**Distribution Agreement**

Lu Lin 04/8/2016

Nonnegative and Volume Constrained Image Deblurring

By

Lu Lin

James Nagy

Adviser

Department of Mathematics and Computer Science

James Nagy

Adviser

John Duncan

Committee Member

Christina DePasquale

Committee Member

2016

# Nonnegative and Volume Constrained Image Deblurring

By

Lu Lin

James Nagy

Adviser

Abstract of
A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Sciences with Honors

Department of Mathematics and Computer Science

2016

# Abstract

Nonnegative and Volume Constrained Image Deblurring
By Lu Lin

In this thesis, we discuss iterative algorithms that can be used for constrained image deblurring. We mainly focus on the gradient projection method, which combines gradient descent with projections that implement constraints, such as nonnegative and volume constraints. Numerical experiments on three test problems using MATLAB illustrate the effctiveness and the efficiency of these methods.

# Nonnegative and Volume Constrained Image Deblurring

By

Lu Lin

James Nagy

Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelors of Sciences with Honors

Department of Mathematics and Computer Science

2016

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Background on Inverse Problems

An inverse problem is the process of calculating the unknown causes from a set of observations. It takes the general form

$$b = G(x)$$

where $b$ is the observed data, $x$ is the unknown we wish to determine, and G is an operator describing the relationship between $b$ and $x$.
In this thesis, we will discuss image deblurring problems, which can be generallly considered as linear inverse problems. We wish to reconstruct a good approximation of the true image from the typically blurred and noisy measurements:

$$g = Kf_{true} + \eta$$

where $g$ is the measured blurred image, $f_{true}$ is the true image, $\eta$ is additive noise, and $K$ is a matrix that serves as the mapping operator generated from the point spread function (PSF). We will have further discussion of image deblurring problems in section 1.3.

There are mainly two computational challenges for image deblurring:

- The problem is usually large-scale: If the images have $m \times n$ pixels (i.e. $g, f \in R^{mn}$), then the dimension of matrix $K$ would be $mn \times mn$ (i.e. $K \in R^{mn \times mn}$).
- The matrix $K$ is extremely ill-conditioned. This implies that regularization and additional constraints are needed to compute good approximations of $f_{true}$.

## 1.2 Iterative Methods

Iterative methods have many advantages for solving image deblurring problems. They can be very efficient for getting solutions, and they can easily incorporate additional constraints on the solutions, such as nonnegative and volume constraints [1, 9]. The basic idea of iterative methods is to generate a sequence of approximate solutions until it converges to a sufficiently good approximation.

Since image deblurring problems are usually ill-conditioned, we focus on iterative methods that can be applied to solve the following least squares problem.

$$\min_f \|Kf - g\|_2^2$$

We notice that if we define the quadratic function $\psi(f) = \frac{1}{2} f^T K^T K f - f^T K^T g$, then the following two minimization problems are equivalent:

$$\min_f \psi(f) \quad \text{and} \quad \min_f \|Kf - g\|_2^2.$$

Hence, our deblurring problem can be transformed into solving: $\min\limits_f \psi(f)$, which can be easily implemented by using iterative methods. Here we have the general form of the iterative methods:

$$f^{(k+1)} = f^{(k)} + \tau_k d^{(k)}$$

which means in the $k^{th}$ step of the iteration, we approach the solution in the direction of $d^{(k)}$ by a step size $\tau_k$.

The general form of an iterative algorithm is represented in the following box.

```
$f_0$ = initial guess of $f_{true}$
for $k = 0, 1, 2, ...$
    • $f_{k+1} = f_k + \tau_k d_k$
    • update $d_k$, $\tau_k$ and other intermediate quantities involved in the computations
    • determine if the stoping criteria are satisfied
end
```

## 1.3 Image Deblurring

In the image processing application, the data is recorded digitally by the intensity value at only finite and discrete points on the image, named pixels. This recording process is modeled as a linear system:

$$g(x, y) = \int_{R^2} k(x, s; y, t) f(s, t) ds dt + \eta(x, y)$$

where, $g(x, y)$ is the observed image, $f(s, t)$ is the original image, $\eta(x, y)$ is the additive noise, and the kernel function $k$ represents the blurring phenomena, which is called the *point spread function* (PSF). Typically, $k$ is estimated from observations and varies from case to case. If the kernel function satisfies $k(x, s; y, t) = k(x - s, y - t)$, then the blur is called spatially invariant, otherwise the blur is called spatially variant. If we partition the image into several small regions, and for each region there is a single PSF that can be used to represent the blurring opertation, then the blur is called locally spatially invariant.

The integration in the model is called a two-dimensional *Fredholm integral equation of the first kind* (IFK) in mathematical literature. Since it is discretely recorded, by discretizing the IFK and approximating the integration with a quadrature rule, we obtain the matrix-vector equation that we discussed in section 1.1

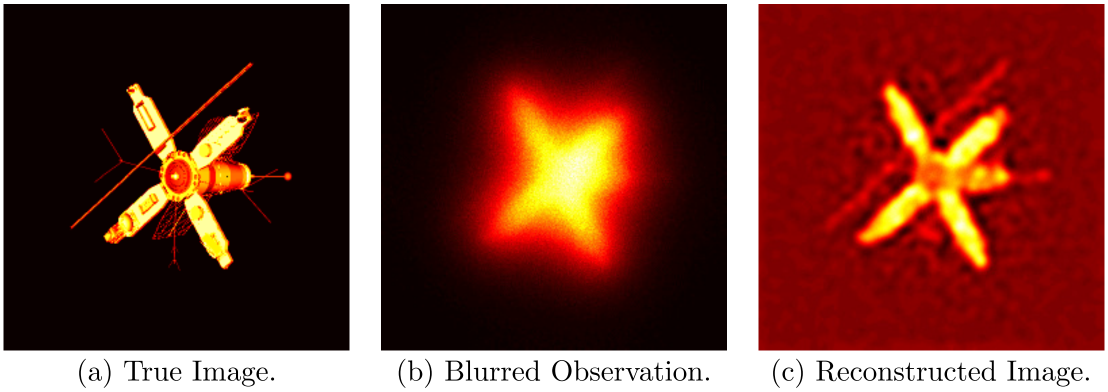
$$g = Kf_{true} + \eta.$$

More details about inverse problems in imaging can be found in [6, 3].

Our goal for image deblurring is to compute an approximated solution $f$ to this matrix-vector equation. For example in Figure 1.1, the left picture is the true image of a satellite, which is $f_{true}$ in the matrix-vector equation; the middle picture is the blurred observation, which is $g$; the right picture is the reconstructed image using 43 iterations of the iterative method *conjugate gradient method for least squares problems* (CGLS), which is $f$ that we computed. In the general cases of image deblurring, the true image is usually unknown and only a blurred observation is given. Our expectation is to get a good image reconstruction from this blurred and noisy observation with the given point spread function.

In the general form of the unconstrained image deblurring problem, the task can be expressed as follows:

$$\min_f \|Kf - g\|_2 \ .$$



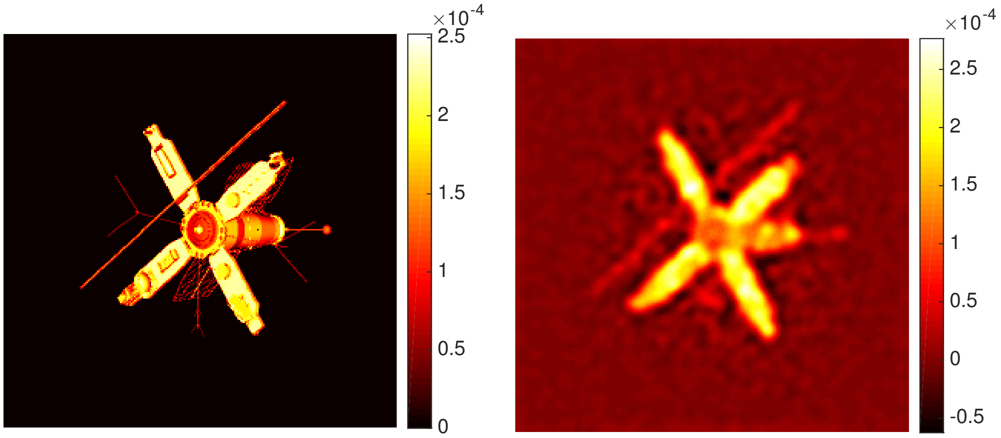
(a) True Image. (b) Blurred Observation. (c) Reconstructed Image.

Figure 1.1: The left is the true image, which is usually unknown in image deblurring problems; the middle picture is the given observation, which is extremely blurry and noisy; the right picture is our reconstructed image after 43 iterations of the CGLS method.

But this may be too simple for a real life application. We may need to introduce additional constraints in some problems in order to improve the accuracy of the solution or to follow the laws of physics. Here are two additional constraints that we believe are valuable and practical:

- **Nonnegative Constraint**: The data of the image is recored by the pixel intensity values, which are in a range of nonnegative values. Thus, the true image should always be nonnegative. However, because of additional noise, nonnegativity is not always satisfied in the observed image, as well as in the reconstructed image based upon it. As represented in Figure 1.2, the true image in the left is nonnegative everywhere, but the reconstruction at 43th iteration, using CGLS, has some negative values. Hence, we include a nonnegative constraint in the iterative algorithm.

- **Volume Constraint**: Intuitively, the total volume of the data is not affected by the blurring operation. It is natural to preserve the volume of data, and to enforce an additional volume constraint in the iterative algorithm.



(a) True Image.

(b) Reconstructed Image.

Figure 1.2: The true image in the left is nonnegative everywhere, but the reconstructed image has some negative values in it. Here the reconstructed image is computed at iteration 43 using the CGLS method.

In addition, we also need to incorporate an appropriate boundary condition for deblurring problems, since the edges of the blurred image may be affected by information outside the field of view. For example, in Figure 1.3, the left is an astronomical image, which has zero boundary condition intuitively; but for the right image, we obviously need to apply the boundary condition other than zero, since there is information extending outside the left top corner.

In the following chapters, we will further discuss the unconstrained iterative methods, such as gradient descent method (GD) and conjugate gradient method for least squares (CGLS), for image deblurring problems. Based on the unconstrained gradient descent method, we introduce a nonnegative projection in chapter 2, and a nonnegative and volume projection in chapter 3 for solving more complicated constrained problems.

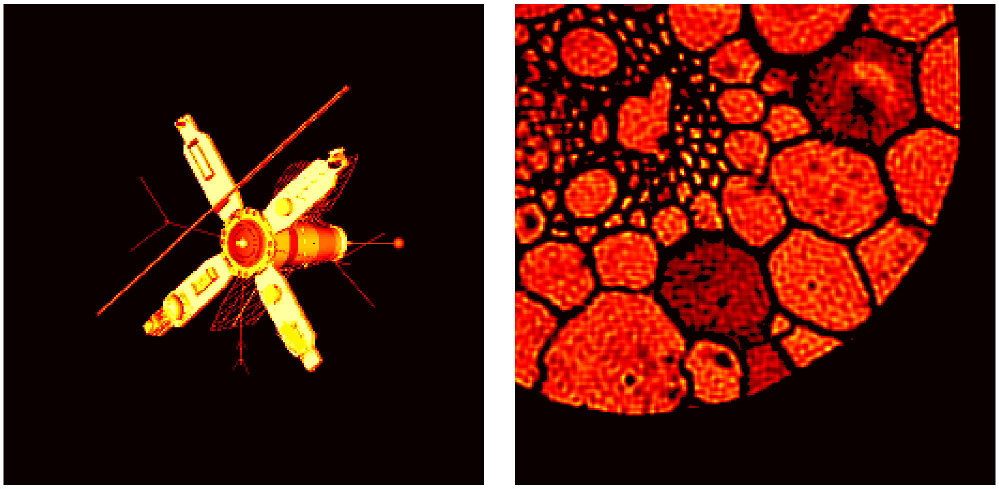


Figure 1.3: It is appropriate to use zero boundary conditions for the left image because it is all black near the edges (and likely outside) of the field of view. The right image has to use boundary condition other than zero, because the image is expected to extend with nonzero pixels outside the left top corner.

# Chapter 2

# Gradient Projection Methods

The gradient projection method can be literally understood as a combination of the gradient descent method, which solves the least squares problem, and a specific projection, which enforces additional constraints for various problems.

In this chapter, we focus on the least squares problem with nonnegative constraint, expressed as follows:

$$\min_f \|Kf - g\|_2, \quad \text{s.t.} \quad f \geq 0.$$

It basically first updates one step of the approximate solution by the gradient descent method, then projects the approximate solution onto the nonnegative orthant.

## 2.1 Gradient Descent Method

As we discussed in section 1.2, if we define $\psi(f) = \frac{1}{2}f^T K^T K f - f^T K^T g$, then the least squares problem $\min\limits_f \|Kf - g\|_2$ is equivalent to $\min\limits_f \psi(f)$. The iterative methods take the form:

$$f^{(k+1)} = f^{(k)} + \tau_k d^{(k)}$$

where $d^{(k)}$ is a step direction, and $\tau_k$ is a step size.

In the gradient descent method, we choose the step direction to be the negative gradient of $\psi(f)$, and choose the step size that minimizes $\psi(f)$ in the direction of the negative gradient. That is:

$$\begin{aligned} d^{(k)} &= -\nabla\psi(f^{(k)}) = K^T(g - Kf^{(k)}) \\ \tau_k &= \arg\min_{\tau} \psi(f^{(k)} + \tau d^{(k)}) \end{aligned}$$

To determine the value of $\tau_k$, we use a line search algorithm [10]:

- **Exact line search** is for the unconstrained problem, where the step size is exactly given by minimizing $\psi$ along the ray $\{f + \tau d \mid \tau \geq 0\}$. We first define function $\rho(\tau) = \psi(f + \tau d)$, then calculate the critical value, $\tau$, of the first derivative $\rho'(\tau) = 0$:

$$\begin{aligned} \rho(\tau) &= \psi(f + \tau d) \\ &= \frac{1}{2}(f + \tau d)^T K^T K(f + \tau d) - (f + \tau d)^T K^T g \\ &= \frac{1}{2}\big(f^T K^T K f + 2\tau d^T K^T K f + \tau^2 d^T K^T K d\big) - \big(f^T K^T g + \tau d^T K^T g\big) \\ &= \frac{1}{2} f^T K^T K f + \tau d^T K^T K f + \frac{1}{2}\tau^2 d^T K^T K d - f^T K^T g - \tau d^T K^T g \\ \implies \quad \rho'(\tau) &= d^T K^T K f + \tau d^T K^T K d - d^T K^T g \end{aligned}$$

  Let $\rho'(\tau) = 0$, then:

$$\tau = \frac{d^T K^T \big(g - Kf\big)}{d^T K^T K d} = \frac{d^T d}{d^T K^T K d} = \frac{\|d\|_2^2}{\|Kd\|_2^2}$$

- **Inexact (backtracking) line search** is for the problems under certain constraints, where the step size $\tau$ approximately minimizes $\psi(f + \tau d)$ along $\{f + \tau d \mid \tau \geq 0\}$. The backtracking line search starts with a relatively large initial guess of the step size $\tau$, and then iteratively shrinks $\tau$ by a factor of $\beta \in (0, 1)$ until the objective function $\psi(f)$ decreases corresponding to a step-wise movement from $f$ to $f + \tau d$. The exit condition of the iteration of backtracking line search is:

$$\psi(f + \tau d) - \psi(f) \leq \alpha \tau \nabla \psi(f)^T d$$

where $\alpha$ is another controlling parameter, such that $\alpha \in (0, 0.5)$, and $\nabla \psi^T(f) d$ is the slope of the function of $\tau$ along the searching direction $d = -\nabla \psi(f)$.

Thus the backtracking line search algorithm can be expressed as follows:

```
$\tau$=initial guess of step size;
while $\psi(f + \tau d) - \psi(f) > \alpha \tau \nabla \psi(f)^T d$
    $\tau = \beta \tau$;
return $\tau$
```

Since the projections are implemented after the line search, we choose step size $\tau_k$ via the exact line search. That is:

$$\tau_k = \arg\min_{\tau} \psi(f^{(k)} + \tau d^{(k)}) = \frac{\|d^{(k)}\|_2^2}{\|K d^{(k)}\|_2^2}$$

Thus, the basic iteration needs to compute:

$$\begin{aligned} r^{(k)} &= g - Kf^{(k)}, \\ d^{(k)} &= K^T r^{(k)}, \\ \tau_k &= \|d^{(k)}\|_2^2 / \|Kd^k\|_2^2, \\ f^{(k+1)} &= f^k + \tau_k d^{(k)}. \end{aligned}$$

We notice that $r^{(k+1)}$ can be computed more efficiently by expanding it to:

$$r^{(k+1)} = g - Kf^{(k+1)} = g - K(f^k + \tau_k d^{(k)}) = r^{(k)} - \tau_k K d^{(k)}.$$

Now, we can write the gradient descent algorithm as follows:

```
Algorithm: Gradient Descent

given:      $K$, $g$
choose:     initial guess for $f$
compute:
            $r = g - Kf$
            while (not stop) do
              $d = K^T r$
              $w = Kd$
              $\tau = \|d\|_2^2 / \|w\|_2^2$
              $f = f + \tau d$
              $r = r - \tau w$
            end while
```

For more details about the gradient descent methods, we refer [8, 10].

## 2.2 Nonnegative Projection

The nonnegative constraint, $f \geq 0$, can be easily applied into the iteration by the projection $\mathcal{P}_{nn}(\cdot)$, such that:

$$[\mathcal{P}_{nn}(f)]_i = \begin{cases} f_i & \text{if} \quad f_i \geq 0, \\ 0 & \text{if} \quad f_i < 0. \end{cases}$$

where the $i^{th}$ entry of a projected vector $f$ will keep the same if it is nonnegative, otherwise it will become zero. Thus, we can obtain the gradient projection algorithm by just choosing a nonnegative initial guess and replacing

$$f = f + \tau d$$

with

$$f = \mathcal{P}_{nn}(f + \tau d).$$

We can write the algorithm of gradient descent with nonnegative constraint as follows:

**Algorithm: GDnn**

```
given:      $K$, $g$
choose:     initial guess for $f \geq 0$
compute:
            $r = g - Kf$
            while (not stop) do
              $d = K^T r$
              $w = Kd$
              $\tau = \|d\|_2^2 / \|w\|_2^2$
              $f = \mathcal{P}_{nn}(f + \tau d)$
              $r = r - \tau w$
            end while
```

## 2.3 Conjugate Gradient Method for Least Squares

Conjugate gradient methods are usually much more efficient than the gradient descent method; for more details, see [12, 7]. But incorporating projections into conjugate gradient methods is not always feasible. In this thesis, we use the conjugate gradient method for least squares (CGLS) as a comparison to the gradient projection methods. For the experiments written in chapter 4, we use a naive way to incorporate the constraints into CGLS, which is simply applying the projection on its approximated solution following the final iteration step.

# Chapter 3

# Projection for Volume and Nonnegative Constraints

In this chapter, we will extend the gradient projection method discussed in chapter 2, to preserve not only the nonnegativity but also the total volume of the data. The task can be described as the following optimization problem:

$$\min_f \|Kf - g\|_2^2,$$

$$\text{s.t.} \quad f \geq 0, \quad \sum_{i=1}^{n} f_i = C$$

where $C$ is the total volume of the data, and everything else is the same as defined in chapter 2.

We use exactly the same algorithm as described in section 2.2: first updating one step of the approximate solution $f$ by the gradient descent method, then projecting $f$ by the specific projection. But instead of using projection $P_{nn}(\cdot)$, which only enforces nonnegative constraint, we develop a more complicated projection $P_{nnv}(\cdot)$ in the following section, so that it enforces both nonnegative and volume constraints on the approximate solution $f$.

The algorithm of gradient descent with nonnegative and volume constraint is depicted as follows:

```
Algorithm: GDnnv

given:      $K$, $g$
choose:     initial guess for $f \geq 0$
compute:
            $r = g - Kf$
            while (not stop) do
              $d = K^T r$
              $w = Kd$
              $\tau = \|d\|_2^2 / \|w\|_2^2$
              $f = \mathcal{P}_{nnv}(f + \tau d)$
              $r = r - \tau w$
            end while
```

## 3.1 Nonnegative and Volume Projection

In this section, we focus on developing the projection $\mathcal{P}_{nnv}(\cdot)$, which preserves the volume and the nonnegativity of $f^{(k)}$. Basically, It can be viewed as an optimization problem:

$$\mathcal{P}_{nnv}(y) = \arg\min_{w} \frac{1}{2}\|w - y\|_2^2$$

$$\text{s.t.} \quad w \geq 0, \quad \|w\|_1 = C$$

where vectors $y$ and $w$ respectively corresponds to the given $f^{(k)}$ and the projection of $f^{(k)}$, and $C$ represents our expectation of the total data volume.

We will solve this optimization problem by the Lagrange multiplier approach [5].

### 3.1.1 Lagrange Multiplier Approach

The Lagrangian of the projection problem is

$$\mathcal{L}(w, \rho, z) = \frac{1}{2}\|w - y\|_2^2 + \rho\Big(\|w\|_1 - C\Big) - z^T w$$

where $\rho$ is a Lagrange multiplier, and $z$ is a vector of non-negative Lagrange multipliers.

Since the optimality condition occurs when $\frac{d\mathcal{L}}{dw_i} = 0$, and the first order derivative of the Lagrangian is $\frac{d\mathcal{L}}{dw_i} = w_i - y_i + \rho - z_i$, it implies that:

$$w_i - y_i + \rho - z_i = 0.$$

If $w_i > 0$ and $z^T w = 0$, then we must have $z_i = 0$. This means that for all positive entries $w_i > 0$, we must have that $w_i = y_i - \rho + z_i = y_i - \rho$. To be more precise:

$$w_i = \max(y_i - \rho, 0).$$

Thus, if the Lagrange multiplier $\rho$ is known, the projection problem can be solved easily as the following equation

$$[\mathcal{P}_{nnv}(y)]_i = \max(y_i - \rho, 0).$$

This $\rho$ can be determined by using Newton's Method, and we will discuss it in the following section.

### 3.1.2 Newton's Method for finding $\rho$

For the projection problem, the volume constraint $\|w\|_1 = C$ also needs to be satisfied. Assume $w \in R^n$, with $w_i \geq 0$. Then $\|w\|_1 = \sum_{i=1}^{n} w_i = C$. Thus for the best-fit solution of this projection problem, it requires

$$\sum_{i=1}^{n} \max(y_i - \rho, 0) = C.$$

If we define function $\phi(\rho) = \sum_{i=1}^{n} \max(y_i - \rho, 0)$, then the $\rho$ needed in section 3.1.1 is the exact solution to the equation $\phi(\rho) - C = 0$. We are able to solve $\phi(\rho) - C = 0$ by using Newton's method [10], which takes the following form:

```
$\rho_0$=initial guess
for $k = 1, 2, ...$
    • $\rho_{k+1} = \rho_k - \frac{\phi(\rho_k)-C}{\phi'(\rho_k)}$
    • determine if the stop criteria are satisfied.
end
```

In order to improve the efficiency of the Newton's method, it demands a good initial guess of $\rho$.

### 3.1.3 Initial Guess of $\rho$

In this section, we focus on finding a good initial guess $\rho_0$, such that it takes only a few iterations in Newton's method to obtain an accurate enough solution $\rho$.

We start by sorting the given vector $y$, so that all of its entries increase monotonically, such that $y_1 \leq y_2 \leq \cdots \leq y_n$ .

Then if $\rho = y_j$, we get

$$\max(y_k - \rho, 0) = \begin{cases} 0 & \text{if} \quad k \leq j \\ y_k - \rho & \text{if} \quad k > j \end{cases}$$

Thus,

$$\begin{aligned} s_j = \phi(y_j) - C &= \sum_{k=j+1}^{n} (y_k - y_j) - C \\ &= \sum_{k=j+1}^{n} y_k - (n-j)y_j - C \end{aligned}$$

Notice that

$$\begin{aligned} s_{j+1} &= \sum_{k=j+2}^{n} y_k - \big(n - (j+1)\big)y_{j+1} - C \\ &= \sum_{k=j+1}^{n} y_k - (n-j)y_{j+1} - C \\ &= s_j - (n-j)(y_{j+1} - y_j) \end{aligned}$$

Therefore, we can generate the sequence of $\{s_i\}_{i=1}^{n}$ recursively:

```
Let $s_1 = \sum_{k=1}^{n} y_k - ny_1 - C$
for $j = 1 : n-1$
    $s_{j+1} = s_j - (n-j)(y_{j+1} - y_j)$
end
```

We will use $y_j$ as the initial guess $\rho_0$, where $j$ is the largest $j$ such that $s_j > 0$.

# Chapter 4

# Numerical Experiments

In this chapter, we display some numerical results that illustrates the effectiveness and efficiency of the gradient projection methods described in chapter 2 and chapter 3. We use three test problems, named Satellite, StarCluster, and Grain in the MATLAB toolbox *Restore Tools* [2, 9]. In each one of the test problems, we include the following:

- Comparing the performance of Gradient descent (GD) and CGLS methods when they are:

    1. Unconstrained
    2. Nonnegative constrained, named GDnn and CGLSnn.
    3. Nonnegative and volume constrained, named GDnnv and CGLSnnv.

    Here, we use the CGLS method, which is introduced in section 2.3, as a comparison to the GD method. We implement the constraints on CGLS by simply applying the corresponding projection after its optimal reconstructed solution is computed.

    We use relative error

    $$\frac{\|f^{(k)} - f_{true}\|_2}{\|f_{true}\|_2}$$

    as a measure for the accuracy of the reconstructions, and use the number of iterations, where the relative error reaches the minimum, as an indicator for the efficiency of the reconstructions. As for the volume constraint, we set it to be the total volume in the given observation.

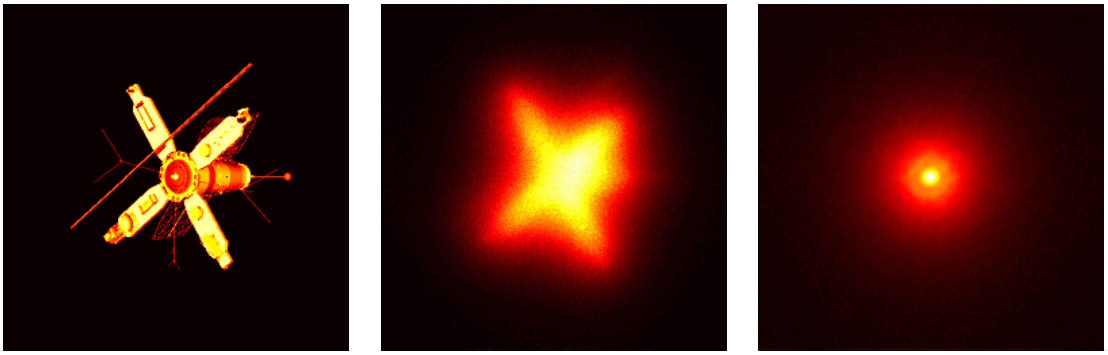
- Testing the efficiency of the nonnegative and volume projection:
  Since constrained methods are based on the projections, we need to check the efficiency of the projections in order to control the total reconstruction time. Nonnegative projection is very simple to apply, so that we ignore it and only test the nonnegative and volume projection.
  We first plot the graph of $\phi(\rho) - C$, described in section 3.1.3, to show that our choice of $\rho_0$ in the projection is visually a good initial guess. Then we determine the number of iterations in Newton's method. For an efficient projection, we expect the number to be very small.

## 4.1 Test Problem: Satellite

This is an example of astronomical image reconstruction. The true image, the blurred observation, and the point spread function of this test problem are displayed in Figure 4.1, where all images are of size 256 × 256. The blur of this test problem is expected to be spatially invariant. The boundary condition is set to zero.

### 4.1.1 Methods Comparison

We compare the performance of the unconstrained and constrained CGLS and GD methods for this example in Figure 4.2 and Figure 4.3, and represent the numerical results explicitly in the following table. The curves in the graphs represent the convergence history of CGLS and GD methods, and the points in the graphs show the occurrence of the optimal reconstruction, such that the relative error of the recon-



(a) True Image. (b) Observation. (c) PSF.

Figure 4.1: (a) is the true image of the test problem Satellite; (b) is the blurred image that is given as the observation; (c) is the given point spread function that describes the blurring phenomenon.

struction reaches its minimum through the entire iteration history. For the purpose of demonstration, we enlarge the partial segment of the left graph, focusing on the optimal reconstruction, and display it on the right side of Figures 4.2 and 4.3.

| | Minimum Relative Error | Number of Iteration |
|---|---|---|
| CGLS | 0.3557 | 43 |
| CGLSnn | 0.1446 | 43 |
| CGLSnnv | 0.1499 | 43 |
| GD | 0.3533 | 1101 |
| GDnn | 0.3555 | 1301 |
| GDnnv | 0.3475 | 1596 |

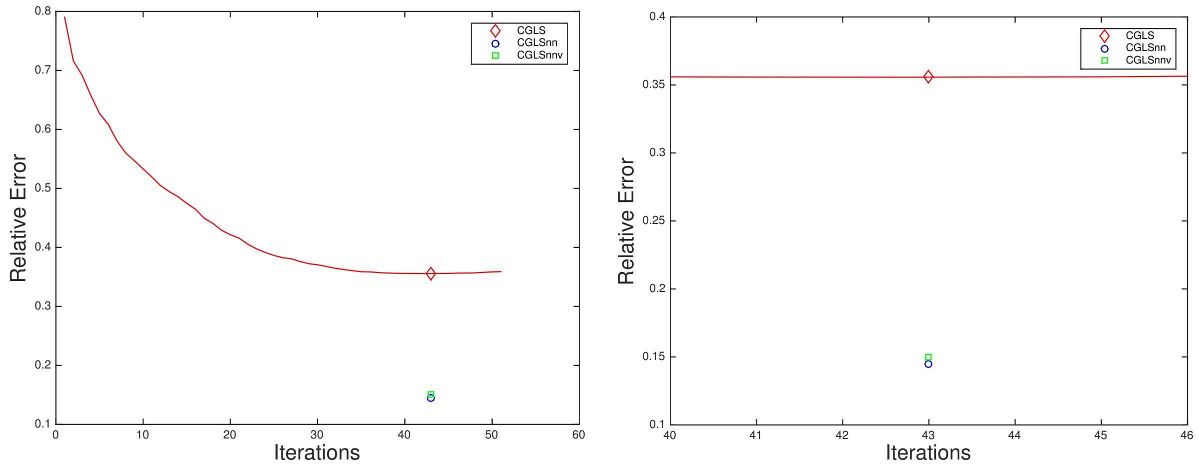


Figure 4.2: Performance of constrained and unconstrained CGLS method for the example of Satellite. The right graph is an enlarged segment of the left graph.

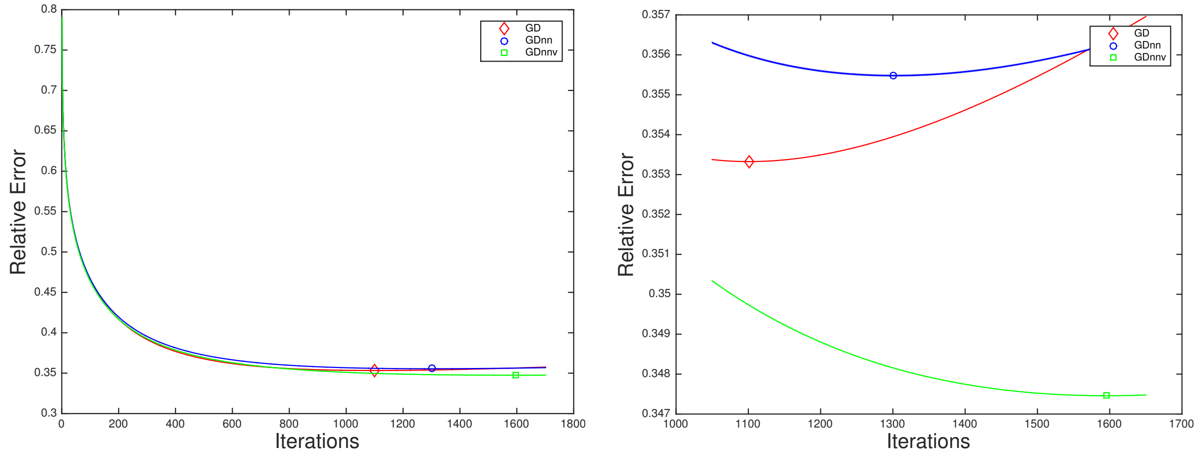


Figure 4.3: Performance of constrained and unconstrained GD method for the example of Satellite. The right graph is an enlarged segment of the left graph.

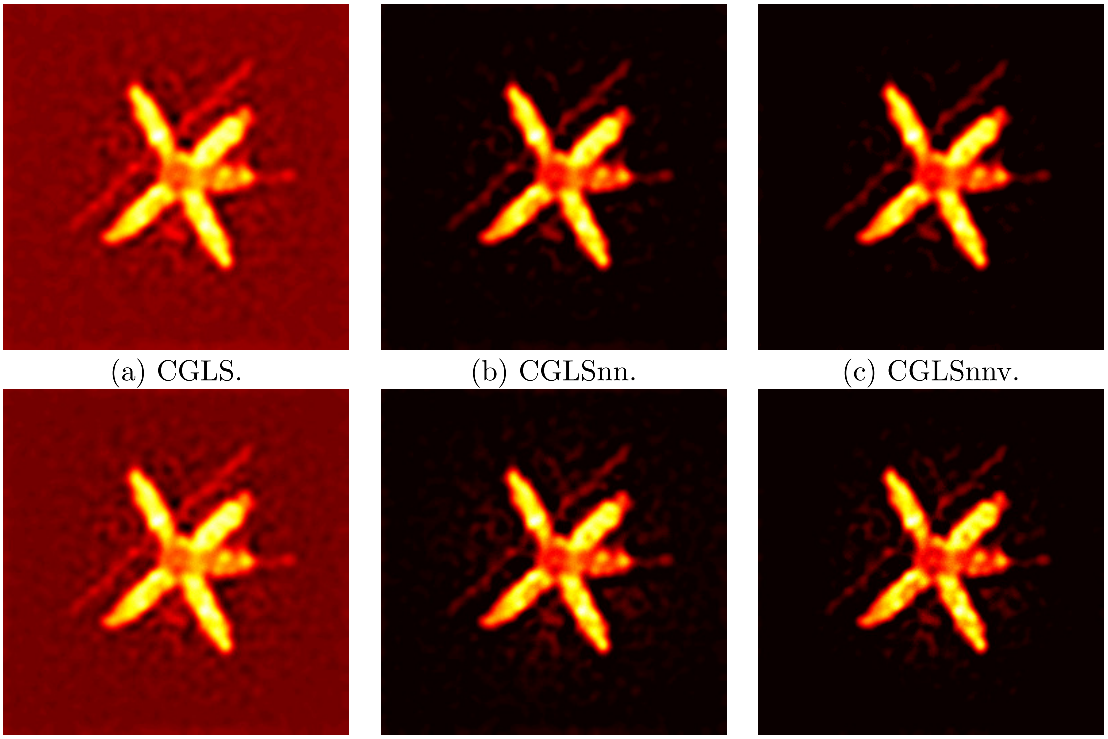
Here are our observations from the numerical results for this example:

- For CGLS based methods, the two projections of the constrained methods significantly improve the accuracy of the reconstruction from the unconstrained CGLS method. In between the constrained methods, CGLSnn is slightly better than CGLSnnv.
- For GD based methods, GDnnv generates the best accuracy of the reconstruction with the largest number of iterations, while GDnn is slightly worse than the unconstrained GD method and requires more iterations.
- In general, CGLS based methods are much more efficient than GD based methods.
- It's interesting to notice that CGLSnn generates the best reconstruction with the fewest number of iterations.

In Figure 4.4, we show the reconstructed images for each method. We can see that the two projections significantly improve the sharpness of the reconstructions for both methods. In general, CLGS based methods generate sharper reconstructed images than GD based methods. It is also interesting to point out that although CLGS with nonnegative projection generates the solution with minimum relative error, the solution actually looks better when also including the volume constraint, achieved by the nonnegative and volume projection.

### 4.1.2 Efficiency of the NNV Projection

As the method described in section 3.1.3, we are required to find the solution $\rho$ to the equation $\phi(\rho) - C = \sum\limits_{i=1}^{n}(y_i - \rho) - C = 0$. Newton's method can be used to solve the problem, but for doing so, we need a good initial guess that decreases the Newton



(a) CGLS. (b) CGLSnn. (c) CGLSnnv.

(a) GD. (b) GDnn. (c) GDnnv.

Figure 4.4: Reconstructed images for CLGS and GD based methods for the example of Satellite.

iterations as much as possible.

We set the initial guess to be $\rho_0 = y_j$, where $j$ is the largest $j$ such that, $\phi(y_j) - C > 0$. From Figure 4.5, we notice that our initial guess of $\rho_0 = y_j$ is very close to the intersection of the curve $\phi(\rho) - C$ and the x-axis. Thus we visually accept it as a good initial guess.

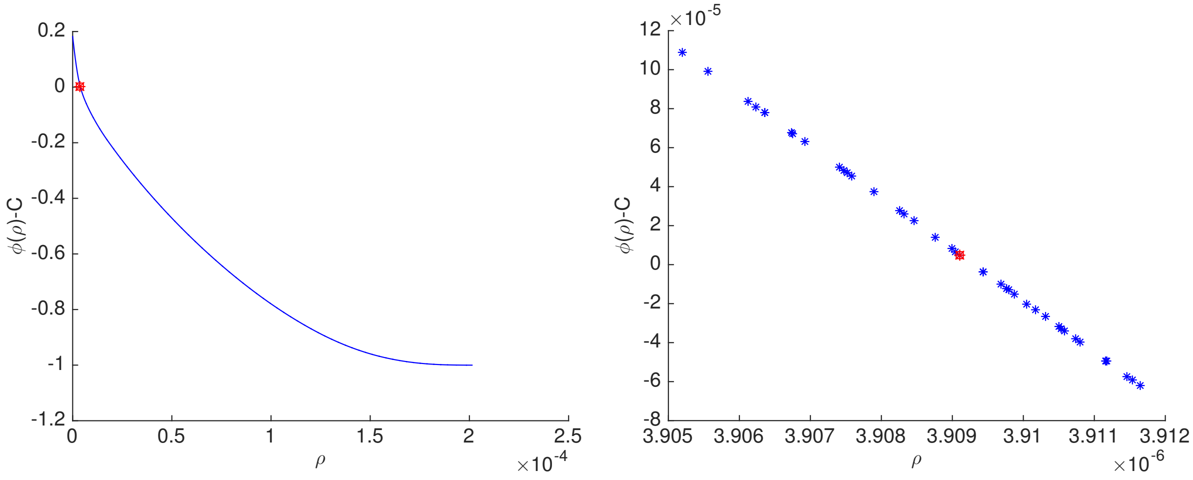


Figure 4.5: This is the plot of $\phi(\rho) - C$ for the example of Satellite. The blue points on the graphs represents the values of $\phi(y_j) - C$, where $y_j$ is the $j^{th}$ entry of the reconstructed solution at iteration 100 of GDnn. The red point is our initial guess of $\rho$. The right graph is the partial enlarged segment of the left graph focusing on our initial guess.

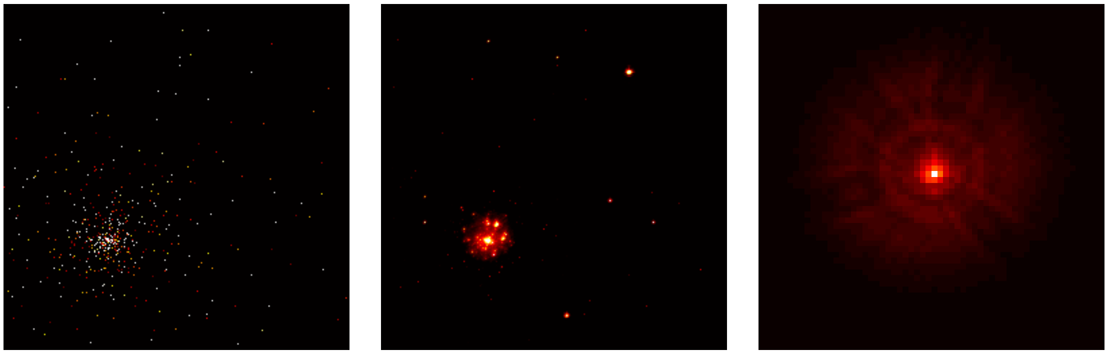
We also record the number of iterations in Newton's method when doing the experiment in Figure 4.3. The numerical results are shown in the following table:

| | Max. # of Iteration | Min. # of Iteration | Avg. # of Iteration |
|---|---|---|---|
| GDnnv | 6 | 1 | 4.9937 |

These results show that our choice of the initial guess $\rho_0$ extremely decreases the number of the iterations in Newton's method, and thus allows the NNV projection to be very efficient.

## 4.2 Test Problem: StarCluster

This is another example of astronomical image reconstruction. The true image, the blurred observation, and the given point spread functions (here we only display one of them as a representation) of this test problem are displayed in Figure 4.6, where all of the images are of size $256 \times 256$. The blur of this test problem is expected to be locally spatially invariant. In this experiment, we use a $2 \times 2$ partitioning (i.e., four regions), and assume each region of the blurred image is formed with one point spread function. As in the previous example in section 4.1, we set the boundary condition to be zero.



(a) True Image. (b) Observation. (c) PSF.

Figure 4.6: (a) is the true image of the test problem StarCluster; (b) is the blurred image that is given as the observation; (c) is one of the four point spread functions of this test problem that represents the blurring operator.

### 4.2.1 Methods Comparison

We compare the performance of the unconstrained and constrained CGLS and GD methods for this example in Figure 4.7 and Figure 4.8, and represent the numerical results explicitly in the following table. All the other setups are the same as in the previous example in section 4.1

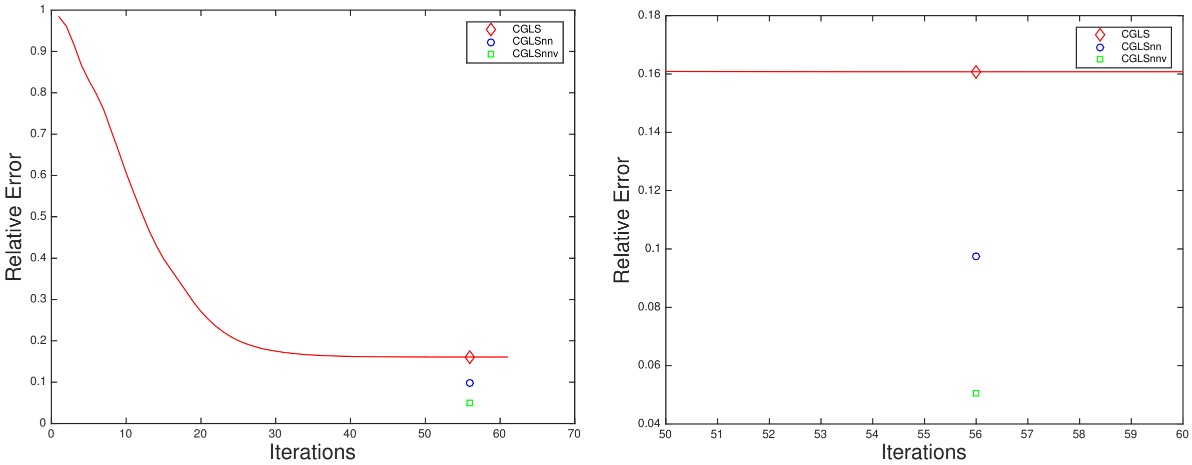


Figure 4.7: Performance of constrained and unconstrained CGLS method for the example of StarCluster. The right graph is an enlarged segment of the left graph.

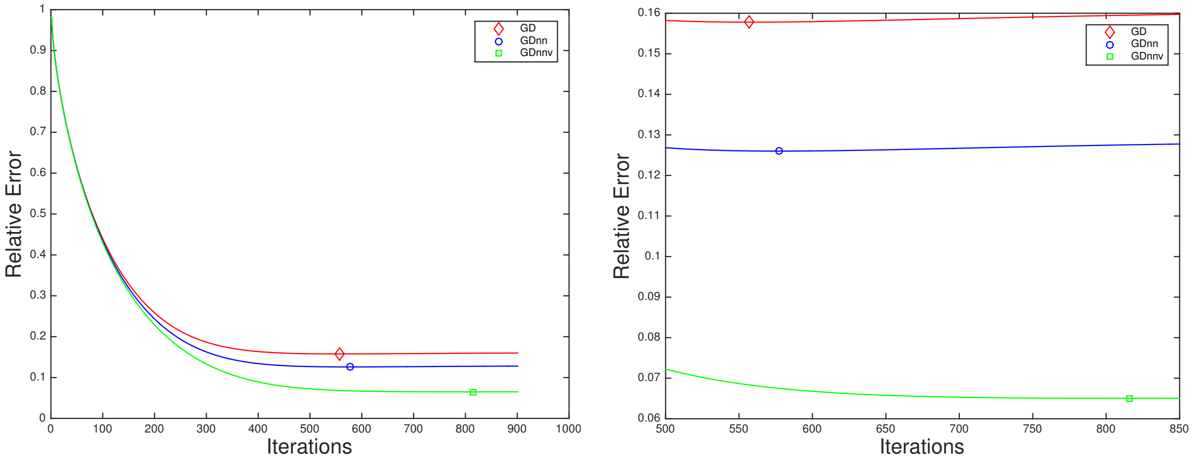


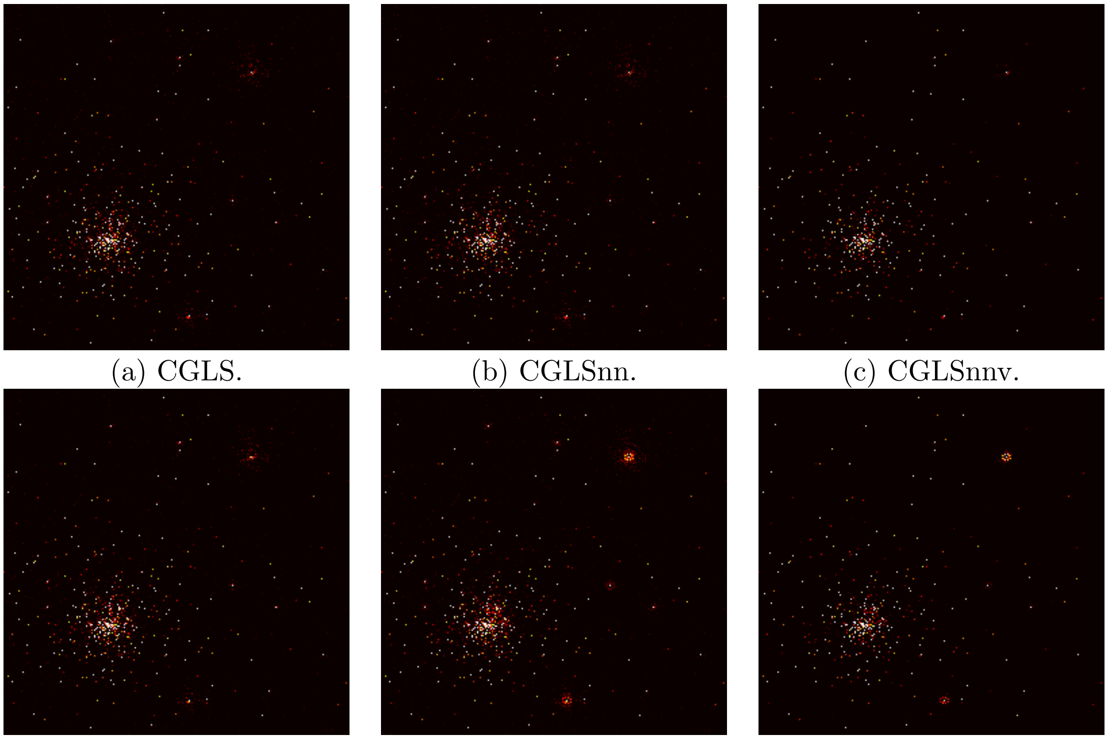
Figure 4.8: Performance of constrained and unconstrained GD method for the example of StarCluster. The right graph is an enlarged segment of the left graph.

| | Minimum Relative Error | Number of Iteration |
|---|---|---|
| CGLS | 0.1608 | 56 |
| CGLSnn | 0.0973 | 56 |
| CGLSnnv | 0.0506 | 56 |
| GD | 0.1578 | 557 |
| GDnn | 0.1260 | 577 |
| GDnnv | 0.0650 | 816 |

Here are our observations from the numerical results for this example:

- For both CGLS and GD based methods, the two projections of the constrained methods significantly improve the accuracy of the reconstruction from the unconstrained methods. In between the constrained methods, CGLSnnv performs better than CGLSnn. GDnnv also performs better than GDnn, but with an increase cost of the iteration.
- Although CGLS is not as accurate as GD methods for the unconstrained problem, when we apply the same projection constraints on the final computed solution, CGLS based methods generate a more accurate solution with much fewer number of iterations than GD based methods.
- The reconstruction with the best accuracy and the fewest number of iterations comes from CGLSnnv.

In Figure 4.9, we show the reconstructed images for each method. For the clarity purpose, here we only display the data with values in the range of $(50, 500)$. In general, CGLS based method generate sharper reconstructed images than GD based methods. We also conclude that for both methods, the volume constraint, incorperated by NNV projection, further improves the sharpness of the reconstruction on the base of the nonnegative constraint.



(a) CGLS. (b) CGLSnn. (c) CGLSnnv.

(a) GD. (b) GDnn. (c) GDnnv.

Figure 4.9: Reconstructed images for CLGS and GD based methods for the example of StarCluster, displaying the values in the range of $(50, 500)$ for clarity purpose.

### 4.2.2 Efficiency of the NNV Projection

From Figure 4.10, we notice that our initial guess of $\rho_0 = y_j$, shown as the red dot, is very close to the intersection of the curve $\phi(\rho) - C$ and the x-axis. Thus we visually accept it as a good initial guess.

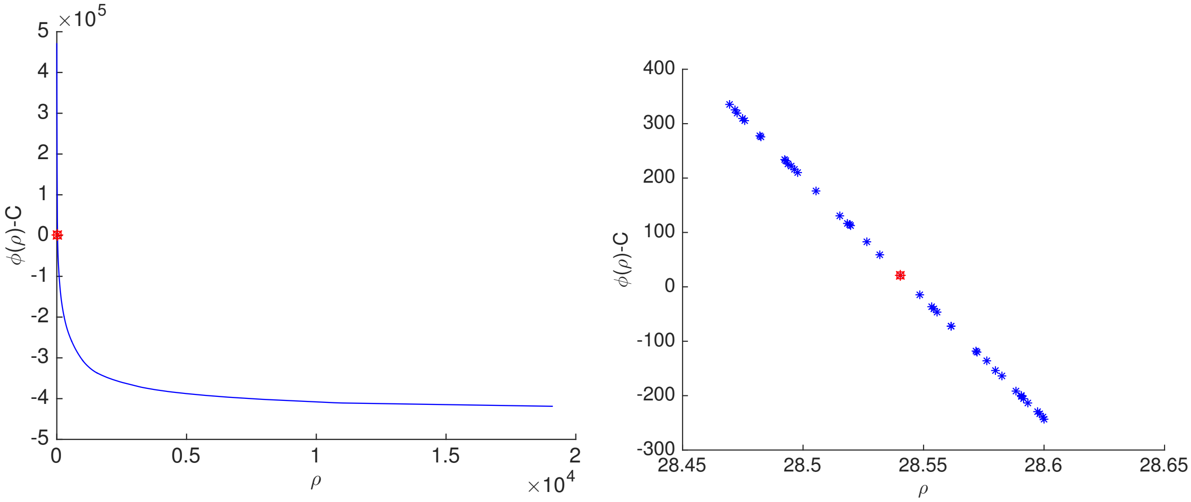


Figure 4.10: This is the plot of $\phi(\rho) - C$ for the example of StarCluster. The blue points on the graphs represents the values of $\phi(y_j) - C$, where $y_j$ is the $j^{th}$ entry of the reconstructed solution at iteration 100 of GDnn. The red point is our initial guess of $\rho$. The right graph is the partial enlarged segment of the left graph focusing on our initial guess.

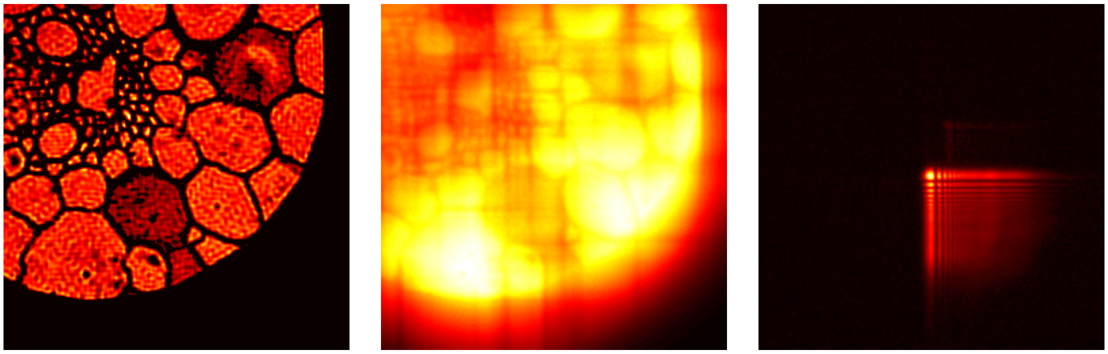
We also record the number of iterations in Newton's method while doing the previous experiment in Figure 4.8. The numerical results are shown in the following table:

| | Max. # of Iteration | Min. # of Iteration | Avg. # of Iteration |
|---|---|---|---|
| GDnnv | 11 | 4 | 7.7479 |

The number of the iterations in Newton's method for this problem is very small, which means that the NNV projection is very efficient. Thus, we can say that our initial guess $\rho_0$ is a really good choice.

## 4.3 Test Problem: Grain

The true image, the blurred observation, and the given point spread function of this test problem are displayed in Figure 4.11, where all of the images are of size 256 × 256. The blur of this test problem is expected to be spatially invariant. We set the boundary condition to be reflective.



(a) True Image. (b) Observation. (c) PSF.

Figure 4.11: (a) is the true image of the test problem Grain; (b) is the blurred image that is given as the observation; (c) is the given point spread function that represents the blurring phenomenon

### 4.3.1 Methods Comparison

We compare the performance of the unconstrained and constrained CGLS and GD in Figure 4.12 and Figure 4.13. The numerical results are presented in the following table (we omit the results for the gradient descent methods because they require more than 5000 iterations):

| | Minimum Relative Error | Number of Iteration |
|---|---|---|
| CGLS | 0.2266 | 176 |
| CGLSnn | 0.1415 | 176 |
| CGLSnnv | 0.1963 | 176 |

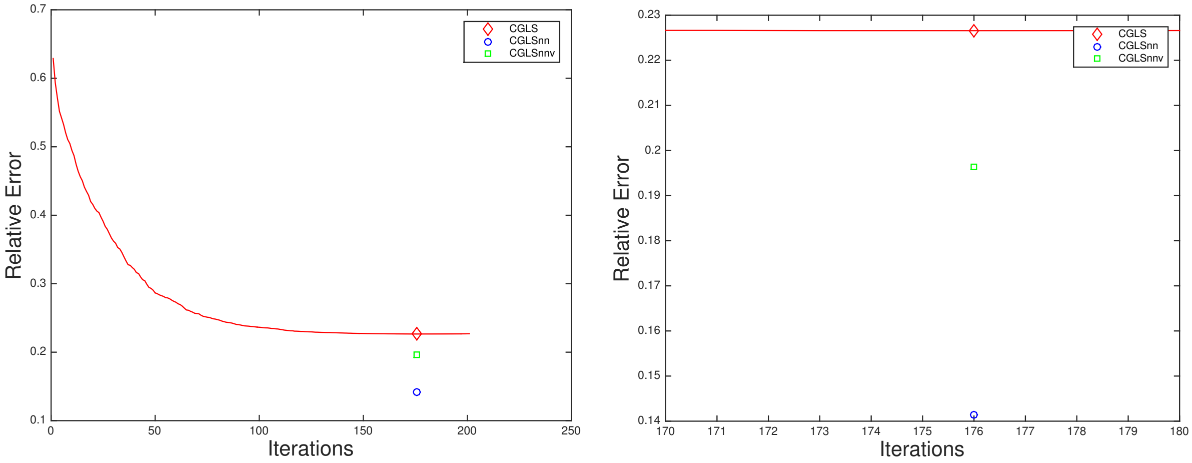


Figure 4.12: Performance of constrained and unconstrained CGLS method for the example of Grain. The right graph is an enlarged segment of the left graph.

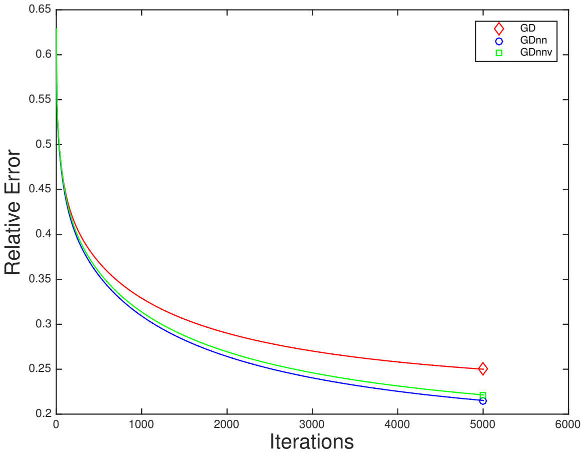
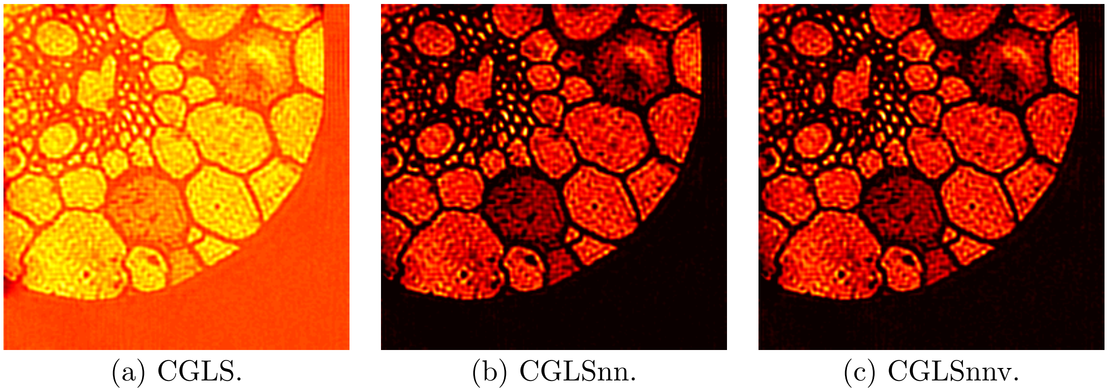


Figure 4.13: Performance of constrained and unconstrained GD method for the example of Grain. The optimal solutions are not reached yet at iteration 5000.

Here are our observations from the numerical results for this example:

- For CGLS, the 2 projections significantly improve the accuracy of the reconstruction from the unconstrained method.
- For GD, the 2 projections improve the accuracy of the reconstruction from the unconstrained method in the first 5000 iterations.
- GD based methods are obviously not very efficient in this case. The optimal solutions are not reached even after 5000 iterations. We might need other tools to accelerate the convergence.

In Figure 4.14, we show the reconstructed images for CGLS based methods. We can see that the two projections significantly improve the sharpness of the reconstruction from the unconstrained CGLS method. For the two constrained methods, the results are very similar. In between them, CGLSnn method produce a slightly better looking reconstructed image.



(a) CGLS. (b) CGLSnn. (c) CGLSnnv.

Figure 4.14: Reconstructed images for CLGS based methods for the example of Grain.

# Chapter 5

# Conclusion and Discussion

In the previous chapters, we have discussed the gradient projection method that can be used for image deblurring, and the two projections that enforce nonnegativity and volume constraints on the solution. We compared the performance of gradient projection methods with CGLS methods on three test problems using MATLAB. The numerical results illustrate the benefits of implementing these constraints and the effectiveness of our projections. But this is just the beginning of our study. More things need to be discussed in the future. For example, it would be interesting to use a projected Newton method instead of projected gradient descent. Because a Newton method uses second derivative information, it will usually converge much faster than gradient descent, which uses only first derivative information. However, it is much more difficult to implement a projected Newton method for large scale problems in image processing [10]. It would also be interesting to try preconditioned Landweber methods [11]. This approach requires additional work to define and implement a preconditioning technique that is beyond the scope of this thesis. Finally we mention that an important application of the volume constraint is in astronomical imaging, especially of stellar objects (e.g., binary stars, galaxies, etc.), where scientists want to preserve photometry [4]. It would be interesting to apply the method in this thesis to real astronomical imaging applications, and compare with the techniques used by astronomers.

# Bibliography

[1] J. M. Bardsley and J. G. Nagy. Covariance-preconditioned iterative methods for non-negatively constrained astronomical imaging. *SIAM Journal on Matrix Analysis Applications*, 27:1184–1197, 2006.

[2] S. Berisha and J. G. Nagy. Iterative methods for image restoration. In R. Chellappa and S. Theodoridis, editors, *Academic Press Library in Signal Processing*, volume 4, pages 193–243. Academic Press, 2014.

[3] M. Bertero and P. Boccacci. *Introduction to Inverse problems in Imaging*. Institute of Physics Publishing, Bristol, UK, 1998.

[4] E. Bratsolis and M. Sigelle. A spatial regularization method preserving local photometry for Richardson-Lucy restoration. *Astronomy and Astrophysics*, 375:1120–1128, 2001.

[5] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 272–279, New York, NY, USA, 2008. ACM.

[6] C. W. Groetsch. *Inverse Problems*. The Mathematical Association of America, Washington, DC, 1999.

[7] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[8] J. G. Nagy and K. M. Palmer. Steepest descent, CG, and iterative regularization of ill-posed problems. *BIT*, 43:1003–1017, 2003.

[9] J. G. Nagy, K. M. Palmer, and L. Perrone. Iterative methods for image deblurring: A Matlab object oriented approach. *Numerical Algorithms*, 36:73–93, 2004.

[10] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 1999.

[11] M Piana and M Bertero. Projected Landweber method and preconditioning. *Inverse Problems*, 13(2):441–463, 1997.

[12] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1994.