Universität Stuttgart
Geodätisches Institut

Change detection using SAR data

Diplomarbeit im Studiengang
Geodäsie und Geoinformatik
an der Universität Stuttgart

Wenxi Cao

Stuttgart, September 2013

Betreuer: Prof. Dr.-Ing. Nico Sneeuw
Universität Stuttgart

Dr. Sandro Martinis
German Aerospace Center (DLR)

Erklärung der Urheberschaft

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Ort, Datum Unterschrift

Abstract

The objective of this thesis is to find changes caused by natural disaster from two co-registered calibrated TerraSAR-X images. Three methods are used in this thesis. The first method histogram thresholding, uses the histogram of the SAR intensity ratio image to classify the ratio image into three classes. This technique was originally proposed by Kittler et al. (1986) and modified by Bazi et al. (2005) and Moser et al. (2006) based on the Bayesian formula. In this thesis their methods are combined together to detect three classes. The relative difference of the cost function is used to detect the number of the classes instead of the determinant of the Hessian matrix suggested by Bazi et al. (2005). The second method formulates the classification problem as a hypothesis testing problem. This idea was originally used by Touzi et al. (1988) and Oliver et al. (1996). In this thesis the analytical method by Touzi et al. (1988) is replaced by using the properties of the Gamma distribution. The third method, graph-cut algorithm, is a post-processing method, which improves classification results from the first and second methods. The improvement is equivalent to the global optimization of an energy function in a Markov random field (MRF). A modern method proposed by Kolmogorov et al. (2004) and Boykov et al. (2004) is used in this thesis. This method transforms the energy function of a MRF into an equivalent graph and solves the global optimization problem using a max-flow/min-cut algorithm. These three methods are applied to the test data on Queensland, Australia, and Leipzig, Germany. Most SAR ratio images can be classified into three classes successfully. The remaining problem is that the interpretation of the changed classes is still ambiguous. Other data sources should be combined to assist or improve the interpretation of the detected change.

Zusammenfassung

Das Ziel der Diplomarbeit ist die von Naturkatastrophen verursachten Änderungen an der Erdoberfläche mittels zwei ko-registrierten kalibrierten TerraSAR-X Intensitätsbildern zu identifizieren. Drei Methoden werden in der Diplomarbeit verwendet. Die erste Methode, Histogram Schwellwertverfahren, vewendet das Histogram vom SAR-Ratiobild um das Ratiobild in drei Klassen zu segmentieren. Diese Technik wurde zuerst von Kittler et al. (1986) veröffentlicht und von Bazi et al. (2005) und Moser et al. (2006) ans Sicht der Bayes'schen Formel weiterentwickelt. Ihre Methoden werden in der Diplomarbeit kombiniert. Der relative Unterschied der Kostenfunktion wird zur Detektion von Klassenanzahl verwerdet anstatt Determinant der Hessematrix, die von Bazi et al. (2005) verwendet wurde. Die zweite Methode beschreibt die Klassifikation als einen statistischen Test. Diese Idee wurde zuerst von Touzi et al. (1988) und Oliver et al. (1996) verwendet. In der Diplomarbeit wird die analytische Methode von Touzi et al. (1988) durch Anwendung der Eigenschaften von Gamma-Verteilungsfunktion ausgewechselt. Die dritte Methode, Graph-cut, ist ein Post-processing Verfahren, das die Klassifikationsmasken der ersten und zweiten Methoden verbessert. Die Verbesserung ist äquivalent zur globalen Optimierung einer Energiefunktion in einem Markov random field (MRF). Eine moderne Methode von Kolmogorov et al. (2004) und Boykov et al. (2004) wird verwendet um die Optimierung durchzuführen. Diese Methode wandelt die Energiefunktion in einen äquivalenten Graph um und löst das Problem der globalen Optimierung mittels ein Max-flow/Min-cut Verfahren auf. Diese drei Methoden werden auf die Testdaten von Queensland, Australien, und Leipzig, Deutschland, angewendet. Die meisten SAR Bilder von Testdaten können erfolgreich in drei Klassen klassifiziert werden. Das übrige Problem ist die nicht eindeutige Interpretation von Änderungen. Zusätzliche Daten sollten kombiniert werden um die Interpretation zu unterstützen oder verbessern.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to change detection

Change detection is the process of identifying differences in the state of an object or phenomenon by observing it at different times (Lu et al., 2004). The goal of change detection is to detect "significant" changes while rejecting "unimportant" ones (Radke et al., 2005). In general these significant changes reflect the process of some natural phenomenons or activities of humans on the Earth's surface. Some applications of change detection to remote sensing are listed in (Lu et al., 2004) as follows:

- land-use and land-cover,
- forest or vegetation change,
- wetland change,
- forest fire,
- environmental change,
- etc.

In general, change detection enables better understanding of relationship and interaction between human and natural phenomena and better management and usage of resources (Lu et al., 2004).

Besides the applications above natural disaster monitoring plays an important role because of its close relation to human life. Natural disaster can affect residence of human beings, cause economic damage and even take lives of thousands of people every year. It is essential to monitor the natural disaster and provide valid near real-time information about development of natural disaster, so that rescue work can be conducted in time to keep the loss as less as possible. To fulfill this task a satellite called TerraSAR-X (*Astrium*, n.d.*a*) was launched to observe the Earth's surface. TerraSAR-X uses a X-band synthetic aperture radar (SAR) sensor. The advantage of SAR is that it is an active sensor and can penetrate the cloud, so its observations are almost independent of weather condition and solar illumination. The department, Center for Satellite Based Crisis Information (ZKI) in the German Remote Sensing Data Center (DFD) at German Aerospace Center (DLR), provides a service for the rapid provision, processing and analysis of satellite imagery during natural and environmental disasters (ZKI, 2013). The calibrated TerraSAR-X data are also used by ZKI to extract relevant crisis information. This thesis will use TerraSAR-X data to find changes caused by natural disaster, the designed methods are expected to be integrated into the crisis response service by ZKI, or at least help ZKI to improve their service.

There are many change detection techniques in the literature. Lu et al. (2004) and Radke et al. (2005) provided good reviews for the present methods. The following categories of change detection were summarized by Lu et al. (2004):

1. Algebraic method. It tries to find a test variable from image series, and compare this test variable with a suitable threshold to identify which areas have changed. This category includes image differencing, image ratioing, image regression, vegetation index differencing, change detector analysis (CDA) and background subtraction.
2. Transformation method. It tries to deal with each image recorded at different times as a single layer of a synthetic multi-spectral image. Then transforms this multi-spectral image into corresponding major and minor components. The changes are then included in the minor component. The mostly used transformations in this category are PCA, KT, Gramm-Schmidt, and Chi-square transformations.
3. Classification. It also merge all images of different dates into a synthetic multi-spectral image, and uses a classification method (Bishop and Nasrabadi, 2006) to identify which area has changed. The mostly used classification techniques are post-classification comparison, spectral-temporal combined analysis, expectation-maximization, unsupervised change detection, hybrid change detection and artificial neural network (ANN).
4. Advanced model. It uses a biophysical model related to the scattering process of vegetation to find changed areas.

There is no ideal algorithm for all problems. The algebraic method is easy to implement, but it is hard to find a suitable threshold to decide which areas have changed. Other methods in the list above outperform the algebraic method, but they are complex and time-consuming. The methods used in this thesis are expected to detect changes caused by natural disaster, so efficient algorithms are preferred in order to get near real-time information as soon as possible.Thesis focuses at first on the algebraic method. One of them based on the hypothesis testing is used in this thesis to find changes. Generally the classification method can be divided into two groups: supervised and unsupervised methods. The first one needs a large volume of learning data and the second one is very time-consuming. However, a special version of classification method is used based on histogram, which can simply the data such that the data volume and the computation time are largely reduced. Although these two algorithms are easy to implement, their performance is only in the intermediate level. According to (Radke et al., 2005) the concept of Markov random field (MRF) can be used as a post-processing step to improve the classification results. So a algorithm based on MRF is also added in this thesis.

1.2 Structure of this thesis

Firstly, the TerraSAR-X satellite and SAR data are introduced in the chapter 2. In the following two chapters, two methods are used to obtain a coarse classification. In chapter 3 a method based on the histogram thresholding is introduced. This technique is originally introduced by Bazi et al. (2005) and Moser et al. (2006) to improve the classical method by Kittler and Illingworth (1986) in the framework of *maximum a posteriori probability* (MAP). However, Moser and Serpico (2006) only restricted their work to two-class classification, but they used a powerful method to estimate parameters of SAR models. Bazi et al. (2005) discussed the three-class problem but used the general Gaussian distribution whose parameters are difficult to estimate. In

chapter 3 these two ideas are combined together. The three-class problem in the framework of MAP is discussed as shown by Bazi et al. (2005), but their model is replaced with other easier mathematical models, so that the parameters can be estimated easily. The two-class problem can be considered as a special case of three-class problem. In order to distinguish them Bazi et al. (2005) suggested the determinant of the Hessian matrix as the criterion. This criterion is replaced by the relative difference of the cost function, which is numerically easier and more robust.

In chapter 4 the second method based on the hypothesis testing is discussed. This idea is at first published by Touzi et al. (1988) to detect edges in SAR images. It was further used by Oliver (1996) to detect changes from two SAR images. But the formula by Oliver (1996) is a difficult series expansion. This formula is replaced using the properties of the Gamma distribution function. At the end the test variable can be related to a special function, which is already a built-in routine in many softwares.

In chapter 5 an optimization method based on the graph-cut algorithm is used to improve the classification results. This method was invented by Boykov et al. (2004) and Kolmogorov et al. (2004). At first, the general concept of graph transformation for an energy function is introduced, then a brief introduction of the max-flow algorithm is given. This graph-cut method has already been implemented in a library of C++ source code. Two of the functions in this library are used to carry out the graph-cut optimization.

In chapter 6 a few test data are processed by the algorithms in this thesis and the results are discussed. Chapter 7 is devoted to conclusion and outlook. Some mathematical proofs and the source codes in this thesis are listed in the appendices. The codes are written in IDL and C++ programming language.

The TerraSAR-X data used in this thesis are two test datasets recorded in horizontal polarized ScanSAR mode. The first one is observed on Queensland, Australia, the second one is near Leipzig, Germany.

Chapter 2

Satellite and data

2.1 Introduction to TerraSAR-X and data

There are many types of satellite sensors to observe the Earth's surface, such as optical cameras and *synthetic aperture radar* (SAR). The advantage of SAR is that it is an active system, the operation time is independent of solar illumination, so that SAR can observe the Earth's surface 24 hours a day without interruption. In addition, SAR system uses microwaves to penetrate the clouds. This character enables SAR sensor to performance all-weather monitoring. Compared to other passive sensors, SAR is a more appropriate and powerful disaster monitoring instrument. SAR also finds other applications in the scientific areas such as landcover mapping, environment monitoring, cartography and others.

To fulfill the demands in scientific research and public interest an SAR satellite called *TerraSAR-X* was launched on 15 June, 2007 by German Aerospace Center (DLR) and the company Astrium of the European Aeronautic Defense and Space Company (EADS) (*Astrium*, n.d.*a*). TerraSAR-X has been in operational service since January 2008 at the height of 514 km above the equator with the orbital inclination 97.44°. This orbital configuration enables TerraSAR-X to synchronize its solar cell panels with the solar position to ensure an optimal energy supply. The antenna length is 0.7 m, along with the X-band signal (wavelength at 31 mm) TerraSAR-X can get a theoretical resolution better than 1 m. The phased-array antenna on TerraSAR-X can change its look angle according to its operation modes. More details about the satellite sensor can be found in (*DLR*, 2009). There are three types of modes for different scientific applications:

- SpotLight, up to 2 m resolution,
- StripMap, up to 3 m resolution,
- ScanSAR, up to 18.5 m resolution.

These modes are demonstrated in figure (2.1). The StripMap mode uses the beam in a fixed direction. As the satellite moves relatively to the Earth's surface, the SAR sensor scans a strip in the flight direction. This is the most commonly used mode for SAR. The ScanSAR mode changes the beam direction during the flight, so there are more strips observed at the same time, in this way the scanned area is increased. The incidence angle range for StripMap and ScanSAR modes is $20^\circ \sim 45^\circ$. The SpotLight mode uses a tracking beam which focuses on a small area for a long time. The consequence is the reduction of scanned area but increase of resolution. The incidence angle range for SpotLight SAR mode is $20^\circ \sim 55^\circ$. If the priority is the coverage of observation, ScanSAR mode is preferred. A higher coverage (swath) also means a higher repetition rate of observation. This is important for disaster monitoring because the

area of interest should be observed as quickly as possible. If the resolution is more important, SpotLight mode is a better choice (*Astrium*, 2002).

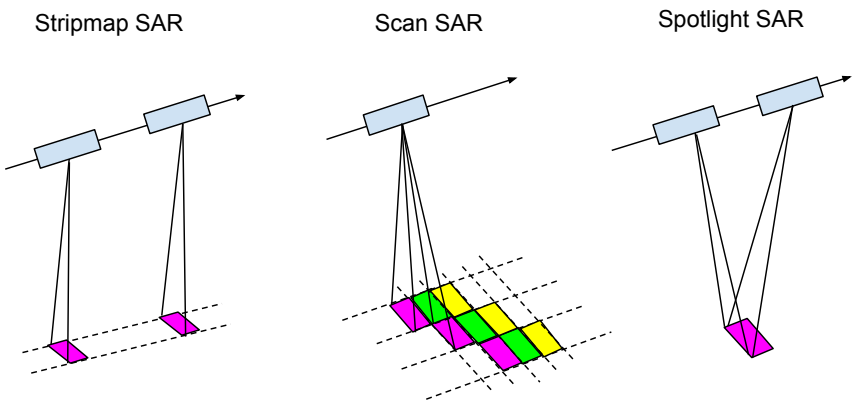


Figure 2.1: *SAR modes for TerraSAR-X (*Astrium*, n.d.*a*).*

Observations from TerraSAR-X sensor are downloaded to the ground segment in DLR, then calibrated by the TerraSAR-X Multi Mode SAR Processor (TMSP). There are generally two types of products (*Astrium*, 2002):

- spatially enhanced (SE) products,
- radiometrically enhanced (RE) products.

The first one is so calibrated that the highest geometrical resolution is possible. The second one is optimized with respect to radiometry, approximately six looks are used to average the signal and reduce the speckle effect.

In terms of the data representation, the processed SAR data can be grouped into following four types (*Astrium*, 2002):

- single-look slant range complex (SSC) data,
- multi-look ground range detected (MGD) data,
- enhanced ellipsoid corrected (EEC) data,
- geocoded ellipsoid corrected (GEC) data.

The SSC data are represented as complex numbers, the pixels in this data are equidistant in azimuth and slant range direction. The SSC product is intended for scientific applications that require the full bandwidth and the phase information, for example, SAR interferometry. MGD data are represented in azimuth and ground range direction. Multi-looking technique is applied to reduce the speckle effect. Slant range coordinates are projected to ground range using a WGS84 ellipsoid and an average, constant terrain height. The advantage of this type of data is that there is no rotation of data to a map coordinate system, so the data are more accurate than the geocoded products. EEC data are also processed through multi-looking to reduce the speckle effect, they are re-sampled to a WGS84 reference ellipsoid assuming one average terrain height and then mapped to a plane through UTM or UPS projection. The last one GEC

data uses further a *digital elevation model* (DEM) to correct the distortion caused by the terrain variation.

2.2 Input data

In this thesis the radiometrically enhanced EEC data are used, which are registered in the ScanSAR mode and calibrated by the TMSP SAR processor (*Astrium* (2002)). The calibration in this step includes geo-referencing, multi-looking, projection of SAR data to a plane sand other radiometric calibrations. The radiometrically enhanced EEC data are represented in *digital number* (DN) values. These data are then further calibrated by the SAR processor in ZKI using the ASTER Global Digital Elevation Model Version 2 (GDEM V2) (LP DAAC, NASA) to adjust the errors of the SAR data from terrain variations. The result is represented in dB values. Because the most models for SAR signals in the literature are based on power/intensity or amplitude data, the dB images are then transformed to intensity images. This calibrated and transformed SAR intensity images are the input data of this thesis.

In the next, the signal calibration in ZKI and the data transformation from dB values to intensity values are explained.

2.2.1 Calculation of intensity images

According to the product information in (*Astrium*, n.d.*b*), the measurements of TerraSAR-X are registered in DN values, which are calculated from the real and imaginary parts I, Q of the received complex signal:

$$\mathrm{DN} = \sqrt{I^2 + Q^2}. \tag{2.1}$$

The DN values are calibrated with the calibration factor k_s:

$$\beta^0 = k_s |DN|^2, \tag{2.2}$$

and then transformed to dB values:

$$\beta^0_{\mathrm{dB}} = 10 \log_{10}(\beta^0). \tag{2.3}$$

The data are images in β^0_{dB} values. In order to retrieve the amplitude or intensity data, the transformations in equation (2.1), (2.2) and (2.3) should be used inversely. Two points should be noticed here. Firstly, the calibration factor k_s only plays a role of scaling, it does not change the properties of the SAR signals at all. Secondly, the term $I^2 + Q^2$ has the physical meaning of power, so $\beta^0 = k_s(I^2 + Q^2)$ has also the physical meaning of power and can be modeled by the SAR intensity models in the literature. For simplicity β^0 can be considered as a scaled/calibrated version of intensity value. In this thesis only following equation is used to transform the dB value to intensity:

$$\beta^0 = 10^{\frac{\beta^0_{\mathrm{dB}}}{10}}. \tag{2.4}$$

2.2.2 Correction of terrain effect

Before direct application of β^0 it is important to make sure that the measured β^0 only reflects backscattering property of the observed object. Otherwise this physical quantity can not represent the change of real objects. One problem of β^0 is that local variation of terrain can affect radar signal. This situation is illustrated in figure 2.2, where a SAR pulse is reflected on a plane. The normal line of the plane is in blue. The pulse with length Δ is demonstrated by two parallel dashed lines, the true reflecting area is in red. From the geometry the true backscattering area is equal to $\Delta / \sin\theta$. This means that the observation depends on the local incidence angle. If the angle is smaller, the measured intensity is larger, vice versa. In order to get a physical quantity that only depends on the object itself, the effect of terrain from the term $1/\sin\theta$ should be compensated , so a new physical quantity called *sigma naught* is defined as (*Astrium*, n.d.*b*):

$$\sigma^0 = \beta^0 \sin\theta. \tag{2.5}$$

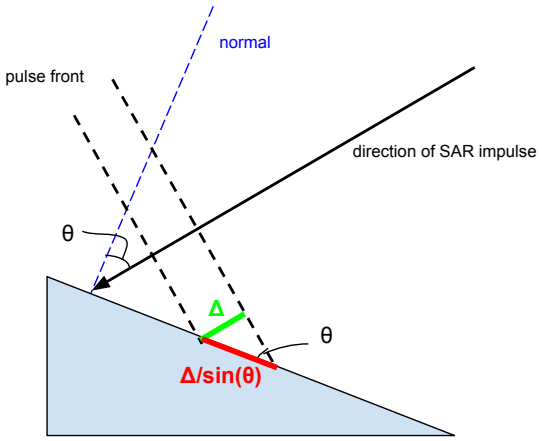


Figure 2.2: *Incidence angle and* β^0.

In this way σ^0 only reflects the backscattering ability of the object, which is independent of the incidence angle of the beam. It is also proportional to the true intensity of the SAR signal. σ^0 can also be interpreted as *scaled/calibrated intensity*. Compared with β^0, σ^0 is more suitable to represent the true backscattering ability of the object. In this thesis σ^0 is used as input data to detect changes from two SAR images.

In the literature a similar physical quantity σ also characterizes the backscattering property of an object. According to Oliver (1996), σ is called *radar cross section* (RCS). Their difference is that σ^0 is defined for point scatters, σ is for distributed scatters. For simplicity no discrimination is made between these two quantities and they are identified generally as the backscattering ability of the observed object. Because SAR signal are coupled with the speckle noise n, which is commonly considered as multiplicative noise. Strictly speaking the observed scaled power/intensity σ in equation (2.5) should be replaced by σn.

Chapter 3

Histogram thresholding

3.1 Introduction

Histogram thresholding is a classical but widely used technique. The idea is easy to understand. The histogram of an image can be considered as a mixture of pixels from two classes: background and foreground class. The pixel intensities of each class follow two different statistical *probability density functions* (pdf). In practice histogram is a good estimation of the theoretical pdf. In the case of two classes, the histogram can be divided into two parts by choosing a threshold. Pixels with intensity values smaller than the threshold are classified to the first class, the others belong to the second class.

There are many image thresholding techniques in the literature. Sezgin et al. (2004) provided a good review of nearly 40 thresholding methods in the literature in terms of segmentation of documents and nondestructive testing (NDT) images. They divided thresholding methods into six categories:

1. Histogram shape-based methods through the analysis of the peaks, valleys and curvature of the smoothed histogram.
2. Clustering-based methods through division of the samples into two parts or modeling as mixture of Gaussians.
3. Entropy-based methods through the usage of the entropy of the background and foreground regions or the cross-entropy between the original and binarized images.
4. Object attribute-based methods through the search of a similarity measure between the original and the binarized images.
5. Spatial methods through the higher-order probability distribution or correlation between pixels.
6. Local methods through the adaption of the threshold value on each pixel to the local image characteristics.

According to the experiments by Sezgin et al. (2004), the quality of the thresholding depends on different applications. But they concluded that the clustering-based method by Kittler and Illingworth (1986) provided almost the best thresholding result in their experiments. In the literature this method is also called *KI thresholding*. It was widely discussed and developed in the literature of SAR application. Because of its popularity and good evaluation the KI thresholding is chosen in this thesis.

However, KI thresholding can not be directly used for change detection using SAR data, because this method is originally limited to the Gaussian pdf. For change detection using SAR data, the Gaussian assumption does not apply any more. At the same time there are three classes to distinguish, so direct application of KI thresholding is difficult for SAR change detection. Some efforts have been made to extend the KI thresholding to deal with a three-class problem or replace the Gaussian model with other pdfs. In this chapter a three-class histogram thresholding method is presented based on (Bazi et al., 2005) and (Moser and Serpico, 2006). Before that a brief introduction of the KI thresholding is given.

3.2 KI thresholding

The quality of a histogram thresholding can be measured by a criterion or penalty function. In order to describe the criterion used by Kittler and Illingworth (1986), firstly, some notations have to be introduced to formulate the two-class thresholding problem.

Mathematical notations

The histogram of an image is denoted as $h(g)$, where g is the value of pixel intensity, h is the frequency of pixels with intensity value g. In the case of two classes, there are two pdfs, each of them can be represented by the *class conditional probability* $h(g|i)$, where i means the number of the class for $i = 1, 2$. These two classes are divided by a threshold denoted as T, where pixels with $g \leq T$ belong to the first class, pixels with $g > T$ belong to second class. After thresholding all pixels are divided into two classes, each class is characterized by its own parameters in the *class conditional probability* $h(g|i)$ and its *class a priori probability* P_i. In case of the Gaussian distribution, the parameters are mean value μ_i and variance σ_i, which can be estimated from the samples of each class. In this viewpoint the class conditional probability $h(g|i)$, class a priori probability P_i and its parameters μ_i, σ_i also depend on the threshold value T, they can be denoted as $h(g|i,T)$,$P_i(T)$ $\mu_i(T)$ and $\sigma_i(T)$.

Criterion function

Kittler and Illingworth (1986) used the following equation to evaluate the threshold value T:

$$e(g,T) = h(g|i,T) \cdot P_i(T)/h(g), i = 1,2, \tag{3.1}$$

they interpreted $e(g,T)$ as "probability of g being replaced in the image by a correct binary value. " In order to make $e(g,T)$ more simple for Gaussian distribution function, $e(g,T)$ is modified as:

$$\epsilon(g,T) = -2\ln[e(g,T)]. \tag{3.2}$$

The quality of the threshold value T for the whole image can be written by the following criterion function:

$$J(T) = \sum_g h(g) \cdot \epsilon(g,T). \tag{3.3}$$

The best threshold value τ is that which minimize the criterion function in equation (3.3):

$$\tau = \operatorname{argmin} J(T). \tag{3.4}$$

Calculation

If the threshold is chosen, other parameters can be estimated from the samples in each class. These parameters are calculated respectively as:

- The class a priori probability:

$$P_i(T) = \sum_{a}^{b} h(g). \tag{3.5}$$

- The mean value of each class:

$$\mu_i(T) = \frac{\sum_a^b h(g)g}{P_i(T)}. \tag{3.6}$$

- The variance value of each class:

$$\sigma_i^2(T) = \frac{\sum_a^b (g - \mu_i(T))^2 h(g)}{P_i(T)}. \tag{3.7}$$

where a, b are boundaries of the pixel intensities of each class:

$$\begin{array}{llll} \text{case} & i = 1: & a = 0, & b = T, \\ \text{case} & i = 2: & a = T + 1, & b = g_{max}. \end{array} \tag{3.8}$$

In addition, the samples from each class are assumed to be Gaussian distributed, so the class conditional probability function $h(g|i, T)$ can be written as:

$$h(g|i, T) = \frac{1}{\sqrt{2\pi}\sigma_i(T)} \exp\left(-\frac{1}{2}\left(\frac{g - \mu_i(T)}{\sigma_i(T)}\right)^2\right). \tag{3.9}$$

Based on equations (3.5), (3.6), (3.7) and (3.9), equation (3.3) can be rewritten as:

$$\begin{array}{rcl} J(T) & = & 1 + 2[P_1(T)\ln(\sigma_1(T)) + P_2(T)\ln(\sigma_2(T))] \\ & & -2[P_1(T)\ln(P_1(T)) + P_2(T)\ln(P_2(T))] \quad . \end{array} \tag{3.10}$$

3.3 Generalized KI thresholding

KI thresholding is an easy method to find the optimal threshold but limited to the two-class case with the Gaussian assumption. In change detection there are usually more than two classes in the images. At the same time the Gaussian model does not applies to SAR data directly. So KI needs two modifications to be applied in change detection. At first, it needs to deal with other statistical pdfs, such as the Gamma distribution function. Secondly, at least two thresholds should be used to distinguish three classes. Because their works are just an update of the traditional KI algorithm, they are also called *generalized KI thresholding* (GKI) (Moser and Serpico, 2006).

3.3.1 Modification of the KI thresholding

Although equation (3.10) provides a compact formula and is easy to use, its previous version (3.1), (3.2) and (3.3) are more general and do not assume any particular statistical pdf. Moser et al. (2006) and Bazi et al. (2005) tried to extend the KI thresholding by interpreting equation (3.1) from the Bayesian viewpoint. Some modifications of the formulas in the KI algorithm will help to see their interpretation.

As known, Bayesian formula is usually formulate as following:

$$p(i|x) = \frac{p(x|i)}{p(x)} p(i), \tag{3.11}$$

where i is the class number, x is the observation value, $p(i)$ is the *a priori probability*, $p(i|x)$ is *a posteriori probability* after the observation x, $p(x|i)$ is the *likelihood function* and $p(x)$ is the *evidence function*. The Bayesian equation (3.11) can be interpreted as a transformation between a priori probability $p(i)$ and a posteriori probability $p(i|x)$ given an observation x. From this viewpoint equation (3.1) can be rewritten as following if the threshold T is omitted here:

$$e(g) = \frac{h(g|i)}{h(g)} P(i), \quad i = 1, 2, \tag{3.12}$$

where $h(g|i)$ is the class conditional probability, it plays a role of the class likelihood function $p(x|i)$, $h(g)$ is also similar to the evidence function $p(x)$. In this way $e(g)$ in equation (3.12) can be considered as the a posteriori probability. So the criterion equation (3.3) can be rewritten in a more general way:

$$\begin{aligned} J(T) &= \sum_g h(g) \cdot \epsilon(g, T) \\ &= -2 \sum_g h(g) \cdot \ln[e(g, T)] \\ &= -2 \sum_g h(g) \cdot \ln[p(i|g, T)]. \end{aligned} \tag{3.13}$$

If the minus symbol in equation (3.13) is omitted , the minimum condition in the criterion (3.4) is equivalent to the maximum, and it can be interpreted as *maximum a posteriori probability* (MAP) condition according to equation (3.13).

Practically the constant 2 in (3.13) can be omitted. As shown in (3.11), the Bayesian equation transforms the a priori to a posteriori probability through the likelihood function, the denominator $p(x)$ plays a role of balancing the left and right hand sides of the equation, it can be omitted without loss of information, so the criterion function (3.13) can be simplified as (Moser and Serpico, 2006):

$$J(T) = -\sum_g h(g) \cdot [\ln(p(g|i, T)) + \ln(P(i|T))], \quad \text{for} \quad i = 1, 2. \tag{3.14}$$

In this generalized criterion equation, $h(g)$ and P_i are calculated from the histogram, $p(g|i, T)$ is the statistical distribution function. Reforming equation (3.3) into (3.14) not only gives a more general viewing angle of KI thresholding but also provides an interface between the traditional KI algorithm and other statistical pdfs.

3.3.2 Substitution of the Gaussian model

The equation (3.14) make the KI algorithm not be limited to the Gaussian pdf. It is now adaptable to SAR signal. In the next other statistical distribution functions for SAR signal are discussed.

Because of a limited resolution the received SAR signal comes from a series of distributed scatters instead of just an ideal point. Every point scatter within a resolution contributes to the received signal individually, the result is the vectorial summation of the components received from every point scatter within the resolution area. According to Oliver (1996) it is formulated as:

$$A \cdot \exp(i\phi) = \sum_{k=1}^{N} A_k \cdot \exp(i\phi_k), \tag{3.15}$$

where A_k is the amplitude value of the k-th received signal, ϕ is the corresponding phase, N is the number of the individual scatters within the resolution area. The amplitude of the resultant signal is A and the phase is ϕ.

If no scatter is dominant within the resolution area, the resultant phase ϕ can be considered as a sample from the uniform distribution between $[-\pi, \pi]$, and the resultant amplitude can be considered as a result of random walk on the complex coordinate system (Oliver, 1996). In this way both of the real part Z_1 and the complex part Z_2 of SAR signal follows the Gaussian distribution. Their joint distribution is given by Oliver (1996) as following:

$$p_{Z_1,Z_2}(z_1, z_2) = \frac{1}{\pi\sigma} \exp\left(-\frac{z_1^2 + z_2^2}{\sigma}\right), \tag{3.16}$$

where the parameter σ is called *Radar Cross Section* (RCS).

The random variables (Z_1, Z_2) of this joint distribution can be transformed into the polar coordinate (A, ϕ). Under the condition that ϕ follows the uniform distribution between $[-\pi, \pi]$ the amplitude A follows finally the Rayleigh distribution:

$$p_A(a) = \frac{2a}{\sigma} \exp\left(-\frac{a^2}{\sigma}\right). \tag{3.17}$$

After transformation $G = A^2$, the intensity G follows then the exponential distribution:

$$p_G(g) = \frac{1}{\sigma} \exp\left(-\frac{g}{\sigma}\right). \tag{3.18}$$

On the other hand, there is the Doppler effect as the platform of the SAR system moves in the radial direction with respect to the observing scatter. This phenomena results in the increased frequency band width of the received signal. Because the spatial resolution is inversely proportional to the band width, equivalently the spatial resolution becomes smaller. However, this frequency band width is not always exploited once a time. A consequence of the coherent SAR signal is the speckle effect, which degrades the practical applicability of SAR and makes the analysis of SAR images more difficult. In order to reduce the speckle in the SAR image, the band width is divided into L parts, each of them can provide an observation, and the final product is their mean value. In this way a balance is achieved between the spatial resolution and suppression of speckle effect. L is called the *number of looks* (Oliver, 1996).

Since L observations $G_k, k = 1, 2...L$ are used to produce the final SAR image, each of them follows the exponential distribution described by (3.18), the resultant mean signal

$$G = \frac{\sum_{k=1}^{N} G_k}{L}$$

is also a random variable and follows a distribution. Neglecting the constant L the mean signal can be considered as a sum of L random variables from the same exponential distribution. In Mathematics, if random variables X, Y have pdfs $f_X(x), f_Y(y)$, then the pdf of their sum $Z = X + Y$ is the convolution between $f_X(x)$ and $f_Y(y)$ (Springer, 1979). Accordingly the final SAR signal after multi-looking is a random variable, whose distribution function equals to the convolution of L exponential distribution functions. The result can be analytically described as a Gamma distribution function (Oliver, 1996):

$$p_G(g; L, \sigma) = \frac{1}{\Gamma(L)} (\frac{L}{\sigma})^L g^{L-1} \exp\left(-\frac{Lg}{\sigma}\right). \tag{3.19}$$

In the literature there are also other statistical models for SAR signal. Those distribution functions can be considered as extensions of the Gamma pdf. For example, if variable transformation $A = \sqrt{G}$ is used, equation (3.19) can be rewritten as a function of amplitude A. This pdf is then called *Nakagami distribution*, which is suitable to model SAR amplitude signals. If an a priori statistical model is used to describe the fluctuation of the RCS σ in equation (3.19), the resultant statistical model is called *K distribution*, which is suitable to model SAR signal from an inhomogeneous region. If the real and complex parts of the SAR signal are assumed to follow the generalized Gaussian model instead of the Gaussian, the resultant model is called *generalized Gaussian Rayleigh distribution*, this modification makes it suitable to model a long tailed distribution.

According to (Oliver, 1996) the Gamma model is a good distribution function to model the SAR signal received from natural objects compared to its variations. In this thesis the focus is detection of natural disasters such as flood water, so the Gamma pdf is still preferred. Here two additional empirical models are also used: *log-normal* and *Weibull* pdf. They do not have so clear physical interpretation as the Gamma pdf, but mathematically they are good approximations of the Gamma pdf and easier to calculate. Their mathematical functions are given:

- Log-normal distribution function

$$p_G(g; m, V) = \frac{1}{g\sqrt{2\pi V}} \exp\left[-\frac{(\ln g - m)^2}{2V}\right]. \tag{3.20}$$

- Weibull distribution function

$$p_G(g; b, \eta) = \frac{\eta g^{\eta-1}}{b^\eta} \exp\left[-\left(\frac{g}{b}\right)^\eta\right]. \tag{3.21}$$

3.3.3 Three-class thresholding

Two thresholds

Bazi et al. (2005) extended the traditional KI algorithm to solve the three-class problem. They added another threshold to the criterion function (3.14) as follows:

$$J(T_1, T_2) = -\sum_g h(g) \cdot [\ln(p(g|i, T_1, T_2)) + \ln(P(i|T_1, T_2))], i = 1, 2. \tag{3.22}$$

The optimal threshold values are (T_1, T_2) which minimize $J(T_1, T_2)$. In practice, every pair of (T_1, T_2) is tried and the corresponding $J(T_1, T_2)$ is calculated, then a matrix of $J(T_1, T_2)$ can be obtained for each pair of (T_1, T_2). In this way the minimum of $J(T_1, T_2)$ is easy to find. However, the potential error is that the position of the minimum value could locate at the boundary, or there could be more than one position with global minimum. In order to avoid these critical solutions Bazi et al. (2005) suggested calculating the determinant of the Hessian matrix for $J(T_1, T_2)$. If the determinant is negative, the extreme value is only at the saddle point, it can not be the global minimum. If the determinant is zero, there must be many extreme values. This means that only one threshold is valid. In this way, the traditional KI algorithm is applicable to the three-classes problem. In the section 3.5 another method using the profiles of the cost function is used to replace the determinant of Hessian matrix.

Ratio image

In the case of change detection the input data are two SAR images. The objective is to find the area where change occurs. In the framework of algebraic thresholding method there are two popular ways to model this changed signals. One method uses the difference between intensity images, the resultant test image is called *difference image*. The other method uses their ratio, the test image is called *ratio image*.

According to equation (3.19) the variance values of pixel intensity are:

$$\text{var}(G) = \frac{\sigma^2}{L}, \tag{3.23}$$

where σ is the RCS, L is the equivalent number of looks. If intensity value at a given pixel in two co-registered images are denoted as G_1, G_2, the variance of the difference $d = G_1 - G_2$ is then:

$$\sigma_d^2 = \frac{\sigma_1^2 + \sigma_2^2}{L}, \tag{3.24}$$

where σ_1, σ_2 are RCS of G_1, G_2. This equation means that the variance σ_d depends on the RCS of the pixel. This implies that simple thresholding will yield more errors for a given change in a bright area than a darker area (Oliver, 1996). So instead of image differencing, the ratio is a better test variable in change detection. The following problem is to find the pdf of ratio signal.

Ratio distribution

As discussed before, the SAR signal can be modeled by the Gamma or Nakagami distribution. There are two random variables x, y following distributions $f_X(x), f_Y(y)$, their joint distribution is denoted as $f_{X,Y}(x, y)$, generally the ratio $r = x/y$ follows the following distribution (Springer, 1979):

$$p_R(r) = \int_{-\infty}^{+\infty} |y| \, f_{X,Y}(ry, y) dy. \tag{3.25}$$

The solution of this integral for two Gamma distributions, i.e intensity values G_1, G_2 of two pixels in the SAR image is given in (Oliver, 1996) as follows:

$$p_R(r) = \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L r^{L-1}}{(q+r)^{2L}}, \tag{3.26}$$

where $q = \sigma_1/\sigma_2$ is the true ratio of RCS, $r = G_1/G_2$ is the observed ratio, L is the equivalent number of looks. This formula was proved analytically by Touzi et al. (1988). In next chapter a few properties of the Gamma distribution will be used to avoid the analytical integration. Both of them get the same result, the derivation of latter one is easier.

For the Weibull distribution Moser and Serpico (2006) provided a mathematical derivation. The following simplified assumptions were used:

- Two independent random variables, so $f_{XY}(ry, y) = f_X(ry) f_Y(y)$,
- The parameters η_1, η_2 for each random variable are same.

The proof is listed in the appendix. The pdf of ratio is described as:

$$p_R(r) = \eta \lambda^\eta \frac{r^{\eta-1}}{(\lambda^\eta + r^\eta)^2}, \tag{3.27}$$

where $\lambda = b_1/b_2$.

For log-normal distribution no new formula has to be derived. After transformation of the variable $y = \ln x$, the log-normal random variable x becomes the normal variable y (see the definition of log-normal distribution in equation (3.20)). For two log-normal variables x_1, x_2 along with $y_1 = \ln x_1, y_2 = \ln x_2$, it is easy to get $d = y_1 - y_2 = \ln(x_1/x_2)$. This means that the quotient of two log-normal random variables x_1/x_2 corresponds to the difference of two normal random variables $d = y_1 - y_2$. As known linear combinations of normal variables are still normal distributed, so $d = y_1 - y_2$ is a normal variable, which also implies that x_1/x_2 is still log-normal. In this way log-normal distribution function can be used to model ratio images directly.

3.4 Parameter estimation by MoLC

Parameter estimation in the statistic models is an important task. Krylov et al. (2013) provided a good review and introduced a new method. The traditional methods include Maximum of likelihood (ML) and Method of Moments (MoM). The first one forms a criterion function as the product of the likelihoods of the samples. The optimal parameters are those which maximize this criterion function. It is usually solved by partial derivation of the criterion function with

respect to the parameters. One problem of this method is that the likelihood function could be very difficult to derivate, so that the procedure would be analytically very complicated. The second method uses the Laplace transform to formulate the moments as the function of the parameters. One problem of this method is that some results from the Laplace transform are also difficult to tackle analytically or numerically. This is the situation for the statistic models of the SAR signal, which is only defined on the interval $(0, +\infty)$. In the literature a method called *method of logarithmic cumulants* (MoLC) were suggested by Nicolas (2002) and Moser et al. (2006) to deal with these problems. According to their studies the *Mellin transform* is better than the Laplace transform to estimate the parameters for pdfs defined on $(0, +\infty)$. This method provides a simpler formula and the variance of estimation is also smaller than that from MoM and ML (Nicolas, 2002).

3.4.1 Introduction of MoLC

The first characteristic function and moments of the second kind

The basis of MoLC is the Mellin transform, which is defined for the pdf $p_X(x)$ on $x \in (0, +\infty]$ as follows:

$$\mathcal{M}[p_X(x)](s) = \int_0^{+\infty} x^{s-1} p_X(x) dx. \tag{3.28}$$

In the situation of MoM, parameters are contained in the function of moments, which can be obtained from the characteristic function through the Laplace transform. Similarly, a so called *the first characteristic function of the second kind* was defined by Nicolas (2002) based on the Mellin transform:

$$\phi_X(s) = \mathcal{M}[p_X(x)](s). \tag{3.29}$$

As known the first characteristic function can be used to generate moments. Similarly $\phi_X(s)$ can also be handled in the same way. The Mellin transform in equation (3.28) can be rewritten as follows:

$$\phi_X(s) = \int_0^{+\infty} \exp\left[(s-1)\ln x\right] p_X(x) dx. \tag{3.30}$$

Using the Taylor series expansion $\exp(x) = 1 + x + \frac{x^2}{2!} + \cdots$ in equation (3.30) provides

$$\begin{aligned} \phi_X(s) &= \int_0^{+\infty} \left[1 + (s-1)\ln x + \frac{\left((s-1)\ln x\right)^2}{2!} + \cdots\right] p_X(x) dx \\ &= 1 + (s-1)\int_0^{+\infty} \ln x \cdot p_X(x) dx + \frac{(s-1)^2}{2!}\int_0^{+\infty} (\ln x)^2 p_X(x) dx + \cdots . \end{aligned} \tag{3.31}$$

The terms in the integral are similar to the moments $m_{X(n)} = \int x^n p_X(x) dx$. The difference is that equation (3.31) is in terms of $(\ln x)^n$ instead of x^n. Similar to the definition of moments Nicolas (2002) also defined *the second kind moments* as follows:

$$\tilde{m}_{X(n)} = \int_0^{+\infty} (\ln x)^n p_X(x) dx. \tag{3.32}$$

So equation (3.31) can be rewritten as:

$$\phi_X(s) = 1 + \sum_{n=1}^{+\infty} \frac{(s-1)^n}{n!} \tilde{m}_{X(n)}. \tag{3.33}$$

Equation (3.33) also provides another way to get $\tilde{m}_{X(n)}$:

$$\tilde{m}_{X(n)} = \left. \frac{d^n \phi_X(s)}{ds^n} \right|_{s=1}. \tag{3.34}$$

The n-th derivation of $\phi_X(s)$ will delete the terms in equation (3.33) with the exponential smaller than n. After setting $s = 1$ the terms with the exponential larger than n will also be deleted, the left one is only the term with $\tilde{m}_{X(n)}$.

The second characteristic function and moments of the second kind

In statistics $\ln(\phi_X(s))$ instead of $\phi_X(s)$ is sometimes used, it provides more simpler theoretical treatments than $\phi_X(s)$ in some cases (Weisstein, Eric W., n.d.). $\ln(\phi_X(s))$ is called *the second characteristic function of the second kind* (Nicolas, 2002) and denoted as:

$$\psi_X(s) = \ln(\phi_X(s)). \tag{3.35}$$

Similar to equation (3.31), using Taylor expansion $\ln x = -\sum_{k=1}^{+\infty} \frac{(1-x)^k}{k}$ provides (Weisstein, Eric W., n.d.):

$$\begin{aligned} \psi_X(s) &= \ln(\phi_X(s)) = -\sum_{k=1}^{+\infty} \frac{1}{k} (1 - \phi_X(s))^k \\ &= -\sum_{k=1}^{+\infty} \frac{1}{k} \left(-\sum_{n=1}^{+\infty} \frac{(s-1)^n}{n!} \tilde{m}_{X(n)} \right)^k \\ &= \tilde{m}_{X(1)}(s-1) + (\tilde{m}_{X(2)} - \tilde{m}_{X(1)}^2) \frac{(s-1)^2}{2!} + \\ &\quad (\tilde{m}_{X(3)} - 3\tilde{m}_{X(2)}\tilde{m}_{X(1)} + 2\tilde{m}_{X(1)}^3) \frac{(s-1)^3}{3!} + \cdots . \end{aligned} \tag{3.36}$$

It is straightforward to rewrite the equation above more compactly:

$$\psi_X(s) = \sum_{n=1}^{+\infty} \frac{(s-1)^n}{n!} \tilde{k}_{X(n)}, \tag{3.37}$$

where $\tilde{k}_{X(n)}$ is defined as:

$$\tilde{k}_{X(1)} = \tilde{m}_{X(1)}, \tag{3.38}$$

$$\tilde{k}_{X(2)} = \tilde{m}_{X(2)} - \tilde{m}_{X(1)}^2, \tag{3.39}$$

$$\tilde{k}_{X(3)} = \tilde{m}_{X(3)} - 3\tilde{m}_{X(2)}\tilde{m}_{X(1)} + 2\tilde{m}_{X(1)}^3. \tag{3.40}$$

$$\cdots$$

Similar to equation (3.34), $\tilde{k}_{X(n)}$ can be defined by derivation of $\psi_X(s)$:

$$\tilde{k}_{X(n)} = \left.\frac{d^n \psi_X(s)}{ds^n}\right|_{s=1}. \tag{3.41}$$

It worth noting that the moments and log-moments are defined by the first and second characteristic functions through equations (3.34) and (3.41). At the same time the characteristic functions are also determined by moments and log-moments through series in equations (3.33) and (3.37). In addition, the moments and log-moments of the second kind are related to each other through equations (3.38), (3.39) and (3.40). Through series expansion each component can be handled respectively and then combined linearly. This property will be used to estimate parameters in the case of product or ratio of two random variables.

Mellin transform and its properties

One advantage of integral transforms is that some operations become easier in the transformed space. As known after the Fourier transform the convolution operation in time space becomes product in frequency space. A similar property is expected for the Mellin transform, but the convolution for the Mellin transform should be redefined. Mathematically the Mellin convolution between two functions f, g on $(0, +\infty)$ is denoted as $f \hat{*} g$ and defined as follows:

$$h(u) = [f \hat{*} g](u) := \int_0^{+\infty} f(y) g\left(\frac{u}{y}\right) \frac{dy}{y}. \tag{3.42}$$

With this definition some fundamental properties of the Mellin transform can be proved. The details of the proofs are demonstrated in the appendix to maintain the fluency of the thesis. Here only a few properties are listed and used to derive the equations of parameter estimation (Nicolas, 2002).

1. **Convolution property**
   For two functions f, g defined on $(0, +\infty)$, the product of their Mellin transforms is equal to the Mellin transform of their Mellin convolution, i.e

$$\mathcal{M}[f \hat{*} g] = \mathcal{M}[f] \cdot \mathcal{M}[g]. \tag{3.43}$$

2. **Product property**
   For random variables X_i with the corresponding pdfs $p_{X_i}(x_i)$, $i = 1, 2$, the pdf of their product $X_3 = X_1 \cdot X_2$ is denoted as $p_{X_3}(x_3)$. The product relation between the random variables is also preserved in the Mellin transforms of their pdfs, i.e

$$X_3 = X_1 \cdot X_2 \Rightarrow \mathcal{M}[p_{X_3}] = \mathcal{M}[p_{X_1}] \cdot \mathcal{M}[p_{X_2}]. \tag{3.44}$$

3. **Inversion property**
   For a random variable X with pdf $p_X(x)$, the pdf of random variable $Y = 1/X$ is denoted as $p_Y(y)$, whose Mellin transform can be calculated by the following relation:

$$\mathcal{M}[p_Y(y)](s) = \mathcal{M}[p_X(x)](2 - s). \tag{3.45}$$

The convolution property is the basis for others. From the product property the corresponding characteristic functions is obtained as follows:

$$\phi_{X_3}(s) = \phi_{X_1}(s) \cdot \phi_{X_2}(s), \tag{3.46}$$
$$\psi_{X_3}(s) = \psi_{X_1}(s) + \psi_{X_2}(s). \tag{3.47}$$

Inserting equation (3.37) in equation (3.47) gets the relation between log-cumulants:

$$\tilde{k}_{X_3(n)} = \tilde{k}_{X_1(n)} + \tilde{k}_{X_2(n)}, \quad n = 1, 2, \cdots. \tag{3.48}$$

This equation enables calculation of the log-cumulants for the product of two random variables very easily. The product and inversion properties can be combined together if the ratio of two random variables is rewritten as follows:

$$R = \frac{X_1}{X_2} = X_1 \cdot \left(\frac{1}{X_2}\right). \tag{3.49}$$

If the pdfs of X_1 and X_2 are $p_{X_1}(x_1)$ and $p_{X_2}(x_2)$, the pdf of R is $p_R(r)$, according to the product and inversion properties the Mellin transform of $p_R(r)$ is:

$$\mathcal{M}[p_R(r)](s) = \mathcal{M}[p_{X_1}(x_1)](s) \cdot \mathcal{M}[p_{X_2}(x_2)](2-s), \tag{3.50}$$

similarly another version represented by the characteristic functions of the second kind is given:

$$\phi_R(s) = \phi_{X_1}(s) \cdot \phi_{X_2}(2-s), \tag{3.51}$$
$$\psi_R(s) = \psi_{X_1}(s) + \psi_{X_2}(2-s). \tag{3.52}$$

Using series expansion to rewrite equation (3.52) also provides a linear combinations of the log-cumulants, which are the coefficients in equation (3.37). Each component can be abbreviated as $a_n(s-1)^n$.

In equation (3.52) variable s in $\psi_X(s)$ is replaced by $2-s$, so $a_n(s-1)^n$ will be replaced by $a_n[(2-s)-1]^n = a_n(1-s)^n = (-1)^n a_k(s-1)^n$. This implies that the log-cumulants of $\psi_{X_2}(2-s)$ are $(-1)^n \tilde{k}_{X_2(n)}$, so the following equation is obtained from equation (3.52):

$$\tilde{k}_{R(n)} = \tilde{k}_{X_1(n)} + (-1)^n \tilde{k}_{X_2(n)}, \quad n = 1, 2, \cdots. \tag{3.53}$$

This equation enable calculation of the log-cumulants for the ratio of two random variables easily. At the same it serves as the equation of parameter estimation. The left hand side can be calculated from the data (ratio image) according to the equations (3.32), (3.38), (3.39) and (3.40). The right hand side can be analytically calculated from the definition of the log-cumulants in equation (3.41) along with equations (3.29) and (3.35), which contain the parameters. The remaining problem is to derive $\tilde{k}_{X(n)}$ analytically for different pdfs of the SAR signal.

3.4.2 MoLC equation for ratio pdfs

In this thesis only three pdfs for SAR signals are used: Gamma, Weibull and log-normal distribution functions. There is a table of the Mellin transform for many commonly used pdfs (Krylov et al., 2013). Their results are directly used here to avoid tedious analytical derivation.

Gamma distribution

The second characteristic function with respect to the Gamma distribution is:

$$\phi_X(s) = \sigma^{s-1}\frac{\Gamma(L+s-1)}{L^{s-1}\Gamma(L)}, \tag{3.54}$$

where $\Gamma(x)$ is the Gamma function. Using equations (3.35) and (3.41) can get the log-cumulants $\tilde{k}_{X(n)}$:

$$\begin{aligned}
\tilde{k}_{X(1)} &= \ln\sigma + \Psi(L) - \ln L, && (3.55)\\
\tilde{k}_{X(2)} &= \Psi(1, L), && (3.56)\\
\tilde{k}_{X(3)} &= \Psi(2, L), && (3.57)
\end{aligned}$$

where $\Psi(n, x)$ is the polygamma function of order n (Weisstein, Eric W., n.d.) with the following definition:

$$\Psi(n, x) = \frac{d^{n+1}}{dx^{n+1}}\ln\Gamma(x). \tag{3.58}$$

There are synonyms for the polygamma functions in special cases of n:

- $\Psi(x) := \Psi(0, x)$ is called *digamma* function,
- $\Psi(1, x)$ is called *trigamma* function,
- $\Psi(2, x)$ is called *tetragamma* function.

For simplicity the parameter L for X_1, X_2 are assumed to be same. Equation (3.53) provides the log-cumulants for the ratio R of two Gamma random variables X_1, X_2:

$$\begin{aligned}
\tilde{k}_{R(1)} &= \tilde{k}_{X_1(1)} - \tilde{k}_{X_2(1)} = \ln\sigma_1 - \ln\sigma_2 = \ln q, && (3.59)\\
\tilde{k}_{R(2)} &= \tilde{k}_{X_1(2)} + \tilde{k}_{X_2(2)} = 2\Psi(1, L), && (3.60)\\
\tilde{k}_{R(3)} &= \tilde{k}_{X_1(3)} - \tilde{k}_{X_2(3)} = 0. && (3.61)
\end{aligned}$$

The third equation provides no information about the parameters. Only the first and second equations are used to estimate the parameter q, L in equation (3.26).

Weibull distribution

Similarly the log-cumulants for Weillbull distributed can be found in (Moser and Serpico, 2006):

$$\begin{aligned}
\tilde{k}_{X(1)} &= \ln b_i + \frac{\Psi(1)}{\eta}, && (3.62)\\
\tilde{k}_{X(2)} &= \frac{\Psi(1,1)}{\eta^2}. && (3.63)
\end{aligned}$$

Under the assumption that the parameter η in equation (3.27) is same for X_1, X_2, log-cumulants of their ratio are then:

$$\begin{aligned}
\tilde{k}_{R(1)} &= \tilde{k}_{X_1(1)} - \tilde{k}_{X_2(1)} = \ln b_1 - \ln b_2 = \ln\lambda, && (3.64)\\
\tilde{k}_{R(2)} &= \tilde{k}_{X_1(2)} + \tilde{k}_{X_2(2)} = 2\frac{\Psi(1,1)}{\eta^2}. && (3.65)
\end{aligned}$$

Log-normal distribution

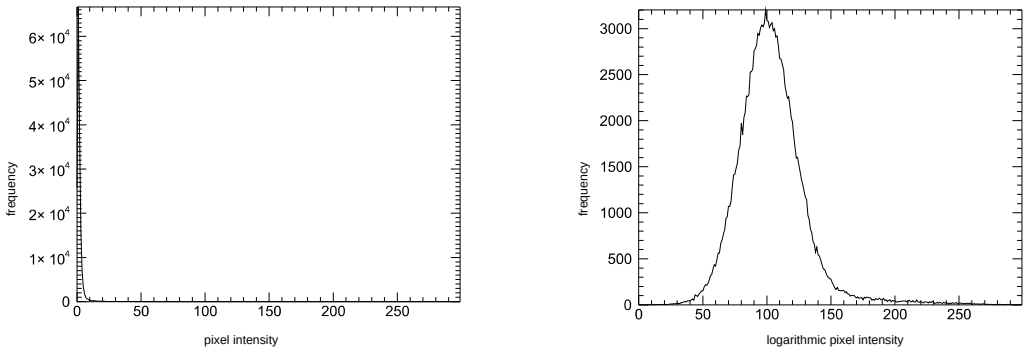
After the variable transformation $Y = \ln X$, the log-normal distributed random variable X is transformed to a Gaussian distributed random variable Y. The ratio of two log-normal variables is then transformed to the difference of two Gaussian ones: $d = Y_1 - Y_2 = \ln(X_1/X_2)$. Since d is normal, so X_1/X_2 is log-normal, this implies that the log-cumulants of the ratio in this case are:

$$\tilde{k}_{R(1)} = m, \tag{3.66}$$

$$\tilde{k}_{R(2)} = V. \tag{3.67}$$

3.4.3 Adaption to log-ratio

Previously three distribution functions are listed to model ratio signal, the MoLC method is used to estimate parameters. Using these equations the likelihood functions in equation (3.22) and the penalty (or called cost) can be calculated. The optimal pair of thresholds is that which minimize the penalty, so principally the three-class problem is solved. However, the calculation



(a) Histogram of ratio image.

(b) Histogram of log-ratio image.

Figure 3.1: *Comparison between the histograms of ratio and log-ratio data.*

of penalty in equation (3.22) also depends on the histogram. In practice the ratio can range from 0 to $+\infty$, this situation results in a very long-tailed histogram, which is very hard to deal with. This situation is illustrated in figure 3.1(a). The tail of the histogram can be cut, but in this way some information will get lost. As discussed in the literature the log-ratio is suggested to avoid this problem. It will compress the original interval of x-axis to a propitiate range. The histogram after the transformation is illustrated in figure 3.1(b). The histogram of log-ratio image is equally distributed compared with the one of ratio image.

As discussed in the previous sections, the logarithm function also appears in the log-cumulants, so he only problem is that the pdfs should also be adapted to the logarithmically transformed data.

Based on log-normal ratio

From equation (3.20) the log-normal data follows the Gaussian distribution after transformation of variable $Z = \ln R$:

$$p_Z(z; m, V) = \frac{1}{\sqrt{2\pi V}} \exp\left[-\frac{(z-m)^2}{2V}\right]. \tag{3.68}$$

Based on the Gamma ratio

For ratio of two Gamma random variables, equation (3.26) can be rewritten as:

$$\begin{aligned} p_R(r)dr &= \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L r^{L-1}}{(q+r)^{2L}} dr \\ &= \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L r^L}{(q+r)^{2L}} d\ln r \\ &= \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L \exp(zL)}{(q+\exp(z))^{2L}} dz, \end{aligned} \tag{3.69}$$

where the last step uses the transformation $Z = \ln R$. So the pdf for the log-ratio image based on the Gamma ratio distribution is then

$$p_Z(z; q, L) = \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L \exp(zL)}{(q+\exp(z))^{2L}}. \tag{3.70}$$

Based on the Weibull ratio

For Weibull ratio, equation (3.27) can be rewritten as follows:

$$\begin{aligned} p_R(r)dr &= \eta\lambda^\eta \frac{r^\eta}{(\lambda^\eta + r^\eta)^2} d\ln r \\ &= \eta\lambda^\eta \frac{\exp(z\eta)}{(\lambda^\eta + \exp(z\eta))^2} dz, \end{aligned} \tag{3.71}$$

where transformation of variable $Z = \ln R$ is used again. This equation implies that the pdf for the log-ratio of two Weibull distributed random variables is:

$$p_Z(z; \lambda, \eta) = \eta\lambda^\eta \frac{\exp(z\eta)}{(\lambda^\eta + \exp(z\eta))^2}. \tag{3.72}$$

It worth noting that the log-cumulants are transformed to the moments through this transformation, as shown below:

$$\tilde{k}_{R(1)} = \tilde{m}_{R(1)} = \int_0^{+\infty} \ln r\, p_R(r) dr = \int_{-\infty}^{+\infty} z\, p_Z(z) dz = m_{Z(1)}, \tag{3.73}$$

$$\begin{aligned} \tilde{k}_{R(2)} &= \tilde{m}_{R(2)} - \tilde{m}_{R(1)}^2 = \int_0^{+\infty} (\ln r)^2 p_R(r) dr - \tilde{m}_{R(1)}^2 \\ &= \int_0^{+\infty} (\ln r - \tilde{m}_{R(1)})^2 p_R(r) dr \\ &= \int_{-\infty}^{+\infty} (z - m_{Z(1)})^2 p_Z(z) dz = m_{Z(2)}, \end{aligned} \tag{3.74}$$

where $m_{Z(1)}, m_{Z(2)}$ are first and second **centered** moments of the transformed data Z, also called *mean* and *variance*. They can be easily calculated from the transformed data. Log-cumulants $\tilde{k}_{R(1)}, \tilde{k}_{R(2)}$ in the section 3.4.2 can be replaced by $m_{Z(1)}, m_{Z(2)}$. In this way the pdfs and parameter estimation equations for the transformed log-ratio data Z are updated. A summary of equations used to process thresholding for three-class problem is listed in the table 3.1.

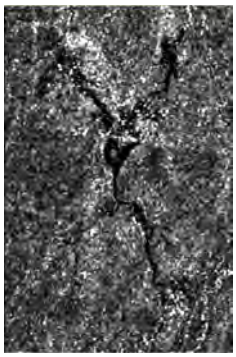
Log-Ratio of	**PDF**	**Parameters Eq.**
Gamma	$p_Z(z;q,L) = \frac{\Gamma(2L)}{\Gamma^2(L)} \frac{q^L \exp(zL)}{(q+\exp(z))^{2L}}$	$m_{Z(1)} = \ln q,\ m_{Z(2)} = 2\Psi(1,L)$
Weibull	$p_Z(z;\lambda,\eta) = \eta\lambda^\eta \frac{\exp(z\eta)}{(\lambda^\eta+\exp(z\eta))^2}$	$m_{Z(1)} = \ln\lambda,\ m_{Z(2)} = 2\frac{\Psi(1,1)}{\eta^2}$
Lognormal	$p_Z(z;m,V) = \frac{1}{\sqrt{2\pi V}} \exp\left[-\frac{(z-m)^2}{2V}\right]$	$m_{Z(1)} = m,\ m_{Z(2)} = V$

Table 3.1: *Distribution functions and parameter equations for log-ratio data based on different models.*

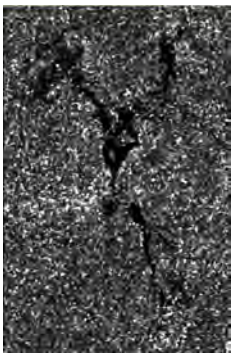
3.5 Application to test data

In this section the thresholding method is used to classify SAR ratio/log-ratio images. The first test data are two horizontal polarized ScanSAR images recorded on Queensland, Australia on 21 January, 2011 and on 27 February, 2012. Here a subset before and after an flood event is chosen on purpose.

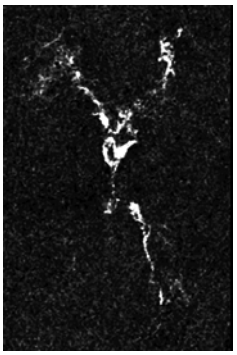
(a) Test area recorded on 21 January, 2011, ©DLR 2013.



(b) Test area recorded on 27 February, 2012, ©DLR 2013.



(c) The first intensity ratio image, ©DLR 2013.



(d) The second intensity ratio image, ©DLR 2013.

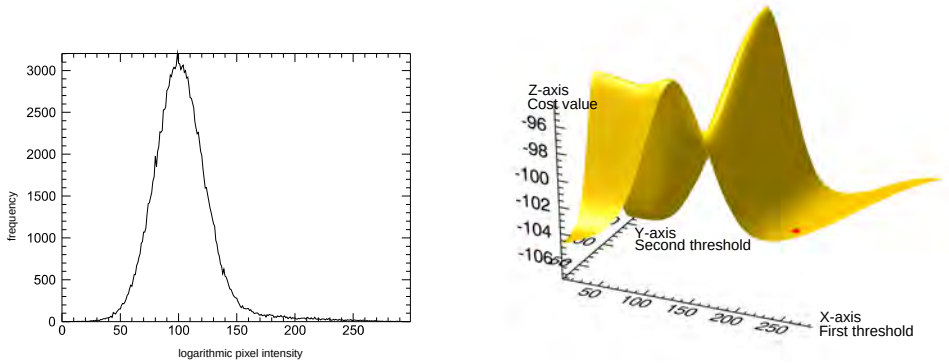
Figure 3.2: *The first test area on Queensland, Australia. Figure 3.2(a) is recorded before the flood event, figure 3.2(b) is after the flood event. In the figure 3.2(c) the dark pixels mean negative-changed areas, which indicate the flooded areas. The gray and bright pixels mean no-changed areas. In the figure 3.2(d) the bright pixels mean positive-changed areas, which indicate the flooded areas. The gray and dark pixels mean no-changed areas.*

Figure 3.2(a) represents the test area before the flood event, figure 3.2(b) represents the test area after the flood event. The gray and bright pixels in both figures represent land in general. The dark pixels in figure 3.2(b) represents the flood water. From figures 3.2(a) and 3.2(b) a clear

difference can be found between two test images. The purpose is to extract the dark pixels in the second SAR image which indicate flood water. In order to highlight the changed signal two ratio images are displayed. One is defined as the quotient between the second and first image, this ratio image is illustrated in figure 3.2(c). For the second ratio image shown in figure 3.2(d) the order of image sequence is changed.

Because the contrast of image is dominated by bright ones, dark pixels can hardly be noticed, that is why here two ratio images are displayed. The bright pixels in the second ratio image are exactly the dark ones in the first image. These are equivalent and complement to each other on the purpose of representation.

The histogram of log-ratio is illustrated in figure 3.3(a). A 3D-plot of the cost/penalty function based on the Gamma pdf is illustrated in figure 3.3(b), where the yellow color means the surface of the cost function. The optimal solution is marked by a red spot. The x-axis indicates the first threshold T_1, the y-axis is for the second threshold T_2. The z-axis represents the cost/penalty value. As mentioned by Bazi et al. (2005), there is ambiguity between three-class and two-class problem. They used the determinant of the Hessian matrix to distinguish them. Their choice is reasonable. The global minimum is always characterized by the concave curvature with positive determinant of the Hessian matrix. For this test data, the cost function in the x-direction is concave near the red spot , in y-direction the cost function is almost a constant, so the determinant of the Hessian matrix is nearly zero. This means that there is only one optimal threshold and there are two classes in this image. The ratio images in figure 3.2(c) and 3.2(d) prove this decision.



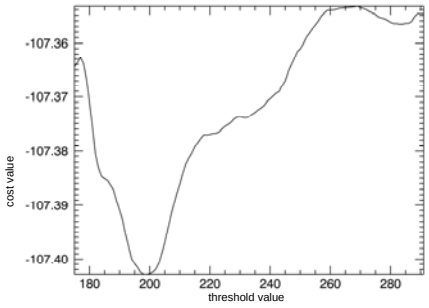
(a) The histogram of log-ratio image.

(b) 3D plot of the cost function.

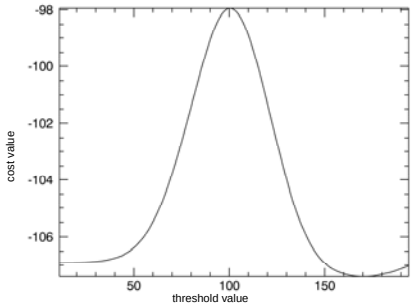
Figure 3.3: *Histogram and cost function of the first test data on Queensland, Australia. Figure 3.3(a) is the histogram of the log-ratio image shown in figure 3.2(c) or figure 3.2(d). Figure 3.3(b) is the 3D plot of the cost function for each pair of threshold values. The x-axis represents the first threshold value, the y-axis represents the second threshold value. The z-axis is the corresponding cost value. The red spot marks the global minimum of the cost function.*

However, Hessian matrix needs two times derivation of the cost function, which can cause some numerical problems. At the same time it is hard to decide whether the determinant is significantly larger or smaller than zero. Here calculation of the determinant of Hessian matrix is not necessary. As mentioned the x- and y-axis represent the first and the second threshold. If there is only one optimal threshold, the cost function must be independent of the first or the second threshold values and behaves like a constant in the direction of the corresponding threshold. So it is easier to solve the problem by just considering the profile of the cost function

along the x- and y-axis at the position of the red spot. In figure 3.4(a) and 3.4(b) the profiles of the cost function in x- and y-axis direction at the global minimum are displayed, the relative variation of the cost function in y-axis is far smaller than that in x-axis. So the relative difference of the cost function values in the profiles can be used to distinguish the two-class and three-class case.



(a) The profile of cost function in y-direction.



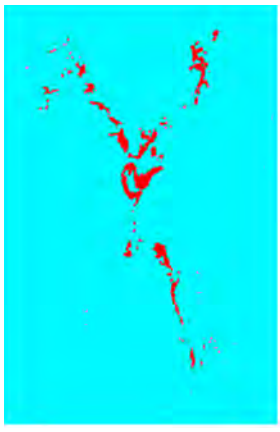
(b) The profile of cost function in x-direction.

Figure 3.4: *Profile of the cost function at the position of global minimum. Figure 3.4(a) illustrates the profile of the cost function in the direction of y-axis, figure 3.4(b) illustrates the profile of the cost function in the direction of x-axis. The variation range of the cost function in figure 3.4(a) is much smaller than 1%, so the cost function seems to be a constant in the direction of y-axis through the global minimum.*

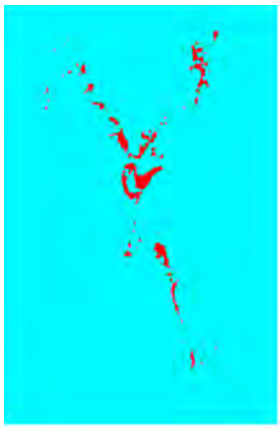
The relative difference is defined as:

$$d = \frac{J_{max} - J_{min}}{J_{min}}, \tag{3.75}$$

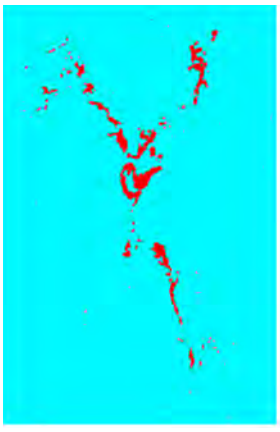
where J_{max}, J_{min} are the maximal and minimal values of cost function along the profile. If the relative difference is smaller than 1%, the cost values in this direction will be considered as constant and the optimal threshold in this direction will also be deleted.



(a) Log-ratio of Gamma.



(b) Log-ratio of Weibull.



(c) Log-ratio of Lognoraml.

Figure 3.5: *The classification results based on different pdfs. Figure 3.5(a) is based on the Gamma log-ratio, figure 3.5(b) is based on the Weibull log-ratio, figure 3.5(c) is based on the log-normal log-ratio.*

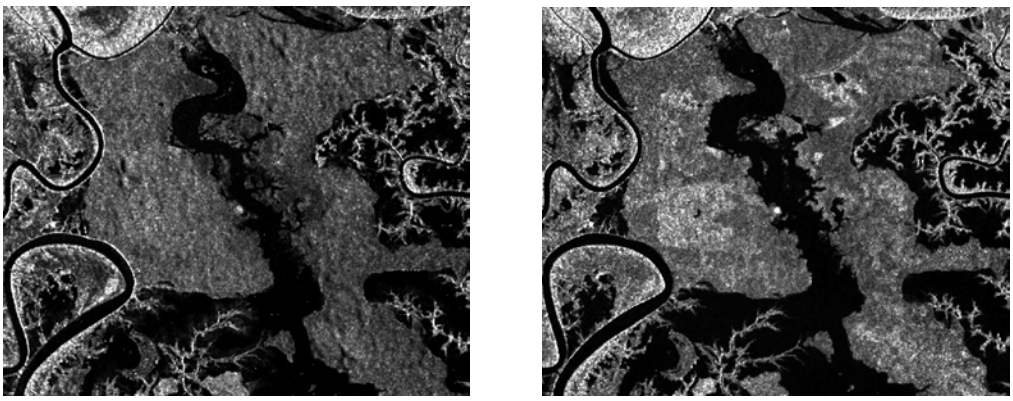
In figure 3.5 the classification results from different pdfs are listed. The cyan color is background and indicates the no-changed area. The red color indicates the negative-changed area. The ratio images in figure 3.2(c) and 3.2(d) prove the classification. The results from the Gamma and the Log-normal ratio pdfs show more detected signals than that from the Weibull ratio pdf. In table 3.2 the runtime for different pdfs is listed. These number has only relative meaning because it can change under different hardware. The runtime of the Gamma pdf is far longer than other pdfs. The Gamma model needs to solve the non-linear equation $m_{Z(1)} = 2\Psi(1, L)$ in table 3.1. A series expansion is used to calculate the trigamma function $\Psi(1, L)$ (see appendix) and a bisection function is used to estimate the parameter L as suggested by Moser and Serpico (2006). Both of them are not so efficient. This could be the reason why the Gamma pdf needs longer time.

Although the runtime of Weibull pdf is shorter, the result in figure 3.5(b) is very conservative than the others. The Log-normal pdf provides a similar good result as Gamma pdf but in surprisingly shorter time. Through this comparison the Log-normal model prove to be a good choice between the quality and efficiency of the classification result.

PDF	**Run time[s]**
Gamma	95.345
Weibull	2.542
Log-noraml	0.967

Table 3.2: *Comparison of run time for different pdfs.*

Because there are rarely positive-changed pixels in the first test data, an another subset is chosen as displayed in figure 3.6 from the same test images. Their ratio images are in figure 3.7.

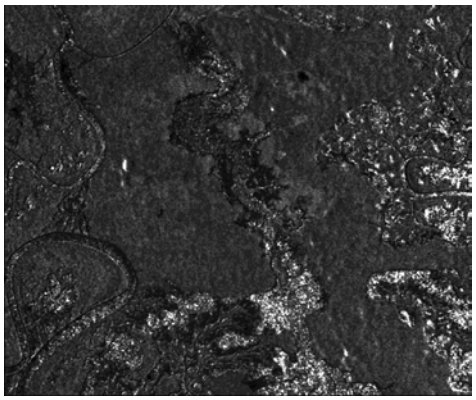


(a) Test area recorded on 21 January , 2011, ©DLR 2013.

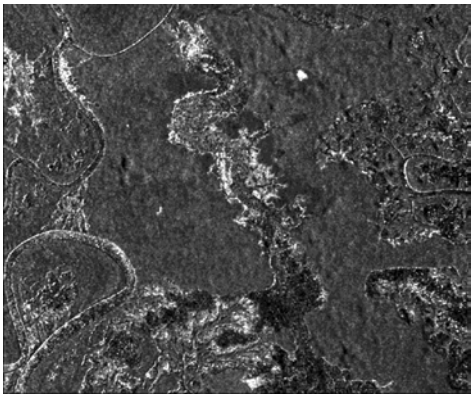
(b) Test area recorded on 27 February, 2012, ©DLR 2013.

Figure 3.6: *The second test data on Queensland, Australia. Figure 3.6(a) is recorded before an flood event, figure 3.6(b) is after an flood event. The gray and bright pixels in both images represent land areas, the dark pixels represent water areas.*

The classification result in figure 3.8(b) is a little bit disappointing. The optimal solution marked with the red spot has only one threshold, because the profile of the cost function through this

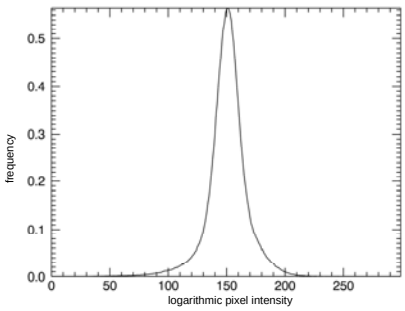


(a) The first ratio image. Quotien between the second and the first image, ©DLR 2013.

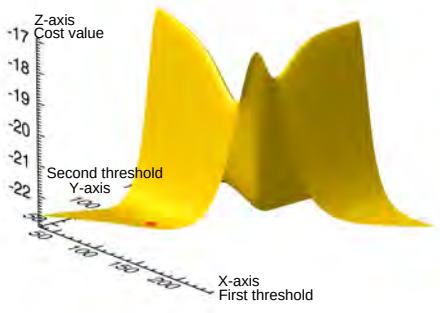


(b) The second ratio image. Quotien between the first and the second image, ©DLR 2013.

Figure 3.7: *Ratio images of the second test area on Queensland, Australia. In the figure 3.7(a) the dark pixels mean negative-changed areas, which indicate the flooded areas, the gray pixels represent no-changed areas, the bright pixels are positive-changed areas, which indicate the receding water. In the figure 3.7(b) the bright pixels mean positive-changed areas, which indicate the flooded areas, the gray bright pixels represent no-changed areas, the dark pixels are negative-changed areas, which indicate the receding water.*



(a) The histogram of log-ratio image.



(b) 3D plot of the cost function.

Figure 3.8: *Histogram and cost function of the second test data on Queensland, Australia. Figure 3.8(a) is the histogram of the log-ratio image shown in figure 3.7(a) or 3.7(b). Figure 3.8(b) is the 3D plot of the cost function for each pair of threshold values. The x-axis represents the first threshold value, the y-axis represents the second threshold value. The z-axis is the corresponding cost value. The red spot marks the global minimum of the cost function.*

point in the x-direction is almost constant. But according to the ratio images in figure 3.7(a) and 3.7(b) a result of three classes is expected: no-changed area, receding flood water and increasing flood water. This is a typical problem of the thresholding method, it sometimes find the optimal solution at a wrong position. Moser et al. (2006) and Bazi et al. (2005) pointed out that this problem was due to the speckle effect, so they filtered the SAR intensity images before the thresholding to reduce the speckle effect. Following their suggestion the Gamma-MAP filter in the software ERDAS Imagine 2011 is used to suppress the speckle effect, the window size is set to 3. The histograms before and after the filtering are displayed in figure 3.9, the blue line is the histogram before filtering, the red line is the histogram after filtering. It can be seen that the histogram gets narrow after filtering, so the samples of each class become easier to distinguish. The 3D plot of the cost function is illustrated in figure 3.10. Because the optimal solution is not clearly seen, the cost function is displayed from two different viewing angles. Three classes are identified from this thresholding, the classification result is displayed in figure 3.11, cyan color means the no-changed area, negative-changed class is marked by red color, blue color indicates the positive-changed area. The classification result is consistent with the ratio images 3.7(a) and 3.7(b).

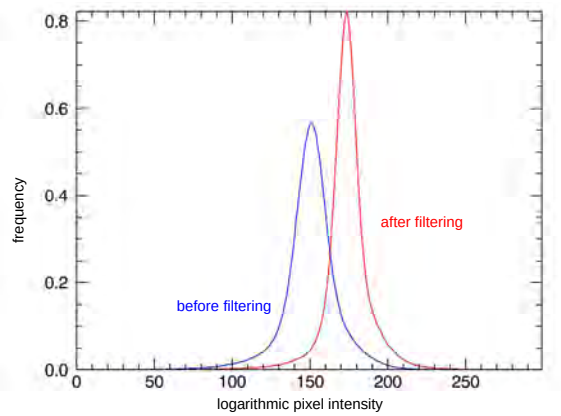
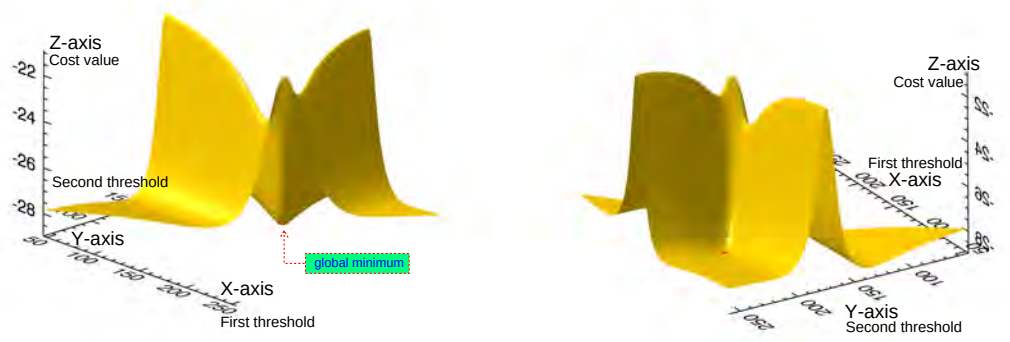


Figure 3.9: *Comparison of the histograms before and after filtering. The blue curve is the histogram before filtering, the red one is after filtering.*

3.6 Conclusion

In this chapter the framework of three-class thresholding proposed in (Bazi et al., 2005) is used to classify the changed areas from two co-registered SAR images. The determinant of Hessian matrix used in (Bazi et al., 2005) is replaced by the relative difference of cost function to distinguish the two-class problem from the three-class problem. Ratio pdfs in (Moser and Serpico, 2006) and the MoLC method in (Nicolas, 2002) were used to model the ratio image and estimate parameters. Because there are not ground truth data for these test area, quantitative validation is impossible, but a good consistence between the classification results and the ratio images can be found.

The histogram thresholding is a simple but efficient method among those of four categories in chapter 1. The runtime is only determined by the complexity of equations for parameter



(a) Cost function, viewed from the front.

(b) Cost function, viewed from the back.

Figure 3.10: *The cost function for the second test area on Queensland, Australia, after two times filtering. Figure 3.10(a) is viewed from the front side, figure 3.10(a) is from the back side.*

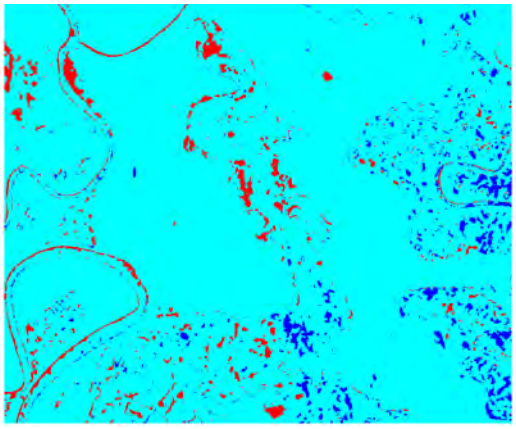


Figure 3.11: *The classification result for the second test area on Queensland, Australia, after two times filtering. Cyan color represents the no-changed area, negative-changed (flooded) areas are marked in red, blue color indicates the positive-changed areas (receding water).*

estimation and the number of bins in the histogram. If the pdf is chosen and the number of bins is constant, the runtime of this method is almost independent of the size of the processed image.

In the literature there is a more general concept called *mixture model* (Bishop and Nasrabadi, 2006) to deal with the problem of classification. It supposes that the data are composed of samples from a few classes and try to segment them into their own class. The commonly used method to solve this problem is called *maximum expectation* (EM). The advantage of EM is that it can estimate parameters even if some data are missing. In the image classification the class ID of each pixel can be considered as missing data and solved in the concept of EM. EM is an iterative method which starts with an initial guess and tries to correct it iteratively. Like many iterative methods, one problem of EM is that a bad initial guess can result in a wrong estimation.

Compared with EM histogram thresholding goes another way. It looks like a greedy algorithm to list all possible solutions and calculate their cost/penalty values directly, then chooses the

best one with the minimal cost value. Usually it is impossible to list all combinations, because the observations are continuous, theoretically there are infinite possibilities. However, the situation will be different if the data are reorganized in a histogram. Histogram plays a role of quantization and simplification of the data. The data within each bin are considered to be same, their variations are neglected. So n bins means that the original data are simplified as data with n scales. This simplification has two advantages:

1. It is not necessary to calculate the cost function for each sample at all, because the data becomes redundant, the samples in each bin are duplicates of each other, each bin need to be calculated only once. It is this simplification that makes the histogram thresholding so efficient.

2. There are only finite possible solutions (possibilities of classification). For example, if the ratio image is reorganized by a histogram with 300 bins, there are only $300^2/2$ possible pairs of thresholds for three-class segmentation. It is this simplification that makes the enumeration of all solutions possible.

Every coin has two sides, this simplification by histogram also introduces errors in the data. In the second test data the method of histogram thresholding made a wrong decision, as shown in figure 3.8(b). From this viewpoint the suggestion of filtering can be better understand. Filtering will smooth the variation of the data, so the simplification by the histogram becomes more reasonable.

Classification result from the histogram thresholding can also be used as an initial guess for EM or other iterative methods. In the chapter 5 a new optimization method in the literature based on graph-cut algorithm will be introduced. This method provides more flexibility and is more general than EM. In the next chapter some information from single images will be used. Compared with the histogram thresholding, which only uses the ratio image, a better classification result will be expected if more information is used.

Chapter 4

Statistical hypothesis test

Observations obtained from an instrument are vulnerable to many factors such as environment impacts, limited instrument accuracy, random error and so on. If two observations are obtained at different dates, they are compared with each other. In general they are not same. This difference could arise from stochastic errors or could suggest that the observed object has already changed. The technique to distinguish this difference is called *statistical hypothesis test* (Benning, 2011). In this chapter, this technique is used to compare pixel intensities on two co-registered SAR images. It should be able to classify the image pixels into three classes: positive-, negative- or no-change. This chapter begins with a general introduction of statistical hypothesis test. Then based on the characters of SAR signal an appropriate test variable and two hypotheses are described to classify SAR images. This technique can be used either pixel by pixel or area by area (Oliver, 1996), these two methods are presented and discussed in the third section.

4.1 Basic hypothesis test theory

Mathematically an one-sided statistical hypothesis test can be formulated as following:

Given a random variable Y following a known statistical distribution with *probability density function* (pdf) $f_Y(y|x)$, where y is a realization or an observation of the random variable Y, x is the parameter of this function, the task is to check whether x is equal to or smaller than a reference value x_0 based on the set of samples y. The decision should be made between two cases: $x = x_0$ and $x < x_0$. They are formulated in statistics respectively by a *zero hypothesis* $\mathcal{H}_0$ and an *alternative hypothesis* $\mathcal{H}_\mathrm{A}$:

$$\mathcal{H}_0 : x = x_0, \tag{4.1}$$

$$\mathcal{H}_\mathrm{A} : x < x_0. \tag{4.2}$$

Similarly another one-sided hypothesis test can also be defined with the same $\mathcal{H}_0$ but different alternative hypothesis $\mathcal{H}_\mathrm{A}$: $x > x_0$. The formulation is similar, without loss of generality only one of them is discussed here.

To conduct a hypothesis test a set of empirical samples (observations) y should be at first selected, and then the parameter x should be estimated from the samples. Instead of comparing x and x_0 directly a test variable t will be generated from x and x_0 so that t follows a known distribution with *cumulative density function* (cdf) $F_T(t)$, then check if this test variable t is probable under an *false alarm rate* α or equivalently *significance level* $1 - \alpha$. This false alarm rate α can also

be described by probability $P(t < q) = \alpha$, where $P(\mathrm{T})$ means the probability of condition T, variable q defines the error rate and is called *quantile*. These can be summarized as:

$$t = t(x, x_0) \sim F_T(t), \tag{4.3}$$
$$F_T(q) = \alpha. \tag{4.4}$$

If the test variable t is not smaller than the quantile q, the zero hypothesis $\mathcal{H}_0$ will be accepted, otherwise alternative hypothesis $\mathcal{H}_\mathrm{A}$ will be accepted. These decision rule can be formulated as:

$$\mathcal{H}_0 \quad \text{accepted}, \quad \text{if } t \geq q, \tag{4.5}$$
$$\mathcal{H}_\mathrm{A} \quad \text{accepted}, \quad \text{if } t < q. \tag{4.6}$$

To illustrate previous discussions an example is given here. Suppose the task is to check whether the length of a desk x is equal to a reference value $x_0 = 1\mathrm{m}$, the length of desk is measured many times and obtain a series of empirical observations y_i for $i = 1, 2, \cdots, N$. If the observations are assumed to be Gaussian distributed with a known variance σ_0^2, the mean value can be estimated as $\mu = \frac{\sum_{i=1}^{N} y_i}{N}$. Then a test variable t can be formed following normal distribution:

$$t = \frac{\frac{1}{N}\sum_{i=1}^{N} y_i - x_0}{\sigma_0 / \sqrt{N}} \sim \mathcal{N}(0, 1), \tag{4.7}$$

where the test variable t follows normal distribution, so the quantile q can be easily calculated given a fixed error rate α.

4.2 Hypothesis test for SAR change detection

The task of change detection is to check whether the objects in two SAR images have changed. As known from last chapter, the SAR signal can be modeled by the Gamma or the Nakagami distribution, so the problem here is to test whether two random variables come from the same Gamma distribution. If the SAR signal are correctly calibrated, the difference in two co-registered SAR images only represents random error or change of the object.

As known from (4.1) a test variable and a quantile are necessary to conduct hypothesis test. At the same time the test variable should be related to a known distribution. This section will demonstrate that the ratio image follows a *incomplete beta distribution*, according to this function an appropriate test variable and quantile can be found. Because there are many variables used in this section, table 4.1 is used to explicate the notation for some variables.

4.2.1 Hypothesis test in the literature

The random observation error in SAR images comes from the speckle effect which is usually considered as a multiplicative noise and formulated as (Oliver, 1996):

$$G = n\sigma, \tag{4.8}$$

Notation	Variable
G	pixel intensity
N	number of pixels
L	number of looks
n	speckle noise
σ	radar cross section
r	observed ratio
γ	true ratio
α	false alarm rate
q	quantile
δ	auxiliary variable for quantile

Table 4.1: *Notations for variables.*

where G is pixel intensity, n is speckle noise and σ is parameter RCS. Equation (4.8) suggests that every pixel in SAR images is characterized by its own RCS σ, which is influenced by the speckle noise n, so that the resultant observation is registered in SAR images as pixel intensity G. RCS can be used to identify the observed object. So the change in the real world can be quantized as the change in RCS parameter. The task is to compare the following parameters:

$$x = \sigma_1, \quad x_0 = \sigma_2. \tag{4.9}$$

The corresponding hypotheses are:

$$\mathcal{H}_0 : \sigma_1 = \sigma_2, \tag{4.10}$$

$$\mathcal{H}_A : \sigma_1 < \sigma_2 \quad \text{or} \quad \sigma_1 > \sigma_2. \tag{4.11}$$

In the literature a hypothesis test can be found by Oliver (1996). Oliver (1996) started with assumption that the SAR signal follows the Gamma distribution and used a pdf function for the ratio image derived by Touzi et al. (1988). At the end they could describe the cdf of ratio image as sum of a series expansion, then the false alarm rate α for hypothesis test could be described as a function of quantile q :

$$\begin{aligned} \alpha &= P_E(q) = P(r > q) \\ &= \frac{\Gamma(N_1 + N_2)}{\Gamma(N_1)\Gamma(N_2)} \sum_{k=0}^{N_2-1} \binom{N_2 - 1}{k} \frac{(-1)^{N_2-k-1}}{N_1 + N_2 - k - 1} \left(1 + \frac{N_2 \cdot q}{(N_1 - 1)\gamma}\right)^{-(N_1+N_2-k-1)}, \end{aligned} \tag{4.12}$$

where N_1, N_2 are number of pixels in two homogeneous regions, $r = \bar{G}_1/\bar{G}_2$ is the observed ratio of mean intensity, $\gamma = \sigma_1/\sigma_2$ is the true ratio of mean intensity, $\Gamma(\cdot)$ is the Gamma function. Using variable γ the hypothesis test (4.10) and (4.11) can be formulated as:

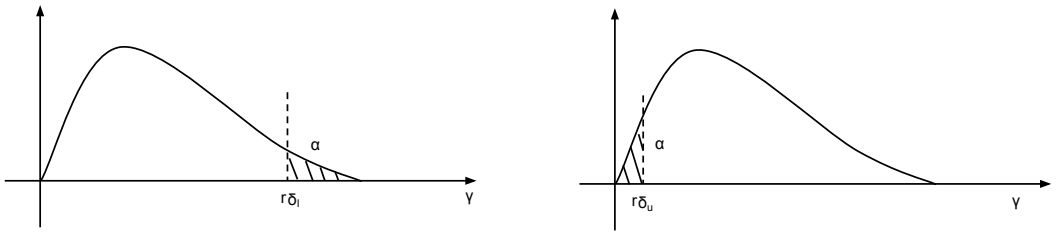
$$\mathcal{H}_0 : \gamma = 1, \tag{4.13}$$

$$\mathcal{H}_A : \gamma < 1 \quad \text{or} \quad \gamma > 1. \tag{4.14}$$

The decision rule by Oliver (1996) is different compared with a traditional hypothesis test. Usually the quantile is calculated and compared with the test variable or parameter. Oliver

(1996) did not try to form a test variable, but calculated the quantile directly, which formed boundary of confidence interval for the parameter. If the parameter was contained in this confidence interval, $\mathcal{H}_0$ would be accepted, otherwise $\mathcal{H}_A$ will be accepted. This formulation is equivalent to the traditional one and but mathematically easier to deal with.

Because the equation (4.12) is actually a function of $r = \frac{q}{\gamma}$, Oliver (1996) defined the quantile for γ as $q = r\delta$. In this way two quantiles for confidence interval are defined as: $q_1 = r\delta_l, q_2 = r\delta_u$,



(a) Interval to acctept $\mathcal{H}_A$.

(b) Interval to acctept $\mathcal{H}_A$.

Figure 4.1: *False alarm rate for both one-sided hypothesis test.*

as illustrated in figure 4.1, then the error probability can be calculated as follows:

$$P(\gamma < \delta_l r) = P(r > \frac{\gamma}{\delta_l}) = P_E(\frac{\gamma}{\delta_l}) = \alpha, \tag{4.15}$$

$$P(\gamma > \delta_u r) = P(r < \frac{\gamma}{\delta_u}) = 1 - P_E(\frac{\gamma}{\delta_u}) = \alpha, \tag{4.16}$$

the parameters δ_l, δ_u can be determined by (4.12), in this way Oliver (1996) found a confidence interval for observation r:

$$\frac{\gamma}{\delta_l} < r < \frac{\gamma}{\delta_u}. \tag{4.17}$$

or equivalently a confidence interval for true ratio γ:

$$\delta_u r < \gamma < \delta_l r. \tag{4.18}$$

If no change occurs, true ratio γ should be nearly 1, thus if $\gamma = 1$ is not in the interval $[\delta_u r, \delta_l r]$, it suggests a positive ($1 < \delta_u r$) or a negative ($1 > \delta_l r$) changed pixel, otherwise it belongs to the no-change class.

However, the binomial term in this series makes this equation difficult to implement and easy to cause some numerical problems. In next section the same method by Oliver (1996) is used, but try to relate the test variable to a known statistical model.

4.2.2 Ratio of Gamma random variables

There are two ways to compare image pixels: one method is to compare each pixel once a time, the other will group the pixels into areas and then compare each area respectively. It can be noticed that the first method is a special case of the second one, so generally the second situation is discussed at first.

Before doing formula manipulation it should be noted that the parameterization of the Gamma distribution function may be different. Oliver (1996) formulate the Gamma pdf from the background of SAR:

$$p_G(g) = \frac{1}{\Gamma(L)}(\frac{L}{\sigma})^L g^{L-1} e^{-Lg/\sigma}, \tag{4.19}$$

where L is the number of looks, G is the pixel intensity, σ is the parameter RCS.

In statistics, the Gamma distribution is usually written as (Lindgren, 1993):

$$\text{Gamma}(x; k, \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}}, \tag{4.20}$$

where x is random variable , k is the so called *shape parameter*, θ is the *scale parameter*.

Comparing equation (4.20) and (4.20) gives then:

$$\text{Gamma}(G; k = L, \theta = \frac{\sigma}{L}) = p_G(g). \tag{4.21}$$

There is not much difference between these two parameterizations. As some mathematical properties will be used during derivation, it will be better to use (4.20) at first and then change the parameter in the final result according to (4.21).

Assuming that SAR images have been segmented into many homogeneous areas with pixel number N_i, the pixels in each area of the first image follow the same Gamma distribution:

$$X_i \sim \text{Gamma}(k_1, \theta_1), \quad \text{for} \quad i = 1, 2, \cdots, N_1, \tag{4.22}$$

where k_1, θ_1 are parameters for the Gamma distribution, which are constant for all pixels in a homogeneous area. For homogeneous area in the second images the situation is similar:

$$X_j \sim \text{Gamma}(k_2, \theta_2), \quad \text{for} \quad j = 1, 2, \cdots, N_2. \tag{4.23}$$

If an area of pixels have changed, there must be also a difference between their mean values, the task now is to find the pdf for the mean value of N Gamma distributed random variables, this can be solved based on following three properties of Gamma distribution (Lindgren, 1993):

1. Summation property:
   If $X_i \sim \text{Gamma}(k_i, \theta)$ for $i = 1, 2, \cdots, N$, then $\sum_{i=1}^{N} X_i \sim \text{Gamma}(\sum_{i=1}^{N} k_i, \theta)$.
2. Scaling property:
   If $X \sim \text{Gamma}(k, \theta)$, then for any $c > 0$, $cX \sim \text{Gamma}(k, c\theta)$.
3. Relation to the χ^2 distribution:
   If $X \sim \text{Gamma}(k, 2)$, then X is identical to $\chi^2(2k)$.

The summation property and the scaling property with $c = 1/N$ provide:

$$\bar{X}_1 = \frac{\sum_{i=1}^{N_1} X_i}{N_1} \sim \text{Gamma}(N_1 k_1, \frac{\theta_1}{N_1}), \tag{4.24}$$

$$\bar{X}_2 = \frac{\sum_{i=1}^{N_2} X_i}{N_2} \sim \text{Gamma}(N_2 k_2, \frac{\theta_2}{N_2}). \tag{4.25}$$

Using the scaling property with $c = \frac{2N_i}{\theta_i}$ for $i = 1, 2$ respectively provides:

$$\frac{2N_1}{\theta_1}\bar{X}_1 \sim \text{Gamma}(N_1k_1, 2) = \chi^2(2N_1k_1), \tag{4.26}$$

$$\frac{2N_2}{\theta_2}\bar{X}_2 \sim \text{Gamma}(N_2k_2, 2) = \chi^2(2N_2k_2). \tag{4.27}$$

where the last equality uses the third property. In addition, χ^2_f distribution is related to F-distribution by following property:

$$X_1 \sim \chi^2(f_1), X_2 \sim \chi^2(f_2) \Rightarrow \frac{X_1/f_1}{X_2/f_2} \sim F(f_1, f_2). \tag{4.28}$$

Applying this property to (4.26) and (4.27) provides:

$$y = \frac{\frac{2N_1}{\theta_1}\bar{X}_1/2N_1k_1}{\frac{2N_2}{\theta_2}\bar{X}_2/2N_2k_2} = \frac{\bar{X}_1}{\bar{X}_2} \cdot \frac{k_2\theta_2}{k_1\theta_1} \sim F(2N_1k_1, 2N_2k_2). \tag{4.29}$$

With the parameter relation in (4.21) this equation is equal to:

$$y = \frac{\bar{G}_1}{\bar{G}_2} \cdot \frac{\sigma_2}{\sigma_1} \sim F(2N_1L_1, 2N_2L_2), \tag{4.30}$$

where $\bar{G}_i$ is the mean intensity of a homogeneous area, σ_i is RCS of this homogeneous area, N_i is pixel number, L_i is the number of looks. Using the variable r, γ defined by Oliver (1996), equation (4.30) can be written as:

$$y = \frac{r}{\gamma} \sim F(2N_1L_1, 2N_2L_2). \tag{4.31}$$

In this way the method by Oliver (1996) can be related to this equation.

In order to see the physical meaning of this equation, *mean speckle noise* can be defined from equation (4.8) as follows:

$$\bar{n} = \frac{\bar{G}}{\sigma}, \tag{4.32}$$

so equation (4.30) can also written as:

$$y = \frac{\bar{n}_1}{\bar{n}_2} \sim F(2N_1L_1, 2N_1L_2), \tag{4.33}$$

which means that the ratio of mean speckle noise in two co-registered SAR images follows F-distribution.

According to (Weisstein, Eric W., n.d.) the cdf of F-distribution $F(x; f_1, f_2)$ can be formulated as:

$$cdf_X(x; f_1, f_2) = I_{\frac{f_1 \cdot x}{f_1 \cdot x + f_2}}(\frac{f_1}{2}, \frac{f_2}{2}), \tag{4.34}$$

where $I_x(d_1, d_2)$ is called *incomplete beta function*, which can be invoked in MATLAB by routine *betainc(·)* or in IDL by *ibeta(·)*. So (4.34) provides a practical way to replace the series representation in equation (4.12).

Some manipulation should be added to obtain the final result. At first equation (4.12) can be rewritten in variable y as described in equation (4.31):

$$P_{\mathrm{E}}(q) = P(r > q) = \int_{r=q}^{+\infty} pdf_R(r)dr = \int_{y=q/\gamma}^{+\infty} pdf_Y(y)dy, \tag{4.35}$$

where the last step uses equality $pdf_R(r)dr = pdf_Y(y)dy$ and $y = r/\gamma$ to perform variable transformation from r to y.

Because $y \sim F(y; 2N_1L_1, 2N_2L_2)$ as in (4.31), using equation (4.34) can rewrite (4.35) as:

$$P_{\mathrm{E}}(q) = 1 - cdf_Y\left(y = \frac{q}{\gamma}; 2N_1L_1, 2N_2L_2\right) = 1 - I_{\frac{N_1L_1 \cdot \frac{q}{\gamma}}{N_1L_1 \cdot \frac{q}{\gamma} + N_2L_2}}(N_1L_1, N_2L_2). \tag{4.36}$$

So the confidence interval defined in (4.15) and (4.16) can be replaced by:

$$P(\gamma < \delta_l r) = P_{\mathrm{E}}(\frac{\gamma}{\delta_l}) = 1 - I_{\frac{N_1L_1}{N_1L_1 + N_2L_2 \cdot \delta_l}}(N_1L_1, N_2L_2) = \alpha, \tag{4.37}$$

$$P(\gamma > \delta_u r) = 1 - P_{\mathrm{E}}(\frac{\gamma}{\delta_u}) = I_{\frac{N_1L_1}{N_1L_1 + N_2L_2 \cdot \delta_u}}(N_1L_1, N_2L_2) = \alpha. \tag{4.38}$$

Then the parameters δ_l, δ_u defining the confidence interval are given by:

$$I_{\frac{N_1L_1}{N_1L_1 + N_2L_2 \cdot \delta_l}}(N_1L_1, N_2L_2) = 1 - \alpha, \tag{4.39}$$

$$I_{\frac{N_1L_1}{N_1L_1 + N_2L_2 \cdot \delta_u}}(N_1L_1, N_2L_2) = \alpha. \tag{4.40}$$

These two equations provide a easy way to find the confidence interval of the ratio. Similar to (Oliver, 1996), if $\gamma = 1$ is not in the interval $[\delta_u r, \delta_l r]$, it suggests a positive ($1 < \delta_u r$) or a negative ($1 > \delta_l r$) changed pixel, otherwise it belongs to no-change class. The decision rules can be summarized as follows:

Change	**Condition**
no	$\delta_l r \leq 1 \leq \delta_u r$
$+$	$\delta_l r > 1$
$-$	$\delta_u r < 1$

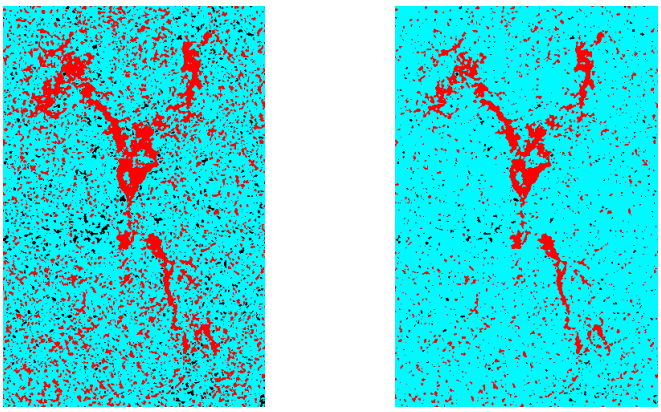
Table 4.2: *Hypothesis test for three-class discrimination.*

There are two ways to use these equations. If $N_1 = N_2 = 1$ then only one pixel is compared or generally an co-registered area can be tested with same pixel number $N_1 = N_2$. Both of them will be used in the next section with different test data.

4.3 Application to test data

4.3.1 Pixel based hypothesis test

In this subsection pixel based hypothesis test is used to classify SAR ratio images. The test data are the same as in the chapter 3. The first test data is a subset extracted from two horizontal polarized ScanSAR images recorded on Queensland, Australia, on 21 January, 2011 and on 27 February, 2012. They are displayed in figure 3.2.

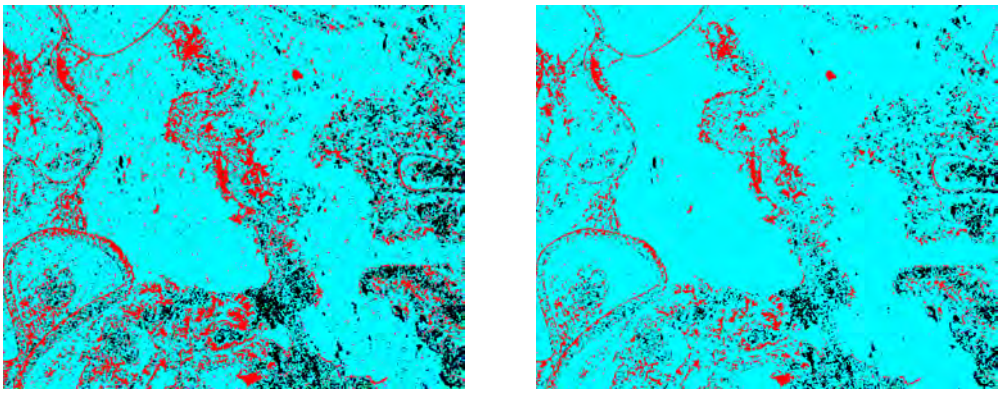


(a) $\alpha = 5\%$ (b) $\alpha = 1\%$

Figure 4.2: *Classification for the first test data on Queensland, Australia, using pixel based hypothesis. Cyan color represents no-changed areas, red color indicates negative-changed areas, black color means positive-changed areas. Figure 4.2(a) is obtained from the false alarm rate $\alpha = 5\%$, figure 4.2(b) is from the false alarm rate $\alpha = 1\%$.*

In figure 4.2 two classified images using pixel based hypothesis test are displayed, where pixels of no-change are marked in cyan, positive-change in red and negative-change in black. Because equations (4.39) and (4.40) depends on the number of looks, the software ERDAS Imagine 2011 is used to estimate the equivalent number of looks (ENL). Within a window size of 5 pixels the estimation are $ENL_1 = 10.6$ and $ENL_2 = 11.3$. Since these two images are recorded by the same SAR system and calibrated by the same processor, it can be approximately assumed that $ENL_1 = ENL_2$. At the same time the false alarm rate α should be set before classification. Figure 4.2(a) is the result with $\alpha = 0.05$, figure 4.2(b) is the result with $\alpha = 0.01$. It can be seen that higher false alarm rate results in more correctly detected signals but at same time more errors. The classification result with error rate $\alpha = 0.01$ seems better, but it can still be noticed that many isolated points on the image, which make the image very "dirty". Practically it is not necessary to set α and ENL respectively, the ENL can be fixed as a constant and attribute all influence of parameters to α, but in this way the meaning of α would be changed.

Because there are rarely negative-change pixels in the first test data, another test area is chosen in the same SAR images shown chapter 3, which are displayed in figure 3.6. Their ratio images are in figure 3.7 and classification result with two different false alarm rates are displayed in figure 4.3. The ENLs calculated for these two images are $ENL_1 = 6.2$, $ENL_1 = 4.3$ respectively. The color table in figure 4.3 is the same as that in the first test data. In figure 4.3 similar effect of α values can also be observed. Pixels of three classes can be noticed directly. Compared with their ratio images shown in figure 3.7, the classification result is plausible. The difference between



(a) $\alpha = 5\%$. (b) $\alpha = 1\%$.

Figure 4.3: *Classification for the second test data on Queensland, Australia using pixel based hypothesis. Cyan color represents no-changed areas, red color indicates negative-changed areas, black color means positive-changed areas. Figure 4.3(a) is obtained from the false alarm rate $\alpha = 5\%$. Figure 4.3(b) is from the false alarm rate $\alpha = 1\%$.*

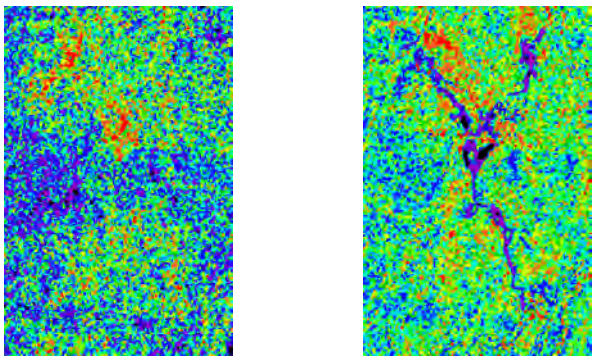
two SAR images in figure 3.6 is attributed to the flood event. The negative-changed (red) areas in the boundary of the rivers indicate the increase of flood water. The positive-changed (black) areas in the boundary of the rivers indicate the decrease of flood water. However, this is a very difficult test data. The detected changed areas are scattered or isolated. There are many small changed areas within the rivers. The reason for this classification result is probably that the alluvial sediment areas within the rivers are inundated after the flood event. The problem whether such fine changes in the classification result should be kept depends on the definition of the reference data. This could be variable in different areas of applications.

4.3.2 Area based hypothesis test

The hypothesis test described in subsection (4.2.2) can also be used on areas of pixels. If at first the images are segmented into homogeneous areas, in which all pixels can be approximately considered as realizations of same random variable, equations (4.39) and (4.40) can be used to calculate the quantile for the confidence interval and perform hypothesis test. However, the classified result is not so good as expected.

In figure 4.4 two segmented images are displayed. The processed image data is the same as the data used in figure 3.2. These two images are segmented by k-means algorithm with 16 classes and then smoothed by the graph-cut algorithm described in the chapter 5. Each area is labeled in different color. The hypothesis test is then applied to every homogeneous area in these images. The classified result is described in figure 4.5. Figure 4.5(a) is the result from $\alpha = 0.1\%$ with 16 classes, figure 4.5(b) is the result from $\alpha = 0.1\%$ with 5 classes. It can be seen that the area based hypothesis test needs only a smaller false alarm rate $\alpha = 0.1\%$ to get considerable signals, whereas the false alarm rate in figure 4.2 is 5% and 1%. However, the exaggerated error in figure 4.5(b) indicates that the approximation error (assuming only 5 classes) from segmentation dominates the final result. The presumption of area based hypothesis test is that each segmented area in the image is homogeneous. If there were error in segmentation procedure, it would transport in the final result. In figure 4.5(a) a larger class number is assumed, so the

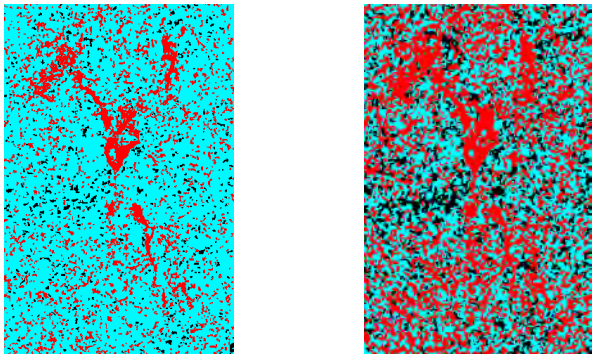
approximation error is smaller and the result looks better. The classification could also be improved if a even larger class number were used, but practically it is not efficient. Comparing the result in figure 4.2 and 4.5 indicates that both pixel and area based hypothesis test can provide a good classification result, but the former one is easier to implement and more efficient than the latter one.



(a) Segmentation for the first SAR intensity image.

(b) Segmentation for the second SAR intensity image.

Figure 4.4: *Segmentation using k-means algorithm. Figure 4.4(a) and 4.4(b) are segmented from the SAR intensity images shown in figure 3.2. Each isolated colored area in figure 4.4(a) and 4.4(b) indicates a homogeneous region.*



(a) 16 classes

(b) 5 classes

Figure 4.5: *Classification for the first test data on Queensland, Australia using area based hypothesis. Figure 4.5(a) and 4.5(b) are obtained from the false alarm rate $\alpha = 0.1\%$, the first one set 16 as the number of classes for k-means algorithm, the second one use 5.*

4.4 Conclusion

After comparison figures 4.2 and 4.3 with 3.5 and 3.11 it can be seen that the classification result from the histogram thresholding is very conservative. In figure 4.2 there are less detected changed pixels but also less error, whereas hypothesis test is too optimistic, it finds more changed pixels but makes many wrong decisions. If the data were not calibrated sufficiently,

there would be more errors in the result by hypothesis test than that by histogram thresholding.

Chapter 5

Markov random field and graph-cut algorithm

5.1 Introduction

Chapter 3 and 4 have introduced two different methods to classify ratio images, but the classification results in these two chapters seem very "dirty". There are many scattered negative- and positive-changed pixels mixed in no-changed areas and vice versa. The reason for this result is the speckle effect of SAR image. Intensity value of SAR image acts like a random variable. Meanwhile the method used in chapter 3 and 4 (except area based hypothesis test) are all pixel based methods, so the randomness of pixel intensity value results in false classification of image pixels, the classification results obtained in chapter 3 and 4 need a post-processing.

The classification result can be considered as a mask, in which every pixel is labeled with its class number, so the post-processing can be seen as correction of the initial labeling of the image. In the literature this task is formulated in the framework of *Markov random field* (MRF) (Li, 1995). In this chapter this concept will be introduced briefly and used to improve the classification results. Section 5.2 and 5.3 will introduce MRF and discuss its traditional solutions. In section 5.4 a modern solution by the graph-cut algorithm in (Boykov and Kolmogorov, 2004) and (Kolmogorov and Zabin, 2004) will be introduced. In the last section this graph-cut algorithm will be used to post-process the classification results.

5.2 Markov random field

The problem of pixel based classification is that it neglects spatial information. Because objects in real world distribute homogeneously, thus observations of the same category should almost gather together and be recorded in this way by sensors. Based on this consideration the expected classification result should be described in area or block instead of pixel.

This problem can be solved in the framework of MRF. The idea of MRF comes from thermostatistics, where the state of molecules in an system is of center interest. In this thermo-system every molecule can have an arbitrary (within a realistic range) state with an corresponding amount of energy, this system is called a *random filed*. In addition molecules in the system can be influenced by their neighbors, this phenomenon can be generally formulated by a neighbor system. The adjacent molecules build then a so called *clique*, in which all molecules are neighbors to each other. Theoretically there are many ways to define a clique based on a corresponding neighbor system. A popular one is based on spatial distance of the molecules. In

this sense this random field is also *Markovian*, so the whole system is called *Markov random field* (MRF).

There are two types of energy in the system: one type is defined for each molecule, the other is for each clique, i.e each set of neighboring molecules. In the framework of MRF, the energy of the clique is called *smoothness energy* because the molecules in the clique are expected to have similar state. The energy for each individual molecule is called *data energy*. The total energy of a MRF is the sum of data and smoothness energy of all molecules in the system. The task is to find the minimum of the total energy and the state of each molecule in that situation. Generally this is a problem of global optimization, the solution depends on how the data and smoothness energy is defined.

The framework of MRF can also be applied to image classification, where every pixel in the image should be labeled with a class ID to represent membership of this class. If all pixels are classified, these labels form a vector or matrix called the *labeling* of the image. The task of image classification is to find the optimal labeling. In order to transform this problem to a MRF, two types of energy are defined similarly: data energy for each pixel and smoothness energy for each pair of adjacent pixels. These two terms should be so chosen that the optimal classification corresponds to the minimum of the total energy. In the following a mathematical formulation are given.

The set of pixels in an image $\mathcal{I}$ is denoted as $\mathcal{V}_0$. A labeling of this image is then denoted as $l = \{l_p, \forall p \in \mathcal{V}_0\}$ where l_p means the label or class ID of pixel $p \in \mathcal{V}_0$. The set of all possible labeling is described as $\mathcal{L}$. The labeling l can also be considered as a mapping from $\mathcal{I}$ to $\mathcal{L}$. If pixel p is classified to the class l_p, its data energy is denoted as $D_p(l_p)$, the smoothness energy of two pixels p and q in a neighbor system $\mathcal{N}$ is denoted as $V_{pq}(l_p, l_q)$. Sometimes $V_{pq}(l_p, l_q)$ is also called *potential*. With these notations the whole energy of the MRF under the labeling l is described as:

$$E(l) = \sum_{p \in \mathcal{V}_0} D_p(l_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(l_p, l_q). \tag{5.1}$$

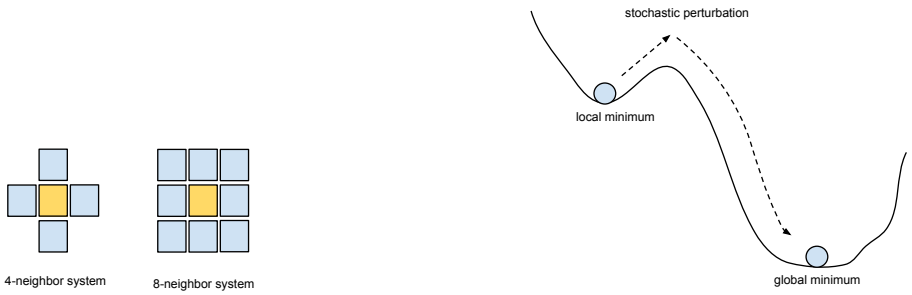
The problem is to find the labeling l that minimizes $E(l)$ globally provided that the data and smoothness energy are suitably defined.

5.3 Traditional solutions

The difficulty in minimizing equation (5.1) is that it is not analytically tractable. In this situation only numeric methods can be used to approximate equation (5.1).

The easiest way is to use a greedy search method, which tests every possible labeling and calculates its total energy in equation (5.1) numerically, at the end choose the labeling with the minimal energy. This method is easy to implement but not practical. Supposing that there is an image with $50\times$ 50 pixels and 16 classes, the whole number of labeling is $16^{2500} \approx 2 \times 10^{3010}$. Such a large computation cost makes the simple greedy search impractical.

Instead of calculating the total energy for all labeling, another method begins at an initial labeling and tries to correct it iteratively until the global minimum is reached. At each iteration only one neighbor of a given pixel is changed, where a usual 4-neighbor or 8-neighbor system is assumed as illustrated in figure 5.1(a). Using equation (5.1) the total energy can be updated



(a) Two neighbor systems.

(b) Illustration of simulated annealing.

Figure 5.1: *Two neighbor systems and illustration of simulated annealing.*

for each iteration and the new labeling will be accepted if it has a smaller energy. This procedure repeats until every pixel has been tested many times, so hopefully the result of the final labeling corresponds to the global minimum of the energy. However, this method suffers from a potential danger of local minimum. If a bad initial labeling is chosen, this algorithm will stop at the position of a local minimum.

The third method to solve the problem of local minimum is called *simulated annealing* (Besag, 1986). This method uses also an iterative procedure to approach a global minimum but with a stochastic feature. Similar to the second method the total energy is updated for the new labeling each time. Their difference is that the second method only accepts the new labeling if the total energy becomes smaller. This condition could be too "hard" for a realistic decision and results in a local minimum. In order to reduce false decision this "hard" rule will be modified to a "soft" one, i.e this "hard" rule can be broken under a probability, such that the estimation can spring out of a local minimum. This is illustrated in figure 5.1(b). Typically this break or perturbation is implemented by a random number, which is then compared with a given threshold T. If the total energy becomes smaller, the new labeling will be accepted. If the total energy is equal or larger than the old one but at the same time the random number is smaller than the given threshold, this new labeling will still be accepted. It is this stochastic feature that makes the algorithm not stop at a local minimum. The stochastic perturbation is controlled by the threshold parameter, which is usually interpreted as *temperature*. Generally this parameter will be so changed during iteration that the stochastic perturbation has decreasingly less influence, at the end the estimation will be stable and converge to the global minimum (Besag, 1986). The whole procedure is similar to annealing metal, so in the figurative sense this algorithm is called simulated annealing. This method is a good idea to solve the problem of local minimum, however, very hard to control. If the threshold parameter (temperature) decreases too quickly during iteration, the stochastic perturbation is not large enough to force the estimation out of a local minimum. If it decreases too slowly, the stochastic perturbation could force the estimated state out of the global minimum. Because of this difficulty simulated annealing does not always provide the global minimum in practice.

5.4 Graph-cut algorithm

There are some similarities between a graph and a MRF. A graph is composed of nodes and edges, whereas a MRF is defined on a field (set of points) with a neighbor system. A point in the MRF can be considered as a node in the graph, and a pair of points defined by the neighbor system can be seen as an edge. The energy defined in the MRF is also similar to the weight or flow in a graph. As known a popular methodology in mathematics is to transform a difficult problem to a easy one, or an unsolved to a solved one. Because of these similarities some studies have tried to relate the energy function in a MRF to a graph, and then solve the problem of global optimization in the framework of graph theory. Among them the work by Kolmogorov et al. (2004) and Boykov et al. (2004) are most important. Both of them have set up a theory to transform the energy function in a MRF to a graph, but the method by Kolmogorov et al. (2004) is more general and easy to implement. Boykov et al. (2004) used many additional nodes to expand a graph, so the number of nodes after transformation was much larger and the structure of the graph was different from the original MRF, whereas Kolmogorov et al. (2004) used only two additional nodes and preserved the structure of the MRF. Both of these methods are limited to a two-class problem, which means that the nodes in the graph are binary variables, and can be labeled as 0 or 1 (two-class problem). Boykov et al. (2004) have invented two methods to expand a two-class problem to a general multi-label(multi-class) problem. In addition, Boykov et al. (2004) have also invented a new max-flow algorithm to solve the global optimization problem in a graph. Based on their work the energy function in a MRF can be equivalently transformed to a graph, and the global optimization problem in a MRF can be solved in the framework of the graph theory. At beginning the general relation between max-flow and min-cut problems is explained.

5.4.1 Max-flow and min-cut

A graph is composed of nodes and edges. Each edge in the graph can transport a mount of *flow* between its two nodes. The maximum of the flow is limited by the *capacity* of the edges. If the flow in one edge has reached its capacity, this edge is *blocked* and can not transport flow any more. The connection between the source and the sink through edges forms a *path* for flow transport. The maximum of flow in this path (max-flow) is reached if the capacity of one edge in the path has been fully used. In this situation this path is called *blocked*. In order to find the max-flow from source to sink, every possible path should be fully used to transport flow until they are all blocked, then the sum of flow in these paths is the maximal flow.

Another interesting viewpoint is given by Ford and Fulkerson (1956). According to their theorem the maximal flow in each path is only determined by those edges with minimal capacities, i.e the blocked edges. So the max-flow in the whole graph is equal to the sum of capacities of the blocked edges. In addition these blocked edges *cut* the graph into two parts, one part is connected with the source, denoted as $\mathcal{G}_s$ the other is connected to the sink, denoted as $\mathcal{G}_t$. The blocked edges are the channel of these two parts. In the case of max-flow, the capacities of the blocked edges are fully used and determine the maximal flow. In this viewpoint Ford and Fulkerson (1956) formulated the max-flow as determination of the best cut of a graph. There are many possible cuts which divide the graph into two parts $\mathcal{G}_s, \mathcal{G}_t$. The best cut is so defined that the sum of capacities of the cut edges is minimal (min-cut). These cut edges are equivalent to the blocked edges in the max-flow problem. According to (Ford and Fulkerson, 1956)

max-flow is equivalent to min-cut, they are dual to each other. In addition, Ford and Fulkerson (1956) provided a method to solve the max-flow problem called *augmentation path algorithm*.

Since max-flow and min-cut are equivalent, each of them provides a way to solve the global optimization problem in a graph. The viewpoint of max-flow is better for numerical implementation, whereas min-cut is easier to formulate.

5.4.2 Transformation to binary graph

Kolmogorov and Zabin (2004) transformed the minimal energy of a MRF to the max-flow/min-cut in a graph. The purpose of transformation is to benefit from some global optimization methods in the graph theory. Before direct application of the max-flow/min-cut algorithm to a transformed graph, three questions should be answered:

1. How to define the nodes in the transformed graph?
2. How to define the edges in the transformed graph?
3. How to set the capacities of the edges in the transformed graph?

The graph should be constructed as simple as possible. It is essentially important to make sure that the optimal result obtained from the max-flow/min-cut algorithm is exactly the same as that from the criterion of minimal global energy in a MRF.

Defining nodes

The solution to the first problem was easily solved by Kolmogorov and Zabin (2004). Their transformation does not add additional nodes (except source and sink) to a graph, so the nodes in the graph are the same as the points in the MRF. The last two problems are more complicated. At first some mathematical notations in (Kolmogorov and Zabin, 2004) are used to assist description.

The pixels in an image $\mathcal{I}$ are denoted as $\mathcal{V}_0 = \{v_1, v_2, \cdots, v_n\}$. The corresponding graph is then denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the set of nodes $\mathcal{V} = \mathcal{V}_0 \cup \{s, t\}$, where each pixel in the image $\mathcal{I}$ is equivalent to a node in the graph $\mathcal{G}$. Only two additional terminals, source s and sink t, are added. $\mathcal{E}$ is the set of edges of the graph. For a two-class problem, the class ID can be simplified as 0 or 1, so a labeling l of the image $\mathcal{I}$ can be described by a vector of binary variables: $l = \{x_1, x_2, \cdots, x_n\}$, where $x_i = 1$ or $x_i = 0$ for $i = 1, 2, \cdots, n$. Given a labeling a measure can be used to describe the quality of this labeling. It is noted by Kolmogorov and Zabin (2004) as $E(x_1, x_2, \cdots, x_n)$, which is also the energy function defined in the MRF.

According to (Ford and Fulkerson, 1956), the max-flow is equivalent to the min-cut of a graph, the global optimization is then characterized by the min-cut which separates the graph into two parts $\mathcal{G}_s$ and $\mathcal{G}_t$. It is natural to define following labeling rule related to a cut:

$$v_i \in \mathcal{G}_s \quad \Leftrightarrow \quad x_i = 0, \tag{5.2}$$

$$v_i \in \mathcal{G}_t \quad \Leftrightarrow \quad x_i = 1. \tag{5.3}$$

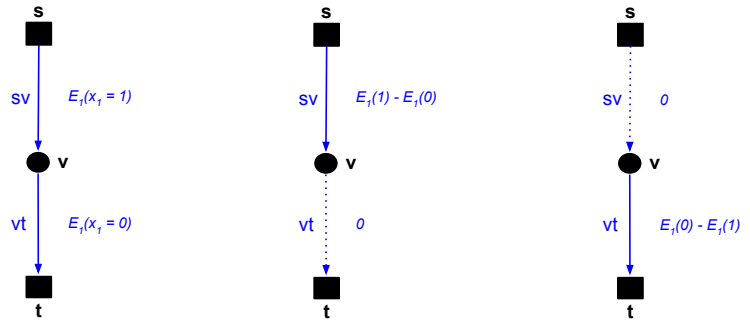
Based this labeling rule the min-cut result is then related to a unique classification of the image. The following concern is to choose edges and their proper capacities.

Defining edges and capacities

The total energy $E(x_1, x_2, \cdots, x_n)$ defined on a graph can be divided into two types. One type is defined on each node, the other is on each pair of nodes or on edges. So generally the energy $E(x_1, x_2, \cdots, x_n)$ can be rewritten as:

$$E(x_1, x_2, \cdots, x_n) = \sum_{v_i \in \mathcal{V}_0} E_1(x_i) + \sum_{(v_i, v_j) \in \mathcal{N}} E_2(x_i, x_j), \tag{5.4}$$

where E_1 is the term defined on each single node, E_2 is the term defined on each pair in the neighbor system $\mathcal{N}$. Equation (5.4) describes the total energy of the whole graph as a linear combination of energies on each node and edge. Further more, Kolmogorov and Zabin (2004) proved a theorem: The whole graph construction can be divided into small steps, in which each node or pair in equation (5.4) can be transformed into a subgraph individually and then all subgraphs are linearly combined together to form the final graph. According to this theorem, the task now is how to construct a graph equivalent to $E_1(x_i)$ and $E_2(x_i, x_j)$ respectively.



(a) Original. (b) Flow $= E_1(0)$. (c) Flow $= E_1(1)$.

Figure 5.2: *Graph construction for energy function $E_1(x)$, (Kolmogorov and Zabin, 2004).*

The first case is easy. As illustrated in figure 5.2(a), the source s and sink t are squares, the circle represents a node v in the graph. There are two edges in this graph, sv and vt. The first one is assigned with capacity $E_1(x = 1)$, the second one is $E_1(x = 0)$. According to the min-cut criterion all possible solutions are listed in the table 5.1, where the first column is the minimal energy, the second one represents the smallest capacity, the third one describes the corresponding cut edge according to the max-flow/min-cut. The result is described in the forth column. It can be seen that the labeling in the forth column is exactly the same as the values of variable in the first column. So if there was a MRF with one node v and its energy function was $E_1(x)$ for $x = 0, 1$, the labeling of the graph in 5.2(a) by the max-flow/min-cut algorithm would be the globally optimal one, i.e the labeling with the global minimal energy.

If the energy function is modified as $\tilde{E}_1(x) = E_1(x) + c$, where is c a constant, the optimization will be still the same as that in figure 5.2(a). This means that a mount (maximum) of capacity is used according to the max-flow/min-cut criterion, at the end the same result is obtained. The graph in figure 5.2(a) can then be modified to the one in figure 5.2(b), where a constant (flow) $E_1(0)$ is subtracted from both edges. If $E_1(1) - E_1(0) < 0$ a constant $E_1(1) - E_1(0)$ can be subtracted from both edges in figure 5.2(b) to avoid a negative capacity, the result is then

described in figure 5.2(c). Graphs in figure 5.2(a), 5.2(b) and 5.2(c) are equivalent, the last two are more efficient for numeric calculation, where edges with zero capacity can be neglected. Because flows and capacities are defined as positive, zero capacity is the minimum, so the edges with zero capacity will always be cut (min-cut), which is described by dashed lines. They can be considered as virtual edges in the graph.

Min Energy	Capacity	Cut Edge	Result	Labeling
$E_1(x=0)$	$E_1(0) \leq E_1(1)$	vt	5.2(b)	x = 0
$E_1(x=1)$	$E_1(1) \leq E_1(0)$	sv	5.2(c)	x = 1

Table 5.1: *Test table of graph construction for $E_1(x)$ energy function.*

After construction of the subgraph for energy function $E_1(x)$, the second step is to construct a graph for energy function $E_2(x_1, x_2)$. This situation is a little complicated that the former one. To simplify notation the following abbreviations were used by Kolmogorov and Zabin (2004):

$$E_2(x_1 = 0, x_2 = 0) = A \tag{5.5}$$
$$E_2(x_1 = 0, x_2 = 1) = B \tag{5.6}$$
$$E_2(x_1 = 1, x_2 = 0) = C \tag{5.7}$$
$$E_2(x_1 = 1, x_2 = 1) = D \tag{5.8}$$

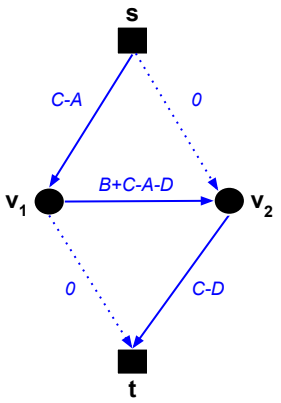


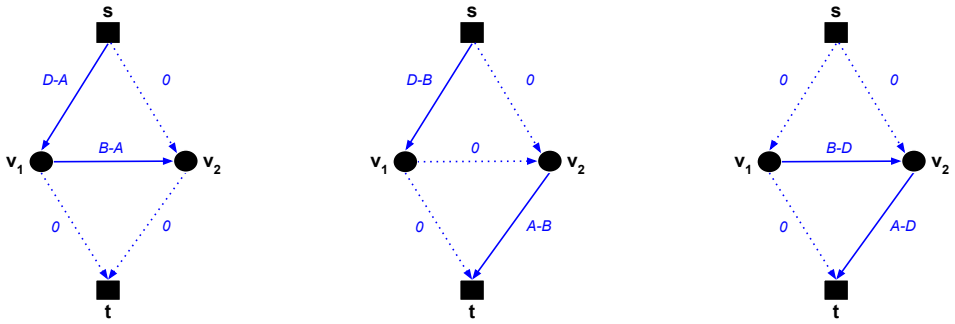
Figure 5.3: *Graph construction for energy function $E_2(x_1, x_2)$ in the case of C = max, (Kolmogorov and Zabin, 2004)*

The graph after transformation in a special case is illustrated in figure 5.3, where C is the maximum among $\{A, B, C, D\}$, so capacities $C - A, C - D$ are positive. Kolmogorov and Zabin (2004) proved a necessary (and sufficient) condition to transform the energy function into a graph: The energy function should be regular. This condition implies that $B + C > A + D$, so the capacity $B + C - A - D > 0$. Similar to the case of $E_1(x)$, the labeling result is summarized in table 5.2. The corresponding graphs after application of the max-flow/min-cut algorithm are listed in figure 5.4. It can be seen that the first and forth columns are exactly the same, this

means the labeling of the max-flow/min-cut criterion is equivalent to that of the min-energy criterion.

Min Energy	Capacity	Cut Edge	Result	Labeling
$E_2(0,0) = A$	$C - D \leq B + C - A - D$ $C - D \leq C - A$	v_2t	5.4(a)	(0, 0)
$E_2(0,1) = B$	$B + C - A - D \leq C - D$ $B + C - A - D \leq C - A$	v_1v_2	5.4(b)	(0, 1)
$E_2(1,1) = D$	$C - A \leq B + C - A - D$ $C - A \leq C - D$	sv_1	5.4(c)	(1, 1)

Table 5.2: *Test table of graph construction for $E_2(x_1, x_2)$ energy function.*

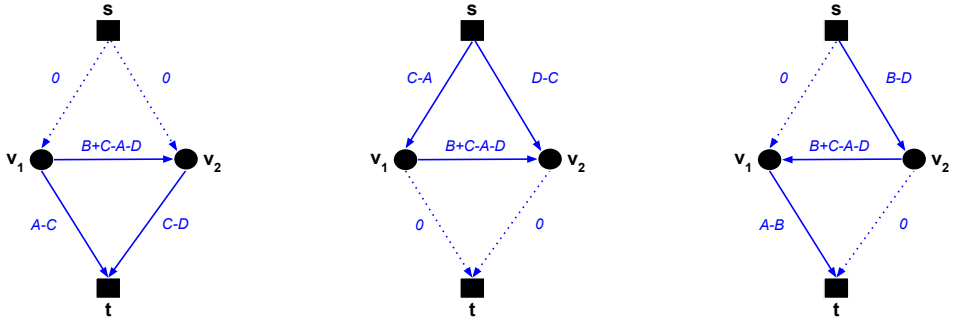


(a) Flow $= C - D$. (b) Flow $= B + C - A - D$. (c) Flow $= C - A$.

Figure 5.4: *Graph in figure 5.3 after application of the max-flow algorithm.*

The graph in figure 5.3 is only for the case $C = \max\{A, B, C, D\}$. If $B < C$, $D < C$ and $C < A$ then $C - A < 0$, a constant $A - C$ can be added to the path sv_1t of the graph in figure 5.3 to avoid negative capacity. The result is shown in figure 5.5(a). In this case A is the maximum, if $D > C$ or $D > B$ then $A + D > B + C$, this result is controversy to the regularity of the energy function, so D must be the minimum, the expected labeling is then $(1, 1)$ according to the labeling rule. At the same, the edges in the graph in figure 5.5(a) with zero capacity will always be cut, this corresponds to the labeling $(1, 1)$ exactly. So generally identical labeling is obtained from two different criteria.

If $B < C$, $A < C$ and $C < D$, then $C - D < 0$, a constant $D - C$ can be added to the path sv_2t, the result is the figure 5.5(b), which is similar to the graph in figure 5.5(a). If $A < C$, $C < D$ and $C < B$, then B must be the maximum according to regularity. C in figure 5.3 can be replaced with B to obtain a similar graph in figure 5.4(c). They can be verified in the same way as described above.



(a) $C < A, D < C$ (b) $C < D, A < C$ (c) $C < A, C < D$

Figure 5.5: *Graph construction for energy function $E_2(x_1, x_2)$ in the case of $C \neq max$*

5.4.3 Multi-label graph and max-flow algorithm

Extension to multi-label graph

Boykov and Kolmogorov (2004) invented a algorithm to construct a equivalent graph. But their method is more complicated. However, they invented two methods called *α-expansion* and *$\alpha\beta$-swap* to extend the graph construction for a multi-labeled graph.

The idea is to approximate the global minimum by repeating the min-cut algorithm for a binary graph. At each iteration only nodes of two classes are considered, in this way a binary subgraph is formed and the max-flow/min-cut algorithm can be applied. This process will repeat for other possible combinations of two classes. If it repeated sufficiently long, the global minimum of energy would be reached. At each iteration α-expansion will add more pixels into the class α, whereas $\alpha\beta$-swap will exchange the labels among the nodes of class α or β. These two methods will repeat many times for all classes. In reality, infinite iterations are impractical, as pointed out by Boykov and Kolmogorov (2004) a good approximation of the global minimum is at least guaranteed. After testing the $\alpha\beta$-swap prove to be a better method than α-expansion.

Max-flow algorithm

There are many algorithms in the literature to find the max-flow of a graph, such as augmenting path in (Ford and Fulkerson, 1956). Boykov and Kolmogorov (2004) proposed a novel algorithm to perform max-flow optimization. According to their experiments their new method significantly outperforms other standard algorithms.

A tree structure is used to organize nodes and edges. There are two search trees S, T with roots at the source and the sink. The starting node of an edge is called the parent node, the end one is children node. If there are non-saturated edges (capacities are not fully used) with terminals as their parent nodes, the search trees begin to grow and absorb these nodes in the trees. These new members of the trees are the boundaries of the trees, which will try to absorb more nodes nearby. The nodes at boundary are called active nodes, the nodes inside the trees are called passive nodes, others are free nodes. As the search trees S, T grow and touch each other, there

will be a path between the source and the sink. The maximum of flow will be filled in the path and results in the reduction of the capacities of the edges in the path. This can also force some nodes or a part of the search trees to be separated from the original trees. These isolated nodes or parts will be rescanned and reconnected to the search trees to maintain a single-tree structure. Then the updated search trees will try to grow again until a new path is formed from the source to the sink. The algorithm terminates if the search trees cannot grow any more.

5.5 Energy functions

In the last section a graph construction method by Kolmogorov and Zabin (2004) and a max-flow by Boykov and Kolmogorov (2004) were introduce briefly. In this section these methods can be used to improve the classification results from chapter 3 and 4. The problem of pixel based classification methods is the speckle effect of SAR images, which makes the result scattered and isolated. In order to obtain a more realistic classification, spatial information should be taken into account. In the framework of MRF the labeling or classification problem can be simply described as the sum of two energy terms, one is called *data energy*, the other is *smoothness energy*:

$$E(l) = \sum_{p \in \mathcal{V}_0} D_p(l_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(l_p, l_q). \tag{5.9}$$

The former one measures the similarity of the intensities, the latter one measures the spatial proximity or connectivity. Optimization of energy function on a MRF will correct the labels of pixels such that the new labeling is similar to the old one but much more smooth. However, as mentioned before direct optimization is very hard, the methods by Kolmogorov and Zabin (2004) and Boykov and Kolmogorov (2004) can be used to fulfill this task. Their algorithms have been implemented in C++ code (GCMex, n.d.). In this thesis two functions in their code are used to improve the classification from chapter 3 and 4: a graph constructor and $\alpha\beta$-swap function. These functions need a data energy and a smoothness energy to initialize the process. In the following more details are given about these two energy matrices.

5.5.1 Data energy

The data energy is defined on each pixel with respect to a given label. Generally for an image with n pixels and m labels, there are $n \times m$ values to define the data energy, which can be described in a $n \times m$ matrix. The i-th row means the i-th pixel, the j-th column means the j-th label. The values at the i-th row in the matrix means how much a label is probable for the i-th pixel. Since the max-flow algorithm searches the minimum of the energy, a large value in the matrix means a small probability and vice versa.

Because pixels of same class have similar intensity values, the pdf value can be used to measure how similar the pixel intensities are. Given an initial classification, the parameters of each class can be estimated from samples of each class. With these parameters pdf value for every pixel can be calculated. Because the max-flow algorithm tries to find the minimum, a minus is added in the data energy. The following equation is used to set the data energy:

$$D_p(l_p) = -\ln\left(pdf\left(x_p; \theta_j(l_p)\right)\right). \tag{5.10}$$

where x_p is the intensity value of pixel p, θ_j are parameters for the j-th class and depends on the labeling. In this way pixels belong to the j-th class will have the lowest energy values at the j-th column, so these pixels will be marked with the j-th label according to the min-energy criterion.

5.5.2 Smoothness energy

The smoothness energy measures the spatial proximity of pixels. The adjacent pixels should have continuous intensities and probably have same label. If this does not happen, a penalty will be given to this pair. The following simple equation is used for smoothness energy:

$$V_{pq}(l_p, l_q) = \delta(l_p, l_q) = \begin{cases} 1 & \text{if} \quad l_p \neq l_q \\ 0 & \text{if} \quad l_p = l_q. \end{cases} \tag{5.11}$$

Because the max-flow/min-cut optimization will try to find the minimal energy, it favors pair of pixels with the same label (no penalty). In this way the smoothness energy will try to smooth the local variation of the labeling.

In the literature the smoothness energy will always weighted instead of being used directly. This will enable more finer control of the result. If the initial segmentation should be more respected, a small weight value should be used, otherwise a larger one is preferred and more labels will be changed in this case. In experiments the magnitude of the data energy is much larger than the smoothness one. In order to keep the weight factor in the range $[0, 1]$, the data energy is modified as:

$$D_p(l_p) = -\frac{\ln\left(pdf\left(x_p; \theta_j(l_p)\right)\right)}{20}. \tag{5.12}$$

and the empirical weight factor used in this thesis is about 0.2.

5.5.3 Prior probability

In practice a single SAR image is about 700 Mb or even larger, this is too large to be processed at once. In order to avoid memory problem a SAR image is partitioned into small tilings and processed respectively, then merged together into a final image. This tiling and merging process should in principle not influence the final result. According to experiments, this is true for most SAR images. However, there is still a problem if a tile contains areas with similar intensity value. In fact these areas belong to one class. But the process still tries to segment them into three classes. The consequence is that a large area is relabeled after optimization.

The idea to solve this problem is to add the original proportion of each class into the data energy to restrict the effect of optimization. For example, if there is only 5% of pixels in the image labeled as $\mathcal{C}_j$, this number in the final result after the graph-cut algorithm should not grow too much. This 5% proportion will reduce the probability of relabeling even if the pixels have similar intensity values and are spatial connected. In statistics this proportion of the j-th class can be interpreted as the *prior probability of the j-th class* , the pdf used in data energy is the *likelihood function of the j-th class*. So the product of these two is proportional to the *a posterior probability of the j-th class*. Similar to smoothness energy another weight factor is added

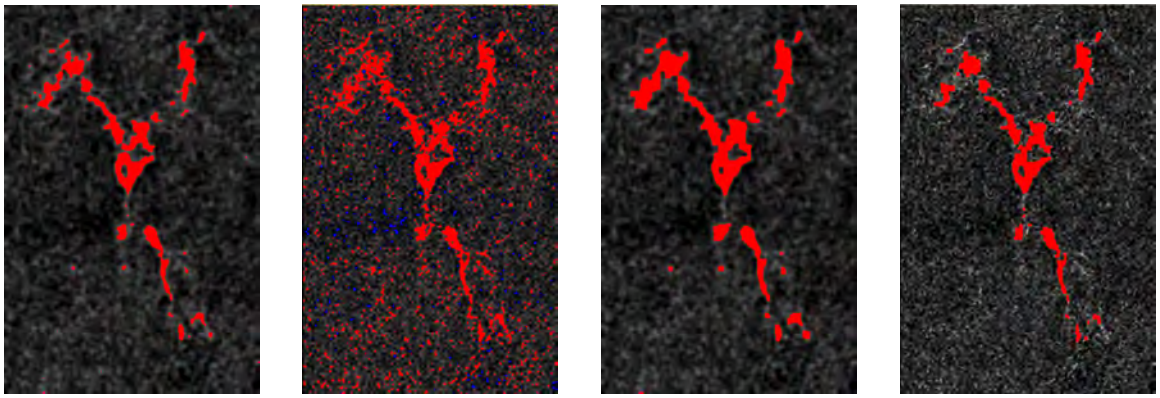
to balance the effect of likelihood function and prior probability, the final data energy can be updated as:

$$D_p(l_p) = -\frac{\ln\left(pdf(x_p;\theta_j)\right) + w_2 \ln\left(Pr(\mathcal{C}_j)\right)}{20}. \tag{5.13}$$

where w_2 is the weight factor between likelihood function and prior probability. $Pr(\mathcal{C}_j)$ is the prior probability of the j-th class. In this thesis w_2 is set equal to 0.2.

5.6 Application to test data

The classification results in chapter 3 and 4 are used by the graph-cut algorithm in this chapter. The test data are the same as shown in figure 3.2 and 3.6.



(a) Histogram thresholding.

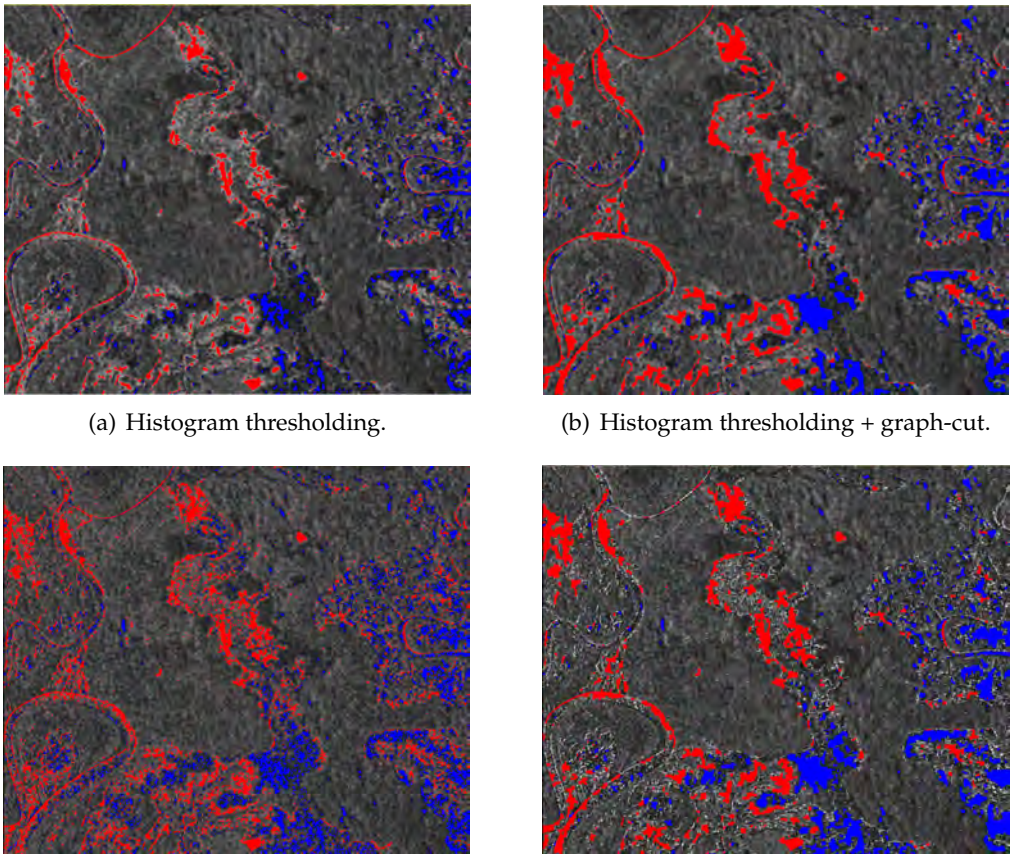
(b) Hypothesis test.

(c) Histogram thresholding and graph-cut.

(d) Hypothesis test and graph-cut.

Figure 5.6: *Classification results for the first test data on Queensland, Australia, using different methods. The classification masks are overlapped with the SAR ratio images. The red color illustrates the negative change, it represents the detected flood water areas. Figure 5.6(a) and 5.6(b) are classification results through a single method, they are post-processed by the graph-cut algorithm described in this chapter and the results are displayed in figure 5.6(c) and 5.6(d)*

In figure 5.6 the classification results from histogram thresholding, hypothesis test and graph-cut are compared. As mentioned before the hypothesis test can generally find more detected signals than histogram thresholding, but at the same time more mistakes in the classification result, whereas the generalized KI histogram thresholding makes less mistakes but the classification result is very conservative. This is verified by comparing figure 5.6(a) and 5.6(b). The optimized results by the graph-cut algorithm in this chapter are displayed in figure 5.6(c) and 5.6(d). Compared with figure 5.6(a) the result in figure 5.6(c) adds more useful signals at the boundary of water areas as detected change. The result in figure 5.6(d) also removes falsely labeled pixels in figure 5.6(b). General speaking, graph-cut algorithm adds plausible signals to the result from the histogram thresholding but removes error obtained from the hypothesis test. The similar effect can be seen from figure 5.7. The graph-cut algorithm can generally reduce the classification error in hypothesis test and expand the detected area in histogram thresholding. The extend of this effect depends on the parameters in the energy function, which can be modified according to different applications.



(a) Histogram thresholding.

(b) Histogram thresholding + graph-cut.

(c) Hypothesis test.

(d) Hypothesis test + graph-cut.

Figure 5.7: *Classification results for the second test data on Queensland, Australia using different methods. The classification masks are overlapped with the SAR ratio images. The red color illustrates the negative change, it represents the detected flooded water. The blue color indicates the positive change, it represents the receding water. Figure 5.7(a) and 5.7(c) are classification results through a single method, they are post-processed by the graph-cut algorithm described in this chapter and the results are displayed in figure 5.7(b) and 5.7(d)*

Chapter 6

Validation and discussion

In this chapter the methods discussed in this thesis are applied to a test area near Leipzig, Germany. This area is observed twice by TerraSAR-X: The first SAR image was recorded on 17 January, 2011 during a flood event, another one was recorded on 08 February, 2011 after the flood event. A co-registered RGB aerial image recorded on 17 January, 2011 at the resolution of 0.5 m is used to generate the validation data. A subset of this aerial image is shown in figure 6.1(d). This RGB aerial image was then processed by the ISODATA algorithm in the software ERDAS Imagine 2011 and segmented into many homogeneous areas, which are later manually identified as water area or not. Because it is time consuming to extract water areas from the whole optical aerial image, only a small area is digitalized in a binary water mask to provide a quantitative analysis. The result is a binary water mask shown in figure 6.2(b), which can be used as a validation image. In figure 6.1 a subset area of the test data near Leipzig is displayed, figure 6.1(a) and 6.1(b) are amplitude data recorded during and after the flood event, 6.1(c) is the intensity ratio image, 6.1(d) is the RGB aerial image recorded during the flood event.

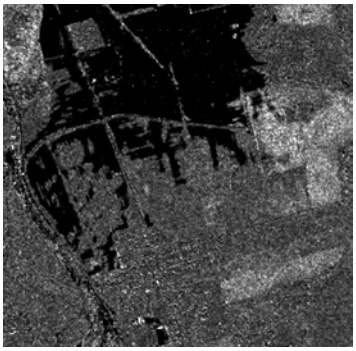
In figure 6.2(a) the classification result from the hypothesis test and graph-cut is shown, In figure 6.2(b) a binary reference water mask is displayed along with the classification result. Here the sequence of the SAR images is changed, the ratio image is calculated as the quotient between the first and the second SAR images. In the classification mask, no-change class is set as transparent, positive class is marked in blue, negative change is in red. Histogram thresholding and hypothesis test are also individually applied to this test area and the result are compared with the reference mask. In table 6.1 three number are given to describe the quality of a classification method. The meaning of each column is described as follows:

- The column "water":
$$\frac{\text{correctly classified water}}{\text{area of the test image}},$$
- The column "non-water":
$$\frac{\text{correctly classified non-water}}{\text{area of the test image}},$$
- The column "error": the rate of falsely classified area.

Table 6.1 demonstrates that all methods provide good results in the accuracy about 8%. The true water proportion calculated from the binary reference mask is about 70%, so the detected water proportion is about 4% less than the true proportion. The difference of classification between the SAR and the aerial image is mostly located on the road and street areas. Because SAR is a squinting sensor, the trees on the both sides of the road can disturb the observation

(a) Amplitude data on 17 January, 2011, ©DLR 2013.

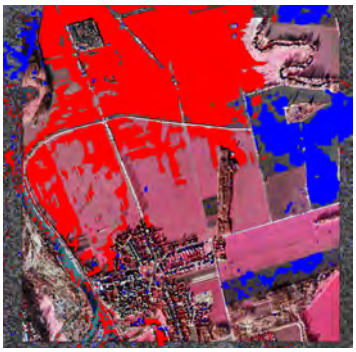
(b) Amplitude data on 08 February, 2011, ©DLR 2013.



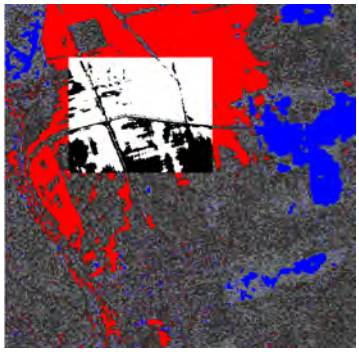
(c) Intensity ratio image between figure 6.1(a) and 6.1(b), ©DLR 2013.

(d) Aerial image, ©DLR 2013.

Figure 6.1: *The first subset of the test area near Leipzig, Germany. Figure 6.1(a) and 6.1(b) are amplitude data recorded during and after the flood event, 6.1(c) is the intensity ratio image, 6.1(d) is the RGB aerial image recorded during the flood event.*



(a) Classification



(b) Reference water mask

Figure 6.2: *Classification result and reference water mask for the first subset of the test data near Leipzig, Germany. In figure 6.2(a) the classification mask is overlapped with the aerial image. Red color indicates the negative-changed flooded water, blue color represents the positive-changed receding water, no-changed areas are transparent. In figure 6.2(b) the reference water mask is overlapped with the classification mask and the SAR ratio image. White pixels in the reference mask are flood water, black pixels are non-water.*

by SAR sensor. In figure 6.1(b) many road areas (thin lines) are displayed as dark pixels while no flood occurs. These areas are marked as no-changed area in figure 6.2(a). This difference is mostly not attributed to the algorithms but the measuring principle of the SAR sensor.

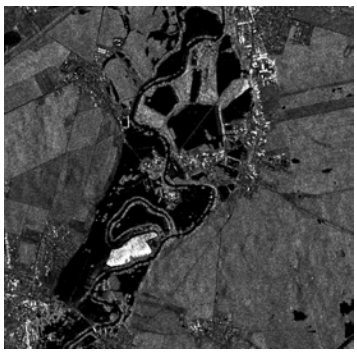
Method	**Water [%]**	**Non-water [%]**	**Error [%]**
Histogram thresholding	65.02	26.72	8.26
Hypothesis test	66.48	25.51	8.01
+ Graph-cut	66.73	25.53	7.74

Table 6.1: *Quantitative comparison of classification results with reference data. The column "water" represents the proportion of correctly detected water area, the column "non-water" represents the proportion of correctly detected non-water area, the column "error" represents the proportion of incorrectly detected area. The true proportion of water area in the reference water mask is* 70%.

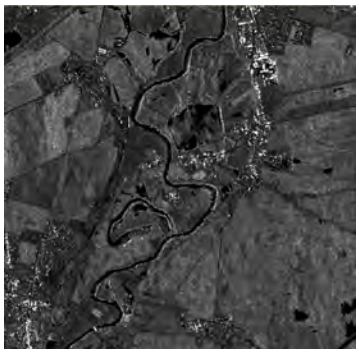
It is interesting to observe three classes within one test area in figure 6.2(a). Usually the areas marked in red are interpreted as flood water, the areas in blue are interpreted as receding water. But all areas in figure 6.1(b) seem to be non-water, so the blue color should not be interpreted as receding water. This positive change could be caused by the double/multiple bounce reflect. If the flood water is not above the total vegetation, the water surface and lower sections of vegetation form a corner reflector and result in a strong signal return (Richards et al., 1987).

A better example of this phenomenon is given in figure 6.3. The difference in the amplitude data can be recognized easily through the ratio image in figure 6.3(c). But it seems strange to observe a positive-changed area near a negative-changed area at the same time. This could also be explained by double/multiple bounce effect. These areas are clearly detected and displayed in figure 6.3(d) using hypothesis test and graph-cut algorithm.

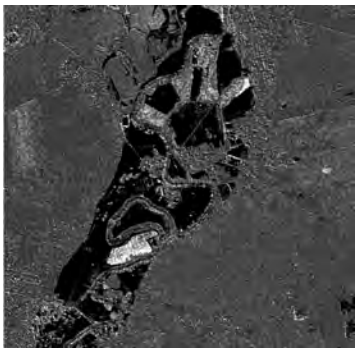
Another subset of the test area with three classes is displayed in figure 6.4. The ratio image is illustrated in figure 6.4(a), the classification results obtained from hypothesis test and graph-cut algorithm are displayed in figure 6.4(b) and 6.4(c). It seems that the result in figure 6.4(c) is improved compared with that in figure 6.4(b). But according to Google maps the areas marked in red are actually permanent water, so they should be classified in no-changed area. This phenomenon could be caused by different weather conditions during two observations. Wind velocity could influence the roughness of water surface and change the backscattered signal. This is another case that the positive change should be interpreted differently instead of receding water.



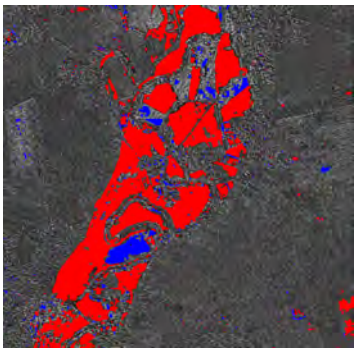
(a) Amplitude image on 17 January, 2011, ©DLR 2013.



(b) Amplitude image on 08 February, 2011, ©DLR 2013.

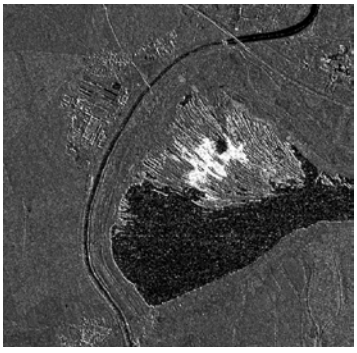


(c) Intensity ratio image between figure 6.3(a) and 6.3(b)

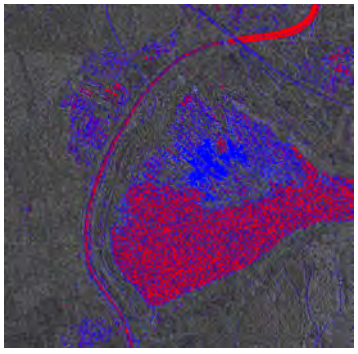


(d) Classification image

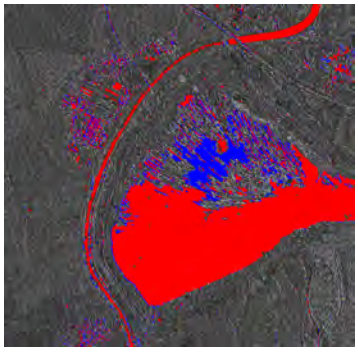
Figure 6.3: *The second subset of the test area near Leipzig, Germany, and the classification mask. Figure 6.3(b) and 6.3(a) are amplitude data recorded before and after the flood event, 6.3(c) is the intensity ratio image, 6.3(d) is the classification mask overlapped with the SAR ratio image. Red color indicates the negative change, blue color represents the positive change, no-changed areas are transparent.*



(a) Intensity ratio image



(b) Classification from hypothesis test



(c) Classification from hypothesis test and graph-cut

Figure 6.4: *The third subset of the test area near Leipzig, Germany, and the classification mask. Figure 6.4(a) is the intensity ratio image, 6.4(b) and 6.4(c) are the classification masks overlapped with the SAR ratio image. Figure 6.4(b) is obtained from hypothesis test algorithm, 6.4(c) is obtained from hypothesis test and graph-cut algorithms. Red color in the classification masks indicates the negative change, blue color represents the positive change, no-changed areas are transparent.*

Chapter 7

Conclusion and outlook

The objective of this thesis is to find changes from two co-registered calibrated TerraSAR-X images. The designed algorithm should be able to distinguish three different classes: positive-, negative- and no-changed classes. Three methods are used to accomplish the task. The first method, histogram thresholding, uses the histogram of the SAR intensity ratio image to classify the ratio image into three classes. This technique was originally proposed by Kittler and Illingworth (1986) and modified by Bazi et al. (2005) and Moser et al. (2006) based on the Bayesian formula. A new parameter estimation method MoLC was also used by Moser and Serpico (2006). In this thesis the methods by Bazi et al. (2005) and Moser et al. (2006) are combined together to detect three classes. In order to determine the number of the classes Bazi et al. (2005) used the determinant of the Hessian matrix, this method is replaced by the relative difference of the cost function. The second method formulates the classification problem as a hypothesis testing problem. This idea was originally used by Touzi et al. (1988) and Oliver (1996). In this thesis the analytical method by Touzi et al. (1988) is replaced by using the properties of the Gamma distribution, at the end a similar test variable can be found to formulate the hypothesis and perform classification. The third method, graph-cut algorithm, is a post-processing method, which improves classification results from the first and second methods. This problem is equivalent to the global optimization of an energy function in a MRF. A modern method proposed by Kolmogorov et al. (2004) and Boykov et al. (2004) is used in this thesis. Their method transforms the energy function of a MRF into an equivalent graph and solves the global optimization problem using a max-flow/min-cut algorithm. These three methods are applied to the test data of Queensland, Australia, and Leipzig, Germany. According to the experiments the histogram thresholding is a conservative method, which find less changed areas but the rate of false classification is also smaller. The hypothesis testing is a optimistic method, it can find more changed areas but also make more false decisions. These two methods can be improved by the graph-cut algorithm. Using these three methods most SAR ratio images can be classified into three classes successfully.

The remaining problem is that the interpretation of the changes is still ambiguous. For a flood event the positive change is usually interpreted as receding water, the negative change is usually interpreted as flood water. However in some cases their meanings could be different. According to the examples in this thesis positive change could be caused by double/multiple bounce effect or different wind velocities above the water surface. It is hard to determine which interpretation is correct without other information. Although this limitation SAR images are still preferred in disaster monitoring because SAR observations are nearly weather independent. The methods in this thesis provides independent information about the event on the ground. They can be used as a primary search, after which other data sources can be combined to assist or improve the interpretation of the detected change. For example a local reference

water mask can be used to filtered out the permanent water from the detected changed areas, in this way the change caused by different wind velocities could almost be excluded in the final interpretation. In this thesis only calibrated intensity SAR data are used, it is also worthwhile using other SAR data such as phase or polarization data to assist identification of detected change.

Bibliography

Astrium (2002), http://www2.astrium-geo.com/files/pmedia/public/r466_9_tx-gs-dd-3302_basic-product-specification-document_v1.7.pdf. [Online; accessed 08-Aug-2013].

Astrium (n.d.*a*), http://www.astrium-geo.com/terrasar-x/. [Online; accessed 08-Aug-2013].

Astrium (n.d.*b*), http://www2.astrium-geo.com/files/pmedia/public/r465_9_tsxx-itd-tn-0049-radiometric_calculations_i1.00.pdf. [Online; accessed 08-Aug-2013].

Bazi, Y., Bruzzone, L. and Melgani, F. (2005), 'An unsupervised approach based on the generalized gaussian model to automatic change detection in multitemporal sar images', *Geoscience and Remote Sensing, IEEE Transactions on* **43**(4), 874–887.

Benning, W. (2011), *Statistik in Geodäsie, Geoinformation und Bauwesen: mit Ausgleichsprogramm auf CD-ROM*, Wichmann.
URL: *http://books.google.de/books?id=-MC2uQAACAAJ*

Besag, J. (1986), 'On the statistical analysis of dirty pictures', *Journal of the Royal Statistical Society. Series B (Methodological)* pp. 259–302.

Bishop, C. M. and Nasrabadi, N. M. (2006), *Pattern recognition and machine learning*, Vol. 1, springer New York.

Boykov, Y. and Kolmogorov, V. (2004), 'An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **26**(9), 1124–1137.

DLR (2009), http://www.dlr.de/eo/en/desktopdefault.aspx/tabid-5725/9296_read-15979/. [Online; accessed 08-Aug-2013].

Ford, L. R. and Fulkerson, D. R. (1956), 'Maximal flow through a network', *Canadian Journal of Mathematics* **8**(3), 399–404.

Kittler, J. and Illingworth, J. (1986), 'Minimum error thresholding', *Pattern recognition* **19**(1), 41–47.

Kolmogorov, V. and Zabin, R. (2004), 'What energy functions can be minimized via graph cuts?', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **26**(2), 147–159.

Krylov, V., Moser, G., Serpico, S. B., Zerubia, J. et al. (2013), 'On the method of logarithmic cumulants for parametric probability density function estimation', *IEEE Transactions on Image Processing* .

Li, S. Z. (1995), *Markov random field modeling in computer vision*, Springer-Verlag New York, Inc.

Lindgren, B. (1993), *Statistical Theory*, Chapman and Hall/CRC Texts in Statistical Science Series, Chapman & Hall.
URL: *http://books.google.co.uk/books?id=2_HPq1ru9nwC*

Lu, D., Mausel, P., Brondizio, E. and Moran, E. (2004), 'Change detection techniques', *International journal of remote sensing* **25**(12), 2365–2401.

Moser, G. and Serpico, S. B. (2006), 'Generalized minimum-error thresholding for unsupervised change detection from sar amplitude imagery', *Geoscience and Remote Sensing, IEEE Transactions on* **44**(10), 2972–2982.

Nicolas, J.-M. (2002), 'Introduction to second kind statistics: application of log-moments and log-cumulants to sar image law analysis', *Traitement du Signal* **19**(3), 139–168.

Oliver, M. (1996), *Understanding disability: From theory to practice.*, St Martin's Press.

Radke, R. J., Andra, S., Al-Kofahi, O. and Roysam, B. (2005), 'Image change detection algorithms: a systematic survey', *Image Processing, IEEE Transactions on* **14**(3), 294–307.

Richards, J., Woodgate, P. and Skidmore, A. (1987), 'An explanation of enhanced radar backscattering from flooded forests', *International Journal of Remote Sensing* **8**(7), 1093–1100.

Sezgin, M. et al. (2004), 'Survey over image thresholding techniques and quantitative performance evaluation', *Journal of Electronic imaging* **13**(1), 146–168.

Springer, M. (1979), *The algebra of random variables*, Probability and Statistics Series, Wiley.
URL: *http://books.google.de/books?id=qUDvAAAAMAAJ*

Touzi, R., Lopes, A. and Bousquet, P. (1988), 'A statistical and geometrical edge detector for sar images', *Geoscience and Remote Sensing, IEEE Transactions on* **26**(6), 764–773.

Appendix A

Proofs

A.1 Ratio of Weibull

According to the equation (3.25) the distribution function for the ratio of two Weibull distributed random variables is:

$$\begin{aligned} P_R(r) &= \int_0^\infty y \frac{\eta_1 (ry)^{\eta_1 - 1}}{b_1^{\eta_1}} \frac{\eta_2 y^{\eta_2 - 1}}{b_2^{\eta_2}} \exp\left(-\left(\frac{ry}{b_1}\right)^{\eta_1}\right) \exp\left(-\left(\frac{y}{b_2}\right)^{\eta_2}\right) dy \\ &= \frac{\eta^2 r^{\eta-1}}{(b_1 b_2)^\eta} \int_0^\infty y^{2\eta-1} \exp\left[y^\eta \left(-\left(\frac{r}{b_1}\right)^\eta - \left(\frac{1}{b_2}\right)^\eta\right)\right] dy. \end{aligned} \tag{A.1}$$

Using change of variable $\rho = y^\eta \left(\left(\frac{r}{b_1}\right)^\eta + \left(\frac{1}{b_2}\right)^\eta\right)$ the equation is then:

$$P_R(r) = \frac{\eta r^{\eta-1}}{(b_1 b_2)^\eta A^2} \int_0^\infty \rho \exp(-\rho) d\rho, \tag{A.2}$$

where $A = \left(\left(\frac{r}{b_1}\right)^\eta + \left(\frac{1}{b_2}\right)^\eta\right)$. At the end the ratio distribution is:

$$P_R(r) = \eta \lambda^\eta \frac{r^{\eta-1}}{(\lambda^\eta + r^\eta)^2}, \tag{A.3}$$

where $\lambda = b_1/b_2$.

A.2 Fundamental property of Mellin transformation

Fundamental property of Mellin transform:

$$\mathcal{M}[f \hat{*} g] = \mathcal{M}[f] \cdot \mathcal{M}[g]. \tag{A.4}$$

Proof: let $h = f \hat{*} g$, according to the definition of Mellin transform in the equation 3.28

$$\text{LHS} = \int_0^\infty u^{s-1} h(u) du = \int_0^\infty u^{s-1} \left[\int_0^\infty f(y) g\left(\frac{u}{y}\right) \frac{dy}{y}\right] du. \tag{A.5}$$

Rewriting $u^{s-1} = y^{s-1}\left(\frac{u}{y}\right)^{s-1}$ and distributing them to dy and $\frac{du}{y}$ respectively:

$$\text{LHS} = \int_0^\infty \left[\int_0^\infty \left(\frac{u}{y}\right)^{s-1} \cdot g\left(\frac{u}{y}\right) d\left(\frac{u}{y}\right)\right] y^{s-1} f(y) dy. \tag{A.6}$$

Rewriting the term is bracket with the following transformation of variable $z = u/y$ then:

$$\begin{aligned} \text{LHS} &= \int_0^\infty \left[\int_0^\infty z^{s-1} g(z)\, dz\right] y^{s-1} f(y) dy \\ &= \int_0^\infty \mathcal{M}[g](s) \cdot y^{s-1} f(y) dy \\ &= \mathcal{M}[g](s) \cdot \int_0^\infty y^{s-1} f(y) dy \\ &= \mathcal{M}[g](s) \cdot \mathcal{M}[f](s) = \text{RHS}. \qquad \text{Q.E.D.} \end{aligned} \tag{A.7}$$

A.3 Product property

For random variables X_i for $i = 1, 2$ with corresponding pdf $f_{X_i}(x_i)$. Their product is $X_3 = X_1 \cdot X_2$ with pdf $f_{X_3}(x_3)$, which can be obtained by Mellin transformation of $f_{X_1}(x_1)$ and $f_{X_2}(x_2)$.

Proof: $X_3 = X_1 \cdot X_2$ implies $\ln(X_3) = \ln(X_1) + \ln(X_2)$. As known the pdf of the sum of two random variables is the convolution of pdfs of these two random variables. Let $f_{\ln(X_i)}(\ln(x_i))$ be the pdf of $\ln(X_i)$, it can be shown:

$$\begin{aligned} f_{\ln(X_3)}(\ln(x)) &= [f_{\ln(X_1)} * f_{\ln(X_2)}](\ln(x)) \\ &= \int_0^\infty f_{\ln(X_1)}(\ln(y)) \cdot f_{\ln(X_2)}(\ln(x) - \ln(y)) d\ln(y) \\ &= \int_0^\infty f_{\ln(X_1)}(\ln(y)) \cdot f_{\ln(X_2)}\left(\ln\left(\frac{x}{y}\right)\right) d\ln(y). \end{aligned} \tag{A.8}$$

At the same time consider the relation of pdfs for X_i and $\ln(X_i)$:

$$f_{\ln(X_i)}(\ln(x)) = x f_{X_i}(x), \tag{A.9}$$

which is obtained just by transforming the variable $\ln(x)$ to x. The terms in the equation (A.8) can be replaced by following equations:

$$f_{\ln(X_1)}(\ln(y)) = y f_{X_1}(y). \tag{A.10}$$

$$f_{\ln(X_2)}\left(\ln\left(\frac{x}{y}\right)\right) = \frac{x}{y} f_{X_2}\left(\frac{x}{y}\right). \tag{A.11}$$

$$f_{\ln(X_3)}(\ln(x)) = x f_{X_3}(x). \tag{A.12}$$

Inserting these equations into the equation (A.8) gets:

$$\begin{aligned} \text{LHS} &= x f_{X_3}(x). \\ \text{RHS} &= \int_0^\infty y f_{X_1}(y) \cdot \frac{x}{y} f_{X_2}\left(\frac{x}{y}\right) \frac{1}{y} dy \\ &= x \int_0^\infty f_{X_1}(y) \cdot f_{X_1}\left(\frac{x}{y}\right) \frac{dy}{y}. \end{aligned} \tag{A.13}$$

Both sides of the equation (A.8) must be equal, this implies:

$$f_{X_3}(x) = \int_0^\infty f_{X_1}(y) \cdot f_{X_1}\left(\frac{x}{y}\right)\frac{dy}{y} = [f_{X_1} * f_{X_2}](x). \qquad \text{Q.E.D.} \tag{A.14}$$

A.4 Inversion property

For a random variable X with pdf $p_X(x)$, the pdf of random variable $Y = 1/X$ is then $p_Y(y) = x^2 p_X(x)$. The Mellin transform of $p_Y(y)$ is

$$\mathcal{M}[p_Y(y)](s) = \int_0^\infty y^{s-1} p_Y(y) dy. \tag{A.15}$$

Inserting $p_Y(y) = x^2 p_X(x)$, $dy = -\frac{1}{x^2}dx$ and $y = x^{-1}$ into the equation gets

$$\mathcal{M}[p_Y(y)](s) = -\int_\infty^0 x^{1-s} x^2 p_X(x)\frac{1}{x^2}dx = \int_0^\infty x^{(2-s)-1} p_X(x)dx = \mathcal{M}[p_X(x)](2-s). \tag{A.16}$$

At the end:

$$\mathcal{M}[p_Y(y)](s) = \mathcal{M}[p_X(x)](2-s). \qquad \text{Q.E.D.} \tag{A.17}$$

Appendix B

IDL and C++ codes

B.1 Preprocessing

The following codes are for preprocessing of the SAR images. This step is implemented in the function "prepare.pro", which includes two utility functions. The first utility function "checksz.pro" checks the size of two co-registered images and makes them equal. The second utility function "db2pow.pro" transforms the dB images to intensity images.

Function: prepare.pro

```
pro prepare, filepath1, filepath2, image1 = image1, image2 = image2,
$ ratio = ratio, powerimage = powerimage


  ; purpose: test if the sizes of two images are same
  ;
  ; filepaht1: local path of the first image data
  ; filepath2: local path of the second image data
  ;
  ; image1: variable to save transformed data of the first image
  ; image2: variable to save transformed data of the second image
  ; ratio: variable to save transformed data of the ratio image
  ;
  ; ##################################################################
  ; powerimage: set "powerimage = 1" to TURN OFF transformation from
   dB to power, if the power/intensity images are used.


  if N_ELEMENTS(powerimage) eq 0 then powerimage = 0

  ;test if the file could be read or not
  query = QUERY_IMAGE(filepath1, info)
  query = QUERY_IMAGE(filepath2, info)*query
  if query ne 1 then $
     MESSAGE, 'images were not read in correctly !! '

  ;read image files
  image1 = read_tiff(filepath1)
  image2 = read_tiff(filepath2)
```

```
  ;check size
  checksz, image1, image2

  ; use keyword: powerimage
  if powerimage eq 0 then begin
    scale = 100
    ; transform from db to power value
    image1 = db2pow((TEMPORARY(image1)), scale)
    image2 = db2pow((TEMPORARY(image2)), scale)
  endif

  ratio = image1/image2

end
```

Function: checksz.pro

```
pro checksz, image1, image2

; check the sizes of two input images and force them equal.

sz1 = size(image1,/DIMENSIONS)
sz2 = size(image2,/DIMENSIONS)

sz = min([[sz1],[sz2]], DIMENSION=2)
image1 = (TEMPORARY(image1))[0:sz[0]-1, 0:sz[1]-1]
image2 = (TEMPORARY(image2))[0:sz[0]-1, 0:sz[1]-1]

end
```

Function: db2pow.pro

```
function db2pow, image, scale

; transfer the dB-image file to power-/intensity-image, a scale is added to
; amplify the amplitude.

  im = (10^(image/10.0))*scale
  id = where(im lt 1e-12)
  if id[0] ne -1 then im[id] = 1e-12
  return, im

end
```

B.2 Histogram thresholding

The following IDL codes are for histogram thresholding based on GKI algorithm. This step is implemented in the function "change_detection1.pro", which includes 3 different

ratio pdfs: log-normal ratio, Weibull ratio and Gamma ratio. The equations for parameter estimation based on MoLC are implemented in the corresponding fitting functions. Function "bisection.pro" is implemented to solve the nonlinear equations in the parameter estimation. "digamma.pro" and "trigamma.pro" are implemented using series expansion.

change_detection1.pro

```
function change_detection1, ratio,  display = display, model = model
   common gmr, m2
   Seconds = SYSTIME( 1)
   ; histogram thresholding for three-class problem based on the
     methods by Bazi and Moser
   ;
   ; image1: power data of the first image
   ; image2: power data of the second image
   ;
   ; display: set this keyword to show result on screen
   ; model: choose a statistic model for SAR intensity data

   if N_ELEMENTS(display) eq 0 then display = 0

   logratio = alog(ratio)
   sz = size(logratio, /DIMENSIONS)

   ; thumb rule
   k = 300
   ;   k = ceil(sqrt(PRODUCT(sz)))
   ;   k = alog(PRODUCT(sz))/alog(2.0) + 1

   h = HISTOGRAM(logratio, NBINS=k, /L64, LOCATIONS= loc, REVERSE_INDICES=r)

   binsz = (loc(k-1)-loc(0))/(k-1)
   h = h/total(binsz*h)

   ; display the histogram on screen
   if display eq 1 then begin
     window, !D.WINDOW+1,/Free, $
               xsize= 700, ysize= 700, XPOS = 0, YPOS = 0, $
               TITLE= 'histogram of log-ratio'
     plot, h
   endif

   ; ------------------------------------------------------------------
   ; initialize cost function with inf. (because min of cost is desired)
   cost = fltarr(k,k) + !values.F_INFINITY

   ; two thresholds loop, each class cover at least 4 bins.
   for i = 5, k-15 do begin
      for j = i+5, k-1 do begin

         ; histogram segmentation --> 3 classes
         ; ------------------------------------------------------
```

```
; histogram is divided in 3 classes
h1 = h[0:i-1]
h2 = h[i:j-1]
h3 = h[j:k-1]

; to avoid error in parameter estimation if there are
; only few samples --> just neglect them
if total(h1) lt 0.001 or total(h3) lt 0.001 then continue

; locations of each classes in histogram --> 3 parts
loc1 = loc[0:i-1]
loc2 = loc[i:j-1]
loc3 = loc[j:k-1]
case (model) of
    0: begin ; gamma ratio
       ; parameter estimation from samples in each class
       prm1 = gammaratio_fit(h1, loc1)
       prm2 = gammaratio_fit(h2, loc2)
       prm3 = gammaratio_fit(h3, loc3)

       ; calc pdf in each segmented histogram part
       logp1 = gammaratio_pdf(loc1, prm1)
       logp2 = gammaratio_pdf(loc2, prm2)
       logp3 = gammaratio_pdf(loc3, prm3)
       end
    1: begin ; weibull ratio
       prm1 = weibullratio_fit(h1, loc1)
       prm2 = weibullratio_fit(h2, loc2)
       prm3 = weibullratio_fit(h3, loc3)

       ; calc pdf in each segmented histogram part
       logp1 = weibullratio_pdf(loc1, prm1)
       logp2 = weibullratio_pdf(loc2, prm2)
       logp3 = weibullratio_pdf(loc3, prm3)
       end
    2: begin ; log-gaussian ratio
       ; parameter estimation from samples in each class
       prm1 = lognormalratio_fit(h1, loc1)
       prm2 = lognormalratio_fit(h2, loc2)
       prm3 = lognormalratio_fit(h3, loc3)

       ; calc pdf in each segmented histogram part
       logp1 = lognormalratio_pdf(loc1, prm1)
       logp2 = lognormalratio_pdf(loc2, prm2)
       logp3 = lognormalratio_pdf(loc3, prm3)
       end
    else: MESSAGE, 'cannnot find the correct model !!'
  endcase

; prior probability
pr1 = total(h[0:i-1])
pr2 = total(h[i:j-1])
pr3 = total(h[j:k-1])
```

```
      ; cost function
      cost[i,j]  =  pr1*alog(pr1) + pr2*alog(pr2) + pr3*alog(pr3)
      cost[i,j] +=  (total(logp1*h[0:i-1]) + $
                    total(logp2*h[i:j-1]) + total(logp3*h[j:k-1]))
      cost[i,j]  = -TEMPORARY(cost[i,j])
   endfor
endfor

; find the minimum of cost and its position
tt = min(cost, min_subscript)
th = ARRAY_INDICES([k,k], min_subscript, /DIMENSIONS)
th1 = th[0]
th2 = th[1]

; ---------------------------------------------------------------------
opt_th = [th1, th2]

; Bazi[04] used Hessian matrix, which is numeric not so good.
; replace it by the profile of cost function in x- and y-axis.
;
; ----> relative difference
;
; 1. df1: difference of the cost function
;         between max and min in the x-axis
; 2. df2: difference of the cost function
;         between max and min in the y-axis
df2 = abs((max(cost[th1,*],/NAN)-min(cost[th1,*],/NAN))$
      /min(cost[th1,*],/NAN))
df1 = abs((max(cost[*,th2],/NAN)-min(cost[*,th2],/NAN))$
      /min(cost[*,th2],/NAN))

; delete threshld if the cost funtion along the profile is constant
; the relative difference must be larger than 0.01 = 1%
if df1 le 0.01 then opt_th[0] = 0
if df2 le 0.01 then opt_th[1] = k-1

; save the result in a matrix
result = bytarr(sz)
id1 = where(logratio lt loc[opt_th[0]])
id2 = where(logratio gt loc[opt_th[1]])
; three classes
if id1[0] ne -1 then result[id1] = 1
if id2[0] ne -1 then result[id2] = 2
; two classes relabeling, test which label is background
if opt_th[0] eq 0 or opt_th[1] eq k-1 then begin
  if N_ELEMENTS(id1) gt PRODUCT(sz)/2.0 then $
   result = 1-TEMPORARY(result)
  if N_ELEMENTS(id2) gt PRODUCT(sz)/2.0 then $
   result = 2-TEMPORARY(result)
endif
Seconds = SYSTIME( 1)- Seconds
; visualize the data
;
; 1. 3D plot of the cost function
```

```
   ; 2. optimal solution marked by red square.
   plot_cost = SURFACE(cost, WINDOW_TITLE = " 3d plot of cost function ")
   plot_cost = plot3d([th1,th1],[th2,th2],[cost[th1,th2],cost[th1,th2]], $
               "r34s", /SYM_FILLED, /OVERPLOT)
   return, result
end
```

gammaratio_pdf.pro

```
function gammaratio_pdf, z, prm

   ; input: log of power ratio
   ; output: log of pdf

   if N_ELEMENTS(log) eq 0 then log = 0
   q = prm[0]
   L = prm[1]

   ; return log of pdf !! instead of pdf
   return, LNGAMMA(2*L) - 2*LNGAMMA(L) + $
            L*alog(q)+ z*L - 2*L*alog(q+exp(z))

end
```

gammaratio_fit.pro

```
function gammaratio_fit, hist, loc
  common gmr, m2


  ; Purpose: estimate the parameters in gammaratio model with MoLC method
  ;
  ; hist: data of histogram
  ; loc: location of the histogram

  ; centered moments of log-ratio
  m1 = total(hist*loc)/total(hist)
  m2 = total(hist*(loc-m1)^2)/total(hist)

  q = exp(m1)
  L = bisection('molc_gammaratio', 1e-6, 1e12, 2000, 1e-7)

  if abs(molc_gammaratio(L))/L gt 0.01 then $
  MESSAGE,'equation solution not accurate enough !!'

  return, [q, L]
end
```

lognormalratio_pdf.pro

```
function lognormalratio_pdf, z, prm

   m = prm[0]
   V = prm[1]

   return,  -0.5*alog(2*!PI*V)-(z-m)^2/2.0/V

end
```

lognormalratio_fit.pro

```
function lognormalratio_fit, hist, loc
  ; Purpose: estimate the parameters in lognormalratio model with
  ; MoLC method
  ;
  ; hist: data of histogram
  ; loc: location of the histogram

  ; centered moments of log-ratio
  m1 = total(hist*loc)/total(hist)
  m2 = total(hist*(loc-m1)^2)/total(hist)

  return, [m1, m2]
end
```

weibullratio_pdf.pro

```
function weibullratio_pdf, z, prm

   ; input: log of power ratio
   ; output: log of pdf

   lambda = prm[0]
   eta = prm[1]

   ; return log of pdf !!
   return,  alog(eta) + eta*alog(lambda) + z*eta - $
            2.0*alog(lambda^eta + exp(z*eta))

end
```

weibullratio_fit.pro

```
function weibullratio_fit, hist, loc
  common gmr, m2

  ; Purpose: estimate the parameters in weibullratio model with MoLC
  ; method
```

```
  ; hist: data of histogram
  ; loc: location of the histogram

  ; centered moments of log-ratio
  m1 = total(hist*loc)/total(hist)
  m2 = total(hist*(loc-m1)^2)/total(hist)

  lambda = exp(m1)
  eta = sqrt(2*trigamma(1)/m2)

  return, [lambda, eta]
end
```

digamma.pro

```
function digamma, x

  x = double(abs(x))

  psi2 = -2.40411380631918857079947632302289998152997258468099776358454
  psi4 = -24.886266123440878231952771674968820033369942068045907487380 6
  psi6 = -726.01147971498443532465423589185366691190176360697186862044 3
  psi8 = -40400.97839874763488532782365545085427877962792824191466403 94
  psi10 = -3.63059331160662871299061884283205410545727903407938742 98*10^6
  em = 0.57721566490153286060651209008240243104215933593992359880576 7

  if x lt 5 and x gt 0.1 then $
     return, digamma(x+1)-1.0/x

  if x le 0.1 then $
     return, -1.0/x -em + !PI^2*x/6.0 + x^2*psi2/2 + !PI^4*x^3/90 + $
             x^4*psi4/24 + !PI^6*x^5/945 + x^6*psi6/720 + $
             !PI^8*x^7/9450 + x^8*psi8/40320 + !PI^10*x^9/93555 $
  else $
     return, -alog(1.0/x) - 1.0/2.0/x - 1.0/12.0/x^2 + 1.0/120.0/x^4 -$
             1.0/252.0/x^6 + 1.0/240.0/x^8 - 1.0/132.0/x^10 + $
             691.0/32760.0/x^12


end
```

trigamma.pro

```
function trigamma, x

  x = double(abs(x))

  psi2 = -2.40411380631918857079947632302289998152997258468099776358454
  psi4 = -24.886266123440878231952771674968820033369942068045907487380 6
  psi6 = -726.01147971498443532465423589185366691190176360697186862044 3
  psi8 = -40400.97839874763488532782365545085427877962792824191466403 94
```

```
  psi10 = -3.6305933116066287129906188428320541054572790340793874298*10^6

  if x lt 5 and x gt 0.1 then $
     return, trigamma(x+1)+1.0/x^2

  if x le 0.1 then $
     return, 1.0/x^2 + !PI^2/6.0 + x*psi2 + !PI^4*x^2/30 + x^3*psi4/6 + $
             !PI^6*x^4/189 + x^5*psi6/120 + !PI^8*x^6/1350 + $
             x^7*psi8/5040 + !PI^10*x^8/10395 + x^9*psi10/362880 $
  else $
     return, 1.0/x + 1.0/2.0/x^2 + 1.0/6.0/x^3 - 1.0/30.0/x^5 + $
             1.0/42.0/x^7 - 1.0/30.0/x^9 + 5.0/66.0/x^11


end
```

bisection.pro

```
function bisection, myfun, a, b, Nmax, Tol

  a = double(a)
  b = double(b)
  ; check a < b
  if a gt b then begin
    c = a
    a = b
    b = c
  endif

  N = 1
  while  N le Nmax do begin
     ; midpoint c
     c = (a+b)/2
     if CALL_FUNCTION(myfun, c) eq 0 or (b-a)/2 lt Tol $
     then return, c

     N = N+1
     if CALL_FUNCTION(myfun, c)*CALL_FUNCTION(myfun, a) gt 0 $
     then a = c else b = c
  ENDWHILE

  print, 'converge at:'
  print,  N

  return, c
end
```

B.3 Hypothesis testing

The following IDL codes are for change detection based on hypothesis testing. This step is implemented in the function "change_detection2.pro", which includes a core function "hyptest.pro". The quantiles are determined in the functions "funtl.pro" and "funtu.pro".

change_detection2.pro

```
function change_detection2, image1, image2, $
                            look1 = look1, look2 = look2, $
                            display = display

  common numberlooks, nlook1, nlook2, alpha

  ; input: image of power
  ; keyword look1, look2: number of multilooking of two images
  ; keyword display: set display = 1 to show result. default = 0


  ; ---------------------------------------------------------------
  ; copy of global number of multilooking
  nlook1_copy = nlook1
  nlook2_copy = nlook2

  if n_elements(display) eq 0 then display = 0
  if N_ELEMENTS(look1) ne 0 then nlook1 = look1
  if N_ELEMENTS(look2) ne 0 then nlook2 = look2

  ; ---------------------------------------------------------
  ; pixel based hypothesis test
  ; ---------------------------------------------------------
  result = hyptest(image1, image2, /PIXEL)
  nlook1 = nlook1_copy
  nlook2 = nlook2_copy

  ; ---------------------------------------------------------
  ; relabeling: change the order of labels
  ; ---------------------------------------------------------
  ;  relabel, result, image1/image2
  ;
  ; ---------------------------------------------------------
  ; display the result
  ; ---------------------------------------------------------
  window_name = 'pixel based hypothesis test'
  colotable = 23
  if display ne 0 then $
     disp, result, window_name = window_name, colotable

  return, result

end
```

hyptest.pro

```
function hyptest, image1, image2, pixel = pixel, area = area, mask = mask

; hypothesis test on pixels (htpix) based on incomplete beta distribution
; ref. oliver.

; ---------------------------------------------------------------
common numberlooks, nlook1, nlook2, alpha

checksz, image1,image2
sz = size(image1,/DIMENSIONS)

if N_ELEMENTS(pixel) eq 0 then pixel = 0
if N_ELEMENTS(area) eq 0 then area = 0
if area + pixel ne 1 then MESSAGE, $
'one of pixel or area keyword can/must be set !!'

if (area) ne 0 then begin
   if N_ELEMENTS(mask) eq 0 then MESSAGE, $
   ' set mask when using area based test !! '
   if PRODUCT(size(mask,/DIMENSIONS) eq sz) ne 1 then $
   MESSAGE, 'dimension of mask doesnt match to images !!'
endif

; ---------------------------------------------------------------
; save global number of multilooking
nlook1_copy = nlook1
nlook2_copy = nlook2

; save result
result = bytarr(sz)


if N_ELEMENTS(mask) ne 0 then begin
   vclass = mask[UNIQ(mask, SORT(mask))]
   nclass = N_ELEMENTS(vclass)

   ; hypothesis test2: region based
   for i = 0, nclass-1 do begin
     idi =  (mask eq vclass[i])
     ; label blob of regions
     b = LABEL_REGION(idi, /ULONG)
     ; get population and members of each blob
     h = HISTOGRAM(b, REVERSE_INDICES=r)

     ; each region
     for j = 1,N_ELEMENTS(h)-1 do begin
        ; linear indices
        p = r[r[j]:r[j+1]-1]
        ; area ratio
        m1 = mean(image1[p])
        m2 = mean(image2[p])
        ratiop = m1/m2
```

```
        ; number of multilooking for area
        nlook1 *= h[j]
        nlook2 *= h[j]

        ; solving equation by bisection
        Tu = bisection('funtu', 0.0, 1.0, 200, 1e-7)
        Tl = bisection('funtl', 0.0, 1.0, 200, 1e-7)

        ; check the accuracy
        if abs(funtu(Tu)) gt 1e-4 or abs(funtl(Tl)) gt 1e-4  $
        then message, 'euqation solution is not accurate enough'

        Tu = (1.0/Tu -1)*nlook1/nlook2
        Tl = (1.0/Tl -1)*nlook1/nlook2

        if (ratiop*Tu lt 1) then result[p] = 1
        if (ratiop*Tl gt 1) then result[p] = 2

        nlook1 = nlook1_copy
        nlook2 = nlook2_copy
     endfor
   endfor
endif else begin
   ; solving equation by bisection
   Tu = bisection('funtu', 0.0, 1.0, 200, 1e-7)
   Tl = bisection('funtl', 0.0, 1.0, 200, 1e-7)

   ; check the accuracy
   if abs(funtu(Tu)) gt 1e-6 or abs(funtl(Tl)) gt 1e-6  $
   then message, 'euqation solution is not accurate enough'

   Tu = (1.0/Tu -1)*nlook1/nlook2
   Tl = (1.0/Tl -1)*nlook1/nlook2

   ratio = image1/image2
   id1 = where(ratio*Tu lt 1)
   if id1[0] ne -1 then result[id1] = 1
   id2 = where(ratio*Tl gt 1)
   if id2[0] ne -1 then result[id2] = 2
endelse

return, result
end
```

funtl.pro

```
function funtl, x

; function for Tl

; global variables reference
```

```
common numberlooks, nlook1, nlook2, alpha

; ibeta() is only defined on interval [0, 1]
x = x<1.0

; ibeta(nlook1, nlook2, x) = 1 - alpha
return, ibeta(nlook1, nlook2, x) - 1 + alpha

end
```

funtu.pro

```
function funtu, x

; function for Tu

; global variables reference
common numberlooks, nlook1, nlook2, alpha

; ibeta() is only defined on interval [0, 1]
x = x<1.0

; ibeta(nlook1, nlook2, x) = alpha
return, ibeta(nlook1, nlook2, x) - alpha

end
```

B.4 Graph-cut optimization

The following IDL codes are for post-processing based on graph-cut optimization. This step is implemented in the function "change_detection3.pro", which includes a core function "clearify.pro". A C++ function is implemented to invoke the graph-cut C++ open codes (GCMex, n.d.) in IDL. Function "relabel.pro" is designed to change the number of each class. Function "disp.pro" is designed to demonstrate the final classification result on the screen.

change_detection3.pro

```
function change_detection3, ratio, initial_mask, weight, weight2,$
                            display = display

  common numberlooks, nlook1, nlook2, alpha

  ; input: image of power
  ; keyword display: set display = 1 to show result. default = 0

  if N_ELEMENTS(display) eq 0 then display = 0
  sz = size(ratio, /DIMENSIONS)
  result = bytarr(sz)
```

```
  ; ------------------------------
  tiledim = 1000
  xtiles = sz[0]/tiledim
  ytiles = sz[1]/tiledim
  print, STRJOIN(['total tilings: = ', $
                  string((xtiles)), '*', string((xtiles))])
  ; image tiling
  for n = 0, xtiles do begin
     idx1 = n*tiledim
     idx2 = min([(n+1)*tiledim, sz[0]])
     for m = 0, ytiles do begin
        PRINT, 'xtiles = '
        print, n
        PRINT, 'ytiles = '
        print, m

        idy1 = m*tiledim
        idy2 = min([(m+1)*tiledim, sz[1]])

        result[idx1:idx2-1, idy1:idy2-1] = clearify( $
          ratio[idx1:idx2-1, idy1:idy2-1], weight, weight2, $
          ini_class_img = initial_mask[idx1:idx2-1, idy1:idy2-1], $
        model = 0, DISPLAY = display)
      endfor
  endfor
  if display ne 0 then begin
    disp, result, 23
  endif
  return, result

end
```

clearify.pro

```
function clearify, data, weight, weight2,$
                                ini_class_img = ini_class_img, $
                                nclass = nclass, $
                                model = model, $
                                display = display

  ; + initialization:
  ;   - this procedure uses "INI_CLASS_IMG" as an inital segmentation
  ;   - otherwise "NCLASS" should be set, then K-means will be used for
  ;   initialization.
  ;
  ; + model:
  ;   - 0: default for Gamma ratio distribution
  ;   - 1: for lognormal ratio distribution
  ;   - 2: for weibull ratio distribution

  ; global variable, set in workflow
  common path, root, sub, function_name, dllname
```

```
; ------------------------------------------------------------------------
; general initialization:
; ------------------------------------------------------------------------
data_img = alog(data)
sz = size(data_img, /DIMENSIONS)

if N_ELEMENTS(ini_class_img) eq 0 and N_ELEMENTS(nclass) eq 0 $
then MESSAGE, 'initial classification or number of classes is required !!'

if N_ELEMENTS(ini_class_img) ne 0 then begin

   ; check dimension
   if total( sz eq size( ini_class_img, /DIMENSIONS ) ) ne 2 $
   then MESSAGE, 'dimension of images dont match !!'

   ; initialize current classification image
   class = ini_class_img

   ; set class number
   vclass = class[UNIQ(class, SORT(class))]
   nclass = N_ELEMENTS(vclass)

endif else begin
   ; use Kmean to initialize the procedure
   ; ### cautious !!!
   ; error by direct kmean, should transform to y = alog(x)
   seed = kmeanpp_seed( (data_img), nclass )
   class = kmeans( reform((data_img), 1, sz[0]*sz[1]), seed, $
                   !values.F_INFINITY, 100)
   class = reform( class, sz)
   relabel, class, data_img

   ; set class number
   vclass = class[UNIQ(class, SORT(class))]
   nclass = N_ELEMENTS(vclass)
endelse

; dll for graph-cut
; copy the dll data to the ROOT directory
full_dll_path = filepath(dllname,ROOT_DIR=root)

; set display, default display = 0, means DON'T show intermediate
; result on screen.
if N_ELEMENTS(display) eq 0 then display = 0
if N_ELEMENTS(model) eq 0 then model = 0
if model ne 0 and model ne 1 and model ne 2 then $
MESSAGE, 'model should be among 0, 1, 2 !!'

; ------------------------------------------------------------------------
; initialization for graph-cut
; ------------------------------------------------------------------------
; change rate of each iteration, convergence test
rate = 1
```

```
; minimum of global energy, convergence test
minEn = 1e8
; pointer to current classification, returned by graph-cut
labeling = ptr_new(make_array(sz[0]*sz[1], value = 9L))
; data- / smooothness-energy
e1 = ptr_new(0.0)
e2 = ptr_new(0.0)
; intialize pointer for each energy term
DataCost = ptr_new(make_array(sz[0]*sz[1]*nclass, value = 1.0))
SmoothnessCost = ptr_new(make_array( nclass*nclass, value = 1.0))
dataEn = fltarr(nclass, sz[0]*sz[1])
smoothnessEn = fltarr(nclass, nclass)
smoothnessEn = weight*((fltarr(nclass, nclass) + 1.0) - $
                  DIAG_MATRIX(intarr(nclass, 1)+1))

; ---------------------------------------------------------
; optimization by graph-cut
; ---------------------------------------------------------
for i = 0,30 do begin

   ; ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   ; update data-/smoothness-cost
   ; ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   ; update current number of classes
   vclass = class[UNIQ(class, SORT(class))]
   nclass = N_ELEMENTS(vclass)

   ; update smoothness energy / cost according to nclass
   smoothnessEn = smoothnessEn[[vclass], *]
   smoothnessEn = smoothnessEn[*,[vclass]]
   *SmoothnessCost = reform(smoothnessEn, nclass*nclass, 1)
   dataEn = fltarr(nclass, sz[0]*sz[1])

   ; ..............................................................
   ; number of bins in the histogram
   ;     k = (sqrt(PRODUCT(sz)) > 300)
   k = 300
   ; ###############
   ; set data energy
   ; ###############
   for j = 0, nclass-1 do begin
      id = where( class eq vclass[j],count )
      if id[0] eq -1 then begin
         continue
      endif else begin
         case (model) of
           0: begin
              h = HISTOGRAM(data_img[id], NBINS=k, /L64, LOCATIONS= loc,$
              REVERSE_INDICES=r)
              binsz = (loc(k-1)-loc(0))/(k-1)
              h = h/total(binsz*h)
              prm =  gammaratio_fit(h, loc)
              logp = gammaratio_pdf(data_img, prm)+$
              weight2*alog(count/product(sz))
```

```
            end
          1: begin
             h = HISTOGRAM(data_img[id], NBINS=k, /L64, LOCATIONS= loc, $
             REVERSE_INDICES=r)
             binsz = (loc(k-1)-loc(0))/(k-1)
             h = h/total(binsz*h)
             prm =  weibullratio_fit(h, loc)
             logp = weibullratio_pdf(data_img, prm)+$
             weight2*alog(count/product(sz))
             end
          2: begin
             h = HISTOGRAM(data_img[id], NBINS=k, /L64, LOCATIONS= loc,$
             REVERSE_INDICES=r)
             binsz = (loc(k-1)-loc(0))/(k-1)
             h = h/total(binsz*h)
             prm =  lognormalratio_fit(h, loc)
             logp = lognormalratio_pdf(data_img, prm)+$
             weight2*alog(count/product(sz))
             end
          else: MESSAGE, 'dont find model !!'
        endcase

        ;*count/width/height
        idd = where( logp lt -12)
        if idd[0] ne -1 then logp[idd] = -12
        dataEn[j,*] = reform(-(logp)/20.0, 1, sz[0]*sz[1])
    endelse
endfor
; ..............................................................

*DataCost = reform(dataEn, sz[0]*sz[1]*nclass, 1)

; ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
; process optimization
; ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
tt = call_external( full_dll_path, function_name, sz[0], sz[1],$
nclass, *DataCost, *SmoothnessCost, *e1, *e2, *labeling, $
VALUE=[1,1,1,0,0,0,0,0], /CDECL, /AUTO_GLUE, /UNLOAD)
if tt ne 0 then MESSAGE, 'Input Array has wrong datatype! ' + $
                                    'Type double * expected!'

if (*e1 + *e2) lt minEn and rate gt 0.02 then begin
   rate = (minEn- *e1 - *e2)/minEn
   minEn = (*e1 + *e2)
   class = reform(*labeling, sz[0], sz[1])

   ; show intermediate result on screen
   if display eq 1 then begin
     disp, class, $
     window_name = $
     strjoin(['this is:',string(i+1),'th optimization']), 23
   endif

endif else begin
```

```
        break
     endelse

   endfor

   ptr_free, DataCost
   ptr_free, SmoothnessCost
   ptr_free, e1
   ptr_free, e2
   ptr_free, labeling

   return, class
end
```

disp.pro

```
  pro disp, image, window_name = window_name, colortable

    ; display the result
    ; ---------------------------------------------------------
    sz = size(image,/DIMENSIONS)
    sz = (sz < [1000,1000])

    if N_ELEMENTS(window_name) eq 0 then $
    window_name = ''

    window, !D.WINDOW+1,/Free, $
               xsize= sz[0], ysize= sz[1], XPOS = 0, YPOS = 0, $
               TITLE= window_name
    device, decomposed = 0
    loadct, colortable
    tvscl, ROTATE(image,7)

  end
```

relabel.pro

```
 pro relabel, classIm, dataIm, revert = revert


  ; purpose: change the order of labels in classIm according
  ; to dataIm, default is increasing order.
  ;
  ; revert: revert the order
  ; ---------------------------------------------------------

  vclass = classIm[UNIQ(classIm, SORT(classIm))]
  nclass = N_ELEMENTS(vclass)
     for i = 0, nclass-2 do begin
     for j = i+1, nclass-1 do begin
        id1 = where(classIm eq vclass[i])
```

```
        id2 = where(classIm eq vclass[j])
        q = 0
        if N_ELEMENTS(revert) ne 0 then $
           q = (mean(dataIm(id1)) lt mean(dataIm(id2))) $
        else q = (mean(dataIm(id1)) gt mean(dataIm(id2)))
        if q then begin
           classIm(id1) = vclass[j]
           classIm(id2) = vclass[i]
        endif
     endfor
  endfor

end
```

wenxi_test.cpp

```
// wenxi_test.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include "graph.h"
#include "GCoptimization.h"
#include "idl_export.h" // For CALL_EXTERNAL within the IDL code


__declspec(dllexport) IDL_INT __cdecl graph_cut (IDL_LONG width,
 IDL_LONG height, IDL_LONG num_labels, float* DataCost,
 float* SmoothnessCost, float* e1, float* e2, IDL_LONG* labeling)
{
IDL_LONG num_pixels = width * height;

    /* First call the constructor for a grid graph */
    GCoptimization *optimize = (GCoptimization *) new GCoptimization(width,
    height,num_labels,SET_ALL_AT_ONCE,SET_ALL_AT_ONCE);

// set cost values
optimize -> setData(DataCost);
optimize -> setSmoothness(SmoothnessCost);

    // optimize
    optimize -> swap();

    // copy result in an external array or pointers
    optimize -> ExportLabels( (GCoptimization::LabelType *) labeling );
    *e1 = optimize -> giveDataEnergy();
    *e2 = optimize -> giveSmoothEnergy();

    delete optimize;
return 0;
}
```