Large Scale Variational Bayesian Inference with Applications to Image Deblurring

A thesis submitted in partial satisfaction of the requirements for the degree Master of Science in Statistics

by

Brian Jonathan Verbaken

2011
The thesis of Brian Jonathan Verbaken is approved.

__________________________________________

Song-Chun Zhu

__________________________________________

Frederic R. Paik Schoenberg

__________________________________________

Yingnian Wu

__________________________________________

Alan L. Yuille, Committee Chair

University of California, Los Angeles
2011
To my parents,

For their love, time, and continuing support.
# Table of Contents

1 Introduction ......................................................... 1  
  1.1 Notation .......................................................... 2  

2 Previous Work ......................................................... 4  
  2.1 Variational Bayesian ............................................. 4  
  2.2 Expectation Propagation ....................................... 5  
  2.3 Double Loop Algorithm ......................................... 6  

3 Variance Estimation .................................................. 8  
  3.1 Woodbury .......................................................... 8  
  3.2 Lanczos Method .................................................. 9  
  3.3 Sampling method ................................................ 9  

4 Data Collection ....................................................... 11  
  4.1 Setup ............................................................. 11  
  4.2 Initial Observations ............................................ 12  

5 Analysis ............................................................ 15  
  5.1 Analysis Method ................................................ 15  
  5.2 Results ........................................................... 16  
  5.3 Follow-up Analysis .............................................. 18  
  5.4 MAP vs. Bayesian ................................................ 20  
  5.5 Interpretation of Results ....................................... 21  

iv
LIST OF FIGURES

5.1 PSNR values for VB/Lanczos vs. VB/sample . . . . . . . . . . . 17
5.2 PSNR values for VB/sample vs. EP/sample . . . . . . . . . . . . 17
5.3 PSNR values for VB/Lanczos vs. EP/sample . . . . . . . . . . . . 17
5.4 Boxplot of Computation Times . . . . . . . . . . . . . . . . . . 19
LIST OF TABLES

4.1 Inferred Images for 3 methods ....................................... 14

5.1 Friedman Test Results for PSNR ..................................... 18
5.2 Pairwise Comparisons for PSNR ..................................... 18
5.3 Friedman Test Results for Computation Time .................... 19
5.4 Paired t-test for Computation Time ................................. 19
5.5 Friedman Test Results for MAP vs. Bayesian ..................... 21
5.6 Pairwise Comparisons Against MAP ............................... 21
ACKNOWLEDGMENTS

I would like to thank all of my teachers and professors from elementary school up through graduate school. It is because of your encouragement and patience that I have gotten to where I am today. A special thank you to all my professors at Cal Poly for nurturing and growing my interest in statistics and its applications.

Specifically, I would like to thank my advisor, Professor Alan Yuille, and George Papandreou for answering my seemingly endless questions and helping me to learn more about computer vision and Bayesian inference.
Abstract of the Thesis

Large Scale Variational Bayesian Inference with Applications to Image Deblurring

by

Brian Jonathan Verbaken

Master of Science in Statistics

University of California, Los Angeles, 2011

Professor Alan L. Yuille, Chair

Huge data sets have now become the norm and put a larger strain on computation time and memory. Within the field of computer vision and compressed sensing this burden is felt because the inference is done on the edges between the pixels of an image. This does not scale linearly and thus the matrices being used grow faster than the computers that can analyze them. This becomes problematic when looking at variances because when exactly calculating variances it is required to invert the data matrix. This is computationally impossible for large images, and thus variances approximations must be used instead. The Lanczos method creates a low-rank matrix approximation by an iterative method similar to a singular value decomposition (SVD). The drawback of the Lanczos method is that it scales badly in terms of computation time and memory for the number of iterations because of the required reorthogonalization step in each iteration. Rather than using the deterministic approach of the Lanczos method, a fairly new method that takes advantage of sampling from a Markov random field (MRF) inserts a stochastic element into an otherwise deterministic algorithm. When sampling from a Gaussian Markov random field (GMRF) the samples are...
selected from a perturbed GMRF. The sampling method is not only unbiased, it can estimate very efficiently with relatively few samples, and it scales much better than the Lanczos method to the dimensionality of the data. This thesis will compare these two methods in terms of peak signal-to-noise ratio (PSNR) when deblurring images with a known smoothing kernel. The data used is from a previous paper [6] and consists of 4 images with 8 smoothing kernels per image. It will also be examined if these Bayesian techniques are more effective than maximum a posteriori (MAP) estimation.
CHAPTER 1

Introduction

Image deconvolution, also know as image deblurring, algorithms were initially used on small images where the algorithm could use exact estimation methods. However, when the size of images kept growing the computational power of computers could not keep up and the estimation methods used had to start using approximations rather than finding the exact values. Statistics research has shown that photographs typically have wavelets coefficients that follow a super-Gaussian distribution, where the majority of coefficients are close to 0 and the remaining number are significantly large. When using Bayesian techniques this leads to using similarly super-Gaussian prior distributions, such as Laplace or Students t distributions, which will help in better estimating the relationship between the edges of the image.

A majority of the research related to Bayesian estimation techniques has been done using MAP estimates which estimate the mode or most likely point of the posterior distribution. It it generally agreed upon that this is the best way of estimating the means and now researchers are looking into how to best estimate the variances of the posterior distributions. Generally, variances can be computed exactly by inverting the data matrix, but this becomes computationally inefficient or impossible as the number of variables (or edges of the image) increase. Thus variance estimation, rather than exact computation, has become popular to use.
The Lanczos method is an iterative algorithm that creates a low rank matrix approximation for sparse matrices. It is similar to a singular value decomposition that iteratively finds the largest eigenvalue of a matrix, reorthogonalizes the estimation matrix, and then finds the next highest eigenvalue [4]. This reduces the number of dimensions of the matrix to a more computationally tractable low-rank approximation. Based on the desired output and if certain criterion are met some computation time can be eliminated, making this method less computationally intensive as the dimension of the data increases. A relatively new sampling method has been proposed which utilizes sampling from a Markov random field and estimates the variances from the samples rather than through matrix manipulation. This method scales much better in terms of computation time and memory than the Lanczos method and also can guarantee a desired relative error by selecting a certain number of samples. These benefits make it a very strong competitor to the Lanczos method. This thesis will attempt to determine if the sampling or Lanczos method is better than the other as well as comparing these two methods to MAP estimation to see if Bayesian inference is beneficial.

1.1 Notation

Let the unknown parameters $u \in \mathbb{R}^n$ be the original image that has been blurred and $y \in \mathbb{R}^m$ be the observed blurred image from a Gaussian distribution. Then we can represent their relationship based on a blurring kernel $X$:

$$y = Xu + \varepsilon \in \mathbb{R}^m \quad \text{where} \quad \varepsilon \sim N(0, \sigma^2 I)$$

We adopt a sparse prior $T_i(s_j)$ along the directions:

$$s = Bu \in \mathbb{R}^q$$
Which results in a posterior distribution:

\[
P(u|D) \propto \mathcal{N}(y|Xu, \sigma^2I) \prod_{j=1}^{q} T_i(s_j)
\]
CHAPTER 2

Previous Work

The research and code that is used in this study is based upon the glm-ie toolbox developed by Hannes Nickisch [8] written in Matlab with additional research by others [17]. For simplicity all of the coding of this study was done in Matlab to remain consistent. The documentation to the glm-ie toolbox describes the main framework and algorithms that were used in this study. The sparse priors that are used in our Bayesian estimation are a subgroup of the inference methods that this toolbox is able to do.

2.1 Variational Bayesian

Variational Bayesian (VB) is an extension of expectation maximization (EM) that is used in MAP estimation because in EM one single point is found while in VB the posterior distribution is computed (or approximated), including all the parameters and latent variables. In our case it maximizes the following functions:

\[ T(s) = \max_{\theta \geq 0} \exp \left( \beta s - \frac{s^2}{2\theta} - \frac{h(\theta)}{2} \right) \]

\[ h(\theta) = \max_{x \geq 0} \frac{x}{y} - 2g(\sqrt{x}) \]

\[ g(s) = \ln T(s) - \beta s \]

The intermediate steps in [8] manipulate the following properties:

1. Positivity: \( T(s) > 0 \)
2. Symmetry: \( \forall s \in \mathbb{R}, \exists \beta \in \mathbb{R} \) such that \( T(s)e^{-\beta s} = T(s)e^{\beta s} \) and for \( T(s) = T(-s), \beta = 0. \)

3. Super-Gaussianity: \( g_{\beta}(x) \) is convex and decreasing where \( g_{\beta}(x) = \ln T(\sqrt{x}) - \beta \sqrt{x} \).

This results in the posterior \( P(u|D) \) being approximated by a Gaussian distribution \( Q(u) = \mathcal{N}(u|m, V) \) where:

\[
\begin{align*}
\text{Covariance} & : \quad V = A^{-1} \\
\text{Mean} & : \quad A^{-1}d \\
d & = \frac{1}{\sigma^2}X^Ty + B^T\beta \\
A & = X^T(\sigma^2I)^{-1} + B^T\Theta^{-1}B \\
\Gamma & = \text{diag}(\theta)
\end{align*}
\]

Nickisch then goes on to show that the concavity of \( \theta^{-1} \mapsto \ln |A| \) which allows us to find an approximate variance estimate. He also notes that for log-concave potentials the univariate minimization of \( \theta_i \) has a closed form.

2.2 Expectation Propagation

Expectation Propagation (EP) is an inference technique that is very similar to VB, but is more general and does not require the potentials to be super-Gaussian. \( \beta \) is treated as a parameter in this case, it is not a set quantity as it was in VB.
The quantity that is minimized for EP is:

\[ \phi_{EP}(\gamma, \beta) = \ln |A| + h_{EP}(\theta, \beta) + \min_u R(u, \theta, \beta) \]

\[ R = \frac{1}{\sigma^2} \|Xu - y\|^2 + s^T \Theta^{-1} s - 2\beta^T s \]

Where:

\[ A = \frac{X^T X}{\sigma^2} + B^T \Theta^{-1} B \]

The EP criterion is very similar to the VB criterion given earlier, the only difference is the additional \( h_{EP}(\theta, \beta) \) term. This leads to a very similar final equation for the mean and covariance of the marginal posterior distribution. Let \( m = \mathbb{E}_Q(u|D) \) and \( V = \mathbb{V}_Q(s|D) \) then \( Q_j(s_j) \) is \( \mathcal{N}(s_j|\mu_j, \rho_j) \) where:

\[ \mu = Bm \]
\[ \rho = \text{diag}(BVB^T) \]

### 2.3 Double Loop Algorithm

The glm-ie toolbox hinges on a double loop algorithm [17]. The outer loop is the variance estimation while the inner loop is the MAP estimation, the computation of the desired variables in a loop are done when the variables of the other loop are held fixed. The inner and outer loop alternate until a solution is found. The minimizations that occur are as follows for VB:

1. Outer loop: \( z^* = \arg \min_z \phi_{VB}(uz) = \text{dg}(BA^{-1}B^T) = \mathbb{V}[s|D] \)

2. Inner loop: \( u^* = \arg \min_u \phi_{VB}(u, z) = \frac{1}{\sigma^2} \|Xu - y\|^2 + 2 \cdot h^*(Bu) = \mathbb{E}[u|D] \)

For EP the double loop algorithm goes as follows:

1. Outer loop: Compute \( \rho = z = \text{diag}(BA^{-1}B^T) \)
2. Update $(\beta, \theta)$ and then recompute $\mu = Bm$

The inner and outer loop for EP and VB both do similar things, the formulas are only tweaked a little. Although EP is more versatile it is dependent on the exact variance computations that are done. It is not very robust against any small flaws that may occur in the variance computation which VB can handle. Thus there is a tradeoff that must be made between the versatility of EP and the robustness of VB.
CHAPTER 3

Variance Estimation

Many variance methods for GMRF are applicable in certain situations, but algorithms such as [13] are not feasible for sizes of matrices that are considered in this study. This includes the Woodbury method, which is quickly discussed below, which cannot be used since it requires matrices much smaller than this study is considering. However, there are many algorithms that are much more computationally efficient than inverting the entire data matrix. Based on the structure of the model and whether there are any hidden variables or additional layers certain algorithms can be utilized that are much more powerful and better in terms of speed and computer capacity.

3.1 Woodbury

For the special case where $B = I$, $z = \text{diag}(A^{-1})$, and the size of the rows and columns, $m$ and $n$ respectively, of $X$ are small, the variances can be found exactly by applying the Woodbury formula [8]:

$$A^{-1} = \left( \frac{X^T X}{\sigma^2} + \Gamma^{-1} \right)^{-1} = \Gamma - \Gamma X^T (\sigma^2 I + X \Gamma X^T)^{-1} X \Gamma$$

When applicable, this option is desirable because it requires no approximation. However, it will not be applicable with the data that will be used in this study.
3.2 Lanczos Method

The application of the Lanczos method for this study is to use an iterative algorithm similar to a singular value decomposition that is run on the sparse matrices that we are considering [4]. When estimating variances, scalability of the matrix becomes an issue because the matrix will need to be inverted, this is why a low-rank approximation is employed in the Lanczos method to reduce the computation time and memory of this matrix inversion [14, 15]. Preconditioning can also be implemented, given certain criterion are met, so that less iterations need to be run thus saving computational time. However, this is very rarely the case and is almost never used in application.

The Lanczos method is utilized in [12] where its relationships with conjugate gradients to find eigenvalues and eigenvectors are in turn used to estimate variances. Malioutov et al. [7] examine how to estimate posterior variances in large-scale GMRF. They create an aliasing matrix that is using in matrix inversion to estimate the variances. This works well when there is only short range correlation, it becomes complicated when long range correlations are present and a wavelet-based approach is used to deal with this problem.

3.3 Sampling method

Rather than using exact or approximate matrix inversion, [10] discusses a new method of estimating variances for very large matrices by first adding local perturbations into the data matrix and then analyzing. This method is shown to
be unbiased for both the mean and covariance. The essence of the algorithm is to perturb the prior means, perturb the measurements, and then compute the posterior mode using MAP estimation. This leads to a very intuitive unbiased variance estimator:

\[
\hat{\Sigma} = \frac{1}{S} \sum_{s=1}^{S} (x_s - \mu)(x_s - \mu)^T
\]

Where \( S \) is the number of samples taken, \( x_s \) is the \( s \)-th GMRF sample, and \( \mu \) is the posterior mean/mode. We can assume (and it can be shown) that the variance estimates calculated in this way are never larger than the true variances because increased samples lead to a decrease in the uncertainty of the Gaussian models. To account for this fact, a clipped estimate that is asymptotically unbiased is used:

\[
\bar{z}_k = \min(\hat{z}_k, \gamma_{k}^{-1})
\]

If the variances are the only values of interest, the diagonal of \( \hat{\Sigma} \) is all that needs to be retained. The error is the variance estimates has a relative error of \( r = \sqrt{2/S} \) which is sufficient for approximate variance estimation. This variance estimation method is highly flexible and can be applied to any class of models among the GMRF family. Another benefit is that this method’s accuracy is independent of the matrix size.
4.1 Setup

The variable of interest for each of the images that are being studied is PSNR which is a ratio between the signal of the image and the additional noise that affects the quality of the image. This metric is most commonly used in engineering where it is used to assess the quality of a compressed image, although structural similarity index measure (SSIM) is also often used [5]. PSNR is a very intuitive measure of the quality of an image and removes the subjectivity of a human trying to determine which image has better quality. Where $I$ and $K$ are the two images that are being compared and one is the noisy representation of the other PSNR can be computed as:

\[
\text{PSNR} = 10 \cdot \log_{10} \left( \frac{\text{max}_I^2}{\text{MSE}} \right) \\
= 20 \cdot \log_{10} \left( \frac{\text{max}_I}{\sqrt{\text{MSE}}} \right) \\
\text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2
\]

Where $\text{max}_I$ is the maximum pixel value, which is the case of our images is 255. Generally the values of PSNR are between 30 and 50 dB. Although PSNR is relative, we can use this range to see where our PSNR measurements fall in general.
The images used in our analysis are from the Levin et al. data [6] which consists of 4 different images with 8 different blurring kernels for a total of 32 images. Each of the four methods: VB/Lanczos, VB/sample, EP/Lanczos, and EP/sample, was run on the 32 images for a total of 128 PSNR data points.

4.2 Initial Observations

The PSNR readings for the EP/Lanczos method were negative which is not expected. EP is highly sensitive to variance estimation errors and Lanczos is a very rough approximation, these two factors together cause the EP/Lanczos method to break down and be irrational. Thus this method has been removed from any further discussion and leaves only 96 usable and rational data points.

Before analyzing the PSNR measurements for each measurement, let’s take a look at the inferred image for each of the 3 working methods for an identical image and deblurring kernel. The first column of images in table 4.1 show the final inferred image for each of the three methods for the same image (image 1, kernel 1 of [6]), in general the images look very similar with only very minute details that differ. These diminutive details are very tough to see unless either zooming in or enlarging the image.

One example of the difference in the images can be seen on the right side of the sandbox along the edge of the image, where the EP/sample image is blurrier than either the VB/Lanczos or VB/sample images. When examining the girl’s hair and sweater, the methods that utilize sampling are crisper and more clear than
the method that uses Lanczos. The images that use sampling are also much more consistent and less choppy when inferring the background surrounding the two adults in the top-left corner of the image.

When looking at the second set of images (image 2, kernel 6 of [6]) there is no noticeable difference between the two methods that employ sampling, but when comparing these two images to the image using Lanczos there a few discrepancies. The images using sampling are more precise when inferring things like the texture of the bushes and grass in the foreground, as well as the trees and parts of the house in the background. It is interesting to note that each of the three methods have trouble inferring the left and bottom borders of the images, although the images using sampling are more fluid and less ambiguous.

The observations that were made may point towards sampling being a better at deblurring images than Lanczos, but observations about very small differences between a couple of images may not result in statistically significant results which will be looked into in the next chapter.
<table>
<thead>
<tr>
<th>Method</th>
<th>Image 1 Kernel 1</th>
<th>Image 2 Kernel 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB/sample</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td>VB/Lanczos</td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td>EP/sample</td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Table 4.1: Inferred Images for 3 methods - Visual discrepancies between methods for the same image are hard to notice with the naked eye, but can be distinguished when zooming in. For an in-depth analysis of some discrepancies in these images reference section 4.2.
CHAPTER 5

Analysis

5.1 Analysis Method

Because each of the 3 methods were run on the same image there was an issue that the independence between samples was violated. Thus the repeated measures analysis of variance seemed like a possible test, however this was also incompatible because it is designed for subjects that are re-evaluated at different time periods to account for changes in other variables. This is not the case for our setup because the images did not “change” between the different treatments. Thus the best option was to use the non-parametric alternative of the one-way repeated measures analysis of variance, the Friedman test [2]. The results of this test are displayed in table 5.1 which has a p-value of approximately 0. This concludes that there is a statistically significant difference between the 3 methods when evaluating how well an image is deblurred using PSNR as a metric.

Now that we know there is a statistically significant difference between the 3 methods we can do pairwise comparisons to find where the differences are. Because nonparametric methods were used earlier, they will be used again in the pairwise comparisons. The binomial distribution can be utilized to determine whether the number of times that one group is larger than the other is significantly higher than expected. For example, if two methods do not have signifi-
cantly different PSNR values then we would expect approximately 16 of the 32 PSNR values of one group to be larger than the other group (for the same image and blurring kernel). Thus a binomial distribution with n=32 and p=0.5 can be utilized and find the probability of having as extreme or more extreme of a number of larger data points for one group than another. This process was used to compare each of the 3 methods for a total of 3 comparisons.

5.2 Results

In table 5.2 and figures 5.1 to 5.3 we can see the results of these pairwise comparisons. There that there is not a statistically significant difference between the EP/sample and VB/sample methods with a p-value of 0.1102; this is also illustrated by the fact that in figure 5.2 the points very closely follow the line and are not consistently above or below the line. This result is different than that from figure 5.1 where we can see that the VB/sample method has 30 of 32 images that have higher PSNR measure than the VB/Lanczos method. The p-value for this comparison is approximately 0 so we can conclude that the VB/sample method has a statistically significantly higher PSNR measurement for an image than the VB/Lanczos method. A similar conclusion can be made when comparing the EP/sample method to the VB/Lanczos method where we conclude that the EP/sample method has a significantly higher PSNR measurement than the VB/Lanczos method.
Figure 5.1: PSNR values for VB/Lanczos vs. VB/sample - VB/sample has a higher PSNR measurement for 30 of 32 images, this is statistically significant.

Figure 5.2: PSNR values for VB/sample vs. EP/sample - EP/sample has a higher PSNR measurement for 21 of 32 images, this is not statistically significant.

Figure 5.3: PSNR values for VB/Lanczos vs. EP/sample - EP/sample has a higher PSNR measurement for 28 of 32 images, this is statistically significant.
Table 5.1: Friedman Test Results for PSNR - There is a statistically significant difference in the PSNR measurements of the 3 methods.

<table>
<thead>
<tr>
<th>Source</th>
<th>SS</th>
<th>df</th>
<th>MS</th>
<th>Chi-sq</th>
<th>Prob &gt; Chi-Sq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>32.6875</td>
<td>2</td>
<td>16.3438</td>
<td>32.6875</td>
<td>7.98e-8</td>
</tr>
<tr>
<td>Error</td>
<td>31.3125</td>
<td>62</td>
<td>0.5050</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>64</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: Pairwise Comparisons for PSNR - Results for pairwise comparisons for figures 5.1 to 5.3.

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Group 2</th>
<th>Group 1 ≥ Group 2</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB/sample</td>
<td>VB/Lanczos</td>
<td>30</td>
<td>2.46e-7</td>
</tr>
<tr>
<td>EP/sample</td>
<td>VB/sample</td>
<td>21</td>
<td>0.1102</td>
</tr>
<tr>
<td>EP/sample</td>
<td>VB/Lanczos</td>
<td>28</td>
<td>1.93e-5</td>
</tr>
</tbody>
</table>

5.3 Follow-up Analysis

Now that it has been determined that the EP/sample and VB/sample methods are equally the “best” method when estimating variances using PSNR, is there some way that we can distinguish between the two? Yes, if the two methods result in PSNR measurements that are not statistically significantly different but one method takes less computational time then isn’t that method more efficient? To be consistent the Friedman test can be used again, but a paired t-test can be used if normality of the samples can be assumed. Let’s run both statistical tests and hope that the conclusions match, making the assumption of normality moot.

Tables 5.3 and 5.4 show the results for the Friedman test and paired t-test respectively. In both cases the p-value is essentially zero which means that there is
Figure 5.4: Boxplot of Computation Times - VB/sample has a significantly longer computation time than EP/sample.

<table>
<thead>
<tr>
<th>Source</th>
<th>SS</th>
<th>df</th>
<th>MS</th>
<th>Chi-sq</th>
<th>Prob&gt;Chi-Sq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>16</td>
<td>1</td>
<td>16</td>
<td>32</td>
<td>1.54e-8</td>
</tr>
<tr>
<td>Error</td>
<td>0</td>
<td>31</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>16</td>
<td>63</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.3: Friedman Test Results for Computation Time - There is a statistically significant difference in the computation times of EP/sample and VB/sample.

<table>
<thead>
<tr>
<th>Test Statistic</th>
<th>17.7234</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>31</td>
</tr>
<tr>
<td>Confidence Interval</td>
<td>(143.05,180.23)</td>
</tr>
<tr>
<td>P-value</td>
<td>8.90e-19</td>
</tr>
</tbody>
</table>

Table 5.4: Paired t-test for Computation Time - There is a statistically significant difference in the computation times of EP/sample and VB/sample.
a statistically significant difference in the computation times of the VB/sample and EP/sample methods. Solely looking at figure 5.4 we could have predicted that there would be statistical significance because the boxplots have little or no overlap. The confidence interval tells us that we are 95% confident that the VB/sample method will have between 143.05 and 180.23 seconds slower computation time than the EP/sample method. This is a quite significant difference, especially if you will be running the deblurring algorithm on multiple images.

The fact that the EP/sample method has a much faster computation time than the VB/sample method while having the same PSNR measurements for the same image makes it that much more appealing and essentially breaks the tie between the two methods.

5.4 MAP vs. Bayesian

One question that could be asked is if the Bayesian priors and analysis that is being used is actually better than the simpler MAP estimation. This is of interest because the Bayesian inference requires more computation than MAP estimation and thus should only be done if it is truly beneficial. To analyze this we ran the Friedman test on the same data as in section 5.1 with an additional column for the MAP estimates’ PSNR measurements. We would expect these results to be statistically significant, as we can see in table 5.5, because the Friedman test was already significant when comparing only the 3 methods so adding MAP cannot make the results of the test any less significant. We can conclude that there is a statistically significant difference between the PSNR measurements of the 3 methods and the MAP estimates.
Table 5.5: Friedman Test Results for MAP vs. Bayesian - There is a statistically significant difference in the PSNR measurements of the 3 methods and the MAP estimate.

<table>
<thead>
<tr>
<th>Source</th>
<th>SS</th>
<th>df</th>
<th>MS</th>
<th>Chi-sq</th>
<th>Prob&gt;Chi-Sq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>94.5625</td>
<td>3</td>
<td>31.5208</td>
<td>56.7375</td>
<td>2.92e-12</td>
</tr>
<tr>
<td>Error</td>
<td>65.4375</td>
<td>93</td>
<td>0.7036</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>160</td>
<td>127</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6: Pairwise Comparisons Against MAP - VB/sample and EP/sample are significantly better than MAP, VB/Lanczos is significantly worse than MAP.

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Group 2</th>
<th>Group 1 ≥ Group 2</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB/sample</td>
<td>MAP</td>
<td>29</td>
<td>2.56e-6</td>
</tr>
<tr>
<td>EP/sample</td>
<td>MAP</td>
<td>26</td>
<td>5.35e-4</td>
</tr>
<tr>
<td>MAP</td>
<td>VB/Lanczos</td>
<td>32</td>
<td>4.66e-10</td>
</tr>
</tbody>
</table>

What is really of interest is the pairwise comparisons, which were done similarly to section 5.2 and the results can be seen in table 5.6. Each of these pairwise comparisons is highly significant and therefore we are very confident in concluding that the sampling method is statistically significantly better than MAP, whether using VB or EP. We can also conclude that MAP is statistically significantly better than the VB/Lanczos method.

### 5.5 Interpretation of Results

EP did not work well with the Lanczos approximation, which resulted in negative PSNR measurements, this is troublesome because it would be easiest to find one method that works best under any circumstances. Although universal useability
is not the only criterion for determining the best method to use when doing large scale variance estimation it should be taken into account. Because each method that used sampling was statistically significantly higher than any method using the Lanczos approximation we can safely conclude that sampling is best to use in the outer loop of the double loop algorithm.

When looking at what to use for the inner loop, there was not a statistically significant difference between EP and VB when looking at the PSNR measurements of images. Because of this result our follow-up analysis looking at the computation time of the two methods showed that EP was faster than VB while having no negative affect on the PSNR measurement. This result leads to the final conclusion that EP and sampling is the best combination when estimating variance in large scale image deblurring.

Additionally the validity of the Bayesian techniques was investigated and found to be useful when using the sampling method, but not when using the Lanczos method. Thus the best to worst ordering would be: EP/sample, VB/sample, MAP, VB/Lanczos, EP/Lanczos.
CHAPTER 6

Final Remarks

6.1 Future Research

This study was limited in certain ways and could be expanded in future research. The images that were used to test which variance estimation method was best had the blurring kernel as a known parameter. What happens if you are given a blurred image (without knowing the blurring kernel), are the results found in this study applicable to this situation? What if we have a blurred image that is small enough that the Woodbury formula can be utilized, which method would be best now? Obviously some of these questions are highly specialized on criterion that happen infrequently or almost never, while the sampling method for high dimension variance estimation proposed and discussed in [9, 10, 11] is flexible enough for any situation yet powerful enough to rival (or even beat) highly specialized algorithms.

Another change that could be explored would be another way of quantifying when one method is “better” than another at deblurring at an image, more specifically in this case is how well the method estimates the variances. PSNR is the way this study chose to quantify this phenomenon. The reasons why PSNR was chosen have already been outlined, the quantitative measure chosen is dependent upon the researcher and exactly what he/she wants to measure from the deblurring
methods that will quantify one method as being better than the other. One possible new measure may be some sort of ratio between PSNR and the time the method takes to run, this will give better rating of the efficiency of a method because now there would be a tradeoff between computational time and the precision of the variance estimation.

One thing that should be kept in future research is to run whatever chosen deblurring methods on the same images, rather than independent random samples of images, because this drastically reduces the variation between samples (borrowing terms from analysis of variance) and leads to greater power in any statistical tests that are ultimately run.

### 6.2 Conclusion

The sampling method is has statistically significantly higher PSNR measurements than the Lanczos method, this coupled with it being much more scalable in both computation time and memory leads one to conclude that it is clearly superior to the commonly used Lanczos method. The sampling method is also useable in any situation without restrictions, making it even more desirable and tractable to use in the future when estimating variances in high dimensional data. The sampling method is significantly better than MAP estimation, which the Lanczos method is not, and has equal results when using VB or EP while VB has a slower computation time than EP. Thus if given the choice when doing large scale variational Bayesian inference in image deblurring one should choose a combination of EP and the sampling method.
CHAPTER 7

Matlab Code

Code was written and edited in conjunction with George Papandreou between June and November 2011.

File name: deblur_dataset_brian.m

% Do nonblind deconvolution to recover the image u that has been blurred by an optical device with known point spread function f. The measurements are y = conv2(u,f) + e, e~N(0,s2).

% Fix start of random number generator so results can be verified
% rand('twister',19791017);
% randn('state',19791017);

%%% Inference parameters

% INNER LOOP PARAMS
opts.innerOutput = 0;
opts.innerExact = 0; % Solve inner problem exactly?
opts.innerMVM = 100; % Number of CG steps in inner loop
% type of variational approximation
% opts.innerType = 'VB';
switch opts.innerType
    case 'VB'
        opts.innerVBpls = 'plsLBFGS'; % PLS algorithm for VB inner loop
    case 'EP'
        opts.innerEPeta = 0.9;
    otherwise
        error('deconv_inference: wrong variational method');
end

% OUTER LOOP PARAMS
opts.outerOutput = 1;
opts.outerNiter = 4; % number of outer loop iterations
% method for marginal variance computation in outer loop
% opts.outerMethod = 'sample';
switch opts.outerMethod
    case {'woodbury','full'}
    case 'lanczos'
        opts.outerVarOpts = struct('MVM',170);
    case 'sample'
        opts.outerVarOpts = struct('NSamples',20,'Ncg',20);
    otherwise
        error('deconv_inference: wrong variance estimation method');
end

%%
res_base_dir = 'results';
data_dir = 'levin';
imw_fun = @(x)1/max(x(:))*x;

% ground truth image ut, corr blur filter kt, and (possibly)
% realistic yt
data_id = 2;
switch data_id
  case 1
    % glm-ie data
    gt_data = load('deblur');
    ut = gt_data.u; % ut = imresize(ut,0.2);
    kt = rot90(gt_data.f,2); kt = kt/sum(kt(:));
    % kt = gt_data.f(4:end,:)
    su = size(ut); sk = size(kt); % sizes
    clear gt_data;
  case 2
    % Levin's data
    gt_data = load(fullfile(data_dir,sprintf('im%02d_ker%02d.mat',im_id,ker_id)));
    ut = gt_data.x;
    kt = gt_data.f; kt = kt/sum(kt(:));
    yt = gt_data.y;
    su = size(ut); sk = size(kt); % sizes
    clear gt_data;
end
% blur operator and noisy measurement

obs_id = 'real';
s2 = 1e-4; % variance

switch obs_id
    case 'artificial'
        % artificial blurring
        conv_fun = @(k)matConv2(k, su,'circ','corr');
        imcorr_type = 'circular';
        Kt = conv_fun(kt);
        ute = ut; sue = su; ue2u_f=@(ue)reshape(ue,su);
        sy = su;
        y = Kt*ut; y = y + sqrt(s2)*randn(size(y)); % observation
    case 'real'
        % real blur: assume that the unknown ut image is bigger
        sy = su;
        sue = su+sk-1;
        conv_fun = @(k)matConv2(k, sue,'valid','corr');
        imcorr_type = 'zero';
        Kt = conv_fun(kt);
        skh = (sk-1)/2;
        ute = nan(sue);
        ute(skh(1)+(1:su(1)),skh(2)+(1:su(2))) = ut;
        ue2u_f=@(ue)reshape(ue(skh(1)+(1:su(1)),skh(2)+(1:su(2))),su);
        y = yt(:);
    case 'realc'
        % real blur: assume that the ut image is same size
        sy = su;
sue = su; ue2u_f = @(ue)reshape(ue,su);
conv_fun = @(k)matConv2(k, sue,'circ','corr');
imcorr_type = 'circular';
Kt = conv_fun(kt);
ute = ut;
y = yt(:);
end
% s2 = mean((y-Kt*ut).^2);

% prior
% B = [matWav(sue); matFD2(sue,'valid')]; % wavelet and total
% variation = finite diff matrix
B = matFD2(sue,'valid'); % total variation = finite diff matrix
tau = 15; % prior inverse scale
pot = @potLaplace;
% pot = @(s) potT(s,2);
% pot = @potGauss;

% exp_id = sprintf(
% '%s_Ns%d_Ncg%d_im%02d_ker%02d_ssq%.1e_tau%.1e', ... % opts.innerType,opts.outerVarOpts.NSamples,
% opts.outerVarOpts.Ncg, ... % im_id,ker_id,s2,tau);
exp_id = sprintf('%s_%s_im%02d_ker%02d_ssq%.1e_tau%.1e', ... % opts.innerType,opts.outerMethod,im_id,ker_id,s2,tau);
res_dir = fullfile(res_base_dir,exp_id);
if ~isdir(res_dir), mkdir(res_dir); end
fprintf('EXP_ID: %s\n',exp_id);

imwrite(ut,fullfile(res_dir,'x.png'));
imwrite(yt,fullfile(res_dir,'y.png'));
imwrite(imw_fun(kt),fullfile(res_dir,'kt.png'));

%% Deblurring with ground truth kernel

dli_fun = @(K,s2,opts) dli(K,y,s2,B,pot,tau,opts,[]);

run_deblur = 1;

if run_deblur

[uinf,ga,b,z,nlZ_deblur,extra] = dli_fun(Kt,s2,opts);
uinf = reshape(uinf,sue);
if isequal(opts.outerMethod,'sample')
    sdinf = sqrt(extra.zu);
else
    % compute standard deviation: sdinf = sqrt(s2*diag(Q*inv(T)*Q'));
    % V = inv(A), A = K'*K/s2 + B'*diag(1./ga)*B \approx Q*T*Q'
    Q_T = extra.Q/chol(extra.T);
    sdinf = sqrt( sum(Q_T.*Q_T,2) );
end
sdinf = reshape(sdinf,sue);

imwrite(uinf,fullfile(res_dir,'uinf.png'));
if any(sue>su)
    [M1 M2] = imalign(uinf,ut,imcorr_type,sk);
    uinfc = ue2u_f(circshift(uinf,[M1 M2]));
    sdinfc = ue2u_f(circshift(sdinf,[M1 M2]));
    imwrite(uinfc,fullfile(res_dir,'uinfc.png'));
    imwrite(imw_fun(sdinfc),fullfile(res_dir,'sdinfc.png'));
else
    uinfc = uinf;
    sdinfc = sdinf;
end

end

%% Estimation with ground truth kernel

run_estimation = 1;

if run_estimation

    opt.nMVM = 250; opt.output = 0;
    u0 = zeros(prod(sue),1);
    uest = reshape( feval(opts.innerVBpls,u0,Kt,y,B,opt,s2,'penVB',pot,
                     tau), sue);
    imwrite(uest,fullfile(res_dir,'uest.png'));

if any(sue>su)
    [M1 M2] = imalign(uest,ut,imcorr_type,sk);
    uestc = ue2u_f(circshift(uest,[M1 M2]));
    imwrite(uestc,fullfile(res_dir,'uestc.png'));
else
    uestc = uest;
end

end

%% Evaluation

fprintf(1,'Blurred: PSNR(y,u_gt) = %.2f\n',psnr(yt,ut,max(ut(:))));
fprintf(1,'Estimation: PSNR(uest,u_gt) = %.2f\n',
    psnr(uestc,ut,max(ut(:))));
fprintf(1,'Inference : PSNR(uinf,u_gt) = %.2f\n',
    psnr(uinf,ut,max(ut(:))));

%% Plotting

run_plot = 1;

if run_plot
    figure(1)
    subplot(2,2,1); imshow(ut); title('true','FontSize',16);
    subplot(2,2,2); imshow(reshape(y,sy)); title('blurred',"FontSize",16);
    subplot(2,2,3); imshow(uinf,sy); title('predicted');
subplot(2,2,3); imshow(uestc); title('mode','FontSize',16);
subplot(2,2,4); imshow(uinfc); title('mean','FontSize',16);

figure(2)
sz = [800,300]; set(gcf,'Position',[50,50,sz])
subplot('Position',[0/3,0,.32,.87])
imagesc(abs(uestc-ut),[0,.2]), axis off, title('|mode-true|','FontSize',16)
subplot('Position',[1/3,0,.32,.87])
imagesc(abs(uinfc-ut),[0,.2]), axis off, title('|mean-true|','FontSize',16)
subplot('Position',[2/3,0,.32,.87])
imagesc(sdinf,[0,3e-3]),axis off, title('standard dev.','FontSize',16)
imagesc(sdinf,[0,0.1]), axis off, title('stdev','FontSize',16)
colormap(gray)

figure(3)
subplot(2,2,1); hist(sdinf(:),100); title('U stdev','FontSize',16);
subplot(2,2,2); hist(sqrt(z(:)),100); title('S stdev','FontSize',16);
subplot(2,2,3); hist(abs(uinfc(:)-ut(:)),100); title('U |mean-true|','FontSize',16);
subplot(2,2,4);
uerrnorm = (uinfc(:)-ut(:))./sdinfc(:);
mu_errnorm = mean(uerrnorm);
sigma_errnorm = std(uerrnorm);
% mu_errnorm = median(uerrnorm);
% sigma_errnorm = 1.4826*mad(uerrnorm,1);
pdfplot(uerrnorm,100);
hold on
plotgauss1d(mu_errnorm,sigma_errnorm^2,'r');
hold off
title(sprintf('U (mean-true)./stdev
[sigma=%.2f]',sigma_errnorm),'FontSize',16);
end

%%% Save workspace

%for i in *.png; do convert $i 'basename $i png'eps; end;

%save(fullfile(res_dir,[exp_id '.mat']));
%copyfile('deblur_dataset_brian.m',res_dir);
psnr_all = [];
times = [];
for im_id = 1:4;
    for ker_id = 1:8;
        temp = [];
        temp2 = [];
        opts.innerType='VB';
        opts.outerMethod='sample'; start = tic;
dehblur_dataset_brian;
        temp = [temp psnr(uestc,ut,max(ut(:)))
psnr(uinfc,ut,max(ut(:)))];
        temp2 = [temp2 toc(start)];
        opts.outerMethod='lanczos'; start = tic;
dehblur_dataset_brian;
        temp = [temp psnr(uestc,ut,max(ut(:)))
psnr(uinfc,ut,max(ut(:)))];
        temp2 = [temp2 toc(start)];
        opts.innerType='EP';
        opts.outerMethod='sample'; start = tic;
dehblur_dataset_brian;
        temp = [temp psnr(uestc,ut,max(ut(:)))
psnr(uinfc,ut,max(ut(:)))];
        temp2 = [temp2 toc(start)];
        opts.outerMethod='lanczos'; start = tic;
dehblur_dataset_brian;
temp = [temp psnr(uestc,ut,max(ut(:)))
        psnr(uinfc,ut,max(ut(:)))];
temp2 = [temp2 toc(start)];
psnr_all = [psnr_all; temp]
times = [times; temp2]
end
end

% DISREGARD EP/lanczos since it doesn’t work

VBsample=psnr_all(:,2); VBlanczos=psnr_all(:,4);
EPsample=psnr_all(:,6); MAP=psnr_all(:,1);

%%%%%%% Friedman test
[p, table] = friedman(psnr_all(:,[2 4 6]),1);

% VB/sample vs. VB/lanczos
scatter(VBlanczos,VBsample); refline(1,0);
ylabel('VB/sample'); xlabel('VB/lanczos');
2*(1-binocdf(sum(VBsample>VBlanczos)-1,size(VBsample,1),0.5))

% VB/sample vs. EP/sample
scatter(VBsample,EPsample); refline(1,0);
xlabel('VB/sample'); ylabel('EP/sample');
2*(1-binocdf(sum(VBsample<EPsample)-1,size(VBsample,1),0.5))

% EP/sample vs. VB/lanczos
scatter(VBlanczos,EPsample); refline(1,0);
ylabel('EP/sample'); xlabel('VB/lanczos');
2*(1-binocdf(sum(EPsample>VBlanczos)-1,size(VBsample,1),0.5))

% Compare computational times between VB/sample and EP/sample
[h, p, ci, stats] = ttest(times(:,1),times(:,2));
[p1, table1] = friedman(times(:,1:2),1);
boxplot(times(:,1:2), vertcat(repmat(['VBsample'],32,1),
        repmat(['EPsample'],32,1)));
ylabel('Computation time'); xlabel('Method');

%%%%%% MAP vs. Bayesian
[p, table] = friedman(psnr_all(:,[1 2 4 6]),1);

2*(1-binocdf(sum(VBsample>MAP)-1,size(VBsample,1),0.5))
2*(1-binocdf(sum(EPsample>MAP)-1,size(VBsample,1),0.5))
2*(1-binocdf(sum(MAP>VBlanczos)-1,size(VBsample,1),0.5))
2*(1-binocdf(sum(MAP>VBsample)-1,size(VBsample,1),0.5))
REFERENCES


