Cosmological field analysis with the principle of maximum entropy

THESIS

submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE
in
ASTRONOMY & MATHEMATICS

Author :	L.A.E. Prins
Student ID :	3302415
Supervisor :	dr. E. Sellentin
2nd corrector :	prof. dr. A. Doelman

Leiden, The Netherlands, June 13, 2025

Cosmological field analysis with the principle of maximum entropy

L.A.E. Prins

Leiden Observatory & Mathematical Institute, Universiteit Leiden
Einsteinweg 55, Leiden

June 13, 2025

Abstract

Cosmological fields can tell us many important things about the structure and evolution of the universe. These fields have complex structures, making them difficult to analyze with standard statistical methods. One method to estimate distributions is the principle of maximum entropy (MaxEnt). In this thesis, two questions were investigated: how can a MaxEnt distribution be found, and how well can the marginal distribution of cosmological fields be described with a MaxEnt distribution based on skewness and kurtosis?

To this end, two methods were implemented to find MaxEnt distributions. The first depends on solving a system of equations, but was unsuccessful due to the complex form of these equations. The second method works by minimizing a strictly convex function, and was found to be reliable and fast. The main limitation of this method arises from a numerical approximation of the function. This approximation can cause bad performance for certain parameters, and make it difficult to use many constraints. This method could be improved by making these approximations more accurate.

Using projection maps from the FLAMINGO cosmological simulations, MaxEnt distributions were found using the skewness and kurtosis of these maps. These distributions were not good approximations of the data due to the large right tails of such cosmological fields.

Overall, a MaxEnt distribution can be found efficiently by minimizing a strictly convex function, but this density is only useful when based on the right constraints.

Contents

Chapter 1

Introduction

Cosmological images can contain a wealth of information. For example, structures like the cosmic web show how galaxy clusters are positioned in thread-like filaments, with large areas of seemingly empty space in between. The way all this matter is distributed throughout the universe can tell us much about the evolution and structure of the universe, like the cosmic expansion rate, or the prevalence of dark matter and dark energy. In order to extract such knowledge from cosmological fields, reliable statistical techniques have to be used.

Due to the non-linear expansion of the universe, these fields have complex structures, consisting of large areas of mostly empty space, with very dense regions being few and far between. These complex images are often difficult to analyze using standard statistical techniques, and developing new methods to this end is an area of active research.

One technique to estimate distributions is the principle of maximum entropy (MaxEnt). It states that the density that best describes the current state of knowledge is a density which maximizes the entropy while satisfying this knowledge. Intuitively, the most general distribution is chosen which complies to the current knowledge. This knowledge, also called constraints, can for instance consist of the first four moments of a distribution.

In this thesis, an algorithm was developed to calculate MaxEnt distributions. This algorithm was applied to cosmological images from the FLAMINGO simulations. Using the skewness and kurtosis of those fields as constraints, MaxEnt distributions were determined with the aim to describe the marginal distribution of these cosmological fields.

Chapter 2 introduces the concept of entropy (2.1) and the principle of maximum entropy (2.2). Some instances of maximum entropy distributions are presented (2.3), along with several cases where no such distribution exists (2.4). In Chapter 3, two algorithms were developed that determine MaxEnt distributions. The first method attempted to solve a system of equations (3.1), while the second algorithm minimized a function (3.2). The performance of the second algorithm was tested on problems with a known solution, and on cases where the skewness and kurtosis is known. In Chapter 4, the second algorithm was applied to some cosmological fields from the FLAMINGO simulations. The results are discussed in Chapter 5, and Chapter 6 summarizes the findings.

Chapter 2

The principle of maximum entropy

Probability density estimation is a common problem in statistics: given prior knowledge about a distribution, how can the most likely density be reconstructed? An example of this problem is choosing a prior density in Bayesian statistics, where one often has some knowledge about results from previous experiments. Usually, many distributions could have given rise to the measured prior knowledge, and choosing one of those densities can be subject to subjective assumptions of the scientist.

For example, a common approach is to estimate a distribution by assuming the distribution is part of a parametric model, like the class of all normal distributions or exponential distributions. From this class, a particular distribution can be selected with parameter inference. However, the method is subjective, since the choice of the model often depends on the ideas of the scientist, and not solely on the data.

This subjectivity largely disappears if we adopt the non-parametric approach of choosing the density with the highest entropy among all distributions which satisfy the prior knowledge. This approach is called the principle of maximum entropy (MaxEnt).

Entropy is a widely used concept in many areas of science, and multiple interpretations exist. In general, two kinds of entropy exist: experimental entropy and informational entropy. The former describes the state of a system and is important in the theory of thermodynamics, while the latter is a property of probability distributions and originated in information theory. There are many similarities between the two forms of entropy, but both theories can be developed independently. This text only concerns itself with the latter notion.

The informational entropy roughly describes the inherent uncertainty of a distribution: if you take samples from a known distribution, how surprised will you be on average?

This Section begins by introducing the concept of entropy as a measure of uncertainty of a distribution (2.1). Then the principle of maximum entropy is defined, along with a general formula for MaxEnt distributions and two ways for finding these distributions (2.2). Afterwards, some examples of MaxEnt distributions are given (2.3). Lastly, three different cases are discussed were no MaxEnt distribution exists (2.4).

2.1 Entropy

First defined in 1948 by Claude Shannon [1, 2], entropy can be seen as the expected 'surprise' of a random experiment, given the underlying probabilities are known. Here, a derivation, inspired by Shannon, is presented of a function which captures this surprise in the case of discrete outcomes. The expectation of this surprise is defined as the entropy, and can be generalized to non-discrete cases.

Let X be a discrete random variable with range E and probability mass function p with $p_i := \mathbb{P}(X = i)$ for $i \in E$. Let $\mathcal{S} \subseteq E$ be the **support** of X, i.e. the set of outcomes with non-zero probability.

Suppose we want to find a function which measures how surprised we are when observing an outcome $i \in \mathcal{S}$. This **information function** or **surprisal** $I : (0, 1] \to \mathbb{R}_{\geq 0}$ should measure how surprised we are when observing an event with a certain probability. It should satisfy three intuitive properties:

(i) the function I should be strictly decreasing (if an event gets more likely, the surprise decreases);

(ii) $I(1) = 0$ (an event that always happens, is not surprising at all);

(iii) for two independent events with probabilities p_1 and p_2, the equality $I(p_1 \cdot p_2) = I(p_1) + I(p_2)$ should hold (measuring two independent events should be as as surprising as the sum of measuring them separately).

The only functions which satisfy these three properties are of the form $I(p) = c \log p$, where $c < 0$ is some negative constant. Changing c is equivalent to changing the base of the logarithm. Since only the relative changes in surprisal are important, this constant is arbitrary. Traditionally $c = -1$ is used, while the base may vary depending on the field of research.

In conclusion, $I(p_i) = -\log(p_i)$ is a measure for how surprised we are when observing some outcome $i \in \mathcal{S}$. The expected value of the surprisal is also called the Shannon entropy.

Definition 2.1. Given a discrete random variable X with probability mass function p and support $\mathcal{S}$, the **(Shannon) entropy** of X is defined as

$$H(X) := \mathbb{E}_{X\sim p}[I(p_X)] = -\sum_{i\in\mathcal{S}} p_i \log p_i.$$

Originally conceived to measure the amount of information in pieces of text, in particular bit strings, the base of the logarithm used by Shannon was 2. In the field of physics and astronomy, however, the natural logarithm is often preferred, and this will be used throughout the rest of the text.

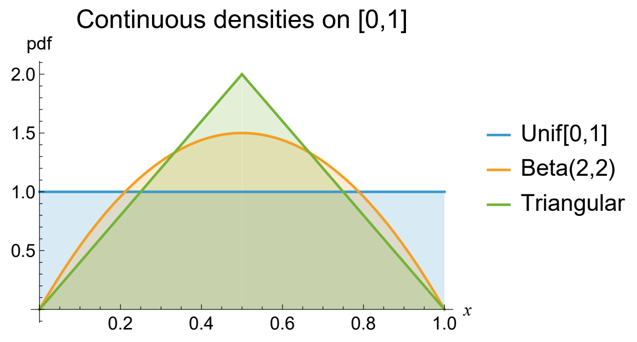
Likewise, the entropy of continuous random variables is given in Definition 2.2.

Definition 2.2. Given a continuous random variable X with probability density function (**pdf**) p and support $\mathcal{S}$, the **(differential) entropy** of X is defined as

$$h(X) := -\int_{\mathcal{S}} p(x) \log p(x) \mathrm{d}x.$$

This version of entropy for continuous random variables was not derived by Shannon. He just assumed that replacing a sum by an integral was a good choice. The differential entropy indeed has many similar properties as the Shannon entropy, but misses some intuitive properties that the Shannon entropy does have, such as non-negativity and invariance under change of variables. An alternate concept which does satisfy these properties is the limiting density of discrete points, introduced by E. T. Jaynes [3]. However, this version of entropy is hard to work with, so in practice Definition 2.2 is often used. The differential entropy is still quite useful and will be used in the rest of the text. From now on, only continuous random variables will be considered.

As an example of differential entropy, Figure 2.1 shows three continuous distributions on the interval $[0, 1]$. Their entropies are shown in Table 2.1. Among these, the uniform distribution has the highest entropy. This can be intuitively understood, since it is quite uncertain what its outcome will be, and in the the other distributions, one would expect the outcome to be closer to 0.5 on average. Furthermore, it can be proven that the uniform distribution has the highest entropy over all possible densities on $[0, 1]$. Thus it can be seen as being the most surprising distribution on this interval.



p	$h(p)$
Unif$[0,1]$	0
Beta$(2,2)$	-0.13
Triangular	-0.19

Figure 2.1 & Table 2.1: *Three continuous densities on the interval* $[0,1]$*, along with their entropy. The uniform density has the highest entropy, and can therefore be considered the 'most surprising'.*

2.2 Maximizing the entropy

As described in the beginning of this Chapter, a common problem in statistics is finding the probability distribution which gave rise to observed data. Since there is no one 'correct' answer, all approaches have to make use of assumptions or subjective interpretations.

One possible solution to this problem is the principle of maximum entropy: among all distributions which could have given rise to our data, choose the one with the highest entropy. Intuitively, this principle states that, among all possible distributions which could describe our data, we should choose the most general one, i.e. the one that makes the least amount of assumptions. Definition 2.3 formalizes this idea.

Definition 2.3. Let X be a random variable with pdf q and support $\mathcal{S}$, and $k \in \mathbb{N}_0$ a number. Let $f_i : \mathbb{R} \to \mathbb{R}$ and $F_i \in \mathbb{R}$ be functions and real numbers such that $F_i = \mathbb{E}_{X\sim q}[f_i(X)]$ for $i = 1, \dots, k$. The **principle of maximum entropy** states the best estimate for q is the non-negative function p which maximizes the entropy

$$h(p) = -\int_{\mathcal{S}} p(x) \log p(x) \, \mathrm{d}x$$

subject to the constraints

$$\int_{\mathcal{S}} p(x) \, \mathrm{d}x = 1, \tag{2.1}$$

$$\int_{\mathcal{S}} p(x) f_i(x) \, \mathrm{d}x = F_i, \qquad i = 1, \dots, k. \tag{2.2}$$

A density p which satisfies these conditions, is called a **maximum entropy distribution**, or **MaxEnt distribution** for short. The equalities in Equation 2.2 are called **constraints**.

In later Chapters, it's useful to use vector notation indicate the constraints. So we write $\boldsymbol{F} = (F_1, \ldots, F_k)$ and $\boldsymbol{f} = (f_1, \ldots, f_k)$.

Note that, when using this principle instead of the more classical parameter inference setting as described above, different assumptions have to be made. Instead of assuming a parametric model, it is assumed that certain expectations of functions of X are known. In particular, no uncertainties on these expectations can be taken into account. Therefore, it is important that these constraints are accurate, which can be ensured with sufficiently large sample sizes.

One MaxEnt distribution has already been discussed: the uniform distribution on $[0, 1]$. Except normalization constraint 2.1, there are no constraints. So with $k = 0$ and support $\mathcal{S} = [0, 1]$, it can be shown that the highest entropy is attained by the uniform distribution. Other more interesting examples of MaxEnt distributions are given in Section 2.3.

Assuming a solution exists, the MaxEnt density can be shown to have a particular form.

Theorem 2.1. Given a MaxEnt distribution p exists, it can be written in the following way:

$$p(x) = Z(\lambda_1, \ldots, \lambda_k)^{-1} \exp\left(\sum_{i=1}^{k} \lambda_i f_i(x)\right), \tag{2.3}$$

where

$$Z(\lambda_1, \ldots, \lambda_k) := \int_{\mathcal{S}} \exp\left(\sum_{i=1}^{k} \lambda_i f_i(x)\right) \mathrm{d}x$$

and $\lambda_1, \ldots, \lambda_k$ are such that

$$F_i = \frac{\delta}{\delta \lambda_i} \log Z(\lambda_1, \ldots, \lambda_k), \qquad i = 1, \ldots, k. \tag{2.4}$$

The proof relies on the method of Lagrange multipliers and functional derivation [4, 5]. In the context of this text, the variables $\lambda_1, \ldots, \lambda_k$, which give a distribution via Equation 2.3, will be called the **Lagrange parameters**. For short, the vector notation $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_k)$ is used. Theorem 2.1 states the maximization problem is equivalent to the problem of solving a system of k equations with k variables. If there does not exist a MaxEnt distribution, the system of equations is often not solvable.

Note that for $k = 0$, one gets the uniform distribution if the convention is used that the empty sum evaluates to 0. In the rest of the text, we assume $k \geq 1$.

The MaxEnt density in Equation 2.3 can be formulated in an equivalent way. Multiplying both sides of the fraction in Equation 2.3 by $\exp(-\sum_i \lambda_i F_i)$ gives an alternative formulation of the MaxEnt distribution:

$$p(x) = Q(\lambda_1, \dots, \lambda_k)^{-1} \exp\left(\sum_{i=1}^{k} \lambda_i (f_i(x) - F_i)\right), \tag{2.5}$$

where Q is a normalization constant:

$$Q(\lambda_1, \dots, \lambda_k) := \int_{\mathcal{S}} \exp\left(\sum_{i=1}^{k} \lambda_i (f_i(x) - F_i)\right) \mathrm{d}x. \tag{2.6}$$

This may seem like a trivial reformulation of the solution. However, it is useful to consider Q instead of Z, and Theorem 2.2 tells us why. The Theorem is inspired by [5], where only the case $f_i(x) = x^i$ was considered.

Theorem 2.2. If f_i are continuously differentiable functions on $\mathcal{S}$, the Lagrange parameters $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_k)$ solve the system of equations 2.4 precisely when they minimize Q.

Proof. Suppose $\dfrac{\partial Q}{\partial \lambda_m} = 0$ for all $m = 1, \dots, k$. If the functions f_i are continuously differentiable on $\mathcal{S}$ for all i, the derivative and integral can be interchanged using the Leibniz integral rule [6, p. 422]. This gives the following expression for the derivative:

$$\frac{\partial Q}{\partial \lambda_m} = \int_{\mathcal{S}} (f_m(x) - F_m) \exp\left(\sum_{i=1}^{k} \lambda_i (f_i(x) - F_i)\right) \mathrm{d}x.$$

Since Q is a positive constant with respect to x, both sided can be divided by Q to see that,

$$0 = \int_{\mathcal{S}} (f_m(x) - F_m) p(x) \mathrm{d}x, \qquad m = 1, \dots, k,$$

which is equivalent to the constraints in Equation 2.2. Therefore, at an extreme value of Q, the corresponding Lagrange parameters give a MaxEnt distribution via Equation 2.3.

In fact, Q has at most one extreme value, and it is a minimum. To see this, consider the second partial derivatives:

$$\frac{\partial^2 Q}{\partial\lambda_m \partial\lambda_n} = \int_S (f_m(x) - F_m)(f_m(x) - F_n) \exp\left(\sum_{i=1}^{k} \lambda_i (f_i(x) - F_i)\right) \mathrm{d}x.$$

Dividing by Q again gives the following:

$$\frac{\partial^2 Q(\boldsymbol{\lambda})}{\partial\lambda_m \partial\lambda_n} \frac{1}{Q(\boldsymbol{\lambda})} = \int_S (f_m(x) - F_m)(f_m(x) - F_n) p(x) \mathrm{d}x,$$

which is the covariance of $f_m(X)$ and $f_n(X)$. The matrix

$$\left(\frac{\partial^2 Q(\boldsymbol{\lambda})}{\partial\lambda_m \partial\lambda_n} \frac{1}{Q(\boldsymbol{\lambda})}\right)_{1 \leq m,n \leq k}$$

is therefore a covariance matrix. This implies it's symmetric and positive semi-definite. For properly chosen constraints, a covariance matrix is positive definite. Multiplying the matrix with the positive value $Q(\boldsymbol{\lambda})$ gives the Hessian of Q evaluated in $\boldsymbol{\lambda}$, which then is positive definite, as well. Since the Hessian is positive definite for every value of $\boldsymbol{\lambda}$, Q is strictly convex. This implies that there exists at most one minimum of Q. $\square$

Since its partial derivatives are often considered, the function Q will be referred to as a **potential function**. Note that the proof assumes Q has an extreme value. If a MaxEnt distribution exists, it is unique.

In conclusion, there are two ways to find the Lagrange parameters $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_k)$ which give a MaxEnt distribution by way of Equation 2.3 or 2.5: solve a system of equations, or minimize a strictly convex potential function Q. In Chapter 3, both of these approaches are attempted.

2.3 Some MaxEnt distributions

Despite the complicated forms of Equations 2.3 and 2.4, many commonplace distributions can be found using the principle of maximum entropy. As discussed before, for a finite support $\mathcal{S} = [l, u]$ and no constraints, the MaxEnt distribution is the uniform distribution on $[l, u]$. For support $\mathcal{S} = \mathbb{R}$ and a specified mean μ and variance σ^2, the MaxEnt distribution is the normal distribution $\mathcal{N}(\mu, \sigma^2)$ [7].

More examples of continuous MaxEnt distributions are shown in Table 2.2. In the Table, it can be seen that the constraints giving rise to some distributions are relatively simple: on the positive reals, a specified mean leads to an exponential distribution. Some slightly more complicated restraints give rise to other densities, such as the Pareto or von Mises distributions.

2.4 Existence of MaxEnt distributions

Sadly, there does not always exist a MaxEnt distribution. Three distinctions can be made. Let $\mathcal{C}$ be the class of all pdfs p which satisfy the constraints of Equation 2.2, and denote by

$$h(\mathcal{C}) := \{h(p) : p \in \mathcal{C}\}$$

the set of all entropies attained by elements of $\mathcal{C}$.

The first case is that the set $\mathcal{C}$ could be empty: no distribution would exist that satisfies all constraints. For example, the mean of X can be constrained as being both 0 and 1. In practice, this is easy to identify and avoid.

Secondly, $h(\mathcal{C})$ could have no upper bound, in which case the entropy could become arbitrarily large. Consider the case with possible support $\mathbb{R}$ and one constraint: $\mathbb{E}\left[X\right] = 0$. To see this, consider the uniform density $p_a = \text{Unif}[-a, a]$ for $a > 0$. The entropy of p_a is $h(p_a) = \log(2a)$, which goes to infinity as a increases. Such a case could be remedied by adding extra constraints, or decreasing the size of the support.

Lastly, and perhaps most interestingly, $h(\mathcal{C})$ could have an upper bound, but no maximum; the supremum of $h(\mathcal{C})$ could then be approached arbitrarily well, but no distribution's entropy would achieve this value. An example is the problem with the first three moments as constraints [9, Ch. 12.3].

k	Distribution	Parameters	Support	Pdf $p(x)$	Constraints	$\boldsymbol{\lambda}$
0	Uniform	$l, u \in \mathbb{R}$ $l < u$	$[l, u]$	$\frac{1}{u-l}$	–	–
1	Normal	$\mu \in \mathbb{R}$ $\sigma^2 > 0$	$\mathbb{R}$	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\mathbb{E}\left[(X-\mu)^2\right] = \sigma^2$	$-\frac{1}{2\sigma^2}$
	Exponential	$c > 0$	$\mathbb{R}_{\geq 0}$	ce^{-cx}	$\mathbb{E}\left[X\right] = 1/c$	$-c$
	Laplace	$\mu \in \mathbb{R}$ $b > 0$	$\mathbb{R}$	$\frac{1}{2b} e^{-\frac{\lvert x-\mu \rvert}{b}}$	$\mathbb{E}\left[\lvert X-\mu \rvert\right] = b$	$1/b$
	Pareto	$\alpha, x_{\mathrm{m}} > 0$	$[x_{\mathrm{m}}, \infty)$	$\frac{\alpha x_{\mathrm{m}}^{\alpha}}{x^{\alpha+1}}$	$\mathbb{E}\left[\log X\right] = \frac{1}{\alpha} + \log(x_{\mathrm{m}})$	$-\alpha - 1$
	Cauchy	–	$\mathbb{R}$	$\frac{1}{\pi(1+x^2)}$	$\mathbb{E}\left[\log 1 + X^2\right] = 2\log 2$	-1
2	Normal	$\mu \in \mathbb{R}$ $\sigma^2 > 0$	$\mathbb{R}$	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\mathbb{E}\left[X\right] = \mu$ $\mathbb{E}\left[X^2\right] = \sigma^2 + \mu^2$	$\frac{\mu}{\sigma^2}$ $-\frac{1}{2\sigma^2}$
	Lognormal	$\mu \in \mathbb{R}$ $\sigma^2 > 0$	$\mathbb{R}_{\geq 0}$	$\frac{1}{\sqrt{2\pi\sigma^2}x} e^{-\frac{(\log x-\mu)^2}{2\sigma^2}}$	$\mathbb{E}\left[\log X\right] = \mu$ $\mathbb{E}\left[\log(X)^2\right] = \sigma^2 + \mu^2$	$\frac{\mu}{\sigma^2} - 1$ $-\frac{1}{2\sigma^2}$
	Von Mises	$\mu \in \mathbb{R}$ $\kappa > 0$	$[0, 2\pi)$	$\frac{1}{2\pi I_0(\kappa)} e^{\kappa\cos(x-\mu)}$	$\mathbb{E}\left[\cos X\right] = \frac{I_1(\kappa)}{I_0(\kappa)}\cos\mu$ $\mathbb{E}\left[\sin X\right] = \frac{I_1(\kappa)}{I_0(\kappa)}\sin\mu$	$\kappa\cos\mu$ $\kappa\sin\mu$
	Rayleigh	$\sigma^2 > 0$	$\mathbb{R}_{\geq 0}$	$\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$	$\mathbb{E}\left[X^2\right] = 2\sigma^2$ $\mathbb{E}\left[\log X\right] = \frac{\log(2\sigma^2) - \gamma_E}{2}$	$-\frac{1}{2\sigma^2}$ 1
	Gamma	$k, \theta > 0$	$\mathbb{R}_{\geq 0}$	$\frac{x^{k-1}e^{-x/\theta}}{\Gamma(k)\theta^k}$	$\mathbb{E}\left[X\right] = k\theta$ $\mathbb{E}\left[\log X\right] = \psi(k) + \log\theta$	$-\frac{1}{\theta}$ $k-1$

Table 2.2: *Various continuous probability distributions with their MaxEnt constraints and Lagrange parameter $\boldsymbol{\lambda}$. Among all densities on a given support which satisfy the constraints, that distribution is the one with the highest entropy. The Lagrange parameter $\boldsymbol{\lambda}$ gives rise to the pdf via Equation 2.3. k indicates the amount of constraints. Here, I_j is the modified Bessel function of the first kind of order j, γ_E the Euler-Mascheroni constant, Γ the Gamma function, and ψ the digamma function [7, 8].*

Chapter 3

Methods for finding a maximum entropy distribution

Two approaches for finding a MaxEnt distribution were discussed in the previous Chapter. The first method solves system of equations 2.4, while the second minimizes the potential function Q defined in Equation 2.6.

In Section 3.1, the first method was implemented in Mathematica and solved many MaxEnt problems with one constraint (3.1.1), but failed when generalizing to two constraints (3.1.2). Section 3.2 describes the second algorithm, which was written in Python (3.2.1). Its performance was tested on a MaxEnt distribution (3.2.2), and on the case with skewness and kurtosis as constraints (3.2.3). Section 3.3 briefly summarizes the finding of the Chapter. All code, as well as most results presented in this thesis, can be found on the GitHub page in [10].

3.1 Solving a system of equations

Given that the constraints admit a MaxEnt distribution, its form can be found by solving system of equations 2.4. These equations can take on complex forms. Using the programming language Wolfram Mathematica, a notebook was created that attempts to solve this set of equations. Mathematica is a powerful programming language equipped with many built-in tools to solve mathematical problems, especially related to calculus.

Algorithm 1 outlines the structure and simplicity of the code. The Algorithm merely defines the necessary equations, and in step 3 the '*Solve*' function is called on the system of equations.

The difficulty lies in providing the right *assumptions* to the Algorithm.

Algorithm 1: Find a MaxEnt distribution by solving the system of equations 2.4, written for Mathematica. The argument *assumptions* is a Boolean statement defining the constraints on the parameters belonging to F_i and f_i, and specifying the Lagrange parameters belonging to $\mathbb{R}$. '$\cdot$' denotes the inner product.

```
Input: $k \in \mathbb{N}_1$;
        $\boldsymbol{F} = (F_1, \ldots, F_k) \in \mathbb{R}^k$;
        $\boldsymbol{f} = (f_1, \ldots, f_k) : \mathbb{R} \to \mathbb{R}^k$;
        $\mathcal{S} = [a, b]$ with $a, b \in [-\infty, \infty]$;
        $assumptions$
Output: Lagrange parameter $\boldsymbol{\lambda} \in \mathbb{R}^k$
1 Define $Z(\boldsymbol{\lambda}, assumptions) := \int_{\mathcal{S}} \exp(\boldsymbol{\lambda} \cdot \boldsymbol{f}(x))\mathrm{d}x$, where
    $assumptions$ are used when evaluating the integral.
2 Define $dZ(\boldsymbol{\lambda}, assumptions) := \left(\frac{\partial \log Z(\boldsymbol{\lambda}, assumptions)}{\partial \lambda_i}\right)_{1 \le i \le k}$.
3 Solve $dZ(\boldsymbol{\lambda}, assumptions) = \boldsymbol{F}$ over $\boldsymbol{\lambda}$ given $assumptions$ are true.
```

This argument consists of Boolean statements such as $\sigma^2 > 0$ or $\lambda_1 \in \mathbb{R}$. In practice, one always needs to specify all normal and Lagrange parameters are real numbers to get sensible results. Often, it's also necessary (or time-efficient) to specify some Lagrange parameter is positive or negative. This can be necessary to let Mathematica perform calculations. For example, to evaluate the integral $\int_{\mathbb{R}} \exp(ax^2)\mathrm{d}x$, it's necessary to know $a < 0$. Note there has to be at least one constraint for the Algorithm to work.

3.1.1 One constraint

All first four distributions with one constraint in Table 2.2 were found this way, with run times varying between 0.10 and 10 s. Plugging the result $\boldsymbol{\lambda}$ into Equation 2.3 The resulting distributions were exact, and no numerical approximations had to be made.

Other MaxEnt densities with one constraint could also be found. Figure 3.1 shows a MaxEnt density arising from the positive real support $\mathbb{R}_{\ge 0}$ and a fixed third moment: $\mathbb{E}\left[X^3\right] = c$ with $c > 0$.

However, this approach did not succeed for the Cauchy distribution. In this case, the system of equations could be solved, but the program could not prove that this solution was the only one. This caused the program to quit and give an error message. Using the function '*FindInstance*', which only searches for one solution, solved this issue. However, this approach

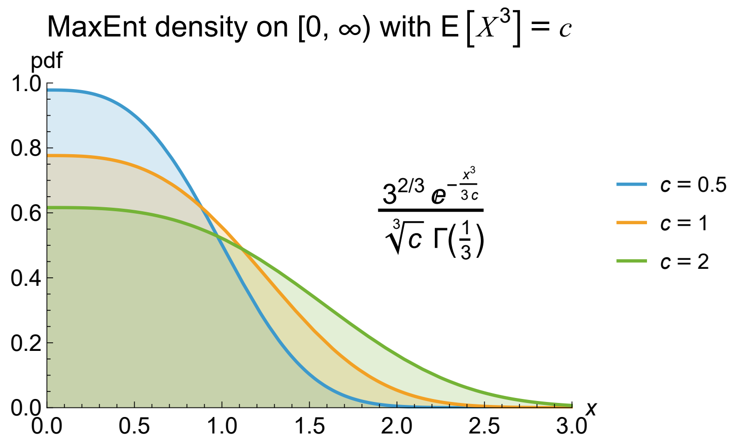


Figure 3.1: *The MaxEnt distribution arising from support $\mathbb{R}_{\geq 0}$ and constraint $\mathbb{E}\left[X^3\right] = c$. The resulting Lagrange parameter of Algorithm 1 was converted into a density via Equation 2.3. The closed-form equation for the solution for is shown on the plot, along with three example densities for fixed c.*

only worked because the Cauchy distribution has no parameters.

The Cauchy distribution is usually a curious case in probability; its mean and variance are undefined due to the divergence of their integrals. Therefore, many standard theorems in probability theory, such as the Central Limit Theorem, do not apply to this distribution. This abnormal behavior of the distribution makes it not very surprising that the program had issues in that case.

Since Algorithm 1 worked well in most cases with one constraint, the next step was to generalize to two constraints.

3.1.2 Multiple constraints

It was attempted to apply Algorithm 1 to the distributions with two constraints in Table 2.2. While it performed well for the normal and lognormal distribution, the other distributions couldn't be found in this way; the system of equations became too complex for the the '*Solve*' function, and the program quit with an error message. It was attempted to provide a solving method to the '*Solve*' function, but this also didn't improve performance.

Since for practical purposes, the objective wasn't necessarily to find closed-form solutions, the problem was relaxed; the values for parameters could be fixed before finding the MaxEnt distribution. The solution could then be achieved using the '*NSolve*' function. This way, the gamma distribution could be found within about 30 s. However, other problems with two constraints could not be solved, providing an error message. Since the envisioned application of finding MaxEnt distributions was more complex than these cases, the approach with Algorithm 1 was discontinued.

Unless more efficient solving-methods can be found, it is not feasible to calculate or approximate complex MaxEnt densities by solving system of equations 2.4.

3.2 Minimizing a potential

Since solving the system of equations was not feasible with Mathematica, a different approach was used to find the MaxEnt distribution. Here, the potential function Q (Eq. 2.6) is minimized using Newton's method for optimization in Python. This approach was inspired by Rockinger & Jondeau [5]. However, it was slightly adapted; Rockinger & Jondeau only considered the first k moments as constraints, i.e. $f_i(x) = x^i$ for $i = 1, \ldots, k$. The algorithm implemented here allows for more general constraint functions.

3.2.1 The algorithm

As discussed in Section 2.2, another way to find the MaxEnt distribution is to minimize the potential Q, defined in Equation 2.6. Theorem 2.2 states that if a solution exists, this minimum is unique. This makes Newton's method for optimization an appropriate way of finding this minimum. See [11, Sect. 3.1] for some background on the technique. Algorithm 2 implements this, and attempts to find the Lagrange parameters which minimize the potential.

In order to apply this Algorithm, two approximations have to be made for practical purposes. First, to efficiently calculate the gradient and Hessian of Q, the integral over $\mathcal{S}$ in the definition of Q is approximated by a sum using an n-point Gaussian quadrature. Furthermore, the support $\mathcal{S}$ is approximated with a finite support of the form $[l, u]$, which is necessary for the n-point Gaussian quadrature.

The **n-point Gaussian quadrature** is numerical technique for evaluating integrals by approximating an integral over the interval $[-1, 1]$ with a sum over n elements. The technique uses so-called **weights** $w_1, \ldots, w_n$ and **nodes** $z_1, \ldots, z_n \in [-1, 1]$. Their values are such that, for some integrable function g,

$$\int_{-1}^{1} g(z)\mathrm{d}z \approx \sum_{i=1}^{n} w_i g(z_i).$$

The values of the nodes and weights do not depend on the function which is integrated. As a consequence of using the Gaussian quadrature, the support $\mathcal{S}$ has to be approximated with a finite interval of the form $[a, b]$. Hence, the potential is approximated as

$$Q(\boldsymbol{\lambda}) \approx \sum_{i=1}^{n} w_i \exp\left(\sum_{j=1}^{k} \lambda_j (f_j(x_i) - F_j)\right), \tag{3.1}$$

where x_i is a linearly rescaled value of z_i, also called a node. The n-point Gaussian quadrature is useful for quickly and easily calculating the gradient and Hessian of Q.

This approximation of Q is minimized in Algorithm 2. The first step defines the nodes and weights, whose values can be found using numerical packages. Step 2 rescales the nodes linearly from the interval $[-1, 1]$ to $[l, u]$. Next, the matrix A is defined to recast Equation 3.1 into matrix-form, and then the potential is defined in step 4. Next, the iterator and the Lagrange parameters are initialized to zero.

For each iteration step, the gradient and Hessian of Q are evaluated in $\boldsymbol{\lambda}^i$. In this case, the automatic differentiation technique was used with the '*autograd*' package in Python. Then a step $\boldsymbol{d}$ is calculated by solving a linear system of equations, and $\boldsymbol{\lambda}^i$ is updated according to Newton's method. This procedure is repeated until either the step size becomes very small, in which case the algorithm is said to **converge**, or until the maximum number of iterations has been reached.

If convergence is reached, the output of Lagrange parameters then identify a MaxEnt distribution via Equation 2.3.

3.2.2 Performance on constraints with a known solution

Once the constraints have been set, several other parameters should be given to Algorithm 2: the quadrature number n, the maximum amount of iterations i_{max}, and the borders of the closed interval $[l, u]$. This raises the question of how these parameters should be chosen. Intuitively, one may expect that the larger the chosen support and the higher the quadrature number n, the more accurate the result. As shown below, however, this is not always the case.

Algorithm 2: Minimize the potential Q with an n-point Gaussian quadrature and Newton's method for optimization. Convergence is said to be reached when the condition in step 11 is satisfied. In case of convergence, the resulting Lagrange parameters give a MaxEnt distribution via Equation 2.3. The exponent 'exp' acts element-wise on vectors. The Euclidian norm is denoted by $||\cdot||_2$.

```
Input: $n, k, i_{\max} \in \mathbb{N}_1$;
       $[l, u]$ with $l, u \in \mathbb{R}, l < u$;
       $\boldsymbol{F} = (F_1, \ldots, F_k) \in \mathbb{R}^k$;
       $\boldsymbol{f} = (f_1, \ldots, f_k) : \mathbb{R} \to \mathbb{R}^k$
Output: Lagrange parameters $\boldsymbol{\lambda} \in \mathbb{R}^k$
1  Let $\boldsymbol{z} = (z_1, \ldots, z_n)$ and $\boldsymbol{w} = (w_1, \ldots, w_n)$ be the nodes and
     weights of the $n$-point Gaussian quadrature.
2  Set $x_i := ((u-l)z_i + (u+l))/2$ for $i = 1, \ldots, n$.
3  Define the matrix $A = (a_{ij})_{1\le i\le k, 1\le j\le n}$ with $a_{ij} := f_j(x_i) - F_j$.
4  Define the potential function $Q(\boldsymbol{\lambda}) := \boldsymbol{w} \cdot \exp(A\boldsymbol{\lambda})$.
5  Initialize $i := 0$ and $\boldsymbol{\lambda}^0 := \boldsymbol{0} \in \mathbb{R}^k$.
6  while $i < i_{max}$ do
7      Let $\boldsymbol{g^i} := \nabla Q(\boldsymbol{\lambda}^i)$ be the gradient, and $G^i := H_Q(\boldsymbol{\lambda}^i)$ the
         Hessian of $Q$ evaluated in $\boldsymbol{\lambda}^i$.
8      Let $\boldsymbol{d}^i \in \mathbb{R}^k$ be the solution to the system of linear equations
         $G^i\boldsymbol{d}^i = -\boldsymbol{g}^i$.
9      $\boldsymbol{\lambda}^{i+1} := \boldsymbol{\lambda}^i + \boldsymbol{d}^i$
10     $i$++
11     if $||\boldsymbol{d}^i||_2 < 10^{-9}$ then
12         STOP by setting $i_{\max} := i$.
```

The value $i_{\max}$ that should be used generally depends on the number of constraints k. In the case of one or two constraints, given the Algorithm doesn't diverge, converges within 10 to 100 iterations. As a consequence, the influence of this parameter does not need to be investigated here, and by default is set to 100 for one or two constraints.

Example: the Laplace distribution

To illustrate the implemented Algorithm, consider the Laplace distribution with parameter $c = 1$. Then the inputs for the algorithm are $k = 1$, $i_{\max} = 100$, $F_1 = 1$, and $f_1(x) = |x|$.

Since the distribution is symmetric around 0, a logical choice for the support interval is $[-u, u]$ for some $u > 0$. It's obvious that u should be chosen such that most of the mass of the distribution is contained in that interval. But the interval shouldn't be too large, either, as can bee seen in Figure 3.2. A bigger interval resulted in a worse result.

Both of these examples, however, did converge to a solution within about ten iterations. When increasing to $u = 40$, the algorithm diverged; see Figure 3.3.

Performance plots

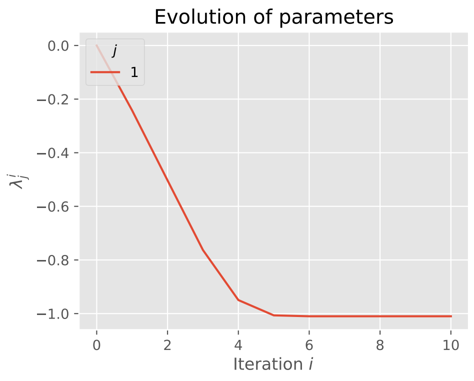
Most distributions in Table 2.2 have infinite support. To apply Algorithm 2, a closed interval has to be fixed. To see how the size of the interval influences the results, the Algorithm was tested for various supports and values for n. Here the interval had the form $[-u, u]$ (for real support $\mathbb{R}$), or $[a, u]$, $a \in \mathbb{R}$ (for real support $[a, \infty]$) is used, for varying u. For each instance, three things were evaluated:

(i) how much of the mass of the 'real' distribution fell within the interval $[l, u]$;

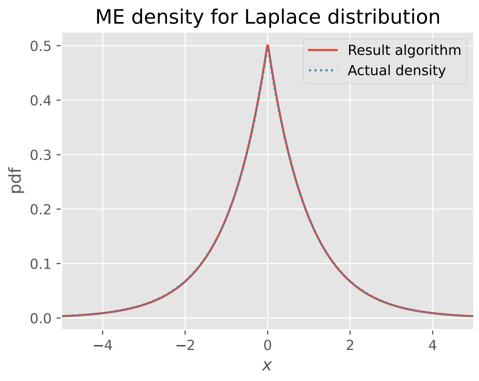
(ii) whether the algorithm converged;

(iii) given that the algorithm converges, how far the solution $\boldsymbol{\lambda}$ was from the theoretical value. This error is denoted by λ_{err}, the Euclidean distance between theoretical value and $\boldsymbol{\lambda}$.

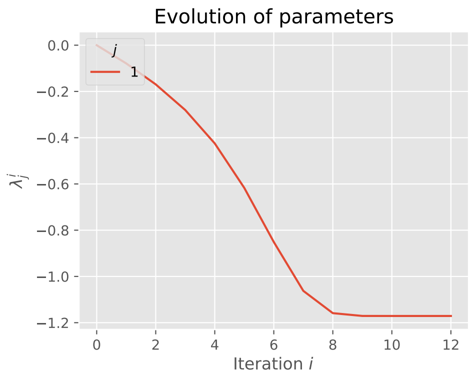
The resulting performance plots are shown in Figure 3.4. In each Figure, the same data is visualized in two plots. The contour plot on the left shows the areas of non-convergence well, while the scatter plot on the right shows the influence of the support size and n more accurately.



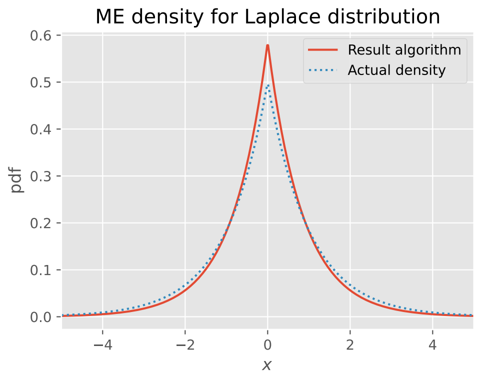
(a) *Lagrange parameter vs. iteration using support* $[-8, 8]$. *The resulting parameter is* $\lambda = -1.01$.



(b) *The resulting MaxEnt density using support* $[-8, 8]$ *with* $\lambda = -1.01$.



(c) *Lagrange parameter vs. iteration using support* $[-20, 20]$. *The resulting parameter is* $\lambda = -1.17$.



(d) *The resulting MaxEnt density using support* $[-20, 20]$ *with* $\lambda = -1.17$.

Figure 3.2: *Two visualizations of Algorithm 2 and resulting densities using the Laplace constraint* $\mathbb{E}[|X - \mu|] = 1/c$ *with* $\mu = 0$ *and* $c = 1$, *varying support and* $n = 50$. *The correct Lagrange parameter is* $\lambda = -1$.

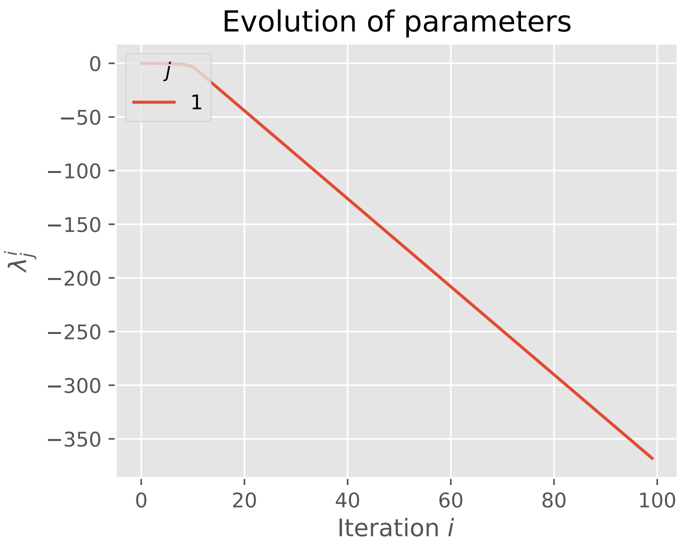
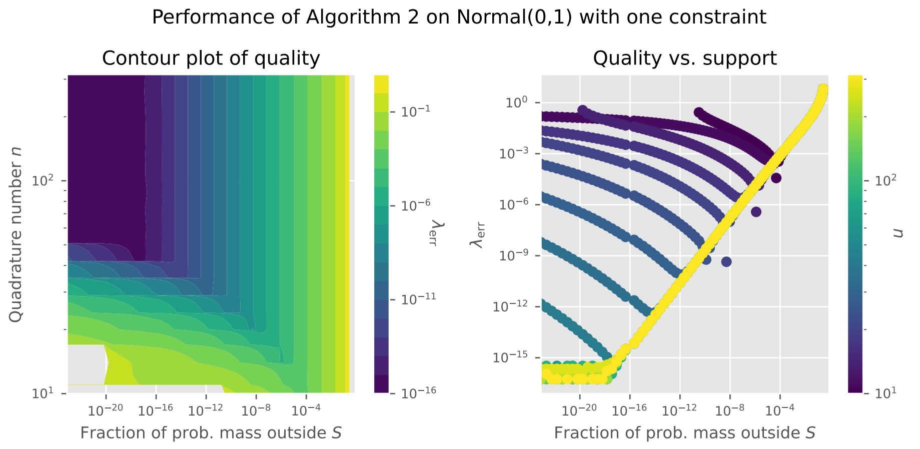
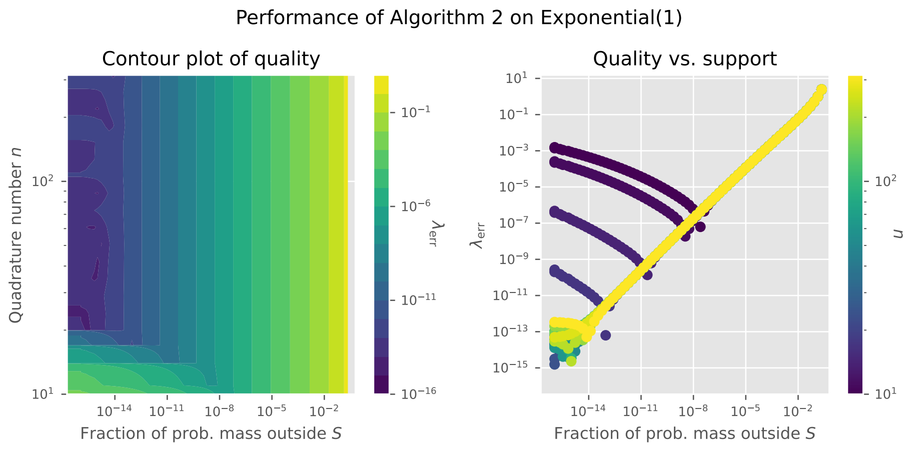


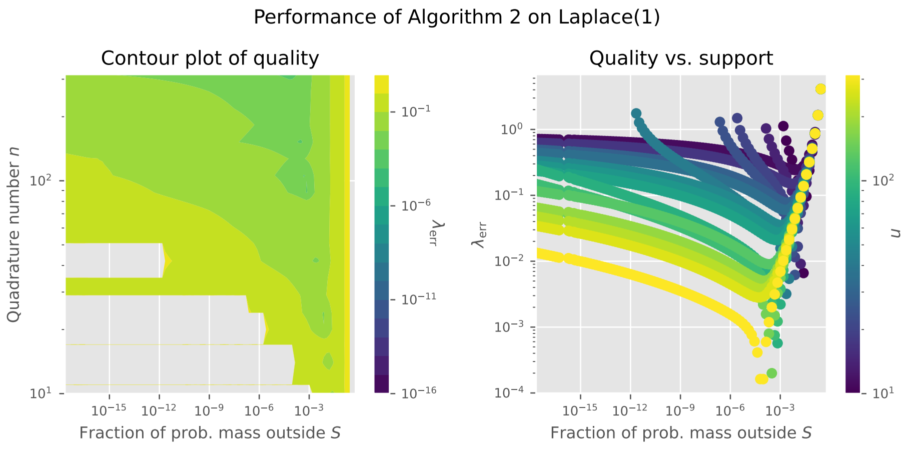
Figure 3.3: *Algorithm 2 does not always converge to a solution. The evolution of the Lagrange parameter is shown during Algorithm 2 on the Laplace constraint* $\mathbb{E}\left[|X-\mu|\right] = 1/c$ *with* $\mu = 0$ *and* $c = 1$*, support* $[-40, 40]$ *and* $n = 50$*. The correct Lagrange parameter is* $\lambda = -1$*.*



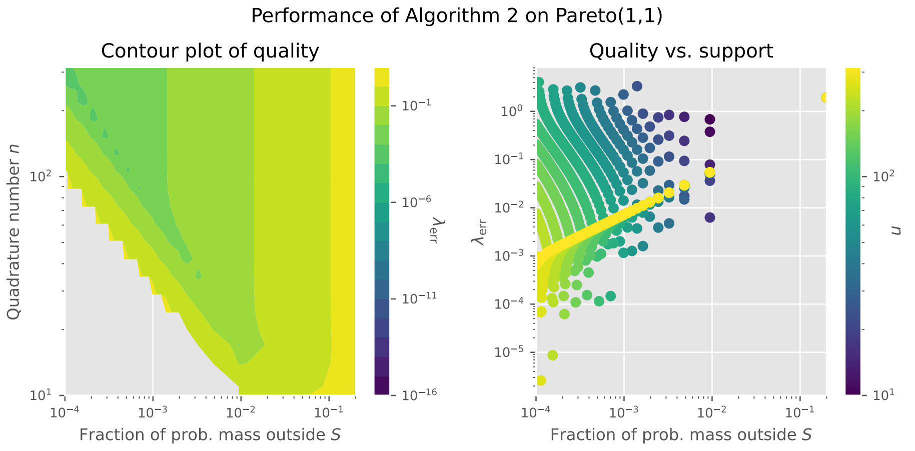
(a) *The performance of Algorithm 2 on the normal distribution with parameters $\mu = 0$ and $\sigma^2 = 1$, using one constraint. The support used for the Algorithm was $\mathcal{S} = [-u, u]$, where u varied from 0.2 to 10 in 100 steps on a linear scale.*



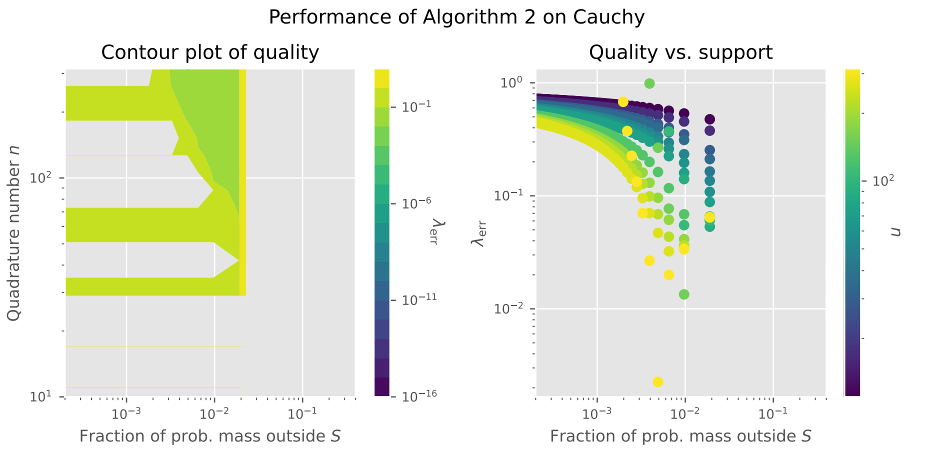
(b) *The performance of Algorithm 2 on the exponential distribution with parameter $c = 1$. The support used for the Algorithm was $\mathcal{S} = [0, u]$, where u varied from 0.5 to 50 in 100 steps on a linear scale.*



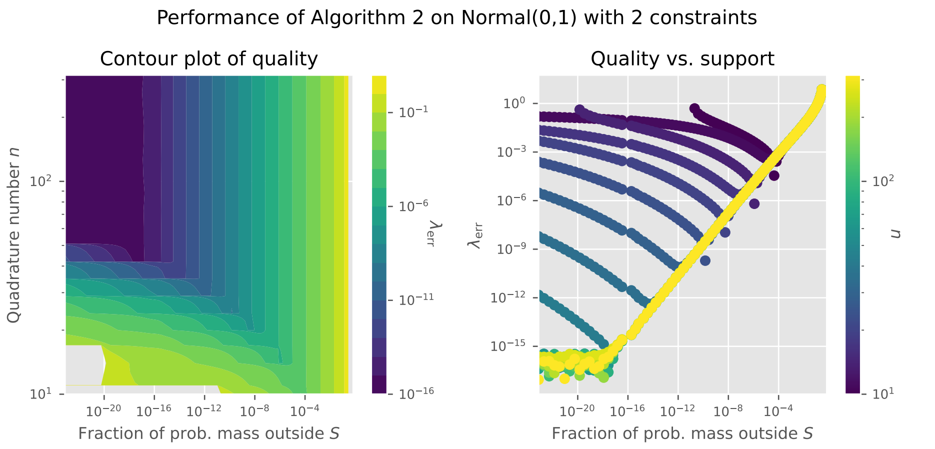
(c) *The performance of Algorithm 2 on the Laplace distribution with parameter $c = 1$. The support used for the Algorithm was $\mathcal{S} = [-u, u]$, where u varied from 0.5 to 40 in 100 steps on a linear scale.*



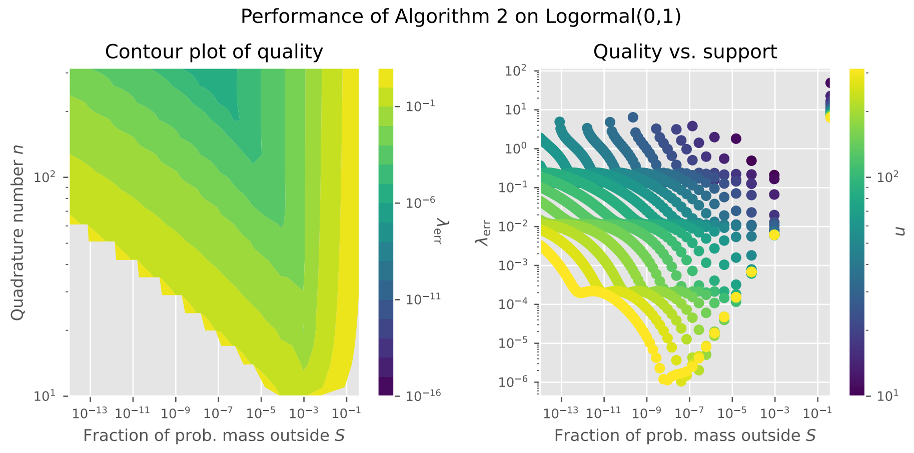
(d) *The performance of Algorithm 2 on the Pareto distribution with parameters $\alpha = 1$ and $x_m = 1$. The support used for the Algorithm was $\mathcal{S} = [1, u]$, where u varied from 5 to $10,000$ in 100 steps on a linear scale.*



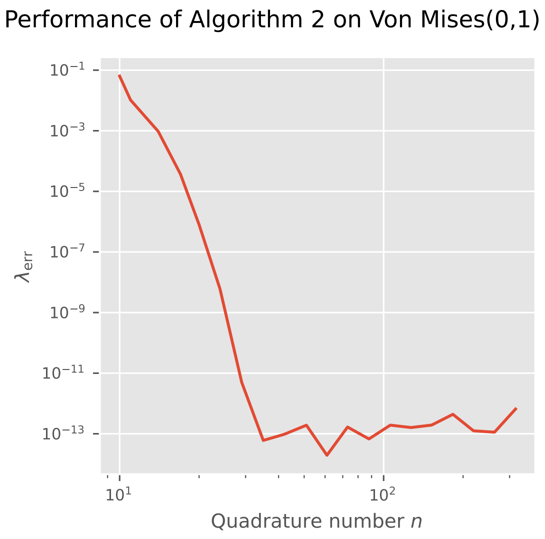
(e) *The performance of Algorithm 2 on the Cauchy distribution. The support used for the Algorithm was $\mathcal{S} = [-u, u]$, where u varied from 1.38 to 3183 in 100 steps on a linear scale.*



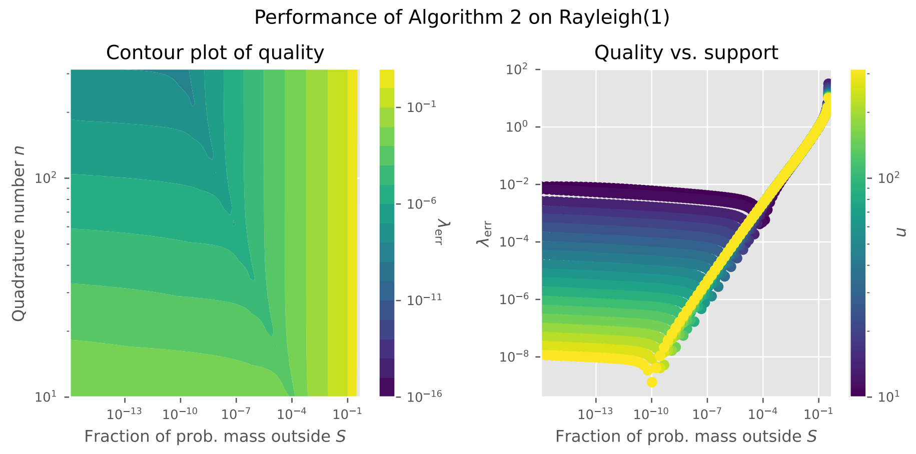
(f) *The performance of Algorithm 2 on the normal distribution with parameters $\mu = 0$ and $\sigma^2 = 1$, using two constraints. The support used for the Algorithm was $\mathcal{S} = [-u, u]$, where u varied from 0.4 to 10 in 100 steps on a linear scale.*



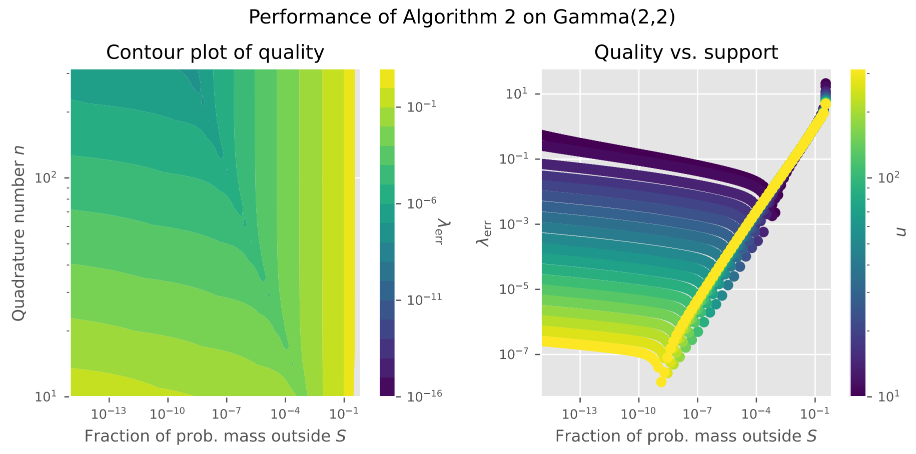
(g) *The performance of Algorithm 2 on the lognormal distribution with parameters $\mu = 0$ and $\sigma^2 = 1$. The support used for the Algorithm was $\mathcal{S} = [0, u]$, where u varied from 1.29 to 2102 in 100 steps on a linear scale.*



(h) *The performance of Algorithm 2 on the von Mises distribution with parameters $\mu = 0$ and $\kappa = 1$. The quality of the result is shown as a function of quadrature number n. The support was fixed at $[0, 2\pi]$.*



(i) *The performance of Algorithm 2 on the Rayleigh distribution with parameter $\sigma^2 = 1$. The support used for the Algorithm was $\mathcal{S} = [0, u]$, where u varied from 1.18 to 8.57 in 100 steps on a linear scale.*



(j) *The performance of Algorithm 2 on the gamma distribution with parameters $k = 2$ and $\theta = 2$. The support used for the Algorithm was $\mathcal{S} = [0, u]$, where u varied from 2.19 to 76.4 in 100 steps on a linear scale.*

Figure 3.4: *The performance of Algorithm 2 on several distributions from Table 2.2. Left, a contour plot shows the error of the solution λ_{err} as a function of the fraction of probability mass outside support $\mathcal{S}$, and n. Empty areas belong to non-converging cases. Right, the same information is plotted in a different way. n was varied from 10 to 316 in 20 steps on a logarithmic scale, and u was varied as specified under the Figure. The von Mises distribution is presented differently.*

Conclusions about performance

From the performance figures, a few observations can be made:

(i) In each instance, there is a certain value for the minimum fraction of probability mass that needs to be in the interval $[l, u]$ before convergence was possible, no matter the value of n. This fraction usually lies below 0.90. This means that in practice, it is often easy to make the interval large enough.

(ii) Once the support fraction is larger than the above-mentioned minimal value, two general cases can be distinguished: situations where the error is decreased by making the interval larger, and ones where the error is decreased by increasing n. Sometimes, increasing the support may cause the result to be worse, or even diverge. Increasing n usually doesn't negatively impact performance.

In general, these results are encouraging: the outcome can be made quite accurate, and the Algorithm can be executed dozens of times per second. Note however, that the performance depends on which distribution which is approximated. The Cauchy and lognormal distribution do not have great convergence, and it's easy to make the support too large.

From these Figures, it is also not known how the amount of constraints k influences performance. In the next Subsection, a particular MaxEnt problem with four constraints is investigated.

3.2.3 Skewness and kurtosis

Two parameters that are often used to describe the shape of a distribution are the skewness and kurtosis. Generally speaking, skewness indicates the *asymmetry* of a distribution, while kurtosis measures the *prominence of the tails*. Formally, they are defined by Definition 3.1.

Definition 3.1. Let X be a random variable with mean μ and non-zero variance σ^2. The **skewness** γ_1 and **kurtosis** γ_2' of X are defined as

$$\gamma_1 := \mathbb{E}\left[\left(\frac{X-\mu}{\sigma}\right)^3\right] \quad \text{and} \quad \gamma_2' := \mathbb{E}\left[\left(\frac{X-\mu}{\sigma}\right)^4\right].$$

Since these two parameters are common ways of describing distributions, it's interesting to consider what the MaxEnt distribution is for some fixed mean, variance, skewness and kurtosis. There is no known closed-form MaxEnt distribution for these constraints [5].

Note that skewness and kurtosis are invariant under linear transformations; without loss of generality, one can set the mean and variance to 0 and 1, respectively. In this case, the skewness and kurtosis are equal to the third and fourth moment. The resulting density can afterwards be rescaled to adjust for the desired mean and variance.

Formally, the following case is considered: the support is $\mathbb{R}$ with $k = 4$ constraints, and constraints $\boldsymbol{f}(x) = (x, x^2, x^3, x^4)$ and $\boldsymbol{F} = (0, 1, \gamma_1, \gamma_2')$.

One instance with $\gamma_1 = 1$ and $\gamma_2' = 3$ is shown in Figure 3.5. The path the Lagrange parameters take toward the minimum can be quite complex.

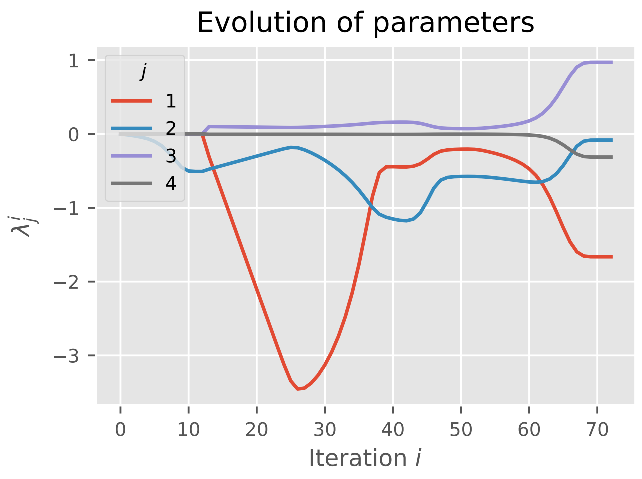
Rockinger & Jondeau investigated this skewness-kurtosis problem in the same paper from which Algorithm 2 was inspired [5]. Figure 3.6 shows the convergence of the algorithm for varying skewness and kurtosis, which corresponds well to Figure 1 of [5]. However, increasing n from $n = 40$ (used in the paper) to $n = 200$, gave a slightly larger domain of convergence. A larger area of convergence than in Figure 3.6 could not be found. Outside the region of convergence in Figure 3.6, there often exists no MaxEnt distribution according to [5]. Adjusting the parameters therefore won't work in this case.

3.3 Conclusions on finding a MaxEnt distribution

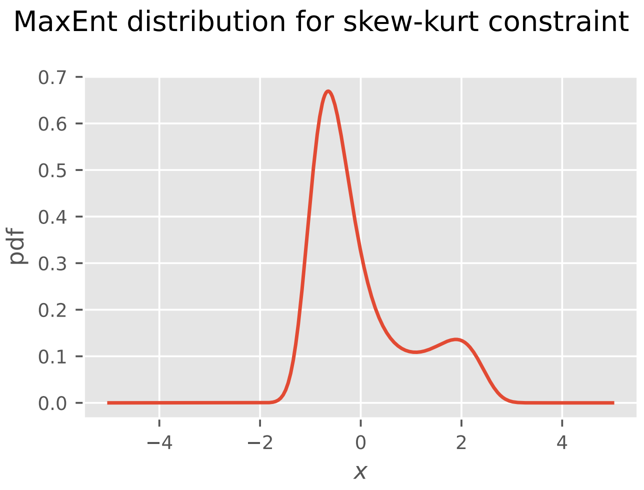
Two methods have been implemented to find a MaxEnt distribution. Algorithm 1 attempts to solve the system of equations 2.4 in Mathematica. However, it could only do so reliably with no more than one constraint, because the form of the equations quickly became too complicated. Also, even if it worked, it was not fast, usually taking over one second. The performance of this Algorithm was hindered by computational complexity, and might be implemented with better results by using another program or function with an appropriate system-solving method.

The second Algorithm performed much better. It was very fast, allowing dozens of runs per second, and performed well under constraints with a known solution. The downside of this method is the approximations which have to be made: the support has to be approximated with a finite interval, and the integral in the definition of the potential is evaluated with an n-point Gaussian quadrature. Depending on the chosen support and value for n, Algorithm 2 can perform badly or not converge at all.

In the next Chapter, Algorithm 2 is applied to some simulated cosmological fields, using their skewness and kurtosis as constraints.



(a) *Lagrange parameters vs. iteration.*



(b) *The resulting MaxEnt density.*

Figure 3.5: *A visualization of Algorithm 2 and resulting density using the constraints* $\boldsymbol{f}(x) = (x, x^2, x^3, x^4)$ *and* $\boldsymbol{F} = (0, 1, 1, 3)$*, support* $[-20, 20]$ *and* $n = 200$.

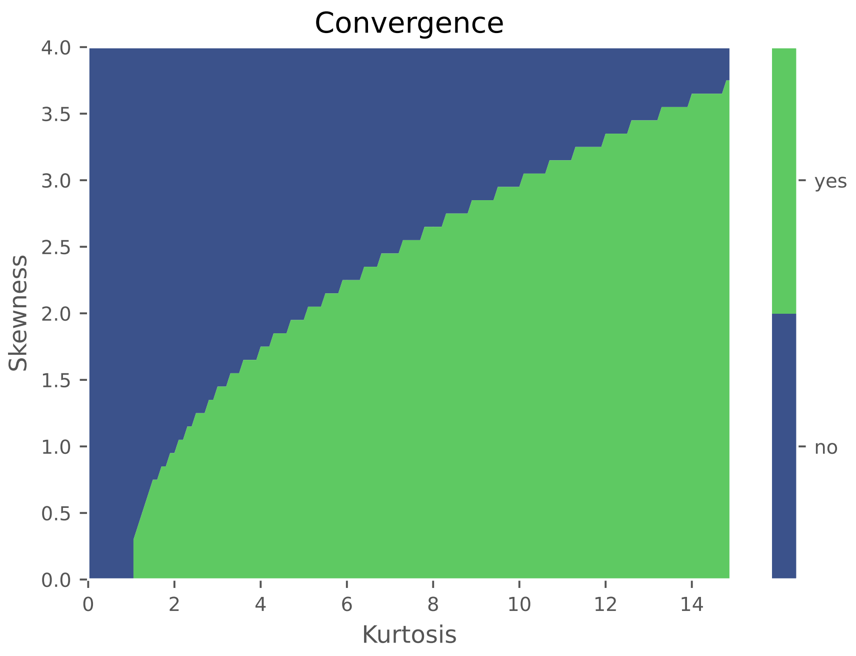


Figure 3.6: *A contour plot showing the convergence of Algorithm 2 with varying skewness γ_1 and kurtosis γ_2'. The constraints were $\boldsymbol{f}(x) = (x, x^2, x^3, x^4)$ and $\boldsymbol{F} = (0, 1, \gamma_1, \gamma_2')$, with support $[-20, 20]$ and quadrature number $n = 200$. This is a recreation of Figure 1 in [5].*

Chapter 4

Analyzing cosmological fields with skewness and kurtosis

If you would describe the universe as consisting only of empty space, you would be mostly correct. However, this would not a very interesting description of the universe; most things we know, like planets, stars or black holes, wouldn't exist.

Likewise, most of the night sky is dark, with sparse bursts of light, way brighter than the background. When statistically describing the night sky, we need to model both the large patches of relative darkness, as well as the small amount of very bright areas. This implies that when considering the marginal distribution of the brightness, it should have a very large tail toward the brighter areas. This motivates the suspicion that the sample skewness and kurtosis might describe these distributions.

In this Chapter, we analyze projection maps from the FLAMINGO project. First, some explanation is given about FLAMINGO and the data (4.1). The analysis is presented in Section 4.2, where Algorithm 2 calculated MaxEnt distributions using the skewness and kurtosis as constraints.

4.1 FLAMINGO

The standard cosmological model can give a good description of the large-scale structure of the universe. It can model the evolution of the universe over billions of years. However, some parameters of the cosmological model have proved to be very hard to determine. One infamous example is the Hubble constant, a parameter indicating how fast the universe is expanding. Over the last decades, two different methods have been used

	DMO		Hydrodynamical	
	Skewness	Kurtosis	Skewness	Kurtosis
Planck	2.8 ± 0.2	22 ± 4	2.9 ± 0.2	24 ± 4
Fiducial	2.8 ± 0.3	23 ± 5	2.9 ± 0.3	24 ± 5
Low sigma8	2.7 ± 0.2	21 ± 3	2.8 ± 0.2	22 ± 3

Table 4.1: *Average skewness and kurtosis of the FLAMINGO projection maps, with their standard deviation, based on five projection maps each.*

to measure this value. Both led to significantly different results and it's still unclear why [12]. Another way to constrain cosmological parameters is be to do simulations with different sets of cosmological parameters (different **cosmologies**), and assessing for which of these parameters the simulation resembles our universe.

Due to the extreme computational difficulty of such large-scale simulations, an often-used simplification is the **dark matter only** (**DMO**) model. However, this only works for large scales. To accurately simulate on smaller scales, baryonic matter also needs to be taken into account. These simulations, which use both dark and baryonic matter, are called **hydrodynamical** simulations, and are computationally much more demanding.

FLAMINGO is a project of the Virgo Consortium for Cosmological Supercomputer Simulations. The aim is to execute accurate hydrodynamical simulations for various cosmologies [13]. Each of these simulations results in slightly different kinds of visible structure.

The images used for the analysis below are projection maps made from the results of the simulations; the 3D state of the simulation was transformed into full-sky maps (images as one would see the sky), and small patches, called projection maps, were taken from these full-sky maps. The projection maps were provided by Maria Marinichenko, a member of the same research group as for which this thesis was written.

Three different cosmologies were considered (Planck, fiducial and LS8/ Low sigma8), and both hydrodynamical and DMO simulations were performed. The cosmological parameters that correspond to the three cosmologies can be found in Table 4 of [13]. For each of these six simulations, we have five projection maps. Table 4.1 shows the skewness and kurtosis of the projection maps. Since there is no significant difference between those statistics, it was chosen to only consider one image per simulation type for the following analysis. Those images are shown in Figure 4.1.

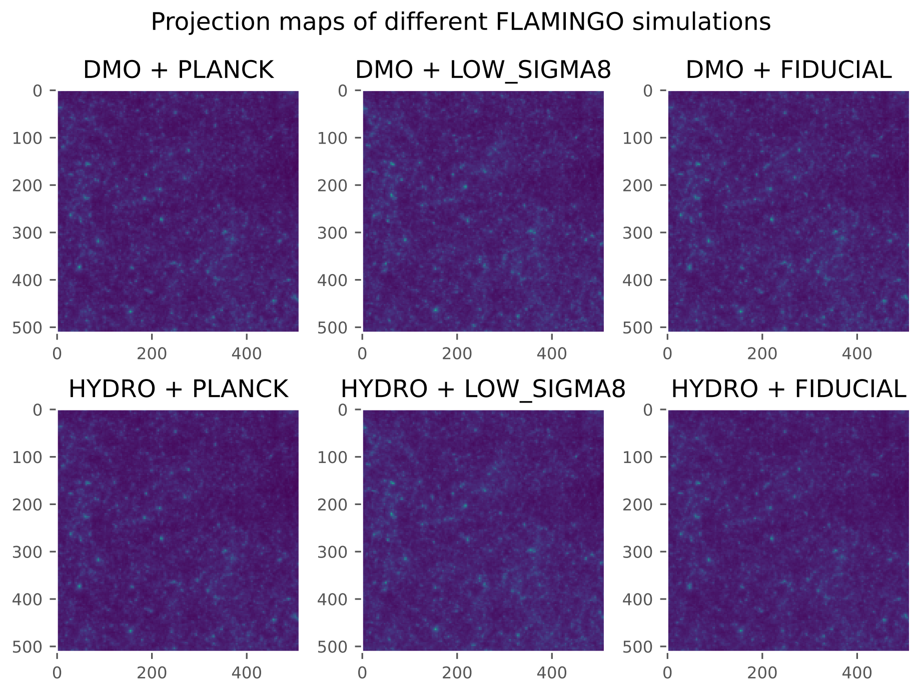


Figure 4.1: *Projection maps of several FLAMINGO simulations. The cosmological parameters that indicate the Planck, low sigma8, and fiducial cosmologies are shown in Table 4 of [13].*

4.2 Analysis of the projection maps

The goal of this Section is to analyze the FLAMINGO fields by determining a MaxEnt distribution based on the skewness and kurtosis of the marginal distribution of the pixel brightness. To determine how well this distribution describes the data, samples were drawn from it, and then compared with the original data using a Kolmogorov-Smirnov test.

The main part of the analysis is shown in Algorithm 3. As explained in Subsection 3.2.3, the image can be standardized using the mean and the variance, which happens in step 3. This makes the skewness and kurtosis equal to the third and fourth moments, which are calculated in step 4. Step 5 fixes the support as the interval with the minimum and maximum values of the image pixel brightness as its borders, which is reasonable given the sample size is of order 10^5. Then Algorithm 2 is performed with the first four moments as constraints.

Once the MaxEnt density has been determined, samples are drawn from it using rejection sampling in step 7. Described in Algorithm 4, rejection sampling is a sampling technique that can produce samples from a density g using a so-called dominating density f. One says f **dominates** g if there exists a constant $C > 0$ such that $g(x) \leq Cf(x)$ for all x. Furthermore, to use rejection sampling, it should be possible to sample from the dominating density. Since the MaxEnt distribution is defined on a finite support $[l, u]$, the uniform distribution is an appropriate dominating density. In step 8 of Algorithm 3, the samples are rescaled with the original mean and variance.

Figures 4.2a to 4.2f show the results of the analysis. As the tails of the distribution are not visible in the normal histogram, the second figure has a logarithmic axis. The Figures also show the p-value of the 2-sample Kolmogorov-Smirnov test (KS test). Appendix A shows the results of the same analysis on other types of images.

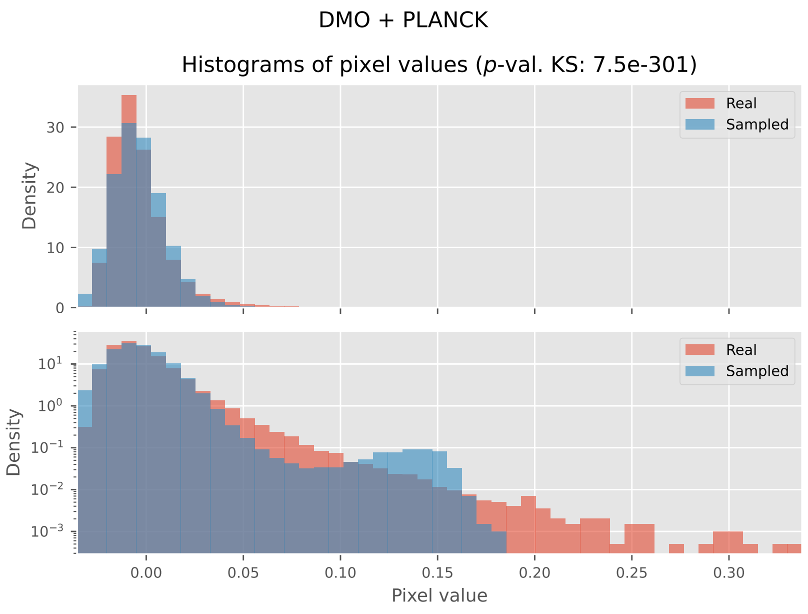
The two samples are markedly different in all Figures. Accordingly, the p-value is very low. This is the case for all six projection maps. There is no big difference between the performance on the different simulations that can't be attributed to chance. Also, the MaxEnt distributions have trouble capturing the tails of the real distribution. However, these regions with high pixel values are important to model correctly, since this is the area where interesting phenomena like stars exist. Therefore, it's crucial to describe the tails accurately. This implies the skewness and kurtosis alone aren't good enough to describe the shape of such cosmological images accurately.

Algorithm 3: Find a MaxEnt distribution of the marginal distribution of an image, with skewness and kurtosis constraints, and take samples from this distribution.

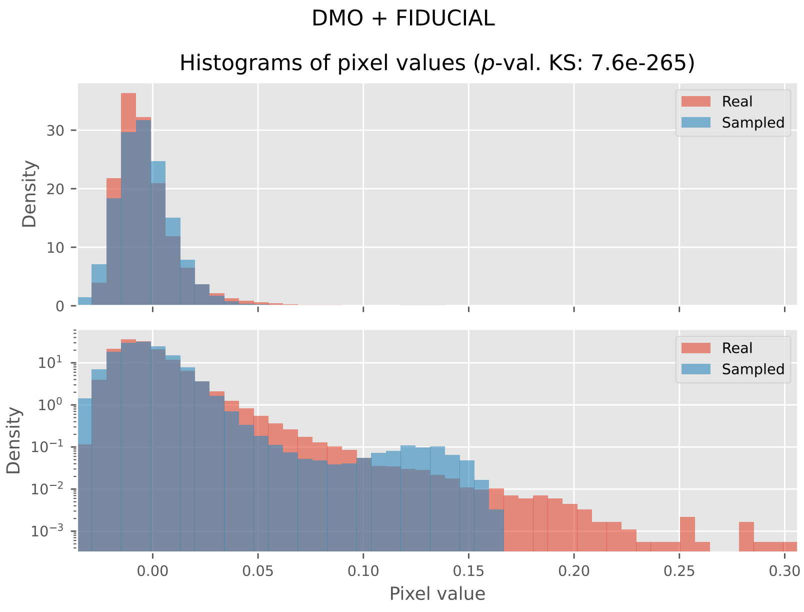
```
Input: image;
       quadrature number $n$
Output: sample from the MaxEnt distribution based on skewness
        and kurtosis, with the same sample size as the amount of
        pixels in image.
1 Let $\mu$ and $\sigma^2$ be the sample mean and sample variance of the pixel
    values of image.
2 Let $m := |image|$ be the amount of pixels.
3 Standardize the image: $image\_st := (image - \mu)/\sigma$.
4 Let the skewness $\gamma_1$ and kurtosis $\gamma_2'$ be the third and fourth sample
    moments of image_st, respectively.
5 Define $l := \min(image\_st)$ and $u := \max(image\_st)$.
6 With support $[l, u]$, quadrature number $n$, find the MaxEnt
    distribution with Algorithm 2 with constraints
    $\boldsymbol{f}(x) = (x, x^2, x^3, x^4)$ and $\boldsymbol{F} = (0, 1, \gamma_1, \gamma_2')$. Call this density $g$.
7 Sample $m$ i.i.d. values from $g$ using rejection sampling
    (Algorithm 4), with dominating density Unif$[l, u]$ and constant
    $C := \max\{g(x) : x \in [l, u]\} \cdot (u - l)$.
8 Rescale the samples by multiplying by $\sigma$, and adding $\mu$.
```

Algorithm 4: Sample from g with rejection sampling. For more background on this technique, see [14, Ch. II.2b].

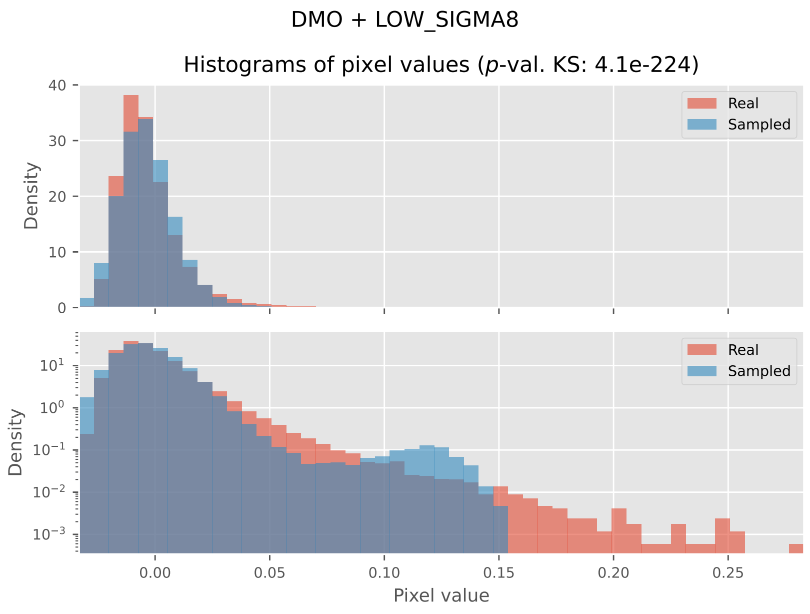
```
Input: density $g$;
       dominating density $f$ on the same support as $g$ from which
         one can sample;
       constant $C$ such that $g(x) \leq Cf(x)$ for all $x$
Output: random sample $X$ from $g$
1 Sample $Y \sim f$ and $U \sim$ Unif$[0, 1]$ independently.
2 if $U > g(Y)/Cf(Y)$ then
3     return to step 1.
4 Return $X := Y$.
```



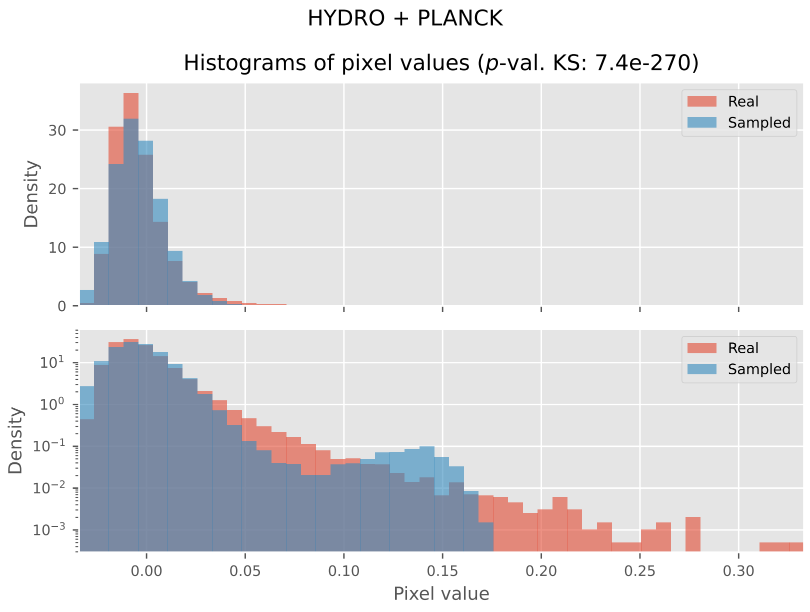
(a) *Results using an image from a dark matter only simulation with the Planck cosmology.*



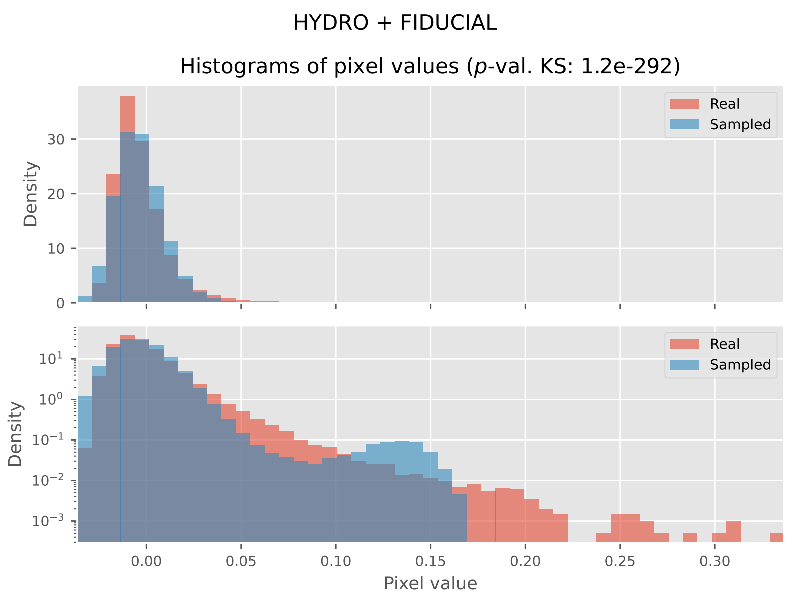
(b) *Results using an image from a dark matter only simulation with the fiducial cosmology.*



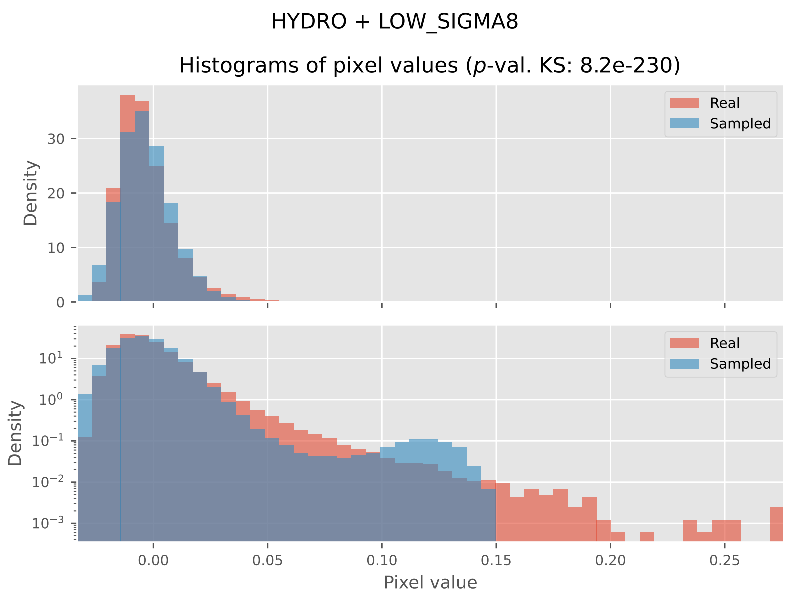
(c) *Results using an image from a dark matter only simulation with the low sigma8 cosmology.*



(d) *Results using an image from a hydrodynamical simulation with the Planck cosmology.*



(e) *Results using an image from a hydrodynamical simulation with the fiducial cosmology.*



(f) *Results using an image from a hydrodynamical simulation with the low sigma8 cosmology.*

Figure 4.2: *The results of Algorithm 3 with various projection maps and $n = 350$ in two histograms. In red, the image values are plotted, while in blue, samples from the MaxEnt distribution based on the skewness and kurtosis are shown. Both histograms show the same data, but the lower plot has a logarithmic scale on the density.*

Chapter 5

Discussion

The method of solving system of equations 2.4 did not succeed in reliably finding MaxEnt distributions for multiple constraints. However, if better solving methods are found which are suited to this type of problem, it would be worth investigating, because this method can provide exact MaxEnt distributions.

Minimizing the potential Q does appear to be a reliable way to find MaxEnt distribution. It performed equally well as in [5] in the case of moment constraints. Due to the more general implementation, it could also find MaxEnt distributions for other types of constraints.

However, it is not perfect: the performance relies heavily on two parameters: the size of the interval of support, and the value for n used in the n-point Gaussian quadrature. These parameters influence the approximation of the potential function, which can shift, or even remove, the minimum. Further research could focus on methods to determine the optimal values for these parameters, and whether a support which is asymmetric about the mean could improve performance.

Even though a larger n generally increases accuracy, the current application does not allow for an unlimited increase in n, since the quadrature nodes and weights used are potentially not accurate for $n > 100$ according to the NumPy documentation. To remedy this, the Algorithm could be adjusted in one of two ways: (i) find more accurate quadrature nodes and weights, or (ii) stop using the quadrature approximation, and evaluate the integral in another way. While this Algorithm uses automatic differentiation, the implementation of (ii) may be better achieved with a finite-difference approach.

A limitation of Algorithm 2 is the amount of constraints: when choosing six or more constraints, the algorithm rarely converges (see e.g. Appendix B). This problem can probably only be solved if the Gaussian quadrature approximation were replaced by a more robust technique to evaluate the integral.

The marginal distribution of cosmological fields couldn't described accurately by the MaxEnt distribution using skewness and kurtosis, since the right tail was too large. This can be seen in Appendix A, where the same analysis was performed on various kinds of patterns. The same analysis worked a lot better when applying the Algorithm to turbulence-like patterns.

It was also attempted to add the eighth moment as a fifth constraint to the skewness-kurtosis analysis. This marginally improved the results, but not by much. A better result could perhaps be achieved with a logarithmic constraint.

While this particular application did not lead to a very accurate description of the data, the principle of maximum entropy is nonetheless a powerful tool for density estimation. When information is sparse, one can still make an educated guess on the original density, which can for instance be used as a prior density, or to simulate data for follow-up simulations.

Chapter 6

Conclusion

An effective way to approximate MaxEnt distributions has been developed in Algorithm 2, which minimizes a strictly convex function Q. This approach was inspired by [5], but instead of only taking moments as constraints, it was generalized to arbitrary constraint functions. As can be seen in Figures 3.4a to 3.4j, all MaxEnt distributions from Table 2.2 with at least one constraint could be approximated reasonably well: for some distributions, like the Laplace, Pareto or Cauchy, the error in the Lagrange parameters was at best about 10^{-3}, while for others, like the normal, exponential and von Mises, the error could reach below 10^{-12}.

Algorithm 2 also determined MaxEnt distributions for skewness and kurtosis constraints. They were the same as in [5], but increasing n from 40 to 200 increased the region of convergence, as shown in Figure 3.6.

A less successful method to calculate the MaxEnt distribution is Algorithm 1, which solves the system of equations 2.4 using Wolfram Mathematica. It worked reasonably well for one constraint, but scaling up to two constraints caused the equations to be too complex to solve in many cases.

Using Algorithm 2, some projection maps from the FLAMINGO project were analyzed. The MaxEnt distribution with skewness and kurtosis constraints did not accurately describe the marginal distribution of these cosmological fields (Figures 4.2a to 4.2f).

All in all, minimizing a potential function is a reliable and fast way to find MaxEnt distributions, although one should take care to use constraints which appropriately describe the distribution.

References

[1] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[2] ——, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 4, pp. 623–656, Oct. 1948.

[3] E. Jaynes, *Information Theory and Statistical Mechanics*. Brandeis University, 1962, vol. 3, pp. 181–218.

[4] E. T. Jaynes, *Probability Theory – The Logic of Science*. Cambridge University Press, 2003, ch. 11, pp. 343–371.

[5] M. Rockinger and E. Jondeau, "Entropy densities with an application to autoregressive conditional skewness and kurtosis," *Journal of Econometrics*, vol. 106, pp. 119–142, 2001.

[6] M. Protter and C. Morrey, *Intermediate Calculus*, ser. Undergraduate Texts in Mathematics. Springer New York, 2012.

[7] S. Y. Park and A. K. Bera, "Maximum entropy autoregressive conditional heteroskedasticity model," *Journal of Econometrics*, vol. 150, no. 2, pp. 219–230, 2009.

[8] J. H. C. Lisman and M. C. A. v. Zuylen, "Note on the generation of most probable frequency distributions," *Statistica Neerlandica*, vol. 26, no. 1, pp. 19–23, Mar. 1972.

[9] T. M. Cover and J. A. Thomas, *Elements of information theory*. Wiley, 2012.

[10] L. Prins, "MaxEnt project," Jun. 2025. [Online]. Available: https://github.com/LiselotPrins/max_entropy

[11] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 1986.

[12] W. L. Freedman and B. F. Madore, “Progress in direct measurements of the hubble constant,” *Journal of Cosmology and Astroparticle Physics*, vol. 2023, no. 11, p. 050, nov 2023.

[13] J. Schaye, R. Kugel, M. Schaller, J. C. Helly, J. Braspenning, W. Elbers, I. G. McCarthy, M. P. van Daalen, B. Vandenbroucke, C. S. Frenk, J. Kwan, J. Salcido, Y. M. Bahé, J. Borrow, E. Chaikin, O. Hahn, F. Huško, A. Jenkins, C. G. Lacey, and F. S. J. Nobels, “The FLAMINGO project: cosmological hydrodynamical simulations for large-scale structure and galaxy cluster surveys,” *Monthly Notices of the Royal Astronomical Society*, vol. 526, no. 4, pp. 4978–5020, Dec. 2023.

[14] S. Asmussen and P. W. Glynn, *Stochastic Simulation: Algorithms and Analysis*. Springer Science+Business Media, LLC, 2007.

[15] S. Cheng, “scattering_transform,” Sep. 2024. [Online]. Available: https://github.com/SihaoCheng/scattering_transform

[16] G. Casella and R. W. Berger, *Statistical inference*. CRC Press, 2024.

[17] P. L. Novi Inverardi and A. Tagliani, “The lognormal distribution is characterized by its integer moments,” *Mathematics*, vol. 12, no. 23, 2024.

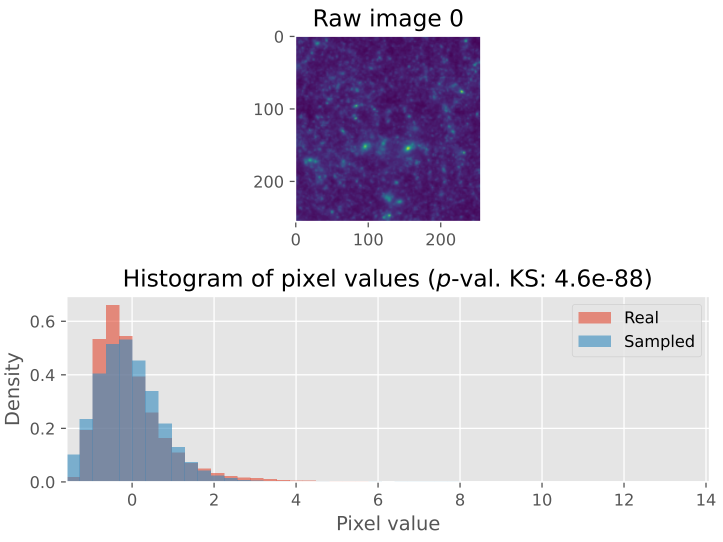
Appendix A

Extra analyses with skewness and kurtosis

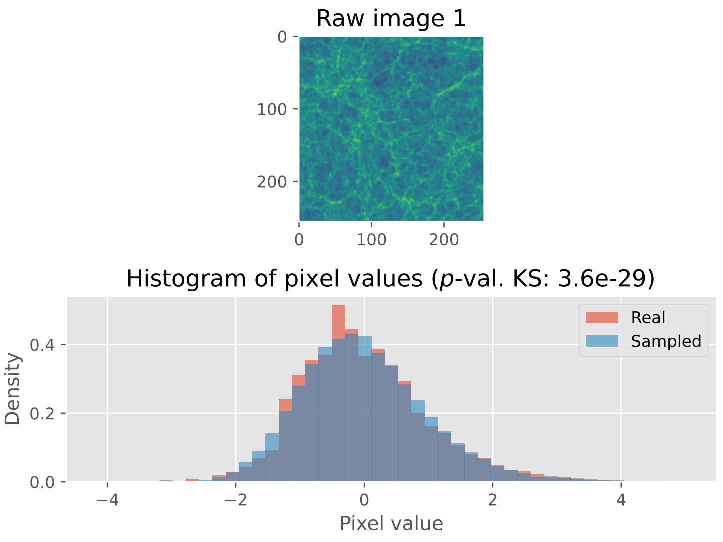
The procedure from Chapter 4 was also applied to other types of patterns than cosmological fields. The images originate from Sihao Cheng's GitHub page on the scattering transform [15]. The file is called '*example_fields.npy*' and is located in the map '*data/physical_fields*'.

Figures A.1a to A.1g show various types of patterns from that file; some resemble the cosmological fields as seen before in this thesis, while others look like turbulence patterns. The upper part of the Figures show the image, and the bottom part shows the image's pixel values in a histogram, along with samples based on the MaxEnt distribution with skewness and kurtosis as generated by Algorithm 3.

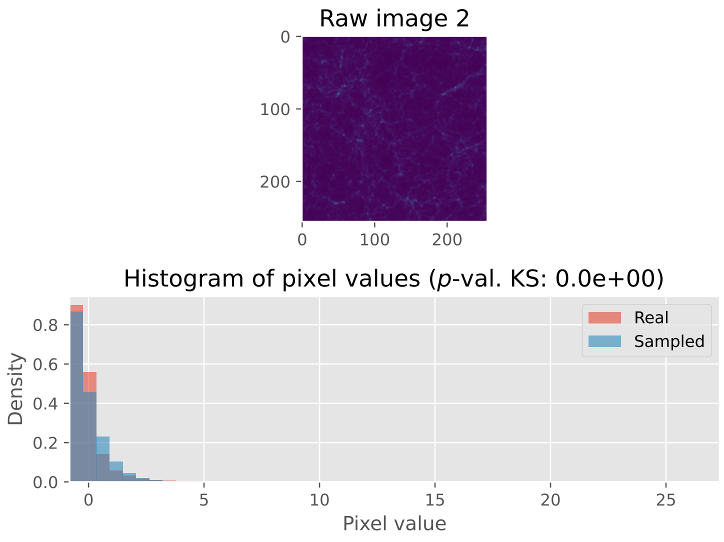
The variation in patterns illustrates how this skewness-kurtosis analysis performs for different kinds of input. The images with long right tails have bad approximations and accordingly low p-values (Figures A.1a and A.1c). However, the other images, which have smaller tails, can be approximated reasonably well with the MaxEnt approach and have markedly higher p-values. Algorithm 3 therefore works better for turbulence-like patterns than for cosmological images.



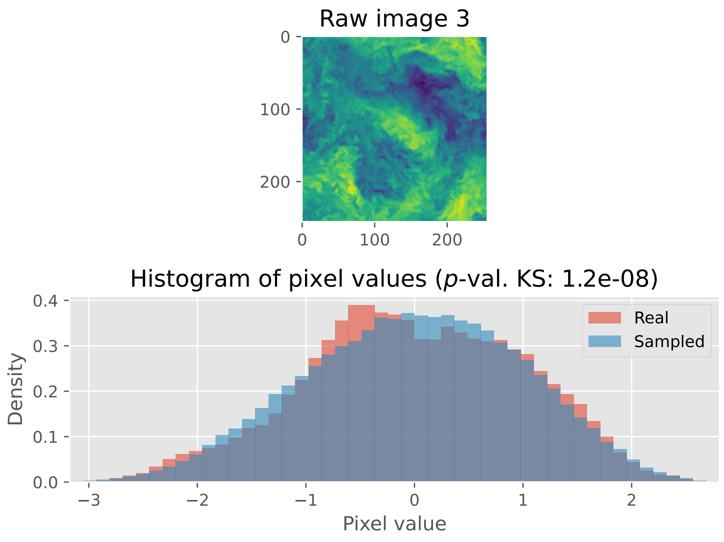
(a) *First image of 'example_fields.npy'.*



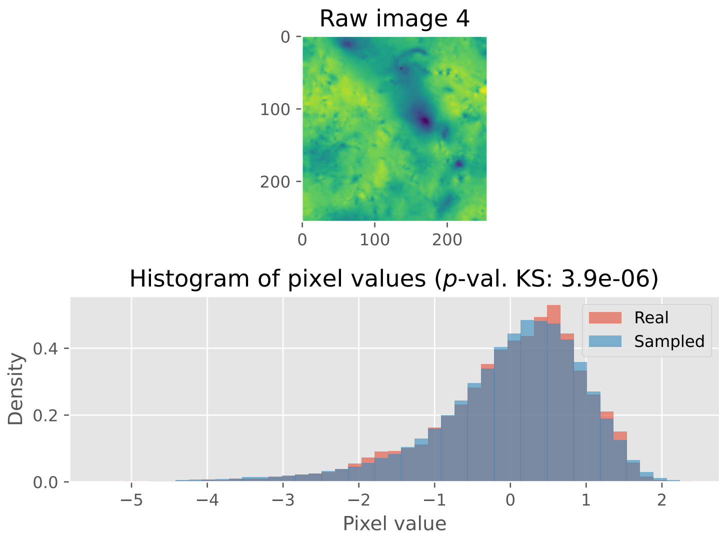
(b) *Second image of 'example_fields.npy'.*



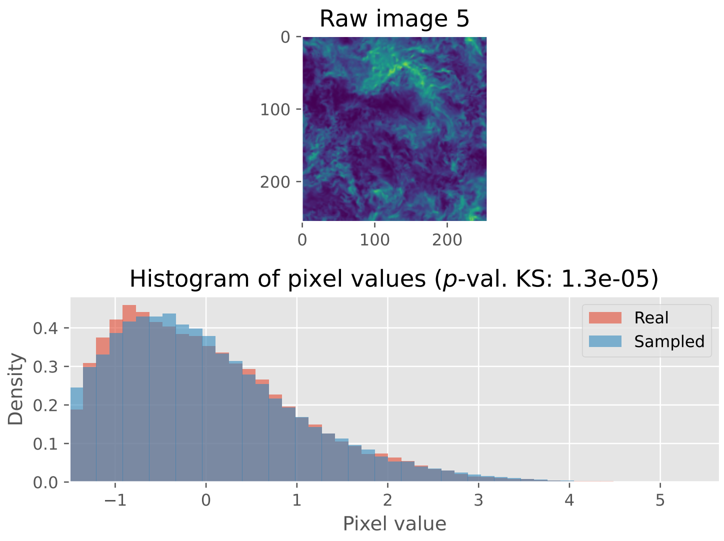
(c) *Third image of 'example_fields.npy'.*



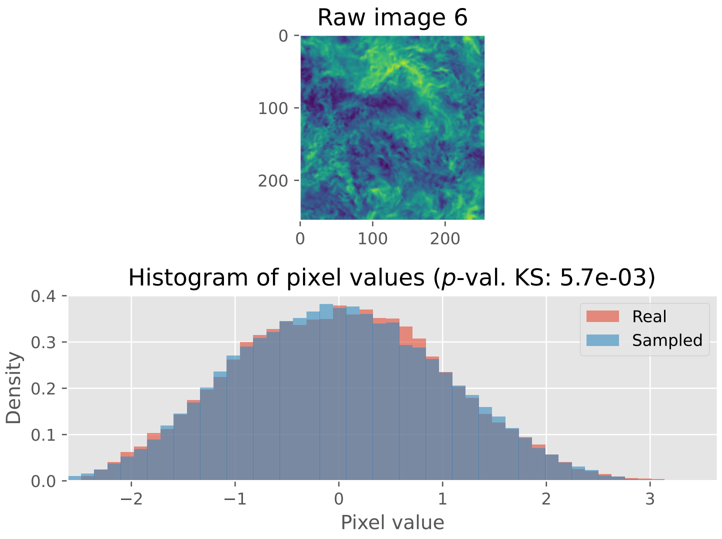
(d) *Fourth image of 'example_fields.npy'.*



(e) *Fifth image of 'example_fields.npy'.*



(f) *Sixth image of 'example_fields.npy'.*



(g) *Seventh image of 'example_fields.npy'.*

Figure A.1: *The results of Algorithm 3 on several images and* $n = 250$*. Below, a histogram shows the results: in red, the image values are plotted, while in blue, the samples from the MaxEnt distribution based on the skewness and kurtosis are shown. The images are the first seven images contained in the file 'example_fields.npy' from Cheng's GitHub page [15].*

Appendix B

Describing the lognormal distribution with its moments

The **moment generating function** (**mgf**) of a random variable X is defined by

$$M_X(t) := \mathbb{E}\left[e^{tX}\right]$$

for all t for which this expectation exists. Since this expectation always exists for $t = 0$, one says the mgf **exists** if, for all t in some neighborhood of 0, the mgf is defined [16, Sect. 2.3].

If the mgf exists, it is unique. This means if two random variables have an mgf which coincide in some neighborhood of 0, they must have the same distribution. Also, for all t in this neighborhood, a sum and expectation can be interchanged in the following way:

$$\begin{aligned} M_X(t) &= \mathbb{E}\left[e^{tX}\right] \\ &= \mathbb{E}\left[\sum_{n=0}^{\infty} \frac{(tX)^n}{n!}\right] \\ &= \sum_{n=0}^{\infty} \frac{t^n \mathbb{E}\left[X^n\right]}{n!}, \end{aligned}$$

from which it becomes clear how the mgf can generate moments:

$$M_X^{(k)}(0) = \mathbb{E}\left[X^k\right].$$

The uniqueness of the mgf implies that for all random variables with existing mgfs, their moments uniquely identify the distribution.

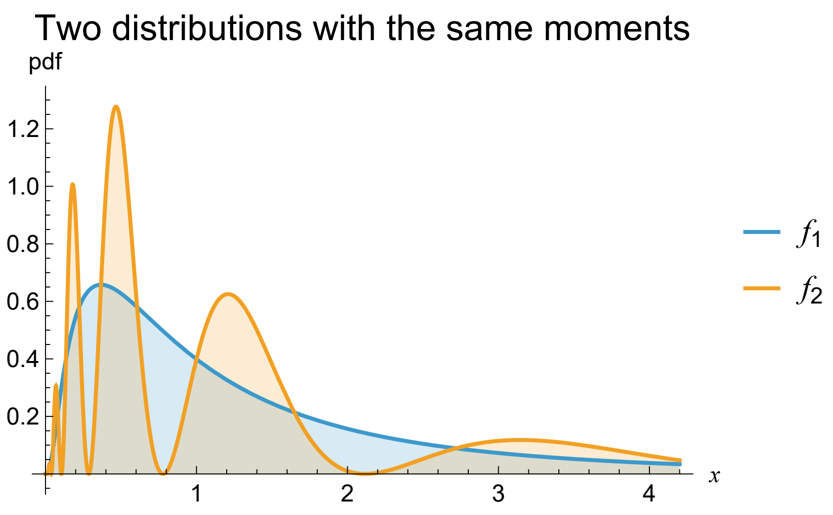


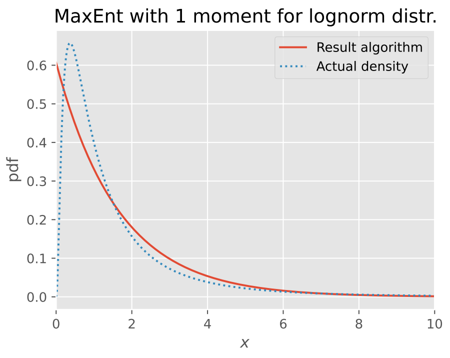
Figure B.1: *Two densities on $\mathbb{R}_{\geq 0}$ with the same moments $\mathbb{E}[X^n] = e^{n^2/2}$: the lognormal distribution with parameters $\mu = 0$ and $\sigma = 1$ and pdf $f_1(x)$, and the distribution with pdf $f_2(x) = f_1(x)(1 + \sin(2\pi \log x))$. Inspiration was taken from Example 2.3.10 in [16].*

However, not every random variable has an mgf. One example is the lognormal distribution. All moments exist, but its mgf does not. For this reason, it's said that the lognormal distribution is not described by its moments. In fact, there exists another distinct distribution with exactly the same moments as the lognormal distribution with $\mu = 0$ and $\sigma = 1$, as shown in Figure B.1.

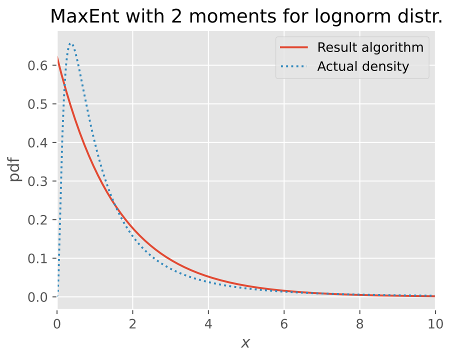
As one can also guess by looking at Figure B.1, the entropy of the second distribution f_2, which is 1.11, is lower than the entropy of the lognormal distribution at 1.42. This raises the question whether, among all distributions with the same moments, the lognormal has the highest entropy. In fact, this is the case, as proven in [17]. The lognormal distribution can then be described by its moments after all.

Therefore, it was attempted with Algorithm 2 to approach the lognormal distribution using the moments as constraints. In theory, the distribution should converge as the amount of moments used is increased. The similarity of two pdfs can be quantified with the KL divergence. A lower KL divergence means a bigger resemblance.

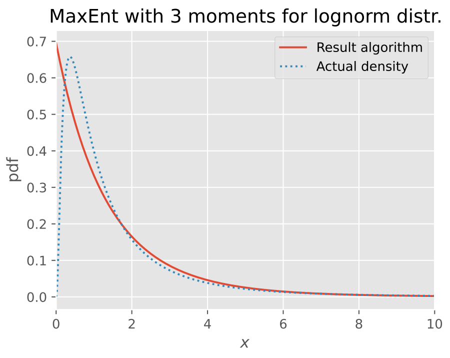
The results are shown in Table B.1, and the resulting densities in Figure B.2. Convergence could only be achieved for at most five constraints. At six, the system of linear equations (in step 8 of the Algorithm) could not be solved. Choosing a different initializing value for $\boldsymbol{\lambda}$ did not help. At seven and eight constraints, the Lagrange parameters diverged.



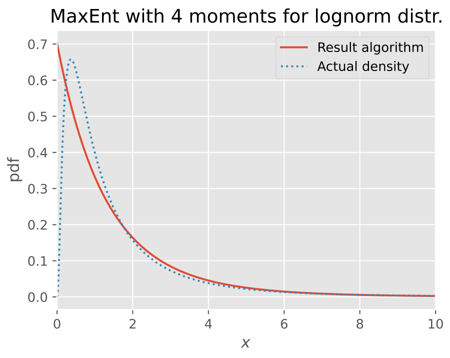
(a) $k = 1$



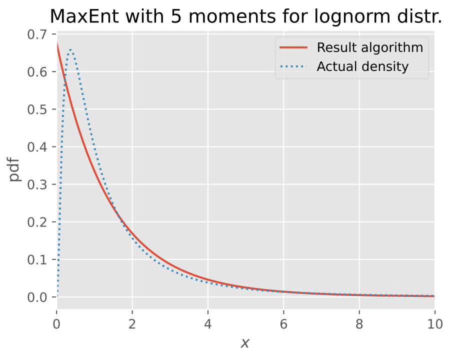
(b) $k = 2$



(c) $k = 3$



(d) $k = 4$



(e) $k = 5$

Figure B.2: *Five MaxEnt distributions from Algorithm 2 with the first k moments of the lognormal distribution as constraints. The lognormal distribution with parameters* $\mu = 0$ *and* $\sigma^2 = 1$ *was considered. Results were obtained with* $n = 350$ *and support* $[0, u]$*, where u is such that a fraction* $1 - 10^7$ *of the mass of the lognormal distribution is inside the interval.*

k	λ_1	λ_2	λ_3	λ_4	λ_5	KL div. $(\cdot 10^{-2})$
1	-0.61	–	–	–	–	8.10
2	-0.63	0.003	–	–	–	7.51
3	-0.76	0.021	$-2.0 \cdot 10^{-4}$	–	–	6.08
4	-0.77	0.023	$-3.0 \cdot 10^{-4}$	$1.08 \cdot 10^{-6}$	–	6.06
5	-0.71	0.006	$8.1 \cdot 10^{-4}$	$-2.1 \cdot 10^{-5}$	$9.2 \cdot 10^{-8}$	6.31

Table B.1: *The results of Algorithm 2 with the first k moments of the lognormal distribution as constraints, for $k = 1, \ldots, 5$. The resulting Lagrange parameters are shown, along with the KL divergence $D_{KL}(p||q)$, where p is the lognormal, and q the MaxEnt density. The lognormal distribution with parameters $\mu = 0$ and $\sigma^2 = 1$ was considered. Results were obtained with $n = 350$ and support $[0, u]$, where u is such that a fraction $1 - 10^7$ of the mass of the lognormal distribution is inside the interval.*

The resulting MaxEnt density appears to get closer to the actual density as k increases. Accordingly, in Table B.1, we see the KL divergence decreasing, albeit slowly. Only for $k = 5$ does it rise again. For five constraints, the Lagrange parameters also deviate from the previous calculations. Perhaps the Algorithm provided an inaccurate MaxEnt distribution in that case.

All in all, no conclusions can be drawn due to the small amount of constraints used and potential numerical inaccuracies. The Algorithm would need to perform better under more constraints to investigate this properly.