Automated Parameter Selection Tool for Solution to Ill-Posed Problems
Mid-Year Report

Brianna Cash (brcash@math.umd.edu)

Advisor: Dianne O'Leary (oleary@cs.umd.edu)

December 19, 2011

Abstract

In many ill-posed problems it can be assumed that the error in the data is dominated by noise which is independent identically normally distributed. Given this assumption the residual should also be normally distributed with similar mean and variance. This idea has been used to develop three statistical diagnostic tests to constrain the region of plausible solutions. This project aims to develop software that automates the generation of a range of plausible regularization parameters based on diagnostic tests.

1 Background

In medical images such as MRI or CT scans (*figure 1*), the images may be distorted and/or noisy due to the physics of the measurement and the structure of the material (human) being imaged. These images are expensive to produce and often are critical in making medical decisions. Image deblurring is an example of an ill-posed inverse problem. To find suitable approximate solutions to ill-posed inverse problems we use our knowledge about the particular problem to come up with constraints [4]. These constraints are used to determine parameters to regularize the problem, replacing the ill-posed problem by one that is well-posed, and thus has an acceptable solution. Finding and selecting good regularization parameters can be very expensive and subject to bias. Researchers often have invaluable information that is crucial in finding a good approximate solution, but without validation, there is risk of seeing what is expected and not the true solution or image (*figure 2*). An effective automated tool that generates a plausible range of regularization parameters is needed to create both a cost effective methodology and control for bias when determining optimal solutions.

The goal of this project is to develop a software package with graphical user interface (GUI) for parameter selection in regularizing ill-posed problems, and apply it to debluring and denoising problems with Gaussian noise. The software will use generalized cross-validation (GCV) for initial parameter selection

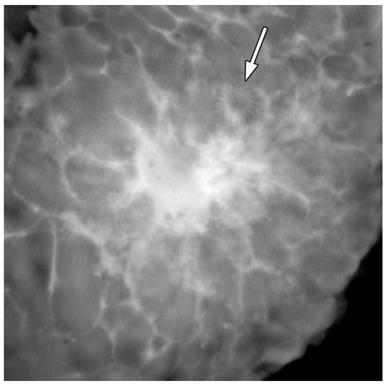


Figure 1: Stevens G M et al. Radiology 2003;228:569-575

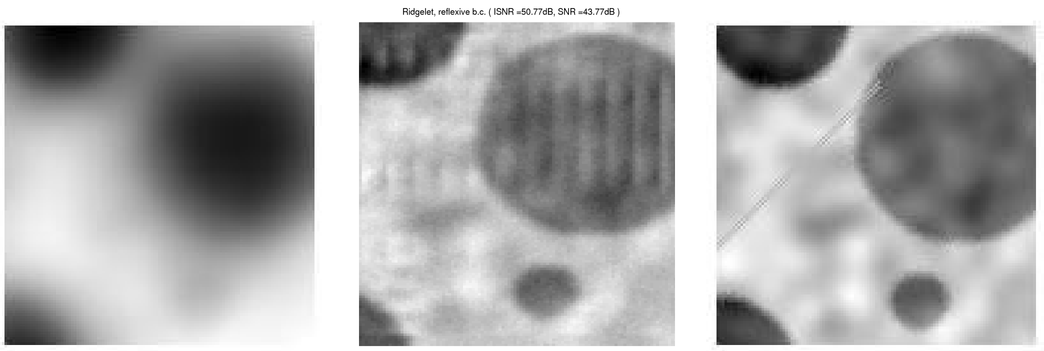


Figure 2: Left: Blurred Image, Center: Deblurred Image, Right: True Image which has “trian tracks”. Without knowlege of the true image having “train tracks” one might accept the deblurred to be a good image without realizing that important information was lost in the process. Images courtesy of Dianne O’Leary

and statistical diagnostics to validate candidate solutions based on user provided parameters. A number of regularization methods will be used in hopes of determining optimal methods for a given image. To start, the software will include Truncated SVD (TSVD) regularization, Tikhonov regularization, and Total Variation (TV) regularization.

2 Regularization methods for ill-posed problem

Consider the ill-posed and ill-conditioned discrete problem

$$\mathbf{Ax} = \mathbf{b} \tag{1}$$

where $\mathbf{A}$ is a known $m \times n$ matrix where $m \geq n$, $\mathbf{b}$ is a known $m \times \mathit{1}$ vector (measured image) and $\mathbf{x}$ is an unknown $n \times \mathit{1}$ vector. In general to solve the inverse problem one would solve $\mathbf{Ax} = \mathbf{b}$ or the equivalent least square problem

$$\min_x \|\mathbf{Ax} - \mathbf{b}\|_2^2 \tag{2}$$

Because $\mathbf{A}$ is ill-conditioned, solving $\mathbf{Ax} = \mathbf{b}$ directly generally does not give good results. To make the problem less sensitive one needs to apply a method which imposes stability to the problem while retaining desired features of the solution. Regularization methods incorporate *a priori* assumptions about the size and smoothness of the desired solution.

$$\min_x \|\mathbf{Ax} - \mathbf{b}\|_\mathbf{2}^\mathbf{2} + \gamma\mathbf{\Omega}(\mathbf{x}) \tag{3}$$

Where the second term of the expression above is the regularization term where $\Omega(x)$ is a smoothing function or penalty function and γ is the regularization parameter.

2.1 Tikhonov's Regularization method

In this method our penalty function is

$$\Omega(x) = \|\mathbf{Lx}\|_\mathbf{2}^\mathbf{2} \tag{4}$$

where $\mathbf{L}$ is the identity matrix, approximation of the first derivative operator, a diagonal weighting matrix [5], or any other type of operator based on the problem and the desired features.

This method has been implemented using a class in *RestoreTool* where we assume that $\mathbf{L}$ is the indentity matrix. It is solved directly where

$$\mathbf{x_{tik}} = (\mathbf{A^T A} + \gamma\mathbf{I_n})^{-1}\mathbf{A^T b} \tag{5}$$

When one takes the Singular Value Decomposition (SVD) of $\mathbf{A}$, where

$$\mathbf{A} = \mathbf{U\Sigma V^T} = \sum_{i=1}^{n} \mathbf{u_i}\sigma_i\mathbf{v_i^T} \tag{6}$$

it is easily shown that

$$x_{tik} = \sum_{i=1}^{n} \frac{\sigma_i \mathbf{u_i} \mathbf{v_i^T} \mathbf{b}}{\sigma_i^2 + \gamma} \tag{7}$$

2.2 Truncated SVD

In Truncated SVD (TSVD) we regularize the problem by truncating $\mathbf{A}$ and therefor ignoring the small singular values which cause $\mathbf{A}$ to be ill-conditioned. The regularization problem becomes

$$\min_x \|\mathbf{A_k x} - \mathbf{b}\|_2^2 \tag{8}$$

where

$$\mathbf{A}_k = \sum_{i=1}^{k} \mathbf{u_i} \sigma_i \mathbf{v_i^T} \tag{9}$$

and the regularization parameter is k or the level of truncation. Again when one takes the SVD of $\mathbf{A}_k$ it is easily shown that the solution is

$$x_{TSVD} = \sum_{i=1}^{n} \phi_i \frac{\mathbf{u_i} \mathbf{v_i^T} \mathbf{b}}{\sigma_i} \tag{10}$$

where $\phi_i = 1$ for $i = 1..k$ and 0 for $i = k + 1..n$.

2.3 Total Variation Regularization method

In Total Variation (TV) regularization we assume that our penalty function is the l_1 norm of the gradient of the solution and thereby help retain steep gradients that may be present, which may be lost in methods where the l_2 norm is used. The penalty function is

$$\Omega(x) = TV(\mathbf{x}) \tag{11}$$

where

$$TV(\mathbf{x}) = \int_\Omega \sqrt{|\nabla \mathbf{x}|^2} d\Omega \tag{12}$$

Unlike the previous methods, our regularization problem is non-linear and the TV term is non-differentiable. There have been many proposed interactive methods to approximate the solution including time marching schemes, steepest descent, Newton's method, lagged diffusivity fixed point iterative method [14], and primal-dual Newton method [3]. In many of these methods the difficulty of the $TV(\mathbf{x})$ term not being differentiable at zero is avoided by adding a small positive constant value so our term becomes

$$TV(\mathbf{x}) = \int_\Omega \sqrt{|\nabla \mathbf{x}_\lambda|^2 + \beta} d\Omega \tag{13}$$

For this project I used Newton Method with Conjugate Gradients (CG) based on the algorithm found in [3]. See section 9.2 for details about implementation.

3 Statistical Based Diagnostics

The use of residual diagnostics for choosing regularization parameters was demonstrated by Bert Rust and Dianne O'Leary [12] as an effective tool to determine plausible regularized solutions. In this project we will use the following three diagnostics to generate a range of plausible regularization parameters.
Assuming that the errors in the data are independently identically normally distributed with mean zero and variance one, the discretized linear regression model is

$$\mathbf{b} = \mathbf{A}\mathbf{x}^{*} + \epsilon \tag{14}$$

where

$$\epsilon \sim N(\mathbf{0}, \mathbf{I_m}) \tag{15}$$

Now if we have an estimate $\tilde{\mathbf{x}}$ of $\mathbf{x}^{*}$ then the residual vector

$$\tilde{\mathbf{r}} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}} \tag{16}$$

for a plausible $\tilde{\mathbf{x}}$ should be a sample from the distribution from which ϵ is drawn since our linear regression model can be written

$$\epsilon = \mathbf{b} - \mathbf{A}\mathbf{x}^{*} \tag{17}$$

This characteristic of the residual inspired three diagnostics [7]:

- Diagnostic 1. The residual norm squared should be within two standard deviations (within the 95% confidence interval) of the expected value of $\|\epsilon\|_{\mathbf{2}}^{\mathbf{2}}$ or
$$\|\tilde{\mathbf{r}}\|_2^2 \in [m - 2\sqrt{2m}, m + 2\sqrt{2m}]$$
where the expected value of $\|\epsilon\|_2^2$ is m and the variance is $2m$. This diagnostic is based on the Chi^2 distribution, where the sum of the squares of m i.i.d. standard normal random variables are Chi^2 distributed. Now if we have m samples and find the sum of squares of the sample then we expect that we should be within the 95% confidence interval if we have a normal sample.

- Diagnostic 2. The graph of the elements of the vector of the residual $\tilde{\mathbf{r}}$ should look like samples from the distribution $\epsilon \sim N(0, 1)$. Quantitatively this test is based on the goodness of fit of the normal curve to the histogram of the elements which is $\tilde{r}_i$. One way to test the goodness of fit is the Chi^2 test where the null hypothesis that our sample is standard normal distributed with significance 0.05 or in other words is our sample normal distributed with 95% confidence. Another significance test is the Fisher test which has been shown to be more acurate because it is an exact test compared to the previous test where the distribution of Chi^2 has to be approximated.

- Diagnostic 3. If we consider the elements of ϵ and $\tilde{\mathbf{r}}$ as time series with index $j = 1, ..., n$ where $\epsilon_j \sim N(0, 1)$ forming a white noise series then $\mathbf{r}$ should also be a white noise series. To measure this quantitatively one needs to find the cumulative periodogram of the residual time-series.

To find the the cumulative periodogram one first needs to find the spectral density of the series by transforming it into its frequency domain by taking the Fourier transform given by

$$x_t = \frac{a_o}{2} + \sum_{k=1}^{m}(a_k \cos(w_k t) + b_k \sin(w_k t))$$

where x_t has n observations where $m = \frac{n-1}{2}$ and the frequency $w_k = \frac{2\pi k}{n}$. And the spectral density or periodogram defined as

$$I(w_k) = \frac{n}{2}(a_k^2 + b_k^2) \tag{18}$$

If we look at the spectral density of white noise we would expect the magnitude to be equal distribution over every frequency. And if we take the cumulative sum over the spectral density over each frequency we would assume that we would expect a straight line between 0 and 1. The cumulative periodogram is given by

$$C_k = \frac{\sum_{(j=0)}^{k} I_n(w_j)}{\sum_{(j=0)}^{m} I_n(w_j)} \tag{19}$$

To test to see if our residual falls within 95% of the standard normal distribution we simple plot the cummulative periodogram and find the 95% interval. The 95% confidence interval is giving by plus (upperbound) or minus (lower bound) $1.36/\sqrt{(m-1)}$[6] to the linear line from 0 to 1, where this relation is said to hold for $m - 1 > 30$. .

4 Choosing an initial regularization paramter

The method of generalized cross-validation (GCV) is to minimize the GCV function

$$G(\lambda) = \sum_{k=1}^{m}[b_k - (\mathbf{A}\tilde{\mathbf{x}}_{\lambda}^{(k)})_k]^2 \tag{20}$$

where the $\tilde{x}^{(k)}$ is the estimate when the k^{th} measurement of $\mathbf{b}$ is omitted. This method will be used to choose an intial λ for each of the regularization methods [7].
This can be greatly simplified for the Tikhonov and the TSVD regularization method. Below are the simplified forms of GCV.

- *GCV for TSVD*: $G(k) = \frac{1}{(m-k)^2}\sum_{i=k+1}^{m}(\mathbf{u}_i^T\mathbf{b})^2$, which has been implemented in *RestoreTool*.
- *GCV for Tikhonov*: $G(\lambda) = \frac{\sum_{i=1}^{m}(\frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i^2+\lambda^2})^2}{\sum_{i=1}^{m}(\frac{1}{\sigma_i^2+\lambda^2})^2}$, (*RestoreTool*).
- *Parameter selection for TV*: As part of the second semester goal, GCV for TV regularization will be implemented.

5 Frontend

The idea of having a frontend or Graphical User Interface (GUI) for this project is to provide an interactive platform for users who may have invaluable information about the image (example. the doctor looking a CT image) to change the regularization parameters to see the affect and also given the statistical diagnostic to validate that they are not losing too much information by under or over regularizing the image that may look good to the eye (Figure 2). The user should not need to have knowlegable about the implementation to effectively use the interface.

6 Software

Below is a discription of the pieces that will form the software package for this project. The software package will be written with the intent of running jobs in parallel using Matlab Parallel Computing toolbox.

Frontend Software

- Graphical User Interface (GUI) using Matlab's GUI toolbox

Backend Software

- Regularization method
  - Regularization methods Tikhonov and TSVD from *RestoreTool* [8]
  - Total Variation regularization method (new code as part of this project)
- Method for initial parameter selection
  - Generalized Cross-Validation (GCV) in *RestoreTool* for regularization methods included.
  - GCV for Total Variation (new code as part of this project)
- Validate candidate solutions using statistical diagnostics (Section 2.2)
  - Apply statistical diagnostics (modify existing code).
- Tikhonov and TSVD for images- already exist (part of RestoreTool [8])
- Generalized cross validation (GCV) for Tikhonov and TSVD regularization - already exist (part of RestoreTool [5])
- TV regularization to be added to RestoreTool [8]- to be built by Brianna Cash in Matlab using RestoreTool
- GCV for TV regularization-to be adapted from RestoreTool[8] GCVforSVD by Brianna Cash
- Statistical diagnostics - based on *checkperiod.m* by Dianne O'Leary
- GUI interface- to be built by Brianna Cash using Matlab GUI tool box
- The software package will be written with the intent of running jobs in parallel using the MATLAB Parallel Computing tool box.

7 Hardware

For the initial stages the software will be designed to run on a modern desktop PC or laptop with no special hardware requirements. For the parallelization stage I will use available computers on either the computer science or mathnetwork that have multi-core machines available.

8 Databases

For this project I will use artificial generated images and PSF functions for development as well as some initial testing of the software. These data sets will be created to any size that is both menagebale and required. For testing and validation I will use the five test data sets in *RestoreTool* which include a 256×256 image and at least one test PSF.

9 Implementation and Results

All the regularization methods have been initially tested on the same set of test images using the same PSF function and boundary conditions and noise levels¿ Currently the PSF function is Gaussian with zero boundary conditions but they are all implemented in a way the it is easily changed as the project progresses.

9.1 Tikhonov and TSVD with GCV for initial parameter selection

- Tikhonov is implemented using the Matlab functions *Tikhonov.m* where the initial parameter is found using *GCVfun.m* and Matlab function *fminbnd* finds the minimum.
  - Note that there was a small mistake in *RestoreTool*s implementation using *fminbnd* within *Tikhonov.m* that I fixed, properly documented in the function and files. I plan to submit this to the authors.
- TSVD is implemented using the Matlab functions *TSVD.m* where the initial parameter is found using *GCVforSVD.m*.

As with all methods in *RestoreTool*, Tikhonov and TSVD regularization method utilizes the *psfMatrix* class to make the implementation more efficient by not storing the $\mathbf{A}$ explicitly but instead creating a *psfMatrix* object is used to store the Point Spread Function (PSF), a function that describes how a single point is blurred or distorted, and the boundary conditions from which the blurring matrix can be formed. Operations like matrix vector multiplication, SVD, as well as operations on $\mathbf{A^T}$ make up the *psfMatrix* class. This greatly reduces the amount of storage because typically the PSF is much smaller than the blurring matrix depending on the amount of distortion/blur.

9.2 Total Variation (TV) Regularization Method

The TV regularization method was implemented using Newton Method with Conjugate Gradients (CG)[3]. The TV term was first discretized using forward

differencing with Neumann Boundary contions, the value of the derivative of the solution is taken at the boundary [3].
Where the discrete formulation of TV term becomes

$$\min \frac{1}{2}\|\mathbf{Ax}-\mathbf{b}\|_2^2 + \lambda \sum_{i=1}^{n} \sqrt{\|\mathbf{D}_i^T\mathbf{x}\|^2+\beta} \tag{21}$$

where

$$\mathbf{D}_i^T\mathbf{x} = (x_{i+1}-x_i, x_{i+n_h}-x_i) \tag{22}$$

To implement Newton's method the first order condition was found to be

$$\mathbf{g}(\mathbf{x}) = \mathbf{A}^T\mathbf{Ax}-\mathbf{b} \ + \lambda \sum_{i=1}^{n} \frac{\mathbf{D}_i\mathbf{D}_i^T\mathbf{x}}{\sqrt{\|\mathbf{D}_i^T\mathbf{x}\|^2+\beta}} = 0 \tag{23}$$

and the Hessian was found to be

$$\mathbf{H}(\mathbf{x}) = \mathbf{A}^T\mathbf{A} + \lambda \sum_{i=1}^{n} \frac{\mathbf{D}_i(I - \frac{\mathbf{D}_i^T\mathbf{x}\mathbf{x}^T\mathbf{D}_i}{\|\mathbf{D}_i^T\mathbf{x}\|^2+\beta})\mathbf{D}_i^T\mathbf{x}}{\sqrt{\|\mathbf{D}_i^T\mathbf{x}\|^2+\beta}} \tag{24}$$

9.3 Pseudo-Code for Newton Method with CG

Newton Method:

1. Initialize: $\mathbf{x}_0$ an initial approximation to the solution, set $k = 0$
2. If $\mathbf{x}_k$ is "good enough", terminate
3. Solve $\mathbf{H}(\mathbf{x})\mathbf{p_k} = -\mathbf{g}(\mathbf{x})$ where $\mathbf{p}$ using conjugate gradient (CG) to find the Newton direction p_k
4. Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{p}_k$ where α_k is determined by a linesearch.
5. Set $k = k+1$

Stopping condition ("good enough") for Newton Step: $\|\mathbf{g}(\mathbf{x}_k)\|/\|\mathbf{g}(\mathbf{x}_0)\| < 10^{-4}$

CG:

1. Let $\mathbf{r} = -\mathbf{g}(\mathbf{x}) - \mathbf{H}(\mathbf{x})\mathbf{p_k}$, $\mathbf{q} = \mathbf{r}$, $\rho = \|\mathbf{r}\|$, $\gamma = \rho$, $k = 0$
2. For $k = 0$, 1, until $\rho = \|\mathbf{r}\|/\rho < tol$
3. $\alpha = \frac{rho}{\mathbf{q^T H(x) q}}$
4. $\mathbf{p} = \mathbf{p} + \alpha\mathbf{q}$
5. $\mathbf{r} = \mathbf{r} - \alpha\mathbf{H}(\mathbf{x})\mathbf{q}$
6. $\hat{\gamma} = \|\mathbf{r}\|$
7. $\beta = \frac{\hat{\gamma}}{\gamma}$, $\gamma = \hat{\gamma}$
8. $\mathbf{q} = \mathbf{r} + \beta\mathbf{q}$

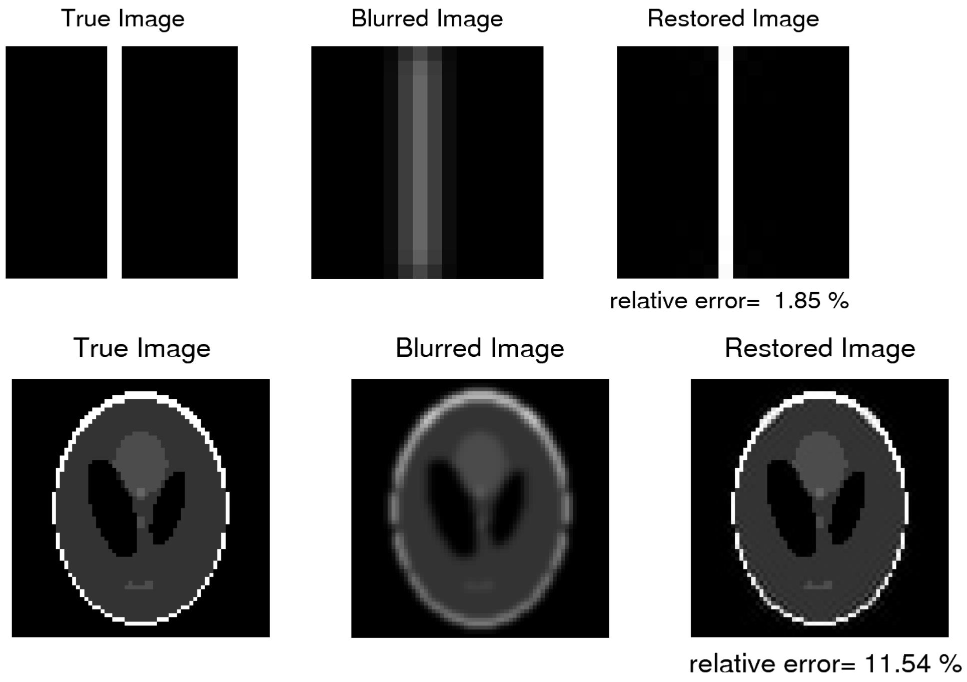


Figure 3: Results on Newton-CG for two images, top is a 16×16 generated in Matlab and the bottom image is 32×32 Modified Shepp-Logan generated in Matlab

9. Set $k = k + 1$

Tolerance for CG: $tol = 0.1$ when $k = 0$ and $tol = \min(0.1, 0.9\|\mathbf{g}(\mathbf{x}_k)\|^2/\|\mathbf{g}(\mathbf{x}_{k-1})\|^2)$ are the suggested conditions in [3].

See *figure 3* for basic results of the algorithm on two images.

9.4 Efficient Implementation

In order to strive for efficient implementation, there is an attempt to make the algorithm as low storage in order to work with larger images while balancing computation time. In the current implementation the Hessian is not stored explicitly which is an advantage of using CG. Instead we use a function to form

$$\mathbf{Hv} = \mathbf{H}(\mathbf{x})\mathbf{v}$$

where $\mathbf{v}$ is an arbitrary vector. The blurring matrix is also not stored explicitly. For smaller images (smaller than 32×32) a sparse representation, this was

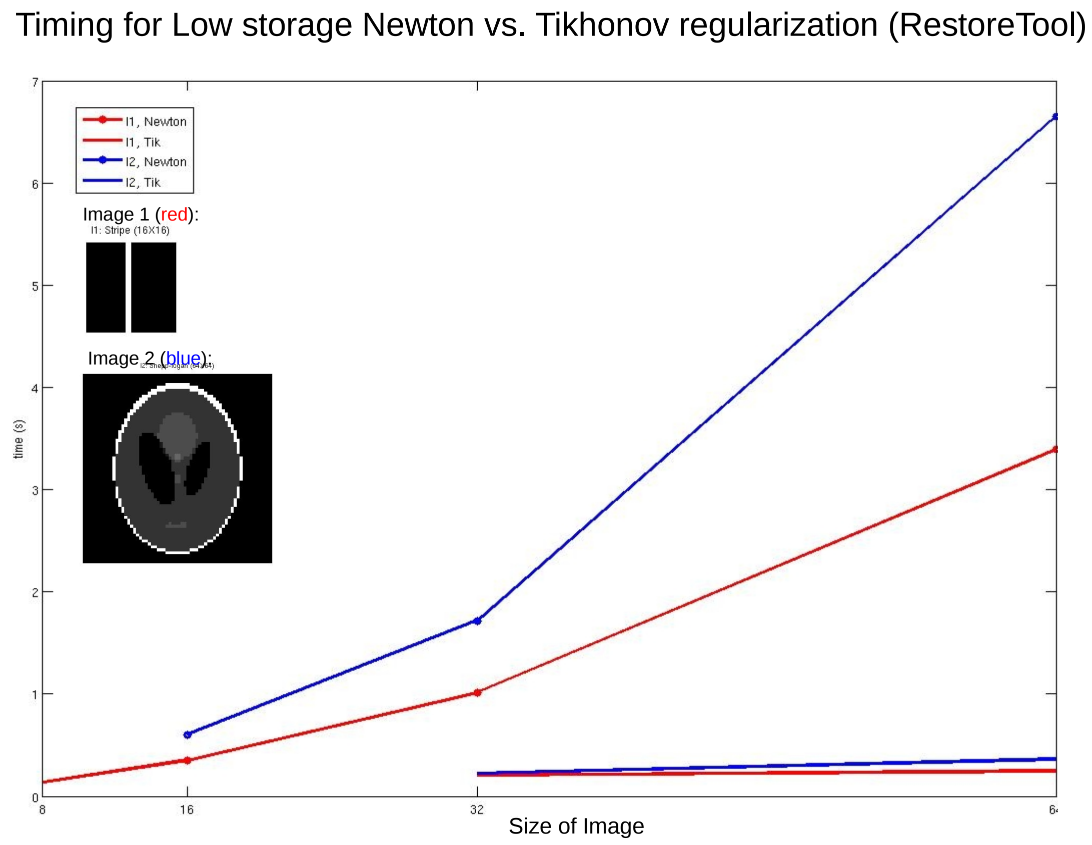


Figure 4: Results on timing the Newton-CG timing for two images compared to timing of the direct Tikhonov regularization method

chosen because the RestoreTool class *psfMatrix* does not seem to work with smaller images. For larger images *psfMatrix* is used.
In addition to low-storage the method should be fast enough to work with the GUI interface; see Figure 4 for initial timing results. As part of the second semester goals efforts will be made to optimize the implementation.

9.5 Statistical Diagnostics

- Diagnostic 1. Is directly implemented where $||\tilde{\mathbf{r}}||_2^2$ is checked to see if it falls within the bounds $[m - 2\sqrt{2m}, m + 2\sqrt{2m}]$ for each solution (for different parameters).

- Diagnostic 2. This diagnostic is both implemented visually using the Matlab plotting function *hist* and the the null hypothesis with 5% significance is tested using the Matlab function *chi2gof(resid,'cdf',@normcdf)* where it assumes there are 10 bins.

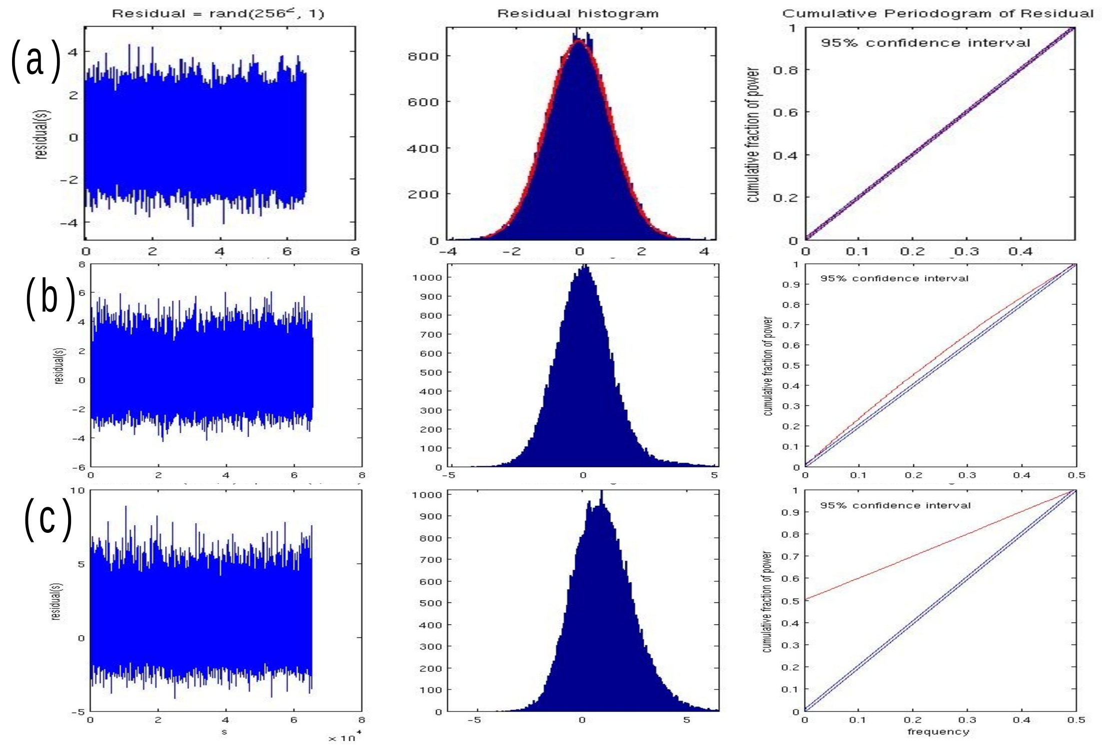


Figure: Plots of residual, histogram of residual, cumulative periodogram:
(a) Normal distributed, (b) Normal plus Poisson (1), (c) Normal plus features

Figure 5: Plot of the residual, histogram of the residual and the cumulative for three different residuals

- Diagnostic 3. The cumulative periodogram was found by first using the Fast Fourier Transform (the Matlab function *fft*) to approximate the Fourier transform, then taking the square of each element, and finally finding the cumulative sum. The diagnostic measured by finding the number of points that fell outside the 95% interval over each frequency. If less than 5% fell outside then the diagnostic was satisfied. The implementation follows the implementation of *checkperiod.m* by Dianne O'Leary which follows a Fortran program by Bert Rust.

9.6 Frontend

The GUI interface was implemented using Matlabs GUI interface toolbox. The GUI is built by first creating a figure window using a figure editor. Within the editor the user has a number components (fields, axes, sliders, dropbox, input fields, push buttons....) to choice from. You can add user-operated controls, dynamic and static components. Once the figure has the desired features Matlab generates the basic code that if run generates the figure you built but it does not do any computations, i.e. you can move the slider but nothing changes (Figure 6). The componments that are user-operated or dyanamic must have

a *Callback function*, within this function you can designate the response to the user-generatated event by coding what the callbacks should perform giving the user input. Each component has a tag and a handle structure is created to store all relavent information about the component (i.e. for a the slider has a handle for max value, min value and current value). Figure 7 has a screen shot of an example of the *Callback function* for the parameter selection tool.
In Figure 8 is one of the GUI I am using as part of this project, in this GUI a user first chooses an test image and regularization method (left corner). Once they press the push button "compute" for that given method the GCV selected parameter is computed and the regularization solution is found, the GUI then displays information about the image (right corner), as well as the blurred image, deblurred image (regularization solution), and the true image. The user can then adjust the slide bar (which is initial set to the GCV selected parameter) and in realtime the new solutions are computed and displayed. In the bottom right corner the user can see whether the first diagnostic is satisfied and change the slider till the solution satisfies the diagnostic. In Figure 9 is a screen shoot of the second GUI that I'm using, it is a lot like the previously described GUI but it inlcudes all three diagnostics including the visual interpretation of diagnostic 2 and 3.

10 Validation and Testing

10.1 Initial Validation of Total Variation

The implementation was programmed modularly so that each piece (Newton Step, CG, function evaluations) can be validated individually.

- CG was validated for small $\mathbf{Ax} = \mathbf{b}$ test problems where the results could be verified.
- Implementation the minimization function $f(\mathbf{x})$, gradient $\mathbf{g}(\mathbf{x})$, and Hessian times a vector $\mathbf{Hv}(\mathbf{x}, \mathbf{v})$ were verified.

The Newton Method (without CG) was also implemented and results were compared to the low-storage Newton method with CG and linesearch. Binary test images without noise and verified the results were close to the true image (see Figure 4).

10.2 Validation of Diagnostics

The validation of the diagnostics was done experimentally based on expected statistical results. Each of the diagnostics were tested first with stardard normal independent, identically distribute (i.i.d.) samples to confirm that that each of the diagnostics is satisfied at least 95% of the time (see table below).

From the table it is seen for 1000 different standard normal i.i.d. sample the three diagnostics and the Fisher normality test (not described) tell us for these 1000 samples, 95% fall satisfy the diagnostics. Then if we perturb the standard normal samples by periodically adding artifacts or adding small faction of poisson distributed samples ones sees that for the diagnostic 2 and the Fisher Test (which should be equivalent) do a very good job of determining that the sample

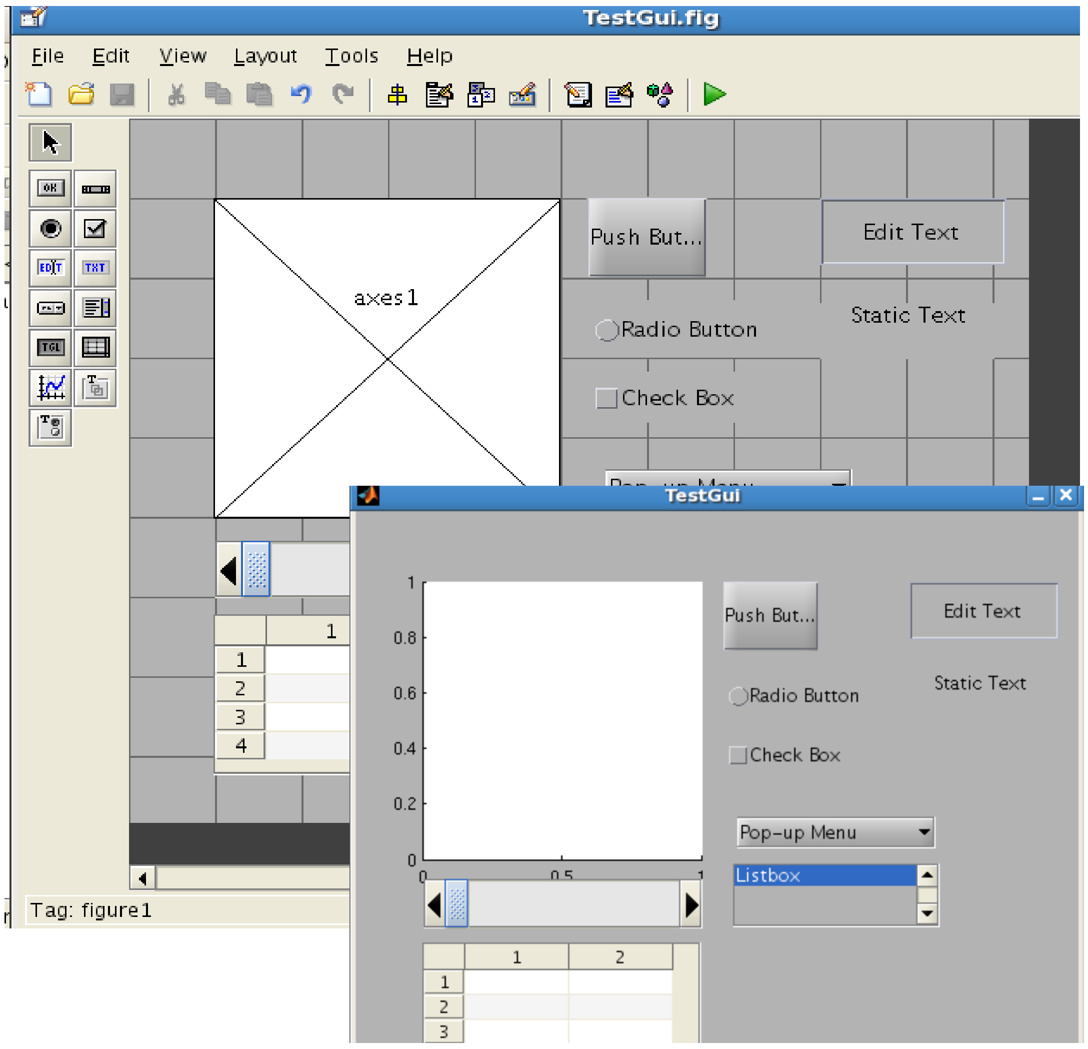
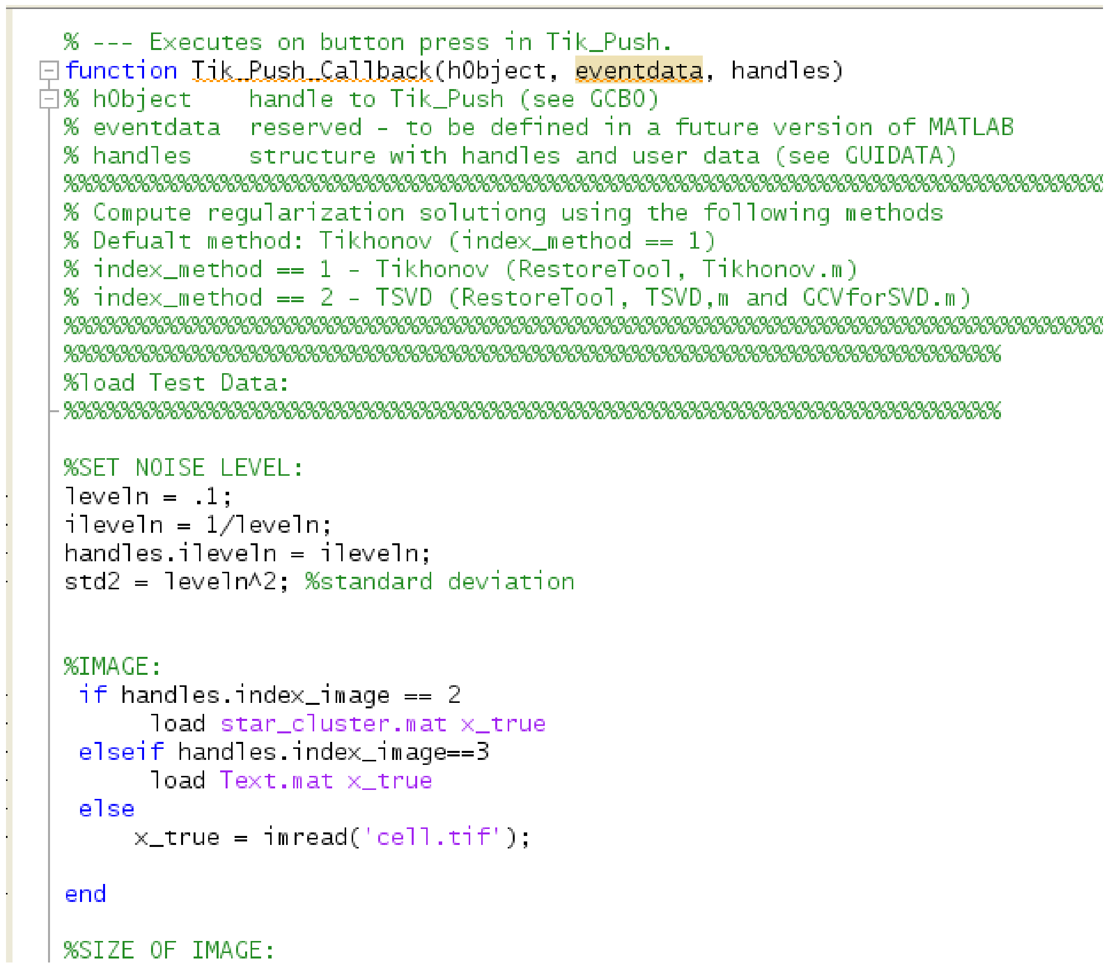
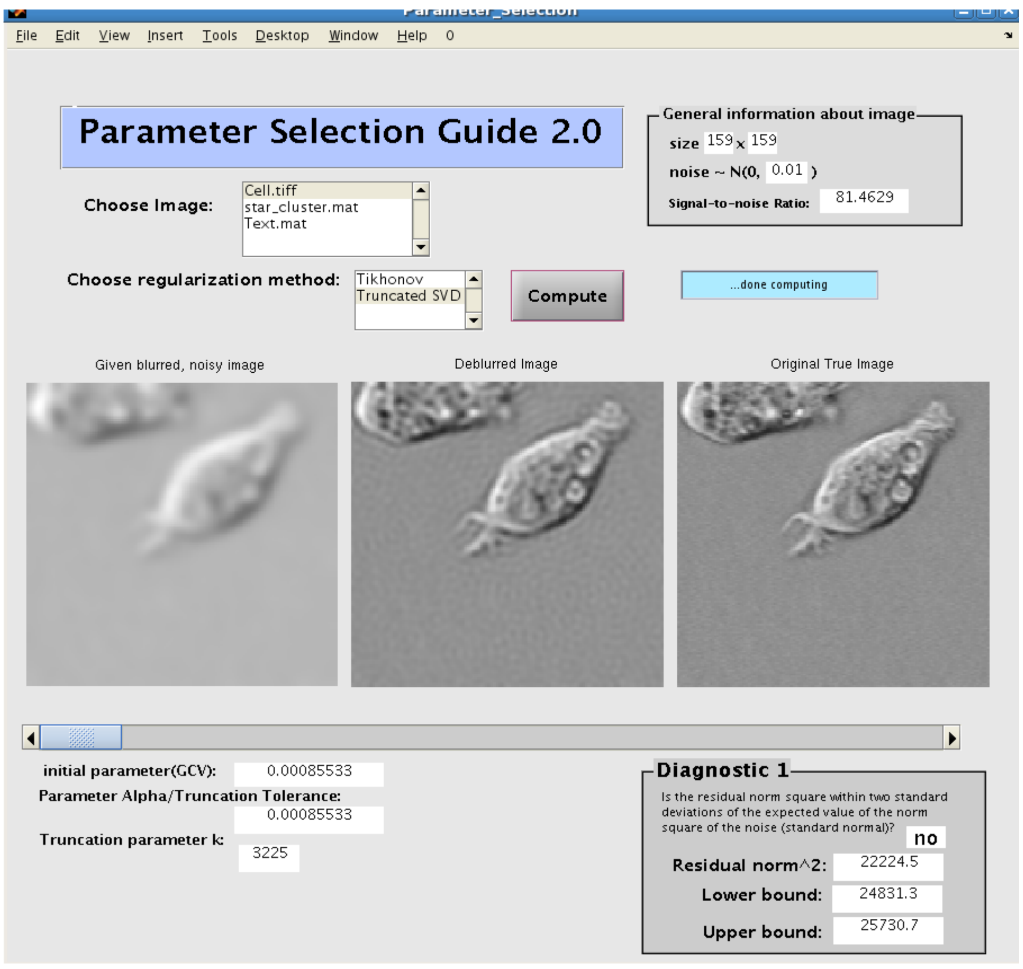


Figure 6: The image in the back is of the figure being using using Matlabs toolbox, the figure in front is the result when matlab creates a GUI from the figure.



```
% --- Executes on button press in Tik_Push.
function Tik_Push_Callback(hObject, eventdata, handles)
% hObject    handle to Tik_Push (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compute regularization solutiong using the following methods
% Defualt method: Tikhonov (index_method == 1)
% index_method == 1 - Tikhonov (RestoreTool, Tikhonov.m)
% index_method == 2 - TSVD (RestoreTool, TSVD,m and GCVforSVD.m)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%load Test Data:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%SET NOISE LEVEL:
leveln = .1;
ileveln = 1/leveln;
handles.ileveln = ileveln;
std2 = leveln^2; %standard deviation


%IMAGE:
 if handles.index_image == 2
      load star_cluster.mat x_true
 elseif handles.index_image==3
      load Text.mat x_true
 else
     x_true = imread('cell.tif');

end

%SIZE OF IMAGE:
```

Figure 7: Snipit of a *callback fucntion* taken from the parameter selection GUI


Parameter_Selection
File Edit View Insert Tools Desktop Window Help
Parameter Selection Guide 2.0
Compute
...done computing
Given blurred, noisy image
Deblurred Image
Original True Image

General information about image
size 159 x 159
noise ~ N(0, 0.01)
Signal-to-noise Ratio: 81.4629

Choose Image: Cell.tiff, star_cluster.mat, Text.mat

Choose regularization method: Tikhonov, Truncated SVD

initial parameter(GCV): 0.00085533

Parameter Alpha/Truncation Tolerance: 0.00085533

Truncation parameter k: 3225

Diagnostic 1

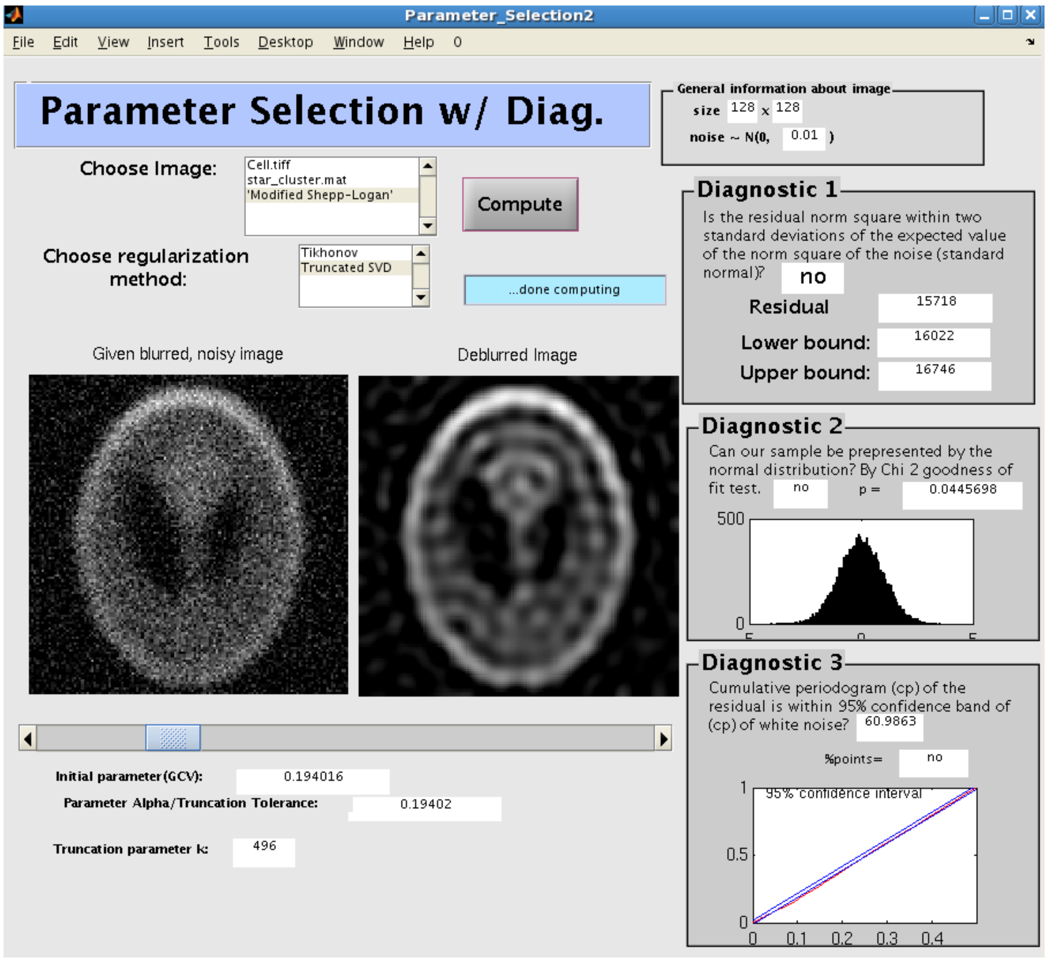
Is the residual norm square within two standard deviations of the expected value of the norm square of the noise (standard normal)? no

Residual norm^2: 22224.5

Lower bound: 24831.3

Upper bound: 25730.7

Figure 8: GUI for parameter selction with diagnostics 1 and the true image along with the blurred and deblurred image



Parameter Selection w/ Diag.

Choose Image: Cell.tiff / star_cluster.mat / 'Modified Shepp-Logan'

Choose regularization method: Tikhonov / Truncated SVD

...done computing

Initial parameter(GCV): 0.194016

Parameter Alpha/Truncation Tolerance: 0.19402

Truncation parameter k: 496

General information about image

size 128 x 128

noise ~ N(0, 0.01)

Diagnostic 1

Is the residual norm square within two standard deviations of the expected value of the norm square of the noise (standard normal)? no

Residual 15718

Lower bound: 16022

Upper bound: 16746

Diagnostic 2

Can our sample be prepresented by the normal distribution? By Chi 2 goodness of fit test. no p = 0.0445698

Diagnostic 3

Cumulative periodogram (cp) of the residual is within 95% confidence band of (cp) of white noise? 60.9863

%points= no

Figure 9: GUI for parameter selction with all three diagnostics

Residual	Diag. 1	Diag. 2	Diag. 3	Fisher
$r_n \sim N(0,1)$	51	46	14	48
$r_n + I(i)$	950	999	539	1000
$r_n + .05 * r_p \; r_p \sim pois(1)$	156	1000	149	1000

Table 1: For 1000 runs, number of times the Diagnostics are **NOT** satisfied. Where $I(i) = 1$ if $(i-1)mod(100) = 0$ or $(i-2)mod(100) = 0$ and $I(i) = 0$ otherwise.

is no longer stardard normally distributed which is what we expected giving the pertubation.

11 Remaining Schedule

- Jan 15-Feb15 - Implement Total Variation in RestoreTool Framework
- Feb 15-Feb 28 - Validate Total Variation tool
- Mar 1 - Mar 15 - Add Total Variation tool to GUI
- Mar 15-April 30 - Optimize Total Variation using tool using parallel toolbox in Matlab (if available), if not look at other ways to implement the software package using other optimization the package as whole available in matlab.
- May - Prepare final report and presentation

12 Milestones

12.1 Completed Milestones

- A basic GUI to use existing tools in RestoreTool [5] or Regularization Tool [3]
- Add validated periodogram diagnostics to RestoreTool
- GUI that determines a range of plausible regions using the periodogram diagnostics
- Outline of Total Variation regularization algorithm and basic implementation in Matlab.
- Deliver mid-year report and presentation

12.2 Remaining Milestones

- Feb 1 - Total Variation (TV) regularization in RestoreTool Framework
- Feb 15 - Generalized Cross Validation (GCV) for Total Variation tool
- Feb 30 - Add validated TV too with GCV to the GUI

- April 1 - Optimize software package finished (using parallel toolbox in Matlab if available)
- May 1 - Poster for SIAM Conference on Image Science to be presented May 20-22
- May 15- Deliver final presentation and report

13 References

1. R. Acar and C.R. Vogel. Analysis of Bounded Variation Penalty Methods for Ill-Posed Problems. *Inverse Problems.* 10: 1217-1229, 1994.
2. Tony F. Chan and Jianhong Shen. Image Processing and Analysis. *SIAM*, Philadelphia, PA, 2005.
3. Tony F. Chan, Gene H. Golub and Pep Mulet. A nonlinear primal-dual method for total variation-based image restoration. *Lecture Notes in Control and Information Sciences*, Vol.219, pp.241-251, 1996.
4. Per Christian Hansen. Regularization Toolbox, http://www2.imm.dtu.dk/ pch/Regutools/regutools.htı
5. Per Christian Hansen. Rank-Deficient and Discrete Ill-Posed Problems: *Numerical Aspects of Linear Inversion. SIAM*, Philadelphia, PA, 1998.
6. Wayne A. Fuller. Introduction to Statistical Time Series, *Wiley-Interscience*, New York, NY, 1996.
7. C.T. Kelly. Iterative Methods for Linear and Nonlinear Equations. *SIAM*, Philadelphia, PA, 1995.
8. James G. Nagy, RestoreTool, http://www.mathcs.emory.edu/ nagy/RestoreTools/
9. James G. Nagy, K. Palmer and L. Perrone. Iterative Methods for Image Deblurring: A Matlab Object Oriented Approach. *Numerical Algorithms.* 36: 73-93, 2004.
10. Stephen G. Nash. Linear and Nonlinear Programming. *McGraw-Hill*, 1996.
11. Dianne P. 0'Leary. Scientific Computing with Case Studies. *SIAM*, Philadelphia, PA, 2009.
12. Bert W. Rust and Dianne P. O'Leary. Residual programs for choosing regularization parameters for ill-posed problems. *Inverse Problems*, 24:034005 (30 pages), 2008. Invited Paper
13. Bert W. Rust. Parameter selection for constrained solutions to ill-posed problems. *Computing Science and Statistics*, 32:333-347, 2000.
14. C. Vogel and M. Oman. Iterative methods for total variation denoising, *SIAM Journal on Scientific Computing*, Vol.17, pp.227 238, Jan.1996.