MMAT5270 Introduction to Inverse Problems

Assignment 2

1 Problems:

5.1 The Discrepancy Principle

The purpose of this exercise is to illustrate the sensitivity of the discrepancy principle to variations of the estimate of the error norm. First generate the `shaw` test problem for $n = 100$, and add Gaussian noise with standard deviation $\eta = 10^{-3}$ to the right-hand side.

Use the discrepancy principle to compute the Tikhonov solution. The discrepancy principle is implemented in the MATLAB function *discrep* from *Regularization Tools*. This function may occasionally have convergence problems; if this happens, then create a new noise realization (or change η slightly) and try again.

You should first use the "safety factor" $\nu_{dp} = 1$. As the "right-hand side" $\|e\|_2$ in (5.6), try to use both the norm estimate $\sqrt{n}\eta$ and the actual 2-norm $\|e\|_2$ of the perturbation vector e (perhaps try with different perturbations). Is there a significant difference in the results?

Still with the "safety factor" $\nu_{dp} = 1$, use several values of $\sqrt{n}\eta$ and/or $\|e\|_2$ that are slightly too large and slightly too small; this simulates a situation where only a rough estimate is available. For example, scale $\sqrt{n}\eta$ and/or $\|e\|_2$ up and down by 5-10%. How sensitive is the regularized solution to overestimates and underestimates of the noise level?

Finally, repeat the experiments with the "safety factor" $\nu_{dp} = 2$, and comment on the new results.

Solution.

Using the "safety factor" $\nu_{dp} = 1$:

```
[A,b,x] = shaw (100);
[U,s,V] = csvd (A);
e = 1e-3*randn (size(b));
b = b + e;
x_lambda = discrep (U,s,V,b,norm(e));
figure(1)
plot ([x,x_lambda]);

x_lambda2 = discrep (U,s,V,b,sqrt(100)*1e-3);
figure(2)
plot ([x,x_lambda2]);

x_lambda = discrep (U,s,V,b,norm(e)*1.1);
figure(3)
plot ([x,x_lambda]);

x_lambda = discrep (U,s,V,b,norm(e)*0.9);
figure(4)
plot ([x,x_lambda]);

x_lambda2 = discrep (U,s,V,b,sqrt(100)*1e-3*1.1);
figure(5)
plot ([x,x_lambda2]);
```

```
x_lambda2 = discrep (U,s,V,b,sqrt(100)*1e-3*0.9);
figure(6)
plot ([x,x_lambda2]);
```

Using the "safety factor" $\nu_{dp} = 2$:

```
[A,b,x] = shaw (100);
[U,s,V] = csvd (A);
e = 1e-3*randn (size(b));
b = b + e;
x_lambda = discrep (U,s,V,b,2*norm(e));
figure(1)
plot ([x,x_lambda]);

x_lambda2 = discrep (U,s,V,b,2*sqrt(100)*1e-3);
figure(2)
plot ([x,x_lambda2]);

x_lambda = discrep (U,s,V,b,2*norm(e)*1.1);
figure(3)
plot ([x,x_lambda]);

x_lambda = discrep (U,s,V,b,2*norm(e)*0.9);
figure(4)
plot ([x,x_lambda]);

x_lambda2 = discrep (U,s,V,b,2*sqrt(100)*1e-3*1.1);
figure(5)
plot ([x,x_lambda2]);

x_lambda2 = discrep (U,s,V,b,2*sqrt(100)*1e-3*0.9);
figure(6)
plot ([x,x_lambda2]);
```

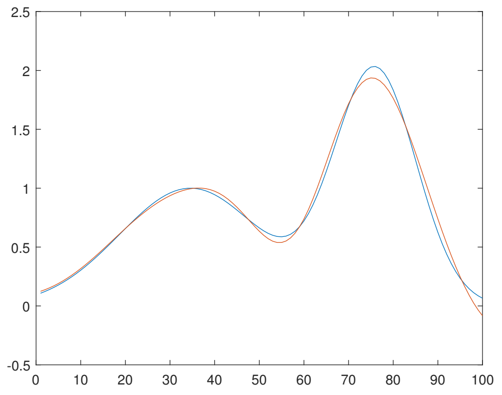

Figure 1: Using the actual 2-norm $\|e\|_2$

Figure 1,2,3 and 5 show that the results are similar if we use the norm estimate $\sqrt{n}\eta$ and the actual 2-norm $\|e\|_2$ of the perturbation vector e or scale $\sqrt{n}\eta$ and/or $\|e\|_2$ up by 5-10%.

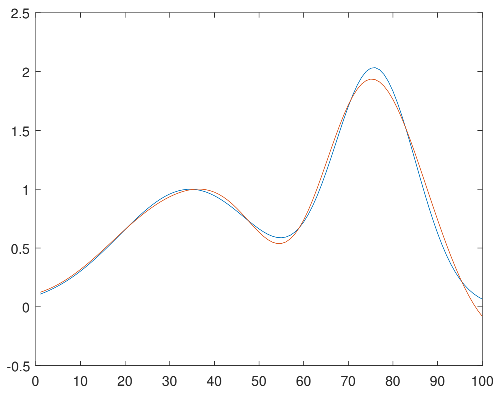

Figure 2: Using the norm estimate $\sqrt{n}\eta$

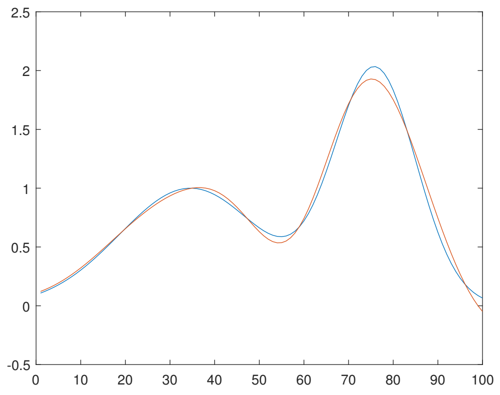

Figure 3: Scaling $\|e\|_2$ up by 10%

Figure 4,6 show that regularized solution is sensitive to underestimates of the noise level. If we use the "safety factor" $\nu_{dp} = 2$, we can get the same figure as figure 1.

$\square$

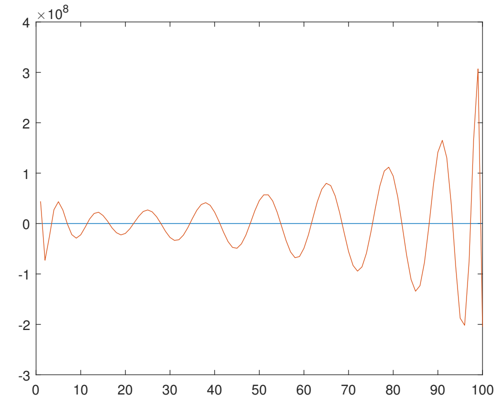

Figure 4: Scaling $\|e\|_2$ down by 10%

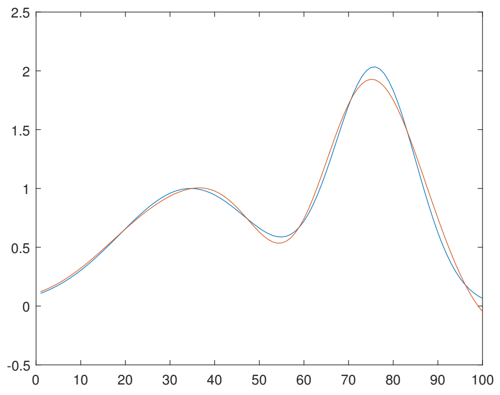

Figure 5: Scaling $\sqrt{n}\eta$ up by 10%

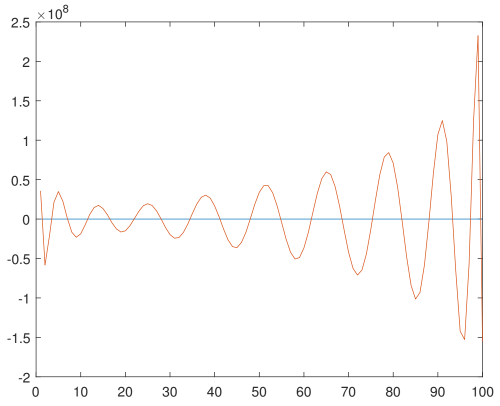

Figure 6: Scaling $\sqrt{n}\eta$ down by 10%

5.2 The GCV and L-curve Methods

This exercise illustrates the use of the GCV and L-curve methods for choosing the regularization parameter, and we compare these methods experimentally. As part of this comparison we investigate how robust-or reliable-the methods are, i.e., how often they produce a regularization parameter close to the optimal one. We use the shaw test problem and the parameter-choice functions `gcv` and `l_curve` from Regularization Tools.

Plot the GCV function for, say, 5 or 10 different perturbations with the same η, and note the general behavior of the GCV function. Is the minimum always at the transition region between the flat part and the more vertical part (cf. Figure 5.6)?

Use the L-curve criterion to compute the regularization parameter for the same 5 or 10 perturbations as above. Does the regularization parameter computed by means of `l_curve` always correspond to a solution near the corner (cf. Figure 5.4)?

For a fixed perturbation of the right-hand side, compute the regularization error and the perturbation error for Tikhonov's method, for a range of regularization parameters λ. Plot the norms of the two error components Δx_{bias} and Δx_{pert}, and relate the behavior of this plot to the two regularization parameters found by the GCV method and the L-curve criterion.

Solution.

For the L-curve criterion, we can find the regularization parameter computed by means of `l_curve` always correspond to a solution near the corner. The norm of Δx_{pert} is larger than the norm of Δx_{bias} before the corner value of λ calculated by the L-curve.

```
[A,b,x] = shaw(100); err=1e-3*randn (size(b));
b = b + err; [U,s,V] = csvd (A);
figure(1);
lambda = gcv (U,s,b);
figure(2);
reg_corner = l_curve(U,s,b);

S=diag(s);
Si=zeros(size(S));
lambda=0.001;
Si=diag(s.^2./(s.^2.+lambda^2));
x_bias=V*(1-Si)*V'*x;
x_pert=V*Si*inv(S)*U'*err;
norm(x_bias)
norm(x_pert)
```

□

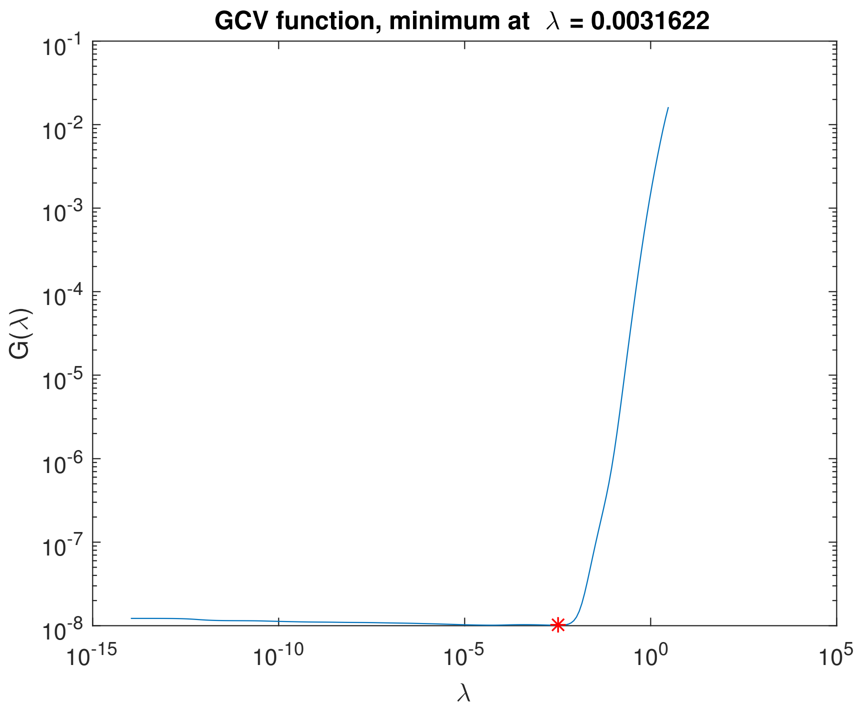

Figure 7: GCV

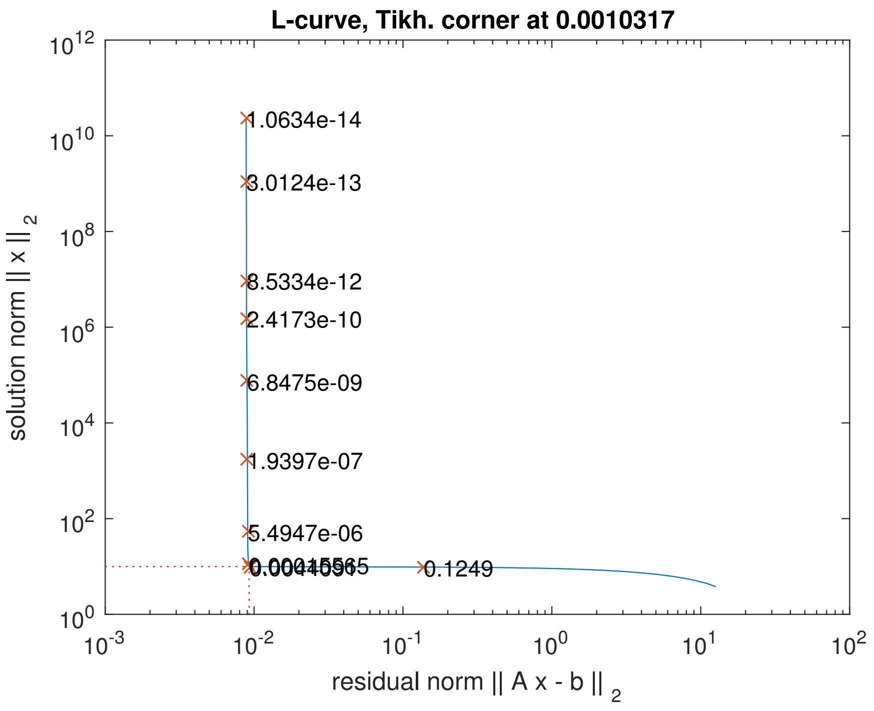

Figure 8: L-curve

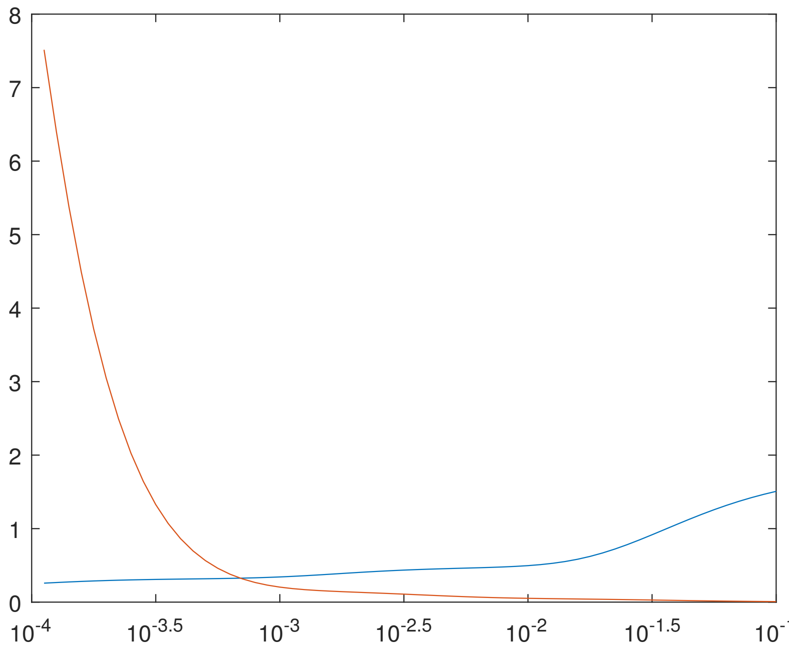

Figure 9: Δx_{bias} and Δx_{pert}

6.2 Landweber Iteration for Nonnegative Solutions

The purpose of this exercise is to illustrate the use of the augmented Landweber iteration (6.3) where the operator $\mathcal{P}$ represents a nonnegativity constraint. In other words, if x and $y = \mathcal{P}(x)$ are vectors, then

$$y_i = \begin{cases} x_i, & x_i \geq 0, \\ 0, & x_i < 0. \end{cases}$$

The test problem to be used here is a slightly modified version of the phillips problem from Regularization Tools:

```
[A,b,x] = phillips(n);
x = x - 0.5;
x(x<0) = 0;
b = A*x;
```

This produces an exact solution `x` which has a smooth "blob" in the middle, surrounded by zeros.

Implement two versions of Landweber's method: the standard version (6.1) and the augmented version (6.3) with the above nonnegativity projection $\mathcal{P}$. Use both methods to solve the modified `phillips` problem; you should perform several hundred iterations, and we suggest that you use a value of ω slightly smaller than $2/\sigma_1^2$. Compute the error histories for the two methods, study the regularized solutions, and comment on the results.

Solution.

```
n=100;
[A,b,x] = phillips(n);
x = x - 0.5;
x(x<0) = 0;
b = A*x;

x_iter=zeros(size(x));
[U,s,V] = csvd (A);
omega=2/s(1)^2-0.001;
k=500;
error_his=zeros(k);
for i=1:k
    x_iter=x_iter+omega*A'*(b-A*x_iter);
    error_his(i)=norm(x-x_iter)/norm(x);
end
plot(error_his)
hold on

x_iter=zeros(size(x));
[U,s,V] = csvd (A);
omega=2/s(1)^2-0.001;
k=500;
error_his=zeros(k,1);
for i=1:k
    x_iter=x_iter+omega*A'*(b-A*x_iter);
    x_iter(x_iter<0)=0;
    error_his(i)=norm(x-x_iter)/norm(x);
end
plot(error_his)
```

The error histories for the two methods show that the augmented version has a better result and converges faster.

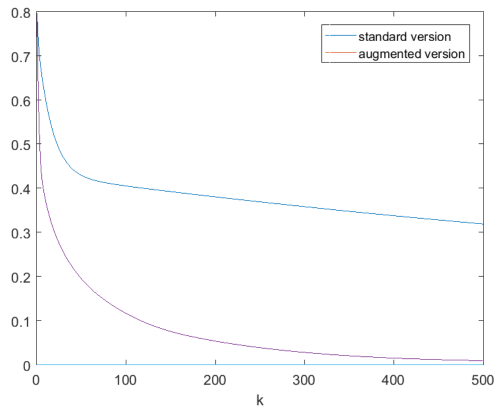

Figure 10: Error history

□

6.3 Illustration of the CGLS Algorithm

The purpose of this exercise is to illustrate the use of regularizing iterations in the form of the CGLS algorithm from Section 6.3.2. This algorithm is implemented in *Regularization Tools* as the function `cgls`. The exercise also previews material coming up in Section 7.5.

The model problem is a small image deblurring problem which is generated by means of the function `blur` from the same package. Both the exact image and the blurred image are $N \times N$, and they are represented by N^2-vectors by stacking their columns. Use `reshape(x,N,N)` to get from the "stacked" representation of the image to the image itself. To display the images, you can use the function `imagesc` together with the commands `axis image` and `colormap gray`.

Choose $N = 64$ and generate the test problem with `blur(N,10,1.4)` (the second parameter controls the sparsity of A, and the third parameter controls the amount of blurring). Plot the sharp and the blurred $N \times N$ images. At this stage, do not add noise to the problem. Perform a number of CGLS iterations; note that the CGLS iterates are returned as columns of the output matrix from `cgls`. Plot some of the iterates as images, and notice how the reconstruction gets sharper as the number of iterations increases.

Now add noise e to the blurred image with relative noise level $\|e\|_2/\|b\|_2 = 0.1$, and repeat the CGLS computations. You should notice that after a certain number of steps, the noise starts to dominate. This illustrates that the number of iterations indeed plays the role of the regularization parameter for the CGLS algorithm.

Solution.

No noise:

```
N=64;
[A,b,x] = blur(N,10,1.4);
x_image=reshape(x,N,N);
figure(1);
imagesc(x_image);
axis image;
colormap gray;
b_image=reshape(b,N,N);
figure(2);
imagesc(b_image);
```

```
axis image;
colormap gray;
k=15;
[X,rho,eta] = cgls (A,b,k);
error_his=zeros(k,1);
for i=1:k
    error_his(i)=norm(x-X(:,i))/norm(x);
end
x_image=reshape(X(:,end),N,N);
figure(3);
imagesc(x_image);
axis image;
colormap gray;
figure(4);
plot(error_his);
```

Adding noise e to the blurred image:

```
N=64;
[A,b,x] = blur(N,10,1.4);
err=1e-1*randn (size(b));
b = b + err;
x_image=reshape(x,N,N);
figure(1);
imagesc(x_image);
axis image;
colormap gray;
b_image=reshape(b,N,N);
figure(2);
imagesc(b_image);
axis image;
colormap gray;
k=15;
[X,rho,eta] = cgls (A,b,k);
error_his=zeros(k,1);
for i=1:k
    error_his(i)=norm(x-X(:,i))/norm(x);
end
x_image=reshape(X(:,end),N,N);
figure(3);
imagesc(x_image);
axis image;
colormap gray;
figure(4);
plot(error_his);
```

□

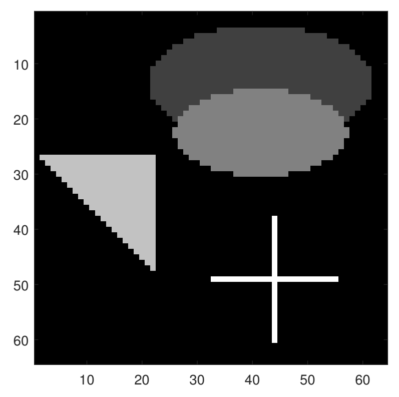

Figure 11: Sharp image

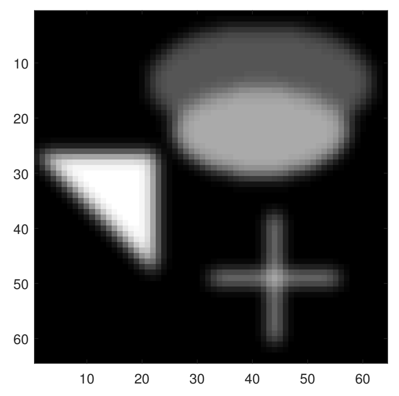

Figure 12: Blurred image

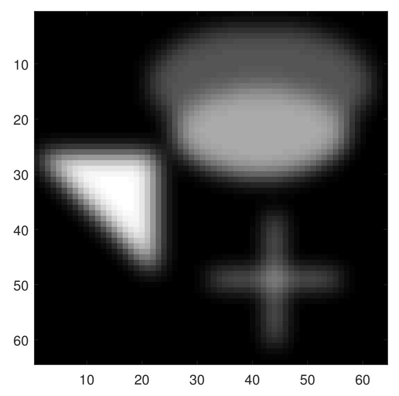

Figure 13: Perform 1 CGLS iteration

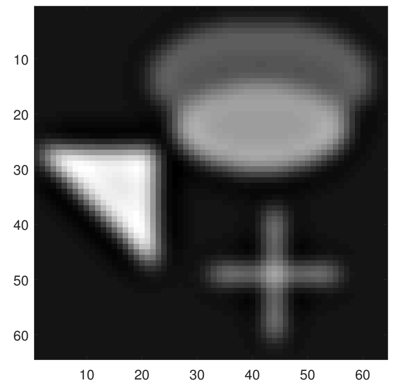

Figure 14: Perform 2 CGLS iterations

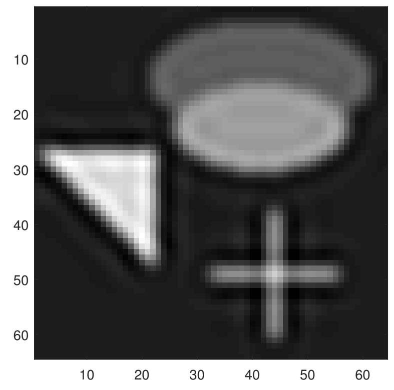

Figure 15: Perform 5 CGLS iteratios

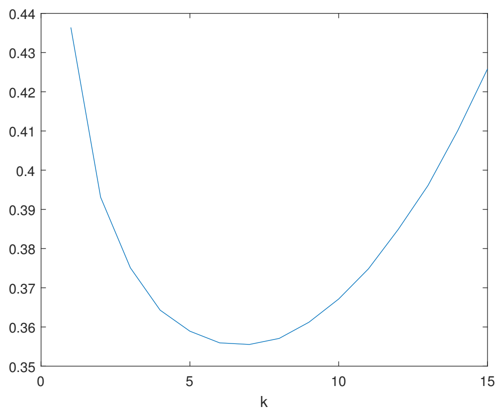

Figure 16: Error history for noisy case