DEPARTMENT OF STATISTICS

University of Wisconsin

1300 University Ave.

Madison, WI 53706

TECHNICAL REPORT NO. 1110

August 25, 2005

# SMOOTHING IN MAGNETIC RESONANCE IMAGE ANALYSIS AND A HYBRID LOSS FOR SUPPORT VECTOR MACHINE [1]

Xianhong Xie

xie@stat.wisc.edu

http://www.stat.wisc.edu/~xie

[1]This research is partially supported by NSF Grant DMS 0072292 and NIH grant EYO9946.

# SMOOTHING IN MAGNETIC RESONANCE IMAGE ANALYSIS AND A HYBRID LOSS FOR SUPPORT VECTOR MACHINE

By

**Xianhong Xie**

A dissertation submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

(Statistics)

at the

**UNIVERSITY OF WISCONSIN – MADISON**

2005

# Abstract

This thesis will focus on applying smoothing splines to magnetic resonance image (MRI) analysis. Some additional work on support vector machine with a hybrid loss function will be discussed.

We apply smoothing splines to both the structural MRI and functional MRI. For the structural MRI, we fit thin plate splines to overlapping blocks of the image with different configurations of knots. The optimal configurations are found by the generalized cross validation with a constant factor (Luo and Wahba, 1997). The fitted splines with the optimal configurations are then blended to get a smoothed image of the brain. Thresholds are found along the way with k-means algorithm and are blended as well. By thresholding the blended image we obtained, we get the boundaries between gray matter, white matter, cerebrospinal fluid, and others. The combination of smoothing and thresholding gives us very good results in terms of segmentation.

For the functional magnetic resonance image analysis, we propose a partial spline model for the model fitting and hypothesis testing. Simulation are done to test the theoretical properties of the model. It appears that the partial spline model can compete with the commonly used smoothing+GLM paradigm.

A support vector machine with a new hybrid loss is studied in the thesis. We propose a loss function that is a hybrid of the hinge loss and the logistic

loss, with the aim to achieve the nice properties of these two loss functions, i.e., giving sparse solutions and being able to estimate the conditional probabilities at the same time. Our results and theoretical derivation show that the new loss function has the properties we expected and serves as a nice loss function for classification as well.

# Acknowledgements

First and foremost, I'd like to thank my advisor Professor Grace Wahba for supporting me in the course of my study and teaching me how to be a researcher and many other things. I learned a lot from her by being a member of the spline group. Special thanks also go to Professor Yi Lin who always unselfishly discusses his ideas before our thursday group and provides a lot of help.

I thank Professors Moo K. Chung, Chunming Zhang, Andrew Alexander, and Wei-Yin Loh for their service on my defense committee. I got the first topic in my thesis from Professors Chung and Wahba, and other topics from Professor Wahba. They have provided lots of advice and help during the research.

Our thursday group meeting is a great place to hear some ideas, discuss one's questions, and do some thinking. The experience is very inspiring. I should thank the hosts Professors Wahba and Lin and the members of the current group: John Carew, Fan Lu, Hyonho Chun, Weiliang Shi, and others. Especially I want to thank John Carew for the discussion on magnetic resonance imaging and related topics. Some former group members: Chenlei Leng, Ming Yuan, Hao Zhang and Yoonkyung Lee, should be thanked as well.

I should not forget the fellow students, the professors, and the staff members in the statistics department. During my stay in Madison, they helped me in various ways and made my life and study enjoyable.

I owe a lot to my parents, my sisters, and my brother, who have always been very supportive of my pursuing an advanced degree in science. Without their support, the experience would have been totally different.

# List of Figures

# List of Tables

# Contents

# Chapter 1

# General Overview

## 1.1 Introduction

In this thesis, we will study different smoothing techniques for magnetic resonance image (MRI) analysis and a hybrid loss for support vector machine.

Smoothing is an important topic in MRI analysis, whether it is structural MRI or functional MRI. For structural data, smoothing can reduce the noise in the data so that subsequent analysis can generate better results. In the functional data case, smoothing can be used to alleviate the effect of not knowing the true covariance structure of the time series (Wahba, 1978). The common smoothing used in MRI analysis is kernel smoothing which could be isotropic or anisotropic with one or more parameters (bandwidth) controlling how the smoothing is done. Not many studies were done on applying smoothing splines to structural MRI or functional MRI. We will study the performance of smoothing splines under these kinds of settings.

For the support vector machine, there seem to be some interests in the sparseness and estimating probabilities for a given loss function. Two of the commonly used loss functions are the hinge loss and the logistic loss which only

have one of the two properties listed, but not the other. We try to look for some new loss function which could have both of the nice properties.

## 1.2 Outline of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we will study thin plate spline smoothing for MRI segmentation. First we do some literature review. Then the basics of the thin plate spline will be discussed. We give the segmentation algorithm for 2-D slices, with slight tweaking when the algorithm is applied to 3-D volumes. The results for 2-D and 3-D segmentations will be presented. Different evaluation criteria will be considered. We will draw some conclusions and discuss the future directions at last.

In Chapter 3, we will show some results on functional MRI data analysis. We will present some collaborative work with John Carew and others in which a GCV smoothing spline+GLM approach to fMRI data analysis was proposed. We will describe a new partial spline approach to the data analysis. Simulation results will be given and comparison will be done.

Chapter 4 will be on the support vector machine with a hybrid loss. We'll give the motivation first, followed by the definition of the loss function. Some theoretical derivation will be done to show the properties of the hybrid loss. The mathematical program for the loss function will be given and choosing the parameters will be discussed. We present the simulation results along with those for a real data example to show the advantages of the new loss function.

The thesis will be concluded with some remarks in Chapter 5.

# Chapter 2

# Magnetic Resonance Image Segmentation with Thin Plate Spline Thresholding

## 2.1 Problem Description

Segmentation of magnetic resonance (MR) images is an important part of brain imaging research. Digital image of the brain are obtained under a strong magnetic field and a weak pulse field. Different tissue types of the brain show different intensity values under the magnetic fields. Our goal is to segment a given image into three tissue types: grey matter (GM), white matter (WM), and cerebrospinal fluid (CSF). The dimension of a structural image is usually huge. And this is often coupled with image inhomogeneity and partial volume effect (i.e., a voxel could contain more than one tissue types). All of these have to be addressed in a segmentation method.

## 2.2 Literature Review

Up to now, many segmentation methods have been proposed. A rough grouping of the methods is to categorize them as either intensity based methods or surface based methods.

The neural network classifier (Morrison and Attikiouzel, 1992; Ozkan *et al.*, 1993; Kollokian, 1996; Wang *et al.*, 1998), the k-nearest neighbor classifier (Bezdek *et al.*, 1993) or a finite Gaussian mixture modeling (Bezdek *et al.*, 1993; Kapur, 1995) can be used for classifying each voxel into 3 different classes. In particular Gaussian mixture modeling assumes the image intensity values follow the mixture of two or more Gaussians and the unknown parameters of Gaussian distributions are estimated by maximizing the likelihood functions possibly via the expectation maximization (EM) algorithm or other optimization techniques. The widely used SPM'99 brain image analysis package (Wellcome Department of Cognitive Neurology, London, UK, URL http://www.fil.ion.ucl.ac.uk/spm) is based on a Bayesian Gaussian mixture modeling with a prior probability image generated by averaging the image intensity for large number of subjects (Ashburner *et al.*, 1997; Ashburner and Friston, 2000). Based on a prior probability of each voxel being the specific tissue type, a Bayesian approach is used to get a better estimate of the posterior probability. This Bayesian update of the probability is iterated many times until the probability converges. The resulting probability is interpreted as the probability of each voxel belonging to one of the three tissue types.

Instead of the above intensity-based segmentation techniques, surface-based segmentation techniques have begun to emerge. The advantage for surface-based segmentation methods is the possible reduction of the partial volume effect (Tohka *et al.*, 2004). When triangular meshes are used, the meshes are not constrained to lie on voxel boundaries. Instead the triangular meshes can cut through a voxel, which can be considered as correcting where the true boundary ought to be and reducing the partial volume effect. Deformable surface modeling (Terzopoulos *et al.*, 1988; Davatzikos and Bryant, 1995; Dale and Fischl, 1999; MacDonald *et al.*, 2000) can be used to segment tissue boundaries by either solving a partial differential equation or optimizing an objective function.

Recently isosurface modeling, known as a level set method (Sethian, 1999) seems to show some promise in tissue boundary segmentation and has been used in segmenting the sagittal section of the corpus callosum (Hoffmann *et al.*, 2004). A related approach to image segmentation problems is the method proposed in Mumford and Shah (1985), where a piecewise smooth function is fitted to the image data, with the discontinuities happening only on the boundaries between different tissue types. The solution can be obtained by solving a variational problem iteratively.

In this thesis, we propose and validate a new thin plate spline thresholding method on both 2-D image slices and 3-D image volumes by comparing our method against the expert manual segmentation (Boesen *et al.*, 2004) and some other segmentation methods.

## 2.3 Basics of Thin Plate Splines

A thin plate spline (TPS) is the minimizer of the following optimization problem (Wahba, 1990, pp. 30–31)

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_1(i),\cdots,x_d(i)))^2 + \lambda J_m^d(f), \tag{2.3.1}$$

where $y_i$ is the $i$-th observation, $(x_1(i),\cdots,x_d(i))$ is the point at which $y_i$ is observed, $\lambda$ is the smoothing parameter, and $J_m^d(f)$ is a summation (penalty functional) of $m$-th derivatives in $d$-dimensions. For the special case $m = 2$, $d = 2$, $J_m^d(f)$ is defined as

$$J_2^2(f) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(f_{x_1x_1}^2 + 2f_{x_1x_2}^2 + f_{x_2x_2}^2)dx_1dx_2. \tag{2.3.2}$$

This is the one to be used in segmentation on 2-D slices. The definition of $J_m^d(f)$ for $m = 2$, $d = 3$ which is used for 3-D segmentation can be found in Wahba (1990).

In order for the thin plate spline to work, we have to impose some constraint on $m$ and $d$: $2m-d > 0$. In addition, by imposing some mild regularity condition on $t_i,\cdots,t_n$: e.g., $t_i's$ do not fall on a straight line for $d = 2$ case, we can show that the minimization problem (2.3.1) has a unique solution

$$f_\lambda(t) = \sum_{\nu=1}^{M} d_\nu\phi_\nu(t) + \sum_{i=1}^{n} c_iE_m(t - t_i), \tag{2.3.3}$$

where $t = (x_1,\cdots,x_d)$, $t_i = (x_1(i),\cdots,x_d(i))$ are $d$-dimensional vectors, and $M = \binom{d+m-1}{d}$ is the dimension of the null space $\{f : J_m^d(f) = 0\}$ spanned by the

$\phi_\nu$'s (for $m = 2, d = 2$, we have $M = 3$, $\phi_1(t) = 1$, $\phi_2(t) = x_1$, and $\phi_3(t) = x_2$). The function $E_m(\cdot)$ in the above equation is defined as

$$E_m(\tau) = \begin{cases} \theta_{m,d}\|\tau\|^{(2m-d)} \ln(\|\tau\|) & , \text{ if } 2m-d \text{ even}, \\ \theta_{m,d}\|\tau\|^{(2m-d)} & , \quad \text{otherwise.} \end{cases} \tag{2.3.4}$$

This function is known as the thin plate spline radial basis. $\theta_{m,d}$ in the above equation is some constant.

The coefficients $c_i$'s and $d$ in equation (2.3.3) are linear in $y_i$'s (Wahba, 1990), and we have

$$\hat{y} \stackrel{\text{def}}{=} (f_\lambda(t_1), \cdots, f_\lambda(t_n))' = A(\lambda)y, \tag{2.3.5}$$

where $y = (y_1, \cdots, y_n)'$, and $A(\lambda)$ is called the smoothing matrix.

The smoothing parameter $\lambda$ in the minimization problem (2.3.1) will be chosen by the generalized cross validation criterion with a constant factor $\alpha$ which modifies the equivalent degrees of freedom of the spline (Luo and Wahba, 1997)

$$\alpha GCV(\lambda) = \frac{\|(I - A(\lambda))y\|^2/n}{[1 - \alpha \text{tr}(A(\lambda))/n]^2}. \tag{2.3.6}$$

The factor $\alpha$ should be a number no less than 1. It is needed sometimes because the assumptions for $\alpha = 1$ might be violated.

Thin plate splines can be further approximated by using a subset of design points $\{t_i : 1 \leq i \leq n\}$ or some other sets as knots (Wahba, 1990, Chapter 7; Bates *et al.*, 1987). We approximate the thin plate spline $f_\lambda(t)$ by

$$\tilde{f}_\lambda(t) = \sum_{\nu=1}^{M} d_\nu \phi_\nu(t) + \sum_{s_l \in \Omega} c_l E_m(t - s_l), \tag{2.3.7}$$

where $\Omega$ is the set of knots ($\Omega \subset \mathbb{R}^d$).

With the approximation of $f_\lambda(\cdot)$, the whole theory described above still holds; we gain more flexibility with the setting and sometimes it may be more appropriate for data fitting.

## 2.4 TPS Thresholding Method

Our method fits thin plate splines to overlapping blocks of an image slice (or volume), and blends the splines together smoothly; a similar idea was used in Wood *et al.* (2002). The main differences are that we choose the smoothing parameters differently and we use explicit subdivisions. In addition, our method obtains thresholds on every block, and the thresholds are blended the same way as we blend the splines.

A brief overview of the algorithm goes as follows (given for 2-D case). First, we divide the slice into overlapping blocks. Second, we fit thin plate splines to the image intensities at each block with different number of knots, and select the knot configuration that gives us the smallest $\alpha$GCV score on the block. Third, we fit the thin plate spline with the knot configuration found in the last step, and predict the spline on a very fine grid. We also find the thresholds on the block with the k-means algorithm. Finally, we blend the predicted block images and the thresholds with some smooth weighting functions. We calculate the averages of the corresponding thresholds on all the blocks, and find the thresholds on the blended smooth image as well. In total, we have three different threshold sets:

blended thresholds, averaged thresholds, and recomputed thresholds. We can apply all 3 thresholding schemes to the blended smooth image, and pick the one that suits our needs best. Images from different sources might need different thresholding schemes because of the variabilities among the images.

For 3-D case, the basic algorithm is the same. But some tweaking is needed. The details are given in the following sections.

### 2.4.1 2-D Algorithm

**Step 1: Partitioning of Slice Images**

With a typical slice of the size 256x256 pixels, we first clip the empty space in the slice image. Many software programs can do this; and manual clipping is quite easy to implement too. Once the clipping is done, we can divide the slice into blocks of size about 50x50 pixels. The users can determine the sizes of the blocks. A rule of thumb is to have all the tissue types (mainly gray matter, white matter, CSF, and others) in every block. The same idea was used in Kovacevic *et al.* (2002). Following a similar line of thinking, we allow some degree of overlapping between adjacent blocks (horizontally, vertically, or diagonally). The overlapping proportion between each pair of horizontally or vertically adjacent blocks is about one half of the pixels in either of the blocks (Figure 2.4.1(a)). This results in each pair of diagonally adjacent blocks having about one fourth of the pixels in the individual blocks overlapped. The histograms for the image intensities of two adjacent blocks are shown in Figure

2.4.1(b). We can see the bumps in the histograms.

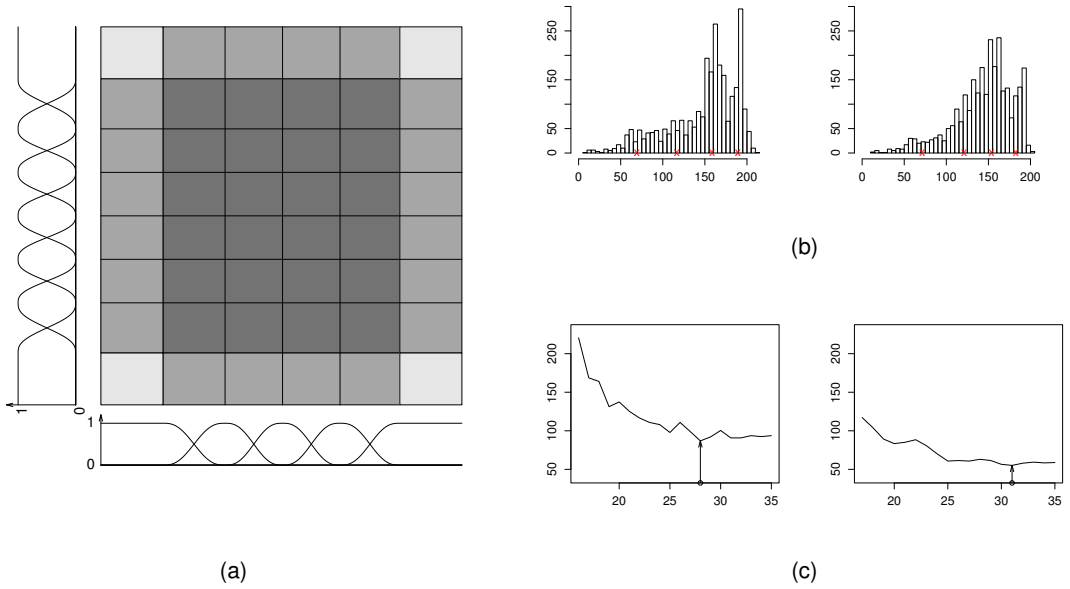



Figure 2.4.1: Overlapping scheme, $\alpha$GCV curves and histograms for one slice: (a) overlapping scheme of a slice with 5 by 7 blocks (horizontally and vertically respectively). Each 4 adjacent shaded rectangles form one block, with different shades representing different numbers of overlapping (light grey=1, medium gray=2, dark gray=4). The weighting functions at each direction are given below and to the left of the plot. (b) histograms of the corresponding blocks (the centers found by k-means on predicted data shown as red x's). (c) $\alpha$GCV curves for 2 adjacent blocks (minimum of $\alpha$GCV shown with arrows).

### Step 2: Finding Optimal $\alpha$GCV Scores

After the partitioning is done, we fit thin plate splines to each block. Note that even for one block (of the size 50x50), there are more than two thousand data points. There can be sharp boundaries between different tissue types within one

block. However, the image should be considered relatively smooth within each tissue type in a block. To fit a thin plate spline with 2500 data points as knots is not only computationally ineffective, but also unnecessary. A remedy for this is to use a subset of the 2500 knots as an approximation to the original spline that uses every data point as knot (Luo and Wahba, 1997). Since the slice image is measured on a regular grid, a further approximation is to allow the knots not fall on the pixel grid, but only require them to be on a regular grid, where the knot grid size is approximately proportional to the pixel grid size and the search is over the integer index of the knot grid (along with the smoothing parameter). In our $\alpha$GCV search, the ratio of the number of the knots to the total number of pixels in every block is between 0.02 and 0.45. In addition, the original GCV score without a constant factor $\alpha$ doesn't seem to be a good criterion for selecting the "optimal" number of knots. With a factor $\alpha = 2$, the criterion gives us the "optimal" number of knots at every block (Figure 2.4.1(c)). The reason for using a constant factor can be attributed to the added flexibility of using different number of knots in different blocks (Luo and Wahba, 1997) and also to the fact that the image data is generally correlated. The standard GCV has some limitation on correlated data (see Wahba, 1990, Section 4.9 and the references therein). The empirical value for $\alpha$ was fine tuned on a number of images, and can be further adjusted by the user if the TPS method with the given setting doesn't produce a good segmentation result.

**Step 3: Predicting TPS and Thresholding**

Using the optimal knot configuration found in the last step, we fit a thin plate spline to each block with the given configuration. A fine grid is laid on the block, with every pixel divided into 8 by 8 subpixels and the thin plate spline predicted on the grid. The use of the fine grid is to get a smoother image, which will be beneficial for the thresholding later. Another advantage of the fine grid is that we can get subpixel level segmentation and smoother boundaries. To calculate the thresholds on every block, we use the k-means algorithm, which is simple, fast and efficient. The k-means is used with 4 centers corresponding to the white matter, gray matter, cerebrospinal fluid, and the empty space, in the order of the intensity values of these tissue types appearing in a T1-weighted MR image from the highest to the lowest. Two examples of the centers found by k-means are given in Figure 2.4.1(c), which shows that the algorithm is doing a reasonably good job. We tried both 3 centers and 4 centers with k-means; it appears that the 4 center setting gives us better tissue boundaries. Once the centers have been found, we calculate the thresholds in the following way (Kovacevic *et al.*, 2002),

$$c_{ec} = (m_e + m_c)/2 \tag{2.4.1}$$

$$c_{cg} = (m_c + m_g)/2 \tag{2.4.2}$$

$$c_{gw} = (m_g + m_w)/2, \tag{2.4.3}$$

where $m_e$, $m_c$, $m_g$, $m_w$ are the centers found by the k-means algorithm, and $c_{ec}$, $c_{cg}$, $c_{gw}$ are the thresholds to be used in the later step.

**Step 4: Blending the Block Images**

Having done all the predicting and thresholding, we can now blend the block images together along with the thresholds using weighting functions. In both the horizontal and the vertical direction, a pair of functions $1 - f(t)$ and $f(t)$ is used, where

$$f(t) = \begin{cases} 0 & , \text{ if } t \leq 0 \\ t^3(6t^2 - 15t + 10) & , \text{ if } 0 < t < 1 \\ 1 & , \text{ if } t \geq 1. \end{cases} \tag{2.4.4}$$

Note $f(t)$ goes smoothly (second order differentiable) from 0 to 1, and takes only nonnegative values (it is a quintic spline). In the case a subblock is covered by only 2 adjacent blocks, either horizontally or vertically, the weighting pair $1-f(t)$ and $f(t)$ is used for left (lower) block and right (upper) block respectively. In the case a subblock is covered by 4 adjacent blocks, a tensor product weighting scheme is used. The weighting functions become $[1 - f(x)][1 - f(y)]$, $f(x)[1 - f(y)]$, $[1 - f(x)]f(y)$, and $f(x)f(y)$ for the lower left block, lower right block, upper left block, and upper right block respectively. For every overlapped subblock, the pixel coordinates are scaled so that the lower left endpoint of the subblock is mapped to $(0, 0)$, and the upper right endpoint of the subblock is mapped to $(1, 1)$.

For the blending of the thresholds, we use the same scheme. The difference between the blending of the thresholds and the blending of the block images is that we have constant matrices in the place of block images. The final number of thresholding matrices is 3 (one less than the total number of classes).

When all the blending is done, we are ready to display the segmentation results. For the averaging of the thresholds, where all the thresholding triples are averaged across the blocks, we draw contour lines at each threshold level on the blended image; for the blending of the thresholds, we draw contours at level = 0 on the difference images between the blended image and the blended thresholds. A side note is that since the blended image can be considered as an approximation to the true slice image with the boundaries between the tissue types smoothed out, we calculate the thresholds on the blended image as well (recomputed thresholds), and draw contours at these thresholds as an alternative to the previous two thresholding schemes.

### 2.4.2 Some 3-D Tweaking

So far we have presented the algorithm for 2-D case. For 3-D segmentation, some extra work is needed. The consideration is as follows. Firstly, the computational load in the 3-D case is much larger that that for the 2-D case. Instead of tens of overlapping blocks, we now have hundreds even thousands of overlapping cubes. Because of the memory and computation time limitation of the ordinary computers, the cube size can not be too big. A cube of the size 20x20x20

can still be handled by a PC with about 1 Giga byte memory. Secondly, we encounter the problem of not having all the tissue types present in every cube due to the cube size limitation. We need to address these two problems in the 3-D algorithm.

Our solution is to tweak the blending and thresholding a little bit. Instead of straightforward blending as we did in the 2-D case. We will blend only adjacent cubes together first after the $\alpha$GCV scores are found and prediction is done. Then we will do thresholding on the intermediately blended cubes. Once this is done, we blend the thresholds and blended cubes as in the 2-D case.

It seems that by combining the 3x3x3 adjacent cubes together, we get very good results. We will use this in our segmentation.

### 2.4.3 Implementation

The algorithm diagram for the TPS method is given in Figure 2.4.2. We use the Condor batch system (URL http://www.cs.wisc.edu/condor/) on the step 2 and 3 of the algorithm. Since the TPS fitting and predicting on the individual blocks (or cubes) is independent of one another, and the input of step 3 depends on the output of step 2 for each block, we deploy the Condor Directed Acyclic Graph (DAG) facility (Tannenbaum *et al.*, 2001; Thain *et al.*, 2004) which is basically a job scheduling system where independent jobs can run on different machines and Condor is in charge of enforcing the dependencies among the jobs.

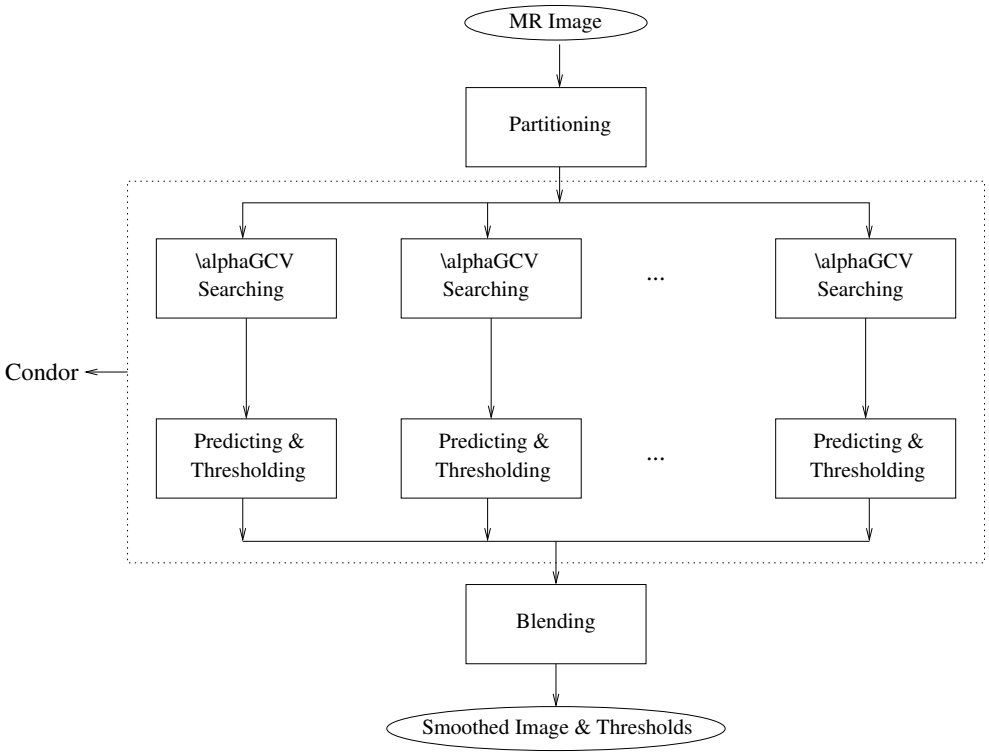



Figure 2.4.2: Diagram of TPS Algorithm. Note Condor is used for the step 2 and step 3 of the algorithm. The input and outputs are denoted by ellipses.

Most of the computation for the TPS segmentation is done with the R software (URL http://www.r-project.org/), with the exception of displaying the results in Matlab (Mathworks, Natick, MA). For the fitting of thin plate splines, I use the *fields* package in R, which seems to be sufficient for the 2-D case. But for the 3-D segmentation, the computation becomes too demanding, and I switch to the *GCVPACK* package (Bates *et al.*, 1987). The reason can be to attributed to 2 factors. First, R currently doesn't work under the Condor standard universe, which has the nice checkpointing feature (Tannenbaum *et al.*, 2001); Second, the R software seems to have some problems on 3-D thin plate splines with special knot configuration. The singular value matrix decomposition fails with the special input of the knots. The *fields* package calls R for the matrix decomposition.

## 2.5 Subjects and Image Acquisition

There are two data sources for our segmentations. The first data source is the Waisman Laboratory for Brain Imaging and Behavior at the University of Wisconsin-Madison (UW Waisman Lab for Brain Imaging and Behavior). High resolution anatomical MRI scans were obtained using a 3-Tesla GE SIGNA (General Electric Medical Systems, Waukesha, WI) scanner with a quadrature head RF coil on 12 normal subjects. A three dimensional, spoiled gradient-echo (SPGR) pulse sequence was used to generate T1-weighted images. The imaging parameters were TR/TE 21/8 ms, flip angle 30, 240 mm field of view, 256x192

in-plane acquisition matrix (interpolated on the scanner to 256x256), and 128 axial slices (1.2 mm thick) covering the whole brain. The data sets have been used in Chung *et al.* (2004) and other papers.

The second data source is the Center for Morphometric Analysis at the Massachusetts General Hospital (http://www.cma.mgh.harvard.edu/ibsr/). We downloaded the 20 normal data with human segmentations from the website. Three-dimensional T1-weighted SPGR MRI scans were performed on two different imaging systems. Ten FLASH scans on four males and six females were performed on a 1.5 Tesla Siemens Magnetom MR System (Iselin, NJ) with the following parameters: TR/TE 40/8 ms, flip angle 50, 30cm field of view, 3.1mm slice thickness, 256x256 matrix. Ten 3D-CAPRY scans on six males and four females were performed on a 1.5 Tesla General Electric Signa MR System (Milwaukee, WI), with the following parameters: TR/TE 50/9 ms, flip angle 50, 24cm field of view, 3.0mm slice thickness, 256x256 matrix. Each image volume has about 60–65 coronal slices. The data sets were used in Shan *et al.* (2002) and other papers.

## 2.6 2-D Segmentation

### 2.6.1 Evaluation of TPS Method

For the data sets obtained from the UW Waisman Lab for Brain Imaging and Behavior, we used the image for one subject in our segmentations. The neural

network classifier (Kollokian, 1996) was applied to the brain volume. And one axial slice was segmented with the TPS method. Since it is a norm to do non-uniformity correction before applying the neural network classifier, both the TPS and the neural network method were applied to the non-uniformity corrected image with N3 (Sled *et al.*, 1998). We used visual inspection on the results from the neural network method and those from the TPS method.

For the data sets downloaded from MGH CMA, we used 5 subjects out of the 20. The selection scheme was as follows: we sorted the subjects based on their id's (1_24, 2_4, 4_8, $\cdots$), and selected the 2nd, 6th, 10th, 14th, 18th subjects. We then applied the SPM segmentation method to the whole volumes of the subjects, and we also applied our TPS method to one coronal slice near the middle of the brain for each subject. Note that we have the manual segmentation for the MGH data, so we get 3 segmentations in total. To compare each pair of segmentations, we used 2 measure of similarity. One is the correlation coefficient. The other one is the kappa index, which is defined as

$$\kappa(S_1, S_2) = \frac{2|S_1 \cap S_2|}{|S_1| + |S_2|}, \tag{2.6.1}$$

where $S_1$, $S_2$ are the sets of pixels classified as one tissue type by the given segmentation methods, and $|\cdot|$ is the number of elements in the set. This measure has been used in Shan *et al.* (2002), Kovacevic *et al.* (2002), and Zijdenbos (1994). It has the nice property that two equally sized regions that overlap each other with half of their areas result in an index $\frac{1}{2}$. Also, the index is sensitive to both differences in sizes and locations of $S_i$'s. A slight variation

of the kappa index is the Jaccard index (Shan *et al.*, 2002), which differs from the kappa index only in the constant and the denominator. These 2 criteria actually gave us the same conclusions, so we will stick with the kappa index, which seems to be used more in the papers.

Since the TPS segmentation gives us the subpixel level results, we need to convert them to the pixel level to be easily compared to the manual and the SPM segmentations. The way we did it was to calculate the proportion of the number of subpixels in every pixel that belong to each class (Figure 2.6.7(c)). And we got a 4-tuple at every pixel, which sums to 1. The pixel level proportions were then used to calculate the correlation coefficients between the TPS and the other 2 methods. To get the kappa index, we thresholded both the TPS proportion outputs and the SPM probability outputs. We are only interested in the gray matter and the white matter proportions. So we computed the above mentioned indices for each pair of the segmentations on gray matter and white matter only.

### 2.6.2 Results and Plots

The results for the segmentation of the data from UW Waisman Lab for Brain Imaging and Behavior are given in Figure 2.6.1 and 2.6.2. We can see that the TPS is doing a quite good job. The neural network is doing a good job too, but it shows more inlands because the method generates 4 discrete classes $(0, 1, 2, 3)$. The averaging thresholding scheme was used with the TPS method

for the data set.

For the MGH CMA data, we used the blended threshold scheme which seems to work the best among the 3 thresholding schemes described before. The plots of the manual segmentation, the TPS segmentation, and the SPM segmentation for one subject on one slice are given in Figure 2.6.4, 2.6.5, 2.6.6 respectively. The original gray level image for the slice is given in Figure 2.6.3. Note that the manual segmentation gives the discrete classification (GM, WM, CSF, and others); TPS generates a predicted image with 3 thresholding fields, while SPM produces one probability image for each tissue class. These are reflected in the contour plots. With the subpixel property built in the algorithm, the TPS segmentation shows smoother boundaries than the other methods. Even at local regions (Figure 2.6.7 (a)-(b)), the TPS method still traces the boundary between WM and GM well without being too wiggly. The similarity measurements (with mean and standard deviation) between the 3 methods on all 5 subjects are given in Table 2.6.1. We can see that the numbers are close, with the mean coefficients for the TPS against manual and those for the SPM against manual all within one standard deviation of each other (well below the 1.96 standard deviation). If we count the number of times TPS is doing better than SPM in terms of each index and vice versa, we can find that there is no definite winner. Note that for one of the subjects (subject 4), both the TPS and the SPM failed on the segmentation. But TPS did a better job than did the SPM. Our method seems to be less sensitive to image non-uniformity.

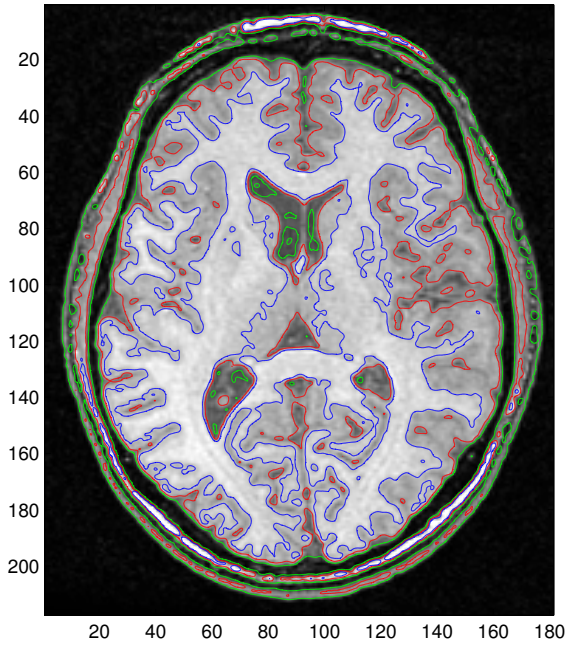



Figure 2.6.1: Overlay of the TPS segmentation on one axial slice

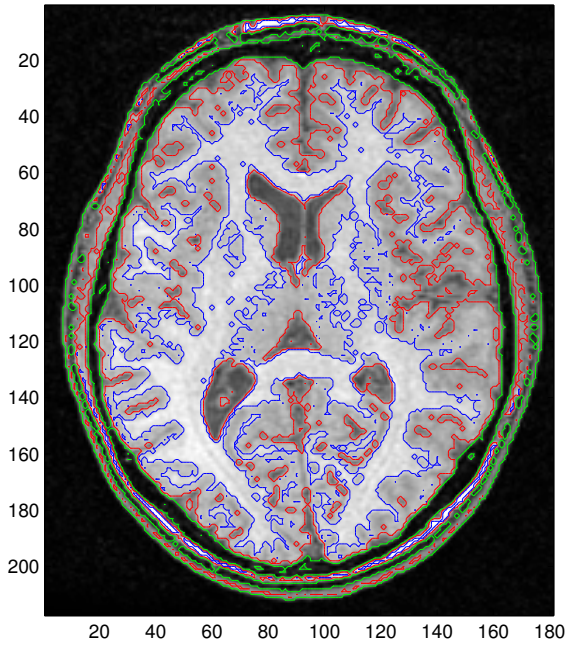



Figure 2.6.2: Overlay of the Neural Network segmentation on the axial slice

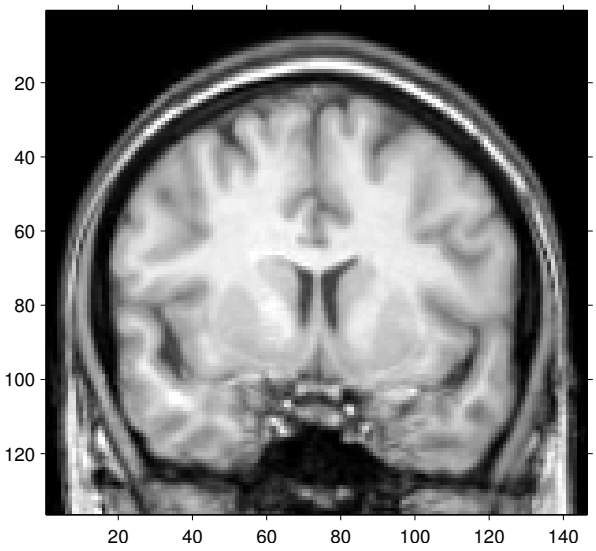



Figure 2.6.3: Original slice image in the gray scale

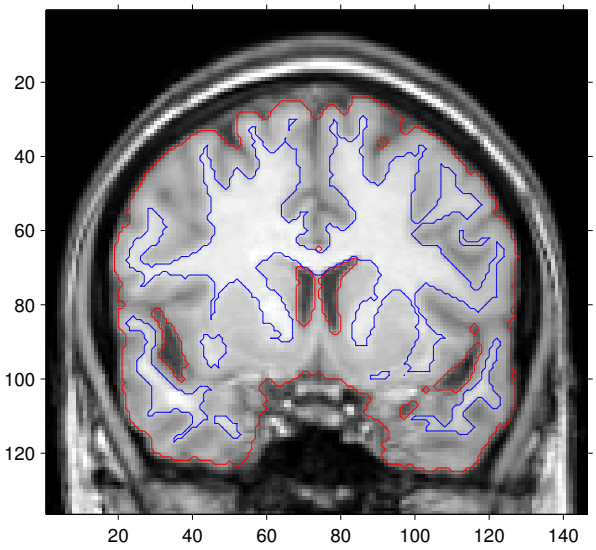



Figure 2.6.4: Overlay of the manual segmentation on the original slice

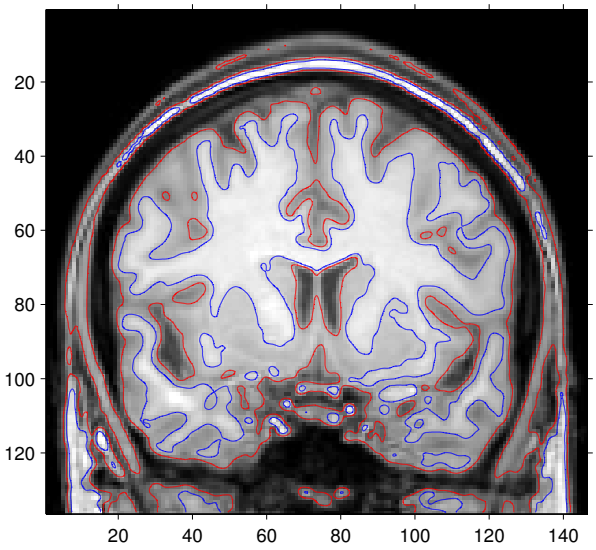



Figure 2.6.5: Overlay of the TPS segmentation on the original slice

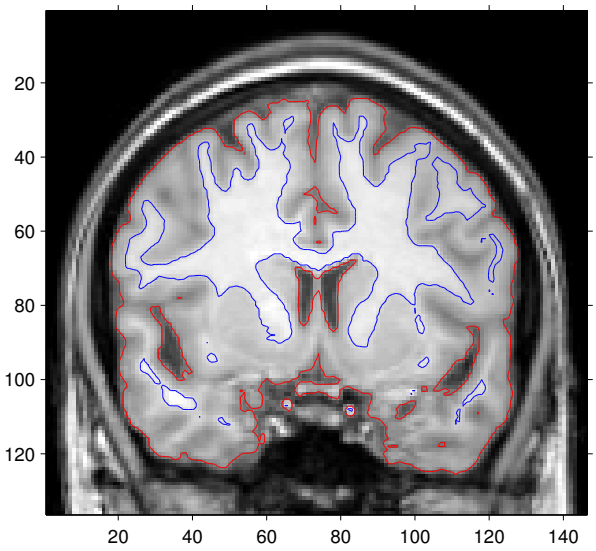



Figure 2.6.6: Overlay of the SPM segmentation on the original slice. Both the gray and the white probability outputs by SPM were thresholded at level=0.2 to get the red and the blue contour lines respectively.

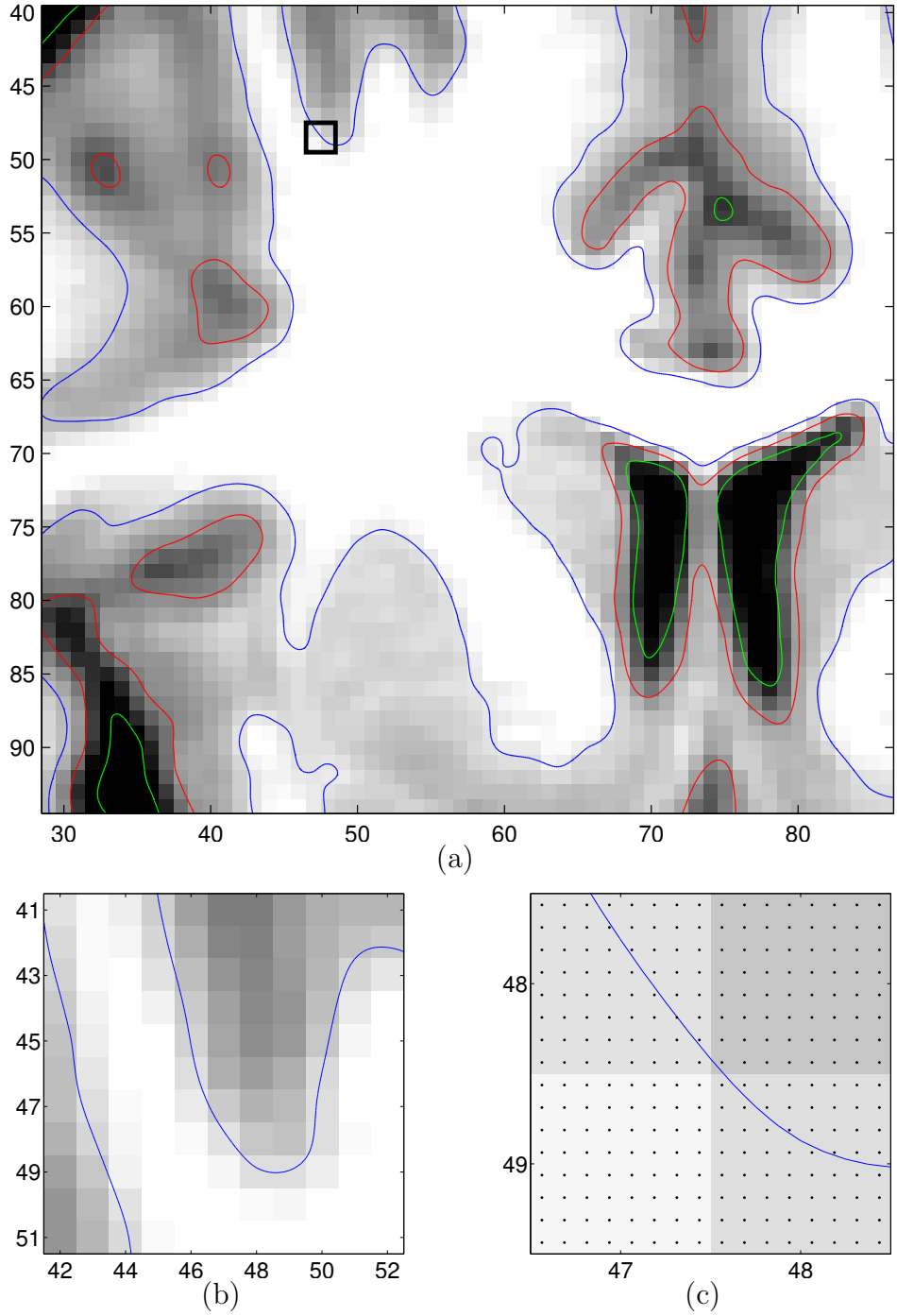



Figure 2.6.7: Zoomed plots of the TPS segmentation: (a) TPS result zoomed to a larger region, (b) TPS result zoomed to a smaller region, (c) dots plot with the centers of the subpixels shown as small dots.

Table 2.6.1: Comparison of 3 Segmentation Methods on MGH CMA Data

| | subject no. | Corr. Coef. | | Kappa Index | |
|---|---|---|---|---|---|
| | | GM | WM | GM | WM |
| TPS vs Manual | 1 | 0.660 | 0.827 | 0.836 | 0.872 |
| | 2 | 0.702 | 0.757 | 0.841 | 0.827 |
| | 3 | 0.654 | 0.787 | 0.811 | 0.850 |
| | 4 | 0.410 | 0.678 | 0.723 | 0.770 |
| | 5 | 0.612 | 0.791 | 0.776 | 0.838 |
| mean (sd) | | 0.608(0.115) | 0.768(0.056) | 0.798(0.049) | 0.831(0.038) |
| SPM vs Manual | 1 | 0.675 | 0.846 | 0.883 | 0.866 |
| | 2 | 0.686 | 0.839 | 0.887 | 0.880 |
| | 3 | 0.637 | 0.810 | 0.863 | 0.842 |
| | 4 | 0.091 | 0.672 | 0.679 | 0.753 |
| | 5 | 0.450 | 0.803 | 0.825 | 0.824 |
| mean (sd) | | 0.518(0.250) | 0.794(0.071) | 0.827(0.087) | 0.833(0.050) |
| TPS vs SPM | 1 | 0.806 | 0.883 | 0.848 | 0.900 |
| | 2 | 0.626 | 0.759 | 0.794 | 0.824 |
| | 3 | 0.734 | 0.822 | 0.808 | 0.861 |
| | 4 | 0.426 | 0.767 | 0.730 | 0.793 |
| | 5 | 0.645 | 0.800 | 0.785 | 0.836 |
| mean (sd) | | 0.647(0.143) | 0.806(0.050) | 0.793(0.043) | 0.843(0.040) |

## 2.7 3-D Segmentation

For the 3-D segmentation, we use the the data from UW Waisman Lab for Brain Imaging and Behavior. The reason for choosing the Waisman data was because the MGH CMA data downloaded from the web doesn't have enough slices. Although the latter has manual segmentation, the number of slices (equivalently the sampling rate) does seem to be a bigger concern here. To evaluate the TPS method, we compare it against the neural network method, simple thresholding, and SPM. We still use the image chosen for the 2-D segmentations, since it has

neural network segmentation available. The data had been corrected with N3 to reduce the image non-uniformity.

### 2.7.1 Evaluation Criteria

The evaluation criteria will be a little bit different from the 2-D case since we don't have the expert manual segmentation. In the 3-D case, we compute the kappa index along with 2 other indices: volume similarity (VS), and Tanimoto index (TN), inspired by Bouix *et al.* (2005). The definition for the last 2 indices is

$$VS(S_1, S_2) = 1 - \frac{\big||S_1| - |S_2|\big|}{|S_1| + |S_2|}, \tag{2.7.1}$$

$$TN(S_1, S_2) = \frac{|S_1 \cap S_2| + |\overline{S_1 \cup S_2}|}{|S_1 \cup S_2| + |\overline{S_1 \cap S_2}|}. \tag{2.7.2}$$

If we let $a_1 = |S_1 \cap S_2|$, $a_2 = |S_1 \cap \bar{S}_2|$, $a_3 = |\bar{S}_1 \cap S_2|$, and $a_4 = |\overline{S_1 \cup S_2}|$ (Figure 2.7.1), then the indices can be expressed as

$$\kappa(S_1, S_2) = \frac{2a_1}{2a_1 + a_2 + a_3}, \tag{2.7.3}$$

$$VS(S_1, S_2) = 1 - \frac{|a_2 - a_3|}{2a_1 + a_2 + a_3}, \tag{2.7.4}$$

$$TN(S_1, S_2) = \frac{a_1 + a_4}{a_1 + 2a_2 + 2a_3 + a_4}. \tag{2.7.5}$$

The volume similarity, by its name, measures how similar two sets are in terms of their volumes; and the Tanimoto index measures the degree of similarity between two sets by also referring to the whole set.

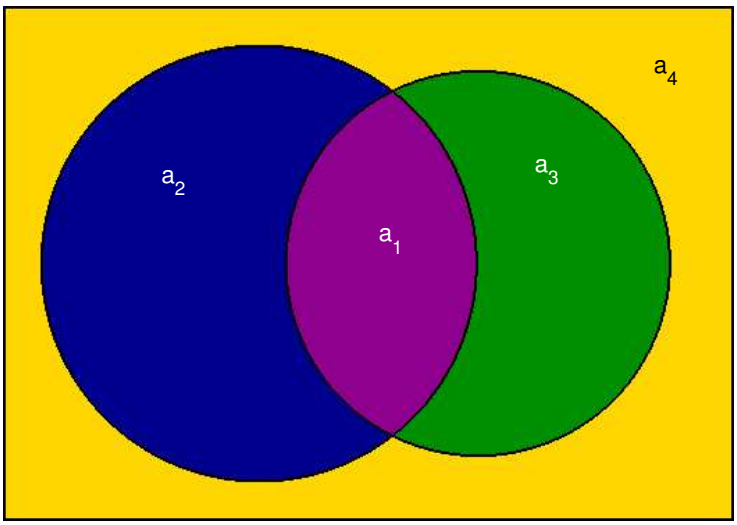



Figure 2.7.1: Venn diagrams for two sets $S_1$ and $S_2$ represented as circles (left and right respectively). The symbols $a_1$, $a_2$, $a_3$, $a_4$ denote the number of elements in the corresponding shaded regions.

After the indices are obtained, one can get a further measurement of agreement of different methods by calculating the Williams' index (Bouix *et al.*, 2005), which is defined as,

$$WI_j = \frac{(r-2)\sum_{1\leq j'\leq r,\, j'\neq j} \alpha(I_{j'}, I_j)}{2\sum_{1\leq j'\leq r,\, j'\neq j} \sum_{1\leq j''\leq (j'-1),\, j''\neq j} \alpha(I_{j'}, I_{j''})}, \tag{2.7.6}$$

where $j$ is the $j$-th method, $r$ is the total number of methods compared, and $\alpha(I_k, I_l)$ is the similarity measure between the 2 methods in question.

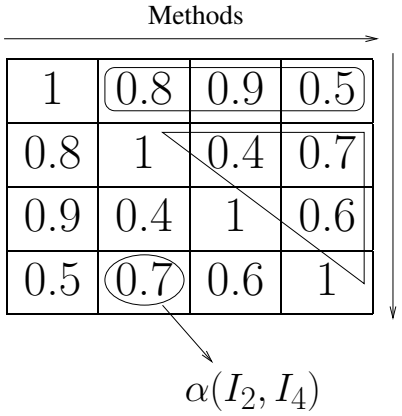



Figure 2.7.2: Illustration of the computation of Williams' index for one method (the first row). Each row (or column) represents one method, with the numbers indicating how a pair of methods agrees. The summands for the numerator of the Williams' index are circled by rounded rectangle, the summands for the denominator are circled by triangle. Plot courtesy of Sylvain Bouix.

In Figure 2.7.2, we show how the index is computed for the method indexed by $j = 1$. For a given method, the index is proportional to ratio of the closeness between the method in question and all the other methods to the closeness within all the other methods. It is argued in Bouix *et al.* (2005) and related

paper that the index is a good proxy to the criterion based on comparing one method against the estimated truth. The Williams' index has the advantage of being cheap in computation. Without the ground truth, the index seems to be a good choice.

### 2.7.2 Visualization

For displaying the segmentations in 2-D, we used the contours. The 3-D generalization of contours are the isosurfaces, which are defined as $\{(x, y, z) : f(x, y, z) = C\}$, where $C$ is some given constant. To visualize an isosurface, we can use either Matlab or VTK (Kitware, Inc., 2003). We use Matlab to calculate the isosurface, which takes 3-D image as input and outputs vertices and faces. We can then feed the output to Matlab or VTK to render the surfaces.

Both Matlab and VTK have almost the same functionalities in terms of visualization. One note about VTK is that it is a free software and is built for visualization. In contrast, Matlab has many other capabilities and is commercial. Some computer code for converting Matlab output to VTK data format and the code for rendering the surface in VTK are given in Appendix A.

### 2.7.3 Results and Plots

The rendered surfaces of the 3-D segmentations are given in Figure 2.7.3 and 2.7.4, which show the TPS segmentation is doing a good job. The GM/CSF surfaces by all 4 methods are not that much different. The GM/WM surfaces

seem to differ more than how the corresponding GM/CSF surfaces differ from one another. Note the GM/WM surface by the TPS segmentation appears to be the smoothest among the GM/WM surfaces by all the methods.

Two tables are given for the 3-D segmentations. Table 2.7.1 shows the pairwise comparisons among the 4 methods. TPS is doing good with respect to the other methods in terms of the kappa index and the volume similarity. It does slightly worse in terms of the Tanimoto index. And the TPS seems to do a better job in white matter segmentation than the gray matter segmentation in terms of 2 of the indices: kappa index and Tanimoto index. Table 2.7.2 shows the Williams' index for each method based on the 3 indices we used for pairwise comparisons. For both gray matter and white matter segmentation, the simple thresholding method has the biggest numbers for each of the 3 indices. TPS is doing better than the SPM, but worse than the neural network method in terms of the Williams' index. Caution should be employed in interpreting the results from the Williams' index since we don't know too well about the properties of the index yet (it's a relatively new concept).

Table 2.7.1: Comparisons of the 3-D Segmentations of 4 Methods

| | Kappa Index | | Vol. Similarity | | Tanimoto Index | |
|---|---|---|---|---|---|---|
| | GM | WM | GM | WM | GM | WM |
| TPS vs NNet | 0.726 | 0.863 | 0.916 | 0.811 | 0.619 | 0.732 |
| TPS vs SPM | 0.725 | 0.734 | 0.853 | 0.738 | 0.595 | 0.658 |
| TPS vs SimpThrh | 0.889 | 0.947 | 0.964 | 0.947 | 0.833 | 0.907 |
| NNet vs SPM | 0.853 | 0.846 | 0.936 | 0.923 | 0.744 | 0.825 |
| NNet vs SimpThrh | 0.810 | 0.863 | 0.952 | 0.863 | 0.711 | 0.808 |
| SPM vs SimpThrh | 0.784 | 0.776 | 0.889 | 0.788 | 0.658 | 0.721 |

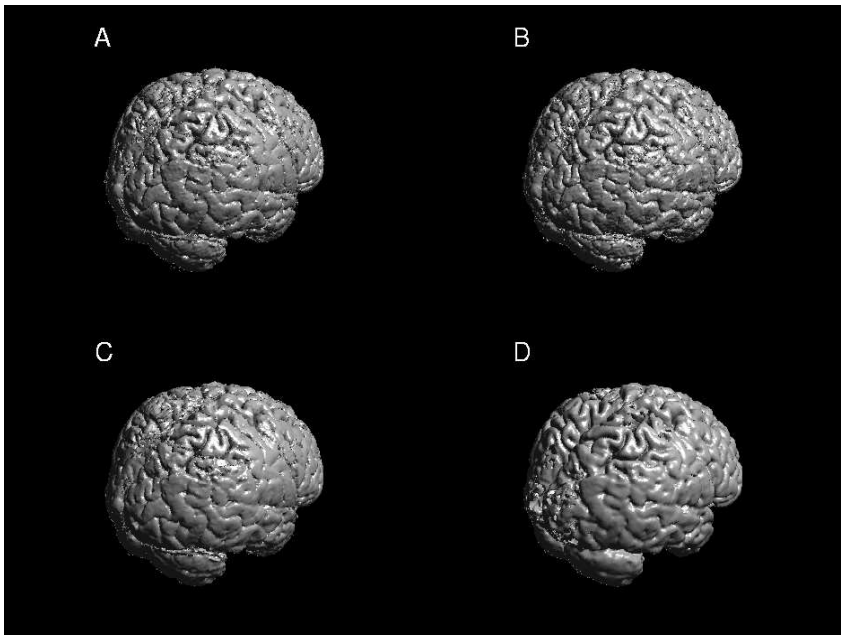



Figure 2.7.3: 3-D segmentation for GM/CSF surfaces. A: Simple Thresholding; B: Neural Network; C: TPS Thresholding; D: SPM.

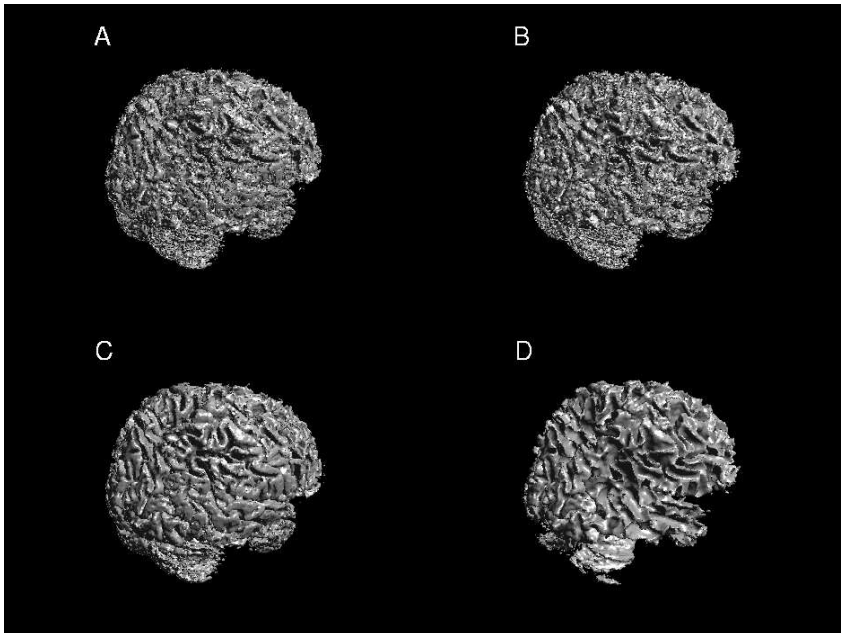



Figure 2.7.4: 3-D segmentation for GM/WM surfaces. A: Simple Thresholding; B: Neural Network; C: TPS Thresholding; D: SPM.

Table 2.7.2: Williams' Indices for the 4 Methods

| GM | NNet | TPS | SimpThrh | SPM |
|---|---|---|---|---|
| KP | 0.996 | 0.957 | 1.077 | 0.974 |
| VS | 1.036 | 0.984 | 1.037 | 0.946 |
| TN | 0.994 | 0.969 | 1.125 | 0.923 |
| WM | NNet | TPS | SimpThrh | SPM |
| KP | 1.026 | 1.003 | 1.082 | 0.899 |
| VS | 1.050 | 0.970 | 1.051 | 0.934 |
| TN | 1.035 | 0.975 | 1.099 | 0.901 |

(Note: KP=Kappa Index; VS=Volume Similarity; TN=Tanimoto Index)

## 2.8 Summary

We proposed an intensity based method for magnetic resonance image segmentation. The method works both in 2-D and 3-D segmentations. Our method does very well in the 2-D segmentation. It is still doing a quite good job in the 3-D case. The TPS method can produce subpixel (or subvoxel) segmentation, which is very useful for the partial volume effect in segmentation. The method generally gives smoother boundary; it has the potential for more accurately estimating the distances between the cortical surfaces and calculating the curvatures of the surfaces. By using the varying thresholds generated by the TPS method, we can overcome the image non-uniformity. The method appears to be a good alternative to the other segmentation methods.

## 2.9 Discussion and Future Directions

Segmentation of the brain involves huge amount of computation. It is no exception with our method. The SPM method handles the problem through simple iteration with some stopping criterion; the neural network method has some way of iteration too which gives the method some speed. We do $\alpha$GCV searching on every block of the brain, which provides us with accuracy and requires more computing at the same time. So our first step in the future work would be to speed up the computation. Under the constraint of not losing too much accuracy, we can consider doing $\alpha$GCV searching on a subset of blocks (or cubes). Some scheme can be devised to select the subset for $\alpha$GCV.

The second step in our future direction would be to get more data and try our method on the data. If our method appears to be work well in general for 3-D segmentation. We can go one step further to use our results in other applications that require segmentations as the inputs. Also, we can do some computation on the brain surfaces we get, such as the calculation of distances and curvatures. The generalization of our method to other applications can also be considered, but it should be done only after we have gotten a good handle on the applications we mentioned before.

# Chapter 3

# GCV Smoothing in Functional Magnetic Resonance Image Analysis

## 3.1 Introduction

In the last chapter, we studied how to do segmentation on structural images. A related topic is the functional magnetic resonance image (fMRI) data analysis. For the functional image, the subject lying in the scanner is supposed to perform some task that is designed to activate parts of the brain. A series of "snapshots" of the brain is obtained and used for statistical analysis.

Preprocessing is usually involved in fMRI data analysis which includes the realignment, spatial normalization, and spatial smoothing of the images. After preprocessing, the generalized linear model (GLM) is applied to the time series at each voxel to get a test statistic. This generates a map on the brain which can be thresholded to locate which parts of the brain are activated while performing

the task.

Because of the huge amount of voxels in the brain, simple statistical methods are preferred for fMRI. The GLM used in fMRI data analysis usually assumes the noise is identically independently distributed (i.i.d.) or has a autocorrelated structure. If the assumption is violated, some corrections have to be done. Two ways for handling this problem is temporal smoothing (Friston *et al.*, 1995; Worsley and Friston, 1995) and whitening (Bullmore *et al.*, 1996). The whitening involves modeling the intrinsic autocorrelation with a pre-specified structure. Since there is always the possibility of misspecifying the autocorrelation, smoothing seems to be more appropriate in this kind of setting compared to whitening (Friston *et al.*, 2000; Wahba, 1978). Carew *et al.* (2003) studied the effects of GCV spline smoothing for the problem and found that the GCV smoothing gives appropriate degree of smoothing and the estimate of the variance of a given contrast after GCV smoothing is applied is generally unbiased. To cast the problem in a related framework, we model each time series as a partial spline and study the hypothesis testing and inferences.

## 3.2 Generalized Linear Model Approach

In fMRI data analysis, the generalized linear model being considered (Friston *et al.*, 1995) is

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{K}\epsilon, \tag{3.2.1}$$

where $\mathbf{y} = (y_1, \cdots, y_n)^T$ is an $n \times 1$ equally-spaced samples of the time series, $\mathbf{X}$ is the model matrix with columns that contain signals of interest and nuisance signals, $\beta$ is an unknown parameter, $\mathbf{K}$ is an unknown matrix which characterizes the intrinsic autocorrelation, and $\epsilon \sim N(\mathbf{0}, \sigma^2\mathbf{I})$.

Note under this setting, even the naive ordinary least square (OLS) estimator of $\beta$ is unbiased, but the variance estimator of the estimator of a given contrast $\mathbf{c}^T\beta$ where $\mathbf{c}$ is a column vector of the same length as $\beta$ will be biased without knowing the true $\mathbf{K}$. By multiplying a smoothing matrix $\mathbf{S}$ to both sides of equation (3.2.1), we have

$$\mathbf{Sy} = \mathbf{SX}\beta + \mathbf{SK}\epsilon. \tag{3.2.2}$$

If we select the smoothing matrix $\mathbf{S}$ properly, we may end up with the following relation $\mathbf{SK}_a\mathbf{K}_a^T\mathbf{S}^T \approx \mathbf{SKK}^T\mathbf{S}^T$, where $\mathbf{K}_a$ is an assumed matrix for $\mathbf{K}$. The variance estimator of the estimator $\mathbf{c}^T\hat{\beta}$ with the OLS $\hat{\beta}$ from equation (3.2.2) will generally be less biased relative to the true variance of $\mathbf{c}^T\hat{\beta}$ than the variance estimator of the corresponding estimator for $\mathbf{c}^T\beta$ without smoothing. The OLS estimator of $\hat{\beta}$ for equation (3.2.2) is

$$\hat{\beta} = (\mathbf{X}^T\mathbf{S}^T\mathbf{SX})^+\mathbf{X}^T\mathbf{S}^T\mathbf{Sy}, \tag{3.2.3}$$

where $^+$ is the pseudo inverse that satisfies the Moore-Penrose conditions.

Let $\mathbf{V} = \mathbf{KK^T}$, then the covariance for the estimator $\hat{\beta}$ is

$$Var(\hat{\beta}) = \sigma^2(\mathbf{X}^T\mathbf{S}^T\mathbf{SX})^+\mathbf{X}^T\mathbf{S}^T\mathbf{SVS}^T\mathbf{SX}(\mathbf{X}^T\mathbf{S}^T\mathbf{SX})^{+T}. \tag{3.2.4}$$

Given a contrast $\mathbf{c}^T\beta$, an approximate t-test statistic for testing $H_0 : \mathbf{c}^T\beta = 0$ can be constructed as

$$T = \frac{\mathbf{c}^T\hat{\beta}}{\sqrt{\mathbf{c}^T\widehat{Var(\hat{\beta})}\mathbf{c}}}. \tag{3.2.5}$$

To get the estimate $\widehat{Var(\hat{\beta})}$, we replace $\sigma^2$ in equation (3.2.4) with

$$\hat{\sigma^2} = \frac{(\mathbf{Ry})^T\mathbf{Ry}}{tr(\mathbf{RV}_a)}, \tag{3.2.6}$$

where $\mathbf{R} = \mathbf{I} - \mathbf{SX}(\mathbf{X}^T\mathbf{S}^T\mathbf{SX})^+\mathbf{X}^T\mathbf{S}^T$, and $\mathbf{V}_a = \mathbf{K}_a\mathbf{K}_a^T$ is an approximation to the true $\mathbf{V}$. The $\mathbf{V}$ in equation (3.2.4) is replaced with $\mathbf{V}_a$ as well.

The degrees of freedom for the t-test statistic can be estimated with

$$\hat{df} = \frac{[tr(\mathbf{RV}_a)]^2}{tr(\mathbf{RV}_a\mathbf{RV}_a)}. \tag{3.2.7}$$

It was found in Carew *et al.* (2003) that $\mathbf{V}_a = \mathbf{SS}^T$ where $\mathbf{S}$ is the smoothing matrix from the GCV spline works quite well, we will use this setting for the later analysis.

## 3.3 Partial Spline Model Approach

Another approach to the same problem is the partial spline model (Wahba, 1990; Chiang *et al.*, 1999; and elsewhere)

$$\mathbf{y} = \mathbf{X_2}\gamma + f(\mathbf{t}) + \epsilon, \tag{3.3.1}$$

Where $\mathbf{X_2}$ is a matrix with the columns for the signals and nuisance as in the GLM approach, $\gamma$ is a parameter vector, $f(\cdot)$ is a smooth function, $\mathbf{t}$ is an $n\times 1$ vector representing the time points at which $\mathbf{y}$ is observed, and $\epsilon \sim N(\mathbf{0}, \sigma^2\mathbf{I})$.

The solution for $\gamma$ and $f(\cdot)$ is found by solving the optimization problem

$$\min_{\gamma\in\mathbb{R}^p, f\in\mathcal{H}} \frac{1}{n}\|\mathbf{y} - \mathbf{X_2}\gamma - f(\mathbf{t})\|^2 + \lambda \int_{t_1}^{t_n} (f''(s))^2 ds, \tag{3.3.2}$$

where $\mathcal{H}$ is a reproducing kernel hilbert space (RKHS), $\lambda$ is the smoothing parameter, $t_1$ and $t_n$ are the first and last time points respectively.

If we cast the partial spline model in a stochastic process setting (Wahba, 1990), we have

$$f(\mathbf{t}) = (f(t_1), f(t_2), \cdots, f(t_n))^T \sim N(\mathbf{X_1}\delta, b\mathbf{\Sigma}), \tag{3.3.3}$$

where $\mathbf{X_1}$ is a matrix composed of the columns that span the null space of the given penalty functional (second part of equation (3.3.2)), $\delta$ is an unknown vector of parameters, $\mathbf{\Sigma}$ is the kernel matrix corresponding to the given penalty, $b$ is a scale parameter, and $f(\mathbf{t})$ is assumed to be independent of $\epsilon$.

If we let $\mathbf{X} = (\mathbf{X_1}\ \mathbf{X_2})$, $\alpha = (\delta^T\ \gamma^T)^T$, then

$$\mathbf{y} = \mathbf{X}\alpha + N_{b,\sigma}, \tag{3.3.4}$$

where $N_{b,\sigma} \sim N(\mathbf{0}, b\mathbf{\Sigma} + \sigma^2\mathbf{I})$.

It is shown in Wahba (1990) that the minimum variance linear unbiased estimator of $\alpha$ is

$$\hat{\alpha} = (\hat{\delta}^T\ \hat{\gamma}^T)^T = (\mathbf{X}^T\mathbf{M}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{M}^{-1}\mathbf{y}, \tag{3.3.5}$$

where $\mathbf{M} = \mathbf{\Sigma} + (\sigma^2/b)\mathbf{I}$. When the parameters $b$, $\sigma^2$, and $\lambda$ satisfy the equation $\lambda = \sigma^2/(nb)$, the optimal estimate of $\mathbf{X}_2\gamma + f(\mathbf{t})$ under the optimization setting will be equal to the corresponding estimate under the stochastic setting.

Using the GCV estimate $\lambda^*$ of $\lambda$ from equation (3.3.2) where the GCV criterion is defined as

$$V(\lambda) = \frac{\frac{1}{n}\|\mathbf{y} - \mathbf{X_2}\gamma - f(\mathbf{t})\|^2}{[\frac{1}{n}tr(\mathbf{I} - A(\lambda))]^2}, \tag{3.3.6}$$

we can get an estimate of $\sigma^2$ as $\hat{\sigma^2} = \frac{\|\mathbf{y}-\mathbf{X}_2\hat{\gamma}-f_{\lambda^*}(\mathbf{t})\|^2}{tr(\mathbf{I}-A(\lambda^*))}$. The $A(\cdot)$ in the previous two expressions is the smoothing matrix for the optimization problem (3.3.2), $f_{\cdot}(\mathbf{t})$ is the solution to $f(\mathbf{t})$ with the given smoothing parameter.

Notice $\hat{\alpha}$ has the distribution $N(\alpha, b(\mathbf{X}^T\mathbf{M}^{-1}\mathbf{X})^{-1})$, an approximate $\chi^2$ statitic for testing $H_0 : \mathbf{c}^T\alpha = 0$ vs $H_1 : \mathbf{c}^T\alpha \neq 0$ can be constructed as

$$T_2 = \frac{(\mathbf{c}^T\hat{\alpha})^2}{\hat{b}\mathbf{c}^T(\mathbf{X}^T\hat{\mathbf{M}}^{-1}\mathbf{X})^{-1}\mathbf{c}}, \tag{3.3.7}$$

where $\hat{b} = \hat{\sigma^2}/(n\lambda^*)$, and $\hat{\mathbf{M}} = \mathbf{\Sigma} + n\lambda^*\mathbf{I}$. The $\chi^2$ statistic has one degree of freedom.

## 3.4 Simulation Study

Two simulations were done to compare the partial spline model against the generalized linear model with GCV smoothing (Smoothing+GLM). The purpose was to see if the partial spline model is viable in the fMRI data analysis.

For the first simulation, we employed the scheme that had been used in Carew *et al.* (2003). First, we used the SPM package to fit a generalized linear

model on a real data set, the resulting t-map was used to select the 100 "most activated" voxeles (i.e., the 100 time series with the largest t-test statistics in magnitude among the whole brain); then the 100 time series were fitted with a linear model containing the constant term and a convolved boxcar function, the residuals of the linear model were used to find the auto-regressive coefficients of an AR(8) model. Using these coefficients, we constructed 100 AR(8) noise sequences (one for each set of coefficients). Each of these noise sequences plus the boxcar function (Figure 3.4.1) is our simulated time series. One example of the time series is given in Figure 3.4.2.

For the second simulation, the data we used to simulate the smoothing functions was from a subject who was resting under the scanner when the images were scanned. The same positions (100 voxels) of the brain as in the last experiment were selected. A smoothing spline was fitted to each time series; the fitted values were scaled to $[0, 1]$ and then used as the smooth background in the simulated time series. One arbitrary part about the simulation was that the time series for the subject at rest is 2 times the length of the time series for the same subject in motion. So I just dropped the additional length of the time series to get length 110 time series. The signal part in the simulation was the boxcar function (Figure 3.4.1) multiplied by the constant 0.6. The simulated time series were the sum of the signal part, the smooth background, and the white noise (sd=0.5). One example of the time series is given in Figure 3.4.3.

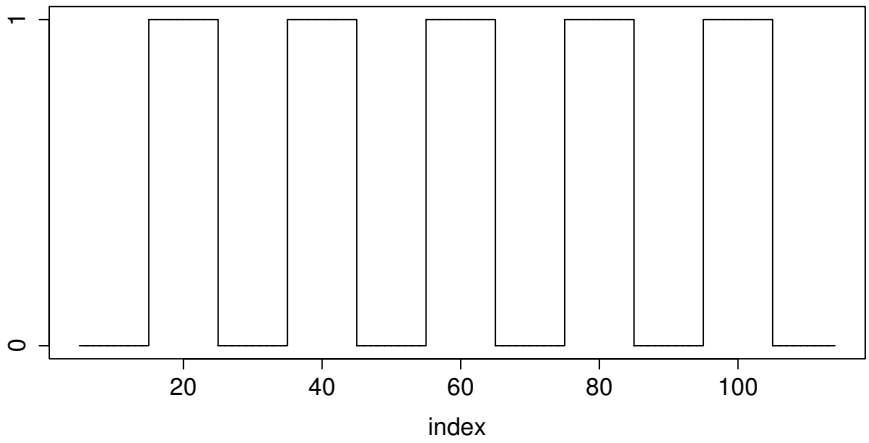



Figure 3.4.1: Boxcar stimulus function for the simulation (length=110)

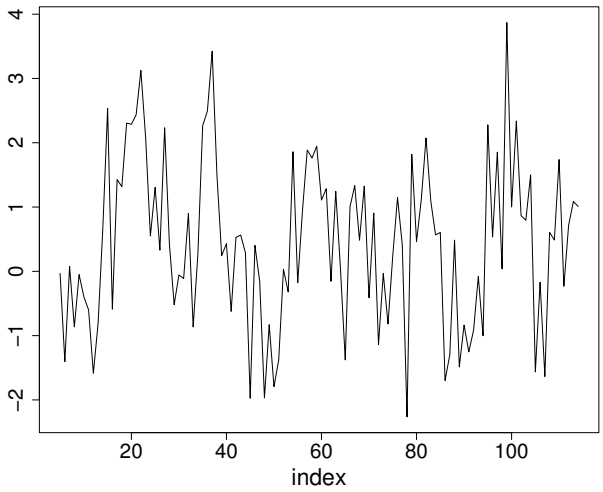



Figure 3.4.2: One example of simulated time series from the signal+AR noise paradigm

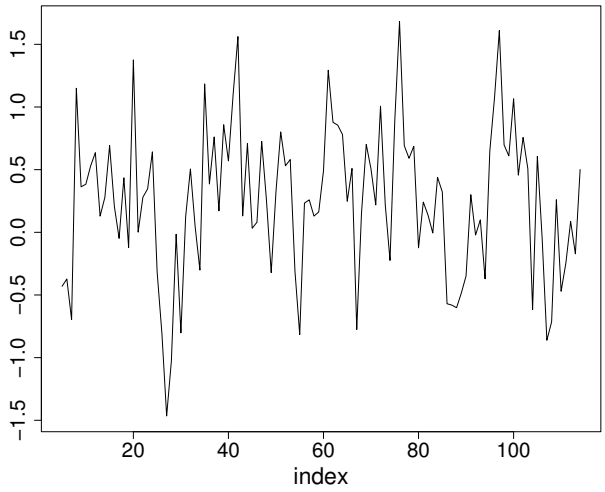



Figure 3.4.3: One example of simulated time series from the signal+smooth background+white noise paradigm

## 3.5 Analysis and Results

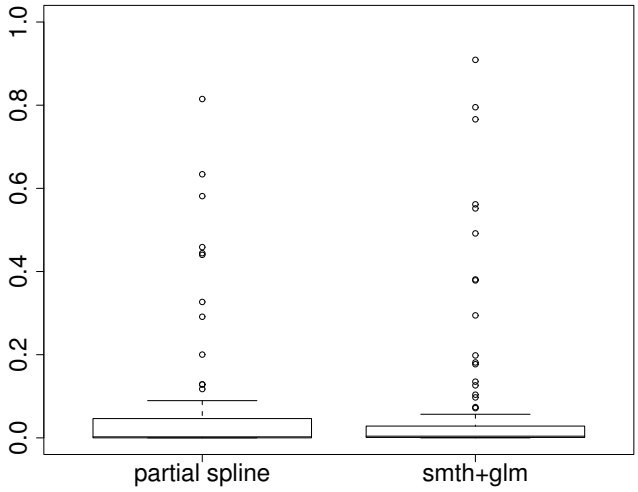
Both the Smoothing+GLM and the partial spline model were fitted to the 2 sets of simulated time series. In both models, the matrix $\mathbf{X}$ has 5 columns. The first 4 are polynomials up to order 3 (in the increasing order), which are used as the drift for the time series. The last column of $\mathbf{X}$ is the boxcar function given in Figure 3.4.1. In the partial spline model, $\mathbf{X_2}$ consists of the last 3 columns of $\mathbf{X}$. The contrast $\mathbf{c}$ of interest is $(0, 0, 0, 0, 1)^T$ for both models. Note this is a slightly different convention as commonly used in the statistics community. The t-test statistics and $\chi^2$-test statistics were calculated for the corresponding models and the $p$-values were obtained. Some model diagnostics were done for

each model under both simulation schemes.

The boxplots of the p-values for the partial spline model and those for the smoothing+GLM are given in Figure 3.5.1 (a) and (b) for simulation scheme 1 and 2 respectively. Figure 3.5.1 (a) shows that under the signal+AR noise setting, the smoothing+GLM seems to detect more time series with signals (at level=.05) than does the partial spline model. Also, the p-values for the smoothing+GLM have a smaller variance than those for the partial spline model. But as far as hypothesis testing is concerned, the results from these two models are close under the simulation setting. Under the other simulation setting (signal+smooth background+white noise), the partial spline model is doing better (Figure 3.5.1 (b)) than the smoothing+GLM. Now the partial spline model can detect almost all the time series with signals.

The diagnostic plots for the model fitting on the time series given in Figure 3.4.2 and 3.4.3 are shown in Figure 3.5.2-3.5.5. It seems that under simulation scheme 1, both the partial spline model and the smoothing+GLM fit the time series quite well, with the residuals from the model fitting and the fitted values appearing to be independent and the normality of the residuals appearing to be met. Under the simulation scheme 2, the model fitting of the partial spline model seems to be better than that of the smoothing+GLM.

(a)




(b)

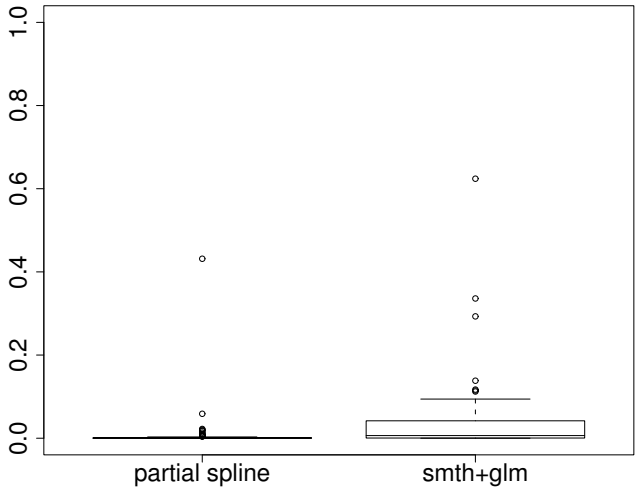
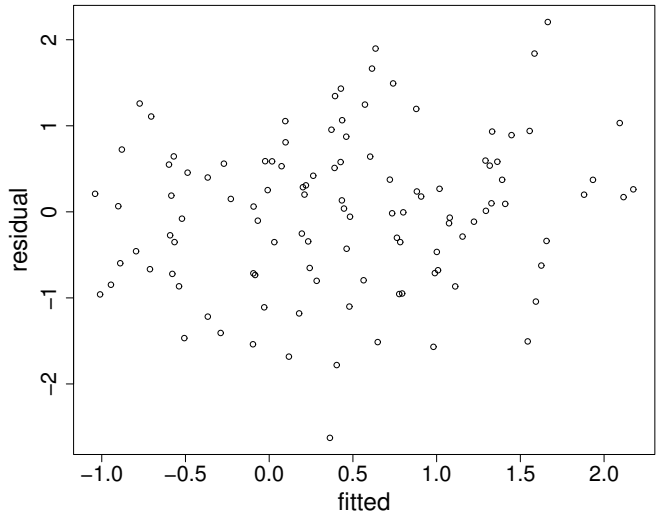



Figure 3.5.1: Boxplots of p-values for the partial spline model and smoothing+GLM fitted to data simulated under different schemes. (a) signal+AR noise; (b) signal+smooth background+white noise.

(a)




(b)

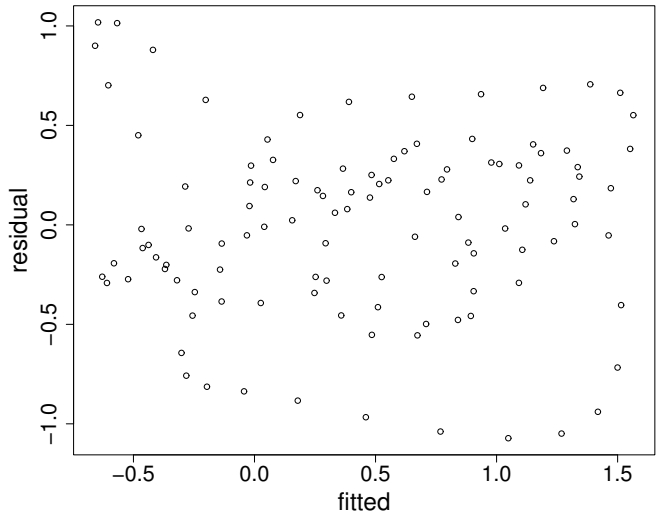
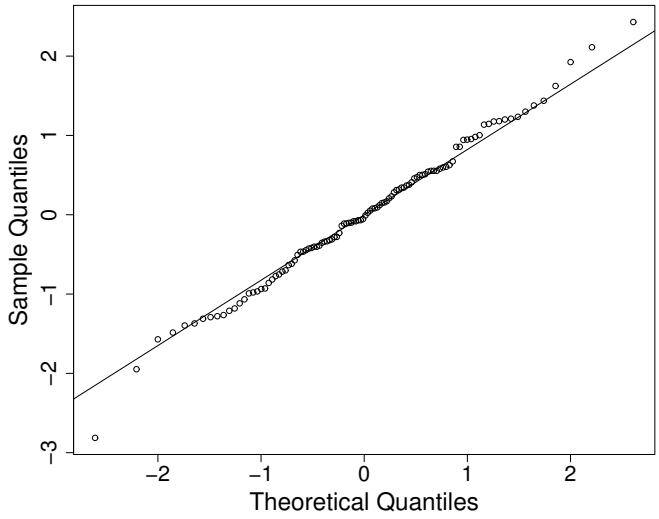



Figure 3.5.2: Diagnositic plots of the residual vs fitted values for the partial spline model and the smoothing+GLM fitted to a time series simulated under signal+AR noise scheme. (a) Partial spline model; (b) Smoothing+GLM.

(a)




(b)

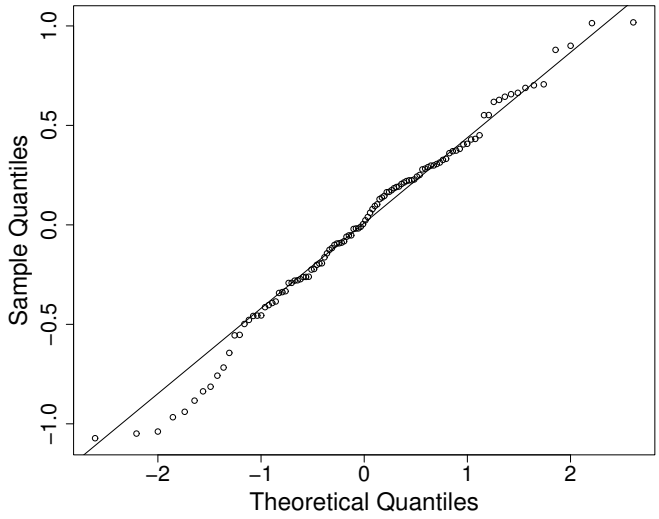
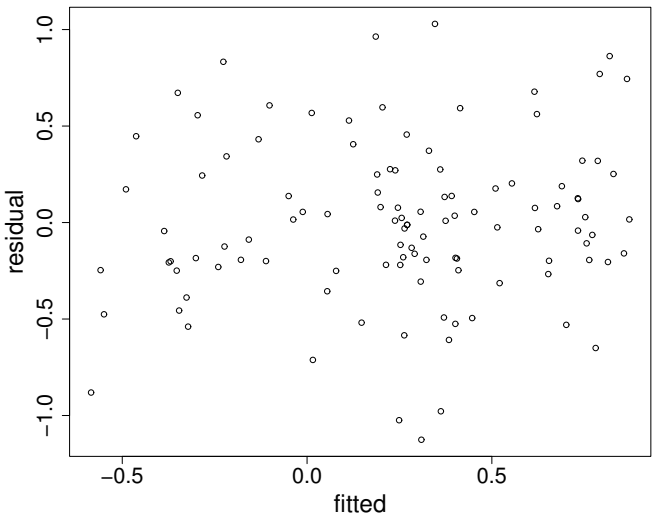



Figure 3.5.3: Quantile-Quantile plots for the partial spline model and the smoothing+GLM fitted to a time series simulated under signal+AR noise scheme. (a) Partial spline model; (b) Smoothing+GLM.

(a)




(b)

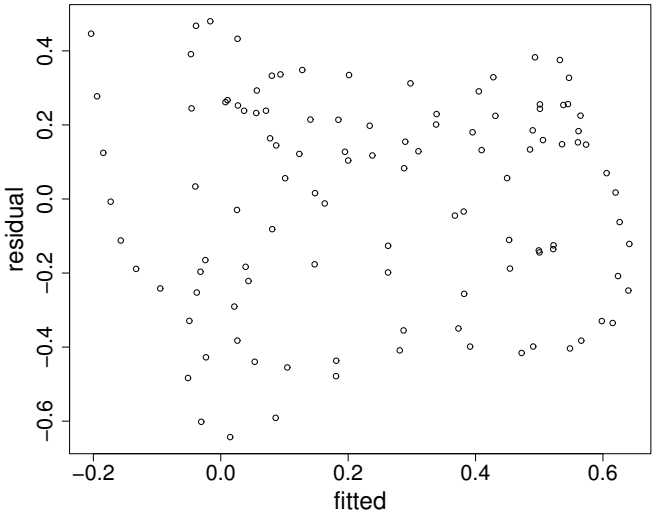
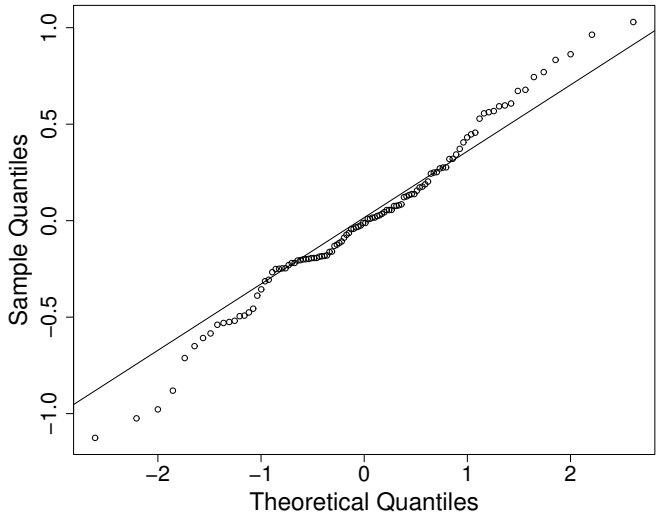



Figure 3.5.4: Diagnositic plots of the residual vs fitted values for the partial spline model and the smoothing+GLM fitted to a time series simulated under signal+smooth background+white noise scheme. (a) Partial spline model; (b) Smoothing+GLM.

(a)




(b)

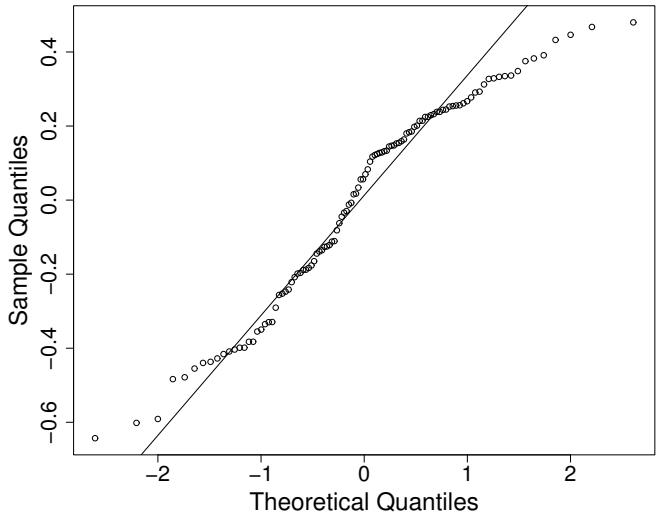



Figure 3.5.5: Quantile-Quantile plots for the partial spline model and the smoothing+GLM fitted to a time series simulated under signal+smooth background+white noise scheme. (a) Partial spline model; (b) Smoothing+GLM.

## 3.6 Summary and Discussion

We proposed applying the partial spline model to fMRI data analysis. A hypothesis testing procedure was given and simulations were done. The partial spline model was compared against the smoothing+GLM (a popular model in fMRI data analysis). Our results show that under different simulation schemes, the partial spline model does a quite good job in terms of model fitting and hypothesis testing. Since it is unkown what the true underlying distribution of the time series is in fMRI data analysis, expert opinions might be needed to tell if the partial spline model is comparable to the traditional model after both models are applied to the real data and other procedures are performed.

# Chapter 4

# A Hybrid Loss for Support Vector Machine

## 4.1 Motivation

In this chapter we consider the problem of classifying an object, based on a vector of attributes, into one of two or more categories. There are many ways for solving the problem which include the support vector machine (SVM), tree methods, neural network, boosting, and etc.. We will study support vector machine for the classification problem, focusing only on the two class case.

Under the SVM setting, we need to solve an optimization problem that consists of the sum of two parts (Wahba, 2002), i.e., the cost of using a given function to classify the data and the smoothness of the function. Different choices of the loss functions used in the cost part generally lead to different solutions (Hastie *et al.*, 2001). And two of the popular choices are the hinge loss and the logistic loss. The square loss is also used quite often because many fast algorithms are available with the corresponding optimization problem. Some

concerns one may have for choosing a loss function include the Fisher consistency, spareness, and whether estimating conditional probabilities (Bartlett and Tewari, 2004) is associated with a given loss function. The latter two are a little more of interest since most convex loss functions we have seen are Fisher consistent, i.e., the sign of the function that minimizes the expected loss with respect to a given loss function is the same as the sign of the Bayes rule (Lin, 2001). Among the loss functions we have listed, the hinge loss has sparse solutions, but it doesn't estimate the conditional probabilities $p(1|x)$ where x is a given attribute vector and 1 represents the positive class. The other 2 loss functions (logistic loss and square loss) estimate the probabilities, but they don't have sparse solutions. Our goal is to find a loss function that has the nice properties of both the hinge loss and the logistic loss. The advantage of having sparsity is the reduction in computing when the decision rule is used in prediction. And the niceness of being able to estimate conditional probabilities is to give one more information about the data.

## 4.2 New Hybrid Loss

Notice the difference between the hinge loss and the logistic loss (Wahba, 2002; Hastie *et al.*, 2001). The logistic loss $log(1 + e^{-\tau})$ is differentiable everywhere in $(-\infty, \infty)$, while the the hinge loss $max\{1 - \tau, 0\}$ is joined together with two linear segments. A natural thought is to change the logistic loss to piecewise function too, while maintaining some level of continuity (Wahba, 2002).

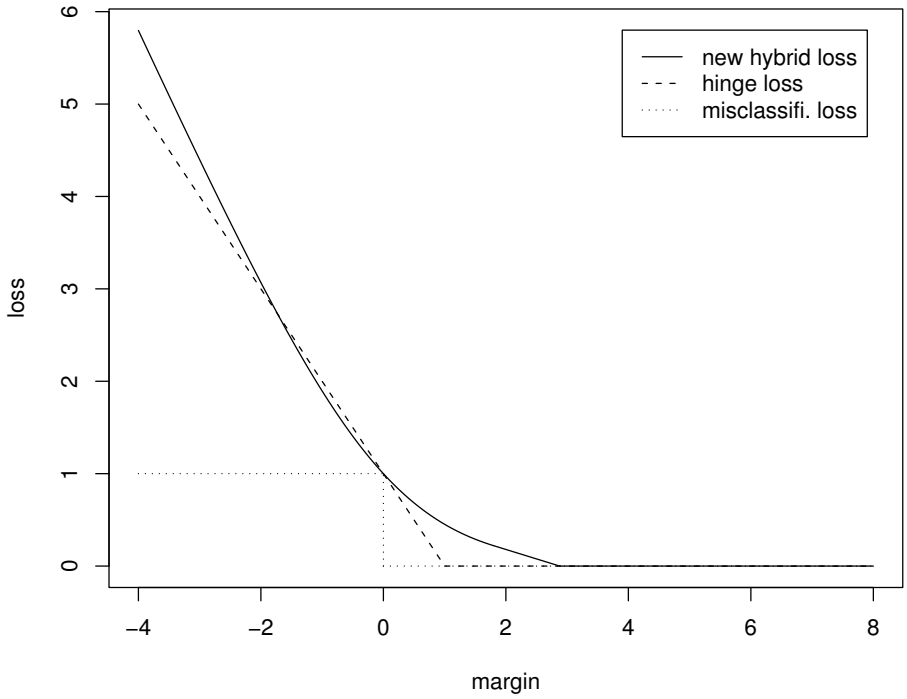



Figure 4.2.1: Plot of the new loss function with parameter $\theta = 1.8$: also overlaid are the hinge loss and the misclassification loss functions. Note we changed the natural logarithm in the new loss function to base 2 logarithm to make all the loss functions pass the same point $(0, 1)$.

The proposed hybrid loss function is given in the following,

$$l(\tau) = \begin{cases} \ln(1+e^{-\tau}) & , \text{ if } \tau \leq \theta \\ \ln(1+e^{-\theta}) - \frac{\tau-\theta}{1+e^{\theta}} & , \text{ if } \theta < \tau \leq (1+e^{\theta})\ln(1+e^{\theta}) - \theta e^{\theta} \\ 0 & , \text{ if } \tau > (1+e^{\theta})\ln(1+e^{\theta}) - \theta e^{\theta}. \end{cases} \quad (4.2.1)$$

In the equation, there is one parameter $\theta$ which controls how much one wants to truncate the original logistic loss. The plot for the hybrid loss with $\theta = 1.8$ is given in Figure 4.2.1. For convenience, we define $(*) = (1+e^{\theta})\ln(1+e^{\theta}) - \theta e^{\theta}$. Note that the loss function has first order continuity at $\tau = \theta$, zero-th order continuity across the whole real line, and is convex, it can serve as a surrogate for the 0-1 misclassification loss (Figure 4.2.1). Since the hybrid loss is a mix of the logistic loss and the hinge loss, we expect the new loss has the nice properties of both loss functions.

Bartlett (2003) studied the sparseness vs estimating conditional probabilities for a family of loss function of the form $\max\{(1-\tau)_+^2, (1-\gamma)^2\}$, where $(\cdot)_+$ is the plus function defined as $max\{\cdot, 0\}$, and $\gamma$ is an unknown parameter in the interval $(0, 1]$. This loss function should have similar properties as our hybrid loss. The differences between our loss function and Bartlett's loss function are similar to those between the logistic loss and the square loss, i.e., we obtain better estimate of probabilities at the price of slightly more computation.

## 4.3 General SVM and RKHS Framework

Under the reproducing kernel hilbert space (RKHS) framework, the optimization problem for support vector machine can be formulated as

$$\arg \min_{\substack{h \in \mathcal{H}_K \\ d \in R}} \frac{1}{n} \sum_{i=1}^{n} l(y_i(h(x_i) + d)) + \lambda \|h\|^2_{\mathcal{H}_K}, \tag{4.3.1}$$

where $\mathcal{H}_K$ is a RKHS with kernel $K$, $\lambda$ is the smoothing parameter, and $n$ is the number of observations. It is known from Wahba (2002) that the solution $h(x)$ to the optimization problem has the form

$$h(x) = \sum_{j=1}^{n} c_j K(x, x_j). \tag{4.3.2}$$

Also, we have $\| \sum_{j=1}^{n} c_j K(\cdot, x_j)\|^2_{\mathcal{H}_K} = c^T K_n c$, where $c = (c_1, \cdots, c_n)^T$, and $K_n$ is a matrix with the $i, j$-th entry $K(x_i, x_j)$.

Plugging the expression for norm and equation (4.3.2) into equation (4.3.1), we get the new form of the optimization problem

$$\arg \min_{c,d} \frac{1}{n} \sum_{i=1}^{n} l(y_i[\sum_{j=1}^{n} c_j K(x_i, x_j) + d]) + \lambda c^T K_n c. \tag{4.3.3}$$

Let $f(x) = h(x) + d$, the optimal solution for the data. The observations $(x_j, y_j)$'s which have corresponding $c_j$'s equal to 0 are called support vectors (SV). Different kernels used can have an influence on the sparseness of the representation of $f(x)$. The common choices for kernels are the polynomial kernel $(1 + \langle x, x' \rangle)^m$ where $m$ is an integer and the Gaussian kernel $exp(-\|x - x'\|^2/(2\sigma^2))$.

## 4.4 Theoretical Properties of the Hybrid Loss

Let $P$ be the distribution of $(X, Y) \in \mathcal{X} \times \{\pm 1\}$, $P_X$ be the distribution of the attribute vector $X$, and denote the conditional distribution $P(Y = 1|X = x)$ by $p(1|x)$ for short, the population minimizer $argmin_f \mathbb{E}[l(Yf(X))|X = x]$ for the hybrid loss $l(\cdot)$ is

$$\hat{f}(x) = \begin{cases} -(*) & , \text{ if } \text{logit}(p) \leq \ln\frac{1+e^{-(*)}}{1+e^{\theta}} \\ -\text{logit}(\frac{1-p}{p}\frac{1}{1+e^{\theta}}) & , \text{ if } \ln\frac{1+e^{-(*)}}{1+e^{\theta}} < \text{logit}(p) \leq -\theta \\ \text{logit}(p) & , \text{ if } -\theta < \text{logit}(p) \leq \theta \\ \text{logit}(\frac{p}{1-p}\frac{1}{1+e^{\theta}}) & , \text{ if } \theta < \text{logit}(p) \leq \ln\frac{1+e^{\theta}}{1+e^{-(*)}} \\ (*) & , \text{ if } \text{logit}(p) > \ln\frac{1+e^{\theta}}{1+e^{-(*)}}. \end{cases} \tag{4.4.1}$$

In the above expression, $(*)$ is defined as before, $p = p(1|x)$, and $\text{logit}(p) = \ln\frac{p}{1-p}, \forall\ 0 < p < 1$. We omit the cases $p(1|x) = 0$ or 1. For $p(1|x) = 0$, $\hat{f}(x)$ can be any number $\leq -(*)$; for $p(1|x) = 1$, $\hat{f}(x)$ can be any number $\geq (*)$. The derivation of this minimizer is given in Appendix B.1. A plot of $\hat{f}(x)$ vs $p(1|x)$ is given in Figure 4.4.1. The population minimizer shows that the loss function estimates the conditional probability when the true probability is away from 0 or 1. With the population minimizer, we can show that the hybrid loss is Fisher consistent (Appendix B.2).

In addition, using the value of $f(x)$ to estimate the true probability $p(1|x)$,

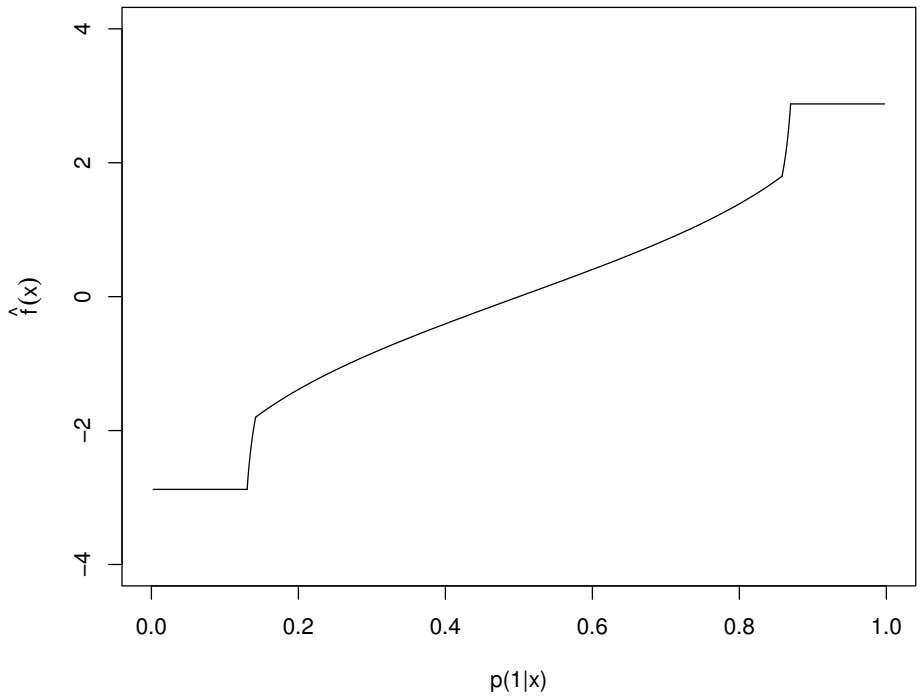



Figure 4.4.1: Plot the population minimizer $\hat{f}(x)$ vs the conditional probability function p(1|x). The middle part is the logit of p(1|x).

we have

$$\hat{p}(1|x) \in \begin{cases} [0,\ \frac{1+e^{-(*)}}{2+e^{\theta}+e^{-(*)}}] & , \text{ if } f(x) \leq -(*) \\ \frac{1+e^{f(x)}}{2+e^{\theta}+e^{f(x)}} & , \text{ if } -(*) < f(x) \leq -\theta \\ \frac{e^{f(x)}}{1+e^{f(x)}} & , \text{ if } -\theta < f(x) \leq \theta \\ \frac{1+e^{\theta}}{2+e^{\theta}+e^{-f(x)}} & , \text{ if } \theta < f(x) \leq (*) \\ [\frac{1+e^{\theta}}{2+e^{\theta}+e^{-(*)}},\ 1] & , \text{ if } f(x) > (*). \end{cases} \tag{4.4.2}$$

Note the hybrid loss doesn't give an exact estimate of the probability when the population minimizer (or the estimate from the data) is above or below some thresholds.

To see if the hybrid loss generates sparse solutions, we use some results from Steinwart (2003). We can show (Appendix B.3) that the proportion of support vectors for the hybrid loss is bounded below by some constant in probability.

**Proposition 4.4.1.** *Let $K$ be a universal kernel (Steinwart, 2001) and $P$ be a probability measure on $\mathcal{X} \times \{\pm 1\}$ with no discrete components on $\mathcal{X}$. Then for the hybrid loss SVM using a regularization sequence $(\lambda_n)$ with $\lambda_n \to 0$, $n\lambda_n^3 \to \infty$, $n\lambda_n^2/\log n \to \infty$, and all $\epsilon > 0$, we have*

$$P^n(T \in (\mathcal{X} \times \{\pm 1\})^n : \#SV(f_{T,\lambda_n}) \geq (S_{L,P} - \epsilon)n) \to 1,$$

*where $f_{T,\lambda_n}$ is the solution to the SVM optimization problem (4.3.1) with the data $T = \{((x_1, y_1), \cdots, (x_n, y_n)) \in (\mathcal{X} \times \{\pm 1\})^n\}$ and smoothing parameter $\lambda_n$, $S_{L,P}$ is equal to*

$$\mathbb{E}\left[p(1|X) I_{\left\{logit(p(1|X)) \leq \ln \frac{1+e^{\theta}}{1+e^{-(*)}}\right\}} + (1 - p(1|X)) I_{\left\{logit(p(1|X)) \geq \ln \frac{1+e^{-(*)}}{1+e^{\theta}}\right\}}\right].$$

If the data is distributed such that the conditional probability $p(1|X) = P(Y = 1|X)$ takes some values close to 0 or 1 with non-zero probabilities with respect to the marginal probability $P_X$, we would expect some sparsity in the solution of the hybrid loss. Note that the Gaussian kernel (among others) is universal (Steinwart, 2001) which basically requires any continuous function from $\mathcal{X}$ to $\mathbb{R}$ can be approximated as close as one wants by some function in $\mathcal{H}_K$ in terms of infinity norm.

## 4.5 The Optimization Problem

Plugging the hybrid loss into equation (4.3.3) and introducing some slack variables $\xi_i$'s to relax the the hybrid loss $l(\cdot)$ at the right corner point $(*)$, we get the mathematical program for the hybrid loss SVM

$$\min_{c,d,\xi} e^T\xi + n\lambda c^T K_n c, \quad \text{subject to} \begin{cases} l_1(y_i(\sum_{j=1}^n c_j K(x_i, x_j) + d)) \leq \xi_i \\ \xi_i \geq 0, \ \forall\ 1 \leq i \leq n. \end{cases} \tag{4.5.1}$$

In the above expression, $\xi = (\xi_1, \xi_2, \cdots, \xi_n)^T$, $e = (\overbrace{1, 1, \cdots, 1}^{n})^T$. The function $l_1(\tau)$ is equal to $l(\tau)$ when $\tau \leq (*)$; passing that point, $l_1(\tau)$ just extends $l(\tau)$ all the way down. $l_1(\cdot)$ has only 2 pieces instead of 3.

A slight transformation of the above problem is

$$\min_{c,d,\xi} e^T\xi + n\lambda c^T K_n c, \quad \text{subject to} \begin{cases} y_i(\sum_{j=1}^n c_j K(x_i, x_j) + d) \geq g(\xi_i) \\ \xi_i \geq 0, \ \forall\ 1 \leq i \leq n, \end{cases} \tag{4.5.2}$$

where $g(\cdot)$ is the inverse function of $l_1(\cdot)$. Since $l_1(\cdot)$ is strictly decreasing and differentiable everywhere, $g(\cdot)$ has the same properties. The final simplification of the optimization problem is

$$\min_{c,d,\eta} e^T l_1(\eta) + n\lambda c^T K_n c, \quad \text{subject to} \begin{cases} y_i(\sum_{j=1}^n c_j K(x_i, x_j) + d) \geq \eta_i \\ \eta_i \leq (*), \ \forall\ 1 \leq i \leq n, \end{cases} \tag{4.5.3}$$

where $\eta = (\eta_1, \eta_2, \cdots, \eta_n)^T$, and $l_1(\eta) = (l_1(\eta_1), l_1(\eta_2), \cdots, l_1(\eta_n))^T$, and $(*)$ is defined as before.

Note the optimization problem now has nonlinear objective and linear constraints which are in a form readily solvable with the MINOS package (Murtagh and Saunders, 1983). I wrote some Fortran code to compute the hybrid loss support vector machine with MINOS. Some further derivation can be done to find the dual problem of the optimization problem. The KKT conditions can be found after the dual problem is derived.

## 4.6 Choosing the Parameters

For the support vector machine under the regularized RKHS framework, we always face the problem of choosing the right smoothing parameter $\lambda$. If we use the Gaussian kernel for the reproducing kernel, we need to choose the kernel parameter $\sigma^2$ as well. Note for the hybrid loss, there is another parameter $\theta$ that controls how much to truncate. All of these parameters requires some way of choosing them.

We use the 5-fold cross validation for choosing both the smoothing parameter $\lambda$ and the kernel parameter $\sigma^2$. The method was used because it is available with any SVM, and we need to compare the hybrid loss against the hinge loss and the logistic loss. Also, 5-fold seems to be a compromise between the leave-out-one and the 10-fold. The 5-fold cross validation actually works well for a lot of applications.

To choose the parameter $\theta$ in the loss function, we have to use a different procedure. Notice the plot for the hybrid loss function (Figure 4.2.1), the loss is nondecreasing with the increase of $\theta$ and a given margin $\tau$. We make a table for different $\theta$'s, where each $\theta$ corresponds to a truncation proportion. We choose the $\theta$ that gives a desired proportion.

## 4.7 Simulation Study

We did some simulation to see if the hybrid loss is doing what we expected. Both one dimensional case and two dimensional case were tried. Gaussian kernels were used with all the SVMs, and 5-fold cross validation was used to choose the smoothing parameters and kernel parameters. The value $\theta$ used in the hybrid loss SVM was 1.0.

### 4.7.1 1-D Simulation

For the 1-D case, we sampled 300 data points from the normal distribution $N(0,2)$ as the values for the attribute variable $X$. The conditional probability function $p(1|x)$ had the form $(0.5\sin(0.4\pi x)+0.5)\cdot I_{\{-1.25<x<1.25\}}+I_{\{x\geq 1.25\}}$. This means about one third $(2(1-\Phi(\frac{1.25}{\sqrt{2}}))\approx 0.38)$ of the data has probabilities equal to either 0 or 1, , where $\Phi(\cdot)$ is the standard normal cumulative distribution function. We wanted to see how the estimates from the hybrid loss differ from the ones from the logistic loss and those from the hinge loss.

The results are shown in Figure 4.7.1. The performance of the logistic loss was quite surprising, which shows the true conditional probability is estimated accurately. The smoothing parameter might have played some role in conditioning the estimates. The estimates from the hinge loss are close to the population minimizer $sign(p(1|x)-1/2)$. The estimates from the hybrid loss go to its population minimizer too. The hybrid loss does give a solution that is a mix of the solution from the hinge loss and the one from the logistic loss.

### 4.7.2 2-D Simulation

For the 2-D simulation, we sampled 400 $(X_1, X_2)$'s from the uniform distribution on the square $[-1,1]^2$. The conditional distribution $p(1|(x_1,x_2))$ was chosen to be $[0.5\sin(\sqrt{2}\pi(r-\frac{3\sqrt{2}}{8}))+0.5]\cdot I_{\{\frac{\sqrt{2}}{8}<r<\frac{5\sqrt{2}}{8}\}}+I_{\{r\geq\frac{5\sqrt{2}}{8}\}}$, where $r=\sqrt{x_1^2+x_2^2}$, the distance between the point $(x_1,x_2)$ and the origin. This conditional distribution implies that about 41% of data points have conditional probability $p(1|(x_1,x_2))$

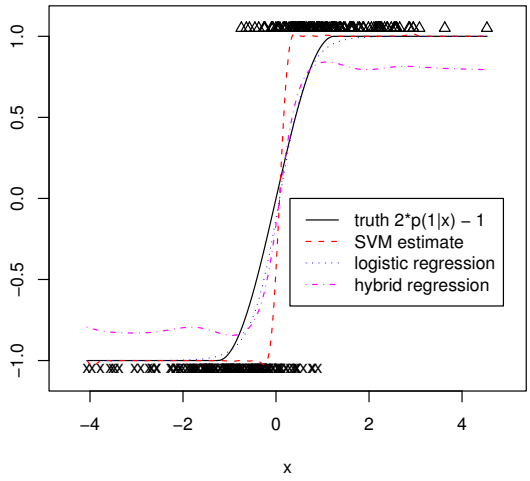



Figure 4.7.1: 1-D simulation results. The triangles and crosses represent the data for positive class and negative class respectively with their horizontal positions corresponding to the $x$ values. The raw estimates from hinge loss SVM were plotted. For the logistic regression and the hybrid method, the transformation $2e^f/(1+e^f) - 1$ was applied to the estimates.

either 0 or 1. The SVMs with the 3 different loss functions were fitted to the data. The result for the hybrid loss is given in Figure 4.7.2.

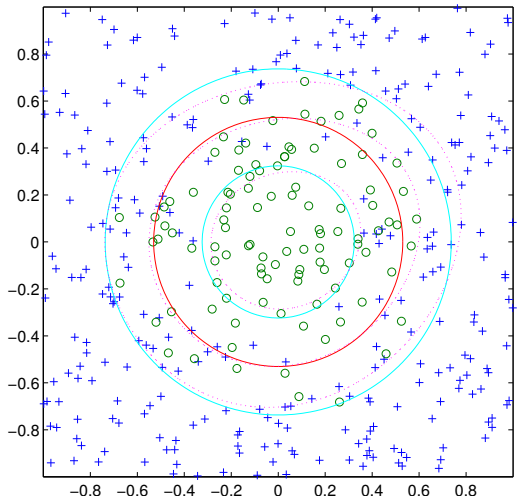



Figure 4.7.2: 2-D simulation result. The solid circles from the innermost to the outermost are the $x$'s with the $logit(p(1|x))$ equal to $-(*)$, 0, $(*)$ respectively. The dotted lines are the contours on the estimates from the hybrid method with the corresponding levels.

The plot shows that the solution from the hybrid loss goes to the population minimizer with the selection of the $\lambda$ and $\sigma^2$ by 5-fold cross validation. This implies that we can use the hybrid loss to target a specific value of the conditional probability, while the new loss provides further conditioning of the estimates from the logistic loss with a smoothing parameter.

Table 4.8.1: Summary of Wisconsin Breast Cancer Data

| item | value |
|---|---|
| number of instances | 699 |
| number of attributes | 9 |
| attribute type | integer value 1-10 |
| number of classes | 2 |
| missing attribute values | 16 with missing values |
| class distribution | 458 benign, 241 malignant |

## 4.8 Real Data Example

To see how the hybrid loss is doing as a classification method and its other properties, we applied the new loss along with the hinge loss and the logistic loss to the Wisconsin Breast Cancer data downloaded from UCI machine learning data repository (URL http://www.ics.uci.edu/∼mlearn/MLRepository.html).

The data set we used was donated by Dr. Olvi Mangasarian of University of Wisconsin-Madison. It has been used in Wolberg and Mangasarian (1990), Zhang (1992), and Aaron (1998). The classification accuracy was high, at around 93%.

The summary of the data is given in Table 4.8.1. We can see all the attributes are ordered numbers, and there are some cases with missing values. To simplify the computation, I just deleted all the cases with missing attributes and treated the scale numbers as truly numeric. The actual data set being used has size 683.

The number of misclassified cases for the hinge loss, the logistic loss, and the

hybrid loss on the 683 cases with the $(\lambda, \sigma^2)$'s selected by 5-fold cross validation and $\theta$ set to 1 are 18, 17, 16 respectively. The accuracy is around 97%. It appears that number of support vectors for hybrid loss is between that for the hinge loss and that for the logistic loss, with the hinge loss having the most sparse solution. The box-plots of of the raw estimates from the 3 loss functions are given in Figure 4.8.1. It shows that the hybrid loss has a truncation effect on the estimates relative to those from the logistic loss. The histogram of the transformed estimates (with the inverse logit transformation) for the logistic loss and that for the hybrid loss are given in Figure 4.8.2 (a) and (b) respectively. We can see the overall shape of the estimates from hybrid loss still looks like that of the logistic loss, with the truncation effect quite evident.

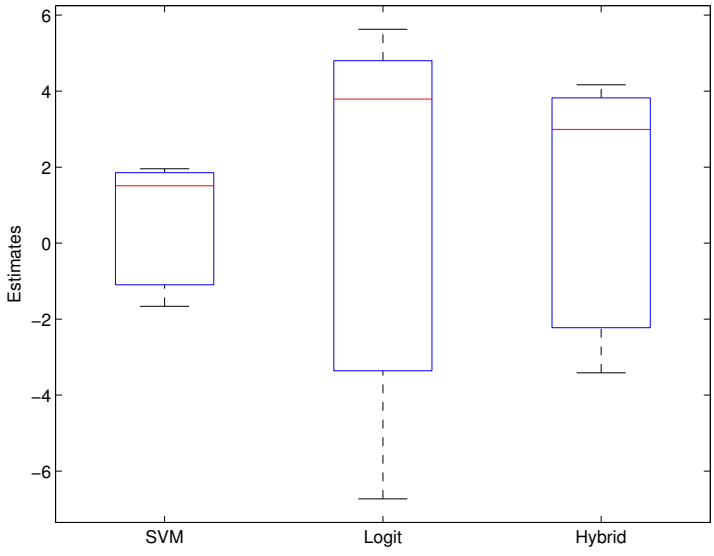



Figure 4.8.1: Box-plots of the estimates from different loss functions on Wisconsin Breast Cancer data. Logit stands for logistic loss in the plot.

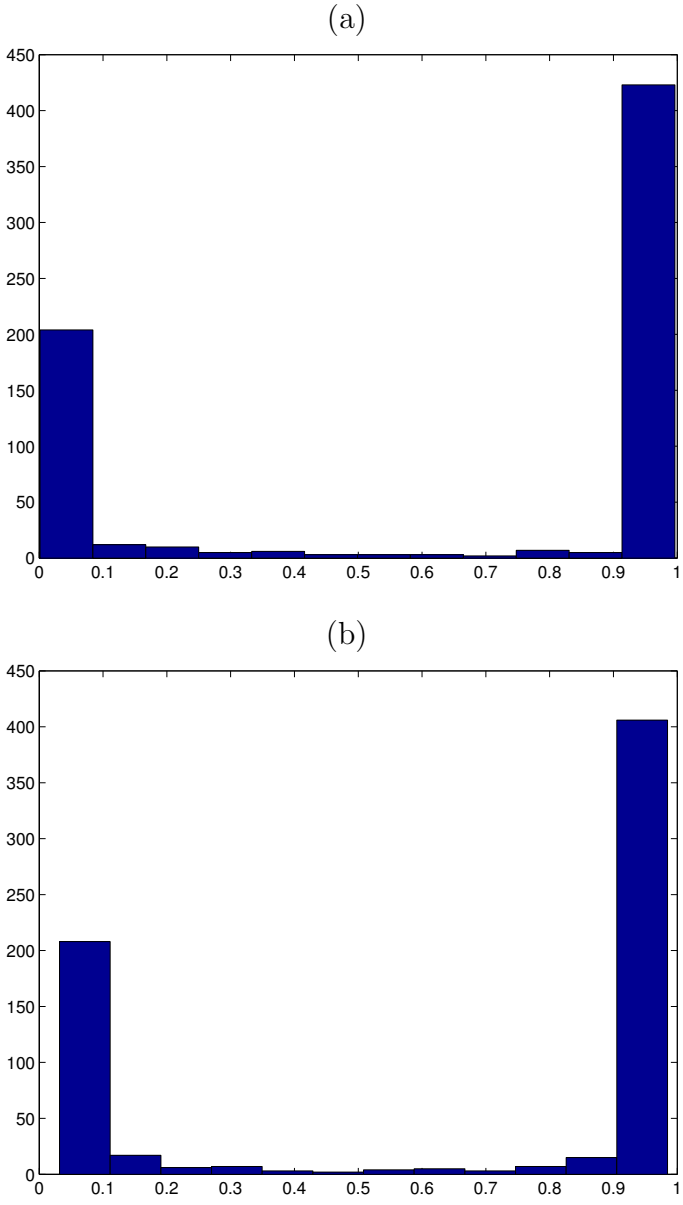



Figure 4.8.2: Histograms of the estimates from 2 loss functions on Wisconsin Breast Cancer data. (a) Logistic loss; (b) Hybrid loss. The estimates were transformed with the function $e^f/(1+e^f)$.

## 4.9 Summary and Discussion

In this chapter, we presented a new loss function that is a hybrid of the logistic loss and the hinge loss. We expected our new loss to have the advantages of the previous 2 loss functions, i.e., it can estimate probabilities in certain region and give sparse solutions at the same time. The population minimizer of the new loss function and the asymptotic bound for the proportion of support vectors were given. The mathematical program for the SVM with the hybrid loss was derived and the SVM was implemented with MINOS. Simulations with the new loss function were done and application to Wisconsin Breast Cancer data were tried. Our new loss function seems to deliver what we expected. And it does a reasonable job in terms of classification as well.

Some further work on the new SVM would be to find some other (maybe better) ways of selecting the smoothing parameter, the kernel parameter, and the parameter in the loss function. The computation might need to be sped up a little bit. Finally, finding more applications for the hybrid loss requires some more work.

# Chapter 5

# Concluding Remarks

We presented 3 topics in the thesis. The first two were related to the magnetic resonance image analysis. And the third one was on studying a new hybrid loss for support vector machine.

For the first topic, MRI segmentation with thin plate spline thresholding, a novel smoothing technique was proposed for structural images. Due to the huge data size of image data, we designed a divide and conquer technique to smooth the data, and thresholds were found during the smoothing process. The combination of the smoothing and thresholding seems to do a very good job in terms of segmentation. Many indices were computed for comparing the new segmentation method against the other methods. And contours and isosurfaces were drawn to show the boundaries and surfaces found by the methods. There are a lot of of directions worth further studies.

The second topic was on functional MRI analysis. Because the real time series will never have the covariance structure as one expects. Many techniques were come up to cope with misspecification. We showed by simulation that partial spline with its corresponding hypothesis testing is a good way for fMRI

analysis. Also, the amount of smoothing determined by generalized cross validation seems to be about right for the splines. Application of partial splines to the image volume of the whole brain warrants some prospects.

The last topic was on the hybrid loss for support vector machine. We proposed a new loss function to achieve the sparsity and to be able to estimate the conditional probabilities in certain region. The theoretical proofs were done and mathematical program was derived. The support vector machine was implemented with MINOS. The hybrid loss seems to have both the nice properties of the hinge loss and the logistic loss. This topic can be related to the MRI analysis if we can reduce the image for each subject to some attribute vector and classification needs to be done.

All of these topics have in common is the smoothing, with the first one using the thin plate spline, the second one using the partial spline, and the third one utilizing the Gaussian kernel smoothing spline in the classification setting. Different ways of choosing the smoothing parameters and other parameters were studied. In all the applications, cross validation seems to be a very useful tool, be it the generalized cross validation with a constant factor, the generalized cross validation, or the 5-fold cross validation.

In summary, we studied smoothing in 3 different settings. We expect more problems can be solved with clever ways of smoothing.

# Appendix A

# Some Computer Code

## A.1 Conversion of Data to VTK Format

```
function vertface2vtk(v,f,fname,description)
% VERTFACE2VTK Save a set of vertice coordinates and faces to
%     vtk format.
% VERTFACE2VTK(v,f,fname,description)
%     v is a Nx3 matrix of vertex coordinates.
%     f is a Mx3 matrix of vertex indices.
%     fname is the filename for the output vtk data.
%     description is a brief description of the data.

if (nargin<=3), description = ' '; end

fid = fopen(fname,'w');

fprintf(fid, '# vtk DataFile Version 1.0\n');
fprintf(fid, '%s\n', description);
```

```
fprintf(fid, 'ASCII\n\n');

npts = size(v,1);
fprintf(fid, 'DATASET POLYDATA\n');
fprintf(fid, 'POINTS %d float\n', npts);
for i=1:npts
  fprintf(fid,'%f %f %f\n',v(i,1),v(i,2),v(i,3));
end

nfaces = size(f,1);
fprintf(fid,'POLYGONS %d %d\n', nfaces, nfaces*4);

for i=1:nfaces
  fprintf(fid,'3 %d %d %d\n',f(i,1)-1,f(i,2)-1,f(i,3)-1);
end

fclose(fid);
```

## A.2 Rendering a Surface with VTK

```
# This example shows how to use VTK to render a surface which
# contains a lot of details. The following is in TCL language.

package require vtk
package require vtkinteraction

# We start by reading some data that was generated by our method
vtkPolyDataReader reader
    reader SetFileName "$VTK_DATA/surf_gray_white_tps.vtk"

# We want to preserve topology (not let any cracks form). This
# may limit the total reduction possible, which we have specified
# at 70%.
vtkDecimatePro deci
    deci SetInput [reader GetOutput]
    deci SetTargetReduction 0.7
    deci PreserveTopologyOn
vtkPolyDataNormals normals
    normals SetInput [reader GetOutput]
    normals FlipNormalsOn
vtkPolyDataMapper isosurfMapper
```

```
    isosurfMapper SetInput [normals GetOutput]
vtkActor isosurfActor
    isosurfActor SetMapper isosurfMapper
    eval [isosurfActor GetProperty] SetColor 0.8 0.7 0.7

# Create the RenderWindow, Renderer and the Interactor
vtkRenderer ren1
vtkRenderWindow renWin
    renWin AddRenderer ren1
vtkRenderWindowInteractor iren
    iren SetRenderWindow renWin

# Add the actor to the renderer, set the background and size
ren1 AddActor isosurfActor
ren1 SetBackground 1 1 1
renWin SetSize 250 250

# add 2 events to the renderer
iren AddObserver UserEvent {wm deiconify .vtkInteract}
iren AddObserver KeyPressEvent KeyPress

vtkWindowToImageFilter w2i
```

```
vtkPostScriptWriter psw


proc KeyPress {} {
    set key [iren GetKeySym]
    if { $key == "z" } {
        puts "write image to ps file."


        w2i SetInput renWin
        w2i Modified


        psw SetInput [w2i GetOutput]
        psw SetFileName "vtkimgout.ps"
        psw Write
    }
}


# set the camera
vtkCamera cam1
    cam1 SetFocalPoint 0 0 0
    cam1 SetPosition 1 1 1
    cam1 ComputeViewPlaneNormal
    cam1 SetViewUp 1 0 0
```

```
    cam1 OrthogonalizeViewUp
ren1 SetActiveCamera cam1
ren1 ResetCamera

iren Initialize

# prevent the tk window from showing up then start the event loop
wm withdraw .
```

# Appendix B

# Derivation and Proof

## B.1 Derivation of the Population Minimizer for the Hybrid Loss

Denoting $\mathbb{E}[l(Yf(X))|X = x]$ by $\mathbb{E}(f)$ for short, we have

$$\mathbb{E}(f) = \begin{cases} p{\cdot}\ln(1+e^{-f}) & , \quad f \leq -(*) \\ p{\cdot}\ln(1+e^{-f}) + (1-p)[\ln(1+e^{-\theta}) + \frac{f+\theta}{1+e^{\theta}}] & , \quad -(*) < f \leq -\theta \\ p{\cdot}\ln(1+e^{-f}) + (1-p){\cdot}\ln(1+e^{f}) & , \quad -\theta \leq f < \theta \\ p[\ln(1+e^{-\theta}) - \frac{f-\theta}{1+e^{\theta}}] + (1-p){\cdot}\ln(1+e^{f}) & , \quad \theta < f \leq (*) \\ (1-p){\cdot}\ln(1+e^{f}) & , \quad f > (*). \end{cases} \tag{B.1.1}$$

Then taking partial derivative of the above function w.r.t. $f$, we get

$$\frac{\partial}{\partial f}\mathbb{E}(f) = \begin{cases} -\frac{p}{1+e^{f}} & , \quad f \leq -(*) \\ -\frac{p}{1+e^{f}} + \frac{1-p}{1+e^{\theta}} & , \quad -(*) < f \leq -\theta \\ -\frac{p}{1+e^{f}} + \frac{(1-p)e^{f}}{1+e^{f}} & , \quad -\theta < f \leq \theta \\ -\frac{p}{1+e^{\theta}} + \frac{(1-p)e^{f}}{1+e^{f}} & , \quad \theta < f \leq (*) \\ \frac{(1-p)e^{f}}{1+e^{f}} & , \quad f > (*). \end{cases} \tag{B.1.2}$$

In the above equations $(*) = (1+e^{\theta})\ln(1+e^{\theta}) - \theta e^{\theta}$, and $p = P(Y=1|X=x)$.

Noting the signs of the derivatives when $f$ is in different intervals, and letting the derivative be 0 in the interval if 0 is attainable, we get the minimizer $\hat{f}(x)$ of the expected conditional loss as in the following,

$$\hat{f}(x) = \begin{cases} -(*) & , \text{ if } \text{logit}(p) \leq \ln\frac{1+e^{-(*)}}{1+e^{\theta}} \\ -\text{logit}(\frac{1-p}{p}\frac{1}{1+e^{\theta}}) & , \text{ if } \ln\frac{1+e^{-(*)}}{1+e^{\theta}} < \text{logit}(p) \leq -\theta \\ \text{logit}(p) & , \text{ if } -\theta < \text{logit}(p) \leq \theta \\ \text{logit}(\frac{p}{1-p}\frac{1}{1+e^{\theta}}) & , \text{ if } \theta < \text{logit}(p) \leq \ln\frac{1+e^{\theta}}{1+e^{-(*)}} \\ (*) & , \text{ if } \text{logit}(p) > \ln\frac{1+e^{\theta}}{1+e^{-(*)}}. \end{cases} \tag{B.1.3}$$

For the special cases of $p(1|x) = 0$ or 1, it is easy to find the corresponding $\hat{f}(x)$ is any number $\leq -(*)$ or any number $\geq (*)$. ■

## B.2 Proof of the Fisher Consistency of the Hybrid Loss

Using some results from Lin (2001), we only need to check 2 conditions and to show that the global minimizer $\hat{f}(\cdot)$ of $\mathbb{E}[l(Yf(X))]$ exists.

The 2 conditions in Lin (2001) are:

1. $l(\tau) < l(-\tau), \forall \tau > 0$.

2. $l'(0) \neq 0$ exists.

The first condition is satisfied by the hybrid loss with the parameter $\theta$ satisfying $(*) = (1 + e^{\theta})\ln(1 + e^{\theta}) - \theta e^{\theta} > 0$. Similarly, the second condition is satisfied with the same restriction.

The global minimizer of of $\mathbb{E}[l(Yf(X))]$ exists based on our previous derivation and the fact that the marginal p.d.f. $p(x) \geq 0, \forall x$.

The thing remaining to show is $(*) > 0, \forall \theta \in \mathbb{R}$. Consider first the $\theta \leq 0$ case, $(1 + e^{\theta})\ln(1 + e^{\theta}) > 0$ in this case, and $-\theta e^{\theta} \geq 0$ too. If $\theta > 0$, $1 + e^{\theta} > e^{\theta}$ and $\ln((1 + e^{\theta}) > \theta > 0$. In both cases, we have $(*) > 0$. So the hybrid loss is Fisher consistent. ■

## B.3 Proof of the Sparseness of the Hybrid Loss

We will prove the sparseness of the hybrid loss for the $\theta \geq 0$ case. The proof of the $\theta < 0$ case is similar, but more technical handling is required.

Using the Theorem in Steinwart (2003), we only need to check a few conditions and calculate some quantities for the hybrid loss. In Steinwart's notation, $L(y, t)$ is equal to $l(yt)$, where $l(\cdot)$ is the hybrid loss.

First we need to show that the hybrid loss is strongly admissible. Notice that the Fisher consistency implies admissibility (i.e. the population minimizer estimates the correct sign when $p(1|x)$ is not equal to $1/2$), the thing remaining to be shown is the cardinality of $F_L^*(a)$ is 1 for all $a \in (0, 1)$ with $a \neq 1/2$. This is evident from equation (B.1.3).

Secondly we need to show that the hybrid loss is regular. Using Steinwart's

definition and noting that the hybrid loss is convex, we have

$$|L_{|Y\times[-a,a]}|_1 = \frac{e^a}{1+e^a}, \quad \forall\, a > 0. \tag{B.3.1}$$

In the above expression, we used the fact that the hybrid loss is differentiable except at one corner point $(*)$. The inequality $f(x_1) - f(x_0) \geq f'(x_0)(x_1 - x_0)$ holds for any convex and differentiable function $f$.

Notice that $e^a/(1+e^a)$ takes value in $(1/2, 1)$ for any $a > 0$, we have

$$|L_{|Y\times[-\gamma a,\gamma a]}|_1 < 1 < 2|L_{|Y\times[-a,a]}|_1, \quad \forall\, a > 0, \gamma > 0. \tag{B.3.2}$$

The infinity norm of the hybrid loss in a given interval is

$$\|L_{|Y\times[-a,a]}\|_\infty = \ln(1+e^a). \tag{B.3.3}$$

Using the trivial inequality $1 + x^\gamma < (1+x)^\gamma$, for any $x > 0$ and $\gamma > 1$, we get

$$\ln(1+e^{\gamma a}) < \ln(1+e^a)^\gamma = \gamma \ln(1+e^a), \quad \forall\, a > 0, \gamma > 1. \tag{B.3.4}$$

For the case $0 < \gamma \leq 1$, we have

$$\ln(1+e^{\gamma a}) \leq \ln(1+e^a), \quad \forall\, a > 0, 0 < \gamma \leq 1. \tag{B.3.5}$$

If we let $c_\gamma = \max\{2, \gamma\}$, and also notice the monotone property of the loss function $L(y,t)$ for given $y = 1$ and $y = -1$, we proved the hybrid loss is regular.

The hybrid loss is convex by our definition. We now need to calculate a few quantities. For the hybrid loss, $\delta_\lambda = \sqrt{\frac{L(1,0)+L(-1,0)}{\lambda}} = \sqrt{\frac{2\ln 2}{\lambda}}$, and $C =$

$\sup\{\sqrt{K(x,x)} : x \in \mathcal{X}\}$ is a constant for a given kernel ($C = 1$ for Gaussian kernel), we have

$$|L_{\lambda_n}|_1 = \frac{e^{\sqrt{\frac{2\ln 2}{\lambda_n}}C}}{1 + e^{\sqrt{\frac{2ln2}{\lambda_n}}C}} = O(1), \quad \text{as } \lambda_n \to 0, \tag{B.3.6}$$

and

$$\|L_{\lambda_n}\|_\infty = \ln(1 + e^{\sqrt{\frac{2ln2}{\lambda_n}}C}) = O(\lambda_n^{-1/2}), \quad \text{as } \lambda_n \to 0. \tag{B.3.7}$$

Since $F_L^*(1/2) = 0$ and the loss function $L(y,t)$ is differentiable at $t = 0$ for both $y = 1$ and $y = -1$ when the parameter $\theta \geq 0$, we then have $\partial_2 L(\pm 1, F_L^*(1/2)) = \mp 1/2$ respectively, this implies $S_{L,P} = P(S)$, where

$$S = \{(x,y) \in \mathcal{X}_{cont} \times \{\pm 1\} : 0 \notin \partial_2 L(y, F_L^*(p(1|x)) \cap \mathbb{R})\}. \tag{B.3.8}$$

In the above expression, $\mathcal{X}_{cont} = \{x \in \mathcal{X} : P_X(\{x\}) = 0\}$ and $\mathbb{R} = (-\infty, \infty)$. By focusing our attention to the class of distribution $P(X,Y)$ with continuous marginal distribution $P_X$ (i.e. $\mathcal{X}_{cont} = \mathcal{X}$), we have

$$S = \left\{(x,1) : \eta(x) \leq \ln \frac{1+e^\theta}{1+e^{-(*)}}\right\} \bigcup \left\{(x,-1) : \eta(x) \geq \ln \frac{1+e^{-(*)}}{1+e^\theta}\right\}, \tag{B.3.9}$$

where $\eta(x) = logit(p(1|x))$.

The proportion of support vectors can be bounded below by

$$S_{L,P} = P(S) = E\left[p(1|X) I_{\left\{\eta(X) \leq \ln \frac{1+e^\theta}{1+e^{-(*)}}\right\}} + (1 - p(1|X)) I_{\left\{\eta(X) \geq \ln \frac{1+e^{-(*)}}{1+e^\theta}\right\}}\right]$$

in probability if the kernel $K$ is universal, and the regularization sequence $(\lambda_n)$ satisfies $\lambda_n \to 0$, $n\lambda_n^3 \to \infty$, and $n\lambda_n^2 / \log n \to \infty$. ■

# Bibliography

Aaron, K. (1998), Group classification in using a mix of genetic programming and genetic algorithms, *in* 'Proceedings of the 1998 ACM Symposium on Applied Computing', pp. 308–312.

Ashburner, J. & Friston, K. J. (2000), 'Voxel-Based Morphometry - The Methods', *NeuroImage* **11**(6), 805–821.

Ashburner, J., Neelin, P., Collins, D. L., Evans, A. C. & Friston, K. J. (1997), 'Incorporating prior knowledge into image registration', *NeuroImage* **6**, 344–352.

Bartlett, P. L. (2003), 'Some notes on estimating posterior probabilities with large margin classifiers', unpublished manuscript.

Bartlett, P. L. & Tewari, A. (2004), 'Sparseness vs estimating conditional probabilities: Some asymptotic results', submitted. URL http://www.stat.berkeley.edu/∼bartlett/papers/bt-secp-04.pdf.

Bates, D. M., Lindstrom, M. J., Wahba, G. & Yandell, B. S. (1987), 'GCVPACK-Routines for generalized cross validation', *Commun. Statist.-Simula.* **16**(1), 263–297.

Bezdek, J. C., Hall, L. O. & Clarke, L. P. (1993), 'Review of MR image segmentation techniques using pattern recognition', *Medical Physics* **20**(4), 1033–1048.

Boesen, K., Rehm, K., Schaper, K., Stoltzner, S., Woods, R., Lüders, E. & Rottenberga, D. (2004), 'Quantitative comparison of four brain extraction algorithms', *NeuroImage* **22**, 1255–1261.

Bouix, S., Martin-Fernandez, M., Ungar, L., McCarley, R. W. & Shenton, M. E. (2005), 'Automatic evaluation of brain MR segmentation techniques by level of agreement', 11th Annual Meeting of the Organization for Human Brain Mapping.

Bullmore, E., Brammer, M., Williams, S., Rabe-Hesketh, S., Janot, N., David, A., Mellers, J., Howard, R. & Sham, P. (1996), 'Statistical methods of estimation and inference for functional MR image analysis', *Magnetic Resonance in Medicine* **35**, 261–277.

Carew, J., Wahba, G., Xie, X., Nordheim, E. & Meyerand, M. (2003), 'Optimal spline smoothing of fMRI time series by generalized cross-validation', *NeuroImage* **18**, 950–961.

Chiang, A., Wahba, G., Tribbia, J. & Johnson, D. (1999), A quantitative study of smoothing spline-ANOVA based fingerprint methods for attribution of global warming, Technical Report 1010, Department of Statistics, University of Wisconsin-Madison.

Chung, M. K., Dalton, K. M., Alexander, A. L. & Davidson, R. J. (2004), 'Less white matter concentration in autism: 2D Voxel-Based Morphometry', *NeuroImage* **23**, 242–251.

Dale, A. M. & Fischl, B. (1999), 'Cortical surface-based analysis: I. segmentation and surface reconstruction', *NeuroImage* **9**, 179–194.

Davatzikos, C. & Bryan, R. N. (1995), Using a deformable surface model to obtain a shape representation of the cortex, *in* 'Proceedings of the Intl. Symp. on Comp. Vision', pp. 212–217.

Friston, K. J., Holmes, A. P., Poline, J.-B., Grasby, P. J., Williams, S. C. R., Frackowiak, R. S. J. & Turner, R. (1995), 'Analysis of fMRI time-series revisited', *Neuroimage* **2**, 45–53.

Friston, K. J., Josephs, O., Zarahn, E., Holmes, A. P., Rouquette, S. & Poline, J.-B. (2000), 'To smooth or not to smooth? Bias and efficiency in fMRI time-series analysis', *Neuroimage* **12**, 196–208.

Hastie, T., Tibshirani, R. & Friedman, J. (2001), *The elements of statistical learning. Data mining, inference, and prediction*, Springer, New York.

Hoffmann, T. J., Chung, M. K., Dalton, K. M., Alexander, A. L., Wahba, G. & Davidson, R. J. (2004), 'Subpixel curvature estimation of the corpus callosum via splines and its application to autism', 10th Annual Meeting of the Organization for Human Brain Mapping.

Kapur, T. (1995), Segmentation of brain tissue from magnetic resonance images, Technical Report 1566, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Kitware, Inc. (2003), *The VTK User's Guide*, Kitware, Inc.

Kollokian, V. (1996), Performance analysis of automatic techniques for tissue classification in MRI of the human brain, Master's thesis, Concordia University, Montreal, Canada.

Kovacevic, N., Lobaugh, N. J., Bronskill, M. J., Levine, B., Feinstein, A. & Black, S. E. (2002), 'A robust method for extraction and automatic segmentation of brain images', *NeuroImage* **17**, 1087–1100.

Lin, Y. (2001), A note on margin-based loss functions in classification, Technical Report 1044, Department of Statistics, University of Wisconsin-Madison.

Luo, Z. & Wahba, G. (1997), 'Hybrid adaptive splines', *Journal of the American Statistical Association* **92**(437), 107–116.

MacDonald, D., Kabani, N., Avis, D. & Evans, A. C. (2000), 'Automated 3-D extraction of inner and outer surfaces of cerebral cortex from MRI', *NeuroImage* **12**, 340–356.

Morrison, M. & Attikiouzel, Y. (1992), A probabilistic neural network based image segmentation network for magnetic resonance images, *in* 'Proc. Conf. Neural Networks', Vol. 3, pp. 60–65.

Mumford, D. & Shah, J. (1985), Boundary detection by minimizing functionals, *in* 'IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 22–26.

Murtagh, B. A. & Saunders, M. A. (1983), MINOS 5.5 user's guide, Technical Report SOL 83-20R, Systems Optimization Laboratory, Stanford University. revised July 1999.

Ozkan, M., Dawant, B. M. & Maciunas, R. J. (1993), 'Neural-network-based segmentation of multi-modal medical images: a comparative and prospective study', *IEEE Trans. Med. Imaging* **12**, 534–544.

Sethian, J. A. (1999), *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science*, Cambridge University Press, Cambridge, UK.

Shan, Z. Y., Yue, G. H. & Liu, J. Z. (2002), 'Automated histogram-based brain segmentation in T1-weighted three-dimensional magnetic resonance head images', *NeuroImage* **17**, 1587–1598.

Sled, J. G., Zijdenbos, A. P. & Evans, A. C. (1998), 'A nonparametric method for automatic correction of intensity nonuniformity in MRI data', *IEEE Transactions on Medical Imaging* **17**(1), 87–97.

Steinwart, I. (2001), 'On the influence of the kernel on the consistency of support vector machine', *Journal of Machine Learning Research* **2**, 67–93.

Steinwart, I. (2003), 'Sparseness of support vector machines', *Journal of Machine Learning Research* **4**, 1071–1105.

Tannenbaum, T., Wright, D., Miller, K. & Livny, M. (2001), Condor – a distributed job scheduler, *in* T. Sterling, ed., 'Beowulf Cluster Computing with Linux', MIT Press.

Terzopoulos, D., Witkin, A. & Kass, M. (1988), 'Constraints on deformable models', *Artificial Intelligence* **36**(1), 91–124.

Thain, D., Tannenbaum, T. & Livny, M. (2004), 'Distributed computing in practice: The Condor experience', *Concurrency and Computation: Practice and Experience*. to appear.

Tohka, J., Zijdenbos, A. & Evans, A. C. (2004), 'Fast and robust parameter estimation for statistical partial volume models in brain MRI', *NeuroImage* **23**(1), 84–97.

Wahba, G. (1978), 'Improper priors, spline smoothing and the problem of guarding against model errors in regression', *Journal of Royal Statistical Society. Series B* **40**(3), 364–372.

Wahba, G. (1990), *Spline models for observational data*, SIAM, Philadelphia.

Wahba, G. (2002), Soft and hard classification by reproducing kernel Hilbert space methods, *in* 'Proceedings of the National Academy of Sciences', pp. 16524–16530.

Wang, Y., Adah, T. & Kung, S.-Y. (1998), 'Quantification and segmentation of brain tissues from MR images: A probabilistic neural network approach', *IEEE Transactions on Image Processing* **7**(8), 1165–1181.

Wolberg, W. H. & Mangsarian, O. L. (1990), Multisurface method of pattern separation for medical diagnosis applied to breast cytology, *in* 'Proceedings of the National Academy of Sciences', pp. 9193–9196.

Wood, S., Jiang, W. & Tanner, M. (2002), 'Bayesian mixture of splines for spatially adaptive nonparametric regression', *Biometrika* **89**(3), 513–528.

Worsley, K. J. & Friston, K. J. (1995), 'Analysis of fMRI time-series revisited-again', *Neuroimage* **2**, 173–181.

Zhang, J. (1992), Selecting typical instances in instance-based learning, *in* 'Proceedings of the Ninth International Machine Learning Conference', pp. 470–479.

Zijdenbos, A. P., Dawant, B. M., Margolin, R. A. & Palmer, A. C. (1994), 'Morphometric analysis of white matter lesions in MR images: method and validation', *IEEE Transactions on Medical Imaging* **13**(4), 716–724.