Using Supervised Learning on maps of total column density to investigate the fractal structure of both the Milky Way and the Andromeda Galaxy.

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

by

MATTHEW LOUIS BATES

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

June, 2023

ABSTRACT

In this thesis, we design, train, and test multiple Convolutional Neural Networks (CNNs) that can take images of surface-density as inputs and estimate the Hurst parameter, $\mathcal{H}$ and, the scaling exponent, $\mathcal{S}$ of each image. $\mathcal{H}$ is a measure of the power spectrum while $\mathcal{S}$ is a measure of the range of the surface-density. The CNNs are trained using exponentiated fractional Brownian motion (xfBm) fields that measure $N_{\rm PIX} \times N_{\rm PIX}$ pixels in size. We produce three CNNs, one for each $N_{\rm PIX} = 128, 64,$ and 32. We find that all three models estimate $\mathcal{H}$ more accurately than Delta-Variance. We use these CNNs to analyse Hi-GAL tiles, using a sliding patch, in order to produce maps of $\mathcal{H}$ and $\mathcal{S}$. Each Hi-GAL tile has been processed using `PPMAP` to produce maps of total column-density. We find that the size and resolution of the patches, (i.e. the range of scales that the patch captures) affects the resultant statistics. We find evidence of bimodality between sightlines with higher surface density ($\Sigma \gtrsim 32$ M$_\odot$pc^{-2}) which correlates with a high $\mathcal{H}$ ($\gtrsim 0.8$) and $\mathcal{S}$ ($\gtrsim 1$), and sightlines with lower surface-density ($\Sigma \lesssim 32$ M$_\odot$pc^{-2}), which correlate with lower $\mathcal{H}$ ($\lesssim 0.8$) and $\mathcal{S}$ ($\lesssim 1$). We find that this bimodality is not present when analysing maps of temperature-differential column density. We find that when $\Sigma \gtrsim 32$ M$_\odot$pc^{-2} the surface densities follow a power law distribution $dP/d\Sigma \propto \Sigma^{-3}$. We also apply these CNNs to *Herschel* maps of the Andromeda Galaxy (M31) that have been processed via `PPMAP`. We find that the distribution of $\mathcal{H}$ for Andromeda has a peak at $\mathcal{H} = 0.67$, which agrees with the literature.

Keywords: *Thesis*, *Dissertation*, *Degree*, *Star Formation*, *ISM*, *Galaxies*, *Machine Learning*.

ACKNOWLEDGEMENT

It is with my deepest gratitude that I thank the following people, without whom I would have not been able to complete my research, and who have thus helped me immensely throughout:

My supervisor, Prof. Anthony Whitworth, whose support, patience, and seemingly infinite subject knowledge has guided me throughout this endeavour.

The rest of the CHART students and faculty, with whom I have had frequent and interesting exchanges of ideas.

My parents for their constant encouragement, and without whom I would not be on this exciting path.

My partner, Amie, for her support and patience, and who has been my foundation.

My children, Isabella and Louie, who fill me with immense joy everyday.

And last but not least, all my friends, family, and colleagues who have supported me along the way.

Date: 30/06/2023 **MATTHEW LOUIS BATES**

CONTENTS

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Introduction

It has long been suggested that the structure of the interstellar medium (ISM) is fractal in nature and thus possesses some degree of self-similarity over a large range of length-scales (Beech, 1987; Falgarone et al., 1991; Hetem and Lepine, 1993; Elmegreen and Falgarone, 1996; Stutzki et al., 1998; Chappell and Scalo, 2001; De La Fuente Marcos and De La Fuente Marcos, 2006; Shadmehri and Elmegreen, 2011; Elia et al., 2018; Beattie et al., 2019a,b; Petkova et al., 2023). This structure is also chaotic. This is mostly due to the fact that the processes that govern the evolution of the ISM are non-linear. Such processes include: self-gravity, magneto-hydrodynamic turbulence, and a range of non-LTE (local thermodynamic equilibrium), thermal, radiative and chemical processes (Habing, 1968; Field et al., 1969; Jura, 1974; Hollenbach and McKee, 1979; Mathis et al., 1983; Sternberg and Dalgarno, 1995; Spitzer, 1998; Wolfire et al., 2003; Sternberg et al., 2014). As a result, the structure of the ISM must be described using statistical metrics.

The main property that is of interest when we concern ourselves with fractal structures is the fractal dimension, $\mathcal{D}$. Simply, the fractal dimension of a structure in an $\mathcal{E}$-dimensional space measures how efficiently that structure fills the space. There are many different definitions of the fractal dimension. Examples include: the box-counting dimension (otherwise known as the Minkowski–Bouligand dimension), the perimeter-area dimension, and the Hausdorff dimension (Schroeder, 1996). Of these, the two types of fractal dimension that have been most commonly used to describe the ISM are: the perimeter-area fractal dimension $\mathcal{D}_{PA}$ (Beech, 1987; Bazell and Desert, 1988; Falgarone et al., 1991; Hetem and Lepine, 1993; Stutzki et al., 1998; Sanchez et al., 2005; Federrath et al., 2009; Rathborne et al., 2015) and the box-counting fractal dimension, $\mathcal{D}_{BC}$ (Sanchez et al., 2005; Federrath et al., 2009; Elia et al., 2018).

The names given to these two metrics suggest the method by which they are defined. The perimeter-area method defines $\mathcal{D}_{PA}$ while the box-counting method defines $\mathcal{D}_{BC}$. Federrath et al. (2009) compare both fractal dimensions with each other and conclude that the perimeter-area method is the less noisy of the two, a point that was also briefly mentioned in Sanchez et al. (2005). In Chapter 2 we explain the two methods as well as the relationship between these two fractal dimensions.

Another method that can be used to measure the fractal dimension is to calculate the Delta-variance (Stutzki et al., 1998). The Delta-variance analysis assumes that the fractal dimension is related to the power spectrum of the image, via the power spectral index, β. Thus this assumes that the structure being investigated is similar to a fractional Brownian motion (fBm) field. We refer the reader to Section 3.4 for more details on Delta-Variance, but in short, it involves finding the variance $\sigma_{\Delta}^{2}(L)$ of the image after it has been convolved with a circular filter, $\odot_L$, of length-scale L and using the relation $\sigma_{\Delta}^{2} = L^{-2\mathcal{D}_{PA}}$. We note that since Delta-Variance relies on a probing filter that acts at all possible length-scales it is less prone to noise and other corrupting artefacts than a simple measurement of its power spectrum. This is because the length-scale at which these artefacts occur can be simply ignored. We must stress however that this filtering of noise sources is a manual procedure and must be done by eye on a case-by-case basis, by inspecting plots of $\log(\sigma_{\Delta}^{2})$ against $\log(L)$. Even then, there is a limitation as to how much the image can be corrupted while still being able to recover accurate estimations of $\mathcal{D}$. We explore this in Section 3.4, particularly when comparing Figures 3.7 and Figures 3.8.

Most of the analyses we have mentioned so far have assumed that a region with a relatively large dynamic range can be described by a single $\mathcal{D}$. A structure that can be described with a single fractal dimension is called a monofractal. This is generally applicable for isolated structures, where each subregion can be described by the same fractal dimension no matter the position or length scale. The reality is however that structures in the ISM do not tend to exist in isolation and even if they do there may be overlapping substructures that possess different fractal dimensions. As a result, the fractal dimension of a region within a structure might differ with position and with scale. The structure is then said to be multifractal. The

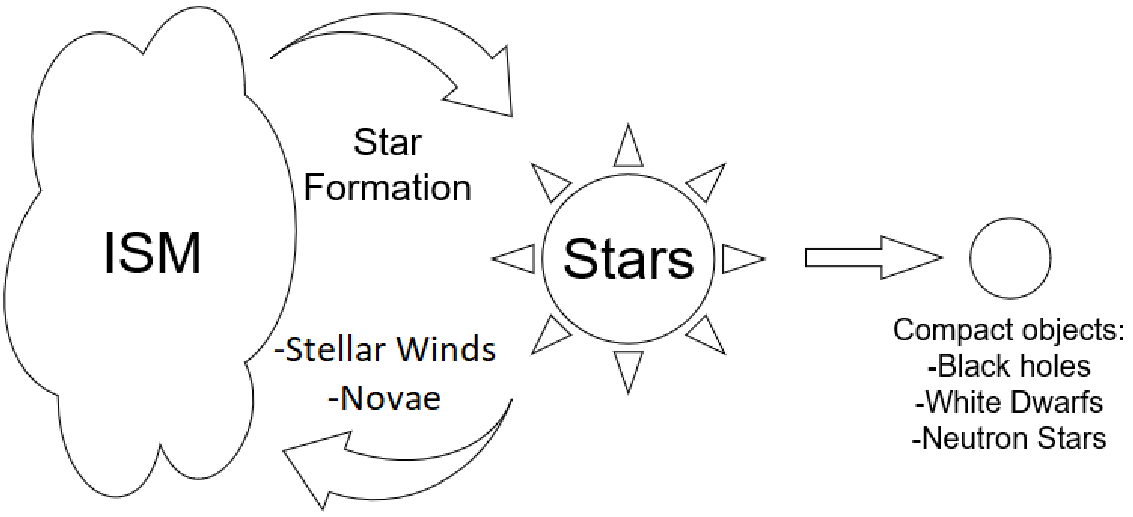


Fig. 1.1 A cartoon showing the recycling of material in the ISM. Material found in the ISM condenses into stars. Much of this material is recycled back into the ISM through stellar winds and supernovae. The material that is not returned to the ISM is locked up in compact objects such as black holes, white dwarfs, or neutron stars. Some material from the intergalactic medium also replenishes the ISM, which is not shown here.

multifractal properties of the Galactic Plane have been explored in Elia et al. (2018) and we briefly revisit this study in Section 3.5.

1.2 The ISM - Observations, Theory, and Background

1.2.1 The ISM

The Interstellar Medium (ISM) refers to the gas and dust that exists between stars, and it is from this gas and dust that stars themselves form. Molecular gas condenses into stars, which then inject material back into the ISM. Much of the material (90%) injected back into the ISM occurs during the stars' lifetime via stellar winds, while a smaller amount of material is released at the end of the stars' life, via runaway fusion (novae/supernovae). Figure 1.1 shows a basic cartoon of how the material of the ISM is recycled. Some of the material is locked up in compact objects, such as black holes, white dwarfs, or neutron stars. Supernovae, along with stellar winds from massive stars ($M \gtrsim 3M_{\odot}$), also produce the bulk of the heavy elements that are then seeded into the ISM. A large fraction of these heavy elements end up condensing into the grains that we call dust (Stahler and Palla, 2004).

The ISM consists of multiple different phases that are in a rough pressure equilibrium with each other, with each of the phases having a different density, temperature, and composition.

Since the bulk of the ISM is composed of hydrogen, then it is useful to group these phases according to the state of the hydrogen:

Atomic/Neutral Hydrogen: The portions of the ISM that contains neutral hydrogen forms the bulk of these phases by mass ($7 \times 10^9 M_\odot$) and is further split into two components; the *warm neutral medium* and the *cold neutral medium*. The warm neutral medium permeates the ISM and forms the bulk of the atomic hydrogen by volume ($\sim 90\%$). Typical values of density and temperature of this medium are $n = 0.5$ cm^{-3} and $T = 8000$ K respectively. The total mass of the warm neutral medium is $\sim 4 \times 10^9 M_\odot$. The cold neutral medium is found as distinct clouds. These HI clouds typically having a density and temperature of $n = 50$ cm^{-3} and $T = 80$ K. The total mass of the cold neutral medium is $\sim 3 \times 10^9 M_\odot$.

Ionised Hydrogen: The ISM that contains ionised hydrogen can also be described using two phases; the *warm ionised medium* and the *hot ionised medium*. The warm ionised medium has roughly the same temperature and density as the warm neutral medium with the main difference being that the warm ionised medium consists of ionised hydrogen. The total mass of the warm ionised medium is roughly $10^9 M_\odot$. The hot ionised medium is a very hot and rarefied phase and is formed due to supernova explosions. These supernovae produce bubbles of expanding material which sweep out a large volume, and, eventually connect with other supernovae remnants to create interconnected regions of this hot gas. Typical values for the density and temperature of these hot ionised regions are $n = 3 \times 10^{-3}$ cm^{-3} and $T = 5 \times 10^5$ K respectively.

Molecular Hydrogen: Regions that contain molecular hydrogen are diverse, ranging from diffuse clouds that are in pressure equilibrium with their surroundings to Giant Molecular Clouds (GMCs), where self-gravity rather than pressure maintains the equilibrium. GMCs also contain the majority of the Galaxy's molecular hydrogen (80%). Since gravity plays such an important role in maintaining a GMCs stability, this is where we see the most star formation. The total mass of molecular hydrogen in the ISM is $\sim 2 \times 10^9 M_\odot$. This accounts for around 25% of the mass of the ISM (Blitz, 1997). Molecular regions are the coldest and densest of all the phases. The mean temperature in a GMC is ~ 15 K while the mean density is 100 cm^{-3}.

An important constituent of the ISM is dust which accounts for around 1% of the mass of the ISM. Dust grains are solid particles that range from sub-micron to nanometre size. Dust is particularly important to observations as it absorbs and scatters incoming starlight. The energy absorbed by the dust grains is then re-emitted in the infrared. As a result dust grains both attenuate and redden observed starlight. Cold dust ($10K \leq T \leq 40K$) emission is particularly useful as a tracer of mass in the ISM, and has been used extensively to map the hydrogen column density along the line of sight, with the assumption that the gas to dust ratio is roughly constant throughout the Galaxy. This dust emission is optically thin in the Far-Infrared (FIR) and submillimetre range ($10^1 - 10^3\mu$m), across much of the Galaxy. As a result this dust emission only depends on the emissivity, column density, and temperature. Thus by observing the luminosity and temperature of the cold dust we can calculate the column density (Molinari et al., 2010). Observatories capable of investigating the FIR include the InfraRed Astronomical Satellite (IRAS) (Neugebauer et al., 1984), the Infrared Space Observatory (ISO) (Kessler et al., 1996), Spitzer (Werner et al., 2004), and the Herschel Space Observatory (HSO) (Pilbratt et al., 2010).

1.2.2 The ISM as observed by the *Herschel Space Observatory*

The *Herschel Space Observatory* (HSO) (Pilbratt et al., 2010) was a space telescope that was launched on the 14th of May 2009, with the aim of taking observations of the ISM in the far-infrared and the submillimetre range. *Herschel* carried three instruments; PACS (Photometric Array Camera and Spectrometer, Poglitsch et al. (2010)), SPIRE (Spectral and Photometric Imaging REceiver, Griffin et al. (2010)), and HIFI (Heterodyne Instrument for the Far Infrared, De Graauw et al. (2010)). Together these three instruments produce broad band photometry with bands centred at 70, 100, 160, 250, 350, and 500 μm (Pilbratt et al., 2010). These wavelengths perfectly capture the peak of the dust thermal emission, and as a result are particularly sensitive to temperature variations, which makes them useful for mapping the temperatures and luminosities of the ISM at greater resolutions than before. *Herschel* has been been able to map molecular clouds over a large range of spatial scales, ranging from the size of the molecular clouds themselves (> 10 pc) to individual cores (< 0.1 pc) (Beuther et al., 2014).

In this study we will mostly be working with the Via-Lactea dataset (Marsh et al., 2017). This dataset is a result of taking the Herschel Infrared GALactic Plane survey or Hi-GAL (Molinari et al., 2010) and applying `PPMAP` (Point Process Mapping) (Marsh et al., 2015) to it. The Hi-GAL study uses the PACS and SPIRE instruments to map the entire Galactic plane in five wavebands, with the PACS instrument providing photometric maps at the 70 and 160 μm and the SPIRE instrument providing the photometric maps at 250, 350 and 500 μm. The spatial resolution in these five wavebands are $8.5''$, $13.5''$, $18.2''$, $24.9''$ and $36.3''$ respectively (Elia et al., 2013; Marsh et al., 2017). The spatial resolution here refers to the full-width half maximum (FWHM) of the beam size. The Hi-GAL survey mapped the entire plane of the Galaxy, in each of these five wavebands, ranging across all Galactic longitudes ($0^\circ \leq \ell \leq 360^\circ$), and approximately 1° above and below the plane ($-1^\circ \leq b \leq +1^\circ$). The Hi-GAL survey produced maps with a different pixel size for each of the wavebands. These are $3.2''$, $4.5''$, $6.0''$, $8.0''$, and $11.5''$ for the 70, 160, 250, 350, and 500 μm wavebands respectively. In order to derive maps of column density the standard way is to perform an SED (Spectral Energy Distribution) fit to each pixel across the five wavelengths. Since the intensity maps have a different pixel resolution then the pixel grid needs to be smoothed to a common resolution and reprojected to a common pixel grid. For *Herschel* maps this is usually done to the 500μm pixel grid resolution of $11.5''$. The pixel to pixel blackbody fit is then performed according to

$$I_\lambda \simeq \kappa_\lambda B_\lambda(\bar{T})\Sigma \tag{1.1}$$

where κ_λ is the dust opacity, $B_\lambda(\bar{T})$ is Planck's law of radiation, and Σ is the integrated column density along the line of sight. The dust opacity is approximated by,

$$\kappa_\lambda = \kappa_{300}\left(\frac{\lambda}{300\mu\text{m}}\right)^{-\beta}. \tag{1.2}$$

where κ_{300} is the dust opacity at 300μm and β is the dust emissivity index which is set to 2 (Hildebrand, 1983; André et al., 2010; Elia et al., 2013; Arzoumanian et al., 2011; Sadavoy et al., 2012). We can convert the total integrated column density, Σ, to that of hydrogen via

$$\Sigma = \mu m_H N(H_2) \tag{1.3}$$

where $\mu = 2.8$ is the mean molecular weight, and m_H is the atomic mass of hydrogen. $N(H_2)$ is the hydrogen column density. Fitting to this model not only provides the column density, Σ, but also the mean line-of-sight temperature, $\bar{T}$, due to Planck's law:

$$B_\lambda(\bar{T}) = \frac{2hc^2/\lambda^5}{e^{hc/\lambda k\bar{T}} - 1}, \tag{1.4}$$

with h, c, and k being the Planck constant, the speed of light, and the Boltzmann constant respectively.

This approach to determining the column density relies on a number of assumptions. Three of these assumptions are as follows:

1. That the gas-to-dust mass ratio is constant. This ratio has been derived from empirical data and is normally assumed to be 100 (Hildebrand, 1983).
2. That the dust opacity law follows $\kappa_\lambda \propto \lambda^{-\beta}$ where β is a constant. This is seen in Equation 1.2, where it is assumed that $\beta = 2$.
3. That the dust along the line-of-sight can be represented by a single mean vibrational temperature.

Assumption 3. is a rather poor one as structures along the line-of-sight are typically not isothermal. As a result, SED fitting this way tends to overestimate the contribution from the warmer than average dust and underestimate the contribution from the cooler than average dust (Marsh et al., 2015; Jáquez-Domínguez et al., 2023; Juvela, 2023). Since this method also requires that the high resolution images from the shorter wavelength bands get smoothed to the resolution of the low-resolution images from the longer wavelength 500μm band, then a lot of the high-resolution information gets ignored.

Marsh et al. (2015) introduce the Point Process Mapping (`PPMAP`) procedure. This procedure modifies Assumption 3. and assumes instead that there is a range of dust temperatures which can be represented by Q discrete dust temperatures, T_q, between $T_{\rm MIN}$ and $T_{\rm MAX}$, where,

$$\begin{aligned} T_1 &= T_{\rm MIN}; \\ T_q &= F^2 T_{q-1}, \qquad 2 \leq q \leq Q; \\ F &= \left(\frac{T_{\rm MAX}}{T_{\rm MIN}}\right)^{1/2(Q-1)}. \end{aligned}$$

The intensity is thus approximated by summing over these discrete dust temperatures,

$$I_\lambda \simeq \sum_{q=1}^{q=Q} B_\lambda(T_q)\Delta\tau_{300:q}\left(\frac{\lambda}{300\mu\text{m}}\right)^{-\beta}. \tag{1.5}$$

Here, $\Delta\tau_{300:q}$ is the contribution to the optical depth at 300μm of the dust at temperature T_q, defined in relation to the total optical depth, τ_{300} as

$$\tau_{300} = \sum_{q=1}^{Q} \Delta\tau_{300:q}. \tag{1.6}$$

The goal of PPMAP is to estimate the values of $\Delta\tau_{300:q}$. This is done by drawing random values of $\Delta\tau_{300:q}$ from a Gaussian distribution, comparing the resulting intensities with the observed intensities, and then adjusting $\Delta\tau_{300:q}$ accordingly. For a full mathematical description of PPMAP we refer the reader to Marsh et al. (2015). Since PPMAP does not require the smoothing of observational images to a common resolution, it is able to deliver images at the finest possible resolution. Furthermore, it is able to produce both the expectation value, $\Delta\tau_{300:q}$, and the uncertainties, $\sigma_{300:q}$, for each of the temperatures, T_q.

Galactic Plane surveys, such as Hi-GAL, and Via Lactea, can let us observe the differences between the inner and outer Galaxy. One notable and observed difference is the amount of star formation, with the outer Galaxy having less star formation than the inner Galaxy, mainly due to the lower density of atomic and molecular hydrogen found in the outer Galaxy (Wouterloot et al., 1990). Another major difference is line-of-sight confusion, with the outer Galaxy having a lower degree of confusion than the inner Galaxy (Elia et al., 2013).

1.2.3 The Fractal Structure of the ISM

Multiple studies have shown that the structure of the ISM is self-similar in nature and can thus be described as a fractal (Beech, 1987; Falgarone et al., 1991; Vogelaar and Wakker, 1994; Elmegreen, 1997; Chappell and Scalo, 2001; Elmegreen, 2002; Elia et al., 2014, 2018). There is evidence to support that supersonic turbulence is a major driver behind this fractal structure (Federrath et al., 2009; Kritsuk et al., 2013; Konstandin et al., 2016; Beattie et al., 2019a,b), with the fractal structure forming as the turbulence cascades from large-scales (where it is injected) to small scales (where it is eventually dissipated as thermal energy). Since turbulence lacks a specific length scale, it produces this fractal structure over a large range of scales. Thus the range of scales of a fractal cloud can determine the injection and dissipation scales of the turbulent energy (Elia et al., 2014). For a gas that is moving at supersonic velocities ($\mathcal{M} > 1$), the energy present in this bulk motion of the gas is dissipated to the smaller scales through shocks. Here $\mathcal{M}$ refers to the mean Mach number. Turbulence can work to aid or hinder star formation as it can provide support against gravitational collapse, or it can produce the high

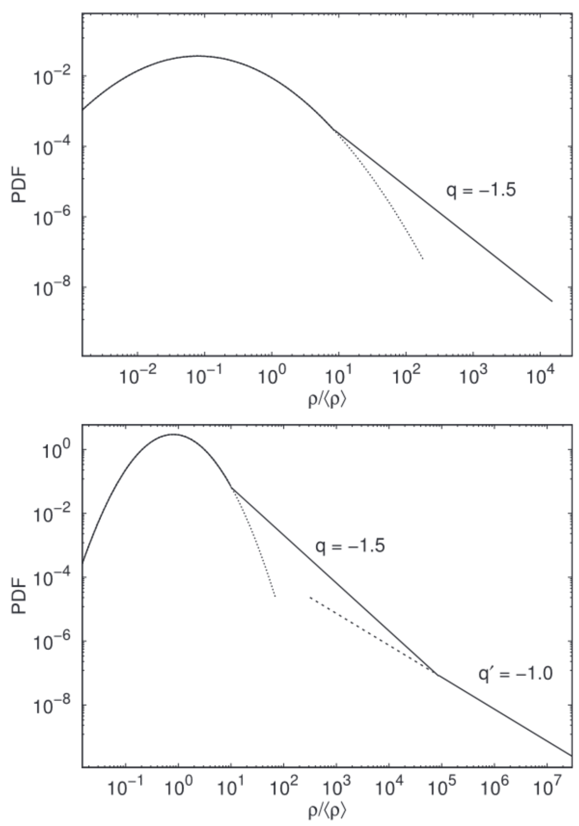


Fig. 1.2 Two examples of density PDFs from Donkov et al. (2021). The top panel shows a log-normal PDF with a power law tail, while the bottom panel shows a log-normal PDF with two power law tails.

density regions where star-formation occurs (Kainulainen et al., 2009; Schneider et al., 2012; Könyves et al., 2015; Konstandin et al., 2016).

An important tool for determining whether or not turbulence is a driver for the fractal structure is the distribution of the column density. Numerous studies have shown that this distribution of the column-density is generally log-normal if the fractal structure is driven purely by isothermal supersonic turbulence, however if self-gravity plays a large role one or more power law tails also develop (Klessen et al., 2000; Vazquez-Semadeni and Garcia, 2001; Kritsuk et al., 2011; Girichidis et al., 2014; Ward et al., 2014; Schneider et al., 2015; Krumholz and McKee, 2020; Jaupart and Chabrier, 2020; Donkov et al., 2021; Schneider et al., 2022). Figure 1.2 shows how this probability density distribution would look with one or two power law tails. The column density represented here is the mass column density, ρ, while the slope of each power law tail is given as q. The second, flatter power law tail tends to develop as self-gravity starts to dominate the small spatial scales ($\lesssim 1$ pc), and corresponds to rotationally supported material such as a disc (Khullar et al., 2021).

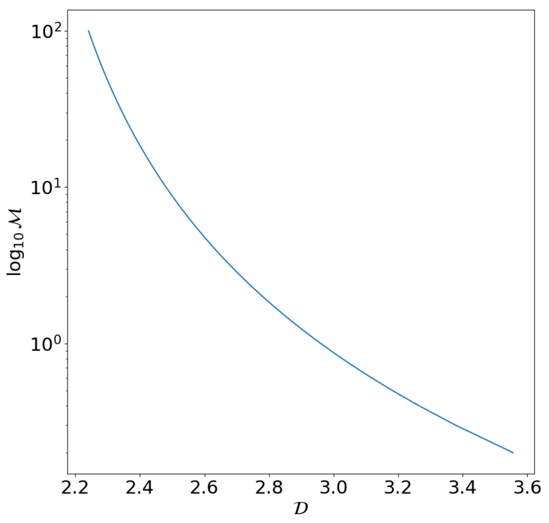


Fig. 1.3 The relation between $\mathcal{M}$ and $\mathcal{D}$ as derived by Konstandin et al. (2016).

Some effort has been made to directly link turbulence to the fractal structure of the ISM by relating the mean Mach number, $\mathcal{M}$, to the fractal dimension, $\mathcal{D}$. The Mach number is important as it can be used to constrain the column density distribution, as well as the star formation rate (Konstandin et al., 2016). Konstandin et al. (2016) use numerical simulations to model driven, compressible, and isothermal turbulence, using both compressive and solenoidal turbulence modes. These simulations are then used to derive the relation

$$\mathcal{D} = 2 + \frac{\alpha_{\mathcal{M}}}{2}\mathcal{M}^{\beta_{\mathcal{M}}}, \tag{1.7}$$

where $\alpha_{\mathcal{M}} = -1.91 \pm 0.001$ and $\beta_{\mathcal{M}} = -0.30 \pm 0.03$. Figure 1.3 shows a plot of this relation,

Beattie et al. (2019b) also use numerical simulations to derive a relation between the fractal dimension and the mean Mach number, $\mathcal{M}$. However it is important to note that the fractal dimension here refers to the mass-length scaling dimension, $\mathcal{D}_p$, which controls the relationship between the mass, m and the length-scale, ℓ/L, by the relation

$$m(\ell/L) \sim (\ell/L)^{\mathcal{D}_p}. \tag{1.8}$$

Nonetheless these two studies (Konstandin et al., 2016; Beattie et al., 2019b) both find that the fractal dimension roughly decreases as the mean Mach number increases. Since the fractal dimension is a measure of how efficiently a structure fills its space, then this finding is consistent with the fact that diffuse structures with low mean Mach number have a high fractal dimension, while structures with a high Mach number, i.e. shocks/filaments/bubbles, have a low fractal dimension. On a more granular level, Brunt and Federrath (2014) use numerical simulations to describe how each of the two supersonic turbulence modes (solenoidal and compressible) can affect the density and velocity structure of molecular clouds.

In contrast, de Vega et al. (1996) suggest that fractal structure can emerge as a result of self-gravity alone.

1.3 Why use Machine Learning?

The recent explosion in the amount of data being generated and collected in the world has gone hand-in-hand with a large increase in the research and application of novel statistical techniques that can handle and analyse these large amounts of data. Most notable is the accelerated pace of research into new Machine Learning techniques and their implementation. New and more powerful processing technology has also become more readily available, which has made the processing and analysis of this data easier. This includes the advent of Graphical Processing Units (GPUs) that are capable of implementing these new techniques and their subsequent deployment on large-scale cloud computing platforms.

This vast increase in the generation of data is also seen in the field of Astronomy. For example, the Square Kilometre Array (SKA) is a radio observatory that is projected to come online in 2027 and when completed will generate 20 terabits of data per second while the comparable Atacama Large Millimetre Array (ALMA) produces around 20 gigabits per second of data[1], an increase of 3 orders of magnitude. This type of increase in dataset size, necessitates the use of algorithms that can reduce and analyse these large datasets quickly and efficiently. As a result of this explosion in dataset sizes there has also been an increase in the application of Machine Learning techniques to astronomical problems (Dieleman et al., 2015; Flamary, 2016; Remez et al., 2017; Khalifa et al., 2017; Kimura et al., 2017; Postnikov, Postnikov; Bialopetravičius et al., 2019; Dawson et al., 2019, 2021). In this thesis we leverage these Machine Learning techniques in order to process large amounts of image data from the *Herschel Space Observatory* (Pilbratt et al., 2010).

We concern ourselves with the fractal structure of the Galactic Disk and thus make use of the

[1]`https://www.skao.int/en/explore/big-data`

Herschel infrared Galactic Plane (Hi-GAL) survey (Molinari et al., 2010). This survey consists of 163 tiles each containing a $2.4^\circ \times 2.4^\circ$ area of the Galactic Plane. Each tile has 1440×1440 pixels across 5 wavebands (70μm, 160μm, 250μm, 350μm, 500μm). The dataset we use thus consists of $163 \times 1440 \times 1440 \approx 3.4 \times 10^8$ pixels. Each of these tiles has been converted into maps of column-density using the `PPMAP` algorithm (Marsh et al., 2015, 2017).

One way of estimating fractal parameters is through the use of Delta-Variance (Stutzki et al., 1998; Ossenkopf et al., 2008a). However, this algorithm is computationally expensive and requires some human input (See Chapter 3 for more detail). It is also only possible to get one value of the so-called Hurst parameter, $\mathcal{H}$, for an input region, no matter the size of this region. However, we are interested in producing maps of $\mathcal{H}$ in order to infer some statistics of the fractal properties of the ISM. Using the Hi-GAL data as an input would result in $\sim 10^8$ values of $\mathcal{H}$. This is not feasible with the computationally expensive Delta-Variance and thus an estimator that can produce values of $\mathcal{H}$ quickly and with a certain degree of accuracy is required. To address this need, we have designed a Machine Learning model that can be used as a quick and reliable estimator.

Supervised Machine Learning models, in particular Neural Networks, are a form of general estimators that can be used to perform either classification tasks, or regression. The term *supervised* simply means that before being deployed, the estimator must be *trained*. This is done by running a series of inputs (known as features) and their known corresponding outputs (known as labels) through the model, comparing the resulting outputs to the known ones, and adjusting the model parameters as needed. In our study, the features would be images of fractal structures while the labels would be each image's corresponding value of $\mathcal{H}$. We also introduce an additional parameter, $\mathcal{S}$, which is an exponentiation factor that controls the scaling of the fields. Therefore each field has both $\mathcal{H}$ and $\mathcal{S}$ as labels. The training of a Machine Learning model might be computationally expensive compared to other techniques such as Delta-Variance, but it only needs to be done once. Once trained, a Machine Learning model is able to estimate the labels of new inputs using less computational resources than required to train it. See Chapter 4 for more details on how the Neural Network used in this thesis is constructed, trained, and tested.

As a result of this increase in speed, we use a Machine Learning model called a Convolutional Neural Network (CNN) to produce an estimator that takes subregions of the Hi-GAL derived tiles of column density and estimates $\mathcal{H}$ and $\mathcal{S}$. By passing all possible subregions to the CNN we can produce multiple values of $\mathcal{H}$ and $\mathcal{S}$, and use these to construct maps of $\mathcal{H}$ and $\mathcal{S}$.

Examples of the use of Machine Learning techniques in the field of astronomy include: star cluster analysis (Lomax et al., 2018; Bialopetravičius et al., 2019), galaxy and supernova classification (Khalifa et al., 2017; Kimura et al., 2017), image reconstruction (Flamary, 2016), the study of galaxy kinematics (Dawson et al., 2019, 2021), gravitational wave detection (Zevin et al., 2017; Gabbard et al., 2018; George and Huerta, 2018; Shen et al., 2019), and exoplanet detection (Shallue and Vanderburg, 2018).

1.4 Thesis Plan

In Chapter 2 we give a brief description of the theory used to describe fractal structures. We look at some examples of fractal structures and describe some statistics that can be used to characterise fractals. We show how to construct fractional Brownian motion (fBm) fields using the Spectral Synthesis method and how this can be extended in order to construct exponentiated fBm (xfBm) fields. We explain the significance of the Hurst parameter, $\mathcal{H}$ and the exponentiating factor, $\mathcal{S}$.

In Chapter 3 we investigate the established techniques that can be used to determine $\mathcal{H}$ and $\mathcal{S}$. For determining $\mathcal{H}$ we investigate both the Perimeter-Area method and the Delta-Variance method. For the Delta-Variance method, we investigate the effect of the periodicity of the fields on its performance. We also investigate the case where the fields are multi-fractal.

In Chapter 4 we propose using a Convolutional Neural Network in order to estimate the values of $\mathcal{H}$ and $\mathcal{S}$. We design, train and test three different CNNs with each one being able to take images of a different linear pixel size, $N_{\text{PIX}} = [32, 64, 128]$. We compare the performance of these three CNNs with that of the methods explored in Chapter 3.

In Chapter 5 we apply the CNNs trained in Chapter 4 to observational data. We investigate two datasets that have both been generated using the `PPMAP` algorithm (Marsh et al., 2015). The first dataset we analyse in this chapter is one based on the Hi-GAL survey, which is a survey of the Galactic Plane (Molinari et al., 2010; Marsh et al., 2017). We then analyse `PPMAP` generated images of the Andromeda Galaxy (M31).

In Chapter 6 we summarise our conclusions and discuss future work.

CHAPTER 2

Exponentiated Fractional Brownian Motion Fields

2.1 Preamble

We have already explained in the previous Chapter (Chapter 1) how the processes that govern the evolution of the interstellar medium tend to be chaotic in nature. Examples of these processes include magnetohydrodynamics, turbulence, and self-gravity. This means that the structure of the ISM tends to be random in nature, but possess an overall statistical self-similarity that can be seen across a large range of spatial scales (Stutzki et al., 1998; Elmegreen, 2002; Ossenkopf et al., 2008b; Shadmehri and Elmegreen, 2011; Elia et al., 2014, 2018). In other words, the structure of the ISM has a fractal geometry.

A number of fractal models have been proposed (Voss, 1985; Hetem and Lepine, 1993; Stutzki et al., 1998; Elmegreen, 2002; Sanchez et al., 2005; Federrath et al., 2009; Shadmehri and Elmegreen, 2011), the majority of which build the fractal structure through numerical or recursive means. For example, Hetem and Lepine (1993), produce three different fractals models, two of which involve recursively subdividing a volume into smaller cubes and the third involves an iterative process applied to a grid. Conversely, Stutzki et al. (1998), find that observed molecular clouds have structures that are well characterised by fractional Brownian motion (fBm) images which are a non-recursive and non-iterative way of producing fractal structures. In this work, we focus on the fractal structures that are generated by a modified version of fBm which we call Exponentiated Fractional Brownian Motion (xfBm).

2.2 Theory of Fractal Images

There is no hard definition of what makes an image/structure fractal. The general consensus is that it must have a large level of detail that is self-similar over a large range of scales. There are two definitions of self-similarity; fractals can be either exactly self-similar (e.g. the von Koch curve, the Sierpinski triangle), or statistically self-similar (e.g. fractal clouds, a coastline). An exactly self-similar fractal is a pattern that repeats exactly at increasingly smaller scales while a statistically self-similar fractal is one where the structure looks similar at different scales but possesses a degree of randomness.

Figure 2.1 shows a von Koch curve which is an example of an exactly self-similar fractal. The rest of Figure 2.1 (a-d) shows how the von Koch curve is constructed using a recursive process. Starting with a straight line, (Figure 2.1a), the line is split into four equal length lines, with the two central lines forming the two sides of an equilateral triangle (Figure 2.1b). For a true fractal, this process is repeated on each of the resulting lines an infinite amount of times, however since we are limited by reality we must define a lower limit to this process. In the case of the von Koch fractal this lower limit is defined by arbitrarily halting the process, and in the case of the fractal produced in Figure 2.1 this process is stopped after four steps. The upper limit of the von Koch fractal and indeed many other generated fractals is simply the size of the bounding box that the fractal resides in.

Defining this length-scale is equally important when constructing a statistically self-similar fractal such as a 2-dimensional xfBm field. Since a 2D xfBm field is generated on a grid of pixels, the small scale limit of this xfBm is thus the size of each pixel, while the large scale limit will be the number of pixels. This defines the dynamic range, $\mathcal{R}$, of the xfBm. The dynamic range defines the range of spatial scales over which the fractal model applies.

When dealing with a fractal structure, it is helpful to evaluate the structure's fractal dimension, $\mathcal{D}$. This is roughly a measure of how efficiently the structure fills its Euclidean space. There are numerous different definitions of $\mathcal{D}$, such as the Box-Counting fractal dimension, $\mathcal{D}_{BC}$, and the Perimeter-Area fractal dimension, $\mathcal{D}_{PA}$.

For a 2-dimensional fractal image (such as a cloud generated by fBm), the $\mathcal{D}_{PA}$ is estimated by measuring the perimeters, P, and areas, A, of a set of contours that have been constructed at different density levels. The relation between these terms is given by,

$$P \propto A^{\mathcal{D}_{PA}/2}. \tag{2.1}$$

Thus by plotting $\log P$ against $\log A$ we can derive $\mathcal{D}_{PA}$.

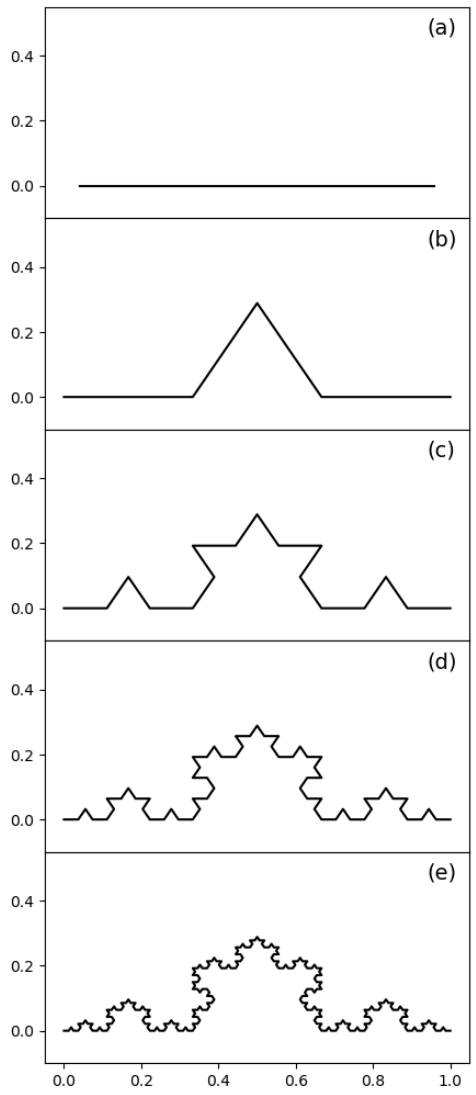


Fig. 2.1 The iterative process that produces a von Koch curve. Each line segment is split into 4 new line segments. This is best illustrated in the differences between panels (a) and (b). This process is then repeated for each line segment many times. A true von Koch curve is infinitely recursive, however in the real world we must define a limit to these recursive steps.

To calculate $\mathcal{D}_{BC}$, a grid of $\mathcal{E}$-dimensional boxes are placed over the structure. For a fractal structure with well defined edges, the number of these boxes, $N(r)$, that cover the fractal structure, give the fractal dimension according to,

$$N(r) \propto r^{-\mathcal{D}_{BC}}, \tag{2.2}$$

where r is the length of each side of each box. For structures that do not have a well-defined edge, such as a continuum astronomy image, a lower limit of the image intensity must be defined. Elia et al. (2018) show that as this limit goes to the global minimum intensity, $\mathcal{D}_{BC}$ approaches $\mathcal{E}$. Here $\mathcal{E}$ is simply the Euclidean dimension, in which the $\mathcal{E}$-dimensional fractal structure resides.

Of the two measures ($\mathcal{D}_{PA}$ and $\mathcal{D}_{BC}$), $\mathcal{D}_{BC}$ is considered the noisier as it relies on several arbitrary factors, such as the position and orientation of the grid of boxes and, in the case of astronomical images, the lower intensity limit that is used to define the structure. There has been some work that attempts to relate the two measures, in particular Stutzki et al. (1998) find that generally $\mathcal{D}_{PA} \simeq \mathcal{D}_{BC} - 1$. For the rest of this text we shall assume the Perimeter-Area dimension when referring to the fractal dimension, i.e. $\mathcal{D} = \mathcal{D}_{PA}$.

2.3 Fractional Brownian Motion

In Section 2.1 we mention that Fractional Brownian Motion fields (fBm) are a statistically self-similar structure that have been used to characterise molecular clouds as well as the interstellar medium (ISM) (Stutzki et al., 1998; Elmegreen, 2002; Shadmehri and Elmegreen, 2011; Elia et al., 2014, 2018). Peitgen and Saupe (1988) describe the theory behind fBm fields and also provide several algorithms to generate them, including the mid-point displacement method and the spectral synthesis method. All fBm and xfBm curves or fields in this work are generated using the spectral synthesis method. The spectral synthesis method is expanded in detail in Section 2.3.3

fBm fields are a generalisation of classical Brownian motion and are defined by three parameters: the Euclidean dimension, $\mathcal{E}$, the dynamic range, $\mathcal{R}$, and the power spectral index, β. The power spectral index, β, and the Euclidean dimension, $\mathcal{E}$, can be combined into a single parameter,

$$\mathcal{H} = \frac{\beta - \mathcal{E}}{2}, \tag{2.3}$$

where $\mathcal{H}$ is known as the Hurst parameter, and takes values $0 \leq \mathcal{H} \leq 1$. $\mathcal{H}$ can be thought of as a smoothing parameter, where as $\mathcal{H} \to 1$, the curve becomes smoother and as $\mathcal{H} \to 0$ the curve becomes rougher. Classical Brownian motion is a special case of fBm where $\mathcal{H} = 0.5$. Stutzki et al. (1998) show that

$$\mathcal{D} \simeq (\mathcal{E} - \beta)/2 = \mathcal{E} - \mathcal{H}. \tag{2.4}$$

The dynamic range, $\mathcal{R}$, describes the range of scales over which the fractal structure is defined. In the case of the artificially generated fBm that are presented in this work, $\mathcal{R}$ will be defined in terms of pixels, such that $\mathcal{R} = [1, N_{\text{PIX}}]$. As a result it is only necessary to define the large scale limit of these fractals, since the small scale limit is simply a single pixel although we must note that in an astronomical image the real lower limit will be the beam size. This large scale limit will be the linear size of the bounding box and is given by N_{PIX}. N_{PIX} is simply the number of pixels in each dimension. In this thesis we will only deal with fields with $N_{\text{PIX}} = 1000, 128, 64,$ or 32.

2.3.1 One-dimensional ($\mathcal{E} = 1$) fBm.

Figure 2.2 shows three examples of one-dimensional ($\mathcal{E} = 1$) fBm curves, $f(x)$. The top panel shows the case for when $\mathcal{H} = 0$, and the bottom panel shows the curve for when $\mathcal{H} = 1$. The middle panel shows the special case for classical Brownian motion (i.e. $\mathcal{H} = 0.5$). These curves have been generated using the same random seed so as to preserve the general shape. We can see that when $\mathcal{H} = 1$ the curve is smoothest and when $\mathcal{H} = 0$ the curve is roughest. This can also be explained using the power spectrum by way of the β parameter. A high β means the the power spectrum has more power on large scales and results in a smooth curve. For a lower β, there is less power on large scales and thus the curve is rougher.

The Hurst parameter $\mathcal{H}$ relates the typical change in $f(x)$, i.e. Δf, to the increment Δx using the simple scaling law (Peitgen and Saupe, 1988)

$$\Delta f \propto \Delta x^{\mathcal{H}}. \tag{2.5}$$

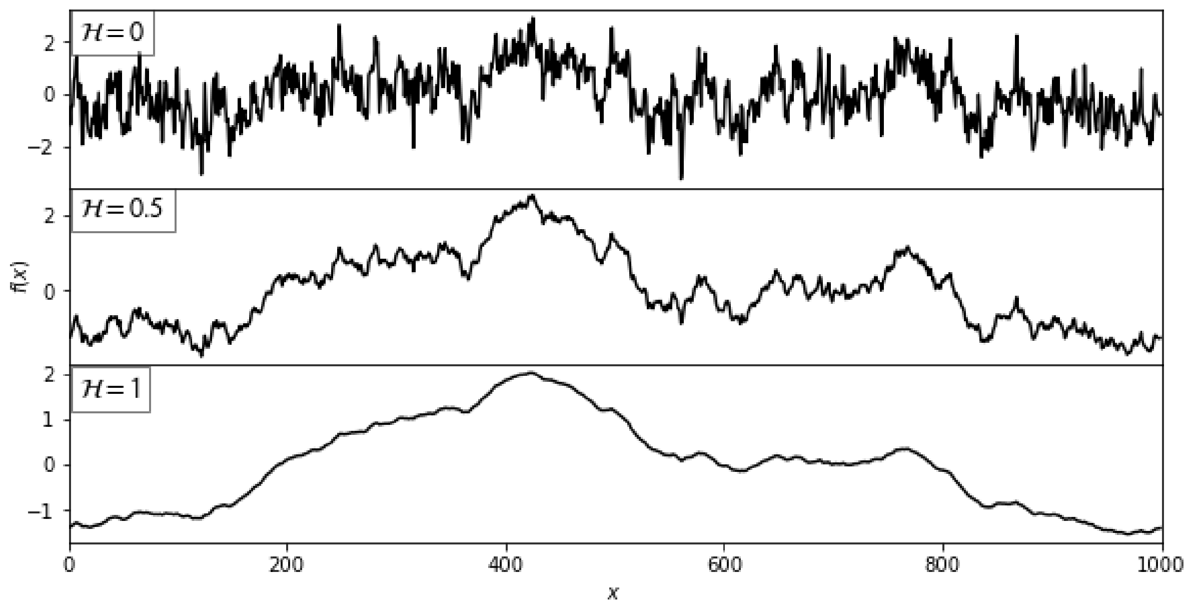


Fig. 2.2 Three examples of 1-dimensional ($\mathcal{E} = 1$) fractional Brownian motion curves for differing values of $\mathcal{H}$. *Top*: $\mathcal{H} = 0$. *Middle*: $\mathcal{H} = 0.5$. *Bottom*: $\mathcal{H} = 1$.

This relationship splits the fBm into three regimes;

- $\mathcal{H} > 0.5$: The increment, Δf is correlated with $f(x)$, producing a smooth curve.
- $\mathcal{H} < 0.5$: The increment, Δf is *anti*-correlated with $f(x)$, producing a rough curve.
- $\mathcal{H} = 0.5$: The increment is $\Delta f \propto \sqrt{\Delta x}$. This is the relationship that produces Classical Brownian motion. In this case Δf can be approximated well by a random Gaussian increment.

Taking the Fourier Transform of $f(x)$ produces a power spectrum, $\hat{f}(k)$, where each k is a wave-vector. x takes integer values of $1 \leq x \leq N_{\text{PIX}}$, while k takes integer values of $-N_{\text{PIX}}/2 \leq k \leq N_{\text{PIX}}/2$. Here N_{PIX} is the number of steps in total that have been taken in order to produce $f(x)$. For higher dimensions this will be the size in pixels of the resulting field. Figure 2.3 shows the power spectra of the three curves shown in Figure 2.2. $\hat{F}(k)$ in this figure is simply $\hat{F}(k) = ||\hat{f}(k)||$. This figure shows that there exists a power law relation between $\hat{F}(k)$ and k with exponent $-\beta$. In Figure 2.3 we distinguish between β (and by Equation 2.3, $\mathcal{H}$) used to generate $f(x)$, that we label β_{TRUE}, and β measured from the slope of $\hat{F}(k)$, which we label β_{EST}. We can see that, for 1-dimensional fBm, β_{EST} agrees with β_{TRUE}.

The top row of Figure 2.4 shows the distribution of x for each of the curves shown in Figure

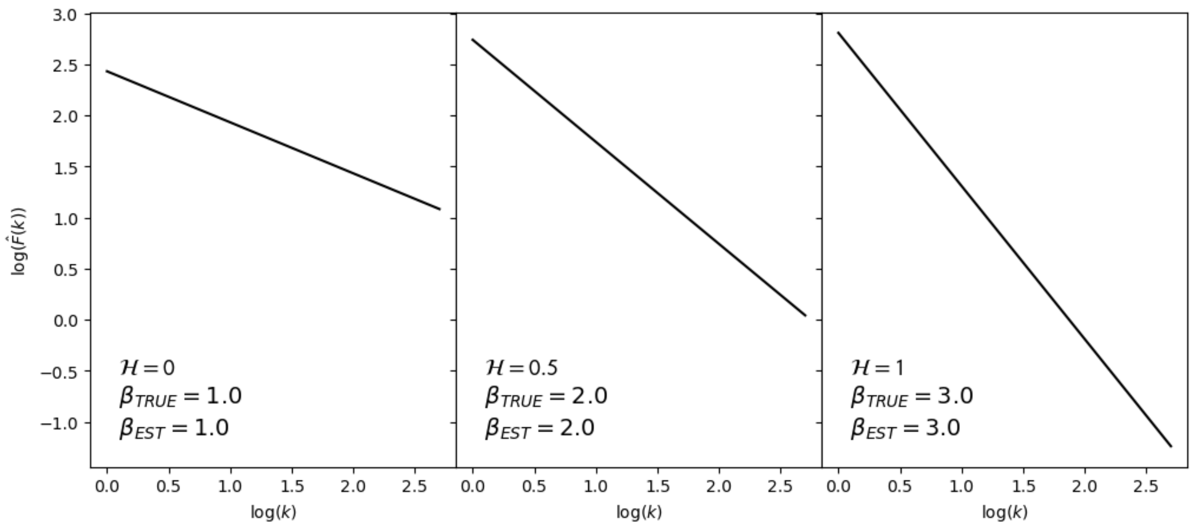


Fig. 2.3 The power spectra of the three curves shown in Figure 2.2. These power spectra have been measured using a Fast Fourier Transform (FFT).

2.2. This shows an interesting property of 1-dimensional fBm, that is, that fBm generated curves have a mean of 0 (shown by the red line) and thus have both positive and negative values. Another feature is that as $\mathcal{H} \to 0$, the distribution of x becomes more Gaussian. The bottom row shows the distribution of the random increments Δf. These are all roughly Gaussian with a mean of 0. The main difference between these distributions is the range of values the Δf takes, with this range being smaller as $\mathcal{H} \to 1$.

2.3.2 Two-dimensional ($\mathcal{E} = 2$) fBm.

In the rest of this thesis we shall work in 2-dimensions ($\mathcal{E} = 2$). Extending, $f(x)$ to 2-dimensions we get the 2-dimensional fBm field, $f(\mathbf{r})$, where $\mathbf{r} \equiv (r_1, r_2)$ is a 2-dimensional grid of integers of values of $1 \leq r_i \leq N_{PIX}$. N_{PIX} is the size in pixels of each of the dimensions of the field (i.e. $N_{\rm PIX} \times N_{\rm PIX}$ in the case of $\mathcal{E} = 2$).

Figure 2.5 show 2-dimensional fBm fields for different values of $\mathcal{H}$ (left: $\mathcal{H} = 0$, centre: $\mathcal{H} = 0.5$, right: $\mathcal{H} = 1$). The fields in this Figure have $N_{\rm PIX} = 1000$. Similar to the 1D case, we can see that as $\mathcal{H} \to 1$ the field is smooth with more large scale structure than small scale structure. Conversely when $\mathcal{H} \to 0$ we can see that the field is rougher with a lot of small scale structure and hardly any large scale structure. The three fields here have been generated with the same random seed so as to preserve the general shape of the fields. It is also evident that these fields are periodic. This is a consequence of the spectral synthesis method which we shall explore in more detail in Section 2.3.3.

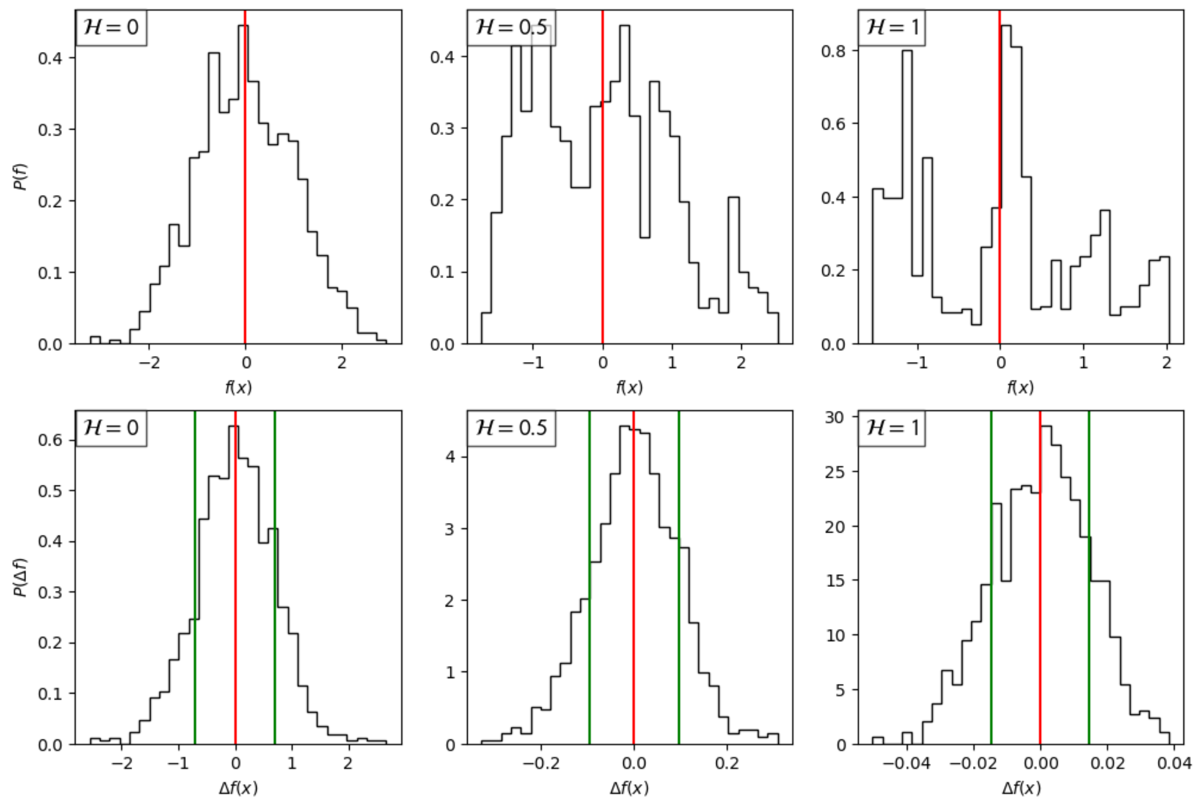


Fig. 2.4 *Top:* The distribution of the points that make up $f(x)$, for the three curves shown in Figure 2.2. The red lines give the mean of the distribution which in the case of all three figures is 0. *Bottom:* The distribution of the increments, Δf, these are all Gaussian with a mean of 0 (red line). The green lines give the standard deviation of these Gaussians. The variances of these three distributions are, 4.86×10^{-1}, 9.29×10^{-3}, and 2.11×10^{-4}, from left to right.

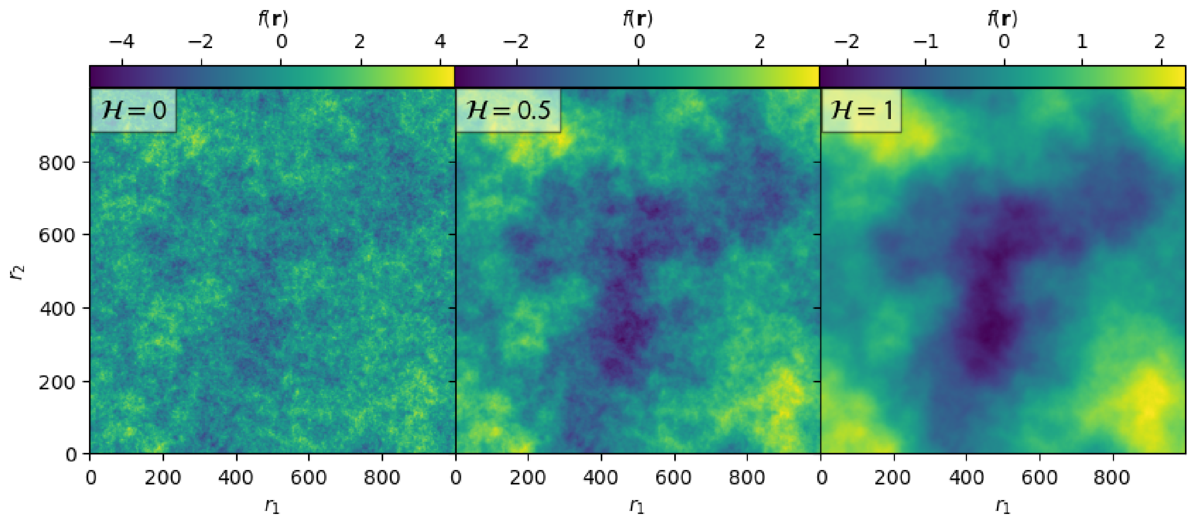


Fig. 2.5 Three 2D ($\mathcal{E} = 2$) fBm images for different values of $\mathcal{H}$ (From left to right: $\mathcal{H} = 0$, $\mathcal{H} = 0.5$, $\mathcal{H} = 1$). All three fields have been generated using the same random seed in order to preserve the same general shape.

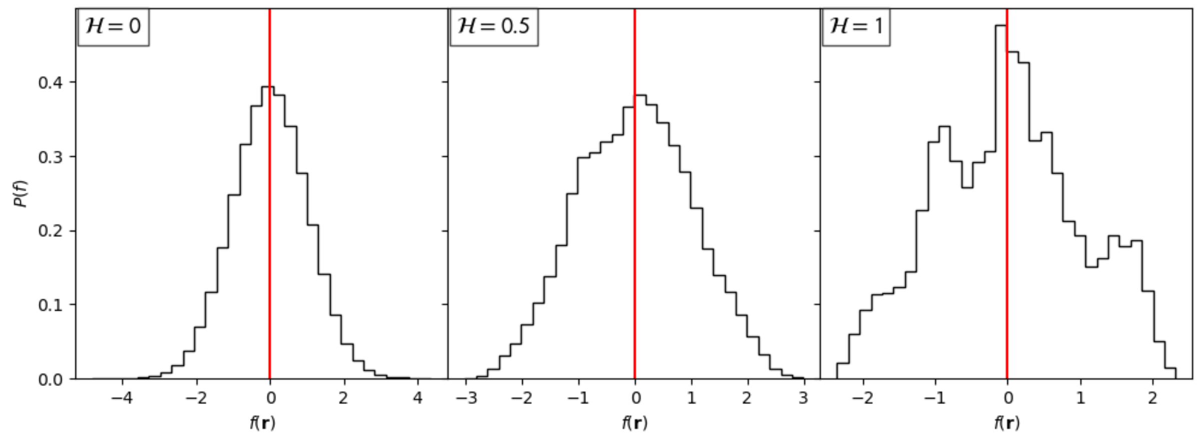


Fig. 2.6 The distributions of the pixel values of the 2D fBm fields shown in Figure 2.5. The mean is shown as a red vertical line. For all fields this is shown to be 0.

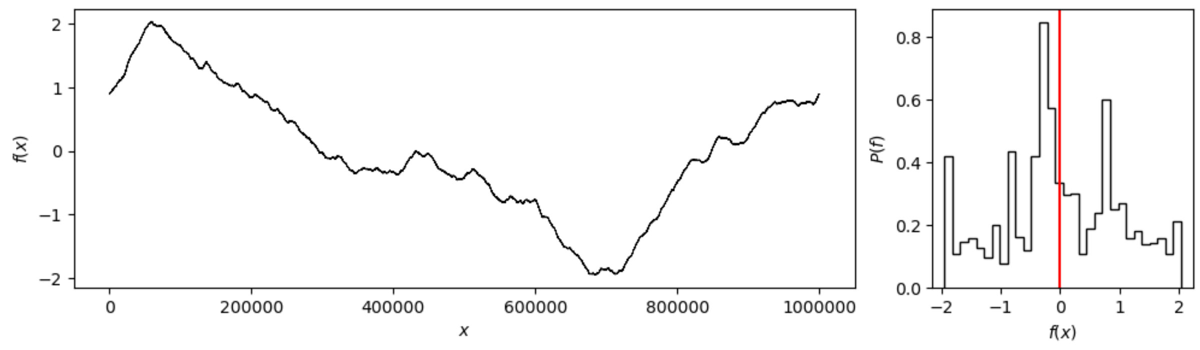


Fig. 2.7 *Left:* A 1-dimensional ($\mathcal{E} = 1$) fBm curve that has $N = 10^6$ and $\mathcal{H} = 1$. *Right:* Its distribution, with the mean marked with a red vertical line, which is 0

There is also a difference in the distribution of the pixel values, $f(\mathbf{r})$, which is shown in Figure 2.6. Similar to the 1-dimensional case we see that the two-dimensional fields also have a mean of 0, shown by the red vertical lines. We also see that as $\mathcal{H} \to 0$ the distribution of $f(\mathbf{r})$ tends towards a Gaussian. There is a slight difference in the distributions from the 1D case: As $\mathcal{H} \to 1$, the distribution remains roughly Gaussian for 2D fBm compared to 1D fBm. One possible explanation for this difference is the resolution of the curves. The 1D curves shown in Figure 2.2 have $N_{\text{PIX}} = 10^3$ total points while the 2D curves shown here have $(N_{\text{PIX}})^2 = 10^6$ total points. Figure 2.7 shows a 1D fBm with $N_{\text{PIX}} = 10^6$ and $\mathcal{H} = 1$ as well as its distribution. Thus we can see that increasing the resolution of the curve does not affect its distribution as it is still not Gaussian.

Taking the Fourier Transform of $f(\mathbf{r})$ produces the power spectrum $\hat{f}(\mathbf{k})$ where $\mathbf{k} \equiv (k_1, k_2)$, is a grid of wave-vectors taking integer values of $-N_{\text{PIX}}/2 \leq k_i \leq N_{\text{PIX}}/2$ for each ith direction. The top row in Figure 2.8 shows $\hat{F}(\mathbf{k})$ of each of the fields in Figure 2.5, where

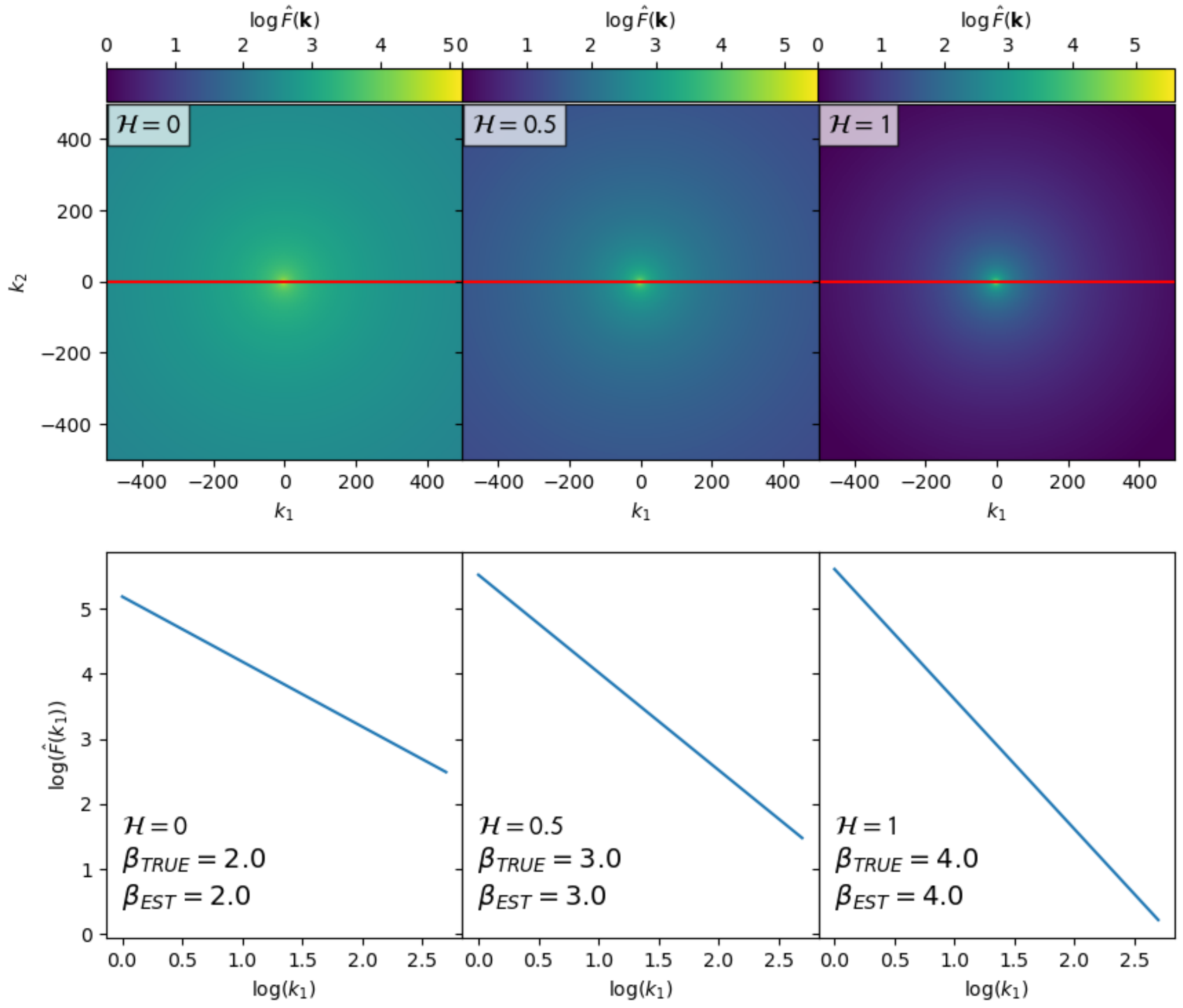


Fig. 2.8 The top row shows the amplitudes $\hat{F}(\mathbf{k})$ of the 2D power spectra of the fields shown in Figure 2.5. The plots in the bottom row show a cross section of each of the 2D $\hat{F}(\mathbf{k})$ plots in the top row. The position of each cross-section is given by the red lines. The slope of each of the graphs has been used to calculate β_{EST}.

$\hat{F}(\mathbf{k}) = ||\hat{f}(\mathbf{k})||$. $\hat{F}(\mathbf{k})$ is symmetrical around $\mathbf{k}_0 = \mathbf{0}$. By taking a cross-section through $\mathbf{k}_0$ we can deduce the β and thus $\mathcal{H}$. In this instance we take a linear cross-section at $k_2 = 0$ (shown by the red lines). The bottom row shows the respective plots of these cross-sections, from which we can deduce β_{EST}. As with the 1D case we see that β_{EST}, agrees with β_{TRUE}.

2.3.3 Spectral Synthesis

All of the fBm curves used in this work have been generated using the Spectral Synthesis method, which is a non-recursive way of generating an fBm. Recursive methods are highly dependent on the initial state of the system, and as a result are noisier than non-recursive methods.

We have already seen how fBm have a well defined power spectrum and thus the main idea of spectral synthesis is that this power spectrum is constructed first, and then an Inverse Fourier Transform is applied which generates the fBm. We shall denote an fBm field as $f(\mathbf{r})$ and its power spectrum as $\hat{f}(\mathbf{k})$.

We start with a grid of integer wave-vectors $\mathbf{k} = (k_1, k_2)$, which take values $-N_{\text{PIX}}/2 \leq k_i \leq N_{\text{PIX}}/2$. The power spectrum is then given by,

$$\hat{f}(\mathbf{k}) = A(\mathbf{k})[\cos\phi(\mathbf{k}) + i\sin\phi(\mathbf{k})]. \tag{2.6}$$

The amplitudes, $A(\mathbf{k})$, are given by

$$A(\mathbf{k}) = \begin{cases} 0, & \text{if } \mathbf{k} = \mathbf{0} \\ \mathcal{K}^{-1/2}||\mathbf{k}||^{-\beta/2}, & \text{if } \mathbf{k} \neq \mathbf{0} \end{cases} \tag{2.7}$$

where, $\mathcal{K}$ is a normalisation factor that scales the total power to unity, and β is the power-law index. These are given by,

$$\mathcal{K} = \sum_{\mathbf{k}} ||\mathbf{k}^{-\beta}||; \tag{2.8}$$

$$\beta = \mathcal{E} + 2\mathcal{H}. \tag{2.9}$$

The phases, $\phi(\mathbf{k})$, are given by,

$$\phi(\mathbf{k}) = \chi(\mathbf{k}) - \chi(-\mathbf{k}) \tag{2.10}$$

where $\chi(\mathbf{k})$ is a random variate that is drawn from a uniform distribution and takes values $0 \leq \chi(\mathbf{k}) \leq 2\pi$. Equation 2.10 ensures that $\hat{f}(\mathbf{k})$ is the complex conjugate of $\hat{f}(-\mathbf{k})$. This is done in order to ensure that $f(\mathbf{r})$ only has real values. Performing an inverse Fourier Transform on $\hat{f}(\mathbf{k})$ produces the fBm field $f(\mathbf{r})$.

This process produces a field that is periodic. Since we want to model observed fields that are not periodic, we need to deal with this periodicity. To do this we generate an intermediary field, $f'(\mathbf{r}')$, with four times as many pixels along each axis, i.e. $1 \leq r'_i \leq 4N_{\text{PIX}}$, and then select and cut out a $N_{\text{PIX}} \times N_{\text{PIX}}$ section.

2.4 Exponentiated fBm

We have already seen how fBm have a roughly Gaussian distribution with a mean of around zero, $\langle f(\mathbf{r})\rangle \sim 0$. As a result fBm fields contain negative values. If we want to model fields that are positive only (such as column density), then we need to make the fBm field positive everywhere. There are a few ways of doing this.

The simplest way to do this is to normalise the field by shifting and scaling the field such that it has particular values for its minimum and maximum. Common values of the minimum and maximum are 0 and 1, respectively (Shadmehri and Elmegreen, 2011). Another way of normalising the field is to set its minimum and mean to particular values such as 0 and 0.5 respectively. However both of these techniques tend to produce noisy results as they can be quite sensitive to outliers.

A better technique is to exponentiate the field according to

$$g(\mathbf{r}) = \exp\left\{\frac{\mathcal{S} f(\mathbf{r})}{\langle f^2(\mathbf{r})\rangle^{1/2}}\right\}. \tag{2.11}$$

$\mathcal{S}$ is a scaling factor that controls the dynamic range of the field, and can take any value. Since Equation 2.11 is a one-to-one function the fractal properties of the field are unchanged. At $\mathcal{S} = 0$ the field is simply a flat field with a uniform value of unity. As $\mathcal{S}$ increases the dynamic range increases and the high-intensity structures gain contrast. $\mathcal{S}$ is closely related to the standard deviation of $\log g(\mathbf{r})$ (Peitgen and Saupe, 1988; Stutzki et al., 1998; Elmegreen, 1997; Lomax et al., 2018; Bates et al., 2020). We refer to fBm fields that have undergone this process as Exponentiated fBm fields, or xfBm.

Figure 2.9 shows how the structure of 2D xfBm fields vary with both $\mathcal{H}$ and $\mathcal{S}$. $\mathcal{H}$ decreases from top to bottom while $\mathcal{S}$ increases from left to right. Shown are nine fields with $\mathcal{H} = [0, 0.5, 1]$ and $\mathcal{S} = [0.1, 0.5, 2]$. All nine fields have been generated with the same random seed in order to preserve the general shape. The fields have also been normalised such that the field has value $0 \leq g(\mathbf{r}) \leq 1$. Each field has had its centre of mass shifted to the centre of the image. There are also contour lines that show the value of the field at those contours. We can clearly see that when $\mathcal{S}$ increases the structure becomes less evenly distributed and the field intensity starts to concentrate into fewer and fewer regions. This can be seen when in the bottom right panel ($\mathcal{H} = 0$, $\mathcal{S} = 2$) where the structure is mostly within a few distinct regions. It is worth noting that for this particular example the fields are still periodic. Although $\mathcal{H}$ and $\mathcal{S}$ differ between each image, we can still see a hint of this periodicity. These plots have $N_{\text{PIX}} = 10^3$.

Equation 2.11 also has the effect of transforming the field from one with a Gaussian distribution into one with a roughly lognormal distribution. This is shown in Figure 2.10. The distributions here are those of the fields shown in Figure 2.9. For each distribution we fit a lognormal curve which is shown in red. This fit is done using the `scipy.stats.lognorm` function in the `SciPy` package (Virtanen et al., 2020). This lognormal approximates the peaks of the distributions very well, especially for low values of $\mathcal{S}$. For higher values of $\mathcal{S}$ we see that the tail of the distribution is lower than the tail of the fit. This is seen especially in panels *(c)*, *(e)*, and *(f)*. In general we find that the distributions are most similar to a lognormal for low values of $\mathcal{H}$ and $\mathcal{S}$. The mean and standard deviation of each lognormal fit is given by μ and σ, respectively. We can see that σ agrees very well with $\mathcal{S}$. This is due to the fact that $\mathcal{S}$ can be thought of as the standard deviation of the field.

Figure 2.11 shows the power spectra of these fields, while Figure 2.12 shows a cross-section of these power spectra taken at $k_1 = 0$, which we denote as $\hat{G}(k_2, k_1 = 0)$. $\hat{G}(\mathbf{k})$ is the amplitude of the power spectrum $\hat{g}(\mathbf{k})$, i.e. $\hat{G}(\mathbf{k}) = ||\hat{g}(\mathbf{k})||$. The location of the cross-section is shown by the red line in Figure 2.11. We can see that these power spectra are noisier then the ones for non-exponentiated fBm (Figure 2.8). This level of noise increases as we increase $\mathcal{S}$, and there is a higher level of noise at high k. In spite of this noise we are still able to estimate $\mathcal{H}$ very well by fitting a straight line to the curve (shown by the dashed red line in Figure 2.12), estimating the power law exponent β, and thus obtaining an estimate of $\mathcal{H}$ (which we denote as $\mathcal{H}_{EST}$) by using Equation 2.3. The value of $\mathcal{H}$ used to generate the fields is denoted by $\mathcal{H}_{TRUE}$. The value of $\mathcal{H}$ is estimated very well here with this estimation being slightly worse in panel (i), which underestimates $\mathcal{H}$ by 0.2. The field in this panel has $\mathcal{H}_{TRUE} = 0$ and $\mathcal{S} = 2.0$.

2.4.1 Noisy xfBm fields

We also investigate the effect of noise on xfBm fields. We generate a white noise field, $w(\mathbf{r})$ with $\mathbf{r} = (r_1, r_2)$ where $1 \leq r_i \leq N_{\text{PIX}}$. Each pixel in $w(\mathbf{r})$ is generated from a Gaussian

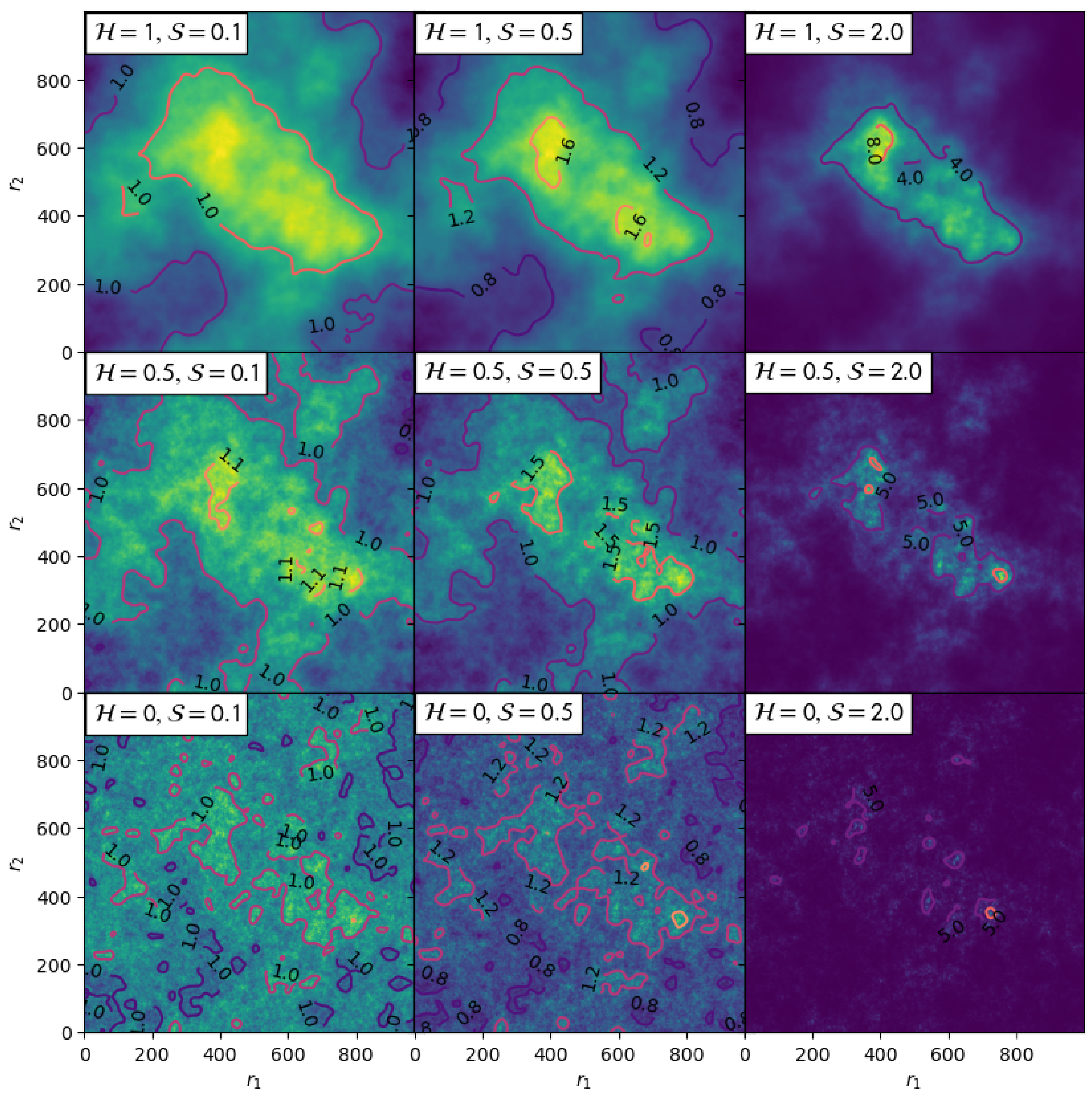


Fig. 2.9 Nine 2D xfBm fields that show how the structure varies with $\mathcal{H}$ and $\mathcal{S}$. $\mathcal{H}$ increases from the bottom to top while $\mathcal{S}$ increases from left to right. There are also contours that show lines of constant intensity. All nine plots have been plotted using the same random seed to preserve the general shape and all nine have $N_{\text{PIX}} = 10^3$. Each field has also been normalised such that the standard deviation is unity. These fields are all periodic and each one has had its centre of mass shifted to the centre of the image.

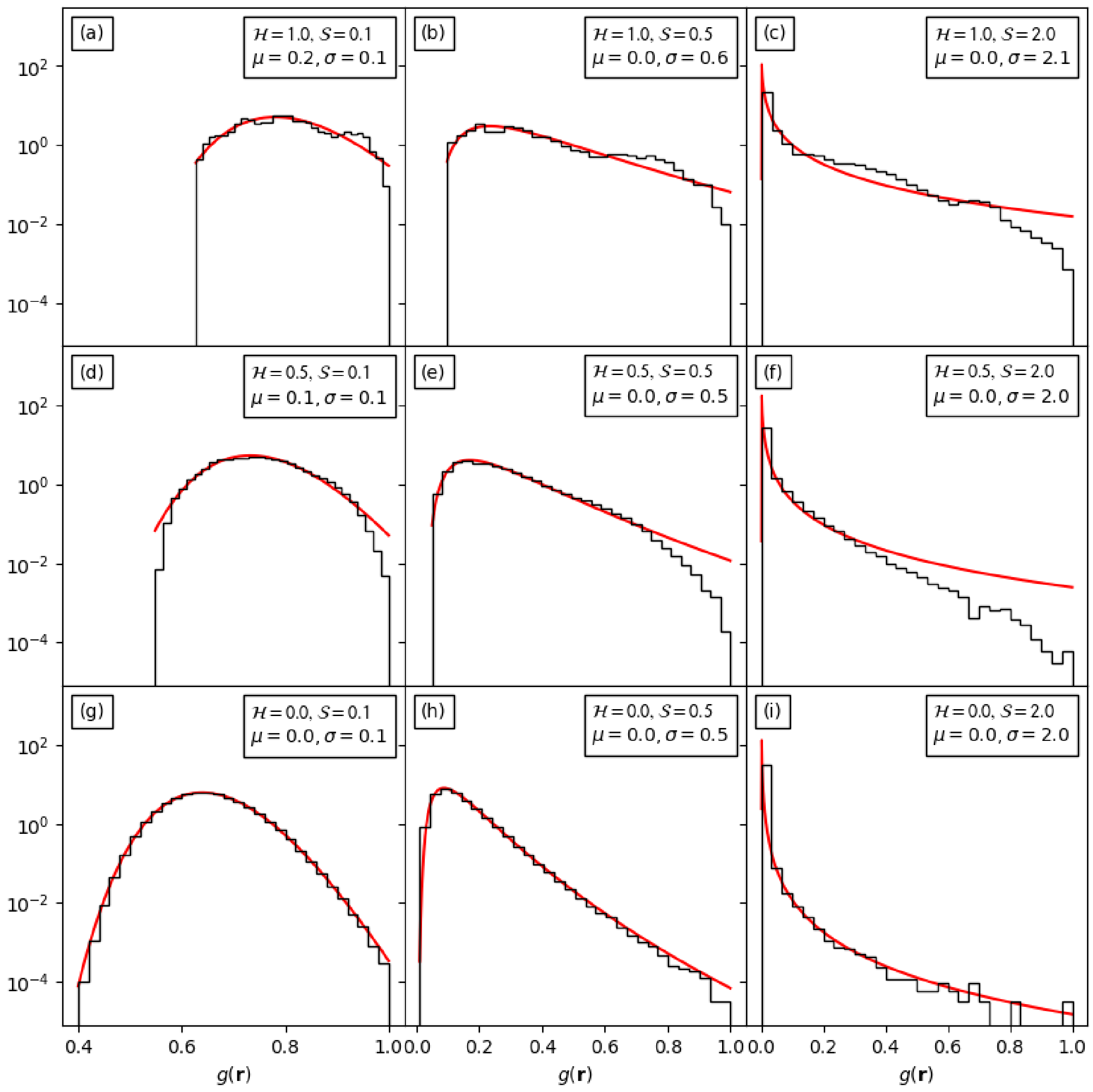


Fig. 2.10 The distributions of the fields shown in Figure 2.9. We have fitted a lognormal to each of these distributions, which is given by the red curves. The mean and standard deviation of the fitted lognormals is given by μ and σ.

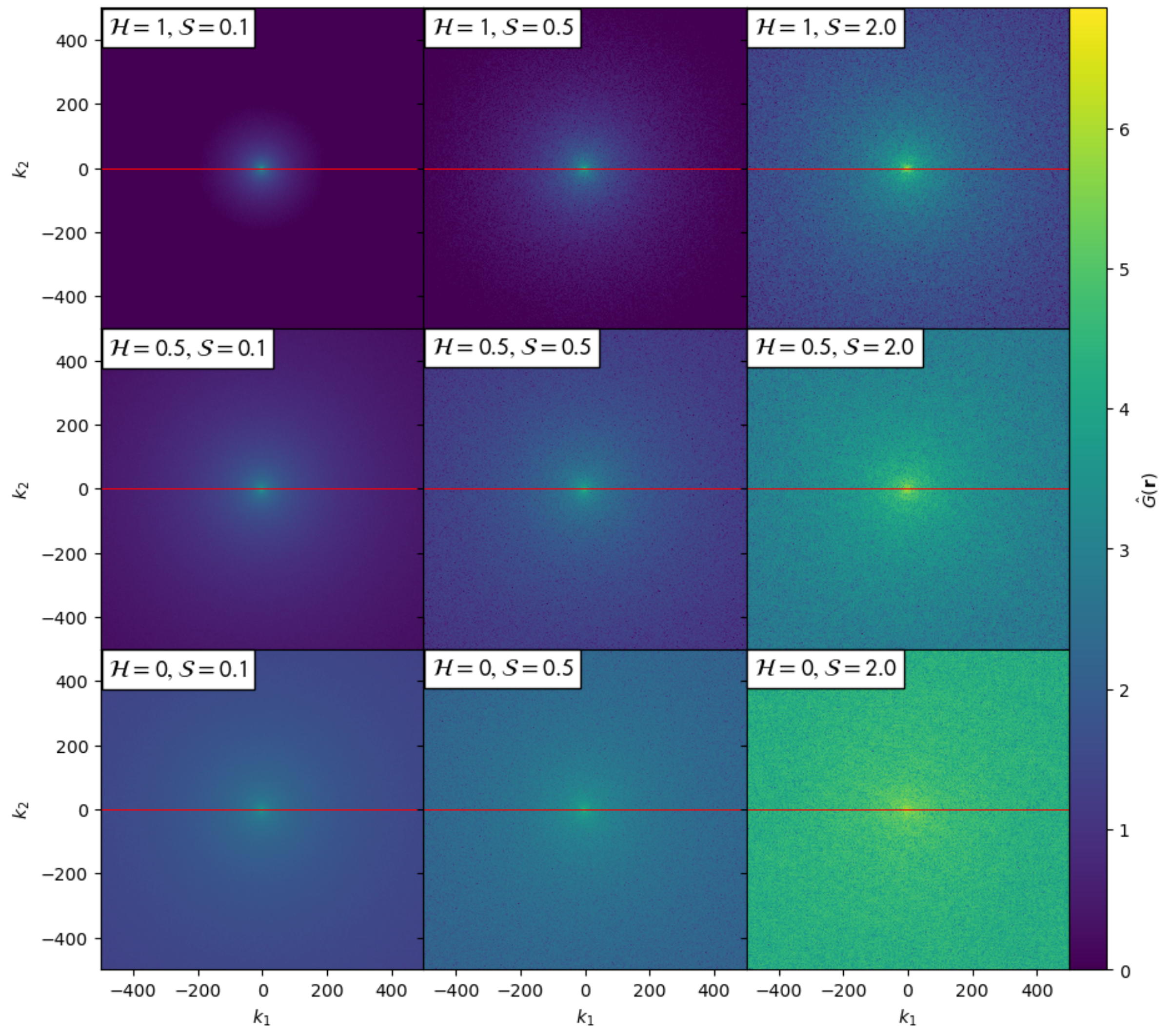


Fig. 2.11 The amplitudes, $\hat{G}(\mathbf{k})$, of the power spectra, $\hat{g}(\mathbf{k})$, of $g(\mathbf{r})$ shown for different values of $\mathcal{H}$ and $\mathcal{S}$. The red line at $k_2 = 0$ marks the cross-section that is taken to generate the plots in Figure 2.12.

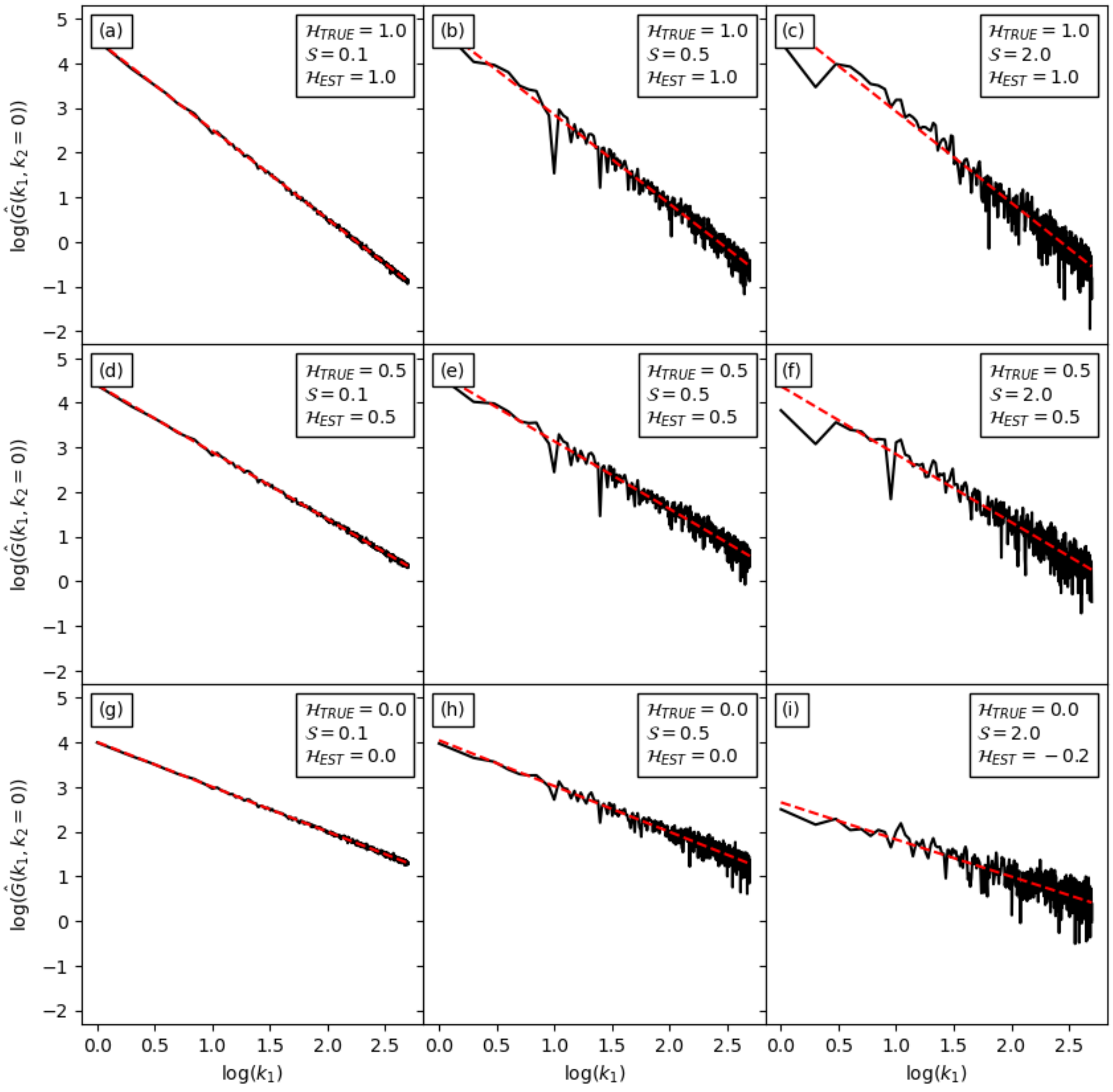


Fig. 2.12 The value of $\hat{G}(k_1, k_2)$ when $k_2 = 0$ for the power spectra shown in Figure 2.11. The black line is $\hat{G}(k_1, k_2 = 0)$ measured from $g(\mathbf{r})$ while the red dashed line is the linear fit of $\hat{G}(k_1, k_2 = 0)$. The slope gives an estimate of $\mathcal{H}$, denoted as $\mathcal{H}_{EST}$.

distribution with mean of 0 and standard deviation of $\eta\sigma_g$, where σ_g is the standard deviation of $g(\mathbf{r})$, and η is a parameter that controls how much noise is added to the field. In this section we set $\eta = 0.05$. We then get the noisy field, $g'(\mathbf{r})$, according to $g'(\mathbf{r}) = g(\mathbf{r}) + w(\mathbf{r})$.

Figure 2.13, shows the cross-sectional amplitude of the power spectrum $\hat{G}'(k_1, k_2 = 0)$ for different $\mathcal{H}$ and $\mathcal{S}$. The solid black curves show the measured amplitude while the red dashed line shows the linear fit. We can see that when $\mathcal{H} = 1$ and $\log(k_1) > 2$ the amplitude flattens off. This is due to the contribution of the noise to the power spectrum. Since high values of $\mathcal{H}$ have less high frequency modes than lower values of $\mathcal{H}$, the high frequency portion of the power spectrum for high $\mathcal{H}$ is dominated by that of the white noise, which has a gradient of 0. Thus in order to estimate $\mathcal{H}$ the linear fit must be done for the portion of this graph that does not have a gradient of 0.

Table 2.1 shows the true and estimated values of $\mathcal{H}$ and $\mathcal{S}$ for the fields shown in Figure 2.13. $\mathcal{H}$ is estimated from the slope of the power spectrum while, $\mathcal{S}$ is estimated by taking the standard deviation of the $\log g'(\mathbf{r})$. We can see that $\mathcal{S}$ is estimated very well when $\mathcal{S}_{TRUE} = 0.1$ and $\mathcal{S}_{TRUE} = 0.5$ for all cases of $\mathcal{H}$, however when $\mathcal{S}_{TRUE} = 2$ taking the standard deviation underestimates $\mathcal{S}$. $\mathcal{H}$ tends to be underestimated in all cases except for a few of the fields, while one of the estimations is larger than the true value.

Table 2.1 Table showing the true and estimated values of each of the xfBm fields from Figure 2.13.

$\mathcal{H}_{TRUE}$	$\mathcal{H}_{EST}$	$\mathcal{S}_{TRUE}$	$\mathcal{S}_{EST}$
1.00	0.98	0.10	0.10
1.00	0.99	0.50	0.50
1.00	1.06	2.00	1.67
0.50	0.50	0.10	0.10
0.50	0.48	0.50	0.50
0.50	0.40	2.00	1.60
0.00	0.00	0.10	0.10
0.00	-0.01	0.50	0.50
0.00	-0.20	2.00	1.46

2.4.2 Summary: Noisy and non-periodic xfBm fields

We construct noisy and non-periodic xfBm fields by combining the field-generating processes from above. First, the non-periodic xfBm field is generated. This is done by generating an xfBm field of size $4N_{\text{PIX}} \times 4N_{\text{PIX}}$ and then slicing the middle $N_{\text{PIX}} \times N_{\text{PIX}}$. As before, we set

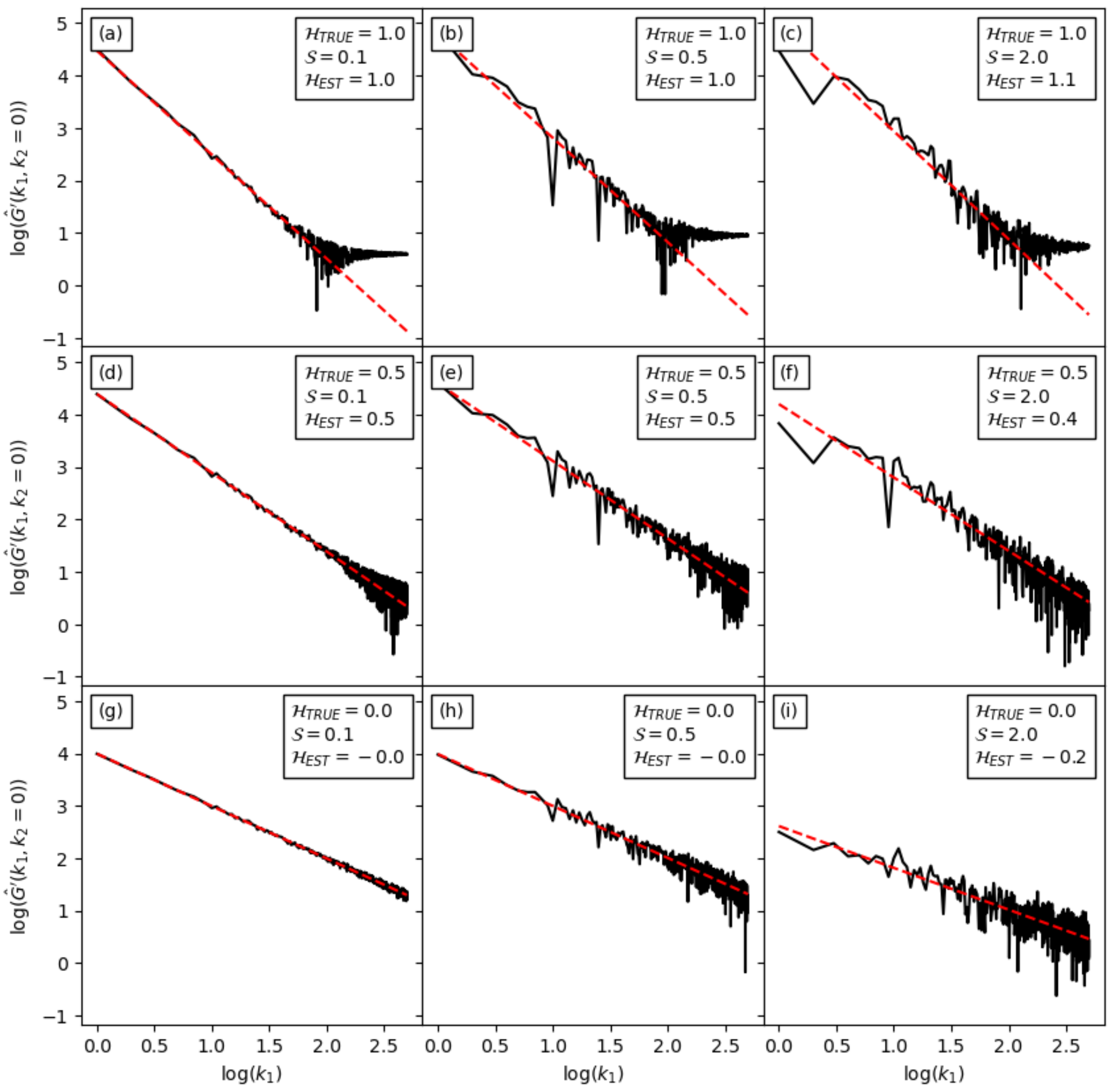


Fig. 2.13 Plotting of $\hat{G}'(k_1, k_2 = 0)$ against k_1. The black line shows the data while the red dashed line shows the linear fit. For the top row (panels (a), (b), and (c)), there are two linear portions of the plot. When $\log(k_1) > 2$ we see that the plot flattens out. This is due to the contribution of white noise to the power spectrum, which is lending more power to the high frequencies. Thus the linear fit was only made for $\log(k_1) < 2$.

$N_{\text{PIX}} = 1000$ and generate nine 2D ($\mathcal{E} = 2$) fields each with a different combination of $\mathcal{H} = [0, 0.5, 1], \mathcal{S} = [0.1, 0.5, 2]$ (See Section 2.4). We then add noise to each field using the process outlined in Section 2.4.1. This gives us a noisy, non-periodic field which we call $x(\mathbf{r})$. Taking the Fourier Transform, we get the power spectrum, $\hat{x}(\mathbf{k})$. We then calculate the amplitudes $\hat{X}(\mathbf{k}) = ||\hat{x}(\mathbf{k})||$ and set $k_2 = 0$. $\mathcal{H}$ is estimated according to

$$\hat{X}(k_1, k_2 = 0) = k_1^{-\frac{\mathcal{E}+2\mathcal{H}}{2}}. \tag{2.12}$$

$\mathcal{S}$ is estimated by taking the standard deviation of the field.

Table 2.2 shows the true, $TRUE$, and estimated, EST, values of $\mathcal{H}$ and $\mathcal{S}$ for each of the nine noisy, non-periodic xfBm outlined above. The performance of these estimations is the worst so far with both $\mathcal{H}$ and $\mathcal{S}$ being severely underestimated in all cases except for when $\mathcal{H} = 0$ and $\mathcal{S} = 0.1$.

Table 2.2 Table showing the true and estimated values of 9 noisy, non-periodic xfBm fields.

$\mathcal{H}_{TRUE}$	$\mathcal{H}_{EST}$	$\mathcal{S}_{TRUE}$	$\mathcal{S}_{EST}$
1.00	0.06	0.10	0.02
1.00	0.06	0.50	0.11
1.00	0.10	2.00	0.44
0.50	0.07	0.10	0.05
0.50	0.05	0.50	0.23
0.50	0.34	2.00	0.95
0.00	0.01	0.10	0.09
0.00	-0.11	0.50	0.45
0.00	-0.35	2.00	1.47

2.5 Next steps

So far we have only been dealing with fields that have $N_{\text{PIX}} = 1000$. The next chapter deals with the effects of N_{PIX} on the estimation of these metrics. In particular we shall look at the cases where $N_{\text{PIX}} = [128, 64, 32]$. We shall also introduce different techniques in particular Δ-variance (Stutzki et al., 1998; Ossenkopf et al., 2008a; Elmegreen, 2002) and Machine Learning.

CHAPTER 3

Estimating $\mathcal{H}$ and $\mathcal{S}$ using Established Techniques

3.1 Preamble

In the previous Chapter (Chapter 2) we saw how the power spectrum of fBm and xfBm fields vary with different parameters ($\mathcal{H}$ and $\mathcal{S}$) and constraints (noise and periodicity). The aim in that chapter was to investigate whether it was possible to recover the original values of $\mathcal{H}$ and $\mathcal{S}$ that were used to generate the fields. This was done on a singular basis, using one field at a time, however in this Chapter we aim to produce large scale statistics by estimating $\mathcal{H}$ and $\mathcal{S}$ for many fields rather than a handful. We investigate two different techniques that have been designed to estimate $\mathcal{H}$, which are; the Perimeter-Area method (this was briefly touched upon in Section 2.2), and the Delta-Variance method (Stutzki et al., 1998). We also investigate the estimation of $\mathcal{S}$. The next chapter (Chapter 4) is dedicated to developing our own method using Machine Learning. We will now primarily focus on 2-dimensional ($\mathcal{E} = 2$) fields that have $N_{\text{PIX}} = [128, 64, 32]$, except where stated otherwise. The pixel sizes chosen here are arbitrary and a machine learning model can be trained on any pixel size (with the caveat that a smaller N_{PIX} can give poorer results).

3.2 Estimating $\mathcal{S}$

Lomax et al. (2018) state that the exponentiating factor, $\mathcal{S}$, is simply the standard deviation of $\log g(\mathbf{r})$, where $g(\mathbf{r})$ is the xfBm field and $\mathbf{r} = (r_0, r_1)$ is the extent of the field and takes values $0 \leq r_i \leq N_{\text{PIX}}$. This is, by definition, a result of performing the exponentiation according to Equation 2.11.

The montage in Figure 3.1 shows the effect of the various processes outlined in Chapter 2 on this estimation of $\mathcal{S}$. Each point on these plots represents a single xfBm field. The value of $\mathcal{S}$

that is used to generate each field is denoted as $\mathcal{S}_{TRUE}$, while the estimation of $\mathcal{S}$ is denoted as $\mathcal{S}_{EST:N_{\text{PIX}}}$. The top row shows fields with $N_{\text{PIX}} = 128$, the middle row shows fields with $N_{\text{PIX}} = 64$, and the bottom row shows fields with $N_{\text{PIX}} = 32$. The different columns show, from left to right:

- Pure xfBm fields, i.e. fBm fields that have then been exponentiated according to Equation 2.11. These fields are periodic.
- xfBm fields that are non-periodic. This is done by generating an xfBm field of size $4N_{\text{PIX}} \times 4N_{\text{PIX}}$ and slicing out the middle $N_{\text{PIX}} \times N_{\text{PIX}}$.
- xfBm fields that have Gaussian noise added to them. This is done using the process described in Section 2.4.1 where each pixel has a variate $\mathcal{G}$ added to it. $\mathcal{G}$ is a random variate drawn from a Gaussian distribution with mean of 0 and a standard deviation of $0.05\sigma_{RMS}$, σ_{RMS} being the standard deviation of the field. These fields are periodic.
- xfBm fields that are both noisy and non-periodic. In this process the field is first sliced (to make it non-periodic), then the noise is applied.

Each field in Figure 3.1 has a randomly generated value of $\mathcal{H}$ and $\mathcal{S}$, sampled from a uniform distribution. $\mathcal{S}$ was sampled from the interval $[0, 3]$ while $\mathcal{H}$ was sampled from the interval $[0, 1]$. The value of $\mathcal{H}$ for each field is shown via the colour-bar. We also include the root-mean-square (RMS) error, ε, for each plot. The black line represents a perfect estimation where $\mathcal{S}_{EST} = \mathcal{S}_{TRUE}$.

The first thing to notice is that there isn't a large amount of difference in the accuracy of this estimation for fields that have different values of N_{PIX}. The error for pure xfBm is non-existent, while the error within the other vertical groupings differs by at most 0.02. This is a negligible error difference.

We can see that for pure xfBm fields this technique recovers $\mathcal{S}$ perfectly with 0 error. This is due to the definition of $\mathcal{S}$ in this case. The worst performing grouping is the one with noisy, non-periodic fields, with the periodicity (or lack thereof) playing a significant role in this error. In fact if we were to add up these errors according to the standard rule $(\varepsilon_{TOTAL})^2 = (\varepsilon_1)^2 + (\varepsilon_2)^2$ we would find that the error of the noisy and periodic panels are stochastic, with independent contributions from the noisy and the non-periodic parts of the xfBm generation process.

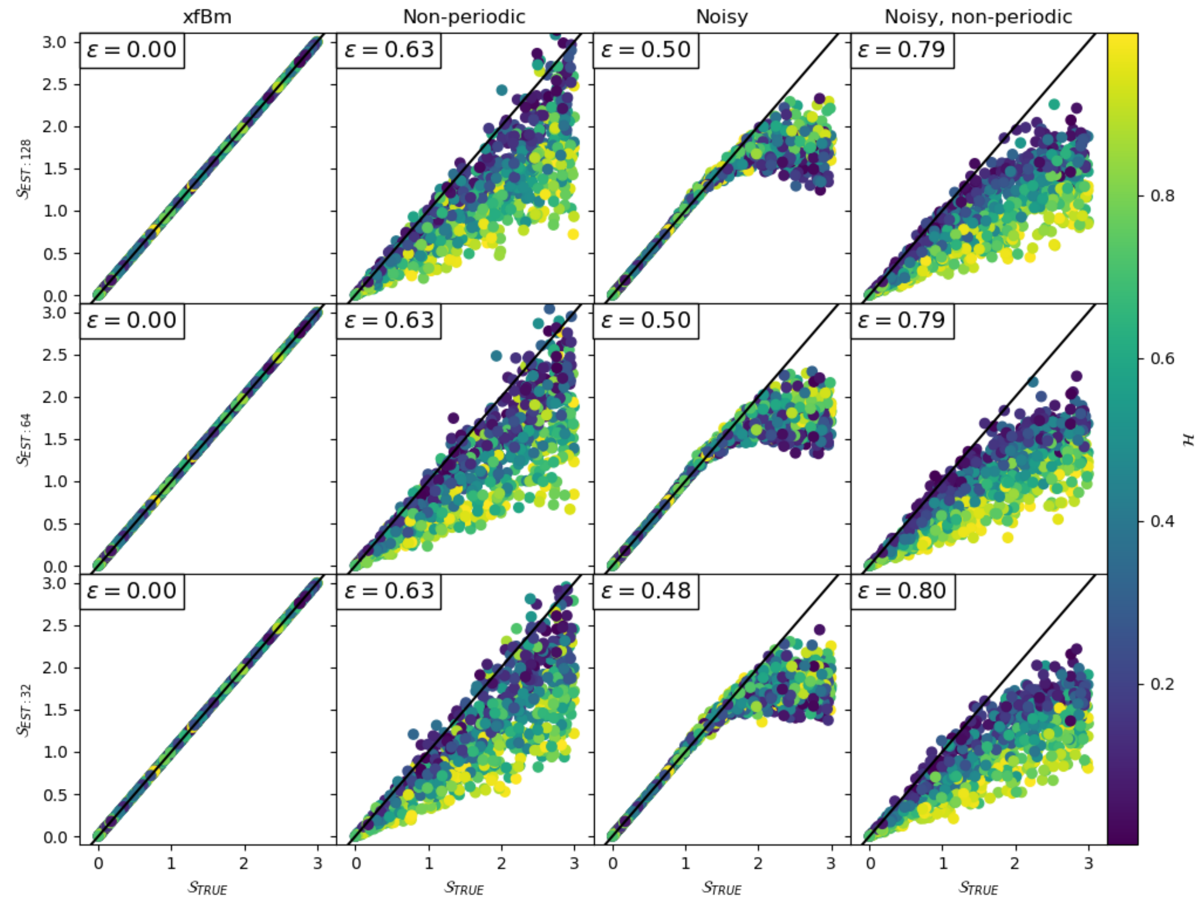


Fig. 3.1 Accuracy plots for different xfBm fields for estimating the value of $\mathcal{S}$. The x-axis shows the true value of $\mathcal{S}$ which we denote as $\mathcal{S}_{TRUE}$, while the y-axis shows the estimated value of $\mathcal{S}$ which we denote as $\mathcal{S}_{EST}$. The top row shows fields with $N_{\text{PIX}} = 128$, the middle row shows fields with $N_{\text{PIX}} = 64$, and the bottom row shows field with $N_{\text{PIX}} = 32$. The different columns correspond to different versions of the xfBm fields. From left to right they are; *(i):* Pure xfBm (noiseless and periodic), *(ii):* Non-periodic xfBm, *(iii):* Noisy xfBm, *(iv):* Noisy and non-periodic xfBm. The black line at at $\mathcal{S}_{EST} = \mathcal{S}_{TRUE}$ represents a perfect estimator. The RMS error is shown as ε. Each of the plots represents 1000 fields each with random $\mathcal{H}$ and $\mathcal{S}$. The value of $\mathcal{H}$ is represented by the colourbar.

When the field is non-periodic only, we find that $\mathcal{S}$ is increasingly underestimated as the values of both $\mathcal{S}$ and $\mathcal{H}$ increase. When the field is noisy only, this estimation works extremely well until $\mathcal{S}_{TRUE} \sim 1.5$ where $\mathcal{S}$ starts being severely underestimated. This is due to the fact that as $\mathcal{S}$ increases, the intensity of the field becomes more concentrated in fewer and fewer regions, leaving the rest of the field with less intensity, thus when a fairly flat noise field is added to the image, these low intensity regions have a lower signal to noise than the high intensity. At a particular threshold of $\mathcal{S}$, these low signal-to-noise regions will make up a larger portion of the field and thus their contribution to the measurement of the standard deviation dominates. The right-most column shows a combination of these two effects, with a consistent underestimation of $\mathcal{S}$ up until the threshold, $\mathcal{S}_{TRUE} > 1.5$, at which point the signal-to-noise dominates, resulting in serious underestimation of $\mathcal{S}$.

3.3 Estimation of $\mathcal{H}$ using the Perimeter-Area method

The Perimeter-Area method was one of the earliest methods used to determine the fractal dimension, $\mathcal{D}$, of a structure (Mandelbrot and Cannon, 1984; Peitgen and Saupe, 1988), and has since been popularly used to describe astronomical structures (Beech, 1987; Bazell and Desert, 1988; Falgarone et al., 1991; Hetem and Lepine, 1993; Vogelaar and Wakker, 1994; Sanchez et al., 2005; Rathborne et al., 2015; Marchuk et al., 2021; Petkova et al., 2023).

This method involves constructing a number, N_{CONT}, of iso-density contours and then measuring the perimeter, P, and area, A, of each contour. $\mathcal{D}$ is then estimated according to Equation 2.1. We can then use Equation 2.4 to estimate $\mathcal{H}$. Vogelaar and Wakker (1994) study the limitations of this method using fBm fields and suggest a number of caveats to the process. One suggestion by Vogelaar and Wakker (1994) is to only use contours that are $\geq 3\sigma_{RMS}$ where σ_{RMS} is the standard deviation of the field. This is because contour placement is quite sensitive to noise and using only $\geq 3\sigma_{RMS}$ contours is a way to eliminate low signal-to-noise pixels. They also suggest to only include the contribution to each A, of that of contour 'islands' that have $A_i > 25$ pixels. This is generally not a problem if the images that we are dealing with have large N_{PIX}, such as $N_{\text{PIX}} = 1000$.

Figure 3.2 shows an example of the Perimeter-Area method on a single fBm image, with $N_{\text{PIX}} = 1000$ and $\mathcal{H} = 0.5$. The left image shows the contours used, while the image on the right shows a plot of $\log_{10}(P)$ against $\log_{10}(A)$. The red line shows the linear fit through the

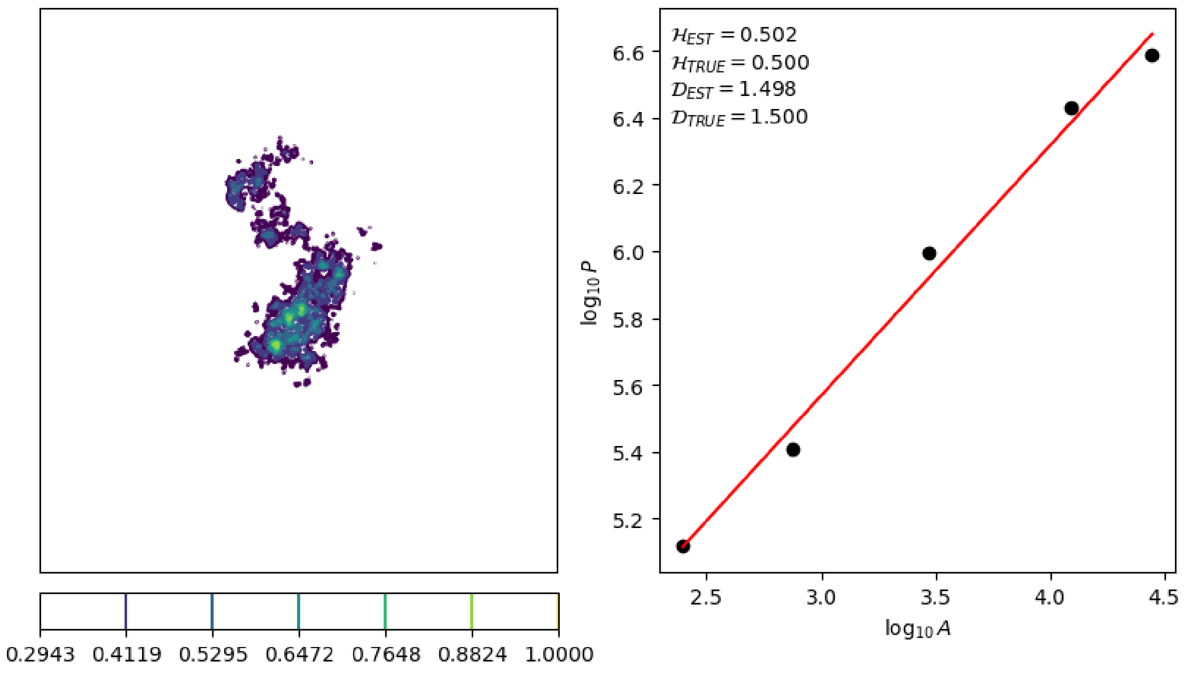


Fig. 3.2 An example of how the Perimeter-Area method of determining the fractal dimension. Our test image is a 2-dimensional ($\mathcal{E} = 2$) fBm image with parameters, $\mathcal{H} = 0.5$, and $N_{\text{PIX}} = 1000$. The left hand side shows the iso-level contours of our test field. Plotting the perimeters and the areas of these contours produces the plot on the right. The red line shows the linear fit to these values which are plotted as black dots. The estimated and true values of $\mathcal{D}$ and $\mathcal{H}$ are shown. The estimation here has $N_{CONT} = 5$.

points. Following the limitations outlined in Vogelaar and Wakker (1994) we find that the estimated value of $\mathcal{H}$ is only 0.002 different from the true value. Vogelaar and Wakker (1994) mention that this model can estimate $\mathcal{D}$ to within 0.05 for $1.2 \leq \mathcal{D} \leq 1.6$ and this is the case here. The number of contours used here is $N_{\text{CONT}} = 5$. It is quite obvious to see that much of the image is not used.

These limitations present a problem when dealing with small fields, particularly the ones we are interested in (i.e. ones with $N_{\text{PIX}} = [128, 64, 32]$). In fact it is frequently the case that no such contours exist for these fields when taking these limitations into account.

Another issue is the fact that the perimeter-area method must only contain closed contours in order for it to work. Taking the lower limit contour $> 3\sigma_{RMS}$, as mentioned above, deals with this, however for small fields such as ours, this can quite frequently rule out most of the field. Another way of minimising the occurrence of open contours is to ensure the structure to be studied is centred in the image. However real-life structures can be quite extended, and in order to model such structures we must allow the generation of xfBm structures that extend beyond the boundaries of the box. For fields such as ours ($N_{\text{PIX}} \leq 128$) we do not produce meaningful results using Perimeter-Area. This is shown below.

Figure 3.3 shows the performance of Perimeter-Area when applied to different flavours of xfBm. Each panel represents 1000 fields with a random $\mathcal{H}$ chosen from the uniform interval $[0, 1]$ and a random $\mathcal{S}$ chosen from the uniform interval $[0, 3]$. From top to bottom we have fields with $N_{\text{PIX}} = 128$, $N_{\text{PIX}} = 64$, and $N_{\text{PIX}} = 32$, and from left to right we have: *(i):* pure xfBm (periodic and noiseless), *(ii):* non-periodic xfBm, *(iii):* noisy xfBm, and *(iv):* noisy and non-periodic xfBm. The black line at $\mathcal{H}_{EST} = \mathcal{H}_{TRUE}$ shows a perfect estimator and the colourbar represents $\mathcal{S}$. For such small fields no contours would exist if we take any of the precautions mentioned in Vogelaar and Wakker (1994) and thus these estimations were made using all available contours. As predicted by Vogelaar and Wakker (1994) this estimation is fruitless and no accurate measurement of $\mathcal{H}$ can be found this way. This is mostly due to the fact that. for such small contour areas, there is substantial amount of noise due to the placement of the individual contours. Thus we do not use Perimeter-Area for our estimations.

3.4 Estimation of $\mathcal{H}$ using the Delta Variance method

Stutzki et al. (1998) show a more reliable way of estimating, $\mathcal{D}$ (and thus $\mathcal{H}$) by introducing the Delta-Variance method. Delta-variance is a generalisation of the Allan Variance (Allan, 1966) and provides a measure of the amount of structure that exists on different length-scales. The Delta-Variance, $\sigma^2_\Delta(L)$, of a 2-dimensional field, $g(\mathbf{r}')$ is the variance of the field after it is convolved with a circular filter function, $\odot_L$, of length-scale L,

$$\sigma^2_\Delta(L) = \frac{1}{2\pi}\left\langle (g \circledast \odot_L)^2 \right\rangle_{\mathbf{r}'}, \tag{3.1}$$

where $\circledast$ denotes the convolution operation. Here, we perform a coordinate transformation such that $\mathbf{r} \longrightarrow \mathbf{r}' = (r'_0, r'_1)$ where $-1/\sqrt{2} \leq r'_i \leq 1/\sqrt{2}$, in contrast to $0 \leq r_i \leq 1$. This also ensures that $|\mathbf{r}'| \leq 1$ under the condition that the bounding box is perfectly square.

The σ^2_Δ measures the amount of substructure that exists at the scale of L, and has the relationship, $\sigma^2_\Delta \propto L^{\beta-\mathcal{E}}$. Thus by substituting β for $\mathcal{H}$ and taking the derivative w.r.t $\log(L)$ we get,

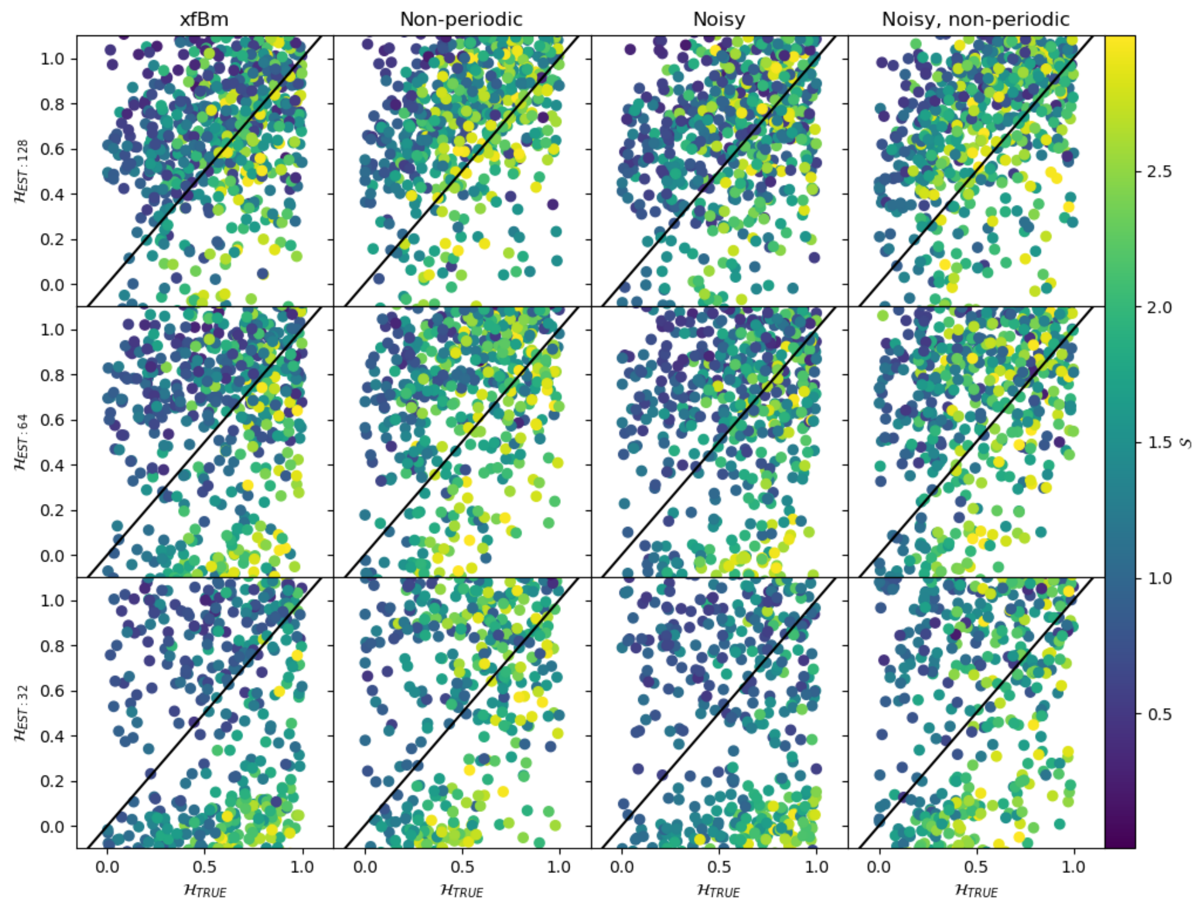


Fig. 3.3 Accuracy plots for the estimation of $\mathcal{H}$ using the perimeter-area method. Each of the points represents an xfBm field. The black line at $\mathcal{H}_{EST} = \mathcal{H}_{TRUE}$ represents a perfect estimator. The top row has fields with $N_{\text{PIX}} = 128$, the middle row has fields with $N_{\text{PIX}} = 64$, and the bottom row has fields with $N_{\text{PIX}} = 32$. The different columns correspond to different versions of the xfBm fields. From left to right they are; *(i):* Pure xfBm (noiseless and periodic), *(ii):* Non-periodic xfBm, *(iii):* Noisy xfBm, *(iv):* Noisy and non-periodic xfBm. Each of the plots represents 1000 fields each with random $\mathcal{H}$ and $\mathcal{S}$. The value of $\mathcal{S}$ is represented by the colourbar.

$$\mathcal{H} = \frac{1}{2} \cdot \frac{d \log(\sigma_\Delta^2)}{d \log(L)}. \tag{3.2}$$

Plotting $\log(\sigma_\Delta^2)$ against $\log(L)$ and taking the gradient we can therefore estimate $\mathcal{H}$.

There is discussion to be made on the choice of the circular filter, $\odot_L$. Stutzki et al. (1998) originally use a French hat filter function,

$$\odot_l = \frac{4}{\pi L^2} \begin{cases} 1 & \text{if } |\mathbf{r}'| \leq L/2 \\ -1/8 & \text{if } L/2 < |\mathbf{r}'| \leq 3L/2 \\ 0 & \text{if } |\mathbf{r}'| > 3L/2. \end{cases} \tag{3.3}$$

Figure 3.4 shows what a French hat filter function looks for $L = [0.1, 0.5, 1]$. The top row shows the 2-dimensional filter function, while the bottom row shows a cross-section taken at $r_1' = 0$. $r_1' = 0$ is shown as a red line in the top row. This filter function consists of a positively valued core (shown in yellow) and a negatively valued annulus (shown in dark blue). At a particular L the convolution operation suppresses structures that are of a different length-scale than L and thus taking the variance of the leftover structures we have a measure for how much variation in the structure is at that particular length-scale.

Ossenkopf et al. (2008b) note that the discontinuities between the core, the annulus, and the background of the French Hat filter function causes inaccurate estimations for σ_Δ^2 at high frequencies. A remedy for this is to use a Mexican Hat filter function instead. The core and annulus of the Mexican Hat filter are two Gaussians that are combined in order to remove any discontinuity in the function. This is the filter that we use and it is given by,

$$\odot_L(\mathbf{r}') = \odot_{CORE,L}(\mathbf{r}') - \odot_{ANN,L}(\mathbf{r}') \tag{3.4}$$

$$\odot_{CORE,L}(\mathbf{r}') = \frac{4}{\pi L^2} \exp\left(\frac{-4r'^2}{L^2}\right) \tag{3.5}$$

$$\odot_{ANN,L}(\mathbf{r}') = \frac{4}{\pi(\nu^2 - 1)L^2} \left\{ \exp\left(\frac{-4r'^2}{\nu^2 L^2}\right) - \exp\left(\frac{-4r'^2}{L^2}\right) \right\}, \tag{3.6}$$

where ν determines the ratio between the width of the annulus and the core. Ossenkopf et al. (2008a) suggest using a value of $\nu = 1.5$, and this is what we do here.

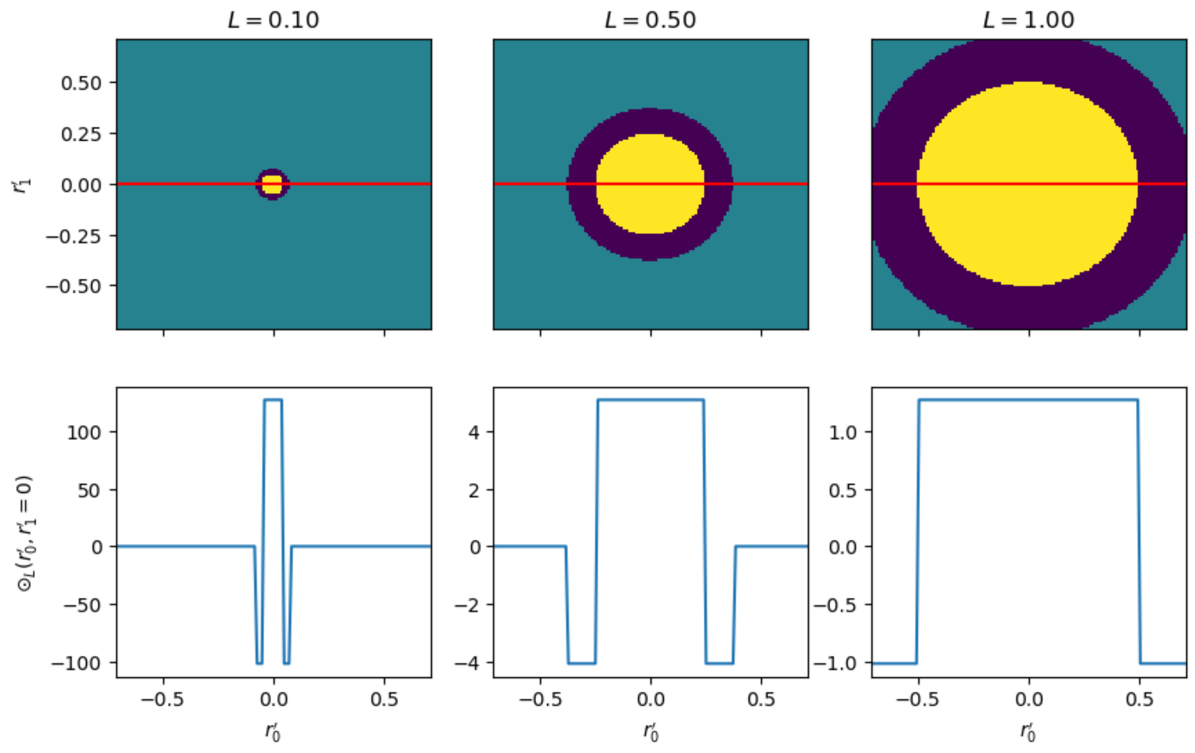


Fig. 3.4 Plots showing the French Hat filter for different L. The top row shows the 2D filter, while the bottom row shows a cross-section through the red line ($r_1' = 0$).

Figure 3.5 shows the Mexican Hat filter for $L = [0.1, 0.5, 1]$. Similar to Figure 3.4, the top row shows the 2D filter while the bottom row shows a cross section at $r_1' = 0$ (shown by the red line). This is clearly a continuous function and as a result $\sigma_\Delta^2(L)$ will not be contaminated by the high frequency modes that are present in the Fourier space of the French Hat filter function.

Figure 3.6 shows the difference in the power spectrum between the two filters. The top row shows the French Hat filter function while the bottom row shows the Mexican hat filter functions. The French hat filter function produces a number of artefacts in the power spectrum while the Mexican Hat function is cleaner, and therefore gives a better result (Ossenkopf et al., 2008b).

3.4.1 Periodic Fields

If the field is known to be periodic then computing the Delta-variance is quite straightforward. Ossenkopf et al. (2008b) note that it is less computationally expensive to integrate the product of the power spectrum of the field and that of the circular filter over **k**,

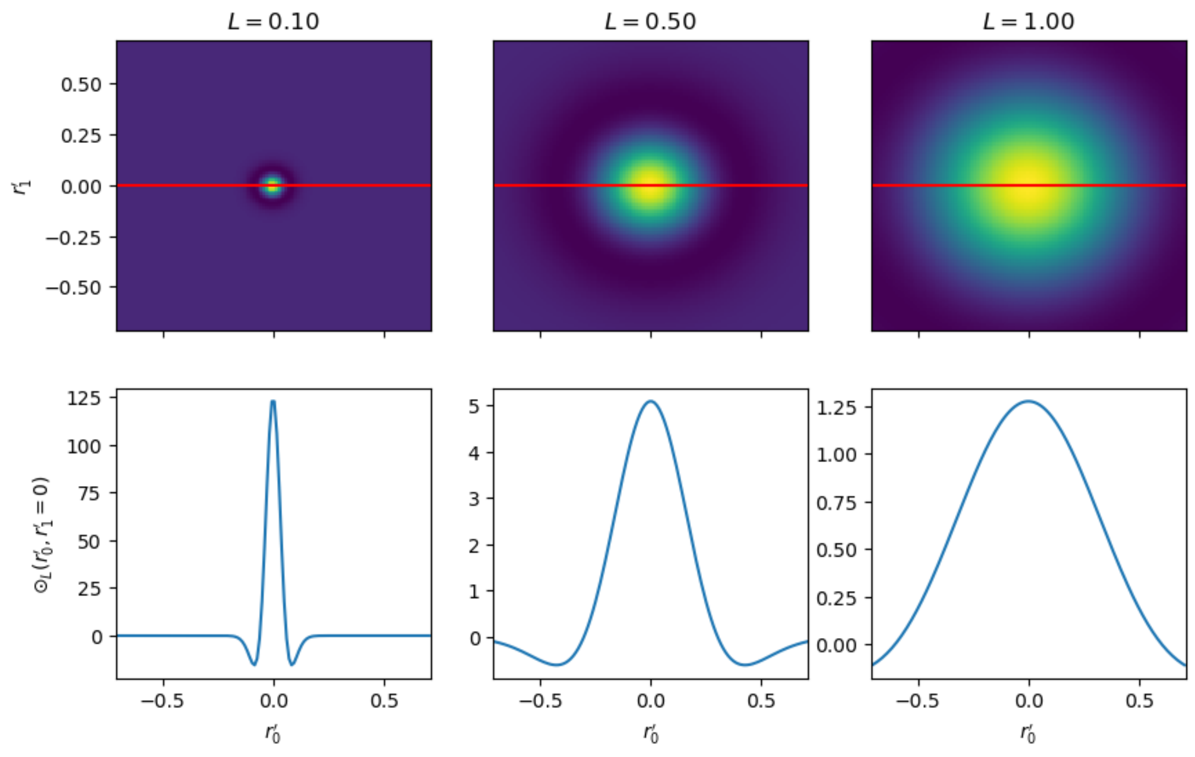


Fig. 3.5 Plots showing the Mexican Hat filter for different L. The top row shows the 2D filter, while the bottom row shows a cross-section through the red line ($r'_1 = 0$).

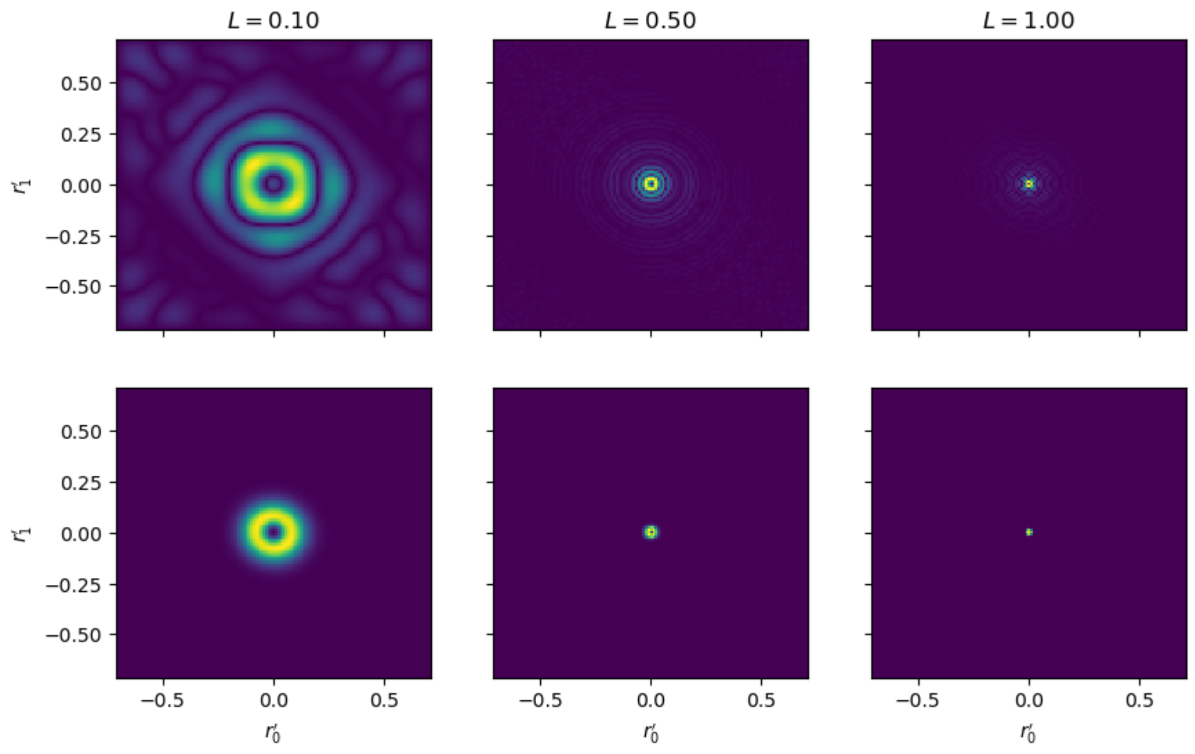


Fig. 3.6 A comparison between the power spectra of the French Hat filter function and the Mexican Hat filter function. The top row shows the power spectrum of the French hat filter function. The bottom row shows the power spectrum of the Mexican Hat filter function. The columns indicate different values of L.

$$\sigma^2_\Delta(L) = \frac{1}{2\pi}\int \hat{g}(\mathbf{k})|\hat{\odot}_L|^2 d^2\mathbf{k}, \tag{3.7}$$

where $\hat{g}(\mathbf{k})$ is the power spectrum of the field, $g(\mathbf{r}')$, $\hat{\odot}_L$ is the power spectrum of the filter function, $\odot_L(\mathbf{r}')$, and $\mathbf{k}$ is the wavenumber. L here denotes the relative size of the filter to the image and has values $0 \leq L \leq 1$.

3.4.2 Non-periodic fields

If the field is non-periodic then a more complex procedure is needed. This is in order to reduce any edge effects since Fourier Transforms assume a periodic field. The field is padded out to twice the size (i.e $2N_{\mathrm{PIX}} \times 2N_{\mathrm{PIX}}$). The padded values are set to zero. The convolution operation is then performed on just the pixels that make up the original image. If we do not zero-pad the image, the convolution operation would wrap around the edge of the field, and would consist of contributions from the opposite side of the field. This is more easily done using the following process:

1. First, the convolution is split into the following four parts,

$$F_{CORE,L}(\mathbf{r}') = f_{PAD}(\mathbf{r}') \circledast \odot_{CORE,L}(\mathbf{r}') \tag{3.8}$$

$$F_{ANN,L}(\mathbf{r}') = f_{PAD}(\mathbf{r}') \circledast \odot_{ANN,L}(\mathbf{r}') \tag{3.9}$$

$$W_{CORE,L}(\mathbf{r}') = w(\mathbf{r}') \circledast \odot_{CORE,L}(\mathbf{r}') \tag{3.10}$$

$$W_{ANN,L}(\mathbf{r}') = w(\mathbf{r}') \circledast \odot_{ANN,L}(\mathbf{r}'). \tag{3.11}$$

   Here, $f_{PAD}(\mathbf{r}')$ is the zero-padded field, while $w(\mathbf{r}')$ is a mask that takes values of one corresponding to the non-zero values of $f_{PAD}(\mathbf{r}')$ and then zero padded to $\mathbf{r}'$. Each of these convolutional operations is calculated by taking the product of their Fourier Transforms.

2. The four different convolutions are then combined according to,

$$F_L(\mathbf{r}') = \frac{F_{CORE,L}(\mathbf{r}')}{W_{CORE,L}(\mathbf{r}')} - \frac{F_{ANN,L}(\mathbf{r}')}{W_{ANN,L}(\mathbf{r}')} \tag{3.12}$$

3. The weight parts are combined to produce a final map of weights,

$$W_{TOT,L}(\mathbf{r}') = W_{CORE,L}(\mathbf{r}')W_{ANN,L}(\mathbf{r}'). \tag{3.13}$$

   This map of weights gives the central pixels of the field a higher weight than the ones closer to the edges. As a result the effect of the discontinuity at the edges of the field can be minimised.

4. Finally the Delta-variance is calculated according to,

$$\sigma^2_\Delta(L) = \frac{\Sigma\{(F_L(\mathbf{r}') - \langle F_L(\mathbf{r}')\rangle)^2 \cdot W_{TOT,L}(\mathbf{r}')\}}{\Sigma\{W_{TOT,L}(\mathbf{r}')\}} \tag{3.14}$$

Figure 3.7 shows a set of Delta-variance plots for fBm fields. These fields are pure (i.e. periodic and noiseless). Each of the Delta-Variance curves represents a single field with each of the plots showing five curves, one for each of $\mathcal{H} = [0.00, 0.25, 0.50, 0.75, 1.00]$. Each panel corresponds to different $N_{\text{PIX}} = [128, 64, 32]$. It is important that an appropriate L-range is chosen such that a good linear fit can be made. The L-range that is chosen is different for different N_{PIX} as a valid range of L will depend on the scale of the whole field. Therefore L must be chosen such that it is not too close to the total size of the field or to the resolution limit. These two limits produce end effects in the curve that are not linear. These end effects can be clearly seen in Figure 3.7. The red shaded region shows the range of L that was used to estimate the gradient and thus $\mathcal{H}$. For $N_{\text{PIX}} = 128$, L is chosen to have values $-1.5 \leq \log_{10} L \leq -0.5$, when $N_{\text{PIX}} = 64$ we choose $-1.25 \leq \log_{10} L \leq -0.5$, and when $N_{\text{PIX}} = 32$ we choose $-1 \leq \log_{10} L \leq -0.4$. We can see that from these plots that, for pure fBm fields, Delta-Variance recovers $\mathcal{H}$ quite well. The estimated and true values of $\mathcal{H}$ are shown in the inset of each panel.

Figure 3.8 shows Delta-variance curves for noisy and periodic xfBm fields with $N_{\text{PIX}} = 32$ with each colour representing a different $\mathcal{H} = [0.00, 0.25, 0.50, 0.75, 1.00]$. Panel *(a)* shows fields with $\mathcal{S} = 0.5$, panel *(b)* shows fields with $\mathcal{S} = 1$, and panel *(c)* shows fields with $\mathcal{S} = 3$. The inset of each panel shows the true and estimated values of $\mathcal{H}$. The coloured red area shows the range of L that is taken to estimate the slope and thus $\mathcal{H}$. For $\mathcal{S} = 0.5$ we choose $-1.25 \leq \log_{10} L \leq -0.75$, for $\mathcal{S} = 1$, we choose $-1.25 \leq \log_{10} L \leq -0.5$, and for $\mathcal{S} = 3$ we choose $-1 \leq \log_{10} L \leq -0.4$.

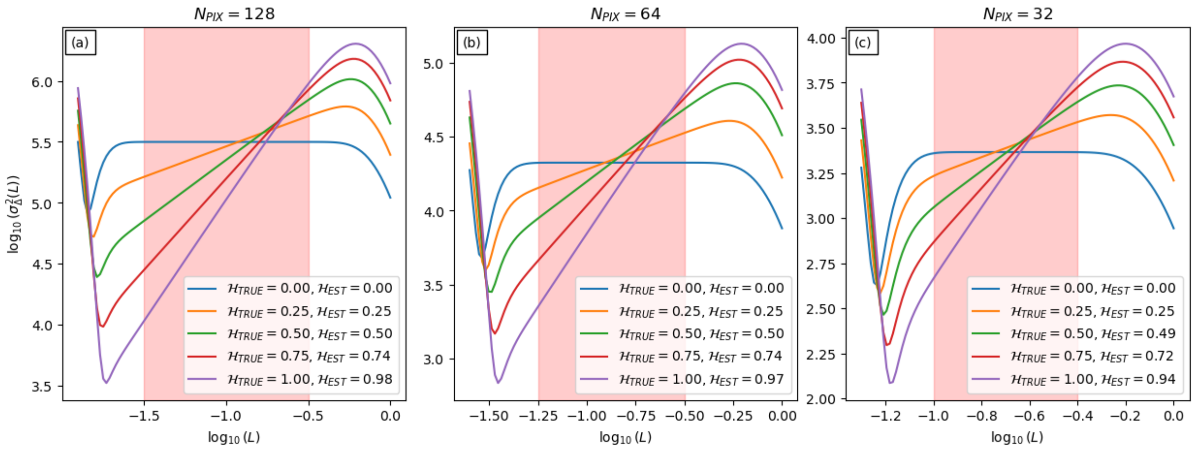


Fig. 3.7 Delta-variance curves for pure (periodic and noiseless) fBm fields. Each of the panels shows fBm fields which have $\mathcal{H} = [0.00, 0.25, 0.50, 0.75, 1.00]$ and are represented by different colours. Panel *(a)* shows fields with $N_{\text{PIX}} = 128$, panel *(b)* shows fields with $N_{\text{PIX}} = 64$, and panel *(c)* shows fields with $N_{\text{PIX}} = 32$. The red shaded region shows the range of L for which a linear fit was estimated and thus $\mathcal{H}$ estimated. The true and estimated values of $\mathcal{H}$ are shown in the inset.

We can already see from Figure 3.8 that the estimation of $\mathcal{H}$ for xfBm is not as good as for fBm (See Figure 3.7). This is due mostly to the fact that the slope is not well-defined for a substantial number of fields. This results in such fields having a Delta-variance curve where it is difficult to find a portion of the curve that is linear, and in some cases there are often more than one. A good example of this is the $\mathcal{S} = 3, \mathcal{H}_{TRUE} = 0.25$ curve in panel (c) of Figure 3.8 which is marked in orange. This curve clearly has two regions ($\log_{10}(L) \leq -0.8$ and $-0.5 \leq \log_{10}(L) \leq -0.4$) that are linear. For a number of fields that do have a well-defined slope we often find that the estimation tends to give incorrect, or even negative, values.

Figure 3.9 shows a large scale estimation of $\mathcal{H}$ using Delta-variance. Each of the panels contains 1000 xfBm fields that have been generated with a random $\mathcal{H}$, chosen from the interval $[0, 1]$, and a random $\mathcal{S}$, chosen from the interval $[0, 3]$. The top row indicates fields with $N_{\text{PIX}} = 128$, the middle row indicates fields with $N_{\text{PIX}} = 64$, while the bottom row indicates fields with $N_{\text{PIX}} = 32$. One interesting feature that we can see across all panels is that as $\mathcal{S}$ increases $\mathcal{H}$ becomes more and more underestimated. This points to a degeneracy between high values of $\mathcal{S}$ and low values of $\mathcal{H}$. In fact, if we have a field where $\mathcal{H}$ is low and $\mathcal{S}$ is also low, there is more small scale structure than large scale structure, and if we have a field where $\mathcal{H}$ is high and $\mathcal{S}$ is also high the structure is more concentrated in a few small regions. These two fields will both have more small scale structure than large scale structure, leading to

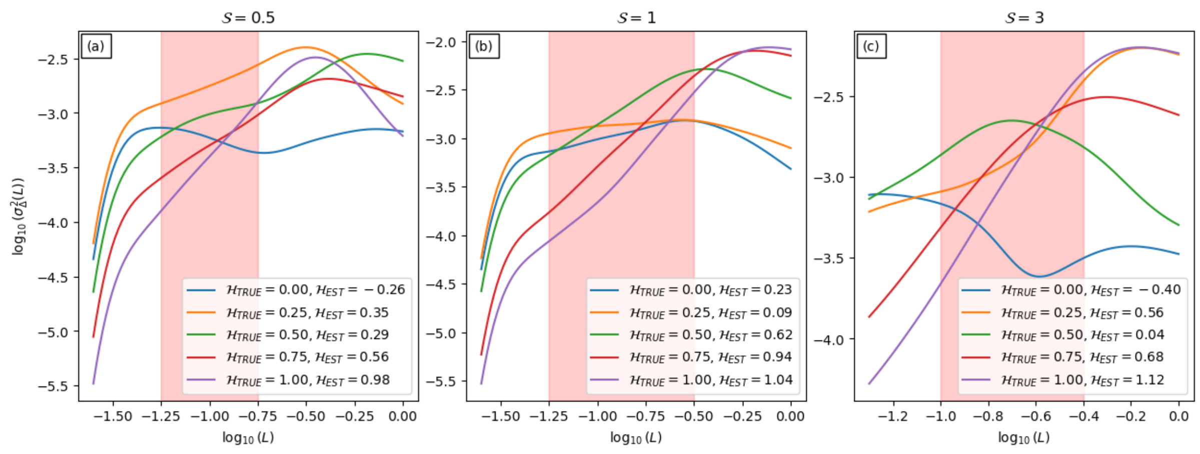


Fig. 3.8 Delta-variance curves for noisy and periodic xfBm fields. Each of the panels shows xfBm fields which have $\mathcal{H} = [0.00, 0.25, 0.50, 0.75, 1.00]$ and are represented by different colours. Panel *(a)* shows fields with $\mathcal{S} = 0.5$, panel *(b)* shows fields with $\mathcal{S} = 1$, and panel *(c)* shows fields with $\mathcal{S} = 3$. The red shaded region shows the range of L for which a linear fit was estimated and thus $\mathcal{H}$ estimated. The true and estimated values of $\mathcal{H}$ are shown in the inset. Each of the fields shown here have $N_{\text{PIX}} = 32$.

Delta-variance underestimating $\mathcal{H}$ in the latter case. The main visual difference between these two fields is that the former (low $\mathcal{H}$ and $\mathcal{S}$) is more extended than the latter (high $\mathcal{H}$ and $\mathcal{S}$). We refer the reader to Figure 2.9 for a visual comparison between two such fields.

The accuracy of these estimations can no doubt be improved with careful analysis of the Delta-variance curves. However this involves checking the curves visually and selecting the appropriate portion of the curve for which there is a well-defined slope. This human element adds a considerable amount of time and effort to this analysis and is thus infeasible in the case of producing maps of $\mathcal{H}$ and $\mathcal{S}$. Furthermore we find that Delta-variance tends to be quite computationally expensive. The estimations for Figure 3.9 took 6 hours to produce on a personal computer. Therefore it only makes sense to use this technique when analysis of a single region or structure is required. In fact this is mostly what Delta-variance has been used for, and in most studies a single value of $\mathcal{H}$, β, or $\mathcal{D}$ is found per region. In Stutzki et al. (1998) and Ossenkopf et al. (2008b) use it to examine the Polaris Flare determining $\beta = 2.77$ ($\mathcal{H} = 0.385$), while Elia et al. (2014) use it to investigate a number of whole Hi-GAL tiles (Molinari et al., 2010) producing single values of β for a handful of region subsets and tiles.

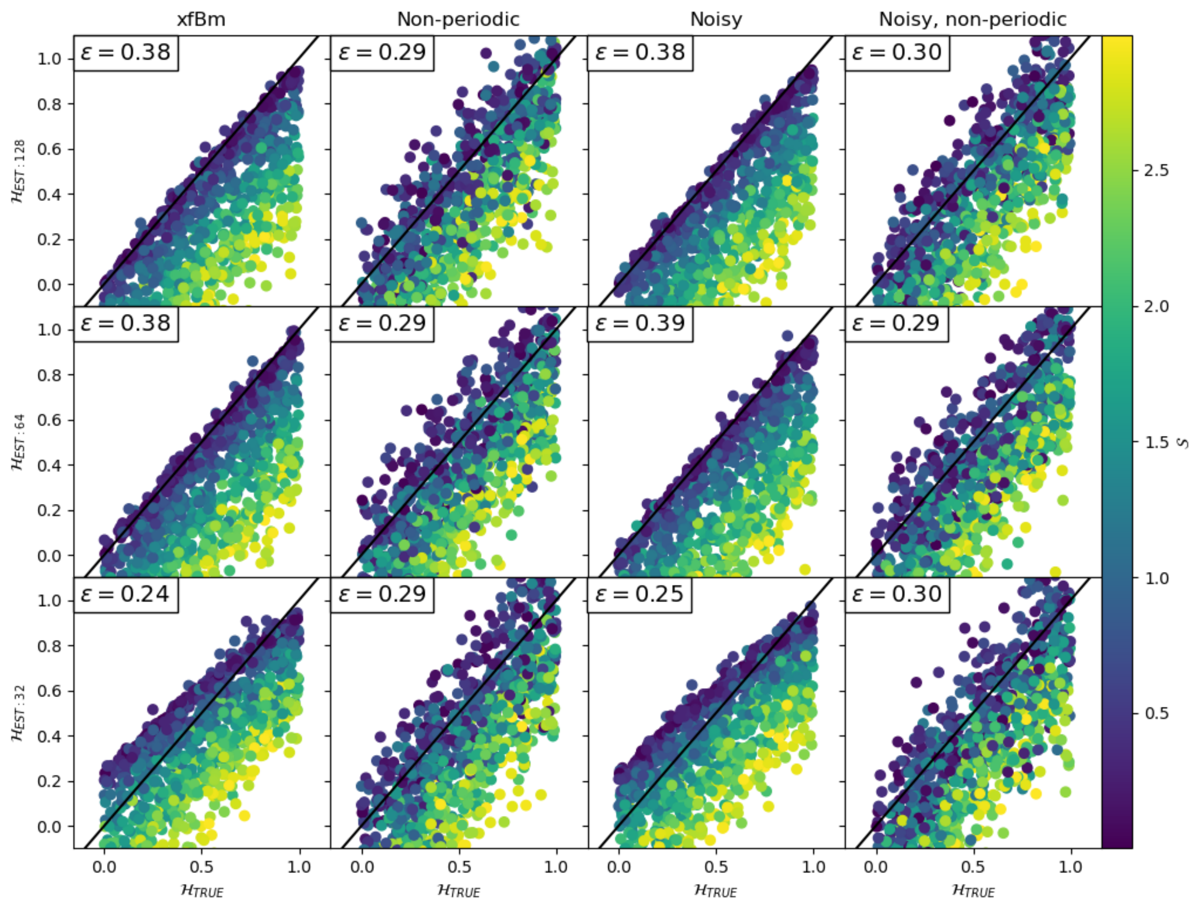


Fig. 3.9 Accuracy plots for the estimation of $\mathcal{H}$ using Delta-variance. Each of the points represents an xfBm field. The black line at $\mathcal{H}_{EST} = \mathcal{H}_{TRUE}$ represents a perfect estimator. The top row has fields with $N_{\text{PIX}} = 128$, the middle row has fields with $N_{\text{PIX}} = 64$, and the bottom row has fields with $N_{\text{PIX}} = 32$. The different columns correspond to different versions of the xfBm fields. From left to right they are; *(i):* Pure xfBm (noiseless and periodic), *(ii):* Non-periodic xfBm, *(iii):* Noisy xfBm, *(iv):* Noisy and non-periodic xfBm. Each of the plots represents 1000 fields each with random $\mathcal{H}$ and $\mathcal{S}$. The value of $\mathcal{S}$ is represented by the colourbar. The RMS error is given by ε which is shown in the corner of each panel.

3.5 Multi-Fractal Analysis

So far we have assumed that the structures are inherently monofractal, i.e the fractal dimension remains constant at all length-scales. However there is evidence to suggest that the ISM is multi-fractal (Chappell and Scalo, 2001; Elia et al., 2018). Multifractals are a generalisation of monofractals and describe structures where the fractal structure is self-similar but has different scaling behaviours at different scales (Mandelbrot and Cannon, 1984; Halsey et al., 1986; Chappell and Scalo, 2001; De La Fuente Marcos and De La Fuente Marcos, 2006). Multifractals are also needed if the structure we are observing is in reality a superposition of more than one fractal structure each with potentially differing fractal properties, such as those seen along the general vicinity of a particular line of sight. This would lead to differing values of fractal dimension at different locations.

By tiling the structure with boxes of size L we can measure the generalised box counting fractal dimension, $\mathcal{D}_q$ by,

$$\mathcal{D}_q = \lim_{L \to 0} \frac{1}{q-1} \cdot \frac{\log Z_q(L)}{\log L}, \tag{3.15}$$

where $Z_q(L)$ is a partition function and is given by,

$$Z_q(L) = \sum_i P_i^q. \tag{3.16}$$

Here P_i is the probability distribution of the pixel intensity in the ith box normalised over the entire field. This is simply the sum of the pixel values in that box divided by the sum of all the pixels in the field. The variable $q \in \Re$ is defined such that $q = 0$ recovers the Euclidean dimension (i.e. $\mathcal{D}_0 = \mathcal{E}$) (Elia et al., 2018). However it is worth noting that for the case where the fractal structure is a set of points (as opposed to a continuous image), then $q = 0$ recovers the mono-fractal dimension, $\mathcal{D}$, of those points (Hentschel and Procaccia, 1983; Halsey et al., 1986). As $q \to +\infty$, $\mathcal{D}_q$ characterises the scaling of the high density structures, and as $q \to -\infty$, $\mathcal{D}_q$ characterises the scaling of the low density structures.

Figure 3.10 shows two plots of $\mathcal{D}_{20}$ against $\mathcal{D}_{-10}$. Each of the plots contains three groupings of points, with each group representing xfBm fields with three different $\mathcal{S}$, and with a random $\mathcal{H}$ that is uniformly chosen from the interval $[0, 1]$. The three values of $\mathcal{S}$ are: $\mathcal{S} = 0.1$ which

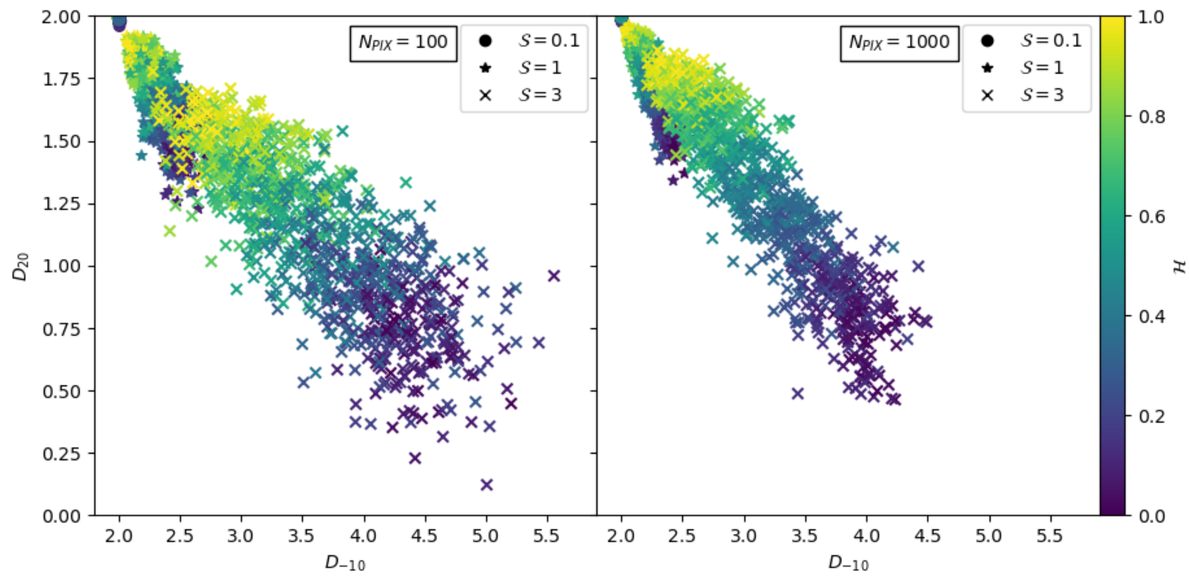


Fig. 3.10 Plots of $\mathcal{D}_{20}$ against $\mathcal{D}_{-10}$, where each point represents an xfBm field. Each of the plots has three distinct groupings for three particular values of $\mathcal{S}$. Fields with $\mathcal{S} = 0.1$ are represented with a circle (•), fields with $\mathcal{S} = 1$ are represented with a star (★), while fields with $\mathcal{S} = 3$ are represented with a cross ($\times$). Each grouping contains 1000 fields. Each field was generated with a random value of $\mathcal{H}$ picked from a uniform distribution in the interval $[0, 1]$. The value of $\mathcal{H}$ is represented by the colour-bar. The panel on the left only contains fields with $N_{\text{PIX}} = 100$, while the panel on the right only contains fields with $N_{\text{PIX}} = 1000$.

are represented with a circle (•); $\mathcal{S} = 1$ which are represented with a star (★); and $\mathcal{S} = 3$ which are represented with a cross ($\times$). Each grouping has 1000 xfBm fields. The fields in the panel on the left have $N_{\text{PIX}} = 100$, while the field in the panel on the right have $N_{\text{PIX}} = 1000$. The values of $\mathcal{H}$ for each field are shown via the colour-bar. This is a reworking of Figure 19 found in Elia et al. (2018), but for xfBm fields. For their study Elia et al. (2018) show $\mathcal{D}_{20}$ and $\mathcal{D}_{-10}$ of fields from: (i) 12 cloud simulations, (ii) 6 Hi-GAL tiles (Molinari et al., 2010), and (iii) 18 fBm images with $\beta = 2$ to 4 (equivalent to $\mathcal{H} = 0$ to 1). These three different sets of data occupy different regimes in the original plot, and are thus easily distinguishable. For our study we find that there is a fairly large area that xfBm fields can take, with this area being more well-defined with higher N_{PIX}. There is a noticeable trend for both plots where as $\mathcal{H}$ decreases the fields move from a high $\mathcal{D}_{20}$, low $\mathcal{D}_{-10}$ region to a low $\mathcal{D}_{20}$, high $\mathcal{D}_{-10}$ region. We also see this trend for when $\mathcal{S}$ increases, with the regions occupying a larger space. We notice that there is quite a bit of overlap between the $\mathcal{S} = 1$ group and the $\mathcal{S} = 3$ group for both plots, where low $\mathcal{H}$ fields with $\mathcal{S} = 1$ occupy the same region as that of high $\mathcal{H}$ fields with $\mathcal{S} = 3$. Thus we can conclude that, due to this degeneracy, this particular plot cannot be reliably used to distinguish between these two types of xfBm fields.

In the next chapter we will focus on designing and constructing our own model for estimating both $\mathcal{H}$ and $\mathcal{S}$ for noisy and non-periodic xfBm fields. The ultimate aim is to use this model to produce maps of $\mathcal{H}$ and $\mathcal{S}$ for real-life observations. Thus we look towards Machine Learning and design a model that is capable of producing fast and efficient estimations of both $\mathcal{H}$ and $\mathcal{S}$.

CHAPTER 4

The Convolutional Neural Network - Design, Training, and Testing

In this chapter we shall give a brief overview of the Convolutional Neural Network (CNN), how it is designed, trained, tested, and deployed. We start off by defining the term Machine Learning and how Neural Networks, including CNNs, fit within this paradigm.

4.1 Machine Learning

Consider a vector $\mathbf{x} = (x_1, x_2, ..., x_m)$ that maps to a vector $\mathbf{y} = (y_1, y_2, ..., y_n)$ via some function f:

$$\mathbf{y} = f(\mathbf{x}) \tag{4.1}$$

Given a known $\mathbf{x}$ and $\mathbf{y}$, the general goal of machine learning algorithms is to determine, or at least approximate, f. There are many Machine Learning algorithms that aim to do this. Examples of Machine Learning algorithms include Support Vector Machines (Cortes and Vapnik, 1995; Angulo, 2008), and Decision Trees (Breiman et al., 1984; Quinlan, 1986, 1993). The algorithm that is the focus of this work is a specialised architecture of the Neural Network, called the Convolutional Neural Network (CNN) (LeCun et al., 1989, 1998). We will give a brief description of the Convolutional Neural Network in the next section below. We recommend both Bishop (1995) for a rundown of Neural Networks and the textbook by Bishop (2006) for a more formal mathematical description of different machine learning techniques, including CNNs.

4.2 What is a Convolutional Neural Network?

In this section we describe what a Convolutional Neural Network (CNN) is, how it is designed, trained, and tested. In the next chapter (Chapter 5) we then apply CNNs to astronomical data. A Convolutional Neural Network is a specialised form of the Neural Network that is designed to work with image data. In Equation 4.1, $\mathbf{x}$ is a vector that consists of the scalar values $(x_1, x_2, ..., x_m)$. Since these are scalar values a Neural Network with 1-dimensional layers is needed. However if $\mathbf{x}$ is a matrix,

$$\mathbf{x} = \begin{bmatrix} x_{11} & \dots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nm} \end{bmatrix} \tag{4.2}$$

then a good architecture to use is the Convolution Neural Network. This is because the CNN is good for when neighbouring values of x_{nm} are closely correlated, such as the pixels in an image. The CNN can also be used in the cases where $\mathbf{x}$ is a rank 3 tensor, such as a colour image with 3 RGB channels or a data cube (LeCun et al., 1998). In both these cases each of the corresponding pixels across different channels are also correlated. In this thesis we will only consider the case where $\mathbf{x}$ is a 2-dimensional matrix as defined by equation 4.2 since most of the data that will be used is in the form of 2-dimensional images.

4.2.1 Artificial Neural Network

A neural network consists of a network of units called neurons. Each neuron takes a vector of inputs, such as $\mathbf{x} = (x_1, x_2, ..., x_m)$ and outputs a single value, y,

$$y = f(\mathbf{w} \cdot \mathbf{x} + b). \tag{4.3}$$

Here $\mathbf{w} = (w_1, w_2, ..., w_m)$ is a weight vector, b is a bias and f is an activation function. Figure 4.1 shows a graphical representation of a neuron. The activation function can represent any nonlinear function, and can be chosen freely. Common examples of activation functions include the Rectified Linear Unit (ReLU) (Nair and Hinton, 2010), and the logistic function. The logistic is defined as,

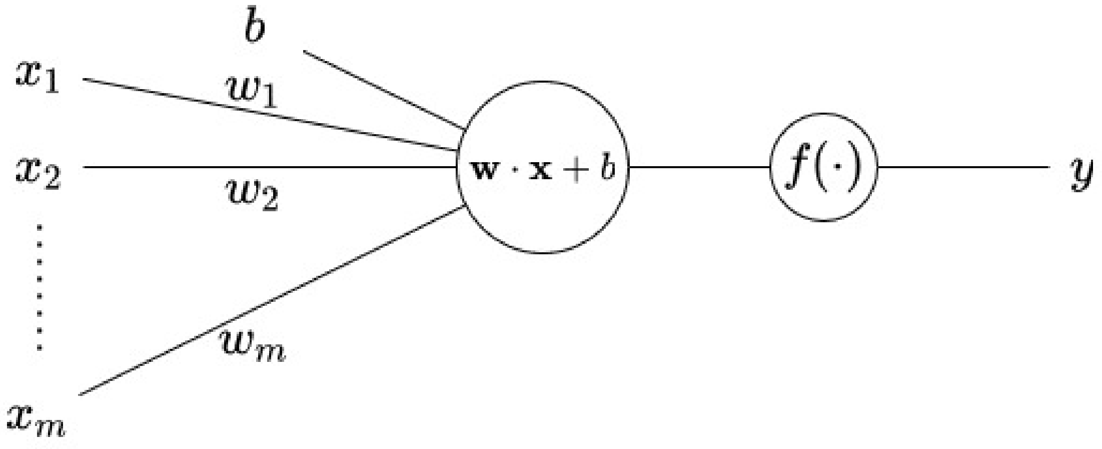


Fig. 4.1 A graphical representation of a neuron, which takes an input $\mathbf{x} = (x_1, x_2, ..., x_m)$ and outputs $y = f(\mathbf{w} \cdot \mathbf{x} + b)$

$$f_{LOG}(x) = \frac{1}{1 + e^{-x}} \tag{4.4}$$

while the ReLU is defined as,

$$f_{ReLU}(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ x, & \text{if } x > 0 \end{cases} \tag{4.5}$$

Figure 4.2 shows the ReLU and logistic activation functions in graphical form. Activation functions serve as a way of constraining the output of the neuron, for example, the logistic function constrains the output of a neuron to lie between 0 and 1, while the ReLU function is linear for all positive inputs but 0 for all negative inputs. The choice of activation function within a larger neural network can affect its performance (Bishop, 2006) and as a result we experiment with different activation functions in order to find one that works best. In the CNNs used in this thesis the activation function used is the ReLU. We find that the ReLU is the activation function that gives the best performing CNN (see Section 4.4.3).

We can chain multiple neurons together to form a layer:

$$\mathbf{y} = \mathbf{f}(\mathbf{W} \cdot \mathbf{x} + \mathbf{b}). \tag{4.6}$$

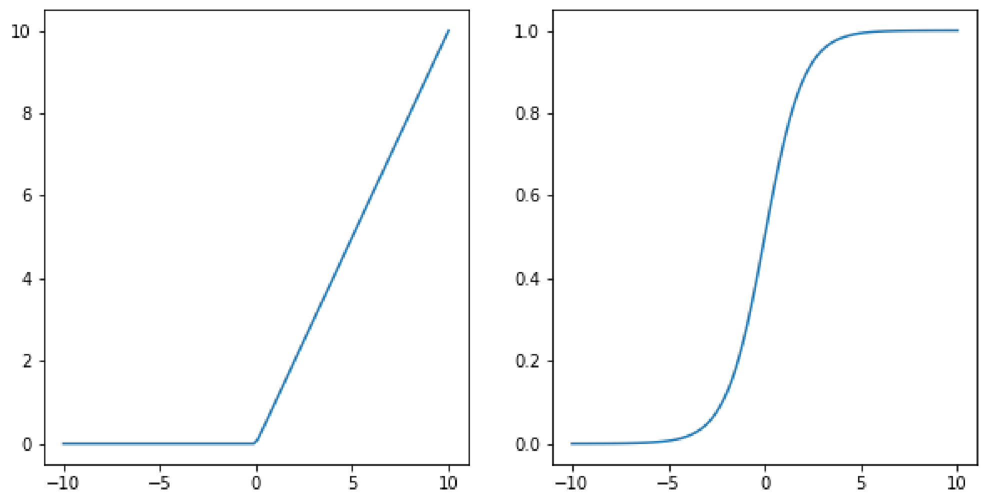


Fig. 4.2 *Left:* Rectified Linear Unit (ReLU) *Right:* Logistic or Sigmoid

Instead of one output, y, we have a vector of outputs, $\mathbf{y} = (y_1, y_2, ..., y_u)$, with each y_u corresponding to the output of each neuron in the layer. The weights are now a matrix of size $u \times m$ which takes the form,

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_u \end{bmatrix} = \begin{bmatrix} w_{11} & \dots & w_{1m} \\ \vdots & \ddots & \vdots \\ w_{u1} & \dots & w_{um} \end{bmatrix} \tag{4.7}$$

while the bias is now a vector of the form $\mathbf{b} = (b_1, b_2, ..., b_u)$. The vector function $\mathbf{f}$ is defined such that each scalar output of $\mathbf{W} \cdot \mathbf{x} + \mathbf{b}$ is passed through an activation function f, such that $\mathbf{f}(\mathbf{W} \cdot \mathbf{x} + \mathbf{b}) = [f(\mathbf{w}_1 \cdot \mathbf{x} + b_1), f(\mathbf{w}_2 \cdot \mathbf{x} + b_2), ..., f(\mathbf{w}_u \cdot \mathbf{x} + b_u)]$. Figure 4.3 shows a graphical representation of a single layer with 3 neurons ($u = 3$). The layer in this example could also serve as the output layer, as by convention the symbol, $\mathbf{y}$, is reserved for the final outputs of the model.

Layers can also be chained together, where the outputs of one layer become the inputs of the next layer, with each layer having their own weights and biases,

$$\mathbf{y} = \mathbf{F}_l(...\mathbf{F}_2(\mathbf{F}_1(\mathbf{z}))...) \tag{4.8}$$

where each $\mathbf{F}_l(\mathbf{z}) = \mathbf{f}_l(\mathbf{W}_l \cdot \mathbf{z} + \mathbf{b}_l)$. This type of network is usually referred to as an Artificial

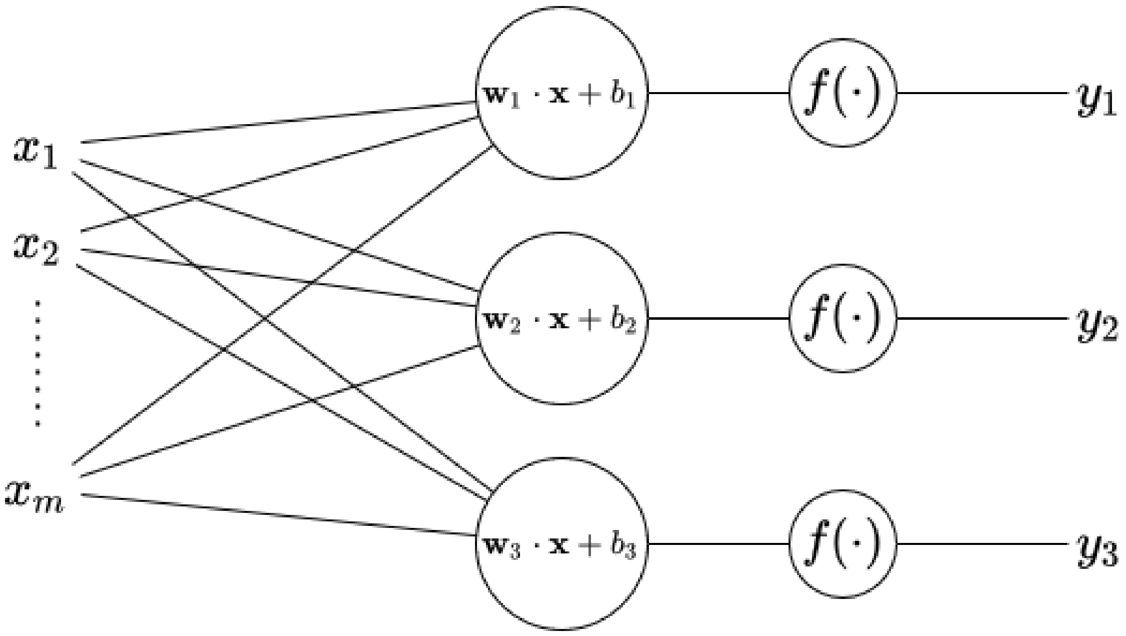

x_1
x_2
x_m
$\mathbf{w}_1 \cdot \mathbf{x} + b_1$
$\mathbf{w}_2 \cdot \mathbf{x} + b_2$
$\mathbf{w}_3 \cdot \mathbf{x} + b_3$
$f(\cdot)$
$f(\cdot)$
$f(\cdot)$
y_1
y_2
y_3

Fig. 4.3 A graphical representation of a single layer, consisting of 3 neurons. This layer takes as an input a vector $\mathbf{x} = [x_1, x_2, \ldots, x_m]$, and outputs $\mathbf{y} = [y_1, y_2, y_3]$.

Neural Network and each layer is usually referred to as a Dense layer. Dense layers can be equivalently referred to as flat or fully connected, which are all terms that refer to the same 1-dimensional layer of neurons, where each neuron is connected to each input from the previous layer and to each output in the next layer. Figure 4.4 shows a graphical representation of an ANN where $l = 3$ (i.e . In this particular example, the input is a vector $\mathbf{x}_1$. The first layer outputs $\mathbf{x}_2 = \mathbf{f}_1(\mathbf{W}_1 \cdot \mathbf{x}_1 + \mathbf{b}_1)$, and the second layer outputs $\mathbf{x}_3 = \mathbf{f}_2(\mathbf{W}_2 \cdot \mathbf{x}_2 + \mathbf{b}_2)$. The third and final layer outputs $\mathbf{y} = \mathbf{f}_3(\mathbf{W}_3 \cdot \mathbf{x}_3 + \mathbf{b}_3)$. In this particular example the neural network outputs two scalar values $\mathbf{y} = [y_1, y_2]$.

The number of scalar outputs in $\mathbf{y}$ is arbitrary and depends on the type of problem at hand. For example, in a typical binary classification problem a single output ($\mathbf{y} = y$) is sufficient. In this case it makes sense that the activation function for the final layer be the sigmoid, $f_{LOG}(x)$. Thus $0 \leq y \leq 1$ and one class can be represented when $y > 0.5$ and the other class is represented when $y \leq 0.5$.

The Convolutional Neural Networks that are used in this work will strictly output two values: $\mathcal{H}$ and $\mathcal{S}$. We do not constrain these outputs and therefore we do not put activation functions on the final layer of our CNNs.

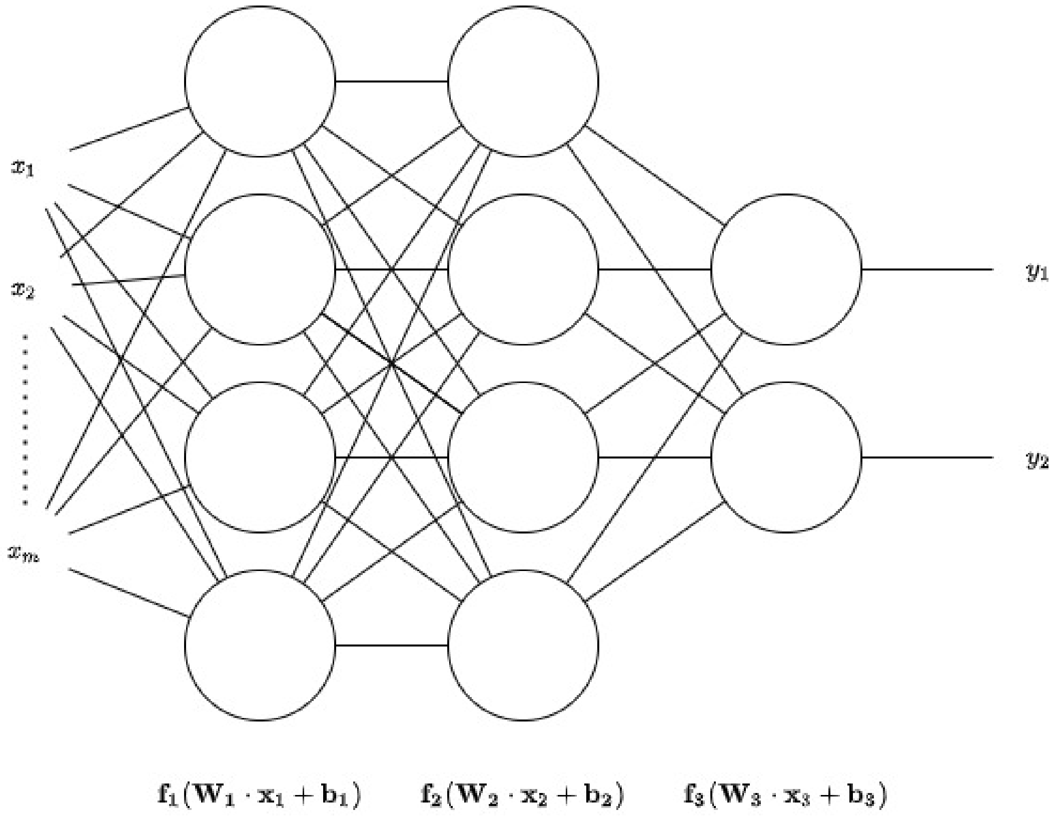

x_1
x_2
x_m
y_1
y_2
$\mathbf{f_1}(\mathbf{W_1} \cdot \mathbf{x_1} + \mathbf{b_1})$
$\mathbf{f_2}(\mathbf{W_2} \cdot \mathbf{x_2} + \mathbf{b_2})$
$\mathbf{f_3}(\mathbf{W_3} \cdot \mathbf{x_3} + \mathbf{b_3})$

Fig. 4.4 A graphical representation of a multilayer perceptron model that has three layers with the first two layers having four neurons. The final layer is the output layer and has two neurons. This neural network takes a vector $\mathbf{x}_1 = [x_1, x_2, \dots x_m]$ as an input and outputs $\mathbf{y} = [y_1, y_2]$

4.2.2 Convolutional Neural Network

A Convolutional Neural Network is a Neural Network where one or more of the layers consists of a convolutional layer (LeCun et al., 1989). In a convolutional layer each node represents a convolution operation followed by an activation function. The input to the convolutional layer used in this thesis is a 2-dimensional matrix, of the form seen in Equation 4.2. The convolution operation works as follows. First, a subsection of the input $\mathbf{x}$ is selected, $\mathbf{x}_{sub}$, and convolved with a filter $\mathbf{G}$. A bias is then added and an activation function is applied,

$$\mathbf{f}(\mathbf{x}_{sub} \circledast \mathbf{G} + b). \tag{4.9}$$

This operation produces a single scalar value.

This operation is then repeated on a different subsection of $\mathbf{x}$ for all desired subsections, keeping $\mathbf{G}$, and b constant. The size of $\mathbf{G}$ is set as required for the task. For the rest of this work we shall work with a filter size of 3×3,

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix} \tag{4.10}$$

The size of $\mathbf{x}_{sub}$ is set by the size of the filter used. However which subset of the original image is selected is set by the user. For each $\mathbf{x}$ we choose a 3×3 window, $\mathbf{x}_{sub}$. The operation shown in Equation 4.9 is applied. This produces a scalar value. The sliding window is then moved over to the next subset of $\mathbf{x}$ and the process is repeated. The amount that this sliding window is moved is called the *stride*, and is set as required. For the rest of this work we will set the stride to be 1×1. This means that the sliding window moves across 1 column after each operation. After reaching the last column, the sliding window then moves 1 row down and repeats the process. Having the stride smaller than the filter size means that each pixel will contribute more than once to the output. It is important to note that in this explanation we have mentioned that the sliding window does the convolution operation sequentially, however each of these convolution operations are independent of one another and as a result can be made in parallel. This embarrassingly parallel[1] task is an ideal usecase for Graphical Processing Units

[1]Embarrassingly parallel tasks are ones where it is trivial to split the task into multiple subtasks that can be

(GPUs) and as a result all the models in this thesis have been done using the GPUs.

Figure 4.5 shows how a single node performs a convolution operation on an input. $\mathbf{x}$. This convolutional node has a 3×3 filter, $\mathbf{G}$, and a 3×3 sliding window $\mathbf{x}_{sub,n}$ with a stride of 3×3. The input to the node is a 6×6 matrix, $\mathbf{x}$. The top panel shows how the sliding window selects the first subset $\mathbf{x}_{sub,1}$ and performs the convolution operation on it. The sliding window then moves 3 spaces to the right to select $\mathbf{x}_{sub,2}$ and performs the convolutional operation again. At this point the sliding window has reached the right end of the matrix, so it then moves back to the start of the columns and moves 3 places down. It then repeats the entire process until the sliding window reaches the bottom right of $\mathbf{x}$. Since, in this example, the stride is the same size as the filter window there will be no overlapping values of $\mathbf{x}$ in each $\mathbf{x}_{sub}$. The output of the node is a 2×2 matrix. This is shown in the bottom panel. This output size can be controlled by modifying the stride. A larger stride will produce a smaller output size, while a stride of 1×1 will only reduce the output size by 2 in each dimension (one for each side of the image). The activation function is then applied to all the resulting values, although this is not shown in the diagram for simplicity.

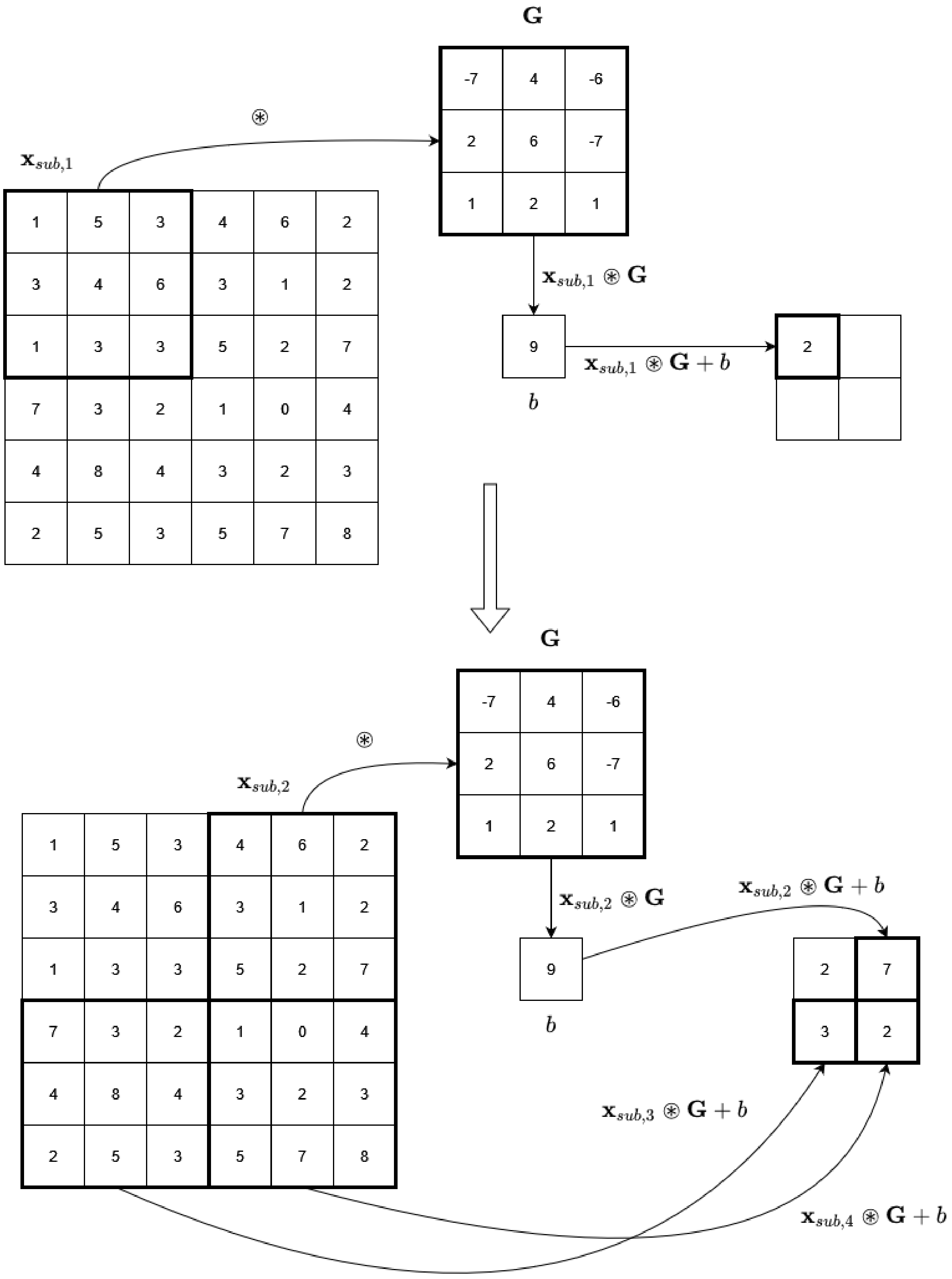
Like in the Artificial Neural Network, we can chain multiple convolutional nodes to form a layer, which we will call l. Each node in the layer will have its own different filter, $\mathbf{G}_n$ and bias, b_n. Since each node will output its own 2-dimensional matrix, the layer will output a rank-3 tensor, $\mathbf{x}_{l+1}$, consisting of N_n of the 2-dimensional matrices. Here N_n refers to the number of nodes in the layer. Thus the input to each node in layer $l + 1$ will be $\mathbf{x}_{l+1}$.

Convolutional layers can also be accompanied by Max Pooling layers. A Max Pooling layer is similar in operation to a Convolutional layer with the difference being that the maximum value of the window is outputted instead of a convolution with some filter. The outputs to a CNN can be fed as an input vector to an Artificial Neural Network. This is the approach that is used in this work.

4.3 Training

Since $\mathbf{W}$ (or $\mathbf{G}$ in the case of a CNN) and $\mathbf{b}$ are unknown quantities they need to be determined. This is done through a process called *training*. Consider a training set consisting of inputs $\{\mathbf{x}_0, ..., \mathbf{x}_N\}$ and targets $\{\mathbf{t}_0, ..., \mathbf{t}_N\}$. During training a large number of these values

performed independently and in parallel.



$\mathbf{G}$
-7 4 -6
2 6 -7
1 2 1
$\circledast$
$\mathbf{x}_{sub,1}$
1 5 3 4 6 2
3 4 6 3 1 2
1 3 3 5 2 7
7 3 2 1 0 4
4 8 4 3 2 3
2 5 3 5 7 8
$\mathbf{x}_{sub,1} \circledast \mathbf{G}$
9
b
$\mathbf{x}_{sub,1} \circledast \mathbf{G} + b$
2
$\mathbf{G}$
$\circledast$
$\mathbf{x}_{sub,2}$
$\mathbf{x}_{sub,2} \circledast \mathbf{G}$
$\mathbf{x}_{sub,2} \circledast \mathbf{G} + b$
2 7
3 2
$\mathbf{x}_{sub,3} \circledast \mathbf{G} + b$
$\mathbf{x}_{sub,4} \circledast \mathbf{G} + b$

Fig. 4.5 An example showing how a convolutional node works. The activation function operation has been omitted for simplicity. The convolution operation in this example has a filter size of 3×3 with a stride of 3×3.

of $\mathbf{x}_n$ are passed through the model and each of the outputs, $\mathbf{y}_n$ is compared with the true value, $\mathbf{t}_n$. This is usually done through the use of an error function. The error function used here is the sum-of-squares error,

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n, \mathbf{W}) - \mathbf{t}_n||^2, \tag{4.11}$$

where we have absorbed $\mathbf{G}$ and $\mathbf{b}$ into $\mathbf{W}$ for simplicity.

By minimising $E(\mathbf{W})$ we can determine the value of $\mathbf{W}$ for which the Neural Network is said to be trained. Minimisation of $E(\mathbf{W})$ is done numerically using *gradient descent*, such that $\nabla E(\mathbf{W}) \approx 0$. Gradient descent is done in a series of steps. For each step, τ, the weights are updated according to,

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} + \Delta\mathbf{W}^{(\tau)} \tag{4.12}$$

where $\Delta\mathbf{W}^{(\tau)}$ is the update to the weight vector's value at that step and is given by,

$$\Delta\mathbf{W}^{(\tau)} = -\eta_l \nabla E_n(\mathbf{W}^{(\tau)}). \tag{4.13}$$

Here η_l is known as the learning rate, and it controls how big of a jump in the weight space the step performs. When the learning rate is too small, the model takes a long time to train, if the learning rate is too big, the model will struggle to find a minimum. It is important to note that the minimum reached by minimising $E(\mathbf{W})$ may not be a global minimum but a local one. One way to mitigate this is to train a mode multiple times and then pick the best performing one.

The process of calculating the gradient of the error function, $\nabla E_n(\mathbf{W}^{(\tau)})$, is called backpropagation. Backpropogation is done via the chain rule, which is composed of the partial derivatives of the error function with respect to each individual weight value. We refer the reader to Chapter 5 of Bishop (2006) for a detailed derivation on backpropagation and gradient descent, as well as a detailed mathematical formulation of Neural Networks. Here we perform each step τ with each data point n, however it is faster to perform each step τ with a batch of data points. This is called batch training.

A single pass of the full data through this training process is called an epoch. Machine

Learning models can be trained for multiple epochs, however there is a point at which the model starts to overfit to the training data. It is therefore common practice to record the value of the *loss* after each epoch which is then compared to the loss of an independent set of data points. This set of independent data points is called the validation dataset. The loss is a metric that is used to keep track of the performance of the model as it trains. Equation 4.11 is one example of a *loss function* The loss that is determined during training is known as the *training loss* while the loss determined using the validation dataset is known as the *validation loss*. By comparing the training loss with the validation loss after each epoch, we can stop the training when the validation loss stops decreasing. The model is then said to be trained and it can then be deployed. This process is know as cross-validation.

In the case of our CNN models, we do not have a fixed size data set as the xfBm images can be generated on the fly. As a result we define the end of an epoch after we have trained for 20,000 xfBm images. The validation loss, $\mathcal{L}_{VAL}$, is then measured using an addition 200 independently generated xfBm fields.

4.4 Constructing Our CNNs

For constructing, training and testing our Convolutional Neural Networks we use the Python package `TensorFlow` (TensorFlow-Developers, 2023). `TensorFlow` is a package that allows the assignment of arrays to array objects called Tensors. These Tensors can then be allocated to Graphical Processing Units (GPUs) in order to make the training process parallelisable. We use the sub-package `Keras` to build the layers. `Keras` provides a high-level API that can be used to define whole layers rather than constructing the layers manually using the Tensors. We train 3 CNNs in total. To differentiate between the 3 CNNs, we designate them with the name CNN:$N_{\rm PIX}$, where $N_{\rm PIX}$ refers to the $N_{\rm PIX} \times N_{\rm PIX}$ pixel image that is the input for CNN:$N_{\rm PIX}$. The values of $N_{\rm PIX}$ are $N_{\rm PIX} = [32, 64, 128]$. Each CNN outputs a value for $\mathcal{H}$ and a value for $\mathcal{S}$.

4.4.1 Architecture

Each CNN consists of enough Convolutional-MaxPool layer combinations to reduce the image into a 2×2 map. Each convolutional layer is 256 nodes wide and consists of a 3×3 kernel with a stride of 1×1. Each convolutional layer is then followed by a max pool layer.

Each max pool layer has a 2×2 kernel size with a stride of 2. The final convolutional layer of each CNN:N_{PIX} has an output shape of $2 \times 2 \times 256$. This output is then flattened into a 1024-length vector. This 1024-vector is then fed into 5 additional feed-forward layers with each feed-forward layer consisting of 256 nodes. An additional final 2-node layer is then used to output $\mathcal{H}$ and $\mathcal{S}$. Tables 4.1, 4.2, and 4.3 show the architecture of CNN:128, CNN:64, and CNN:32 respectively.

Each of these models has a large number of trainable parameters with the number of trainable parameters being dependant on the size of the model (i.e. how many layers and nodes it has). The architecture of CNN:128 is shown in Table 4.1. This table gives the number of trainable parameters at each layer. The Input layer does not have any trainable parameters as it is simply loading the data in and sending it to the next layer. The Conv.1 layer is the first convolutional layer. This layer consists of 256 different 3×3 kernels and a single bias per kernel, making 10 trainable parameters per kernel, for a total of 2560 trainable parameters. This layer takes the 128×128 input image and applies each of the convolutional filters to the image. This produces 256 different maps each of size 126×126. This reduction in size of the map is due to the convolution filter having a stride of 1×1, which reduces the size of the map by 2 in each dimension. The MaxPool.1 layer simply picks the maximum value in a moving 2×2 window, and thus has no trainable parameters. This moving window has a stride of 2×2. As a result, this layer halves the size of each of the 256 maps in each direction, producing an output of size $63 \times 63 \times 256$. The subsequent Conv layers have more trainable parameters than Conv.1 and this is purely due to the fact that Conv.1 only had a single map as an input while the rest have 256 maps as inputs. Each of the input maps will have 256 distinct trainable kernels, thus each layer will have 256^2 total trainable kernels. Each kernel has 3×3 trainable parameters giving $256^2 \times 3 \times 3 = 589,824$ trainable parameters. Adding a bias for each of the maps gives an additional 256 trainable parameters giving $589,824 + 256 = 590,080$ total trainable parameters for that layer. This matches what we see in Table 4.1.

The output of MaxPool.5 is a tensor consisting of 256 2×2 maps. These are then flattened by the Flatten layer to produce a 1-dimensional vector of size 1024. This vector is then fed into an ANN, consisting of 5 fully connected layers (See Section 4.2.1), each with 256 nodes. Dense.1 is the first of these fully connected layers and has 262,400 trainable parameters. This number of trainable parameters is the result of each of the 256 nodes having 1024 weights

Table 4.1 The architecture of CNN:128. The CNN takes an 128×128 pixel map that has been normalised and outputs an estimate for $\mathcal{H}$ and $\mathcal{S}$. The CNN does this by passing the image through 5 convolutional layers and 5 fully connected layers. Each convolutional layer outputs the convolution of a 3x3 windows with a trainable kernel and is followed by a max pooling layer that outputs the maximum value of a 2×2 sliding window with a stride of 2×2. This model has 2,888,962 trainable parameters.

Layer	Output Size	Operation	Trainable Parameters
Input	$128 \times 128 \times 1$	input layer	0
Conv.1	$126 \times 126 \times 256$	3×3 kernel	2560
MaxPool.1	$63 \times 63 \times 256$	2×2 max pooling	0
Conv.2	$61 \times 61 \times 256$	3×3 kernel	590080
MaxPool.2	$30 \times 30 \times 256$	2×2 max pooling	0
Conv.3	$28 \times 28 \times 256$	3×3 kernel	590080
MaxPool.3	$14 \times 14 \times 256$	2×2 max pooling	0
Conv.4	$12 \times 12 \times 256$	3×3 kernel	590080
MaxPool.4	$6 \times 6 \times 256$	2×2 max pooling	0
Conv.5	$4 \times 4 \times 256$	3×3 kernel	590080
MaxPool.5	$2 \times 2 \times 256$	2×2 max pooling	0
Flatten	$1 \times 1 \times 1024$	flattens into 1D layer	0
Dense.1	$1 \times 1 \times 256$	fully connected	262400
Dense.2	$1 \times 1 \times 256$	fully connected	65792
Dense.3	$1 \times 1 \times 256$	fully connected	65792
Dense.4	$1 \times 1 \times 256$	fully connected	65792
Dense.5	$1 \times 1 \times 256$	fully connected	65792
Output	$1 \times 1 \times 2$	one channel each for $\mathcal{H}$ and $\mathcal{S}$	0

(due to input size) and a bias, giving $256 \times (1024 + 1) = 262,400$. All subsequent Dense layers have 65,792 trainable parameter, which is different from Dense.1 due to the different input sizes ($256 \times [256 + 1] = 65792$).

For each convolutional layer we apply a ReLU activation function while for the Dense layers we do not apply any activation functions.

4.4.2 Training

An initial exploration on the training of a CNN was carried out in Bates et al. (2020) using CNN:128. As a result, there are a few improvements in the way the CNNs in this thesis have been trained compared to the one explored in Bates et al. (2020). This initial study is briefly covered in the next section (Section 4.4.2.1) where we outline the training method and some findings. In Section 4.4.2.2 we explore the improvements that were made to the training compared to Bates et al. (2020). It is also worth noting that the CNN:128 trained in Bates et al. (2020) uses β instead of $\mathcal{H}$ however since β scales linearly with $\mathcal{H}$ (See Equation 2.3)

Table 4.2 The architecture of CNN:64. The CNN takes a 64×64 pixel map as input. CNN:64 has 4 convolutional layers and 5 fully connected layers. This model has 2,298,882 trainable parameters.

Layer	Output Size	Operation	Trainable Parameters
Input	$64 \times 64 \times 1$	input layer	0
Conv.1	$62 \times 62 \times 256$	3×3 kernel	2560
MaxPool.1	$31 \times 31 \times 256$	2×2 max pooling	0
Conv.2	$29 \times 29 \times 256$	3×3 kernel	590080
MaxPool.2	$14 \times 14 \times 256$	2×2 max pooling	0
Conv.3	$12 \times 12 \times 256$	3×3 kernel	590080
MaxPool.3	$6 \times 6 \times 256$	2×2 max pooling	0
Conv.4	$4 \times 4 \times 256$	3×3 kernel	590080
MaxPool.4	$2 \times 2 \times 256$	2×2 max pooling	0
Flatten	$1 \times 1 \times 1024$	flattens into 1D layer	0
Dense.1	$1 \times 1 \times 256$	fully connected	262400
Dense.2	$1 \times 1 \times 256$	fully connected	65792
Dense.3	$1 \times 1 \times 256$	fully connected	65792
Dense.4	$1 \times 1 \times 256$	fully connected	65792
Dense.5	$1 \times 1 \times 256$	fully connected	65792
Output	$1 \times 1 \times 2$	one channel each for $\mathcal{H}$ and $\mathcal{S}$	0

Table 4.3 The architecture of CNN:32. The CNN takes a 32×32 pixel map as input. CNN:32 has 3 convolutional layers and 5 fully connected layers. This model has 1,708,802 trainable parameters.

Layer	Output Size	Operation	Trainable Parameters
Input	$32 \times 32 \times 1$	input layer	0
Conv.1	$30 \times 30 \times 256$	3×3 kernel	2560
MaxPool.1	$15 \times 15 \times 256$	2×2 max pooling	0
Conv.2	$13 \times 13 \times 256$	3×3 kernel	590080
MaxPool.2	$6 \times 6 \times 256$	2×2 max pooling	0
Conv.3	$4 \times 4 \times 256$	3×3 kernel	590080
MaxPool.3	$2 \times 2 \times 256$	2×2 max pooling	0
Flatten	$1 \times 1 \times 1024$	flattens into 1D layer	0
Dense.1	$1 \times 1 \times 256$	fully connected	262400
Dense.2	$1 \times 1 \times 256$	fully connected	65792
Dense.3	$1 \times 1 \times 256$	fully connected	65792
Dense.4	$1 \times 1 \times 256$	fully connected	65792
Dense.5	$1 \times 1 \times 256$	fully connected	65792
Output	$1 \times 1 \times 2$	one channel each for $\mathcal{H}$ and $\mathcal{S}$	0

then we can safely compare this version of CNN:128 with the final versions.

Each field that has been generated for training is normalised. This is done by first setting the standard deviation of the field to 0.25 and the mean to 0. The reason for this is to eliminate any biases due to scaling of the field as the CNN should estimate $\mathcal{H}$ and $\mathcal{S}$ purely on the correlation between the pixels rather than the scale of the field as a whole. Then any pixel, $g > 1$ is set to $g = 1$. Similarly any pixel, $g < -1$ is set to $g = -1$. This process serves two purposes. The first is to simulate cases where pixels in observational images are oversaturated, such as in the case of cosmic ray strikes. The second is that in order for the Neural Networks to train efficiently and in a stable way, it is helpful to have the inputs be of a similar scale. From a performance perspective this normalisation technique prevents certain features from dominating the contribution to the training process, and ensures equal contributions from each feature of each field.

4.4.2.1 Initial study of CNN:128

We shall briefly cover the initial study that explored this CNN method (Bates et al., 2020). First a dataset of 20,000 xfBm images were generated according to Section 2.4. Each of these xfBm field has a random value of β chosen uniformly from the interval $2 \leq \beta \leq 4$ (equivalent to $0 \leq \mathcal{H} \leq 1$) and a random value of $\mathcal{S}$ chosen uniformly from the interval $0 \leq \mathcal{S} \leq 3$. Each field is also generated with a random amount of Gaussian noise. This is done according to the process outlined in Section 2.4.1, where η is chosen from a uniform distribution in the interval $0 \leq \eta \leq 0.05$. Each field has $N_{\rm PIX} = 128$.

At each epoch a random 70% of the fields (i.e. 14,000) are allocated as a training set. This training set is then used to train the model, with the fields being used in batches of 32. The remaining 30% of the fields (6,000) are allocated as a validation set. At the end of each epoch, the training loss, $\mathcal{L}_{TRAIN}$, is recorded. The validation set is then passed through the model and the validation loss, $\mathcal{L}_{VAL}$, is recorded. The validation loss represents the performance of the model when it predicts on unseen data. This process of splitting the dataset into training and validation sets is called *cross-validation* (Bishop, 2006). This process is repeated for a total of 500 epochs. The purpose of cross-validation is to see whether the model is overfitting to the training data. It is ideal if the validation loss decreases during training, however, if the validation loss starts to increase while the training loss is still decreasing, then the model has

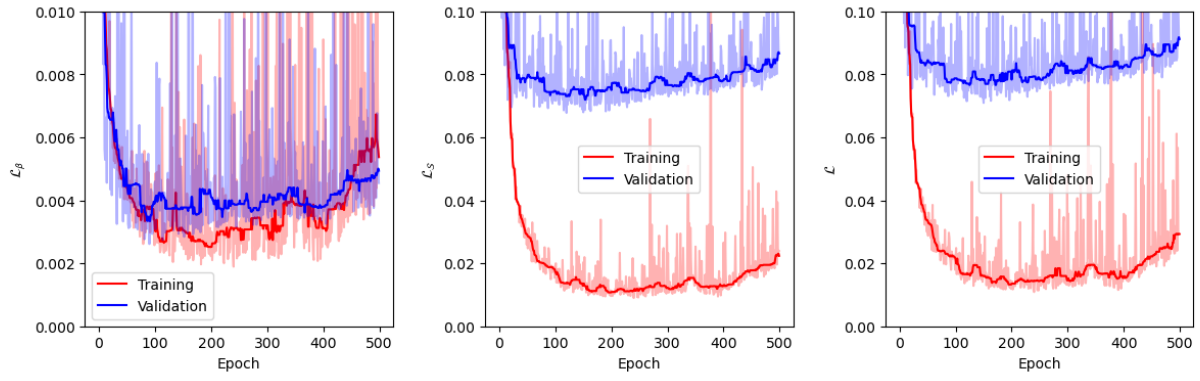


Fig. 4.6 Panels showing the training loss (red) and validation loss (blue) of the pathfinder CNN:128 that was trained in Bates et al. (2020). We have panels that represent the loss of: β ($\mathcal{L}_\beta$, left), $\mathcal{S}$ ($\mathcal{L}_\mathcal{S}$, middle), the total ($\mathcal{L} = \mathcal{L}_\beta + \mathcal{L}_\mathcal{S}$, right). The pale lines represent the actual loss values while the dark lines shows a smoothed version that is calculated by taking the median of the surrounding 20 points.

been overfitted to the training data.

Figure 4.6 shows the values of both the validation loss (in blue) and the training loss (in red) at every epoch for this version of CNN:128. Each of the panels in this figure show (from left to right): the loss for $\mathcal{H}$, $\mathcal{L}_\beta$, the loss for $\mathcal{S}$, $\mathcal{L}_\mathcal{S}$, and the total loss across both parameters, $\mathcal{L}$, which is simply the sum of the previous two losses (i.e. $\mathcal{L} = \mathcal{L}_\beta + \mathcal{L}_\mathcal{S}$). The pale lines represent the actual values of the loss while the dark lines are the smoothed out version, which is done by taking the median of a moving window consisting of 20 data points. We can see by eye that the validation loss starts to increase at ~ 100 epochs while the training loss remains roughly constant until ~ 400 epochs. At this point the training loss also starts to increase. We therefore only work with CNNs that are trained to 100 epochs when training new models.

4.4.2.2 Final versions of CNN:128, CNN:64 and CNN:32

The final version of CNN:128 as well as CNN:64 and CNN:32 were trained slightly different to the version of CNN:128 explored in Bates et al. (2020) and described in the previous section. For the reasons mentioned above these new models were only trained for 100 epochs. In the final versions, instead of generating a set number of xfBm fields that are then used for the entire training process, we generate 20,000 entirely new xfBm fields at the beginning of each epoch, which are used to train the CNN. These are fed in batches of 32 and are then discarded. At the end of each epoch an additional new 200 xfBm images are generated which serve as our validation set and thus are used to calculate the validation loss for that epoch.

These are then discarded. A new set of 20,000 xfBm images are then generated for the next epoch's training set, as well as a new validation set. This process is repeated for each of the 100 epochs. This process is better at avoiding overfitting as the model never 'sees' the same image twice and as a result generalises well. As in the previous section, for each field, $\mathcal{H}$ and $\mathcal{S}$ were randomly picked from the uniform intervals $[0, 1]$ and $[0, 3]$. Each field is also generated with a random amount of Gaussian noise as described in the previous section. In addition, we also randomise the position of the centre of mass of each image. This is done since real-life observations the ISM might not have perfectly centred structures.

Figure 4.7 shows the loss against epoch for the training of the revised version of CNN:128 as well as for CNN:64 and CNN:32. We see that both the validation loss and the training loss for all three models decrease systematically, without showing evidence of overfitting. The training loss for all three models settles by the time the training is finished. The training loss roughly decreases and does not start to increase. For CNN:64 and CNN:128 we observe that the validation loss roughly follows the training loss quite closely. For CNN:32 we find that the validation loss is greater than the training loss until ~ 40 epochs after which the validation loss becomes consistently lower than the training loss. In a truly generalised model we would expect the validation loss to be very similar to the training loss. CNN:64 most resembles this type of behaviour.

4.4.3 Testing

At the end of each models' training, a further 10,000 unique $N_{\rm PIX} \times N_{\rm PIX}$ xfBm images are generated per model, for a total of 30,000 images (10,000 $N_{\rm PIX} = 128$ fields for CNN:128, 10,000 $N_{\rm PIX} = 64$ fields for CNN:64, and 10,000 $N_{\rm PIX} = 32$ fields for CNN:32). These are generated in the same way as the training and validation xfBm images and are referred to as the testing set. These are then passed through the models in order to get an estimate of each models' performance. For each CNN the known values of $\mathcal{H}_{TRUE}$ and $\mathcal{S}_{TRUE}$ are compared with the models' estimate, $\mathcal{H}_{EST}$ and $\mathcal{S}_{EST}$, using the error function (Eq. 4.11).

Figure 4.8 shows this comparison with each panel showing the estimated value plotted against the true value. The top row of panels shows this comparison for $\mathcal{H}$ while the bottom row shows this for $\mathcal{S}$. Each of the columns show comparisons for different $N_{\rm PIX}$. The left column is for fields with $N_{\rm PIX} = 128$, the middle column is for fields with $N_{\rm PIX} = 64$, and the right

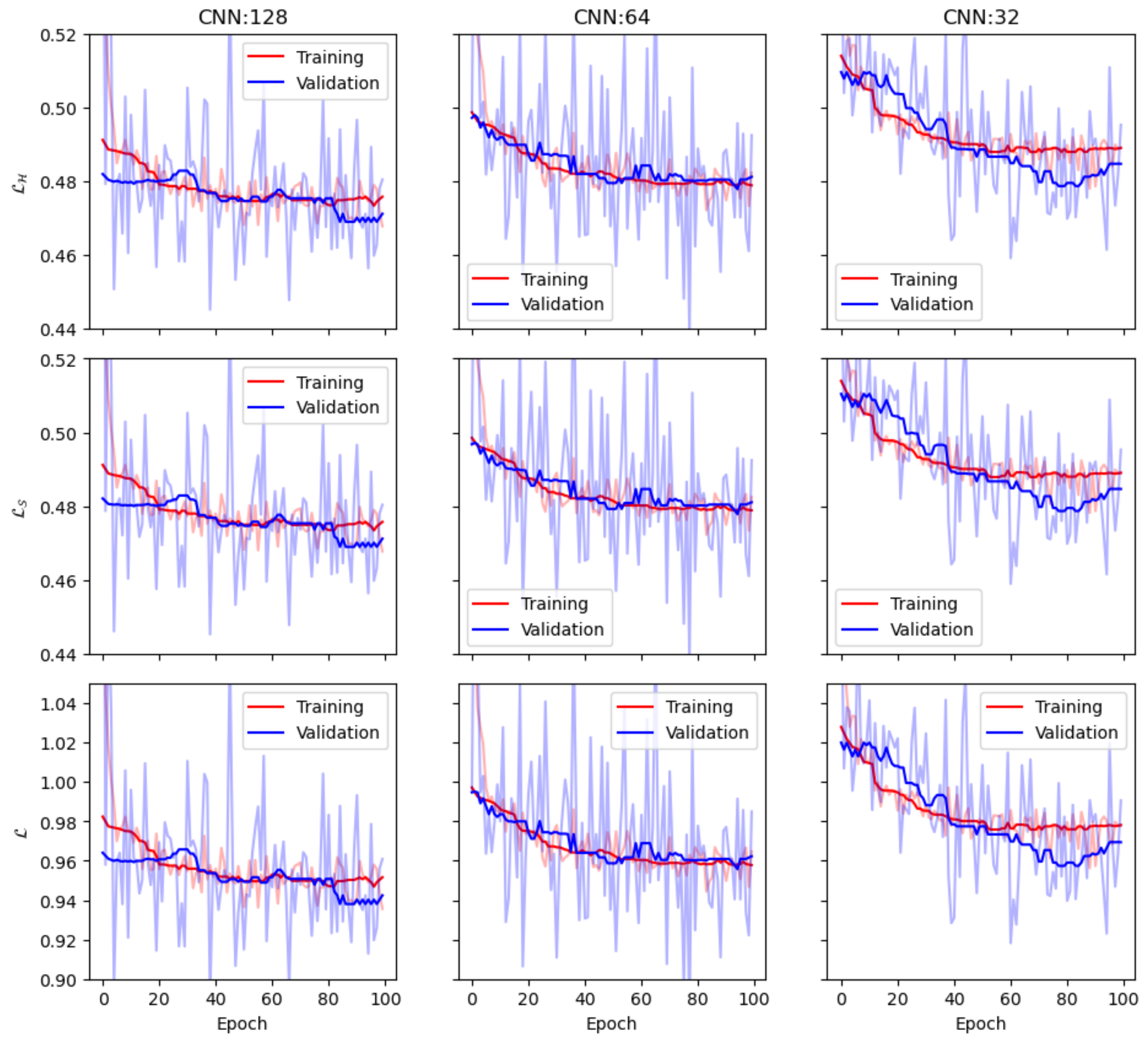


Fig. 4.7 Panels showing the training loss (red) and validation loss (blue) of: CNN:128 (left column), CNN:64 (middle column), CNN:32 (right column). Each of the rows represent the loss of: $\mathcal{H}$ ($\mathcal{L}_\mathcal{H}$, top row), $\mathcal{S}$ ($\mathcal{L}_\mathcal{S}$, middle row), the total ($\mathcal{L} = \mathcal{L}_\mathcal{H} + \mathcal{L}_\mathcal{S}$, bottom row). The pale lines represent the actual loss values while the dark lines shows a smoothed version that is calculated by taking the median of the surrounding 20 points.

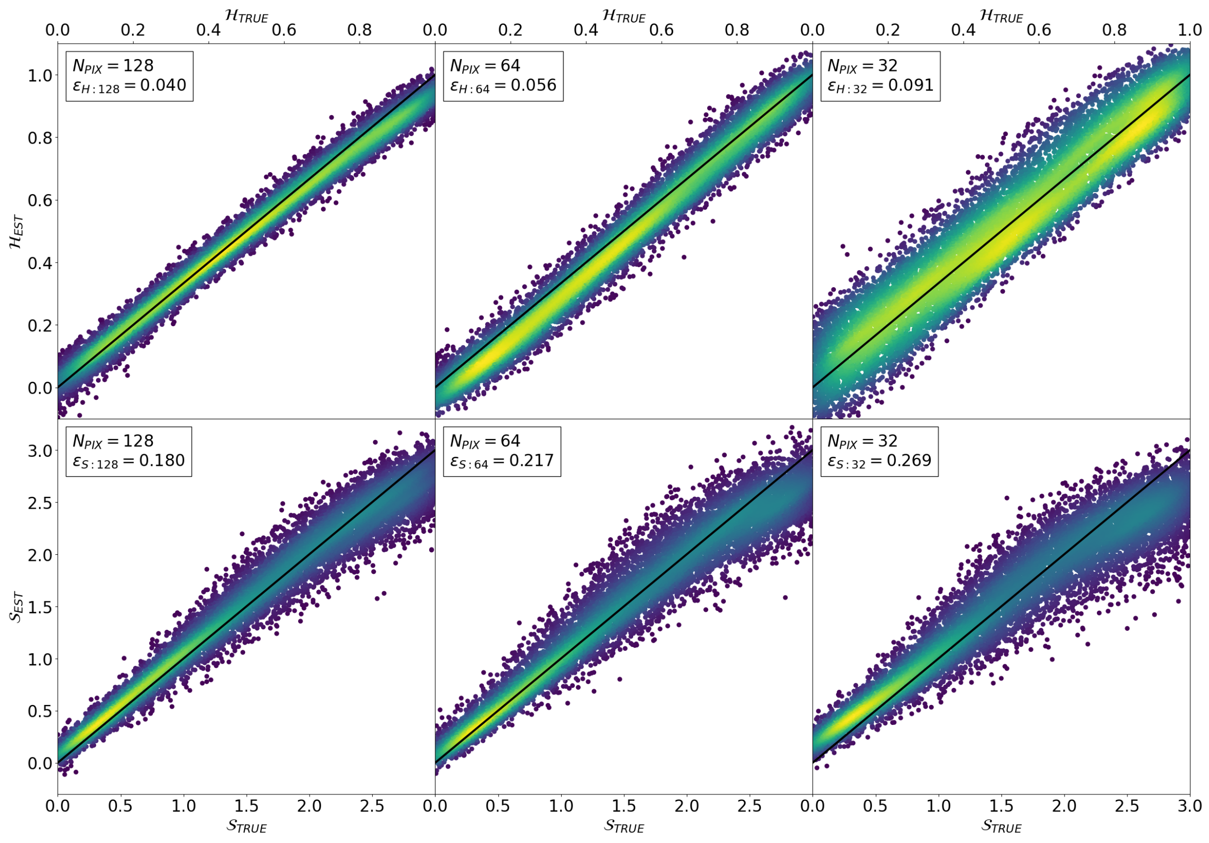


Fig. 4.8 Plots showing the estimated values of $\mathcal{H}$ and $\mathcal{S}$ against the true values when estimated using each of the CNNs. Each point represents an xfBm field with a particular $\mathcal{H}$ and $\mathcal{S}$. There are 10,000 fields per panel. The black line at $y = x$ represent the result given by a perfect estimator. The top row shows $\mathcal{H}_{EST}$ plotted against $\mathcal{H}_{TRUE}$ while the bottom row shows $\mathcal{S}_{EST}$ plotted against $\mathcal{S}_{TRUE}$. From left to right the panels in each column represent fields with $N_{\text{PIX}} = 128$, $N_{\text{PIX}} = 64$, and $N_{\text{PIX}} = 32$, respectively. The colour represents a kernel density estimation that gives the density of the points. The root mean square error of each set of points compared to the perfect estimator is given by ε in the inset.

column is for $N_{\text{PIX}} = 32$. The black 45° line represents a perfect estimator. The colour of the points represents their density. This is done using a kernel density estimation implementation in the SciPy Python package (Virtanen et al., 2020). The root mean square error ε is shown in the inset. We see that as we increase N_{PIX}, then ε decreases. This is expected since the lower N_{PIX} fields are of a lower resolution and hence have less pixels for the CNN to work with. A useful feature to look at is the spread of points along the black line, which is aided by the colour. Notably we see that for $\mathcal{H}_{EST}$ the spread of points appears roughly uniform as $\mathcal{H}_{TRUE}$ increases. By contrast, the points for $\mathcal{S}_{EST}$ are more concentrated for low $\mathcal{S}$ and more spread out for high $\mathcal{S}$. We also notice that the models overestimate $\mathcal{S}$ when $\mathcal{S} \lesssim 1.5$ while underestimating $\mathcal{S}$ when $\mathcal{S} \gtrsim 2$. This effect is more pronounced for CNN:32.

Figure 4.9 shows box-and-whisker plots for the residuals ($\mathcal{H}_{EST} - \mathcal{H}_{TRUE}$ and

$\mathcal{S}_{EST} - \mathcal{S}_{TRUE}$) of the data shown in Figure 4.8 where each set of residuals is binned into 5 different buckets. Each of the bins have width $\Delta\mathcal{H} = 0.2$ or $\Delta\mathcal{S} = 0.6$. The orange lines in the figure show the median of the residuals in each bucket. Each box marks the lower and upper quartile, $\mathcal{Q}_1$ and $\mathcal{Q}_3$ respectively. Each of the upper whiskers extends to highest point less than $\mathcal{Q}_3 + 1.5\Delta\mathcal{Q}$, where $\Delta\mathcal{Q} = \mathcal{Q}_3 - \mathcal{Q}_1$ is the interquartile range. Each of the lower whiskers extends to the lowest point greater than $\mathcal{Q}_1 - 1.5\Delta\mathcal{Q}$. The points that lie outside the whiskers are plotted individually. The blue line represents a perfect estimator. For CNN:128 we can see that the model overestimates $\mathcal{H}$ when $\mathcal{H} \lesssim 0.5$ and underestimates $\mathcal{H}$ when $\mathcal{H} \gtrsim 0.5$. CNN:128 overestimates $\mathcal{S}$ for $\mathcal{S} \lesssim 2.4$ and underestimates $\mathcal{S}$ when $\mathcal{S} \gtrsim 2.4$. CNN:64 underestimates $\mathcal{H}$ for all values of $\mathcal{H}$, with this underestimation being worse at low $\mathcal{H}$. CNN:64 overestimates $\mathcal{S}$ for $\mathcal{S} \lesssim 2.10$, while underestimates $\mathcal{S}$ when $\mathcal{S} \gtrsim 2.10$. CNN:32 overestimates $\mathcal{H}$, however this overestimation effect is small due to the larger spread of the residuals. CNN:32 overestimates $\mathcal{S}$ when $\mathcal{S} \lesssim 1.80$ and underestimates when $\mathcal{S} \gtrsim 1.80$. We can also see that for $\mathcal{S}$ all three models have roughly the same spread of the residuals, while for $\mathcal{H}$ we see that there is significant difference in the spread of the points in CNN:32 compared to CNN:64 and CNN:128. This degradation in the estimation of $\mathcal{H}$ could denote the lower limit on the number of pixels in an image in order to estimate $\mathcal{H}$.

Figure 4.10 shows a distribution of the residuals that were shown in Figure 4.9, with the mean being shown with a red vertical line. The bottom row shows the residuals for $\mathcal{S}$, while the top row shows the residuals for $\mathcal{H}$. The inset on the left side of each figure shows the first four moments of each distribution. These are the mean (μ_1), the standard deviation (μ_2), the skewness (μ_3), and the kurtosis (μ_4). We can see that the distributions are all roughly Gaussian with a mean of ~ 0. For $\mathcal{S}$ the peak of each distribution is at a higher residual value than the mean which give the distributions a negative skewness. This is consistent with previous finding that each of the models overestimates $\mathcal{S}$. This is most evident for CNN:32 ($\mu_3 = -0.73$) and least evident for CNN:64 ($\mu_3 = -0.28$). The estimation of $\mathcal{H}$ made by CNN:32 shows a slight positive skewness of $\mu_3 = 0.22$, while CNN:128 shows a slight negative skewness of $\mu_3 = -0.18$. CNN:64 shows a negligible skewness of $\mu_3 = 0.09$.

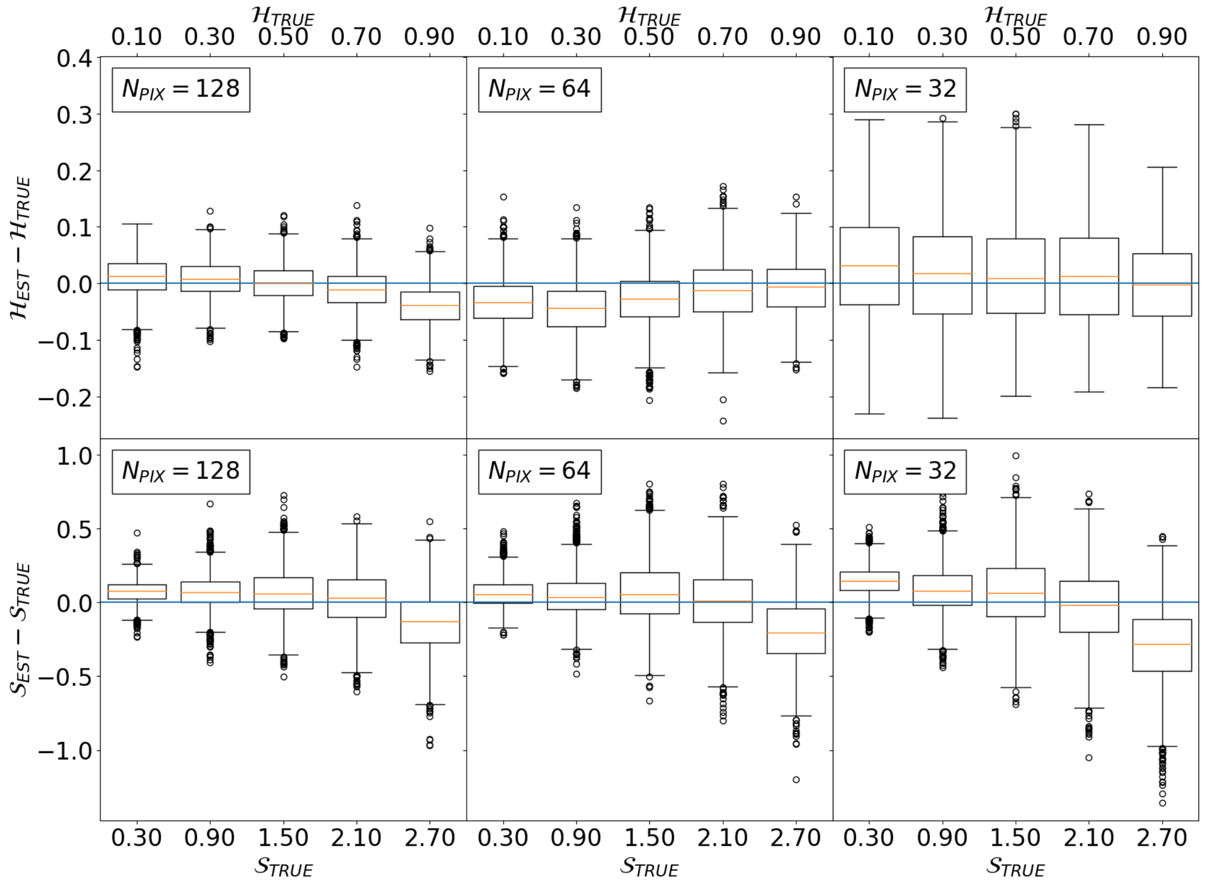


Fig. 4.9 Box-and-whisker plots for the residuals of the the data shown in Figure 4.8. The top row shows $\mathcal{H}_{EST} - \mathcal{H}_{TRUE}$ while the bottom row shows $\mathcal{S}_{EST} - \mathcal{S}_{TRUE}$. The bins in the top row have width $\Delta\mathcal{H} = 0.2$, while the bins in the bottom row have width $\Delta\mathcal{S} = 0.6$. The orange lines show the median of each bin, while the blue line shows a perfect estimator. The box extends to the upper, $\mathcal{Q}_1$, and lower, $\mathcal{Q}_3$, quartiles. The upper whisker extends to the highest point less than $\mathcal{Q}_1 - 1.5\Delta\mathcal{Q}$, while the lower whisker extends to the lowest point greater than $\mathcal{Q}_3 + 1.5\Delta\mathcal{Q}$, where $\Delta\mathcal{Q} = \mathcal{Q}_3 - \mathcal{Q}_1$. All points that lie outside this range are plotted individually.

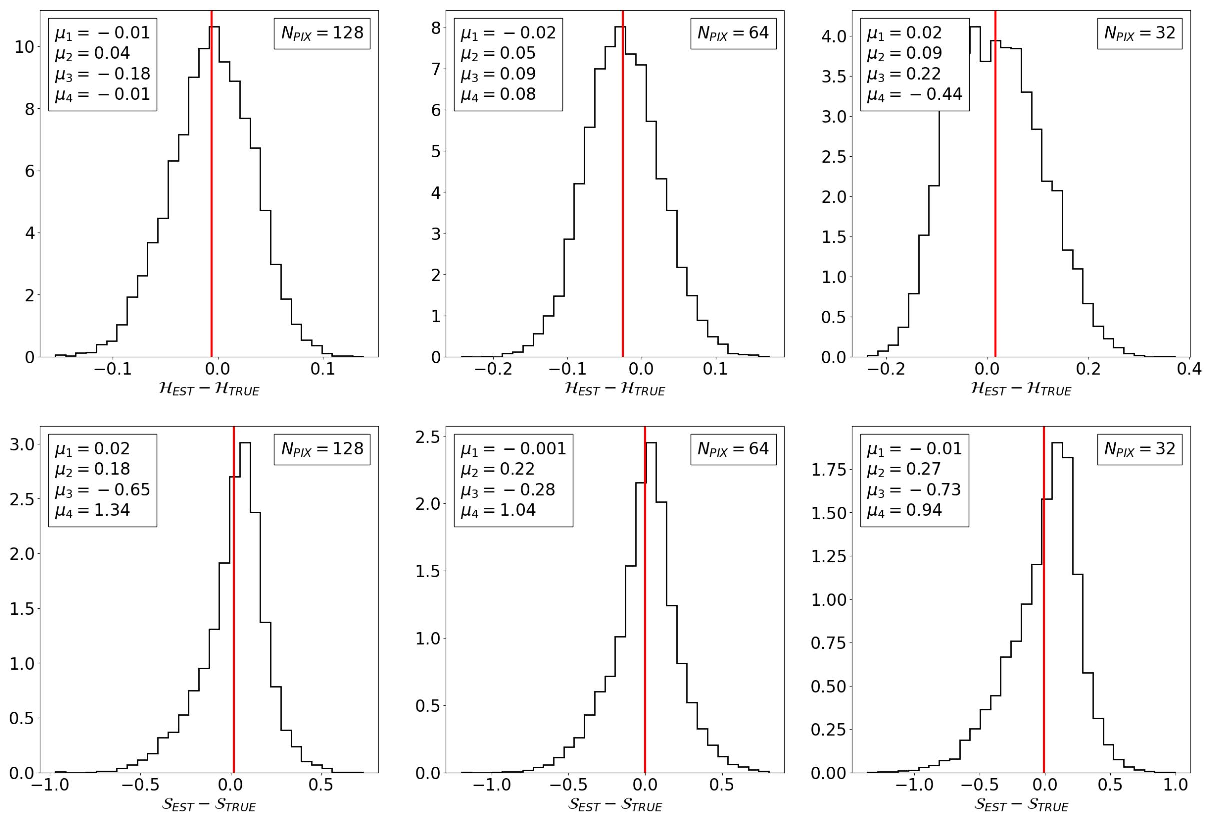


Fig. 4.10 Histograms of the residuals of the estimated values of $\mathcal{H}$ and $\mathcal{S}$ that are shown in Figure 4.8. The top row shows the distribution of the residuals for $\mathcal{H}$ while the bottom row shows the distributions of the residuals for $\mathcal{S}$. The red vertical line shows the mean of the residuals. The inset on the left shows the first four moments of each distribution. These are: the mean (μ_1), the standard deviation (μ_2), the skewness (μ_3), and the kurtosis ($4\mu_4$).

4.5 Application of models

In the next chapter we will apply the saved models to observations of the dust column density of the ISM. We will compare estimations of $\mathcal{H}$ and $\mathcal{S}$ that are returned by CNN:64 and CNN:32.

4.5.1 Caveats and Assumptions

It is important to mention a major caveat with using this method. Each CNN that has been trained on xfBm fields inherently expects that the inputs to the CNN are also xfBm fields. As a result we must mention the requirement that the observed data that is being put through each model is similar to xfBm fields.

CHAPTER 5

Applying CNN Estimator to Observations

5.1 Preamble

In the previous chapter (Chapter 4) we described and trained a set of models that can quickly estimate $\mathcal{H}$ and $\mathcal{S}$ for images with pixel sizes of 32×32, 64×64, and 128×128. In this Chapter we apply these models to real observations of the interstellar medium in the Galactic Plane. We also apply these models to observations of the Andromeda galaxy (M31). We forego the use of CNN:128 and only use CNN:64 and CNN:32. Thus we will only work with patches with pixel sizes of 32×32 and 64×64.

5.2 Via Lactea

The data used to investigate the Galactic Plane is the Via Lactea dataset (Marsh et al., 2017) which is based on observations from the Herschel infrared Galactic Plane survey (Hi-GAL) (Molinari et al., 2010). The initial Hi-GAL survey covers the area of the Galactic plane with galactic latitude $-1^\circ \lesssim b \lesssim 1^\circ$ and longitude $-60^\circ \lesssim \ell \lesssim 60^\circ$, however this was then extended to cover the entire Galactic plane. A notable exception is the region spanning $216.5^\circ \lesssim l \lesssim 225.5^\circ$ where the area surveyed spans $-2^\circ \lesssim b \lesssim 0^\circ$ since the plane of the Galaxy is warped in this region (Elia et al., 2013). The data is organised in 163 tiles each encompassing an $2.4^\circ \times 2.4^\circ$ area of the Galactic plane. Each tile is labelled as ℓ followed by the approximate longitude of its centre. For example tile $\ell 000$ is the tile that contains $l = 0^\circ$ i.e. the Galactic Centre, while the tile that contains the Anti-Centre is tile $\ell 180$ as it contains $l = 180^\circ$.

The Hi-GAL data is collected using the Photodetector Array Camera and Spectrometer (PACS, Poglitsch et al. (2010)) and Spectral and Photometric Imaging REceiver (SPIRE, Griffin et al.

(2010)) instruments found on the Herschel Space Observatory (Pilbratt et al., 2010). Together these instruments produced maps at five wavelengths: the PACS instrument produces maps at 70μm and 160μm, while the SPIRE instrument produces maps at 250μm, 350μm, and 500μm. The PACS instrument has bandwidths of $60 - 85\mu$m and $125 - 210\mu$m (Poglitsch et al., 2010), while the SPIRE instrument has bandwidths such that $\lambda/\Delta\lambda \sim 3$ (Griffin et al., 2010).

The Via Lactea dataset consists of images of surface-density that have been generated by using `PPMAP` (Point Process Mapping) on the Hi-GAL data (Marsh et al., 2017). `PPMAP` is an analytical procedure that is able to produce dust temperature-dependant maps based on multi-wavelength observations (such as Hi-GAL). For a full mathematical description of `PPMAP` we refer the reader to Marsh et al. (2015). `PPMAP` differs from traditional SED fitting in that it does not assume a single temperature along each line of sight, instead it gives the contributions to the dust optical depth at 300μm, $\Delta\tau_{300:q}$, from different temperature intervals around T_q. The sum of each $\Delta\tau_{300:q}$ gives the total dust optical depth at 300μm,

$$\tau_{300} = \sum_{q=1}^{q=Q} \left\{\Delta\tau_{300:q}\right\}, \tag{5.1}$$

where Q is the number of temperature intervals that are used in the PPMAP algorithm.

Table 5.1 shows the 12 temperature intervals that are used to produce the Via Lactea dataset (Marsh et al., 2017). This table shows the values of T_q for each q as well the lower limit, $0.920T_q$, and upper limit, $1.087T_q$ of each interval. We must also note that the temperature interval for T_1, actually represents dust below its listed lower limit of 7.36K, and likewise the temperature interval for T_{12}, actually represents dust above its listed upper limit of 54.3K.

Another advantage of using `PPMAP` over SED fitting is that SED fitting requires that the maps be smoothed to the largest spatial resolution, which for Hi-GAL would be the 36 arcsec resolution of the 500μm band maps. This results in a lot of the information in the high-resolution images being ignored. `PPMAP` however is able to produce the temperature-differential maps to the finest resolution by processing the Hi-GAL images at each bands native resolution. This results in the angular size of the pixels, $\phi_{\text{min}} = 6''$, which is the Hi-GAL resolution at the 70μm band. This angular size corresponds to a linear size of $0.029\text{pc}[D/\text{kpc}]$ for a structure at distance D along the line of sight. We must stress that any structure seen, is likely to be the superimposition of multiple structures that lie along the line

Table 5.1 The dust temperature intervals defined in Marsh et al. (2017) and subsequently used in this thesis and in Bates and Whitworth (2023). For each q there is a discrete dust temperature T_q. This table also shows the lower limit, $0.920T_q$, and the upper limit, $1.087T_q$, of each temperature interval. The temperature interval for T_1 actually represents dust below its listed lower limit of 7.36K, while the temperature interval for T_{12} actually represents dust above its listed upper limit of 54.3K.

q	1	2	3	4	5	6
T_q/K	8.00	9.45	11.2	13.2	15.6	18.4
$> 0.920\,T_q/\mathrm{K}$	'>7.36'	> 8.69	> 10.3	> 12.1	> 14.3	> 16.9
$\leq 1.087\,T_q/\mathrm{K}$	≤ 8.69	≤ 10.3	≤ 12.1	≤ 14.3	≤ 16.9	≤ 20.0

q	7	8	9	10	11	12
T_q/K	21.7	25.7	30.3	35.8	42.3	50.0
$> 0.920\,T_q/\mathrm{K}$	> 20.0	> 23.6	> 27.9	> 33.0	> 38.9	> 46.0
$\leq 1.087\,T_q/\mathrm{K}$	≤ 23.6	≤ 27.9	≤ 33.0	≤ 38.9	≤ 46.0	'≤ 54.3'

of sight, each at a different physical length-scale. This can result with a structure that is multifractal due to the possibility that each structure along the line of sight being described by a different $\mathcal{H}$ and $\mathcal{S}$. For simplicity, in this study we assume that each line of sight can be described by a single monofractal.

We convert the Via Lactea data into units of surface-density ($\mathrm{M_\odot pc^{-2}}$). The `PPMAP` process assumes; (i) that the Hi-GAL data consists of primarily thermal emission of dust, (ii) that this thermal emission is optically thin, and (iii) that this thermal emission follows a mass opacity law of the form

$$\kappa(\lambda) = 0.11\mathrm{cm^2g^{-1}}\left(\frac{\lambda}{300\mu\mathrm{m}}\right)^{-\beta} \tag{5.2}$$

where $\beta = 2$ (Sadavoy et al., 2012; Marsh et al., 2017). The net dust opacity at 300μm is thus

$$\kappa_{300} = 0.11\mathrm{cm^2g^{-1}} = 2.1\times10^{-5}\mathrm{pc^2M_\odot^{-1}}, \tag{5.3}$$

which gives a total surface density of

$$\Sigma = \frac{\tau_{300}}{\kappa_{300}} = [4.8\times10^4\mathrm{M_\odot pc^{-2}}]\tau_{300}, \tag{5.4}$$

and a temperature-differential surface density of

$$\Delta\Sigma_q = [4.8\times10^4\mathrm{M_\odot pc^{-2}}]\Delta\tau_{300:q}. \tag{5.5}$$

Marsh et al. (2017) discuss the validity of assumption (ii) that the dust emission is optically thin. This is valid over most of the Galactic Plane at the Herschel wavebands used here, with the lower wavelengths being the least compatible with this assumption. For 70μm and 160μm, the fractions of pixels for which the optical depth > 1 are $\sim 10^{-4}$ and $\sim 10^{-6}$. These regions of high optical depth are mostly confined to localised regions in dense molecular clouds. Since in this study we are more interested in large scale statistics, this violation of assumption (ii) has a negligible effect on our results. However, it is still important to keep this caveat in mind.

Figure 5.1 shows temperature -differential surface density maps for the Galactic Centre, which consist of the $\ell002$, $\ell000$, and $\ell358$ tiles. Each of the panels show surface density maps for the following dust-temperature intervals:

(*a*). $T \leq$ 12.1K. This is calculated by summing the contributions to the surface densities from $q = [1, 2, 3]$, i.e. $\sum_{q=1}^{3}\{\Delta\Sigma_q\}$.

(*b*). 12.1K $\leq T \leq$ 14.3K. This corresponds to the surface density maps, $\Delta\Sigma_4$, at temperature interval T_4.

(*c*). 14.3K $\leq T \leq$ 16.9K. This corresponds to $\Delta\Sigma_5$, at T_5.

(*d*). 16.9K $\leq T \leq$ 20.0K. This corresponds to $\Delta\Sigma_6$, at T_6.

(*e*). 20.0K $\leq T \leq$ 23.6K. This corresponds to $\Delta\Sigma_7$, at T_7.

(*f*). $T >$ 23.6K. This is calculated by summing the contributions to the surface densities from $q = [8, 9, 10, 11, 12]$, i.e. $\sum_{q=8}^{12}\{\Delta\Sigma_q\}$.

Panels (a) and (b) show a discontinuity between the three tiles. This discontinuity is more pronounced between the $\ell002$ and $\ell000$ tiles. This is a consequence of the $\ell002$ and $\ell358$ having less dust than $\ell000$. This results in a lower signal to noise than in other maps and therefore the estimates of the surface density are uncertain. Panels (c) through (f) have a sufficiently high signal to noise such that there isn't such a pronounced discontinuity. The `PPMAP` algorithm returns `NaN` (Not a Number) values for pixels that do not have enough information to evaluate a surface-density at that temperature interval. These `NaN` values are located mostly at the edge of the tiles but there are a few interior regions that contain these `NaN` values. We can see a few examples of these `NaN` regions in Figure 5.1 mostly in the $\ell002$ tile with the $\ell002$ tile of panel (f) ($T >$ 23.6K) having the largest examples of these regions.

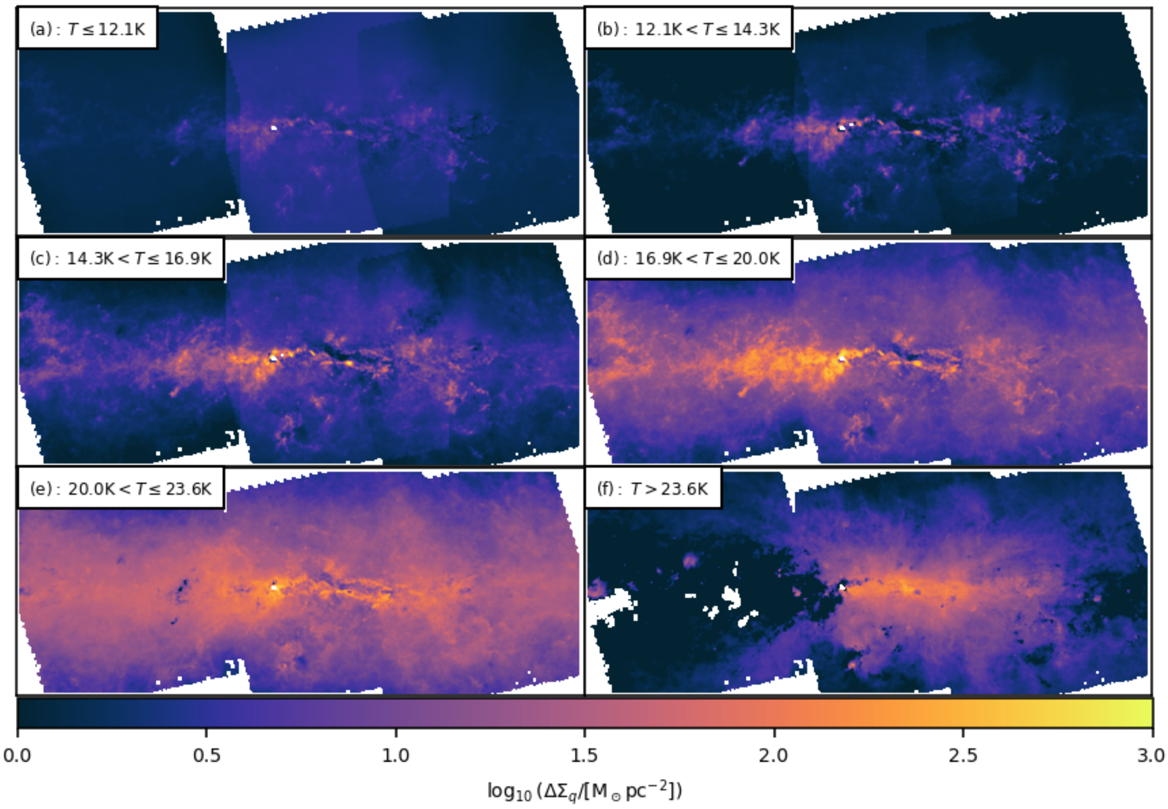

(a) : $T \leq 12.1$K
(b) : $12.1\text{K} < T \leq 14.3$K
(c) : $14.3\text{K} < T \leq 16.9$K
(d) : $16.9\text{K} < T \leq 20.0$K
(e) : $20.0\text{K} < T \leq 23.6$K
(f) : $T > 23.6$K
0.0
0.5
1.0
1.5
2.0
2.5
3.0
$\log_{10}(\Delta\Sigma_q/[\text{M}_\odot\,\text{pc}^{-2}])$

Fig. 5.1 Images of the temperature-differential surface-density, $\Delta\Sigma_q$, for the Hi-GAL tiles ℓ002, ℓ000, and ℓ358, which show an approximately $7° \times 2°$ region surrounding the Galactic Centre. Each panel shows the surface density at a different range of dust temperature: (a) shows $\Delta\Sigma_1 + \Delta\Sigma_2 + \Delta\Sigma_3$ for $T \leq 12.1$K; (b) shows $\Delta\Sigma_4$ for $12.1\text{K} \leq T \leq 14.3$K; (c) shows $\Delta\Sigma_5$ for $14.3\text{K} \leq T \leq 16.9$K; (d) shows $\Delta\Sigma_6$ for $16.9\text{K} \leq T \leq 20.0$K; (e) shows $\Delta\Sigma_7$ for $20.0\text{K} \leq T \leq 23.6$K; and (f) shows $\Delta\Sigma_8 + \Delta\Sigma_9 + \Delta\Sigma_{10} + \Delta\Sigma_{11} + \Delta\Sigma_{12}$ for $T > 23.6$K.

The total column density, Σ, for the same three tiles is shown on Figure 5.2. The total column density for each tile is obtained using Equation 5.4. This total column density does not have any of the discontinuities that are present between the tiles shown in Figure 5.1. The stitching of the tiles was done using the `ReProject` package (Robitaille et al., 2020).

5.3 Estimating $\mathcal{H}$ and $\mathcal{S}$ of the total column density using the CNN models.

For each of the 163 tiles of total surface density, Σ, we create maps of $\mathcal{H}$ and $\mathcal{S}$ using CNN:64 and CNN:32. Each pixel in the $\mathcal{H}$ and $\mathcal{S}$ maps is generated as follows:

1. First a $N_{\text{PIX}} \times N_{\text{PIX}}$ size patch is isolated from the surface density tiles.
2. The patch is then normalised such that the standard deviation of the field is 0.25 and the mean is 0. Each pixel with value, $g > 1$, is set to $g = 1$, and each pixel with value

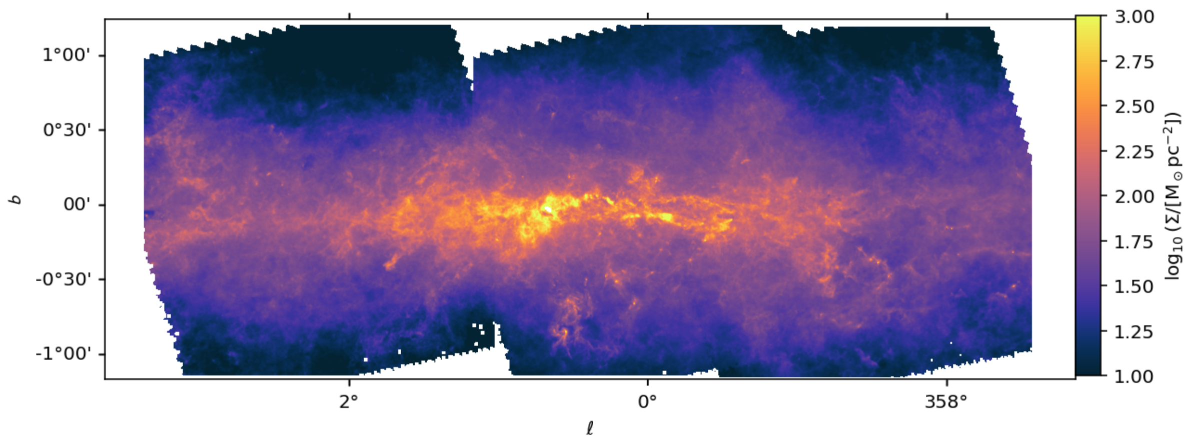


Fig. 5.2 The total surface density, Σ, for the same three tiles shown in Figure 5.1 ($\ell002$, $\ell000$, and $\ell358$). ℓ is the Galactic

Table 5.2 A table showing the parameters that affect the estimations of $\mathcal{H}$ and $\mathcal{S}$.

$\mathcal{H}_{N_{\text{PIX}}\times\phi_{\text{min}}}$	$\mathcal{S}_{N_{\text{PIX}}\times\phi_{\text{min}}}$	N_{PIX}	ϕ_{min}	ϕ_{max}
$\mathcal{H}_{64\times6}$	$\mathcal{S}_{64\times6}$	64	$6''$	$384''$
$\mathcal{H}_{32\times12}$	$\mathcal{S}_{32\times12}$	32	$12''$	$384''$
$\mathcal{H}_{32\times6}$	$\mathcal{S}_{32\times6}$	32	$6''$	$192''$

$g < -1$ is set to $g = -1$. This is the same normalisation process that is done to the training data during the training of the CNNs (See Section 4.4.2). This normalisation ensures that the patches are of the same scale as the training data.

3. The normalised $N_{\text{PIX}} \times N_{\text{PIX}}$ size patch is then inputted into CNN:N_{PIX} and a single value of $\mathcal{H}$ and $\mathcal{S}$ is recorded.

4. The process is repeated for all available patches.

The maps of $\mathcal{H}$ and $\mathcal{S}$ depend on two parameters: the angular size of each pixel, ϕ_{min}; and the patch size N_{PIX}. ϕ_{min} of the raw surface density tiles is $6''$, however we also use rebinned surface density tiles such that $\phi_{\text{min}} = 12''$. This degrades the resolution but increases the signal-to-noise. ϕ_{min} determines the size of the smallest structures that can be resolved, which is $\sim 2\phi_{\text{min}}$. N_{PIX} determines the size of the largest structure that can be resolved, which is $\phi_{\text{max}} = N_{\text{PIX}}\phi_{\text{min}}$. This is because ϕ_{max} is the angular size of each patch.

For each combination of ϕ_{min} and N_{PIX} we denote the resulting values of $\mathcal{H}$ and $\mathcal{S}$ as $\mathcal{H}_{N_{\text{PIX}}\times\phi_{\text{min}}}$ and $\mathcal{S}_{N_{\text{PIX}}\times\phi_{\text{min}}}$. For example using patches of angular pixel size of $\phi_{\text{min}} = 6''$ with CNN:32 produces the $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$ maps. Table 5.2 shows the values of the different parameters for each of the different estimations of $\mathcal{H}$ and $\mathcal{S}$ that are used here.

5.4 $\mathcal{H}$ and $\mathcal{S}$ for the Galactic Centre

Figure 5.3 show the $\mathcal{H}$ and $\mathcal{S}$ maps for the three tiles that are closest to the Galactic Centre ($\ell 002$, $\ell 000$, $\ell 358$). These are the same three tiles that are shown in Figures 5.1 and 5.2. Panels (a) and (b) show $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ respectively, thus capturing the statistics of the substructure at scales between $\phi_{\rm min} \sim 6''$ and $\phi_{\rm max} \sim 384''$. Panels (c) and (d) show $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ which capture the substructure between $\phi_{\rm min} \sim 12''$ and $\phi_{\rm max} \sim 384''$. Panels (e) and (f) show $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$ which capture the substructure between $\phi_{\rm min} \sim 6''$ and $\phi_{\rm max} \sim 192''$. The contours represent lines of constant surface density given at the values of $\log_{10}(\Sigma/[M_\odot \mathrm{pc}^{-2}]) = [1.2, 1.8, 2.4]$.

5.4.1 Comparing maps of $\mathcal{H}$ and $\mathcal{S}$ generated with the same patch size, $\phi_{\rm max}$, but different dynamic range of angular scales, $\phi_{\rm max}/\phi_{\rm min}$.

The mapping that produces $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ (top row of Figure 5.3), uses patches of angular size $\phi_{\rm max} = 384''$. This is the same for the mapping that produces $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ (middle row of Figure 5.3. On the other hand the dynamic range of angular scales of the two mappings are different. $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ have a larger dynamic range ($\phi_{\rm max}/\phi_{\rm min} = 64$) than $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ ($\phi_{\rm max}/\phi_{\rm min} = 32$). Consequently the mapping for $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ does not ignore the substructures between $6''$ and $12''$, while the mapping for $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ does.

There is a morphological similarity between the maps of$\mathcal{H}_{64\times6}$ (panel (a)) and $\mathcal{H}_{32\times12}$ (panel (c)), however we can see that, in general, $\mathcal{H}_{64\times6} > \mathcal{H}_{32\times12}$. This increase in $\mathcal{H}$ implies that there is an amount of substructure between scales $\sim 6''$ and $\sim 12''$ that systematically steepens the power spectrum (See Chapter 2).

There is also a morphological similarity between the maps of $\mathcal{S}_{64\times6}$ (panel (b)) and $\mathcal{S}_{32\times12}$ (panel (d)) where we also see that, in general, $\mathcal{S}_{64\times6} > \mathcal{S}_{32\times12}$. This implies that the substructures that exist between $\sim 6''$ and $\sim 12''$ have a higher surface-density than the more extended substructures present in the map. This is expected if the small-scale substructures tend to be nested within the larger ones.

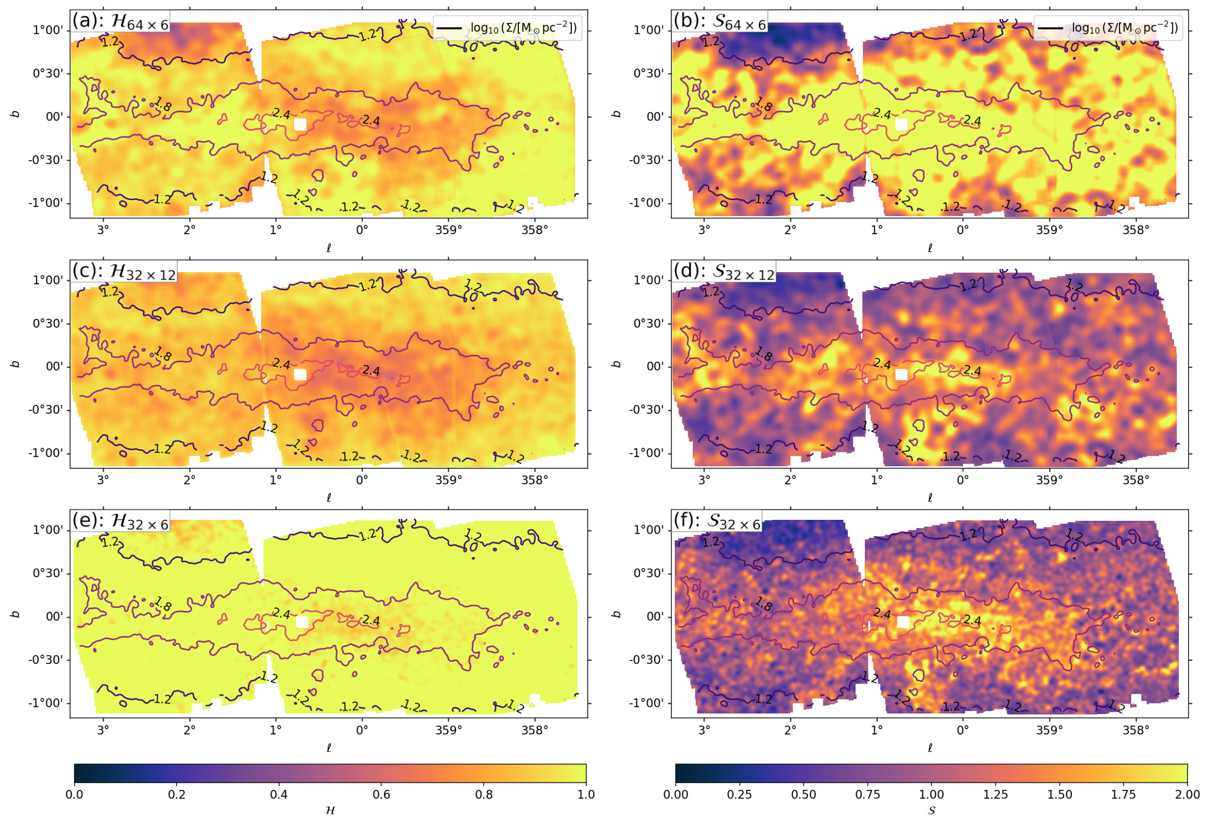

(a): $\mathcal{H}_{64\times6}$
(b): $\mathcal{S}_{64\times6}$
(c): $\mathcal{H}_{32\times12}$
(d): $\mathcal{S}_{32\times12}$
(e): $\mathcal{H}_{32\times6}$
(f): $\mathcal{S}_{32\times6}$
$\log_{10}(\Sigma/[M_\odot pc^{-2}])$
1.2
1.8
2.4
1°00'
0°30'
00'
-0°30'
-1°00'
3°
2°
1°
0°
359°
358°
b
ℓ
$\mathcal{H}$
$\mathcal{S}$

Fig. 5.3 $\mathcal{H}$ and $\mathcal{S}$ maps for the three tiles closest to the Galactic Centre (ℓ002, ℓ000, ℓ358). The left column shows the different maps of $\mathcal{H}$, while the right column shows the different maps of $\mathcal{S}$. The top row (panels (a) and (b)) shows the values generated by applying CNN:64 to patches of resolution $\phi_{\rm min} = 6''$ and angular size $\phi_{\rm max} = 384''$. These patches have a pixel size of 64×64. The middle row (panels (c) and (d)) shows the values generated by applying CNN:32 to patches of resolution $\phi_{\rm min} = 12''$ and angular size $\phi_{\rm max} = 384''$. These patches originally have a pixel size of 64×64 but have been rebinned to 32×32 in order to pass them through CNN:32. The bottom row (panels (e) and (f)) shows the values generated by applying CNN:32 to patches of resolution $\phi_{\rm min} = 6''$ and angular size $\phi_{\rm max} = 192''$. The patches have a pixel size of 32×32. The contours show lines of constant surface density at values of $\log_{10}(\Sigma/[M_\odot {\rm pc}^{-2}]) = [1.2, 1.8, 2.4]$.

5.4.2 Comparing maps of $\mathcal{H}$ and $\mathcal{S}$ generated with the same dynamic range of angular scales, $\phi_{\rm max}/\phi_{\rm min}$, but different patch size, $\phi_{\rm max}$.

In the previous section (Section 5.4.1) we compared $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ (middle row of Figure 5.3) with $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ (top row of Figure 5.3). Here we compare $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ (middle row of Figure 5.3) with $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$ (bottom row of Figure 5.3). $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ have the same dynamic range of angular scales as $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$, which is $\phi_{\rm max}/\phi_{\rm min} = 32$. The maps of $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$ have been generated using a smaller patch size ($\phi_{\rm max} = 192''$) compared to the patch size used to generate maps of $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ ($\phi_{\rm max} = 384''$). The mapping for $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$ takes into account the substructures between $6''$ and $12''$, while the mapping for $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ ignores them. The mapping for $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$, however, ignores the substructures between $192''$ and $384''$ which are taken into account when estimating $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$.

The maps of $\mathcal{H}_{32\times12}$ (panel (c)) and $\mathcal{H}_{32\times6}$ (panel (e)) have similar morphology but $\mathcal{H}_{32\times6}$ is generally larger than $\mathcal{H}_{32\times12}$, with $\mathcal{H}_{32\times6}$ having values mostly in the interval $[0.8, 1.0]$, and $\mathcal{H}_{32\times12}$ having values in the interval $[0.6, 0.8]$. This is due to $\mathcal{H}_{32\times6}$ having a steeper power spectrum.

The maps of $\mathcal{S}_{32\times12}$ and $\mathcal{S}_{32\times6}$ are also morphologically similar. We can see clearly that the map of $\mathcal{S}_{32\times6}$ has more well-defined structure, which is due to the smaller angular size of the patches ($\sim 192''$) as opposed to the larger patch size of $\mathcal{S}_{32\times12}$ ($\sim 384''$). If we were to convolve the $\mathcal{S}_{32\times6}$ map with a circular beam of width $\sim 384''$, the resulting map would look very similar to $\mathcal{S}_{32\times12}$.

5.5 $\mathcal{H}$ and $\mathcal{S}$ for the Whole Galactic Plane.

5.5.1 The Distributions of $\mathcal{H}$ and $\mathcal{S}$.

Figure 5.4 shows $\mathcal{H}$ and $\mathcal{S}$ compared to each other as well as their distributions. The top row shows the distributions of: (a) $\mathcal{H}_{32\times6}$; (b) $\mathcal{H}_{32\times12}$; (c) $\mathcal{H}_{64\times6}$. The middle row shows the distributions of: (d) $\mathcal{S}_{32\times6}$; (e) $\mathcal{S}_{32\times12}$; (f) $\mathcal{S}_{64\times6}$. The bottom row shows two-dimensional distributions of: (g) $\mathcal{H}_{32\times6}$ against $\mathcal{S}_{32\times6}$; (h) $\mathcal{H}_{32\times12}$ against $\mathcal{S}_{32\times12}$; (i) $\mathcal{H}_{64\times6}$ against $\mathcal{S}_{64\times6}$.

For $\mathcal{H}_{32\times6}$ (panel (a)) and $\mathcal{S}_{32\times6}$ (panel (d)), the angular scales between $6''$ and $192''$ are considered whilst the angular scales between $192''$ and $384''$ are not. This means that the power spectrum is biased towards larger wavenumbers and is thus steeper. This gives a larger $\mathcal{H}$. This is shown in panel (a) where $\mathcal{H} \sim 1$. Panel (d) shows that $\mathcal{S}_{32\times6}$ is relatively small ($\mathcal{S} \sim 0.9$). This is a result of the relatively small dynamic range ($\phi_{\text{max}}/\phi_{\text{min}} = 32$).

For $\mathcal{H}_{32\times12}$ (panel (b)) and $\mathcal{S}_{32\times12}$ (panel (e)), the angular scales between $12''$ and $384''$ are considered whilst the angular scales between $6''$ and $12''$ are not. This means that the power spectrum is biased towards the smaller wavenumbers and is thus shallower. This gives a smaller $\mathcal{H}$. Panel (b) shows this where we see that in general $\mathcal{H}_{32\times12} < \mathcal{H}_{32\times6}$. Panel (e) shows that, similarly to $\mathcal{S}_{32\times6}$ in panel (d), $\mathcal{S}_{32\times12}$ is also small with most of the values of $\mathcal{S} \lesssim 0.9$. This is not surprising since the dynamic range is also relatively small ($\phi_{\text{max}}/\phi_{\text{min}} = 32$). The main difference we see between $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$, and $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$ is the emergence of a bimodality in the distributions of $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$. Panel (h) suggests this bimodality has peaks at $(\mathcal{H}_{32\times12}, \mathcal{S}_{32\times12}) \sim (0.9, 0.9)$ and $(\mathcal{H}_{32\times12}, \mathcal{S}_{32\times12}) \sim (0.6, 0.4)$.

For $\mathcal{H}_{64\times6}$ (panel (c)) and $\mathcal{S}_{64\times6}$ (panel (f)), the angular scales between $6''$ and $384''$ are considered. This means that the power spectrum has contributions from both the large and small wavenumbers. This is reflected in the broader distribution of $\mathcal{H}_{64\times6}$ compared to those of $\mathcal{H}_{32\times12}$ and $\mathcal{H}_{32\times6}$. There is also a large dynamic range ($\phi_{\text{max}}/\phi_{\text{min}} = 64$) which gives a broad distribution of $\mathcal{S}$. Similar to $\mathcal{H}_{32\times12}$ and $\mathcal{S}_{32\times12}$, $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ show a bimodality.

We split this bimodality into two modes which we call the 'monolithic mode' and the 'undulating mode'. The monolithic mode represents lines of sight with large angular scales being dominant, which give a large $\mathcal{H} \gtrsim 0.8$, and with large scaling parameter, which gives large $\mathcal{S} \gtrsim 1$. The undulating mode represents lines of sight with a broader range of angular scales, giving $\mathcal{H} \lesssim 0.8$, and lower scaling parameter, which gives $\mathcal{S} \lesssim 1$). When CNN:32 is operated on patches with $\phi_{\text{min}} = 6''$ and $\phi_{\text{max}} = 192''$ the distinction between the two lines of sight is weak since the small dynamic range ($\phi_{\text{max}}/\phi_{\text{min}} = 32$) does not capture the large range of angular scales of the undulating mode, while the small ϕ_{max}, does not capture the large angular scale of the monolithic mode. Operating CNN:32 on patches with larger angular scales i.e. $\phi_{\text{min}} = 12''$, and $\phi_{\text{max}} = 384''$, causes the monolithic mode to be better captured, which results in the bimodality starting to emerge. With CNN:64 we see that, in addition to the monolithic mode being captured (due to $\phi_{\text{max}} = 384''$), the undulating mode is being captured too. This is due to the larger dynamic range ($\phi_{\text{max}}/\phi_{\text{min}} = 64$). This causes an even clearer bimodality.

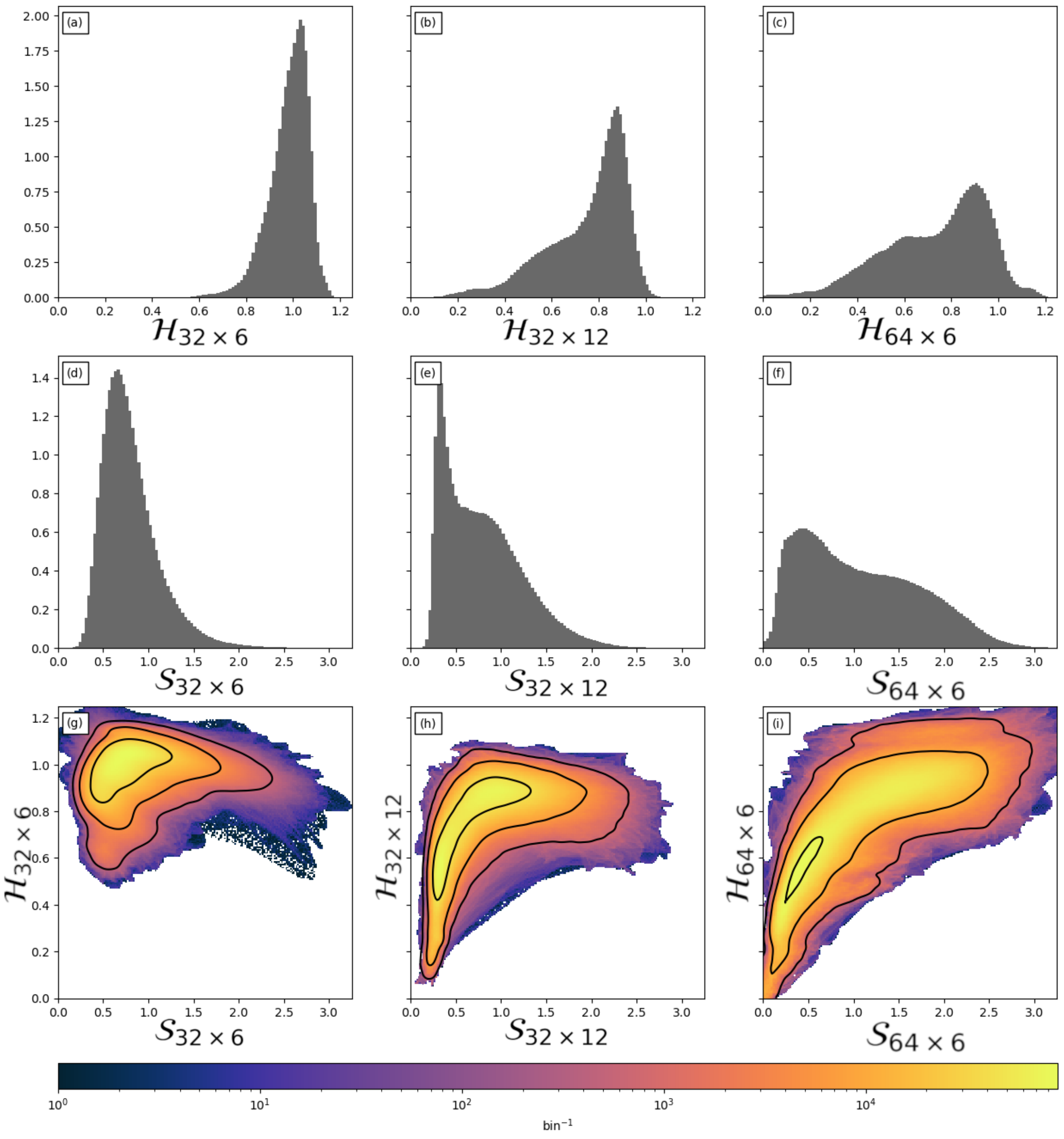


Fig. 5.4 *Top:* Distributions of: (a) $\mathcal{H}_{32\times6}$; (b) $\mathcal{H}_{32\times12}$; and (c) $\mathcal{H}_{64\times6}$. *Middle:* Distributions of: (d) $\mathcal{S}_{32\times6}$; (e) $\mathcal{S}_{32\times12}$; (f) $\mathcal{S}_{64\times6}$. *Bottom:* Two-dimensional distributions of: (g) $\mathcal{H}_{32\times6}$ against $\mathcal{S}_{32\times6}$; (h) $\mathcal{H}_{32\times12}$ against $\mathcal{S}_{32\times12}$; (i) $\mathcal{H}_{64\times6}$ against $\mathcal{S}_{64\times6}$. The ordinate axes for the distributions in panels (a) to (f) have been divided by 10^7. The bins in panels (a) to (c) have size $\Delta\mathcal{H} = 6.70 \times 10^{-3}$. The bins in panels (d) to (f) have size $\Delta\mathcal{S} = 1.80 \times 10^{-2}$. The contours in panels (g) to (i) are at values of 10^3, 10^4, and 7×10^4.

5.5.2 $\mathcal{H}$ and $\mathcal{S}$ compared to the surface density.

Figure 5.5 shows two-dimensional distributions of: (a) $\mathcal{H}_{32\times6}$, (b) $\mathcal{H}_{32\times12}$, and (c) $\mathcal{H}_{64\times6}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$ on the top row; and (d) $\mathcal{S}_{32\times6}$, (e) $\mathcal{S}_{32\times12}$, and (f) $\mathcal{S}_{64\times6}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$ on the bottom row, where Σ is the surface density in units of $\mathrm{M_\odot\ pc^{-2}}$. We can see that $\mathcal{H}$ and $\mathcal{S}$ tend to increase with surface density, with this relationship being the clearest for $\mathcal{H}_{64\times6}$ and $\mathcal{S}_{64\times6}$ mapping and the weakest for $\mathcal{H}_{32\times6}$ and $\mathcal{S}_{32\times6}$. The strength of this correlation mirrors the emergence of the bimodality in the distributions seen in the previous section (Section 5.5.1). For example, the $(\phi_{\text{min}}, \phi_{\text{max}}) = (6'', 384'')$ mapping shows the strongest correlation with surface density and the strongest bimodality. For pixels that have a large surface-density, the structure surrounding that pixel has a steep power spectrum (which results in a large $\mathcal{H}$) and a large variation of the surface density (which results in a large $\mathcal{S}$). Thus these large surface-density pixels correspond to the monolithic mode, while the low surface-density pixels correspond to the undulating mode. These two modes are clearly seen in Figure 5.5, particularly in panels (b) and (c). These two regimes meet at approximately $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}]) = 1.5$, which corresponds to $\Sigma \approx 32\ \mathrm{M_\odot pc^{-2}}$. Therefore we can say that the monolithic mode corresponds to high surface density, $\Sigma \gtrsim 32\ \mathrm{M_\odot pc^{-2}}$, and the undulating mode corresponds to low surface density, $\Sigma \lesssim 32\ \mathrm{M_\odot pc^{-2}}$.

The monolithic mode of $\Sigma \gtrsim 32\ \mathrm{M_\odot pc^{-2}}$ corresponds to a visual extinction of $A_V \gtrsim 2.1$ mag. Numerous studies have shown that there is a threshold of visual extinction above which self-gravity plays an important role in molecular clouds, with the most recent studies revising this threshold to $A_V \sim 2.5$ mag (Kainulainen et al., 2009; Girichidis et al., 2014; Schneider et al., 2015, 2022). Our threshold value of $A_V \sim 2.1$ mag is only slightly lower than the threshold found in the literature, and therefore there is some degree of agreement. This suggests that the monolithic mode is equivalent to this visual extinction regime, representing regions where self-gravity is important, and are thus undergoing star formation. These regions have surface-density distributions that have a power-law tail at higher surface densities. On the other hand the undulating mode describes the more turbulent regions that exist at the lower surface densities. These turbulent regions produce a log-normal distribution of surface-density.

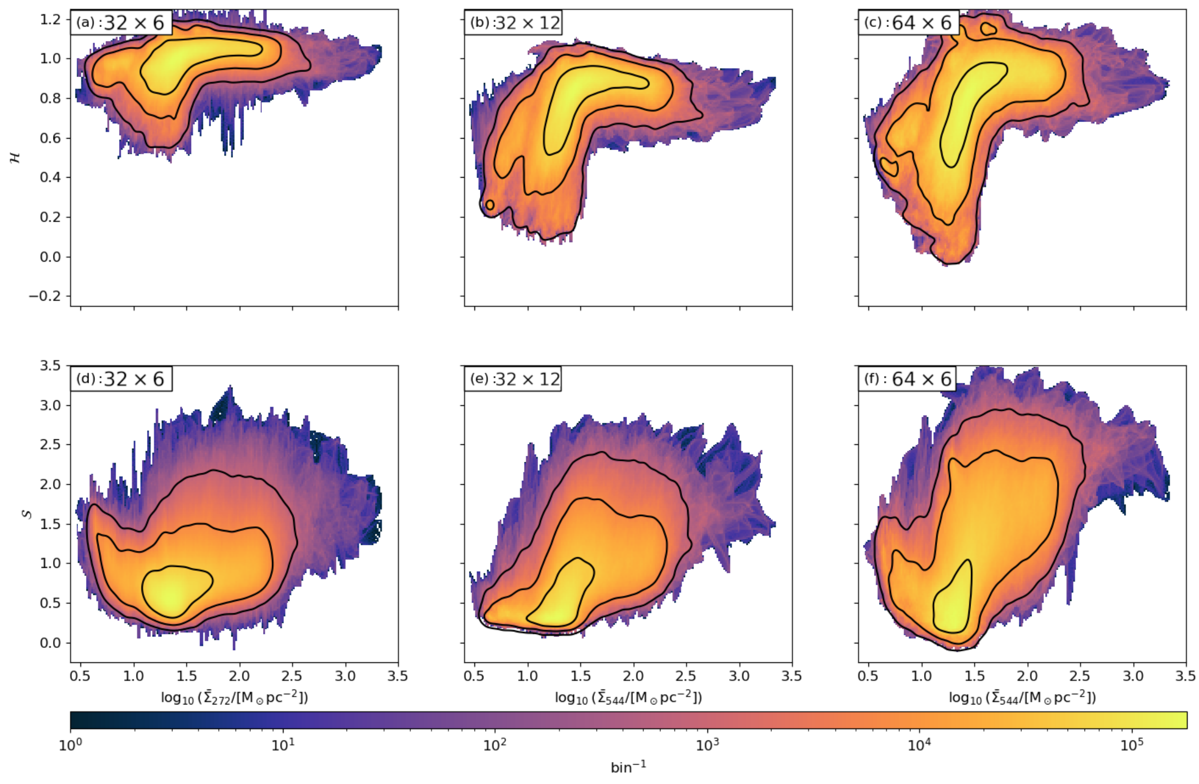


Fig. 5.5 Two-dimensional distributions of $\mathcal{H}$ and $\mathcal{S}$ against the surface density, Σ. *Top row:* Distributions of (a) $\mathcal{H}_{32\times6}$, (b) $\mathcal{H}_{32\times12}$, and (c) $\mathcal{H}_{64\times6}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$. *Bottom row:* Distributions of (d) $\mathcal{S}_{32\times6}$, (e) $\mathcal{S}_{32\times12}$, and (f) $\mathcal{S}_{64\times6}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$. The bin sizes for each variable are: $\Delta\mathcal{H} = 6.70 \times 10^{-3}$, $\Delta\mathcal{S} = 1.80 \times 10^{-2}$, and $\Delta \log_{10}(\Sigma) = 1.44 \times 10^{-2}$. The contours are at values of 10^3, 10^4, and 7×10^4.

5.5.3 $\mathcal{H}$ and $\mathcal{S}$ compared to Galactic longitude.

Figure 5.6 shows two-dimensional distributions of: (a) $\mathcal{H}_{32\times6}$, (b) $\mathcal{H}_{32\times12}$, and (c) $\mathcal{H}_{64\times6}$ against Galactic longitude, ℓ, on the top row; (d) $\mathcal{S}_{32\times6}$, (e) $\mathcal{S}_{32\times12}$, and (f) $\mathcal{S}_{64\times6}$ against ℓ on the middle row; and (g) the surface density, $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$ against ℓ on the bottom row. Each of the black dots show the mean values for each of the 163 tiles i.e. the black dots in the top row show the mean of $\mathcal{H}$, the black dots in the middle row show the mean $\mathcal{S}$, and the black dots in the bottom panel show the mean of $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$. The distributions have been generated such that there are 163 bins (equal to the number of Hi-GAL tiles). This sets the bin sizes to $\Delta\mathcal{H} = 9.2 \times 10^{-3}$, $\Delta\mathcal{S} = 2.3 \times 10^{-2}$, $\Delta \log_{10}\Sigma = 1.9 \times 10^{-2}$ $\mathrm{M}_\odot\mathrm{pc}^{-2}$, and $\Delta\ell = 2.21°$. The red vertical dashed lines in panel (g) represent the longitudes of the tangent lines to the following spiral arms of the Galaxy: the Scutum Arm ($\ell \sim 32°$); the Sagittarius Arm ($\ell \sim 51°$); the Carina Arm ($\ell \sim 282°$); the Centaurus Arm ($\ell \sim 308°$); and the Norma Arm ($\ell \sim 327°$) (Hou and Han, 2014, 2015). There are some features at these longitudes however there are similar features at other longitudes where there are no arms, such as at $\ell \sim 80°$ and at $\ell \sim 170°$.

The central parts of the Galaxy are generally more dense than the outer parts. This is seen quite clearly in Figure 5.6g, where the outer region ($90° \lesssim \ell \lesssim 270°$) has a lower surface density than the rest of the Galactic plane (i.e. towards the Galactic centre). The monolithic mode (large values of $\mathcal{H}$ and $\mathcal{S}$) represents dense regions (Section 5.5.2) and as a result we see from panels (a) to (f) that the monolithic mode is mostly present in the central regions of the Galaxy while the undulating mode (smaller values of $\mathcal{H}$ and $\mathcal{S}$) is mostly present in the outer regions of the Galaxy. Similar to previous findings we see that the two different modes are most prevalent for $\mathcal{H}_{64\times6}$ (panel (c)) and $\mathcal{S}_{64\times6}$ (panel (f)), and least prevalent for $\mathcal{H}_{32\times6}$ (panel (a)) and $\mathcal{S}_{32\times6}$ (panel (d)).

5.5.4 Temperature Differential Surface Densities

In Section 5.2 we discussed the temperature-differential surface density fields that the `PPMAP` process produces as well as how these were used to generate high-resolution maps of surface density. In Figure 5.1 we showed examples of these temperature-differential surface-density maps for the three tiles closest to the Galactic centre (ℓ002, ℓ000, and ℓ358). Figure 5.7 shows distributions of the temperature surface density maps for all 163 tiles for different dust

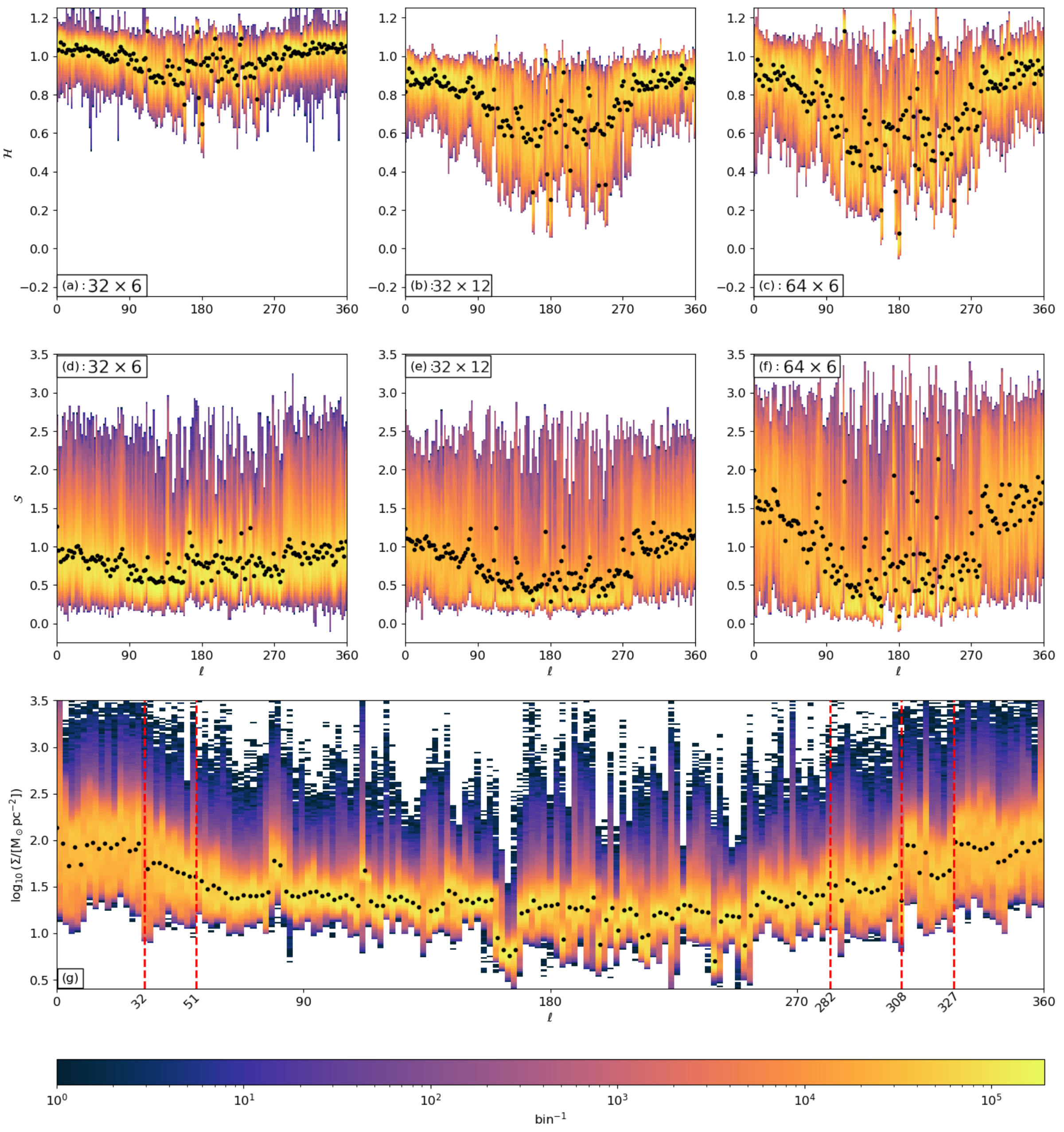


Fig. 5.6 Two-dimensional distributions of: *Top:* (a) $\mathcal{H}_{32\times6}$, (b) $\mathcal{H}_{32\times12}$, and (c) $\mathcal{H}_{64\times6}$ against Galactic longitude, ℓ; *Middle:* (d) $\mathcal{S}_{32\times6}$, (e) $\mathcal{S}_{32\times12}$, and (f) $\mathcal{S}_{64\times6}$ against ℓ; *Bottom:* (g) the surface density, $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$ against ℓ. Here, ℓ is measured in degrees, with $0°$ being the Galactic centre. For each of the Hi-GAL tiles we produce the mean of $\mathcal{H}$, $\mathcal{S}$ and $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$ for that tile. This is shown by the black dots. There are 163 bins for ℓ, with each bin roughly corresponding to each Hi-GAL tile. Each of the red dashed vertical lines in panel (g) represent the longitudes of the tangent lines to the spiral arms of the Galaxy (Hou and Han, 2014, 2015). These are (from left to right): the Scutum Arm ($\ell \sim 32°$); the Sagittarius Arm ($\ell \sim 51°$); the Carina Arm ($\ell \sim 282°$); the Centaurus Arm ($\ell \sim 308°$); and the Norma Arm ($\ell \sim 327°$). The bin sizes are $\Delta\mathcal{H} = 9.2 \times 10^{-3}$, $\Delta\mathcal{S} = 2.3 \times 10^{-2}$, $\Delta\log_{10}\Sigma = 1.9 \times 10^{-2}$, and $\Delta\ell = 2.21°$.

temperature and different total surface density thresholds. The black histograms show the distributions of the surface-density for all dust-temperatures. The blue histograms show the distributions of the surface density for dust that has temperature above a separatrix-temperature, T_S, while the red histograms show the distribution of the surface-density for dust that has temperature below T_S. This separatrix-temperature is a way to explore whether or not the two structural modes, (i.e. the monolithic or undulating modes), are associated with different temperature intervals, given that the monolithic mode seems to be correlated with higher surface densities than the undulating mode. Each column of panels in Figure 5.7 shows the distributions for different separatrix temperatures. These are, from left to right, $T_S = 12.1$ K, 14.3 K, 16.9 K, 20.0 K, and 23.6 K.

The top row of Figure 5.7 shows the distributions for all the available pixels. The number of pixels here are $\sim 2 \times 10^8$. The bottom two rows show the distributions for the surface density where the only pixels considered are ones that have a total surface-density, Σ, above a certain threshold. For the middle row this threshold is $\Sigma > 23$ M$_\odot$pc^{-2}. This threshold corresponds to a visual extinction of $A_V \gtrsim 1$ mag. The number of pixels that satisfy this threshold is $\sim 5 \times 10^7$. For the bottom row this threshold is $\Sigma > 180$ M$_\odot$pc^{-2}, which corresponds to a visual extinction of $A_V \gtrsim 8$ mag. This surface-density threshold is chosen as it is the surface-density above which we observe high rates of low-mass star formation (Johnstone et al., 2004; Könyves et al., 2015; Marsh et al., 2016; Ladjelate et al., 2020; Howard et al., 2021). This total surface threshold is visible in the bottom two rows as the vertical cutoff on the black histograms.

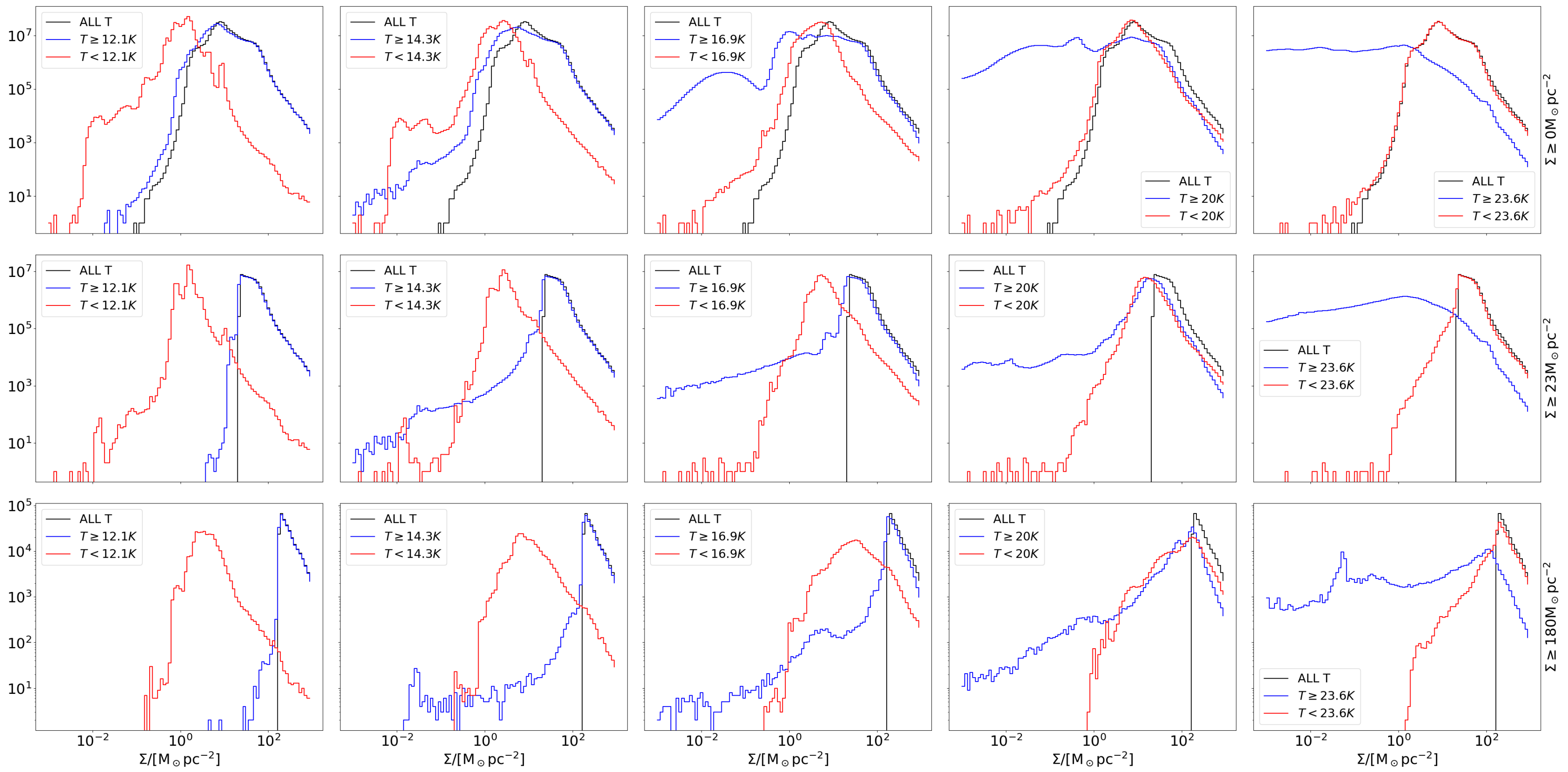


Fig. 5.7 Panels showing different distributions for the surface-density. Each panel shows three histograms. The black histograms show the distributions of the surface-density at all dust-temperatures. The blue histograms show the distributions of the surface density of the dust above a separatrix-temperature, T_S. The red histograms show the distributions of the surface-density of the dust below T_S. Each column of panels shows a different T_S. From left to right the separatrix temperature has values $T_S = 12.1$ K, 14.3 K, 16.9 K, 20.0 K, and 23.6 K. Each row of panels shows the distributions for pixels at different total surface-density, Σ, thresholds. In the top row are histograms for all pixels ($\sim 2 \times 10^8$ pixels). The middle row shows histograms where only the pixels with $\Sigma > 23 M_\odot pc^{-2}$ have been considered ($\sim 5 \times 10^7$ pixels). The bottom row shows histograms where only the pixels with $\Sigma > 180\ M_\odot pc^{-2}$ have been considered ($\sim 3 \times 10^5$ pixels).

It is important to note that the blue and red histograms in each panel do not necessarily sum to the black histogram. This is because for each pixel, the total surface density, Σ, is the sum of both the surface density of the dust at $T < T_S$ and the surface density of the dust at $T > T_S$. As a result these two surface densities are individually less than the total surface density and must thus occupy lower (i.e. different) surface density bins in their respective histograms. This also means that the blue and red histograms are not necessarily confined to be below the black histogram. This can easily be seen in the bottom two rows of Figure 5.7 where the blue and red histograms extend to surface-density values that are less than the vertical cutoff of the black histograms.

For all plots we see that, above a particular value of surface-density, all the histograms follow a power law. This power law holds for both the cooler and warmer dust temperatures and takes the form

$$\frac{dP}{d\Sigma} \propto \frac{1}{\Sigma^3}. \tag{5.6}$$

In the top row of Figure 5.7 we see that at low surface-density the histograms roughly follow a log-normal distribution. This only shows clearly in the top row as once the surface-density cutoff is applied, the lower surface-density regime is not adequately sampled. This suggests that neither the monolithic mode or undulating mode are associated with a particular temperature regime.

5.6 Measuring $\mathcal{H}$ and $\mathcal{S}$ for M31, the Andromeda Galaxy.

In this section we describe results from applying CNN:32 to the Andromeda Galaxy (M31). The map of column density used here is one analysed by Whitworth et al. (2019). In Whitworth et al. (2019), observations from *Herschel* are analysed by the `PPMAP` algorithm in order to provide a high physical resolution map of optical depth with a pixel resolution of $4''$. Since M31 is at a distance of ~ 0.78 Mpc, this arcsecond resolution corresponds to a spatial resolution of ~ 15 pc across each pixel. The pixel size of the map is 2250×500, which corresponds to an area of $2.50^\circ \times 0.56^\circ$. This map of optical depth is then converted to a column density in units of $\mathrm{M_\odot pc^{-2}}$ using a modified version of Equation 5.4,

$$\Sigma \cos(i) = \frac{\tau_{300}}{\kappa_{300}}, \tag{5.7}$$

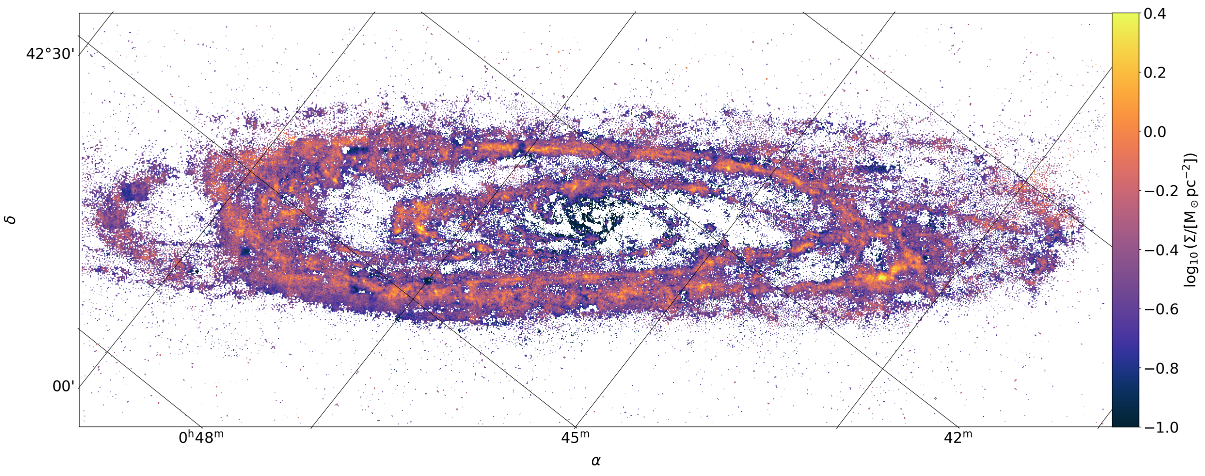


Fig. 5.8 A map of the column density of the Andromeda given in $\mathrm{M}_{\odot}\mathrm{pc}^{-2}$. The grid shows the equatorial coordinate system, given in terms of the right ascension, α, and the declination, δ. The data used here is from Whitworth et al. (2019)

where we have replaced the column density, Σ, with its projection, $\Sigma\cos(i)$. This is done to account for the inclination of M31. Here, i is the inclination angle, where $i = 0°$ is for a face on galaxy and $i = 90°$ is for an edge on galaxy. For the Andromeda Galaxy $i = 77.7°$. For the mass opacity index of dust at 300 μm, we use $\kappa_{300} = 2.7\ \mathrm{cm}^2\ \mathrm{g}^{-1} = 5.6 \times 10^{-4}\ \mathrm{pc}^2\ \mathrm{M}_{\odot}^{-1}$ from Whitworth et al. (2019). This gives

$$\Sigma = \frac{\tau_{300}}{\kappa_{300}\cos(i)} = [8.4 \times 10^3\ \mathrm{M}_{\odot}\mathrm{pc}^{-2}]\tau_{300}. \tag{5.8}$$

Figure 5.8 shows the resulting column density map of M31 given in the units of $\mathrm{M}_{\odot}\mathrm{pc}^{-2}$. We overlay a grid that shows the equatorial coordinate system given in terms of right ascension, α, and declination, δ. In this image we can clearly see the three rings of Andromeda with the middle ring clearly contributing the bulk of the disk as well as having the regions with the highest column density. It is important to note that this map contains far more NaN (Not-a-Number) pixels than the Hi-GAL tiles used in the previous Section, with approximately 65% of the pixels in the M31 image being NaN pixels. Moreover in the Hi-GAL tiles the NaN regions were mostly confined to the outside of the maps with minimal large NaN regions within each tiles. In contrast the M31 map contains quite large NaN regions interspersed within the disk of the galaxy, with multiple regions of column density data completely separated from the main bulk of the disk. This necessitates a different treatment of these NaN values compared to the treatment of the NaNs in the Hi-GAL tiles.

The CNN architecture used for CNN:32 has the consequence of giving values of NaN as the output if even a single NaN pixel is present in the input map. As a result of this, the equivalent NaN region in the resultant map (of $\mathcal{H}$ and $\mathcal{S}$) is expanded by at least an additional $N_{\text{PIX}}/2$ where N_{PIX} is the pixel size of the CNN patches (See Section 5.3). Applying this treatment to the map of M31 would lead to maps of $\mathcal{H}$ and $\mathcal{S}$ with even larger regions of NaN values compared to the already large NaN regions in the column density map. Thus the process of applying CNN:32 is modified by first replacing every NaN pixel with a zero. These *replaced pixels* ensure that CNN:32 returns numerical values for $\mathcal{H}$ and $\mathcal{S}$ in place of NaNs. The column density map is then split into $N_{\text{PIX}} \times N_{\text{PIX}}$ patches and each of these patches are normalised and passed through CNN:32. The normalisation process is the same normalisation process that was done for the patches in the Hi-GAL tiles (See Section 5.3). For each pair of $\mathcal{H}$ and $\mathcal{S}$ values generated for each patch an efficiency value is calculated by dividing the number of replaced pixels, $N_{\text{RP, i}}$, in patch i by the total number of pixels in each patch $(N_{\text{PIX}})^2$, i.e. $(N_{\text{RP, i}})/(N_{\text{PIX}})^2$. The resulting $\mathcal{H}$ and $\mathcal{S}$ maps contain values of $\mathcal{H}$ and $\mathcal{S}$ where the efficiency value is greater than a particular threshold value. Here we set the threshold value to be 0.2. This means that we only consider patches where less than 20% of pixels were originally NaN-values.

In this analysis we set $N_{\text{PIX}} = 32$. Since the pixel resolution is $4''$, then $\phi_{\text{min}} = 4''$ and $\phi_{\text{max}} = 128''$. In terms of spatial distances these become $r_{\text{min}} = 15$ pc and $r_{\text{max}} = 240$ pc respectively. It is straightforward to determine these spatial resolutions as the Andromeda Galaxy, and hence the material intersecting most of the lines of sight, is at a constant distance of $D \approx 0.78$ Mpc. This is in contrast to the analysis that was done on the Milky Way Plane as in that case it is highly likely that the column density contains contributions from material that is at different distances along the line of sight.

Figure 5.9 shows the resulting map of $\mathcal{H}$. We can see that the centre of the galaxy tends to have a low $\mathcal{H}$ compared to the rings, while the dense regions in the rings have a high $\mathcal{H}$. The region with the highest $\mathcal{H}$ is the region at $(\alpha \sim 0^h46^m, \delta \sim 42°10')$ which is within the outermost ring. This is consistent with a power spectrum dominated by large wavelengths. The regions with the lowest $\mathcal{H}$ are the small isolated regions around the edges of the disk and between the rings. These isolated regions tend to produce patches with NaN-values, which are then converted to zeros before being inputted into the CNN. These zeros would make the patch

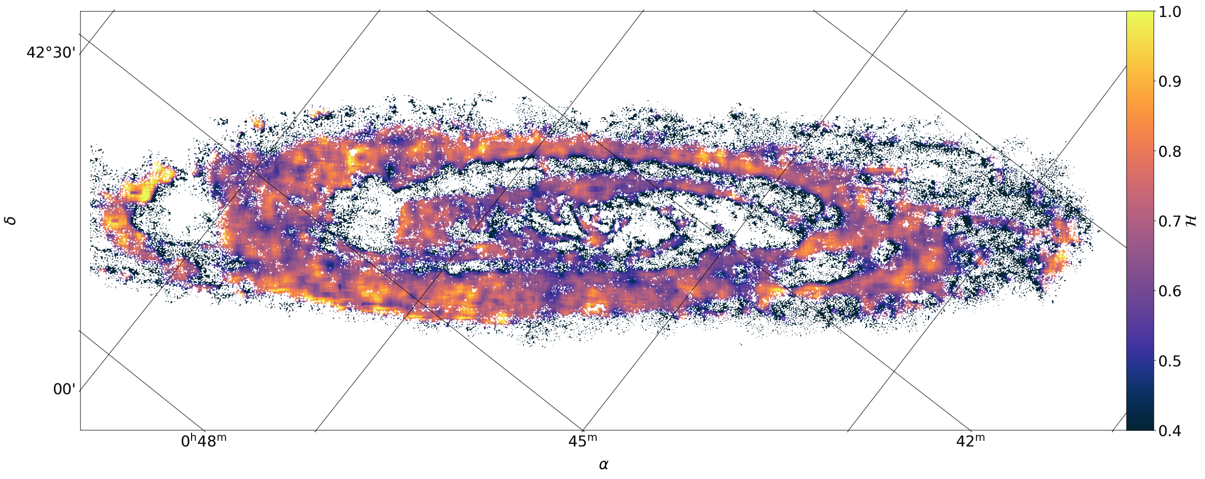


Fig. 5.9 A map of $\mathcal{H}$ for M31.

appear to have rougher structures than patches with high column density and would thus have a power spectrum that is steeper at the short wavelengths. This would produce a relatively low $\mathcal{H}$. We refer the reader to Figure 2.9 to have a rough idea of what these regions look like.

Figure 5.10 shows the resulting map of $\mathcal{S}$. We have flipped the colourbar so that the darker colours represent high $\mathcal{S}$. This is different from Figure 5.9 where the darker colours represent low $\mathcal{H}$. This was done in order to provide a better contrast of the isolated regions against the white background, which is a direct result of these isolated regions having high values of $\mathcal{S}$. These high values $\mathcal{S}$ can also be explained by the abundance of zero pixels in their patches. These pixels provide a relatively large contrast between them and the non-zero pixels which results in a relatively large $\mathcal{S}$. Figure 2.9 shows a region with high $\mathcal{S}$, for comparison. Low values of $\mathcal{S}$ are more present within the contiguous portions of the disk, i.e. the rings. This implies that these regions have a relatively low contrast and as a result have a low variance in column density across neighbouring regions. We note that regions of high $\mathcal{H}$ tend to have low $\mathcal{S}$ and vice versa. This is shown more clearly in Figure 5.11.

Figure 5.11 shows the distributions of $\mathcal{H}$ and $\mathcal{S}$ for M31. The left panel shows the distribution of $\mathcal{H}$, while the middle panel shows the distribution of $\mathcal{S}$. The peak of each of these distributions have been marked with a red line. The right panel shows a two-dimensional distribution of $\mathcal{H}$ against $\mathcal{S}$. The peak of this 2D distribution corresponds to the peaks of the 1D distributions and is at $(\mathcal{H} = 0.67, \mathcal{S} = 1.08)$. This value of $\mathcal{H} = 0.67$ corresponds to a fractal dimension of $\mathcal{D} = 1.33$. This agrees with other measurements of fractal dimension in the ISM, which generally find $1.2 \lesssim \mathcal{D} \lesssim 1.5$ (Beech, 1987; Bazell and Desert, 1988;

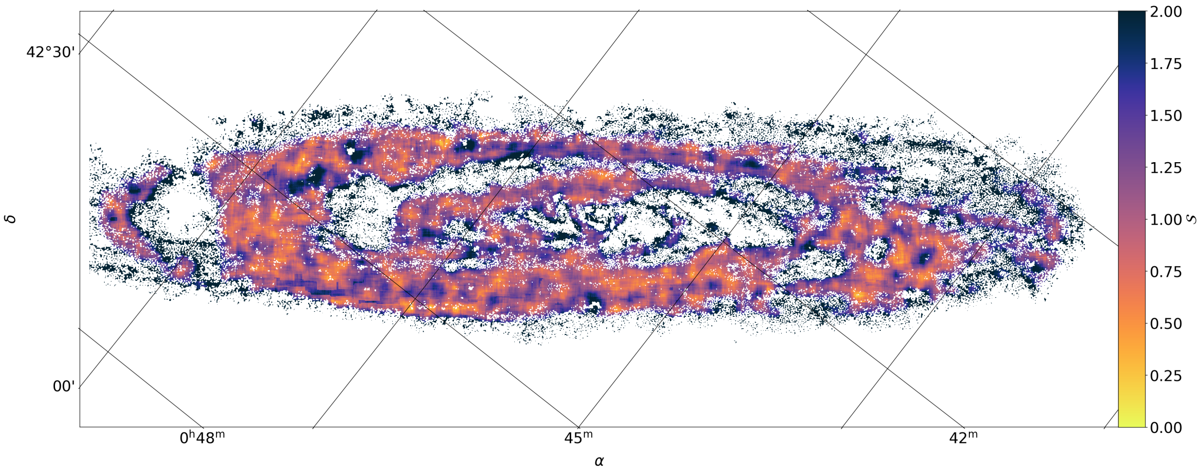


Fig. 5.10 A map of $\mathcal{S}$ for M31. We have flipped the colourbar to provide enough contrast for the isolated regions of high $\mathcal{S}$ to show against the background.

Dickman et al., 1990; Falgarone et al., 1991; Vogelaar and Wakker, 1994; Sanchez et al., 2005; Marchuk et al., 2021). The 2D distribution also confirms that there is a negative correlation between $\mathcal{H}$ and $\mathcal{S}$ where high values of $\mathcal{H}$ tend to have low values of $\mathcal{S}$. This was previously noted in the maps of $\mathcal{H}$ and $\mathcal{S}$ (Figures 5.9 and 5.10). This is in contrast with what was found for the Milky Way Galaxy, which generally had a positive correlation between $\mathcal{H}$ and $\mathcal{S}$ (Figure 5.4). We also find no evidence of any bimodality in these distributions, however this is not too surprising since the spatial resolution is much larger than that used in the analysis of the Milky Way. This larger spatial resolution means that any information that is needed to tease out the smaller undulating mode is mostly present at a smaller scale than that of a single pixel, i.e. 15 pc. This implies that the process that produces the undulating mode is of a scale < 15 pc. In Section 5.5.2 we postulate that this undulating mode is due to turbulence.

Figure 5.12 shows 2D distributions of $\mathcal{H}$ and $\mathcal{S}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$. The panel on the left shows $\mathcal{H}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$ while the panel on the right shows $\mathcal{S}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$. We see that the peak at $\mathcal{H} = 0.67$ and $\mathcal{S} = 1.08$ coincides with a column density peak at $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}]) \sim 0.3$, which gives $\Sigma \sim 2\ \mathrm{M_\odot pc^{-2}}$. If we compare this image with Figure 5.5 we see that there are some similarities. For $\mathcal{H}$ against $\log_{10}(\Sigma/[\mathrm{M_\odot pc^{-2}}])$ we see that all plots generally form an inverted L shape, consisting of two regions meeting at a peak at a relatively high $\mathcal{H}$. For $\mathcal{S}$ we see that the peak tends to be at low $\mathcal{S}$ but there is a steep upwards trend towards high values of $\mathcal{S}$.

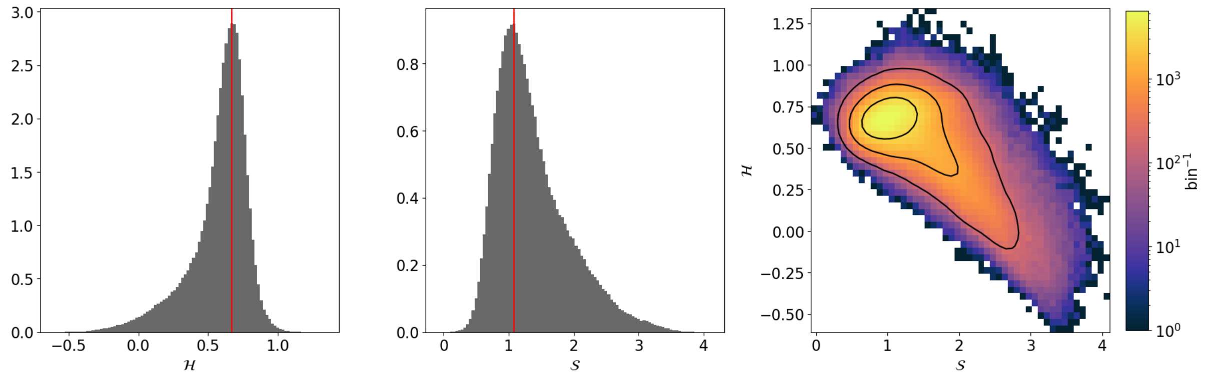


Fig. 5.11 The distribution of $\mathcal{H}$ and $\mathcal{S}$ values for the map of M31 shown in Figure 5.9. *Left:* Histogram showing the distribution of $\mathcal{H}$. The peak is at $\mathcal{H} = 0.67$ and is shown as a vertical red line. Each bin has size $\Delta\mathcal{H} = 1.95 \times 10^{-2}$. *Middle:* Histogram showing the distribution of $\mathcal{S}$. The peak is at $\mathcal{S} = 1.08$. Each bin has size $\Delta\mathcal{S} = 4.17 \times 10^{-2}$. *Right:* A two-dimensional distribution of $\mathcal{H}$ against $\mathcal{S}$. The bin sizes here are $\Delta\mathcal{H} = 3.90 \times 10^{-2}$ and $\Delta\mathcal{S} = 8.35 \times 10^{-2}$. The contours show lines of constant density at the following values: 2×10^2, 10^3, and 3×10^3.

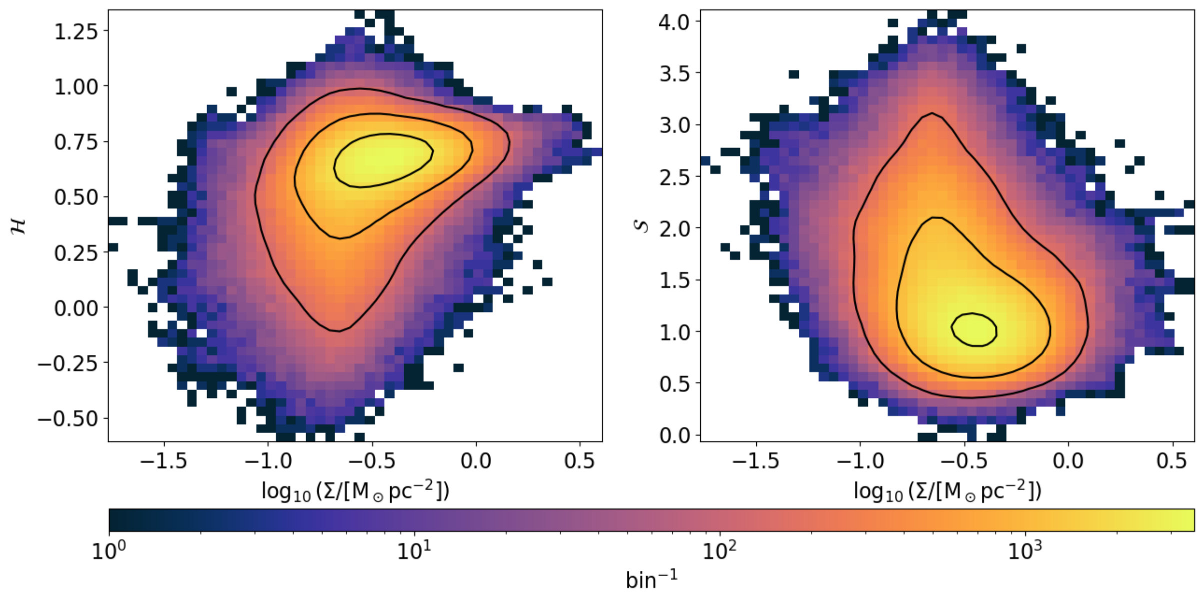


Fig. 5.12 Two-dimensional distributions of: *Left:* $\mathcal{H}$ against the column density, $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$, and *Right:* $\mathcal{S}$ against $\log_{10}(\Sigma/[\mathrm{M}_\odot\mathrm{pc}^{-2}])$. The contours are the same as those in Figure 5.11. The bins for $\mathcal{H}$ and $\mathcal{S}$ are the same as in Figure 5.11. The column density bins are $\Delta\log_{10}\Sigma = 4.74 \times 10^{-2}$.

5.7 Caveats and Assumptions

We must mention multiple assumptions that we have made in the analysis of both the Milky Way plane and the Andromeda Galaxy. One important assumption is made when we assume a constant mass opacity index at 300μm, κ_{300}. It is highly likely that κ_{300} varies spatially within both the Milky Way Galaxy and Andromeda. This is because there are both variations in the amount of dust in the ISM as well as a variation in the metallicity (Wenger et al., 2019; Galliano et al., 2021). The assumption that κ_{300} is constant has two implications. The first is that we assume that κ_{300} is the same for each pixel, and thus there is a linear conversion from the dust optical-depth τ_{300} and the surface density, Σ (See Equation 5.4). The second implication is that κ_{300} is constant along the line of sight. This is an assumption that is made as part of the PPMAP algorithm (Marsh et al., 2015, 2017; Whitworth et al., 2019; Bates and Whitworth, 2023).

We must also stress that although the same techniques have been applied to both Andromeda and the Milky Way, these two analyses are conducted under very different circumstances. The Milky Way analysis has been done from inside the Galaxy and encompasses a wide range of distances, with many structures being superimposed along the line-of-sight, while the Andromeda analysis has been done from an external point-of-view, with the distances to the structures within the galaxy being roughly equal and thus comparable.

Another few assumptions are made with respect to the fractal structure. Firstly, this analysis assumes that each patch that is inputted into the CNNs can be described as an xfBm field. xfBm fields are mono-fractal, ($\mathcal{H}$ does not change with spatial scale or position), however there is a strong possibility that the ISM is multi-fractal (Elia et al., 2014, 2018; Robitaille et al., 2020), and thus each patch can not be described by a single $\mathcal{H}$. Each particular line of sight also includes the contributions from multiple fractal structures and thus cannot be described by a single $\mathcal{H}$ either. These issues might be mitigated by higher resolution images such that there is less confusion between neighbouring structures.

5.8 Chapter Summary

To summarise, we create maps of $\mathcal{H}$ and $\mathcal{S}$ using the maps of total surface density of both the Milky Way Plane and M31. The maps of total surface density have been generated by applying the PPMAP algorithm (Marsh et al., 2015, 2017) on Hi-GAL data (Molinari et al., 2010). We assign an $N_{\text{PIX}} \times N_{\text{PIX}}$ area patch around each of the pixels in these surface density maps. We then pass this patch through the appropriate CNN in order to assign an estimated value of both $\mathcal{H}$ and $\mathcal{S}$ to that pixel. We compare the maps of $\mathcal{H}$ and $\mathcal{S}$ to each other, to the column density, Σ, and (in the case of the Galactic Plane) to the galactic longitude, ℓ. In addition to passing $N_{\text{PIX}} = 32$ and $N_{\text{PIX}} = 64$ sized patches to their respective CNNs, we also pass $N_{\text{PIX}} = 64$ patches to CNN:32 by rebinning the resolution of each patch to a pixel size of 32×32. As a result each pixel gets assigned three values of $\mathcal{H}$ and three values of $\mathcal{S}$.

The distributions of the estimates for $\mathcal{H}$ and $\mathcal{S}$ (Figure 5.4) shows evidence of bimodality in the plane of the Milky Way, with peaks centred at $(\mathcal{H}, \mathcal{S}) \sim (0.9, 0.9)$, which we call the monolithic mode, and $(\mathcal{H}, \mathcal{S}) \sim (0.6, 0.4)$, which we call the undulating mode. The terms 'monolithic' and 'undulating' roughly describe, in simple terms, the structures that these values of $\mathcal{H}$ and $\mathcal{S}$ produce. Structures with $(\mathcal{H}, \mathcal{S}) \sim (0.9, 0.9)$ have a smooth, monolithic structure, while structures with $(\mathcal{H}, \mathcal{S}) \sim (0.6, 0.4)$ have a rough, undulating structure with notable substructures. We find that these two modes occupy different regimes in the column density space that are separated by a point at $\Sigma \sim 32\text{M}_{\odot}\text{pc}^{-2}$, with the monolithic mode corresponding to $\Sigma \gtrsim 32\text{M}_{\odot}\text{pc}^{-2}$ and the undulating mode corresponding to $\Sigma \lesssim 32\text{M}_{\odot}\text{pc}^{-2}$. We find that the monolithic mode represents regions where self-gravity is important (and are thus undergoing star formation), while the undulating mode represents regions where turbulence is dominant. Since the monolithic mode is associated with denser regions we see that this mode is mostly present in the Galactic centre, while the undulating mode is present in the outer parts of the Galaxy. This bimodality is only present when the original patch is of pixel size $N_{\text{PIX}} = 64$, regardless of whether we've used CNN:32 (by rebinning the image) or CNN:64.

We try to find evidence of these two modes by exploring the surface-density at different temperature bins (Figure 5.7). These temperature-differential surface densities are generated using the PPMAP algorithm (Marsh et al., 2015, 2017). We compare contributions at different surface-densities and temperatures in order to find evidence of these modes. Since the monolithic mode is associated with a higher surface density, we investigate whether the distributions for the surface densities show any differences at different surface density and temperature cutoffs. We find no obvious differences in the distributions at high surface densities.

We then apply CNN:32 to a surface-density map of the Andromeda Galaxy (M31). We find no evidence of the previous bimodality in the distributions of $\mathcal{H}$ and $\mathcal{S}$. This is probably due to the larger length-scale of the patches, which is approximately 15pc. Instead, we find that there is a peak at $(\mathcal{H}, \mathcal{S}) = (0.67, 1.08)$, with a negative correlation between $\mathcal{H}$ and $\mathcal{S}$. This peak also coincides with a peak at column density, $\Sigma \sim 2\mathrm{M}_{\odot}\mathrm{pc}^{-2}$. The value of the peak at $\mathcal{H} = 0.67$ corresponds to a fractal dimension of $\mathcal{D} = 1.33$, which is in agreement with the current literature.

CHAPTER 6

Conclusions

6.1 Summary and Conclusions

We have trained and tested three Convolutional Neural Networks (CNNs) using exponeniated fractional Brownian motion (xfBm) images. xfBm images are a modified form of fBm images and are described using three parameters: the Hurst parameter $\mathcal{H}$, the scaling parameter $\mathcal{S}$ and the pixel size of the image, $N_{\rm PIX}$. The Hurst parameter is related to the perimeter-area fractal dimension, $\mathcal{D}_{PA} = 2 - \mathcal{H}$, and to the power-law exponent, β, of the power spectrum via $\beta = 2(1 + \mathcal{H})$. Each of the CNNs, which we name CNN:128, CNN:64, CNN:32, takes as an input a 2-dimensional $N_{\rm PIX} \times N_{\rm PIX}$ pixel sized image and produces an estimate for $\mathcal{H}$ and $\mathcal{S}$. The three CNNs differ only by $N_{\rm PIX}$, with values of $N_{\rm PIX} = 128, 64,$ and 32. We use the naming convention of CNN:$N_{\rm PIX}$, when referring to a particular CNN.

We compare the performance of the CNNs with that of widely used estimators of $\mathcal{H}$ such as the Perimeter-Area method and Delta-Variance. We find that, when trained, the CNNs outperform both methods in both speed and accuracy. We find that the difference in accuracy between Delta-Variance and the CNNs becomes greater when operating on images with smaller $N_{\rm PIX}$. We also note that the CNNs estimate $\mathcal{S}$ as well as $\mathcal{H}$, and while $\mathcal{S}$ is analogous to the standard deviation of the field, we find that using the CNNs produces a more accurate estimation of $\mathcal{S}$ than simply taking the standard deviation of the field.

If we consider the angular size of each pixel as being $\phi_{\rm min} \times \phi_{\rm min}$, then we can assume that the CNNs capture structures with a range of scales between $\phi_{\rm min}$ and $\phi_{\rm max} = N_{\rm PIX}\phi_{\rm min}$. Thus we can use $N_{\rm PIX}$ to set the range of angular scales of the image that is being analysed. There is a trade-off to consider here. At large $N_{\rm PIX}$, the CNNs are more accurate as they can use more pixels in their calculations. This, however, assumes that the image is mono-fractal (i.e. $\mathcal{H}$ is constant everywhere) and thus we have to assume that there is no confusion between different structures with different $\mathcal{H}$ and $\mathcal{S}$. A smaller $N_{\rm PIX}$ gives a more localised estimation of $\mathcal{H}$ and $\mathcal{S}$, albeit a less accurate one.

We apply CNN:64 and CNN:32 to images of surface density of the interstellar medium (ISM) of the Galactic Plane. The images used here are those of the Hi-GAL survey, which have been further processed into maps of the total column density using the `PPMAP` algorithm. These maps have an angular resolution of $6''$ per pixel. For each pixel in these maps we take the surrounding $N_{\rm PIX} \times N_{\rm PIX}$ patch as an input to the CNNs and assign the resulting value of $\mathcal{H}$ and $\mathcal{S}$ to that pixel. When we apply CNN:64 and CNN:32 to these maps we label the results as '64×6' and '32×6' respectively. These have $(\phi_{\rm min}, \phi_{\rm max}) = (6'', 384'')$ and $(6'', 192'')$ respectively. In addition we also apply CNN:32 to a $N_{\rm PIX} = 64$ patch by degrading this patch to a resolution of 32×32 pixels. This gives $\phi_{\rm min} = 12''$ and $\phi_{\rm max} = 384''$. The results here are labelled '32×12'.

We find that regions of higher surface density ($\Sigma \gtrsim 32$ M$_{\odot}$pc^{-2}) correlate with large $\mathcal{H}$ ($\gtrsim 0.8$) and $\mathcal{S}$ ($\gtrsim 1$). We call this the *monolithic mode*. The high $\mathcal{H}$ of this mode implies a steep power spectrum, with power being more concentrated in the larger scales, while the high $\mathcal{S}$ implies a large range of surface densities. These regions are more commonly found towards the Galactic Centre, i.e in the inner parts of the Galaxy.

Regions of low surface density ($\Sigma \lesssim 32$ M$_{\odot}$pc^{-2}) correlate with small $\mathcal{H}$ ($\lesssim 0.8$) and $\mathcal{S}$ ($\lesssim 1$). The lower $\mathcal{H}$ implies a shallower power spectrum and thus we expect to see more small scale structure than in the monolithic mode. The lower $\mathcal{S}$ implies a smaller range of surface densities. These regions are more commonly found in the outer portions of the Galaxy, as well as other regions where the surface density is lower. We call this the *undulating mode*.

The `PPMAP` algorithm also produces maps of the surface-density in different dust temperature intervals. We compare the distributions of surface-density at dust temperatures above and below a given separatrix temperatures, T_S. The values of the separatrix temperatures used are $T_S = 12.1$ K, 14.3 K, 16.9 K, 20.0 K, and 23.6 K. We find that both the cooler dust and the warmer dust have surface densities that have a power-law relation, of the form, $dP/d\Sigma \propto \Sigma^{-3}$ at large surface-densities. This is independent of T_S. We observe that the median dust temperature is $\sim 20\ K$. This can be seen at the point at which the distributions for both the cooler and warmer dust-temperature surface-densities are roughly equivalent. We find no evidence of the two modes (monolithic and undulating) in the temperature-differential surface-density distributions.

We also apply CNN:32 to a map of the Andromeda Galaxy (M31). This map is a *Herschel* derived map of surface-density that has been produced using `PPMAP`. The pixel resolution here is $4''$, which gives $\phi_{\rm min} = 4''$ and $\phi_{\rm max} = 128''$. We convert this to spatial distances of $r_{\rm min} = 15$ pc and $r_{\rm max} = 240$ pc respectively, due to the relatively constant distance to M31 of $\mathcal{D} \approx 0.78$ Mpc. We note that this is different to the analysis of the Milky Way Plane in two ways: firstly, the column density of the Milky Way Plane likely has contributions from material at different distances along the line of sight, and secondly, the distance to the material in the Milky Way is generally less than that of M31 and thus the spatial resolution of each pixel in M31 is likely much larger than that of the Milky Way.

The distributions of $\mathcal{H}$ and $\mathcal{S}$ in M31 have peaks at $\mathcal{H} = 0.67$ and $\mathcal{S} = 1.08$, with $\mathcal{H}$ and $\mathcal{S}$ having an anti-correlation. $\mathcal{H} = 0.67$ is equivalent to $\mathcal{D} = 1.33$, which agrees with the current literature (Beech, 1987; Bazell and Desert, 1988; Dickman et al., 1990; Falgarone et al., 1991; Vogelaar and Wakker, 1994; Sanchez et al., 2005; Marchuk et al., 2021). These peaks (in both $\mathcal{H}$ and $\mathcal{S}$) correspond to a peak in the surface-density distribution of $\Sigma \sim 2$ M$_\odot$pc^{-2}.

6.2 Future Work

Fractal structure is not limited to surface density. These CNN models can also be applied to maps of temperature or velocity. Maps of density need not be a continuous field either. The kernel density estimation of a set of points (for example a star cluster) can also be used as an input to the model.

There are also avenues in which the xfBm models can be extended. Features such as dense cores, and filaments can be added.

We must also mention that the field of Machine Learning is accelerating quickly. In fact, by the time of this writing, the CNN architecture have already been expanded upon; namely by autoencoders and by generative transformers. This is an exciting field and the usage of these new models can be used for more complex dynamical systems (e.g. the addition of the time domain, the training of which necessitates the numerical simulation of the ISM).

References

Allan D., 1966, Proceedings of the IEEE, 54, 221

André P., et al., 2010, Astronomy and Astrophysics, 518, L102

Angulo C., 2008, IEEE Transactions on Neural Networks, 19, 1990

Arzoumanian D., et al., 2011, Astronomy and Astrophysics, 529, L6

Bates M. L., Whitworth A. P., 2023, Monthly Notices of the Royal Astronomical Society, 523, 233

Bates M. L., Whitworth A. P., Lomax O. D., 2020, Monthly Notices of the Royal Astronomical Society, 493, 161

Bazell D., Desert F. X., 1988, The Astrophysical Journal, 333, 353

Beattie J. R., Federrath C., Klessen R. S., 2019a, Monthly Notices of the Royal Astronomical Society, 487, 2070

Beattie J. R., Federrath C., Klessen R. S., Schneider N., 2019b, Monthly Notices of the Royal Astronomical Society, 488, 2493

Beech M., 1987, Astrophysics and Space Science, 133, 193

Beuther H., Klessen R. S., Dullemond C. P., Henning T., eds, 2014, Protostars and planets VI. Lunar and Planetary Institute, Houston

Bialopetravičius J., Narbutis D., Vansevičius V., 2019, Astronomy and Astrophysics, 621, A103

Bishop C. M., 1995, Neural networks for pattern recognition. Clarendon Press ; Oxford University Press, Oxford : New York

Bishop C. M., 2006, Pattern recognition and machine learning. Information science and statistics, Springer, New York

Blitz L., 1997, in Latter W. B., Radford S. J. E., Jewell P. R., Mangum J. G., Bally J., eds, , CO: Twenty-Five Years of Millimeter-Wave Spectroscopy. Springer Netherlands, Dordrecht,

pp 11–18, doi:10.1007/978-94-011-5414-7˙2,
http://link.springer.com/10.1007/978-94-011-5414-7_2

Breiman L., Gordon A. D., Friedman J. H., Olshen R. A., Stone C. J., 1984, Biometrics, 40, 874

Brunt C. M., Federrath C., 2014, Monthly Notices of the Royal Astronomical Society, 442, 1451

Chappell D., Scalo J., 2001, The Astrophysical Journal, 551, 712

Cortes C., Vapnik V., 1995, Machine Learning, 20, 273

Dawson J. M., Davis T. A., Gomez E. L., Schock J., Zabel N., Williams T. G., 2019, Monthly Notices of the Royal Astronomical Society, p. stz3097

Dawson J. M., Davis T. A., Gomez E. L., Schock J., 2021, Monthly Notices of the Royal Astronomical Society, 503, 574

De Graauw T., et al., 2010, Astronomy and Astrophysics, 518, L6

De La Fuente Marcos R., De La Fuente Marcos C., 2006, Monthly Notices of the Royal Astronomical Society, 372, 279

Dickman R. L., Margulis M., Horvath M. A., 1990, The Astrophysical Journal, 365, 586

Dieleman S., Willett K. W., Dambre J., 2015, Monthly Notices of the Royal Astronomical Society, 450, 1441

Donkov S., Stefanov I. Z., Veltchev T. V., Klessen R. S., 2021, Monthly Notices of the Royal Astronomical Society, 505, 3655

Elia D., et al., 2013, The Astrophysical Journal, 772, 45

Elia D., et al., 2014, The Astrophysical Journal, 788, 3

Elia D., et al., 2018, Monthly Notices of the Royal Astronomical Society, 481, 509

Elmegreen B. G., 1997, The Astrophysical Journal, 477, 196

Elmegreen B. G., 2002, The Astrophysical Journal, 564, 773

Elmegreen B. G., Falgarone E., 1996, The Astrophysical Journal, 471, 816

Falgarone E., Phillips T. G., Walker C. K., 1991, The Astrophysical Journal, 378, 186

Federrath C., Klessen R. S., Schmidt W., 2009, The Astrophysical Journal, 692, 364

Field G. B., Goldsmith D. W., Habing H. J., 1969, The Astrophysical Journal, 155, L149

Flamary R., 2016, arXiv e-prints, p. arXiv:1612.04526

Gabbard H., Williams M., Hayes F., Messenger C., 2018, Physical Review Letters, 120, 141103

Galliano F., et al., 2021, Astronomy and Astrophysics, 649, A18

George D., Huerta E., 2018, Physics Letters B, 778, 64

Girichidis P., Konstandin L., Whitworth A. P., Klessen R. S., 2014, The Astrophysical Journal, 781, 91

Griffin M. J., et al., 2010, Astronomy and Astrophysics, 518, L3

Habing H. J., 1968, Bulletin of the Astronomical Institutes of the Netherlands, 19, 421

Halsey T. C., Jensen M. H., Kadanoff L. P., Procaccia I., Shraiman B. I., 1986, Physical Review A, 33, 1141

Hentschel H. G. E., Procaccia I., 1983, Physical Review A, 27, 1266

Hetem Jr. A., Lepine J. R. D., 1993, Astronomy and Astrophysics, 270, 451

Hildebrand R. H., 1983, Quarterly Journal of the Royal Astronomical Society, 24, 267

Hollenbach D., McKee C. F., 1979, The Astrophysical Journal Supplement Series, 41, 555

Hou L. G., Han J. L., 2014, Astronomy and Astrophysics, 569, A125

Hou L. G., Han J. L., 2015, Monthly Notices of the Royal Astronomical Society, 454, 626

Howard A. D. P., Whitworth A. P., Griffin M. J., Marsh K. A., Smith M. W. L., 2021, Monthly Notices of the Royal Astronomical Society, 504, 6157

Jaupart E., Chabrier G., 2020, The Astrophysical Journal Letters, 903, L2

Johnstone D., Di Francesco J., Kirk H., 2004, The Astrophysical Journal, 611, L45

Jura M., 1974, The Astrophysical Journal, 191, 375

Juvela M., 2023,] 10.48550/ARXIV.2304.05102

Jáquez-Domínguez J. M., et al., 2023,] 10.48550/ARXIV.2304.04864

Kainulainen J., Beuther H., Henning T., Plume R., 2009, Astronomy and Astrophysics, 508, L35

Kessler M. F., et al., 1996, Astronomy and Astrophysics, 315, L27

Khalifa N. E. M., Taha M. H. N., Hassanien A. E., Selim I. M., 2017, arXiv:1709.02245 [cs]

Khullar S., Federrath C., Krumholz M. R., Matzner C. D., 2021, Monthly Notices of the Royal Astronomical Society, 507, 4335

Kimura A., Takahashi I., Tanaka M., Yasuda N., Ueda N., Yoshida N., 2017, arXiv e-prints, p. arXiv:1711.11526

Klessen R. S., Heitsch F., Mac Low M., 2000, The Astrophysical Journal, 535, 887

Konstandin L., Schmidt W., Girichidis P., Peters T., Shetty R., Klessen R. S., 2016, Monthly Notices of the Royal Astronomical Society, 460, 4483

Kritsuk A. G., Norman M. L., Wagner R., 2011, The Astrophysical Journal, 727, L20

Kritsuk A. G., Lee C. T., Norman M. L., 2013, Monthly Notices of the Royal Astronomical Society, 436, 3247

Krumholz M. R., McKee C. F., 2020, Monthly Notices of the Royal Astronomical Society, 494, 624

Könyves V., et al., 2015, Astronomy and Astrophysics, 584, A91

Ladjelate B., et al., 2020, Astronomy and Astrophysics, 638, A74

LeCun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D., 1989, Neural Computation, 1, 541

LeCun Y., Bottou L., Bengio Y., Haffner P., 1998, Proceedings of the IEEE, 86, 2278

Lomax O., Bates M. L., Whitworth A. P., 2018, Monthly Notices of the Royal Astronomical Society, 480, 371

Mandelbrot B. B., Cannon J. W., 1984, The American Mathematical Monthly, 91, 594

Marchuk A. A., Smirnov A. A., Mosenkov A. V., Il'in V. B., Gontcharov G. A., Savchenko S. S., Román J., 2021, Monthly Notices of the Royal Astronomical Society, 508, 5825

Marsh K. A., Whitworth A. P., Lomax O., 2015, Monthly Notices of the Royal Astronomical Society, 454, 4282

Marsh K. A., et al., 2016, Monthly Notices of the Royal Astronomical Society, 459, 342

Marsh K. A., et al., 2017, Monthly Notices of the Royal Astronomical Society, 471, 2730

Mathis J. S., Mezger P. G., Panagia N., 1983, Astronomy and Astrophysics, 128, 212

Molinari S., et al., 2010, Publications of the Astronomical Society of the Pacific, 122, 314

Nair V., Hinton G. E., 2010, Proceedings of the 27th International Conference on International Conference on Machine Learning, pp 807–814

Neugebauer G., et al., 1984, The Astrophysical Journal, 278, L1

Ossenkopf V., Krips M., Stutzki J., 2008a, Astronomy and Astrophysics, 485, 719

Ossenkopf V., Krips M., Stutzki J., 2008b, Astronomy and Astrophysics, 485, 917

Peitgen H.-O., Saupe D., eds, 1988, The science of fractal images. Springer, New York Berlin

Petkova M. A., et al., 2023, Monthly Notices of the Royal Astronomical Society, 520, 2245

Pilbratt G. L., et al., 2010, Astronomy and Astrophysics, 518, L1

Poglitsch A., et al., 2010, Astronomy and Astrophysics, 518, L2

Postnikov E.,

Quinlan J. R., 1986, Machine Learning, 1, 81

Quinlan J. R., 1993, C4.5: Programs for Machine Learning. Morgan Kaufmann, doi:10.1016/C2009-0-27846-9, `https://linkinghub.elsevier.com/retrieve/pii/C20090278469`

Rathborne J. M., et al., 2015, The Astrophysical Journal, 802, 125

Remez T., Litany O., Giryes R., Bronstein A. M., 2017, arXiv e-prints, p. arXiv:1701.01687

Robitaille T., Deil C., Ginsburg A., 2020, Astrophysics Source Code Library, p. ascl:2011.023

Sadavoy S. I., et al., 2012, Astronomy and Astrophysics, 540, A10

Sanchez N., Alfaro E. J., Perez E., 2005, The Astrophysical Journal, 625, 849

Schneider N., et al., 2012, Astronomy and Astrophysics, 540, L11

Schneider N., et al., 2015, Monthly Notices of the Royal Astronomical Society: Letters, 453, L41

Schneider N., et al., 2022, Astronomy and Astrophysics, 666, A165

Schroeder M. R., 1996, Fractals, chaos, power laws: minutes from an infinite paradise, nachdr. edn. Freeman, New York, NY

Shadmehri M., Elmegreen B. G., 2011, Monthly Notices of the Royal Astronomical Society, 410, 788

Shallue C. J., Vanderburg A., 2018, The Astronomical Journal, 155, 94

Shen H., George D., Huerta E. A., Zhao Z., 2019, in ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, Brighton, United Kingdom, pp 3237–3241, doi:10.1109/ICASSP.2019.8683061, `https://ieeexplore.ieee.org/document/8683061/`

Spitzer L., 1998, Physical Processes in the Interstellar Medium, 1 edn. Wiley, doi:10.1002/9783527617722, `https://onlinelibrary.wiley.com/doi/book/10.1002/9783527617722`

Stahler S. W., Palla F., 2004, The formation of stars. Wiley-VCH, Weinheim

Sternberg A., Dalgarno A., 1995, The Astrophysical Journal Supplement Series, 99, 565

Sternberg A., Le Petit F., Roueff E., Le Bourlot J., 2014, The Astrophysical Journal, 790, 10

Stutzki J., Bensch F., Heithausen A., Ossenkopf V., Zielinsky M., 1998, Astronomy and Astrophysics, 336, 697

TensorFlow-Developers 2023, TensorFlow, doi:10.5281/ZENODO.4724125, `https://zenodo.org/record/4724125`

Vazquez-Semadeni E., Garcia N., 2001, The Astrophysical Journal, 557, 727

Virtanen P., et al., 2020, Nature Methods, 17, 261

Vogelaar M. G. R., Wakker B. P., 1994, åp, 291, 557

Voss R. F., 1985, in , Fundamental Algorithms for Computer Graphics. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 805–835, doi:10.1007/978-3-642-84574-1˙34, `http://link.springer.com/10.1007/978-3-642-84574-1_34`

Ward R. L., Wadsley J., Sills A., 2014, Monthly Notices of the Royal Astronomical Society, 445, 1575

Wenger T. V., Balser D. S., Anderson L. D., Bania T. M., 2019, The Astrophysical Journal, 887, 114

Werner M. W., et al., 2004, The Astrophysical Journal Supplement Series, 154, 1

Whitworth A., et al., 2019, Monthly Notices of the Royal Astronomical Society, 489, 5436

Wolfire M. G., McKee C. F., Hollenbach D., Tielens A. G. G. M., 2003, The Astrophysical Journal, 587, 278

Wouterloot J. G. A., Brand J., Burton W. B., Kwee K. K., 1990, Astronomy and Astrophysics, 230, 21

Zevin M., et al., 2017, Classical and Quantum Gravity, 34, 064003

de Vega H. J., Sánchez N., Combes F., 1996, Nature, 383, 56

LIST OF PUBLICATIONS

1. Lomax O., Bates M. L., Whitworth A. P., 2018, Modelling the structure of star clusters with fractional Brownian motion, Monthly Notices of the Royal Astronomical Society, 480, 371
2. Bates M. L., Whitworth A. P., Lomax O. D., 2020, Characterizing lognormal fractional-Brownian-motion density fields with a convolutional neural network, Monthly Notices of the Royal Astronomical Society, 493, 161
3. Bates M. L., Whitworth A. P., 2023, A statistical analysis of the structure of the interstellar medium in the disc of the Milky Way, Monthly Notices of the Royal Astronomical Society, 523, 233