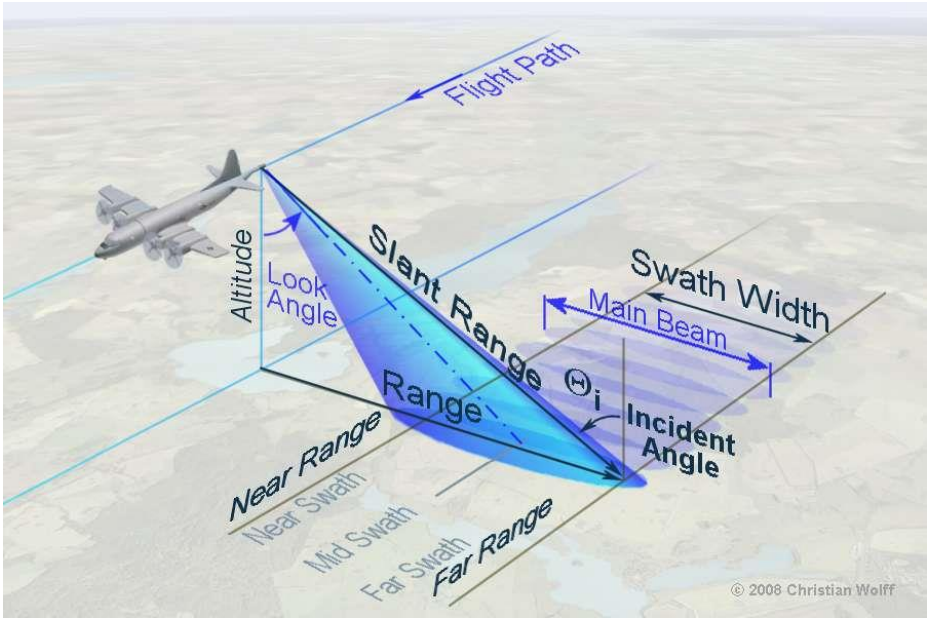
DTU Technical University of Denmark

Remote Sensing (30350) Project Report:

A Test Statistic in the Complex Wishart Distribution and Its Application to Change Detection in Polarimetric SAR Data



By:
Roberto Prestigiacomo (s132275)
Athanasios Belantonas (s162704)

Autumn 2017

TABLE OF CONTENTS

1 - INTRODUCTION

1.1 - SAR Systems Generalities

A SAR (*Synthetic Aperture Radar*) is a radar system used in remote sensing, where an antenna is directed towards the ground at a certain incidence angle. The antenna can be mounted on a satellite (spaceborne) or on one side of an aircraft (airborne systems). In the following, we will refer to airborne systems since it is the case under examination in this context.

.

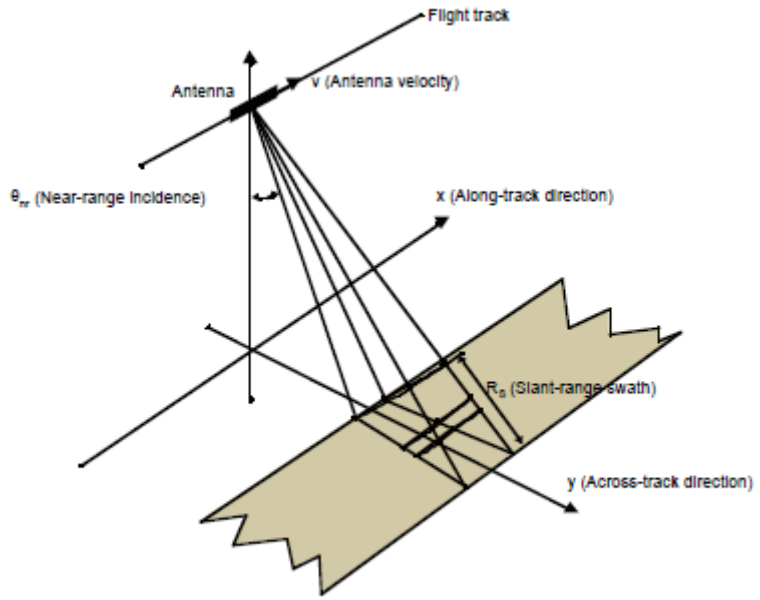


Fig 1 – Basic Diagram of an airborne SAR system

Targets within the observed area of the radar (swath) will reflect and backscatter the signal from the radar. The backscattered signal is detected and processed to create an image of the observed targets. Due to different characteristics of the targets, they can be discriminated by the system.

Frequencies commonly used in radar remote sensing fall in the *L-, C-, or X-Band*. In this case, we work on data acquired in L-Band (1 to 2 GHz, in wavelength: 15 to 30 cm).

The SAR technique is a powerful technique, in the sense that it is capable of mapping in most weather and light conditions. It holds strong potential for many applications of remote sensing, including change detection which is under investigation in this report.

Different targets are sensitive to differently polarized signals. Therefore using a SAR system, different combinations of transmitted and received polarized signals can be used, to create a system typically referred to as *Polarimetric SAR*.

In a Polarimetric SAR, the amplitude and phase of backscattered signals are thus measured in four combinations of linear transmitted and received polarizations : HH, HV, VH and VV. This data is used to build the so-called *complex scattering matrix*.

SAR data is affected by the inherent *speckle noise*, therefore a filter is applied by the system -at the expense of resolution- establishing the corresponding measurement as a multilook case. In multilook cases, a more proper representation of the data is given through the so-called *covariance matrix*, where the properties of a group of resolution cells can be expressed in a single matrix.

The covariance matrix is a 3x3 complex matrix defined as

$$\langle C \rangle = \begin{bmatrix} \langle S_{hh} S_{hh}^{*} \rangle & \langle S_{hh} S_{hv}^{*} \rangle & \langle S_{hh} S_{vv}^{*} \rangle \\ \langle S_{hv} S_{hh}^{*} \rangle & \langle S_{hv} S_{hv}^{*} \rangle & \langle S_{hv} S_{vv}^{*} \rangle \\ \langle S_{vv} S_{hh}^{*} \rangle & \langle S_{vv} S_{hv}^{*} \rangle & \langle S_{vv} S_{vv}^{*} \rangle \end{bmatrix} \quad (1)$$

Such a covariance matrix is a *Hermitian*, meaning that the elements above the main diagonal are the complex conjugates of the element below it. This matrix contains the data manipulated in this report, each of them corresponding to a specific pixel.

1.2 - Description of the System

The data used in this report were acquired by the Danish airborne *EMISAR system*. EMISAR is a fully polarimetric system operating at 2 frequencies:

1. 5.3 GHz / 5.7 cm wavelength (C-Band)
2. 1.25 GHz / 24 cm wavelength (L-Band)

As mentioned in the previous paragraph, only the L-Band data has been used for the demonastrion of the presented techniques in this report. Further specifications of the system are listed below:

- Altitude: ~ 12500 m
- Spatial resolution: 2 x 2 m (one look)
- Ground range swath: ~ 12 km
- Incidence angles (Typ.): 35° to 60°
- Radiometric calibration: better than ±0.5 dB
- Channel imbalance: less than ±0.5 dB in amplitude and ±5° in phase
- Cross-Polarisation contamination suppressed by more than 30 dB.

All acquisitions were co-registered by identifying ground control points in the images and using an interferometric DEM acquired by the EMISAR system. Before resampling, the original one-look scattering matrix data were transformed to covariance matrix data, and these data were averaged to reduce the speckle by a cosine-squared weighted 9 by 9 filter. The new pixel spacing in the images is 5 m by 5 m, and the effective spatial resolution is approximately 8 m by 8 m at mid-range. After the averaging the equivalent number of looks is estimated to be 9-11 from homogenous areas in the images. This corresponds to a standard deviation for the backscatter coefficient of approximately 1.1 – 1.8 dB.

1.3 - Description of the Test Area

The data used here consists of acquisitions, from March 21, 1998 to August 16 of the same year.

The site is is located at the *Research Centre Foulum of the Danish Institute of Agricultural Sciences*. The site contains a large number of agricultural fields with different crops, as well as several lakes, forests, areas with natural vegetation, grasslands, and urban areas.

Fig 2 - L-band EMISAR image of the Foulum test site (16 June 1998)

We have worked only on part of the image above, as demonstrated later in this report, which contains crops of different types and sizes, a forest area and a lake.

The area is relatively flat, and corrections of the local incidence angle due to terrain slope are therefore, as a first approximation, not necessary.

1.4 - Theory

The core of this project is testing the equality of pairs of pictures - formed using the polarimetric SAR data - and so determining if the covariance matrices for each pixel are equal or not for two different times (i.e. two different pictures).

Under the assumption that the covariance matrices follow a complex Wishart distribution, which is a reasonable assumption in this application, it can be demonstrated that for two p*p Hermitian, positive, definite, Wishart distributed matrices **X** and **Y** under the *null hypothesis* H_0: the two matrices are equal, the *likelihood-ratio test statistic* is

$$Q = \frac{(n+m)^{p(n+m)} |\boldsymbol{X}|^n |\boldsymbol{Y}|^m}{n^{pn} m^{pm} |\boldsymbol{X}+\boldsymbol{Y}|^{(n+m)}} \qquad (2)$$

which in the case n = m that is typically the case for change detection, leads to

$$\ln Q = n\left(2p\ln 2 + \ln|\boldsymbol{X}| + \ln|\boldsymbol{Y}| - 2\ln|\boldsymbol{X}+\boldsymbol{Y}|\right) \qquad (3)$$

It can also be demonstrated that the probability of finding a smaller value of the variable (-2ρlnQ) than z (the cumulative distribution function of the distribution -2ρlnQ) is

$$P\{-2\rho\ln Q\leqslant z\}\approx P\{\chi^2(p^2)\leqslant z\}+\omega_2[P\{\chi^2(p^2+4)\leqslant z\}-P\{\chi^2(p^2)\leqslant z\}] \quad (4)$$

With

$$\rho=1-\frac{2p^2-1}{6p}\left(\frac{1}{n}+\frac{1}{m}-\frac{1}{n+m}\right) \quad (5)$$

$$\omega_2=-\frac{p^2}{4}\left(1-\frac{1}{\rho}\right)^2+\frac{p^2(p^2-1)}{24}\left(\frac{1}{n^2}+\frac{1}{m^2}-\frac{1}{(n+m)^2}\right)\frac{1}{\rho^2} \quad (6)$$

Thus, -2ρlnQ should fit a χ^2 distribution.

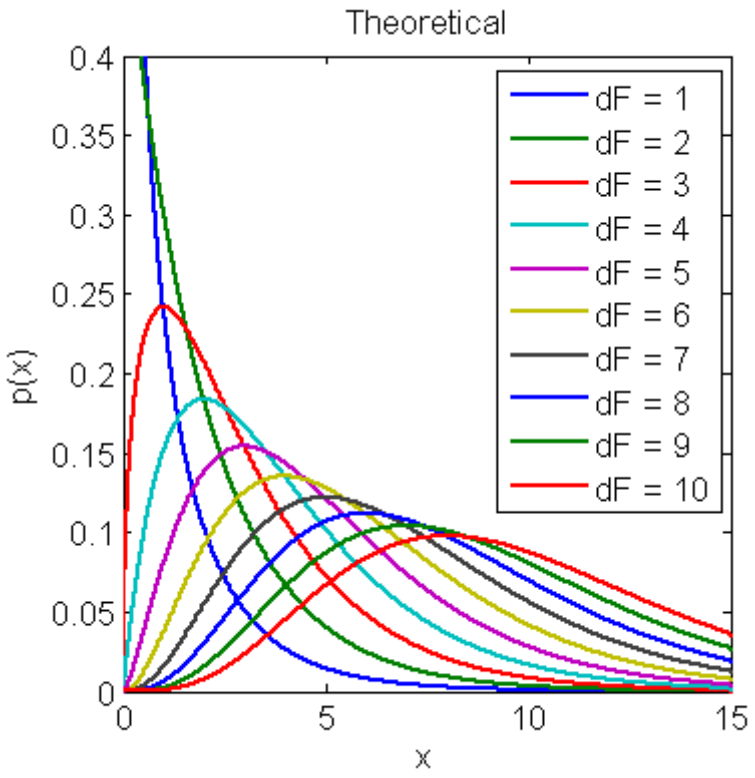


Fig 3 - Family of χ^2 distributions as a function of the degrees of freedom

The null hypothesis is rejected when the probability in (4) is high, which corresponds to the tail of the distribution of -2ρlnQ. According to *'Test statistic in complex Wishart*

distribution' by *Conradsen et al*., the null hypothesis is rejected when the probability of finding a larger value of $-2\rho \ln Q$ is below the 1-5 % significance levels.

2 - PRELIMINARY DATA HANDLING

2.1 - Reading Data, Building the Covariance Matrices, Filtering Speckle

Each of the provided data sets, consists of the elements of the main diagonal (hhhh, hvhv, vvvv) and those above it (hhvv, hhhv, hvvv), with reference to the covariance matrices, for each pixel. The former are real numbers whereas the latter are complex numbers. These elements are stored in separate binary files (one for every combination of channels) that are separately reshaped into 6 different arrays 1x(1024x1024). The arrays are then used to build a 3x3x(1024x1024) three-dimensional covariance matrix where for a certain value of the third dimension, say k, we have a 3x3 submatrix representing the covariance matrix of the k-th pixel. Lastly, the elements below the main diagonals are formed by taking the complex conjugate of the arrays of hhvv, hhhv, and hvvvv following the order shown below:

hhhh	hhhv	hhvv
hhhv*	hvhv	hvvv
hhvv*	hvvv*	vvvv

Fig 4 – 3 x 3 matrix correspondent to the k-th position in the third dimension.

Although this data had already been through speckle filtering, we have performed further filtering using a 3x3 box filter, to demonstrate its effect on the change detection images. In this case the algorithm is quite straightforward: a 3x3 window scans a 1024x1024 matrix and at each step the central pixel of the new picture is set equal to the mean value in the window, as shown in Fig.5. As for the border of the averaged image, the pixel values are set to 0, therefore no filtering is performed for them. Note that the border includes only the peripheral pixels, since a 3x3 averaging window was used. In general, if an n*n window is used (n being an odd number) the width of the border is equal to $\lfloor n/2 \rfloor$ (Fig. 6).

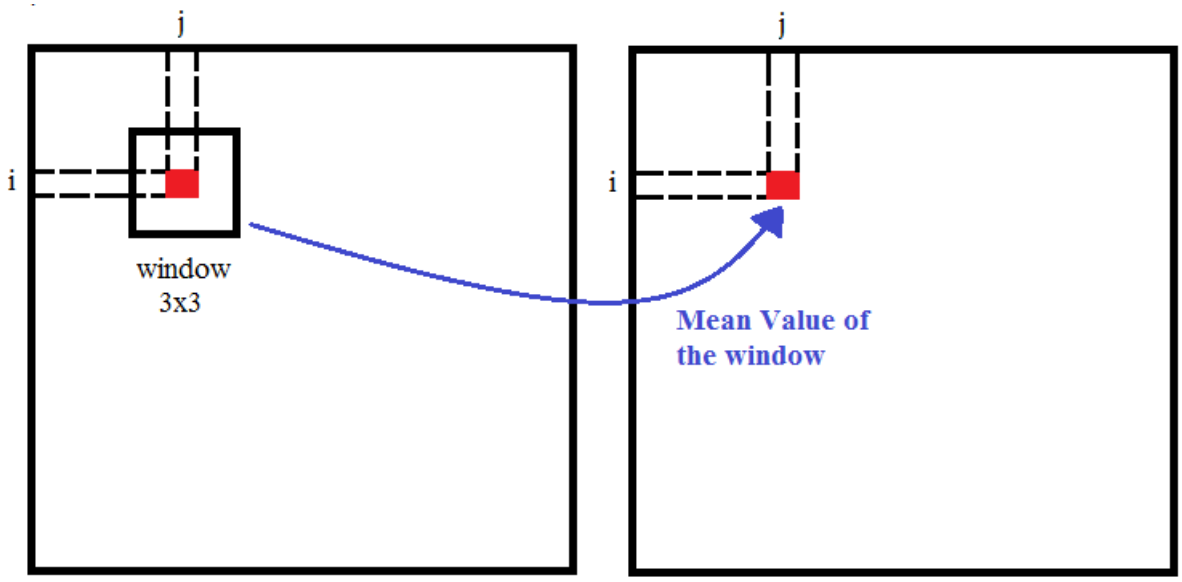


Fig 5 – Filtering operation

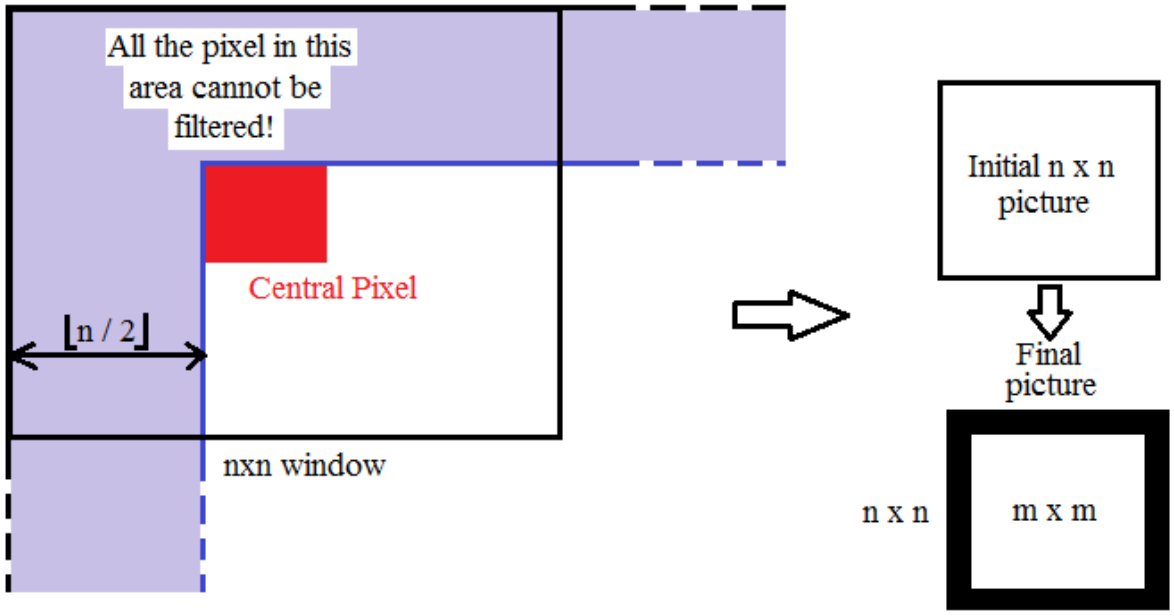


Fig 6 – Treatment of the border during the filtering

2.2 – Showing RGB - Coded dB Pictures

By using only the real elements of the main diagonals, namely hhhh, hvhv, and vvvv, we built three 1024x1024 matrices. These matrices can be displayed as images and each matrix can represent the weight for the red, green, and blue component to produce an RGB-coded image. By selecting the hvhv elements for the red component, hhhh for green and vvvv for blue, after a logarithm transformation and scaling the values, we obtained the following pictures:

Fig 7 – RGB Coded db Pictures at two different times.

In the picture some changes are already evident, especially in the state of the crops whereas the forest (the yellowish, bright area) does not seem to be changed. The lake in the upper right part of the pictures has a shadow cast near the opposite shores in the 2 pictures, this will be detected as a change as well.

After filtering, the speckle is reduced, but the resolution of the picture is degraded and the edges are smeared out as shown in the following pictures of March 21.

Fig 8 – Comparison of the picture for March 21 before (left) and after (right) the spekle filtering

The details of the above images can be observed in figure 9, while speckle reduction is show in figure 10.

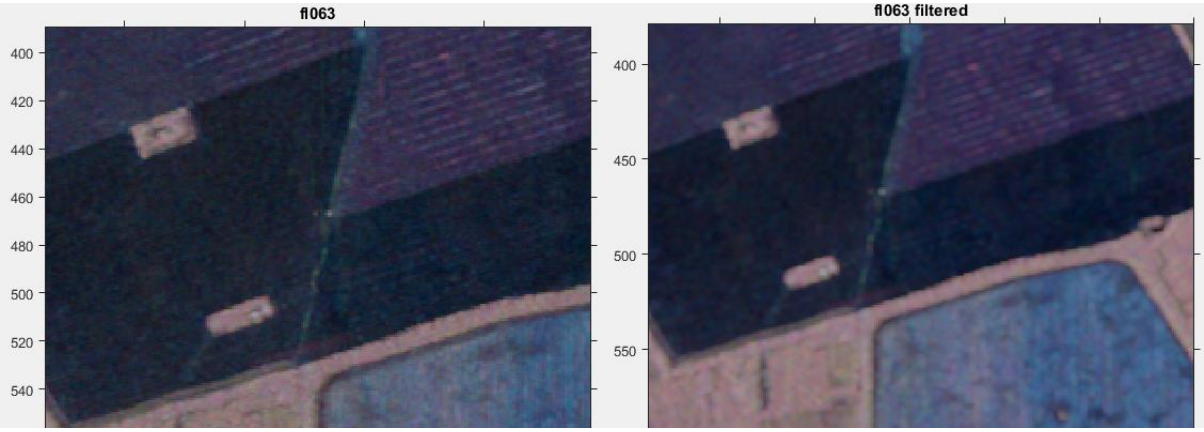


Fig 9 – Loss of resolution in the filtering operation.

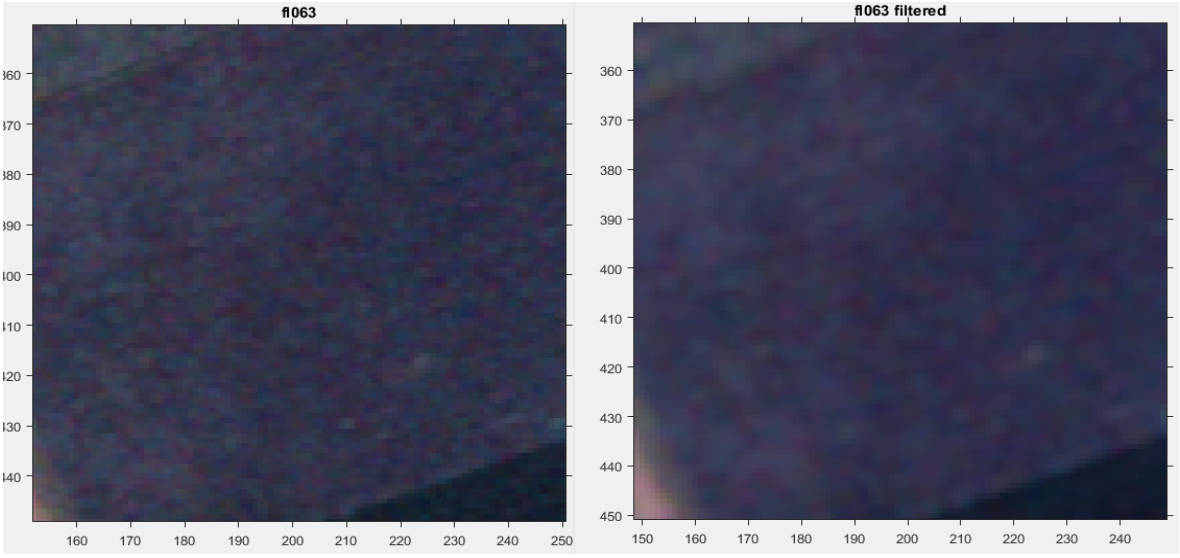


Fig 10 – Visualisation of the speckle reduction.

2.3 - Characterising the System

As mentioned in paragraph 1.2, a speckle filtering operation had already been carried out on the data leading to the multilook condition. In the same paragraph, it is noted that an *equivalent number of looks* (*ENL*) of 9-11 is expected in the case under investigation.

Therefore, after reading the data, we proceeded to the estimation of the ENL. In order to do that, the hhhh, hvhv, and vvvv data (March 21) were used, in the form of 1024x1024 matrices. Three homogeneous areas were selected, as shown in Fig. 11 and for these areas we estimated the ENL through.

$$ENL = \frac{\mu^2(I)}{\sigma^2(I)} \qquad (7)$$

This operation was repeated for all three matrices and 9 estimates were obtained.

By taking the mean value of the estimates and rounding it, the final ENL was obtained as 10, a value consistent with what was expected.

The same process was repeated for the speckle filtered images. The filtering operation causes the ENL to increase and in fact we obtained a value of 23 after it was applied.

Fig 11 – Selection of homogeneous areas for the ENL estimation.

2.4 - Issue with Data after Filtering

In paragraph 2.1 the speckle reduction filtering has been described and the fact that after filtering, the border of the picture is reduced to 0, has been emphasized. It should be stressed once more in this point, that one must keep in mind this processing artifact because it affects the processing of the filtered data. For example, to estimate the ENL we need to select three windows, as described in the previous paragraph and although it didn't present any problem here, it could in general.

As will be discussed later in this report, a no-change area will have to be selected from the $-2\rho \ln Q$ picture (paragraph 3.2) and the same issue described above arises.

In general, it is a good precaution to take out the border of the image and work on 1022x1022 pictures, but when displaying the images and selecting areas within them, activities sensitive to the dimensions of the pictures, if one does not want to deal with these issues manually, it is necessary to include the border again to prevent dimension mismatches (e.g. a window in an unfiltered and in a filtered image can correspond to different areas and give incorrect results in some cases).

3 - CHANGE DETECTION

3.1 - Ratios test statistic

A simple ratios test was initially performed, using the 1% fractile threshold taken from the pixel distribution of the selected forest -no change- area, as seen in figure 12. The following figures and images in this part of the report, are based on the HV polarization measurements, as taken from the provided fl063 and fl064 SAR data

Fig 12 – Selection of a homogeneous forest area

Figure 13 illustrates the pixel ratios distribution (HV63/HV64) for the complete ratios-based image. Pixels whose values approach 1 should now be considered as no-change pixels, as the ratio of two given parameters approaches 1 when those parameters are equal. Values that are bigger or smaller than 1 indicate that the two pixels from two different times are not equal and therefore a change has occurred.

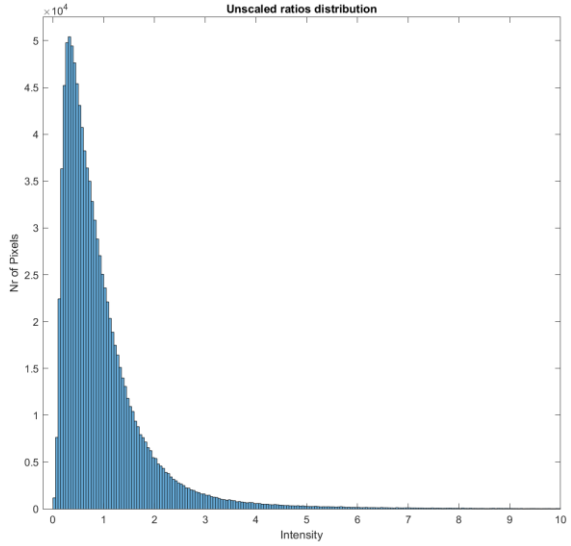


Fig 13 – Unprocessed pixel ratios' distribution [HV-fl0063/HV-fl0064]

In order to visualise the ratios image as a change/no change image through MatLab, the ratios were scaled by an algorithm that inverts the pixel values . Therefore, the distribution of the ratios is rearranged, so that all the values are now between 0 and 1, where 0 indicates that a change has occurred and 1 is treated as the no-change value. Figure 14 shows the distribution of the pixel values after scaling, while figure 15 and figure 16 show the produced ratios image before and after some further scaling (1-scaled_pixels for visualization purposes), so that areas with changes appear as white and no-change areas appear as black (fig 16).

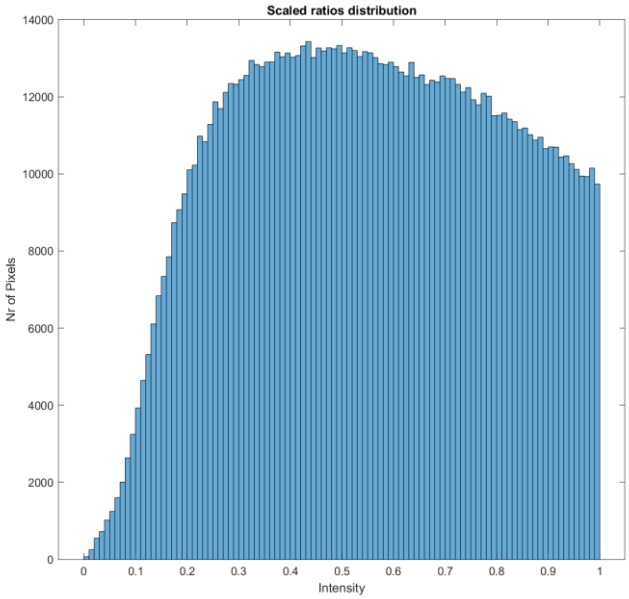


Fig 14 – Processed pixel ratios' distribution

Fig 15 – Unscaled ratios image

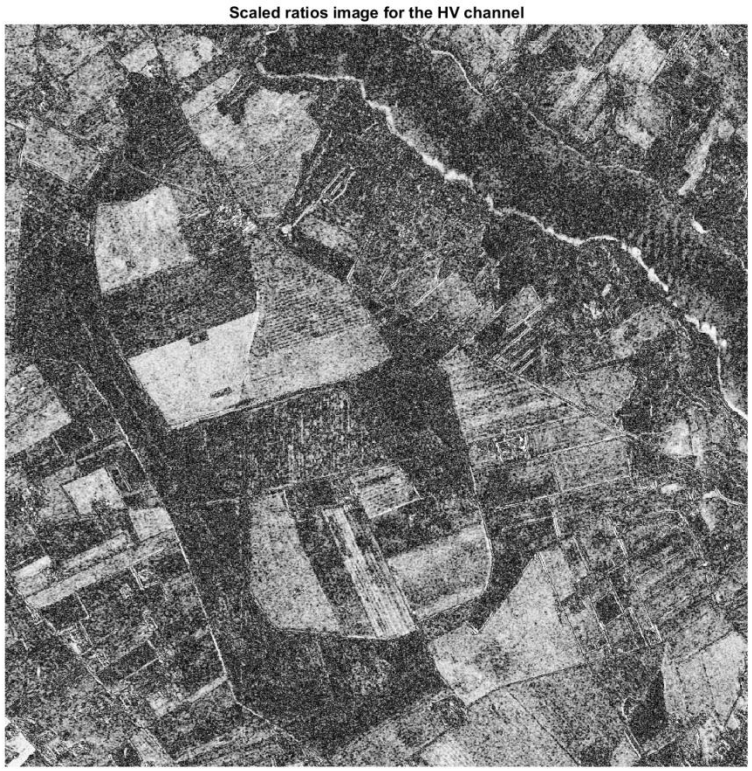


Fig 16 –Scaled ratios image [White pixels: change. Black pixels: no-change]

The threshold correspondent to the accepted 1% false alarm rate for the ratios test, can now be deduced from the distribution of the scaled pixels, using a homogeneous forest area, which is typically a no-change area. Figure 17 shows the distribution of the scaled pixels and the 1% threshold, both corresponding to the selected area. Any pixel values greater than the selected threshold will be treated as no-change pixels and those below it as changed ones.

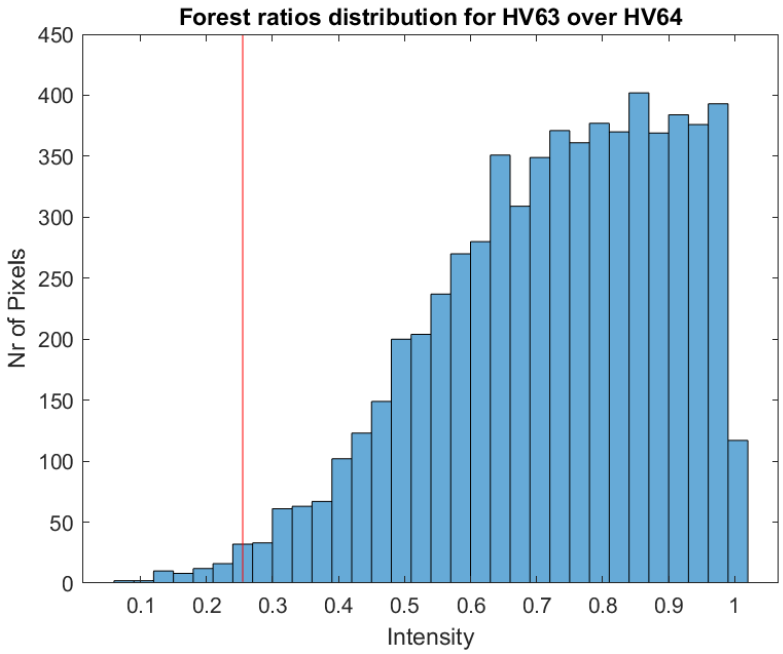


Fig 17 -Selected area's pixel values distribution

Furthermore, an algorithm that transforms the pixel values into a binary state, where zeros are the no-change pixels and ones the changed pixels, was implemented. Figure 18 shows the resulting binary image.

After the speckle reduction filter has been applied, the process described above was repeated for the averaged pixels once more. The initial ratios image and the binary ratios image produced by the process is depicted in figure 19 and 20 respectively.

Fig 18 – 1% FAR Change detection binary image (ratios before the box filter)

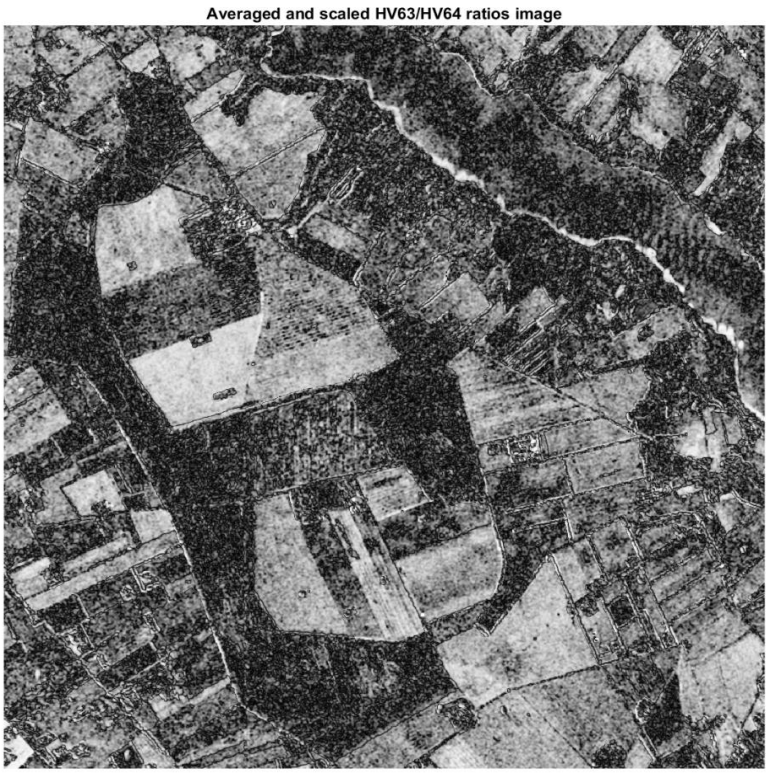


Fig 19- 1% FAR Change detection image (ratios after the box filter)

Fig 20 - 1% FAR Change detection binary image (ratios after the box filter)

3.2 - [-2ρlnQ] test statistic

In this part of the report, we demonstrate another test statistic using a 1% false alarm rate, based on the calculation of lnQ, as described in chapter 1.4. For this purpose, each pixel value of the two compared images will be transformed to its lnQ equivalent value, using the full covariance matrix corresponding to that pixel and scaled by a factor of 2ρ.

For visualization purposes, each pixel is further transformed to its additive inverse, so that each of them contains a real, positive value that can be manipulated and visualized using Matlab. Figure 21 shows how (-lnQ) is visualized after the calculation of lnQ, using the full covariance matrices of the 2 images under comparison, as given in file *fl063* and *fl064* of the provided SAR data.

The scaling factors of 2ρ are then applied, while the same forest area used in the threshold determination for the ratios image, is used once more in order

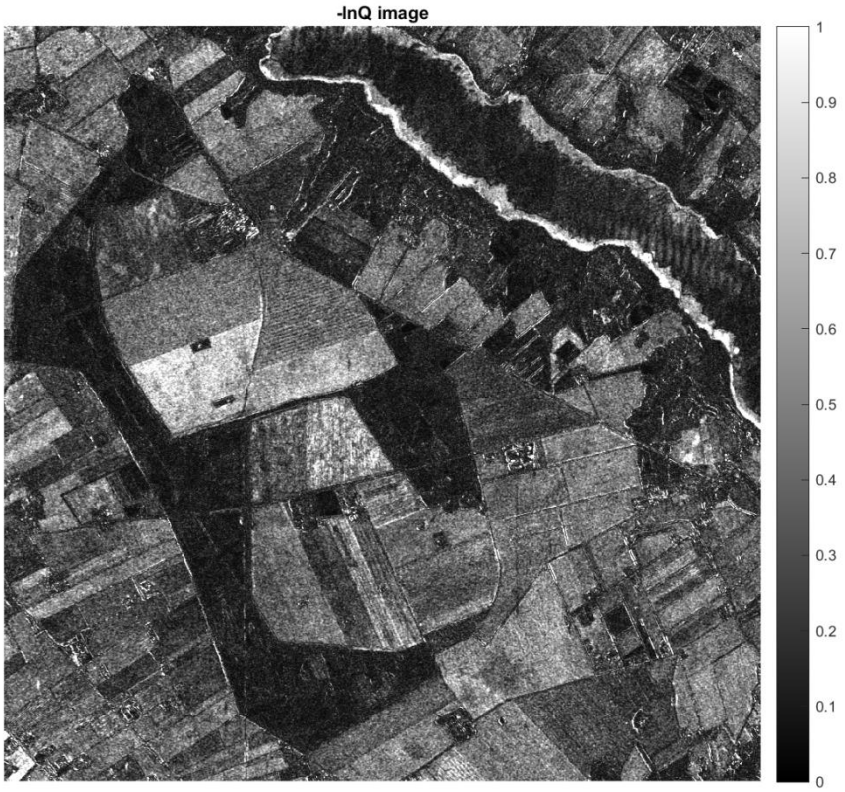


Fig 21 – (-lnQ) visualization

to determine the threshold correspondent to the accepted 1% false alarm rate, based on the -2ρlnQ image for this case.

The distribution of the pixels taken from the no-change forest area, should theoretically fit a χ^2 distribution, as described in chapter 1.4 of this report. Figure 22 contains the histogram of the -2ρlnQ pixels for the forest area, along with the theoretical χ^2 distribution, which should now fit the histogram of the pixels taken from the selected area. The selected 1% threshold is given by the vertical red line, in the same figure.

The threshold is then applied to each pixel of the -2ρlnQ image, where any pixels with a value greater than the threshold are considered as changes, or 1s in the final binary image, while values below it, correspond to no-change pixels or 0s. Figure 23 shows the binary change detection image using the selected threshold corresponding to the predetermined 1% false alarm rate.

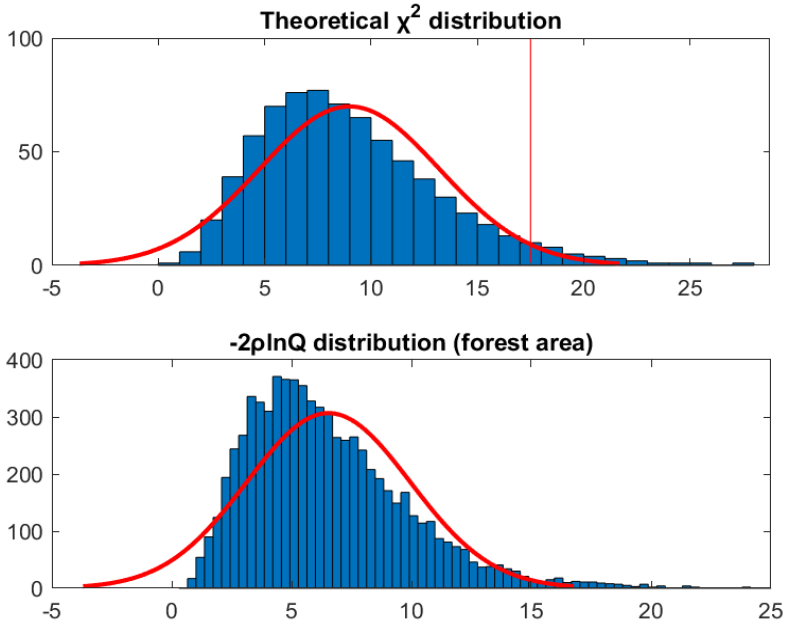


Fig-22 Distribution of pixel values for the forest area, chi squared distribution fit

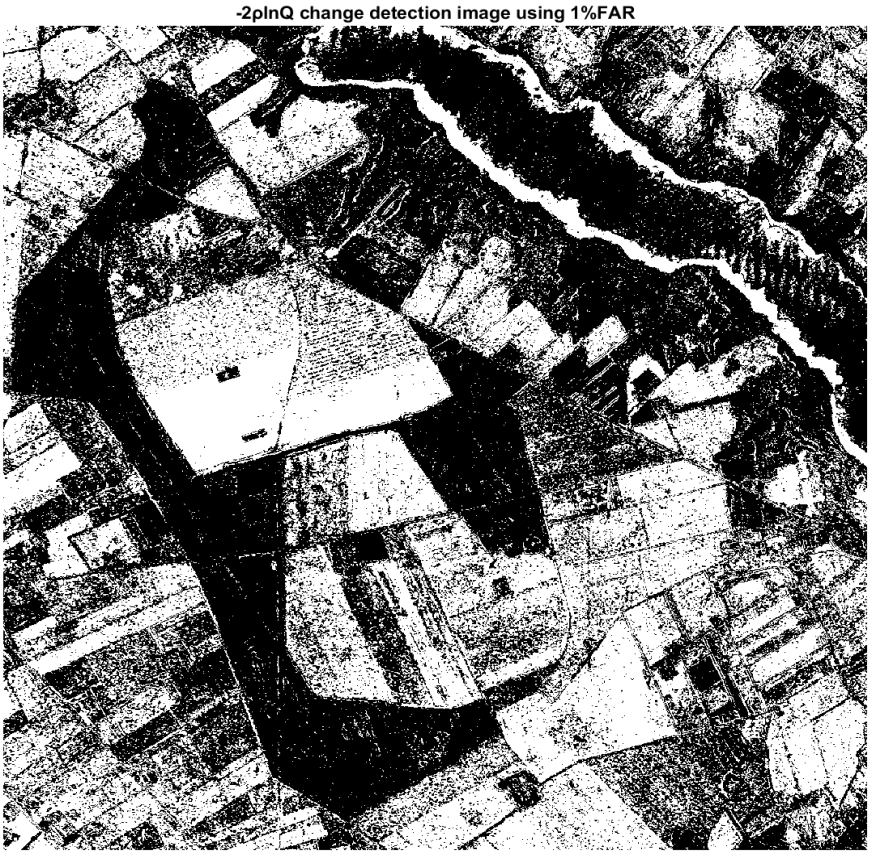


Fig 23 – (-2ρlnQ) binary image after the threshold implementation (before box filter)

As performed for the ratios case, the previously described process must be undergone once more, using -in this case- the averaged covariance matrices of the images under comparison. Figure 24 shows the -lnQ image using the averaged covariance matrices.

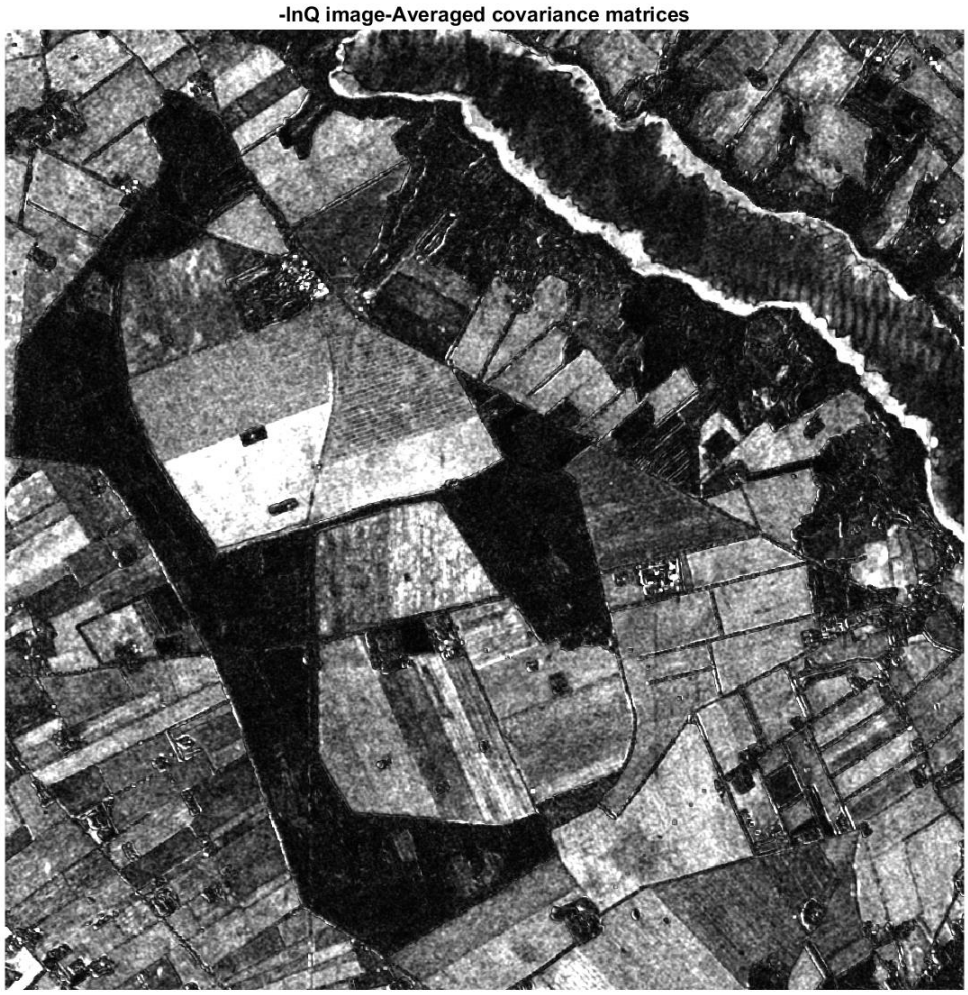


Fig 24 - (-lnQ) visualization for the averaged covariance matrices case

Finally, figure 25 shows the resulting binary change detection image for the averaged covariance matrices case, again using a false alarm rate of 1%.

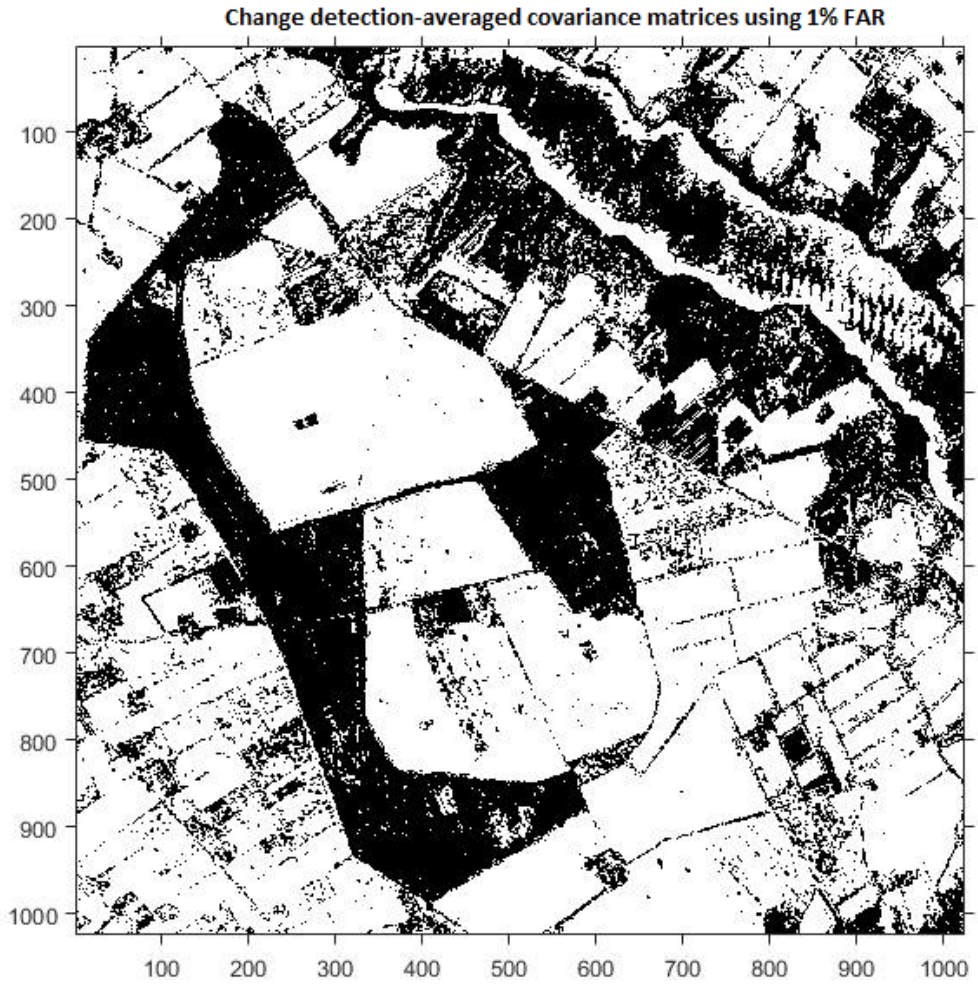


Fig 25 – (-2ρlnQ) binary image after the threshold implementation (before box filter)

3 - DISCUSSION

Two different test statistics have been implemented, to obtain a binary change detection image across 2 polarimetric datasets, each representing the same area at a different time.

Firstly, a simple ratios test statistic was implemented using the unprocessed, HV polarization elements of the pixels' covariance matrices, where a threshold corresponding to a false alarm rate of 1% was determined, as dictated by a -typically- no-change forest area.

Furthermore, the same test statistic was implemented on the -2ρlnQ image, after the calculation of lnQ, using the full covariance matrix for each pixel.

The 2 obtained binary images, based on the ratios change-detection images before and after the box filter was applied, are shown in figure 26. Additionally, figure 27 illustrates the initial and speckle reduced -2ρlnQ change detection images.

The change detection image for the ratios case, seems to be detecting less changes both before and after the box filter was applied, in comparison with the case of the -2ρlnQ image. In particular, areas where more changes were expected to be detected, such as the shores, did not show up in the binary image, in some cases big portions of the fields were wrongly deemed as no-change areas, even though they show up as changed areas in the -2ρlnQ based change detection image.

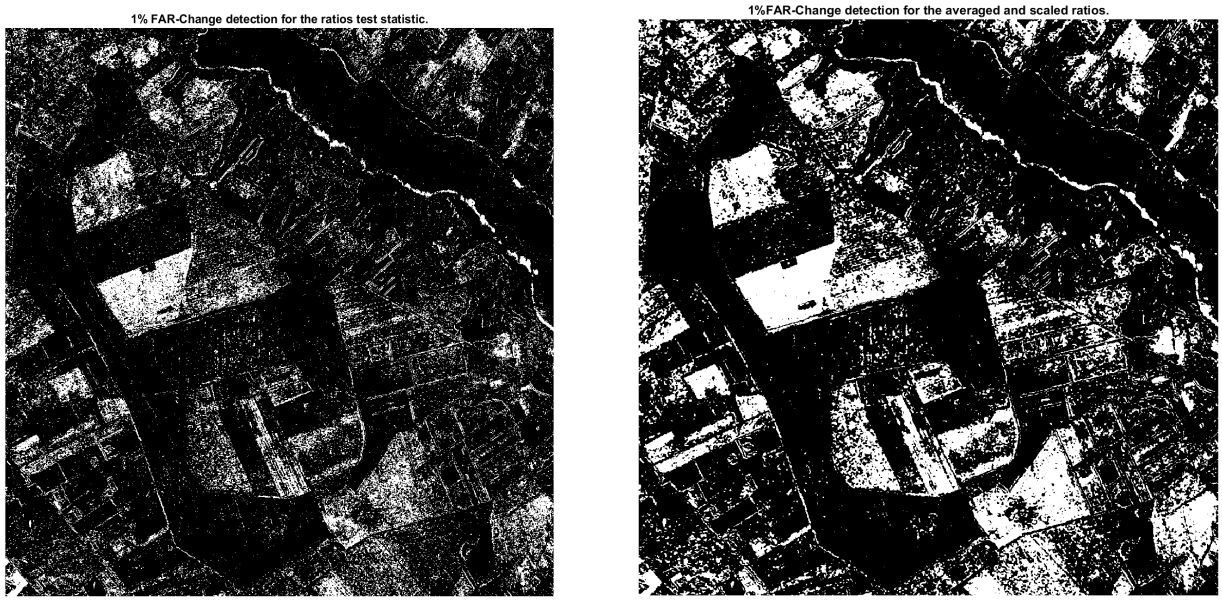


Fig 26 – Ratios-based change detection images before(left) and after the box filter(right)

Fig 27 – (-2ρlnQ)-based change detection images before(left) and after the box filter(right)

The latter, seems to be more sensitive to change detection, something which was expected since the full covariance matrices were included in the change detection algorithm, in contrast with the ratios-based change detection image where only the diagonal elements of the covariance matrices were use. Consequently, the resulting binary image provides better change detection qualities.

Furthermore, the -2ρlnQ based change detection image shows improved qualities on edge detection, line detection and segmentation, as figure 27 clearly illustrates.

In both cases, before and after the box filter was applied, homogeneity seems to improve and therefore classification is is facilitated. However, after the speckle reduction filter, edges in the images seem to be relatively smeared out compared to the initial ones, as averaging pixel values introduces, on some level, a blurring factor.

	Window	***Before filtering (% of window pixels)***	***After filtering (% of window pixels)***
Ratios	***Change area***	***1%***	***1%***
	No-change area	***83.9%***	***94.8%***
-2ρlnQ	***Change area***	***4.2%***	***9%***
	No-change area	***99.3%***	***99.6%***

Table 1 – Percentages of “change pixels” in different cases

The quantitative results, with reference to how many changes were detected in both the ratios and the -2ρlnQ case, are recorded in table-1. Two windows were used in both cases, for 2 different areas typically representing a change and a no-change area, while ensuring that the windows include values that the ratios image was able to detect correctly, in comparison with the -2ρlnQ change detection image.

The ratios-based algorithm was able to detect 83.9% of the corresponding pixels as changes, for the change-area window, while in the -2ρlnQ case, 99.3% of the pixels representing the same areas, were detected as changes, which shows improved change detection qualities.

Change detection seems to deteriorate after the speckle filtering, as dictated by the fact that, for the same no-change area, the test statistic using the -2ρlnQ image resulted in the detection of 9% of the pixels as changed ones, a percentage almost doubled, compared to the one for the image before filtering.

Appendix

```
clear all;
f1 = fopen('fl063_lhhhh','r','b');
HH = fread(f1, [1,inf],'float32');
fclose(f1);
f2 = fopen('fl063_lhvhv','r','b');
HV = fread(f2, [1,inf],'float32');
fclose(f2);
f3 = fopen('fl063_lvvvv','r','b');
VV = fread(f3, [1,inf],'float32');
fclose(f3);
Cm63=get_data63;
Cm64=get_data64;
%% Scale values for HH [ GREEN ],from 0 to 1 for imshow(x)

HH=10*log10(HH); %convert to DB
maximum=max(HH);
scaleHH=0;
minimum=min(HH);
rangeHH=maximum+abs(minimum);
 for i=1:length(HH)
    if HH(i)>10
        scaleHH=scaleHH+1;
        HH(i)=10;
    end
 end
 maximum=10;
 rangeHH=maximum+abs(minimum);
HH=(HH+abs(minimum))./rangeHH;
clear maximum; clear minimum;
%% Scale values for HV [ RED ],from 0 to 1 for imshow(x)

HV=10*log10(HV); %convert to DB
maximum=max(HV);
scaleHV=0;
minimum=min(HV);
rangeHV=maximum+abs(minimum);
 for i=1:length(HV)
    if HV(i)>-1.5
        scaleHV=scaleHV+1;
        HV(i)=-1.5;
    end
 end
 maximum=-1.5;
 rangeHV=maximum+abs(minimum);
HV=(HV+abs(minimum))./rangeHV;
 clear maximum; clear minimum;

%% Scale values for VV [BLUE],from 0 to 1 for imshow(x)
VV=10*log10(VV); %convert to DB
maximum=max(VV);
scaleVV=0;
minimum=min(VV);
rangeVV=maximum+abs(minimum);
 for i=1:length(VV)
    if VV(i)>15
        scaleVV=scaleVV+1;
        VV(i)=15;
    end
 end
 maximum=15;
 rangeVV=maximum+abs(minimum);
VV=(VV+abs(minimum))./rangeVV;
clear maximum; clear minimum;
%% Print RGB image
HVmatr = reshape(HV, 1024, 1024)'; % [R]
HHmatr = reshape(HH, 1024, 1024)'; % [G]
VVmatr = reshape(VV, 1024, 1024)'; % [B]
figure
C=cat(3,HVmatr,HHmatr,VVmatr); % (R,G,B)=(hv,hh,vv)
imshow(C);
saveas(gcf,'63 RGB image.png');
%% Statistics test for HH, between 2 images (simple ratios statistic A=pix1/pix2)
%63
f1 = fopen('fl063_lhhhh','r','b');
HH = fread(f1, [1,inf],'float32');
fclose(f1);
f2 = fopen('fl063_lhvhv','r','b');
HV = fread(f2, [1,inf],'float32');
fclose(f2);

f3 = fopen('fl063_lvvvv','r','b');
VV = fread(f3, [1,inf],'float32');
fclose(f3);
%64
f1 = fopen('fl064_lhhhh','r','b');
HH64 = fread(f1, [1,inf],'float32');
fclose(f1);
f2 = fopen('fl064_lhvhv','r','b');
HV64 = fread(f2, [1,inf],'float32');
fclose(f2);
f3 = fopen('fl064_lvvvv','r','b');
VV64 = fread(f3, [1,inf],'float32');
fclose(f3)
%ratios
HVmatr = reshape(HV, 1024, 1024)'; % [R]
HHmatr = reshape(HH, 1024, 1024)'; % [G]
VVmatr = reshape(VV, 1024, 1024)'; % [B]
```

```
HV64matr = reshape(HV64, 1024, 1024)'; % [R]
HH64matr = reshape(HH64, 1024, 1024)'; % [G]
VV64matr = reshape(VV64, 1024, 1024)'; % [B]
Ar=HVmatr./HV64matr;
Br=HHmatr./HH64matr;
Cr=VVmatr./VV64matr;
%% Display ratios images
imshow(Ar); colorbar; title('Unscaled ratios image')
saveas(gcf,'HV63 to HV64.png');
histogram(Ar); title('Unscaled ratios distribution')
axis([-0.2 10 0 52500])
xlabel('Intensity')
ylabel('Nr of Pixels')
saveas(gcf,'Unscaled ratios distribution.png');
%%
for i=1:1024
    for k=1:1024
        if Ar(i,k)>1
            Arscaled(i,k)= 1/Ar(i,k);
        else
        Arscaled(i,k)=Ar(i,k);
        end
    end
end
%
 histogram(Arscaled); title('Scaled ratios distribution')
 xlabel('Intensity')
ylabel('Nr of Pixels')
 saveas(gcf,'Scaled  ratios distribution.png');
 imshow(1-Arscaled);  title('Scaled ratios image for the HV channel')
 saveas(gcf,'Scaled ratios image for HV63 over HV64.png');
%% Pick threshold for the images using a homogenuous forest area

Arforest=Arscaled(290:369, 35:114);
figure
histogram(Arforest);hold on;  xlabel('Intensity')
ylabel('Nr of Pixels')
title('Forest ratios distribution for HV63 over HV64');
t=0.2549;
z=[1:450];
z1=[ones(1,450)];
z1=t.*z1;
plot(z1,z,'r');
saveas(gcf,'Forest ratios distribution for HV63 over HV64.png');
hold off;
u=0;
for i=1:80
    for k=1:80
        if Arforest(i,k)<=t      % 1% FAR for the ratios image using the 0-5 scaled image.
           u=u+1;
        end
    end
end
FARratios=u/6400
%%
for i=1:1024
    for k=1:1024
        if Arscaled(i,k)<t    %Change
            Aratios(i,k)=1;
        end
        if Arscaled(i,k)>t  %No change
                Aratios(i,k)=0;
        end
    end
end
figure
imshow(Aratios);
title('1% FAR-Change detection for the ratios test statistic.');
saveas(gcf,'1% fractile change detection for ratios statistic test(HV-HV64) .png');
%% Count detected changes for the ratios image
RatC=Aratios(290:369, 35:114);       %no change area
u=0;
for i=1:80
    for k=1:80
        if RatC(i,k)==1
           u=u+1;
        end
    end
end
uNC=u/(80*80)*100
RatNC=Aratios(450:530,210:290);      %change area
u=0;
for i=1:80
    for k=1:80
        if RatNC(i,k)==1
           u=u+1;
        end
    end
end
uC=u/(80*80)*100
%% lnQ
n=10; %nr of looks
p=3; %For a 3x3 covariance matrix p=3
X = CMIm(Cm63);
Y = CMIm(Cm64);
XpY = CMIm(Cm63 + Cm64);
lnQ = n*(6*log(2) + log(X) + log(Y) - 2*log(XpY));
%% Scale lnQ for display
maxlnQ=max(max(-lnQ));
minlnQ=min(min(-lnQ));
u=0;
for i=1:1024
```

```
    for k=1:1024
        if -lnQ(i,k)>=32
            u=u+1;
        end
    end
end
%% Display -lnQ and calculate rho, omega, and -2rholnQ
rangelnQ=maxlnQ-minlnQ;
lnQscaled=-lnQ./32;
rho=1-((2*p^2-1)/(6*p))*(2/10-1/20);
omega=-p^2/4*(1-1/rho)^2+p^2*(p^2-1)/24;
P=-2*rho.*lnQ;
figure
imshow(lnQscaled); colorbar;
title('-lnQ image');
saveas(gcf,'inverted lnQ image.png');
figure
imshow(P./32); colorbar;
title('2rholnQ image (inverted and scaled)');
saveas(gcf,'2rholnQ image (inverted and scaled).png');
%% Test lnq for a forest area
lnqforest=-1.*lnQ(350:440 , 30:110);
imshow(lnqforest);
%% False alarm rate
o=2*rho*lnqforest;
histogram(o);
t=17.5;
u=0;
for i=1:91
    for k=1:81
       if o(i,k)>t                    %FAR
           u=u+1;
    end
    end
end
u=u/(91*81);
%% Chi squared fit
a=[0:0.00135:10];                    %~91*81 values
prof9=chi2inv(a,9);
prof13=chi2inv(a,13);
probforest=prof9                     % Last terms ignored
subplot(2,1,1)
histfit(probforest);
title('Theoretical ?^2 distribution');
hold on;
z=[1:100];
z1=[ones(1,100)];
z1=t.*z1;
plot(z1,z,'r');
hold off;
subplot(2,1,2)
o=reshape(o,1,7371);
histfit(o);
axis([-5 25 0 400])
title('-2?lnQ distribution (forest area)');
saveas(gcf,'Theoretical vs actual forest distribution.png');
%% Threshold implementation

for i=1:1024
    for k=1:1024
        if P(i,k)>=t

            LNQtest(i,k)=1;                 %No change
        end
        if P(i,k)<17.5
             LNQtest(i,k)=0;
        end
    end
end
imshow(LNQtest);
title('-2?lnQ change detection image using 1%FAR');
saveas(gcf,'1% fractile test.png');
%% Count detected changes for the -2rholnQ image
lnQC=LNQtest(290:369, 35:114);       %no change area
u=0;
for i=1:80
    for k=1:80
        if lnQC(i,k)==1
           u=u+1;
        end
    end
end
uNC=u/(80*80)*100

lnQNC=LNQtest(450:530,210:290);      %change area
u=0;
for i=1:80
    for k=1:80
        if lnQNC(i,k)==1
           u=u+1;
        end
    end
end
uC=u/(80*80)*100
```

%%Speckle reduction script%%

```
%% The results are 1024x1024 matrices where the border are set equal to 0
% Load 63
clear all;
file_hhhh = fopen('fl063_lhhhh','r','b');
hhhh=fread(file_hhhh,[1,inf],'float32');
fclose(file_hhhh);
HH3=hhhh;
file_hhhv = fopen('fl063_lhhhv','r','b');
hhhv0=fread(file_hhhv,[2,inf],'float32');
hhhv=hhhv0(1,:)+j*hhhv0(2,:);
fclose(file_hhhv);
HHHV3=hhhv;
file_hvhv = fopen('fl063_lhvhv','r','b');
hvhv=fread(file_hvhv,[1,inf],'float32');
fclose(file_hvhv);
HV3=hvhv;
file_hhvv = fopen('fl063_lhhvv','r','b');
hhvv0=fread(file_hhvv,[2,inf],'float32');
hhvv=hhvv0(1,:)+j*hhvv0(2,:);
fclose(file_hhvv);
HHVV3=hhvv;
file_hvvv = fopen('fl063_lhvvv','r','b');
hvvv0=fread(file_hvvv,[2,inf],'float32');
hvvv=hvvv0(1,:)+j*hvvv0(2,:);
fclose(file_hvvv);
HVVV3=hvvv;
file_vvvv = fopen('fl063_lvvvv','r','b');
vvvv=fread(file_vvvv,[1,inf],'float32');
fclose(file_vvvv);
VV3=vvvv;
%%  First averaged data set
    HH3f = BoxFilter(HH3);
    HV3f = BoxFilter(HV3);
    VV3f = BoxFilter(VV3);
    HHHV3f = BoxFilter(HHHV3);
    HHVV3f = BoxFilter(HHVV3);
    HVVV3f = BoxFilter(HVVV3);
%% Scale HH3fs
HH3fs=10.*log10(HH3f); %convert to DB
maximum=max(max(HH3fs));
scaleHH3fs=0;
minimum=min(min(HH3fs));
rangeHH3fs=maximum+abs(minimum);
 for i=1:length(HH3fs)
    if HH3fs(i)>10
        scaleHH3fs=scaleHH3fs+1;
        HH3fs(i)=10;
    end
 end
 maximum=10;
 rangeHH3fs=maximum+abs(minimum);
HH3fs=(HH3fs+abs(minimum))./rangeHH3fs;
clear maximum; clear minimum;
   %% Scale HV3fs
   HV3fs=10.*log10(HV3f); %convert to DB
maximum=max(max(HV3fs));
scaleHV3fs=0;
minimum=min(min(HV3fs));
rangeHV3fs=maximum+abs(minimum);
 for i=1:length(HV3fs)
    if HV3fs(i)>10
        scaleHV3fs=scaleHV3fs+1;
        HV3fs(i)=10;
    end
 end
 maximum=10;
 rangeHV3fs=maximum+abs(minimum);
HV3fs=(HV3fs+abs(minimum))./rangeHV3fs;
clear maximum; clear minimum;
%% Scale VV3fs
VV3fs=10.*log10(VV3f); %convert to DB
maximum=max(max(VV3fs));
scaleVV3fs=0;
minimum=min(min(VV3fs));
rangeVV3fs=maximum+abs(minimum);
 for i=1:length(VV3fs)
    if VV3fs(i)>10
        scaleVV3fs=scaleVV3fs+1;
        VV3fs(i)=10;
    end
 end
 maximum=10;
 rangeVV3fs=maximum+abs(minimum);
VV3fs=(VV3fs+abs(minimum))./rangeVV3fs;
clear maximum; clear minimum;
%% Plot averaged RGB
% HV3fsmatr = reshape(HV3fs, 1022, 1022)'; % [R]
% HH3fsmatr = reshape(HH3fs, 1022, 1022)'; % [G]
% VV3fsmatr = reshape(VV3fs, 1022, 1022)'; % [B]
figure
C3fs=cat(3,HV3fs,HH3fs,VV3fs); % (R,G,B)=(hv,hh,vv)
imshow(C3fs);
saveas(gcf,'Averaged 63 RGB image.png');
%% Load 64
file_hhhh = fopen('fl064_lhhhh','r','b');
hhhh=fread(file_hhhh,[1,inf],'float32');
fclose(file_hhhh);
HH4=hhhh;
file_hhhv = fopen('fl064_lhhhv','r','b');
hhhv0=fread(file_hhhv,[2,inf],'float32');
hhhv=hhhv0(1,:)+j*hhhv0(2,:);
fclose(file_hhhv);
HHHV4=hhhv;
file_hvhv = fopen('fl064_lhvhv','r','b');
hvhv=fread(file_hvhv,[1,inf],'float32');
fclose(file_hvhv);
HV4=hvhv;
file_hhvv = fopen('fl064_lhhvv','r','b');
hhvv0=fread(file_hhvv,[2,inf],'float32');
hhvv=hhvv0(1,:)+j*hhvv0(2,:);
fclose(file_hhvv);
HHVV4=hhvv;
file_hvvv = fopen('fl064_lhvvv','r','b');
hvvv0=fread(file_hvvv,[2,inf],'float32');
hvvv=hvvv0(1,:)+j*hvvv0(2,:);
fclose(file_hvvv);
HVVV4=hvvv;
file_vvvv = fopen('fl064_lvvvv','r','b');
vvvv=fread(file_vvvv,[1,inf],'float32');
fclose(file_vvvv);
```

```
VV4=vvvv;
%% Second averaged data set
    HH4f = BoxFilter(HH4);
    HV4f = BoxFilter(HV4);
    VV4f = BoxFilter(VV4);
    HHHV4f = BoxFilter(HHHV4);
    HHVV4f = BoxFilter(HHVV4);
    HVVV4f = BoxFilter(HVVV4);


%% Scale HH4fs
HH4fs=10.*log10(HH4f); %convert to DB
maximum=max(max(HH4fs));
scaleHH4fs=0;
minimum=min(min(HH4fs));
rangeHH4fs=maximum+abs(minimum);
 for i=1:length(HH4fs)
    if HH4fs(i)>10
        scaleHH4fs=scaleHH4fs+1;
        HH4fs(i)=10;
    end
 end
 maximum=10;
 rangeHH4fs=maximum+abs(minimum);
HH4fs=(HH4fs+abs(minimum))./rangeHH4fs;
clear maximum; clear minimum;
%% Scale HV4fs
   HV4fs=10.*log10(HV4f); %convert to DB
maximum=max(max(HV4fs));
scaleHV4fs=0;
minimum=min(min(HV4fs));
rangeHV4fs=maximum+abs(minimum);
 for i=1:length(HV4fs)
    if HV4fs(i)>10
        scaleHV4fs=scaleHV4fs+1;
        HV4fs(i)=10;
    end
 end
 maximum=10;
 rangeHV4fs=maximum+abs(minimum);
HV4fs=(HV4fs+abs(minimum))./rangeHV4fs;
clear maximum; clear minimum;
%% Scale VV4fs
VV4fs=10.*log10(VV4f); %convert to DB
maximum=max(max(VV4fs));
scaleVV4fs=0;
minimum=min(min(VV4fs));
rangeVV4fs=maximum+abs(minimum);
 for i=1:length(VV4fs)
    if VV4fs(i)>10
        scaleVV4fs=scaleVV4fs+1;
        VV4fs(i)=10;
    end
 end
 maximum=10;
 rangeVV4fs=maximum+abs(minimum);
VV4fs=(VV4fs+abs(minimum))./rangeVV4fs;
clear maximum; clear minimum;
%% Plot Averaged 64 image
figure
C4fs=cat(3,HV4fs,HH4fs,VV4fs); % (R,G,B)=(hv,hh,vv)
imshow(C4fs);
saveas(gcf,'Averaged 64 RGB image.png');
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Ratios for the averaged images
Ara=HV3f./HV4f;
Bra=HH3f./HH4f;
Cra=VV3f./VV4f;
figure % (R,G,B)=(hv,hh,vv)
imshow(Ara);
saveas(gcf,'Averaged HV63-HV64 ratios image.png');
%%
for i=1:1022
    for k=1:1022
        if Ara(i,k)>1
            Arascaled(i,k)= 1/Ara(i,k);



        else
        Arascaled(i,k)=Ara(i,k);
        end
    end
end
figure
 histogram(Arascaled);title('Averaged and scaled HV63/HV64 ratios distribution');
xlabel('Intensity')
ylabel('Nr of Pixels')
saveas(gcf,'Averaged and scaled HV63-HV64 ratios distribution.png');
figure
imshow(1-Arascaled);title('Averaged and scaled HV63/HV64 ratios image')
saveas(gcf,'Averaged and scaled HV63-HV64 ratios image.png');
%% Pick threshold for the images using homogenuous forest area
Araforest=Arascaled(290:369, 35:114);
figure
histogram(Araforest);hold on;
title('Forest ratios distribution for averaged 63 over 64 HH image');
t=0.373;                                        %% Averaged FAR
z=[1:700];
z1=[ones(1,700)];
z1=t.*z1;
plot(z1,z,'r');
saveas(gcf,'Forest ratios distribution for averaged HV63 over HV64.png');
hold off;
u=0;
for i=1:80
    for k=1:80
        if Araforest(i,k)<t      % 1% FAR for the ratios image using the 0-5 scaled image.
           u=u+1;
        end
    end
end
FARratiosa=u/6400
%% %%
for i=1:1022
    for k=1:1022
        if Arascaled(i,k)<t    %Change
            Araatios(i,k)=1;
        end
        if Arascaled(i,k)>=t  %No change
                Araatios(i,k)=0;
        end
    end
```

```
end
figure
imshow(Araatios);
title('1%FAR-Change detection for the averaged and scaled ratios.');
saveas(gcf,'1%FAR-Change detection for the averaged and scaled ratios.png');
%% Count detected changes for the ratios image
RatNC=Araatios(290:369, 35:114);       %no change area
u=0;
for i=1:80
    for k=1:80
        if RatNC(i,k)==1
           u=u+1;
        end
    end
end
uNC=u/(80*80)*100
RatC=Araatios(450:530,210:290);      %change area
u=0;
for i=1:80
    for k=1:80
        if RatC(i,k)==1
           u=u+1;
        end
    end
end
uC=u/(80*80)*100
%% Averaged CMs for lnQ %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Averaged CM63
HH3fr=reshape(HH3f,1,1044484);
HHHV3fr=reshape(HHHV3f,1,1044484);
HHVV3fr=reshape(HHVV3f,1,1044484);
HV3fr=reshape(HV3f,1,1044484);
HVVV3fr=reshape(HVVV3f,1,1044484);
VV3fr=reshape(VV3f,1,1044484);
Cm3a(1,1,:)=HH3fr;
Cm3a(1,2,:)=HHHV3fr;
Cm3a(1,3,:)=HHVV3fr;
Cm3a(2,1,:)=conj(HHHV3fr);
Cm3a(2,2,:)=HV3fr;
Cm3a(2,3,:)=HVVV3fr;
Cm3a(3,1,:)=conj(HHVV3fr);
Cm3a(3,2,:)=conj(HVVV3fr);
Cm3a(3,3,:)=VV3fr;
%% Averaged CM64
HH4fr=reshape(HH4f,1,1044484);
HHHV4fr=reshape(HHHV4f,1,1044484);
HHVV4fr=reshape(HHVV4f,1,1044484);
HV4fr=reshape(HV4f,1,1044484);
HVVV4fr=reshape(HVVV4f,1,1044484);
VV4fr=reshape(VV4f,1,1044484);
Cm4a(1,1,:)=HH4fr;
Cm4a(1,2,:)=HHHV4fr;
Cm4a(1,3,:)=HHVV4fr;
Cm4a(2,1,:)=conj(HHHV4fr);
Cm4a(2,2,:)=HV4fr;
Cm4a(2,3,:)=HVVV4fr;
Cm4a(3,1,:)=conj(HHVV4fr);
Cm4a(3,2,:)=conj(HVVV4fr);
Cm4a(3,3,:)=VV4fr;
%% lnQ
na=24; %nr of looks
p=3; %For a 3x3 covariance matrix p=3
Xa = CMImAverage(Cm3a);
Ya = CMImAverage(Cm4a);
XpYa = CMImAverage(Cm3a + Cm4a);
lnQa = na*(6*log(2) + log(Xa) + log(Ya) - 2*log(XpYa));
%% Scale lnQ for display
maxlnQa=max(max(-lnQa));
minlnQa=min(min(-lnQa));
u=0;
for i=1:1022
    for k=1:1022
        if -lnQa(i,k)>=62
            u=u+1;
        end
    end
end
%% Display -lnQ and calculate rho, omega, and -2rholnQ
rangelnQa=maxlnQa-minlnQa;
lnQscaleda=-lnQa./52;
rhoa=1-((2*p^2-1)/(6*p))*(2/24-1/48);
omega=-p^2/4*(1-1/rhoa)^2+p^2*(p^2-1)/24*(1/(na^2)+1/(na^2)-1/((2*na)^2))*1/(rhoa^2);
Pa=-2*rhoa.*lnQa;
figure
imshow(lnQscaleda');
title('-lnQ image-Averaged covariance matrices');
saveas(gcf,'Averaged CM case-inverted  lnQ image.png');
% 1% fractile for the averaged case%%%%%%%%%%%%%%%
%% Test lnq for a forest area
oa=Pa(350:440,30:110);
histogram(oa);
%% False alarm rate
u=0;
for i=1:91
    for k=1:81
       if oa(i,k)>150                    %FAR
           u=u+1;
    end
    end
end
u=u/7371*100
%% Chi squared fit
omega=-(p^2/4)*(1-1/rhoa)^2+(p^2*(p^2-1)/24)*(1/(na^2)+1/(na^2)-1/((2*na)^2))*1/(rhoa^2);
probforesta=chi2stat(oa)+omega.*(chi2stat(oa)-chi2stat(oa));
histogram(probforesta);
%% Threshold implementation
for i=1:1022
    for k=1:1022
        if Pa(i,k)>=150

            LNQtesta(i,k)=1;                 %No
        end
        if Pa(i,k)<150
             LNQtesta(i,k)=0;
        end
    end
end
 imshow(LNQtesta');colorbar;
title('1% fractile image-Averaged covariance matrices');
saveas(gcf,'1% fractile for the averaged CM case.png');
%% Count detected changes for the -2rholnQ image
lnQC=LNQtesta(290:369, 35:114);       %no change area
```

```
u=0;
for i=1:80
    for k=1:80
        if lnQC(i,k)==1
           u=u+1;
        end
    end
end
uNC=u/(80*80)*100

lnQNC=LNQtesta(450:530,210:290);      %change area
u=0;
for i=1:80
    for k=1:80
        if lnQNC(i,k)==1
           u=u+1;
        end
    end
end
uC=u/(80*80)*100
```

Functions

```
%%%%%%%%%%
function [ X ] = CMIm( Cm )
    L = length(Cm);
    for k = 1:L
        V(k) = real(det(Cm(1:3, 1:3, k)));
    end

    X = (reshape(V, 1024, 1024))';
End

 %%%%%%%%%%
function [ X ] = CMImAverage( Cm )
    L = length(Cm);
    for k = 1:L
        V(k) = real(det(Cm(1:3, 1:3, k)));
    end

    X = (reshape(V, 1022, 1022))';
End

%%%%%%%%%%
function CovMat = get_data();
file_hhhh = fopen('f1063_1hhhh','r','b');
hhhh=fread(file_hhhh,[1,inf],'float32');
fclose(file_hhhh);

file_hhhv = fopen('f1063_1hhhv','r','b');
hhhv0=fread(file_hhhv,[2,inf],'float32');
hhhv=hhhv0(1,:)+j*hhhv0(2,:);
fclose(file_hhhv);

file_hvhv = fopen('f1063_1hvhv','r','b');
hvhv=fread(file_hvhv,[1,inf],'float32');
fclose(file_hvhv);

file_hhvv = fopen('f1063_1hhvv','r','b');
hhvv0=fread(file_hhvv,[2,inf],'float32');
hhvv=hhvv0(1,:)+j*hhvv0(2,:);
fclose(file_hhvv);

file_hvvv = fopen('f1063_1hvvv','r','b');
hvvv0=fread(file_hvvv,[2,inf],'float32');
hvvv=hvvv0(1,:)+j*hvvv0(2,:);
fclose(file_hvvv);

file_vvvv = fopen('f1063_1vvvv','r','b');
vvvv=fread(file_vvvv,[1,inf],'float32');
fclose(file_vvvv);
l=length(hhhh);
CM=zeros(3,3,l);
CM(1,1,:)=hhhh;
CM(1,2,:)=hhhv;
CM(1,3,:)=hhvv;
CM(2,1,:)=conj(hhhv);
CM(2,2,:)=hvhv;
CM(2,3,:)=hvvv;
CM(3,1,:)=conj(hhvv);
CM(3,2,:)=conj(hvvv);
CM(3,3,:)=vvvv;
CovMat=CM;

%%%%%%%%%%
function [ N ] = EqNLooks( HH, HV, VV )
% Area 1 - 3 channels
    x1 = int64(360);
    x2 = int64(410);
    y1 = int64(205);
    y2 = int64(255);

    [N(1)] = looks(HH, x1, x2, y1, y2);
    [N(2)] = looks(HV, x1, x2, y1, y2);
    [N(3)] = looks(VV, x1, x2, y1, y2);

% Area 2 - 3 channels
    x1 = int64(735);
    x2 = int64(770);
    y1 = int64(360);
    y2 = int64(395);

    [N(4)] = looks(HH, x1, x2, y1, y2);
    [N(5)] = looks(HV, x1, x2, y1, y2);
    [N(6)] = looks(VV, x1, x2, y1, y2);

% Area 3 - 3 channels
    x1 = int64(110);
    x2 = int64(170);
    y1 = int64(610);
    y2 = int64(665);

    [N(7)] = looks(HH, x1, x2, y1, y2);
    [N(8)] = looks(HV, x1, x2, y1, y2);
    [N(9)] = looks(VV, x1, x2, y1, y2);
```

ENL Script

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Equivalent Number of Looks Estimation   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Assumed that the data for the three channels hhhh, hvhv, vvvv are already
% available (execute after "images")

% Normal
    HH = reshape(HH3, 1024, 1024)';
    HV = reshape(HV3, 1024, 1024)';
    VV = reshape(VV3, 1024, 1024)';
    N = EqNLooks(HH, HV, VV);
    n = round(mean(N));

% After Despeckle
%      Z1 = zeros(1024, 1024);
%      Z2 = zeros(1024, 1024);
%      Z3 = zeros(1024, 1024);
%
%      Z1(2:1023,2:1023) = HH3f;
%      Z2(2:1023,2:1023) = HV3f;
%      Z3(2:1023,2:1023) = VV3f;
%      Nf = EqNLooks(Z1, Z2, Z3);
%      nf = round(mean(Nf));
end
```