TP IMINV : Statistical Linear Destriping of Satellite-Based Pushbroom - Type Images

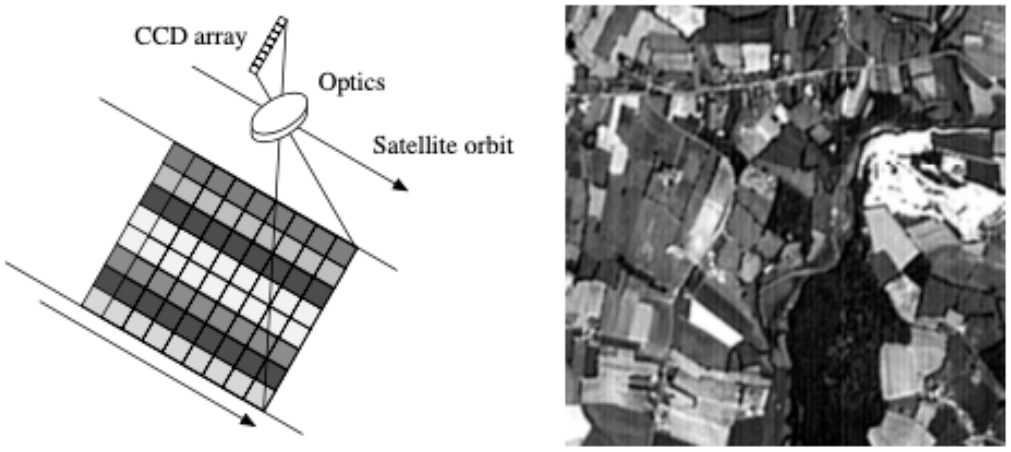

Figure 1: (left) Image formation of pushbroom-type satellite imaging instrument. (right) Zoom on a small part (200 × 200) of an original SPOT3 image. The image has been enhanced to emphasize the striping effect.

Supervised by : Jérome IDIER

Student : Chaitanya Krishna Viriyala

Theoretical Part and Practical Part :

First of all, let us try to understand the title ‘Statistical Linear Destriping of Satellite-Based Pushbroom - Type Images’

There is a paper by the same name by the supervisor and co. dating back to 2009

The breakdown of the title : (definitions taken from Wikipedia, will be quoted at the end)

Destriping (in the context of images) : It is the process of removing stripes or streaks from images and videos.

Pushbroom scanner : It is basically a device for obtaining images by stereoscopic sensors.

Problem Statement :

Pushbroom-type imaging instruments are designed to acquire complete rows of images using a linear array of detectors. The other dimension is obtained by the column-wise scanning that results from the motion of the satellite along its orbit. There are other methods of solving the same problem using Forier filtering, wavelet based methods but they have their own side effects. Essentially what we are trying to solve is an inverse problem. It is based on the statistical estimation of each detector gain from the observed image, assuming a linear response.

In this lab, the aim is to take the methods studied in tutorial on matlab and to evaluate them. Note first that all the methods can be considered as applications of different estimators around the same general model: the model of the nearest neighbor Markov fields.

- For the image, it involves assigning a probability to a local variation in pixel intensity. For a landscape, we would rather expect gentle variations in most cases, we will want to give less chances to large differences in intensity (the reverse would promote noise).

Corresponding MATLAB code :

```
%% Load the data and visualise


load('TP1_Exemple.mat');
% Once we load, we see that there are two variables gth and
Z
% gth = 453 * 1 the *gain* of the detector
% and Z = 2530 * 453  the *true pixel intensity*
W = transpose(gth) .* Z;
fth = log(gth);
fth_centre = fth-mean(fth);
colormap gray;
imagesc(W);
figure()
colormap gray
imagesc(Z);
```

WORKSPACE

Name	Value	Size	Class
gth	*453×1 double*	453×1	double
Z	*2380×453 …*	2380×453	double

Figure 2 : Load the data in Matlab, we can see the following variables, they are self explanatory

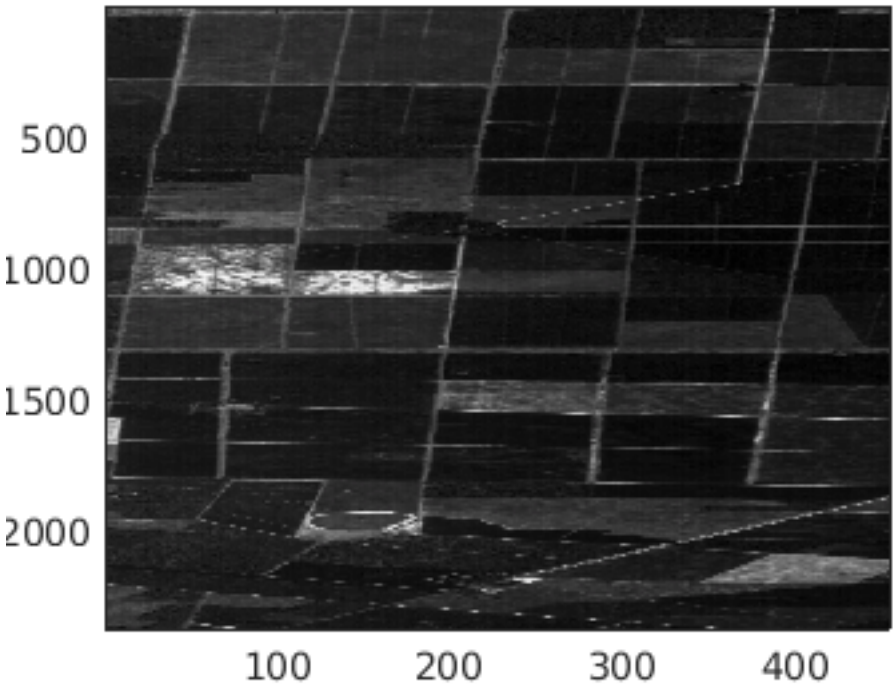

Figure 3 : Visualise the push broom sensor image

Important : This penalty is made by choosing a phi function which is tested in two forms in this practical work: absolute value and square function.

Corresponding MATLAB code :

```
%% Function Phi

abs_x = -2:0.01:2;
phi1 = abs(abs_x);
phi2 = abs_x.^2;
plot(abs_x,phi1,abs_x,phi2);
legend('Absolute Value', 'Squared')
```

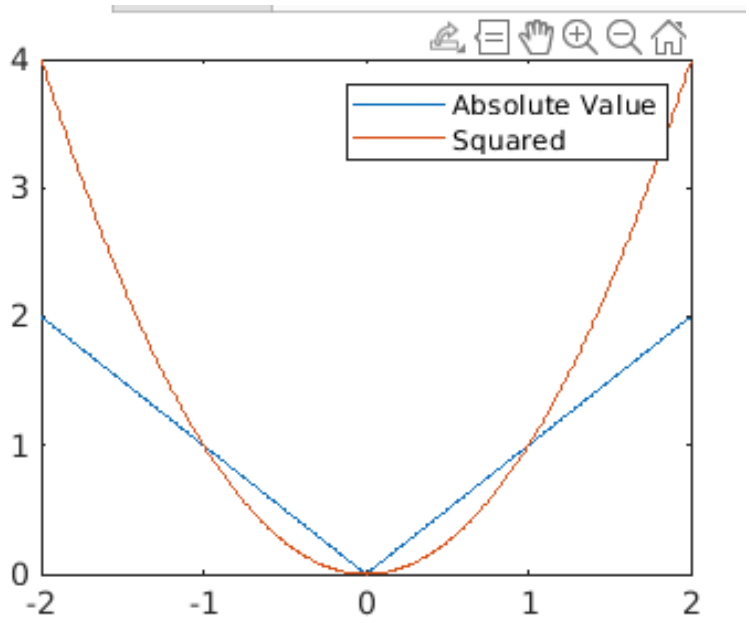

Figure 4 : 2 chosen Phi Functions one is squared and absolute for the Penalty Function

Remark : The absolute value penalizes the breaks in landscape variations less than the square.

Objective :

The objective of this report is to answer two simple questions :

1 . Is the hypothesis of a Gaussian distribution of logarithmic gains well founded? Which parameter mu 'best approximates the model?

2. The absolute value penalises the breaks in landscape variations less than the square. But which one is the most suitable? (recall figure 4 : squared vs absolute value)

To answer these questions, we have a set of data (theoretical gains and ideal images) which simulate the real problem. It is not a question here of questioning the simulation, but of answering the preceding questions as well as possible by assuming the adapted simulation. These data will be used to :
-simulate the image with the pyjama effect, from which we will try to find the gains.
-evaluate the quality of our estimates.

Method 0 : All gains are equal

As mentioned in the TP, this is the "do-nothing" approach. There is no image correction

Corresponding MATLAB code :

```
%% Method 0


g_meth0 = ones(size(gth));
f_meth0 = log(g_meth0);
err_0 = sqrt((sum((f_meth0-fth_centre).^2)));
```

Method 1 : All gains are not equal

As mentioned in the TP, there is an image correction

```
%% Method 1


V=log(W);
Y=log(Z);
f_meth1=mean(V,1);
f_meth1_centre = f_meth1-mean(f_meth1);
%g_meth1=exp(f_meth1)';
err_1 = sqrt((sum((f_meth1_centre'-fth_centre).^2)));
```

Note then that we have studied two types of (classical) estimators for the gains based on this probability: the maximum likelihood (MV), then the maximum a priori (MAP).

With the MV, we look for what gain will make it possible to reconstruct the most probable image. With the MAP, we make an additional assumption: we also give ourselves a probability law on the gains which will weight the likelihood in a way. We no longer seek to blindly maximise the likelihood, even if it means choosing outlier gains but to maximise the likelihood given the fact that the gains are localised (here for lack of a better way we choose a Gaussian law, whose parameters mu and sigma are to be determined). It is therefore not surprising that the solution given for the MAP corresponds to the solution of the problem MV to a regularisation in addition to the gains, but it is remarkable that the problem which has two parameters

to optimise (mu, sigma) a priori only has one (mu ') when it is set as a MAP optimisation.

Corresponding MATLAB code :

```
%% Differences between method 0 and 1


plot(fth_centre)
hold on
plot(f_meth0)
hold on
plot(f_meth1_centre)
legend('Theoritical Gain centered','Method 0 Gain
centred','Method 1 Gain centred')



%% Meth 2
close all
f_meth2=zeros(size(gth));
for i =2:453
    deltaV = V(:,i)-V(:,i-1);
    delta_f = median(deltaV); f_meth2(i) =
f_meth2(i-1)+delta_f;
end
g_meth2=exp(f_meth2);
f_meth2_centre = f_meth2-mean(f_meth2);
err_2 = sqrt((sum((f_meth2_centre-fth_centre).^2)));
close all
plot(fth_centre)
hold on
plot(f_meth0)
hold on
plot(f_meth1_centre)
hold on
plot(f_meth2_centre)
legend('Theoritical Gain','Method 0 Gain','Method 1
Gain','Method 2 Gain’)
```

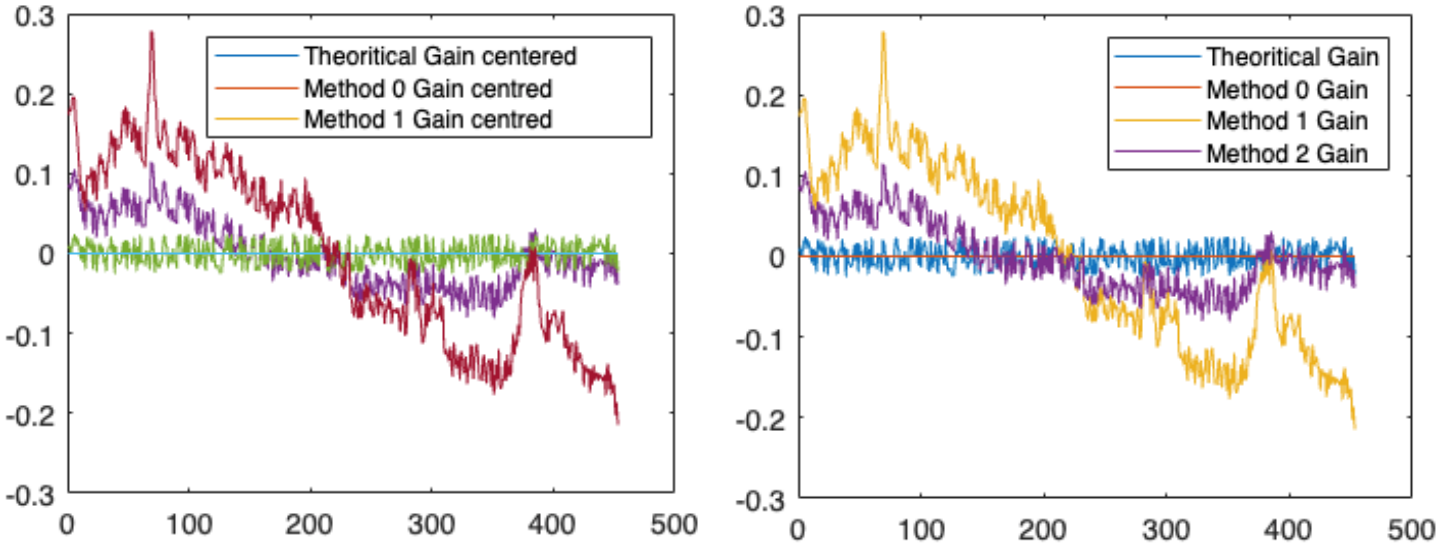

Figure 5 : a) Comparisons between Theoretical Gain, method 0 and method 1 gains

b) Comparisons between Theoretical Gain, method 0, method 1 and method 2 gains

Remarks :

It can be seen that the high-frequency variations have similarities with those of the theoretical gain. On the other hand, low-frequency variations of high amplitudes are observed, which highlight the problems of the model. It is assumed in this model that the average landscape (average intensity) does not depend on the column photographed. However, this is not true: one can see in the photograph of the statement that the average landscape of the first columns is much darker than the average landscape of the columns on the right. So we are making a mistake. The deviations from this average are due to low frequencies (great homogeneity of the landscape on a slice). This error could be reduced if M were increased: each photograph would represent a larger and therefore more varied slice of landscape, more representative of the overall average landscape. This would correspond to increasing the scale and thus limiting the impact of these homogeneities in favour of the variations (high frequencies).

The choice of the median rather than the average as the reference in Method 2 brings the theoretical gain closer. This seems to reduce the amplitude of the low frequency

oscillations (see abscissa points 80 and 350). The high-frequency rate is still satisfactory.

Choice of error measurement :

The theoretical study showed that the gains were determined to within a multiplicative (respective additive) constant, which is quite intuitive: it is the ratio (respective difference) of the gain (respective logarithmic gain) between two columns that creates the pyjama effect. For practical reasons, we work on a logarithmic scale. It is therefore advisable to centre our gains around the same value in order to be able to compare them. The simplest way to do this is to subtract from all our logarithmic earnings their average (centering at zero).

We make the somewhat arbitrary choice of an L2 standard for the measurement of the error, which will penalise quite strongly large deviations from the theoretical values. A graphical confirmation will be interesting.

Corresponding MATLAB code :

```
%% Meth3


D = diag(ones(1,453))+diag(-ones(1,452),-1);
D(1,1)=0;
%mu = mean(Y,'all');
s=mean(V,1)';
errs3=zeros(20,1);
mus = logspace(-4,6,20);
i=0;
mu_min=mus(1);
min=100000;
for mu=mus
    i=i+1;
    f_meth3=s-inv(D'*D/mu+eye(453))*s;
    %g_meth3=exp(f_meth3);
    f_meth3_centre = f_meth3-mean(f_meth3);
    err_3 = sqrt((sum((f_meth3_centre-fth_centre).^2)));
errs3(i)=err_3;
```

```
    if err_3<min
        min=err_3;
        mu_min=mu;
    end
end
semilogx(mus,errs3);
xlabel("mu'");
ylabel('Error');
title('Method 3 (phi(x)=x^2)');
f_meth3=s-inv(D'*D/mu_min+eye(453))*s;
%g_meth3=exp(f_meth3);
f_meth3_centre = f_meth3-mean(f_meth3);
err_3 = sqrt((sum((f_meth3_centre-fth_centre).^2)));



%% Meth4
close all
errs=zeros(10,1);
mus = logspace(-4,6,10);
i=0;
min=10^5;
mu_min=mus(1);


for mu=mus
    i=i+1;
    D=diff(eye(453));
    Vt=V';
    lbd = mu;
    %mean(Y,'all');
    f_meth4 = MAPL1(Vt,D,lbd);
    f_meth4_centre= f_meth4-mean(f_meth4);
    err_4 = sqrt((sum((f_meth4_centre-fth_centre).^2)));
errs(i,1) = err_4;
    if err_4<min
        min=err_4;
        mu_min=mu;
    end
end

semilogx(mus,errs);
xlabel("mu'");
ylabel('Error');
title('Method 4 (phi(x)=|x|)');
```

```
ylim([-1 6])
lbd = mu_min;
%mean(Y,'all');
f_meth4 = MAPL1(Vt,D,lbd);
f_meth4_centre= f_meth4-mean(f_meth4);
err_4 = sqrt((sum((f_meth4_centre-fth_centre).^2)));
```

Application and results :

The application of methods 1 and 2 (MV) do not pose any particular problems. For MAP, there is a little bit of parameter research to be done. We can try to be very precise, we have only looked here for its order of magnitude.

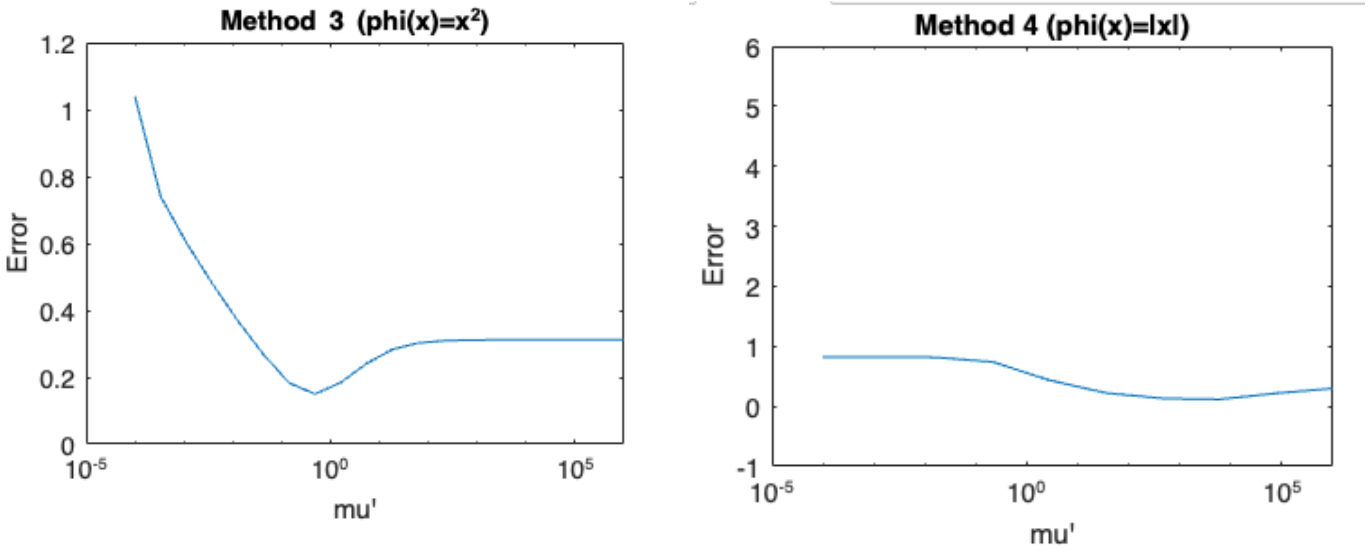

Figure 6 : Varying phi's : a) Method 3 b) Method 4

Putting methods into context :

	MV	MAP
Phi(x)=x^2	Method 1	Method 3
Phi(x)=|x|	Method 2	Method 4

Method results (with error-minimising regularisation parameters) :

	MV	MAP
Phi(x)=x^2	2.4	0.15
Phi(x)=|x|	0.84	0.12

```
%% Comparison Methods 3,4 and 5
close all
%close all
%plot(fth_centre)
%hold on
%plot(f_meth2_centre)
%hold on
plot(abs(f_meth3_centre-fth_centre)/max(abs(fth_centre)))
hold on
plot(abs(f_meth4_centre-fth_centre)/max(abs(fth_centre)))
legend('Method 3 Gain centered','Method 4 Gain centered')
```

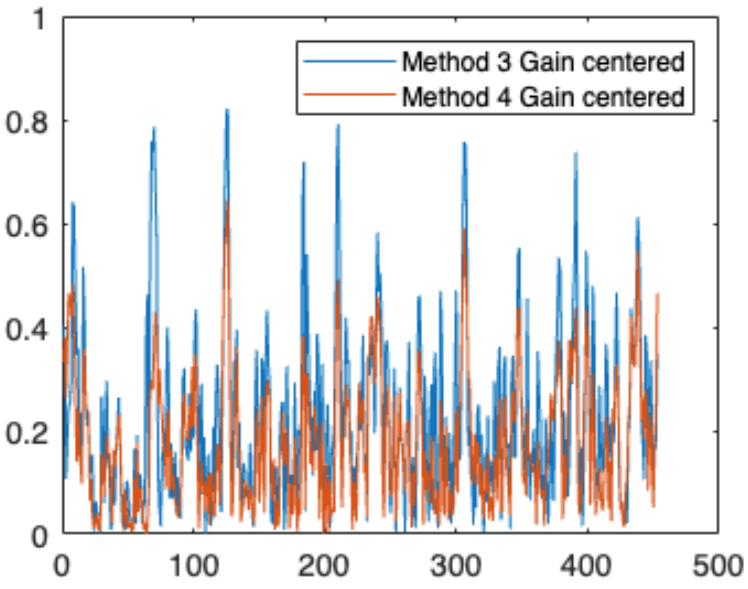

Figure 7 : Comparison between Method 3 and Method 4 Gains

```
%% Last Graph comparing methods
close all
plot(fth_centre)
hold on
plot(f_meth0)
hold on
plot(f_meth3_centre)
hold on
plot(f_meth4_centre)
legend('Theorical Gain ','Method 0 Gain','Method 3
Gain','Method 4 Gain')
```

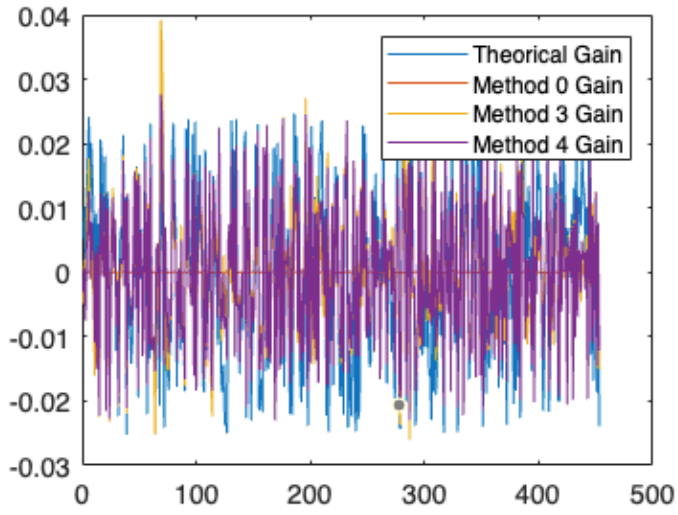

Figure 8 : Comparison between Method 0, Method 3 and Method 4 Gains

Remarks :

With regularisation, the last two methods really come closer to the goal. It seems that giving oneself a Gaussian a priori on the distribution of the gains corresponds to high-pass filtering on their estimation. Indeed, with this information, there is a tendency to choose earnings reasonably close to a certain average. This limits the oscillations as we have seen in the first two methods where the first estimated earnings are all rather above the average and then all rather below. There is no reason for this to happen with a Gaussian hypothesis.

Conclusion :

This TP illustrates the influence of the choices that can be made on an estimation model. All of them were legitimate and it was difficult to decide only on a theoretical part. Here it is therefore with a MAP estimator (with Gaussian a priori) and a penalty for local disparities in |x| that we obtain the best results. But it is quite possible that for other types of problems, a more aggressive penalty in x2 is more realistic and that a VM is more than sufficient. It should also be noted that there is a link between the estimation model and the content of the frequency analysis, which makes it possible to shed new light on the influence of the choice of estimators.

Other Remarks :

For computing the inverse of a matrix, MATLABs inv function is particularly slower or more inaccurate because it requires the condition number of the system in question, whereas we can use A \b to solve it as well.

References :

1. SIRIM_part1_annotated [Class Notes, available on Hippocampus]

2. SIRIM_part2_annotated [Class Notes, available on Hippocampus]

3. TP SIRIM annotated [TP, available on Hippocampus]

4. Research Paper : https://ieeexplore.ieee.org/document/5340644

5. Wiki page for Image Destriping: https://en.wikipedia.org/wiki/Image_destriping

6. Wiki page for Pushbroom Scanner: https://en.wikipedia.org/wiki/Push_broom_scanner

7. MATLABs inv function : http://blogs.mathworks.com/loren/2007/05/16/purpose-of-inv/