PROJECTED BARZILAI-BORWEIN METHOD WITH INFEASIBLE ITERATES FOR NONNEGATIVE IMAGE DECONVOLUTION

by

Kathleen Fraser

Submitted in partial fulfillment of the
requirements for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
July 2011

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "PROJECTED BARZILAI-BORWEIN METHOD WITH INFEASIBLE ITERATES FOR NONNEGATIVE IMAGE DECONVOLUTION" by Kathleen Fraser in partial fulfillment of the requirements for the degree of Master of Computer Science.

Dated: July 22, 2011

Supervisors: _____

_____

Reader: _____

DALHOUSIE UNIVERSITY

DATE: July 22, 2011

AUTHOR: Kathleen Fraser

TITLE: PROJECTED BARZILAI-BORWEIN METHOD WITH INFEASIBLE ITERATES FOR NONNEGATIVE IMAGE DECONVOLUTION

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: M.C.Sc. CONVOCATION: October YEAR: 2011

Signature of Author

Table of Contents

List of Tables . . . . . **vi**

List of Figures . . . . . **vii**

Abstract . . . . . **x**

List of Abbreviations and Symbols Used . . . . . **xi**

Acknowledgements . . . . . **xiii**

Chapter 1 Introduction **1**

Chapter 2 Background **4**

2.1 The Image Degradation Model 4
2.1.1 Image Blur and Noise 4
2.1.2 Formalism 6
2.1.3 The Inverse Problem 7

2.2 Digital Image Restoration 7

2.3 Evaluation 10

2.4 Deconvolution Microscopy 10
2.4.1 Fluorescence Microscopy 10
2.4.2 Principles of Fluorescence 11
2.4.3 Degradations 12

Chapter 3 Methods **14**

3.1 Direct Methods 14
3.1.1 Inverse Filtering 14
3.1.2 Wiener Filtering 15
3.1.3 Tikhonov Filtering 15

3.2 Iterative Methods 17
3.2.1 Steepest Descent 18
3.2.2 Conjugate Gradient 19
3.2.3 Barzilai-Borwein 21

3.3 Constrained Methods 22
3.3.1 Gradient Projection 24

3.3.2 Gradient Projection-Conjugate Gradient 24
3.3.3 Projected Barzilai-Borwein 25

3.4 Barzilai-Borwein with Infeasible Iterates 27

3.5 Stopping Criteria 31

Chapter 4 Results **34**

4.1 Two-Dimensional Image Restoration 34
4.1.1 Test Cases 34
4.1.2 Parameters 36
4.1.3 Evaluation and Efficiency 37
4.1.4 Restoration Error 38
4.1.5 Discussion 51

4.2 Three-Dimensional Microscope Image Restoration 53
4.2.1 Test Cases 53
4.2.2 Parameters 55
4.2.3 Evaluation 55
4.2.4 Discussion 56

Chapter 5 Conclusion **62**

Bibliography **65**

Appendix A Matlab Code **69**

List of Tables

Table 2.1 Different structures for the blurring matrix **A**. 9

Table 4.1 Minimum mean error: head test case. 51

Table 4.2 Minimum mean error: satellite test case. 51

Table 4.3 Minimum mean error: text test case. 52

Table 4.4 Microscope parameters for the real 3D test case. 55

List of Figures

Figure 2.1 Graphical representation of the image degradation process. Typically, the image is blurred by the imaging system (e.g., camera), and affected by noise in the recording system (e.g., CCD). . . 6

Figure 3.1 Inverse filtering. Top left, the original image. Top right, the point spread function used to blur the image (called"PSF1" in Chapter 4). Bottom left, the image blurred by the PSF and then degraded by additive Gaussian noise such that the BSNR = 20 dB. Bottom right, the results of inverse filtering. 16

Figure 3.2 Wiener filtering. The image is the same same test case as in Fig. 3.1. On the left, the blurred and noisy image. On the right, the solution after Wiener filtering. 17

Figure 3.3 Slow convergence of steepest descent. This figure shows how the steepest descent method can be very slow to converge when the minimum lies in a long, narrow valley. (From *Numerical Recipes* [30].) 19

Figure 3.4 Iterative regularization. On top, the restoration error shows semiconvergent behaviour. On the bottom left, the solution after 20 iterations is over-regularized. In the middle, the solution after 450 iterations is near the minimum of the restoration error curve. On the bottom right, the solution after 2000 steps is under-regularized. 20

Figure 3.5 The Barzilai-Borwein method. We consider again the same test case as in Fig. 3.1. On top, the restoration error shows non-monotonic behaviour. Below, the step sizes span several orders of magnitude and do not show any discernible pattern [38]. . . 23

Figure 3.6 Comparing the performance of BB and PBB. On top, the restoration error of PBB decreases smoothly but does not reach as low a value as BB in 200 iterations. Below, there is a marked difference in the projected BB step size as compared to the original BB step size (see Fig. 3.5 for comparison). Data is for the same test case as Fig. 3.1. 26

Figure 3.7 Relationship between restoration error and r value. On top, the blue (solid) line shows the restoration error for the BB method. The green (dashed) line shows what the restoration error *would* be, if the solution was projected on that iteration. In the middle, the value of r, given by (3.24), at each iteration. On the bottom, the median value of r over the past 10 iterations. Data is for the same test case as Fig. 3.1. 30

Figure 3.8 Good performance of PBB. Shown is the restoration error for the head test image from Chapter 4, blurred by a motion blur of length 20 at an angle of 45°, with BNSR = 30. BB is the blue (solid) line and PBB is the green (dashed) line. 31

Figure 3.9 Comparing the performance of BB and BBII. On top, the restoration error for the two methods is the same until the first projection in BBII. After the point that the BB error starts to increase again, the BBII error is still decreasing. On the bottom, the step sizes for BBII. Iterations on which a projection occurs are marked with green circles. Data is for the same test case as Fig. 3.1. 32

Figure 4.1 Test images. 35

Figure 4.2 Test PSFs. All PSFs have been padded with zeros to size 256×256. The images have been scaled for display purposes. 36

Figure 4.3 Head test image with PSF1 39

Figure 4.4 Head test image with PSF2 40

Figure 4.5 Head test image with PSF3 41

Figure 4.6 Head test image with PSF4 42

Figure 4.7 Satellite test image with PSF1 43

Figure 4.8 Satellite test image with PSF2 44

Figure 4.9 Satellite test image with PSF3 45

Figure 4.10 Satellite test image with PSF4 46

Figure 4.11 Text test image with PSF1 47

Figure 4.12 Text test image with PSF2 48

Figure 4.13 Text test image with PSF3 49

Figure 4.14 Text test image with PSF4 50

Figure 4.15 Bars test cases. On the top left, the true image. On the top right, the PSF. On the bottom left, the blurred image with SNR = 15. On the bottom right, the blurred image with SNR = 30. (Images have been scaled for visualization.) 54

Figure 4.16 Results for 3D bars test case with SNR = 15. On top, the restoration error vs. the number of FFTs. Bottom left, the result after 2000 FFTs for PBB. Bottom right, the result after 2000 FFTs for BBII. 57

Figure 4.17 Results for 3D bars test case with SNR = 30. On top, the restoration error vs. the number of FFTs. Bottom left, the result after 2000 FFTs for PBB. Bottom right, the result after 2000 FFTs for BBII. 58

Figure 4.18 Results for the 3D cell image, using BB, PBB, and BBII, after 400 FFTs. 59

Figure 4.19 Results for the 3D cell image, compared to those produced by the SlideBook software normally used with the microscope. . . 60

Abstract

The Barzilai-Borwein (BB) method for unconstrained optimization has attracted attention for its "chaotic" behaviour and fast convergence on image deconvolution problems. However, images with large areas of darkness, such as those often found in astronomy or microscopy, have been shown to benefit from approaches which impose a nonnegativity constraint on the pixel values. We present a new adaptation of the BB method which enforces a nonnegativity constraint by projecting the solution onto the feasible set, but allows for infeasible iterates between projections. We show that this approach results in faster convergence than the basic Projected Barzilai-Borwein (PBB) method, while achieving better quality images than the unconstrained BB method. We find that the new method also performs comparably to the Gradient Projection-Conjugate Gradient (GPCG) method, and in most test cases achieves a lower restoration error, despite being a much simpler algorithm.

List of Abbreviations and Symbols Used

A	the blurring operator
B	the Fourier transform of the blurred image
F	the Fourier transform of the true image
G	the Fourier transform of the blurred and noisy image
H	the Fourier transform of the point spread function, also known as the optical transfer function (OTF)
R	a frequency domain filter
W	the Fourier transform of the noise vector
Ω	the feasible set
α_k	the step size on the k-th iteration
$\boldsymbol{\gamma}$	the gradient vector
$\hat{\mathbf{f}}$	discrete representation of the de-blurred image, or solution image
$\hat{f}$	the de-blurred image, or solution image
λ	the regularization parameter
$\mathbf{A}$	the blurring matrix
$\mathbf{B}$	the discrete Fourier transform of the blurred image
$\mathbf{F}$	the discrete Fourier transform of the true image
$\mathbf{G}$	the discrete Fourier transform of the blurred and noisy image
$\mathbf{H}$	the discrete Fourier transform of the point spread function
$\mathbf{W}$	the discrete Fourier transform of the noise vector
$\mathbf{b}$	discrete representation of the blurred image
$\mathbf{d}_k$	the search direction on the k-th iteration
$\mathbf{f}$	discrete representation of the true image
$\mathbf{g}$	discrete representation of the blurred and noisy image

$\mathbf{h}$	discrete representation of the the point spread function (PSF)
$\mathbf{w}$	discrete representation of the noise
$\mathcal{A}(\mathbf{f})$	the active set
$\mathcal{F}(\mathbf{f})$	the search face
$\mathcal{H}$	the Hessian matrix
$\mathcal{I}(\mathbf{f})$	the inactive set
$\mathcal{P}_\Omega$	the projection operator
ρ	decay rate for τ for BBII
τ	threshold for BBII
b	the blurred continuous image
f	the true continuous image
g	the blurred and noisy continuous image
h	the point spread function (PSF)
w	the noise

Acknowledgements

I am deeply grateful to my supervisors Dr. Dirk Arnold and Dr. Graham Dellaire for their guidance, inspiration, and encouragement. In particular I would like to thank Dr. Arnold for hiring me as a research assistant well before I had any practical skills to offer, and for his patient introduction into the world of image processing. I would also like to thank my reader Dr. Norman Scrimger for taking the time to offer his advice and share his expertise.

Many thanks also to all the professors and students I have had the pleasure to meet during my time here at Dalhousie. My mind, as well as my circle of friends, has been expanded since day one. Finally, I would like to thank my family and friends for their love and endless support.

The research for this thesis was financially supported by an NSERC CGS-M scholarship.

Chapter 1

Introduction

The goal of image restoration is to take an image which has been degraded by noise or blur and, through various mathematical techniques, recover the true image. Image restoration algorithms are used widely in many different fields of study. One of the early applications of computer image processing was the restoration of the first images of the moon taken during the Ranger mission in 1964 [15]. The images were corrected for distortion caused by the on-board television camera and blur due to the motion of the spacecraft. More recently, the first images from the Hubble Space Telescope were notoriously blurry due to an error in the main mirror, but were corrected using restoration techniques for three years before the mirror could be fixed [3]. Astronomical imaging is still one of the main applications of image restoration, as the images are unavoidably blurred by the bending of light as it travels through the atmosphere, as well as by the optical components in telescopes and cameras.

Medical imaging is another important area in which image restoration methods are applied. Images obtained through different techniques such as X-rays, magnetic resonance imaging (MRI), computerized tomography (CT), and positron emission tomography (PET), can all be enhanced by post-processing of the images [15]. Obtaining a clear image is often crucial to making an accurate diagnosis.

Another popular imaging technique in medicine and other scientific research is fluorescence microscopy. In fluorescence microscopy, biological specimens can be imaged at the cellular level, either by their natural fluorescence or by treatment with fluorescent chemicals. This has allowed researchers to study sub-cellular components in living cells, something which was previously impossible [23, 41]. Image deblurring is particularly important in this field because light is emitted from all parts of the fluorescent specimen, not just the part which is in the focal plane, and thus the raw images appear blurry.

Other fields in which image restoration techniques are used include geography,

which makes use of aerial and satellite photos; law enforcement, where photographs and security videotapes of crime scenes can be enhanced to provide important details; archaeology, where restoration techniques have been used to restore photographs which are the only records of artifacts which have since been broken or lost, and so on [3, 15].

With some knowledge of the imaging system, image blur and noise can be modelled mathematically. Image restoration algorithms attempt to reverse the processes which led to the degradation, and thus recover the original or "true" image. A large number of algorithms exist, and the choice of an appropriate algorithm may depend on a number of factors, including the type of blur and whether or not a good estimate of the blurring function is known, the type and amount of noise, the computing power and time available, the importance of accuracy in the solution, and the application towards which the end result will be directed [7, 15, 19].

In this thesis, we focus on image restoration algorithms which impose a nonnegativity constraint. Such constrained algorithms are particularly appropriate for astronomical and microscope images, or any images with large dark areas [17, 26, 39]. Nonnegatively constrained algorithms include Richardson-Lucy [33], projected Landweber [7], Modified Residual Norm Steepest Descent [26], Gradient Projection [40], and Gradient Projection-Conjugate Gradient [25, 4]. We propose a new algorithm called Barzilai-Borwein with Infeasible Iterates, based on the existing unconstrained Barzilai-Borwein method [6]. Other projected Barzilai-Borwein methods have been proposed [8, 42], but we find that by projecting less often we achieve superior results. The new approach is also much simpler than many of the existing algorithms for nonnegative image deconvolution.

The thesis is organized as follows. In Chapter 2, we present an overview of important concepts in digital image processing, such as mathematical models of image blurring and noise, the discrete inverse problem, and methods of evaluating the results of restoration. We also include a discussion of fluorescence microscopy and the specific problems in that area. In Chapter 3 we present a number of different restoration algorithms, beginning with the most basic approaches and moving on to more sophisticated techniques which take into account the nonnegativity constraint. We

then propose our new approach, and explain why it avoids some of the slow convergence behaviour observed in other projected Barzilai-Borwein methods. In Chapter 4 we show the results of our algorithm on a number of test cases, and compare it with several pre-existing algorithms. We find that Barzilai-Borwein with Infeasible Iterates performs comparably to the others in all cases, and much better in some cases. We also apply the new algorithm to real, three-dimensional microscope images and show that it achieves acceptable results. Finally, in Chapter 5 we discuss some conclusions based on the results of our study.

Chapter 2

Background

In this chapter, we give a brief overview of how image degradation is modelled mathematically, including the effects of blurring and noise. Some simplifying assumptions are explained. We then discuss some of the challenges associated with solving inverse problems, particularly discrete inverse problems, and methods of evaluating the results. Finally, we discuss fluorescence microscopy in particular, including how it works, its applications, and some limitations.

2.1 The Image Degradation Model

In order to develop an algorithm to remove the effects of blurring and noise, we must first understand how these degradations affect the true image, and develop a mathematical model to describe these processes.

2.1.1 Image Blur and Noise

Image degradation consists of two parts: blurring and noise. Image blurring may result from a number of factors, including motion blur, aberrations in the lenses of cameras, atmospheric turbulence in the case of astronomical or satellite images, or the contribution of light from outside the focal plane in light microscopy [19, 36]. In many cases the blur is unavoidable, and must be removed from the image after the acquisition process.

In most situations image blur is linear [19]. Image blur may be described mathematically by a blurring operator, commonly known as the point spread function or PSF. It is called this because it describes the effect of the blur on a single point of light. Linear system theory states that the response of a linear system can be characterized completely by its response to an impulse, in this case a point source of light [15]. In general the PSF might change depending on the location of the point source, but here we assume that the blur is spatially invariant, as is commonly the case [19].

We also require that the PSF integrates to one (that is, no light is lost or gained in the blurring process).

If we let the true image be denoted by the function $f(\mathbf{x})$, where typically $\mathbf{x}$ will be two-dimensional, the blurry image by $b(\mathbf{x})$, and the PSF by $h(\mathbf{x})$, then the linear blurring process can be modelled as

$$b(\mathbf{x}) = \int h(\mathbf{x} - \mathbf{x}')f(\mathbf{x}')\mathrm{d}\mathbf{x}' \tag{2.1}$$

which can also be expressed as

$$b = h * f \tag{2.2}$$

where $*$ represents the convolution operator. Because image blurring is mathematically equivalent to convolution, the process of de-blurring images is often known as image deconvolution.

The second component of image degradation is noise. Noise typically enters the system while the image is being recorded or transmitted, and can depend on a number of different factors. For instance, if the imaging system uses a charge-coupled device (CCD), then the light levels and sensor temperature are known to affect the amount of noise [15]. In microscope imaging, the primary source of noise is photon shot noise, especially at low light levels [35]. Image noise can be modelled by a number of different probability distributions, such as Gaussian, Poisson, uniform, or exponential, depending on the situation [15].

In many cases the PSF is known, or can be estimated, either through mathematical analysis of the components of the imaging system (mirrors, lenses, and so on), by carrying out experiments in which something approximating a point source is recorded and the resulting image used to approximate the PSF, or by computational methods. In general the noise is not known, although in some cases it may be possible to estimate statistical properties of the noise, such as the variance or mean.

The image acquisition process is represented graphically in Fig. 2.1. If we denote the blurred, noisy image by g, then we have

$$g = b + w = h * f + w. \tag{2.3}$$

When dealing with noise as well as blur, it is useful to quantify the amount of noise relative to the amount of information in the image. One way to do this is by calculating the blurred signal-to-noise ratio, or BSNR. We use the definition given in [3],

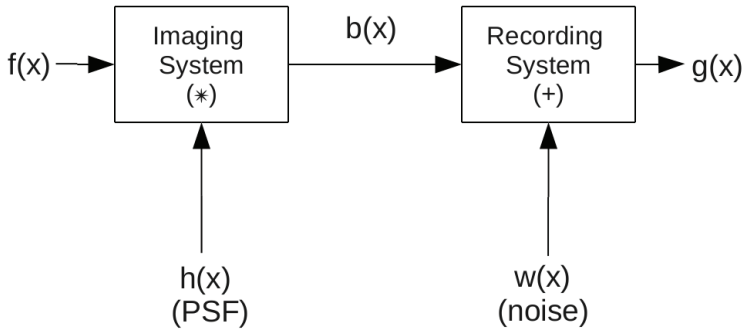


Figure 2.1: Graphical representation of the image degradation process. Typically, the image is blurred by the imaging system (e.g., camera), and affected by noise in the recording system (e.g., CCD).

namely

$$\text{BSNR} = 10 \log_{10} \left(\frac{\sigma_b^2}{\sigma_w^2} \right) \text{dB} \tag{2.4}$$

where σ_b^2 is the variance of the blurred image, and σ_w^2 is the variance of the noise. For a given image, increasing the amount of noise will decrease the BSNR, and decreasing the amount of noise will increase the BSNR.

2.1.2 Formalism

Because we will often want to work in the Fourier domain, it is worth pointing out here that (2.2) and (2.3) are equivalent to

$$B = HF \tag{2.5}$$

and

$$G = HF + W \tag{2.6}$$

by the Convolution Theorem, where H denotes the Fourier transform of h, F denotes the Fourier transform of f, and so on. In particular, H is called the optical transfer function, or OTF, of the imaging system. So performing convolution in the spatial domain is equivalent to performing multiplication in the frequency domain, when H is bounded and F is square-integrable [7].

In the linear model, there exists an operator A which could multiply with f to achieve the same blurring result as convolution. This operator is rarely computed

in practice, for reasons explained in Section 2.2. However it is useful to reformulate (2.1) as

$$b(\mathbf{x}) = (Af)(\mathbf{x}) = \int h(\mathbf{x} - \mathbf{x}')f(\mathbf{x}')\mathrm{d}\mathbf{x}' \tag{2.7}$$

and (2.3) as

$$g = Af + w \tag{2.8}$$

Here we consider A to be an integral operator on the continuous function f. We will discuss the structure of A in more detail in the next section, but note that A is linear, bounded, and continuous [7].

2.1.3 The Inverse Problem

Given an equation of the form $Af = b$, where we can think of A as a system, f as the input, and b as the output, the forward problem is to find b if A and f are known. Usually this is not difficult. The inverse problem is to find f if A and b are known, or to find A if f and b are known. This problem is usually more difficult, and can be made much more difficult if the output b is known only with errors. Inverse problems belong to the class of ill-posed problems [18], which can be difficult to solve.

Image restoration is a classic example of an ill-posed problem. For reasons to be discussed in Section 3.1.1, the solution may not exist if the OTF is zero at any point. The uniqueness of a solution is not guaranteed, particularly in band-limited systems such as telescopes and microscopes. Finally, depending on the nature of the blurring function and the noise, the blurred and noisy image g may not depend continuously on the true image f [7].

2.2 Digital Image Restoration

Up to this point we have assumed all functions and variables to be continuous; however this is not the case with digital images. Rather, continuous data is converted to discrete data by two processes: quantization and sampling.

Quantization refers to the discretization of the amplitude values, typically represented as integers between 0 and $2^k - 1$, where k is the number of bits used for storage [15]. For example, 8-bit images would have values ranging from 0 to 255,

with 0 representing black and 255 representing white. In applications such as microscopy, 12-bit images or higher are typical [24]. The image values do not have to be integers; sometimes instead they are normalized to lie between 0 and 1.

Sampling refers to the discretization of the coordinate values into m rows and n columns [15]. Once this has been done, the image can be represented by a $m \times n$ matrix $\mathbf{F}$, such that $\mathbf{F}(i,j) = v, 1 \leq i \leq m, 1 \leq j \leq n, v \in 0, 1, 2, \ldots, v_{max}$. Each element in the matrix is called a picture element, or pixel.

We can still use the notation of Eqn. (2.8) if we reinterpret the continuous functions as discrete vectors: let $\mathbf{f}$ be a $(mn \times 1)$-dimensional vector made up of the rows of the matrix $\mathbf{F}$, $\mathbf{w}$ be a $(mn \times 1)$-dimensional vector representing the noise, and $\mathbf{A}$ be a $(mn \times mn)$-dimensional blurring matrix. The blurred, noisy image is then represented by a $(mn \times 1)$-dimensional vector $\mathbf{g}$. The problem is now equivalent to a system of linear equations:

$$\mathbf{g} = \mathbf{A}\mathbf{f} + \mathbf{w}. \tag{2.9}$$

The matrix $\mathbf{A}$ depends on two things: the PSF and the boundary conditions [19]. It is clear from looking at a PSF that the blurring of one pixel affects the pixel values of its neighbours. So pixels on or near the boundary of the image are being affected by theoretical pixels which are just outside the boundary. To deal with this problem, we must make assumptions about what the pixel values outside the boundary would be, if they existed. The assumptions we make define the boundary conditions of the problem, and can have a serious effect on the solution.

One possible assumption is *zero boundary conditions*, i.e., that all the pixel values outside the image boundary be zero. This is a simple choice, but is not usually physically accurate except in cases such as astronomical or microscope imaging. Another choice is *periodic boundary conditions*, which assume that the image repeats itself periodically both vertically and horizontally. A third common choice is *reflexive boundary conditions*, which assume that the image pixels are reflected across the boundary, to form a mirror image on the other side. Each of these options is valid, and the choice will depend on the nature of the image at hand.

The exact structure of $\mathbf{A}$ will vary depending on the PSF and the choice of boundary conditions. Table 2.1 is taken from [19] and summarizes the different structures

for the three types of boundary conditions discussed above, and separable and nonseparable PSFs. A separable PSF is one in which the vertical and horizontal components of the blur can be described separately. The acronym BTTB stands for "block Toeplitz with Toeplitz blocks," BCCB stands for "block circulant with circulant blocks," BTHB stands for "block Toeplitz with Hankel blocks," BHTB stands for "block Hankel with Toeplitz blocks," and BHHB stands for "block Hankel with Hankel blocks." Properties of each of these different matrices can be found in [19] or [18].

Boundary Condition	Nonseparable PSF	Separable PSF
Zero	BTTB	Kronecker product of Toeplitz matrices
Periodic	BCCB	Kronecker product of circulant matrices
Reflexive	BTTB + BTHB + BHTB + BHHB	Kronecker product of Toeplitz-plus-Hankel matrices

Table 2.1: Different structures for the blurring matrix $\mathbf{A}$.

Given a PSF it is theoretically possible to use these guidelines to construct the blurring matrix $\mathbf{A}$. In practice, however, it is often not practical to compute $\mathbf{A}$. Even for a relatively small image of size 256×256 pixels, the matrix $\mathbf{A}$ would have 256^4, or $4.2950 \cdot 10^9$, elements. For this reason, it is often more useful to compute multiplications by the OTF in the Fourier domain, than by the blurring matrix in the spatial domain.

The discretization of an ill-posed problem usually leads to an ill-conditioned problem [7]. The condition number of $\mathbf{A}$ is given by $\text{cond}(\mathbf{A}) = s_{\max}/s_{\min}$, where $s_{\max}$ and $s_{\min}$ are the maximum and minimum singular values of $\mathbf{A}$. A high condition number means the problem is ill-conditioned, while a condition number close to one means that the problem is well-conditioned. In general terms, the value of the condition number represents the numerical stability of the solution – in an ill-conditioned problem, some small perturbations in $\mathbf{A}$ may lead to large changes in the solution. In the context of image restoration, this means that a small amount of noise can lead to arbitrarily large perturbations in the solution. This could be image noise from any of the sources mentioned above, or even small perturbations introduced due to

the limits of the numerical accuracy of the deconvolution program. Some methods of regularizing the solution are discussed in Chapter 3.

2.3 Evaluation

In order to compare different methods of image deconvolution we must have some way of evaluating the results. Choosing an appropriate image quality metric can be a difficult task. Because the goal of most deblurring tasks is to produce an acceptable image for human viewers, it has been argued that the best metric would be a subjective measure of quality, such as the mean opinion score [43]. On the other hand, it is more practical in most cases to use an objective measure, such as the mean square error [39], improved signal-to-noise ratio [3], Cramer-Rao bound [35], or some other well-defined mathematical quantity [10].

One common approach is to use the restoration error [5, 11, 26, 27, 42]. The restoration error, also known as the relative error, measures the difference between the restored image $\hat{\mathbf{f}}$ and the true image $\mathbf{f}$:

$$E_{\text{rest}} = \frac{\|\hat{\mathbf{f}} - \mathbf{f}\|_2}{\|\mathbf{f}\|_2} \tag{2.10}$$

where $\|\cdot\|_2$ is the ℓ^2 norm. The limitation of this measure is that it requires knowledge of the true image $\mathbf{f}$, which is possible when using constructed test cases but not in real-life applications.

2.4 Deconvolution Microscopy

One area in which image deblurring is widely used is fluorescence microscopy. Fluorescence microscopes can be used to obtain 3D images of living cells and biological tissues, but the images are distorted by the imaging system and require computational post-processing.

2.4.1 Fluorescence Microscopy

Fluorescence microscopy was invented almost 100 years ago, but was originally limited to imaging samples which were naturally fluorescent. The field grew as scientists developed ways to stain biological specimens with fluorescent dyes. In 1994 a living

organism was imaged using green fluorescent protein, motivating a wave of popularity for the field [41]. Today, fluorescence microscopy is commonly used to capture three-dimensional or even four-dimensional images showing the location and interactions of proteins in living cells [23].

There are two main types of fluorescence microscopes: wide-field and confocal. In wide-field microscopes, the 3D image is produced by recording a series of 2D focal planes. (In 4D microscopy, time acts as the fourth dimension, and the 4D data is a series of 3D data sets.) Any light that is emitted from within each focal plane is in focus; however light from outside the focal plane is also recorded. Thus each resulting slice contains out-of-focus light as well as in-focus light, and appears blurry.

Confocal microscopes mitigate this problem by adding a pinhole in front of the detector, which removes most of the out-of-focus light. The downside is that it also reduces the total amount of light reaching the detector, which can lead to a poor signal-to-noise ratio [35, 41]. This type of microscopy also requires more time, which can make it inappropriate for imaging live, moving specimens [36].

In both cases the images can be improved by deconvolution, although the results are more pronounced in the case of wide-field microscopy [39]. Typically the term "deconvolution microscopy" refers to the application of restoration algorithms to wide-field microscopy [24].

2.4.2 Principles of Fluorescence

Fluorescence is a natural phenomenon which occurs when a molecule emits light of a particular wavelength after light of a different wavelength is incident upon it. The incident photon gives the molecule enough energy to move into a higher energy state. When it returns to its ground state, it emits a photon (usually of a lower energy, i.e., longer wavelength, than the incident photon). Compounds which exhibit fluorescence are called *fluorophores*; dyes which can be used to make other materials fluoresce are called *fluorochromes* [41].

Almost any protein can be labelled using a fluorescent protein, which is part of what makes fluorescence microscopy so popular. In the past 20 years, a wide range of fluorescent proteins have been discovered or engineered, with different properties applicable to a variety of experimental situations. An excellent review of how fluorescent

proteins can be used to label living cells can be found in [23].

2.4.3 Degradations

As mentioned above, the main source of blur in fluorescence microscope images is due to light contributions from structures outside the plane of focus. However, blur can also be caused by other aspects of the imaging system. Because the lens aperture is of finite size, diffraction ring patterns are observed [24, 36]. In addition, the lenses may have slight irregularities, and spherical aberrations can be observed when the light travels through different materials [36]. All of these effects should be accounted for in an accurate PSF measurement.

There are three methods of obtaining the PSF: experimental, analytical, and computational [35]. Experimental techniques involve imaging tiny fluorescent beads. If the beads are small enough, then the resulting images can be used as estimates of the effect of the system on a point source. Analytical techniques involve calculating the PSF based on mathematical models of diffraction. Finally, computational methods involve estimating the PSF along with the deblurred image simultaneously, using algorithms which perform so-called "blind deconvolution." Blind deconvolution uses image information to generate or improve the PSF estimate, and can be useful when the other two methods are impractical or impossible.

Microscope images are also affected by noise. The main source of noise is photon noise. Fluorescence is caused by the emission of photons from the sample, which is a random process that follows a Poisson distribution. The number of photons arriving at the detector is proportional to the intensity of the fluorescence signal [41]. Gaussian noise due to electronic devices can also be a factor, although it is typically less of an issue with modern equipment [36].

Fluorescence microscope images may also suffer degradations due to the nature of the samples. Normally when a fluorescent molecule is exposed to an appropriate light source it enters the excited state, then returns to the ground state, and so on. However prolonged exposure to the light source may mean that the molecule does not return to the ground state, and instead may undergo a chemical reaction and even become non-fluorescent. This is known as *photobleaching* [41, 36]. Another potential problem occurs when prolonged exposure to light damages the living sample. The

energy can result in the production of free radicals, which can lead to the death of the cell [36]. These issues impose a practical limit on the exposure time and the intensity of the illumination. Again, wide-field microscopes capture more light than confocal microscopes in the same exposure time, which may help reduce some of these negative effects, but the images will require deconvolution.

Chapter 3

Methods

There are a vast number of image deblurring algorithms. In this chapter we offer a quick introduction to some of the most basic and most popular algorithms, and then focus on the ones which will be used as benchmarks in the next chapter; namely projected gradient-conjugate gradient, Barzilai-Borwein, and projected Barzilai-Borwein. We then propose Barzilai-Borwein with Infeasible Iterates as an improvement over existing methods.

3.1 Direct Methods

These methods directly compute the deblurred solution, in contrast to iterative methods. Their advantage is their speed, which can be an important factor in real-time applications. However as we shall see, they also have some serious drawbacks, especially when the image data is noisy. These algorithms are included for completeness but will not be used as a basis of comparison in the upcoming sections.

3.1.1 Inverse Filtering

The simplest approach to image restoration is to estimate the solution $\hat{\mathbf{F}}$ in the frequency domain as to divide both sides of (2.5) by $\mathbf{H}$, such that

$$\hat{\mathbf{F}} = \frac{\mathbf{B}}{\mathbf{H}} \tag{3.1}$$

where the division is element-wise division. If there is any noise, then the expression becomes

$$\hat{\mathbf{F}} = \frac{\mathbf{B} + \mathbf{W}}{\mathbf{H}} = \mathbf{F} + \frac{\mathbf{W}}{\mathbf{H}}. \tag{3.2}$$

If there is no noise then this approach can work well, although it can be affected by numerical issues. However if there is noise, then even if we know $\mathbf{H}$ we cannot recover $\mathbf{F}$ exactly without knowing $\mathbf{W}$. Furthermore, if $\mathbf{H}$ is ever zero then the solution may

not even exist. Even if the values of $\mathbf{H}$ are merely very small, the second term can dominate the whole expression. For these reasons, inverse filtering is not useful in practice.

Fig. 3.1 shows the results of inverse filtering on a test image of a satellite. The solution does exist for this particular PSF, as its corresponding OTF does not contain any zeroes. However the solution is completely contaminated by noise, and therefore useless.

3.1.2 Wiener Filtering

Another approach to image deblurring is to minimize the mean square error between the true image $\mathbf{f}$ and the solution $\hat{\mathbf{f}}$, given by $\mathrm{E}\{\|\mathbf{f} - \hat{\mathbf{f}}\|^2\}$, where $\mathrm{E}\{x\}$ denotes the expectation value of x. In the Fourier domain, the minimum of this function is given by $\hat{\mathbf{F}} = \mathbf{R}^{\mathrm{W}}\mathbf{G}$, with

$$\mathbf{R}^{\mathrm{W}} = \frac{\mathbf{H}^*}{|\mathbf{H}|^2 + \frac{|\mathbf{W}|^2}{|\mathbf{F}|^2}} \tag{3.3}$$

where $\mathbf{H}^*$ denotes the complex conjugate of $\mathbf{H}$, $|\mathbf{H}|^2 = \mathbf{H}^*\mathbf{H}$, and $|\mathbf{W}|^2$ and $|\mathbf{F}|^2$ are defined similarly. This filter is known as a *least square error filter* or *Wiener filter* [15].

The second term in the denominator helps alleviate some of the problems of inverse filtering by providing regularization. As the noise increases, the regularization term increases, which reduces the effect of small values in $\mathbf{H}$. Note that in the absence of noise and numerical inaccuracies, Wiener filtering reduces to inverse filtering. Comparing the results of filtering noisy images in Fig. 3.1 and Fig. 3.2, it is clear that Wiener filtering performs much better than inverse filtering in the noisy case. However, Wiener filtering requires knowledge of both the noise $\mathbf{W}$ and the true image $\mathbf{F}$, which are not typically available.

3.1.3 Tikhonov Filtering

Tikhonov filtering [37] is a popular approach, where $\hat{\mathbf{F}} = \mathbf{R}^{\mathrm{Tik}}\mathbf{G}$ and the filter $\mathbf{R}^{\mathrm{Tik}}$ is given by

$$\mathbf{R}^{\mathrm{Tik}} = \frac{\mathbf{H}^*}{|\mathbf{H}|^2 + \lambda\mathbf{S}}. \tag{3.4}$$

$\mathbf{S}$ is a regularization operator, often chosen to be a derivative matrix, and λ is called the regularization parameter and controls the amount of smoothing. If λ is chosen

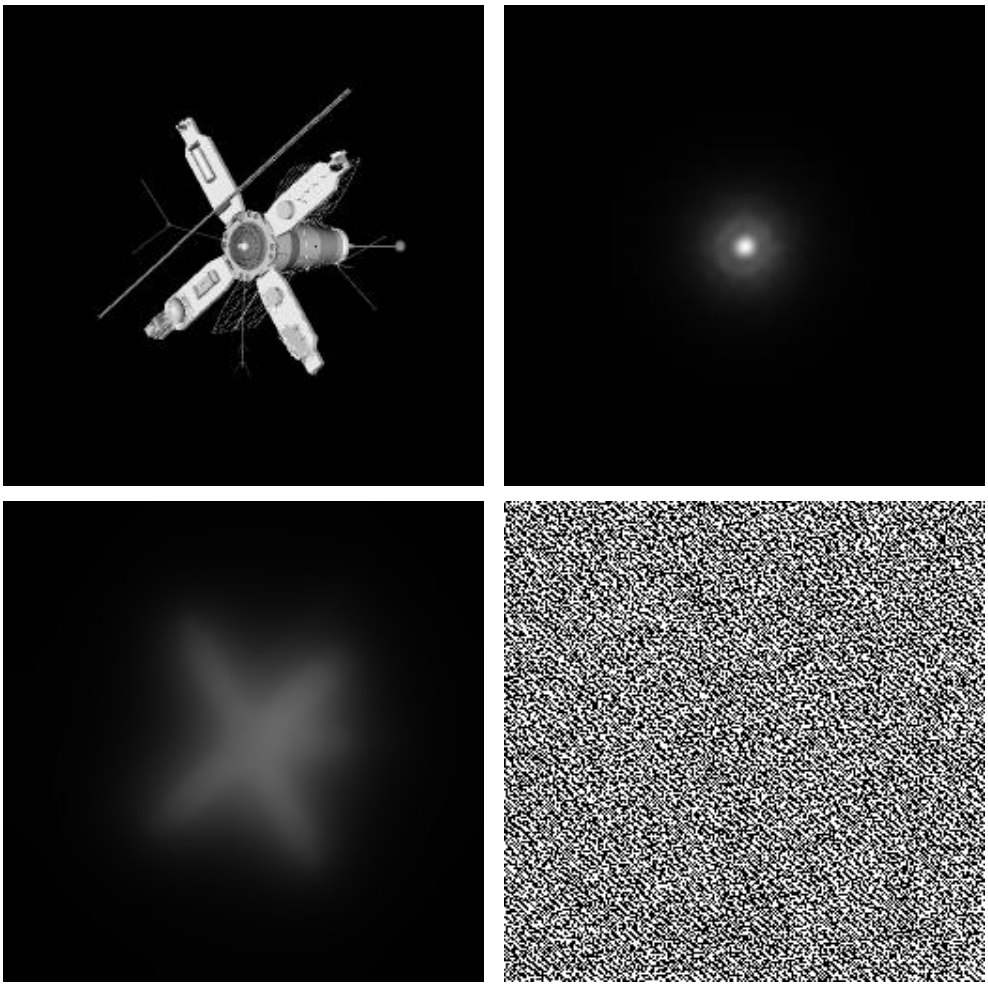


Figure 3.1: Inverse filtering. Top left, the original image. Top right, the point spread function used to blur the image (called"PSF1" in Chapter 4). Bottom left, the image blurred by the PSF and then degraded by additive Gaussian noise such that the BSNR = 20 dB. Bottom right, the results of inverse filtering.

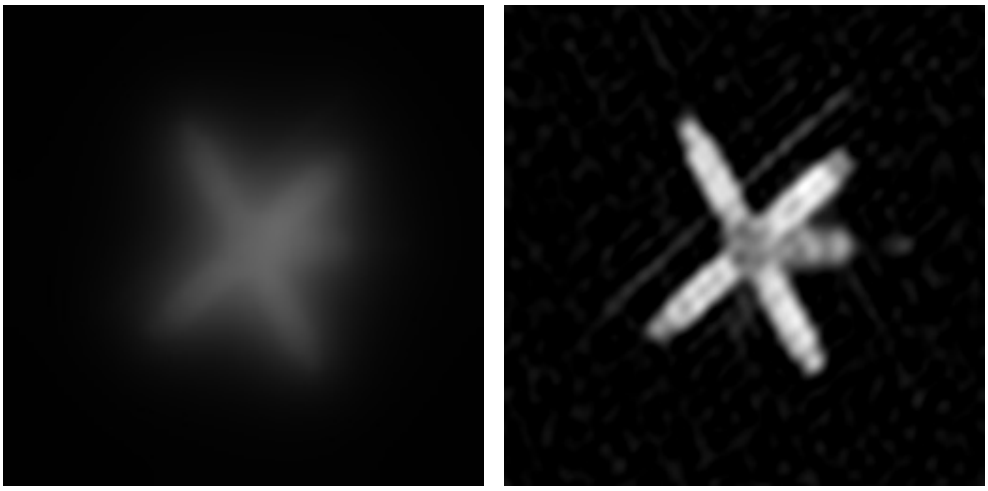


Figure 3.2: Wiener filtering. The image is the same same test case as in Fig. 3.1. On the left, the blurred and noisy image. On the right, the solution after Wiener filtering.

too small, then noise dominates the solution and it is said to be under-regularized. On the other hand, if λ is chosen too large, then the solution will be over-regularized and will be very blurry. Clearly, choosing an appropriate value of λ is crucial to the success of this algorithm. A number of papers have been written on different methods of choosing the regularization parameter, including the discrepancy criterion, the L-curve criterion, and the generalized cross-validation method [14, 20, 29, 18]. However, this is still an area of active research. As Hansen states in his recent book on inverse problems [18], "no one has devised a general-purpose parameter-choice algorithm which will always produce a good [regularization parameter], such that the regularized solution is a good approximation to the unattainable exact solution. What we currently have at our disposal is a collection of methods which, under certain assumptions, tend to work well; but all of them can and will occasionally fail to produce good results."

3.2 Iterative Methods

A different approach to image restoration is to treat it as a least squares problem. The residual of (2.8) is given by $\mathbf{A}\hat{\mathbf{f}} - \mathbf{g}$, ignoring the noise, so the least squares formulation is given by

$$\min \left\{ \tfrac{1}{2} \|\mathbf{A}\hat{\mathbf{f}} - \mathbf{g}\|_2^2 \right\}. \tag{3.5}$$

As an aside, the least squares formulation is equivalent to the maximum likelihood formulation in the case of Gaussian noise with zero mean [7]. Other likelihood functions may be more appropriate for other types of noise [11]. However, even in applications such as fluorescence microscopy, where the noise is more accurately described by a Poisson distribution, Verveer et al. [39] have found that, "even for high levels of noise the assumption of Gaussian noise did not yield results significantly inferior to those resulting from the (correct) assumption of the Poisson noise."

We can reformulate (3.5) as a quadratic programming problem:

$$\tfrac{1}{2}\|\mathbf{Af}-\mathbf{g}\|_2^2 \;=\; \tfrac{1}{2}\mathbf{f}^T\mathbf{A}^T\mathbf{Af}-\mathbf{f}^T\mathbf{A}^T\mathbf{g}+\tfrac{1}{2}\mathbf{g}^T\mathbf{g}. \tag{3.6}$$

It follows that the gradient is given by $\boldsymbol{\gamma}=\mathbf{A}^T\mathbf{Af}-\mathbf{A}^T\mathbf{g}=\mathbf{A}^T(\mathbf{Af}-\mathbf{g})$ and the Hessian is given by $\mathcal{H}=\mathbf{A}^T\mathbf{A}$. Iterative line search methods for minimizing (3.6) take the general form

$$\mathbf{f}_{k+1}=\mathbf{f}_k+\alpha_k\mathbf{d}_k \tag{3.7}$$

where $\mathbf{d}_k$ is the search direction for the k-th iteration, and α_k is the step size.

In many iterative methods, the number of iterations can be used as the regularization parameter [7]. In a plot of restoration error against number of iterations, the error decreases and then increases again. Too few iterations lead to an over-regularized, blurry solution. Too many iterations lead to a noisy, under-regularized solution, as the iterates converge to the inverse solution. This property is known as *semiconvergence* and is common to most iterative methods [18].

3.2.1 Steepest Descent

One way to find the minimum of (3.6) would be to step repeatedly in the direction of the negative of the gradient. In the classic steepest descent (SD) method, the search direction is given by

$$\mathbf{d}_k=-\boldsymbol{\gamma}_k \tag{3.8}$$

and the step size is given by

$$\alpha_k=\frac{\boldsymbol{\gamma}_k^T\boldsymbol{\gamma}_k}{\boldsymbol{\gamma}_k^T\mathbf{A}^T\mathbf{A}\boldsymbol{\gamma}_k}. \tag{3.9}$$

This method results in a strictly monotonic decrease in the function to be minimized. It has been proved to converge to the exact solution for quadratic functions in the

Figure 3.3: Slow convergence of steepest descent. This figure shows how the steepest descent method can be very slow to converge when the minimum lies in a long, narrow valley. (From *Numerical Recipes* [30].)

absence of noise [7]. However, one characteristic of steepest descent is that the direction of each step is always perpendicular to the direction of the previous step [30]. This can lead to very slow convergence, as illustrated in Fig. 3.3.

See Fig. 3.4 for an example of iterative regularization using the steepest descent method. We can see that as the number of iterations increases, the error decreases and then increases again, illustrating the semiconvergence property discussed in Sec. 3.2.

The Landweber method is similar to steepest descent but uses a fixed step size α such that $0 < \alpha < 2/\|\mathbf{H}\|^2$. For a complete discussion, see [7].

3.2.2 Conjugate Gradient

One of the reasons that the SD method can be slow to converge is that the algorithm takes repeated steps in the same direction. It would be better to take only one step in each direction, leading to convergence after n steps in n dimensions. The conjugate gradient (CG) method takes steps which are conjugate to the gradient, as well as conjugate to the previous steps, thus avoiding those unnecessary iterations [30]. In the 2D case illustrated in Fig. 3.3, for any point on one of the level curves, the conjugate gradient points towards the center of the ellipse [7]. Thus from any starting point, the CG method can reach the minimum point in at most two steps.

The CG update rule is given by (3.7) with

$$\alpha_k = \frac{\|\boldsymbol{\gamma}_k\|^2}{\mathbf{d}_k^T \mathbf{A}^T \mathbf{A} \mathbf{d}_k}. \tag{3.10}$$

The first step is taken in the direction of the negative of the gradient, with the search direction update given by

$$\mathbf{d}_{k+1} = -\boldsymbol{\gamma}_{k+1} + \beta_k \mathbf{d}_k \tag{3.11}$$

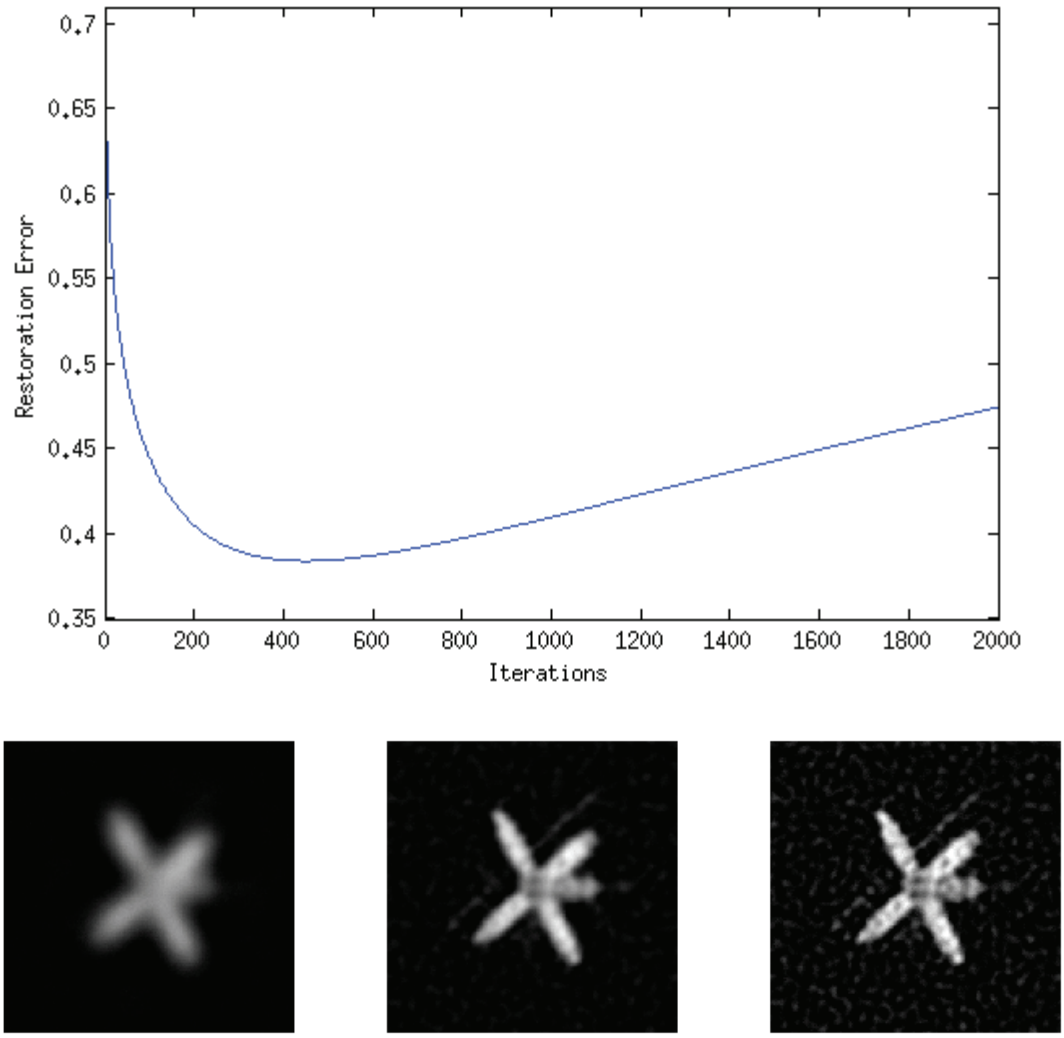


Figure 3.4: Iterative regularization. On top, the restoration error shows semiconvergent behaviour. On the bottom left, the solution after 20 iterations is over-regularized. In the middle, the solution after 450 iterations is near the minimum of the restoration error curve. On the bottom right, the solution after 2000 steps is under-regularized.

with

$$\beta_k = \frac{\|\boldsymbol{\gamma}_{k+1}\|^2}{\|\boldsymbol{\gamma}_k\|^2} \quad \text{and} \quad \boldsymbol{\gamma}_{k+1} = \boldsymbol{\gamma}_k + \alpha_k \mathbf{A}^T\mathbf{A}\mathbf{d}_k \tag{3.12}$$

for positive quadratic functionals [40]. The CG method results in good solutions for quadratic or nearly quadratic problems. However its performance worsens in more general problems or when constraints are added, and it has been shown to exhibit unstable behaviour on some discrete, ill-posed problems [27, 12, 38].

3.2.3 Barzilai-Borwein

The Barzilai-Borwein (BB) method was introduced by Jonathan Barzilai and Jonathan Borwein in 1988 [6]. The BB method has been observed to be faster and less sensitive to ill-conditioning than the classical steepest descent method [6]. Dai and Fletcher state that it has "completely changed our perspectives on the effectiveness of gradient methods" [8]. It uses the same search direction as steepest descent ($\mathbf{d}_k = -\boldsymbol{\gamma}_k$) but replaces the step size with the option of two different possible step sizes:

$$\alpha_k^{BB1} = \frac{\boldsymbol{\gamma}_{k-1}^T\boldsymbol{\gamma}_{k-1}}{\boldsymbol{\gamma}_{k-1}^T\mathbf{A}^T\mathbf{A}\boldsymbol{\gamma}_{k-1}} \tag{3.13}$$

$$\alpha_k^{BB2} = \frac{\boldsymbol{\gamma}_{k-1}^T\boldsymbol{\gamma}_{k-1}}{\boldsymbol{\gamma}_{k-1}^T(\mathbf{A}^T\mathbf{A})^2\boldsymbol{\gamma}_{k-1}} \tag{3.14}$$

Most subsequent research has focused on (3.13), although there is evidence which suggests the two are not dissimilar [12], and strategies which alternate between the two or which adaptively choose one over the other have been proposed [8, 44]. Note that (3.13) is equivalent to the SD step length, but using the gradient information from the previous iteration. This has led to the idea of using gradient information from two or more iterations ago, but in most cases the benefits were negligible, if any [13, 8, 38].

The Barzilai-Borwein method is non-monotonic; that is, the objective function does not strictly decrease but may in fact increase in some iterations. Barzilai and Borwein proved that it converges for the two-dimensional quadratic case [6]; it has since been proven to converge in the n-dimensional convex quadratic case [31]. For the non-quadratic case, it must be combined with a line-search in order to guarantee convergence [32].

In general, however, the behaviour of the BB method is not well understood. It often leads to faster convergence than SD but it is not clear why this is the case. In his 2005 review paper [12], Fletcher states, "One thing that I think emerges from this review is just how little we understand about the BB method." More recently, van den Doel and Ascher [38] discuss BB along with other similar methods, which they together call *faster gradient methods*, and write, "The state of the theory for these methods is currently unsatisfactory. There exist essential convergence theorems, but they are at best as strong as the simple convergence theorems available for SD, and they do not explain why the rate of convergence of these faster gradient descent methods is more like that of CG than SD." The step sizes can vary across different orders of magnitude, and a plot of the restoration error shows seemingly erratic peaks and valleys as a result (see Fig. 3.5). It has been suggested that the large step sizes have the effect of increasing the large components of the gradient, and lead to the spikes in the error. This in turn reduces the contribution of the small components of the gradient, and allows those components to be removed more easily than in monotonic methods, such as steepest descent [12]. However, as stated by van den Doel and Ascher in [38], "no one we know expects any gradient descent variant ever to perform better than CG for the solution of $\mathbf{Ax} = \mathbf{b}$, provided that matrix-vector multiplications $\mathbf{Av}$ for any given vector $\mathbf{v}$ are carried out accurately."

3.3 Constrained Methods

We want to use as much *a priori* information about a problem as we can when computing its solution. One piece of information relevant to image restoration problems is that because the pixel values represent light intensities, they can never be negative. In mathematical terms, this means we now want to consider the constrained optimization problem given by

$$\min J(\mathbf{f}) = \min \left\{ \tfrac{1}{2} \|\mathbf{Af} - \mathbf{g}\|_2^2 \right\} \text{ such that } \mathbf{f} \geq 0 \tag{3.15}$$

where by $\mathbf{f} \geq 0$ we mean that each element of $\mathbf{f}$ is nonnegative. The set defined by $\mathbf{f} \geq 0$ is closed and convex, and is known as the *feasible set* Ω. This is a special case of the more general box-constrained quadratic programming problem, in which we let the lower bound be zero and the upper bound be infinity. We let the projection

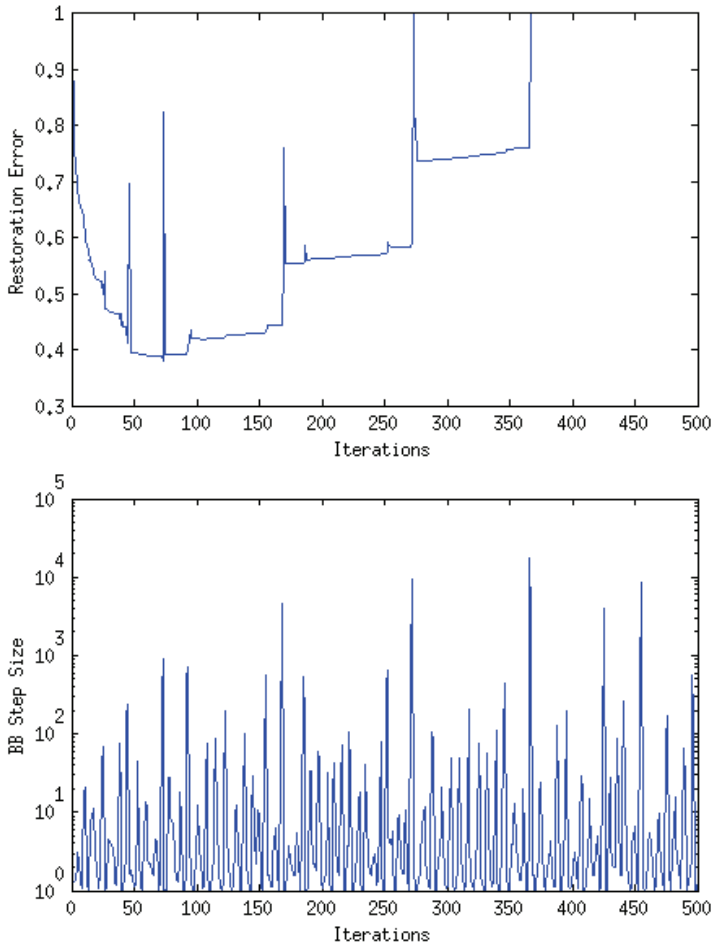


Figure 3.5: The Barzilai-Borwein method. We consider again the same test case as in Fig. 3.1. On top, the restoration error shows non-monotonic behaviour. Below, the step sizes span several orders of magnitude and do not show any discernible pattern [38].

operator $\mathcal{P}_\Omega$ be defined as

$$\mathcal{P}_\Omega(\mathbf{f}) = \mathbf{f}^{\mathrm{P}} \text{ s.t. } \mathbf{f}^{\mathrm{P}} = \begin{cases} f_i^{\mathrm{P}} = f_i \text{ for } f_i \geq 0 \\ f_i^{\mathrm{P}} = 0 \text{ otherwise .} \end{cases} \tag{3.16}$$

Enforcing the nonnegativity constraint may in some cases be expensive compared to the methods outlined in previous sections, but it can produce much more accurate solutions [17, 26]. It is particularly useful to constrain the solution in images which have large dark areas, such as astronomical images [17] and microscope images [39]. Nonnegativity constraints have also been shown to have a regularizing effect [5].

3.3.1 Gradient Projection

The gradient projection (GP) method is basically an extension of the steepest descent method, adapted to handle the nonnegativity constraint [40]. The search direction is given by

$$\mathbf{d}_k = -\boldsymbol{\gamma}_k \tag{3.17}$$

and the step size is chosen by a line search such that

$$\alpha_k = \arg\min_{\alpha>0} \; J(\mathcal{P}_\Omega(\mathbf{f}_k + \alpha_k \mathbf{d}_k)). \tag{3.18}$$

The update rule is then given by

$$\mathbf{f}_{k+1} = \mathcal{P}_\Omega(\mathbf{f}_k + \alpha_k \mathbf{d}_k). \tag{3.19}$$

Given a box-constrained optimization problem, GP has been shown to identify the optimal face in a finite number of iterations, after which it behaves like the SD method [4]. This can result in slow convergence and the "zig-zagging" effect that was seen with steepest descent [42].

3.3.2 Gradient Projection-Conjugate Gradient

The gradient projection-conjugate gradient method (GPCG) aims to combine the benefits of GP and CG in one algorithm. It was initially developed by Moré and Toraldo [25].

We first review some necessary terms. With the nonnegativity constraint, the *active set* of a feasible point $\mathbf{f} = (f_1, f_2, \ldots, f_n)$ is

$$\mathcal{A}(\mathbf{f}) = \{i \mid f_i = 0\}. \tag{3.20}$$

Points which are not in the active set are in the *inactive set*

$$\mathcal{I}(\mathbf{f}) = \{i \mid f_i > 0\}. \tag{3.21}$$

The set of points currently under consideration in the k-th iteration is called the current face

$$\mathcal{F}(\mathbf{f}_k) = \{\mathbf{f} \in \Omega \mid f_i \ = 0 \text{ for } i \in \mathcal{A}(\mathbf{f}_k) \text{ and } f_i > 0 \text{ otherwise}\}. \tag{3.22}$$

The general idea of using GPCG for box-constrained problems is to use the GP method to identify the optimal face, and then use the CG method to search the face. In the first step of GPCG, a sequence of GP iterates are produced until either it is determined that GP is not making sufficient progress, or we are near the optimal face. In either scenario, the algorithm now uses CG to create a series of iterates in the current face. If CG is not making sufficient progress, then perform a line search to determine α_k, and calculate $\mathbf{f}_{k+1}$. If $\mathbf{f}_{k+1}$ is in the optimal face, then return to the CG step and continue searching that face. Otherwise, go back to the GP step to find a new face.

GPCG is known to be successful for large-scale constrained quadratic programming problems, and therefore makes a good baseline against which to compare new algorithms [42, 8].

3.3.3 Projected Barzilai-Borwein

The BB method was originally developed to solve the unconstrained minimization problem, but it can be adapted to work in the constrained case by projecting the solution onto the feasible set after every iteration [42, 8]. It is then known as the projected Barzilai-Borwein (PBB) method. The new update rule becomes

$$\mathbf{f}_{k+1} = \mathcal{P}_\Omega(\mathbf{f}_k + \alpha_k^{BB}\mathbf{d}_k). \tag{3.23}$$

The PBB method requires the addition of a line search strategy in order to safeguard the decrease of the objective function and guarantee global convergence [42]. However, it has been reported that using a line search reduces the effectiveness of the PBB method, and that PBB does not usually fail to converge except on specially constructed test cases, even without a line search [8].

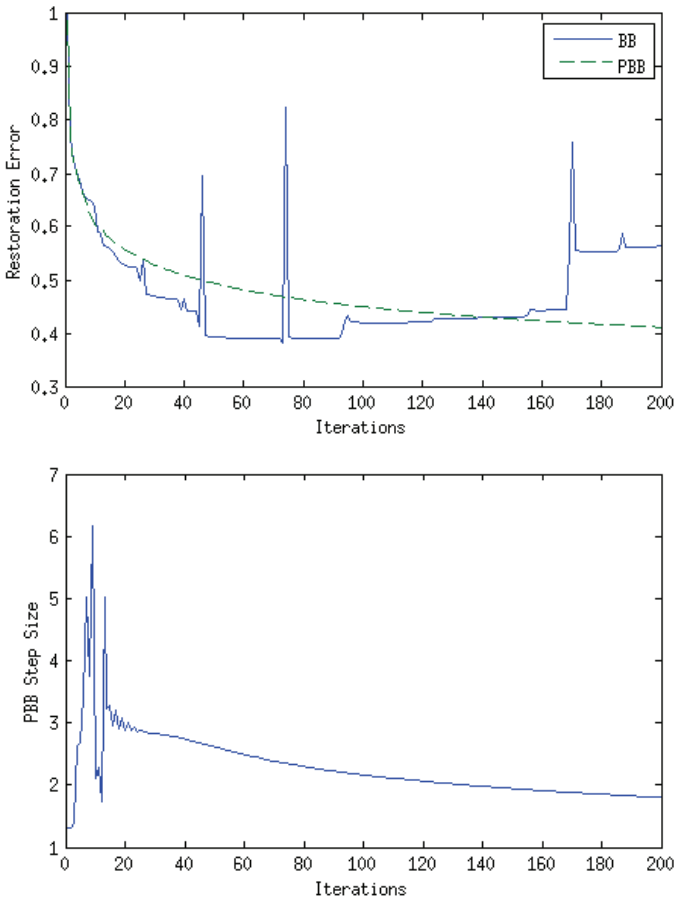


Figure 3.6: Comparing the performance of BB and PBB. On top, the restoration error of PBB decreases smoothly but does not reach as low a value as BB in 200 iterations. Below, there is a marked difference in the projected BB step size as compared to the original BB step size (see Fig. 3.5 for comparison). Data is for the same test case as Fig. 3.1.

3.4 Barzilai-Borwein with Infeasible Iterates

The projected Barzilai-Borwein method adapts the BB method for use in the constrained case; however in the process it diminishes some of the benefits offered by the conventional BB method. In Fig. 3.6, the restoration error for BB reaches a minimum of 0.38 after 73 iterations; by 200 iterations PBB has achieved an error value of 0.41. The descent of PBB is smooth, rather than showing the jumps which are characteristic of the BB method. The reason for this lies in the step size, which after the first several steps settles into a smooth, slowly decreasing sequence, in contrast to the BB step sizes shown in Fig. 3.5.

We propose a new algorithm, which projects the solution onto the feasible set less frequently than PBB. If a projection does not occur on a given iteration, some of the pixel values may be less than zero. These are called *infeasible iterates*, because the solution does not satisfy the constraints. In a recent paper [2], Audet and Dennis state, "It is not unreasonable to expect in nonlinear optimization that allowing constraint violations in the course of solving a problem often enables one to solve the problem with fewer function and constraint evaluations."

Pseudo-code for the new algorithm, which we call Barzilai-Borwein with Infeasible Iterates (BBII), is given in Algorithm 1. For Matlab code, see Appendix A.

The goal of the algorithm is to allow for much of the non-monotonic behaviour which characterizes BB, with projection steps occurring only when the solution violates the constraints too strongly. The question is: how do we define "too strongly"? The PBB method never allows any pixel values to fall below zero without projecting. A softer criterion might be to let only some percentage of pixel values fall below zero, or let them fall only to some minimum value $v_{min} < 0$ before projecting. Here, after some experimentation, we choose the following rule: calculate the average squared value of all the negative pixel values, and divide by the average squared value of all the pixels. That is,

$$r = \frac{\frac{1}{M} \sum_{i \mid f_i < 0} f_i^2}{\frac{1}{N} \sum_i f_i^2} \tag{3.24}$$

where M is the number of negative pixels in $\mathbf{f}$ and N is the total number of pixels in

Algorithm 1 Barzilai-Borwein with Infeasible Iterates

Input: Blurred and noisy image $\mathbf{g}$, PSF $\mathbf{h}$, number of iterations $k_{\max}$, initial threshold $\tau > 0$, decay rate $0 < \rho \leq 1$.

Select initial guess $\mathbf{f}_0$, and set $k = 1$.
Calculate the initial gradient $\boldsymbol{\gamma}_1$.
for $k = 1 : k_{\max}$ **do**
  if $k == 1$ or a projection occurred on the previous step **then**
    $\alpha_k = \alpha_k^{SD}$
  else
    $\alpha_k = \alpha_k^{BB1}$
  end if
  $\mathbf{f}_k = \mathbf{f}_{k-1} - \alpha_k \boldsymbol{\gamma}_k$
  Calculate r_k using (3.24) with $\mathbf{f} = \mathbf{f}_k$.
  if $k \geq 10$ and $\text{median}(r_{k-9}, r_{k-8}, \ldots, r_k) > \tau$ **then**
    Project onto the feasible set.
    Reduce threshold $\tau = \rho\tau$.
  end if
  Calculate the new gradient $\boldsymbol{\gamma}_{k+1}$.
end for

f. When the value of r increases past some threshold τ, then project.

For an example of why this rule might work, consider Fig. 3.7. For the first several iterations, projecting has little benefit in terms of reducing the restoration error. However, it seems that after around 75 iterations, the restoration error would be significantly decreased if the solution was now projected onto the nonnegative set. Looking at the plot of the average negative value, this corresponds to projecting when the value of r increases past some threshold, in this case perhaps $\tau = 0.05$.

In fact, it is not quite this simple, for reasons which are also apparent from Fig. 3.7. The increase in r which occurs at 75 iterations corresponds to the "jumpy" behaviour in the restoration error. In order to avoid projecting in a peak, which could increase the restoration error, we instead keep track of the r values for the past 10 steps. Rather than project when the *current* value reaches τ, we project when the *median* of the values over the past 10 steps reaches τ. This effectively postpones the projection step until after the solution has settled down, leading to a larger decrease in the error. Furthermore, using the median has the effect of low-pass filtering the r values. This effectively removes "spikes" in the r values, such as that seen at 45 iterations in the middle plot in Fig. 3.7, which rapidly increase and then immediately decrease the r value. In the bottom plot of Fig. 3.7, we can see that this spike has been removed by considering the median.

In some cases, particularly cases with low BSNR, the PBB method actually does quite well. As an example, consider Fig. 3.8. The BB method does well for the first few iterations, but the PBB method soon achieves a lower error norm by projecting in every step. Ideally, we would like an algorithm which can range between never projecting, in cases where BB does well on its own, and always projecting, in cases where PBB does better. To allow for this, we choose an initial threshold $\tau > 0$ so that the algorithm can benefit from the fast convergence the BB method shows in the first several iterations. Then every time a projection occurs, we decrease the threshold slightly, by a factor of ρ. For cases where the pixel values do not tend to fall much below zero, the threshold stays close to the initial value and projections occur rarely. For cases where the average squared negative pixel value is consistently high, the value of the threshold decreases and approaches zero. Thus the behaviour of the algorithm approaches that of PBB as the number of projections approaches infinity.

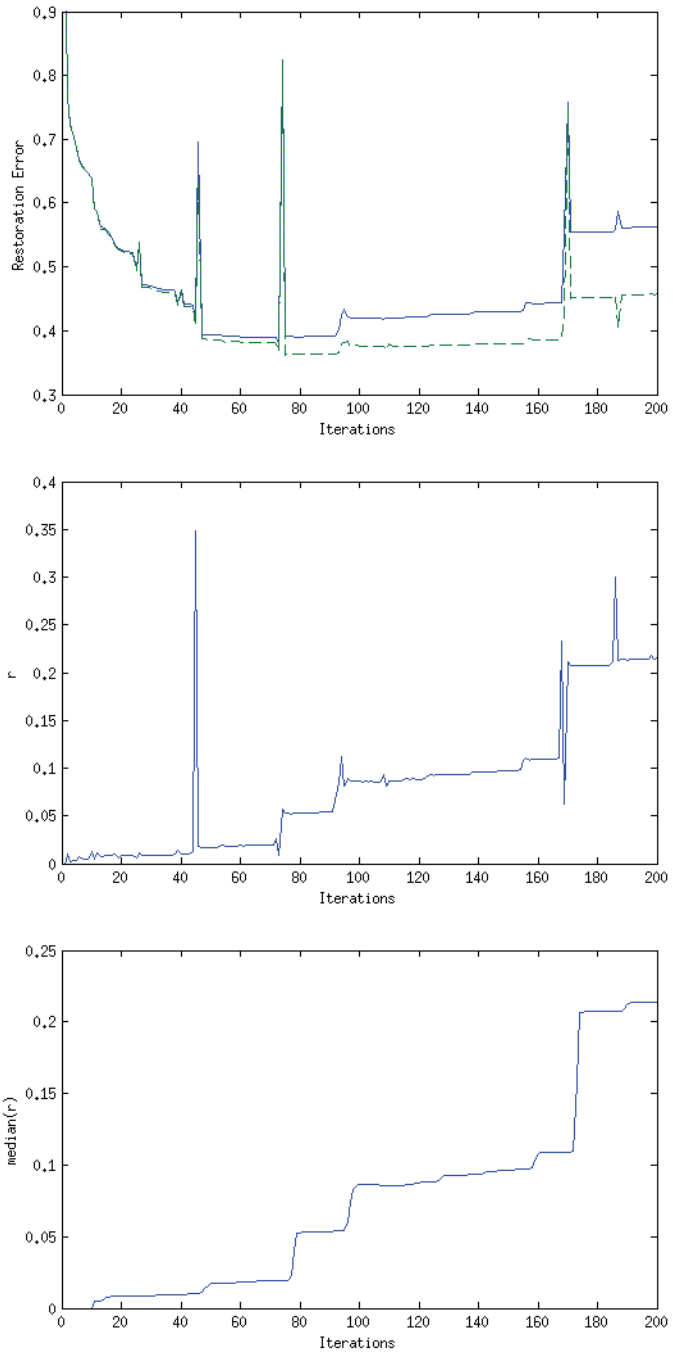


Figure 3.7: Relationship between restoration error and r value. On top, the blue (solid) line shows the restoration error for the BB method. The green (dashed) line shows what the restoration error *would* be, if the solution was projected on that iteration. In the middle, the value of r, given by (3.24), at each iteration. On the bottom, the median value of r over the past 10 iterations. Data is for the same test case as Fig. 3.1.

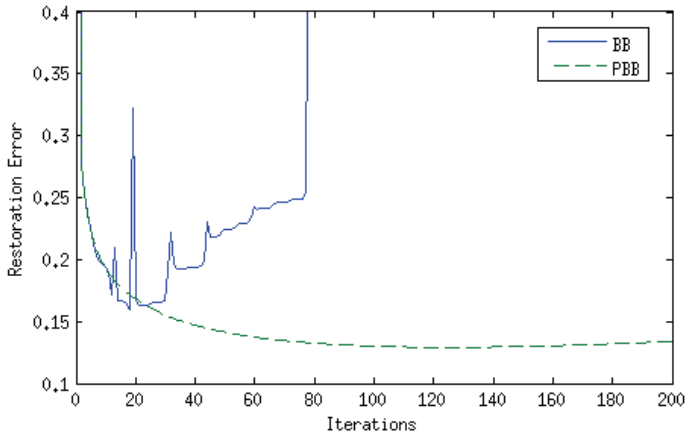


Figure 3.8: Good performance of PBB. Shown is the restoration error for the head test image from Chapter 4, blurred by a motion blur of length 20 at an angle of 45°, with BNSR = 30. BB is the blue (solid) line and PBB is the green (dashed) line.

The general idea of allowing infeasible iterates and then gradually tightening the constraints has been the focus of a number of recent papers in the field of sequential quadratic programming, appearing under different names: *trust funnel* method [16], *progressive barrier* method [2], and *tolerance tube* method [34]. However it has never been applied to the problem of image deconvolution or to any variant of the BB method.

Compare the results of BBII in Fig. 3.9 with those of PBB in Fig. 3.6. The restoration error of BBII is non-monotonic, like that of BB. The step sizes cover a wide range of values. Rather than projecting on every step, projections occur occasionally. The projections are important though: at around 90 iterations, the solution is projected on several iterations, and at the same time the restoration error for BBII decreases while the error for BB increases. It is our hope that the BBII algorithm will be able to give us "the best of both worlds," that is, that we can benefit from the large step sizes and fast convergence of BB, while at the same time use the nonegativity constraint to increase the quality of the solution.

3.5 Stopping Criteria

Choosing an appropriate stopping point for iterative methods is not an easy task [27]. One approach is called the discrepancy principle, and it involves stopping the

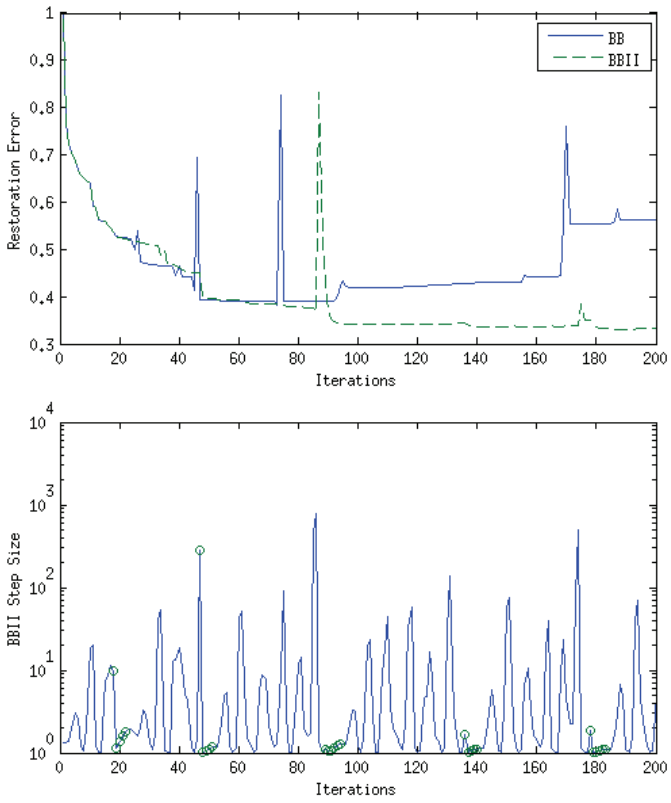


Figure 3.9: Comparing the performance of BB and BBII. On top, the restoration error for the two methods is the same until the first projection in BBII. After the point that the BB error starts to increase again, the BBII error is still decreasing. On the bottom, the step sizes for BBII. Iterations on which a projection occurs are marked with green circles. Data is for the same test case as Fig. 3.1.

iterations when $\|\mathbf{A}\mathbf{f}_k - \mathbf{g}\| = e$, where e is an estimate of the energy of the noise [7]. The discrepancy principle is useful only in cases where a good estimate of e is available. Another common stopping criterion is $\|\boldsymbol{\gamma}_k\| \leq \epsilon\|\boldsymbol{\gamma}_1\|$, where ϵ is a tolerance which can be set as high or low as necessary for the application at hand [8, 42, 44]. Yet another approach is to calculate the relative error norm, given by $\|\mathbf{f}_{k+1} - \mathbf{f}_k\|/\|\mathbf{f}_{k+1}\|$, and stop the iterations when the relative error norm falls below some threshold [21]. In a similar vein, the relative residual norm $\|\mathbf{A}\mathbf{f}_k - \mathbf{b}\|/\|\mathbf{A}\mathbf{f}_0 - \mathbf{b}\|$ can be computed, and iterations stopped when it falls below some threshold [38].

Rather than get involved in a comparative study of all possible stopping criteria, we assume here that an appropriate stopping criterion has been established, and concern ourselves instead with which methods achieve the lowest error norm, and how long it takes to get there. This is a common methodology in research on image restoration algorithms [17, 27].

Chapter 4

Results

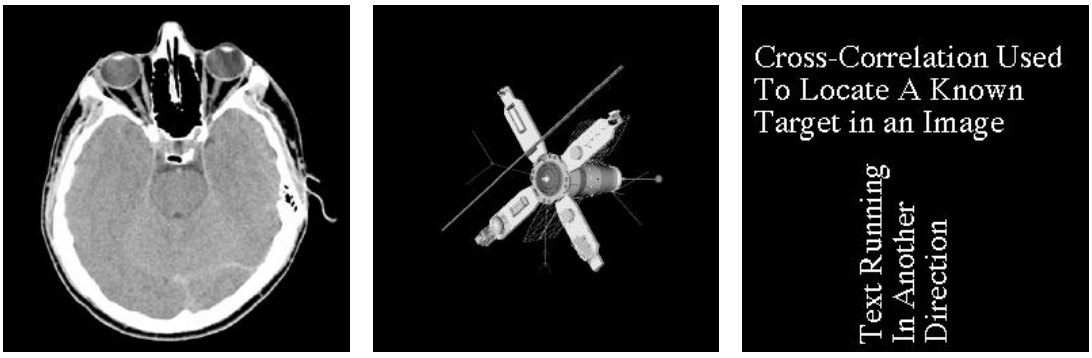
We calculate the restoration error for BBII for a number of test cases, and compare it to the restoration error for unconstrained BB. Additionally, we compare with PBB, which has been the subject of recent research on how to apply BB to a constrained optimization problem [8, 42]. Finally we also compare with GPCG, which has been used as a basis of comparison for PBB in the previously mentioned works [8, 42], and was studied in depth in a recent PhD thesis [4].

4.1 Two-Dimensional Image Restoration

Here we test the algorithms on three two-dimensional images, with four blurring matrices and three different noise levels, for a total of 36 test cases. We show the resulting restoration error curves, as well as the minimum error achieved in each test case.

4.1.1 Test Cases

Nonnegativity constraints are most useful in cases where the images contain large dark areas, where pixel values are close to zero [17]. For this reason, and also to avoid boundary value effects, we choose three well-known test problems with black backgrounds, shown in Fig. 4.1. The first image we consider is a CT scan of a human head, from [15] and available at `http://www.imageprocessingplace.com/root_files_V3/image_databases.htm`. The second image is a satellite test image from the US Air Force Phillips Laboratory, Laser and Imaging Directorate, Kirtland Air Force Base, New Mexico [27]. We consider the version included with the Matlab package RestoreTools, available at `http://www.mathcs.emory.edu/nagy/RestoreTools`. More information about the RestoreTools package can be found in [28]. Along with the true image, the data includes a PSF which simulates the blurring effect of atmospheric turbulence. We call this blurring matrix "PSF1". The third image we consider is also



(a) Head (b) Satellite (c) Text

Figure 4.1: Test images.

from the RestoreTools toolbox, and consists of white text on a black background. Each of the images is of size 256×256.

As well, we include three additional PSFs. The PSFs are shown in Fig. 4.2. The first is a Gaussian with a standard deviation of seven. It can be reproduced in Matlab by typing:

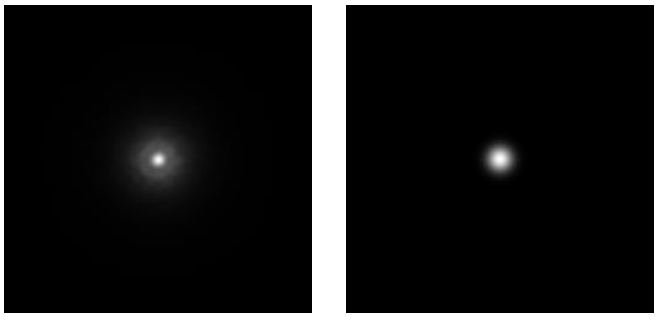
```
PSF=fspecial('Gaussian',256,7);
```

We label it "PSF2". Gaussian blur can be used to model atmospheric turbulence [3]. (Note however that PSF1 is not a Gaussian.) Next we consider "PSF3", which models motion blur. It can be reproduced by:

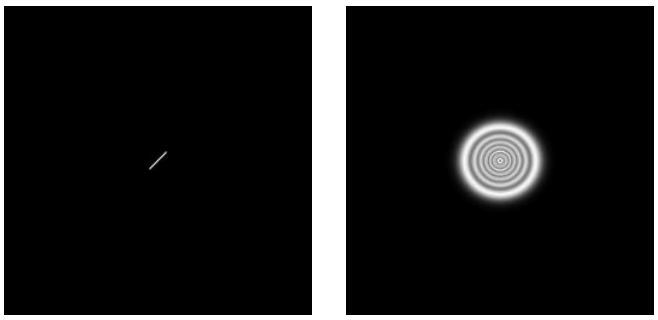
```
PSF=fspecial('motion',20,45);
```

The parameters indicate a motion across 20 pixels and in the direction of 45° below the horizontal. Finally, "PSF4" is a two-dimensional slice taken from a three-dimensional microscope PSF, from the test data available at `http://bigwww.epfl.ch/deconvolution/`.

In addition to blurring the images, we add zero-mean Gaussian noise to the images. Gaussian noise is a good model for the noise in many real-life applications [3]. Even in cases where the noise is not actually Gaussian, it can serve as a good approximation, especially in high SNR cases, due to the central limit theorem [35, 36]. We test at three different noise levels: 20 dB, 30 dB, and 40 dB. At 40-50 dB the noise is not visible in the image anymore [7], and below 20 dB the images are generally too corrupted and it is impossible to achieve a good solution.



(a) PSF1 (b) PSF2



(c) PSF3 (d) PSF4

Figure 4.2: Test PSFs. All PSFs have been padded with zeros to size 256×256. The images have been scaled for display purposes.

4.1.2 Parameters

It has been shown that the Barzilai-Borwein method is more sensitive to the initial starting point than either the steepest descent or conjugate gradient methods [38]. Given the findings of the aforementioned paper, we choose the initial starting point for BB, PBB, and BBII to be a matrix of all zeros. The first BB step length is arbitrary, so we use a steepest descent step length for α_1, given by (3.9).

For GPCG, we use the default parameters given in the code by Johnathan Bardsley which accompanies [40], available at `http://www.math.montana.edu/~vogel/`.

For BBII, we must choose both an inital threshold value τ and a decay rate ρ. To choose τ, we use the following heuristic: decide on the average minimum value, and then calculate what the corresponding value of r would be. In these cases, we choose an average negative value of -0.01, which, using (3.24), leads to an r value of

$$\frac{(-0.01)^2}{\frac{1}{N}\sum_i g_i^2} \tag{4.1}$$

where $\mathbf{g}$ is the test image at hand and $N = 256^2 = 65536$, since each image is of size 256×256. We choose this value as the initial threshold. For example, for the head test case with PSF1 and SNR 20, this leads to $\tau = 0.00065$. In contrast, for the text test case with PSF4 and SNR 40, it leads to $\tau = 0.0161$. If a lower average minimum value was chosen, then the initial threshold would be higher, and the method would explore further into the infeasible region before a projection occured. Other heuristics could surely be derived, but after much experimentation we find that the important thing is to keep in mind the physical meaning of the quantities, rather than choose numbers blindly.

For the decay rate, we use $\rho = 0.97$, and we compute the median average negative value over the previous 10 iterations.

4.1.3 Evaluation and Efficiency

We use the restoration error, defined by (2.10), to evaluate the performance of the algorithms. We do not plot the error vs. the number of iterations, because the GPCG algorithm computes many inner GP and CG iterations within a single outer GPCG iteration, making it many times more computationally intensive per iteration than the other algorithms we consider. Rather, in all of the algorithms we consider, the most computationally costly operation is the matrix-vector multiplication, which is computed using a Fast Fourier Transform (FFT). The complexity of the FFT is $\Theta(mn \log(mn))$ for an image of size m by n [15]. The next most expensive operation in our algorithms is element-wise multiplication, which is $O(mn)$, and calculation of the ℓ_2 norm, which is also $O(mn)$. So instead we plot the restoration error against the total number of FFTs (including inverse Fast Fourier Transforms, or iFFTs).

One advantage to BB is that, like SD, it only requires two FFTs per iteration using the recurrent gradient update rule

$$\boldsymbol{\gamma}_{k+1} = \boldsymbol{\gamma}_k - \alpha_k \mathbf{A}^T \mathbf{A} \boldsymbol{\gamma}_k. \tag{4.2}$$

In PBB, the recurrent gradient update does not hold, and the true gradient must be computed at each step. The overall cost per iteration is three FFTs. (For the reasons discussed in Sec. 3.3.3, we do not include a line search in our implementation.) One advantage of BBII over PBB is that the true gradient need only be computed after a

projection step, so the average cost per iteration is between two and three FFTs per iteration, depending on how many projections are necessary.

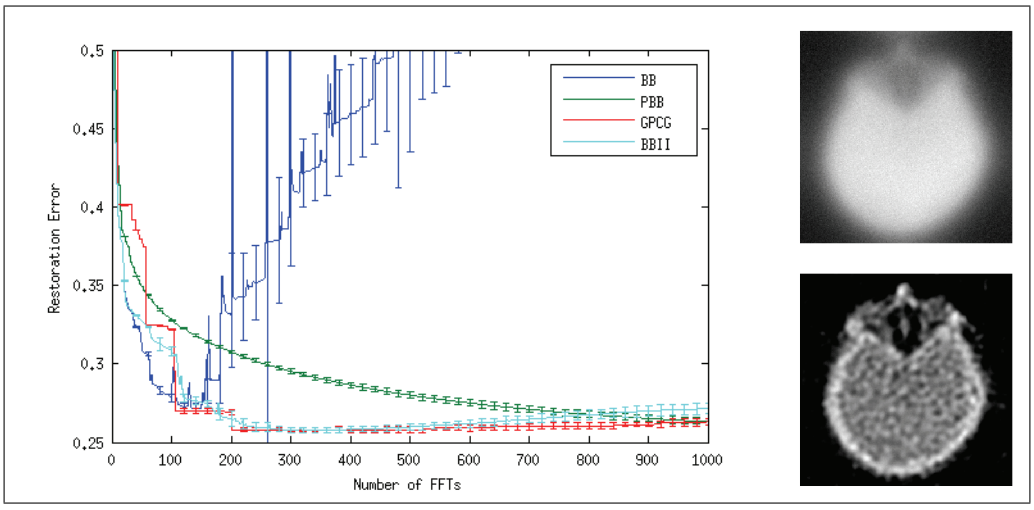
For GPCG, the cost per iteration depends on how many GP and CG iterations are performed. The number of FFTs is output by Bardsley's implementation. In some cases, many FFTs are computed before the solution update is performed, which accounts for the "boxy" nature of the GPCG plots.

4.1.4 Restoration Error

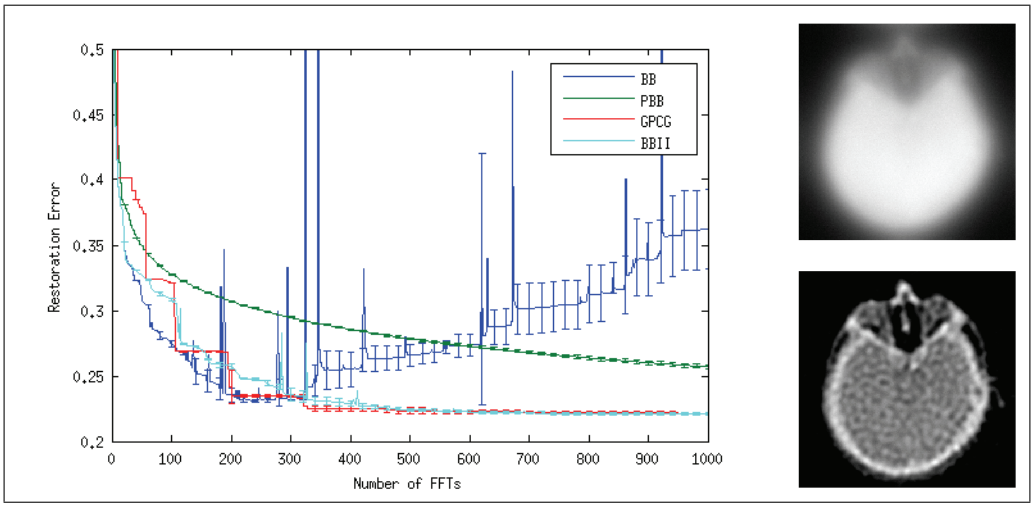
In this section we present plots of the restoration error vs. the number of FFTs for the four algorithms. Note the different scales on the x- and y-axes. The curves are averaged over five different noise configurations. The error bars represent one standard deviation from the mean, and are displayed only every 20 FFTs to improve clarity.

To the right of each plot, two images are displayed. The top image is the blurred and noisy image (for one noise configuration), and the bottom image is the resulting image after deconvolution using BBII, where we stop the iterations at the minimum point in the restoration error curve.

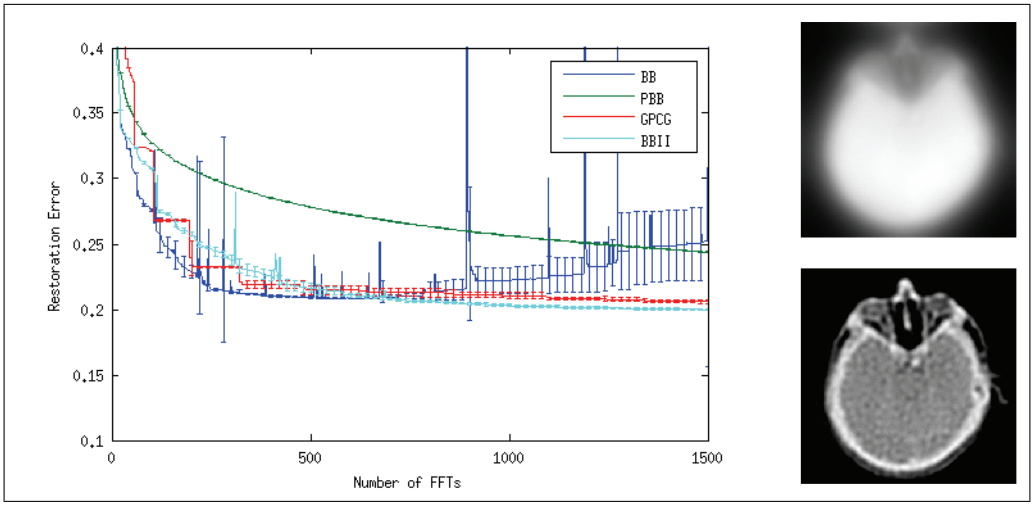
We then display tables showing the minimum restoration error for each method and test case, and the number of FFTs required to reach the minimum value. The method which achieves the lowest value in each test case is indicated with bold font. Note that the algorithms were stopped after a maximum of 2000 FFTs.



(a) SNR 20

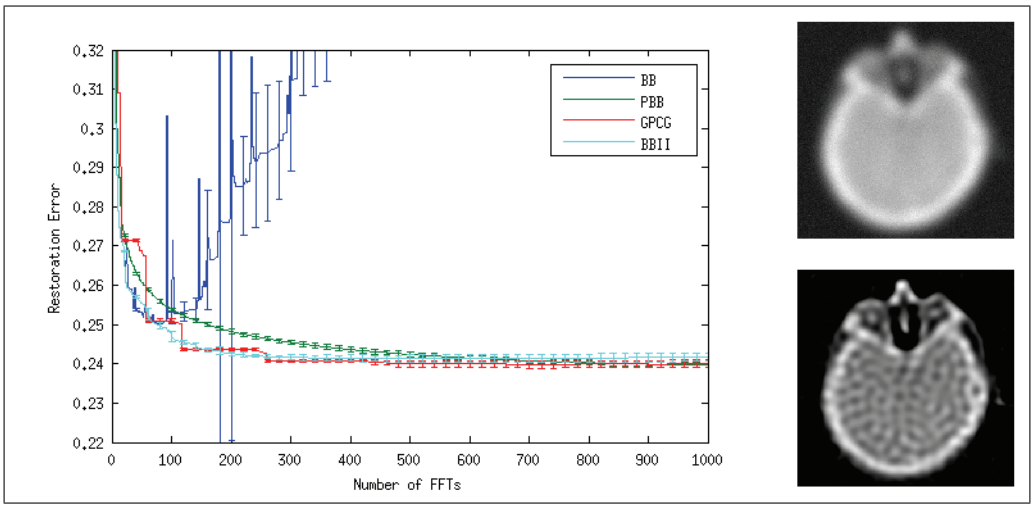


(b) SNR 30

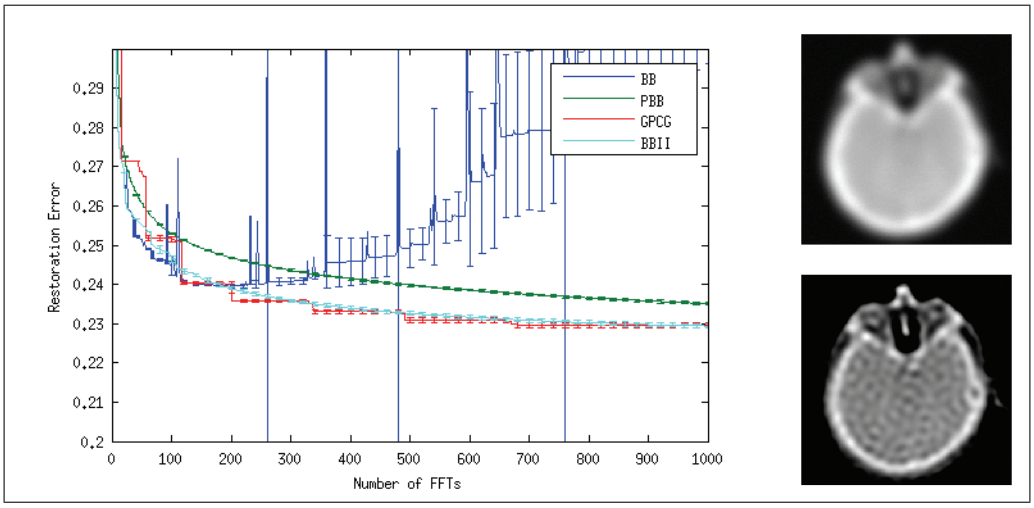


(c) SNR 40

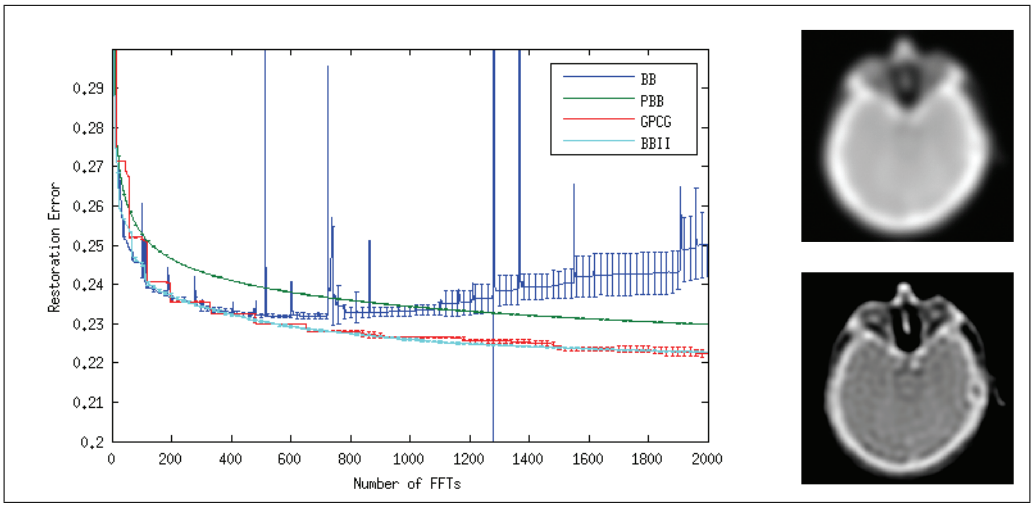
Figure 4.3: Head test image with PSF1



(a) SNR 20

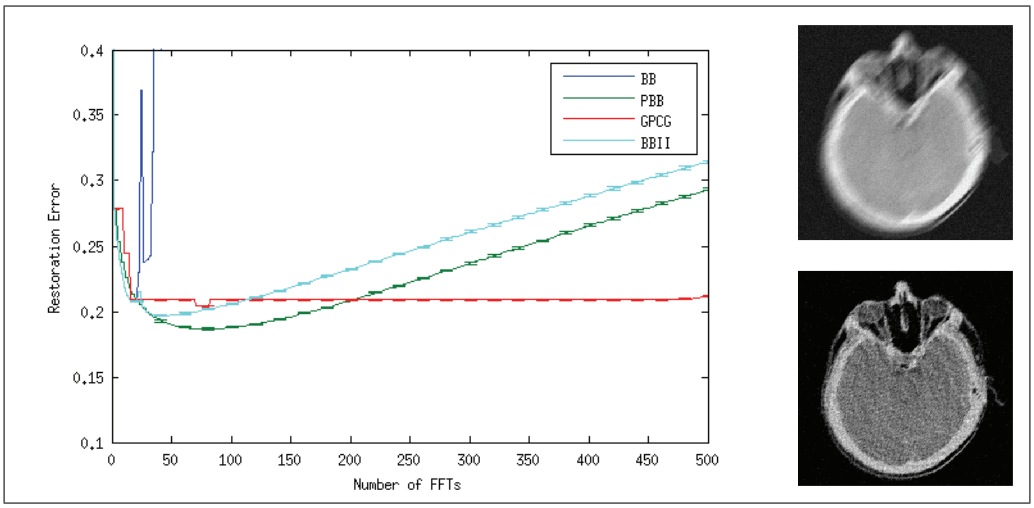


(b) SNR 30

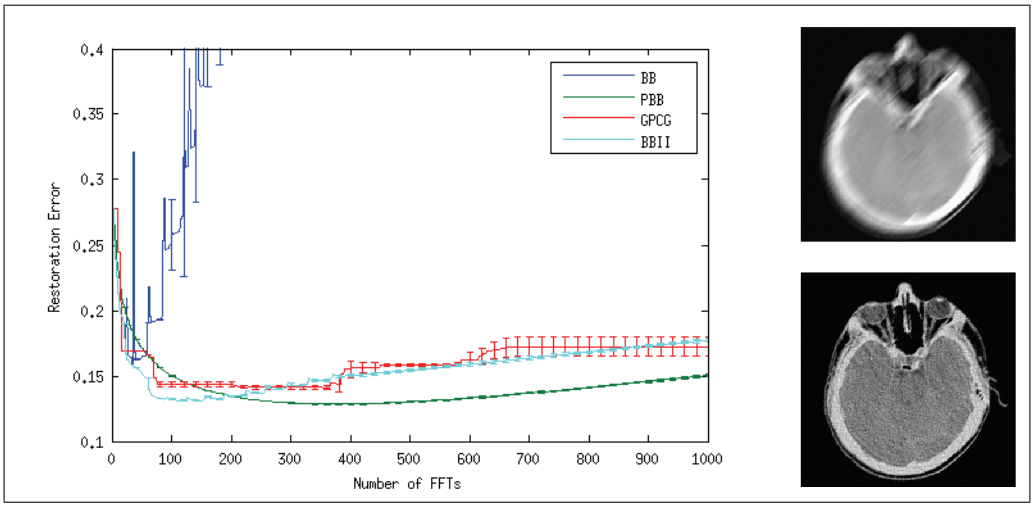


(c) SNR 40

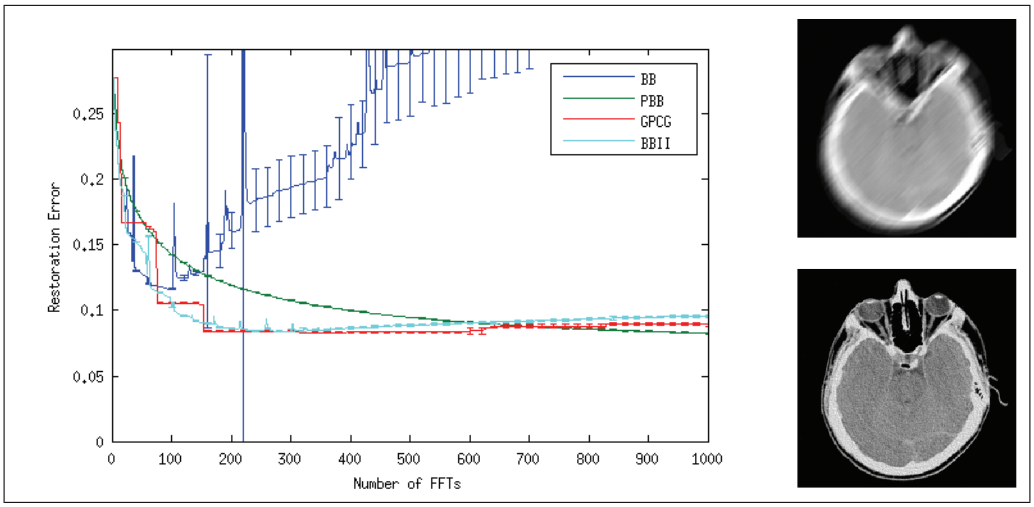
Figure 4.4: Head test image with PSF2



(a) SNR 20

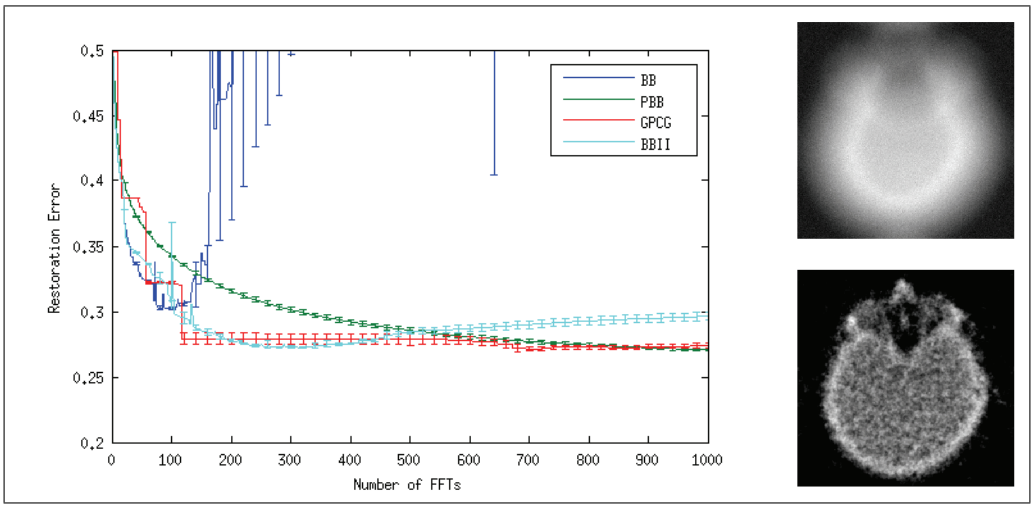


(b) SNR 30

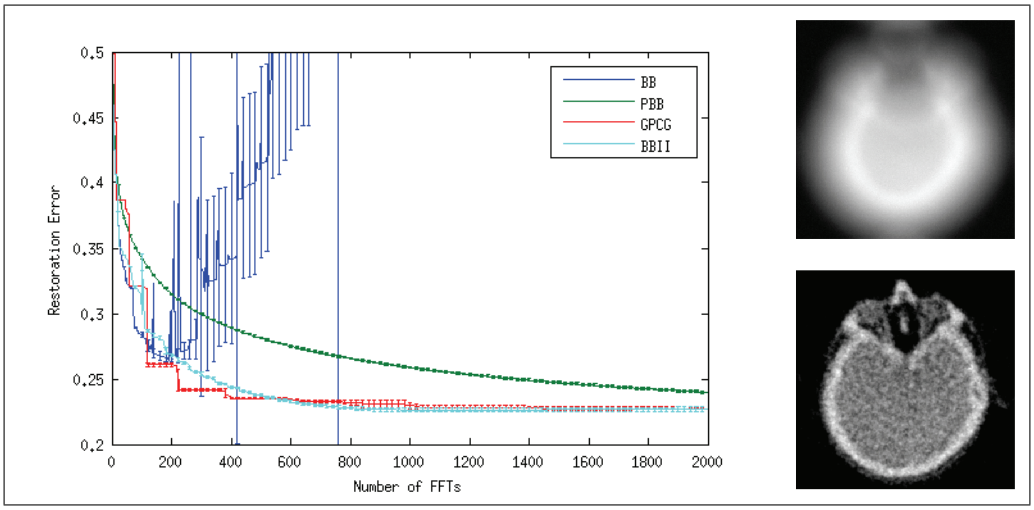


(c) SNR 40

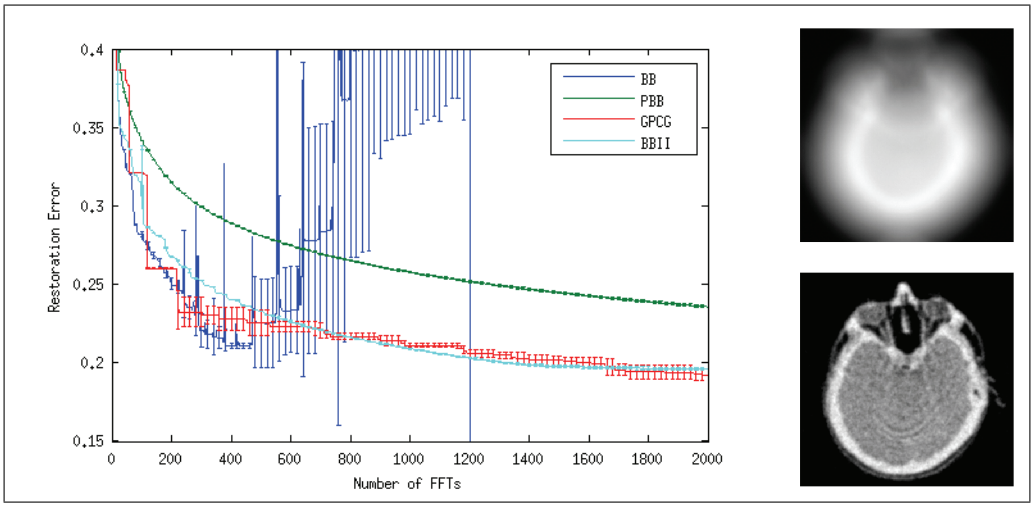
Figure 4.5: Head test image with PSF3



(a) SNR 20

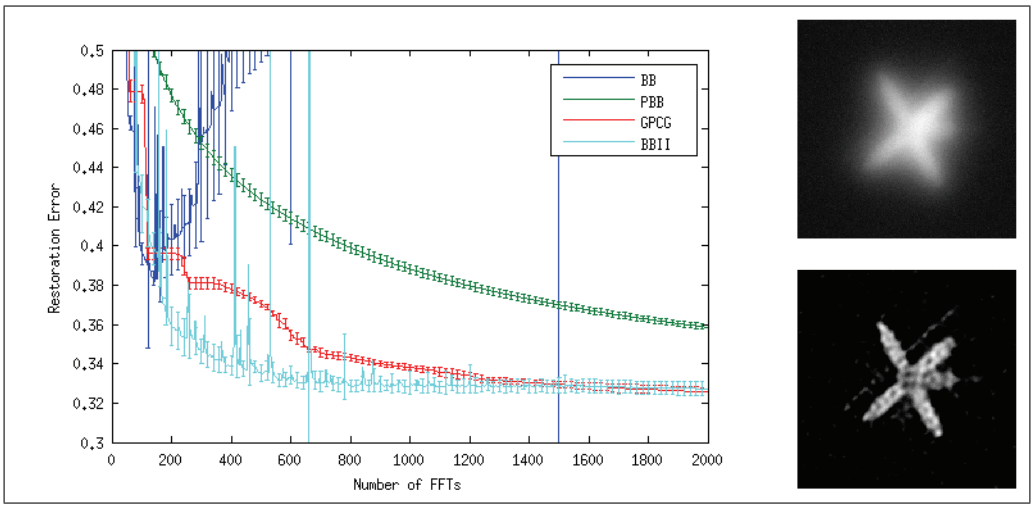


(b) SNR 30

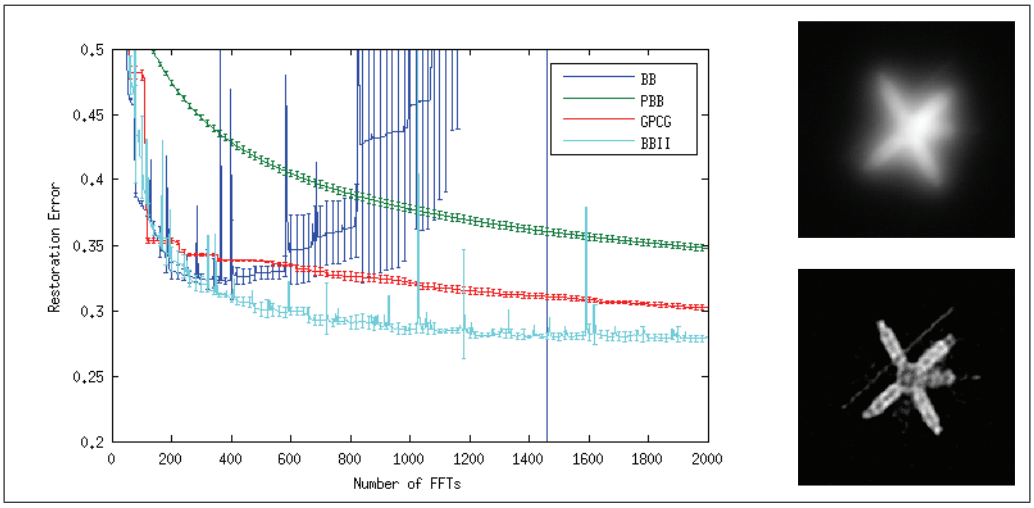


(c) SNR 40

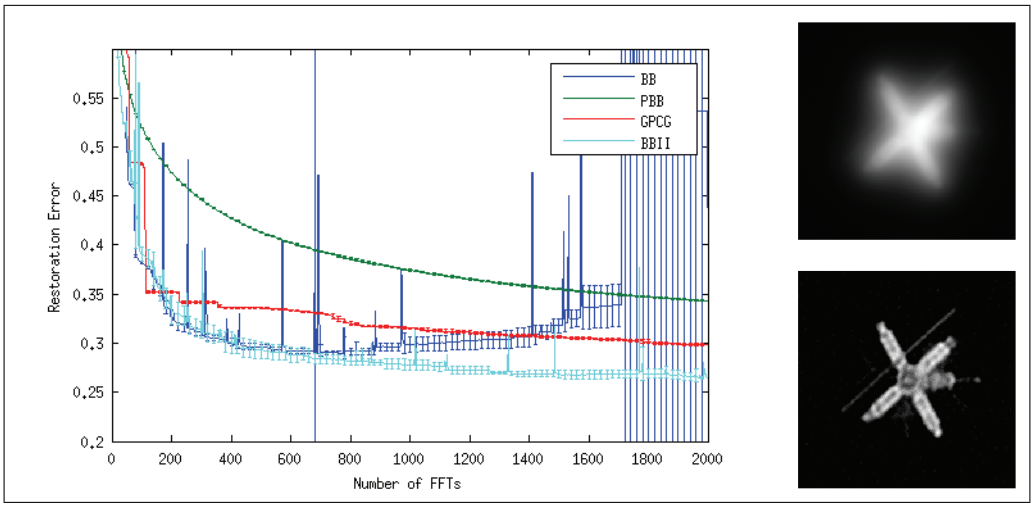
Figure 4.6: Head test image with PSF4



(a) SNR 20

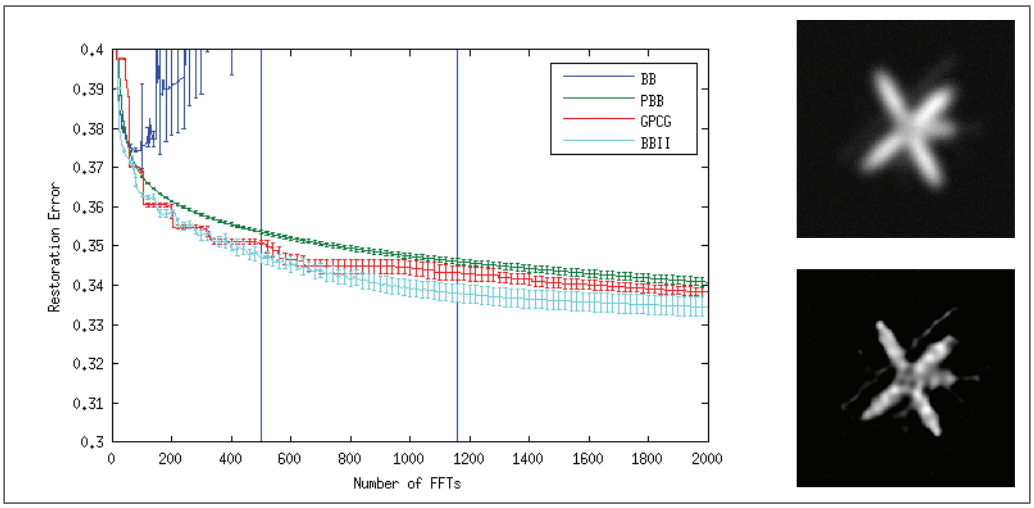


(b) SNR 30

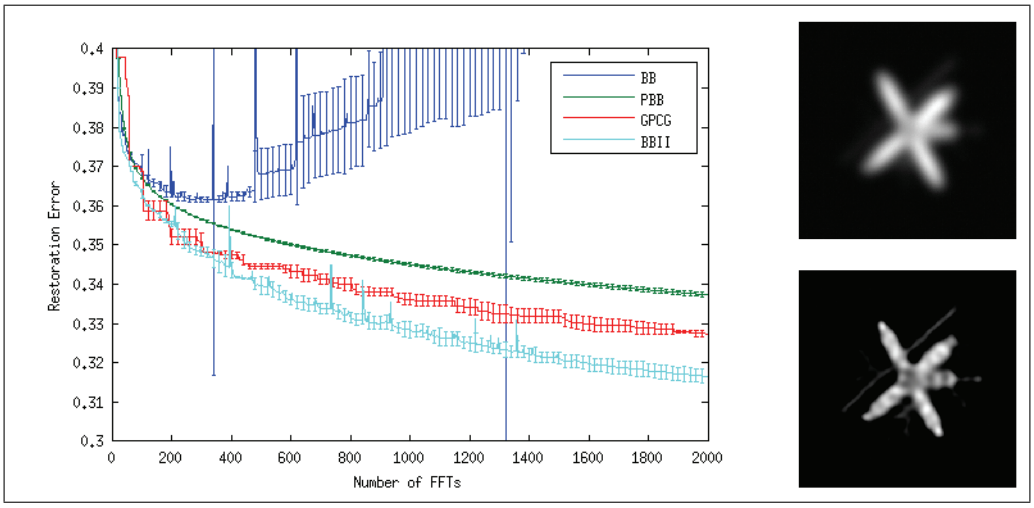


(c) SNR 40

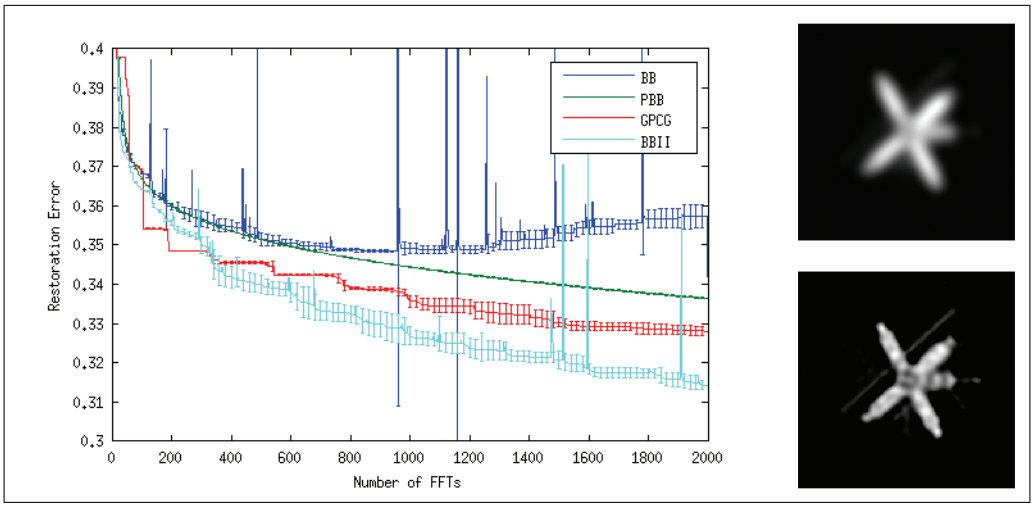
Figure 4.7: Satellite test image with PSF1



(a) SNR 20

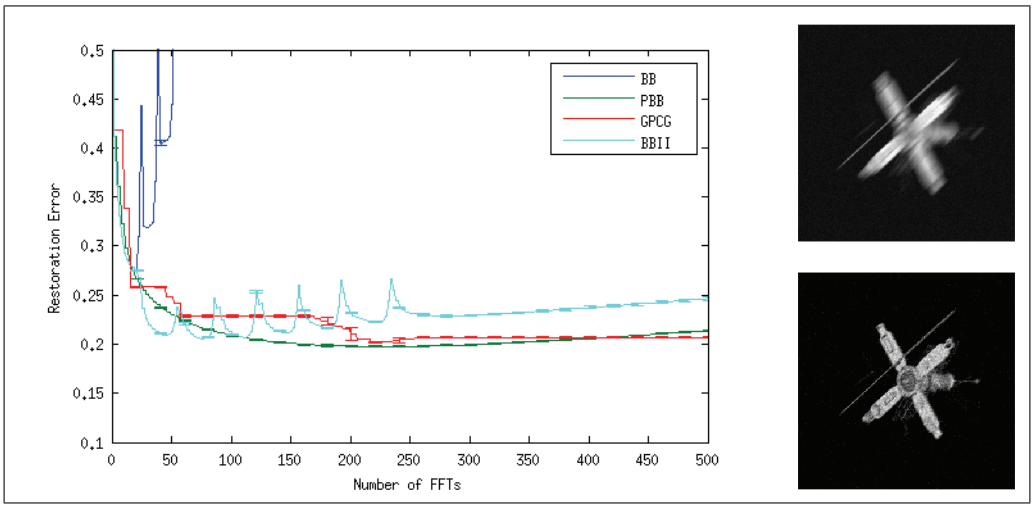


(b) SNR 30

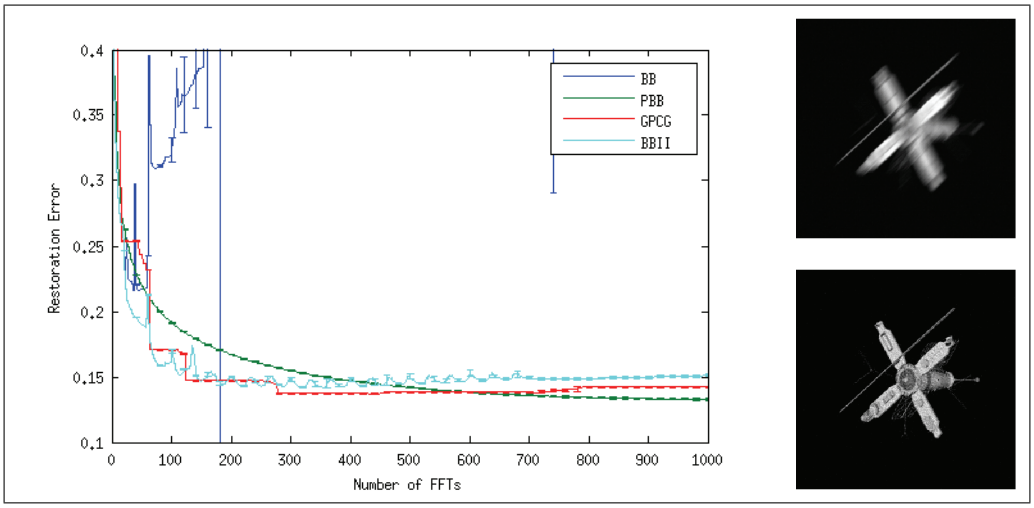


(c) SNR 40

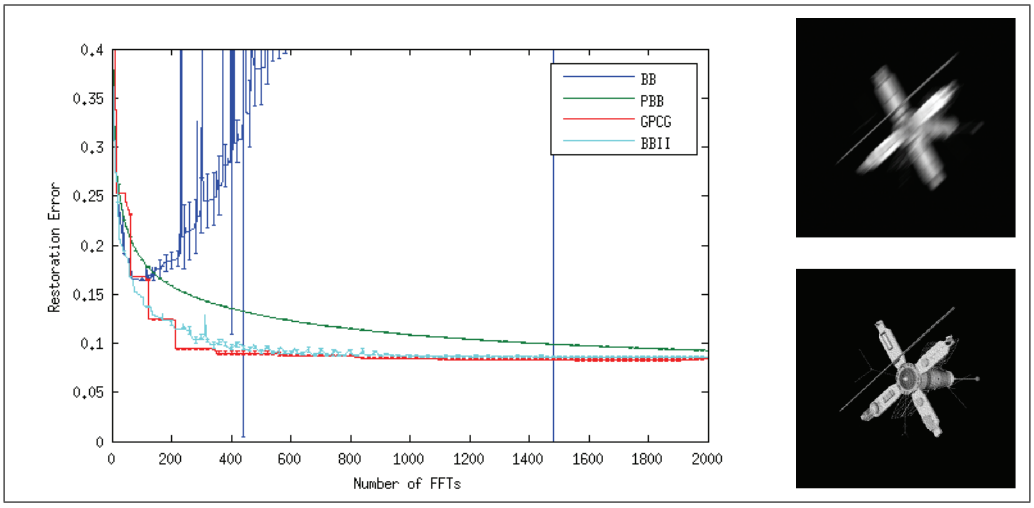
Figure 4.8: Satellite test image with PSF2



(a) SNR 20

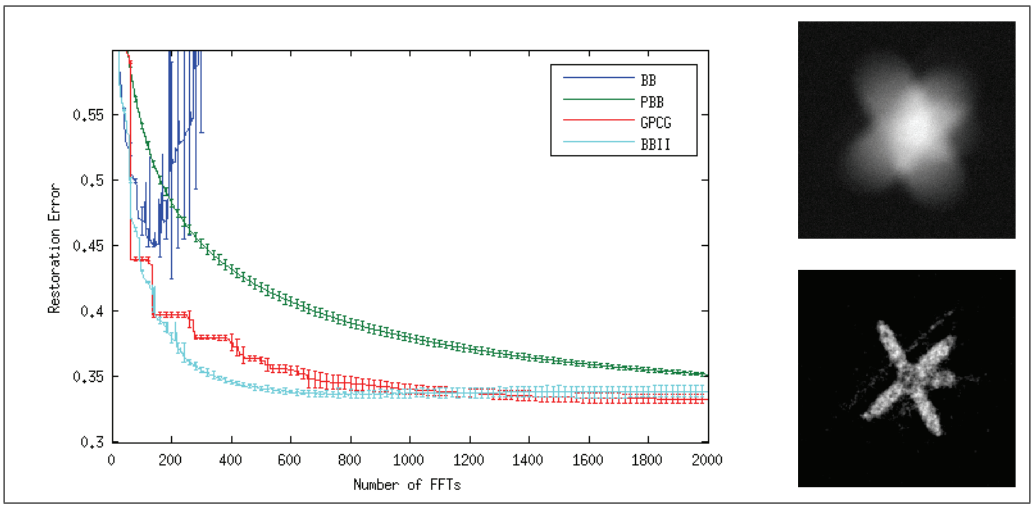


(b) SNR 30

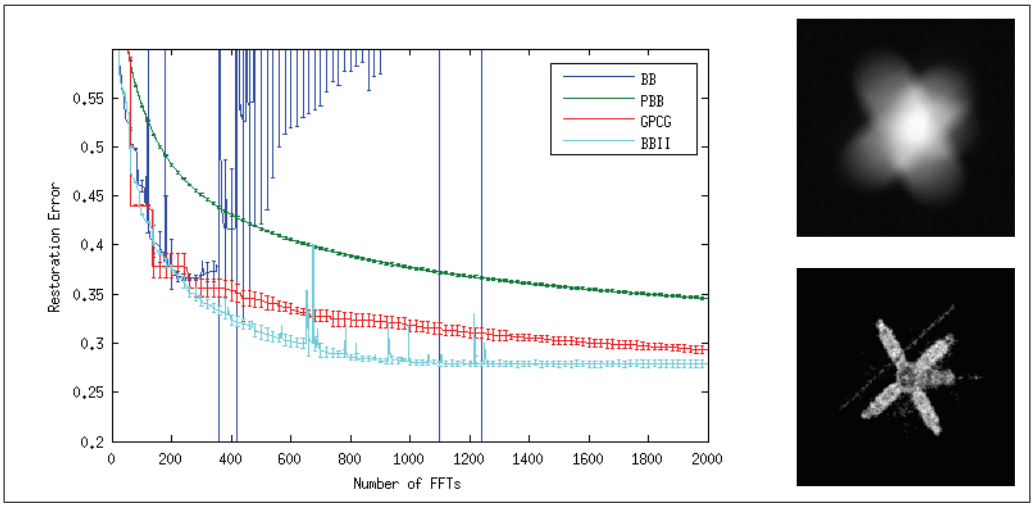


(c) SNR 40

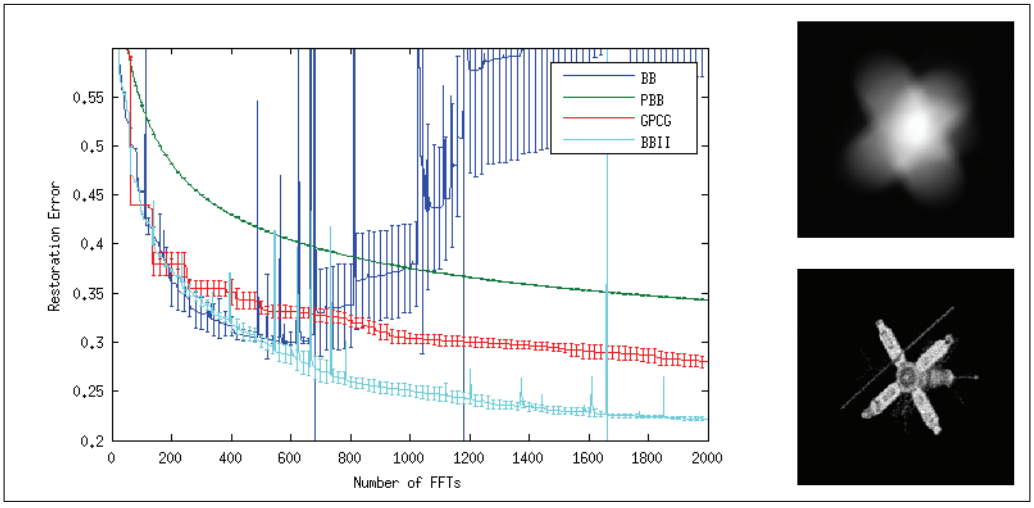
Figure 4.9: Satellite test image with PSF3



(a) SNR 20

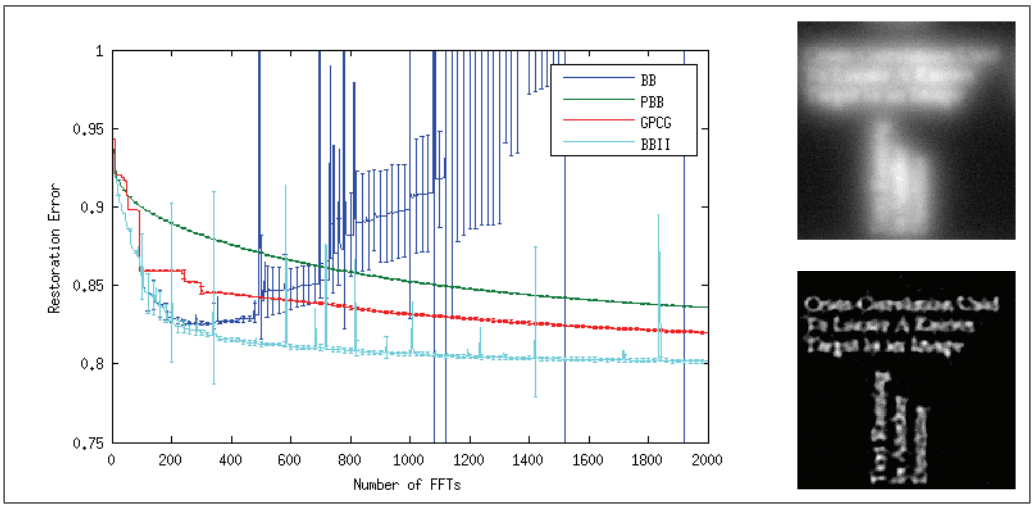


(b) SNR 30

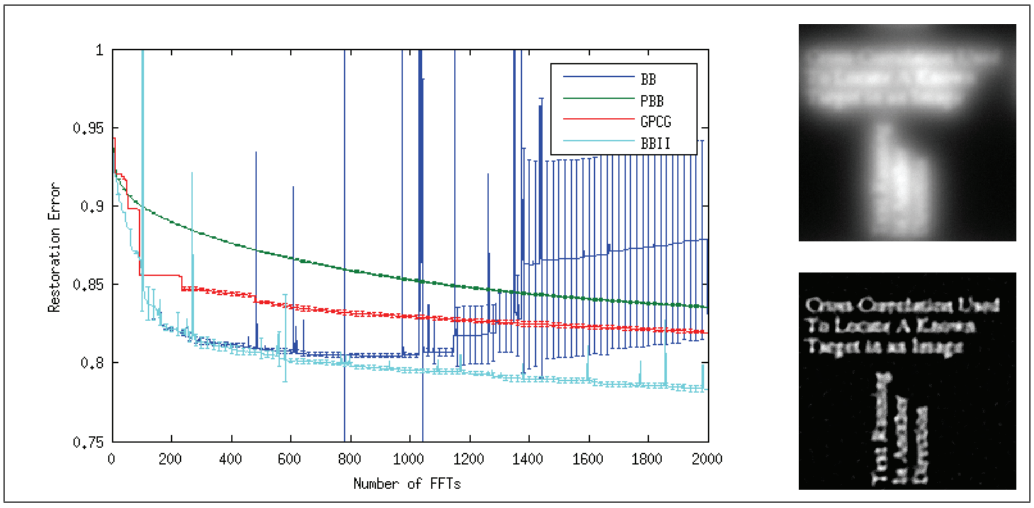


(c) SNR 40

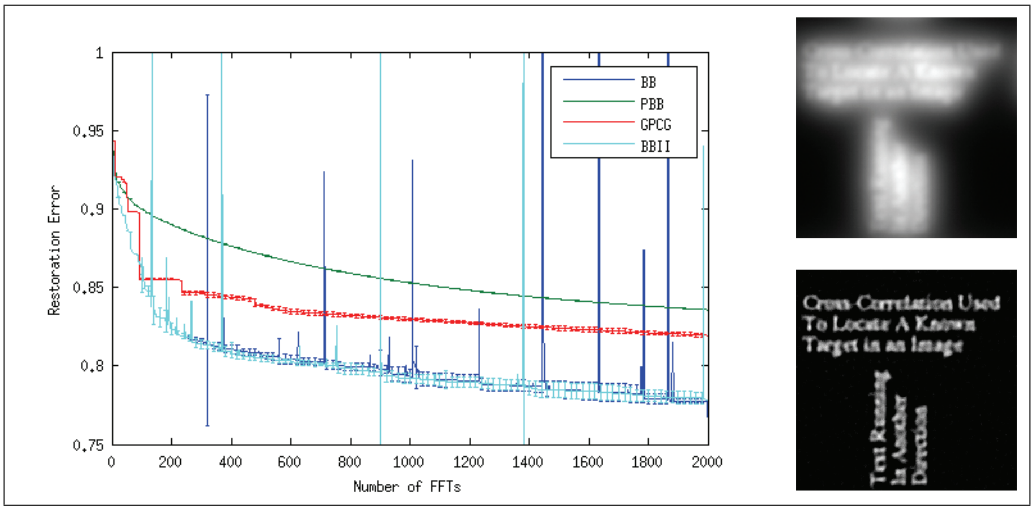
Figure 4.10: Satellite test image with PSF4



(a) SNR 20

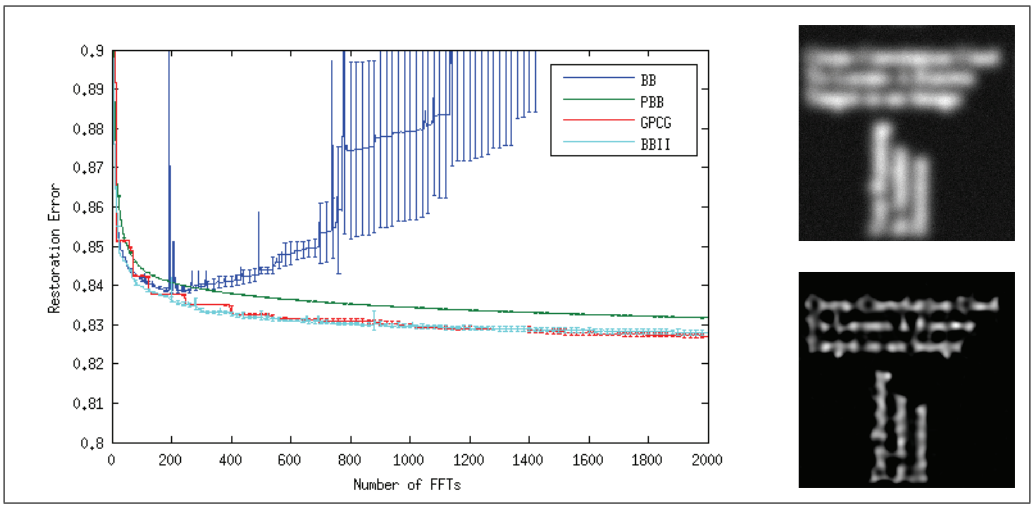


(b) SNR 30

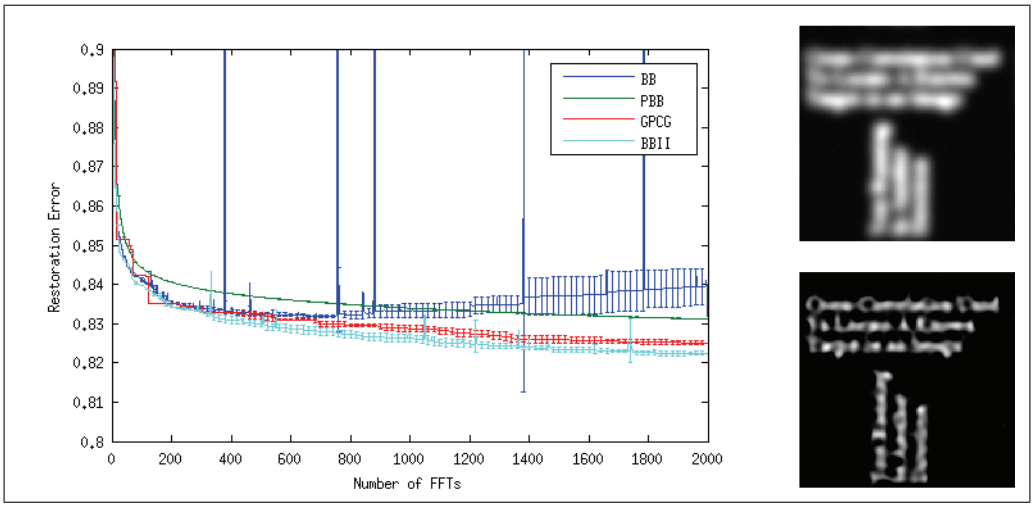


(c) SNR 40

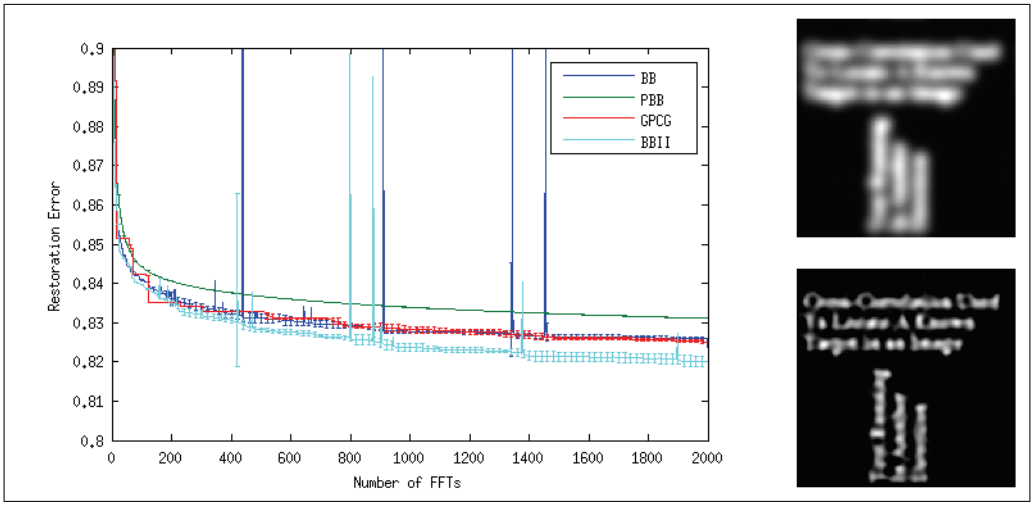
Figure 4.11: Text test image with PSF1



(a) SNR 20

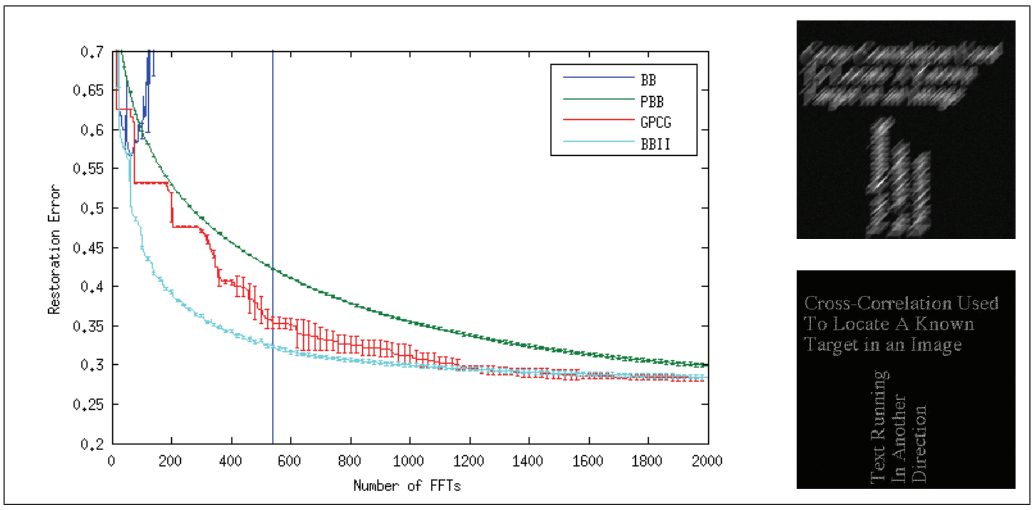


(b) SNR 30

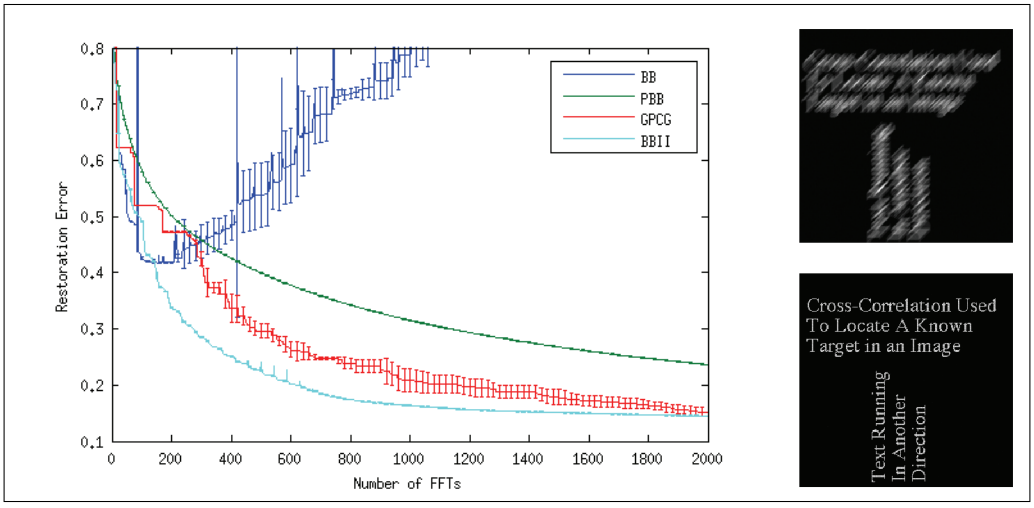


(c) SNR 40

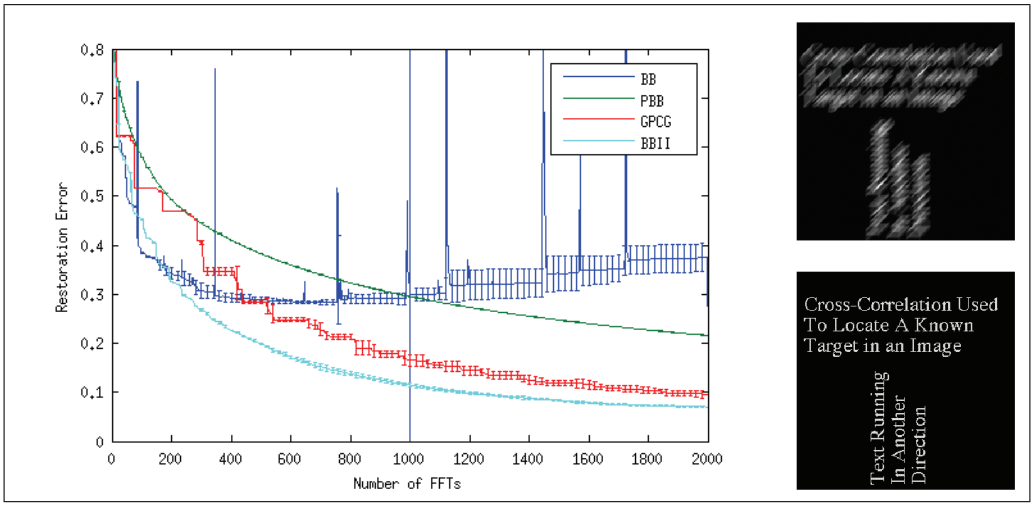
Figure 4.12: Text test image with PSF2



(a) SNR 20

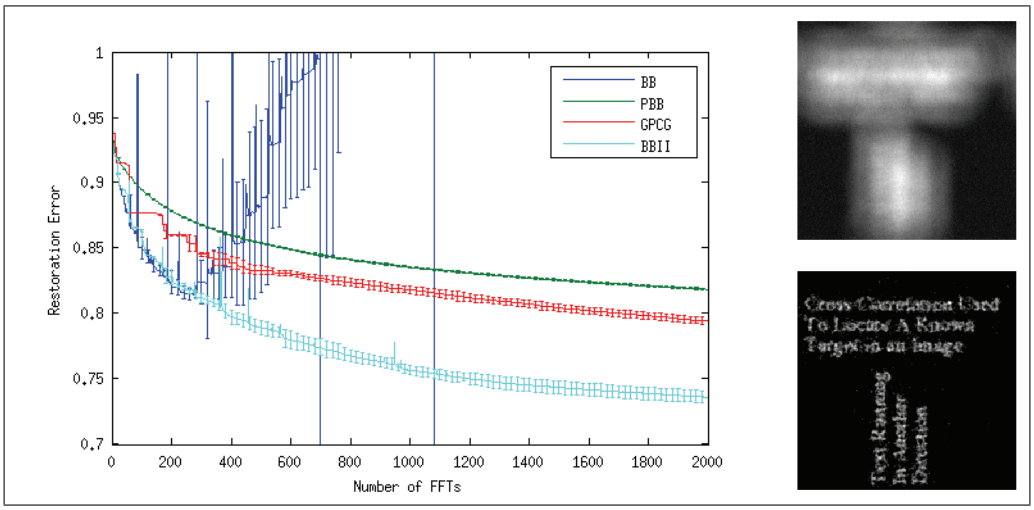


(b) SNR 30

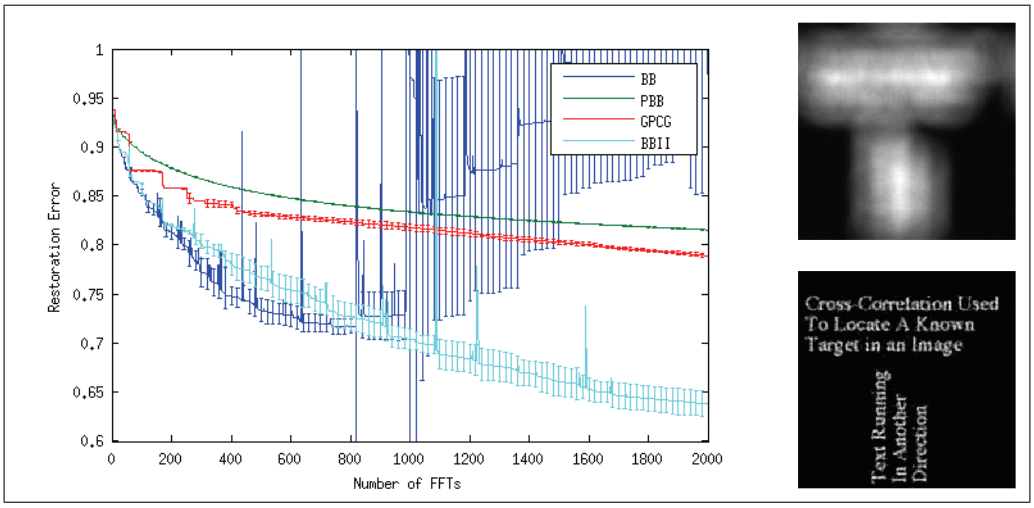


(c) SNR 40

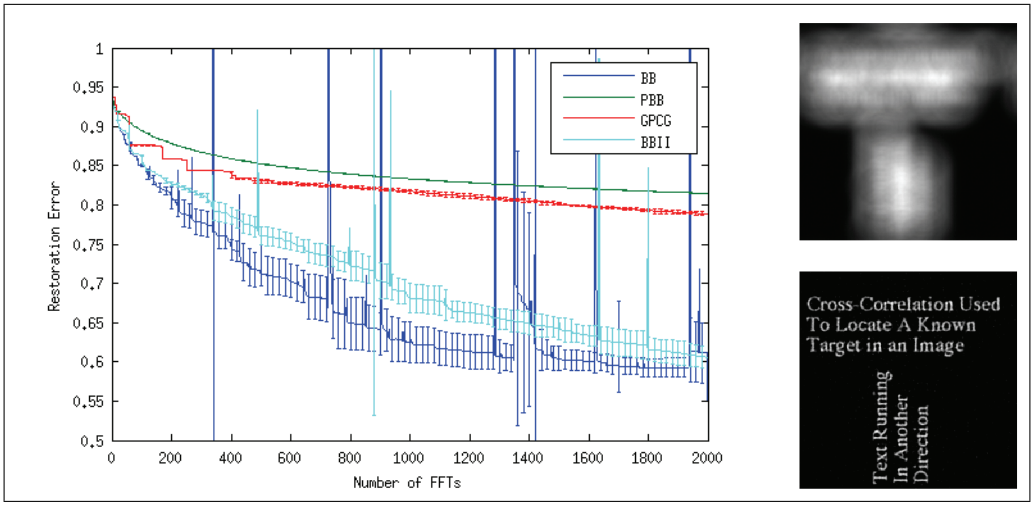
Figure 4.13: Text test image with PSF3



(a) SNR 20



(b) SNR 30



(c) SNR 40

Figure 4.14: Text test image with PSF4

	BB		PBB		GPCG		BBII	
	min	FFTs	min	FFTs	min	FFTs	min	FFTs
PSF1 BSNR 20	0.2713	138	**0.2567**	1895	0.2578	362	0.2578	342
PSF1 BSNR 30	0.2310	244	0.2378	1997	0.2210	1690	**0.2208**	1160
PSF1 BSNR 40	0.2083	620	0.2353	1997	0.2028	1968	**0.1985**	1999
PSF2 BSNR 20	0.2504	88	**0.2392**	1427	0.2398	718	0.2413	447
PSF2 BSNR 30	0.2395	228	0.2310	1997	**0.2268**	1998	0.2281	1999
PSF2 BSNR 40	0.2318	600	0.2298	1997	**0.2224**	1994	0.2228	1999
PSF3 BSNR 20	0.2075	20	**0.1866**	77	0.2034	76	0.1968	43
PSF3 BSNR 30	0.1593	34	**0.1285**	374	0.1410	360	0.1310	118
PSF3 BSNR 40	0.1163	102	**0.0802**	1529	0.0830	344	0.0840	298
PSF4 BSNR 20	0.3023	84	**0.2686**	1385	0.2714	718	0.2725	320
PSF4 BSNR 30	0.2643	190	0.2399	1997	0.2269	1994	**0.2263**	1097
PSF4 BSNR 40	0.2108	414	0.2358	1997	**0.1920**	1998	0.1960	1999

Table 4.1: Minimum mean error: head test case.

	BB		PBB		GPCG		BBII	
	min	FFTs	min	FFTs	min	FFTs	min	FFTs
PSF1 BNSR 20	0.3818	144	0.3587	1997	**0.3256**	1998	0.3273	1999
PSF1 BNSR 30	0.3225	376	0.3477	1997	0.3023	1998	**0.2784**	1946
PSF1 BNSR 40	0.2895	690	0.3431	1997	0.2983	1998	**0.2642**	1983
PSF2 BNSR 20	0.3742	96	0.3408	1997	0.3382	1998	**0.3345**	1999
PSF2 BNSR 30	0.3616	282	0.3373	1997	0.3274	1998	**0.3165**	1993
PSF2 BNSR 40	0.3420	1998	0.3364	1997	0.3281	1998	**0.3142**	1999
PSF3 BNSR 20	0.2753	20	**0.1977**	227	0.2021	230	0.2056	75
PSF3 BNSR 30	0.2158	36	**0.1328**	1079	0.1368	444	0.1420	322
PSF3 BNSR 40	0.1646	100	0.0928	1997	**0.0830**	1580	0.0859	1581
PSF4 BNSR 20	0.4508	150	0.3519	1997	**0.3332**	1992	0.3361	842
PSF4 BNSR 30	0.3668	252	0.3459	1997	0.2934	1996	**0.2785**	1417
PSF4 BNSR 40	0.3000	624	0.3431	1997	0.2801	1998	**0.2219**	1988

Table 4.2: Minimum mean error: satellite test case.

4.1.5 Discussion

BBII tends to benefit from the fast decrease of BB in the first several iterations, but then continues to decrease after the BB error starts increasing again. A good example of this behaviour is seen in Fig. 4.7(b). It also differs from BB in the sense that the error curve is smoother, which means that finding a good stopping criterion is not as essential. For instance, consider Fig. 4.11(b). For BB, the choice between stopping

	BB		PBB		GPCG		BBII	
	min	FFTs	min	FFTs	min	FFTs	min	FFTs
PSF1 BNSR 20	0.8254	276	0.8359	1997	0.8200	1998	**0.8015**	1975
PSF1 BNSR 30	0.8045	972	0.8357	1997	0.8192	1998	**0.7830**	1999
PSF1 BNSR 40	**0.7677**	1998	0.8357	1997	0.8194	1998	0.7783	1999
PSF2 BNSR 20	0.8386	250	0.8317	1997	**0.8270**	1994	0.8280	1999
PSF2 BNSR 30	0.8318	752	0.8311	1997	0.8251	1996	**0.8224**	1915
PSF2 BNSR 40	0.8236	1998	0.8311	1997	0.8251	1998	**0.8201**	1998
PSF3 BNSR 20	0.5684	58	0.2991	1997	**0.2836**	1998	0.2845	1999
PSF3 BNSR 30	0.4178	166	0.2363	1997	0.1516	1998	**0.1447**	1999
PSF3 BNSR 40	0.2753	1998	0.2159	1997	0.0956	1980	**0.0705**	1999
PSF4 BNSR 20	0.8146	274	0.8186	1997	0.7944	1998	**0.7361**	1999
PSF4 BNSR 30	0.7168	816	0.8156	1997	0.7891	1996	**0.6384**	1999
PSF4 BNSR 40	**0.5519**	1998	0.8145	1997	0.7882	1998	0.6057	1999

Table 4.3: Minimum mean error: text test case.

at 1000 FFTs or 1100 FFTs could have a significant effect on the quality of the solution. The BBII curve tends not to exhibit this unstable behaviour. In addition, the BBII curve has smaller error bars, indicating that is less dependent on particular noise configurations than BB. Note as well that although the error bars are the same length in the upward and downward directions, we know from our experience with BB that the sudden spikes typically increase the error, rather than decrease it, as in Fig. 3.5. Additionally, the error is lower-bounded by zero.

BBII achieves results similar to GPCG in many cases (see Fig. 4.3, 4.4, 4.6, 4.9, and 4.12, for example). In a number of other cases it does significantly better, such as in Fig. 4.14.

Not surprisingly, a visual inspection of the resulting images suggests that the reconstruction results are better for higher signal-to-noise ratios. For instance, if we consider the head test image with PSF1 and BSNR 20 (Fig. 4.3(a)), the resulting image appears very noisy in the mostly flat grey area in the middle of the head, and there is very little detail visible around the nose and eyes. The case with BNSR 40, shown in Fig. 4.3(c), has fewer noise artifacts and more detail. Comparing across PSFs, the best results for the head test image are seen in the PSF3 case (Fig. 4.5), which blurs the image less than the other PSFs. Similar results are seen with the other test images.

Tables 4.1, 4.2, and 4.3 help to illustrate the trade-off between time and the quality

of the restored image. For example, in Table 4.1, for test case PSF3 and BSNR 40, PBB attained the minimum restoration error of 0.0802 in 1529 FFTs. However, BBII reached a minimum of 0.0840 in only 298 FFTs. In some applications, the difference of 0.0038 in the restoration error might be meaningful, but in time-sensitive applications, the improvement is negligible compared to the 1231 extra FFTs required to achieve it.

For the head test image (Fig. 4.1), the constrained methods outperform unconstrained BB, confirming that adding a nonnegativity constraint can improve the quality of the results. PBB achieves the minimum error 50% of the time, and GPCG and BBII each achieve the minimum 25% of the time. However, the results are very close for this test image, with the difference in error generally showing up in the second or even third decimal place. For every case in which all three constrained methods reach their minimum before the maximum number of FFTs is reached, BBII achieves its minimum error value in the least number of FFTs.

In the satellite test case (Fig. 4.2), BBII achieves the minimum error 58% of the time, GPCG 25% of the time, and PBB 17% of the time. There are only two cases in which PBB converges before the maximum number of FFTs are reached, which is a testament to its slow convergence.

In the text test case (Fig. 4.3), BBII has the minimum error value in 66% of cases, with GPCG and BB each doing best 17% of the time.

4.2 Three-Dimensional Microscope Image Restoration

We test the algorithms on three-dimensional data as well, since we are interested in fluorescence microscopy. Increasing the dimensionality does not change the code, but it does mean that the stored variables are larger and the operations take longer, which may present problems for some algorithms.

4.2.1 Test Cases

For the 3D test cases, first we consider the test data provided by the Biomedical Imaging Group at the École Polytechnique Fédérale de Lausanne in Switzerland, available at `http://bigwww.epfl.ch/deconvolution/?p=bars`. The data are synthetic test images of six hollow bars, blurred with a theoretical widefield microscope PSF and

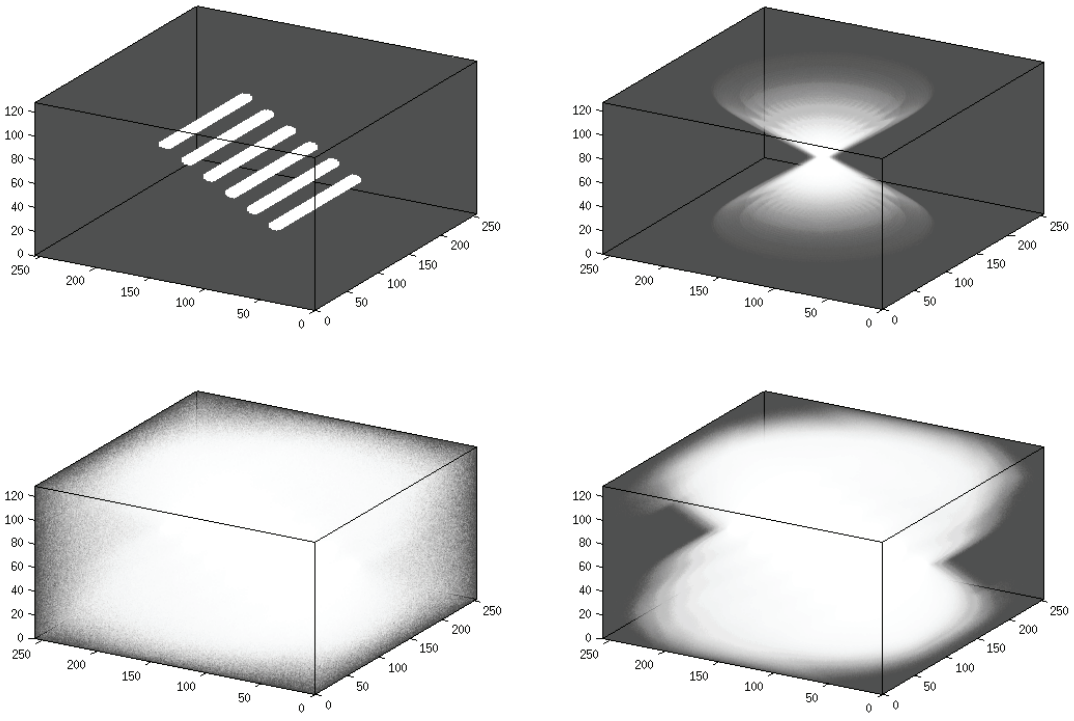


Figure 4.15: Bars test cases. On the top left, the true image. On the top right, the PSF. On the bottom left, the blurred image with SNR = 15. On the bottom right, the blurred image with SNR = 30. (Images have been scaled for visualization.)

corrupted with Gaussian and Poisson noise (see Fig. 4.15). There are two levels of noise, SNR = 15 dB and SNR = 30 dB. Although these images are synthetic, they are useful in making comparisons because the true image is also available, and thus we can compute the restoration error.

We then test the algorithm on real widefield microscope data. The images were acquired in wide-field mode on an Axio Observer Z1 microscope (Zeiss, Jena, Germany) equipped with a 63X Plan-Aprochromat lens, a solid-state light engine (Lumencor, Beaverton USA) and a CoolSNAP HQ2 charge-couple device (CCD) camera (Photometrics, Tucscon, USA) using SlideBook 5.0 software (Intelligent Imaging Innovations (3i), Boulder, USA). The test image consists of two fixed human HeLa cervical cancer cells that were first immunolabelled with sheep anti-PRP4 kinase antibodies [9] to identify the subnuclear compartment known as *splicing speckles* domains [22]. These splicing speckles were then detected by indirect immunofluorescence using donkey anti-sheep antibodies labelled with the fluorophore Alexa 488 (Invitrogen) that fluoresces green when illuminated with 488 nm light. A series of micrographs

was then taken at a spacing of 200 nm in the z-axis to allow the generation of a three-dimensional image series.

4.2.2 Parameters

The bar images are of size $256 \times 256 \times 128$. We scale the PSF so that all its elements sum to one, and scale the true image to lie between zero and one.

To choose the initial threshold, we use the same heuristic described in Section 4.1.2. In these test cases, although the true image is still scaled to have a maximum value of one, the blurry images have a much lower maximum value than in the the 2D cases. For this reason we choose the average minimum value to be -0.001, resulting in inital thresholds of 0.041 in the SNR 15 case, and 0.038 in the SNR 30 case. We again use $\rho = 0.97$ and calculate the median over the previous 10 iterations.

The original image of the cells was of size $956 \times 984 \times 30$, which we cropped to $756 \times 784 \times 30$, and scaled to between zero and one. Since an experimental PSF was not available, we used the "PSF Generator" plug-in for ImageJ (available at `http://bigwww.epfl.ch/deconvolution/?p=plugins`) to generate a theoretical PSF. To perform the calculation, the program requires a number of parameters describing the microscope, which are given in Table 4.4. We use an initial threshold of 0.056, corresponding to an average minimum value of -0.05.

Numerical Aperture	1.4
Refractive Index	1.518
Spatial Resolution	102.383 nm
Axial Resolution	100 nm
Wavelength	488 nm (green)

Table 4.4: Microscope parameters for the real 3D test case.

4.2.3 Evaluation

For the parallel bars test case, we plot the restoration error against the number of FFTs. Note that in this case the FFTs are three-dimensional rather than two-dimensional, so the time required to run the algorithms increases. Additionally, the GPCG program consistently caused Matlab to run out of memory after only a few iterations. As a result, the GPCG curves are not included in the plots.

For the cell test case, we are unable to calculate the restoration error since there is no "true" image. Instead we simply show a slice from the middle of the image after deconvolution, and compare the results visually. For the sake of comparison, we also include the results which are output by the SlideBook software. We consider two of its deconvolution algorithms: Nearest Neighbours and Constrained Iterative. The Nearest Neighbour (NN) algorithm is based on the assumption that most of the out-of-focus light in a given slice comes from the slices directly on either side. Thus, it removes only the blur originating from the two neighbouring planes [24]. The Constrained Iterative (CI) algorithm is based on work by David Agard [1]. It uses a combination of Wiener filtering and a modified van Cittert update [36] to remove blur, and Gaussian filtering to reduce noise. It also imposes a nonnegativity constraint.

The NN and CI algorithms use a PSF which is generated internally by the software and is inaccessible to the user. For this reason, the comparison between our methods and the software methods is not precise as they do not use the same PSF. However, it is useful to compare our results with those that would be typically generated and used in a real laboratory setting.

4.2.4 Discussion

The restoration error curves for the bars test cases, shown in Fig. 4.16 and Fig. 4.17, look similar to those in 2D. The PBB method leads to a slow, smooth descent, while the BB method leads to a fast initial descent but does not achieve a very low error value before increasing again. The BBII method initially behaves similarly to BB, but then smoothes out and achieves the lowest error value of the three methods after 2000 FFTs. The resulting images shown in Fig. 4.16 and Fig. 4.17 indicate that the images produced by PBB are blurrier than those produced by BBII.

For the cell test image, the analysis is more subjective. Each image is displayed such that its lowest pixel value displays as black, and the highest pixel value displays as white. Fig. 4.18 shows a comparison of BB, PBB, and BBII, while Fig. 4.19 shows the comparison with NN and CI.

In Fig. 4.18(b), the BB result overfits the noise. The speckles are blurry and the whole image is covered with a mottled pattern. The PBB and BBII methods lead to fairly similar results, but the BBII image shows a greater reduction in the amount of

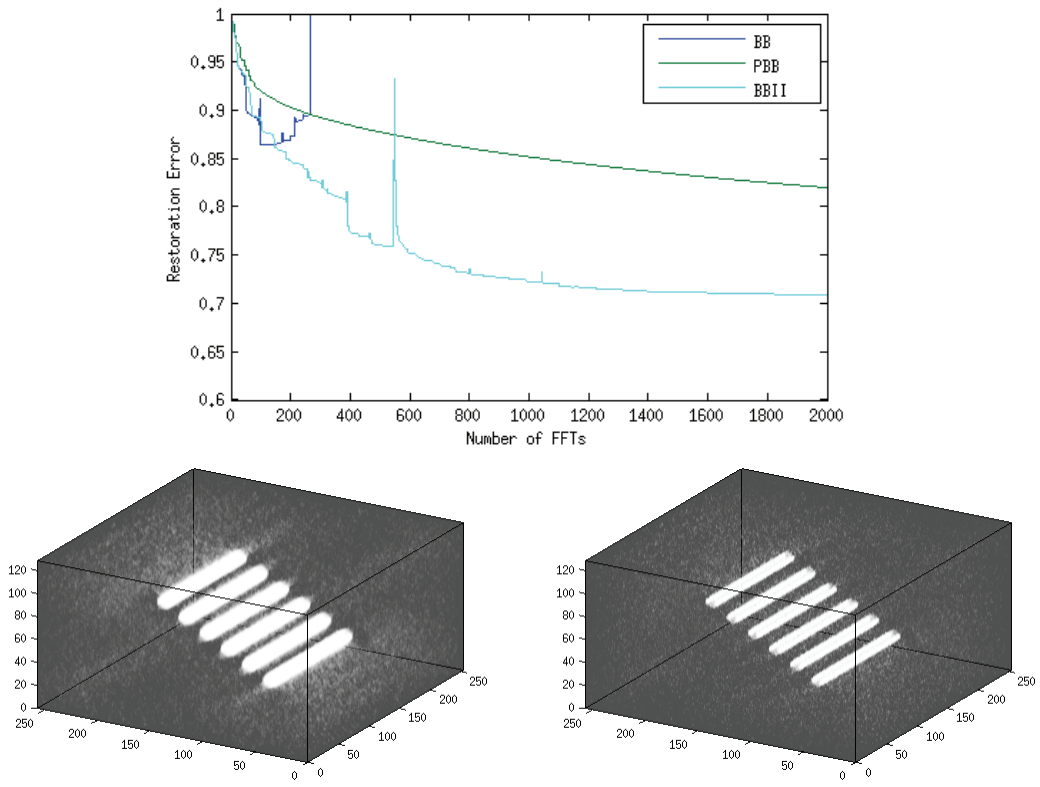


Figure 4.16: Results for 3D bars test case with SNR = 15. On top, the restoration error vs. the number of FFTs. Bottom left, the result after 2000 FFTs for PBB. Bottom right, the result after 2000 FFTs for BBII.

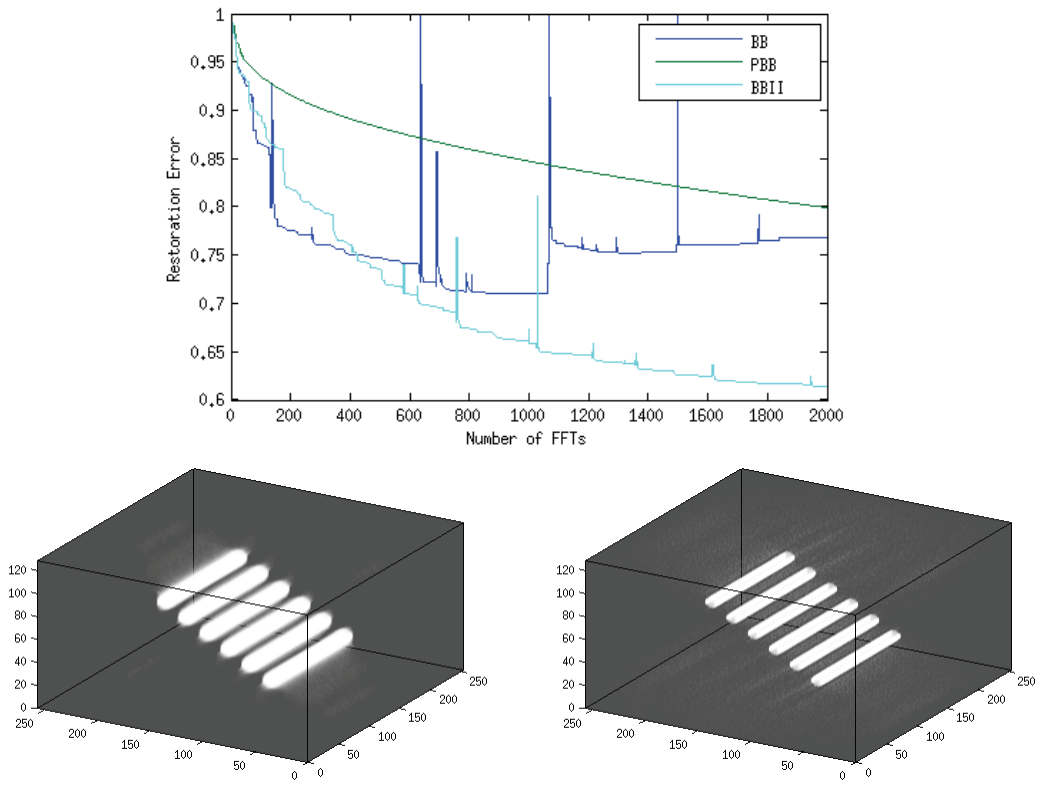
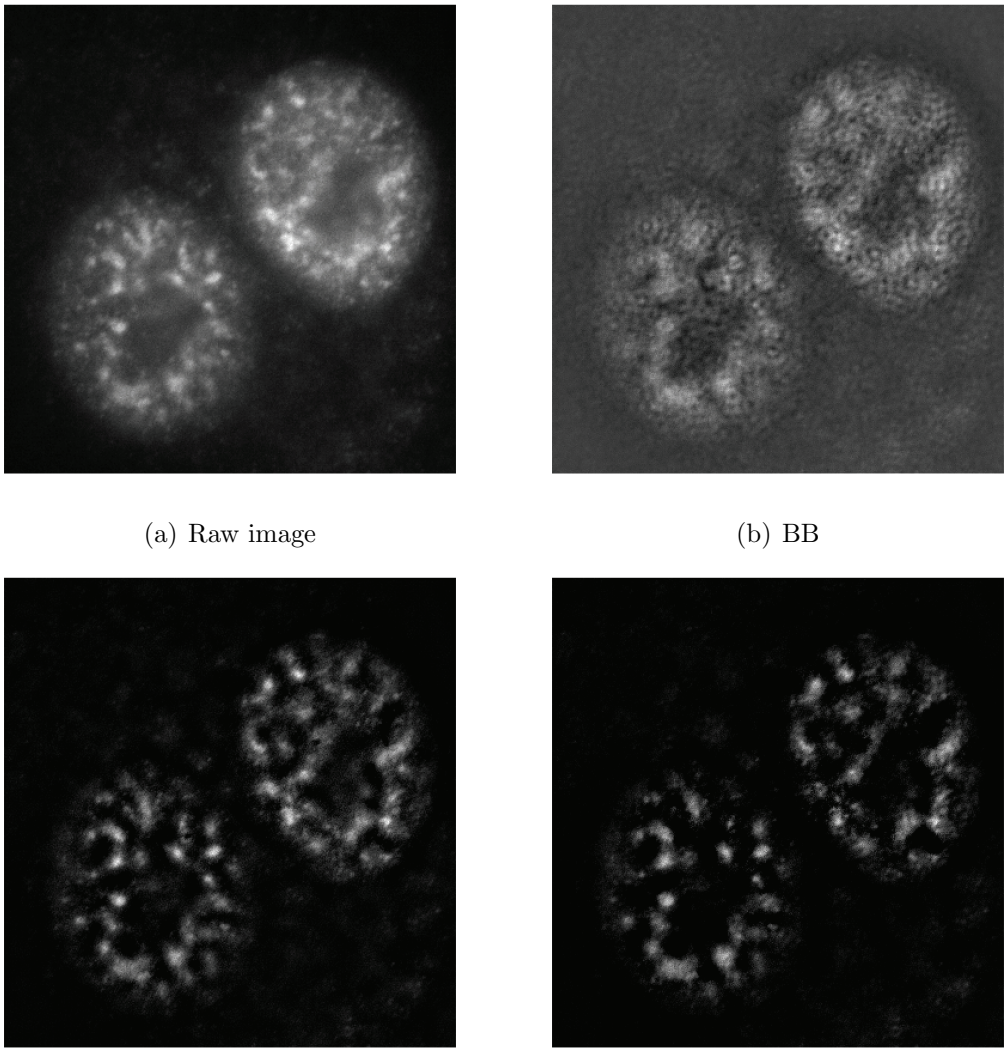


Figure 4.17: Results for 3D bars test case with SNR = 30. On top, the restoration error vs. the number of FFTs. Bottom left, the result after 2000 FFTs for PBB. Bottom right, the result after 2000 FFTs for BBII.



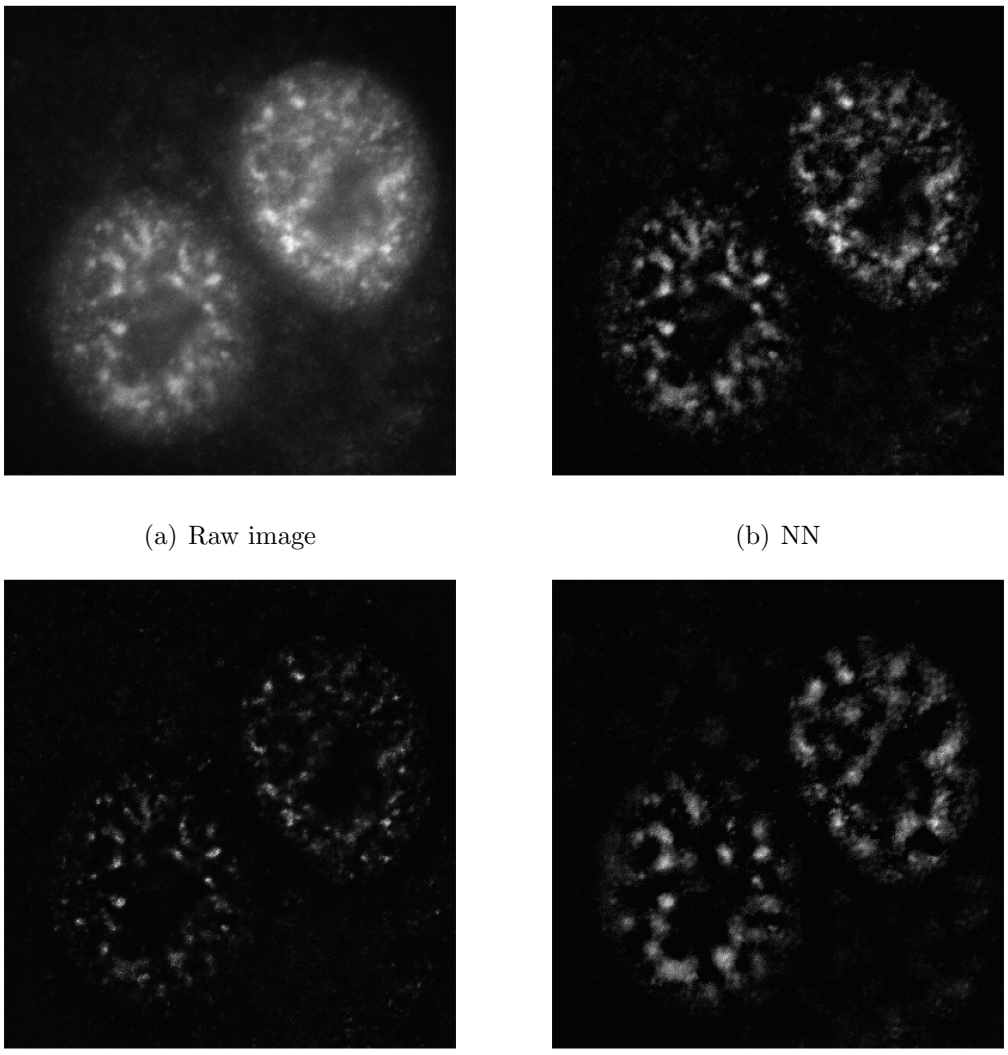
(a) Raw image

(b) BB

(c) PBB

(d) BBII

Figure 4.18: Results for the 3D cell image, using BB, PBB, and BBII, after 400 FFTs.



(a) Raw image

(b) NN

(c) CI

(d) BBII

Figure 4.19: Results for the 3D cell image, compared to those produced by the SlideBook software normally used with the microscope.

background haze inside the cells.

In Fig. 4.19(b) and Fig. 4.19(d), BBII appears to remove more of the background haze than NN, which is to be expected since it considers light contributions from outside the nearest planes. However the speckles in the BBII result are still blurrier than those in the CI image (Fig. 4.19(c)).

One difference between the cell test case and all the previous test cases is that it is the only case in which the exact blurring function is not known. The theoretical PSF which we used is only an estimate of the real PSF. For instance, the top and bottom "cones" of a measured PSF are typically not exactly symmetric due to spherical aberration [24]. Additionally, a theoretical PSF does not model the blur which results from light reflecting off the cover slip. However, it is the nature of experimental work that the PSF will never be known exactly. We have shown here that it is still possible to achieve a good result using BBII in conjunction with a simple and free PSF generator.

Chapter 5

Conclusion

Image deconvolution is difficult due to the ill-posed nature of the problem; however it has many important applications and is an active area of research in a number of different fields. We have presented an overview of the basic aspects of the problem and different approaches to its solution. Direct methods, such as inverse filtering and Wiener filtering, are simple and fast but do not yield good results in all cases. Iterative methods, such as steepest descent and conjugate gradient, generate a sequence of estimates before converging on the inverse solution. By stopping the iterations prematurely, we can achieve a good solution which does not overfit the noise. Another iterative method is the Barzilai-Borwein method, which is non-monotonic and uses a wide range of step-lengths, and has shown remarkably fast convergence given its simplicity.

Because pixel values represent light intensities, they cannot be negative. This suggests that it can be useful to incorporate a nonnegativity constraint into a deconvolution method. The gradient projection algorithm is a simple algorithm for constrained optimization, but it can be slow to converge. Better performance can be achieved by combining the benefits of GP and CG into one method, called GPCG. Additionally, the BB method has been adapted to solve the constrained problem by projecting the BB iterates onto the feasible set.

We have proposed a new algorithm which is based on the BB method but imposes a nonnegativity constraint. However, in contrast to the existing PBB method, we do not project on every iteration. Instead, we allow infeasible iterates with pixel values of less than zero. This has the effect of reducing the number of FFTs required per iteration, and often allows the fast convergence which is characteristic of the unconstrained BB method. To impose the nonnegativity constraint, we keep track of the ratio of the average squared value of negative pixels to the average squared value of all the pixels. When this quantity increases past a pre-defined threshold,

we project the solution onto the feasible set. Over time, we tighten the constraint by decreasing the threshold. As a result, we see that the restoration error decreases more quickly than PBB over the first several iterations, but then does not tend to increase again as rapidly as with the BB method.

Our method represents a significant improvement over the existing PBB method. Based on a comparison between the minimum value of the average error curves, BBII achieved a lower restoration error than PBB in 78% of the 2D test cases, and both of the 3D test cases for which the restoration error could be calculated. It also showed faster convergence.

We also compared BBII to GPCG, which was designed for solving large-scale bound-constrained quadratic programming problems. When compared to GPCG, BBII performed surpisingly well, given that it is a much simpler algorithm and does not use the conjugate gradient method, which is well-known for its performance on unconstrained quadratic problems. BBII achieved a minimum error less than or equal to GPCG in 65% of the 2D test cases. Furthermore, BBII did not cause Matlab to run out of memory in the 3D case, as the GPCG implementation did.

In general the constrained methods did better than the unconstrained BB method on these test cases, which all contain large dark areas, supporting the claim that nonnegative approaches are to be preferred when faced with images of this type.

We have mentioned in Sec. 4.1.2 a heuristic method for choosing the initial threshold τ; future work could explore different methods for choosing τ as well as the scaling factor ρ. We experimented briefly with choosing a different window over which to calculate the median and found that the differences were negligible; however an in-depth study might yield more information.

We have applied BBII to image restoration, focusing particularly on its applications to fluorescence microscopy. It would be interesting to explore its application to real astronomical images, since that is another area in which nonnegatively constrained deconvolution is commonly used. Another possibility would be to incorporate an upper bound as well as a lower bound. An upper bound is not imposed by the physics of the situation, as the lower bound is, but it might be applicable in some situations. For instance, we know that the maximum value of our 2D test images is one, so we could impose the constraint that all solution pixel values lie between zero

and one.

Another possible extension would be to tackle the problem of blind deconvolution, which is used when either the PSF is unknown, or only a rough estimate is known. In real-life applications, such as the 3D microscopy example, the exact PSF is rarely known. Iterative blind deconvolution algorithms update the estimate of the PSF as well as the estimate of the true image at each iteration. If it was possible to incorporate into BBII some method of improving an estimate of the PSF, the algorithm might be useful in a wider range of applications.

Looking ahead in a broader context, this type of algorithm can be applied to any problem in quadratic programming. Applications of quadratic programming can be found in science and engineering, as well as fields such as management, economics, and finance. However, it must be noted that nowhere in our results do we consider the objective function itself; instead we are concerned with maximizing the quality of the restored image, in this case as measured by the restoration error. Further investigation would be required to determine how BBII compares to other quadratic programming methods in terms of minimizing the objective function.

It should be clear by now that aspects of the BB method are not well understood. However, we have shown that by allowing infeasible iterates rather than projecting on every step, we can preserve and benefit from some of the BB behaviour while still constraining the solution to the feasible domain.

Bibliography

[1] David A. Agard, Yasushi Hiraoka, Peter Shaw, and John W. Sedat. Fluorescence microscopy in three dimensions. *Methods in Cell Biology*, 30:353–377, 1989.

[2] Charles Audet and John E. Dennis Jr. A progressive barrier for derivative-free nonlinear programming. *SIAM Journal of Optimization*, 20:445–472, 2009.

[3] Mark R. Banham and Aggelos K. Katsaggelos. Digital image restoration. *IEEE Signal Processing Magazine*, pages 24–41, 1997.

[4] Johnathan M. Bardsley. *Constrained Optimization Techniques for Image Reconstruction*. PhD thesis, Montana State University, 2002.

[5] Johnathan M. Bardsley, Jorma K. Merikoski, and Roberto Vio. The stabilizing properties of nonnegativity constraints in least-squares image reconstruction. *International Journal of Pure and Applied Mathematics*, 42:71–81, 2008.

[6] Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.

[7] Mario Bertero and Patrizia Boccacci. *Introduction to Inverse Problems in Imaging*. Institute of Physics Publishing, 1998.

[8] Yu-Hong Dai and Roger Fletcher. Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. *Numerische Mathematik*, 100:21–47, 2005.

[9] Graham Dellaire, Evgeny M. Makarov, Jeff J. Cowger, Dasa Longman, Heidi G. Sutherland, Reinhard L̀uhrmann, Joseph Torchia, and Wendy A. Bickmore. Mammalian PRP4 kinase copurifies and interacts with components of both the U5 snRNP and the N-CoR deacetylase complexes. *Molecular and Cellular Biology*, 14:5141–5156, 2002.

[10] Ahmet M. Eskicioglu and Paul S. Fisher. Image quality measures and their performance. *IEEE Transactions on Communications*, 43:2959–2965, 1995.

[11] Paola Favati, Grazia Lotti, Ornella Menchi, and Francesco Romani. Performance analysis of maximum likelihood methods for regularization problems with nonnegativity constraints. *Inverse Problems*, 26:1–18, 2010.

[12] Roger Fletcher. On the Barzilai-Borwein method. *Applied Optimization*, 96:235–256, 2005.

[13] Ana Friedlander, José M. Martínez, and Marcos Raydan. Gradient method with retards and generalizations. *SIAM Journal on Numerical Analysis*, 36:275–289, 1999.

[14] Nikolas P. Galatsanos and Aggelos K. Katsaggelos. Methods for choosing the regularization parameter and estimating the noise variance in image restoration and their relation. *IEEE Transactions on Image Processing*, 1:322–336, 1992.

[15] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Pearson Prentice Hall, third edition, 2008.

[16] Nicholas I. M. Gould and Philippe L. Toint. Nonlinear programming without a penalty function or a filter. *Mathematical Programming Series A*, 122:155–196, 2010.

[17] Martin Hanke, James G. Nagy, and Curtis Vogel. Quasi-Newton approach to nonnegative image restorations. *Linear Algebra and its Applications*, 316:223–236, 2000.

[18] Per Christian Hansen. *Discrete Inverse Problems: Insight and Algorithms*. SIAM, 2010.

[19] Per Christian Hansen, James G. Nagy, and Dianne P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM, 2006.

[20] Per Christian Hansen and Dianne P. O'Leary. The use of the L-curve in the regularization of discrete ill-posed problems. *SIAM Journal on Scientific Computing*, 14:1487–1503, 1993.

[21] Hui Huang and Uri Ascher. Faster gradient descent and the efficient recovery of images. Preprint (2011), available at `http://www.cs.ubc.ca/~ascher/publications.html`.

[22] Angus I. Lamond and William C. Earnshaw. Structure and function in the nucleus. *Science*, 280:547–553, 1998.

[23] Jennifer Lippincott-Schwartz and George H. Patterson. Development and use of fluorescent protein markers in living cells. *Science*, 300:87–91, 2003.

[24] James G. McNally, Tatiana Karpova, John Cooper, and José A. Conchello. Three-dimensional imaging by deconvolution microscopy. *Methods*, 19:373–385, 1999.

[25] Jorge J. Moré and Gerardo Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1:93–113, 1991.

[26] James Nagy and Zdenek Strakos. Enforcing nonnegativity in image reconstruction algorithms. In David C. Wilson et al., editor, *SPIE Conference on Mathematical Modeling, Estimation, and Imaging*, pages 182–190, 2000.

[27] James G. Nagy and Katrina M. Palmer. Steepest descent, CG, and iterative regularization of ill-posed problems. *BIT Numerical Mathematics*, 43:1003–1017, 2003.

[28] James G. Nagy, Katrina M. Palmer, and Lisa Perrone. Iterative methods for image deblurring: a Matlab object-oriented approach. *Numerical Algorithms*, 36:73–93, 2004.

[29] Soontorn Oraintara, William C. Karl, David A. Castanon, and Truong Q. Nguyen. A method for choosing the regularization parameter in generalized Tikhonov regularized linear inverse problems. In *International Conference on Image Processing 2000, Proceedings*, pages 10–13, 2000.

[30] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition, 2002.

[31] Marcos Raydan. On the Barzilai and Borwein choice of steplength for the gradient method. *IMA Journal of Numerical Analysis*, 13:321–326, 1993.

[32] Marcos Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal on Optimization*, 7:26–33, 1997.

[33] William H. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62:55–59, 1972.

[34] Miguel A. Rodríguez-Toral, William Morton, and David R.Mitchell. The use of SQP methods for the optimization of utility systems. *Computers and Chemical Engineering*, 25:287–300, 2001.

[35] Pinaki Sarder and Arye Neorai. Deconvolution methods for 3D flouresence microscopy images. *IEEE Signal Processing Magazine*, 23:32–45, 2006.

[36] Jean-Baptiste Sibarita. Deconvolution microscopy. *Advances in Biochemical Engineering/Biotechnology*, 95:201–243, 2005.

[37] Andrei N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. V. H. Winston and Sons, 1977.

[38] Kees van den Doel and Uri Ascher. The chaotic nature of faster gradient descent methods. Preprint (2010), available at `www.cs.ubc.ca/~ascher/papers/doas1.pdf`.

[39] Peter J. Verveer, Mark Gemkow, and Thomas M. Jovin. A comparison of image restoration approaches applied to three-dimensional confocal and wide-field fluorescence microscopy. *Journal of Microscopy*, 193:50–61, 1999.

[40] Curtis R. Vogel. *Computational Methods for Inverse Problems*. SIAM, 2002.

[41] Cédric Vonesch. *Fast and Automated Wavelet-Regularized Image Restoration in Fluorescence Microscopy*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2009.

[42] Yanfei Wang and Shiqian Ma. Projected Barzilai-Borwein method for large-scale nonnegative image restoration. *Inverse Problems in Science and Engineering*, 15:559–583, 2007.

[43] Zhou Wang, Alan C. Bovik, and Ligang Lu. Why is image quality assessment so difficult? In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3313–3316, 2002.

[44] Bin Zhou, Li Gao, and Yu-Hong Dai. Gradient methods with adaptive step sizes. *Computational Optimization and Applications*, 35:69–86, 2006.

Appendix A

Matlab Code

```
function [f,alpha,proj]=bbii(PSF,b,iter,tau,rho)

% Barzilai-Borwein with Infeasible Iterates
% Kathleen Fraser, July 2011
%
% Input:
%           PSF - point spread function
%           b - blurry, noisy image
%           iter - number of iterations
%           tau - initial threshold (tau < 0)
%           rho - scaling factor (0 <= rho <= 1)
%
% Output:
%           f - deconvolved image
%
% Optional output:
%           alpha - step sizes
%           proj - record of projections (1 or 0 for projection or no projection)
%


f=zeros(size(b));
alpha=zeros(1,iter+1);
proj=zeros(1,iter);

K=psf2otf(PSF,size(b));
F=fftn(f);
B=fftn(b);

G = conj(K).*(K.*F-B);
g = real(ifftn(G));
Kg = real(ifftn(K.*G));
alpha(1) = (g(:)'*g(:))/(Kg(:)'*Kg(:));

for k=1:iter,

    if(k>10 && proj(k-1) ~= 0)                  % projection on last step
```

```
        alpha(k)=(g(:)'*g(:))/(Kg(:)'*Kg(:));   % use steepest descent step size
    end

    f=f-alpha(k)*g;                             % update solution
    less=f<0;                                   % number of elements < zero

    if(sum(less(:))>0)
        neg=(f(:).*less(:));
        r(k)=(sum(neg(:).^2)/sum(less(:)))/(sum(f(:).^2)/numel(b));
    else
        r(k)=0;
    end

    alpha(k+1)=(g(:)'*g(:))/(Kg(:)'*Kg(:));     % step size for next iteration

    if(k<=10)                                   % finding the median r value
        n(k)=r(k);
    else
        n(1:9)=n(2:10);
        n(10)=r(k);
        med(k)=median(n);
    end

    if(k>10 && med(k)>tau)                      % project if median(r) > tau
        f=max(f,0);
        num=3;
        proj(k)=1;
        tau=rho*tau;
    else
        num=2;
        proj(k)=0;
    end

    if(num==3)                                  % if projection, use gradient definition
       F=fftn(f);                               % requires an addition FFT
       G = conj(K).*(K.*F-B);
    else                                        % otherwise use iterative update
        G=G-alpha(k)*(conj(K).*K.*G);
    end

    g = real(ifftn(G));
    Kg=real(ifftn(K.*G));

    if(k==iter)                                 % always project on last iteration
        f=max(f,0);                             % so output solution is feasible
    end

end
```