Fast automatic deforestation detection

Jesper Muren

Mathematical Statistics
Stockholm University
Master Thesis **2021:10**
http://www.math.su.se

Fast automatic deforestation detection

Jesper Muren[*]

June 2021

Abstract

The aim of this thesis is to create algorithms for automatic deforestation detection. The detection of forests, or the lack thereof, is interesting in many ways. Presumably everyone is aware of the ongoing climate crisis and our forests role in mitigating it. It may also be of interest to land owners managing crop portfolios, cities measuring percentages of park areas in the city or detection of natural disasters such as wildfires. In this thesis we propose two algorithms using high altitude images for this purpose. The two algorithms include one parametric and one non parametric. The parametric model assumes independent stable distributions for the color intensities of the pixels of forest images. It uses the Cramér-von Mises test statistic and obtained 96.7% accuracy when separating forest and non forest in the test set. The non parametric model uses the squared Mahalanobis distance and obtained an accuracy of 98.4% on the test set. The thesis is also accompanied by a small R package including the proposed algorithms.

[*]Postal address: Mathematical Statistics, Stockholm University, SE-106 91, Sweden. E-mail: jespermuren@gmail.com. Supervisor: Dmitry Otryakhin.

Acknowledgements

I would like to thank my supervisor, Dmitry Otryakhin, for introducing me to the topic of this thesis. I also want to thank him for all the hours he spent giving me guidance and support. It would not have been possible without him.

Finally, I would also like to extend my gratitude to my partner Ia for her patience and support throughout this time.

Contents

1 Introduction

Automatic deforestation detection in near real time is interesting in a number of ways. The first thing that spring to mind might be our forests role in the ongoing climate change with their ability to mitigate it by capturing and storing carbon. According to The State of the World's Forests, by the United Nations (2020) [7], the leading cause of deforestation is agricultural expansion, predominantly in Africa and South America. In these regions, there has been a net forest area loss for the last 30 years, unlike most other parts of the world where we are seeing net increases in forest area. Ironically, the reduction of biodiversity that comes with deforestation has a negative impact on the agricultural sector making the current rate of deforestation by the large scale commercial agriculture unsustainable. Further use of change detection in forests could be interesting for land owners or governments in regards to prevention of illegal logging, management of crop portfolios, computing the percentage of park areas in cities, detection of wildfires or other natural disasters as well as just monitoring forest plantations or other green areas.

The problem of deforestation detection lies within the remote sensing domain and the natural way of obtaining data is through satellite imagery. While aerial photos in general might be better and likely would give better results, they are taken too infrequently to make sense for large scale applications. Therefore satellite imagery is the natural way of obtaining repeated images of large areas of land. The satellite imagery intended to be used for deforestation detection by our algorithms will be obtained using Sentinel-2 operated by the European Space Agency, which can effectively provide global coverage on a five-day basis. However, as of writing this thesis, the infrastructure needed to effectively obtain these satellite images is not in place. For this reason, the images used for the results presented in this thesis will be obtained from the satellite and aerial images available through the Google Earth engine.

The Sentinel-2 satellite infrastructure, as well as deforestation detection results using this data, will be a part of the paper being written on this topic. This paper is being written with hopes of journal publishing and in addition to the algorithms developed in this thesis, it will include a bayesian approach.

As of writing this thesis, there are no algorithms for deforestation detection using the type of frequentist inference we aim to investigate, therefore the algorithms proposed in this thesis are originally our ideas. Most literature on the topic seems to be geared towards machine learning approaches such as Ortega et.al (2020) [18], using deep learning for deforestation detection in the Brazilian Amazon and Shermeyer & Haack (2015) [23], using k-nearest neighbour classification to track deforestation in Peru. A Bayesian approach has also seen use in Reiche et. al (2015) [20] on time series data from Fiji. We hope to create a lightweight algorithm that can detect areas covered by forest or not without being too computationally intensive, as machine learning applications such as deep learning often are. Our goal is also to make the algorithm interpretable, which often is an issue with black-box machine learning techniques.

We intend to investigate two different approaches to the problem: a parametric one and a non parametric one. We take a supervised learning approach where we use training data to learn the distribution of the intensity of red, green and blue colors of pixels in two different ways. The idea in general is to split images into much smaller square sub images, then test distribution of these sub images against reference images that we know contains forest. In one case, we assume a parametric distribution for the intensity of red, green and blue colors of pixels and use training data to estimate the parameters of said distribution. New data is then tested against this distribution. This type of anomaly detection has been successfully applied to automatic detection of ships in Wang, Liao & Li (2008) [26]. In the other case, we do not assume any known distribution for the color intensities but instead use a non parametric test to test if the distribution of the color intensities of the training data and new data is equal.

1.1 Aim

The main aim of this thesis is to create and test algorithms used to automatically detect if areas of satellite images are covered by forest or not. We aim to create multiple models and compare their performance, including a non parametric model as well as a model based on the stable distribution. Further, the work done should result in an R package including the proposed algorithms, as well as a paper intended for journal publishing.

1.2 Disposition

The disposition of this thesis from this point on is as follows. In Section 2 we outline theory of techniques and concepts used in the thesis. In Section 3, we describe the data used in modeling and testing as well as explore its characteristics. In Section 4, we give the outline of the algorithms as well as the training of the models. Section 5 includes information about the R package accompanying the thesis with example code snippets and output, a link to the package is available through [33]. After this, we move on to Section 6, where we present the numerical results obtained when applying the models to the test set. These results, special performance considerations and potential improvements are then discussed in Section 7. Lastly, Section 8 gives the finishing conclusions and takeaways of the thesis.

2 Theory

This section will include the theory of techniques and concepts used in this thesis. This thesis is written for the reader with at least the equivalent of a Master's degree in mathematical statistics. Hence theory of concept that should be familiar to such a reader is omitted.

2.1 Stable distribution

Unless otherwise specified, the following theory follows that of Samorodnitsky and Taqqu (1994) [22].

There are several equivalent definitions of a stable distribution, this section will outline two of them where one is based on a generalized central limit theorem and one the characteristic function. Proofs of these will be omitted but can be found in, for example, Gnedenko and Kolmogorov (1954) [9].

Definition 1. *A random variable X is said to have a stable distribution if there is a sequence of i.i.d random variables $Y_1, Y_2, \ldots$ and sequences of positive numbers $\{d_n\}$ and real numbers $\{a_n\}$, such that*

$$\frac{Y_1 + Y_2 + \cdots + Y_n}{d_n} + a_n \xrightarrow{d} X. \tag{1}$$

So basically, if you sum a sequence of i.i.d random variables and there is a limiting distribution, it must be stable. Further, you can see the resemblance of the above definition with the conventional central limit theorem where the $Y's$ are taken to also have finite variance and X is Gaussian. This means that the normal distribution belongs to the family of stable distributions.

The probability density function of a stable distribution cannot be expressed in closed form except for in special cases, this means that there are no exact likelihood estimation techniques to estimate the parameters of the distribution. Because there is no explicit way of expressing the probability density function in the general case the stable distribution is often expressed with its characteristic function which is why we provide the following definition equivalent to the previous one.

Definition 2. *A random variable X is said to have a stable distribution if there are parameters $0 < \alpha \leq 2$, $\sigma \geq 0$, $-1 \leq \beta \leq 1$ and $\delta \in \mathbb{R}$ such that its characteristic function has the following form:*

$$\phi(t) = E[e^{itX}] = \begin{cases} \exp\Big\{ -\sigma^\alpha |t|^\alpha (1 - i\beta(\text{sign } t)\tan\frac{\pi\alpha}{2}) + i\delta t\Big\} & \text{if } \alpha \neq 1 \\ \exp\Big\{ -\sigma|t|(1 + i\beta\frac{2}{\pi}(\text{sign } t)\ln|t|) + i\delta t\Big\} & \text{if } \alpha = 1. \end{cases} \tag{2}$$

Where sign $t = 0$ if $t = 0$. The parameter α is known as the index of stability and controls the decay of the tails. The parameters β is not the classical statistical skewness but it indicates how the distribution is skewed, indicating left skewness if $\beta < 0$, right skewness if $\beta > 1$ and symmetry if $\beta = 0$. The parameter σ is a scale parameter controlling variability and δ a location parameter which shifts the distribution. The location parameter only corresponds to the mean when $\alpha > 1$, otherwise the mean is undefined. In fact, for $\alpha < 2$ the p-th absolute moment $E[|X|^p] < \infty$ iff $p < \alpha$.

The most famous special cases where the probability density function can be expressed explicitly is when the stable distribution follows a Gaussian, Cauchy or Lévy distribution. A stable distribution is Gaussian when $\alpha = 2$ and (typically) $\beta = 0$. This can be seen by considering the characteristic function, in this case, given by

$$\phi(t) = E[e^{itX}] = \exp\Big\{ -\sigma^2 t^2 + i\delta t \Big\}$$

which is the characteristic function of a Gaussian distributed random variable with mean δ and variance $2\sigma^2$. Further, a stable random variable is Cauchy when $\alpha = 1$ and $\beta = 0$ and Lévy when $\alpha = 1/2$ and $\beta = 1$.

2.1.1 Stable density and distribution function expressions

For this thesis, we will need to compute the density and distribution function of stable distributions for various reasons. An example being that the distribution function will be used for the Cramér-von Mises test described in Section 2.2. This means that even if, in general, there are no explicit expressions for these functions, we need to know how to compute them. This is done numerically using integral formulas and the characteristic function. The theory of this section will follow that of J.P Nolan (1997) [16], where the reader can find the proofs which will be excluded from this thesis due to their very technical nature.

Nolan finds that even if the parametrization of the characteristic function in Equation (2) is the most commonly used one it is numerically favourable to compute the density and distribution functions using another parametrization. This parametrization is a version of Zolotarev's (M)-parametrization, Zolotarev (1986) [28]. This alternate parametrization relates to the one stated in Equation (2) by the location parameter in the following way

$$\delta^* = \begin{cases} \delta + \beta\sigma \tan\frac{\pi\alpha}{2} & \text{if } \alpha \neq 1 \\ \delta & \text{if } \alpha = 1. \end{cases}$$

For the rest of this section, we will denote the alternate location parameter δ^* as δ to keep notation consistent.

It also turns out that numerically it is favourable to compute the density and distribution functions of stable distributions which have been standardized to have location parameter $\delta = 0$ and scale parameter $\sigma = 1$. For a standardized stable random variable the characteristic function, using the alternate parametrization, can be expressed in the following way

$$\phi(t) = E[e^{itX}] = \begin{cases} \exp\Big\{ -|t|^\alpha \big(1 + i\beta(\text{sign } t)\tan\frac{\pi\alpha}{2}(|t|^{1-\alpha} - 1)\big) \Big\} & \text{if } \alpha \neq 1 \\ \exp\Big\{ -|t|(1 + i\beta\frac{2}{\pi}(\text{sign } t)\ln|t|) \Big\} & \text{if } \alpha = 1. \end{cases} \tag{3}$$

The theorem stating the integral formulas for the standardized stable random variables require us to state a number of quantities. These quantities as well as the Theorem itself is not of special interest to us and is quite lengthy, therefore we will include it into Section 9.1 of the Appendix.

In practice, densities and distribution functions are computed by numerically evaluating the integrals stated in the appendix. The proofs involve taking the inverse Fourier transform of the characteristic function and some quite tedious algebra and integral solving for each of the four cases in Theorem 2. Again, these will be omitted but the interested reader is directed to [16] for these proofs as well as numerical considerations when evaluating the integrals.

2.1.2 Parameter estimation

This section will cover the estimation of the four parameters of the univariate stable distribution used in this thesis. There are a number of ways available to estimate these parameters, including quantile-based estimators, Maximum-likelihood-based estimators and moment-based estimators. A good general overview of several estimators can be found in Kharrat Boshnakov (2016) [32]. For this thesis, we have chosen to use the Koutrouvelis regression-type estimator, this estimator is one of the fastest estimator for the parameters with good performance. We tried other estimators, which performed well but were very slow, including ones based on generalized method of moments. Similarly, we tried estimators which were faster than the Koutrouvelis one but these did not perform well enough for our purposes, this includes the McCulloch estimator based on quantiles. A more in-depth study on the performance of different parameter estimators may be of interest to individuals interested in improving the algorithm proposed in this thesis.

The Koutrouvelis regression-type estimator is quite fast, but one big downside is that the estimator of the location parameter δ is often quite unreliable. For our purposes this turns out to not be a big issue though because experimentally we have seen that the parameter $\alpha > 1$ in all our forest data and when this is the case the location parameter δ equals the mean of the distribution,[22]. Which means that the sample mean is a consistent estimator for the location parameter δ for our data. The theory on the Koutrouvelis regression-type estimator of this section will follow that of Koutrouvelis (1980) [14].

The method Koutrouvelis suggested assumes $\alpha \neq 1$, which is exclusively the case for the data used in this thesis so it does not pose a problem. The method is initially based on two observations, firstly the characteristic function $\phi(t)$ taking the form of Equation (2) implies that

$$\log(-\log|\phi(t)|^2) = \log(2\sigma^\alpha) + \alpha \log|t|. \tag{4}$$

The second observation follows from Eulers formula and is that the real and imaginary part of $\phi(t)$ are given by

$$\mathrm{Re}(\phi(t)) = \exp(-\sigma^\alpha|t|^\alpha)\cos\big(\delta t - \sigma^\alpha|t|^\alpha\beta(\mathrm{sign}\ t)\tan\frac{\pi\alpha}{2}\big)$$

and

$$\mathrm{Im}(\phi(t)) = \exp(-\sigma^\alpha|t|^\alpha)\sin\big(\delta t - \sigma^\alpha|t|^\alpha\beta(\mathrm{sign}\ t)\tan\frac{\pi\alpha}{2}\big).$$

From here, we note that Equation (4) only depends on the index of stability parameter α and the scale parameter σ. Koutrouvelis then suggests estimating these parameters by using linear regression with $y = \log(-\log|\phi_n(t)|^2)$ where

$$\phi_n(t) = \frac{1}{n}\sum_{j=1}^{n} \exp(itx_j)$$

is the sample characteristic function for a random sample $x_1, x_2, ..., x_n$ which, by the law of large numbers, is a consistent estimator for $\phi(t)$. With this we can estimate the parameters α and σ of our distribution using the model

$$y_k = \mu + \alpha w_k + \epsilon_k, \quad k = 1, 2, ..., K \tag{5}$$

where $\mu = \log(2\sigma^\alpha)$, $w_k = \log|t_k|$ and ϵ_k is a residual error term which should converge to zero as the sample size grows. Further, $(t_k, k = 1, 2, ..., K)$ is a set of real numbers which needs to be carefully chosen.

When the parameters α and σ are estimated, we move on to consider the real and imaginary part of $\phi(t)$, specifically we consider

$$\arctan\left(\frac{\text{Im}(\phi(t))}{\text{Re}(\phi(t))}\right) = \delta t - \sigma^\alpha |t|^\alpha \beta(\text{sign } t)\tan\frac{\pi\alpha}{2}. \tag{6}$$

The idea is then to fix α and σ to their estimated values and then estimate the remaining parameters β and δ by regressing $z = \arctan\left(\frac{\text{Im}(\phi(u))}{\text{Re}(\phi(u))}\right) + \pi k_n(u)$ on u and $\text{sign}(u)|u|^\alpha$ in the model

$$z_l = \delta u_l - \sigma^\alpha |u_l|^\alpha \beta(\text{sign } u_l)\tan\frac{\pi\alpha}{2} + \eta_l, \quad l = 1, 2, ..., L \tag{7}$$

where η_l again is a residual error term and $(u_k, l = 1, 2, ..., L)$ is an appropriate set of real numbers. Further, $k_n(u)$ is an integer introduced to account for possible nonprincipal branches of the arctan function.

Koutrouvelis found that the estimators of α and σ from Equation (5) depend on the true values of α, β and σ as well as n and the choice of $t_k's$. Similarly, the estimators of β and δ from Equation (7) also depend on the true values of α, β, σ and δ as well as n, the choice of t_k's and u_l's.

The two rounds of regression described above is repeated until satisfactory convergence. For each round a type of standardization, like the one described in Section 2.1.1, of data is performed in order to alleviate some of the dependencies on δ and σ. We consider a sample of data points $x_1, x_2, ..., x_n$, which are standardized in the following way

$$x'_j = \frac{x_j - \delta_0}{\sigma_0}, \quad j = 1, 2, ..., n.$$

Where δ_0 and σ_0 are chosen in advance with some more rudimentary technique. Then $x'_1, x'_2, .., x'_n$ are used as data to get the estimates $\hat{\alpha}_1$ and $\hat{\sigma}_1$ from Equation (5). The estimate for α that would be returned at this stage is $\hat{\alpha}_1$ but since we already standardized by dividing by σ_0 the estimate for σ would be $\sigma_0\hat{\sigma}_1$. We now move on to estimate the parameters β and δ, for this we standardize again by dividing with $\hat{\sigma}_1$ in the following way

$$x_j'' = \frac{x_j'}{\hat{\sigma}_1}, \quad j = 1, 2, ..., n.$$

We then move on to consider the integer $k_n(u_l)$, since this is a function of the u_l's we would have to find this for each of the u_l's. Instead, we opt to find a constant δ_c to subtract from each point x_j'' in order to make sure that the arctan function is continuous on the range of the u_l's. This is done by numerical search starting from $\delta_c = 0$.

Finally, we estimate β and δ using Equation (7) with $\hat{\alpha}_1$ replacing α and the data points $s_j = x_j'' - \delta_c$, $j = 1, 2..., n$. Under the assumption that our estimation of the scale parameter σ is good the x_j'''s should have a scale close to 1 and hence we can ignore σ in Equation (7). From this, we get the estimates $\hat{\beta}_2$ and $\hat{\delta}_2$ where $\hat{\beta} = \hat{\beta}_2$ can be returned as the estimation of β from the original distribution but $\hat{\delta} = \delta_0 + \hat{\sigma}(\hat{\delta}_2 - \delta_c)$.

Since this whole procedure can seem quite involved, a simplified version is described in pseudo-code in Algorithm 1.

Algorithm 1 Koutrouvelis parameter estimation pseudo-code

```
1: **Inputs:** Data points $x_1, x_2, ..., x_n$ and convergence criteria.
2: **while** not converged **do**
3:     **if** first round **then** Estimate $\delta_0$ and $\sigma_0$ with simple technique.
4:     **else** $\delta_0 = \hat{\delta}$, $\sigma_0 = \hat{\sigma}$.
5:     **end if**
6:     Transform data $x_j' = (x_j - \delta_0)/\sigma_0, \quad j = 1, 2, ..., n$.
7:     Estimate $\hat{\alpha}, \hat{\sigma_1}$ using Eq. (5) with $x_j'$'s as data. Set $\hat{\sigma} = \sigma_0\hat{\sigma}_1$.
8:     Transform data $x_j'' = x_j'/\hat{\sigma}_1, \quad j = 1, 2, ..., n$.
9:     Find $\delta_c$ s.t $s_j = x_j'' - \delta_c$ makes LHS of Eq. (7) continous.
10:    Estimate $\hat{\beta}$, $\hat{\delta}_2$ from Eq. (7) with $s_j$'s as data. Set $\hat{\delta} = \delta_0 + \hat{\sigma}(\hat{\delta}_2 - \delta_c)$.
11:    Check convergence
12: **end while** if converged
13: **return** $\hat{\alpha}, \hat{\beta}, \hat{\gamma}$ and $\hat{\delta}$.
```

For more information on things like appropriately choosing the t_k's and u_l's, the dependence of the estimators as well as simulation results the reader is directed to the original paper by Koutrouvelis [14]. The specifics of this is left out because the regression-type technique of Koutrouvelis was later improved in a paper by Kogon & Williams (1998) [13]. They, instead of the standard parametrization of the characteristic function stated in Equation (2), used the Zolotarev (M)-parametrization we previously gave in Section 2.1.1. The characteristic function in this parametrization is continuous for all values of α and β unlike the standard parametrization which is discontinuous for $\alpha = 1$ and $\beta \neq 0$. They also used better initial estimates of δ_0 and σ_0 which were obtained with the McCulloch quantile-based estimator. Further, they choose the interval $[0.1, 1]$

for the frequencies, (t_k's and u_l's) where the sample characteristic function estimates the characteristic function well. These changes lead to the algorithm not needing the iterative approach that Koutrouvelis suggested, which means that it estimates the parameters faster. They also obtain better estimates for values of α and β close to the discontinuity of the standard parametrization. The one downside of the new algorithm is worse performance for α close to 0, it is however unusual to have $\alpha < 0.5$ in practice.

2.2 Empirical distribution function tests

This section will cover the theory of testing whether a sample has been drawn from a specified distribution using cumulative distribution functions (CDF). The theory will follow that of Anderson & Darling (1952) [1] and Anderson & Darling (1954) [2].

For this thesis, we first considered the Anderson-Darling test proposed to test whether a sample of size n with empirical CDF $F_n(x)$ is drawn from a specified distribution with CDF $F(x)$. The test measures the distance between the two distributions in the following way

$$W_n^2 = n \int_{-\infty}^{\infty} [F_n(x) - F(x)]^2 \psi[F(x)] dF \tag{8}$$

where $\psi[F(x)]$ is a weighting function used to emphasize specific regions of the distributions which are of interest to the experimenter. The empirical CDF at any given point t in a sample of n points, $X_1, X_2 \ldots, X_n$ is calculated as

$$F_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{X_i \leq t\}.$$

The classical Anderson-Darling test used the weighting function $\psi[F(x)] = [F(x)(1 - F(x)]^{-1}$, which takes small values when $F(x)$ is close to zero or one, and hence places extra emphasis on difference in tails of $F(x)$ and $F_n(x)$. We realized that this weighting function would not be optimal for our application because large scale images such as the ones used in this thesis will almost unavoidably contain outliers. We noticed that these outliers are often white or black, which would return values in far ends of the support of the distribution of the color intensities. Further, the distributions of our data, the color intensities, are only defined between the values 0 and 1 while this is not the case for the estimated distributions we will test against. This is another reason to give less weight to distributions in the tails. With these realizations, we instead considered the case where weighting function $\psi[F(x)] = 1$ is used, which corresponds to the Cramér-von Mises criterion.

Asymptotic distributions can be found for the statistic W_n^2, see [1], using the values of the statistics which correspond to certain significance levels of the test can be used to compute p-values. However, for our purposes, we chose to work directly with the value of the statistics, this is done for a few reasons. Firstly, as mentioned in [2], these significance points are not available for small sample sizes, which can be the case for us after we split images into smaller sub images for testing. Further, we are not as interested in using the test to answer whether a sample belongs to a specified distribution but instead we are interested in it as a goodness of fit measure, to see how well the empirical distribution of the sample fits to our specified distribution. Finally, we saw better results using the value of the statistic in practice then using p-values, one reason for this is presumably due to truncation of small p-values in the implementation of the test in R. For these reasons, as well as the fact that we value the speed of the algorithm we want to avoid the unnecessary computation of p-values.

2.3 Mahalanobis Distance

Unless otherwise specified, the theory of this section follows that of Varmuza & Filzmoser (2009) [25].

The Mahalanobis distance D was originally introduced as a distance between a given multivariate point $\boldsymbol{p}$ and a distribution. For a point $\boldsymbol{p}$ and distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$; the distance is given by

$$D = \sqrt{(\boldsymbol{p} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{p} - \boldsymbol{\mu})}.$$

Where, in practice, the mean and covariance matrix of the distribution would be estimated by the sample mean and covariance matrix. Further extensions can be made to calculate the distance between two multivariate populations with

$$D = \sqrt{(\bar{\boldsymbol{X}}_1 - \bar{\boldsymbol{X}}_2)^T \hat{\boldsymbol{\Sigma}}^{-1} (\bar{\boldsymbol{X}}_1 - \bar{\boldsymbol{X}}_2)}. \tag{9}$$

Where $\bar{\boldsymbol{X}}_1$ and $\bar{\boldsymbol{X}}_2$ are the sample mean vectors of the two populations. The variable $\hat{\boldsymbol{\Sigma}}^{-1}$ is the inverse of the pooled sample covariance matrix, given by $\hat{\boldsymbol{\Sigma}} = (n_1 \hat{\boldsymbol{\Sigma}}_{\boldsymbol{1}} + n_2 \hat{\boldsymbol{\Sigma}}_{\boldsymbol{2}})/(n-2)$ for samples with respective sample sizes n_1, n_2, total sample size $n = n_1 + n_2$ and sample covariance matrices $\hat{\Sigma}_1$, $\hat{\Sigma}_2$, Mardia, Kent & Bibby (1979) [15]. This case of the Mahalanobis distance between populations will be further explored in Section 2.4

The Mahalanobis distance is often used in cases where there is correlation present in the considered multivariate distribution. As the reader might have noticed the Mahalanobis distance expression is closely related to the statistical concept of standardization, i.e subtracting the mean and dividing by the standard deviation. The idea is to decorrelate and scale each dimensions variance to be 1 and then calculate the euclidean distance. In Figure 1, we can see an example in two dimensions, which illustrates why the Mahalanobis distance is favourable to the Euclidean distance in the case of correlated variables. We see in Figure 1, how the Mahalanobis distance takes into account the correlation between the variables while the euclidean distance does not. For example the Euclidean distance would indicate that a point at $(-2, 2)$ and $(2, 2)$ is equally close to the distribution, which clearly is not the case.

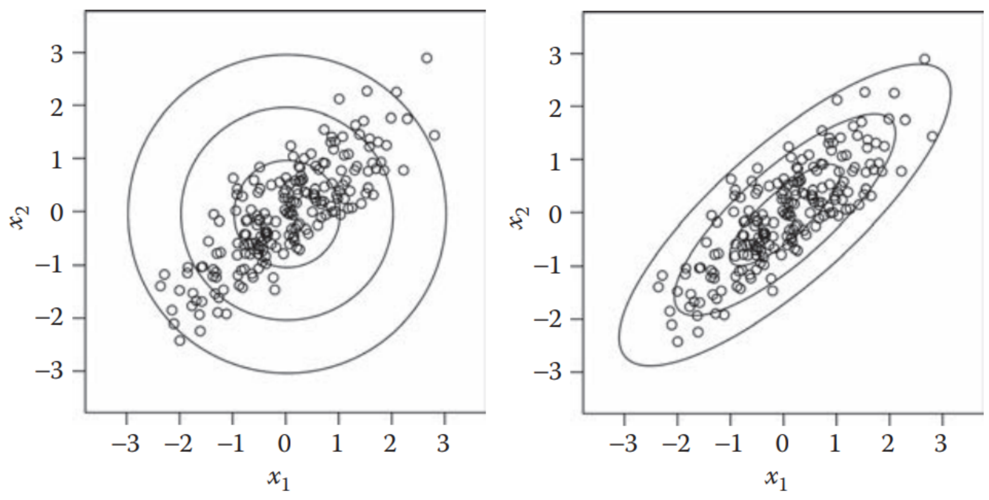


Figure 1: Comparison of Euclidean distance (left) and Mahalanobis distance (right) with distance lines corresponding to distances of 1, 2 and 3 from $(0, 0)$ [25].

2.4 Hotelling T^2 statistic

The theory of this section follows Mardia, Kent & Bibby (1979) [15] unless otherwise specified.

The Hotelling T^2 distribution is often said to be the multivariate version of the well known t-test. It is considered when comparing means of multivariate distributions and is classically defined as follows.

Definition 3. *Let $\boldsymbol{d}$ and $\boldsymbol{M}$ be independently distributed as Normal $N_p(\mathbf{0}, \boldsymbol{I})$ and Wishart $W_p(\boldsymbol{I}, m)$ respectively. We then say that $X = m\boldsymbol{d}'\boldsymbol{M}^{-1}\boldsymbol{d}$ follows a Hotelling T^2 distribution with parameters p and m, denoted $X \sim T^2(p, m)$.*

It can also be shown that properly scaled a T^2 distributed random variable is F distributed, interested readers are directed to Section 3.5 of [15] for more information on this.

Since the Hotelling T^2 distribution would have limited use if it could only be applied in the cases of standard normal and Wishart distributed random variables there are of course extensions. Consider independent $\boldsymbol{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\boldsymbol{M} \sim W_p(\boldsymbol{\Sigma}, m)$. We can show that

$$m(\boldsymbol{X} - \boldsymbol{\mu})'\boldsymbol{M}^{-1}(\boldsymbol{X} - \boldsymbol{\mu}) \sim T^2(p, m)$$

by using a $N_p(\mathbf{0}, \boldsymbol{I})$ distributed random variable $\boldsymbol{d}^* = \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{X} - \boldsymbol{\mu})$ and $W_p(\boldsymbol{I}, m)$ distributed random variable $\boldsymbol{M}^* = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{M}\boldsymbol{\Sigma}^{-1/2}$. The fact that

$\boldsymbol{M}^* \sim W_p(\boldsymbol{I}, m)$ follows from the definition of the Wishart distribution, which unfamiliar readers can see in Section 3.4 of [15]. Using Definition 3, we can then see that

$$m\boldsymbol{d}^{*\prime}\boldsymbol{M}^{*-1}\boldsymbol{d}^* = m(\boldsymbol{X}-\boldsymbol{\mu})'\boldsymbol{M}^{-1}(\boldsymbol{X}-\boldsymbol{\mu}) \sim T^2(p,m).$$

Similarly, if considering a sample of size n from $N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with sample mean vector and covariance matrix $\bar{\boldsymbol{X}}$ and $\hat{\boldsymbol{\Sigma}}$ one can see that

$$(n-1)(\bar{\boldsymbol{X}}-\boldsymbol{\mu})'\hat{\boldsymbol{\Sigma}}^{-1}(\bar{\boldsymbol{X}}-\boldsymbol{\mu}) \sim T^2(p, n-1)$$

by substituting $m = n-1$, $\boldsymbol{M} = n\hat{\boldsymbol{\Sigma}}$ and $(\boldsymbol{X}-\boldsymbol{\mu}) = n^{1/2}(\bar{\boldsymbol{X}}-\boldsymbol{\mu})$.

We can now begin to see the similarity between the Hotelling T^2 statistic and the Mahalanobis distance discussed in the previous section. In fact we will show that the square of the Mahalanobis distance, D^2, between two populations, as seen in Equation (9) follows a Hotelling T^2 distribution under certain conditions when properly scaled. This is called the Hotelling's two-sample T^2 statistic.

Theorem 1. *Let $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$ be independent samples of size n_1 and n_2 from i.i.d $N_p(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2$ and $n = n_1 + n_2$. Then, if $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$, $(n_1 n_2/n)D^2$ is $T^2(p, n-2)$ distributed.*

Proof. The sample means $\bar{\boldsymbol{X}}_i \sim N_p(\boldsymbol{\mu}_i, n_i^{-1}\boldsymbol{\Sigma}_i)$, $i = 1, 2$. This means that $\boldsymbol{d} = \bar{\boldsymbol{X}}_1 - \bar{\boldsymbol{X}}_2 \sim N_p(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}, n_1^{-1}\boldsymbol{\Sigma}_1 + n_2^{-1}\boldsymbol{\Sigma}_2)$. When $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$ this means that $\boldsymbol{d} \sim N_p(\boldsymbol{0}, \frac{n}{n_1 n_2}\boldsymbol{\Sigma})$.

Further, if $\boldsymbol{M}_i = n_i\hat{\boldsymbol{\Sigma}}_i$, then $\boldsymbol{M}_i \sim W_p(\boldsymbol{\Sigma}_i, n_i - 1)$ and if $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$, this means that

$$\boldsymbol{M} = \boldsymbol{M}_1 + \boldsymbol{M}_2 = (n-2)\hat{\boldsymbol{\Sigma}}_p \sim W_p(\boldsymbol{\Sigma}, n-2).$$

Thus $\frac{n}{n_1 n_2}\boldsymbol{M} \sim W_p(\frac{n}{n_1 n_2}\boldsymbol{\Sigma}, n-2)$. Further, $\boldsymbol{M}$ is independent of $\boldsymbol{d}$ since the sample mean and sample variance are independent under the assumption of normality and the two samples are independent of each other. From this we get that

$$(n-2)\boldsymbol{d}'\left(\frac{n}{n_1 n_2}\boldsymbol{M}\right)^{-1}\boldsymbol{d} \sim T^2(p, n-2).$$

From this the proof is finished by seeing that the left hand side is equal to $(n_1 n_2/n)D^2$ by inserting $\boldsymbol{d} = \bar{\boldsymbol{X}}_1 - \bar{\boldsymbol{X}}_2$ and $\boldsymbol{M} = (n-2)\hat{\boldsymbol{\Sigma}}_p$. $\square$

While there is an assumption of underlying multivariate normality here, under certain conditions, the T^2 statistic is asymptotically non parametric, Oja & Randles (2004) [17]. This is is because, asymptotically, the T^2 statistic follows a χ^2 distribution, this can be seen by considering the following scenario:

Let $\boldsymbol{X_1}$ & $\boldsymbol{X_2}$ be samples of size n_1 and n_2, drawn from a p-variate distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The multivariate central limit theorem then tells us that, with increasing sample sizes, the sample mean vectors of these samples, $\bar{\boldsymbol{X}}_1$ & $\bar{\boldsymbol{X}}_2$, tend to multivariate normal distributions with mean vector $\boldsymbol{\mu}$ and covariance matrices $n_1^{-1}\boldsymbol{\Sigma}$ and $n_2^{-1}\boldsymbol{\Sigma}$, respectively. This means that $\bar{\boldsymbol{X}}_1$ - $\bar{\boldsymbol{X}}_2$ is asymptotically $N(\mathbf{0}, (n_1^{-1} + n_2^{-1})\boldsymbol{\Sigma})$. Hence $((n_1^{-1}+n_2^{-1})\hat{\boldsymbol{\Sigma}}))^{-1/2}(\bar{\boldsymbol{X}}_1$ - $\bar{\boldsymbol{X}}_2)$ is asymptotically $N(\mathbf{0}, \boldsymbol{I})$, where $\hat{\boldsymbol{\Sigma}}$ is the pooled sample covariance matrix. Finally, this means that the T^2 statitistic $(\bar{\boldsymbol{X}}_1$ - $\bar{\boldsymbol{X}}_2)'((n_1^{-1}+n_2^{-1})\hat{\boldsymbol{\Sigma}}))^{-1}(\bar{\boldsymbol{X}}_1$ - $\bar{\boldsymbol{X}}_2)$, under these assumptions, is asymptotically $\chi^2(p)$ distributed.

This would allow the use of χ^2 quantiles for testing means of distributions which are not normal. While we will not make use of these quantiles specifically, or p-values at all for that matter— for the same reasons as discussed in Section 2.2—it is worth noting that the assumption made above hold for our data, since it is bounded between 0 and 1 the mean and variance will always be finite.

2.5 Performance analysis

This section will briefly cover the tools and metrics used to analyze the performance of our model. At the core of most of these metrics will be the quantities true positive (tp), true negative (tn), false positive (fp) and false negative (fn), which respectively denote the number of correctly classified as positive and negative, as well as incorrectly classified as positive and negative. For this thesis, we will consider the label "forest" as positive and "not forest" as negative.

2.5.1 Performance metrics

This section will follow the theory of Hossin & Sulaiman (2015) [11] unless otherwise specified.

A natural starting point for visualizing the results of a binary classifiers like the ones we will create is to create a confusion matrix. The confusion matrix uses the quantities tp, tn, fp and fn mentioned above and gives an overview of the performance of the classifier on the test set. Further, performance metrics are then easily calculated from the confusion matrix. A confusion matrix for a binary classifier is a two-by-two matrix with rows representing predicted class and columns representing true labels. This is visualized in Table 1.

Table 1: Visualization of confusion matrix for binary classifier.

Confusion Matrix		
	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True Positive	False Positive
Predicted Negative Class	False Negative	True Negative

While the confusion matrix gives a good overview of how the classifier performed, we need more tangible metrics which can easily be compared between models. The most natural and commonly used one is accuracy, which is just the ratio of correctly classified samples in the entire sample. This is easily computed as well as easily understood by virtually everyone, which is presumably why it is so widely used. We will, however, not limit ourselves to using the accuracy as a performance metric since it also has some drawbacks. The biggest one is probably that it can be very misleading when working with unbalanced data, which is often the case. Imagine a data set consisting of 90% positive samples, then a classifier which only predicts positive would yield an accuracy of 0.9 which at first glance looks very good. To try and give a comprehensive performance analysis, we will also include the metrics error rate, sensitivity, specificity, precision and F-score. These metrics are explained in Table 2.

Table 2: Performance metrics for binary classifier.

Metrics	**Formula**	**Description**
Accuracy (Acc)	$\frac{tp+tn}{tp+fp+fn+tn}$	Fraction of correctly classified in the entire sample.
Error rate (Err)	1−Accuracy	Fraction incorrectly classified in the entire sample.
Sensitivity (Sn)	$\frac{tp}{tp+fn}$	Fraction of positive samples correctly classified.
Specificity(Sp)	$\frac{tn}{fp+tn}$	Fraction of negative samples correctly classified.
Precision(P)	$\frac{tp}{tp+fp}$	Fraction of samples classified as positive correctly.
F-Score (F)	$\frac{2\cdot Sn\cdot P}{Sn+P}$	Performance metric based on both recall and precision.

We can see from Table 2 that with the newly introduced metrics, we can also take into account the overall performance on the positive samples (Sensitivity). The overall performance on the negative samples (Specificity) as well as how good the performance is on specifically positive samples (Precision). The F-score gives us a sort of average between sensitivity and precision by taking the harmonic mean of the two. What we denote F-score is also known as F_1-score, in general, one can consider the F_β-score given by

$$\mathrm{F}_\beta = (1+\beta^2)\frac{Sn \cdot P}{\beta^2 \cdot P + Sn}$$

where one can vary β to give more importance to sensitivity or precision depending on what kinds of performance you want from your classifier.

Finally, we adopt the metric alarm area (AA) [18], which is used when assuming that there will be human double-checking of the classifier when it predicts "not forest". We then want the AA, given by

$$\mathrm{AA} = \frac{tn+fn}{tp+fp+fn+tn},$$

to be as close to the total fraction of ”not forest” samples as possible. This is in order to detect the non-forest areas while not giving too many false alarms.

2.5.2 Receiver operating characteristic curve

This section will cover the concepts of receiver operating characteristic (ROC) curve and the area under the curve (AUC). The theory will follow that of Fawcett (2006) [8].

The ROC curve is another widely used way to visualize the performance of a classifier, as well as a way to choose between different classifiers. A ROC curve is drawn in two-dimensions, with axes corresponding to true positive rate (TRP) and false positive rate (FPR). These are given by

$$\text{TPR} = \frac{tp}{n_p} \quad \text{and} \quad \text{FPR} = \frac{fp}{n_f}$$

where n_p and n_f is the total number of actual positive and negative samples tested in the data set respectively. These quantities are then plotted against each other for varying thresholds. In our case this threshold corresponds to the Mahalanobis distance in the non parametric model and the Cramér-von Mises test statistics in the parametric case, in other models these threshold often correspond to probabilities. In Figure 2, we see an example of a ROC curve with values calculated for seven different thresholds of a model.

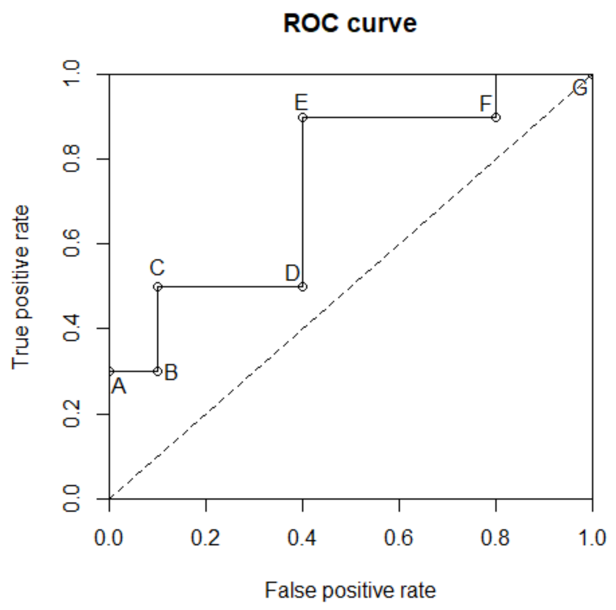


Figure 2: Example ROC curve plotted.

Looking at Figure 2, we first consider the dashed line $y = x$—this line corresponds to the value you would expect to get from random guessing, for example guessing positive 10% of the time would be expected to give 10% true positive and 10% false positives. Any values below the dashed line would hence be worse than random guessing. Classifiers that perform better than random guessing will take values above the dashed line with perfectly scoring classifiers landing on $(1, 0)$. The (uninteresting) cases of always predicting positive or negative yields $(1, 1)$ and $(0, 0)$. Another nice property of the ROC curve is that as long as the threshold is one dimensional it is non-decreasing, since a sample classified as positive will never not be classified as positive when lowering the threshold.

All of this means that drawing a horizontal line to the left and vertical line up from a point yields the area in the ROC space that corresponds to better performance than the given point. This is because any point in this area has either lower FPR but the same TPR, higher TPR but the same FPR or both higher TPR and lower FPR. With this, we can see in Figure 2 that, for example, the point labeled C is better than D. However, a decision between B and D would have to be made while taking into consideration how many false positives to allow in relation to true positives.

A further attractive property of the ROC curves is that they are still effective on unbalanced data since they use the true positive and false positive rate, so they do not depend on the class distribution.

Finally, the ROC curve also offers a good way of comparing different models using the area under the curve (AUC). Since the ROC curve only takes values in the unit square, the AUC takes values between 0 and 1, while you would expect the value to always be above 0.5 since this is the value random guessing yields. The fact that the ROC curve is non-decreasing means that a value of 1 would mean that the classifier could get a perfect TPR with no false positives.

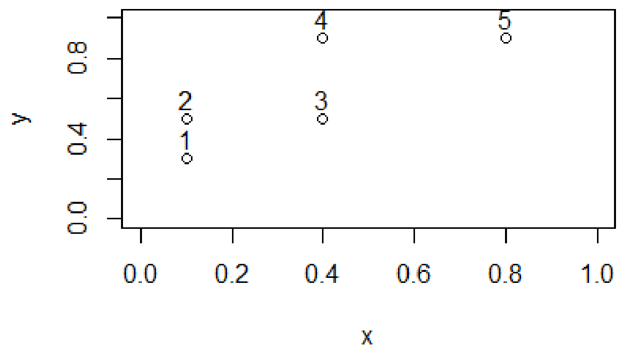
Further, if you consider a classifier that ranks positive samples higher than negative samples, the AUC corresponds to the probability that the model will rank a randomly chosen positive sample higher than randomly chosen negative sample. This means that you would expect a classifier, which yields a higher AUC, to have a better performance on average, leading to a way to choose models using the AUC.

2.6 Hierarchical Clustering

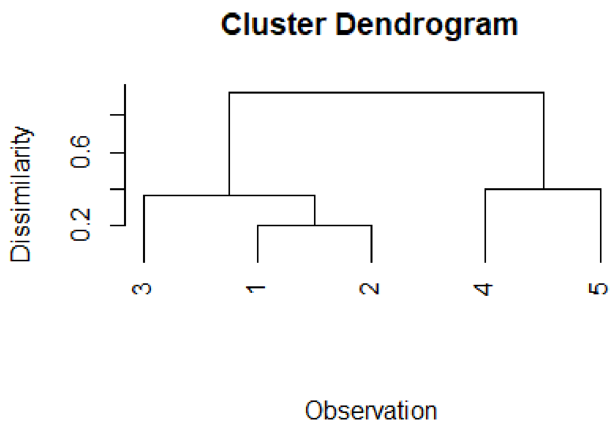
The theory of this section follows that of Section 14.3.12 of Hastie, Tibshirani, Friedman (2001) [10].

Hierarchical clustering creates clusters using a dissimilarity measure, most commonly a distance. Given a set of observations, the dissimilarity measure is calculated between each pair of observations. Observations are then clustered by the magnitude of their dissimilarity measure. At the lowest level of the hierarchy all observations are in their own cluster and at the top level, all observations are in one cluster. Every level of the hierarchy is created by combining the clusters with the pairwise smallest dissimilarity on the level below. This can be done in two ways: bottom-up, starting with one cluster for every observation and combining clusters with small dissimilarity. The other way is top-down, starting with one cluster for all observations and splitting recursively to get the largest dissimilarity. When clusters contain more than one observation, the dissimilarity can be decided in different ways; the most common one is to just average the dissimilarities between the observations of the two clusters.

The easiest way to understand hierarchical clustering is to visualize it with a dendrogram; this also makes it quite easy to interpret. We see an example is Figure 3 using euclidean distance as dissimilarity measure. Here we can see that, for example, observation 1 and 2 are grouped together into a cluster at the lowest level and then they are merged with observation 3. The number of clusters can be chosen either by just deciding the number of clusters you want the observations divided into or by choosing the maximum value of dissimilarity to allow.



(a) Scatter plot of observations



(b) Clustering visualization with dendrogram

Figure 3: Example of hierarchical clustering using euclidean distance as dissimilarity measure.

3 Data

The intention is for the automatic deforestation detection models in this thesis to use data from Sentinel-2. Sentinel-2 is a constellation of two satellites, Sentinel-2A and Sentinel-2B, developed and operated by the European Space Agency. Sentinel-2 provides high-resolution images of the Earth's land surface as often as every five days between the two satellites. This makes Sentinel-2 attractive for near real time applications such as deforestation detection or agricultural monitoring. Sentinel-2 has, for example, seen use in rice crop detection, Campos et al. (2017) [5], and yield forecasting, Pagani et al. (2019) [19].

The infrastructure needed to effectively obtain images from Sentinel-2 was not quite ready as of writing this thesis; for this reason, images from the Google Earth engine is used for the results presented in this thesis instead.

Sentinel-2 provides multispectral imaging in 13 bands, but for our application we will only make use of the three bands corresponding to the colors red, green and blue. These bands are of course also available in the data obtained from Google Earth. Hence, our data will consist of images where for each pixel, we have a color intensity value in $[0, 1]$ for each of the three colors red, green and blue.

For training and validating, we will be using a data set consisting of 45 images, where 22 are of forests and the remaining 23 are of non-forest. While the test set consists of 5 forest images and 7 non-forest images for a total of 12 images. However, this does not accurately depict how balanced the data set is or how big it is because for testing, the images are split into equal-sized sub-images and then classified. This means that the size of each individual picture matters because they will generate a different amount of sub-images. When split into sub-images of 7 by 7 pixels, the training set consists of roughly 29% forest data and 71% non-forest data and the test set roughly 20% forest and 80% non-forest. This should be taken into account when considering the performance, since just guessing non-forest on all samples in the training data would yield 72% accuracy.

3.1 Exploratory data analysis

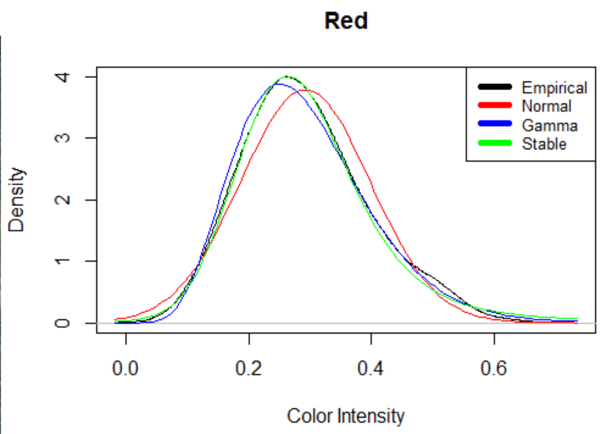
The first method of deforestation detection we were interested in is based on testing the distribution of color intensities of pixels from an image against a distribution with parameters obtained from a training set. With this in mind, we wanted to start by finding suitable distributions for the color intensities. This was done by exploring images obtained from the satellite and airplane images used in Google Earth.

Already at the start of this thesis, we knew that the stable distribution was one we were interested in trying, as it has already seen use in remote sensing areas such as ship detection [26] and video foreground detection [4]. Further, initial inspection of the empirical density of the data gave us some additional ideas for potentially suitable distributions. We looked at distributions including, but not limited to, stable, Normal, Log-Normal, Gamma, Cauchy, Weibull and Levy. We estimated parameters for each of the color intensity distributions and calculated the root-mean-square-error (RMSE)

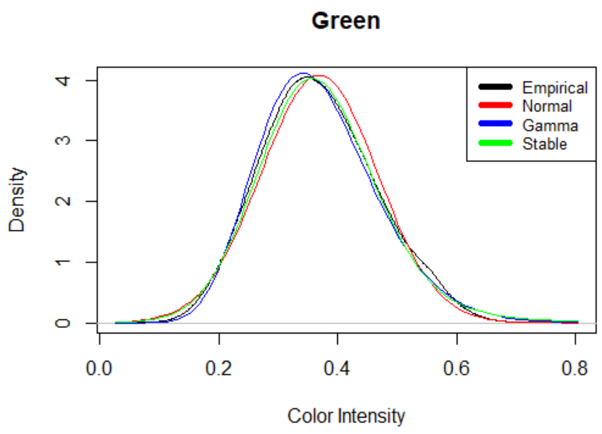
$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{f}(x_i) - f^*(x_i))^2}{n}}$$

where $\hat{f}(\cdot)$ is the probability density function using the estimated parameters, $f^*(\cdot)$ is the empirical density and x_i is color intensity of data point $i = 1, \ldots, n$.

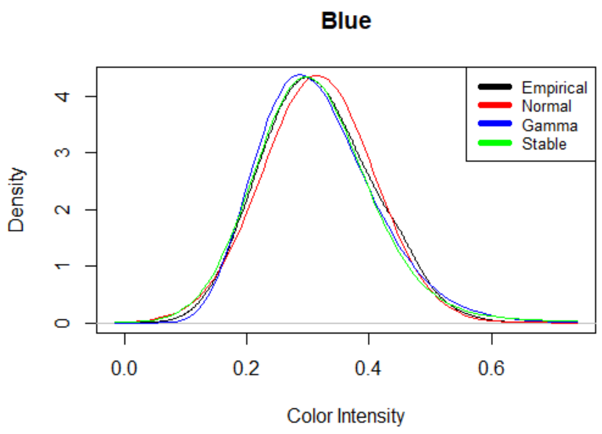
(a) Forest image analysed



(b) Comparison of densities for red color intensity



(c) Comparison of densities for green color intensity



(d) Comparison of densities for blue color intensity

Figure 4: Plots of empirical and estimated probability density functions for distribution of color intensities in an image of a forest

To choose the best fitting distributions we then plotted the densities of the three distributions with the lowest RMSE's together with the empirical densities and compared. Examples of these comparisons can be seen in Figure 4 where we can see that the three best fitting distributions for a forest image according to the RMSE's was Normal, Gamma and Stable. Further from the plots we see that the Normal distributions do not seems to fit very well while the Gamma and Stable distributions are quite close, with Stable edging out Gamma especially for the intensity of the color red.

Since the focus in this thesis is to be able to detect if an area has forest or not, we will work with images of forest. However, our exploratory data analysis shows us that the framework developed could be used to detect other natural objects from satellite images as well. In Figures 20 and 21 in the Appendix, we can see promising results from, again using the stable distribution to describe the color intensities of images of mountain and sea. When it comes to man-made objects such as, for example a city, we would assume a multimodal mixture model of some kind to be appropriate since satellite images would contain distinctly different parts, i.e., an orange roof, green grass and gray pavement.

3.2 Separability of terrain distributions

In this section, we will attempt to show that, assuming independent stable distributions for the color intensities, yields well-separated distributions for different terrains. We will consider the overlap between forest and the terrains of mountain and farmland. In order to show this, we consider the overlap of different distributions in the following way. Let X and Y be stable random variables representing one of the color intensities of a forest image and an image of a different terrain respectively. Let X and Y have location parameters δ_X and δ_Y. We compute the value a of the 5^{th} or 95^{th} percentile of X according to the following rule

$$a = \begin{cases} 5^{th} \text{ percentile of } X & \text{if } \delta_Y < \delta_X \\ 95^{th} \text{ percentile of } X & \text{if } \delta_Y \geq \delta_X. \end{cases}$$

Using this, we then compute the probability p of sampling a value above a, if $\delta_Y < \delta_X$, or below a, if $\delta_Y \geq \delta_X$, when sampling from the distribution of Y. Hence, the probability p is calculated in the following way

$$p = \begin{cases} P(1 > Y > a) & \text{if } \delta_Y < \delta_X \\ P(0 < Y < a) & \text{if } \delta_Y \geq \delta_X. \end{cases}$$

This is illustrated using densities in Figure 5, where, in this example, the red curve is the density of non forest image, the black curve is the density of the forest image and the area under the curve corresponds to the probability p.

The calculation of this probability is done for all three channels separately and the probabilities are finally multiplied. This multiplied value then give us a measure of how likely you would be to see all three values of the color intensities of a pixel, from a non forest image, take values in the range where the most of the pixels from the forest image would fall. This measure will take values between 0 and 1 since it is a product of three probabilities, further this metric calculated using the same image as baseline and comparison image will yield the value $0.95^3 = 0.857$.

To illustrate how small this overlap is, we compute these measures between the forest images and the mountain and farmland images and plot them in a

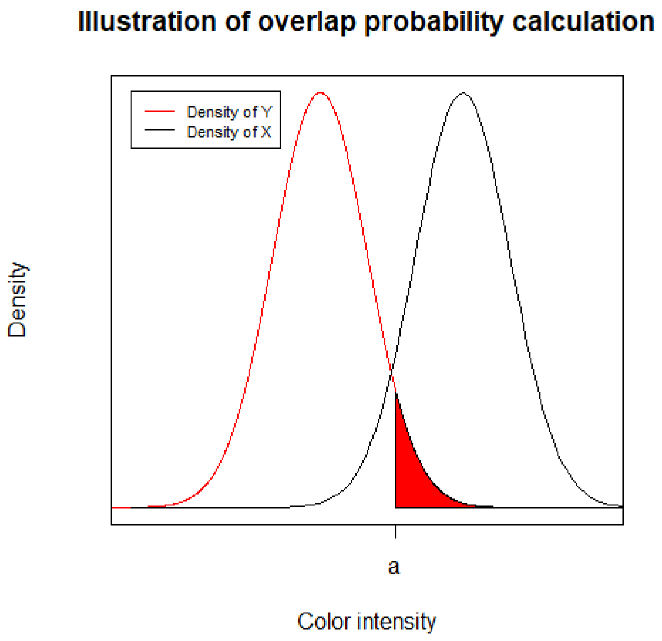


Figure 5: Illustration of the area corresponding to the probability of overlap between color intensity distributions of forests and other terrains.

heat map that we can see in Figure 6. We can see in this heatmap that the overwhelming majority of values are close to 0, indicating good separation between terrains. The outlier seems to be the image "Mountains_Norway_6.jpeg" with higher overlap measures across the board. To explain this, we look at Figure 7 depicting this image, we see that a good portion of the image seems to contain forest, or at least greenery, which would explain the higher overlap measure.

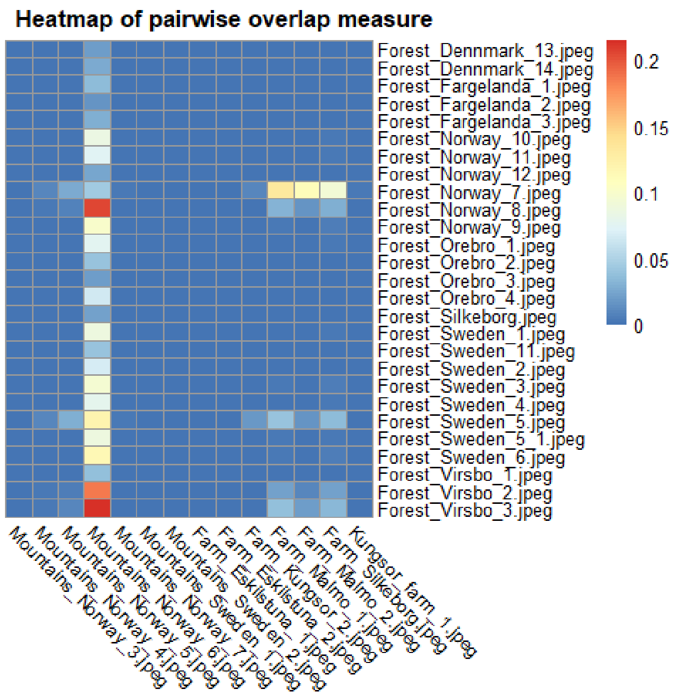


Figure 6: Heatmap of overlap measure between forest images and other terrains.

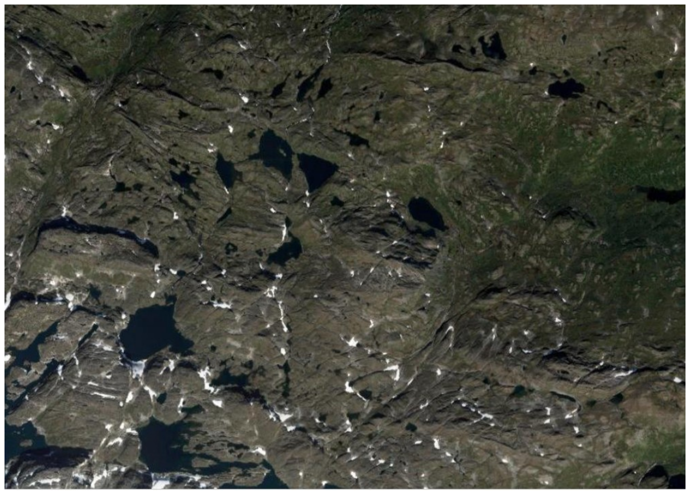


Figure 7: Image "Mountains_Norway_6.jpeg".

4 Modeling

As previously mentioned, this section will include two approaches to testing using training data. The first approach is parametric, where we assume a stable distribution for the color intensities of pixels. This distribution is multivariate—more precisely, it is three-dimensional with dimensions corresponding to the color intensities of the colors red, green and blue. For this approach, we will assume independence of these three random variables and perform tests on them separately. One reason for this is that, in R, we could not really find a standardized parametric goodness-of-fit test when normality cannot be assumed. Furthermore since many multivariate parametric goodness of fit tests compare cumulative distribution functions (CDF), they quickly become very computationally intensive since computation of the empirical CDF involves ordering observations and for each observation computing the sum of the probabilities less than or equal to that of said observation. Since the goal of this thesis is to create not only an accurate but also a relatively fast classifier, this kind of computational intensity is undesirable. There are also already some problems of estimating parameters of the stable distributions in the univariate case, which we did not wish to extend into the multivariate version of the stable distribution.

As previously mentioned, the second approach of this section is a non parametric one. We chose to include a non parametric approach, partly for the reason of comparing performance to that of a parametric one, as well as because of the intrinsic difficulties of multivariate parametric testing described above. Further, a non parametric approach was taken because our exploratory data analysis, as well as trouble in estimation of the parameters of the stable distribution, discussed in Section 2.1.2, lead us to think that our data does not follow any of our candidate distributions exactly, which is an assumption in the parametric case. For this non parametric approach, we have decided to use the squared Mahalanobis distance or unscaled Hotelling's two-sample T^2 statistic described in Section 2.3 and 2.4 respectively.

The time constraint of this thesis means that an in-depth comparison of non parametric multivariate tests is not a possibility for us, so we make no claim that the one we chose is definitely the best choice, as there are a variety of similar tests implemented in packages for R. Examples include the multivariate version of the Cramér test proposed in Baringhaus & Franz (2004) [3], implemented in the cramer package available at [29]. Another choice is using Multiple Response Permutation Procedures proposed in Mielke Jr, Berry & Johnson (1976) with implementation in, for example, the vegan package available at [34]. Another possible test is the energy test proposed by Szekely & Rizzo (2013) [24], implemented in the energy packages [35]. A final example is the crossmatch test proposed in Rosenbaum (2005) [21] implemented in the crossmatch package available at [30]. We again stress that this is not an exhaustive list.

4.1 Non-parametric modeling

This section will cover the procedures gone through when creating and training our non parametric model.

In order for any image to be tested, a training set of images containing only forest needs to be provided. The image to be tested is then split into equally sized square sub-images, where the size of these sub-images needs to be decided in advance. As previously mentioned, we have decided to use the size 7-by-7 pixels. The distribution of the color intensities in each sub-image is then tested against the color intensities of all the images of forest in the training set. For each sub-image the score of the best performing test is returned and then compared to a threshold. For the non parametric model, the test score will be the squared Mahalanobis distance outlined in Section 2.3 and 2.4. We chose to use the squared Mahalanobis distance D^2 instead of its scaled version corresponding to the Hotelling's two-sample T^2 statistic, the reason for this is mostly computational. After reading the reference forest images, we reduce the number of pixels used for each image to the number of the smallest image by randomly sampling. This reduces the computational times in two ways: it reduces the time it takes to compute the square Mahalanobis distance for each forest image (except the smallest one) and it allows us to not compute the scaling factor. Because the size of all reference images is the same and the size of all sub-images is the same, the scaling will be identical for all calculations. Further, as long as the smallest reference image is not extremely small, the procedure of reducing the sample sizes by random sampling should still give a good representation of the distribution. For reference the smallest forest picture in our training data contains over 7000 pixels.

Because the Mahalanobis distance is non parametric, the training of this model is essentially only deciding the threshold for the distance. The threshold should be chosen so that as many non-forest sub-images as possible has a distance above the threshold and as many forest sub-images as possible has a distance below the threshold.

This threshold is decided using k-fold cross-validation; we decided to use $k = 5$ in order to keep the size of the holdout folds relatively large. The cross-validation is started by first randomly shuffling the training data and then splitting it into k folds. The data (images) in one holdout fold is then split into smaller sub-images of predecided size. After this the squared Mahalanobis distance, see Eq. (9), between each sub-image and all images of forest in the remaining $k - 1$ folds is calculated. For each sub-image, the smallest squared Mahalanobis distance is then chosen. These steps are repeated for all k folds. Finally, with the squared Mahalanobis distances computed for all the sub-images in all the folds, we adopt the accuracy metric, described in Section 2.5.1, to decide the best threshold for the distance to use. The best threshold is found by comparing the distances of all sub-images to a range of thresholds, from 0 to T_{max}, and the threshold in this range that gives the best accuracy is chosen. For the non parametric model, we have found empirically that $T_{max} = 15$ works well.

It may be a good idea to run cross-validation multiple times and average the results in order to reduce the error in the estimates we have, however, decided against this. Partly due to the already large amount of time consuming computations needed in our training and partly because we did not see large differences in the estimates in practice. A more concise outline of this procedure can be seen in Algorithm 2.

Algorithm 2 Non parametric cross validation procedure

```
1: Inputs: Data set of images labeled forest or non-forest, sub image size $m$,
   number of folds $k$ & max threshold range $T_{max}$.
2: Randomly shuffle images and split into $k$ folds.
3: for $i = 1$ to $k$ do
4:     Split images in fold $i$ into smaller sub images of size $m$-by-$m$.
5:     Reduce sample size of all forest images in the $k - 1$ training folds to the
       size of the smallest forest image in data set.
6:     for each sub-image in fold $i$ do
7:         Compute squared Mahalanobis distance $D^2$, Eq. (9), between sub-image
           and all images labeled forest in the remaining $k - 1$ folds.
8:         Choose smallest $D^2$ for sub-image.
9:     end for
10: end for
11: for $t$ in 0 to $T_{max}$ do
12:     For all sub-images, predict sub-images with $D^2 < t$ as forest and $D^2 > t$
        as non forest.
13:     Compute the accuracy for the given threshold $t$, Acc($t$).
14: end for
15: return $\arg\max_t \mathrm{Acc}(t)$ and data set of reduced size forest images.
```

In Figure 8, we can see a plot of the accuracy versus threshold for the non parametric classifier with sub images of size 7-by-7 pixels. A vertical red line is drawn at the best performing threshold. The threshold that gives the best performance according to the accuracy is 4.16, giving an accuracy of roughly 93.8% across the training set.

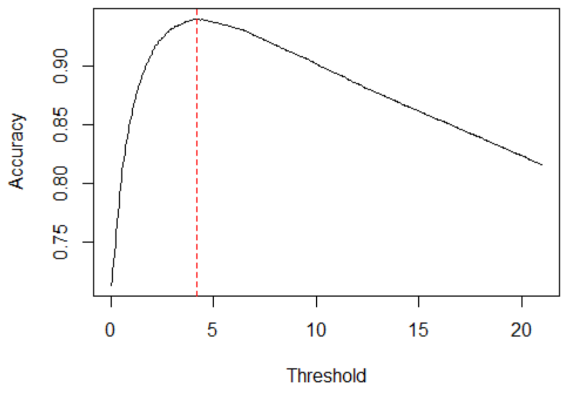


Figure 8: Threshold versus accuracy from 5-fold cross-validation using the non parametric model and sub-images of size 7-by-7 pixels.

4.2 Parametric Modeling

This section will cover the procedures gone through when creating our parametric model. The training part consists mainly of parameter estimation and deciding thresholds for the test statistics. As previously mentioned, each color intensity distribution is considered independently in this method.

For this model, we use the Cramér-von Mises test statistic to decide if an image is of forest or not. The training procedure is similar to the one used in the non parametric case described in Section 4.1. Just like before, we use 5-fold cross validation. Before this is done, stable distribution parameters are estimated for the three color intensities of all forest images in the data set. This is done prior to the cross-validation in order to avoid multiple computations of the time consuming estimations described in Section 2.1.2.

Another difference in this model is that due to the extreme amount of computations needed when computing the Cramér-von Mises statistic with multiple parameter set, we decided to include the option of clustering in this cross-validation. The forest images are clustered using hierarchical clustering, described in Section 2.6, with the squared Mahalanobis distance as a dissimilarity measure. Images which are clustered together are assumed to have the same distribution, so their parameters are averaged and returned, effectively reducing the number of parameter sets which need to be tested. We found that the number of clusters $n_{clust} = 7$ gave a good balance of speed and performance.

We chose to consider the minimum sum of the three statistics when choosing the best test statistics result to return for each sub-image. With the Cramér-von Mises statistics computed for all the sub-images in all the folds, we again adopted the accuracy metric to decide the best threshold for the distance to use. Since we have three statistics in this case, one for each color intensity, this is not as straightforward as in the non parametric case. In this case, to get as close as possible to the maximum accuracy, we need to perform a grid search over a range of threshold values for the three dimensions. Where we must compute the accuracy for all combinations of threshold values in this range. The choice of the range of thresholds to test over is not as easy in the parametric case, this is because the Cramér-von Mises test statistic depends on the sub image size m. We have found that for smaller values (which we recommend), it works well to use $T_{max} = 2 \cdot m$. However, when increasing m this will not work, as you may need to go as high as $T_{max} = 10 \cdot m$. The cross validation procedure for the parametric model is outlined in Algorithm 3.

As previously mentioned, we use sub-images of 7-by-7 pixels. Performing the cross validation to find the best thresholds for the Cramér-von Mises statistics yielded an accuracy of roughly 96.2%. With the best result given by the thresholds 5.76, 6.18, 5.76 for the color intensities of red, green and blue, respectively.

Algorithm 3 Parametric cross validation procedure

```
1: Inputs: Data set of images labeled forest or non forest, sub image size $m$,
   number of clusters $n_{clust}$, number of folds $k$ & max threshold range $T_{max}$.
2: for All images labeled forest do
3:     Estimate the stable distribution parameters using the scheme outlined
       in Section 2.1.2.
4: end for
5: Randomly shuffle data and split into $k$ folds.
6: for $i = 1$ to $k$ do
7:     Split images in fold $i$ into smaller sub images of size $m$-by-$m$.
8:     for each sub-image in fold $i$ do
9:         Cluster forest images in the remaining $k-1$ folds into $n_{clust}$ clusters and
           return averaged stable distribution parameters within clusters.
10:        Compute Cramér-von Mises statistics, Eq. (8), for each color intensity
           using the $n_{clust}$ parameter sets.
11:        Sum the Cramér-von mises statistics for the three color intensities.
12:        Choose the three Cramér-von Mises statistics which produces the small-
           est value when summed.
13:    end for
14: end for
15: for All combinations $\boldsymbol{t}$ in $(0,0,0)$ to $(T_{max}, T_{max}, T_{max})$ do
16:     For all sub-images, predict sub-images with Cramér-von Mises test statis-
        tics smaller than all three corresponding elements of $\boldsymbol{t}$ as forest and the
        rest as non forest.
17:     Compute the accuracy for the given threshold $\boldsymbol{t}$, Acc($\boldsymbol{t}$).
18: end for
19: return $\arg\max_{\boldsymbol{t}}$ Acc($\boldsymbol{t}$) and the set of stable parameters obtained from clus-
    tering forest images in data set.
```

5 The package: deforeStable

As previously mentioned, the work presented in this thesis also resulted in a small R package. This section will briefly cover the functionality of this package and include some examples. A more thorough run through of the package can be found in the manual of the package. The package and manual are available at the link provided in Muren & Otryakhin (2021) [33]. In the github repository containing the package we will also include the training and test set, as well as mixed terrain images for visual testing.

Further, many functions in the package are computationally expensive; because of this, we have included parallelization of the computations performed in many functions. This means that a user with many cores available for these computations will get results much faster.

A natural place to start is with reading images, as this is the core data used in our algorithms. The package contains various options for reading data, they are all based on reading JPEG images using the jpeg package [36], but return the data in different forms. We go through the data reading below.

- Firstly, we load the package as with any R package and provide a directory containing the image we want to read. We also load the doParallel package [37] and let it use all but one of the computer cores.

```
library("deforeStable")
library("doParallel")
cores <- detectCores()
cl <- makeCluster(cores[1]-1)
registerDoParallel(cl)
dir <- "C:\\Users\\User\\images\\"
```

- We then continue with reading an image into an object of class RasterStack. The raster package [31] is the industry standard for working with these kinds of images and this gives it many advantages. Naturally, there are already many functions implemented in packages across R that uses RasterStack object. Further, you can tag images with a geolocation, making comparisons of altered images and the original easier. Another example, that we show below, is that you can plot an image directly in R. The name of the image we want to read must be provided. In Figure 9 we see an example of plotting images using raster.

```
library("raster")
filename <- "image.jpeg"
image_raster <- read_data_raster(filename, dir)
plotRGB(image_raster, scale = 1, asp = 1)
```

- Another way of reading data included is to read it into a matrix with one row for each pixel and columns for the color intensities of the colors red, green and blue. This is useful when considering only the distributions

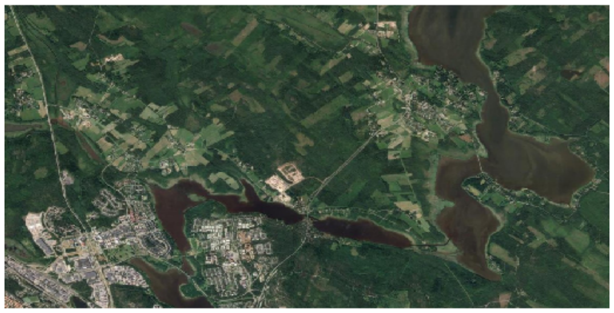


Figure 9: Example output of plotRGB.

of the color intensities. Example with histogram of red color intensity is included below in Figure 10.

```
image_mat <- read_data(filename, dir)
hist(image_mat[,1])
```

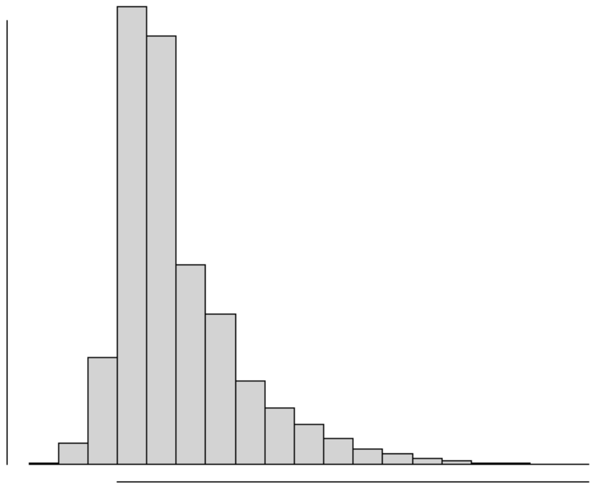


Figure 10: Example histogram plot of read data.

- Finally, the data can be read into a list of three matrices with dimensions corresponding to the pixel dimensions of the image, one matrix for each color intensity. This is useful to keep the image representation of the data and when splitting images into smaller sub-images.

```
image_mat_list <- read_data_matrix(filename, dir)
str(image_mat_list)
List of 3
 $ : num [1:836, 1:1297] 0.133 0.118 0.098 0.137 0.118 ...
 $ : num [1:836, 1:1297] 0.22 0.204 0.192 0.231 0.22 ...
 $ : num [1:836, 1:1297] 0.169 0.153 0.141 0.18 0.176 ...
```

- For individuals interested in testing the package but not gathering their own data, the package contains a sample of 49 images on the RasterStack format. Below, we show how to get this data, its descriptions, and how to change image number 7 into the other two formats described above.

```
data("geoimages") #a list of images
data("geoimages_desc") #image names & descriptions
obj <- geoimages[[7]]
image_mat <- as.matrix(obj)
image_mat_list <- list(as.matrix(obj$layer.1),
                       as.matrix(obj$layer.2),
                       as.matrix(obj$layer.3))
```

The function for estimating the parameters of the stable distributions of the color intensities, described in Section 2.1.2, uses the single matrix representation of the data shown above. The function that does these computations, using the package StableEstim [32], returns a data frame with rows corresponding to the colors red, green and blue. The columns contain to the estimations of the parameters α, β, γ, and δ. We see how this is done, with example output below.

```
params <- Koutparams(image_mat)
params
          alpha       beta      gamma     delta
red    1.703146  0.4100056 0.01862494 0.1406803
green  1.998507 -0.9990000 0.02513797 0.2027838
blue   1.998673 -0.9990000 0.02305076 0.1753170
```

Now that we have described the data reading process and parameter estimation, we continue examining the functions handling the computation of the statistics used in our classifiers. These are the Cramér-von Mises statistics, described in Section 2.2, and the squared Mahalanobis distance, described in Sections 2.3 and 2.4. We begin with the Cramér-von Mises function; this function takes a data frame with parameters as returned from the above Koutparams function, as well as the color intensities of pixels on the matrix form as input. It returns a vector with the values of the Cramér-von Mises test statistics, performed using stable distributions with the provided parameters and the color intensity distribution of the provided data. Below, we see how this is done with the data and parameters we read and computed above, and example output.

```
CVMtest <- Forest_Tester_CVM(params, image_mat)
CVMtest
      X1        X2        X3
 21.23581  18.27714  14.16928
```

For the squared Mahalanobis distance function, the input is reference data and data to test in matrix form. The function returns a numeric, which is the value of the squared Mahalanobis distance. Below, we see how this is done, and example output.

```
reference_image <- geoimages[[5]]
reference_data <- as.matrix(reference_image)
SqMahaladist <- Mahala_dist(reference_data, image_mat)
SqMahaladist
2.967856
```

Now that we have covered the core functions used in the cross-validation, described in Sections 4.1 & 4.2, we move on to the functions that handle the deciding of optimal thresholds for the test statistics through cross-validation. These functions take almost identical inputs for both parametric and non parametric models. These inputs are: the path to a directory containing the forest images of the data set, the path to a directory containing the non-forest images of the data set. The number of folds to use in cross-validation, as well as the number of points (pixels) used when splitting the images into smaller sub-images, needs to be provided. Further, the function for the parametric model contains an option for using the clustering, mentioned in Section 4.2, due to the time consuming nature of this function when the data set grows. The parametric function also includes an argument for the top end of the threshold to calculate the accuracy over. This value is set to two times the number of points used when splitting the data; this works well for small sub-images (which we recommend) but may need to be increased for larger sub-images. The functions are used as follows:

```
forestdir <- "C:\\Users\\User\\images\\forests\\"
Nonforestdir <- "C:\\Users\\User\\images\\nonforests\\"
ParametricCV <- ParamCV(forestdir = forestdir,
                        Nonforestdir = Nonforestdir,
                        n_pts = 7,nrfolds = 5,
                        clustering = TRUE,
                        maxt = 14)
NonparametricCV <- NonParamCV(forestdir = forestdir,
                              Nonforestdir = Nonforestdir,
                              n_pts = 7, nrfolds = 5)
```

These functions return lists where the first element contains a vector with the best thresholds according to the accuracy, as well as the accuracy. The second element of the list contains a data frame with all thresholds tried, corresponding accuracies obtained, as well as the number of true positives, false positives, false negatives and true negatives. For the non parametric function, the second element of the list can, for example, be used to create the plot shown in Figure 8. The third element is a list with the sets of the stable distribution parameters used in the parametric case and a list of matrices containing the data of the reduced size forest images in the non parametric case. These should be used for testing together with the threshold found through cross-validation.

With this, we can move on to the functions that actually perform the classification of images. These functions take as inputs: the image you want to classify, in the RasterStack format, the number of points for splitting the image, which should be the same number used in the cross-validation to get the correct threshold. Finally, the threshold preferably obtained from cross-validation needs to be input, as well as a list of parameter sets in the parametric case and a list of reference images in the matrix form for the non parametric case. The parameter and reference list should be obtained from the forest images used in cross-validation since these are the images the optimal threshold is decided with. The functions return a matrix of zeroes and ones with the same dimensions as the pixel dimension of the original image, where zero indicates that the pixel does not contain forest and one indicates that it does contain forest. In the code snippet below, we show how these functions are called, as well as a way to visualize the results.

```
Pthres <- ParametricCV[[1]][1:3]
NPthres <- NonparametricCV[[1]][[1]]
ref_list <- NonparametricCV[[3]]
param_list <- ParametricCV[[3]]
Nonpar_pred <- Nonparam_classifier(test_image, n_pts = 7,
                                   references = ref_list,
                                   thres = NPthres)
Para_pred <- Param_classifier(test_image, n_pts = 7,
                              pars = param_list,
                              thres = Pthres)
#Save visualization
jpeg::writeJPEG(image=Para_pred, target='predicted.jpeg')
```

It is worth noting that the parametric classifier is up to 20 times slower than the non parametric one, even when implementing clustering, effectively reducing the number of parameter sets from 22 to 7.

Finally, we give an example image and output obtained when using the parametric model to classify an image with patch-wise deforestation—this is shown in Figure 11. In the left panel, we see the original image and we see the classification result in the right panel. White indicates predicted forest and black indicates predicted non-forest.

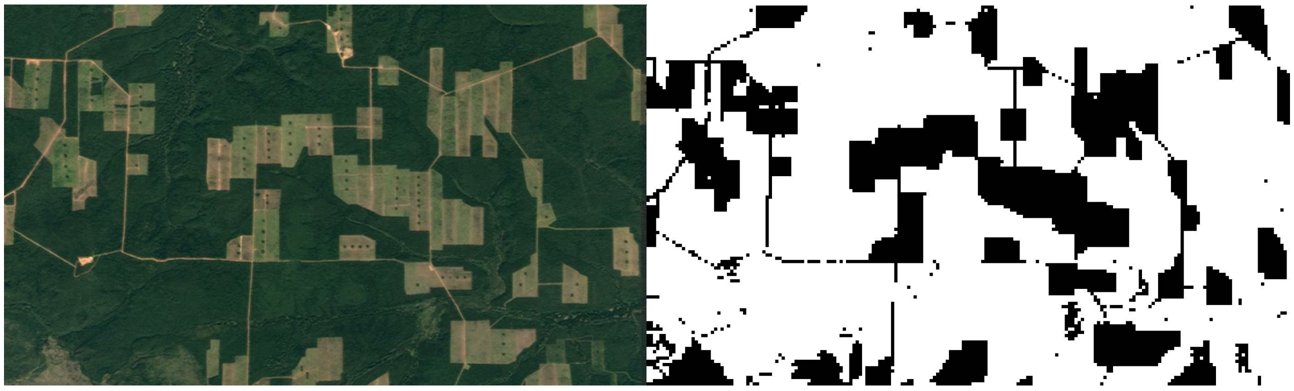

(a) Original image to be tested (b) Prediction using parametric classifier

Figure 11: Example input and output using the parametric classifier proposed in the thesis.

6 Results

This section will cover the results obtained when applying the two models proposed in this thesis to the portion of the data set aside for testing. We will limit this section to the numerical results and discuss these and comparisons between the models further in Section 7.

6.1 Model performance metrics

For an initial overview of the results from the two models, we provide the confusion matrices obtained when applying the models to the test set. These can be seen in Table 3 for the parametric model and Table 4 for the non parametric model.

Table 3: Confusion matrix for parametric model.

Confusion Matrix		
	Actual Forest	**Actual Non Forest**
Predicted Forest	2008	216
Predicted Non Forest	156	8836

Table 4: Confusion matrix for non parametric model.

Confusion Matrix		
	Actual Forest	**Actual Non Forest**
Predicted Forest	2068	76
Predicted Non Forest	96	8976

For more in-depth results, we look in Table 5, where we see the values of the performance metrics discussed in Section 2.5.1, for the two models.

Table 5: Performance metrics for tested models.

Metrics	Parametric	Non parametric
Accuracy (Acc)	0.967	0.984
Error rate (Err)	0.033	0.016
Sensitivity (Sn)	0.927	0.955
Specificity(Sp)	0.976	0.991
Precision(P)	0.902	0.964
F-Score (F)	0.915	0.960
Alarm Area (AA)	0.801	0.808

6.2 ROC curves

In this section, we present the ROC curves of the two models. The theory these are based on was discussed in Section 2.5.2.

In Figure 12 we see the ROC curve for the parametric model. The red point indicates the true positive rate (TPR) and false positive rate (FPR) obtained for the threshold that gave the best possible accuracy on the test set. This accuracy was 0.976. The green point indicates the TPR and FPR obtained for the threshold given by our cross-validation training, with corresponding accuracy of 0.967. Because the thresholds tested are three dimensional the parametric model can give different TPR for a given FPR, we have plotted the best possible TPR for each FPR between 0 and 1. This is why the green point is not on the curve. This means that, in other words, it is possible to get a better TPR for the FPR obtained by using the threshold given by the cross-validation.

In Figure 13, we see the ROC curve for the non parametric model. Again, the red point indicates the best TPR and FPR obtained for the threshold that gives the best possible accuracy on the test set, which was 0.985. Similarly, the green point indicated the TPR and FPR obtained for the threshold given by our cross-validation training, with accuracy 0.984.

Finally, for this section, we provide a plot with the ROC curves from both models for comparison. This can be seen in Figure 14, where the red curve for

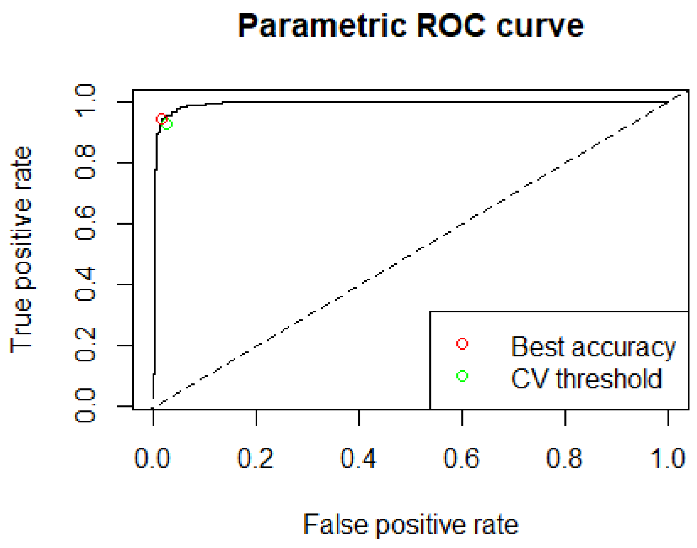


Figure 12: ROC curve for the parametric model.

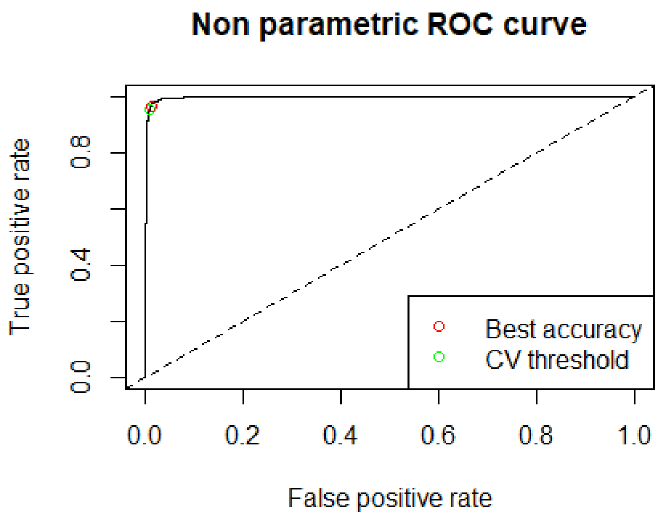


Figure 13: ROC curve for the non parametric model.

the parametric model gives the AUC value of 0.993, while the black curve for the non parametric model gives the AUC value of 0.998.

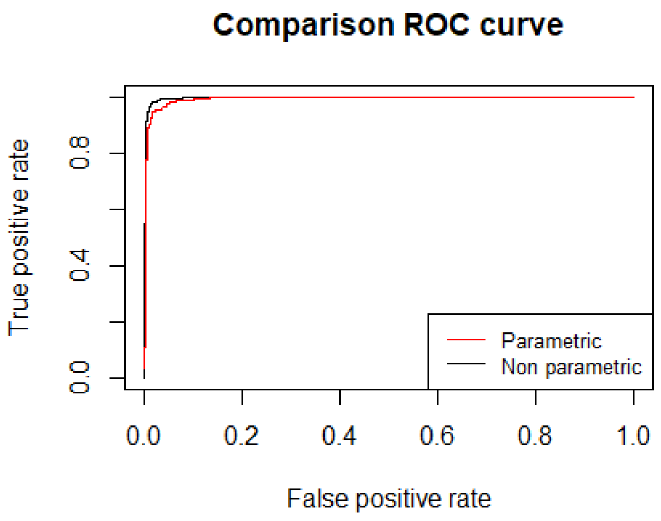


Figure 14: Comparison between the ROC curves for the two models.

7 Discussion

In this section, we will mainly discuss the results, presented in Section 6, comparisons and differences in the performance of our proposed models, as well as potential improvements. However, first we will discuss some things that should be taken into consideration when judging the performance of the models.

7.1 Performance considerations

When studying the results of the models proposed in this thesis, there are a few things that should be taken into consideration.

Firstly, just like in any model that is trained on data, the quality of the data has a big impact on the performance. Here, we mean how well the data is labeled, often denoted the ground truth. If the ground truth is incorrect, the data will train under incorrect assumptions, which will lead to worse performance on unseen data. Similarly, if there are errors in the ground truths of the test data, performance metrics can suffer where they should not. This is a problem that most likely will have some effect on the metrics presented in this thesis, due to the nature of satellite and aerial images. Images covering large areas can easily contain some parts of different terrains. This problem is illustrated in Figure 15, which is an image in the test set labeled as not forest, we see that there are parts of this image that might be considered as forest.

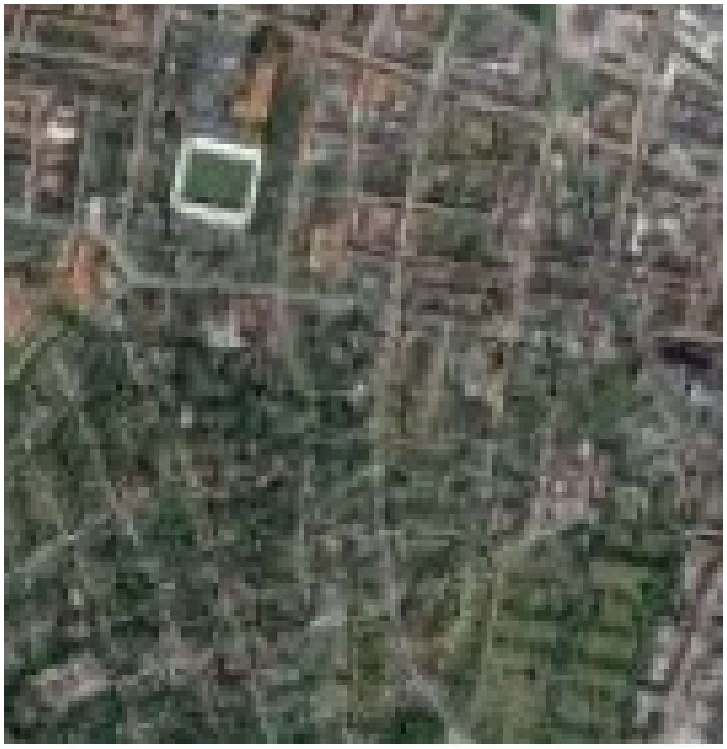


Figure 15: Example image from test set with questionable ground truth.

This naturally leads us into another consideration, which is the size m to use when splitting images into smaller sub-images. As previously mentioned, we have chosen to use $m = 7$, which gives sub-images of 7-by-7 pixels. This produces an image like the one shown in Figure 11. Naturally, lower values of m makes it possible to more finely divide images into forest and non-forest parts. However, lower values of m also comes with downsides—for example, when m is small, outliers have more effect on the predictions of the sub-images they are in. Conversely, using large values of m means larger samples and hence a better representation of the distribution of the color intensities of the sub-image. Using large values of m, would not give good performance on images with mixed terrains, but it would give better results when testing on whole images with homogeneous terrains. This is because the larger sub-images will contain mostly pixels of the label of the picture, making the ground truths of the sub-images more accurate and easier to distinguish. This is illustrated in Figure 16, where we plot the best accuracies obtained using the non parametric model on the test set, with increasing m. In this figure we can see that the accuracy on the test set trends upward with increasing sub image size and perfect accuracy is obtained for all sub image sizes above 28.

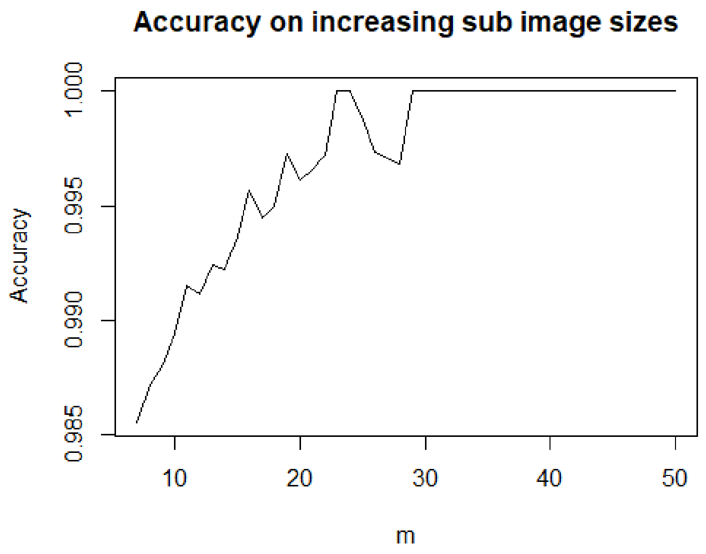


Figure 16: Non parametric model test set accuracy when increasing the size of the sub images.

7.2 Parametric model performance

The main metric we have considered in this thesis is accuracy, from Table 5 we see that the parametric model achieved an accuracy of 0.967 on the test set. Even with the test set being split roughly 80/20 in favor of non-forest, this result is considered quite good. From the confusion matrix in Table 3, we see that a larger proportion of data labeled forest is incorrectly classified than data labeled not forest. Due to the unbalanced nature of the test set, this may not be entirely relevant to the classifiers ability to distinguish forest and not forest, since an even proportion of incorrectly classified samples could lead to worse accuracy.

This is why we consider a variety of metrics, specified in Section 2.5.1. Other than accuracy we have its natural counterpart error rate, which in Table 5 tells us that only 3.3% of the samples in the test set were incorrectly classified. The better performance on non-forest data can also be seen in the smaller sensitivity score of 0.927, compared to the specificity score of 0.976. The metric that shows the worst result for the parametric model is precision, with a value of 0.902, it says that there are relatively many non forest images incorrectly classified as forest. On the other hand, the alarm area, which tells us the fraction of images classified as not forest, is 0.801, which is very close to the true fraction in the data set, which is 0.807 so within 1%. This would be good in applications where the interest is in deforestation with human double-checking. The lower precision would, however, indicate worse performance in applications interested in finding forest, such as computing the percentage of park areas in cities. The final metrics from Table 5 to consider in this section is the F-Score, which is 0.915. The F-score is a good alternate metric to the accuracy for the overall performance of the classifier. We see a score that is comparably quite a bit lower than the accuracy for the parametric model—this is because it takes into account the poor precision score.

The performance of the classifier is also visualized in the ROC curve, plotted in Figure 12, and with the corresponding AUC score of 0.993, which is very good. The plot also contains the points corresponding to the threshold our cross-validation returned (green point) and the best possible result the classified could obtain on the test set (red point). While they are quite close, the fact that the green point is not on the curve means that it is possible to find a threshold with a better TPR for this FPR.

Finally, for the parametric model, we considered how good the individual color intensities are at classifying the data set. We classified the test set using only the Cramér-von Mises test statistic for the red, green and blue color intensities, respectively. The best possible accuracies obtainable with these tests are 0.927 for red, 0.924 for green and 0.958 for blue. This suggests that the blue color intensity is the best at classifying the test set alone. This is relatively close to the best accuracy obtainable using all three tests, which was 0.976. Since the parametric modeling is quite slow with the current implementation, this tells us that it may be worth investigating using only one, or two color intensities.

7.3 Non parametric model performance

For the non parametric model we see, in Table 5, a very good accuracy of 0.984, which translates to only 1.6% of observations being misclassified. The confusion matrix seen in Table 4 shows that the performance is better on data labeled non-forest, just like in the parametric case. This again can be seen in the sensitivity score of 0.955, being lower than the specificity score of 0.991, even if this is more a case of very high specificity than low sensitivity. For the non parametric model we see good performance on the fraction of forest images classified as forest correctly, with a precision score of 0.964. The model still retains an alarm area extremely close to the actual fraction of non forest in the test set with an alarm area of 0.808, compared to the true fraction of 0.807. Finally, for the metrics seen in Table 5, we have an F-score of 0.960, which is quite natural with the sensitivity and specificity being very high for the model.

When considering the ROC curve in Figure 13 we again see very good performance which is translated into the extremely good AUC value of 0.998. We also see that the green dot indicating the TPR and FPR values obtained with the cross validation threshold is very close to the red dot for the best possible accuracy threshold on the test set. This indicates that the cross-validation procedure is good at finding the optimal threshold for this model.

7.4 Model comparison

In this section, we explore the differences in the results and touch on how the properties of the models affect them and their results.

Initially, we can see in Table 5 and in the comparison of the ROC curves, seen in Figure 14, that the non parametric model performed better across the board. This may be surprising because you would often expect to see better results from a parametric model compared to the non parametric one. This expectation was further solidified for us because of the more computationally expensive and time consuming training needed for the parametric model, as well as the better performance in the cross-validation. The accuracy obtained on the test set was very close to the accuracy obtained in the training for the parametric model, with 0.962 in training and 0.967 in testing. The non parametric model substantially outperformed the accuracy achieved in training on the test set, with 0.938 and 0.984, respectively. This may be an indication that the training set is not a perfect representation of the data in the test set.

Further, possible reasons for the worse performance of the parametric classifier is the clustering of the pictures used for computing the Cramér-von Mises statistics, as well as the fact that finding an optimal threshold is much harder. The clustering is needed because for the Cramér-von Mises statistics we compute needs the CDF of the stable distributions. Computing the CDF repeatedly with the numerical computations described in Section 2.1.1, is very computationally expensive, so reducing the number of CDF's needed takes the training from extremely slow to just slow. Unfortunately, reducing the number of parameter sets to test against will likely affect the performance. When it comes to finding the optimal threshold in training, there is again, an issue of computational complexity. Because we need to test all combinations in the range chosen for the thresholds of the Cramér-von Mises test statistic, the number of thresholds tested, as can be seen in Algorithm 3, grows very fast. This means that when trying to keep the computational time down, the possibility of missing the optimal threshold increases. This can be seen in Figure 12 by the positioning of the green point being further away from the optimal red point than in the non parametric case.

7.5 Potential improvements

While the performance of the model is considered good based on the metrics described in Section 2.5.1, there is of course room for improvement. Potential improvement of the models can be divided into two categories. First improvements in the computational times of the models, especially the parametric one. Secondly, improvements in the numeric results of the models.

When it comes to computational times, the biggest improvement would be in finding faster ways to compute the CDF of stable distributions. These are quite time consuming and are computed an extreme amount of times. As an example, consider applying the parametric model to the test set. Even with the forest pictures clustered into 7 clusters, we have 7 parameter sets where each set contains the parameters for 3 stable distributions. When split into 7-by-7 pixel sub-images, the test set contains 11216 sub-images. This means that we need to compute $7 \cdot 3 \cdot 11216 = 235536$ CDF's for stable distributions. Making algorithms of this kind extremely fast is, however, probably not realistic—this is due to the size of images obtained from satellites or planes. Since the images need to be split into small sub frames in order to detect small changes in terrains, this means a large amount of tests need to be performed. Further, when it comes to computational times, there is likely room for improvements in the estimation of the stable parameters as well as in the coding implementation. The code is written in R and includes some loops, which are notoriously slow in R, compared to if they were written in C++, Wickham (2019) [27]. It is also likely that computational times can be greatly reduced in practice by applying knowledge of the area in training. By this we mean that with knowledge of the forests in the area, a smaller number of images, which represent the different terrains of the area well, could be chosen. Perhaps images of coniferous forest can be excluded if you know the area only contains deciduous forest. Hence, reducing the number of images trained against using knowledge of the area could reduce the computational time greatly and even potentially improve performance. For this thesis, we have considered images throughout most of Scandinavia, so the 22 forest images used as references may be warranted. However, the good performance of the parametric classifier with the reference forest images clustered into 7 clusters could be an indication that this number could be reduced while preserving performance. Using smaller images that are representative of the area will also reduce computation times. In our data set, there are images up to the size of almost 18000 pixels, which is likely unnecessarily large. Further, for applications of the algorithms in a given area, training should only have to be done once per season, since the color intensity distribution of a given forest should be mostly the same, except for seasonal changes.

For performance, there are potentially improvements to be made in the estimation of the stable parameters. As mentioned in Section 2.1.2, we tried some different techniques for estimation before deciding on the Koutrouvelis regression-type technique. However, the limited time scope of the thesis made it impossible to include a more extensive study on the different parameter estimation techniques. Looking into using the multivariate stable distribution for the color intensities may also provide improvement in performance. For the parametric model, there is likely room for improvement in finding the optimal threshold for the Cramér-von Mises statistics, as discussed in Section 7.4.

The last, and in our opinion, probably the most drastic improvement in performance could be obtained from improving the data used to train and test on. One potential improvement in the data is to solve the ground-truth problem discussed in Section 7.1. Another problem with the data from Google Earth is difference in shading—this can be seen in the different shades of the right and left side of Figure 17, the shading difference produces a line, vertically, through the middle of the image, this is highlighted in the top right corner. Because the images from Google Earth can be composites of images taken at different times or by different cameras, there are sometimes shading differences in images. Shading differences like this will affect the results because it will change the distribution of the color intensities.

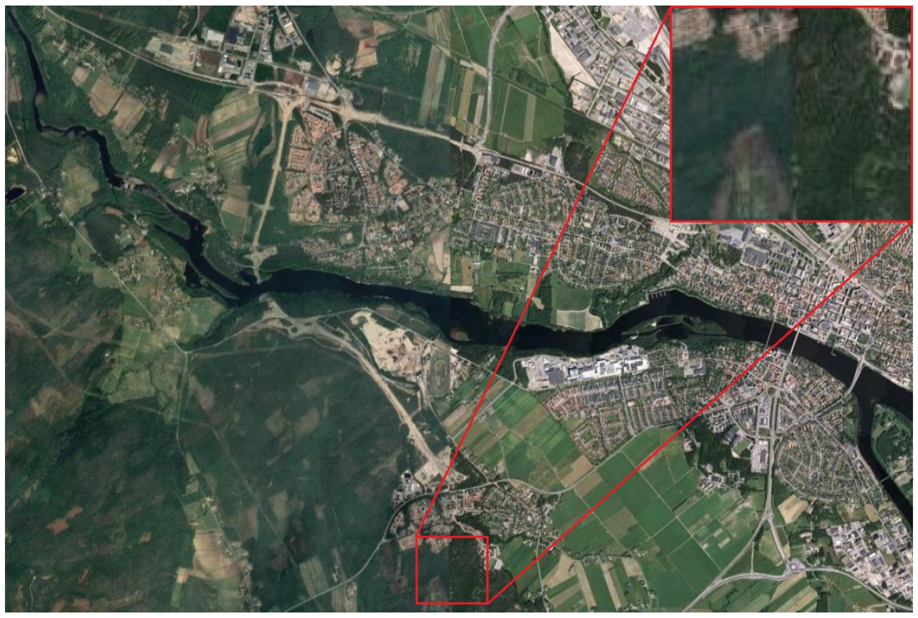


Figure 17: Google Earth image with different shades, zoomed in area highlighted in top right corner.

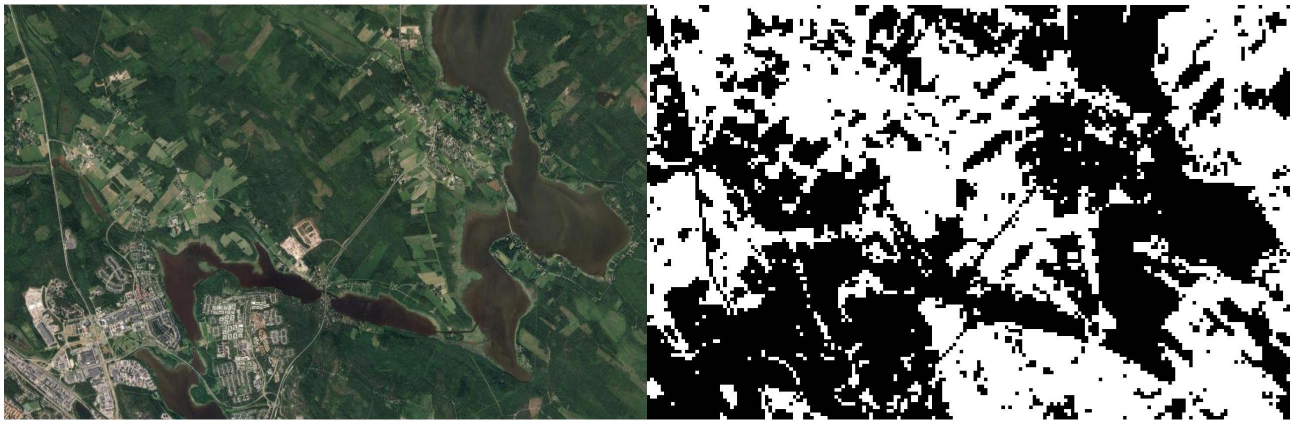
Finally, images with higher resolution would likely translate into better performance. Higher resolution would mean that you could keep sub-images small, to detect small changes in terrain, while still getting a good representation of the color intensity distribution with more pixels. This should theoretically have a similar effect to what we saw for increasing sizes of sub-images in Figure 16.

We expect data obtained through the Sentinel-2 infrastructure that will be used in the paper to alleviate many of these problems with the data.

8 Conclusion

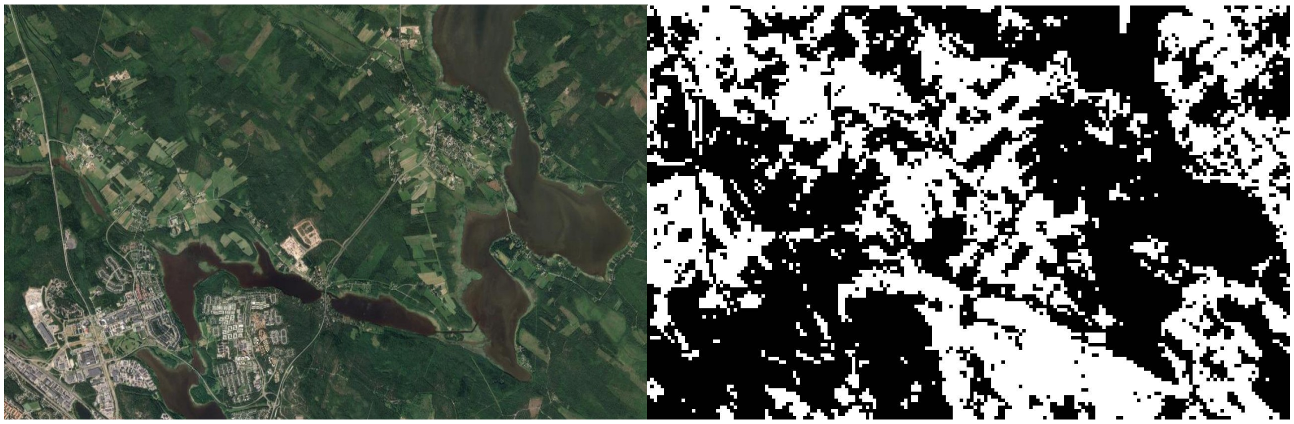
In this thesis, we have proposed two algorithms to automatically detect if areas of images are covered by forest or not. We saw good overall performances from both models with the non parametric one having the edge with an accuracy of

98.4% on the test set, compared to the parametric models accuracy of 96.7%. For more visual examples of the performance of the models we see an example of the non parametric models performance on an image of mixed terrain in Figure 18 and the performance of the parametric model on the same image in Figure 19. We can see that they are quite close, with the parametric model predicting a bit too much non-forest perhaps.



(a) Original image to be tested (b) Prediction using non parametric classifier

Figure 18: Example input and output using the non parametric classifier proposed in the thesis.



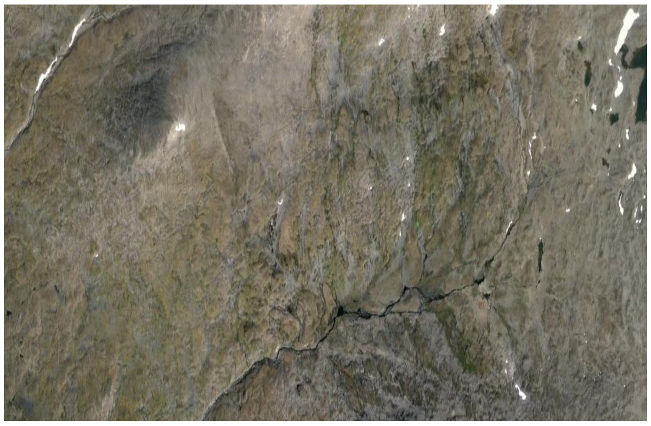
(a) Original image to be tested (b) Prediction using non parametric classifier

Figure 19: Example input and output using the parametric classifier proposed in the thesis.

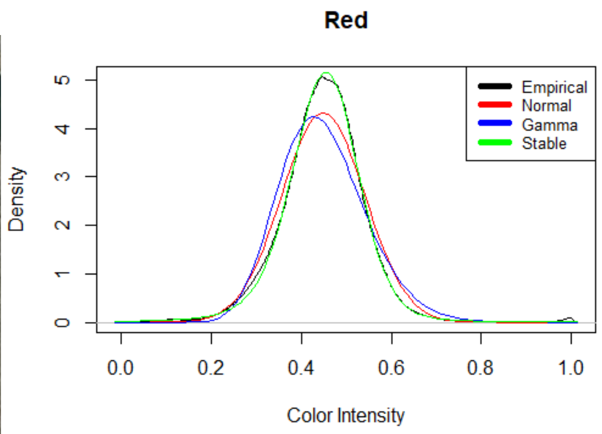
The algorithms we propose and test have also been made available in a small R package accompanying this thesis. While we are happy with the performance of the algorithms proposed and included in the package, the computation time of some functions is extremely long when performed on a regular laptop. This is especially true for the cross-validation performed when training the thresholds. However, this may not be as big of an issue in practice because the training should not have to be as extensive as in this thesis, or done many times, as discussed in Section 7.5.

The exploratory data analysis performed for this thesis showed promising results in use of the stable distribution to represent the color intensities of not only forest images, but also other terrains. Further, studying the Google Earth data used in this thesis and the results it provided gave us even more reason to believe that satellite data from Sentinel-2 will be a better fit for the algorithms we propose. We believe that our algorithms can perform even better with this data. Results using Sentinel-2 data, as well as comparisons to Bayesian algorithms developed, will be presented in the upcoming paper being written on this subject.

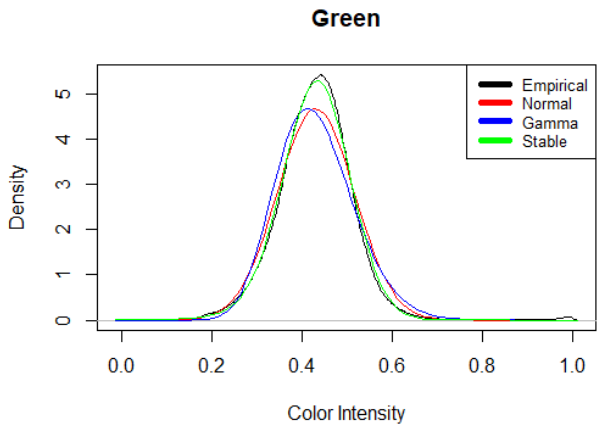
9 Appendix



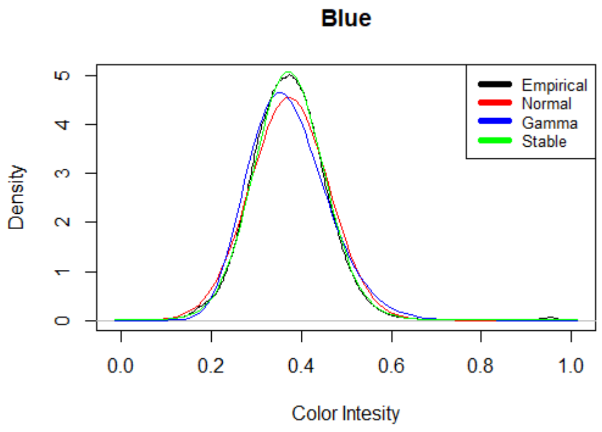
(a) Mountain image analysed



(b) Comparison of densities for red color intensity



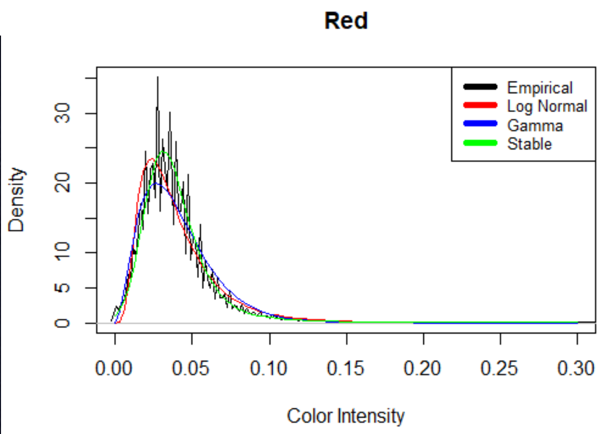
(c) Comparison of densities for green color intensity



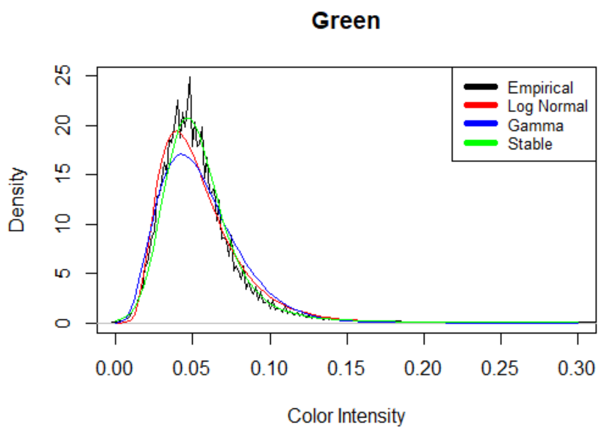
(d) Comparison of densities for blue color intensity

Figure 20: Plots of empirical and estimated probability density functions for distribution of color intensities in an image of mountain.

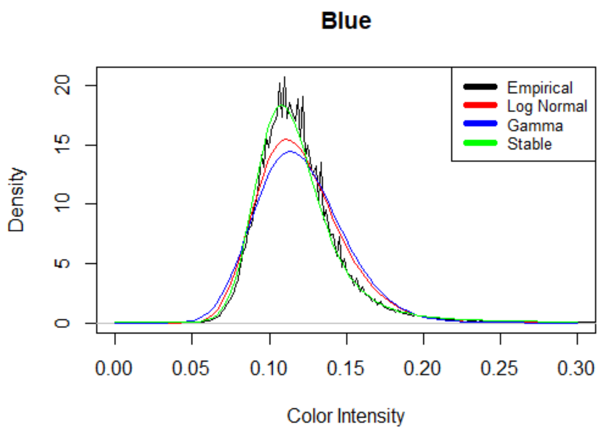
(a) Sea image analysed



(b) Comparison of densities for red color intensity



(c) Comparison of densities for green color intensity



(d) Comparison of densities for blue color intensity

Figure 21: Plots of empirical and estimated probability density functions for distribution of color intensities in an image of sea.

9.1 Stable density theorem

Define the following quantities

$$\zeta = \zeta(\alpha, \beta) = \begin{cases} -\beta \tan \frac{\pi\alpha}{2} & \text{if } \alpha \neq 1 \\ 0 & \text{if } \alpha = 1 \end{cases}$$

$$\theta_0 = \theta_0(\alpha,\beta) = \begin{cases} \frac{1}{\alpha}\arctan(\beta\tan\frac{\pi\alpha}{2}) & \text{if } \alpha \neq 1 \\ \frac{\pi}{2} & \text{if } \alpha = 1 \end{cases}$$

$$c_1(\alpha,\beta) = \begin{cases} \frac{1}{\pi}\left(\frac{\pi}{2} - \theta_0\right) & \text{if } \alpha < 1 \\ 0 & \text{if } \alpha = 1 \\ 1 & \text{if } \alpha > 1 \end{cases}$$

$$V(\theta;\alpha,\beta) = \begin{cases} (\cos\alpha\theta_0)^{\frac{1}{\alpha-1}}\left(\frac{\cos\theta}{\sin\alpha(\theta_0+\theta}\right)^{\frac{\alpha}{\alpha-1}}\frac{\cos(\alpha\theta_0+(\alpha-1)\theta)}{\cos\theta} & \text{if } \alpha \neq 1 \\ \frac{2}{\pi}\left(\frac{\frac{\pi}{2}+\beta\theta}{\cos\theta}\right)\exp\left(\frac{1}{\beta}\left(\frac{\pi}{2}+\beta\theta\right)\tan\theta\right) & \text{if } \alpha = 1, \beta \neq 0. \end{cases}$$

With these we are ready to state Theorem 2 from [16] which gives expressions for the density and distribution functions of standardized stable random variables. Because we are only considering standardized random variables with location parameter $\delta = 0$ and scale parameter $\sigma = 1$ these will be omitted from the notation. We will use $f(x;\alpha,\beta)$ and $F(x;\alpha,\beta)$ for the density and distribution function of a standardized stable random variable X.

Theorem 2. *Let X have characteristic function* (3)*. the density and distribution function of X are given by:*

(a) When $\alpha \neq 1$ and $x > \zeta$,

$$f(x;\alpha,\beta) = \frac{\alpha(x-\zeta)^{\frac{1}{\alpha-1}}}{\pi|\alpha-1|}\int_{-\theta_0}^{\frac{\pi}{2}} V(\theta;\alpha,\beta)\exp\left(-(x-\zeta)^{\frac{\alpha}{\alpha-1}}V(\theta;\alpha,\beta)\right)d\theta$$

and

$$F(x;\alpha,\beta) = c_1(\alpha,\beta) + \frac{\operatorname{sign}(1-\alpha)}{\pi}\int_{-\theta_0}^{\frac{\pi}{2}}\exp\left(-(x-\zeta)^{\frac{\alpha}{\alpha-1}}V(\theta;\alpha,\beta)\right)d\theta.$$

(b) When $\alpha \neq 1$ and $x = \zeta$,

$$f(\zeta;\alpha,\beta) = \frac{\Gamma\left(1+\frac{1}{\alpha}\right)\cos(\theta_0)}{\pi\left(1+\zeta^2\right)^{1/(2\alpha)}}$$

and

$$F(\zeta;\alpha,\beta) = \frac{1}{\pi}\left(\frac{\pi}{2}-\theta_0\right).$$

(c) When $\alpha \neq 1$ and $x < \zeta$,

$$f(x;\alpha,\beta) = f(-x;\alpha,-\beta)$$

and

$$F(x;\alpha,\beta) = 1 - F(-x;\alpha,-\beta).$$

(d) When $\alpha = 1$,

$$f(x:1,\beta) = \begin{cases} \frac{1}{2|\beta|}e^{-\frac{\pi x}{2\beta}}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} V(\theta;1,\beta)\exp\left(-e^{-\frac{\pi x}{2\beta}}V(\theta;1,\beta)\right)d\theta & \beta \neq 0 \\ \frac{1}{\pi(1+x^2)} & \beta = 0 \end{cases}$$

and

$$F(x;1,\beta) = \begin{cases} \frac{1}{\pi}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \exp\left(-e^{-\frac{\pi x}{2\beta}} V(\theta;1,\beta)\right) d\theta & \beta > 0 \\ \frac{1}{2} + \frac{1}{\pi}\arctan x & \beta = 0 \\ 1 - F(x;\alpha,-\beta) & \beta < 0. \end{cases}$$

References

Literature

[1] Anderson, T. W., Darling, D. A. (1952). Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *Annals of Mathematical Statistics, 23*, 193–212. doi:10.1214/aoms/1177729437

[2] Anderson, T. W., Darling, D. A. (1954). A Test of Goodness of Fit. *Journal of the American Statistical Association*, 49, 765-769, doi: 10.2307/2281537

[3] Baringhaus, L., Franz, C. (2004). On a new multivariate two-sample test. *Journal of Multivariate Analysis, 88*, 190-206. doi:10.1016/S0047-259X(03)00079-4

[4] Bhaskar, A., Mihaylova, LS., Achim, AM. (2010). Video foreground detection based on symmetric alpha-stable mixture models. *IEEE Transactions on Circuits and Systems for Video Technology, 20 (8)*, 1133 - 1138. doi:10.1109/TCSVT.2010.2051282

[5] Campos-Taberner, M., García-Haro, F., Camps-Valls, G., Grau-Muedra, G., Nutini, F., Busetto, L., ... Boschetti, M. (2017). Exploitation of SAR and Optical Sentinel Data to Detect Rice Crop and Estimate Seasonal Dynamics of Leaf Area Index. *Remote Sensing, 9(3)*, 248. doi:10.3390/rs9030248

[6] Cramér, Harald. (1928). On the composition of elementary errors: First paper: Mathematical deductions. *Scandinavian Actuarial Journal, 1928*(1), 13–74. doi:10.1080/03461238.1928.10416862

[7] FAO and UNEP. 2020. *The State of the World's Forests 2020. Forests, biodiversity and people.* Rome. doi:10.4060/ca8642en

[8] Fawcett, T. (2006). An Introduction to ROC Analysis. *Pattern Recognition Letters,* 27, 861–874. doi: 10.1016/j.patrec.2005.10.010

[9] Gnedenko, B. V., Kolmogorov, A. N. (1954). *Limit distributions for sums of independent random variables.* Cambridge, Mass: Addison-Wesley Pub. Co

[10] Hastie, T., Tibshirani, R., Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction. 2nd ed. New York: Springer

[11] Hossin, M., Sulaiman, M.N. (2015). A Review On Evaluation Metrics For Data Classification Evaluations. International Journal of Data Mining Knowledge Management Process, 5, 01-11. doi:0.5121/ijdkp.2015.5201

[12] Kharrat, T. & Boshnakov, G. (2014). StableEstim: An R Package for Estimating the Stable Laws Parameter and Running Monte Carlo Simulations. *Journal of Statistical Software.*

[13] Kogon, S. & Williams, D. (1998). Characteristic function based estimation of stable distribution parameters. *A practical guide to heavy tails: statistical techniques and applications.* 311–335 Birkhauser Boston Inc., USA,

[14] Koutrouvelis, I. (1980). Regression-Type Estimation of the Parameters of Stable Laws. *Journal of the American Statistical Association, 75(372)*, 918-928. doi:10.2307/2287182

[15] Mardia, K. V., Kent, J. T., Bibby, J. M. (1979). *Multivariate analysis.* London: Academic Press.

[16] Nolan, J. P. (1997). Numerical calculation of stable densities and distribution functions. *Comm. Statist. Stochastic Models*, 13, 759–774. doi: 10.1080/15326349708807450

[17] Oja, H., Randles, R. (2004). Multivariate Nonparametric Tests. *Statistical Science, 19(4)*, 598-605. doi: 10.1214/088342304000000558

[18] Ortega Adarme, M., Queiroz Feitosa, R., Nigri Happ, P., Aparecido De Almeida, C., Rodrigues Gomes, A. (2020). Evaluation of Deep Learning Techniques for Deforestation Detection in the Brazilian Amazon and Cerrado Biomes From Remote Sensing Imagery. *Remote Sensing, 12(6)*, 910. doi:10.3390/rs12060910

[19] Pagani, V., Guarneri, T., Busetto, L., Ranghetti, L., Boschetti, M., Movedi, E., Campos-Taberner, M., Garcia-Haro, F.J., Katsantonis, D., Stavrakoudis, D., Ricciardelli, E., Romano, F., Holecz, F., Collivignarelli, F., Granell, C., Casteleyn, S., Confalonieri, R., (2019). A high-resolution, integrated system for rice yield forecasting at district level. *Agricultural Systems 168*, 181–190. doi: 10.1016/j.agsy.2018.05.007

[20] Reiche, J., de Bruin, S., Hoekman, D., Verbesselt, J., Herold, M. (2015). A Bayesian Approach to Combine Landsat and ALOS PALSAR Time Series for Near Real-Time Deforestation Detection. *Remote Sensing, 7(5)*, 4973–4996. doi:10.3390/rs70504973

[21] Rosenbaum, P. (2005). An Exact Distribution-Free Test Comparing Two Multivariate Distributions Based on Adjacency. *Journal of the Royal Statistical Society. Series B, Statistical Methodology, 67(4)*, 515–530. doi:10.1111/j.1467-9868.2005.00513.x

[22] Samoradnitsky, G. (1994). *Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance (1st ed.).* Routledge. https://doi.org/10.1201/9780203738818

[23] Shermeyer, J., Haack, B. (2015). Remote sensing change detection methods to track deforestation and growth in threatened rainforests in Madre de Dios, Peru. *Journal of Applied Remote Sensing, 9.* doi:10.1117/1.JRS.9.096040.

[24] Székely, G. J. Rizzo, M. L. (2013). Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143, 1249 - 1272. doi:10.1016/j.jspi.2013.03.018

[25] Varmuza, K., Filzmoser, P. (2009). *Introduction to multivariate statistical analysis in chemometrics.* doi:10.1201/9781420059496

[26] Wang, C., Liao, M., Li, X. (2008). Ship Detection in SAR Image Based on the Alpha-stable Distribution. *Sensors, 8(8)*, 4948–4960. doi:10.3390/s8084948

[27] Wickham, H. (2019). Advanced r. CRC press

[28] Zolotarev, V. M. (1986). *One-dimensional stable distributions* (Vol. 65). American Mathematical Society, Providence, RI. ISBN: 0-8218-4519-5

Packages

[29] Carsten Franz (2019). cramer: Multivariate Nonparametric Cramer-Test for the Two-Sample-Problem. R package version 0.9-3. `https://CRAN.R-project.org/package=cramer`

[30] Ruth Heller, Dylan Small and Paul Rosenbaum (2012). crossmatch: The Cross-match Test. R package version 1.3-1. `https://CRAN.R-project.org/package=crossmatch`

[31] Robert J. Hijmans (2020). raster: Geographic Data Analysis and Modeling. R package version 3.4-5. `https://CRAN.R-project.org/package=raster`

[32] Tarak Kharrat and Georgi N. Boshnakov (2016). StableEstim: Estimate the Four Parameters of Stable Laws using Different Methods. R package version 2.1. `https://CRAN.R-project.org/package=StableEstim`

[33] Jesper Muren and Dmitry Otryakhin (2021). deforeStable: Classify jpeg images into forest or not forest using the color intensities of red, green and blue. R package version 0.1.0. Currently available at `https://github.com/Jmuren/Jesper-Muren-master-thesis-package`

[34] Jari Oksanen, F. Guillaume Blanchet, Michael Friendly, Roeland Kindt, Pierre Legendre, Dan McGlinn, Peter R. Minchin, R. B. O'Hara, Gavin L. Simpson, Peter Solymos, M. Henry H. Stevens, Eduard Szoecs and Helene Wagner (2020). vegan: Community Ecology Package. R package version 2.5-7. `https://CRAN.R-project.org/package=vegan`

[35] Maria Rizzo and Gabor Szekely (2021). energy: E-Statistics: Multivariate Inference via the Energy of Data. R package version 1.7-8. `https://CRAN.R-project.org/package=energy`

[36] Simon Urbanek (2019). jpeg: Read and write JPEG images. R package version 0.1-8.1. `https://CRAN.R-project.org/package=jpeg`

[37] Microsoft Corporation and Steve Weston (2020). doParallel: Foreach Parallel Adaptor for the 'parallel' Package. R package version 1.0.16. `https://CRAN.R-project.org/package=doParallel`