is to examine the algorithm behavior in the space between $L = 3$ and $L = MaxL$. Note the data for Section 4.1 was collected along the $L = MaxL$ step-function just above the hypotenuse (up to $N = 2^9$) but with data collected for values of N between powers of 2.

The (nearly) parallel bands of color in Figure 4.6(a), (b), (d), and (e) make it seem like those heatmaps represent linear surfaces, but in fact only Figure 4.6(b) is linear. An indicator of sparsity, v_0 does not depend on the dilation level L and is in fact a simple multiple of N as indicated in Figure 4.4. The pseudo-linear appearances of the other three figures make it seem that $Time$ and v_1 depend on N in a polynomial fashion (since those are log-log plots) and on L in an exponential manner, and make it seem $Loops$ has a logarithmic relationship with N and a linear one with L. However, these three surfaces are actually not as linear as these color gradients make them appear, i.e., they still contain curvature, so these relationships are not so straightforward. However, it can still be said that $Loops$, $Time$, and v_1 grow faster with L than they do with N. Interestingly, since v_0 does not depend on L but v_1 grows rapidly with L, this means that as the dilation level increases and $\boldsymbol{D}$ provides more vectors to choose from, the vector $\boldsymbol{v}$ is not picking up additional entries but is choosing more extreme values to scale the vectors in $\boldsymbol{D}$.

Of special note, there is less yellow in the corner of the $Time$ plot in Figure 4.6(d) (which is plotted on a log scale) than the amount of yellow in the corner of $Loops$ plot in Figure 4.6(a). This means that if you were to plot $log(Time)$ against $Loops$ then it would curve upward, indicating that execution $Time$ grows **much** faster than one might expect based on the number of loops.

Finally, the behavior of $PSNR$ and $SSIM$ is different from the other metrics. Both of these metrics are largest at $N = 2^5 = 32$ along the hypotenuse where $L = MaxL = 5$, which corresponds with the peak at the very left end in Figure 4.5. This indicates that to emphasize accuracy, one should break the image of interest into 4×8 or 8×8 patches and use $L = MaxL$. Note that since the horizontal axis for Figure 4.6 is $log_2(N)$, most of the

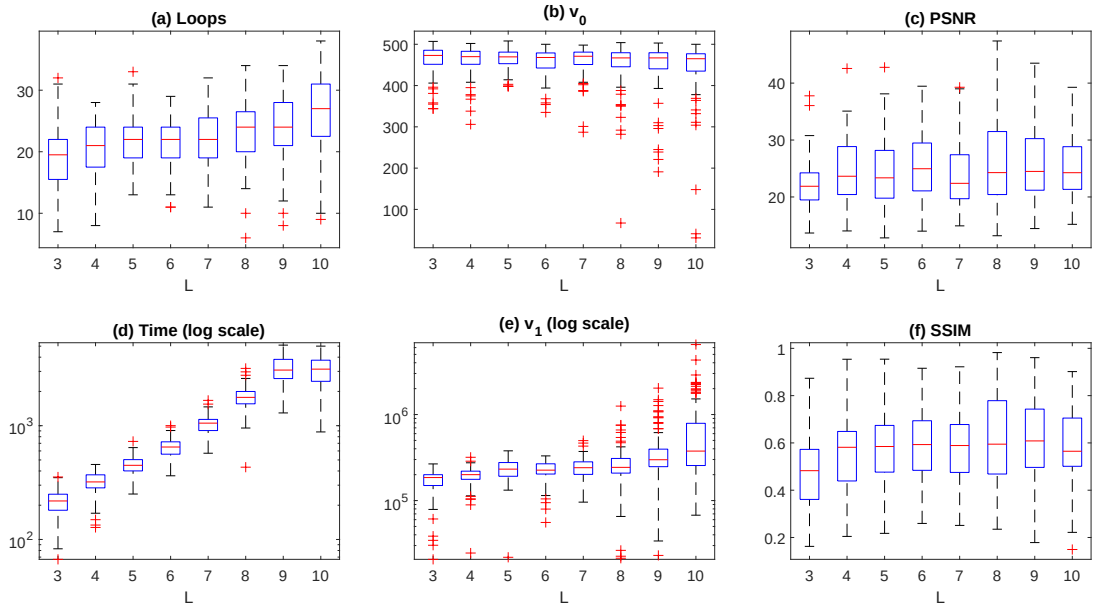


Figure 4.7: Boxplots of metrics for $N = 2^{10}$, with $L \in \{3, 4, \ldots, 10\}$, corresponding with the rightmost edge of the triangles in Figure 4.6. Whiskers extend to the largest and smallest values that are not outliers. Outliers are indicated by '+'.

heatmap area corresponds to small values of N.

Larger values of N may be of some interest, and Figure 4.7 shows boxplots of the observations where $N = 2^{10}$, corresponding with the rightmost vertical edge of the triangular parameter space shown in Figure 4.6. The boxes span the 1^{st} to the 3^{rd} quartiles and the whiskers extend to the largest and smallest non-outlier values, where outliers are any observations more than 1.5 times the inner-quartile range away from the box. Outliers are shown as '+'. Note that $Time$ and v_1 are still plotted logarithmically, but that v_0 is not, because it made that chart harder to read. As noted before, for the most part v_0 does not depend on L, however, higher L permits lower v_0 outliers. Because $Time$ and v_1 appear to have a linear relationship with L on a logarithm plot (except for $L = 10$), they grow exponentially with L. Boxplots for $PSNR$ and $SSIM$ may indicate a slight improvement going from $L = 3$ to $L = 4$, but no discernible improvement for larger values of L. As observed for the heatmaps in Figure 4.6, $Loops$ increases with L.

4.3 Effect of WMP Factor w on Performance Metrics

The WMP algorithm was originally chosen for Step 1, because of the need for faster run-time to permit more observations on larger image patch sizes. Taken alone, WMP executes faster than OMP and our early experimentation showed it gave similar results for the objective function in Step 1. The *Time* metric in Figure 4.8 showing a faster execution time at $w = 1$, however, indicates that OMP in Step 1 coupled with *fminsearch* for Step 2 is faster than using WMP coupled with *fminsearch*. That is, OMP takes slightly longer in Step 1, but the resulting $\boldsymbol{v}$ permits Step 2 to run faster, enough to make up for the slightly longer Step 1.

The WMP function was written to exit when it encountered the first local minimum, because experimentation showed that it was nearly as good as the global minimum. This took less time than guaranteeing the global minimum had been found, and it emphasized the sparsity of $\boldsymbol{v}$.

4.3.1 Single Factor Analysis with w

As discussed above, WMP uses a factor $w \in (0, 1]$, permitting it to quickly choose a vector from a frame that, when compared to the vector OMP would choose, is a less optimal fit to the target by this factor. In this numerical exploration, testing values of the parameter $w \in \{0.05, 0.1, 0.15, \ldots, 0.95, 1.0\}$ was performed. To avoid confounding the results, data collection was organized so that all possible values of w would be used once with the same sensing matrix $\boldsymbol{A}$ and 16×16 image patch. This was repeated with new random sensing matrices and image patches for a total of $n = 200$ observations at each value of w. Figure 4.8 shows these observations as box plots for each metric and value of w. The boxes span the 1^{st} to the 3^{rd} quartiles and the whiskers extend to the largest and smallest non-outlier values, where outliers are any observations more than 1.5 times the inner-quartile range away from the box. Outliers are shown as '+'.

Examination shows that there is not one value of w that optimizes all of the metrics.

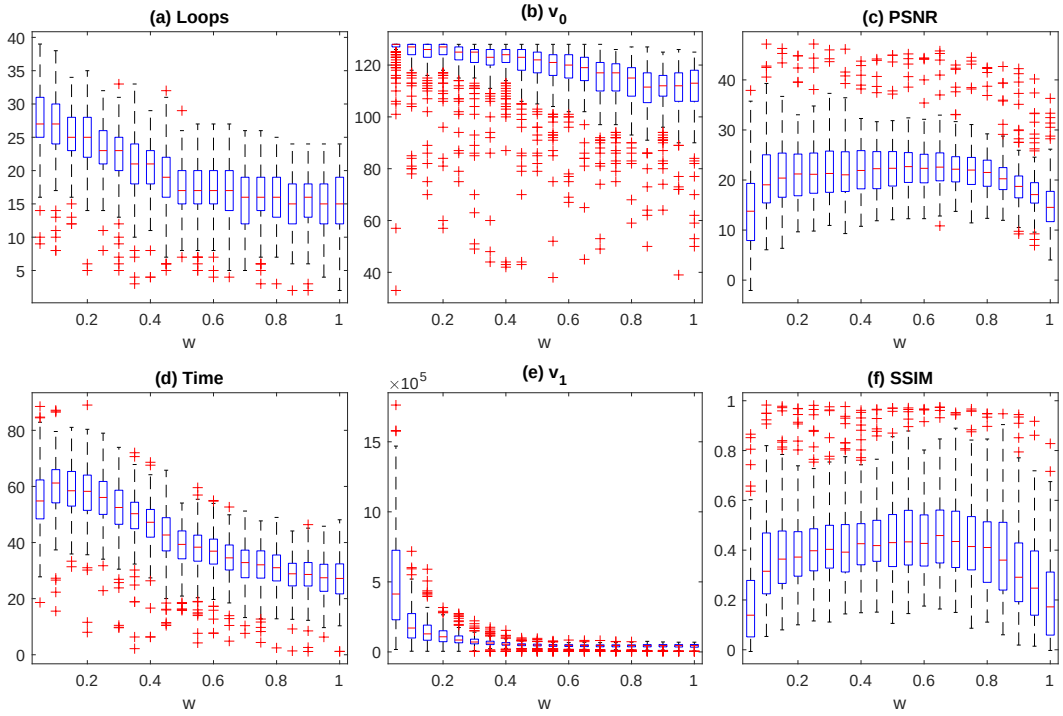


Figure 4.8: Boxplots of metrics for each WMP value of $w \in \{0.05, 0.1, \ldots, 0.95, 1.0\}$. Whiskers extend to the largest and smallest values that are not outliers. Outliers are indicated by '+'.

Loops, *Time*, v_0, and v_1 tend to improve as w approaches 1 (i.e., as WMP approaches OMP), because it is generally desirable to have these as small as possible. In contrast, where larger values of *PSNR* and *SSIM* indicate better accuracy, $w = 0.65$ gives the highest means for these metrics, the highest median for *SSIM* and nearly tied for the highest median of *PSNR*.

4.3.2 Two Factor Analysis with *w* and Sampling Proportion *p*

A parameter not yet explored is the number of measurements taken of the original image, specified in this numerical experiment by the sampling rate p. The value of p controls the number of columns in the sensing matrix $\boldsymbol{A}$, whose the dimensions are $N \times (pN)$. This experiment combines p with the relaxation parameter w from the WMP algorithm in a design of experiments approach to see the effect of both parameters on the six performance

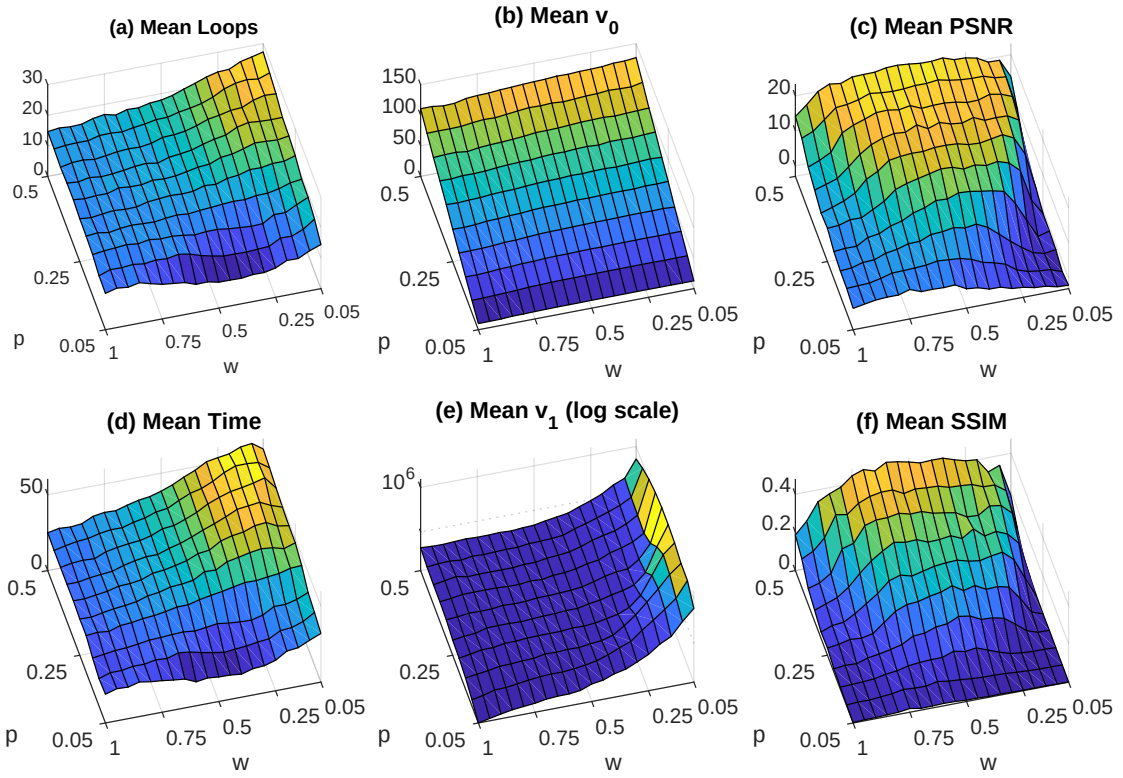


Figure 4.9: Surface plots for means of metrics plotted by w and sampling rate p. The plot for v_1 is on a log scale.

metrics.

Data collection for this numerical experiment used all possible combinations of the factor $w \in \{0.05, 0.1, 0.15, \ldots, 1.0\}$ and sampling rate $p \in \{0.05, 0.1, 0.15, \ldots, 0.50\}$. The same 16×16 image patch and sensing matrix were applied with all 200 combinations of w and p. A total of $n = 120$ observations were collected using this procedure. Figure 4.9 shows the results of the six metrics as surface plots with independent variables w and p. Note that v_1 is plotted on a log scale to better understand the effect of w and p on that metric.

Ideally, one would like combinations of w and p that produce the largest possible values for $PSNR$ and $SSIM$, but the smallest possible values for $Loops$, $Time$, v_0 and v_1. However, no single point in the (w, p) plane will provide the optimal point for all of the performance metrics simultaneously. To understand what will work best, one must consider

what the needs of the user will be.

The surface plots for the accuracy measures, $PSNR$ and $SSIM$, exhibit similar behavior patterns in Figure 4.9(c) and Figure 4.9(f). The w range showing reasonably good results is widest when $p = 0.5$ and gets narrower as p decreases. However, as p decreases, the plateau for $PSNR$ is sustained longer than for $SSIM$. For these data, the mean $PSNR$ has a local maximum of 23.2 at the point $w = 0.55$, $p = 0.5$, and the mean $SSIM$ has local maximum of 0.4766 at the point $w = 0.6$, $p = 0.5$.

Another important requirement might be that of sparsity, as measured in this research by v_0. Examination of Figure 4.9(b) suggests that as p increases and more measurements of the original image are taken, the vector $\boldsymbol{v}$ contains more nonzero elements, i.e. becomes less sparse. Also note that when $w \longrightarrow 1$, there is a slight decrease in the number of nonzero entries in $\boldsymbol{v}$, but, according to the $PSNR$ and $SSIM$ plots, there is a considerable decrease in accuracy. The surface plot for v_1, shown in Figure 4.9(e) presents a more straightforward analysis. It remains relatively flat for values of $w \geq 0.5$, then increases rapidly as $w \longrightarrow 0$.

The surface plots for $Loops$ and $Time$, shown in Figure 4.9(a) and Figure 4.9(d) have similar structure, peaking near the point where $w = 0.05$ and $p = 0.5$, and falling away as w increases and p decreases. On both of these plots there are valleys on the edge where $p = 0.05$. Those might appear to be optimal positions for measures of execution time, but it's more likely they represent scenarios in which the representation algorithm reached a local minimum from which it could not move away.

In conclusion, it is clear that small p should be avoided, because of the very small accuracy observed, regardless of the value of w. For $p \geq 0.2$, w does not depend on p, and choosing a moderate value of w, say $w \in [0.55, 0.65]$ would maximize the accuracy, while not affecting the execution duration and sparsity too adversely.

4.4 Algorithm Behavior for 3×3 vs 5×5 Filterbank

Previous work on this project used the piecewise linear B-spline framelet $FB = 3 \times 3$ shown in Equation (2.2) to initialize the synthesis operator $\boldsymbol{D}$. In this section, the results of the larger, cubic B-spline framelet $FB = 5 \times 5$ shown in Equation (2.3) was compared to the results of the smaller filterbank. Before comparing the mean performance metrics results for the two filterbank sizes, a sensitivity analysis was performed to determine if the regularization parameters used in the research utilizing $FB = 3 \times 3$ should be adjusted to values more appropriate for the larger $FB = 5 \times 5$. Table 4.1 displays the regularization parameters used for each filterbank in the data collection for this section. Section 3.2.1 and Section 3.2.3 explained how these regularization parameters were determined.

Table 4.1: Regularization parameters for comparison of 3×3 and 5×5 filterbanks.

Filterbank	μ_1	μ_2
3×3	1.4^t	10^{13}
5×5	1.6^t	10^{14}

To compare the behavior of the metrics under the two different filterbank sizes, $n = 100$ observations were generated. For each of these, a random sensing matrix $\boldsymbol{A}$ and 16×16 image patch were created. The algorithm first used the 3×3 filterbank to find the representation and reconstruction of the image patch, then repeated the process by using the 5×5 filterbank. The mean metrics were calculated by filterbank across the total set of observations. These data are reported in Table 4.2.

It is noted that, although the larger filterbank takes longer to process, fewer loops are required. In addition, when $FB = 5 \times 5$ the reconstructions were more accurate and resulted in much lower sparsity than that obtained by $FB = 3 \times 3$. My conclusion is that if time is not a serious constraint, better results can be obtained by using the larger 5×5 filterbank.

Table 4.2: Means of metrics for 3×3 and 5×5 filterbanks.

Filterbank	$Time(s)$	$Loops$	$PSNR$	$SSIM$	v_0	v_1
3×3	17.5	16.8	23.0	0.47	117.9	45035
5×5	25.8	7.5	27.4	0.67	86.9	52797

4.5 Algorithm Behavior for Integer and Floating Point Image Formats

The representation algorithm studied here used four grayscale images shown in Figure 2.1 as initial image signals to be reconstructed. These images were stored with integer pixel values in the interval $[0, 255]$, but sometimes grayscale images are stored with floating point values in the interval $[0, 1]$. It was possible the image format might affect the execution of the representation algorithm and the metrics used to characterize it. This section compares the means of the metrics under each of the image formats described above. The maximum dilation level $L = 3$ and the 3×3 filterbank shown in Equation (2.2) were utilized for both formats.

Table 4.3: Regularization parameters for comparison of image formats.

Image Format	μ_1	μ_2
Integer	1.4^t	10^{13}
Floating Point	14^t	10^5

Regularization parameters were selected for the floating point format using the method described in Section 3.2.2. The increasing μ_1 and constant μ_2 parameters chosen for the floating point format are given in Table 4.3, along with the corresponding parameters used for the integer format comparison. Data were collected by generating $n = 100$ observations. For each of these, a sensing matrix $\boldsymbol{A}$ and 16×16 image patch were randomly chosen. The

Table 4.4: Means of metrics for integer and floating point formats.

Format	$Time(s)$	$Loops$	$PSNR$	$SSIM$	v_0	v_1
Integer: $[0, 255]$	15.2	16.8	22.7	0.48	117.5	176.4[a]
Floating Point: $[0, 1]$	2.2	7.2	22.5	0.47	125.3	170.6

[a] Integer format v_1 was divided by 255 to make the comparison between formats meaningful

representation algorithm was applied to each patch stored in the integer format $[0, 255]$, then to the same patch stored in the floating point format $[0, 1]$. The means of the metrics are shown in Table 4.4. Note that for the integer format $[0, 255]$ the metric v_1 was divided by 255 to make the appropriate comparison with the same metric for the floating point format $[0, 1]$.

It was noted that compared to the integer format, the floating point format required fewer loops through the representation algorithm, and considerably less time to execute. The sparsity, as measured by v_0, however, was better (i.e., smaller) for the integer format than for the floating point format. The metrics $PSNR$, $SSIM$, and v_1 did not seem to differ between the two formats.

Even though a single byte **uint8** variable can represent integer values from 0 to 255, we used double data type variables to represent pixel values for both integer and floating point image formats. Thus the differences seen in Table 4.4 by switching to the floating point format do not come from being able to represent a higher number of values close to valid pixel values. These differences might be due to the orders of magnitude of the sets of values that represent the shades of gray.

4.6 Looking for Interactions Among Filterbanks, Image Formats and Maximum Dilation Levels

In Section 4.4, the metrics were compared for 3×3 and 5×5 filterbanks and in Section 4.5 the metrics were compared for the two image formats commonly used in image representation. For most of the research, the maximum dilation level to was set to $L = 3$. In [42] it was found that increasing the maximum dilation level to $L = 4$ might increase the accuracy, but increasing to maximum dilation levels larger than $L = 4$ did not seem to provide additional improvement. In this section, these algorithm parameters are combined in a three-way analysis.

The factors analyzed and their levels are shown in Table 4.5. They were Image Format (integer and floating point), Filterbank ($FB = 3\times3$ and $FB = 5\times5$), and Maximum Dilation Level ($L = 3$ and $L = 4$). Each factor has two levels, and all combinations of those two levels were used in this analysis. A total of $n = 100$ observations were collected for each of the eight possible combinations of the three factors.

For data collection, different regularization parameters were used for the two image formats. For the four combinations involving integer formats, the default regularization parameters developed in Section 3.2.1 ($\mu_1 = 1.4^t$, $\mu_2 = 10^{13}$) were used. For the four combinations involving floating point format combinations, the regularization parameters developed in Section 3.2.2 ($\mu_1 = 14^t$, $\mu_2 = 10^5$), were used.

Table 4.5: Factors and levels for three-way analysis of image format, filterbank size (FB), and maximum dilation level (L).

Analysis Factors and Levels		
Image Format	Filterbank	Max Dilation Level
Integer: $[0, 255]$	$FB = 3\times3$	$L = 3$
Floating Point: $[0, 1]$	$FB = 5\times5$	$L = 4$

Table 4.6: Arithmetic Mean (Standard Error) of each metric for two-way analysis of integer format by filterbank size and maximum dilation level L, based on $n = 100$ observations per combination.

Factor Combinations - Integer Format							
FB	L	$Time(s)$	$Loops$	$PSNR$	$SSIM$	v_0	v_1
3×3	3	16.6(0.4)	16.6(0.5)	22.2(0.6)	0.48(0.02)	116.6(1.3)	177.8(5.5)
3×3	4	27.8(0.6)	18.4(0.4)	25.1(0.6)	0.57(0.02)	117.9(0.9)	213.2(6.3)
5×5	3	13.0(0.8)	3.2(0.1)	25.2(0.6)	0.62(0.02)	53.2(1.4)	194.5(6.6)
5×5	4	68.1(7.6)	12.5(1.4)	26.8(0.7)	0.64(0.02)	80.4(3.3)	275.2(11.2)

Preliminary analyses include the statistics shown in Table 4.6 and Table 4.7, i.e., the arithmetic means and standard errors of the performance metrics for the integer and floating point formats, respectively. It can be noted that in general, the floating point configurations were faster and took fewer loops to reach convergence, except for the combination $FB = 5\times5$ and $L = 3$. The two image formats show similar accuracy, though $PSNR$ and $SSIM$ tend to be slightly higher for the integer format, but it will be shown that the differences are not statistically significant. The integer format configurations show better sparsity (lower v_0), especially those for $FB = 5 \times 5$. To get a better idea of how these factors affected the metrics, statistical results were generated using SAS®, and the relevant output is shown in Appendix **??** and summarized below.

Table 4.7: Arithmetic Mean (Standard Error) of each metric for two-way analysis of floating point format by filterbank size and maximum dilation level L, based on $n = 100$ observations per combination.

Factor Combinations - Floating Point Format							
FB	L	$Time(s)$	$Loops$	$PSNR$	$SSIM$	v_0	v_1
3×3	3	2.5(0.2)	7.4(0.6)	22.7(0.5)	0.47(0.01)	125.0(0.1)	180.7(5.1)
3×3	4	3.1(0.1)	7.3(0.2)	24.1(0.6)	0.56(0.02)	124.5(0.2)	205.3(6.7)
5×5	3	20.8(1.6)	12.1(0.9)	26.2(0.7)	0.62(0.02)	124.5(0.2)	222.2(7.0)
5×5	4	19.9(1.1)	7.7(0.4)	25.1(0.7)	0.60(0.02)	116.9(2.8)	265.3(11.9)

A three-way ANOVA was initially run for each of the metrics, $Time$, $Loops$, $PSNR$, $SSIM$, and v_0. The three-way interaction $Format * FB * L$ was examined first and decisions about the next steps proceeded from there. Results were considered statistically significant if $p < 0.05$.

I expected to see means of the metrics $Time$, $Loops$, $PSNR$, and $SSIM$ become larger and the mean for metric v_0 become smaller as the maximum dilation level increased from $L = 3$ to $L = 4$ or as the size of the filterbank increased from $FB = 3 \times 3$ to $FB = 5 \times 5$. The statistical results that explain if this was true are summarized in the tables shown below. The means in the tables are sorted in ascending order. The least squares means, i.e., means that are adjusted for other factors in the model, and relevant p-values are reported for any significant effects that were observed. In some cases, a transformation of the response variable was required to meet model assumptions for the ANOVA method, and the transformations used are noted. In the event that a transform was required, the least squares means reported were back-transformed from the results shown in the SAS output. Note that when there are more than two possible factor combinations, Tukey Groups are used to show which group means are significantly different. Two means having the same Tukey Group letter are not significantly different.

4.6.1 *Time* and *Loops*

Analysis of Variance models require homogeneity of variances and normality in the sub-populations of the combinations of the three factors. These were not satisfied by either $Time$ or $Loops$, but a natural log transform on each allowed these model assumptions to be satisfied for both. Because normality was still not satisfied, it was determined that analyzing both $Time$ and $Loops$ separately by image format would alleviate the problem. These metrics displayed significant two-way interactions between maximum dilation level L and filterbank size FB for both image formats. Post-hoc tests were performed for the significant interactions and are shown in Table 4.8 for $Time$ and in Table 4.9 for $Loops$.

Under the integer image format, the means of $Time$ for all four factor combinations are significantly different from each other. For the integer format, $Time$ decreased as L decreased for both FB sizes, $Time$ decreased for $L = 4$ when FB size decreased, and Time decreased for $L = 3$ when FB size increased. When using the floating point format, $Time$ was significantly different for all combinations except the two involving $FB = 5 \times 5$. $Time$ decreased for both levels of L as FB size decreased, and $Time$ decreased for $FB = 3 \times 3$ when L decreased. SAS output for these results is in Appendix B.1.

Table 4.8: Least squares means of $Time$ for significant two-way interactions between L and FB. Response variable was $ln(Time)$, but means reported below have been back-transformed.

$Time$							
Integer Format: $p < 0.0001$				Floating Point Format: $p = 0.019$			
L	FB	$Time(s)$	Tukey	L	FB	$Time(s)$	Tukey
$L = 3$	5×5	10.9	A	$L = 3$	3×3	2.3	A
$L = 3$	3×3	16.0	B	$L = 4$	3×3	3.0	B
$L = 4$	3×3	26.8	C	$L = 3$	5×5	17.3	C
$L = 4$	5×5	47.7	D	$L = 4$	5×5	17.4	C

The metric $Loops$ behaved very differently than $Time$, exhibiting unexpected reversals as L and FB decreased. For the integer format, the means of $Loops$ were significantly different for all combinations except the two involving $FB = 3 \times 3$. $Loops$ decreased for both levels of L as FB size increased, and decreased for $FB = 5 \times 5$ when L decreased. For the floating point format the combination $L = 3$ and $FB = 5 \times 5$ was significantly different from the remaining combinations. $Loops$ decreased for $FB = 5 \times 5$ when L increased, and for $L = 3$ when FB size decreased. SAS output for these results is in Appendix B.2.

Table 4.9: Least squares means of $Loops$ for significant two-way interaction between L and FB. Response variable was $ln(Loops)$, but means reported below have been back-transformed.

$Loops$							
Integer Format: $p < 0.0001$				Floating Point Format: $p < 0.0001$			
L	FB	$Loops$	Tukey	L	FB	$Loops$	Tukey
$L = 3$	5×5	2.9	A	$L = 3$	3×3	6.9	A
$L = 4$	5×5	9.3	B	$L = 4$	3×3	7.0	A
$L = 3$	3×3	15.8	C	$L = 4$	5×5	7.1	A
$L = 4$	3×3	17.8	C	$L = 3$	5×5	10.2	B

4.6.2 *PSNR* and *SSIM*

The three-way ANOVA models for $PSNR$ and $SSIM$ easily satisfied the assumptions of homogeneity of variances and normality. In both cases, the three-way interactions were not significant with $p = 0.5291$ for $PSNR$ and $p = 0.3246$ for $SSIM$. For each metric, reduced models were performed that included all possible 2-way interactions.

For $PSNR$, two of the interactions were significant, with $p = 0.0331$ for $L * FB$ and $p = 0.0280$ for $Format * L$. The interaction $Format * FB$ was not significant, with $p = 0.8778$. Post-hoc tests were performed and are shown in Table 4.10. For the interaction $L * FB$, only the mean for the combination of $L = 3$ and $FB = 3 \times 3$ was significantly different (i.e., smaller) than the means for the other three combinations. $PSNR$ increased for $L = 3$ when FB size increased, and for $FB = 3 \times 3$ when L increased. For the interaction $Format * L$ only the means for the two combinations involving integer format were significantly different, and $PSNR$ increased for this image format when L increased. SAS output for these results is in Appendix B.3.

For $SSIM$, only the interaction $L * FB$ was significant, with $p = 0.0002$. The interactions $Format * FB$ and $Format * L$ were not significant with $p = 0.7628$ and $p = 0.3170$,

Table 4.10: Least squares means of $PSNR$ for significant two-way interactions $L * FB$ and $Format * L$.

$PSNR$							
$L * FB$: $p = 0.0331$				$Format * L$: $p = 0.0280$			
L	FB	$PSNR$	Tukey	$Format$	L	$PSNR$	Tukey
$L = 3$	3×3	22.4	A	Integer	$L = 3$	23.7	A
$L = 4$	3×3	24.6	B	Floating Point	$L = 3$	24.4	AB
$L = 3$	5×5	25.7	B	Floating Point	$L = 4$	24.6	AB
$L = 4$	5×5	26.0	B	Integer	$L = 4$	25.9	B

respectively. Post-hoc tests are reported in Table 4.11. All the means were different, except for the means involving $FB = 5 \times 5$. The metric $SSIM$ increased for both levels of L as FB size increased, and for $FB = 3 \times 3$ as L increased. SAS output for these results is in Appendix B.4.

Table 4.11: Least squares means of $SSIM$ for significant two-way interaction $L * FB$.

$SSIM$			
$L * FB$: $p = 0.0002$			
L	FB	$SSIM$	Tukey
$L = 3$	3×3	0.475	A
$L = 4$	3×3	0.568	B
$L = 3$	5×5	0.621	C
$L = 4$	5×5	0.621	C

4.6.3 Sparsity Measure v_0

The ANOVA model for metric v_0 failed to satisfy assumptions of homogeneity of variances and normality. A Box-Cox procedure suggested the transformation v_0^4, but it also failed to

satisfy assumptions. As had been performed for $Time$ and $Loops$, the data were separated by image format, and two-way ANOVA models were run on each.

For the integer format, model assumptions were satisfied and a transpose was not required. The interaction $L * FB$ was significant with $p = 0.0020$. The post-hoc test results are shown on the left side of Table 4.12. All the means were significantly different, except for the two involving $FB = 3 \times 3$. The metric v_0 decreased for both levels of L as FB size decreased, and decreased for $FB = 5 \times 5$ when L decreased.

The metric v_0 for the floating point data required a transformation, and the Box-Cox procedure suggested that v_0^{12} would satisfy the model assumptions. For this floating point analysis the interaction $L * FB$ was not significant, $p = 0.7560$, and a main effects model was performed. Both main effects were significant, with $p = 0.0045$ for L and $p = 0.0067$ for FB. The means are shown on the right side of Table 4.12. For the floating point data, the mean v_0 decreased as L increased and as FB size increased. SAS output for these results is in Appendix B.5.

Table 4.12: Least squares means of v_0 by image $Format$ for the significant interaction $L * FB$ under the image format, and for main effect L and FB under the floating point format.

Sparsity: v_0							
Integer				Floating Point			
$L * FB$: $p = 0.0020$				Significant Main Effects			
L	FB	v_0	Tukey	Effect	p-value	Levels	v_0
$L = 3$	5×5	53.2	A	L	$p = 0.0045$	$L = 4$	124.27
$L = 4$	5×5	80.4	B			$L = 3$	124.87
$L = 3$	3×3	116.6	C	FB	$p = 0.0067$	5×5	124.29
$L = 4$	3×3	117.9	C			3×3	124.86

It should be pointed out that the least squares (LS) means for each of these main effects are significantly different, but appear to be very close in value. By comparing to the

Table 4.13: Least squares means (back-transformed) compared to arithmetic means from Table 4.7.

Sparsity: v_0		
Main Effect Level	LS Means	Arithmetic Means
$L = 4$	124.27	120.70
$L = 3$	124.87	124.75
5×5	124.29	120.70
3×3	124.86	124.75

arithemetic means that appear in Table 4.7, it can be seen that the means are indeed different, as shown in Table 4.13. Note that averaging the pairs of means from Table 4.7 for each main effect is feasible, because the sample sizes are $n = 100$ for each combination. One must also be reminded that statistically significant differences are not always meaningful differences, and common sense must be applied when claiming differences are meaningful or important.

4.7 Chapter Summary

In this chapter, the results of several numerical experiments were reported. In Section 4.1, the effect of image patch size, N, and dilation level was explored by setting the maximum dilation level to $L = MaxL$ and then to $L = 3$, for comparison. The plots for $Loops$, $Time$, and v_1 displayed a non-linear swoosh pattern that indicated image patch sizes close to, but smaller than, powers of 2 would yield desirable values for these metrics. Closer examination of a set of N, all having $MaxL = 8$, showed that as the dilation level increased from $L = 3$ to $L = 8$, the swoosh gradually appeared. The other three metrics did not display this pattern, and were analyzed separately. The sparsity metric, v_0, was plotted as a nearly linear function of N. A linear fit through the points indicated that for $L = MaxL$, sparsity can be estimated as 87% of the number of measurements made, and for $L = 3$ as 91% of the number of

measurements made. The plots for both $PSNR$ and $SSIM$ displayed maximum values near $N = 2^5 = 32$, and gradual decreases for N larger than that.

The exploration of the effect of image patch size was extended in Section 4.2 to include dilation levels $L \in \{3, 4, \ldots, MaxL\}$ for image patch sizes $N = 2^j$ for $j \in \{3, 4. \ldots, 10\}$. It was observed that $Loops$, $Time$, and v_1 grow faster with L than they do with N. In contrast, v_0 does not depend on the dilation level L and depends only on N. Metrics $PSNR$ and $SSIM$ are largest near $N = 2^5 = 32$ along the hypotenuse where $L = MaxL$, also suggesting that one should break the image of interest into 4×8 or possibly 8×8 patches and use $L = MaxL$.

In Section 4.3 it was found that regardless of the value of the sampling proportion p, a moderate value of WMP relaxation factor, such as $w = 0.6$, would yield better accuracy, without pushing execution duration into unacceptable ranges. A comparison of metric means for the two filterbank sizes was examined in Section 4.4. Using the smaller 3×3 filterbank was faster but using the larger 5×5 filterbank required fewer loops, was more accurate, and resulted in better sparsity. In Section 4.5 a comparison was made of metrics means for the two image formats studied here. Using the floating point format resulted in faster execution time with fewer loops, but using the integer format allowed for better sparsity.

A three-way analysis of filterbank size, image format, and dilation levels $L \in \{3, 4\}$ was reported in Section 4.6. The results were very similar to those reported for individual analyses, but with a one exception involving the $Loops$ metric: for the integer format, the 5×5 filterbank, required fewer loops than the 3×3 filterbank, while for the floating point format, the opposite was true, but only for $L = 3$. For the metric, $Time$, a significant decrease occurred for the integer format when L decreased, but for the floating point format $Time$ decreased when the filterbank size decreased. Sparsity was best when using the 5×5 filterbank with both image formats. Accuracy improved when using the 5×5 filterbank for either image format, but only significantly for $L = 3$.

Chapter 5

Parseval Relaxation

The authors of the algorithm restricted themselves to a Parseval frame due to the existence of a theorem [24] that ensured the reconstruction error would be bounded for tight frames and since tight frames have received increased attention in image processing applications [10]. However, Candès et al. point out that their theorem does not require a tight frame, and that constraint was employed to simplify the analysis. This, coupled with the fact that this reconstruction algorithm does not seem to rely on the properties of tight frames in any way, led us to explore the algorithm performance as the Parseval condition was relaxed.

The authors implemented this condition by including a constraint in the second step of the representation algorithm that requires the frame $\boldsymbol{D}$ to be Parseval. Using the definition of a frame in Equation (1.4), if there are frame bounds that satisfy the condition that $A = B$, the frame is tight. A special case of tight frame is a Parseval frame, in which the optimal frame bounds are $A = B = 1$ ([19], pg 23). If $\boldsymbol{D}$ is Parseval, then the frame operator $\boldsymbol{D}\boldsymbol{D}^* = \boldsymbol{I}$ and the condition number of $\boldsymbol{D}\boldsymbol{D}^*$ is $\kappa = 1$.

One way to relax the Parseval condition is to let the condition number be larger than 1 (note, this would also permit non-tight frames since tight frames also have condition number 1). The frame operator $\boldsymbol{D}\boldsymbol{D}^*$ is clearly Hermitian (and normal), so its condition number is $\kappa = \frac{\lambda_1}{\lambda_m}$, where λ_1 and λ_m are, respectively, the largest and smallest eigenvalues of

$\boldsymbol{DD}^*$. The proof of this formula can be found in any linear algebra textbook. The Parseval condition will be relaxed whenever $\frac{\lambda_1}{\lambda_m} - 1 > 0$.

5.1 Parseval Relaxation Functions

For each of the three relaxation methods proposed here, a tuning parameter $\alpha > 0$ will be introduced to represent the amount of relaxation applied to the Parseval constraint. For the first method, in Step 2 the expression $\|\boldsymbol{DD}^* - \boldsymbol{I}\|_F^2$ is replaced by $\kappa 1$ shown in Equation (5.1):

$$\kappa 1 := \max\left(\alpha, \frac{\lambda_1}{\lambda_m} - 1\right) \tag{5.1}$$

This method provides a constant cost when $\kappa \leq 1 + \alpha$ and a gradually increasing cost when $\kappa > 1 + \alpha$. The second method proposed also provides a constant penalty for a small condition number, but imposes an infinite penalty once $\kappa > 1 + \alpha$. In this second method, the expression $\|\boldsymbol{DD}^* - \boldsymbol{I}\|_F^2$ is replaced by the step function $\kappa 2$ shown in Equation (5.2):

$$\kappa 2 := \begin{cases} 0 & \text{if } \kappa \leq 1 + \alpha \\ \infty & \text{if } \kappa > 1 + \alpha \end{cases} \tag{5.2}$$

These first two methods relaxed the equality $\kappa = 1$. The third proposed method relaxes the equality $\boldsymbol{DD}^* = \boldsymbol{I}$, specified in the constraint of Step 2 shown in Equation (2.1). This option is considered, because calculating the Frobenius norm might be faster than finding the eigenvalues. The Parseval constraint is modified as shown in Equation (5.3):

$$P3 := \max\left(\alpha, \|\boldsymbol{DD}^* - \boldsymbol{I}\|_F\right) \tag{5.3}$$

To test the effectiveness of these relaxation methods, they were compared to the original Step 2 Parseval constraint. This baseline is referred to as the Original method.

5.2 Regularization Parameters for Parseval Relaxation

Before data could be collected, it was necessary to determine the appropriate values of the regularization parameters, μ_1 and μ_2 appropriate for Step 1 and Step 2 of the representation algorithm under these conditions. Past experience suggested that testing only increasing μ_1 and constant μ_2 would be required.

A bit of preliminary exploration gave direction to focus the search for both regularization parameters. The round of values tested using $\kappa 1$ were increasing values of $\mu_1 = B^t$ where $B \in \{1.1, 1.2, 1.3, \ldots, 1.9\}$, and constant $\mu_2 = 10^x$, where $x \in \{15, 16, 17, \ldots, 29\}$. The final results selected were $\mu_1 = 1.1^t$ and $\mu_2 = 10^{23}$. Complete results for the search for regularization parameters are shown in Appendix A.4.

5.3 Data Collection and Results for Parseval Relaxation

To help the user make the best choice of α, testing values that were powers of 2, ranging from 2^0 to 2^{56} were used. The upper limit of this wide range is near the maximum possible machine value, because it was important to see if there was a noticeable change in behavior at any relaxation level for any of the methods proposed. The default regularization parameters were retained for the original method, but for the three relaxation methods the regularization parameters were adjusted to allow more degradation. These values were chosen as indicated in Section 5.2. For each observation, a randomly chosen 16×16 image was sampled using the same sensing matrix $\boldsymbol{A}$ and applied using all four methods. There were $n = 100$ observations collected, each one using the original format and three relaxation methods.

Line plots, shown in Figure 5.1, were generated for all metrics to compare the three methods of relaxing the Parseval constraint to the original method. Accuracy, as measured by $PSNR$ and $SSIM$, did not differ among the four relaxation methods, but two of these methods showed important differences for $Loops$, $Time$, and v_0. Both $\kappa 1$ and $P3$ are faster,

use fewer loops,, and generate a more sparse coefficient vector v, i.e. v_0 is smaller, for values of α larger than 2^{15}. This suggests the Parseval constraint could be relaxed, using either $\kappa 1$ or $P3$, allowing greater sparsity while running faster and without loss of accuracy.

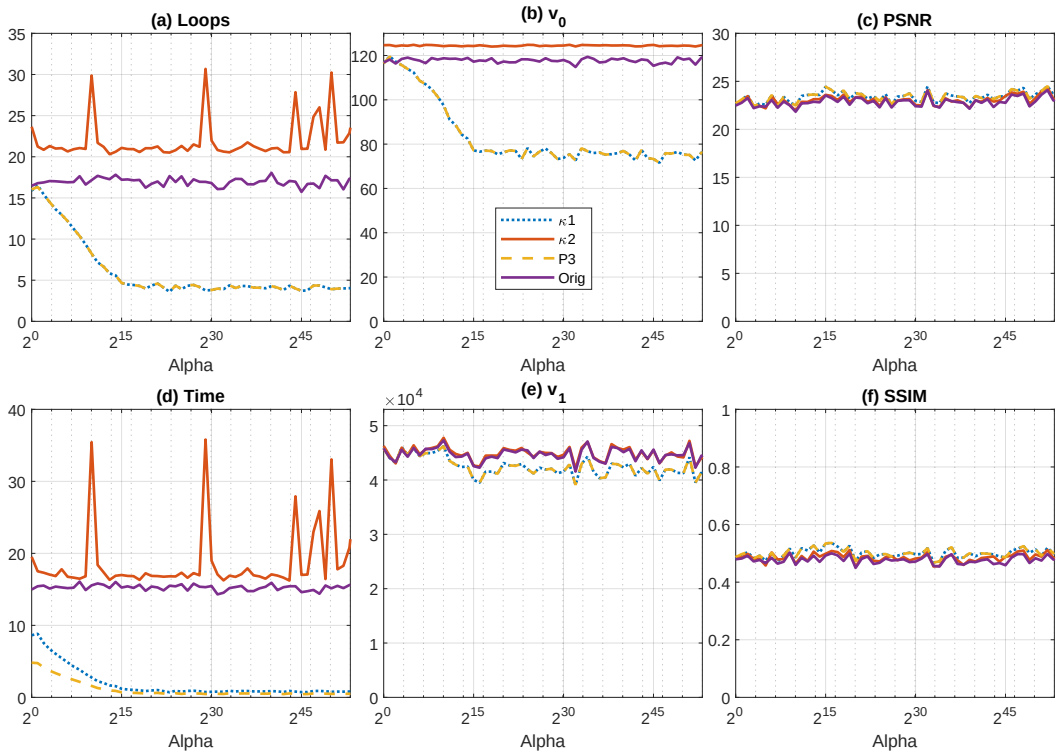


Figure 5.1: Line plots of metric means for three methods to relax the Parseval constraint and original unrelaxed constraint. The horizontal axis is on a log_2 scale.

5.4 Chapter Summary

The algorithm being researched specifies a tight frame to find the representation of the image, but uses a Parseval constraint to achieve that condition. This chapter shows that relaxing the Parseval constraint can provide results that are faster, use fewer loops, and improve sparsity, while maintaining accuracy at levels obtained when the Parseval constraint is observed. Two methods were found to be effective. One directly relaxes the Parseval condition $\|\boldsymbol{DD}^* - \boldsymbol{I}\|_F$, the other allows the condition number of the frame operator DD^*

to become larger than 1. Note that these relaxation methods would also permit non-tight frames as well. An avenue of additional research would be to omit the Parseval constraint completely from this algorithm.

Chapter 6

Visual Results and Conclusions

All the numerical experiments in the previous sections were performed by selecting random patches from a set of test images shown in the top row of Figure 6.1. These grayscale images have dimension 256×256, resulting in a total of $N = 2^{16}$ pixels. This size posed some difficulty, because the extrapolated time required to find the representation for one of these would be considerable. To reduce the time needed, each test image was divided into four quadrants and the algorithm applied independently to each one, using the default values $w = 0.6$ and $L = 3$. These reconstructions were then pieced together to make the images in the bottom row of Figure 6.1. The recombined versions were compared with the original images and their accuracies computed, and Table 6.1 displays the means of metrics $PSNR$ and $SSIM$. It is likely the Cameraman image has a lower value of $SSIM$, relative to the other images, because the sky and dark coat in the original image are noticeably free of texture.

Means of the other metrics could not be calculated for the entire 256×256 pixel reconstructed image, and can only be reported as separate quadrants. See Table 6.2 for the quadrant labels and Table 6.3 for the other performance metrics reported by quadrant.

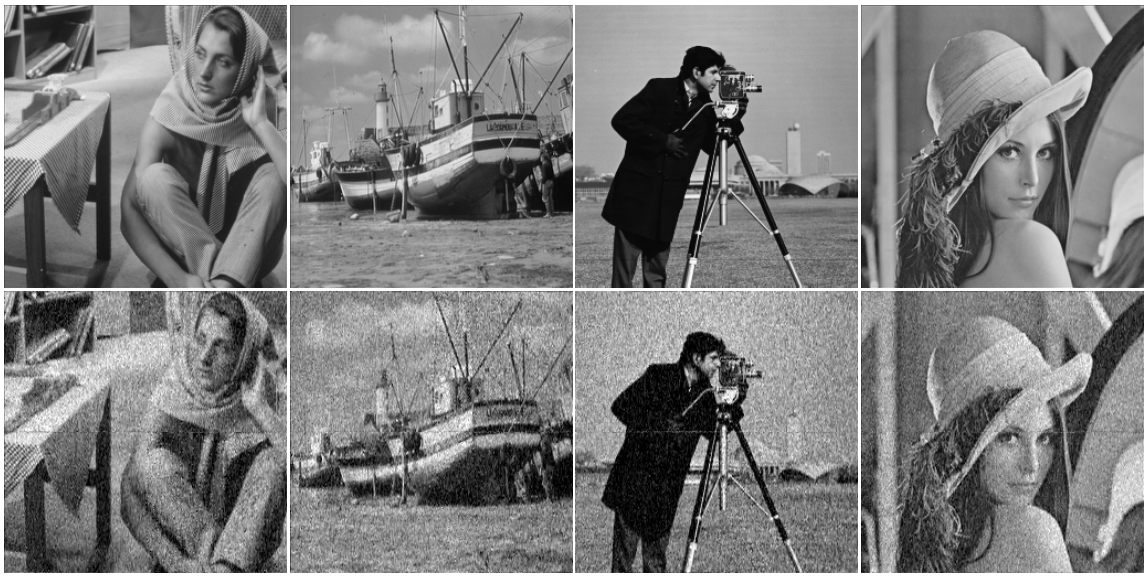


Figure 6.1: Original test images (top row) with their reconstructions (bottom row). The images were reconstructed one quadrant at a time and then placed back together

Table 6.1: Accuracy metrics comparing the reconstructions with the originals, both shown in Figure 6.1.

	Barbara	Boat	Cameraman	Lena
PSNR	21.3	21.1	21.2	23.1
SSIM	.47	.45	.36	.48

Table 6.2: Quadrant labels for Table 6.3.

A	B
C	D

Table 6.3: Metrics available for only individual quadrants of reconstructed images. Quadrant A is upper left, B is upper right, C is lower left, D is lower right.

	Quadrant	$Loops$	$Time$(days)	v_0	$v_1(\times 10^6)$
Barbara	A	26	8.54	7658	3.13
	B	38	13.69	7985	2.81
	C	30	10.23	7797	2.30
	D	31	10.10	7811	2.41
Boat	A	18	4.67	7180	3.23
	B	34	11.65	7904	3.20
	C	18	4.84	7154	2.89
	D	29	9.16	7786	2.33
Cameraman	A	27	8.20	7607	2.73
	B	34	11.87	7918	3.32
	C	17	4.35	6902	1.89
	D	28	9.06	7449	2.86
Lena	A	20	5.31	7220	2.58
	B	31	9.97	7849	2.91
	C	18	4.59	7076	2.26
	D	30	9.37	7776	2.88

6.1 Conclusions Regarding Original Algorithm

It is not clear whether Cao and Gao were able to run their algorithm on these full-sized images, and if so, how long it took, nor how they were able to get the accuracy results shown in [10]. Because image size was not discussed in the paper, I believed the authors had used full-sized 256×256 pixel images. Thus, the need to find faster solvers and memory-reducing strategies was paramount. The authors revealed in their correspondence they had used OMP for Step 1 and *fminsearch* for Step 2. It was not possible to get the algorithm to run fast enough using these functions, and investigating sparse allocation became another avenue of exploration, and ultimately used in some of the MATLAB functions appearing in Appendix C.

The algorithm as described in the paper indicated it ran for an unspecified, but fixed number of loops, whereas in this research, to try to meet the reported accuracy levels, the algorithm was run until convergence criteria were met. Further attempts to reach the authors to obtain additional information were unsuccessful. It is possible they may have stitched together smaller images, perhaps 8×8 patches, using averaging along the borders to avoid artifacts such as those seen in the reconstructed images in Figure 6.1.

6.2 Important Findings

The effect of image size on the performance of the algorithm studied here, was evident in several ways. Performance metrics relating to execution indicate the algorithm will perform faster when image size N (pixels) is a power of 2, or somewhat smaller ($85 - 100\%$ of a power of 2), compared to when N is larger than the nearest power of 2. Accuracy metrics suggest that choosing image size close to $N = 2^5$ will be optimal, additionally, maximum dilation level $L = 4$ yields higher accuracy than smaller values of L, but larger values of L do not provide a significant increase in accuracy. Sparsity was found to have a linear relation

to image size N, improving somewhat with increases in L.

It was found that using values near $w = 0.6$ in the WMP function provides a nice balance between accuracy and speed, but slightly smaller values of w may be better when the sampling rate, p, approaches zero.

It is desirable to have small execution-related and sparsity metrics. The smallest value of $Time$ and $Loops$ occurred for $L = 3$, but for $FB = 5 \times 5$ in the integer format and for $FB = 3 \times 3$ in the floating point format. The behavior of the sparsity metric, v_0, differed somewhat between image formats. For the integer format, sparsity was smallest when using $FB = 5 \times 5$ and $L = 3$, but for the floating point format when using $FB = 5 \times 5$ and $L = 4$. Accuracy measures are better when they have large values. For these data, using $FB = 5 \times 5$ resulted in the largest $PSNR$ and $SSIM$ values for both image formats, regardless of maximum dilation level L.

Two methods were found to relax the Parseval constraint, specified in the algorithm. One relaxed the condition number of the frame operator DD^*, and the other relaxed the constraint that $DD^* = I$. The improvements in sparsity (i.e., v_0), and speed, as measured by $Loops$ and $Time$, are considerable.

6.3 Future Work

Future work will include performing the algorithm on smaller 8×8 image patches from the original full size 256×256 test images, and piecing them together. Of major interest will be changes in execution speed, and accuracy. This research will also include further exploration of the floating point image format and the 5×5 filterbank, both separately and together.

Future research will include an examination of the Parseval Relaxation results. It is intended to further explore the feasibility of removing the Parseval condition completely. Results will be compared to those found with the constraint in place.

Bibliography

[1] INSPIRATION / TECH TODAY. How many photos will be taken in 2020?, January 10, 2020. URL https://focus.mylio.com/tech-today/how-many-photos-will-be-taken-in-2020.

[2] EJ Candes and DL Donoho. Recovering edges in ill-posed inverse problems: optimality of curvelet frames. *The Annals of Statistics*, 30(3):784–842, 2002.

[3] E. J. Candes and D. L. Donoho. New tight frames of curvelets and optimal representations of objects with piecewise c-2 singularities. *Communications on Pure and Applied Mathematics*, 57:219–266, 2004. ISSN 00103640.

[4] I. Daubechies, B. Han, A. Ron, and Z. Shen. Framelets: Mra-based constructions of wavelet frames, 2003.

[5] Yuhui Quan, Hui Ji, and Zuowei Shen. Data-driven multi-scale non-local wavelet frame construction and image recovery. *Journal of Scientific Computing*, 63(2):307–329, 2015. ISSN 0885-7474, 1573-7691. doi: 10.1007/s10915-014-9893-2.

[6] Jian-Feng Cai, Hui Ji, Zuowei Shen, and Gui-Bo Ye. Data-driven tight frame construction and image denoising. *Applied and Computational Harmonic Analysis*, 37(1): 89–105, 2014. ISSN 10635203. doi: 10.1016/j.acha.2013.10.001.

[7] Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-svd an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006. doi: 10.1017/CBO9780511794308.

[8] R Rubenstein, T Peleg, and M Elad. Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model. *IEEE Transactions on Signal Processing, Signal Processing, IEEE Transactions on, IEEE Trans. Signal Process.*, 61(3):661, 2013. ISSN 1941-0476, 1053-587X. doi: 10.1109/TSP.2012.2226445.

[9] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12): 3736–3745, 2006. ISSN 1057-7149. doi: 10.1109/tip.2006.881969.

[10] Chunhong Cao and Xieping Gao. Compressed sensing image restoration based on data-driven multi-scale tight frame. *Journal of Computational and Applied Mathematics*, 309:622–629, 2017. ISSN 03770427. doi: 10.1016/j.cam.2016.03.011.

[11] Bin Dong and Zuowei Shen. Image restoration: a data-driven perspective. In *Proceedings of the 8th International Congress on Industrial and Applied Mathematics*, pages 65–108. Higher Ed. Press, Beijing, 2015.

[12] Jae Kyu Choi, Bin Dong, and Xiaoqun Zhang. An edge driven wavelet frame model for image restoration. *Applied and Computational Harmonic Analysis*, 2018. ISSN 1063-5203.

[13] Qiu Wang, Jun Liu, Nirmal Janardhanan, Michael Zenge, Edgar Mueller, and Mariappan S Nadar. Tight frame learning for cardiovascular mri. In *2013 IEEE 10th International Symposium on Biomedical Imaging*, pages 290–293. IEEE, 2013. ISBN 146736455X.

[14] Weifeng Zhou, Jian-Feng Cai, and Hao Gao. Adaptive tight frame based medical

image reconstruction: a proof-of-concept study for computed tomography. *Inverse problems*, 29(12):125006, 2013. ISSN 0266-5611.

[15] Lina Liu, Gerlind Plonka, and Jianwei Ma. Seismic data interpolation and denoising by learning a tensor tight frame. *Inverse Problems*, 33(10):105011, 2017. ISSN 0266-5611.

[16] Min Zhang, Yunhui Shi, Na Qi, and Baocai Yin. Data-driven redundant transform based on parseval frames. *Applied Sciences*, 10(8):2891, 2020.

[17] Genjiao Zhou and Jinhong Huang. Adaptive wavelet tight frame construction for accelerating mri reconstruction. *Statistics, Optimization and Information Computing*, 5(3):200–211, 2017. ISSN 2310-5070.

[18] Wenkun Zhang, Hanming Zhang, Linyuan Wang, Ailong Cai, Lei Li, and Bin Yan. Limited angle ct reconstruction by simultaneous spatial and radon domain regularization based on tv and data-driven tight frame. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 880:107–117, 2018. ISSN 0168-9002.

[19] Peter G Casazza and Gitta Kutyniok. *Finite frames: Theory and applications*. Springer, 2012. ISBN 0817683739.

[20] Michal Aharon, Michael Elad, and Alfred M. Bruckstein. On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them. *Linear Algebra and Its Applications*, 416:48–67, 2006. ISSN 0024-3795. doi: 10.1016/j.laa.2005.06.035.

[21] Bin Dong, Jia Li, and Zuowei Shen. X-ray ct image reconstruction via wavelet frame based regularization and radon domain inpainting. *Journal of Scientific Computing*, 54 (2-3):333–349, 2013. ISSN 0885-7474. doi: 10.1007/s10915-012-9579-6.

[22] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4): 1289–1306, 2006. ISSN 0018-9448. doi: 10.1109/tit.2006.871582.

[23] Yonina C Eldar and Gitta Kutyniok. *Compressed sensing: theory and applications*. Cambridge university press, 2012. ISBN 1107394392.

[24] Emmanuel J. Candès, Yonina C. Eldar, Deanna Needell, and Paige Randall. Compressed sensing with coherent and redundant dictionaries. *Applied and Computational Harmonic Analysis*, 31(1):59–73, 2011. ISSN 10635203. doi: 10.1016/j.acha.2010.10.002.

[25] Jian-Feng Cai, Raymond H. Chan, and Zuowei Shen. A framelet-based image inpainting algorithm. *Applied and Computational Harmonic Analysis*, 24(2):131–149, 2008. ISSN 10635203. doi: 10.1016/j.acha.2007.10.002.

[26] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001. ISSN 0036-1445.

[27] D. L. Donoho, M. Elad, and V. N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1):6–18, 2006. ISSN 0018-9448. doi: 10.1109/tit.2005.860430.

[28] X Zhan, R Zhang, D Yin, and C Huo. Sar image compression using multiscale dictionary learning and sparse representation. *IEEE Geoscience and Remote Sensing Letters, Geoscience and Remote Sensing Letters, IEEE, IEEE Geosci. Remote Sensing Lett.*, 10(5):1090, 2013. ISSN 1558-0571, 1545-598X. doi: 10.1109/LGRS.2012.2230394.

[29] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993. ISSN 1053-587X.

[30] Geoffrey Davis, Stephane Mallat, and Zhifeng Zhang. Adaptive time-frequency approximations with matching pursuits. In *Wavelet Analysis and Its Applications*, volume 5, pages 271–293. Elsevier, 1994. doi: 10.1016/B978-0-08-052084-1.50018-1.

[31] Simon Foucart. Stability and robustness of weak orthogonal matching pursuits. In *Recent advances in harmonic analysis and applications*, volume 25, pages 395–405. Springer, 2012. doi: 10.1007/978-1-4614-4565-4_30.

[32] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008. ISSN 0176-4276. doi: 10.1007/s00365-007-9003-x.

[33] Zuowei Shen and Zhiqiang Xu. On b-spline framelets derived from the unitary extension principle. *SIAM Journal on Mathematical Analysis*, 45(1):127–151, 2013. ISSN 0036-1410.

[34] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. ISBN 1424475414.

[35] Benjamin Seide. Adm visual effects wiki; floating point vs integer, 2021 2021. URL http://admvfx.com/knowledge-base/floating-point-vs-integer/.

[36] Wayne W. Daniel and Chad L. Cross. *Biostatistics: A foundation for analysis in the health sciences*. Wiley, 2018. ISBN 1119282373.

[37] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo. An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems. *IEEE Trans Image Process*, 20(3):681–95, 2011. ISSN 1941-0042 (Electronic) 1057-7149 (Linking). doi: 10.1109/TIP.2010.2076294. URL https://www.ncbi.nlm.nih.gov/pubmed/20840899.

[38] Alessandro Buccini, Yonggi Park, and Lothar Reichel. Comparison of a-posteriori parameter choice rules for linear discrete ill-posed problems. *Journal of Computational and Applied Mathematics*, 373:112138, 2020. ISSN 0377-0427. doi: 10.1016/j.cam.2019.02.005.

[39] Nikolas P Galatsanos and Aggelos K Katsaggelos. Methods for choosing the regularization parameter and estimating the noise variance in image restoration and their relation. *IEEE Transactions on image processing*, 1(3):322 – 336, 1992. ISSN 1057-7149. doi: 10.1109/83.148606.

[40] Michiel E Hochstenbach, Lothar Reichel, and Giuseppe Rodriguez. Regularization parameter determination for discrete ill-posed problems. *Journal of Computational and Applied Mathematics*, 273:132 – 149, 2015. ISSN 0377-0427. doi: 10.1016/j.cam.2014.06.004.

[41] Lothar Reichel and Giuseppe Rodriguez. Old and new parameter choice rules for discrete ill-posed problems. *Numerical Algorithms*, 63(1):65 – 87, 2013. ISSN 1017-1398. doi: 10.1007/s11075-012-9612-8.

[42] Travis Bemrose and Beverly K Grunden. On the characteristics of a data-driven multiscale frame convergence algorithm. In review: copy available from corresponding author, 2021.

Appendix A

Regularization Parameters

A.1 Regularization Parameters for 3×3 Filterbanks Using an Integer Format

Pivot tables with conditional formatting are shown for all metrics and all combinations of (μ_1,μ_2) that were tested in Round 1 for integer format with 3×3 filterbank. Descriptive statistics taken from these tables are displayed in Table 3.1. Because accuracy is generally considered most important, for Round 2 testing, the focus moved to combinations that would yield optimal $PSNR$ and $SSIM$ values, but would also provide good sparsity and be cost-effective in terms of execution duration, i.e., $Loops$ and $Time$. The next area of the parameter space to be searched included only increasing μ_1 and constant μ_2.

Mean Loops		Mu2	Constant							Increasing			R1
		1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	sqrt(2)	2	5	10
Mu1	1E+05	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
Constant	1E+10	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+15	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+20	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+25	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+30	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+35	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
	1E+40	2.00	2.00	2.00	2.00	2.00	3.00	3.00	3.00	2.00	2.00	2.00	2.00
Increasing	sqrt(2)	23.05	13.58	11.98	8.78	3.28	3.45	3.45	3.45	42.50	30.65	37.10	29.45
	2	17.30	8.85	7.60	6.40	3.50	4.70	4.70	4.70	42.20	27.58	35.98	29.48
	5	13.03	5.05	5.28	4.83	4.13	4.28	4.28	4.28	33.08	24.15	35.55	29.15
	10	9.10	3.80	4.68	3.75	3.43	3.63	3.63	3.63	21.33	18.23	35.80	27.90

Figure A.1: Round 1 means for *Loops* in search for regularization parameters for integer data with 3×3 filterbank. Desirable values are formatted in green, and undesirable values in red.

Mean Time		Mu2	Constant							Increasing			R1
		1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	sqrt(2)	2	5	10
Mu1	1E+05	0.25	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
Constant	1E+10	0.21	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+15	0.21	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+20	0.20	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+25	0.20	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+30	0.20	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+35	0.20	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
	1E+40	0.20	0.30	0.41	0.54	0.67	2.64	2.64	2.64	0.18	0.18	0.18	0.18
Increasing	sqrt(2)	7.69	5.65	8.43	4.76	1.49	3.23	3.23	3.23	12.55	10.67	18.40	16.25
	2	5.09	3.22	4.58	2.97	1.66	4.88	4.88	4.88	10.48	8.03	16.45	15.40
	5	3.18	1.51	2.57	1.99	2.08	4.32	4.32	4.32	7.13	5.85	14.90	14.08
	10	1.97	0.92	1.93	1.43	1.62	3.47	3.47	3.46	4.10	4.04	15.07	12.93

Mean PSNR		Mu2	Constant							Increasing			R1
		1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	sqrt(2)	2	5	10
Mu1	1E+05	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
Constant	1E+10	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+15	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+20	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+25	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+30	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+35	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
	1E+40	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79	22.79
Increasing	sqrt(2)	23.16	23.20	23.31	22.87	23.42	23.37	23.37	23.37	22.87	22.61	22.96	22.68
	2	23.11	23.27	23.02	22.89	23.13	23.02	23.02	23.02	23.25	23.02	23.08	23.20
	5	22.95	23.12	22.87	22.79	22.80	22.80	22.80	22.80	22.83	22.72	22.87	22.70
	10	23.10	22.87	22.85	22.80	22.80	22.79	22.79	22.79	23.18	23.18	23.02	22.94

Mean SSIM		Mu2	Constant							Increasing			R1
		1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	sqrt(2)	2	5	10
Mu1	1E+05	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
Constant	1E+10	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+15	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+20	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+25	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+30	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+35	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	1E+40	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
Increasing	sqrt(2)	0.54	0.53	0.53	0.51	0.53	0.53	0.53	0.53	0.52	0.50	0.53	0.51
	2	0.53	0.53	0.52	0.51	0.52	0.52	0.52	0.52	0.53	0.52	0.52	0.53
	5	0.52	0.53	0.52	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.50
	10	0.52	0.52	0.51	0.51	0.51	0.51	0.51	0.51	0.52	0.52	0.52	0.51

Figure A.2: Round 1 means for *Time*, *PSNR*, *SSIM* in search for regularization parameters for integer data with 3×3 filterbank. Desirable values are formatted in green, and undesirable values in red.

A.2 Regularization Parameters for Floating Point Image Format

For the floating point image format, the 3×3 filterbank was utilized in all data collection. The tables of mean metric values for Round 1 testing are shown below with conditional formatting. Desirable values are formatted in green, and undesirable values in red. To get a better separation of mean values in that part of the space, the lower left portion of the pivot table results have been conditionally formatted separately from the rest of the table for four of the metrics: $Loops$, $Time$, $PSNR$, and $SSIM$. The increasing μ_1, constant μ_2 combinations were selected as the best area of the parameter space for choosing the regularization parameters in the next round of testing. The final results are shown in Section 3.2.2.

Mean Loops		Mu2 Constant 1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	Increasing sqrt(2)	2	5	FPR1 10
Mu1	1E+05	2.00	2.00	2.00	2.00	2.00	2.25	2.25	2.25	2.00	2.00	2.00	2.00
Constant	1E+10	2.00	2.00	2.00	2.00	2.00	2.25	2.25	2.25	2.00	2.00	2.00	2.00
	1E+15	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
	1E+20	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
	1E+25	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
	1E+30	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
	1E+35	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
	1E+40	2.00	2.00	2.00	2.00	2.00	2.15	2.15	2.15	2.00	2.00	2.00	2.00
Increasing	sqrt(2)	10.55	2.55	2.25	2.25	2.25	2.25	2.25	2.25	25.60	33.85	33.50	28.35
	2	8.15	3.65	3.65	3.65	3.65	3.65	3.65	3.65	19.35	15.40	33.35	26.85
	5	7.90	6.75	6.75	6.75	6.75	6.75	6.75	6.75	13.40	11.25	19.75	25.60
	10	6.45	5.75	5.80	5.80	5.80	5.80	5.80	5.80	13.90	11.90	18.15	20.70

Mean Time		Mu2 Constant 1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	Increasing sqrt(2)	2	5	FPR1 10
Mu1	1E+05	0.17	0.24	0.33	0.45	0.56	1.37	1.39	1.37	0.13	0.13	0.14	0.13
Constant	1E+10	0.17	0.24	0.33	0.44	0.55	1.40	1.39	1.39	0.14	0.13	0.14	0.13
	1E+15	0.16	0.25	0.34	0.45	0.56	1.29	1.30	1.29	0.14	0.13	0.14	0.14
	1E+20	0.16	0.24	0.34	0.45	0.56	1.30	1.30	1.30	0.14	0.13	0.14	0.14
	1E+25	0.16	0.25	0.34	0.45	0.56	1.29	1.29	1.30	0.14	0.13	0.14	0.14
	1E+30	0.16	0.25	0.34	0.45	0.57	1.30	1.30	1.30	0.14	0.13	0.14	0.14
	1E+35	0.16	0.25	0.33	0.45	0.56	1.29	1.30	1.29	0.14	0.13	0.14	0.14
	1E+40	0.16	0.25	0.33	0.45	0.56	1.29	1.29	1.29	0.14	0.14	0.14	0.14
Increasing	sqrt(2)	2.50	0.37	0.41	0.55	0.70	1.39	1.39	1.38	210.30	243.37	275.07	245.61
	2	1.68	0.62	0.89	1.18	1.46	2.96	2.95	2.98	134.86	125.44	254.60	232.82
	5	1.32	1.35	1.92	2.56	3.19	6.38	6.35	6.36	69.43	61.67	140.58	193.91
	10	0.85	1.12	1.59	2.14	2.66	5.29	5.32	5.34	59.15	49.54	108.63	146.68

Figure A.3: Round 1 means for *Loops*, *Time* in search for regularization parameters for floating point data with 3×3 filterbank. Desirable values are formatted in green, and undesirable values in red.

Mean PSNR		Mu2 Constant 1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	Increasing sqrt(2)	2	5	FPR1 10
Mu1	1E+05	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88
	1E+10	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88	21.88
Constant	1E+15	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
	1E+20	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
	1E+25	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
	1E+30	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
	1E+35	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
	1E+40	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74	23.74
Increasing	sqrt(2)	17.05	7.87	6.99	6.99	6.99	6.99	6.99	6.99	17.14	17.18	12.37	14.49
	2	19.03	14.60	14.60	14.60	14.60	14.60	14.60	14.60	17.48	17.72	16.39	15.25
	5	22.08	21.23	21.34	21.34	21.34	21.34	21.34	21.34	19.31	19.85	19.91	19.25
	10	22.51	21.62	21.73	21.73	21.73	21.73	21.73	21.73	21.71	21.48	21.51	22.50

Mean SSIM		Mu2 Constant 1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	Increasing sqrt(2)	2	5	FPR1 10
Mu1	1E+05	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405
	1E+10	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405
Constant	1E+15	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
	1E+20	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
	1E+25	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
	1E+30	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
	1E+35	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
	1E+40	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428	0.428
Increasing	sqrt(2)	0.173	0.031	0.026	0.026	0.026	0.026	0.026	0.026	0.228	0.241	0.126	0.194
	2	0.266	0.165	0.165	0.165	0.165	0.165	0.165	0.165	0.247	0.273	0.228	0.184
	5	0.420	0.381	0.385	0.385	0.385	0.385	0.385	0.385	0.333	0.352	0.347	0.324
	10	0.439	0.398	0.401	0.401	0.401	0.401	0.401	0.401	0.404	0.375	0.402	0.439

Mean V0		Mu2 Constant 1E+05	1E+10	1E+15	1E+20	1E+25	1E+30	1E+35	1E+40	Increasing sqrt(2)	2	5	FPR1 10
Mu1	1E+05	31.75	31.75	31.75	31.75	31.75	31.75	31.75	31.75	31.75	31.75	31.75	31.75
	1E+10	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85
Constant	1E+15	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
	1E+20	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
	1E+25	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
	1E+30	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
	1E+35	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
	1E+40	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70	31.70
Increasing	sqrt(2)	10.55	1.90	1.40	1.40	1.40	1.40	1.40	1.40	26.50	24.90	27.75	30.35
	2	16.00	5.80	5.80	5.80	5.80	5.80	5.80	5.80	31.40	30.60	30.85	30.85
	5	31.40	27.25	27.25	27.25	27.25	27.25	27.25	27.25	31.85	31.80	31.75	31.95
	10	31.90	30.70	30.70	30.70	30.70	30.70	30.70	30.70	31.95	31.95	31.80	31.70

Figure A.4: Round 1 means for $PSNR$, $SSIM$, v_0 in search for regularization parameters for floating point data with 3×3 filterbank. Desirable values are formatted in green, and undesirable values in red.

A.3 Regularization Parameters for 5×5 Filterbanks Using an Integer Format

Experience with previous searches for regularization parameters suggested that the parameter space to be searched would include only increasing μ_1 and constant μ_2. The pivot tables shown below include conditional formatting where green is more desirable and red is less desirable. These are the means that were plotted in Figure 3.8.

MLoops	Mu2						
Mu1	1E+12	1E+13	1E+14	1E+15	1E+16	1E+17	1E+18
1.1	21.90	8.90	5.10	2.90	2.20	2.20	2.20
1.2	14.90	8.40	6.70	2.50	2.30	2.30	2.30
1.3	16.50	11.50	5.20	2.90	2.30	2.30	2.30
1.4	15.90	6.50	3.30	2.30	2.30	2.30	2.30
1.5	10.60	6.30	3.80	3.00	3.00	3.00	3.00
1.6	19.20	6.70	3.80	2.90	2.90	2.90	2.90
1.7	15.90	6.60	4.10	3.20	3.20	3.20	3.20

MTime	Mu2						
Mu1	1E+12	1E+13	1E+14	1E+15	1E+16	1E+17	1E+18
1.1	206.62	49.18	28.19	19.00	18.21	19.20	20.30
1.2	112.90	43.87	37.35	15.73	16.23	17.11	18.13
1.3	117.88	45.84	33.92	19.12	18.60	19.69	20.69
1.4	156.65	30.85	27.22	17.65	18.57	19.86	20.79
1.5	74.73	36.21	27.09	26.53	28.14	30.43	31.58
1.6	218.38	43.27	18.54	17.67	18.30	19.48	20.61
1.7	140.82	35.96	19.40	18.82	19.36	20.51	21.67

MPSNR	Mu2						
Mu1	1E+12	1E+13	1E+14	1E+15	1E+16	1E+17	1E+18
1.1	26.77	26.76	26.81	25.56	25.72	25.72	25.72
1.2	27.57	26.72	26.43	25.56	25.56	25.56	25.56
1.3	26.87	26.77	26.40	25.32	25.43	25.43	25.43
1.4	27.16	26.74	26.77	25.62	25.62	25.62	25.62
1.5	27.05	27.50	26.99	26.96	26.96	26.96	26.96
1.6	27.13	27.19	27.87	27.00	27.00	27.00	27.00
1.7	26.90	27.16	27.61	26.87	26.87	26.87	26.87

MSSIM	Mu2						
Mu1	1E+12	1E+13	1E+14	1E+15	1E+16	1E+17	1E+18
1.1	0.65	0.65	0.66	0.61	0.62	0.62	0.62
1.2	0.67	0.65	0.64	0.61	0.61	0.61	0.61
1.3	0.64	0.65	0.64	0.61	0.60	0.60	0.60
1.4	0.66	0.65	0.65	0.61	0.61	0.61	0.61
1.5	0.65	0.67	0.65	0.65	0.65	0.65	0.65
1.6	0.66	0.66	0.68	0.65	0.65	0.65	0.65
1.7	0.65	0.65	0.68	0.65	0.65	0.65	0.65

MV0	Mu2						
Mu1	1E+12	1E+13	1E+14	1E+15	1E+16	1E+17	1E+18
1.1	19.40	14.30	13.30	12.40	12.10	12.10	12.10
1.2	21.90	17.50	15.10	12.40	12.40	12.40	12.40
1.3	24.50	19.20	15.60	13.90	12.60	12.60	12.60
1.4	23.90	20.10	14.00	12.80	12.80	12.80	12.80
1.5	23.10	20.60	16.00	15.10	15.10	15.10	15.10
1.6	25.60	22.40	17.10	15.20	15.20	15.20	15.20
1.7	26.70	22.30	19.10	16.80	16.80	16.80	16.80

Figure A.5: Means for all metrics in search for regularization parameters for integer data with 5×5 filterbank. Green formatting is more desirable and red is less desirable.

A.4 Regularization Parameters for Parseval Relaxation Methods

The search for regularization parameters to be used with the Parseval Relaxation function relied on the experience obtained from earlier searches and restricted the exploration to increasing μ_1 and constant μ_2. The pivot tables shown below include conditional formatting where green in more desirable and red is less desirable. These are the means from which the choice of final regularization parameters was made.

Mean Loops		Mu2							Constant							Pars
		1E+15	1E+16	1E+17	1E+18	1E+19	1E+20	1E+21	1E+22	1E+23	1E+24	1E+25	1E+26	1E+27	1E+28	1E+29
Mu1	1.1	6.8	3.1	2.3	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2
Increasing	1.2	5.8	3.3	2.5	2.3	2.3	2.3	2.3	2.3	2.3	2.3	2.3	2.3	2.3	2.3	2.3
	1.3	5.2	3.4	2.8	2.6	2.6	2.6	2.6	2.6	2.6	2.6	2.6	2.6	2.6	2.6	2.6
	1.4	5.0	3.6	3.1	2.9	2.8	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9
	1.5	5.0	3.6	3.2	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
	1.6	4.5	3.4	3.0	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.9
	1.7	4.6	4.1	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8
	1.8	4.9	4.1	3.8	3.9	3.9	3.9	3.9	3.9	3.9	3.9	3.9	3.9	3.9	3.9	3.9
	1.9	4.8	4.1	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8	3.8

Mean Time		Mu2							Constant							Pars
		1E+15	1E+16	1E+17	1E+18	1E+19	1E+20	1E+21	1E+22	1E+23	1E+24	1E+25	1E+26	1E+27	1E+28	1E+29
Mu1	1.1	371.1	116.5	59.0	50.0	47.2	46.6	46.9	46.7	46.7	46.8	46.7	46.6	46.6	46.8	46.7
Increasing	1.2	302.8	116.1	67.7	55.3	53.0	52.8	53.0	52.7	52.9	52.7	52.7	52.8	52.7	52.7	52.7
	1.3	258.0	121.3	78.5	66.9	65.2	65.1	65.4	65.3	65.0	64.9	65.0	65.5	65.0	65.1	64.8
	1.4	243.2	128.5	92.8	78.1	75.2	75.2	75.3	75.3	75.5	75.0	75.2	75.2	75.3	75.2	75.2
	1.5	236.6	129.1	97.4	82.1	79.5	79.0	79.5	79.6	79.4	79.3	79.4	79.6	79.2	79.4	79.2
	1.6	204.2	120.5	90.0	79.4	77.3	77.3	77.4	77.3	77.3	77.2	77.1	77.5	77.1	77.1	77.2
	1.7	210.1	150.5	122.3	113.4	111.6	111.3	111.5	111.4	111.7	111.6	111.3	111.5	111.4	111.7	111.5
	1.8	225.4	151.2	125.0	118.9	117.5	117.7	117.7	117.8	117.6	117.4	117.5	117.6	117.4	117.3	117.6
	1.9	212.9	150.2	124.8	115.7	113.8	113.8	113.7	113.7	113.7	113.3	113.7	113.4	113.4	113.5	113.5

Figure A.6: Round 1 means for *Loops* and *Time* in search for regularization parameters for Parseval relaxation methods. Desirable values are formatted in green, and undesirable values in red.

Mean Time		Mu2							Constant							Pars
		1E+15	1E+16	1E+17	1E+18	1E+19	1E+20	1E+21	1E+22	1E+23	1E+24	1E+25	1E+26	1E+27	1E+28	1E+29
Mu1	1.1	371.1	116.5	59.0	50.0	47.2	46.6	46.9	46.7	46.7	46.8	46.7	46.6	46.6	46.8	46.7
	1.2	302.8	116.1	67.7	55.3	53.0	52.8	53.0	52.7	52.9	52.7	52.7	52.8	52.7	52.7	52.7
	1.3	258.0	121.3	78.5	66.9	65.2	65.1	65.4	65.3	65.0	64.9	65.0	65.5	65.0	65.1	64.8
	1.4	243.2	128.5	92.8	78.1	75.2	75.2	75.3	75.3	75.5	75.0	75.2	75.2	75.3	75.2	75.2
	1.5	236.6	129.1	97.4	82.1	79.5	79.0	79.5	79.6	79.4	79.3	79.4	79.6	79.2	79.4	79.2
	1.6	204.2	120.5	90.0	79.4	77.3	77.3	77.4	77.3	77.3	77.2	77.1	77.5	77.1	77.1	77.2
	1.7	210.1	150.5	122.3	113.4	111.6	111.3	111.5	111.4	111.7	111.6	111.3	111.5	111.4	111.7	111.5
	1.8	225.4	151.2	125.0	118.9	117.5	117.7	117.7	117.8	117.6	117.4	117.5	117.6	117.4	117.3	117.6
	1.9	212.9	150.2	124.8	115.7	113.8	113.8	113.7	113.7	113.7	113.3	113.7	113.4	113.4	113.5	113.5
Increasing																

Mean SSIM		Mu2							Constant							Pars
		1E+15	1E+16	1E+17	1E+18	1E+19	1E+20	1E+21	1E+22	1E+23	1E+24	1E+25	1E+26	1E+27	1E+28	1E+29
Mu1	1.1	0.608	0.594	0.574	0.563	0.561	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560
	1.2	0.610	0.591	0.572	0.562	0.561	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560	0.560
	1.3	0.597	0.587	0.576	0.570	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568	0.568
	1.4	0.590	0.589	0.577	0.574	0.573	0.572	0.572	0.572	0.572	0.572	0.572	0.572	0.572	0.572	0.572
	1.5	0.585	0.580	0.569	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565
	1.6	0.581	0.572	0.564	0.560	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559
	1.7	0.583	0.576	0.568	0.566	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565	0.565
	1.8	0.574	0.568	0.564	0.562	0.561	0.561	0.561	0.561	0.561	0.561	0.561	0.561	0.561	0.561	0.561
	1.9	0.569	0.564	0.556	0.553	0.552	0.552	0.552	0.552	0.552	0.552	0.552	0.552	0.552	0.552	0.552
Increasing																

Mean V0		Mu2							Constant							Pars
		1E+15	1E+16	1E+17	1E+18	1E+19	1E+20	1E+21	1E+22	1E+23	1E+24	1E+25	1E+26	1E+27	1E+28	1E+29
Mu1	1.1	18.2	16.5	15.9	15.8	15.8	15.8	15.8	15.8	15.8	15.8	15.8	15.8	15.8	15.8	15.8
	1.2	19.6	17.6	16.6	16.4	16.3	16.3	16.3	16.3	16.3	16.3	16.3	16.3	16.3	16.3	16.3
	1.3	20.8	18.8	18.0	17.7	17.7	17.8	17.8	17.8	17.8	17.8	17.8	17.8	17.8	17.8	17.8
	1.4	21.6	20.0	19.2	18.6	18.5	18.6	18.6	18.6	18.6	18.6	18.6	18.6	18.6	18.6	18.6
	1.5	23.0	20.7	19.6	19.2	19.2	19.2	19.2	19.2	19.2	19.2	19.2	19.2	19.2	19.2	19.2
	1.6	23.4	21.1	20.2	19.9	19.9	19.9	19.9	19.9	19.9	19.9	19.9	19.9	19.9	19.9	19.9
	1.7	24.6	23.4	22.6	22.5	22.5	22.5	22.5	22.5	22.5	22.5	22.5	22.5	22.5	22.5	22.5
	1.8	26.0	24.2	23.3	23.4	23.4	23.4	23.4	23.4	23.4	23.4	23.4	23.4	23.4	23.4	23.4
	1.9	26.0	24.4	23.6	23.5	23.5	23.5	23.5	23.5	23.5	23.5	23.5	23.5	23.5	23.5	23.5
Increasing																

Figure A.7: Round 1 means for $PSNR$, $SSIM$ and v_0 in search for regularization parameters for Parseval relaxation methods. Desirable values are formatted in green, and undesirable values in red.

Appendix B

SAS Output

Three-way analyses were performed for each metric outcome using three independent variables: image format, filterbank size, and maximum dilation level. The results for each metric are summarized in individual sections below.

B.1 $Time$

A three-way ANOVA was performed for the metric $Time$. Model assumptions require normality and homogeneity (equality) of variances among all the subpopulations, but these were not sufficiently satisfied. The response variable $Time$ was transformed by the natural log and the analysis was re-run using $ln(Time)$ as the response variable. Because normality was still an issue, the data were analyzed separating the data into two populations by image format. Both groups were analyzed using a two-way ANOVA with the response $ln(Time)$, and factors filterbank size FB and maximum dilation level L.

For data using the integer image format, the two-way interaction between L and FB was strongly significant with $p < 0.0001$. The means for all four combinations of these two factors were significantly different from each other.

For data using the floating point image format, the two-way interaction between L and FB was strongly significant with $p = 0.0019$. The means for all combinations of these two factors were significantly different from each other, except for the two combinations involving $FB = 5 \times 5$.

Time Full Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

Time Full Model: Assumptions Violated

The GLM Procedure

Dependent Variable: Time Time

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	300813.4317	42973.3474	54.20	<.0001
Error	792	627957.2979	792.8754		
Corrected Total	799	928770.7295			

R-Square	Coeff Var	Root MSE	Time Mean
0.323883	131.1431	28.15804	21.47123

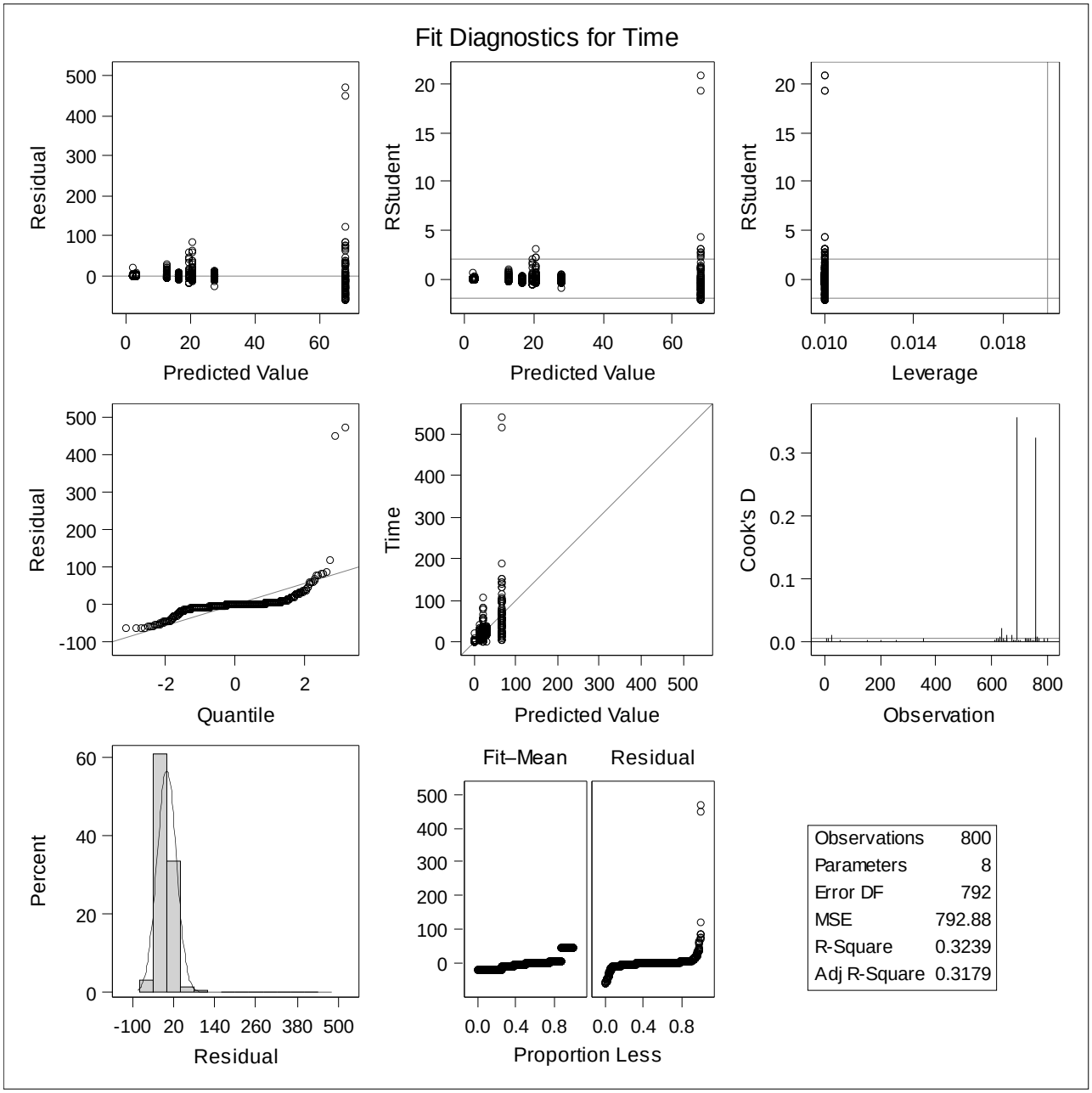
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	78263.46219	78263.46219	98.71	<.0001
L	1	54558.29049	54558.29049	68.81	<.0001
Format*L	1	55330.38529	55330.38529	69.78	<.0001
FB	1	64343.31538	64343.31538	81.15	<.0001
Format*FB	1	34.62660	34.62660	0.04	0.8345
L*FB	1	22441.07397	22441.07397	28.30	<.0001
Format*L*FB	1	25842.27774	25842.27774	32.59	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	78263.46219	78263.46219	98.71	<.0001
L	1	54558.29049	54558.29049	68.81	<.0001
Format*L	1	55330.38529	55330.38529	69.78	<.0001
FB	1	64343.31538	64343.31538	81.15	<.0001
Format*FB	1	34.62660	34.62660	0.04	0.8345
L*FB	1	22441.07397	22441.07397	28.30	<.0001
Format*L*FB	1	25842.27774	25842.27774	32.59	<.0001

Time Full Model: Assumptions Violated

The GLM Procedure

Dependent Variable: Time Time



Time Full Model Natural Log Transformation

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

Time Full Model Natural Log Transformation

The GLM Procedure

Dependent Variable: ln_Time

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	757.3817149	108.1973878	411.60	<.0001
Error	792	208.1908207	0.2628672		
Corrected Total	799	965.5725357			

R-Square	Coeff Var	Root MSE	ln_Time Mean
0.784386	20.56637	0.512706	2.492933

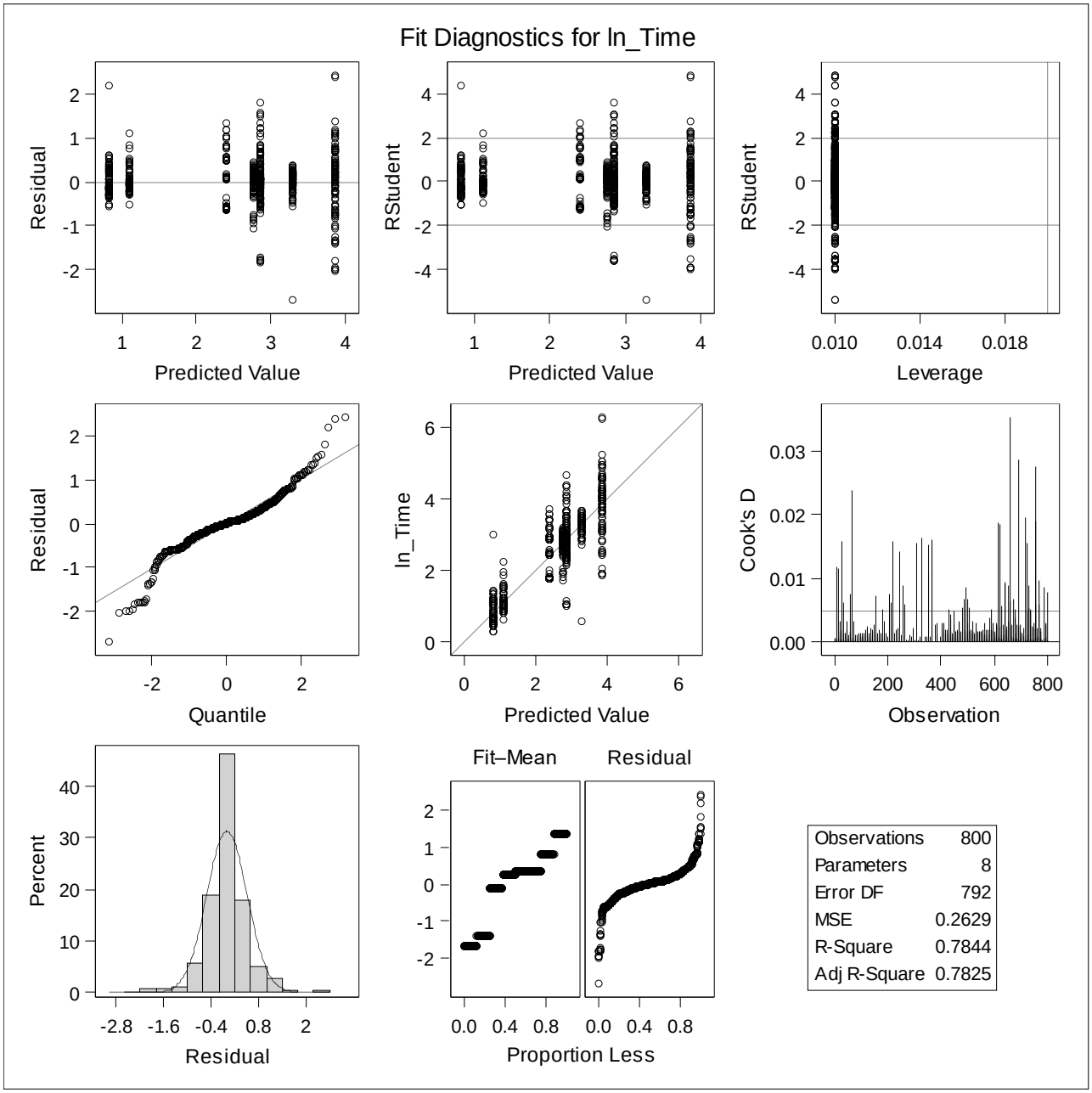
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	273.9646072	273.9646072	1042.22	<.0001
L	1	65.4964304	65.4964304	249.16	<.0001
Format*L	1	35.9768193	35.9768193	136.86	<.0001
FB	1	197.0888959	197.0888959	749.77	<.0001
Format*FB	1	159.8538872	159.8538872	608.12	<.0001
L*FB	1	5.6663570	5.6663570	21.56	<.0001
Format*L*FB	1	19.3347180	19.3347180	73.55	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	273.9646072	273.9646072	1042.22	<.0001
L	1	65.4964304	65.4964304	249.16	<.0001
Format*L	1	35.9768193	35.9768193	136.86	<.0001
FB	1	197.0888959	197.0888959	749.77	<.0001
Format*FB	1	159.8538872	159.8538872	608.12	<.0001
L*FB	1	5.6663570	5.6663570	21.56	<.0001
Format*L*FB	1	19.3347180	19.3347180	73.55	<.0001

Time Full Model Natural Log Transformation

The GLM Procedure

Dependent Variable: ln_Time



Integer: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	ln_Time
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information

Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates

Cov Parm	*Estimate*
Residual	0.3181

Integer: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	688.7
AIC (Smaller is Better)	690.7
AICC (Smaller is Better)	690.7
BIC (Smaller is Better)	694.7

Type 3 Tests of Fixed Effects

Effect	*Num DF*	*Den DF*	*F Value*	*Pr > F*
L	1	396	312.07	<.0001
FB	1	396	3.06	0.0810
*L*FB*	1	396	72.20	<.0001

Least Squares Means

Effect	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Alpha*	*Lower*	*Upper*
*L*FB*	L=3	3x3	2.7702	0.05640	396	49.12	<.0001	0.05	2.6593	2.8811
*L*FB*	L=3	5x5	2.3897	0.05640	396	42.37	<.0001	0.05	2.2788	2.5005
*L*FB*	L=4	3x3	3.2874	0.05640	396	58.28	<.0001	0.05	3.1765	3.3982
*L*FB*	L=4	5x5	3.8653	0.05640	396	68.53	<.0001	0.05	3.7544	3.9762

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Adjustment*	*Adj P*	*Alpha*	*Lower*	*Upper*
*L*FB*	L=3	3x3	L=3	5x5	0.3806	0.07977	396	4.77	<.0001	Tukey	<.0001	0.05	0.2238	0.5374
*L*FB*	L=3	3x3	L=4	3x3	-0.5171	0.07977	396	-6.48	<.0001	Tukey	<.0001	0.05	-0.6740	-0.3603
*L*FB*	L=3	3x3	L=4	5x5	-1.0951	0.07977	396	-13.73	<.0001	Tukey	<.0001	0.05	-1.2519	-0.9382
*L*FB*	L=3	5x5	L=4	3x3	-0.8977	0.07977	396	-11.25	<.0001	Tukey	<.0001	0.05	-1.0545	-0.7409
*L*FB*	L=3	5x5	L=4	5x5	-1.4756	0.07977	396	-18.50	<.0001	Tukey	<.0001	0.05	-1.6324	-1.3188
*L*FB*	L=4	3x3	L=4	5x5	-0.5779	0.07977	396	-7.25	<.0001	Tukey	<.0001	0.05	-0.7347	-0.4211

Differences of Least Squares Means

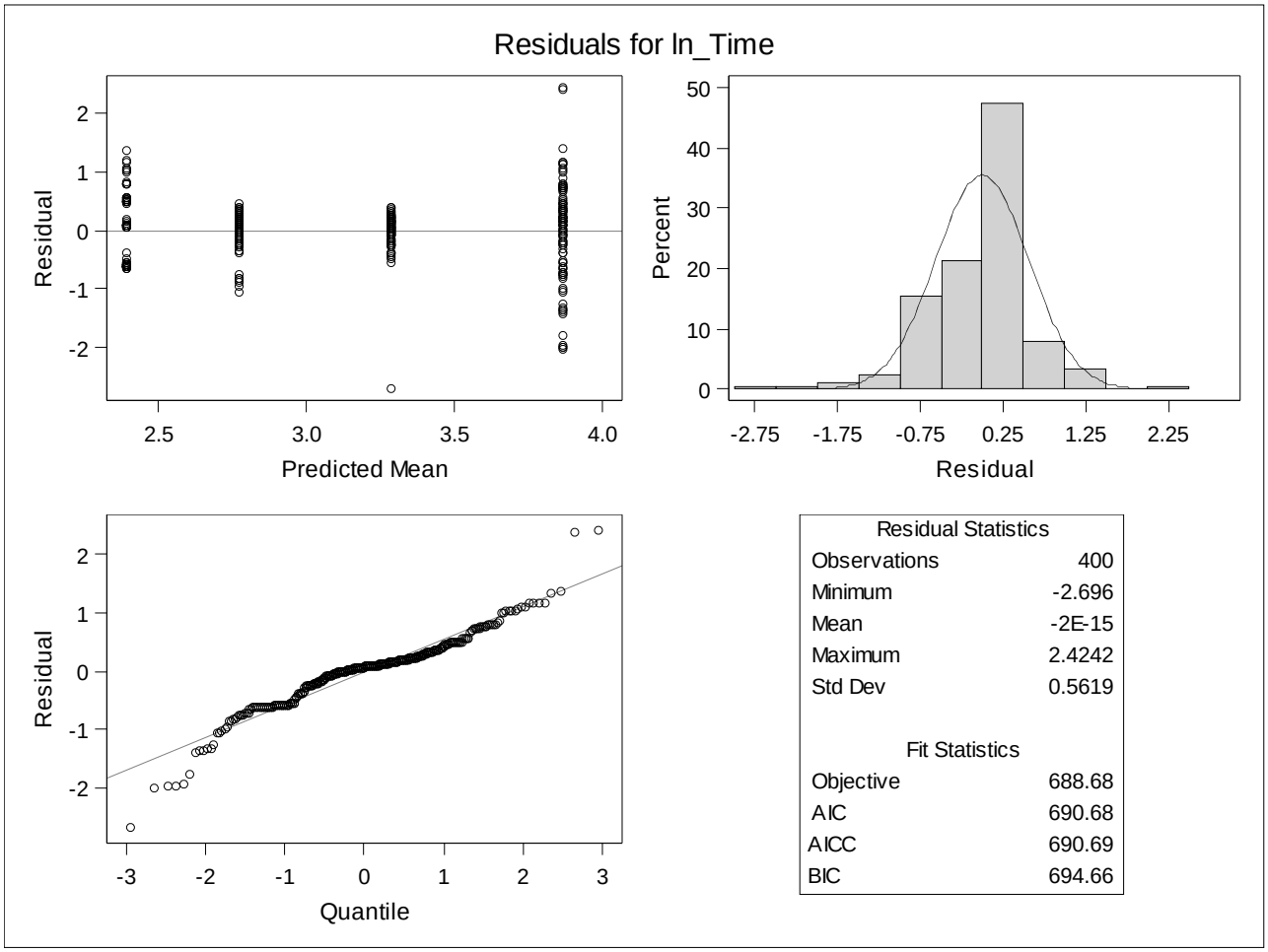
Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	3x3	L=3	5x5	0.1748	0.5864
*L*FB*	L=3	3x3	L=4	3x3	-0.7229	-0.3114
*L*FB*	L=3	3x3	L=4	5x5	-1.3009	-0.8893
*L*FB*	L=3	5x5	L=4	3x3	-1.1035	-0.6919

Integer: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	5x5	-1.6814	-1.2698
*L*FB*	L=4	3x3	L=4	5x5	-0.7837	-0.3721



Integer: Time Two-way Interaction Model Natural Log Transformation

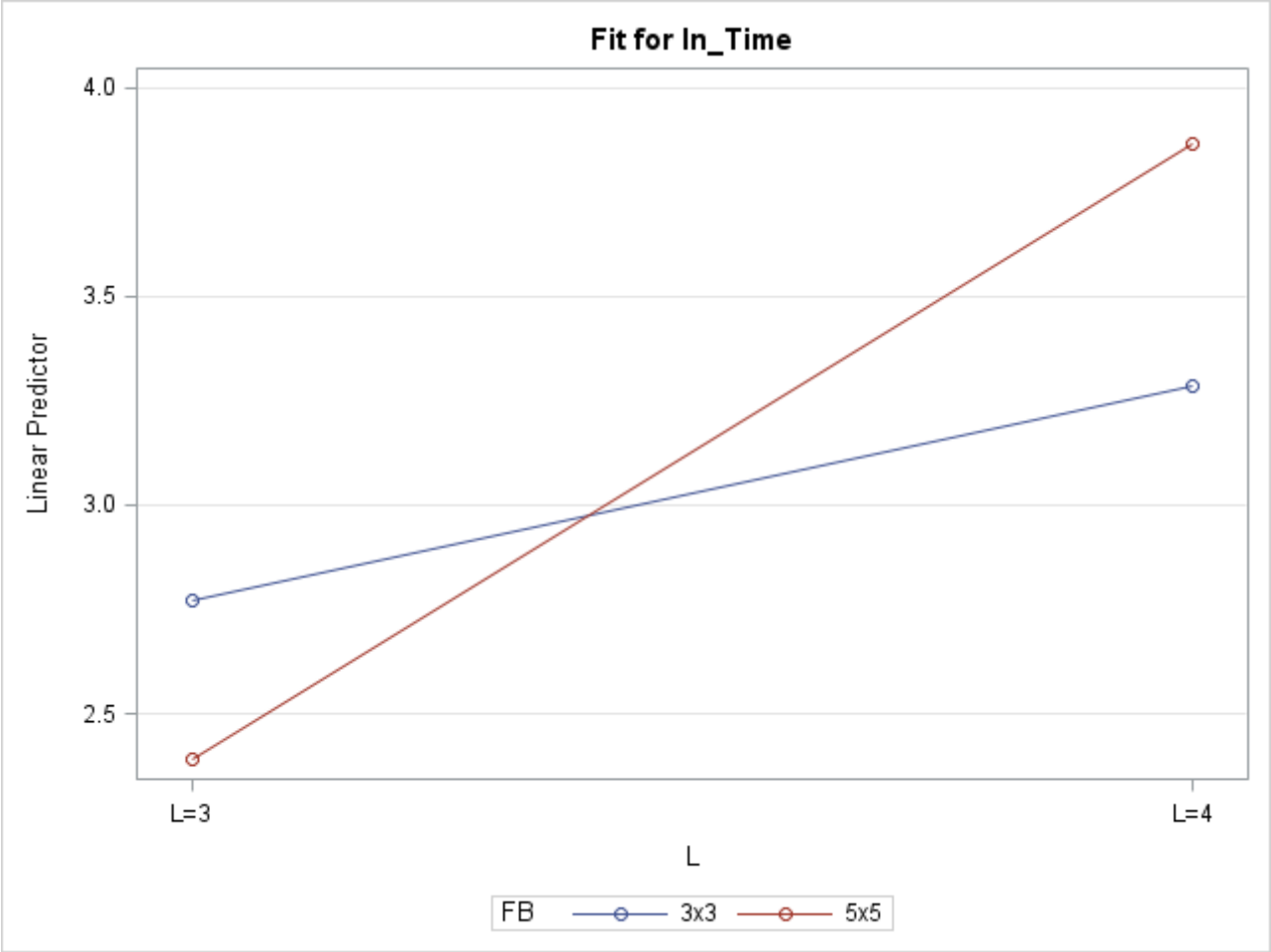
The PLM Procedure

Store Information	
Item Store	WORK.INTEGERTIMEPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:03:58
Response Variable	ln_Time
Distribution	Normal
Class Variables	L FB
Model Effects	Intercept L FB L*FB

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Integer: Time Two-way Interaction Model Natural Log Transformation

The PLM Procedure



Floating Point: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	ln_Time
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	0.2076

Floating Point: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Fit Statistics

-2 Res Log Likelihood	519.7
AIC (Smaller is Better)	521.7
AICC (Smaller is Better)	521.7
BIC (Smaller is Better)	525.6

Type 3 Tests of Fixed Effects

Effect	*Num DF*	*Den DF*	*F Value*	*Pr > F*
L	1	396	10.57	0.0012
FB	1	396	1714.62	<.0001
*L*FB*	1	396	9.80	0.0019

Least Squares Means

Effect	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Alpha*	*Lower*	*Upper*
*L*FB*	L=3	3x3	0.8190	0.04556	396	17.97	<.0001	0.05	0.7294	0.9086
*L*FB*	L=3	5x5	2.8483	0.04556	396	62.51	<.0001	0.05	2.7588	2.9379
*L*FB*	L=4	3x3	1.1097	0.04556	396	24.36	<.0001	0.05	1.0202	1.1993
*L*FB*	L=4	5x5	2.8539	0.04556	396	62.63	<.0001	0.05	2.7643	2.9434

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Adjustment*	*Adj P*	*Alpha*	*Lower*	*Upper*
*L*FB*	L=3	3x3	L=3	5x5	-2.0293	0.06444	396	-31.49	<.0001	Tukey	<.0001	0.05	-2.1560	-1.9026
*L*FB*	L=3	3x3	L=4	3x3	-0.2907	0.06444	396	-4.51	<.0001	Tukey	<.0001	0.05	-0.4174	-0.1641
*L*FB*	L=3	3x3	L=4	5x5	-2.0348	0.06444	396	-31.58	<.0001	Tukey	<.0001	0.05	-2.1615	-1.9082
*L*FB*	L=3	5x5	L=4	3x3	1.7386	0.06444	396	26.98	<.0001	Tukey	<.0001	0.05	1.6119	1.8653
*L*FB*	L=3	5x5	L=4	5x5	-0.00553	0.06444	396	-0.09	0.9317	Tukey	0.9998	0.05	-0.1322	0.1212
*L*FB*	L=4	3x3	L=4	5x5	-1.7441	0.06444	396	-27.07	<.0001	Tukey	<.0001	0.05	-1.8708	-1.6174

Differences of Least Squares Means

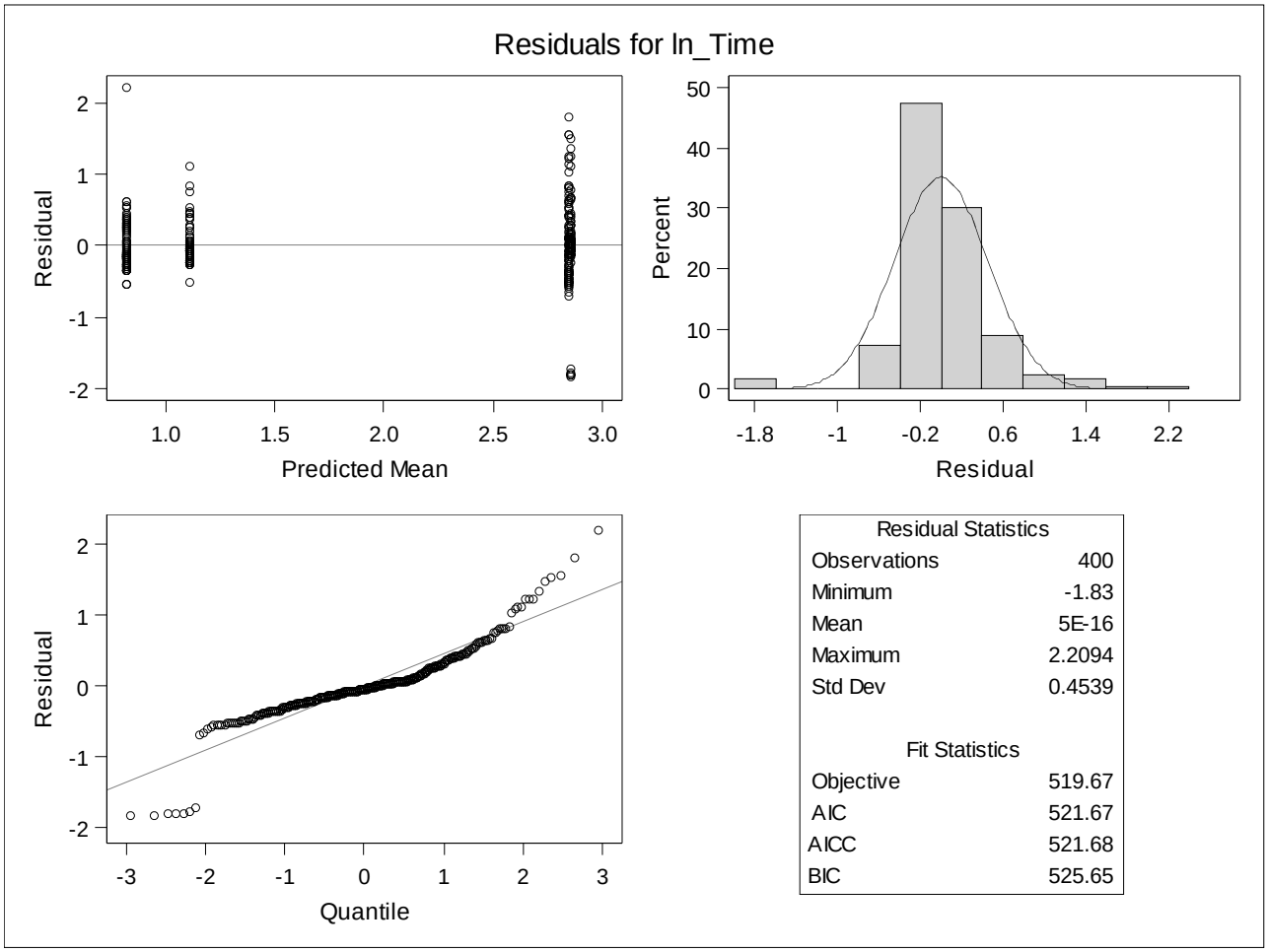
Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	3x3	L=3	5x5	-2.1956	-1.8631
*L*FB*	L=3	3x3	L=4	3x3	-0.4570	-0.1245
*L*FB*	L=3	3x3	L=4	5x5	-2.2011	-1.8686
*L*FB*	L=3	5x5	L=4	3x3	1.5723	1.9048

Floating Point: Time Two-way Interaction Model Natural Log Transformation

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	5x5	-0.1718	0.1607
*L*FB*	L=4	3x3	L=4	5x5	-1.9104	-1.5779

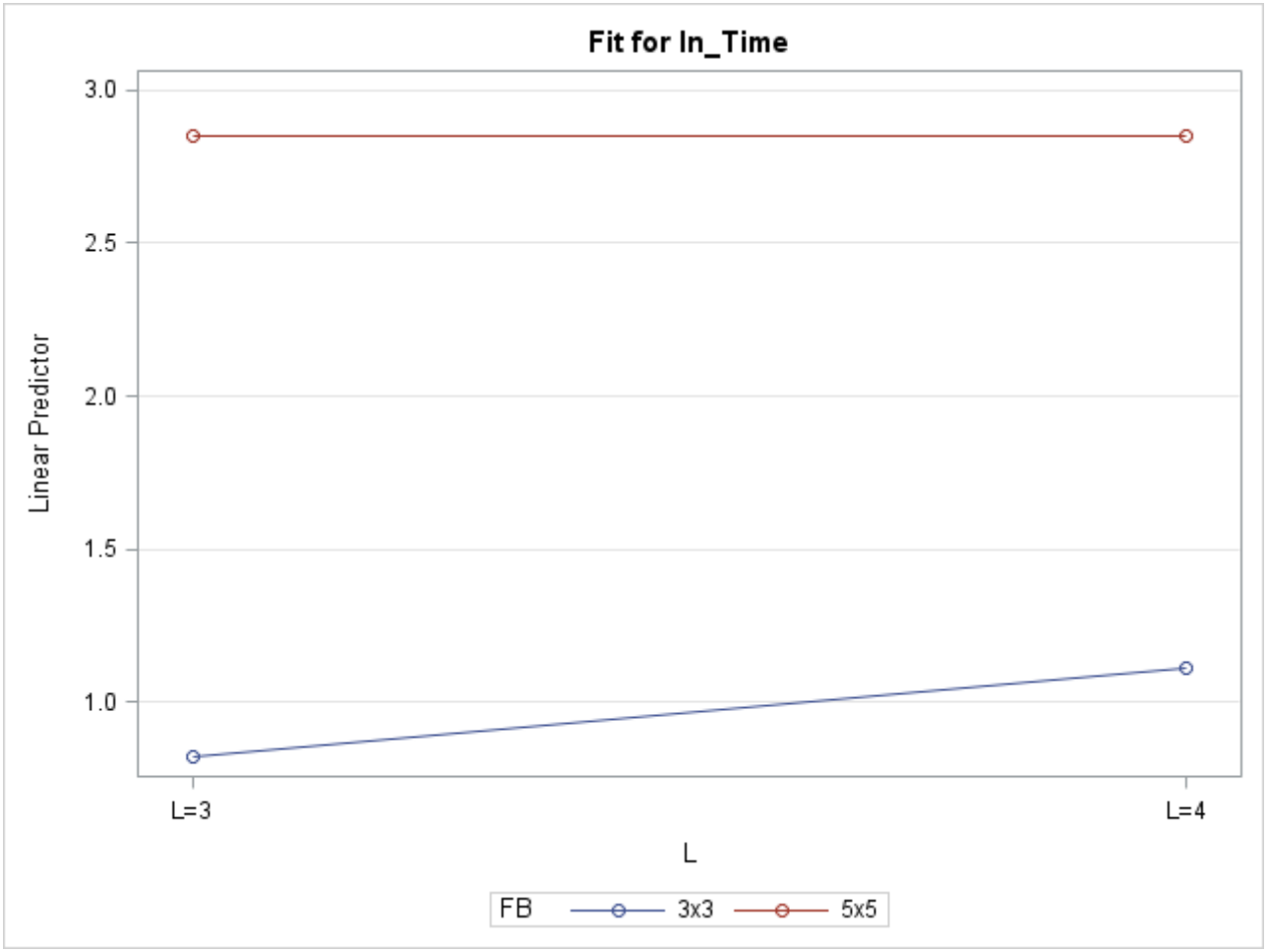


Floating Point: Time Two-way Interaction Model Natural Log Transformation

The PLM Procedure

Store Information	
Item Store	WORK.FLOATINGTIMEPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:04:05
Response Variable	ln_Time
Distribution	Normal
Class Variables	L FB
Model Effects	Intercept L FB L*FB

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5



B.2 *Loops*

A three-way ANOVA was performed for the metric $Loops$. Model assumptions were not sufficiently satisfied. The response variable $Loops$ was transformed by the natural log and the analysis was re-run using $ln(Loops)$ as the response variable. Because normality was still an issue, the data were analyzed separating the data into two populations by image format. Both groups were analyzed using a two-way ANOVA with the response $ln(Loops)$, and factors filterbank size FB and maximum dilation level L.

For data using the integer image format, the two-way interaction between L and FB was strongly significant with $p < 0.0001$. The means of $Loops$ for all combinations of these two factors were significantly different from each other, except for the two combinations involving $FB = 3 \times 3$.

For data using the floating point image format, the two-way interaction between L and FB was strongly significant with $p < 0.0001$. The mean of $Loops$ for combination $L = 3$ and $FB = 5 \times 5$ was significantly larger than the means for all other combinations. No other pairings were significantly different.

Loops Full Model: Assumptions Violated

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

Loops Full Model

The GLM Procedure

Dependent Variable: Loops Loops

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	18739.75500	2677.10786	58.38	<.0001
Error	792	36317.12000	45.85495		
Corrected Total	799	55056.87500			

R-Square	Coeff Var	Root MSE	Loops Mean
0.340371	63.65808	6.771628	10.63750

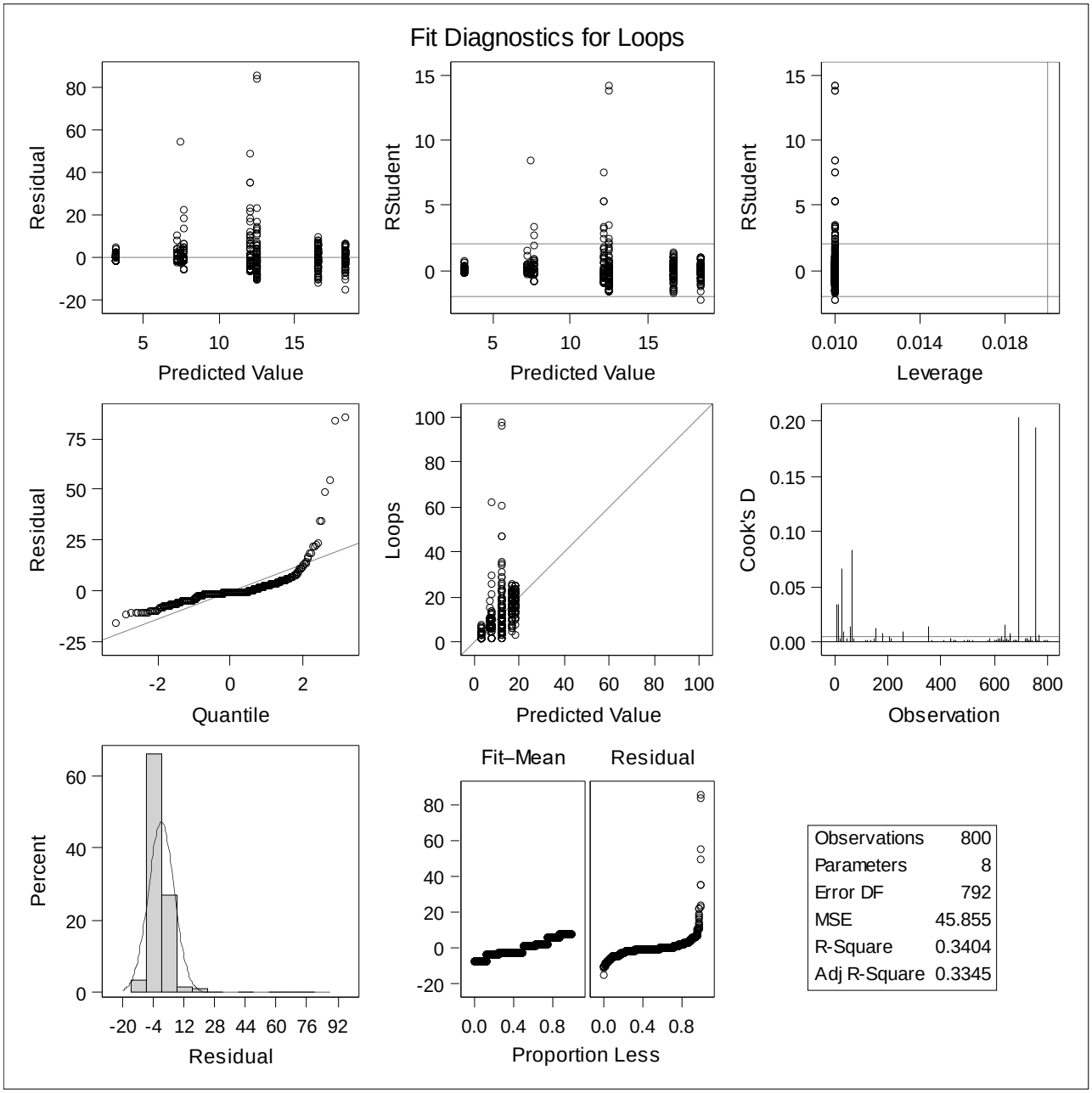
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	3256.245000	3256.245000	71.01	<.0001
L	1	531.380000	531.380000	11.59	0.0007
Format*L	1	3073.280000	3073.280000	67.02	<.0001
FB	1	2513.405000	2513.405000	54.81	<.0001
Format*FB	1	7478.645000	7478.645000	163.09	<.0001
L*FB	1	134.480000	134.480000	2.93	0.0872
Format*L*FB	1	1752.320000	1752.320000	38.21	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	3256.245000	3256.245000	71.01	<.0001
L	1	531.380000	531.380000	11.59	0.0007
Format*L	1	3073.280000	3073.280000	67.02	<.0001
FB	1	2513.405000	2513.405000	54.81	<.0001
Format*FB	1	7478.645000	7478.645000	163.09	<.0001
L*FB	1	134.480000	134.480000	2.93	0.0872
Format*L*FB	1	1752.320000	1752.320000	38.21	<.0001

Loops Full Model

The GLM Procedure

Dependent Variable: Loops Loops



Natural Log Loops Full Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

Natural Log Loops Full Model

The GLM Procedure

Dependent Variable: ln_Loops

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	222.6157205	31.8022458	173.26	<.0001
Error	792	145.3698871	0.1835478		
Corrected Total	799	367.9856076			

R-Square	Coeff Var	Root MSE	ln_Loops Mean
0.604958	20.03717	0.428425	2.138151

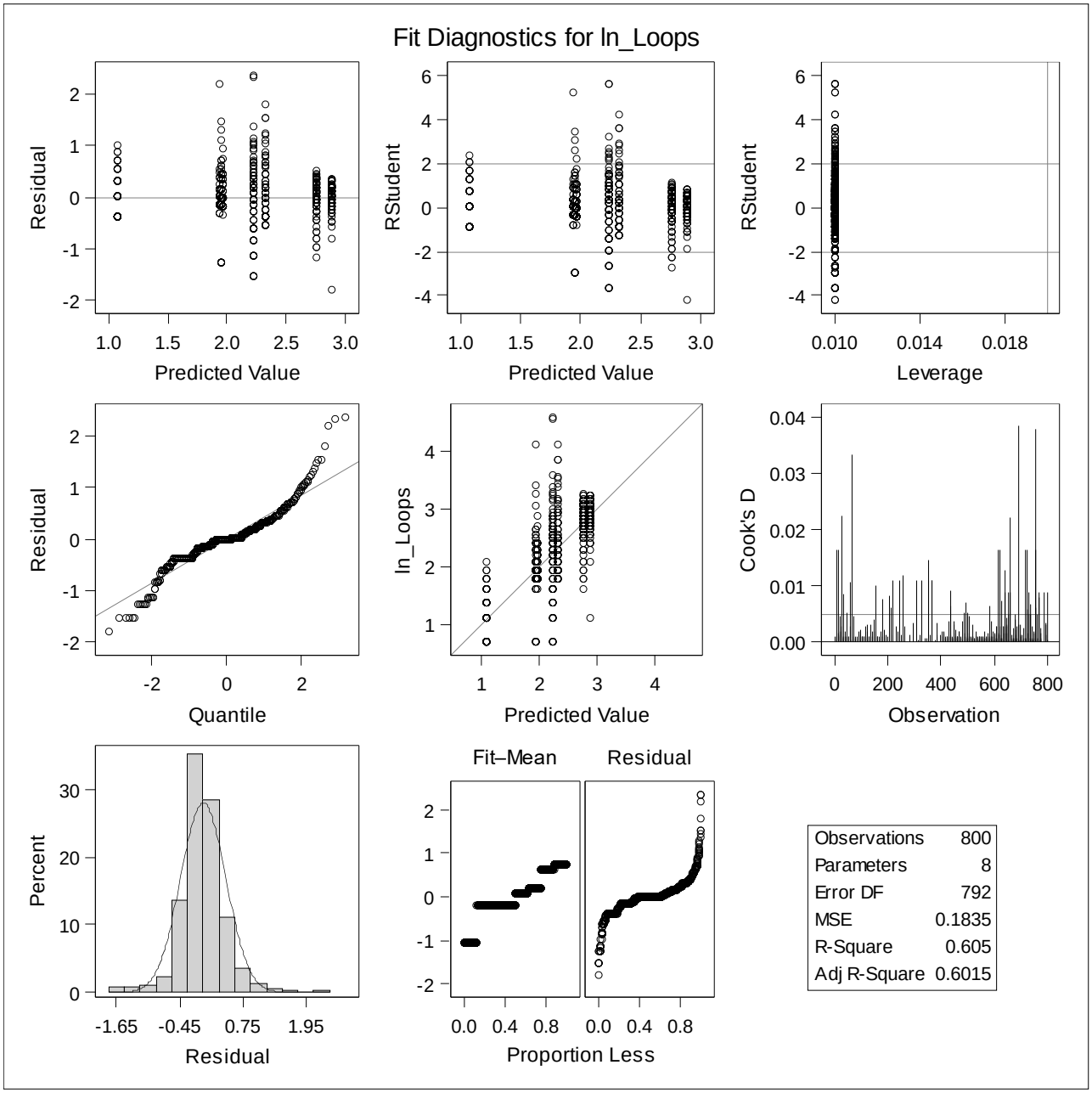
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	7.65313736	7.65313736	41.70	<.0001
L	1	10.75897934	10.75897934	58.62	<.0001
Format*L	1	32.99065379	32.99065379	179.74	<.0001
FB	1	48.50128198	48.50128198	264.24	<.0001
Format*FB	1	91.70439457	91.70439457	499.62	<.0001
L*FB	1	5.16613541	5.16613541	28.15	<.0001
Format*L*FB	1	25.84113803	25.84113803	140.79	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	7.65313736	7.65313736	41.70	<.0001
L	1	10.75897934	10.75897934	58.62	<.0001
Format*L	1	32.99065379	32.99065379	179.74	<.0001
FB	1	48.50128198	48.50128198	264.24	<.0001
Format*FB	1	91.70439457	91.70439457	499.62	<.0001
L*FB	1	5.16613541	5.16613541	28.15	<.0001
Format*L*FB	1	25.84113803	25.84113803	140.79	<.0001

Natural Log Loops Full Model

The GLM Procedure

Dependent Variable: ln_Loops



Integer: Natural Log Loops Full Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	ln_Loops
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	0.2202

Integer: Natural Log Loops Full Model

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	543.0
AIC (Smaller is Better)	545.0
AICC (Smaller is Better)	545.0
BIC (Smaller is Better)	549.0

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
L	1	396	184.88	<.0001
FB	1	396	621.17	<.0001
L*FB	1	396	122.87	<.0001

Least Squares Means

Effect	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
L*FB	L=3	3x3	2.7618	0.04693	396	58.85	<.0001	0.05	2.6695	2.8541
L*FB	L=3	5x5	1.0720	0.04693	396	22.84	<.0001	0.05	0.9798	1.1643
L*FB	L=4	3x3	2.8797	0.04693	396	61.36	<.0001	0.05	2.7875	2.9720
L*FB	L=4	5x5	2.2303	0.04693	396	47.53	<.0001	0.05	2.1380	2.3225

Differences of Least Squares Means

Effect	L	FB	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P	Alpha	Lower	Upper
L*FB	L=3	3x3	L=3	5x5	1.6898	0.06637	396	25.46	<.0001	Tukey	<.0001	0.05	1.5593	1.8202
L*FB	L=3	3x3	L=4	3x3	-0.1179	0.06637	396	-1.78	0.0764	Tukey	0.2861	0.05	-0.2484	0.01256
L*FB	L=3	3x3	L=4	5x5	0.5315	0.06637	396	8.01	<.0001	Tukey	<.0001	0.05	0.4010	0.6620
L*FB	L=3	5x5	L=4	3x3	-1.8077	0.06637	396	-27.24	<.0001	Tukey	<.0001	0.05	-1.9381	-1.6772
L*FB	L=3	5x5	L=4	5x5	-1.1583	0.06637	396	-17.45	<.0001	Tukey	<.0001	0.05	-1.2887	-1.0278
L*FB	L=4	3x3	L=4	5x5	0.6494	0.06637	396	9.79	<.0001	Tukey	<.0001	0.05	0.5189	0.7799

Differences of Least Squares Means

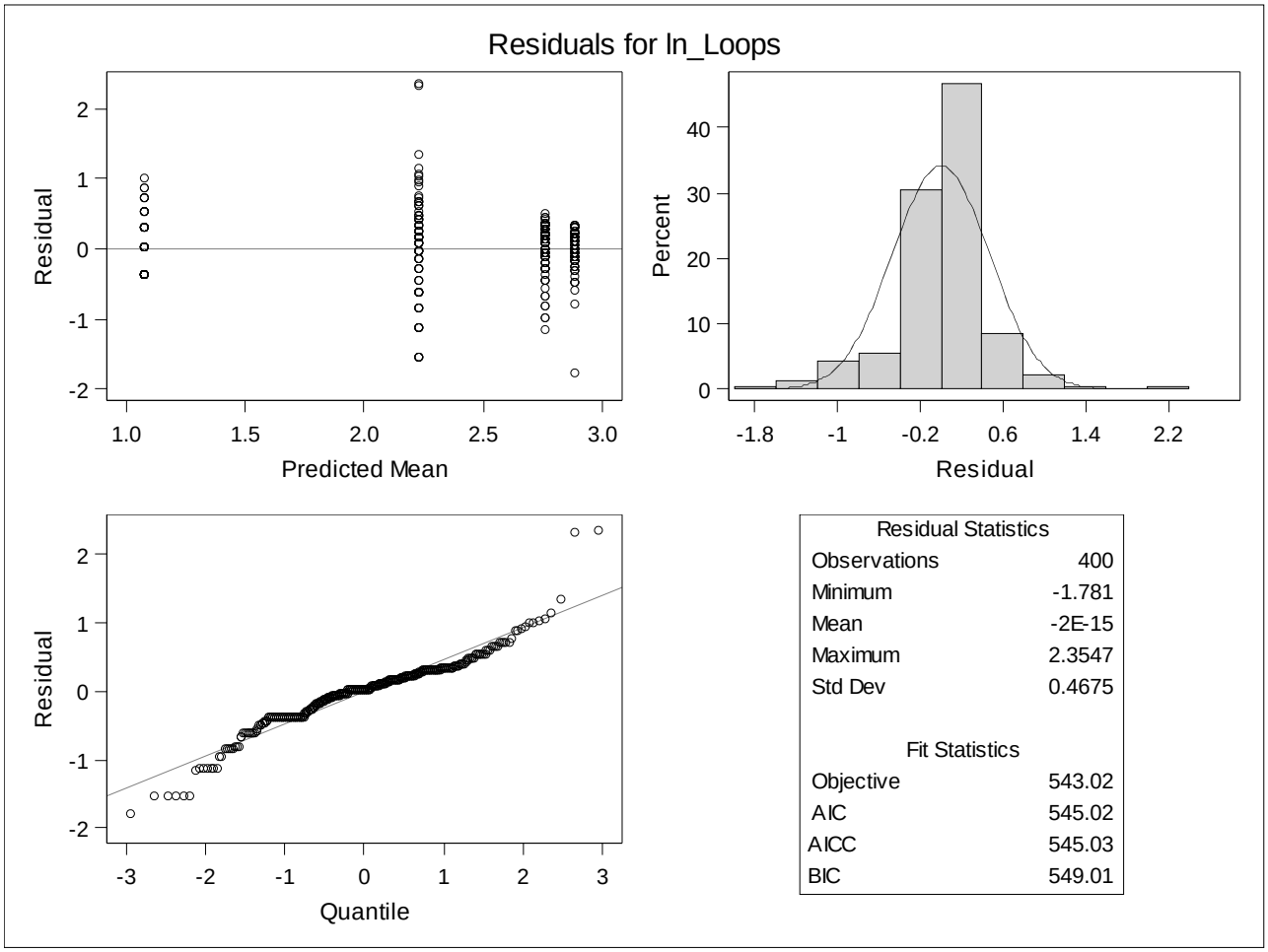
Effect	L	FB	L	FB	Adj Lower	Adj Upper
L*FB	L=3	3x3	L=3	5x5	1.5185	1.8610
L*FB	L=3	3x3	L=4	3x3	-0.2891	0.05331
L*FB	L=3	3x3	L=4	5x5	0.3603	0.7027
L*FB	L=3	5x5	L=4	3x3	-1.9789	-1.6365

Integer: Natural Log Loops Full Model

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	5x5	-1.3295	-0.9870
*L*FB*	L=4	3x3	L=4	5x5	0.4782	0.8206



Integer: Natural Log Loops Full Model

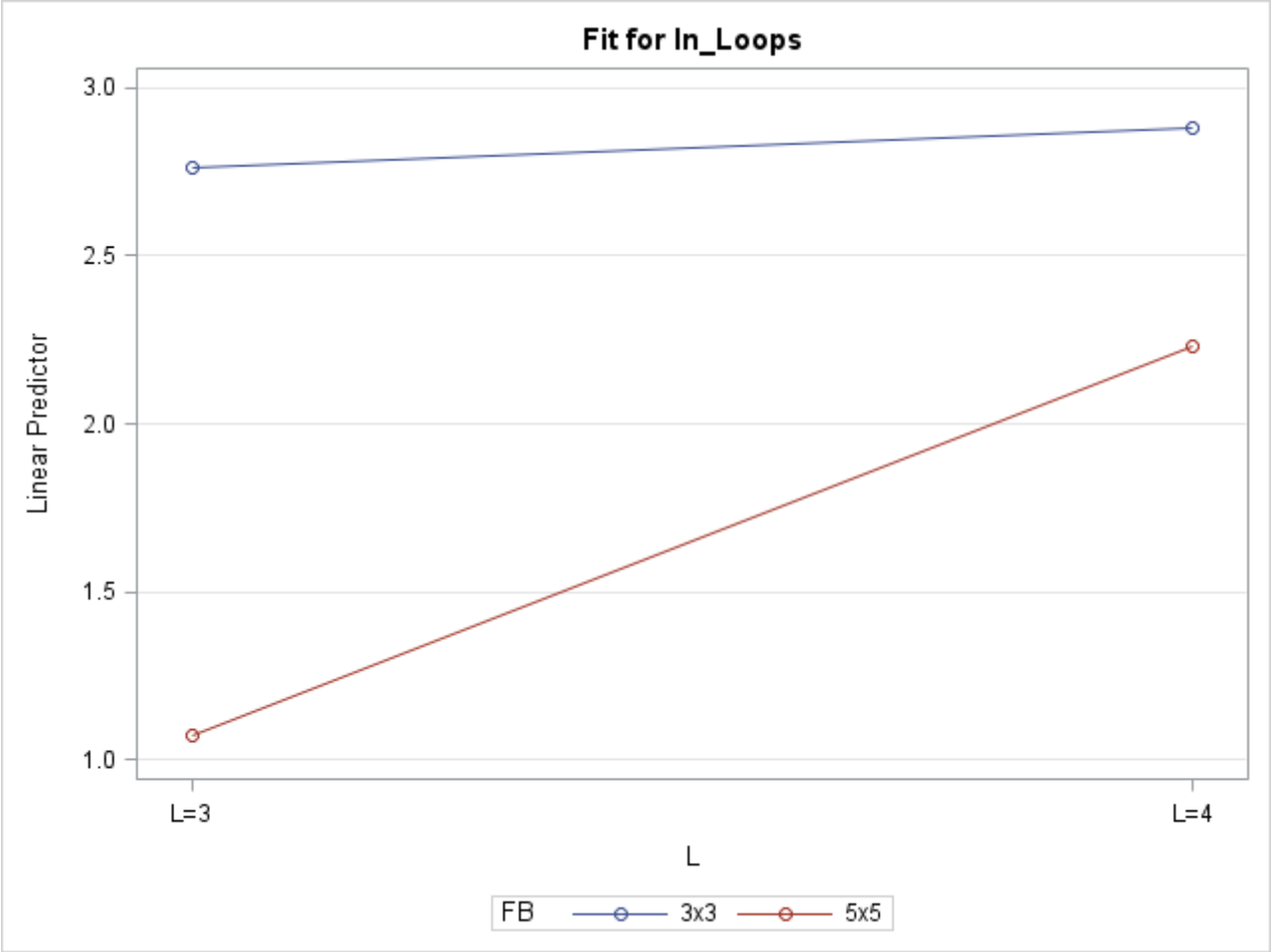
The PLM Procedure

Store Information	
Item Store	WORK.INTEGERLOOPSPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:04:25
Response Variable	ln_Loops
Distribution	Normal
Class Variables	L FB
Model Effects	Intercept L FB L*FB

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Integer: Natural Log Loops Full Model

The PLM Procedure



Floating Point: Natural Log Loops Full Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	ln_Loops
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	0.1469

Floating Point: Natural Log Loops Full Model

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	382.6
AIC (Smaller is Better)	384.6
AICC (Smaller is Better)	384.6
BIC (Smaller is Better)	388.6

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
L	1	396	20.66	<.0001
FB	1	396	23.23	<.0001
L*FB	1	396	26.89	<.0001

Least Squares Means

Effect	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
L*FB	L=3	3x3	1.9357	0.03832	396	50.51	<.0001	0.05	1.8604	2.0111
L*FB	L=3	5x5	2.3192	0.03832	396	60.51	<.0001	0.05	2.2438	2.3945
L*FB	L=4	3x3	1.9603	0.03832	396	51.15	<.0001	0.05	1.8849	2.0356
L*FB	L=4	5x5	1.9462	0.03832	396	50.78	<.0001	0.05	1.8709	2.0216

Differences of Least Squares Means

Effect	L	FB	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P	Alpha	Lower	Upper
L*FB	L=3	3x3	L=3	5x5	-0.3834	0.05420	396	-7.07	<.0001	Tukey	<.0001	0.05	-0.4900	-0.2769
L*FB	L=3	3x3	L=4	3x3	-0.02453	0.05420	396	-0.45	0.6511	Tukey	0.9691	0.05	-0.1311	0.08203
L*FB	L=3	3x3	L=4	5x5	-0.01049	0.05420	396	-0.19	0.8467	Tukey	0.9974	0.05	-0.1170	0.09607
L*FB	L=3	5x5	L=4	3x3	0.3589	0.05420	396	6.62	<.0001	Tukey	<.0001	0.05	0.2523	0.4655
L*FB	L=3	5x5	L=4	5x5	0.3729	0.05420	396	6.88	<.0001	Tukey	<.0001	0.05	0.2664	0.4795
L*FB	L=4	3x3	L=4	5x5	0.01404	0.05420	396	0.26	0.7957	Tukey	0.9939	0.05	-0.09251	0.1206

Differences of Least Squares Means

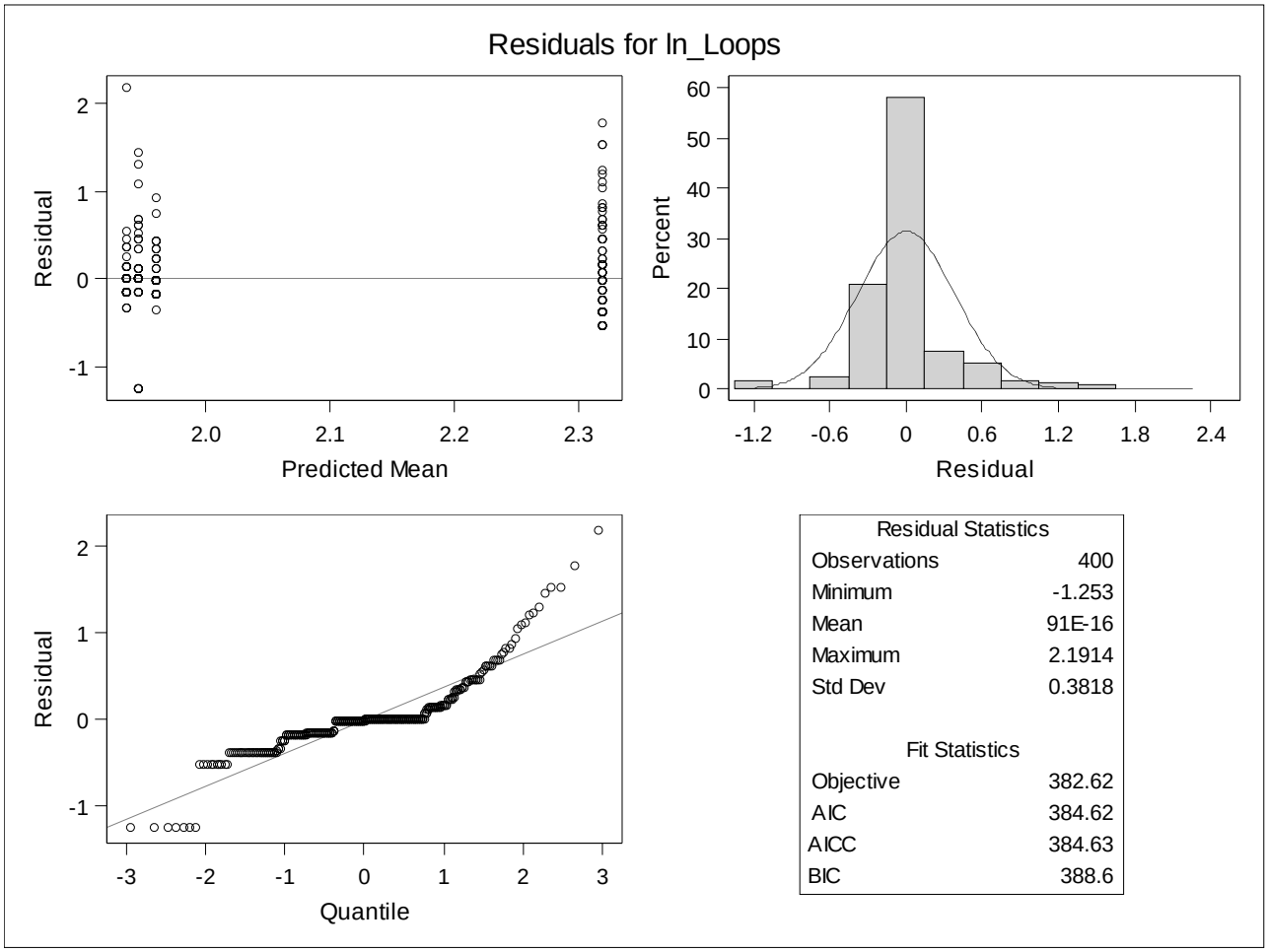
Effect	L	FB	L	FB	Adj Lower	Adj Upper
L*FB	L=3	3x3	L=3	5x5	-0.5233	-0.2436
L*FB	L=3	3x3	L=4	3x3	-0.1644	0.1153
L*FB	L=3	3x3	L=4	5x5	-0.1503	0.1293
L*FB	L=3	5x5	L=4	3x3	0.2191	0.4987

Floating Point: Natural Log Loops Full Model

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	5x5	0.2331	0.5128
*L*FB*	L=4	3x3	L=4	5x5	-0.1258	0.1539

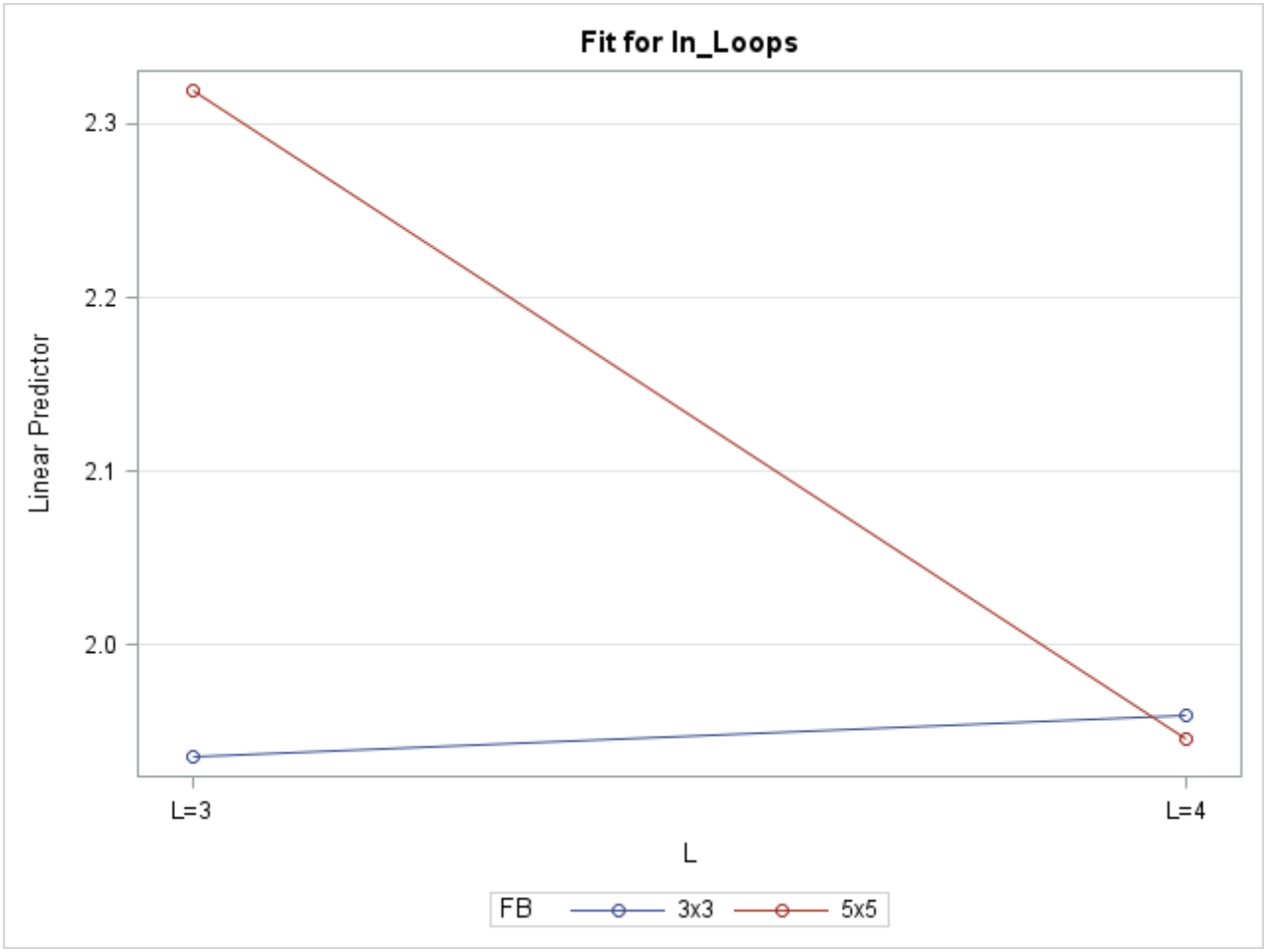


Floating Point: Natural Log Loops Full Model

The PLM Procedure

Store Information	
Item Store	WORK.FLOATINGLOOPSPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:04:30
Response Variable	ln_Loops
Distribution	Normal
Class Variables	L FB
Model Effects	Intercept L FB L*FB

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5



B.3 *PSNR*

A three-way ANOVA was performed for the metric $PSNR$. Model assumptions were satisfied, and no transformation was required. The three-way interaction $Format * L * FB$ was not significant with $p = 0.5291$, and a follow-up model with only two-way interactions was performed.

The interaction $L * FB$ was significant with $p = 0.0331$. The mean of $PSNR$ for combination $L = 3$ and $FB = 3 \times 3$ was significantly smaller than the means for all other combinations. No other pairings were significantly different.

The interaction $Format * L$ was significant with $p = 0.0280$. The mean of $PSNR$ for the combination of integer format with $L = 4$ was significantly larger than the mean for the combination of integer format with $L = 3$. No other pairings were significantly different.

The interaction $Format * FB$ was not significant, $p = 0.8778$.

PSNR Full Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

PSNR Full Model

The GLM Procedure

Dependent Variable: PSNR PSNR

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	1787.27192	255.32456	6.22	<.0001
Error	792	32488.73745	41.02113		
Corrected Total	799	34276.00937			

R-Square	Coeff Var	Root MSE	PSNR Mean
0.052144	25.95938	6.404774	24.67230

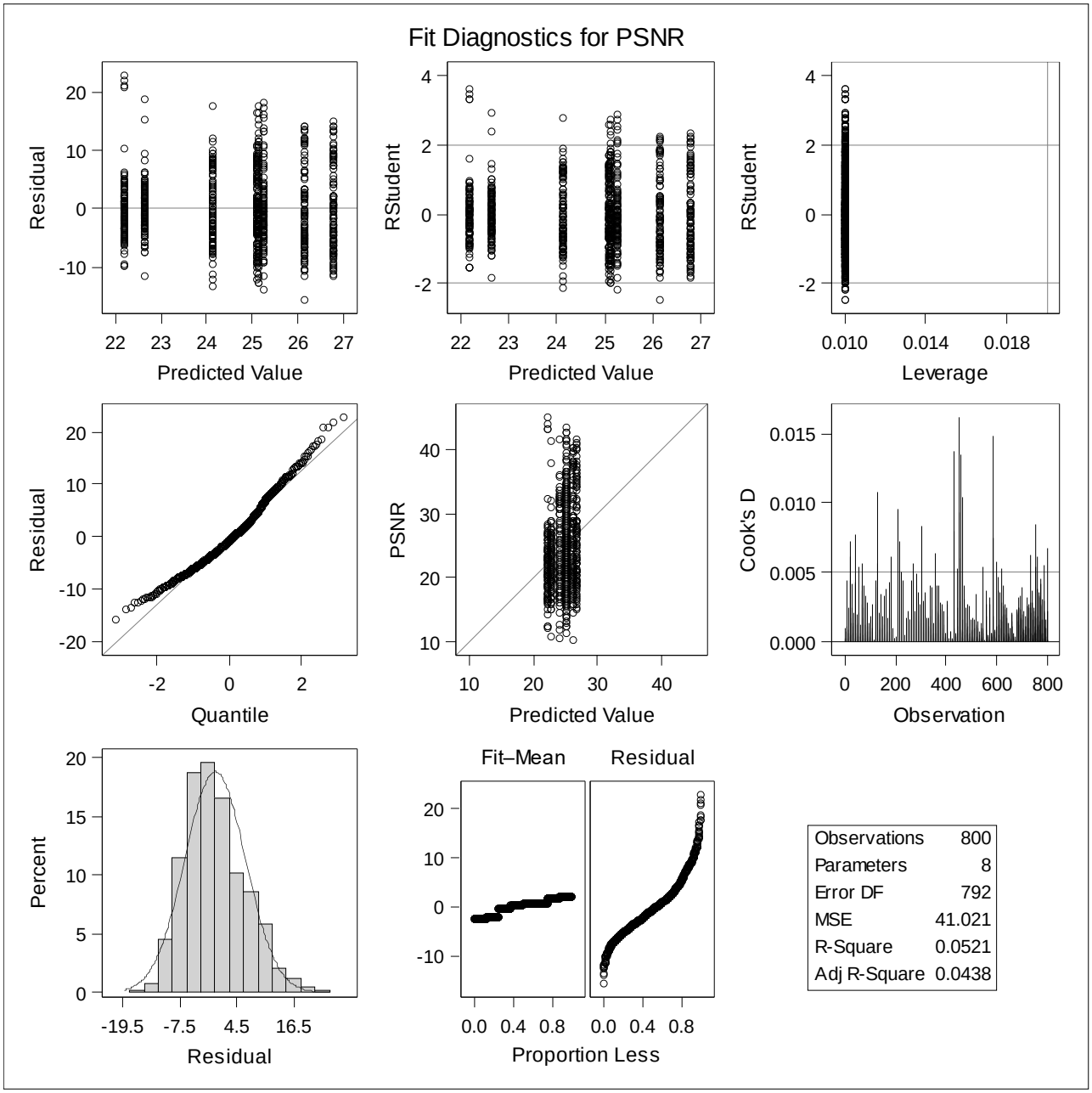
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	18.910467	18.910467	0.46	0.4974
L	1	298.080437	298.080437	7.27	0.0072
Format*L	1	198.436470	198.436470	4.84	0.0281
FB	1	1068.081877	1068.081877	26.04	<.0001
Format*FB	1	0.970429	0.970429	0.02	0.8778
L*FB	1	186.530328	186.530328	4.55	0.0333
Format*L*FB	1	16.261911	16.261911	0.40	0.5291

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	18.910467	18.910467	0.46	0.4974
L	1	298.080437	298.080437	7.27	0.0072
Format*L	1	198.436470	198.436470	4.84	0.0281
FB	1	1068.081877	1068.081877	26.04	<.0001
Format*FB	1	0.970429	0.970429	0.02	0.8778
L*FB	1	186.530328	186.530328	4.55	0.0333
Format*L*FB	1	16.261911	16.261911	0.40	0.5291

PSNR Full Model

The GLM Procedure

Dependent Variable: PSNR PSNR



PSNR Two-way Interaction Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

PSNR Two-way Interaction Model

The GLM Procedure

Dependent Variable: PSNR PSNR

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	1771.01001	295.16833	7.20	<.0001
Error	793	32504.99936	40.98991		
Corrected Total	799	34276.00937			

R-Square	Coeff Var	Root MSE	PSNR Mean
0.051669	25.94949	6.402336	24.67230

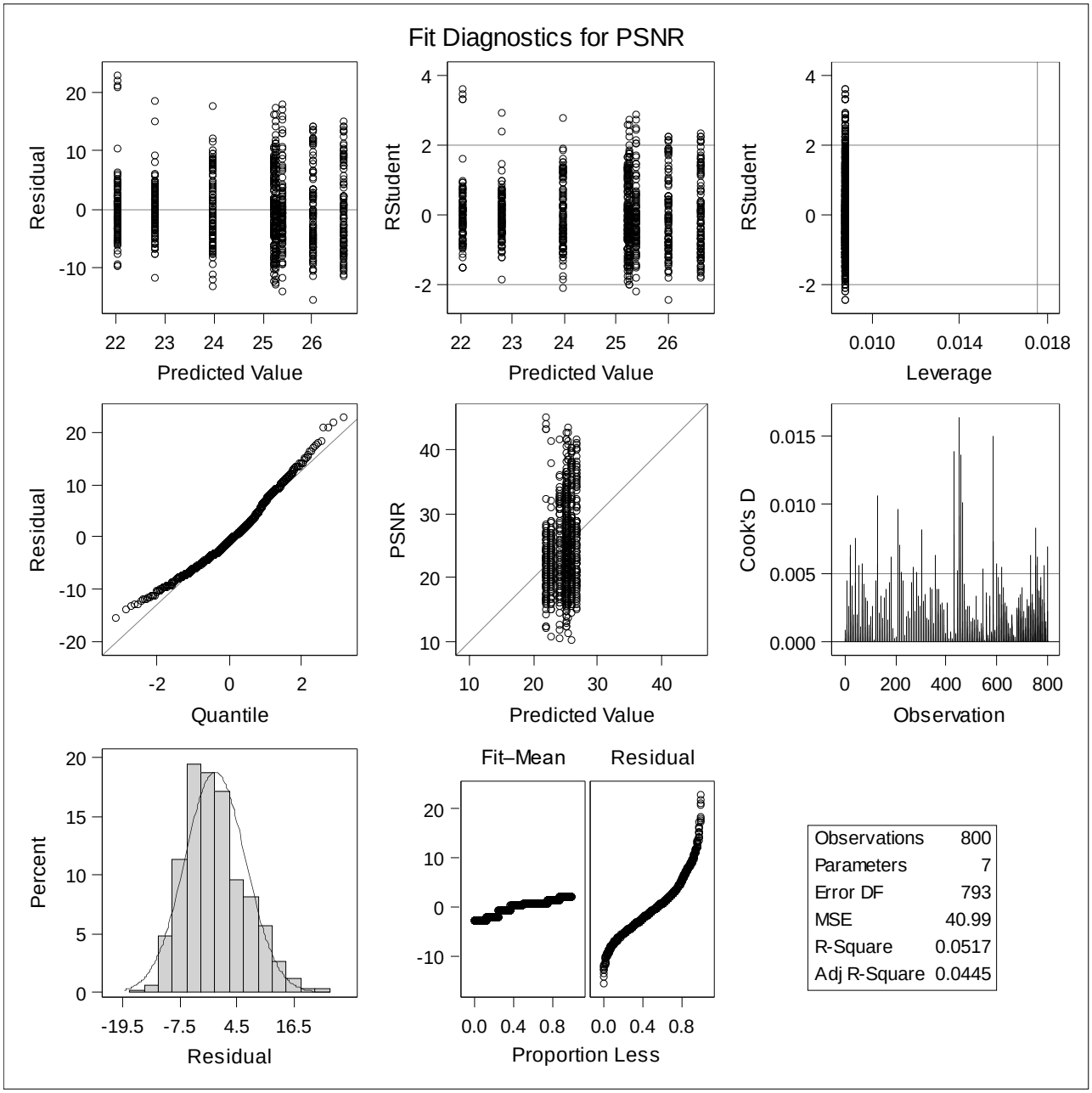
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	18.910467	18.910467	0.46	0.4972
L	1	298.080437	298.080437	7.27	0.0072
Format*L	1	198.436470	198.436470	4.84	0.0281
FB	1	1068.081877	1068.081877	26.06	<.0001
Format*FB	1	0.970429	0.970429	0.02	0.8778
L*FB	1	186.530328	186.530328	4.55	0.0332

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	18.910467	18.910467	0.46	0.4972
L	1	298.080437	298.080437	7.27	0.0072
Format*L	1	198.436470	198.436470	4.84	0.0281
FB	1	1068.081877	1068.081877	26.06	<.0001
Format*FB	1	0.970429	0.970429	0.02	0.8778
L*FB	1	186.530328	186.530328	4.55	0.0332

PSNR Two-way Interaction Model

The GLM Procedure

Dependent Variable: PSNR PSNR



PSNR Reduced Two-way Interaction Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	PSNR
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	15
Columns in Z	0
Subjects	1
Max Obs per Subject	800

Number of Observations	
Number of Observations Read	800
Number of Observations Used	800
Number of Observations Not Used	0

Covariance Parameter Estimates

Cov Parm	*Estimate*
Residual	40.9395

PSNR Reduced Two-way Interaction Model

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	5231.1
AIC (Smaller is Better)	5233.1
AICC (Smaller is Better)	5233.1
BIC (Smaller is Better)	5237.8

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Format	1	794	0.46	0.4969
L	1	794	7.28	0.0071
FB	1	794	26.09	<.0001
L*FB	1	794	4.56	0.0331
Format*L	1	794	4.85	0.0280

Least Squares Means

Effect	Format	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
L*FB		L=3	3x3	22.4236	0.4524	794	49.56	<.0001	0.05	21.5354	23.3117
L*FB		L=3	5x5	25.7002	0.4524	794	56.80	<.0001	0.05	24.8121	26.5883
L*FB		L=4	3x3	24.6101	0.4524	794	54.39	<.0001	0.05	23.7220	25.4982
L*FB		L=4	5x5	25.9553	0.4524	794	57.37	<.0001	0.05	25.0672	26.8434
Format*L	Floating Point	L=3		24.4062	0.4524	794	53.94	<.0001	0.05	23.5181	25.2943
Format*L	Floating Point	L=4		24.6309	0.4524	794	54.44	<.0001	0.05	23.7428	25.5190
Format*L	Integer	L=3		23.7176	0.4524	794	52.42	<.0001	0.05	22.8295	24.6057
Format*L	Integer	L=4		25.9345	0.4524	794	57.32	<.0001	0.05	25.0464	26.8226

Differences of Least Squares Means

Effect	Format	L	FB	Format	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P	Alpha
L*FB		L=3	3x3		L=3	5x5	-3.2767	0.6398	794	-5.12	<.0001	Tukey	<.0001	0.05
L*FB		L=3	3x3		L=4	3x3	-2.1866	0.6398	794	-3.42	0.0007	Tukey	0.0037	0.05
L*FB		L=3	3x3		L=4	5x5	-3.5318	0.6398	794	-5.52	<.0001	Tukey	<.0001	0.05
L*FB		L=3	5x5		L=4	3x3	1.0901	0.6398	794	1.70	0.0888	Tukey	0.3224	0.05
L*FB		L=3	5x5		L=4	5x5	-0.2551	0.6398	794	-0.40	0.6902	Tukey	0.9785	0.05
L*FB		L=4	3x3		L=4	5x5	-1.3452	0.6398	794	-2.10	0.0358	Tukey	0.1532	0.05

PSNR Reduced Two-way Interaction Model

The Mixed Procedure

Differences of Least Squares Means

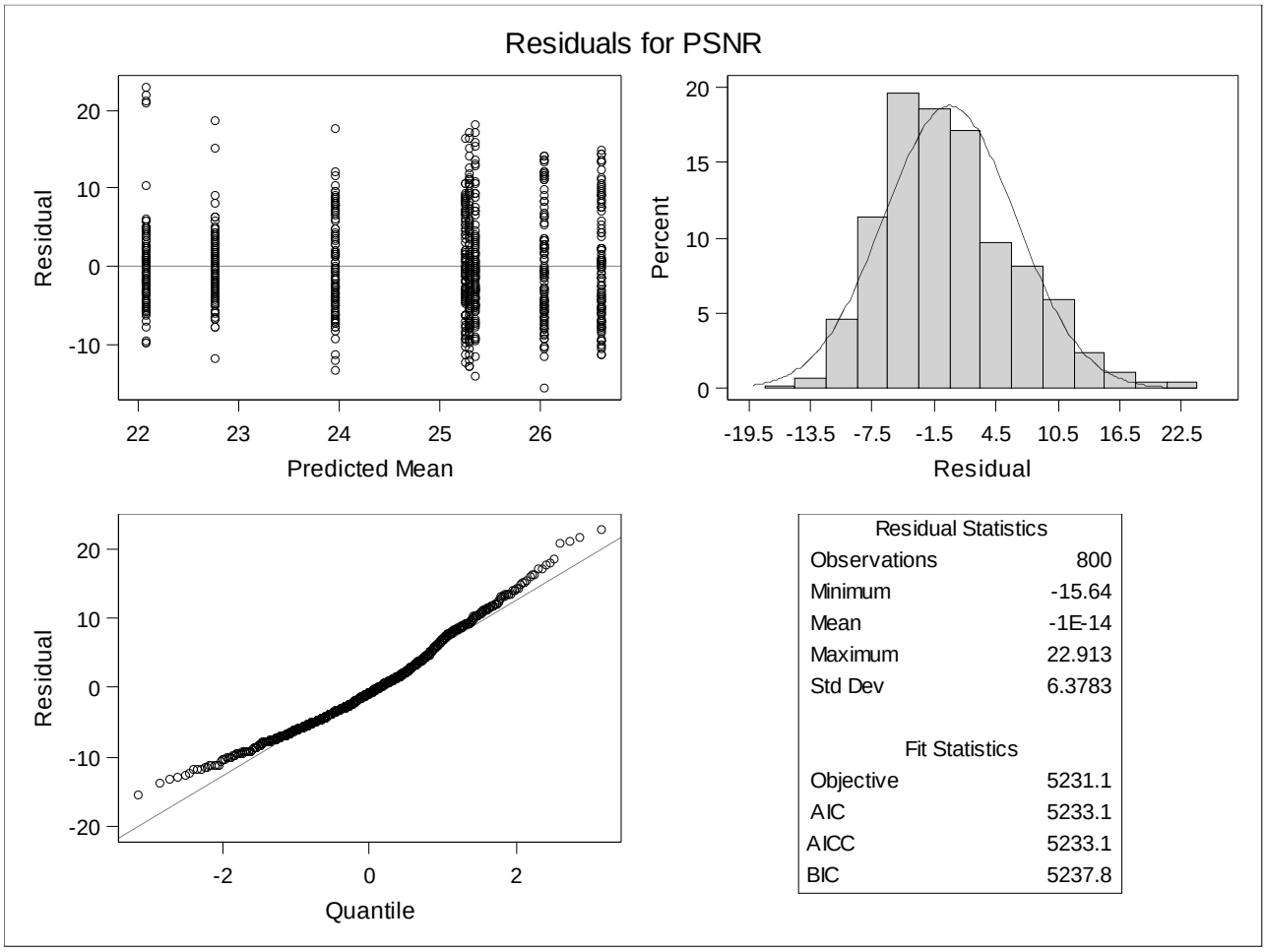
Effect	*Format*	*L*	*FB*	*Format*	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Adjustment*	*Adj P*	*Alpha*
*Format*L*	Floating Point	L=3		Floating Point	L=4		-0.2247	0.6398	794	-0.35	0.7255	Tukey	0.9851	0.05
*Format*L*	Floating Point	L=3		Integer	L=3		0.6886	0.6398	794	1.08	0.2822	Tukey	0.7042	0.05
*Format*L*	Floating Point	L=3		Integer	L=4		-1.5283	0.6398	794	-2.39	0.0171	Tukey	0.0801	0.05
*Format*L*	Floating Point	L=4		Integer	L=3		0.9133	0.6398	794	1.43	0.1538	Tukey	0.4825	0.05
*Format*L*	Floating Point	L=4		Integer	L=4		-1.3036	0.6398	794	-2.04	0.0419	Tukey	0.1752	0.05
*Format*L*	Integer	L=3		Integer	L=4		-2.2169	0.6398	794	-3.46	0.0006	Tukey	0.0031	0.05

Differences of Least Squares Means

Effect	*Format*	*L*	*FB*	*Format*	*L*	*FB*	*Lower*	*Upper*	*Adj Lower*	*Adj Upper*
*L*FB*		L=3	3x3		L=3	5x5	-4.5326	-2.0207	-4.9239	-1.6294
*L*FB*		L=3	3x3		L=4	3x3	-3.4425	-0.9306	-3.8338	-0.5393
*L*FB*		L=3	3x3		L=4	5x5	-4.7877	-2.2758	-5.1790	-1.8845
*L*FB*		L=3	5x5		L=4	3x3	-0.1659	2.3461	-0.5571	2.7373
*L*FB*		L=3	5x5		L=4	5x5	-1.5111	1.0009	-1.9023	1.3921
*L*FB*		L=4	3x3		L=4	5x5	-2.6012	-0.08922	-2.9924	0.3020
*Format*L*	Floating Point	L=3		Floating Point	L=4		-1.4807	1.0312	-1.8720	1.4225
*Format*L*	Floating Point	L=3		Integer	L=3		-0.5674	1.9446	-0.9586	2.3358
*Format*L*	Floating Point	L=3		Integer	L=4		-2.7843	-0.2723	-3.1755	0.1189
*Format*L*	Floating Point	L=4		Integer	L=3		-0.3427	2.1693	-0.7339	2.5605
*Format*L*	Floating Point	L=4		Integer	L=4		-2.5596	-0.04760	-2.9508	0.3436
*Format*L*	Integer	L=3		Integer	L=4		-3.4729	-0.9609	-3.8641	-0.5697

PSNR Reduced Two-way Interaction Model

The Mixed Procedure



Residual Statistics	
Observations	800
Minimum	-15.64
Mean	-1E-14
Maximum	22.913
Std Dev	6.3783

Fit Statistics	
Objective	5231.1
AIC	5233.1
AICC	5233.1
BIC	5237.8

PSNR Reduced Two-way Interaction Model

The PLM Procedure

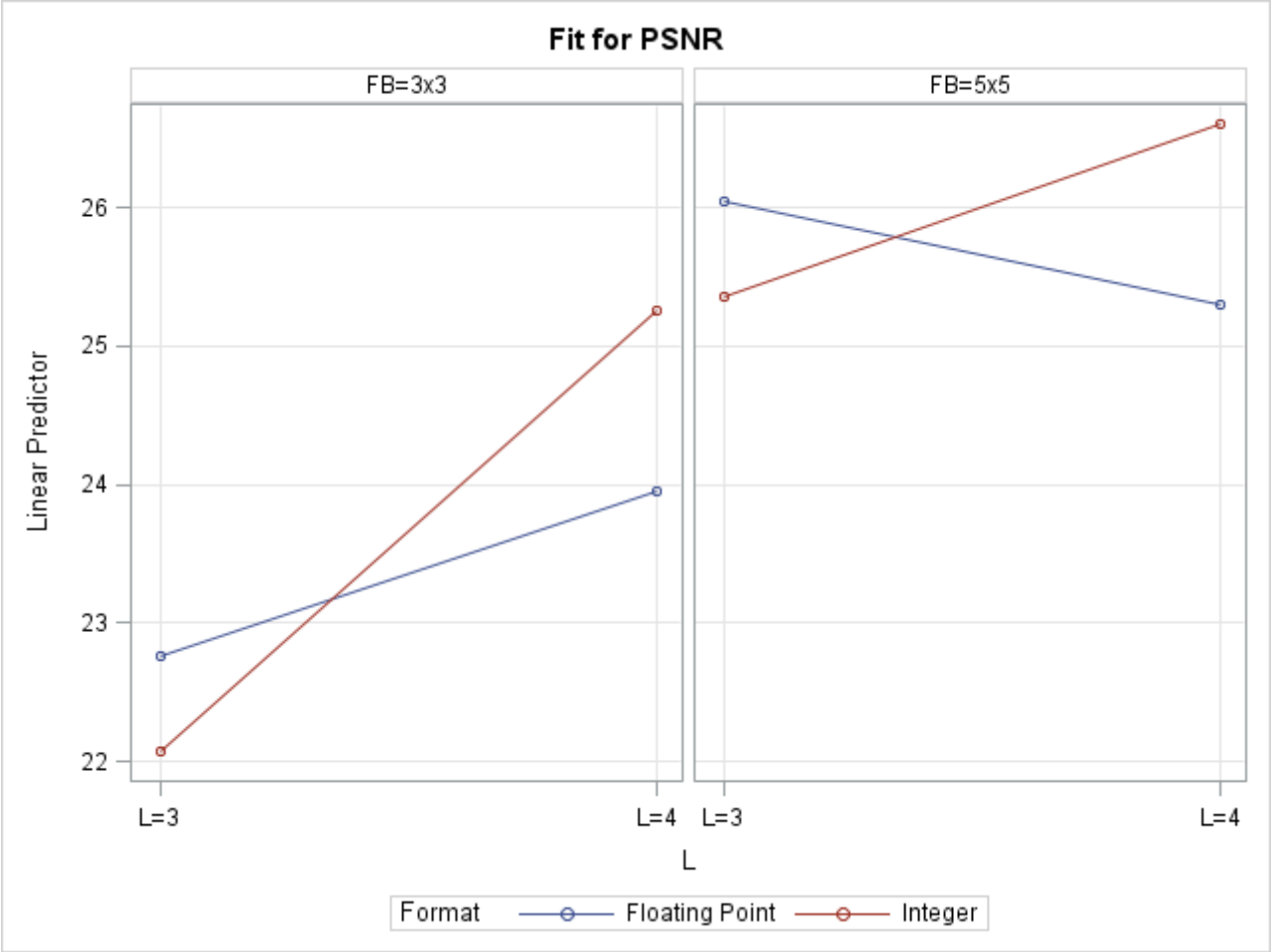
Store Information	
Item Store	WORK.PSNRPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:02:43
Response Variable	PSNR
Distribution	Normal
Class Variables	Format L FB
Model Effects	Intercept Format L FB L*FB Format*L

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

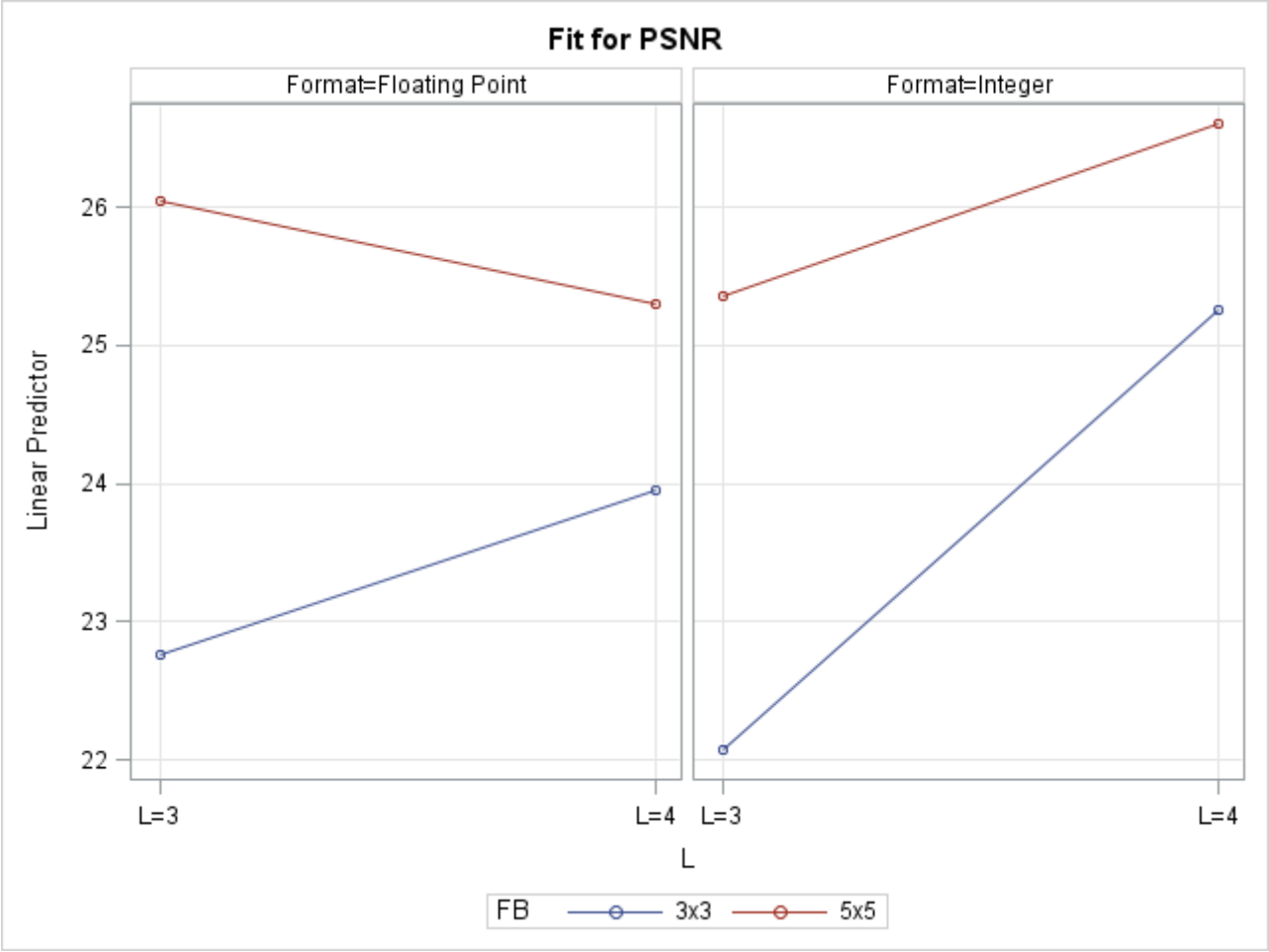
PSNR Reduced Two-way Interaction Model

The PLM Procedure



PSNR Reduced Two-way Interaction Model

The PLM Procedure



B.4 $SSIM$

A three-way ANOVA was performed for the metric $SSIM$. Model assumptions were satisfied, and no transformation was required for the metric $SSIM$. The three-way interaction $Format * L * FB$ was not significant with $p = 0.3246$, and a follow-up model with only two-way interactions was performed.

The interaction $L * FB$ was significant with $p = 0.0002$. The means of $SSIM$ for all combinations of these two factors were significantly different from each other, except for the two combinations involving $FB = 5 \times 5$.

The other two interactions were not significant, $p = 0.7628$ for $Format * FB$ and $p = 0.3170$ for $Format * L$

SSIM Full Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

SSIM Full Model

The GLM Procedure

Dependent Variable: SSIM SSIM

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	2.92749990	0.41821427	13.68	<.0001
Error	792	24.21415492	0.03057343		
Corrected Total	799	27.14165482			

R-Square	Coeff Var	Root MSE	SSIM Mean
0.107860	30.60756	0.174853	0.571273

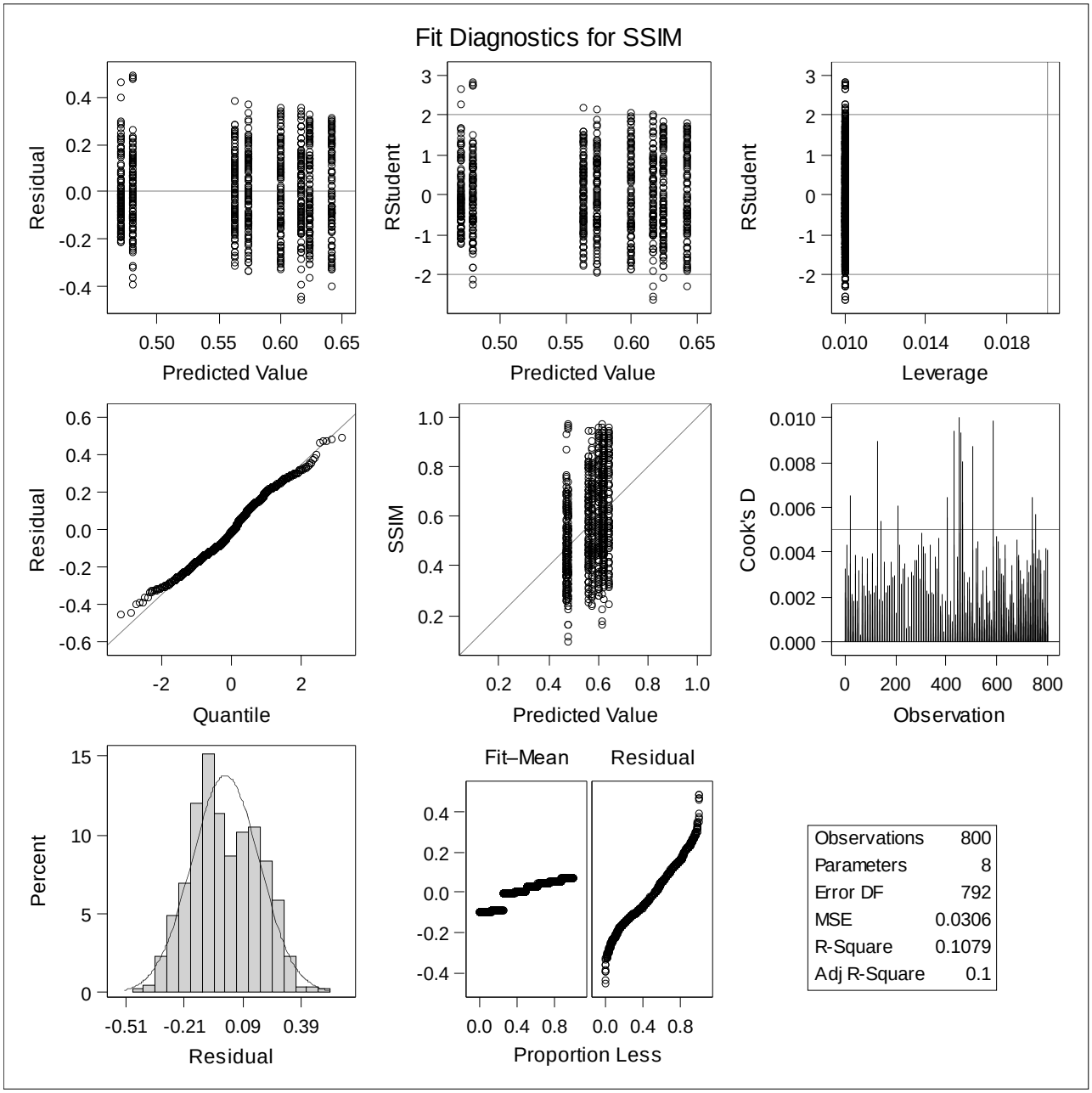
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	0.03856769	0.03856769	1.26	0.2617
L	1	0.43834248	0.43834248	14.34	0.0002
Format*L	1	0.03064809	0.03064809	1.00	0.3170
FB	1	1.96044004	1.96044004	64.12	<.0001
Format*FB	1	0.00278652	0.00278652	0.09	0.7628
L*FB	1	0.42701430	0.42701430	13.97	0.0002
Format*L*FB	1	0.02970078	0.02970078	0.97	0.3246

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	0.03856769	0.03856769	1.26	0.2617
L	1	0.43834248	0.43834248	14.34	0.0002
Format*L	1	0.03064809	0.03064809	1.00	0.3170
FB	1	1.96044004	1.96044004	64.12	<.0001
Format*FB	1	0.00278652	0.00278652	0.09	0.7628
L*FB	1	0.42701430	0.42701430	13.97	0.0002
Format*L*FB	1	0.02970078	0.02970078	0.97	0.3246

SSIM Full Model

The GLM Procedure

Dependent Variable: SSIM SSIM



SSIM Two-way Interaction Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

SSIM Two-way Interaction Model

The GLM Procedure

Dependent Variable: SSIM SSIM

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	2.89779912	0.48296652	15.80	<.0001
Error	793	24.24385570	0.03057233		
Corrected Total	799	27.14165482			

R-Square	Coeff Var	Root MSE	SSIM Mean
0.106766	30.60700	0.174849	0.571273

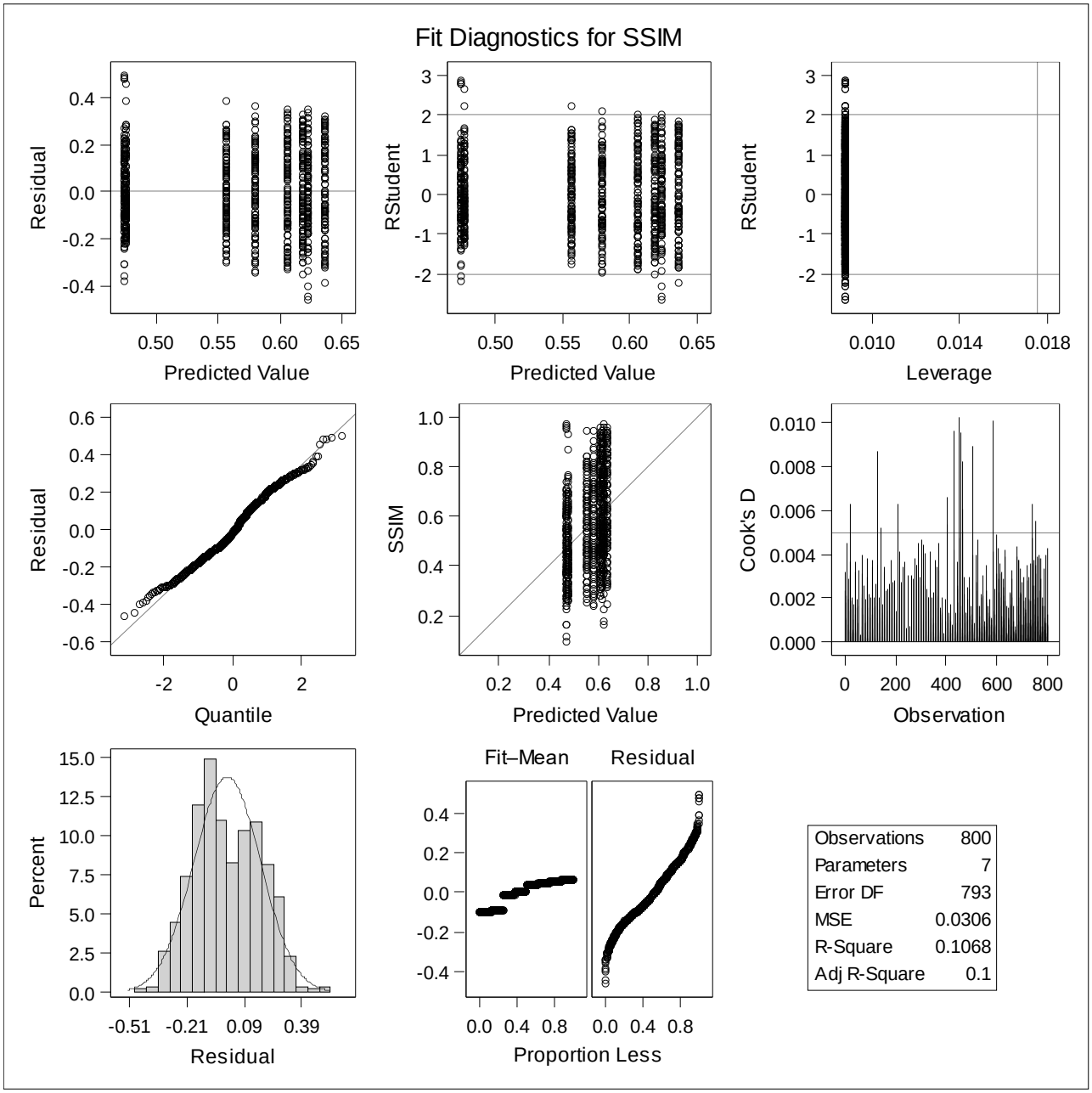
Source	DF	Type I SS	Mean Square	F Value	Pr > F
Format	1	0.03856769	0.03856769	1.26	0.2617
L	1	0.43834248	0.43834248	14.34	0.0002
Format*L	1	0.03064809	0.03064809	1.00	0.3170
FB	1	1.96044004	1.96044004	64.12	<.0001
Format*FB	1	0.00278652	0.00278652	0.09	0.7628
L*FB	1	0.42701430	0.42701430	13.97	0.0002

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Format	1	0.03856769	0.03856769	1.26	0.2617
L	1	0.43834248	0.43834248	14.34	0.0002
Format*L	1	0.03064809	0.03064809	1.00	0.3170
FB	1	1.96044004	1.96044004	64.12	<.0001
Format*FB	1	0.00278652	0.00278652	0.09	0.7628
L*FB	1	0.42701430	0.42701430	13.97	0.0002

SSIM Two-way Interaction Model

The GLM Procedure

Dependent Variable: SSIM SSIM



SSIM Reduced Two-way Interaction Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	SSIM
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	11
Columns in Z	0
Subjects	1
Max Obs per Subject	800

Number of Observations	
Number of Observations Read	800
Number of Observations Used	800
Number of Observations Not Used	0

Covariance Parameter Estimates

Cov Parm	*Estimate*
Residual	0.03054

SSIM Reduced Two-way Interaction Model

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	-491.0
AIC (Smaller is Better)	-489.0
AICC (Smaller is Better)	-489.0
BIC (Smaller is Better)	-484.3

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Format	1	795	1.26	0.2614
L	1	795	14.35	0.0002
FB	1	795	64.20	<.0001
L*FB	1	795	13.98	0.0002

Least Squares Means

Effect	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
L*FB	L=3	3x3	0.4753	0.01236	795	38.46	<.0001	0.05	0.4510	0.4995
L*FB	L=3	5x5	0.6205	0.01236	795	50.21	<.0001	0.05	0.5962	0.6447
L*FB	L=4	3x3	0.5683	0.01236	795	45.99	<.0001	0.05	0.5440	0.5925
L*FB	L=4	5x5	0.6211	0.01236	795	50.26	<.0001	0.05	0.5968	0.6453

Differences of Least Squares Means

Effect	L	FB	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P	Alpha	Lower	Upper
L*FB	L=3	3x3	L=3	5x5	-0.1452	0.01747	795	-8.31	<.0001	Tukey	<.0001	0.05	-0.1795	-0.1109
L*FB	L=3	3x3	L=4	3x3	-0.09302	0.01747	795	-5.32	<.0001	Tukey	<.0001	0.05	-0.1273	-0.05872
L*FB	L=3	3x3	L=4	5x5	-0.1458	0.01747	795	-8.34	<.0001	Tukey	<.0001	0.05	-0.1801	-0.1115
L*FB	L=3	5x5	L=4	3x3	0.05219	0.01747	795	2.99	0.0029	Tukey	0.0154	0.05	0.01789	0.08649
L*FB	L=3	5x5	L=4	5x5	-0.00061	0.01747	795	-0.03	0.9722	Tukey	1.0000	0.05	-0.03491	0.03369
L*FB	L=4	3x3	L=4	5x5	-0.05280	0.01747	795	-3.02	0.0026	Tukey	0.0138	0.05	-0.08710	-0.01850

Differences of Least Squares Means

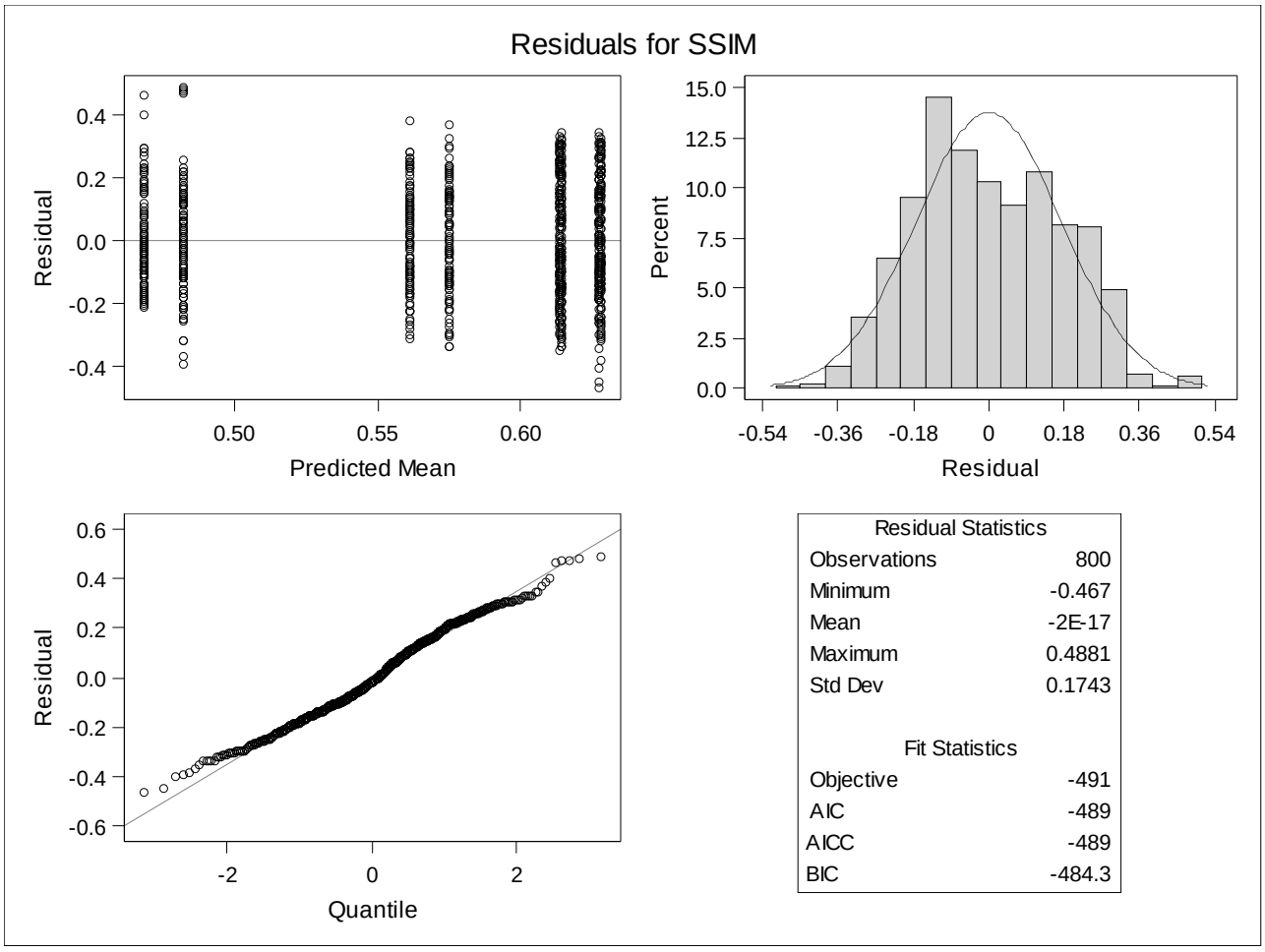
Effect	L	FB	L	FB	Adj Lower	Adj Upper
L*FB	L=3	3x3	L=3	5x5	-0.1902	-0.1002
L*FB	L=3	3x3	L=4	3x3	-0.1380	-0.04803
L*FB	L=3	3x3	L=4	5x5	-0.1908	-0.1008

SSIM Reduced Two-way Interaction Model

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	3x3	0.007202	0.09718
*L*FB*	L=3	5x5	L=4	5x5	-0.04560	0.04438
*L*FB*	L=4	3x3	L=4	5x5	-0.09779	-0.00781



SSIM Reduced Two-way Interaction Model

The PLM Procedure

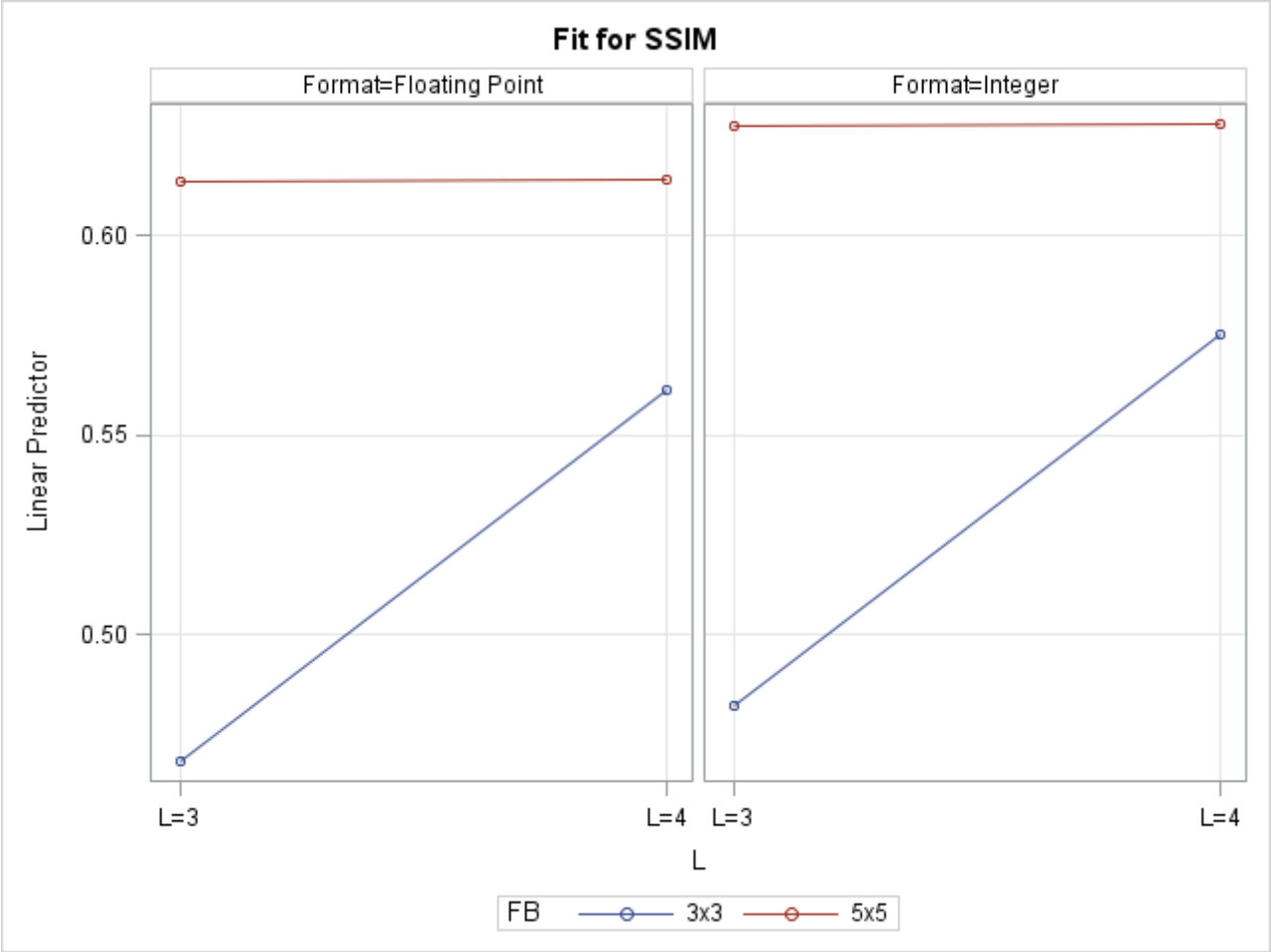
Store Information	
Item Store	WORK.SSIMPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:03:03
Response Variable	SSIM
Distribution	Normal
Class Variables	Format L FB
Model Effects	Intercept Format L FB L*FB

Class Level Information

Class	*Levels*	*Values*
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

SSIM Reduced Two-way Interaction Model

The PLM Procedure



B.5 v_0

A three-way ANOVA was performed for the metric v_0. The model assumptions requiring homogeneity of variances and normality was not satisfied. A Box-Cox procedure determined that using response variable v_0^4 should be tried, but model assumptions were still not satisfied. The data were separated into two populations by image format, and two-way ANOVAs were performed for each image format.

For the integer format model assumptions were satisfied, and the two-way interaction $L * FB$ was strongly significant with $p < 0.0001$. The means of v_0 for all combinations of these two factors were significantly different from each other, except for the two combinations involving $FB = 3 \times 3$. The 5×5 filterbank combinations resulted in the best (smallest) mean sparsity.

For the floating point format, model assumptions were not satisfied, and the Box-Cox procedure was run again. For these data the transformation v_0^{12} was the response variable that allowed model assumptions to be satisfied. For the transformed model, the two-way interaction $L * FB$ was not significant, with $p = 0.7560$, however, both main effects were significant. For L, the p-value was $p = 0.0045$ and for FB, the p-value was $p = 0.0067$. The mean sparsity for $L = 4$ was significantly smaller (better) than the mean for $L = 3$. The mean sparsity for $FB = 5 \times 5$ was significantly smaller (better) than the mean for $FB = 3 \times 3$.

V0 Full Model - Constant Variance Violation

The GLM Procedure

Class Level Information

Class	Levels	Values
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	800
Number of Observations Used	800

V0 Full Model - Constant Variance Violation

The GLM Procedure

Dependent Variable: V0 V0

Source	*DF*	*Sum of Squares*	*Mean Square*	*F Value*	*Pr > F*
Model	7	484424.3200	69203.4743	238.77	<.0001
Error	792	229548.1600	289.8335		
Corrected Total	799	713972.4800			

R-Square	*Coeff Var*	*Root MSE*	*V0 Mean*
0.678492	15.85592	17.02450	107.3700

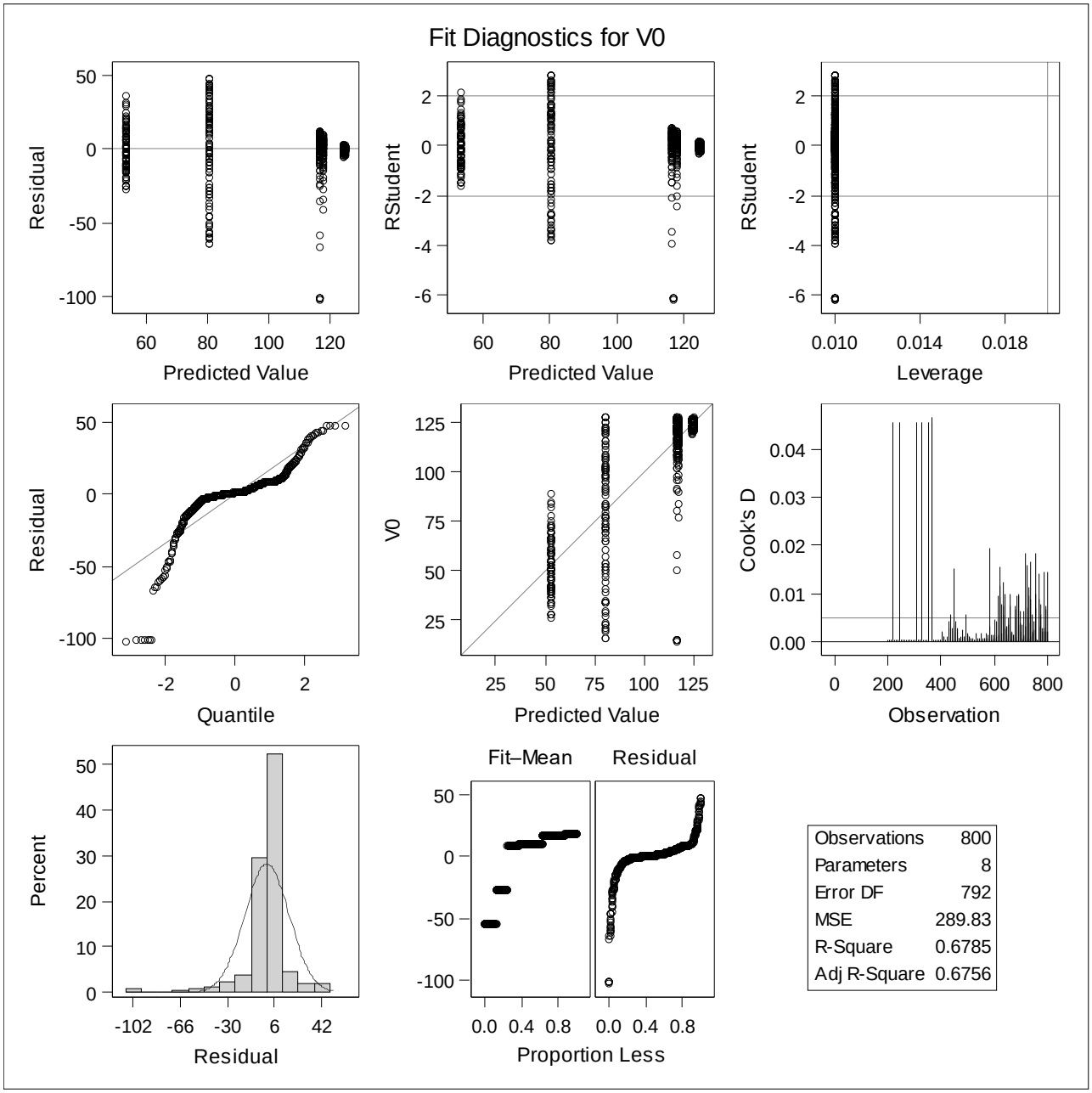
Source	*DF*	*Type I SS*	*Mean Square*	*F Value*	*Pr > F*
Format	1	188375.2200	188375.2200	649.94	<.0001
L	1	5202.0000	5202.0000	17.95	<.0001
*Format*L*	1	16891.2200	16891.2200	58.28	<.0001
FB	1	148621.5200	148621.5200	512.78	<.0001
*Format*FB*	1	107369.7800	107369.7800	370.45	<.0001
*L*FB*	1	4418.0000	4418.0000	15.24	0.0001
*Format*L*FB*	1	13546.5800	13546.5800	46.74	<.0001

Source	*DF*	*Type III SS*	*Mean Square*	*F Value*	*Pr > F*
Format	1	188375.2200	188375.2200	649.94	<.0001
L	1	5202.0000	5202.0000	17.95	<.0001
*Format*L*	1	16891.2200	16891.2200	58.28	<.0001
FB	1	148621.5200	148621.5200	512.78	<.0001
*Format*FB*	1	107369.7800	107369.7800	370.45	<.0001
*L*FB*	1	4418.0000	4418.0000	15.24	0.0001
*Format*L*FB*	1	13546.5800	13546.5800	46.74	<.0001

V0 Full Model - Constant Variance Violation

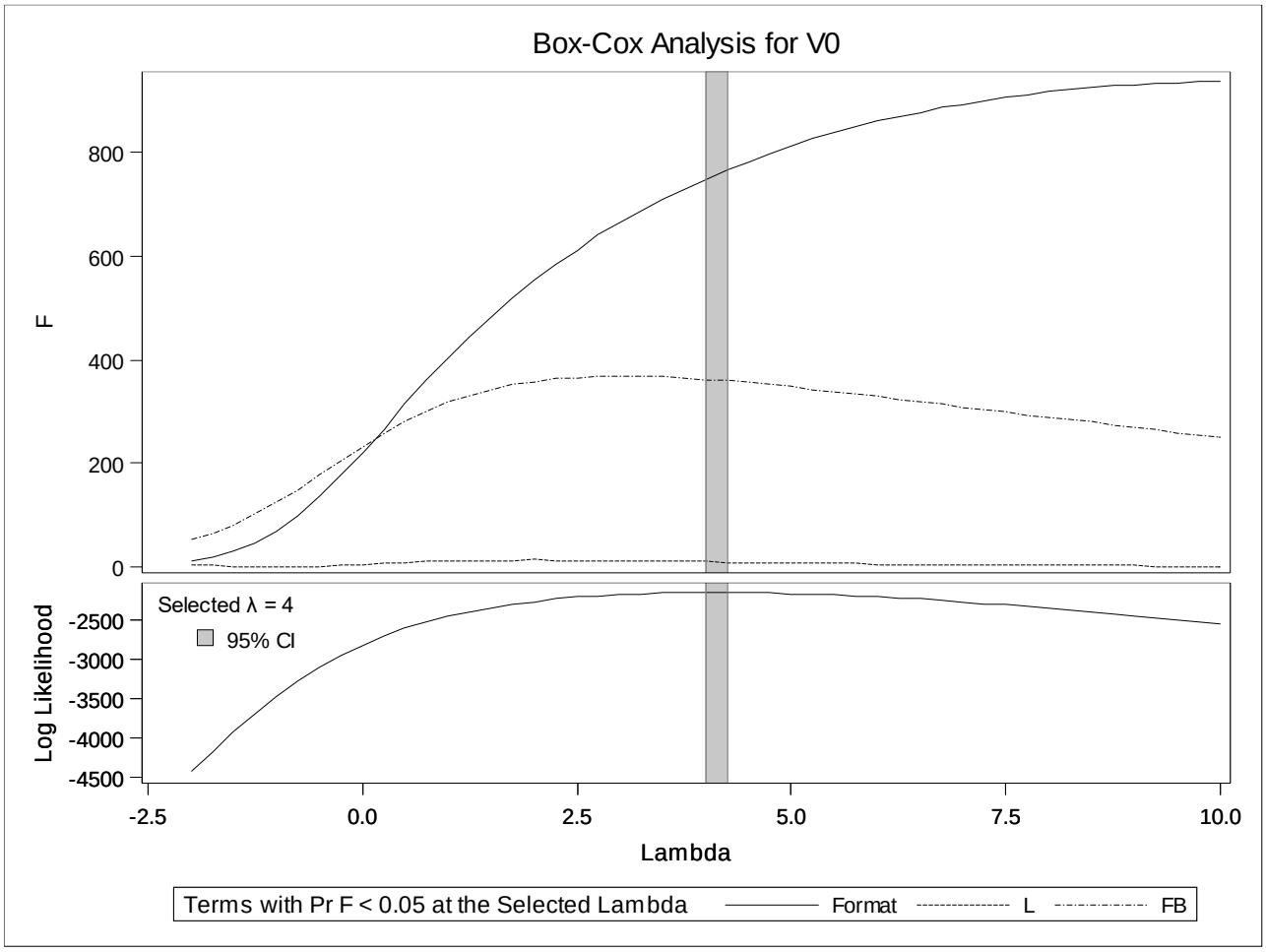
The GLM Procedure

Dependent Variable: V0 V0



V0 Full Model - Constant Variance Violation

The TRANSREG Procedure



Dependent Variable BoxCox(V0)
V0

Number of Observations Read	800
Number of Observations Used	800

The TRANSREG Procedure Hypothesis Tests for BoxCox(V0)
V0

V0 Full Model - Constant Variance Violation

The TRANSREG Procedure

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	*DF*	*Sum of Squares*	*Mean Square*	*F Value*	*Liberal p*
Model	3	2.502E17	8.341E16	374.05	>= <.0001
Error	796	1.775E17	2.23E14		
Corrected Total	799	4.277E17			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Root MSE	14932817	*R-Square*	0.5850
Dependent Mean	44880857	*Adj R-Sq*	0.5835
Coeff Var	33.27213	*Lambda*	4.0000

V0 Full Model: V0^4 Transformation - Assumptions Still Violated

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	V04
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	Levels	Values
Format	2	Floating Point Integer
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	27
Columns in Z	0
Subjects	1
Max Obs per Subject	800

Number of Observations	
Number of Observations Read	800
Number of Observations Used	800
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	Estimate
Residual	2.063E15

V0 Full Model: V0^4 Transformation - Assumptions Still Violated

The Mixed Procedure

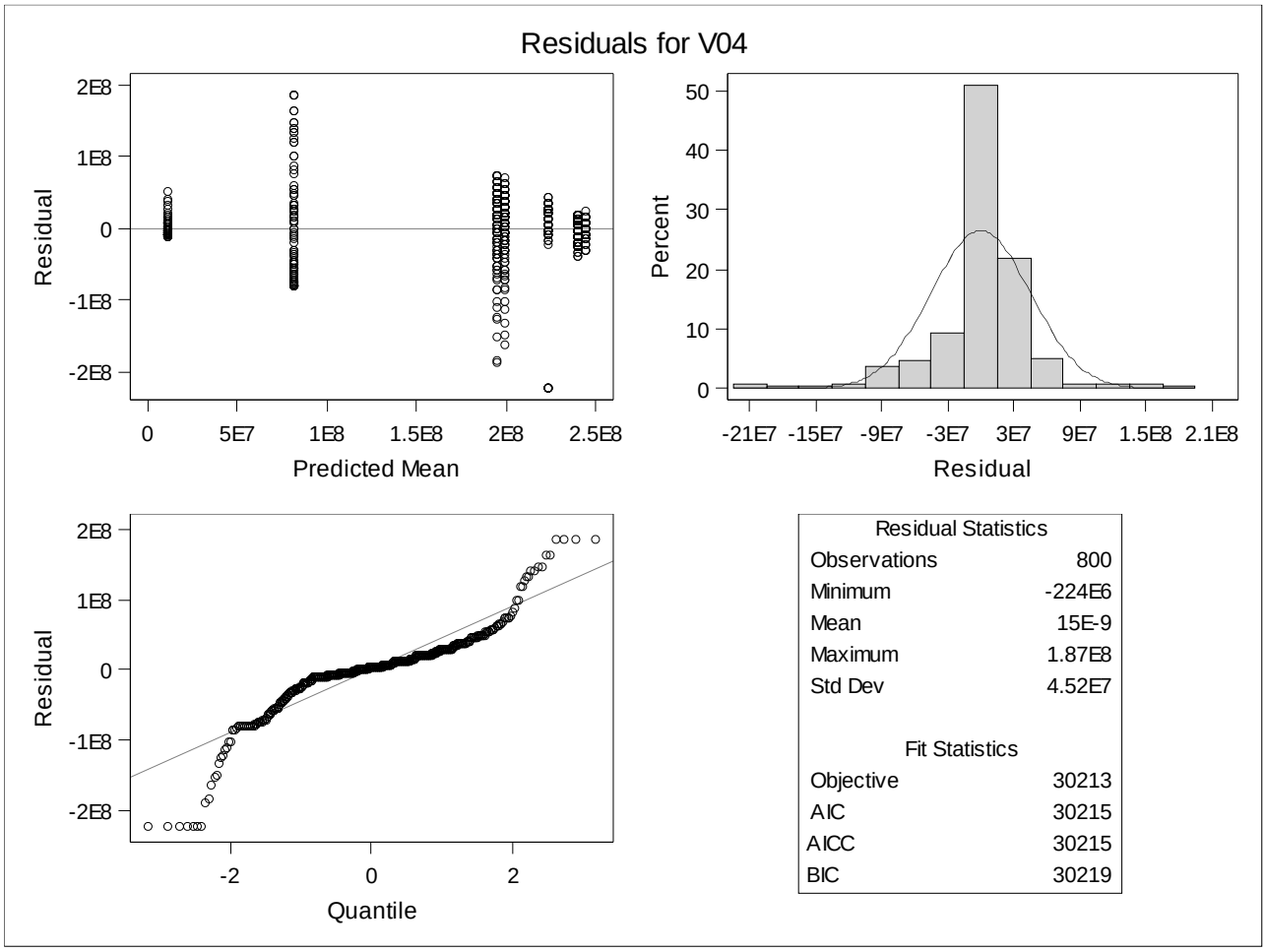
Fit Statistics	
-2 Res Log Likelihood	30212.6
AIC (Smaller is Better)	30214.6
AICC (Smaller is Better)	30214.7
BIC (Smaller is Better)	30219.3

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Format	1	792	1297.07	<.0001
L	1	792	17.06	<.0001
Format*L	1	792	53.99	<.0001
FB	1	792	626.64	<.0001
Format*FB	1	792	476.75	<.0001
L*FB	1	792	17.34	<.0001
Format*L*FB	1	792	36.62	<.0001

V0 Full Model: V0^4 Transformation - Assumptions Still Violated

The Mixed Procedure



Residual Statistics	
Observations	800
Minimum	-224E6
Mean	15E-9
Maximum	1.87E8
Std Dev	4.52E7

Fit Statistics	
Objective	30213
AIC	30215
AICC	30215
BIC	30219

Integer: V0 Full Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	V0
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	379.03

Integer: V0 Full Model

The Mixed Procedure

Fit Statistics	
-2 Res Log Likelihood	3493.5
AIC (Smaller is Better)	3495.5
AICC (Smaller is Better)	3495.5
BIC (Smaller is Better)	3499.5

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
L	1	396	53.88	<.0001
FB	1	396	670.97	<.0001
L*FB	1	396	44.11	<.0001

Least Squares Means

Effect	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
L*FB	L=3	3x3	116.56	1.9469	396	59.87	<.0001	0.05	112.73	120.39
L*FB	L=3	5x5	53.2000	1.9469	396	27.33	<.0001	0.05	49.3725	57.0275
L*FB	L=4	3x3	117.92	1.9469	396	60.57	<.0001	0.05	114.09	121.75
L*FB	L=4	5x5	80.4200	1.9469	396	41.31	<.0001	0.05	76.5925	84.2475

Differences of Least Squares Means

Effect	L	FB	L	FB	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P	Alpha	Lower	Upper
L*FB	L=3	3x3	L=3	5x5	63.3600	2.7533	396	23.01	<.0001	Tukey	<.0001	0.05	57.9471	68.7729
L*FB	L=3	3x3	L=4	3x3	-1.3600	2.7533	396	-0.49	0.6216	Tukey	0.9604	0.05	-6.7729	4.0529
L*FB	L=3	3x3	L=4	5x5	36.1400	2.7533	396	13.13	<.0001	Tukey	<.0001	0.05	30.7271	41.5529
L*FB	L=3	5x5	L=4	3x3	-64.7200	2.7533	396	-23.51	<.0001	Tukey	<.0001	0.05	-70.1329	-59.3071
L*FB	L=3	5x5	L=4	5x5	-27.2200	2.7533	396	-9.89	<.0001	Tukey	<.0001	0.05	-32.6329	-21.8071
L*FB	L=4	3x3	L=4	5x5	37.5000	2.7533	396	13.62	<.0001	Tukey	<.0001	0.05	32.0871	42.9129

Differences of Least Squares Means

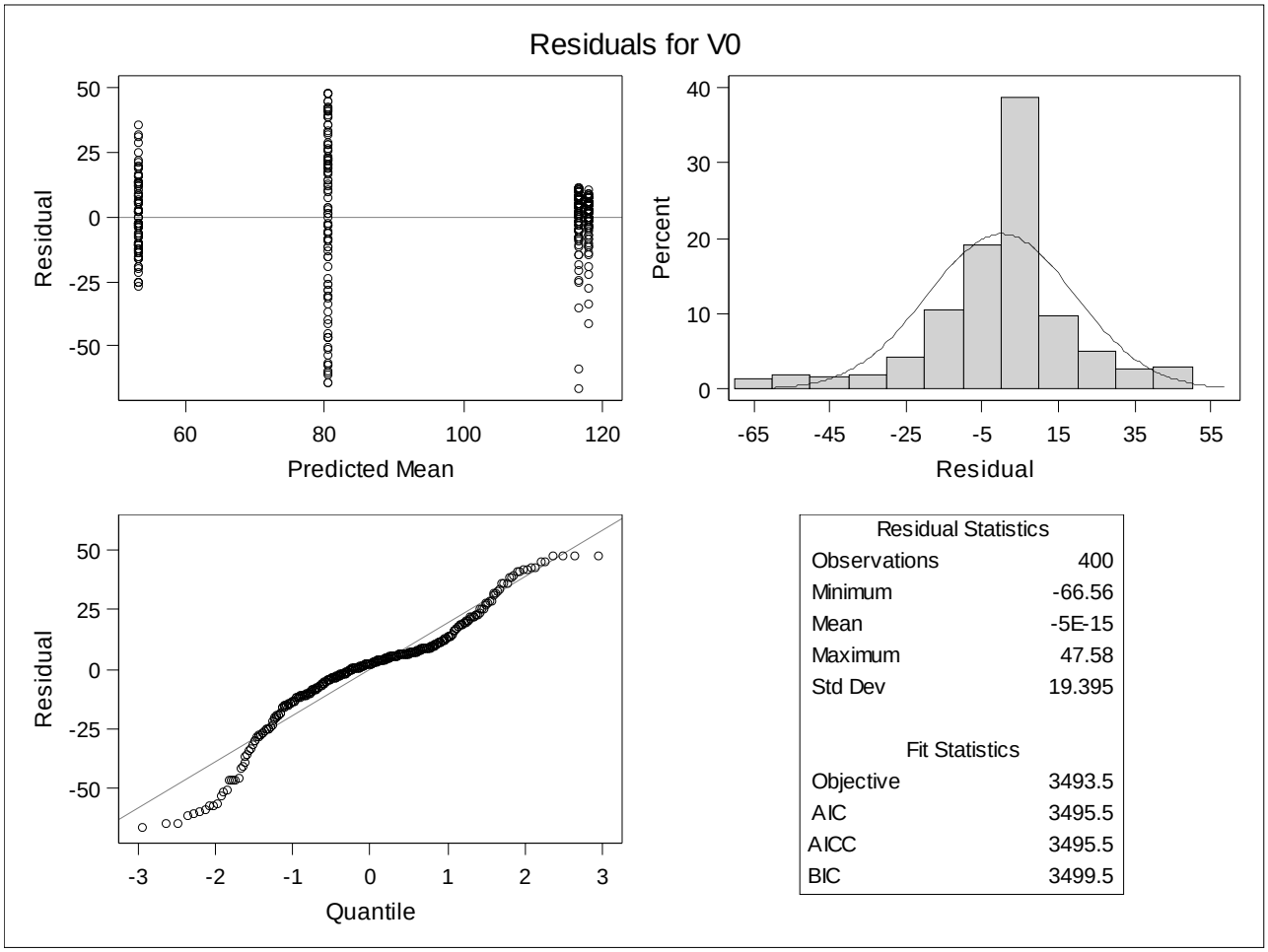
Effect	L	FB	L	FB	Adj Lower	Adj Upper
L*FB	L=3	3x3	L=3	5x5	56.2566	70.4634
L*FB	L=3	3x3	L=4	3x3	-8.4634	5.7434
L*FB	L=3	3x3	L=4	5x5	29.0366	43.2434
L*FB	L=3	5x5	L=4	3x3	-71.8234	-57.6166

Integer: V0 Full Model

The Mixed Procedure

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Adj Lower*	*Adj Upper*
*L*FB*	L=3	5x5	L=4	5x5	-34.3234	-20.1166
*L*FB*	L=4	3x3	L=4	5x5	30.3966	44.6034



Integer: V0 Full Model

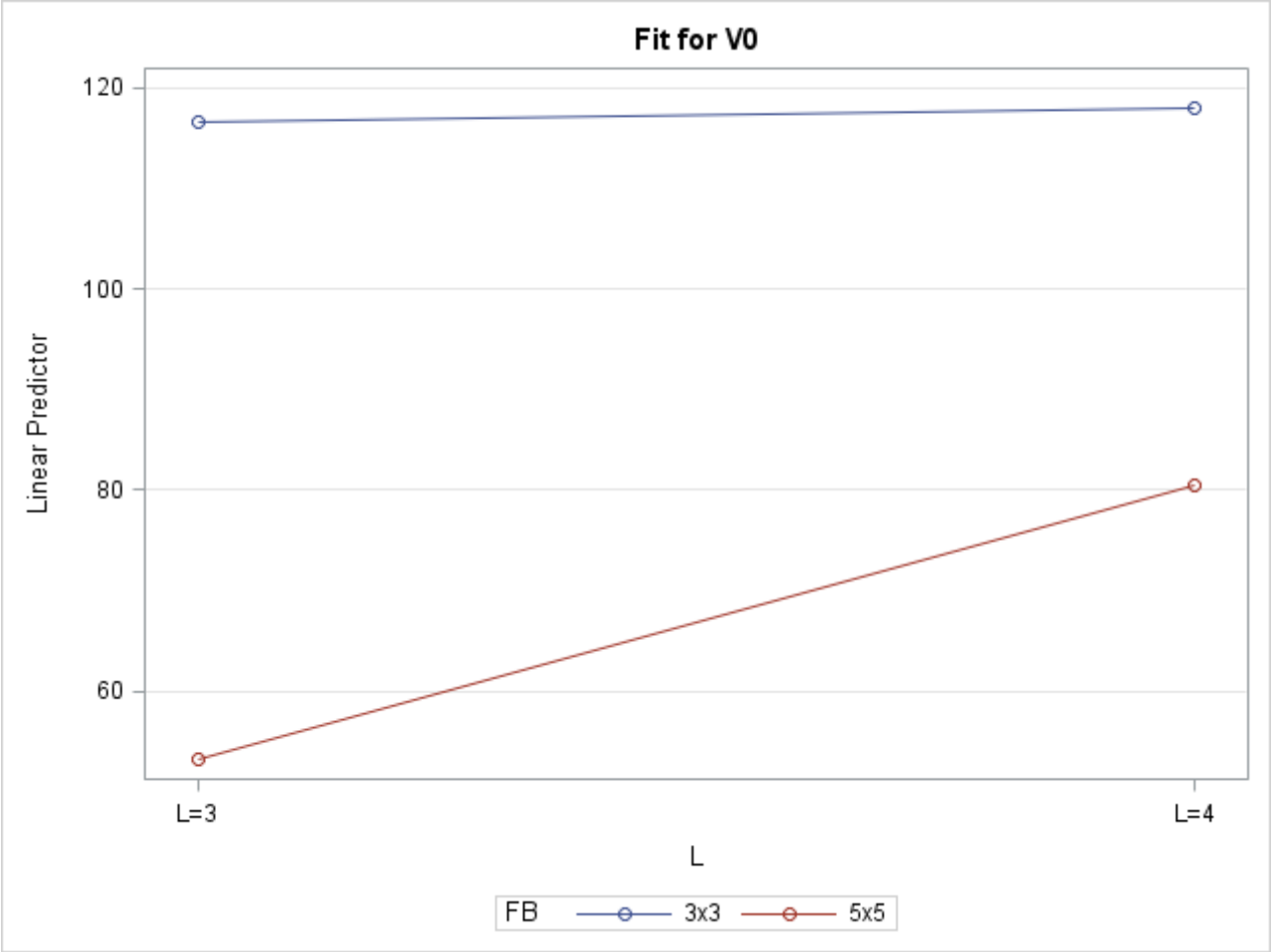
The PLM Procedure

Store Information	
Item Store	WORK.INTEGERVOPLOT
Data Set Created From	WORK.BEV
Created By	PROC MIXED
Date Created	10MAY21:13:03:22
Response Variable	V0
Distribution	Normal
Class Variables	L FB
Model Effects	Intercept L FB L*FB

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Integer: V0 Full Model

The PLM Procedure



Floating Point: V0 Full Model

The GLM Procedure

Class Level Information

Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Number of Observations Read	400
Number of Observations Used	400

Floating Point: V0 Full Model: Assumptions Violated

The GLM Procedure

Dependent Variable: V0 V0

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	4591.71000	1530.57000	7.63	<.0001
Error	396	79451.80000	200.63586		
Corrected Total	399	84043.51000			

R-Square	Coeff Var	Root MSE	V0 Mean
0.054635	11.54268	14.16460	122.7150

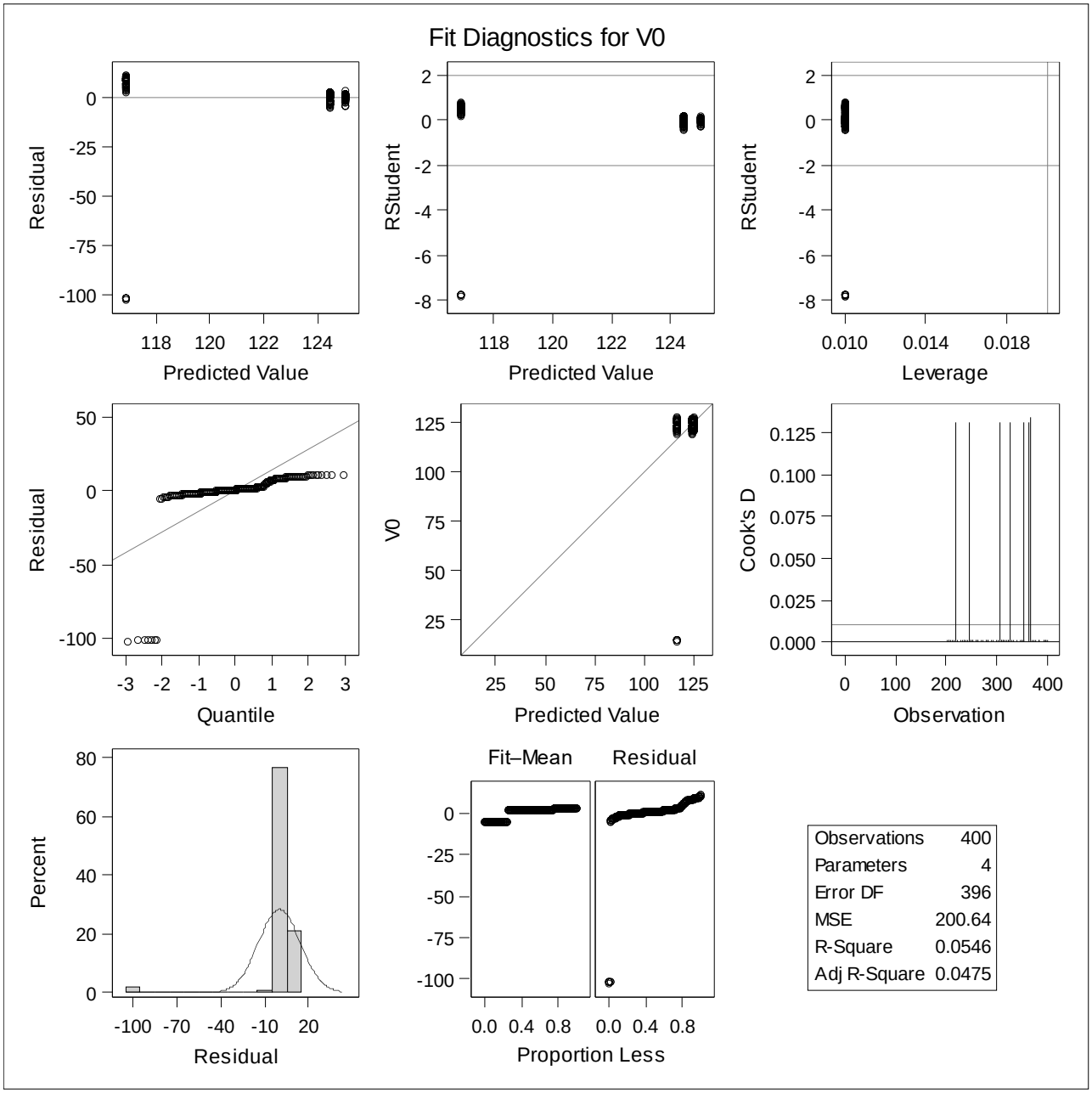
Source	DF	Type I SS	Mean Square	F Value	Pr > F
L	1	1672.810000	1672.810000	8.34	0.0041
FB	1	1672.810000	1672.810000	8.34	0.0041
L*FB	1	1246.090000	1246.090000	6.21	0.0131

Source	DF	Type III SS	Mean Square	F Value	Pr > F
L	1	1672.810000	1672.810000	8.34	0.0041
FB	1	1672.810000	1672.810000	8.34	0.0041
L*FB	1	1246.090000	1246.090000	6.21	0.0131

Floating Point: V0 Full Model: Assumptions Violated

The GLM Procedure

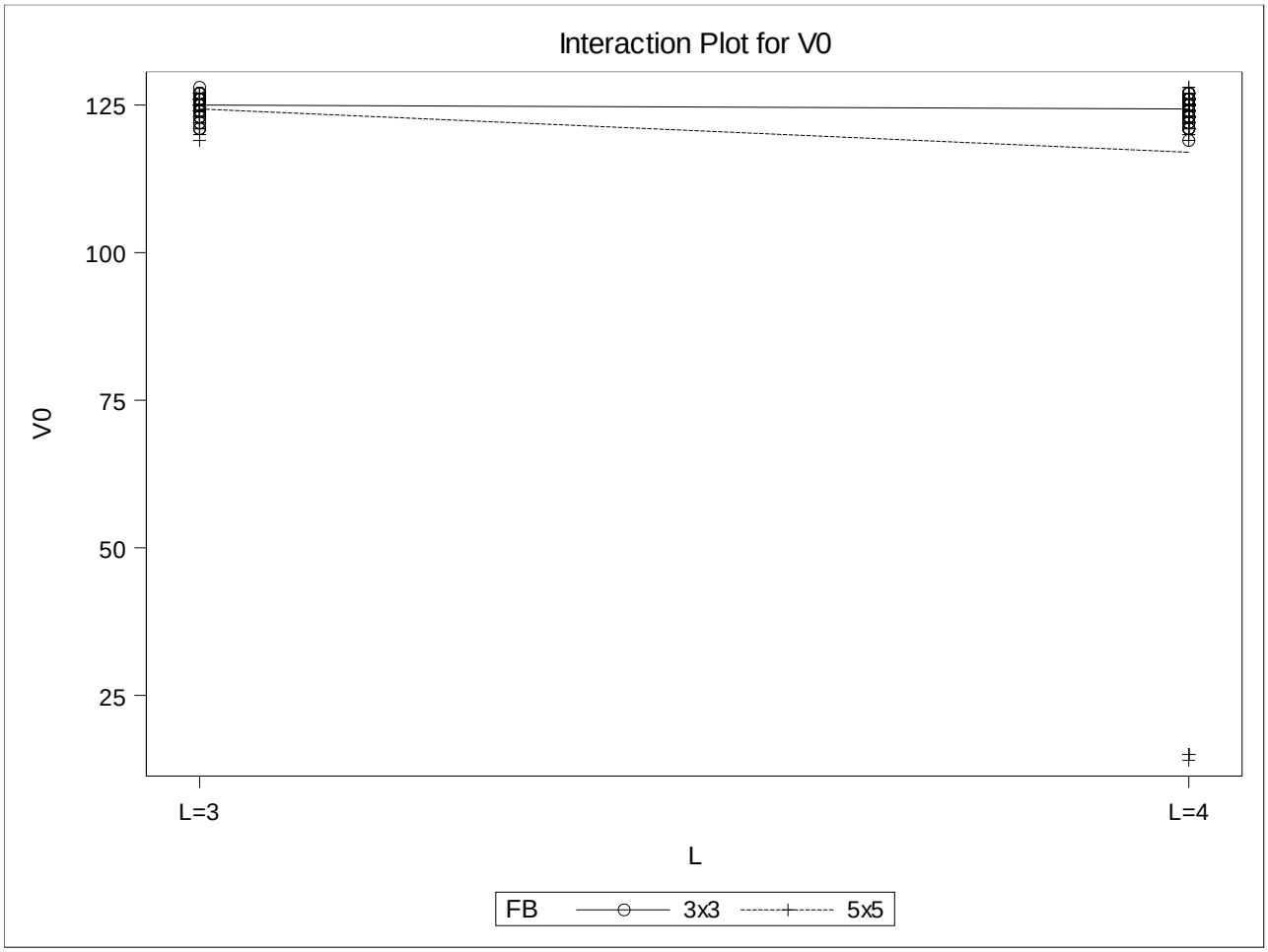
Dependent Variable: V0 V0



Floating Point: V0 Full Model: Assumptions Violated

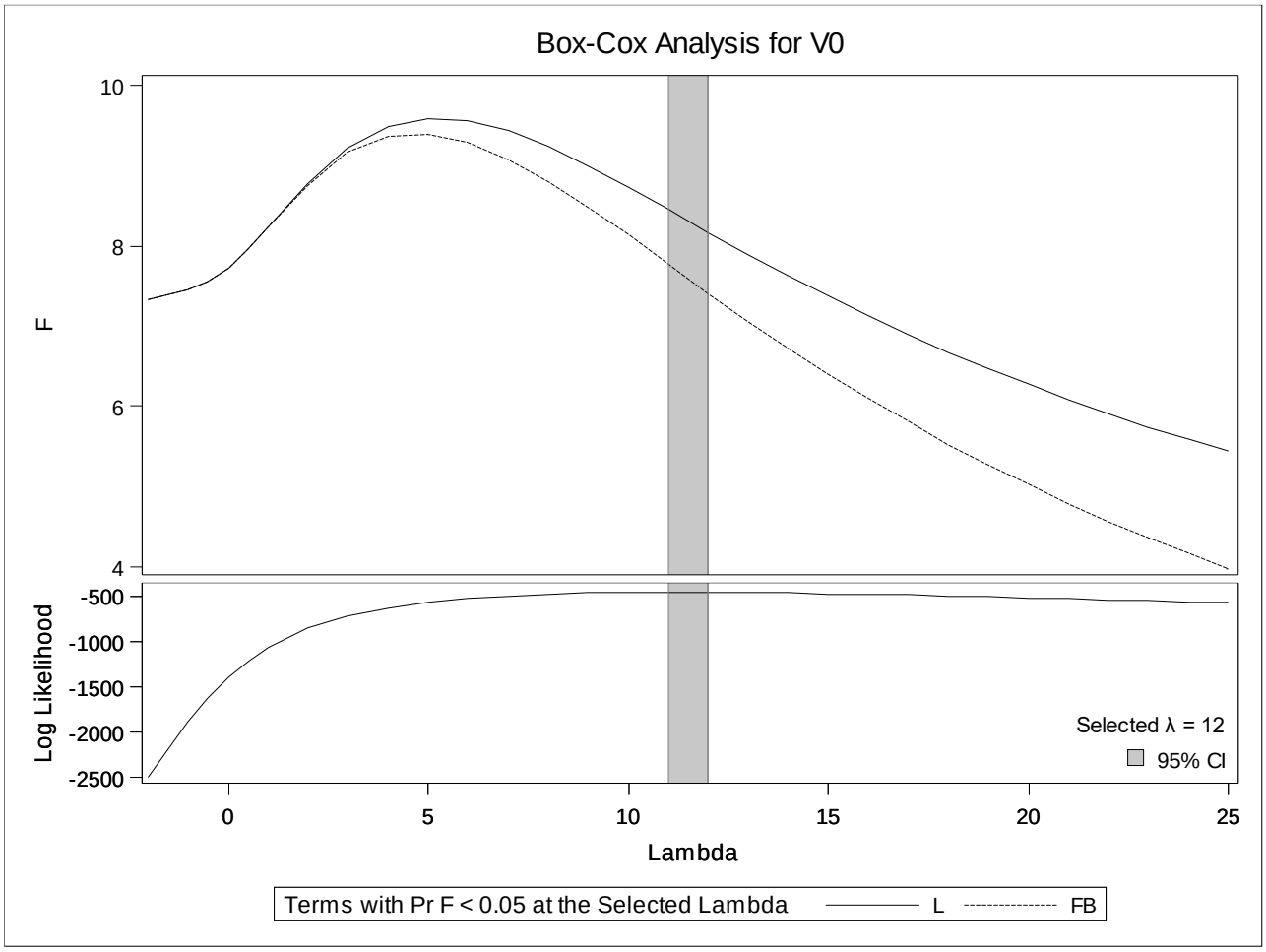
The GLM Procedure

Dependent Variable: V0 V0



Box Cox for Floating Point V0

The TRANSREG Procedure



Dependent Variable BoxCox(V0)
V0

Number of Observations Read	400
Number of Observations Used	400

The TRANSREG Procedure Hypothesis Tests for BoxCox(V0)
V0

Box Cox for Floating Point V0

The TRANSREG Procedure

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	*DF*	*Sum of Squares*	*Mean Square*	*F Value*	*Liberal p*
Model	2	8.72E47	4.36E47	7.79	>= 0.0005
Error	397	2.221E49	5.594E46		
Corrected Total	399	2.308E49			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Root MSE	2.365248E23	*R-Square*	0.0378
Dependent Mean	1.164425E24	*Adj R-Sq*	0.0329
Coeff Var	20.31258	*Lambda*	12.0000

Floating Point: V0^12 Full Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	V0_12
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	9
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	8.074E48

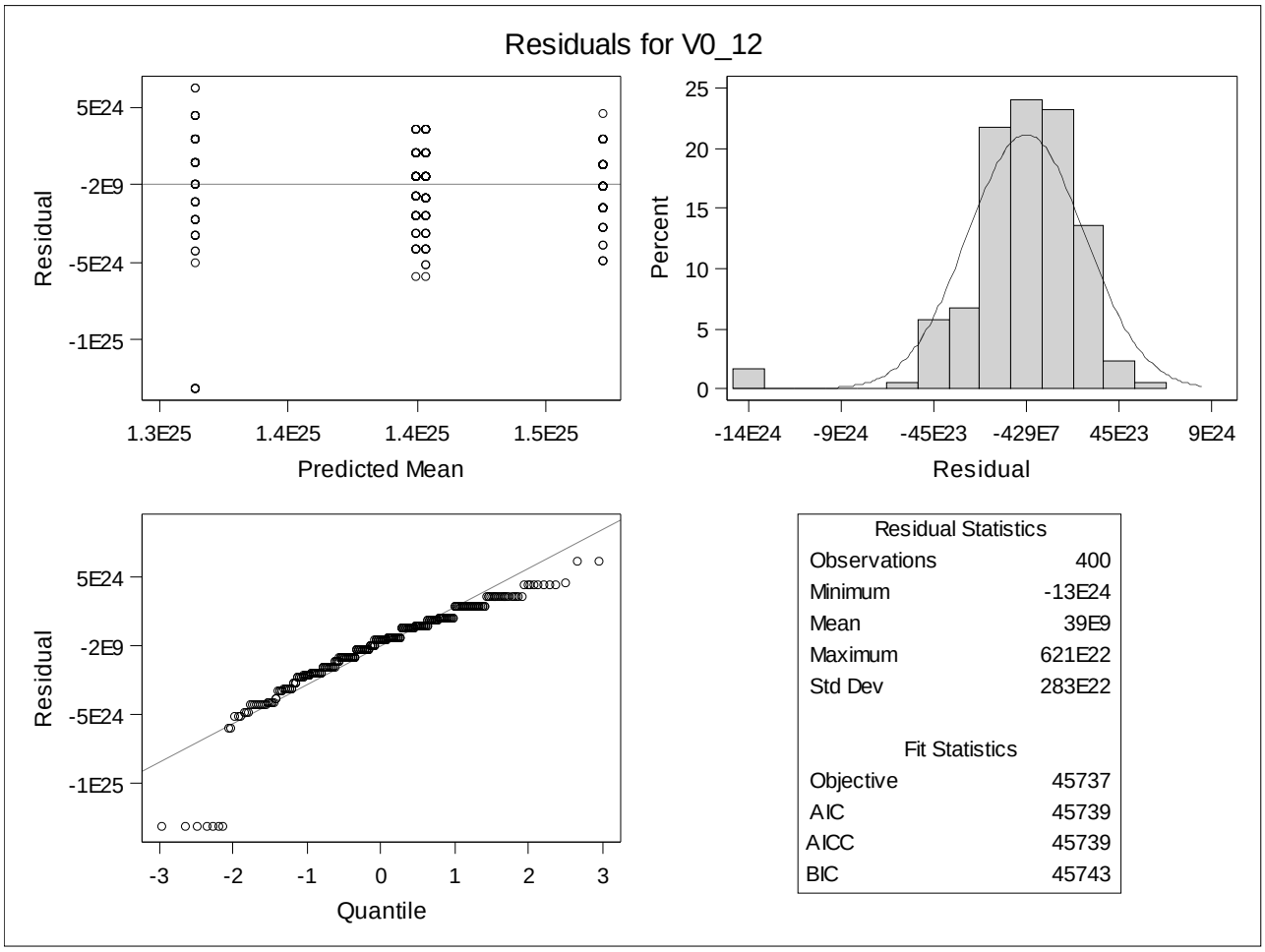
Fit Statistics	
-2 Res Log Likelihood	45736.9
AIC (Smaller is Better)	45738.9
AICC (Smaller is Better)	45738.9
BIC (Smaller is Better)	45742.9

Floating Point: V0^12 Full Model

The Mixed Procedure

Type 3 Tests of Fixed Effects

Effect	*Num DF*	*Den DF*	*F Value*	*Pr > F*
L	1	396	8.15	0.0045
FB	1	396	7.40	0.0068
*L*FB*	1	396	0.10	0.7560



Floating Point: V0^12 Main Effects Model

The Mixed Procedure

Model Information	
Data Set	WORK.BEV
Dependent Variable	V0_12
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	*Levels*	*Values*
L	2	L=3 L=4
FB	2	3x3 5x5

Dimensions	
Covariance Parameters	1
Columns in X	5
Columns in Z	0
Subjects	1
Max Obs per Subject	400

Number of Observations	
Number of Observations Read	400
Number of Observations Used	400
Number of Observations Not Used	0

Covariance Parameter Estimates	
Cov Parm	*Estimate*
Residual	8.056E48

Fit Statistics	
-2 Res Log Likelihood	45848.2
AIC (Smaller is Better)	45850.2
AICC (Smaller is Better)	45850.2
BIC (Smaller is Better)	45854.2

Floating Point: V0^12 Main Effects Model

The Mixed Procedure

Type 3 Tests of Fixed Effects

Effect	*Num DF*	*Den DF*	*F Value*	*Pr > F*
L	1	397	8.17	0.0045
FB	1	397	7.42	0.0067

Least Squares Means

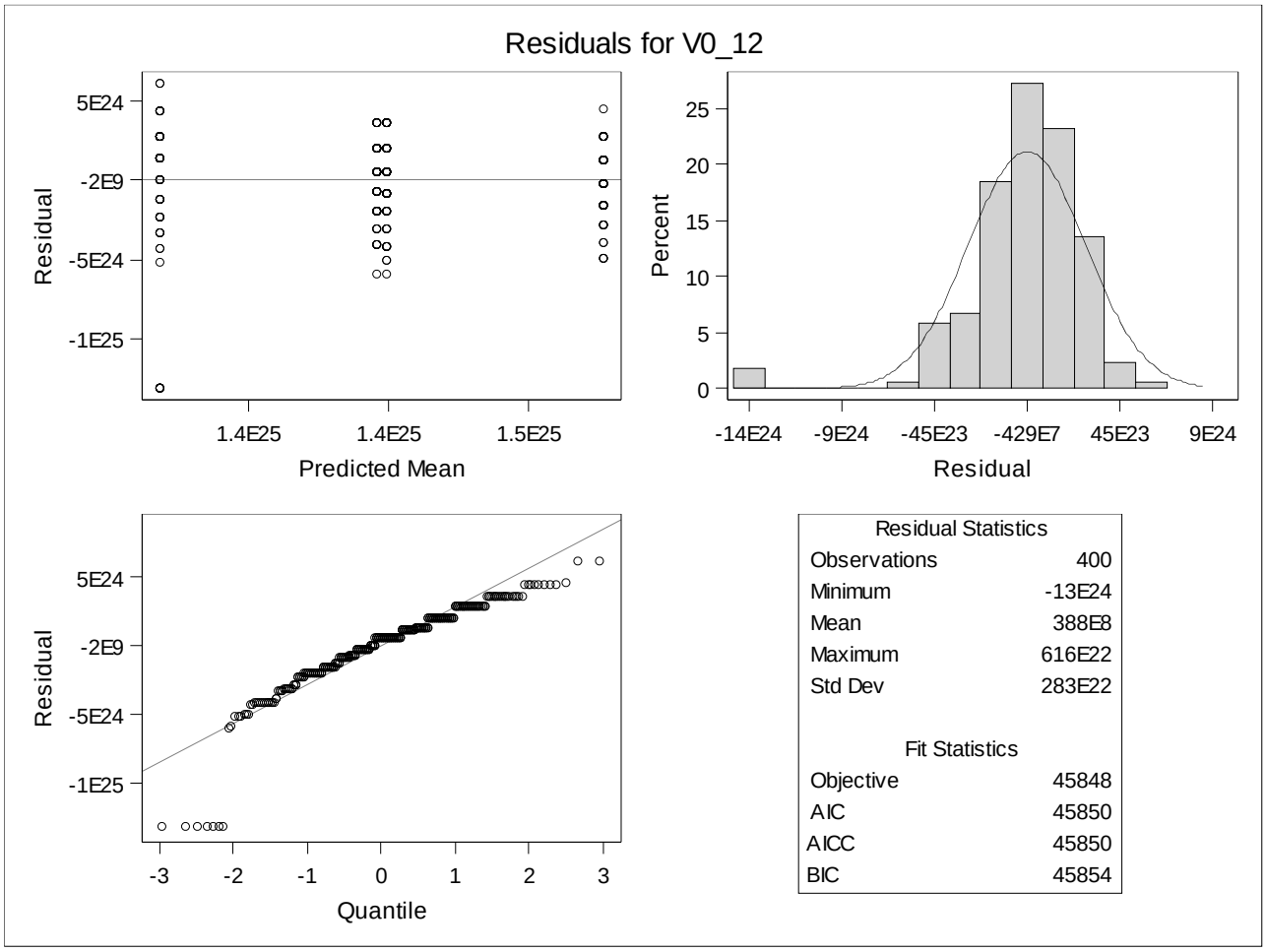
Effect	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Alpha*	*Lower*	*Upper*
L	L=3		1.438E25	2.007E23	397	71.64	<.0001	0.05	1.398E25	1.477E25
L	L=4		1.357E25	2.007E23	397	67.60	<.0001	0.05	1.317E25	1.396E25
FB		3x3	1.436E25	2.007E23	397	71.55	<.0001	0.05	1.397E25	1.475E25
FB		5x5	1.359E25	2.007E23	397	67.70	<.0001	0.05	1.319E25	1.398E25

Differences of Least Squares Means

Effect	*L*	*FB*	*L*	*FB*	*Estimate*	*Standard Error*	*DF*	*t Value*	*Pr > |t|*	*Alpha*	*Lower*	*Upper*
L	L=3		L=4		8.113E23	2.838E23	397	2.86	0.0045	0.05	2.533E23	1.369E24
FB		3x3		5x5	7.73E23	2.838E23	397	2.72	0.0067	0.05	2.15E23	1.331E24

Floating Point: V0^12 Main Effects Model

The Mixed Procedure



Residual Statistics	
Observations	400
Minimum	-13E24
Mean	388E8
Maximum	616E22
Std Dev	283E22

Fit Statistics	
Objective	45848
AIC	45850
AICC	45850
BIC	45854

Appendix C

MATLAB Code

StartCode.m
This script file calls another script file to initialize most of the variables and calls all the functions needed to run the algorithm. It also stores the values of six performance metrics in the cell array Data.

```
%%%%%%%% LOAD IMAGE FILE %%%%%%%%

load Lena.mat img;              % Example uses a MATLAB data set
%%%%%%%% OUTSIDE INITIALIZATION %%%%%%%%
rng('shuffle');

InitializeVariables
HT = 16;
WD = 16;
N = HT * WD;
RndRow = randi(256 - HT + 1);   % choose random row
RndCol = randi(256 - WD + 1);   % choose random column

%%%%%%%% GET IMAGE PATCH, MEASUREMENT MATRIX A AND VECTOR g %%%%%%%%
block = img(Rows(RandRow):Rows(RandRow)+HT-1, RandCol:RandCol+WD-1);
fvec = block(:);                % vectorize image patch
A = randn(round(p*N),N)/sqrt(N);
g = A*fvec;
%%%%%%%% GET ESTIMATED IMAGE FOR THIS IMAGE PATCH %%%%%%%%
cbank1 = cbank;

[cbank1,v,loops,time] = Fitg_WMP_FMS(g, A, cbank1, L, maxLoops,...
```

```
                mu1, mu2, tol, wgt, FMSopts);
%%%%%%%% CALCULATIONS AND STORE DATA %%%%%%%%
estimg = reshape(Dv(cbank1, v, N, L), [HT,WD]);

Data{1} = time;
Data{2} = loops;
Data{3} = psnr(estimg,block,255);
Data{4} = ssim(estimg,block,'DynamicRange',255);
Data{5} = nnz(v);
Data{6} = norm(v,1);
```

InitializeVariables.m

This script file initializes important parameters before the main execution begins.

```
sq2 = sqrt(2);
cbank = [ 1,  sq2, -1;          % initial filterbank
          2,    0,  2;
          1, -sq2, -1; ] / 4;
b = 3;

L = 3;
p = 0.5;

mu1 = @(t) 1.4^t;
mu2 = @(t) 1e13;

tol = sqrt(eps);                % tolerance for convergence test
wgt = 0.6;                      % wgt for WMP - Step 1
maxLoops = 1027;

FMSopts = optimset('Display','none');
```

blockgen.m

The script file generates a random image block of dimension $HT \times WD$ from the chosen test image.

```
function  [block, idx, RndRow, RndCol] = blockgen (imgs, HT, WD,
   numimg)

    idx = randi(numimg);
    RndRow = randi(256 - HT + 1);
    RndCol = randi(256 - WD + 1);
    block = imgs(RndRow:RndRow+HT-1,RndCol:RndCol+WD-1,idx);
end
```

Fitg_WMP_FMS.m

This file takes in g, A, $cbank$, L, $maxLoops$, $mu1$, $mu2$, tol, wgt, and $FMSopts$ as inputs and executes the two-step algorithm to fit a representation to the measurement vector g, returning updated filterbank $cbank$ and coefficient vector v, as well as t, the number of $Loops$, and $time$.

```
function [cbank, v, t, time] = Fitg_WMP_FMS(g, A, cbank, L, maxLoops,
   mu1, mu2, tol, wgt, FMSopts)
    tic;

    N = size(A,2);
    m = size(cbank,2);
    r = ((m - 1)*L + 1)*N;

    v = zeros(r,1);

    oldcbank = zeros(size(cbank));

    for t = 1:maxLoops
        % STEP 1
        oldv = v;
        v = WMPe22x11(AD(cbank, A, N, L), g, mu1(t), tol, wgt);
        % STEP 1

        if Converged(oldv, v, oldcbank, cbank, tol)
            break;
        end

        % STEP 2
        oldcbank = cbank;
        ObjFunc = @(cbank) FroNorm2(A*Dv(cbank, v, N, L) - g) ...
            + mu2(t)*norm(spDDtMinusId(cbank, N, L), 'fro')^2;
        cbank = fminsearch(ObjFunc, cbank, FMSopts);
        % STEP 2

        if Converged(oldv, v, oldcbank, cbank, tol)
            break;
        end
    end

    time = toc;
end

function conv = Converged(oldv, v, oldcbank, cbank, tol)
    conv = (    max(abs(oldv     - v    )           ) < tol ...
```

```
            && max(abs(oldcbank - cbank), [], 'all') < tol );
end
```

WMPe22x11.m

This function calls the Weak Matching Pursuit function written for this research. It takes in product AD, g, $mu1$, tol, and wgt as inputs and outputs the updated coefficient vector v. Note that wgt is referred to as w in the text. This function then calls the function Converged to test convergence. Inputs are $oldv$, v, $oldcbank$, $cbank$, and tol and the output is the logical variable $conv = 1$, if 'converged' and $conv = 0$, if not converged. Then this function calls the built-in MATLAB function *fminsearch*, which requires a call to function FroNorm2 and a call to the built-in MATLAB norm function, followed by another call to the function Converged.

```
function x = WMPe22x11(M,b,mu,tol,wgt)
% Weakly Orthogonal Matching Pursuit algorithm that sparsely finds x
% to minimize  mu*|Mx-b|_2^2 + |x|_1

    szM = size(M);
    szb = size(b);

    if nargin < 5
        wgt = 0.6;
        if nargin < 4
            tol = 1e-10;
            if nargin < 3
                mu = 1;
            end
        end
    end

    if length(szM) > 2 || prod(szM) == 0
        warning('M must be a matrix. Exiting.')
    elseif length(szb) > 2 || szb(2) ~= 1
        warning('b must be a column vector. Exiting.')
    elseif szM(1) ~= szb(1)
        warning('M and b must have the same number of rows. Exiting.')
    else
        itermax = min(szM);
        tempx = zeros(szM(2),1);    % initialize tempx (=v) to zeros
        x = tempx;
        f = Inf;
        nmM = vecnorm(M);           % The norms of the columns.
        nM = M ./ nmM;              % M with the columns normalized.

        r = b;                      % first residual is the vector b
```

```
            (=g)
33          nmr = r'*r;                 % norm of the residual
34
35          used = false(1,szM(2));     % matrix of logical zeros
36
37          for k = 1 : itermax         % itermax is the total number of
               atoms
38              if nmr < tol            % if the norm of residual is very
                   small
39                  break               %     break out of loop
40              end
41
42              IPs = abs(r'*nM(:,~used));      % IP of r with all unused
                   atoms
43              mx = max(IPs);                  % find mx = max IP (duh)
44              idx = find(IPs >= wgt*mx,1);    % 1st atom with IP large
                   enough
45              temp = find(~used,idx);
46              idx = temp(end);
47
48              used(idx) = true;
49
50              B = M(:,used);
51
52              tempx(used) = (B'*B)\B'*b;
53              r = b - B*tempx(used);
54              nmr = r'*r;
55
56              tempf = mu*nmr + norm(tempx,1);
57              if tempf > f
58                  break
59              else
60                  f = tempf;
61                  x = tempx;
62              end
63          end
64      end
65  end
```

AD.m

This function calculates the product AD. It takes in the $cbank$, A, N, and L and outputs the product of matrices A and D.

```
1   function ret = AD(cbank, A, N, L)
2      if size(A,2) == N
```

```
        m = size(cbank,2);
        r = ((m - 1)*L + 1)*N;

        ret = zeros(size(A,1),r);

        ret(:, 1:N) = A;

        for ell = 1 : L
           for s = 2 : m
             ret(:, ((m-1)*(L-ell)+s-1)*N+1 : ((m-1)*(L-ell)+s)*N) =
                ...
                   ret(:, 1:N) * spTpHt(cbank(:,s), ell, N);
           end

           ret(:, 1:N) = ret(:, 1:N) * spTpHt(cbank(:,1), ell, N);
        end
    else
        warning('AD : The dimensions of A are not compatible with Dt.')
        ret = zeros(size(A));
    end
end
```

FroNorm2.m
This function takes in the matrix X and outputs the square of the Frobenius norm of this matrix.

```
function ret = FroNorm2(X)
    ret = X(:)' * X(:);
end
```

spDDtMinusId.m
This function takes in $cbank$, N, and L and calls another function spDDt.m to first calculate a sparse version of the product DD^*, then output the difference $DD^* - I$.

```
function ret = spDDtMinusId(cbank, N, L)
    ret = spDDt(cbank, N, L) - speye(N);
end
```

spDDT.m
This function takes in $cbank$, N, and L, calls the function Recurse.m, which in turn calls the function spTpH.m and returns a sparse version of the product DD^*.

```
function ret = spDDt(cbank, N, L)
        % spDDt(cbank, N, L) returns a sparse D*D'
    ret = Recurse(cbank, N, L, 1);
```

```
end

function ret = Recurse(cbank, N, L, ell)
    if ell < L
        ret = Recurse(cbank, N, L, ell+1);
        H = spTpH(cbank(:,1), ell, N);
        ret = H' * ret * H;
    else
        H = spTpH(cbank(:,1), ell, N);
        ret = H' * H;
    end

    for k = 2 : size(cbank,2)
        H = spTpH(cbank(:, k), ell, N);
        ret = ret + H'*H;
    end
end
```

spTpH.m

This function takes in h, k, and N, dilates filter h to level k, then outputs an $N \times N$ Toeplitz plus Hankel matrix $H = H_i^{(k)}$.

```
function H = spTpH(h, k, N)
sz = size(h);
b = prod(sz);

if ~ismatrix(h) || ( sz(1) > 1 && sz(2) > 1 ) || ~mod(b,2)
    error("spTpH : Filter h must be a vector of odd length.")
%     H = sparse(N,N);      %On error, return the zero matrix.
else
    if k > floor(1 + log2( (2*N - 2)/(b - 1) ))
        error("spTpH : dilation level " + k + " is too large for N=" +
            N + " and filter size " + b)
%         H = sparse(N,N);      %On error, return the zero matrix.
    else

        if k > 0
            % Start with the middle
            m = (b + 1)/2;

            % To build the sparse matrix, array will contain:
            % array = [ row indices;
            %           col indices;
            %                values ];
%             array = zeros(N,3);
```

```
            array = zeros(b*N,3);
            Start = 1;

            num = N;
            array(Start:Start-1+num, 1) = 1:N;
            array(Start:Start-1+num, 2) = 1:N;
%             array(Start:Start-1+num, 3) = val(h(m), num);
            array(Start:Start-1+num, 3) = h(m);
            Start = Start + num;

            step = 2^(k-1);

            for i = 1 : m-1
                istep = i*step;

                % Below the diagonal
                num = istep;
                array(Start:Start-1+num, 1) = num:-1:1;
                array(Start:Start-1+num, 2) = 1:num;
%                 array(Start:Start-1+num, 3) = val(h(m+i),num);
                array(Start:Start-1+num, 3) = h(m+i);
                Start = Start + num;

                num = N - istep;
                array(Start:Start-1+num, 1) = (N-num+1):N;
                array(Start:Start-1+num, 2) = 1:num;
%                 array(Start:Start-1+num, 3) = val(h(m+i),num);
                array(Start:Start-1+num, 3) = h(m+i);
                Start = Start + num;

                % Above the diagonal
                num = N - istep;
                array(Start:Start-1+num, 1) = 1:num;
                array(Start:Start-1+num, 2) = (N-num+1):N;
%                 array(Start:Start-1+num, 3) = val(h(m-i),num);
                array(Start:Start-1+num, 3) = h(m-i);
                Start = Start + num;

                num = istep;
                array(Start:Start-1+num, 1) = N:-1:(N-num+1);
                array(Start:Start-1+num, 2) = (N-num+1):N;
%                 array(Start:Start-1+num, 3) = val(h(m-i),num);
                array(Start:Start-1+num, 3) = h(m-i);
                Start = Start + num;
            end
```

```
68
69              H = sparse(array(:,1), array(:,2), array(:,3), N, N);
70
71          else
72              % At level k=0, return the identity.
73              H = speye(N);
74          end
75      end
76  end
77  end % function
```