Regression Models for Spatial Images

By

Jun Zhang

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

(STATISTICS)

at the

UNIVERSITY OF WISCONSIN – MADISON

2008

A dissertation entitled

Regression Models for Spatial Images

submitted to the Graduate School of the
University of Wisconsin-Madison
in partial fulfillment of the requirements for the
degree of Doctor of Philosophy

by

Jun Zhang

Date of Final Oral Examination: 11/17/08

Month & Year Degree to be awarded: **December** 2008 **May** **August**

****

Approval Signatures of Dissertation Committee

Signature, Dean of Graduate School

Committee's Page. This page is not to be hand-written except for the signatures

Abstract

In spatial statistics, data can be in the form of spatial images and there are numerous situations, such as the analysis of geographic information system data, or remotely sensed (satellite) data, where there is interest in relating the information in one image to the information in another image or set of images.

Specifically, in this thesis, we present a Functional Concurrent Linear Model (FCLM) with varying coefficients to model the relationship among two-dimensional spatial images. To address overparameterization issues, the parameter surfaces in this model are transformed into the wavelet domain and a sparse representation is found by using a large scale l_1 constrained least squares algorithm. Once the sparse representation is identified, an inverse wavelet transform is applied to obtain the estimated parameter surfaces. The optimal penalty term in the objective function is determined using Bayesian Information Criterion (BIC). Subsampling and partial wavelet bases models are developed to reduce computational requirements when large images are involved, and we introduce measures of model quality. Finally, we also address the analysis of spatial images when missing data occur. One valuable finding is that, with appropriate modeling, an FCLM can handle both random missing and non-random missing patterns and select the best parts of different covariate images to fit the response image.

Acknowledgments

First I would like to thank my advisor Professor Murray Clayton for his excellent guidance. Not only I learned knowledge from him, but also I learned effective ways of communication and a very positive altitude towards everything from him.

I would like to acknowldge the help of many people to the completion of this disseration. Professor Phil Townsend gave me advice on many problems related to satellite images. Mr. Wei Zheng had a lot of discussion with me on the regression models for spatial images. I also want to thank my parents and parents-in-law for helping on taking care of my daughter. Finally, I would like to thank my wife Jie Zhou for her support and patience during this long process of pursuing my Ph.D. degree.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Description

Spatial statistics is about analyzing spatially correlated data. The simplest version of spatial data are two-dimensional spatial data where observations lie on a plane or a sphere. Perhaps the first and most famous study of two-dimensional spatial data can be traced back to 1854. In that year, Dr. John Snow crafted a map on which he clearly marked the positions for cholera cases and water pumps. Based on this map, he concluded that the outbreak of cholera was due to contaminated water and not miasma[17]. One hundred and fifty years have passed, however the principles of John Snow's method are still quite useful when dealing with spatial data.

In the 20th century statisticians accomplished great progress in many areas such as robust methods, survival models and bootstrapping. However the progress in data analysis accelerated after fast digital computers became available and mass data collection became routine. In spatial statistics, many new data are obtained as satellite images and these satellite images can contain

different types of information. Scientists are often interested in finding the relationships among different types of satellites images. In the simplest case, when looking at two types of satellites images for the same region, scientists want to make one type as the predictor and the other type as the response. In other words, scientists are looking for regression models for the two types of images. However, with satellite images scientists often cannot find ready-to-use regression tools, and so in this thesis, we try to develop regression tools for satellite images.

We next look at a concrete problem to illustrate why we need a new regression tool for images and we also discuss difficulties that arise with image data. Suppose we have two types of satellite images for Wisconsin: crop yield and levels of a certain mineral. In Figure 1, these two simulated satellite images are the only available images at hand. This is a typical case in the field of remote sensing: scientists often have one set of images available to them for analysis.

It is reasonable to guess that the local mineral level will affect local crop yield linearly. For example, the mineral level in the Madison area will linearly affect crop yield in that area. However, in the La Crosse area there could be a different linear relationship for crop yield and mineral level. Overall we might be interested in whether we can use a linear model to describe the relationship between crop yield and mineral level locally. Also, if a linear model fits the data, what kind of trends can we find in the model parameters. For example, we may find that for Madison and La Crosse, the linear models have similar

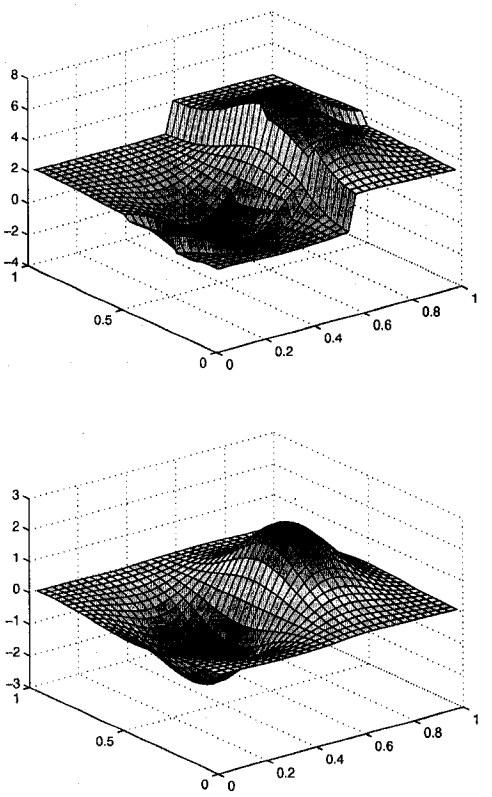


Figure 1: *Top: simulated satellite image for crop yield plotted as 3-D surface. Bottom: simulated satellite image for mineral level plotted as 3-D surface*

constant terms, but the slope term is 1.5 in the Madison area and 3.5 in the La Crosse area.

How can we fit a linear regression model for the set of satellite images in Figure 1? The default linear regression model assumes the same set of parameters for all data points. By looking at Figure 1, our guess is that one set of parameters for all data points may not work for this problem since the mineral level image is smooth but the crop yield image has some hills and valleys. One way to fit a linear regression model is by dividing each image into equal size windows and assuming in each window that all pixels share the same coefficients for the linear model. However one disadvantage associated with a quad-based method is the size of the window. How big shall we make a window? ten-by-ten, or twenty-by-twenty, or larger? Even if we have chosen a size for the windows, for each window we may still have very different coefficients for the linear model and this violates our assumption about having the same coefficients for each window. Even we allow different sizes for different windows we still need to decide on the size of the windows at each location. Ultimately, a quad-based method is ad hoc in its nature.

One may try to fit a conventional spatial linear regression model to the data. The first issue to consider is that the number of pixels is much bigger than the sample size a spatial regression model can typically handle. A spatial regression model requires users to calculate a variogram. For a small 640-by-480 image, to obtain a variogram requires computing the variogram estimator almost $\binom{300,000}{2}$

times. In the field of remote sensing, image sizes are often much larger than 640-by-480 and it would be prohibitively expensive to compute the variogram for these images. Also, a spatial regression model assumes the same coefficients for the linear model everywhere in the image which is probably not valid in many real world applications.

From the above comments, we see that satellite images are very different from traditional spatial data. First, images are huge matrices with very many elements. A small 640 by 480 image contains about 300,000 pixels. But traditionally spatial data were collected only in the hundreds and so applying traditional spatial methods directly on images is not going to work very well. There is an excellent summary in [1]: "the vast amount of data collected by satellites, radar and sonar measurements needs to be organized and reduced in complexity. While statistics originally emphasized obtaining maximal information from minimal data, the challenge from these new data sources is to summarize eloquently and to increase understanding of enormous quantities of information." High resolution satellite images are very popular. That means we have more and more pixels in each image. However pixel level details are not necessarily useful in the inference. For example, imagine that we stare at a mosaic mural. If our face is too close to the mural, we only see colored dots. If we move away from the mural, we start seeing a picture. If we move further away from the mural, we only see some rough big structures. One possible way to change image scale is a discrete wavelet transform (DWT). But we will need to decide what

the proper scale should be. Secondly, the relationship among different types of satellite images could be very complicated. Suppose we have n samples of two tuples of images, (Y_i, X_i) and n is much less than the number of pixels. Let us temporarily move away from the $n = 1$ situation in the crop yield example and assume we have enough "real" replicates in n samples, and we would like to predict image y given a new image x. What kind of model should we build? One possible model is:

$$y_p = f_p(x_1, x_2, ..., x_m) \tag{1.1}$$

where m is the number of pixels in each image and p is the index of pixels. This can be called a "full" model. What kind of function, $f(\cdot)$ are we looking at? If we want to apply a non-parametric approach, then $f(\cdot)$ would be a class of unknown functions. However m is usually much larger than the sample size n. It is probably impractical to estimate a very high dimensional function with relatively few samples. Even if we have enough real replicates, we would still have trouble implementing the algorithm on any common computers. If we limit $f(\cdot)$ to linear functions, we are still facing very complicated linear functions with m variables.

Now let us return to our crop yield example. If we want to fit a linear model then we have to consider adapting a functional linear model and focusing on local relationships. In other words, we do not try to identify the linear model which predicts Madison's crop yield with the mineral levels both in Madison

and in La Crosse. This kind of model may match the data very well. However it may be hard for scientists to interpret such relationship. A "local" linear model which predicts Madison's crop yield only with mineral levels in Madison or possibly with mineral levels in the area around Madison is more appealing. Of course, when scientists look at the local linear model, they would also like to consider the inherent spatial relationship in the image data. The scientists perhaps would ask themselves: "How can we build a local linear model for the image data which also represents the spatial relationship in an elegant way?" A very practical answer for the question is: how about changing the coefficients according to the spatial relationship in the image data. That means for the Dane and La Crosse county areas, the coefficients for the linear model could be very different and totally unrelated. On the other hand, within each county, the coefficients for the linear model should be similar and correlated. This kind of model is called a concurrent functional linear model with varying coefficients:

$$y_p = a_p + b_p x_p + e_p \tag{1.2}$$

A concurrent functional model or point-wise model with varying coefficients is a simple yet flexible model for image data. The spatial relationship in neighboring coefficients plays a important role in this model, which means we should expect similar values for a_p and b_p for most neighboring coefficients. This spatial relationship is very easy to visualize: if we put each a_p at its corresponding location

determined by its index p, then the a_p's form a two-dimensional parameter surface. Similarly the b_p's form another two-dimensional parameter surface. Now the problem has been translated into this form: how do we find two smooth parameter surfaces based on one set of images as in Figure 1?

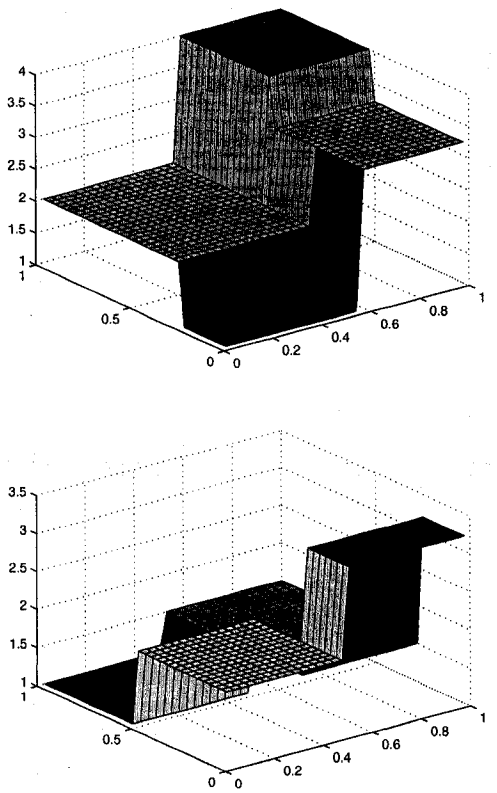


Figure 2: *Top: true parameter surface for A plotted as 3-D surface. Bottom: true parameter surface for B plotted as 3-D surface. In figure 1,* $Crop\ Yield = A + B \circ Mineral\ Level$.

As we can see in Figure 2, the true parameter surfaces of A and B in model $Y = A + B \circ X + E$ are three-dimensional step functions. In our crop yield

example, the relationship between crop yield and mineral level is a linear relationship. It is quite possible that for different regions, the constant terms and slopes are relatively stable inside each region and very different among different regions. Thus they could form three-dimensional step functions such as the surfaces shown in Figure 2. The parameter surfaces shown in Figure 2 will be used in the examples in Section 3 and the corresponding estimated parameter surfaces can be found there. Details about how to estimate the parameter surfaces will also be discussed in the Section 3.

Let us summarize what we have done so far:

1. For crop yield data, we try to find a linear model that can be locally fitted everywhere. This local relationship significantly simplifies the problem. From the interpretability point-of-view, the model is reasonable for many real world applications with local linear causal relationships among variables.

2. We assume that the real parameter surfaces are smooth or at least locally smooth and can be sparsely expressed. Without the smoothness assumption it is impossible to solve the problem effectively.

3. If we want to predict Madison's crop yield based on the mineral level of Madison and its immediate neighborhood, then we have a local inference functional linear model. This model can be reduced to a functional concurrent linear model [23].

With the above example, we have shown that it is possible to fit a concurrent linear model with varying coefficients for satellite images. However, many questions still need to be answered. How do we know that the model is a good fit for the data? A diagnostic tool is needed for this purpose. A good diagnostic tool could point to which areas have a good fit and which areas do not have a good fit. Can we decide for what types of images this model behaves well without knowing the true parameters? How do we make the model more robust under different noise structures? Some of these questions will be addressed in the rest of this thesis. This thesis is organized as follows: Chapter 1 contains a literature review; Chapter 2 includes the main theories and examples; Chapter 3 is about missing data; in Chapter 4 we will discuss some possible future extensions of our current framework.

1.2 Literature Review

1.2.1 Functional Regression Models

In Section 1.1, we mentioned that we will use a functional linear model in our research project. Actually, Functional Regression Analysis has been applied in the time series domain successfully in the past and many researchers from different fields have written papers on the topic. The basic idea behind functional regression according to Ramsay and Silverman [23] is: we can view a series of data in the time domain as a smooth function and express this function as the

summation of *a few* weighted basis functions:

$$X(t) = \sum_{j=1}^{K} a_j \phi_j(t) \tag{1.3}$$

We should note that *a few* is very important here. The direct consequence of Equation (1.3) is we have much fewer parameters a_j than the number of original data points. This, in some sense, serves as a way to reduce the original data dimension if we view each time point as one dimension. Now let us look at some papers on the different applications of functional linear models.

Faraway [10] studied reach motion data with a functional linear regression model. The paper was based on experiments done in the Human Motion Simulation Laboratory at the University of Michigan. The reach motions of 20 right handed subjects were recorded as angle curves for different body parts in the time domain under different environments. The author treated factors x_i^T as scalar variables like those in ordinary linear regression and treated the responses $y_i(t)$ and parameters $\beta(t)$ as time domain functions. His model is:

$$y_i(t) = x_i^T \beta(t) + \epsilon_i(t) \tag{1.4}$$

where the $y_i(t)$'s are angle curves and the x_i are factors such as the location of the target being reached, the age of the subjects and so on. The author expanded $y_i(t)$ and $\beta(t)$ with B-spline bases and essentially translated the problem into a common linear regression problem. He mentioned that using many basis

functions will not help the fitting when the underlying angle curves are smooth. The author wanted to apply a functional model because the model must have functions as responses.

Sometimes covariates contain functions which benefits from functional models. Ratcliffe et al. [24] used a functional linear regression model to study periodically stimulated foetal heart rate data. This paper, like the previous one, is also based on the idea of basis expansion. The variation here is that the model in this paper includes both functional and scalar covariates. The actual model is:

$$y_i = z_i^T \alpha + \sum_j \gamma_j \int x_{i,j}(t)\beta(t)dt + \epsilon_i \tag{1.5}$$

The response y_i is Psychomotor Development Index(PDI) which is a scalar development index for infants. α includes many scalar control variables such as the sex of infants, age at delivery, etc. The stimulus was applied every minute for 19 minutes. γ_j is the parameter for each stimulus. $x_{i,j}(t)$ is foetal heart rate measurements recorded every 0.2 seconds after each stimulation and $\beta(t)$ is a functional parameter. The goal of the research is to decide whether the foetal heart rate response to the stimulus is related to birth results and the child's development at 18 months. They concluded that heart rate changes caused by stimulation is an effective way to predict the PDI scores. The estimated $\hat{\beta}(t)$ shown in the paper is the summation of weighted Fourier basis functions. Based on the shape of $\hat{\beta}(t)$, they could tell what kinds of $x_{i,j}(t)$ indicate higher PDI

score. In their paper, they compare results from usual linear regression and functional regression. However, they did not use any functional covariates in the usual linear model. This, I believe, makes their comparison results not very useful.

In above two papers, functional models can be called "hybrid" linear models since they contain both functional and scalar variables. Next we will review a paper with a "full" functional linear model. Yamanishi and Tanaka [33] presented a spatially weighted functional linear model with functional responses, covariates and parameters. Although the authors put "spatially weighted" in their paper's title, they actually described a functional linear model in the time series domain. Their model can be written as:

$$y_i(t) = \beta_0(t) + \sum_j \int x_{ij}(s)\beta_j(s,t,p_i)ds + \epsilon_i(t) \tag{1.6}$$

where p_i is the index of geographical locations, $y_i(t)$ is an annual daily temperature curve for a certain location and the $x_{ij}(t)$'s are curves such as daily precipitation curves and daylight time curves for the same location. In their example, they tried to predict temperature curves from precipitation and daylight time curves for four different geographical locations. They also included a Monte Carlo hypothesis testing algorithm to see whether spatial variability for the functional parameter $\beta_j(s,t,p_i)$ exists.

Goutis [13] studied a functional regression model with second-derivatives.

This paper demonstrates the flexibility of functional models by showing that we can also include the second derivatives in the model. The model would be useful if we want to predict responses from the acceleration of certain explanatory variables. The model discussed in this paper is:

$$y = \alpha + \int (D^2\xi)(t)(D^2\beta)(t)dt + \epsilon \tag{1.7}$$

where D^2 is the second-order differential operator, y, α and ϵ are scalars and $\xi(t)$ and $\beta(t)$ are functional variables. The author solved the problem with the roughness penalized least squares estimation:

$$\frac{1}{n}\sum_{i=1}^{n}\{y_i - \alpha - \int (D^2\xi_i)(t)(D^2\beta)(t)ds\}^2 + \lambda \int ((D^2\beta)(t))^2 ds \tag{1.8}$$

Almost all of the above papers are in the time domain. We have not seen many papers on functional regression analysis in the spatial domain. We guess that this is partly because two dimensional data in the spatial domain are more complicated than one dimensional data in time series domain. However general abstract concepts in functional data analysis should be the same for time domain and spatial domain data. Thus the success stories in the time domain should encourage us to apply functional methods in the spatial domain.

We notice that quite a number of papers including some of above papers are mentioned in Ramsay and Silverman's book, Functional Data Analysis [23].

This book covers many aspects such as registration, principle component analysis, and functional linear models. Its coverage on functional linear models is probably the most complete literature review available. For the linear models, the book reviews the model with functional responses and scalar covariates, the concurrent model with functional responses and functional covariates, the functional model with scalar responses and functional covariates and the model with both functional responses and functional covariates. The links among the different models are also discussed. For example, the authors indicated that a general functional linear model could be reduced to the functional concurrent linear model.

A functional concurrent model in the time domain can be expressed as:

$$y_i(t) = x_i(t)\beta(t) + \epsilon_i(t) \tag{1.9}$$

This model is called a varying-coefficient model by Hastie and Tibshirani [14]. However, earlier in 1985, West et al. [32] investigated dynamic linear models which includes AR(1) type varying coefficients. Nevertheless Hastie and Tibshirani's paper covered more details and examples of this type of model. Gelfand et al. [12] studied a concurrent model in the spatial setup based on a Bayesian framework. Eubank et al. [9] investigated smoothing spline estimation with a concurrent model.

The last thing we want to mention is that asymptotics are also studied by

some researchers. For example, Cuevas et al. [3] did some asymptotic study from the frequentist perspective. In this paper they proved certain asymptotic results under two conditions: new sample functions must fill function space with an increasing number of orthogonal directions, and for each direction there are many functions. Also Cressie [2] described asymptotics in the spatial domain for two cases: one with increasing spatial resolution and fixed spatial coverage and the other with increasing spatial coverage and fixed spatial resolution. However asymptotics in the functional analysis domain is a hard topic and still in its infancy. It is possible that we can spend some effort on studying how pixel number increasing to infinity will affect the result. This is quite different from the usual asymptotics where we have the sample size increasing to infinity.

1.2.2 Discrete Wavelet Transform

In our approach, Discrete Wavelet Transforms will be used to obtain sparse representations of the parameter surfaces. Walker [31] has written a good introductory book on the Wavelet Transform and offers readers a very lucid explanation about the basics. Readers familiar with Fourier Transform may ask why we need to use Wavelet Transform. For many application with stationary signals, a Fourier Transformation is probably enough. For a stationary signal, its frequency components last through the whole duration of the signal. A Fourier Transform translates the signal into the frequency domain and gives a nice sparse representation of the original signal in the frequency domain. However,

if the signal is non-stationary, then a Fourier Transform will only give frequency domain information and totally ignore time domain information in the signal. However, the time information actually is very important for the non-stationary signals. For example, a signal contains a 10 Hz component in the first half of its duration and a 20 Hz component in the second half of its duration. With a Fourier Transform, we only have two peaks at 10 and 20 Hz in the frequency domain. Looking at the frequency domain plot, we do not know that there is a frequency change in the middle of the duration. If we assume that the original signal is stationary, then we will mistakenly believe that both frequencies exist for the whole duration of the signal. A Wavelet Transform fixes this problem by performing *localized* analysis on the signals. Thus the time domain information is well preserved in the Wavelet Transform. Similarly, with non-stationary 2-D spatial signals, a Wavelet Transform will keep the spatial domain information while extracting the frequency domain information from the signals. Because many satellite images are non-stationary 2-D spatial discrete signals, we decided to use a Discrete Wavelet Transform in our algorithm.

Next we will review some basic concepts of the Discrete Wavelet Transform by describing a very simple example. Let us look at a one-dimensional signal S with equal time intervals:

$$S: \quad 5 \quad 8 \quad 9 \quad 11 \quad 8 \quad 8 \quad 8 \quad 8$$

The simplest wavelets are *Haar wavelets.* For this example, the one-level Haar wavelets are:

$$\begin{aligned} W_1^1 &= (1/\sqrt{2}, -1/\sqrt{2}, 0, 0, 0, 0, 0, 0) \\ W_2^1 &= (0, 0, 1/\sqrt{2}, -1/\sqrt{2}, 0, 0, 0, 0) \\ &\vdots \\ W_4^1 &= (0, 0, 0, 0, 0, 0, 1/\sqrt{2}, -1/\sqrt{2}) \end{aligned}$$

The *details* of S, D^1 can be obtained as follows:

$$\begin{aligned} D^1 &= (S \cdot W_1^1, S \cdot W_2^1, \ldots, S \cdot W_4^1) \\ &= (-3/\sqrt{2}, -\sqrt{2}, 0, 0) \end{aligned}$$

For this example, the one-level *Haar scaling signals* are:

$$\begin{aligned} V_1^1 &= (1/\sqrt{2}, 1/\sqrt{2}, 0, 0, 0, 0, 0, 0) \\ V_2^1 &= (0, 0, 1/\sqrt{2}, 1/\sqrt{2}, 0, 0, 0, 0) \\ &\vdots \\ V_4^1 &= (0, 0, 0, 0, 0, 0, 1/\sqrt{2}, 1/\sqrt{2}) \end{aligned}$$

The *approximation* of S, A^1 can be obtained as follows:

$$\begin{aligned} A^1 &= (S \cdot V_1^1, S \cdot V_2^1, \ldots, S \cdot V_4^1) \\ &= (13/\sqrt{2}, 10\sqrt{2}, 8\sqrt{2}, 8\sqrt{2}) \end{aligned}$$

We can easily verify that S can be written as a *one-level 1-D discrete wavelet expansion* in the wavelet domain, or as a weighted summation of wavelets and scaling signals :

$$\begin{aligned} S &= \sum_{i=1}^{8} a_i \phi_i = 13/\sqrt{2}V_1^1 + 10\sqrt{2}V_2^1 + 8\sqrt{2}V_3^1 + 8\sqrt{2}V_4^1 \\ &\quad -3/\sqrt{2}W_1^1 - \sqrt{2}W_2^1 + 0W_3^1 + 0W_4^1 \end{aligned}$$

The above is the 1-level Discrete Wavelet Transform (DWT) of signal S. For a multiple level DWT, let us look at a 2-level DWT with Haar wavelets. Still using the same signal S, we have level 2 Haar wavelets:

$$\begin{aligned} W_1^2 &= (1/2, 1/2, -1/2, -1/2, 0, 0, 0, 0) \\ W_2^2 &= (0, 0, 0, 0, 1/\sqrt{2}, 1/2, 1/2, -1/2, -1/2) \end{aligned}$$

and level 2 Haar scaling signals:

$$\begin{aligned} V_1^2 &= (1/2, 1/2, 1/2, 1/2, 0, 0, 0, 0) \\ V_2^2 &= (0, 0, 0, 0, 1/2, 1/2, 1/2, 1/2). \end{aligned}$$

The level 2 *details* of S, D^2 can be obtained as follows:

$$\begin{aligned} D^2 &= (S \cdot W_1^2, S \cdot W_2^2) \\ &= (-7/2, 0) \end{aligned}$$

and the level 2 *approximation* of S, A^2 can be obtained as follows:

$$\begin{aligned} A^2 &= (S \cdot V_1^2, S \cdot V_2^2) \\ &= (33/2, 16) \end{aligned}$$

Similarly, we can write S as the *two-level 1-D discrete wavelet expansion* in the wavelet domain, or the weighted summation of two-level wavelets and scaling signals:

$$\begin{aligned} S &= \textstyle\sum_{i=1}^{8} a_i \phi_i = 33/2V_1^2 + 16V_2^2 - 7/2W_1^2 + 0W_2^2 \\ &\quad -3/\sqrt{2}W_1^1 - \sqrt{2}W_2^1 + 0W_3^1 + 0W_4^1 \end{aligned} \tag{1.10}$$

A method called *thresholding* is often used to eliminate small wavelet coefficients. For instance, if we choose a threshold to eliminate coefficients whose absolute values are less than 1.5, then we will set the smallest non-zero wavelet coefficient, $-\sqrt{2}$, to zero in expression (1.10). If we raise the threshold to 2.5, then the two smallest non-zero wavelet coefficients, $-3/\sqrt{2}$ and $-\sqrt{2}$, will be set to zero. As we can see from expression (1.10), raising the threshold will leave us fewer non-zero wavelet coefficients and give us a simpler, but less accurate version of the original signal.

Chapter 2

Functional Concurrent Linear Models

2.1 Main Theories

In Chapter 1, we introduced a Functional Concurrent Linear Model (FCLM) for satellite images. Henceforth we assume that the real parameter surfaces of the model are smooth or at least locally smooth and can be sparsely expressed in the wavelet domain. Without the smoothness assumption it is impossible to solve the problem effectively.

We illustrate the functional concurrent linear model assuming that there is one explanatory variable — the following argument can be easily extended to cases with multiple explanatory variables. Suppose we have n pairs of images, (Y_i, X_i), $i = 1, \ldots, n$, $n \geq 1$. A functional linear concurrent model with varying coefficients for these n pairs of images can be written as:

$$Y_i(t_1, t_2) = \beta_0(t_1, t_2) + X_i(t_1, t_2)\beta(t_1, t_2) + E_i(t_1, t_2) \tag{2.1}$$

where $\beta_0(t_1, t_2)$ is a constant term, $\beta(t_1, t_2)$ is the slope term for $X_i(t_1, t_2)$, $E_i(t_1, t_2)$ is the error term and t_1 and t_2 are spatial indexes in two (orthogonal) directions. We rewrite our model in matrix form:

$$Y_i = A + X_i \circ B + E \tag{2.2}$$

where Y_i, X_i, B and E are M by N matrices, "$\circ$" stands for the Schur product, and E is the error matrix. For matrix B, the element at row t_1 column t_2 is $\beta(t_1, t_2)$. For matrix A, the element at row t_1 column t_2 is $\beta_0(t_1, t_2)$.

We next expand the parameter matrices A and B (2.2) with a 2-D discrete wavelet expansion and obtain:

$$Y_i = \sum_{j=1}^{H} v_j \phi_j + X_i \circ \left\{ \sum_{j=1}^{H} w_j \phi_j \right\} + E = \sum_{j=1}^{H} v_j \phi_j + \sum_{j=1}^{H} w_j (X_i \circ \phi_j) + E \tag{2.3}$$

where $H = M \times N$ and ϕ_j is the 2-D wavelet base [31]. To estimate the coefficients in this expression we use a shrinkage procedure based on LASSO [15]. Specifically, we seek to solve

$$\min_{v,w} \sum_{i=1}^{n} \|Y_i - \sum_{j=1}^{H} v_j \phi_j - \sum_{j=1}^{H} w_j (X_i \circ \phi_j)\|_F^2 \tag{2.4}$$
$$st. \quad \sum_{j=1}^{H} |v_j| + \sum_{j=1}^{H} |w_j| \leq t$$

where $\|\cdot\|_F$ is the Frobenius matrix norm. One major advantage of using

l_1 constraints is that many of the wavelet coefficients v_j and w_j in (2.4) will become exactly zero [28]. This is appealing since under our local smoothness assumption we should have only a few non-zero v_js and w_js. Moreover, this means that our original problem in (1.2) will no longer be over-parameterized.

To proceed, we want to utilize the l_1 constrained Least Squares Estimation (LSE) algorithm mentioned in [18], and this requires us to change (2.4) to the form used by [18]. First, if we define $X_i^j = X_i \circ \phi_j$, then (2.3) becomes:

$$Y_i = \sum_{j=1}^{H} v_j \phi_j + \sum_{j=1}^{H} w_j X_i^j + E. \tag{2.5}$$

We can write (2.5) in a vectorized form:

$$vec(Y_i) = \left(\begin{array}{cccccc} vec(\phi_1) & \dots & vec(\phi_H) & vec(X_i^1) & \dots & vec(X_i^H) \end{array} \right) (v_1 \dots v_H \; w_1 \dots w_H)^T$$

where $vec(Z)$ is defined by writing a matrix Z as a vector column-wise. Now define:

$$\begin{aligned} u &= (v_1 \quad \dots \quad v_H \; w_1 \quad \dots \quad w_H)^T \\ T &= \left(vec(Y_1)^T \quad (vec(Y_2)^T \quad \dots \quad (vec(Y_n)^T\right)^T \\ Z &= \left(\begin{array}{cccccc} vec(\phi_1) & \dots & vec(\phi_H) & vec(X_1^1) & \dots & vec(X_1^H) \\ vec(\phi_1) & \dots & vec(\phi_H) & vec(X_2^1) & \dots & vec(X_2^H) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ vec(\phi_1) & \dots & vec(\phi_H) & vec(X_n^1) & \dots & vec(X_n^H) \end{array} \right) \end{aligned}$$

Then we can rewrite (2.4) in the standard form:

$$\min_{u} ||T - Zu||_2^2 + \lambda ||u||_1 \tag{2.6}$$

With (2.6), we can directly apply the large-scale l_1 penalized least squares algorithm described in [18] to obtain $\hat{u}$. Now applying a 2-D inverse wavelet transform on the first half of $\hat{u}$ or $(\hat{v_1} \quad \ldots \quad \hat{v_H})$, we obtain the estimated constant surface $\hat{A}$. Similarly, the estimated slope surface $\hat{B}$ can be obtained by applying a 2-D inverse wavelet transform on the second half of $\hat{u}$ or $(\hat{w_1} \quad \ldots \quad \hat{w_H})$. Our estimation algorithm can be described as follows:

1. Transform parameter surfaces into the wavelet domain to obtain their sparse representations.

2. Use a large-scale l_1 penalized least squares algorithm to find appropriate sparse representations in the wavelet domain.

3. Perform an inverse discrete wavelet transform to obtain estimated parameter surfaces.

As we mentioned earlier in Chapter 1, a typical case would be that (Y_1, X_1)

is the only set of images available. In this case u, T and Z in (2.6) would be:

$$\begin{aligned} u &= (v_1 \quad \dots \quad v_H \, w_1 \quad \dots \quad w_H)^T \\ T &= (vec(Y_1)) \\ Z &= \begin{pmatrix} vec(\phi_1) & \dots & vec(\phi_H) & vec(X_1^1) & \dots & vec(X_1^H) \end{pmatrix}. \end{aligned}$$

We should note that the problem becomes underdetermined since Z has only H element and u has $2H$ unknown variables. From now on, we will focus on the case with only one set of images.

2.2 Finding the Penalization Term λ

An essential step in LASSO is the selection of the penalization term λ. There are many papers and books on selecting the penalization term λ [6, 15], and in particular Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC) are two useful criteria for picking a good λ. AIC and BIC are defined here as follows:

$$AIC(\lambda) = \frac{1}{N}\left(\frac{\| Y - \hat{Y} \|_F^2}{\hat{\sigma^2}} + 2k(\lambda)\right)$$

$$BIC(\lambda) = \frac{1}{N}\left(\frac{\| Y - \hat{Y} \|_F^2}{\hat{\sigma^2}} + log(N)k(\lambda)\right)$$

where N is the sample size, Y is the observed response image, $\hat{Y}$ is the fitted response image, $\hat{\sigma^2}$ is the estimated variance of the noise in the response image

and $k(\lambda)$ is the degrees of freedom or the effective parameter number of the model. We will focus on BIC here.

The selection of the penalization term λ is done by finding the λ_{opt} which minimizes BIC or

$$\lambda_{opt} = \arg\min_{\lambda} BIC(\lambda).$$

Before we can minimize BIC, we need to estimate $k(\lambda)$ and σ^2 first. For $k(\lambda)$, we will use the number of nonzero coefficients as an unbiased estimate of the degrees of freedom of the LASSO based on the arguments in [34]. To estimate $\hat{\sigma}^2$, we just apply a local linear model to obtain the point-wise residuals and use the sample variances of the point-wise residuals as $\hat{\sigma}^2$.

Searching for λ_{opt} requires us to compute the BIC scores through a sequence of equally spaced values of λ since we do not have an analytical form of BIC scores in terms of λ. However there exists a λ_{max} such that, for any λ greater than λ_{max}, a zero vector is the solution for (2.6) [22]. Thus we only need to search between 0 and λ_{max} to find the λ_{opt}, and λ_{max} is easy to find by $\lambda_{max} = ||Z^T T||_\infty$ using Z and T in (2.6) [22].

2.3 Evaluation of the Fit

Since we focus on the case of a single pair of input and response images, we use a measure suggested by Ramsay and Silverman [23] and defined by

$$R^2 = 1 - \frac{||Y - \hat{Y}||_F^2}{||Y - \bar{Y}||_F^2}$$

where Y is the observed response image, $\hat{Y}$ is the estimated response image and $\bar{Y}$ is the mean image whose pixel values are equal to the mean of all pixel values in Y.

R^2 alone is not enough for evaluating the model fit: because we have twice as many parameters as the sample size we can potentially obtain a degenerate solution in which $\hat{B}$ is almost everywhere zero and $\hat{A}$ mimics Y. Thus, a similarity between the estimated constant surface $\hat{A}$ and the observed response image Y can be used to decide whether we have such a degenerate case. We define the similarity, S, by

$$S = 1 - \frac{\|Y - \hat{A}\|_F^2}{\|Y\|_F^2}.$$

A large R^2 and a small S indicate we have a good fit while a large R^2 and a large S tell us that we have a degenerate case.

For simulated data we know the true parameters surfaces and hence we can measure the similarity between the estimated and true parameter surfaces by

calculating a similarity measure for them. For example, we propose measuring the similarity between the estimated slope surface $\hat{B}$ and the true slope surface B with

$$S_b = 1 - \frac{\|B - \hat{B}\|_F^2}{\|B\|_F^2}.$$

2.4 Numerical Results

In this section, we first use a series of simulated examples to demonstrate our methods, and then apply our model to remote sensing images from gypsy moth defoliation. As noted previously, we focus on one-replicate examples since this is a typical situation. For each example, the penalty term λ is determined by BIC, we implement our algorithm with MATLAB version 7.1 and we use a commodity laptop with a $2G$ Hz Intel Core 2 T7200 CPU and $2G$ memory.

2.4.1 Examples Based on Simulated Data

The functional concurrent linear model we used to generate simulated data is:

$$Y(t_1, t_2) = A(t_1, t_2) + X(t_1, t_2)B(t_1, t_2) + E(t_1, t_2)$$

where X is the covariate, Y is the response, A is the constant parameter surface, B is the slope parameter surface and E is the error matrix. The following two

parameter surfaces A and B will be used in every simulated data example:

$$A(t_1,t_2)=\begin{cases} 1 & 0\le t_1\le 0.25, 0\le t_2\le 0.53 \\ 2 & 0.25< t_1\le 1.0, 0\le t_2\le 0.53 \\ 3.2 & 0\le t_1\le 0.5, 0.53< t_2\le 1.0 \\ 4 & 0.5< t_1\le 1.0, 0.53< t_2\le 1.0 \end{cases}$$

$$B(t_1,t_2)=\begin{cases} 2.0 & 0\le t_1\le 0.5, 0\le t_2\le 0.5 \\ 1.0 & 0.5< t_1\le 1.0, 0\le t_2\le 0.5 \\ 3.2 & 0\le t_1\le 0.25, 0.5< t_2\le 1.0 \\ 1.5 & 0.25< t_1\le 1.0, 0.5< t_2\le 1.0 \end{cases}$$

The above two parameter surfaces are step functions. We purposely set 0.53 as one of the boundaries for parameter surface A. This makes the estimation harder because we now have a thin stripe-like region in the middle of the image which has very different combinations of values for surfaces A and B.

In the first two simulated data examples, the images sizes are 64×64, which means we have total of 8192 wavelet coefficients for the parameter surface A and B. We use a total of 6 levels of Haar wavelet bases in these two examples. For Haar wavelet bases, widths and heights of support of wavelet bases increase by a factor of two when moving from a finer wavelet level to a coarser wavelet level. Thus, in our examples the support of the smallest base is 2×2 and the support of the largest base is 64×64. The wavelet coefficients in the lower level

correspond to finer details of the image while the wavelet coefficients in the higher level correspond to coarser details [31]. For the true parameter surfaces, we have a total of 42 nonzero wavelet coefficients out of a total of 8192 wavelet coefficients. The true A surface has 37 nonzero wavelet coefficients and the true B surface has 5 nonzero wavelet coefficients. Thus true representation of parameter surfaces in the wavelet domain is very sparse and this guarantee that we can use a l_1 norm constrained LSE to find the sparsest solution [4, 5].

In the first example, X was set equal to

$$X(t_1, t_2) = 3\cos\left(\frac{8\pi}{t_1}\right) + 4\sin\left(\frac{8\pi}{t_2}\right) t_2 + 4 \tag{2.7}$$

and $Y = A + X \circ B + E$. We set E's elements to be i.i.d.$N(0, \sigma_e^2)$ with $\sigma_e/\sigma_y = 10\%$, where σ_y is the estimated standard deviation of the pixel values of image Y. For this example, λ_{max} was 100.8593, the estimated $\hat{\lambda}_{opt}$ was 1.6137, and the computing time required was 2.78 minutes. The R^2 was 0.9865 and the similarity S between $\hat{A}$ and the observed response Y was 0.4190. Together, this high R^2 and relatively low S indicate that we have a good fit. This is supported by the coefficient surface similarities: the similarity between $\hat{B}$ and B was 0.9975, while the similarity between $\hat{A}$ and A was 0.9669. Note that in the fitted surfaces we have 25 nonzero estimated wavelet coefficients for $\hat{A}$ and 6 nonzero estimated wavelet coefficients for $\hat{B}$. $\hat{A}$ has fewer nonzero coefficients than the true surface A does. Based on the results shown in Figure 3, we

conclude that our model appears to work properly in the first example.

In the second example, X is equal to

$$X(t_1, t_2) = 5(4t_1 - 2)e^{(-((4t_1-2)^2+(4t_2-2)^2))} \tag{2.8}$$

and $Y = A + X \circ B + E$. Again, we set E's elements to be i.i.d.$N(0, \sigma_e^2)$ with $\sigma_e/\sigma_y = 10\%$, where σ_y is the estimated standard deviation of the pixel values of image Y. For this example, λ_{max} was 114.1170, the estimated $\hat{\lambda}_{opt}$ was 1.3694, and the computing time required was 5.19 minutes. The R^2 was 0.9843 and the similarity S between $\hat{A}$ and the observed response Y was 0.8715. Together, this high R^2 and relatively high S indicate that we have a degenerate case. This is supported by the coefficient surface similarities: the similarity between $\hat{B}$ and B was 0.6966, while the similarity between $\hat{A}$ and A was 0.9933. Note that in the fitted surfaces we have 59 nonzero estimated wavelet coefficients for $\hat{A}$ and 11 nonzero estimated wavelet coefficients for $\hat{B}$. It seems likely that the degenerate case arises because we have a relatively flat X in this example, and this hampers the ability to estimate B well, somewhat analogous to the challenge of estimating a slope in simple linear regression when the range of the independent variable is small. The results are shown in Figure 4.

We have seen two simulated examples with different input image X's and same true parameter surfaces and signal noise ratios. Based on the above results, the condition of X can greatly affect the fit of the model. However we can always

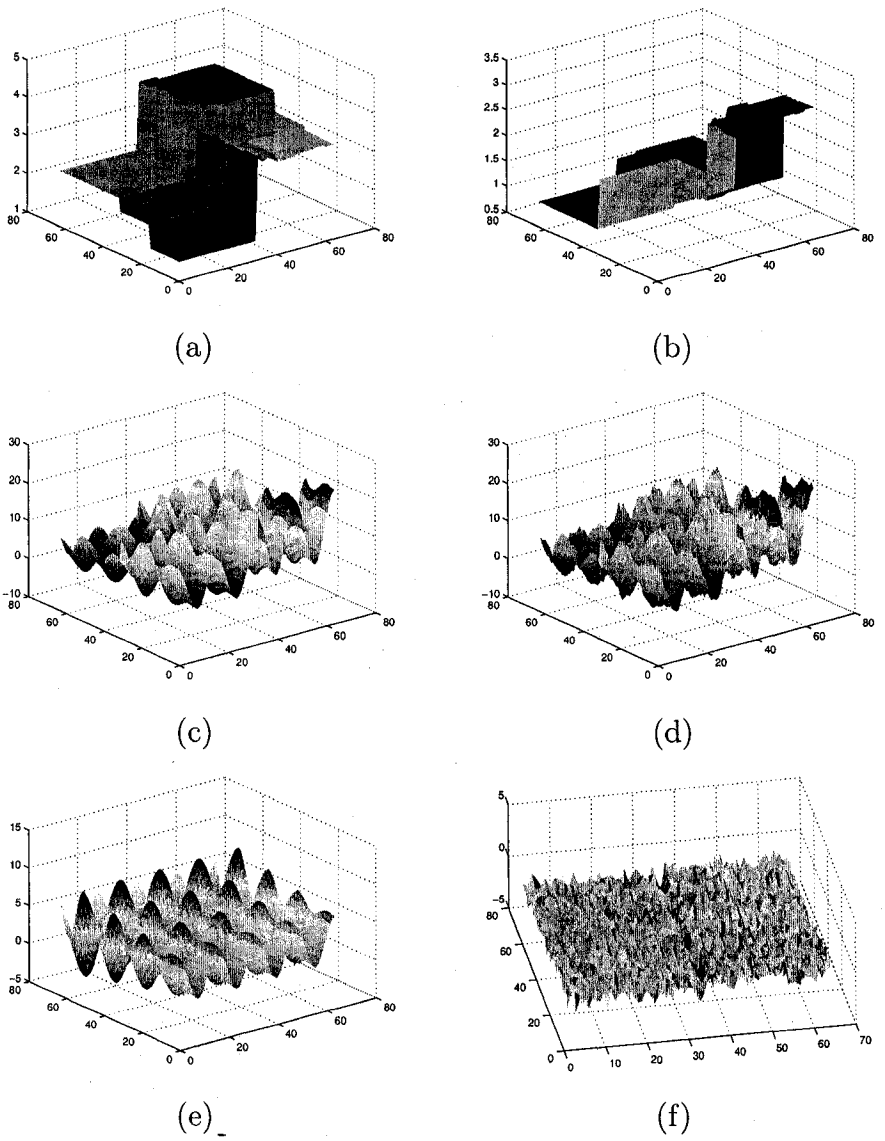


Figure 3: *Figures for the first simulated example. Image size is* 64×64. *(a) estimated parameter surface* $\hat{A}$. *(b) estimated parameter surface* $\hat{B}$. *The estimated parameter surfaces are still fine although we have doubled the noise standard deviation. More results for Example 1. (c) estimated response* $\hat{Y}$. *(d) observed noisy response* Y. *(e) covariate image* X. *(f) residual plot.* $Y = A + B \circ X + E$ *and noise standard deviation is equal to 10 % of the standard deviation of pixel values of* Y.

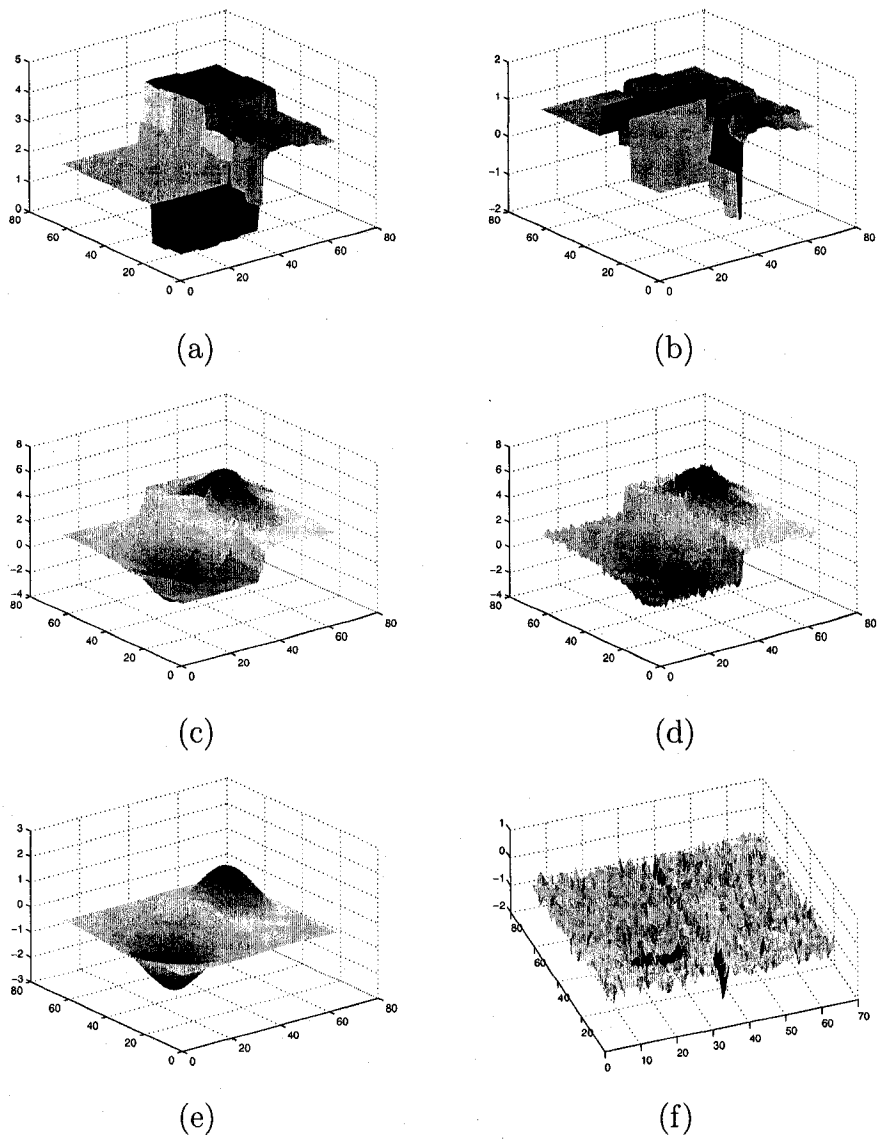


Figure 4: *Figures for the second simulated example. Image size is* 64×64. *(a) estimated parameter surface* $\hat{A}$. *(b) estimated parameter surface* $\hat{B}$. *The estimated parameter surfaces are still fine although we have doubled the noise standard deviation. More results for Example 2. (c) estimated response* $\hat{Y}$. *(d) observed noisy response* Y. *(e) covariate image* X. *(f) residual plot.* $Y = A + B \circ X + E$ *and noise standard deviation is equal to 10 % of the standard deviation of pixel values of* Y.

compute R^2 and S, the similarity between $\hat{A}$ and Y to evaluate the fitness of the model. On the other hand, the wavelet coefficients corresponding to large details in parameter surfaces normally will not have identifiability issues since $X_i \circ \phi_j$ and ϕ_j cannot be highly correlated unless covariate image X is very flat inside the support of ϕ_j. We also note that although Haar wavelet bases are coarse, they are symmetric and have very small support. This makes the design matrix Z in (2.6) very sparse which in turn make the computation very fast.

In Figure 3 and Figure 4, we notice that both residual plots have an narrow dip in the middle which also indicates that the fit in that area is not good. As we have mentioned in the beginning of this section, We purposely set 0.53 as one of the boundaries for parameter surface A. This creates a very narrow strip with very different combinations of values for the A and B surfaces. Our model identifies different combinations of values for surfaces A and B, but if that combination exists in a region which cannot be cheaply recreated with wavelet bases, then our model will not be able to find this particular combination.

One advantage of a wavelet expansion is its ability to capture multilevel properties of images under analysis. Small details like spikes can be represented with low level or fine wavelet bases and large details like trends can be represented with high level or coarse wavelet bases. Lacking high level or coarse wavelet bases will result in inefficiently using many low level or fine wavelet bases to represent large details in parameter surfaces, which means the true representation in the wavelet domain is no longer sparse. For example, we have

mentioned that we have total 42 nonzero wavelet coefficients in our examples using 6 levels of Haar wavelet bases. If we only use one level of Haar wavelet bases, we will have 2048 nonzero wavelet coefficients for the true parameter surfaces. Using 4 levels of Haar wavelet bases will bring that number down to 60. We suggest an empirical guideline for determining how many levels of wavelet bases we should use: we should make the support size of the largest wavelet bases at least a quarter of the original image for high Noise Signal Ratio (NSR) situations or at least one sixteenth of the original image for low NSR situations.

Next, we increase the size of our images to demonstrate the scalability of our algorithm. We use the X in (2.7) and the same noise matrix E with i.i.d.$N(0, \sigma_e^2)$ elements. We still set σ_e equal to 10% of σ_y. Again Haar wavelet bases are used in each case and the minimal support size is 2×2 and the maximal support size is equal to the current image size. We will increase the width and height of the images by two-fold each time. The results are listed in Table 1. According to Kim et al. [18], their large scale l_1 penalized LSE will have computational time proportional to $O(n^{1.2})$ where n is the number of total coefficients. Based on Table 1, we estimate the computational complexity of our algorithm to be about $O(n^{1.3})$. So, if we were to continue to increase the size to 512×512, we would need approximately 592 minutes. Of course with a faster machine, the computational time in Table 1 can be easily reduced.

Table 1: Scalability of the algorithm

image size	nonzero coefficients	total coefficients	λ_{opt}	time(min)
64×64	42	8192	1.6137	2.78
128×128	74	32768	1.6126	15.77
256×256	138	131072	1.6132	97.75

2.4.2 Gypsy Moth Defoliation Example

Backgrounds

Gypsy moth defoliation of oak trees is of concern in the Appalachian Mountains because it disrupts nitrogen cycling in forests [8, 20, 21]. Since gypsy moth defoliation varies widely in intensity and usually occurs over large areas, Townsend et al. [29] found it useful to use remote sensing data to predict disturbances to nitrogen cycling as a consequence of defoliation status. Figure 5 shows satellite images of defoliation rates caused by gypsy moth and surface elevation for the Savage River State Forest in Western Maryland, USA. The defoliation image is for an insect infestation that occurred in June-July of 2006, generated from Landsat satellite images with algorithms developed by Townsend et al. [29]. The elevation surface is derived from the National Elevation Data set of the US Geological Survey. White in the defoliation image corresponds to high defoliation rates and black indicates foliage growth. On the elevation image, low to high elevations are depicted in black to white tones, respectively. Both the elevation data and the defoliation image have 30 meter pixel resolution. These

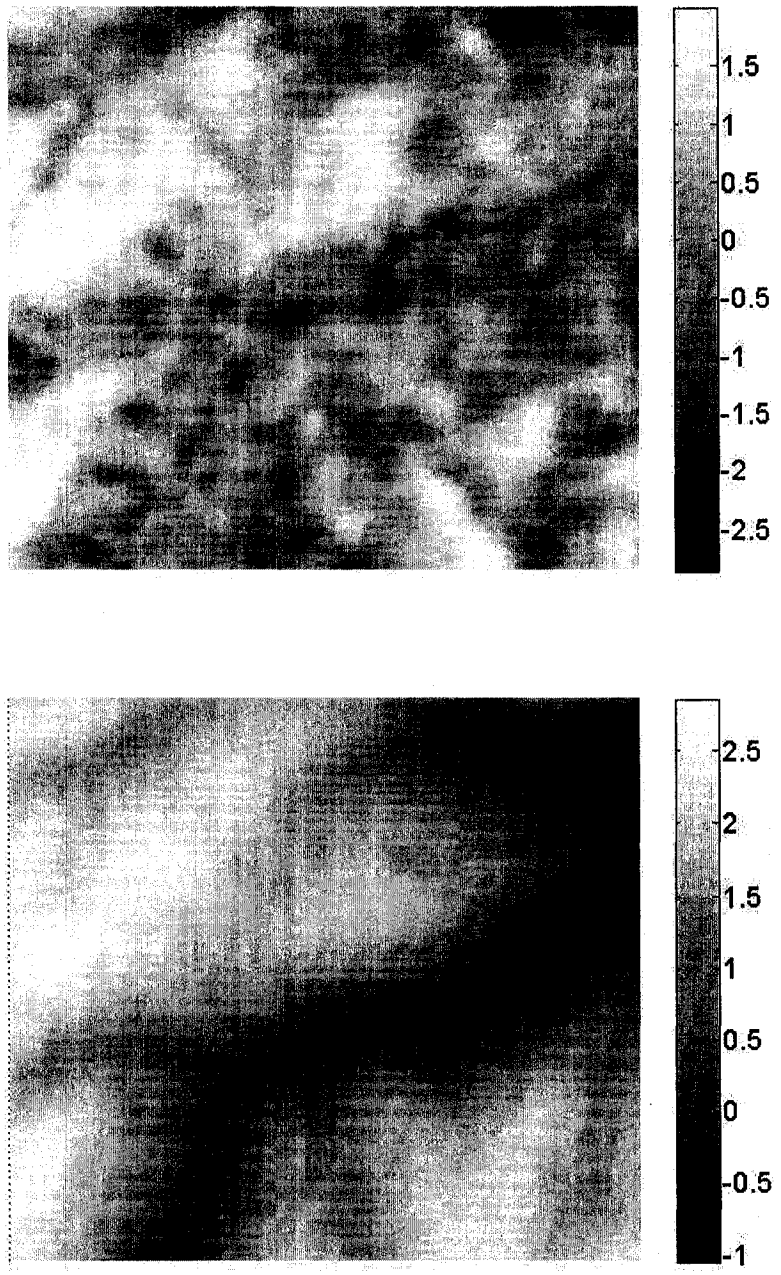


Figure 5: *Top: satellite image for gypsy moth defoliation rates. Bottom: elevation.*

images resulted from the only available cloud-free Landsat data corresponding to the insect disturbance – a typical case in the field of remote sensing.

Several authors have observed that gypsy moth defoliation rates increase with increased elevation [16, 19]. In Figure 5, we can see that defoliation rates are generally higher on ridges than in the valleys. Having a quantitative description of the spatial variation in the relationship between elevation and gypsy moth defoliation rates would provide researchers with a more accurate view of the true relationship, and would facilitate the identification of other possible factors impacting the dynamics of defoliation caused by gypsy moth.

From Figure 5 it seems reasonable to assume that, locally, elevation affects defoliation rates linearly, although from one subregion to the next there could be a different linear relationship for these two attributes; e.g. the slopes could vary across the region. This implies that a conventional spatial linear regression model is not suitable for the data and models based on non-stationary Gaussian processes should be considered instead. Another issue with the application of conventional spatial linear regression models to remote sensing data is that satellite images are huge matrices with very many elements (often 8000 by 8000 grid cells, with multiple channels). For common regression data analyzed by geostatistical methods typical data sets might number in the hundreds. However, for even a small 640-by-480 image, to obtain a variogram requires computing the summand in a variogram estimator almost $\binom{300,000}{2} \approx 4.5 \times 10^{10}$ times. The

extraordinary richness of remotely sensed data implies that, even when a variogram can be calculated, common models are unlikely to fit. Regardless, even when such models might fit, it would be prohibitively expensive to compute the variogram for typical remotely sensed images. This leads to the application of new modeling approaches.

Results

We first explore the model with only one covariate

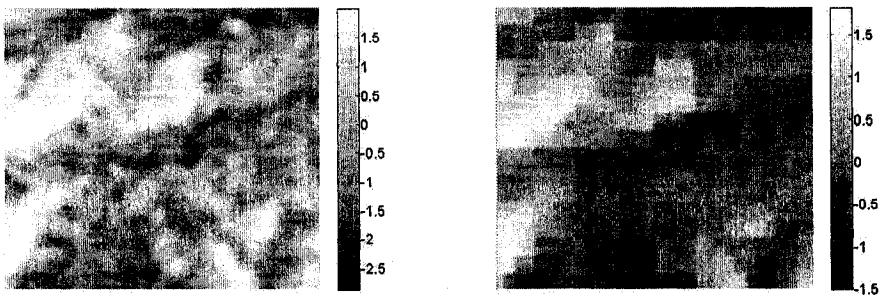
$$Y = A + X_1 \circ B_1 + E \tag{2.9}$$

where Y is defoliation rate and X_1 is elevation. Here, the image size is 64×64 and Haar wavelet bases were used, with the smallest base support being 2×2 and the largest 64×64. For this example, λ_{max} was 65.3873, the estimated $\hat{\lambda}_{opt}$ was 4.4463, and the computing time required for this example was 2.06 minutes. The R^2 was 0.6940 and the similarity S between $\hat{A}$ and the observed response Y was 0.1091. The results of the modeling are shown in Figure 6. The large R^2 value and very small similarity between $\hat{A}$ and Y indicate that the concurrent linear relationship between defoliation rate and elevation is very strong. The estimated parameter surface $\hat{B}_1$ for elevation clearly shows a positive relation between defoliation rates and elevation. Low to high elevations in X_1 usually match black to white tones in $\hat{B}_1$. What is more important is that the spatial variation of the

relationship between defoliation rate and elevation captured by the estimated parameter surfaces can be easily visualized and further inspected by experts. For example, experts who are concerned with some other factors of interest, such as the presence of unfavorable host tree species, nutrient availability, spraying with pesticides for insect suppression, etc. can simply locate zones where color tones mismatch between X_1 and $\hat{B}_1$ and carry out follow-up research for these zones.

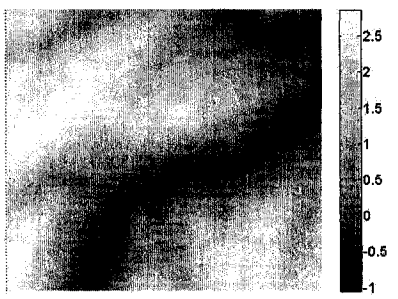
We next inquire whether additional variables could be added to the model. Specifically, from Figure 6 we notice that in the lower-right corner of the defoliation rate image there is a V-shaped area with negative defoliation rates and high elevation which results in a similar V-shaped area in the estimated slope surface $\hat{B}_1$. Inside this V-shaped area of $\hat{B}_1$, the slope terms are smaller than those in the surrounding area and the elevation levels are higher than those in the surrounding area. Again the usual positive relationship between defoliation rate and elevation is reversed.

A possible explanation can be found by examining a species composition image [11]. In Figure 7, the species composition image is a binary image, with white indicating forest types that are susceptible to gypsy moths and black indicating types less preferred by gypsy moths. By examining the species composition image we can clearly see that this V-shaped area in the defoliation rate image is similar to a V-shaped region corresponding to forest types susceptible

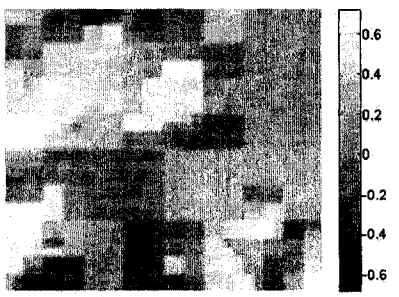


Y: defoliation rate

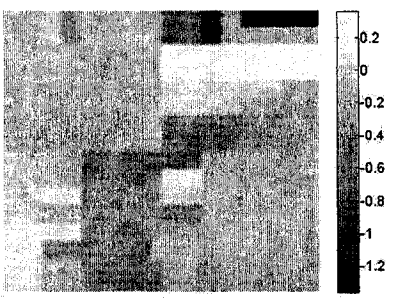
$\hat{Y}$



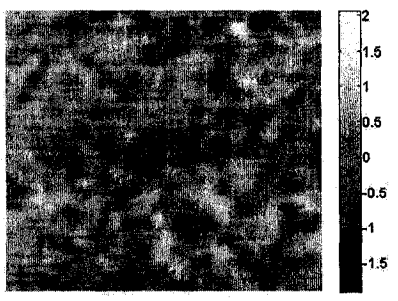
X_1:elevation



$\hat{B}_1$



$\hat{A}$



residual plot

Figure 6: *The model is* $Y = A + X_1 \circ B_1 + E$. *Elevation is the explanatory variable* X_1. *Defoliation rate is the response* Y.

to gypsy moth damage. Thus, we expand our model to be:

$$Y = A + X_1 \circ B_1 + X_2 \circ B_2 + E \tag{2.10}$$

where Y is defoliation rate, X_1 is elevation, and X_2 is binary species composition. Here, the image size is still 64×64 and Haar wavelet bases were used, with the smallest base support being 2×2 and the largest 64×64. For this example, λ_{max} was 65.3873, the estimated $\hat{\lambda}_{opt}$ was 5.4925, and the computing time required for this example was 2.15 minutes. The R^2 was 0.6723 and the similarity S between $\hat{A}$ and the observed response Y was 0.0444. The results of the modeling are shown in Figure 7. Although the V-shaped corner does not completely disappear in $\hat{B}_1$, the new estimated slope surface for elevation, it becomes less pronounced than before. This occurs because, X_2, the species composition, explains some defoliation rates in that region.

Contribution from each regressor can be measured with s sort of "partial" R^2, namely,

$$R_i^2 = 1 - \frac{||Y - X_i \circ \hat{B}_i||_F^2}{||Y - \bar{Y}||_F^2}.$$

For elevation R_1^2 was 0.4164 and for species composition R_2^2 was 0.1746. So we can roughly say that overall the elevation contributes twice as much as the species composition does. The model in (2.10) could be called a concurrent Analysis of Covariance (ANCOVA) model. When we try to understand the estimated parameter surface $\hat{B}_1$ for covariate X_1, we should realize that $\hat{B}_1$

now consists of the slope terms of elevations for two types of trees. So if the regression of Y on elevation for each tree type is of interest, then the model in (2.10) is the one we should consider.

Overall, R^2 for the model in (2.9) is slightly higher than R^2 in (2.10) and the similarity S between $\hat{A}$ and Y for the model in (2.9) is a bit higher than S in (2.10). From Figures 6 and 7, we do not observe any patterns in the residual plots. This suggests that our models fit the data well. Also, fairly low similarities between $\hat{A}$ and Y suggest degeneracy did not occur in these two cases. When we check the estimated defoliation rates $\hat{Y}$ in Figures 7, we find that it has more details than the $\hat{Y}$ in Figures 6. Thus we have two different models and both seem to explain the response well. However in this case it is difficult to recommend a better model purely based on R^2 and S. Each model has its own applications. We should make the choice of the model based on real situations.

We close this section by applying Coiflets wavelet bases to the gypsy moth data with the model in (2.9) to exam the effects of wavelet basis family choice. The Coiflets wavelet family has larger support than Haar wavelets do. They were first constructed by Daubechies at the request of Coifman. Coiflets are almost symmetric; a wavelet transform with Coiflets will keep the approximation subsignal very close to the original signal [31]. Image sizes are still 64×64 and images cover the same area of Savage River State Forest as before. We use coif1 wavelet bases and the highest level is 3. The smallest base support is 6×6 and

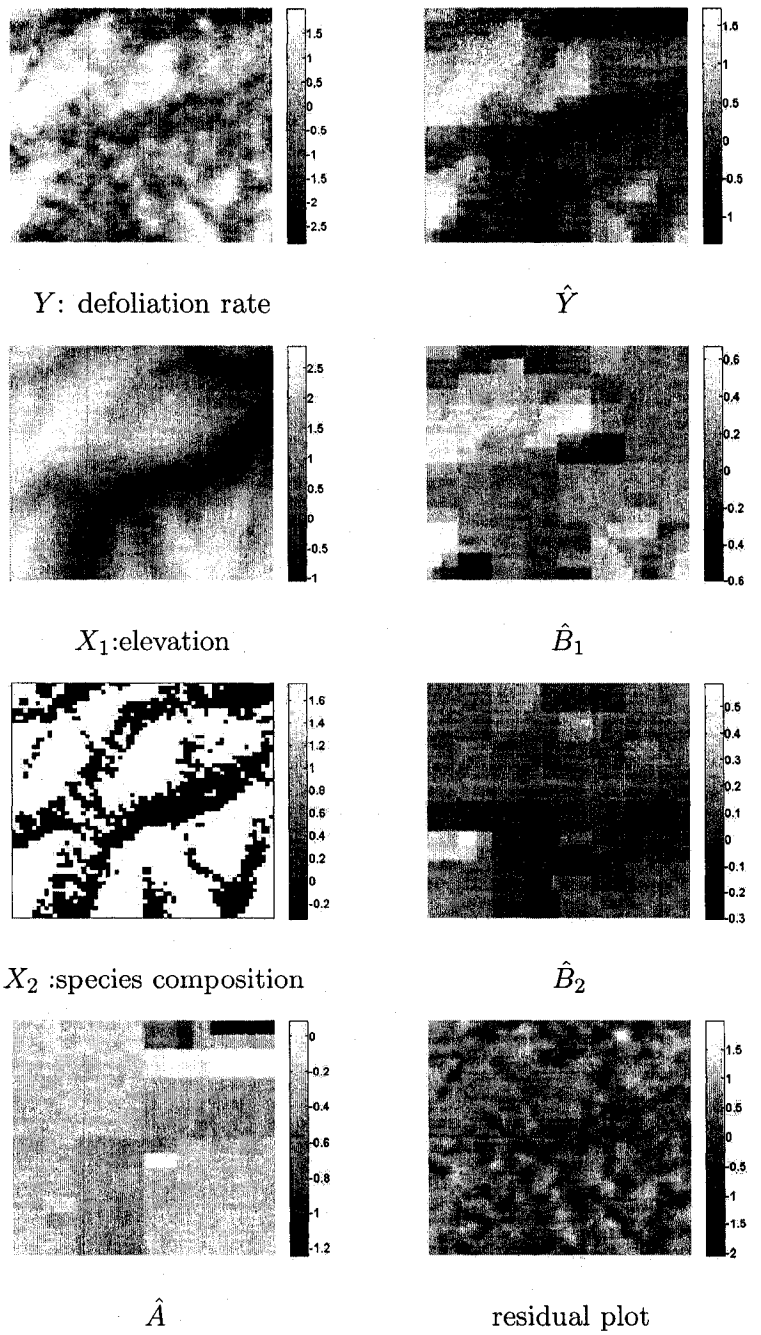


Figure 7: *The model is* $Y = A + X_1 \circ B_1 + X_2 \circ B_2 + E$ *Elevation is* X_1 *and species composition is* X_2. *Defoliation rate is the response* Y.

the largest is 36×36. For this example, λ_{max} was 78.8032, the estimated $\hat{\lambda}_{opt}$ was 4.7282, and the computing time required was 11.93 minutes. The R^2 was 0.6928 and the similarity S between $\hat{A}$ and the observed response Y was 0.2006. From Figure 8, we find that the estimated parameter surfaces are smoother than those in Figure 6 and Figure 7. But the smoother results are obtained at a price: the computational time is about five times as much as the time when Haar wavelet bases are used.

2.5 Computational Issues

2.5.1 Accelerating the Search of Optimal Penalization Term Using Subsampling

From Table 1, we know that our computational cost is about proportional to $O(n^{1.3})$. For a relatively big image, like a 512×512 image, this will require several hours to finish the computation. Most of computational time is spent on finding the optimal penalty term, because, for example, we need to go through a grid of lambda values sequentially to compute the BIC score for each λ value and find the λ_{opt}.

To improve the computational performance, we discuss in this subsection a subsampling method to accelerate the search for an optimal λ. The key idea is that we will randomly sample several sets of subimages from the original response and covariate images and use these subimages to estimate $\hat{\lambda}_{opt}$. Suppose

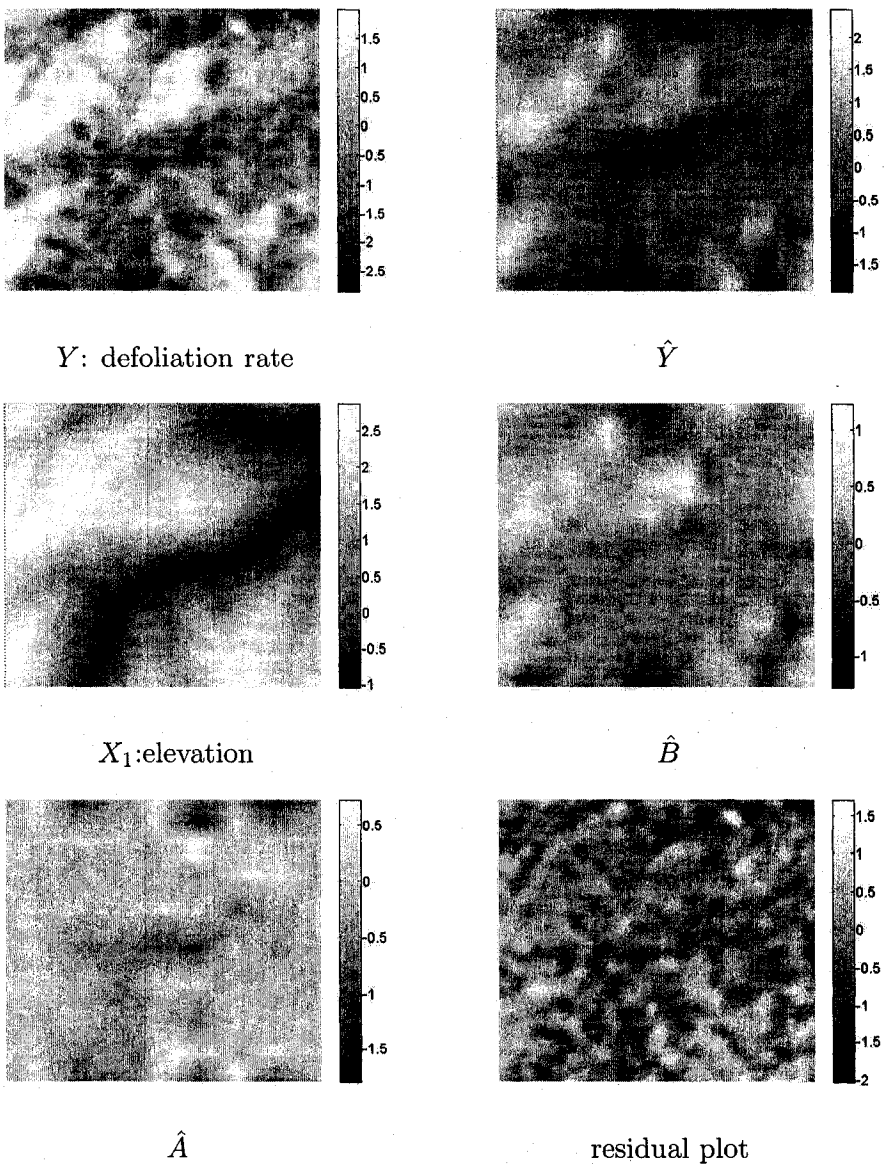


Figure 8: *Coif1 wavelet bases are used and the model is* $Y = A + X_1 \circ B + E$. *Elevation is the explanatory variable. Defoliation rate is the response.*

we randomly sample n sets of subimages (y_i, x_i) and we obtain the optimal $\lambda_{opt,i}$ for each set of subimages with

$$\hat{\lambda}_{opt,i} = \arg\min_{\lambda_i} BIC(\lambda_i, y_i, x_i). \tag{2.11}$$

The final estimated $\hat{\lambda}_{opt}$ can be obtained as the mean of $\lambda_{opt,i}$'s

$$\hat{\lambda}_{opt} = \frac{1}{n}\sum_{i=1}^{n} \hat{\lambda}_{opt,i}. \tag{2.12}$$

Subsampling can substantially accelerate the search of the optimal penalization term. Based on empirical work, the $\hat{\lambda}_{opt}$ obtained with subsampling is consistent with $\hat{\lambda}_{opt}$ obtained with complete images. Of course the conditions of the images such as signal to noise ratio, size and total number of subimages will affect the estimation accuracy.

Next, we discuss, intuitively, why subsampling works. Suppose Y is the complete image and suppose we have a sub-image $Y_i \subset Y$. For Y_i we have a penalty term λ_i and target function $f(\lambda_i)$

$$f(\lambda_i) = ||Y_i - \hat{Y}_i||_F^2 + \lambda_i||\alpha||_1 \tag{2.13}$$

where α is the parameter vector. For the complete image Y we have the penalty

term λ and target function $g(\lambda)$

$$g(\lambda) = ||Y - \hat{Y}||_F^2 + \lambda||\beta||_1 \tag{2.14}$$

where β is the associated parameter vector. Since $Y_i \subset Y$ and if the noise is homogenous, the sum of residual squares should be proportional to the image sizes:

$$||Y - \hat{Y}||_F^2 \approx c||Y_i - \hat{Y}_i||_F^2 \tag{2.15}$$

where c is a constant. Similarly, if the parameter surfaces are homogenous then $k(\lambda_i)$, the number of non-zero wavelet coefficients in sub-image Y_i, is proportional to the image size:

$$k(\lambda) \approx ck(\lambda_i) \tag{2.16}$$

With (2.15) and (2.16) we have

$$\begin{aligned}
\hat{\lambda}_{opt,i} &= \arg\min_{\lambda_i}\{\frac{||Y_i - \hat{Y}_i||_F^2}{N_i\hat{\sigma}^2} + \frac{\log(N_i)k(\lambda_i)}{N_i}\} \\
&= \arg\min_{\lambda_i}\{\frac{c||Y_i - \hat{Y}_i||_F^2}{cN_i\hat{\sigma}^2} + \frac{\log(N_i)ck(\lambda_i)}{cN_i}\} \\
&= \arg\min_{\lambda_1}\{\frac{c||Y_i - \hat{Y}_i||_F^2}{N\hat{\sigma}^2} + \frac{\log(N_i)ck(\lambda_i)}{N}\} \\
&\approx \arg\min_{\lambda}\{\frac{||Y - \hat{Y}||_F^2}{N\hat{\sigma}^2} + \frac{\log(N_i)k(\lambda)}{N}\}
\end{aligned} \tag{2.17}$$

where N_i is the number of the pixels in the subsample image. If we directly

search for $\hat{\lambda}_{opt}$ without subsampling, then we have

$$\hat{\lambda}_{opt} = \arg\min_{\lambda}\{\frac{||Y - \hat{Y}||_F^2}{N\hat{\sigma}^2} + \frac{\log(N)k(\lambda)}{N}\} \tag{2.18}$$

where N is the number of the pixels in the complete image. Comparing (2.17) with (2.18), we can see that subsampling will result in smaller $\hat{\lambda}_{opt}$ than the usual search method with whole images since subsampling gives less weight to the number of non-zero parameters. However when N_i and N are very large, $log(N_i)/log(N)$ will be close to 1. For instance, with a 1000×1000 image and a 250×250 subimage, $log(N_i)/log(N) = log(250^2)/log(10^6) \doteq 0.8$. Thus the mean of the $\hat{\lambda}_{opt,i}$'s should be close to $\hat{\lambda}_{opt}$ estimated without subsampling.

We next use a numerical example to illustrate how to use subsampling to accelerate the search for an optimal penalty term. In this example, the image size for the complete images is 200×200 and the size for each subsample is 64×64. (Another approach for accelerating the search, by using an FCLM with partial wavelet bases (Haar wavelet from level 3 to level 6) will be discussed in Section 2.5.2.) The estimated $\hat{\lambda}_{opt}$ with 10 subsamples was 1.1596, while the $\hat{\lambda}_{opt}$ directly obtained by computing BIC scores over the complete image was 1.5007. Although we used different search step sizes for $\hat{\lambda}_{opt}$ with the whole image and with the subsamples, we nonetheless obtained a good estimate of the optimal λ with subsampling. With 10 subsamples, we spent only 11.91 minutes finding $\hat{\lambda}_{opt}$, which is considerably shorter than 42.04 minutes needed without

subsampling. The results of this example are shown in Figure 9 and Figure 10.

2.5.2 Functional Concurrent Linear Models with Partial Wavelet Bases

So far we have discussed FCLMs with a complete set of wavelet bases. We can rewrite (2.3) explicitly with levels of wavelet bases:

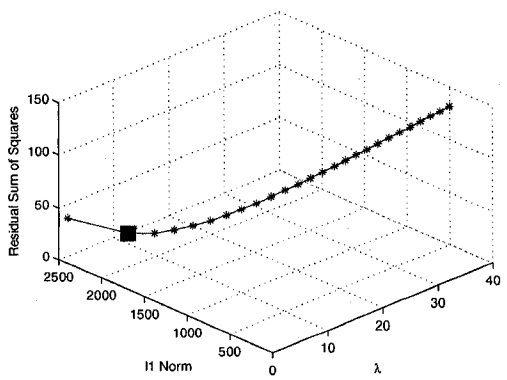
$$Y_i = \sum_{k=1}^{L}\sum_{j=1}^{N_k} v_{kj}\phi_{kj} + X_i \circ \left\{\sum_{k=1}^{L}\sum_{j=1}^{N_k} w_{kj}\phi_{kj}\right\} + E \tag{2.19}$$

$$= \sum_{k=1}^{L}\sum_{j=1}^{N_k} v_{kj}\phi_{kj} + \sum_{k=1}^{L}\sum_{j=1}^{N_k} w_{kj}\{X_i \circ \phi_{kj}\} + E \tag{2.20}$$

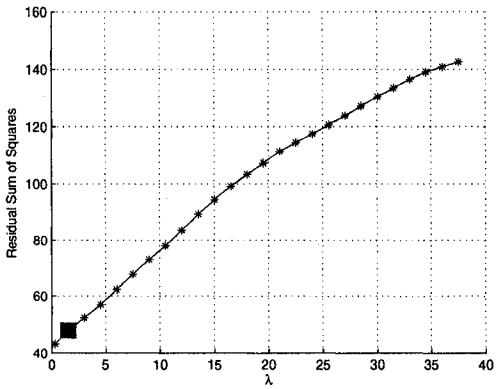
where L is the highest level of wavelet bases and N_k is the number of wavelet bases for level k. If we remove the first level or finest level from the model, then we will remove three quarters of the parameters and our model will become

$$Y_i = \sum_{k=2}^{L}\sum_{j=1}^{N_k} v_{kj}\phi_{kj} + \sum_{k=2}^{L}\sum_{j=1}^{N_k} w_{kj}\{X_i \circ \phi_{kj}\} + E. \tag{2.21}$$

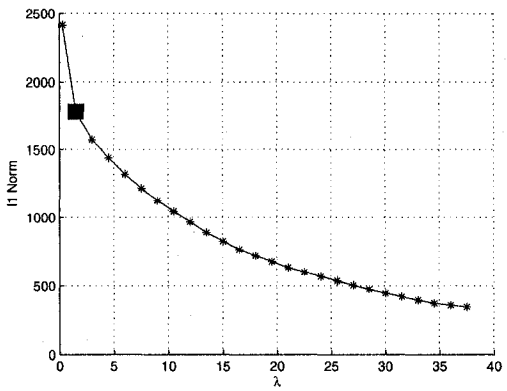
The model has become over-determined. By removing the first level bases, we can improve the computational speed and shrink the memory footprint considerably. If we have relative large images and believe it is safe to assume that the parameter surfaces do not have many small details, then we suggest using an FCLM with partial wavelet bases to accelerate the search for λ_{opt}.



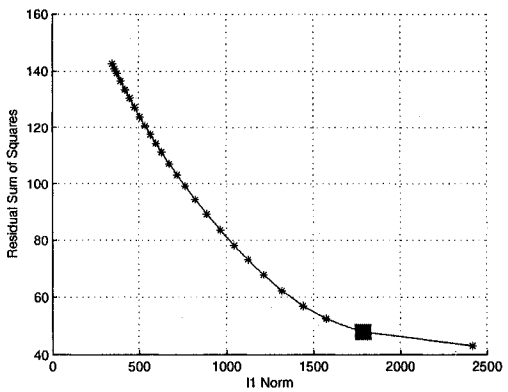
3-D plot of RSS vs. l_1 norm vs. λ



RSS vs. λ

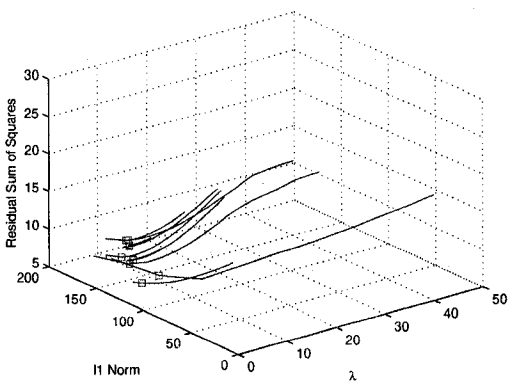


l_1 norm vs. λ

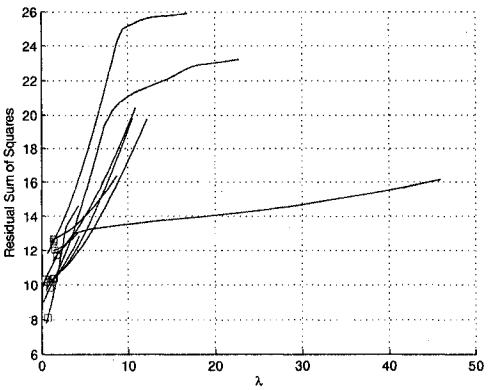


RSS vs. l_1 norm

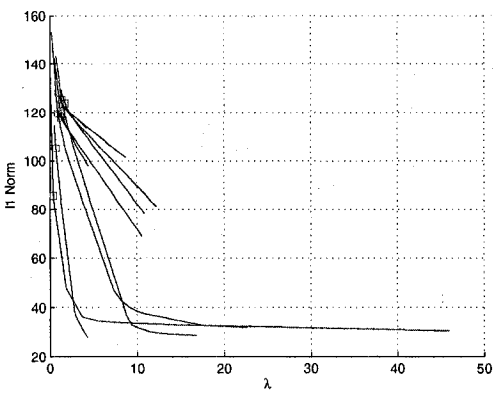
Figure 9: *Plots of the relationship of Residual Sum of Squares (RSS) vs.* l_1 *norm vs.* λ *for a direct search of* $\hat{\lambda}_{opt}$ *over the complete images. The* $\hat{\lambda}_{opt}$ *is marked by a solid square and* $\hat{\lambda}_{opt}$ *is* 1.5007. *The computational time was* 42.04 *minutes.*



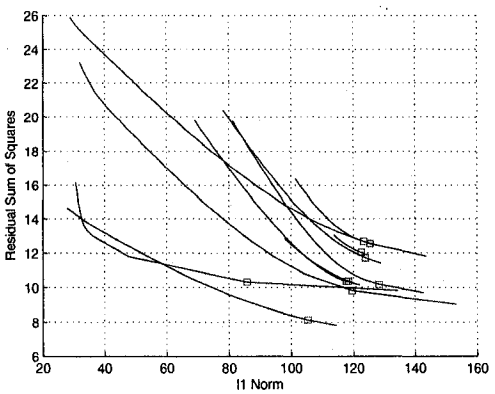
3-D plot of RSS vs. l_1 norm vs. λ



RSS vs. λ



l_1 norm vs. λ



RSS vs. l_1 norm

Figure 10: *Plots of the relationship of Residual Sum of Squares (RSS) vs.* l_1 *norm vs.* λ *for searching* $\hat{\lambda}_{opt}$ *with* 10 *subsamples.* $\hat{\lambda}_{opt,i}$ *is marked by a square on each line and* $\hat{\lambda}_{opt}$ *is* 1.1596. *The computational time was* 11.91 *minutes.*

Chapter 3

Missing Data

3.1 Functional Concurrent Linear Models With Missing Pixels

In applications of all kinds, missing data problems are ubiquitous, and certainly we face such problems when we use a Functional Concurrent Linear Model to do image-based regression with satellite images. For example, in a previous chapter, we applied our models to a set of images with gypsy moth defoliation rates as response and elevation as covariate. Every pixel was used in the analysis done there, but when we have a larger satellite image for defoliation rates, we may have highways, parking lots, agricultural fields, and lakes, etc. in the defoliation rate image, and defoliation rates in those areas are plainly meaningless. In other words, in many situations we may have some subregions in the satellites images that do not have useful information in them and should be excluded from the modeling process. This is equivalent to having missing pixels in some areas of the image. How can we fit our concurrent linear model in such cases?

First, it should be clear that we can divide our image data into two parts - an *incomplete part* and a *complete part.* In our image-based regression setup, the incomplete part includes all pixels that have missing or uninformative pixel values for at least one of the images under analysis, and the complete part includes all pixels that have valid pixel values for all of the images. Depending on the goals, usually there are two strategies for coping with missing data: the first one is that we only utilize the complete part of the data and ignore the incomplete data; the second one is that we use some form of imputation to fill in the missing pixels for the incomplete part of the data and then we fit our model pretending that we have no missing data.

For the gypsy moth defoliation problem, we can adopt the first strategy by introducing a missing data mask matrix M, a matrix with elements 1 or 0:

$$M_{i,j} = \begin{cases} 1 & \text{if pixel value exists at row } i \text{ col } j \\ 0 & \text{if pixel value is missing at row } i \text{ col } j. \end{cases} \tag{3.1}$$

We can think of M as the image "boundaries" inside the image. Then our functional concurrent linear model with missing data can be written as:

$$Y \circ M = A \circ M + B \circ X \circ M + E \tag{3.2}$$

and the estimate of $Y \circ M$ is

$$\hat{Y} \circ M = \hat{A} \circ M + \hat{B} \circ X \circ M \tag{3.3}$$

Like we have done in Chapter 2, we can expand A and B in (3.2) with a wavelet expansion:

$$Y \circ M = \left\{ \sum_{j=1}^{H} v_j \phi_j \right\} \circ M + \left\{ \sum_{j=1}^{H} w_j \phi_j \right\} \left\{ X \circ M \right\} + E \tag{3.4}$$

and after rearranging (3.4) we get

$$Y \circ M = \sum_{j=1}^{H} v_j \left\{ \phi_j \circ M \right\} + \sum_{j=1}^{H} w_j \left\{ \phi_j \circ M \circ X \right\} + E. \tag{3.5}$$

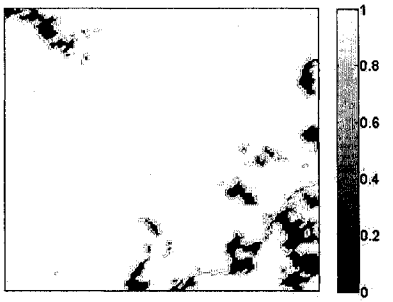
Equation (3.5) is similar to (2.3) in Chapter 2. With (3.5), we can use the same large scale l_1 constrained LSE to estimate v_j and w_j as in Chapter 2. Also, if there is no valid pixel value inside the support of ϕ_j, or if $\phi_j \circ M$ is equal to a zero matrix, then the corresponding v_j and w_j will be zero. As long as there are some valid pixel values inside the support of ϕ_j, $\phi_j \circ M$ will not be a zero matrix and the values of v_j and w_j will be determined by those valid pixel values. The "new" wavelet base $\phi_j \circ M$ is quite flexible and can adapt well to different configurations of missing patterns and different missing proportions.

To illustrate this, in Figure 11, we have 128×128 images for missing mask, defoliation rate and elevation. It can be seen in the original defoliation rate

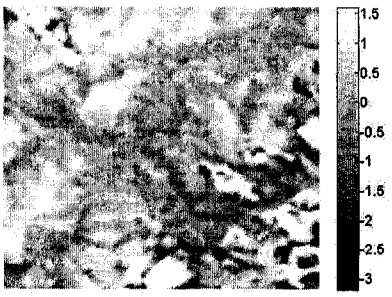
image Y that some unusually bright areas in the northwest and southeast corners of the images perhaps resemble lakes and are therefore not related to the gypsy moth defoliation study. We want to exclude those areas from the study.

In Figure 12, we also show the masked defoliation rate image $Y \circ M$, the masked elevation image $X \circ M$, the estimated defoliation rate $\hat{Y} \circ M$, estimated constant surface $\hat{A} \circ M$, estimated slope surface $\hat{B} \circ M$ and residual surface from applying the above procedure. We still use Haar wavelet bases for this example with the smallest base support being 2×2 and the largest wavelet base support being 128×128. The computational time was 5.87 minutes, λ_{opt} was 6.6404, and λ_{max} was 72.1786. The overall R^2 was 0.5335 which was a bit lower than that in the example from Chapter 2 without missing data. The similarity between $Y \circ M$ and estimated constant surface $\hat{A} \circ M$ was 0.0885 which means we did not have a degenerate case.

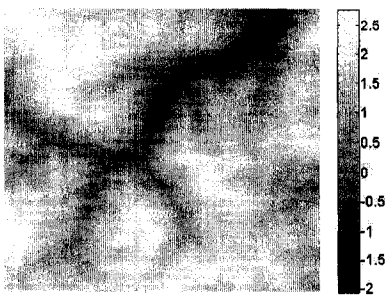
To examine the effects of basis family choice, we next switch from Haar wavelet bases to Coif1 wavelet bases with the smallest base support being 6×6 and the largest wavelet base support being 76×76. In this case the computational time was 52.38 minutes, λ_{opt} was 5.6131, and λ_{max} was 58.4698. The overall R^2 was 0.6154 which was higher than that with Haar wavelet bases. The similarity between $Y \circ M$ and estimated constant surface $\hat{A} \circ M$ was 0.1369 which meant we did not have a degenerate case for these wavelets, either. The results are shown in Figure 13. The fitted $\hat{Y} \circ M$ is smoother than that obtained with Haar wavelet bases. However, the computational time with Coif1 wavelet



M: missing mask

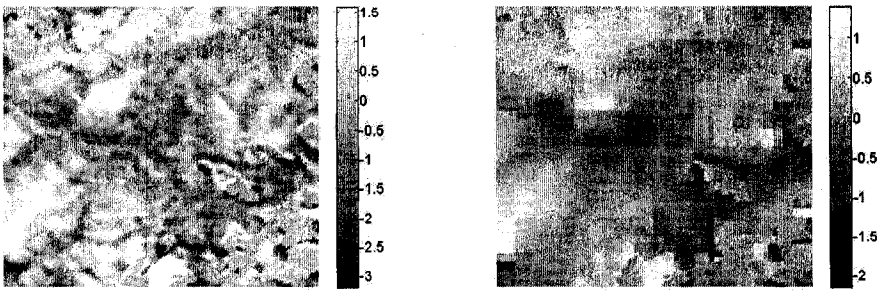


Y: original defoliation rate image

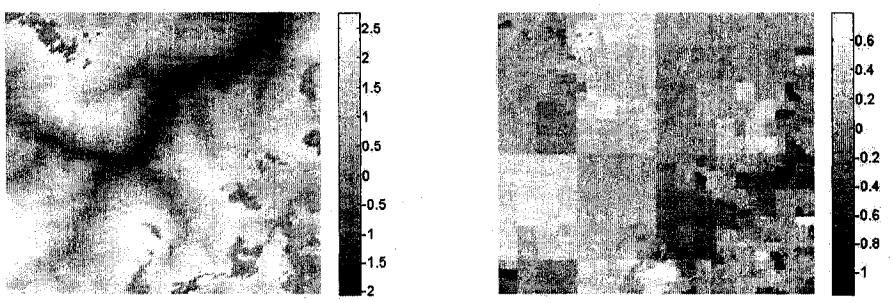


X: original elevation image

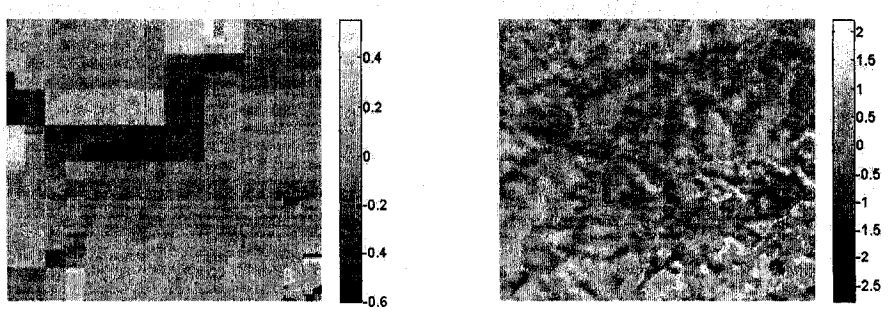
Figure 11: *Missing mask M, original X elevation image and original Y defoliation rate image. In the missing mask M, dark areas stand for the missing subregions. In Figure 12, we will show the masked defoliation rate image $Y \circ M$, the masked elevation image $X \circ M$.*



$Y \circ M$: masked defoliation rate $\hat{Y} \circ M$



$X \circ M$:masked elevation $\hat{B} \circ M$



$\hat{A} \circ M$ residual plot

Figure 12: *Haar wavelet is used. The model is* $Y \circ M = A \circ M + B \circ X \circ M + E$ *. Elevation is the explanatory variable. Defoliation rate is the response. Note that the area with missing data is masked off in elevation and defoliation rate image. The overall* R^2 *was* 0.5335 *which was a bit lower than that in the example from Chapter 2 without missing data. The similarity between* $Y \circ M$ *and estimated constant surface* $\hat{A} \circ M$ *was* 0.0885 *which meant we did not have a degenerate case.*

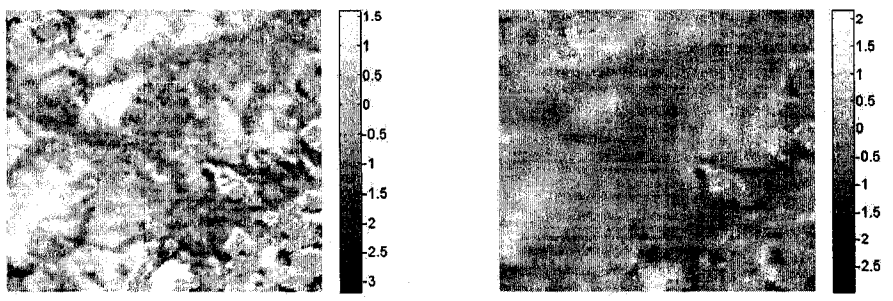
bases is considerably longer than that with Haar wavelet bases.

The above two examples illustrate how an FCLM can be used to utilize the complete part of the data and ignore the incomplete part, at least, if we are only interested in using one mask M for all of the images involved in the modeling process. However, we often face a more challenging situation where different images may have different missing areas. In other words, the response image and covariate images may each have different missing regions which may or may not overlap. Can we still apply an FCLM to such data? Can our algorithm automatically select relevant non-missing regions in the covariate images to fit the non-missing regions in the response image? This may look like a daunting task if we do not allow imputation for each image. In the next section, we will use an example to illustrate that we can indeed fit an FCLM for images with different missing subregions.

3.2 Interpolation and Pixel Selection: Filling in the Missing Gaps for Scan Line Corrector Off (SLC-off) Landsat 7 Images

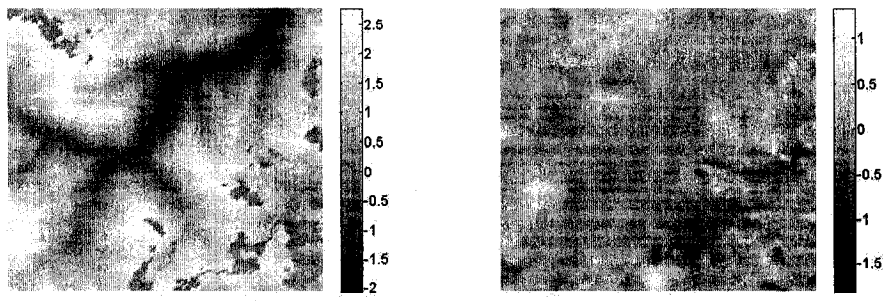
3.2.1 Background on Landsat 7 Images

Landsat 7 was launched on April 15, 1999 and was equipped with an Enhanced Thematic Mapper Plus (ETM+) instrument. However, on May 31, 2003 the



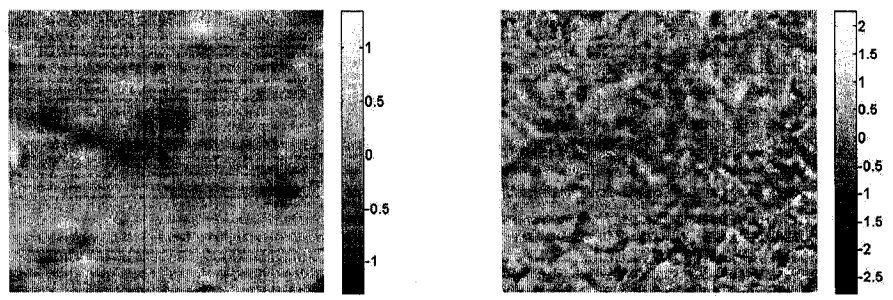
$Y \circ M$: masked defoliation rate

$\hat{Y} \circ M$



$X \circ M$:masked elevation

$\hat{B} \circ M$



$\hat{A} \circ M$

residual plot

Figure 13: *Coif1 wavelet is used. The model is* $Y \circ M = A \circ M + B \circ X \circ M + E$. *Elevation is the explanatory variable. Defoliation rate is the response. Note that the area with missing data is masked off in elevation and defoliation rate image. The overall* R^2 *was* 0.6154 *which was a bit higher than that with Haar wavelet bases. The similarity between* $Y \circ M$ *and estimated constant surface* $\hat{A} \circ M$ *was* 0.1369 *which meant we did not have a degenerate case. The fitted* $\hat{Y} \circ M$ *is smoother than that obtained with Haar wavelet bases. However, the computational time with Coif1 wavelet bases is considerably longer than that with Haar wavelet bases.*

Scan Line Corrector (SLC) in the ETM+ instrument failed [30]. The function of the SLC is to compensate for the forward motion of the satellite, and the failure of the SLC creates missing stripes in images acquired by Landsat 7 (see Figure 14). Unfortunately there is no immediate replacement planned for Landsat 7 and researchers still rely on Landsat imagery to conduct research. As a result there is a substantial interest in filling in the missing stripes in Landsat 7 images. Currently NASA uses several methods to correct the SLC-off mode Landsat 7 images, including an earlier version called a "localized linear histogram matching algorithm" which only incorporates a SLC-on image into a SLC-off image, and a later version called an "Adaptive Local Linear Histogram Adjustment (ALLHA)" which utilizes multiple SLC-off and SLC-on images to fill in the missing stripes in a SLC-off image[25, 26].

We want to try our functional concurrent linear model to see whether our model is capable of filling in the stripes and achieve some improvement over existing methods. But, in a broader perspective, we also want to exam how an FCLM interpolates unobserved subregions between two observed subregions for the response image and how an FCLM reacts to the quality of each covariate image when there are different missing subregions or cloud covers. These two attributes are very important to real world applications. First, there are many practical reasons for the interpolating capability. Often for a large area during a certain period of time satellites may not be able to fully cover the area perhaps due to very poor acquisition conditions in some subregions. In the SLC-off

case, the coverage for some subregions is just simply unavailable due to a mechanical failure. The point is there are many unexpected incidents that will obstruct a complete perfect acquisition of the area we want to study. However, a complete global view of the whole area including uncovered subregions is what many researchers desire to obtain since it will give them uninterrupted images. Sometimes utilizing available images to interpolate their neighboring areas is the major goal of the research because it may be very expensive or difficult to achieve a complete full coverage of the area under interest. In this case, being able to interpolate the pixel values in uncovered subregions is required. Thus, a regression model for spatial images which provides an interpolating function will be useful in many practical situations.

Secondly, adaptation to the quality of each covariate image is actually a variable selection function based on the quality of pixel values. Pixel values in the missing subregions of covariate images are of no value in terms of interpretation. In some cases, like SLC-off images, these missing pixels can be clearly marked in a missing mask matrix. However, even if we have the locations of the missing pixels, different missing patterns in different images still complicate the pixel selection. More difficult situation arises when missing pixels are unmarked. For example, cloud covers represent missing subregions which cannot be easily marked. Moreover, sometimes we cannot draw a clear line between missing pixels and non-missing pixels. For instance, sun glares, snow covers or other transient changes in the landscape and/or in the atmosphere will make

some regions in satellite images become less informative than those in other subregions.

Putting this together, we may have marked missing pixels, unmarked missing pixels, low-quality pixels, and high-quality pixels all mingled together in a single satellite image. A useful regression models for spatial images should be able to select the best possible combination of pixel values across all covariate images to predict the pixel values in the response image. As we will see, an FCLM implemented with a variable selection procedure can provide good interpolation and pixel selection capabilities despite a simple looking model.

In Section 3.2.2, we will first show two examples using a related SLC-on image to fill in the missing stripes in a target SLC-off image. These two examples mainly demonstrate how an FCLM can interpolate the pixel values between two known regions. Next, we will show four examples using multiple SLC-off and SLC-on images to fill in the missing stripes in a target SLC-off image. These examples will show how an FCLM automatically selects useful information in different covariate images. For every example in Section 3.2.2, we apply the scene sub-sampling mentioned in Section 2.5.1 to accelerate the search for an optimal penalization term. Also, partial wavelet bases models will be used since we have relatively large images for each example.

We use real satellite images in all examples but we generate all SLC-off images from SLC-on images by masking off stripes. The reason for using simulated

SLC-off images is to make it easy to check our results against the original SLC-on images. Similar to real SLC-off images, in our simulated missing data the typical width of missing stripes is about 7 to 9 pixels. For these simulations, the image sizes are all 200×200 unless otherwise mentioned. Images in our examples are downloaded from the U.S. Geological Survey Earth Explorer and the link is http://edcsns17.cr.usgs.gov/EarthExplorer/. These images are for the region around Phoenix, AZ and were captured on different dates from 2000 to 2001. All computations were done with MATLAB version 7.1 and we still use a commodity laptop with a 2 GHz Intel Core 2 T7200 CPU and $2G$ memory.

3.2.2 Using FCLMs to Fill in Missing Stripes

Interpolation: Using a SLC-on image to fill in a SLC-off image

First, we will discuss the basic case — using one SLC-on image to fill in the missing stripes in another SLC-off image. In Figure 14, the SLC-off target image Y with missing stripes is the image we want to "fix." This target image Y actually is generated from the complete SLC-on image Y_0, and we have

$$Y = Y_0 \circ M$$

where M is the missing data matrix defined in (3.1). Another SLC-on image X will be treated as a covariate in our model. This X image covers the same area as Y does, but comes from a different date. The target image Y was acquired

on May 24, 2001 and the covariate image X was acquired on May 21, 2000. By choosing two images acquired on similar dates in different years, we hope to minimize the seasonal effects and maximize the similarity between the target image Y and covariate image X.

Now we can write our model as:

$$Y_0 \circ M = Y = A \circ M + B \circ X \circ M \tag{3.6}$$

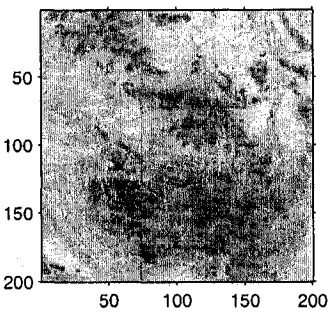
where M is the missing data mask matrix defined in (3.1). After we have calculated $\hat{A}$ and $\hat{B}$, then the estimated *complete* target image $\hat{Y}_0$ is

$$\hat{Y}_0 = \hat{A} + \hat{B} \circ X \tag{3.7}$$

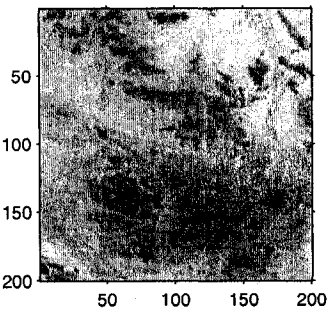
With (3.7), if we assume the concurrent linear relationship remains true in the missing regions, then $\hat{Y}_0$ includes the estimated missing stripes. By combining the estimated stripes in $\hat{Y}_0$ with the non-missing parts in the target image Y, the final filled image $\hat{Y}_{fill}$ can be expressed as

$$\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M \tag{3.8}$$

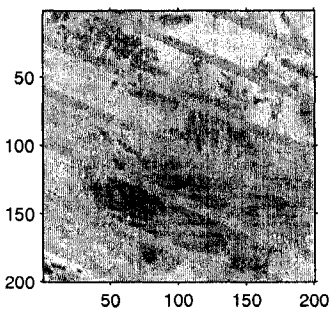
where $\widetilde{M}$ is equal to $J - M$ and J is a matrix whose elements are all equal to 1. As noted, since we have large images, to speed up the computation we accelerate the optimal penalty term search with scene sub-sampling and use an



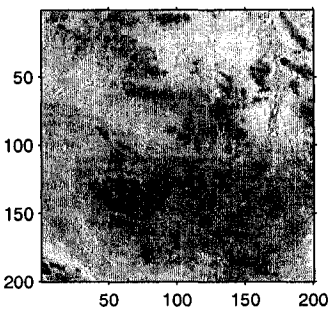
original SLC-on image Y_0



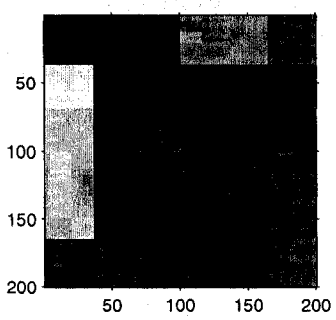
covariate X image



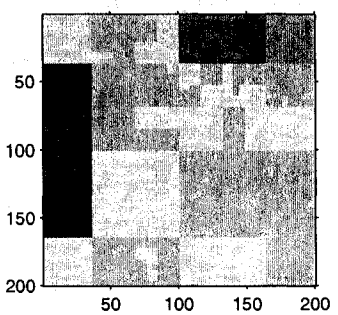
simulated SLC-off target image $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$



$\hat{B}$

Figure 14: *Filling in the gaps. Missing Stripes in* $\hat{Y}_{fill}$ *are from* $\hat{Y}_0 = \hat{A} + \hat{B} \circ X$ *and non-missing parts are from simulated SLC off target image* Y. $\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$. *For our model, the partial* R_p^2 *was* 0.9516 *and* S_p, *the partial similarity between* $\hat{A}$ *and* Y_0 *for the missing regions was* -0.5632.

FCLM with partial wavelet bases. Specifically, we randomly extracted 5 sub-images, estimated the $\lambda_{opt,i}$ for each one and set the final λ_{opt} to be the mean of all $\lambda_{opt,i}$'s. We use Haar basis level 3 to level 6 in our model which means that our smallest base support is 8×8 and our largest base support is 64×64. To evaluate the filled stripes, we define a partial R_p^2 and S_p for the fitted pixel values in missing regions as

$$R_p^2 = 1 - \frac{||Y_0 \circ \widetilde{M} - \hat{Y}_0 \circ \widetilde{M}||_F^2}{||Y_0 \circ \widetilde{M} - \bar{Y} \circ \widetilde{M}||_F^2}$$

and

$$S_p = 1 - \frac{||Y_0 \circ \widetilde{M} - \hat{A} \circ \widetilde{M}||_F^2}{||Y_0 \circ \widetilde{M}||_F^2}.$$

The results are shown in Figure 14. In this case λ_{opt} was 1.46 and λ_{max} was 41.57. The computational time was 8.91 minutes. For our model, the partial R_p^2 was 0.9516 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.5632. For the Adaptive Local Linear Histogram Adjustment (ALLHA), the partial R_p^2 was 0.9614 and S_p was -0.9148. S_p's were negative for these two models. A negative S_p means that the estimated constant term surface $\hat{A}$ is very far from the Y_0 in the missing subregions. Because the FCLM limits the l_1 norm of the parameter surfaces and an $\hat{A}$ too far from the Y_0 can cost a lot of l_1 norm, the S_p from the FCLM will generally be smaller than that from the ALLHA, which is exactly the situation we have observed in this example. The high R_p's for the FCLM and the ALLHA indicates the fit

is good for both models. For this set of data, the FCLM seems to successfully interpolate the missing subregions between the two non-missing subregions, and although we do not show specific results here, we have observed that our results are comparable to the results from the adaptive linear histogram adjustment algorithm. By observation, we have a very good $\hat{Y}_{fill}$ and do not see any visible artifacts in $\hat{Y}_{fill}$ from Figure 14.

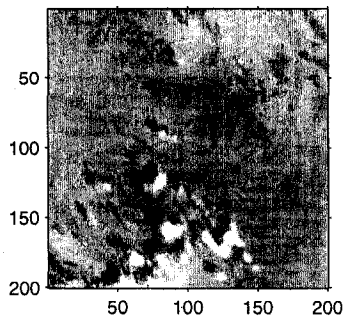
Next, we consider what happens if we have clouds in the target SLC-off image Y. We expect in that situation that we will have difficulties filling in the missing stripes in the region covered by clouds. In Figure 15, the simulated SLC-off target image Y with some cloud cover in the bottom was acquired on September 29, 2001 and the cloud free covariate image X was acquired on September 26, 2000. Again we fit the model in (3.6). In this case λ_{opt} was 1.95 and λ_{max} was 37.39, and the computational time was 8.09 minutes. For our model, the partial R_p^2 was 0.5731 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.1301. For the ALLHA, the partial R_p^2 was 0.5249 and S_p was -0.9564. Here we also observed that S_p from the ALLHA is smaller that the that from the FCLM due to the l_1 norm constraint of the FCLM. Comparing the two, our model was slightly better in terms of R_p^2. However R_p^2's for this example are both much lower than those for the previous example which has a cloud free response image. The filled image $\hat{Y}_{fill}$ in Figure 15 has some artifacts in the bottom where clouds cover some area.

Areas covered by clouds in the target image are additional missing parts

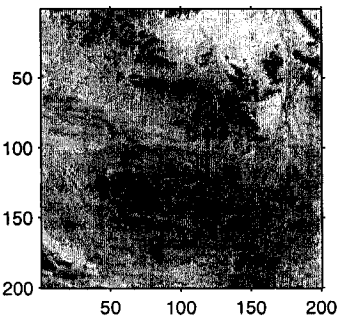
besides missing stripes. However, the key problem is we do not have the missing mask matrix for the randomly missing pixels caused by clouds. Thus in applying our methods, clouds will be treated as the useful information in the scene and the normal parameter estimation procedure will be applied to regions covered by clouds. As a result, the estimated constant surface $\hat{A}$ clearly contains some "clouds" in the bottom part. On the other hand, the "clouds" in $\hat{A}$ shows that our algorithm does treat the cloudy region differently and adapts to use the constant parameter surface to estimate the regions covered by clouds in the response image.

Pixel Selection: Using Multiple SLC-off and SLC-on images to fill in a SLC-off image

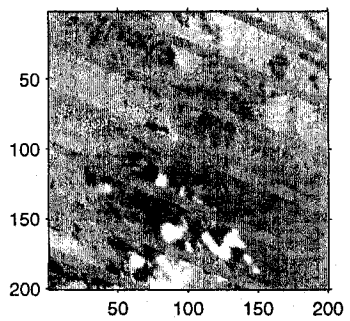
In the previous section, we use one SLC-on image to fill in the missing stripes in a target SLC-off image. However, SLC-on images were all acquired before May 31, 2003 and some scenes covered by SLC-on images may have changed substantially. Thus, sometimes we may not have a suitable SLC-on image for a scene but we may have multiple good SLC-off images available for the same scene. In that case, we have to rely on SLC-off images or a combination of SLC-off images and SLC-on images to fill in the stripes in the target SLC-off image. Although this will require dealing with extra missing stripes in the covariate images and potentially even more incomplete data, we have the potential advantage of more current SLC-off images available. In addition, we generally are



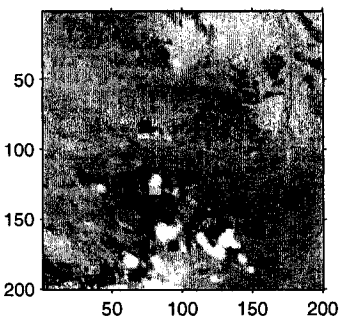
original SLC-on image Y_0



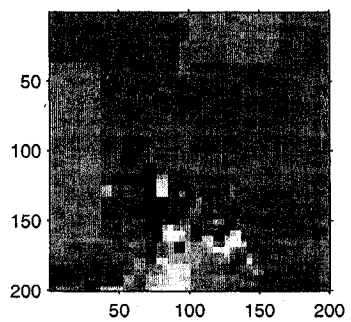
covariate X image



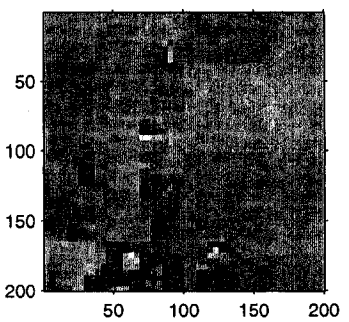
simulated SLC-off target image $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$



$\hat{B}$

Figure 15: *Filling in the gaps. The clouds in the Y cause trouble for our algorithm. $\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$. The partial R_p^2 was 0.5731 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.1301. The estimated constant surface $\hat{A}$ clearly contains some "clouds" in the bottom part.*

able to take advantage of having more information with which to fill in images.

To utilize SLC-off images as covariate images, we have to introduce a missing data mask matrix for each SLC-off image in additional to the initial missing mask matrix M for the response image. In our next example, we will utilizes two SLC-off images to fill in the missing stripes in the target SLC-off image. The corresponding model is

$$Y_0 \circ M = Y = A \circ M + B_1 \circ M \circ X_1 \circ M_1 + B_2 \circ M \circ X_2 \circ M_2 \qquad (3.9)$$

where M, M_1 and M_2 are the missing data mask matrices for the response and covariate images. Here we assume that M, M_1 and M_2 are not the same. After this model has been fit, the estimated *complete* target image $\hat{Y}_0$ will be

$$\hat{Y}_0 = \hat{A} + \hat{B}_1 \circ X_1 \circ M_1 + \hat{B}_2 \circ X_2 \circ M_2 \qquad (3.10)$$

and (3.8) can be used to compute the filled image $\hat{Y}_{fill}$. Of course, if $\widetilde{M}_1 \circ \widetilde{M}_2 \neq O$ where O is a zero matrix, then we must have some areas that still cannot be filled in.

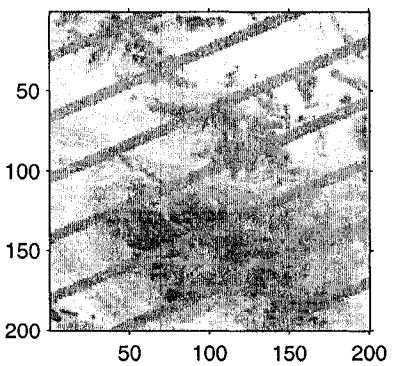
In Figure 16 and Figure 17, we have three simulated SLC-off images. One covariate, $X_1 \circ M_1$, was acquired on May 05, 2000 and the other covariate, $X_2 \circ M_2$, was acquired on April 19, 2000. The acquisition date for the SLC-off target image $Y \circ M$ was May 21, 2000. The acquisition dates are very close for the three images in this example and we hope close acquisition dates will

minimize the scene differences.

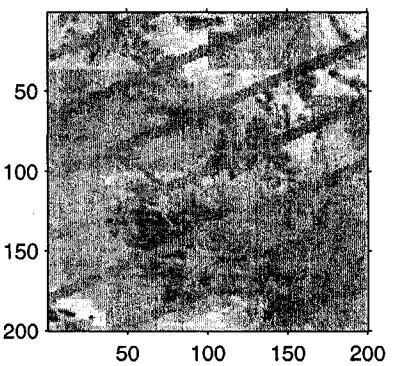
For this case, λ_{opt} was 3.6410, λ_{max} was 46.3739, and the computational time was 9.65 minutes. For our model, the partial R_p^2 was 0.7597 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was 0.3126. A relatively high R_p^2 and a small S_p means that we have a good fit and do not have a degenerated case. From Figure 16, we notice that one SLC-off image $X_1 \circ M_1$ fills in the major part of the missing stripes in the target image. We conjecture that this occurs because the missing stripes of $X_1 \circ M_1$ overlap less with the missing stripes of Y than the missing stripes of $X_2 \circ M_2$ do. In a sense, our model has done pixel selection automatically for this example to produce a filled image $\hat{Y}_{fill}$ that looks good.

In Figure 17, we can see that filled image $\hat{Y}_{fill}$ has some parts still missing. This is caused by the missing stripes in the SLC-off covariate images. We can alleviate this phenomenon by increasing the number of covariate images. In our next example, we use three simulated SLC-off covariate images, which were acquired separately on May 21, 2000, May 12, 2000, and April 19, 2000. The target SLC-off image was acquired on May 21, 2000. Each image has different (simulated) missing stripes which overlap each other. All covariate images and the target images are shown in Figure 18 and Figure 19.

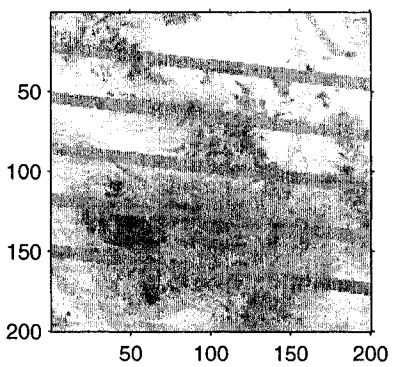
For this example, λ_{opt} was 2.2836 and λ_{max} was 265.5230. The computational time was 28.9675 minutes. For our model, the partial R_p^2 was 0.8478 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.2426. A



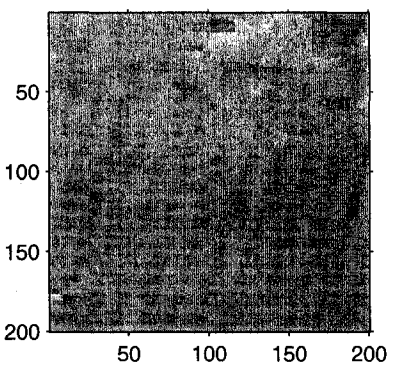
simulated SLC-off $X_1 \circ M_1$



$\hat{B}_1 \circ X_1 \circ M_1$

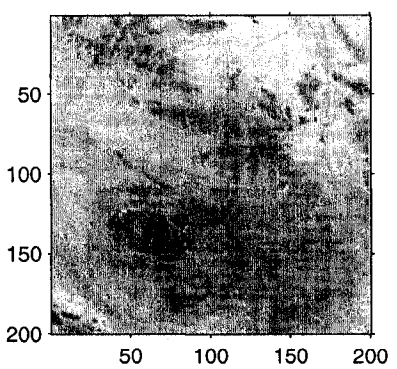


simulated SLC-off $X_2 \circ M_2$

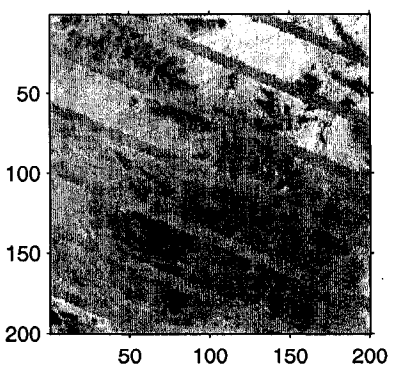


$\hat{B}_2 \circ X_2 \circ M_2$

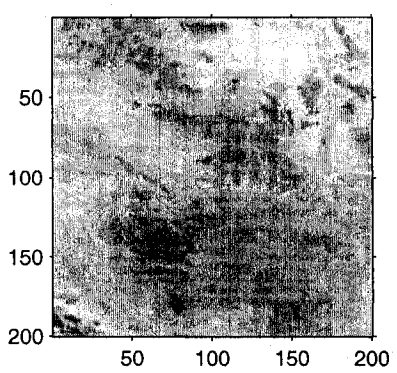
Figure 16: *Filling in the gaps with 2 SLC-off images.* $X_1 \circ M_1$ *explains most part of the target image. The reason we see this may be that the missing stripes of* $X_1 \circ M_1$ *overlap less with the missing stripes of* Y *than the missing stripes of* $X_2 \circ M_2$ *do.*



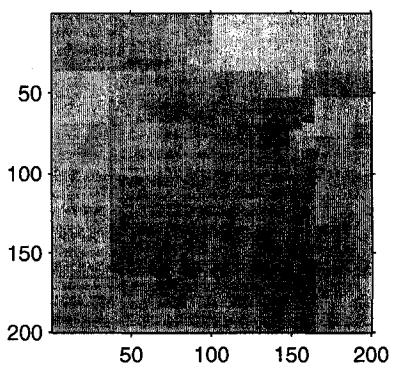
original SLC-on image Y_0



simulated SLC-off target image $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$

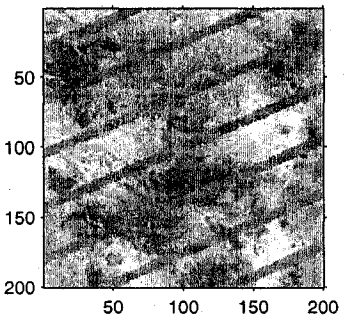
Figure 17: *Filling in the gaps with 2 SLC-off images. Missing Stripes in* $\hat{Y}_{fill}$ *are from* $\hat{Y}_0 = \hat{A} + \hat{B}_1 \circ X_1 \circ M_1 + \hat{B}_2 \circ X_2 \circ M_2$ *and non-missing parts are from simulated SLC off target image* Y. $\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$. *The partial* R_p^2 *was* 0.7597 *and* S_p, *the partial similarity between* $\hat{A}$ *and* Y_0 *for the missing regions was* 0.3126. *Overall our model has done pixel selection automatically for this example and the filled image* $\hat{Y}_{fill}$ *actually looks good.*

negative S_p means that the estimated constant surface $\hat{A}$ is further away from the Y_0 than that from the example with only two SLC-off covariate images. The results are shown in Figure 18 and Figure 19.

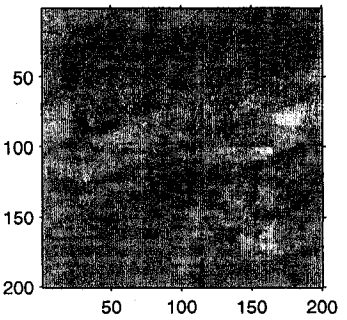
By adding one additional covariate image, we improved R_p^2 from 0.7597 to 0.8478 and have a visibly better filled image $\hat{Y}_{fill}$ in Figure 19. We note from Figure 18 that two of the three SLC-off images play the major role in filling in the missing stripes in target image. It is interesting to see that our algorithm adaptively choose $X_1 \circ M_1$ to describe the middle section of the response and $X_2 \circ M_2$ to explain the top and bottom part of the response. To the eye, it is easy to see that $X_1 \circ M_1$ is better in the middle and $X_2 \circ M_2$ is better in the top and bottom. However, it is encouraging that our algorithm appears to automatically select pixels across the three covariate images with different missing subregions.

In our next example, we will use two simulated SLC-off images and one SLC-on image to fill in the stripes in the target SLC-off image. We replace $X_3 \circ M_3$ in Figure 18 with a new SLC-on X_2 in Figure 20 which has some clouds in the left half part. The acquisition date for the SLC-on image was September 26, 2000, and the two SLC-off images $X_1 \circ M_1$ and $X_2 \circ M_2$ in Figure 20 are the same as those in Figure 18, which were acquired separately on May 05, 2000 and May 21, 2000. The SLC-off target image in Figure 21 is the same as the target image in Figure 19 and its acquisition date was May 21, 2000.

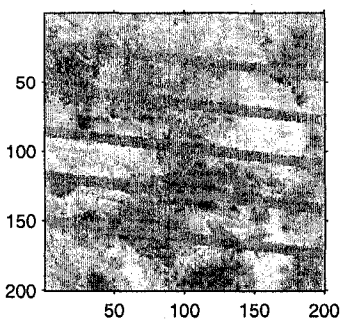
For this example, λ_{opt} was 2.9820 and λ_{max} was 265.5230; the computational



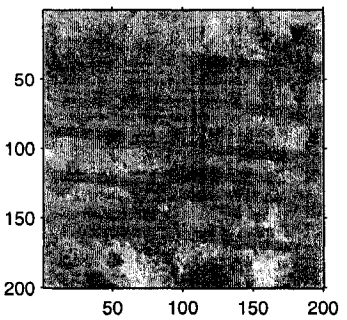
simulated SLC-off $X_1 \circ M_1$



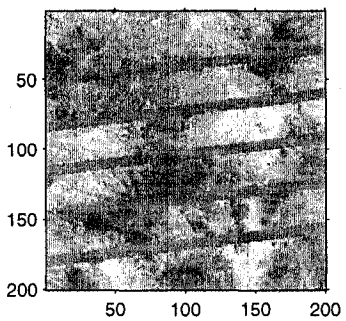
$\hat{B}_1 \circ X_1 \circ M_1$



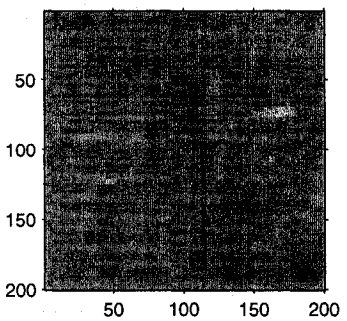
simulated SLC-off $X_2 \circ M_2$



$\hat{B}_2 \circ X_2 \circ M_2$

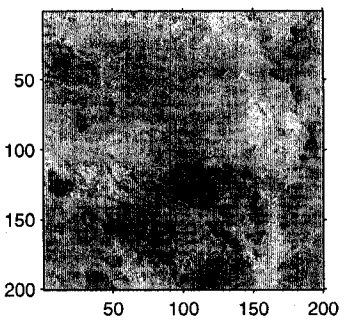


simulated SLC-off $X_3 \circ M_3$

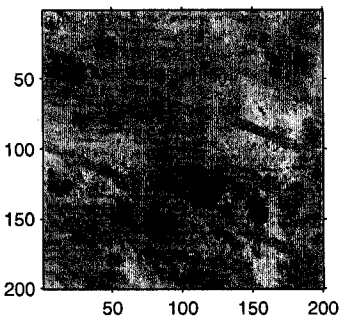


$\hat{B}_3 \circ X_3 \circ M_3$

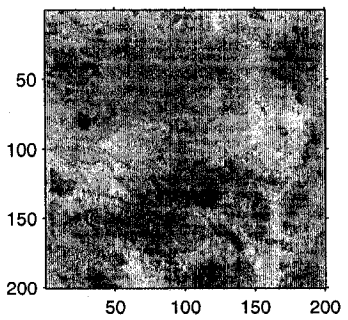
Figure 18: *Filling in the gaps with* 3 *SLC-off images.* $X_1 \circ M_1$ *contributes to the middle part and* $X_2 \circ M_2$ *contributes the top and bottom part.*



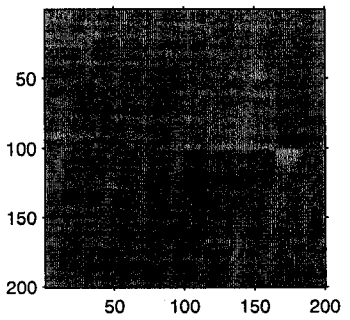
original Y_0



Simulated SLC-off $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$

Figure 19: *Filling in the gaps with* 3 *SLC-off images. Missing Stripes in* $\hat{Y}_{fill}$ *are from* $\hat{Y}_0 = \hat{A} + \hat{B}_1 \circ X_1 \circ M_1 + \hat{B}_2 \circ X_2 \circ M_2 + \hat{B}_3 \circ X_3 \circ M_3$ *and non-missing parts are from simulated SLC off target image* Y*.*$\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$*. The partial* R_p^2 *was* 0.8478 *and* S_p*, the partial similarity between* $\hat{A}$ *and* Y_0 *for the missing regions was* -0.2426*. By adding one extra covariate image, we improved* R_p^2 *from* 0.7597 *to* 0.8478 *and have a visibly better filled image* $\hat{Y}_{fill}$ *in Figure 19.*

time was 33.85 minutes. For our model, the partial R_p^2 was 0.8805 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.4775. The results are shown in Figure 20 and Figure 21.

We can see that we slightly improved R_p^2 after replacing $X_3 \circ M_3$ with a new SLC on image X_3. However, it is interesting to see that our model automatically selects X_3's cloud-free right half to fill in the stripes although the cloud cover in X_3 is not marked. In Figure 20, $X_2 \circ M_2$ still contributes the top and bottom part of the filled image. But $X_1 \circ M_1$ and the new SLC-on X_3 together explain the middle part of the filled image. This example shows that even with very different missing mechanisms and patterns in different covariate images, our algorithm seems to adapt well to the situation and automatically selects the best useful region in each covariate to fill in the stripes in the target image.

In the last example, we still use two simulated SLC-off images and one SLC-on image to fill in the stripes in the target SLC-off image. But the SLC-on covariate X_3 in Figure 22 is mostly covered by clouds. We hope that our algorithm will not use pixels in X_3 since most of pixels in X_3 are either distorted by the thin cloud cover or totaly covered by the thick cloud cover. The acquisition date for the SLC-on image was September 26, 2000, and the two SLC-off images $X_1 \circ M_1$ and $X_2 \circ M_2$ in Figure 22 were acquired separately on May 05, 2000 and May 21, 2000. The SLC-off target image in Figure 23 was acquired on May 21, 2000. These images cover some urban areas of Phoenix, AZ. Urban areas are coarser than the mountain area covered in the previous example and potentially

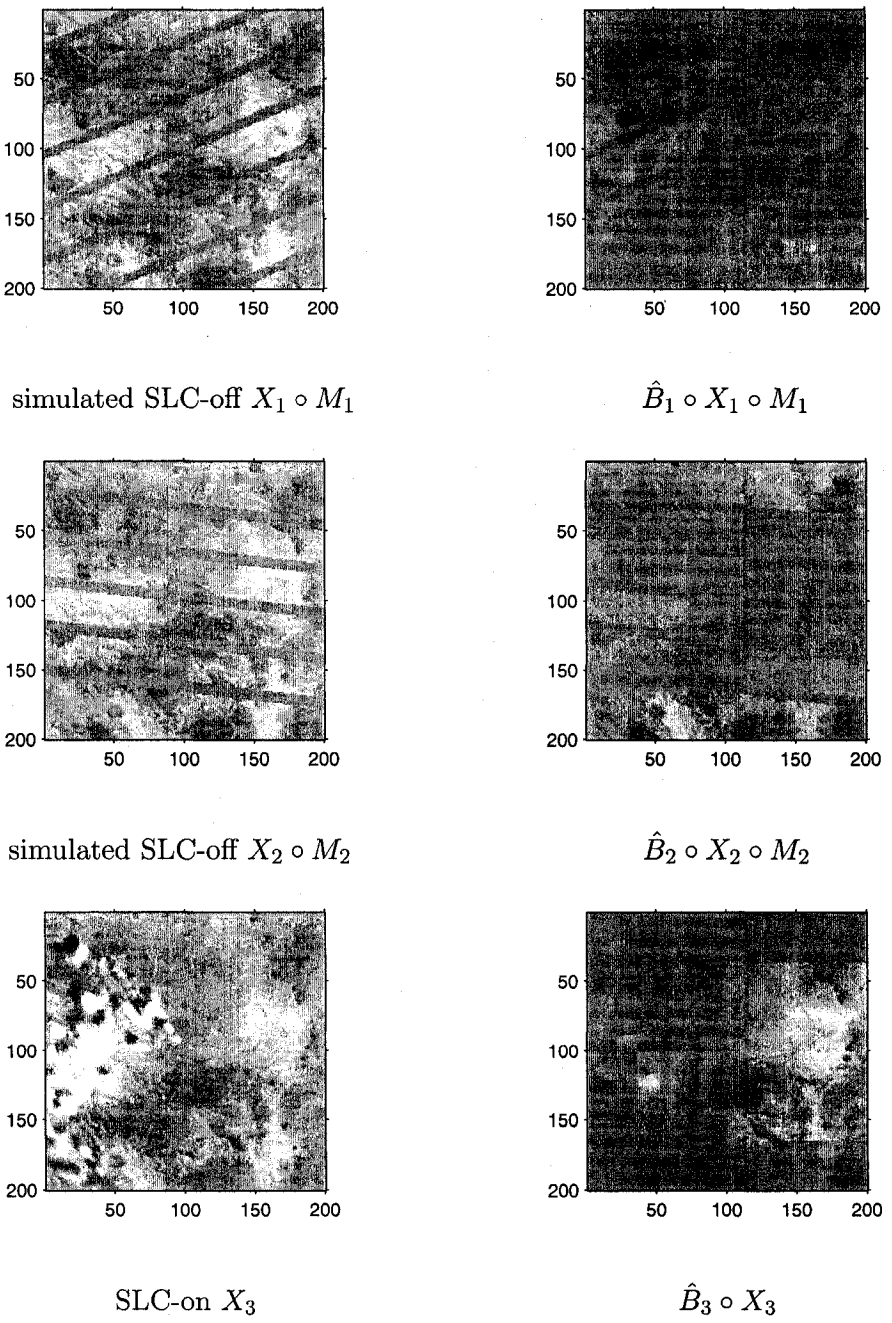
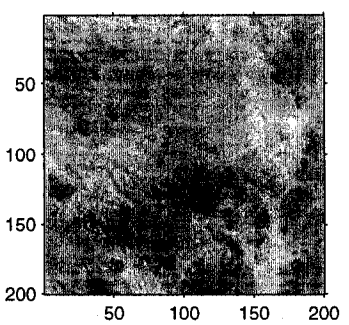
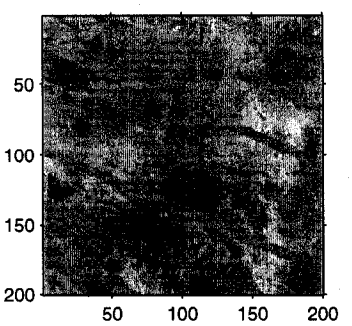


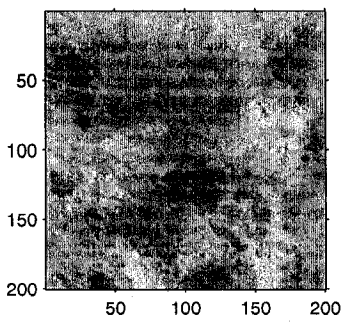
Figure 20: *Filling in the gaps with 2 SLC-off image and one cloudy SLC-on image. It is quite interesting to see that our model automatically selects X_3's cloud-free right half to fill in the stripes although the cloud cover in X_3 is not marked.*



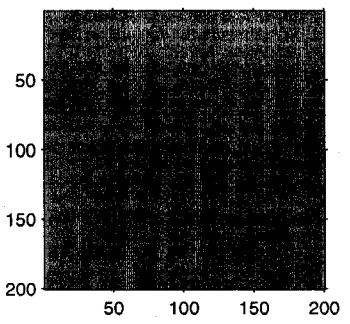
original Y_0



Simulated SLC-off $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$

Figure 21: *Filling in the gaps with 2 SLC-off image and one cloudy SLC-on image. Missing Stripes in* $\hat{Y}_{fill}$ *are from* $\hat{Y}_0 = \hat{A} + \hat{B}_1 \circ X_1 \circ M_1 + \hat{B}_2 \circ X_2 \circ M_2 + \hat{B}_3 \circ X_3$ *and non-missing parts are from simulated SLC off target image* Y.$\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$. *The partial* R_p^2 *was* 0.8805 *and* S_p, *the partial similarity between* $\hat{A}$ *and* Y_0 *for the missing regions was* -0.4775. *We slightly improved* R_p^2 *after replacing* $X_3 \circ M_3$ *in Figure 18 with a new SLC on image* X_3 *in Figure 20.*

hard to fit.

For this example, λ_{opt} was 2.2984 and λ_{max} was 117.1395; the computational time was 21.3771 minutes. For our model, the partial R_p^2 was 0.7952 and S_p, the partial similarity between $\hat{A}$ and Y_0 for the missing regions was -0.5332. The results are shown in Figure 22 and Figure 23.

From Figure 22, we can see that our algorithm now almost entirely ignores the SLC-on image X_3 and only utilizes the two SLC-off images $X_1 \circ M_1$ and $X_2 \circ M_2$. Thus our algorithm automatically adapts to the new SLC-on image X_3 and works as we have expected. Also, thé quality of $\hat{Y}_{fill}$ seems to be unaffected by the coarse urban landscape in the images.

The above six examples illustrate that we can apply an FCLM when we have different missing parts in the response and covariate images. Moreover, when used appropriately, it appears that an FCLM could adapt to the different missing scenarios and do pixel selection across covariate images. Most importantly, it seems that we do not need to resort to imputation for the covariate images and can simply rely on the algorithm to select the best non-missing parts from each covariate to fit the response. This flexibility combined with easy interpretability of the model could make an FCLM very appealing to researchers in a very wide range of fields.

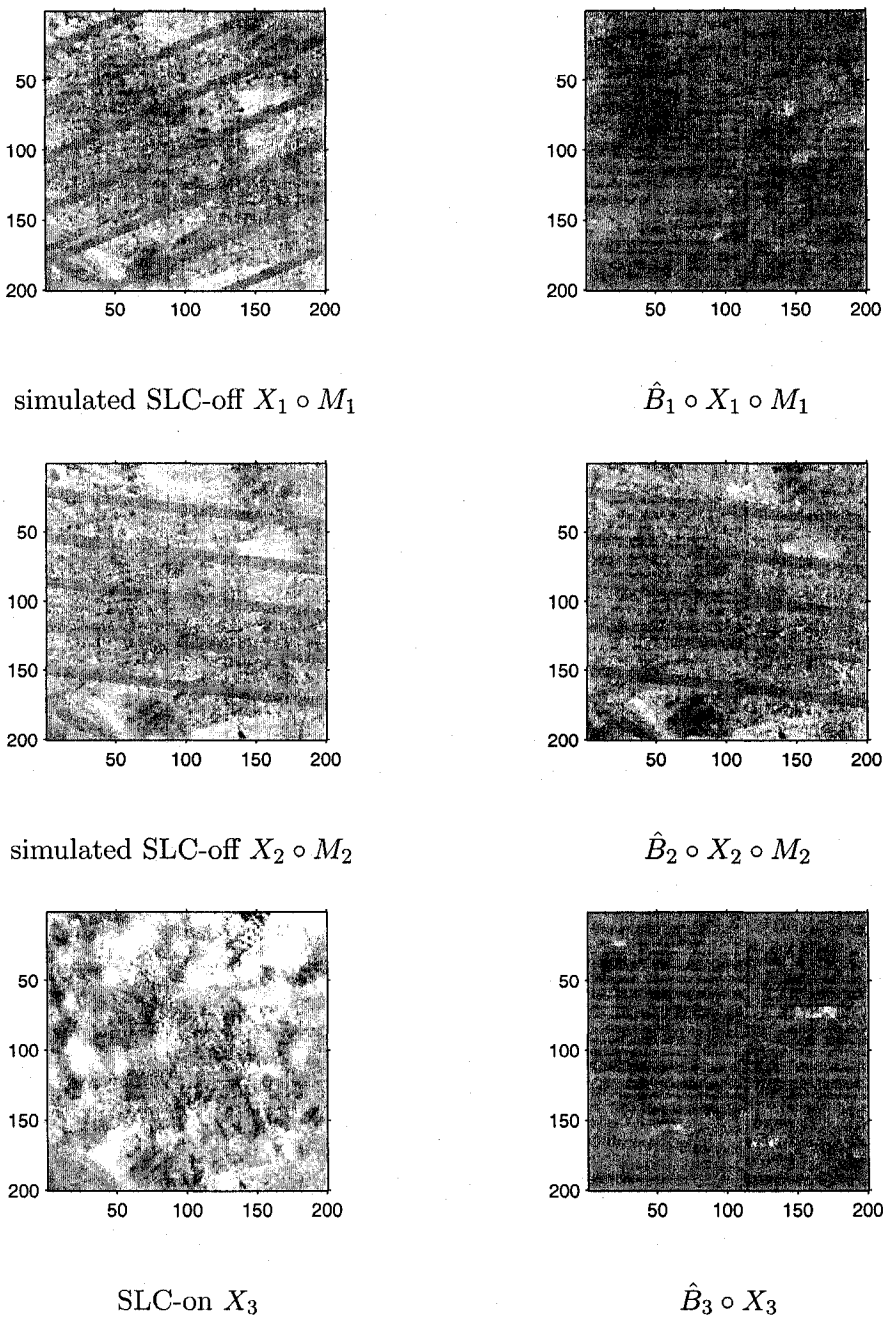
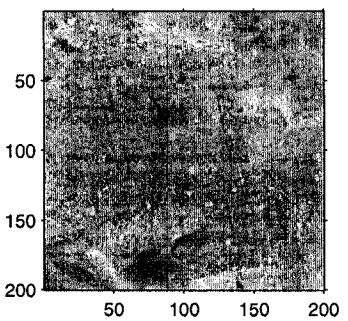
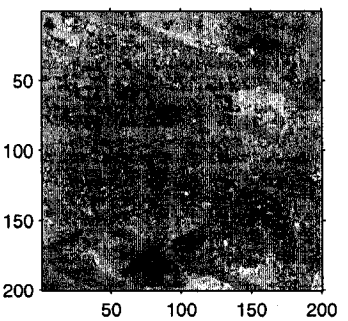


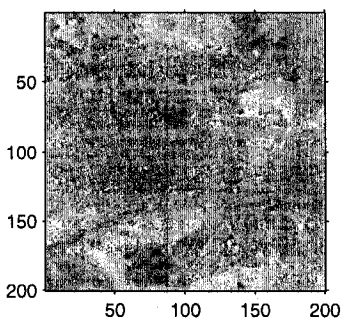
Figure 22: *Filling in the gaps with 2 SLC-off images and one cloudy SLC-on image. Now our algorithm only uses 2 SLC-off images and ignores the SLC-on image which is mostly covered by clouds.*



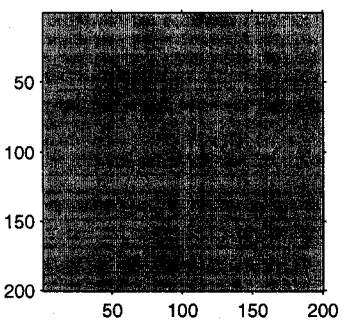
original Y_0



Simulated SLC-off $Y = Y_0 \circ M$



Filled target image $\hat{Y}_{fill}$



$\hat{A}$

Figure 23: *Filling in the gaps with* 2 *SLC-off image and one cloudy SLC-on image. Missing Stripes in* $\hat{Y}_{fill}$ *are from* $\hat{Y}_0 = \hat{A} + \hat{B}_1 \circ X_1 \circ M_1 + \hat{B}_2 \circ X_2 \circ M_2 + \hat{B}_3 \circ X_3$ *and non-missing parts are from simulated SLC off target image* Y.$\hat{Y}_{fill} = \hat{Y}_0 \circ \widetilde{M} + Y_0 \circ M$. *The computational time was* 21.3771 *minutes. For our model, the partial* R_p^2 *was* 0.7952 *and* S_p, *the partial similarity between* $\hat{A}$ *and* Y_0 *for the missing regions was* -0.5332.

Chapter 4

Discussion

We have shown that general regression tools can be developed for image data. With examples, we have demonstrated that it is practical to fit a functional concurrent linear model with varying coefficients for images, and we have used wavelets to help constrain and model these coefficients. The potential applications of regression models for spatial images are not limited for remote sensing satellite images and can be extended to other types of images. For instance, Elad et al. [7] described a parallel matching pursuit algorithm in the wavelet domain for image denoising. Their model actually can be considered as a special case of the functional concurrent linear model $Y = A + B \circ X + E$ if we set X to 0 and Y equal to the image to be denoised for the model .

In the future, we intend to explore a Bayesian framework for our spatial implementation of a functional concurrent linear model. Smith and Fahrmeir [27] showed that an Ising prior is very efficient for spatial Bayesian variable selection. We plan to borrow from this notion and apply a hierarchical Ising prior on the wavelet coefficients and achieve a large scale efficient Bayesian variable selection approach in the wavelet domain.

We close by noting that the functional concurrent linear model with varying coefficients is quite flexible. Not only can we include both continuous and discrete variables in the model, but also we can incorporate a neighborhood influence functional linear model into a functional concurrent linear model. A neighborhood influence functional linear model can be written as:

$$y_i(t_1,t_2) = \int_{\Omega(t_1,t_2)} z_i(s_1,s_2)\beta(s_1,s_2,t_1,t_2)ds_1ds_2 + \epsilon_i(t_1,t_2) \tag{4.1}$$

where, for example, $\Omega(t_1,t_2)$ is a small square-shaped neighborhood area around location (t_1,t_2). This model says that y_i's value at location (t_1,t_2) will depend on z_i's values at location (t_1,t_2) and the area around (t_1,t_2). To proceed, we could expand $\beta(s_1,s_2,t_1,t_2)$ with a 2-D discrete wavelet transform:

$$\beta(s_1,s_2,t_1,t_2) = \sum_k b_k(t_1,t_2)\phi_k(s_1,s_2) \tag{4.2}$$

If we then plug (4.2) into (4.1):

$$\begin{aligned} y_i(t_1,t_2) &= \textstyle\int_{\Omega(t_1,t_2)} z_i(s_1,s_2)\sum_k b_k(t_1,t_2)\phi_k(s_1,s_2)ds_1ds_2 + \epsilon_i(t_1,t_2) \\ &= \textstyle\sum_k b_k(t_1,t_2)\int_{\Omega(t_1,t_2)} z_i(s_1,s_2)\phi_k(s_1,s_2)ds_1ds_2 + \epsilon_i(t_1,t_2) \\ &= \textstyle\sum_k b_k(t_1,t_2)z^\star_{ik}(t_1,t_2) + \epsilon_i(t_1,t_2) \end{aligned} \tag{4.3}$$

The last line of (4.3) essentially means that we can express the neighborhood influence model as a functional concurrent linear model. In future work we

will explore the neighborhood influence functional linear model — one question is whether the complexity of the neighborhood influence model is needed, or whether the regular functional concurrent linear model is adequate. This might be the case because the concurrent model already is intended to account for local influence through the smoothing that takes place. If the neighborhood model is needed in some situations, then we will try to develop a test or diagnostic tool to determine when to choose it over the regular concurrent model.

Bibliography

[1] Mathematics Commission on Physical Sciences and Applications (CPSMA), editors. *Spatial Statistics and Digital Image Analysis.* National Academy Press, 1991.

[2] Noel Cressie. *Statistics for Spatial Data.* Wiley-Interscience, 1993.

[3] Antonio Cuevas, Manuel Febrero, and Ricardo Fraiman. Linear functional regression: The case of fixed design and functional response. *The Canadian Journal of Statistics*, 30(2):285–300, June 2002.

[4] David Donoho and Victoria Stodden. Breakdown point of model selection when the number of variables exceeds the number of observations. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1916 – 1921, 16-21 July 2006.

[5] David L. Donoho and Jared Tanner. Thresholds for the recovery of sparse solutions via l1 minimization. In *Proceedings of the Conference on Information Sciences and Systems*, pages 202 – 206, 22-24 March 2006.

[6] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics,*, 32(2):494–499, Apr. 2004.

[7] Michael Elad, Boaz Matalon, and Michael Zibulevsky. Image denoising

with shrinkage and redundant representations. June 17-22, 2006, CVPR, June 17-22 2006.

[8] K.N. Eshleman, R.P. Morgan, J.R. Webb, F.A. Deviney, and J.N. Galloway. Temporal patterns of nitrogen leakage from mid-appalachian forested watersheds: Role of insect defoliation. *Water Resources Research*, 34:2005–2016, 1998.

[9] R.L. Eubank, Chunfeng Huang, Y. Munoz Maldonado, and R.J. Buchanan. Smoothing spline estimation in varying-coefficent models. *J.R. Statist. Soc. B.*, 66(3):653–667, August 2004.

[10] Julian J. Faraway. Modeling reaching motions using functional regression analysis. In *Digital Human Modeling for Design and Engineering, Conference and Exposition, Dearborn, Michigan*, June 2000.

[11] J.R. Foster and P.A. Townsend. Mapping forest composition in the central appalachians using AVIRIS: Effects of topography and phenology. in r.o. green (ed.). *Proceedings of the Eleventh JPL Airborne Earth Science Workshop. Pasadena, CA: Jet Propulsion Laboratory*, 2002.

[12] Alan E. Gelfand, Hyon-Jung Kim, C.F. Sirmans, and Sudipto Banerjee. Spatial modeling with spatially varying coefficient processes. *J. Ameri. Statist. Associ.*, 98(462):387–396, June 2003.

[13] Constantinos Goutis. Second-derivative functional regression with applications to near infrared spectroscopy. *J.R. Statist. Soc. B*, 60(1):103–114, 1998.

[14] Trevor Hastie and Robert Tibshirani. Varying-coefficient models. *J.R. Statist. Soc. B*, 55(4):757–796, 1993.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learining.* Springer, 2001.

[16] D.R. Houston and H.T. Valentine. Comparing and predicting forest stand susceptibility to gypsy moth. *Canadian Journal of Forest Research*, 7: 447–461, 1977.

[17] Steven Johnson. *The Ghost Map.* Riverhead Hardcover, October 2006.

[18] Seung-Jean Kim, Kwangmoo Koh, Michael Lustig, Stephen P. Boyd, and Dimitry Gorinevsky. An interior-point method for large-scale l_1-regularized least squares. *IEEE Journal on Selected Topics in Signal Processing*, 1(4): 606–617, December 2007.

[19] K.W. Kleiner and M.E. Montgomery. Forest stand susceptibility to the gypsy-moth (lepidoptera, lymantriidae) - species and site effects on foliage quality to larvae. *Environmental Entomology*, 23(19 L19406):699–711, 1994.

[20] Gary M. Lovett, Lynn M. Christenson, Peter M. Groffman, Clive G. Jones, Julie E. Hart, and Myron J. Mitchell. Insect defoliation and nitrogen cycling in forests. *Bioscience*, 52(4):335–341, April 2002.

[21] B.E. McNeil, K.M. de Beurs, K.N. Eshleman, J.R. Foster, and P.A. Townsend. Maintenance of ecosystem nitrogen limitation by ephemeral forest disturbance: An assessment using modis, hyperion, and landsat ETM. *Geophysical Research Letters*, 34(19 L19406):73–102, 2007.

[22] Michael R. Osborne, Brett Presnell, and Berwin A. Turlach. On the lasso and its dual. *Journal of Computational and Graphical Statistics,*, 9(2): 319–337, Jun. 2000.

[23] Jim Ramsay and Bernard Silverman. *Functional Data Analysis 2nd Edition.* Springer, 2 edition, 2005.

[24] S.J. Ratcliffe, L.R. Leader, and G.Z. Heller. Functional data analysis with application to periodically stimulated foetal heart rate data. I: Functional regression. *Statistics in Medicine*, 21:1103–1114, 2002.

[25] Pat Scaramuzza, Esad Micijevic, and Gyanesh Chander. *SLC Gap-Filled Products Phase One Methodology.* U.S. Geological Survey Earth Resources Observation and Science (EROS) Center, http://landsat.usgs.gov/documents/SLC_Gap_Fill_Methodology.pdf, March 2004.

[26] Pat Scaramuzza, Esad Micijevic, and Gyanesh Chander. *SLC-off Gap-Filled Products Gap-Fill Algorithm Methodology Phase 2 Gap-Fill Algorithm.* U.S. Geological Survey Earth Resources Observation and Science (EROS) Center, http://landsat.usgs.gov/documents/L7SLCGapFilledMethod.pdf, October 2004.

[27] Michael Smith and Ludwig Fahrmeir. Spatial Bayesian variable selection with application to functional magnetic resonance imaging. *J. Ameri. Statist. Associ.*, 102(478):417–431, June 2007.

[28] Robert Tibshirani. Regression shrinkage and selection via the lasso. *The Annals of Statistics*, 58(1):267–288, 1996.

[29] Philip A. Townsend, Keith N. Eshleman, and Chris Welcker. Remote sensing of gypsy moth defoliation to assess variations in stream nitrogen concentrations. *Ecological Applications*, 14(2):504–516, 2004.

[30] *Enhanced Thematic Mapper Plus Scan Line Corrector Geometric Processing Algorithm Theoretical Basis.* U.S. Geological Survey Earth Resources Observation and Science (EROS) Center, http://landsat.usgs.gov/documents/SLCOff_Processing_ATBDV1.1.pdf, September 2003.

[31] James S. Walker. *A Primer on Wavelets and their Scientific Applications.* CRC Press, 1999.

[32] Mike West, P.J. Harrison, and H.S. Migon. Dynamic generalized linear models and Bayesian forecasting. *J. Ameri. Statis. Associ.*, 80(389):73–83, March 1985.

[33] Yoshihiro Yamanishi and Yutaka Tanaka. Geographically weighted functional multiple regression analysis: A numerical investigation. *J. Jpn. Soc. Comp. Statist.*, 15(2):307–317, June 2003.

[34] Hui Zou, Trevor Hastie, and Robert Tibshirani. On the degrees of freedom of the LASSO. *The Annals of Statistics*, 35(5):2173-2192, 2007.