UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

MSc thesis

# Estimating cosmic dust properties using Bayesian inference for data collected with the James Webb Space Telescope

**Mikolaj Mazurczyk**

**Supervisor: Asst. Prof. Oswin Krause**

**Co-supervisor: Assoc. Prof. Christa Gall**

**Submitted: September 14, 2023**

## Abstract

Cosmic dust plays a crucial role in the evolution of interstellar objects. Even though its role in astronomical processes is relatively well understood, there is still ongoing debate regarding its origin. The most likely sources of cosmic dust in the universe are thought to be core-collapse supernovae. To confirm that hypothesis, there is a need for accurate estimates of the quantities of dust ejected by a dying star. The dust formed around supernovae is measured through the electromagnetic radiation that they emit, which can be captured by advanced telescopes, such as the James Webb Space Telescope (JWST). Recently, there have been attempts to apply machine learning to test this hypothesis. Since real-world data is scarce, models are trained with a simulated dataset of JWST observations of supernova explosions. However, optimizing regular models predicting the dust features based on JWST observations displayed several limitations.

For instance, collecting data with JWST is resource-consuming, and since not all of its instruments can be used at once, this implies that, typically, there is missing data. Furthermore, many of the dust features turn out to be nearly impossible to predict accurately due to their complex distributions. Hence, there is a need for robust estimates of uncertainties around the predictions.

The approach presented in this thesis aims to address these challenges through the application of Bayesian inference to the issue. I propose to represent the Bayesian likelihood density with the deep learning model. Subsequently, employing the Markov Chain Monte Carlo technique allows me to sample from the posterior. Choosing the distribution predicted by the model so that one can easily marginalize filters from it addresses the first limitation. Furthermore, the representation of the distribution of dust features with the Bayes rule allows for an intricate and accurate estimate of the posterior. Additionally, this work offers a comparative study between a regular neural network and a more advanced architecture, the Conditional Variational Autoencoder (CVAE), in their ability to represent likelihood density.

In conclusion, my results prove that CVAE does not provide significantly better results than the regular neural network. Nonetheless, both still performed better than the non-Bayesian standard method, even with some of the JWST observations marginalized, achieving the primary objective of the thesis.

# Contents

# 1 Introduction

Cosmic dust, once regarded solely as a nuisance in the observations of astronomical objects, upon the advancements in infrared astronomy of the 1960s, became understood as a crucial factor in the processes taking place in the universe, starting from its very early years. The dust is composed of solid matter scattered across the space, which contains various materials in their make-up, including carbon, silicates, metals, and organic compounds, where the size of the grain can vary from just a few molecules to clumps measuring 0.1 mm in diameter. Today, the role of space dust in astronomical processes is relatively well understood. For instance, it contributes to the interstellar medium (ISM), influencing ISM's physical conditions and dynamics, it takes part in stellar and planetary formation, and the grains serve as the catalysts for chemical reactions. However, what remains a topic of debate, is its origin, and finding the answer to this question may unveil how the universe formed and give more insight into understanding the future evolution of interstellar objects.

According to Gall et al. [2011], significant quantities of dust observed in the early and local Universe suggest that it must have formed rapidly. Evidence is emerging that the core-collapse supernovae (CCSNe) explosions are efficient dust producers and are likely responsible for the abundance of dust observed in galaxies. CCSNe occur in particularly massive stars and are the effect of the collapse of the balance keeping them alive for millions of years. For most of the star's life cycle, as an effect of the immense pressure and extreme temperatures in the star's core, nuclear fusion reactions occur, converting lighter elements into heavier ones. These reactions release vast amounts of energy that sustain the star's temperature, which in turn creates a pressure applying outwards force that counteracts the inwards gravity force caused by the star's mass. However, as the fused elements get heavier and the core becomes predominantly composed of iron, the balance gets thrown since iron fusion, instead of releasing energy, requires energy input, resulting in the star collapsing under its gravity. The shockwave induced by this violent process propagates outward through the star, causing it to explode in a burst of light and energy. As a result, part of the outer layers gets scattered into space, enriching the ISM with heavy elements synthesized during the supernova, forming a cloud of gas and cosmic dust around the dying star.

Confirming this hypothesis requires a deep understanding of the complex processes occurring in CCSNe, which is particularly challenging, as performing direct measurements of such distant and massive objects is impossible. Thus, based on the current understanding of laws of physics and experiments simulating conditions during supernovae (SNe) explosions, physicists develop the theoretical models of dust formation around SNe, which they then confront with the indirect observations made by instruments in the Earth's closest surroundings. Therefore, the aim of this thesis is to refine the estimates of

quantities and properties of the cosmic dust formed around CCSNe based on these observations using advanced deep learning (DL) and Monte Carlo techniques, which potentially can enhance our understanding of what role do CCSNe play in the origin of the space dust. However, to motivate the design choices made in this work, it is in place to first introduce how the data is collected and the multiple caveats that come with it.

## 1.1 Obtaining the measurements

Considering the cosmic scales, scientists need to rely only on limited data about the objects in space. From the beginning of astrophysics, light has been the most common source of that information, which is part of a more broad class of phenomena called electromagnetic radiation (EMR). Broadly speaking, it can be described as a wave consisting of individual packets of energy, called photons, that propagates through space. Classification of EMR based on its wavelength ranges from extremely short and high energy waves of ionizing radiation through visible light and up to radio waves. Each photon carries a specific amount of energy proportional to its wavelength, and the distribution of collective energy of multiple photons at different wavelengths is called spectral energy distribution (SED). This thesis focuses on the infrared EMR, which has a wavelength range slightly longer than visible light and is naturally emitted by any physical object with a temperature above absolute zero. Even though the dust in space is predominantly too cold to emit measurable amounts of IR radiation, preventing a more detailed inference of its properties, the one formed around SNe gets heated by blown energy, providing a much stronger signal. To capture it, physicists employ IR telescopes, and since Earth's atmosphere absorbs a significant portion of IR radiation, ground-based observations are challenging, which means that the most informative measurements are done using space telescopes. Artificially generated examples of IR SEDs of dust and CCSNe are presented in Figure 1.

The recently launched James Webb Space Telescope (JWST) [Gardner et al., 2006] is the most advanced of such instruments, specifically designed to capture the IR portion of electromagnetic spectrum. Therefore the following thesis focuses on the data collected by JWST, as it will be one of the primary sources of observations of cosmic dust emissions in the following years. The telescope is equipped with a set of 38 filters, where each covers a different wavelength range of the IR light spectrum. During the measurement, the filter captures the magnitude of the dust's IR emission wave at its specific wavelength range, giving a single value per filter as an output. Therefore, instead of observing the full IR light spectra, one can only access a few data points per observation, which poses the first difficulty when analyzing the IR emissions captured by JWST. The next obstacle arises from physical limitations of how the observations are collected, as only a limited number of filters can be used to capture the light at one moment. Also, obtaining an accurate measurement

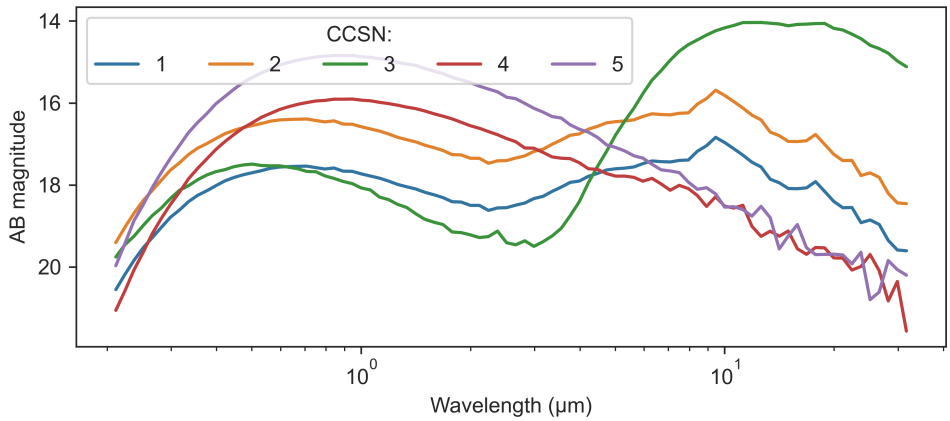


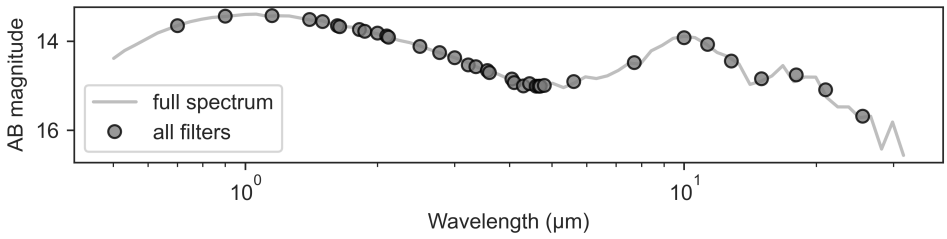

Figure 1: Example of SEDs of different CCSNe and the dust around them. AB magnitude measures the amount of energy that EMR carries, where the lower the value, the stronger the signal is, and thus the y-axis of the plot is flipped. The x-axis is presented on a log scale to magnify the variation of different SEDs.

requires long observation times, and considering the high demand for the usage of JWST, it is also financially expensive. Thus commonly, outputs of only a subset of all filters are available, limiting the information accessible for analysis even more. This contrast between the case where complete information about the spectra is available and the more common one where observations from only a subset of filters are accessible is depicted in Figure 2.

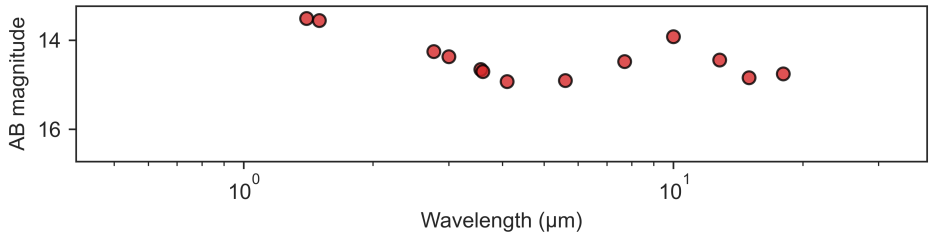
When inferring the properties of the space dust based on this limited information, the usual approach is to generate sample spectral emissions based on the theoretical models and fit them to the observations. Based on the best match, one can estimate the dust's temperature, based on which its mass is derived. Further characteristics, such as the grain size and dust species (i.e. its chemical composition), are more challenging to determine, and since they are especially susceptible to the assumptions made about the SN type and conditions around it, the uncertainties of the estimates are extensive and often vary by an order of magnitude.

## 1.2 Inference of the cosmic dust properties with Machine Learning

Since there is no access to the ground-truth data regarding CCSNe and the dust around it, obtaining reliable estimates of the dust's properties requires consideration of multiple variables and dependencies between them. In recent years, machine learning (ML) proved to excel in these regards, where models applied in a vast range of domains managed to capture intricate patterns




(a) Whole SED with responses from all JWST filters overlaid.




(b) Usual case, where only information from a subset of JWST filters is available.

Figure 2: Example SED of CCSNe and the surrounding dust with a scatter plot showing how it would be observed by all JWST filters in comparison with the more common situation where responses from only a subset of filters are available.

within the data, often exceeding human capabilities. Thus, there is great potential in applying ML to the challenge of inferring the space dust properties, which could potentially aid in narrowing down the uncertainties around the predictions. However, the success of the ML algorithms heavily depends on the quantity and quality of the data they learn from, and in the case of predicting the space dust's properties from IR emissions, this resource is quite scarce. Only a few SNe have been studied well enough to have confident predictions, and considering how expensive it is in terms of resources and time to collect additional data, these circumstances are not expected to change anytime soon.

However, Ansari et al. [2022] overcame this constraint by generating the artificial data set precisely for that purpose. Using the fully three-dimensional radiative transfer code `MOCASSIN` [Ercolano et al., 2006] authors simulated IR emissions of different CCSNe explosions, and by calculating how they would be observed by all JWST filters, they obtained a data set of filter responses and their corresponding dust and SNe parameters, further denoted as vectors of real numbers $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively. By training a DL model to maximize the likelihood of $p(\boldsymbol{y}|\boldsymbol{x})$, they managed to quite accurately predict the dust's mass and temperature, while the dust's specie and the grain size still proved to be rather challenging to estimate. To consider the variability in the number

of JWST filters available for analysis, the authors conducted a feature importance analysis to find the minimum set of JWST filters required to predict the selected dust quantities with an accuracy comparable to standard methods in the literature. However, even though it may serve as a valuable consideration when creating the setup for measurement, in practice, it means that for every set of filters, one would have to train and tune a separate model, making it a cumbersome process.

The following thesis aims to bypass this limitation by utilizing the Bayesian inference:

$$p\left(\boldsymbol{y}|\boldsymbol{x}\right) = \frac{p\left(\boldsymbol{x}|\boldsymbol{y}\right)p\left(\boldsymbol{y}\right)}{p\left(\boldsymbol{x}\right)}.$$

Instead of training a DL model maximizing $p\left(\boldsymbol{y}|\boldsymbol{x}\right)$ directly, I propose a model $\mathcal{M}_\theta$ parameterized by $\theta$ that for each entry $\boldsymbol{y}$ predicts the mean $\boldsymbol{\mu}_x$ and covariance matrix $\boldsymbol{\Sigma}_x$ of the multivariate normal distribution $\mathcal{N}\left(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x\right)$:

$$\mathcal{M}_\theta\left(\boldsymbol{y}\right) \rightarrow \mathcal{N}\left(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x\right) = p_\theta\left(\boldsymbol{x}|\boldsymbol{y}\right),$$

which is trained to minimize the negative log-likelihood (NLL) of true samples under the predicted distribution. Then, with a fixed prior $p\left(\boldsymbol{y}\right)$, the posterior density $p\left(\boldsymbol{y}|\boldsymbol{x}\right)$ can be retrieved using methods such as Markov Chain Monte Carlo (MCMC), since the evidence $p\left(\boldsymbol{x}\right)$ is intractable due to integral

$$p\left(\boldsymbol{x}\right) = \int p\left(\boldsymbol{x}|\boldsymbol{y}\right)p\left(\boldsymbol{y}\right)d\boldsymbol{y},$$

for a fixed $\boldsymbol{x}$, it can be considered as a normalization constant.

The possible advantages of this proposal are manifold. First, because $p\left(\boldsymbol{x}|y\right)$ is represented as a multivariate normal distribution, any missing filters can be easily marginalized from the predicted $\mathcal{N}\left(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x\right)$, resolving the need to train additional models. Furthermore, any prior beliefs motivated by external domain knowledge can be incorporated into $p\left(\boldsymbol{y}\right)$, allowing for flexibility even after model $\mathcal{M}_\theta$ is trained. Moreover, representing $p\left(\boldsymbol{y}|\boldsymbol{x}\right)$ with Bayes' theorem provides much more expressive, possibly multimodal density, compared to merely predicting $\boldsymbol{y}$ along with single-value uncertainties of the predictions, as done by Ansari et al. [2022].

Another goal of this work is to evaluate an improved representation of $p\left(\boldsymbol{x}|\boldsymbol{y}\right)$. The simulation used to generate the data set considers many, but still a limited number of parameters, and thus it incorporates a certain degree of randomness when producing the results. Thus it suggests the presence of significant confounders that might be unavailable for the likelihood model $p\left(\boldsymbol{x}|\boldsymbol{y}\right)$, implying a possible need to represent these uncertainties with the latent variables. Therefore, I plan to represent the likelihood model using a Conditional Variational Autoencoder (CVAE) [Sohn et al., 2015], an accommodation of Variational Autoencoders (VAE) [Kingma and Welling, 2022] for

a supervised setting. Specifically, instead of training $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ directly by minimizing its NLL under the data, with the introduction of the latent variable $\boldsymbol{v}$, its variational lower bound is maximized instead

$$\log p_\theta(\boldsymbol{x}|\boldsymbol{y}) \geq -KL\left(q_\psi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y})\,||p_\theta(\boldsymbol{v}|\boldsymbol{y})\right) + \mathbb{E}_{q_\psi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{v})\right],$$

where inference density $q_\psi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y})$, prior $p_\theta(\boldsymbol{v}|\boldsymbol{y})$, and $p_\theta(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{v})$ are represented by deep learning networks, similarly as presented by Sohn et al. [2015]. Then, the likelihood density $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ can be retrieved from trained models by marginalizing $\boldsymbol{v}$ with methods like Monte Carlo or importance sampling. Therefore, the described approach introduces multimodality in the representation of $p_\theta(\boldsymbol{x}|\boldsymbol{y})$, which might further improve the quality of posterior samples.

# 2 Background

## 2.1 Residual Neural Networks

The 2010s marked a significant breakthrough in the machine learning (ML) field, particularly in the area of deep learning (DL). Thanks to advances in computing power, availability of large datasets, and improvements in optimization techniques, neural networks (NN) became more powerful and easier to train, and along with the complexity and depth of NN architectures, expanded the range and difficulty of the tasks they could solve. This increase in the model's complexity sparked new issues like overfitting and vanishing/exploding gradients, which gave birth to different regularization techniques and methods, such as batch normalization. However, only the introduction of Residual Neural Networks (ResNets) [He et al., 2015] marked the breakthrough in developing extremely large neural networks (over 100 layers), aiming to solve the issue referred to by authors as the degradation problem.

The name refers to a phenomenon where increasing the depth of NN degrades the prediction accuracy, and importantly, it is not caused by overfitting. As the authors point out, the problem seems counterintuitive at first, considering that by construction, any deeper models should perform at least as well as their more shallow counterparts since the additional layer could learn an identity mapping at worst. Thus the primary objective of the ResNet architecture is to ease the process of finding these identity mappings by optimization algorithms during training. Notably, this approach does not increase the parameter space of the model and can be applied to most of the standard DNN architectures, both linear and convolutional ones, and the construction is as follows. Let $\mathcal{H}(\boldsymbol{x})$ denote the mapping that the model is learning for a given input $\boldsymbol{x}$, where $\mathcal{H}$ can represent any subset of non-linear layers of the NN. The authors propose the model to learn a residual mapping instead $\mathcal{F}(\boldsymbol{x}) = \mathcal{H}(\boldsymbol{x}) - \boldsymbol{x}$, meaning that the original $\mathcal{H}(\boldsymbol{x})$ is transformed into $\mathcal{F}(\boldsymbol{x}) + \boldsymbol{x}$ (Figure 3). This forms the basic block of the ResNet architecture, which can further be stacked on top of each other, constituting a significantly deep network. In case $\mathcal{F}$ changes the dimensionality of $\boldsymbol{x}$, one can accommodate it by replacing the identity mapping with a simple projection, such as a linear one

$$\mathcal{H}(\boldsymbol{x}) = \mathcal{F}(\boldsymbol{x}) + \boldsymbol{W}\boldsymbol{x} \tag{1}$$

using a trainable square matrix $\boldsymbol{W}$.

Even though strikingly simple, it has been shown by multiple experimental results that ResNets easily outperform their corresponding non-residual models on multiple standard optimization tasks/datasets. To motivate the success, the authors hypothesize that learning identity mappings is easier in the case of ResNets, as it only requires pushing the weights of the residual function $\mathcal{F}$ to zero. In contrast, the regular models struggle with learning identity mapping through a sequence of nonlinear layers, which renders the training of ResNets

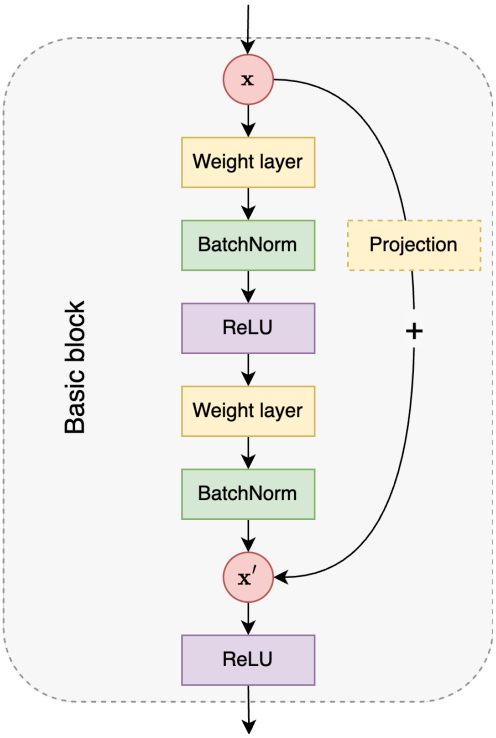



Figure 3: Example structure of the basic block of ResNet architecture. The composition of the block's subcomponents is the same as presented in the original ResNet paper [He et al., 2015]. The projection block with dashed borders designates the projection used whenever the dimensionality of the input changes after passing through the network.

comparatively smoother. Further, even though it is unlikely that the identity mappings are an optimal solution, it might be easier to train the model by making updates in reference to input $\boldsymbol{x}$ than to learn an entirely new mapping. The generalization capabilities of ResNets have been further explored in formalized fashion by He et al. [2019] and Chen et al. [2019], solidifying the motivation behind using these networks. Since then, ResNet architecture has become a cornerstone of the DL field, and residual blocks were incorporated into other state-of-the-art architectures like recurrent networks [Prakash et al., 2016] and transformers [Vaswani et al., 2017]. Thus, residual neural networks are often a default choice in modern ML applications.

## 2.2 Markov Chain Monte Carlo

Since posterior density $p(\boldsymbol{y}|\boldsymbol{x})$ in Bayesian inference is intractable due to the integral in the evidence $p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{y})\, p(\boldsymbol{y})\, d\boldsymbol{y}$, inferring any of its properties is a challenging task. One way of dealing with this problem is to generate multiple samples from $p(\boldsymbol{y}|\boldsymbol{x})$ to get a Monte Carlo approximation

of the desired characteristics of the posterior. Unfortunately, with complex $p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x}|\boldsymbol{y})\, p(\boldsymbol{y})$ and missing closed form for evidence, sampling directly from $p(\boldsymbol{y}|\boldsymbol{x})$ is impossible. Often, the only accessible information is the probability density function (PDF) of joint $p(\boldsymbol{x}, \boldsymbol{y})$, which puts forward the following idea: Since $p(\boldsymbol{x})$ for a fixed $\boldsymbol{x}$ is considered a normalization constant, we could generate a path or sequence of samples in a way such, that each sample in the series appears with the same frequency as when sampled from the posterior, where most of its volume resides. This aim is modeled particularly well by Markov chains. We can treat each sample in the sequence as a state, and by adequately setting the transition probabilities so they align with the posterior's PDF, we could efficiently traverse its volume, giving a result close to one obtained by directly sampling from $p(\boldsymbol{y}|\boldsymbol{x})$.

The above description is a foundation of Markov Chain Monte Carlo (MCMC) algorithms that compute Monte Carlo estimates of expectations using samples obtained from Markov chains. MCMC algorithms aim to generate a Markov chain having a stationary distribution matching the target distribution of interest, and they mainly differ in how they propose new states and evaluate the probabilities of transitioning to those states. Nevertheless, most of these algorithms are highly complex, often based on advanced mathematical fields such as differential geometry. Since, in this work, they are used mainly as a tool to obtain the final results and are not the focal point of the project, in the following, I introduce the key aspects shared by most MCMC algorithms, which are required to properly apply them, along with the diagnostic approaches enabling evaluation of the obtained results.

### 2.2.1 Fundamentals of MCMC

As previously introduced, the foundation of MCMC is based on picturing that the target distribution $p(\boldsymbol{\theta})$, from which we want to sample $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$, is the stationary distribution $\pi(\boldsymbol{\theta})$ of a given Markov chain and then implicitly constructing that chain as the algorithm runs. Different variants of MCMC methods have different ways of ensuring that $\pi(\boldsymbol{\theta})$, in fact, will asymptotically converge to $p(\boldsymbol{\theta})$. Considering the scope of this thesis, I focus on the principles governed by some of the most common MCMC algorithms, including Metropolis-Hastings [Metropolis et al., 1953] and Hamiltonian Monte Carlo (HMC) [Neal, 2010], which also apply to the MCMC method used in this work. Let $P(\boldsymbol{\theta}'|\boldsymbol{\theta})$ denote the probability of transitioning from state $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$ in the induced Markov chain. Foremost, we need to guarantee that the unique stationary distribution exists, which would hold provided the chain is ergodic, meaning it is aperiodic and positive recurrent. To suffice the first, we can ensure that $\forall_{\boldsymbol{\theta}}\, P(\boldsymbol{\theta}|\boldsymbol{\theta}) > 0$, while ensuring that the chain is positive recurrent can be done by enforcing that the probability of transitioning from any given state to any other is greater than zero. Then, a sufficient but not necessary condition for making sure that the Markov process will eventually converge

to $\pi(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$ is the so-called detailed balance

$$\pi(\boldsymbol{\theta}) P\left(\boldsymbol{\theta}'|\boldsymbol{\theta}\right) = \pi\left(\boldsymbol{\theta}'\right) P\left(\boldsymbol{\theta}|\boldsymbol{\theta}'\right),$$

meaning that the transitions are reversible.

To account for that, let us decouple $P(\boldsymbol{\theta}'|\boldsymbol{\theta})$ into a product of two densities

$$P\left(\boldsymbol{\theta}'|\boldsymbol{\theta}\right) = q\left(\boldsymbol{\theta}'|\boldsymbol{\theta}\right) \alpha\left(\boldsymbol{\theta}'|\boldsymbol{\theta}\right),$$

where the proposal distribution $q(\boldsymbol{\theta}'|\boldsymbol{\theta})$ marks the likelihood of proposing a transition from state $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$, and the acceptance distribution $\alpha(\boldsymbol{\theta}'|\boldsymbol{\theta})$ is the probability of accepting or rejecting the proposal of $q$. From this basic framework, different MCMC approaches take different paths in defining $q$ and $\alpha$. For example, in Metropolis-Hastings, the detailed balance is rewritten as

$$\frac{\alpha(\boldsymbol{\theta}'|\boldsymbol{\theta})}{\alpha(\boldsymbol{\theta}|\boldsymbol{\theta}')} = \frac{p(\boldsymbol{\theta}')}{p(\boldsymbol{\theta})} \frac{q(\boldsymbol{\theta}|\boldsymbol{\theta}')}{q(\boldsymbol{\theta}'|\boldsymbol{\theta})},$$

subsequently reformulated to get a closed-form expression for $\alpha$

$$\alpha\left(\boldsymbol{\theta}'|\boldsymbol{\theta}\right) = \min\left\{1, \frac{p(\boldsymbol{\theta}')}{p(\boldsymbol{\theta})} \frac{q(\boldsymbol{\theta}|\boldsymbol{\theta}')}{q(\boldsymbol{\theta}'|\boldsymbol{\theta})}\right\}, \tag{2}$$

where either $\alpha(\boldsymbol{\theta}'|\boldsymbol{\theta}) = 1$ or $\alpha(\boldsymbol{\theta}|\boldsymbol{\theta}') = 1$. Returning to the initial problem of sampling from intractable posterior in Bayesian inference, we can see that MCMC can do so without evaluating the normalization constant, as it cancels out in Equation (2). However, the success heavily depends on the choice of distribution generating the proposals since if it differs substantially from the target distribution, most of the samples will be rejected, hindering the method's efficiency. This introduces us to the first important aspect of MCMC algorithms, the acceptance ratio, which measures the fraction of accepted proposals compared to all generated. Its value can pose as a first diagnostic tool of sampling's performance since if it is close to one, it can mean that the parameter space is traversed within a small range, leaving other regions explored inadequately. On the other hand, as described above, a too-low acceptance rate may indicate that the newly proposed samples fall too far from the target density's typical set[1], which can result in a slow convergence of the stationary distribution of the underlying Markov chain.

After choosing the proper $q$, the next fundamental aspect influencing the algorithm's performance is the initial starting point of generating the Markov chain. If the initial state is far from the target density, it may take many sampling steps until MCMC starts exploring the typical set of $p(\boldsymbol{\theta})$, and the initial samples might be of low quality. To mitigate this, typically, MCMC sampling starts with a warm-up/burn-in phase, which is not much different

[1] the set of $\boldsymbol{\theta}$ for which the likelihood of the target density is close to its expected value

from the regular operation of the method, except that any generated samples are discarded. This allows the chain to get closer to the space where most of the distribution's mass resides before obtaining the samples, improving the final quality of the result. Also, during that phase, the adaptation of method-specific parameters takes place, after which they remain fixed, so increasing the number of warm-up steps may also improve the accuracy of their estimates. Still, there are no ways to decisively determine the moment when the stationary distribution of the generated chain converges to $p(\boldsymbol{\theta})$. Therefore, results require an extensive evaluation to decide their quality.

### 2.2.2 MCMC diagnostics

Since every $\boldsymbol{\theta}$ generated by MCMC is determined with the use of some previous sample, there is a risk that they might be heavily correlated, which means that in practice, only a few $\boldsymbol{\theta}$ out of the total number of retained samples can be considered as a proper representation of the target $p(\boldsymbol{\theta})$. Therefore, upon completion of the MCMC run, two aspects need to be assessed: the level of correlation between the samples and whether the sampling process has reached a stable state with the desired stationary distribution adequately explored. An approach for the evaluation of these issues was presented by Vehtari et al. [2021], which measures the quality of the Markov chain using a set of theoretically derived metrics.

The first precondition that helps to solidify the result's assessment is to run multiple independent MCMC chains, ideally starting from various positions. Then, comparing the results across chains allows for assessing convergence: if chains seem to have converged to different stationary distributions, it shows that each chain covered only parts of the density of interest, meaning the search must be extended. However, it is crucial to note that solely the convergence of multiple chains does not guarantee that sampling was successful, as they might have only explored the same small subset of the typical set. Vehtari et al. [2021] propose the rank-normalized $\hat{R}$ statistic to represent the convergence of chains with a single value metric, which measures the ratio of within-chain and between-chain variances of the sampled values. If the chains have mixed well, both variance estimates should be the same, resulting in $\hat{R}$ equal to 1. Otherwise, its value is greater than 1, suggesting possible convergence issues. Next, to estimate the level of correlation between the samples, one can use the estimate of effective sample size (ESS). It divides the total number of samples from all chains by an intricate estimate of autocorrelation, which attempts to represent the number of independent simulation samples from the target density. The convergence of the MCMC chains might differ across the mode and the tails of the posterior density. Thus, apart from calculating ESS with all samples (ESS-bulk), Vehtari et al. [2021] also advise computing it at the estimates of tail quantiles, obtaining a metric marked as ESS-tail.

Based on the experimental results, Vehtari et al. [2021] advise that to achieve meaningful diagnostics, MCMC should be run with at least four independent chains, and for that setting, as a minimum requirement, one should expect $\hat{R} < 1.01$, ESS-bulk and ESS-tail of at least 400, and per-chain ESS of at least 50. However, it is necessary to note that the recommendations for ESS are a rule of thumb based upon the fact that for multiple problems and families of distributions, many quantities of interest can be approximated well with 400 samples, which does not necessarily have to apply to all of them.

## 2.3 Representing uncertainties with latent space

ML tasks usually are hindered by incomplete or noisy data and unobserved factors, meaning the found solutions are almost always flawed to a certain degree, as models try to give the best guess out of multiple possible ones. Still, single-prediction models have seen much success, reducing this error to acceptable limits, often surpassing all other methods. However, relying solely on a single prediction may be too simplistic sometimes, especially in cases where we observe a one-to-many relationship between the inputs $\boldsymbol{\theta}$ and outputs $\boldsymbol{y}$ of the model. One approach to tackle this issue is to train models predicting the uncertainties of returned $\boldsymbol{y}$, represented by some distribution, such as multivariate normal. Nevertheless, even this can be insufficient, for example, when the true distribution of $\boldsymbol{y}$ is multimodal. Therefore, as described in the Introduction section, considering the limitations and inherent randomness of the simulation generating the dataset, I aim to evaluate if it is the case in this project. For that purpose, as an architecture capable of generating these one-to-many mappings, I chose Conditional Variational Autoencoder (CVAE) [Sohn et al., 2015], a deep conditional generative model that represents the aforementioned confounding variables in the auxiliary latent distribution, that by sampling from it is capable of generating multiple predictions for a given input value. Since the mathematical foundation of CVAE is directly based on Variational Autoencoders (VAE) [Kingma and Welling, 2022], a class of probabilistic generative models for unsupervised learning, in the following, I start with the introduction of concepts motivating VAE before describing CVAEs in more detail.

### 2.3.1 Variational Autoencoders

VAEs introduce a probabilistic perspective to the class of deep generative models, and they aim to learn the distribution of the dataset $p(\boldsymbol{x})$ through the auxiliary latent density $p(\boldsymbol{v})$, creating a continuous representation of the data within it. Specifically, the likelihood of the dataset (also referred to as evidence) can be rewritten as

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{v})\, d\boldsymbol{v} = \int p(\boldsymbol{x}|\boldsymbol{v})\, p(\boldsymbol{v})\, d\boldsymbol{v}, \tag{3}$$

which introduces the first two central components VAEs: prior $p(\boldsymbol{v})$, which in the standard form is a fixed distribution, such as Gaussian $p(\boldsymbol{v}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and decoder $p_\theta(\boldsymbol{x}|\boldsymbol{v})$ represented by a deep neural network (DNN) parametrized by $\theta$, which given a sample from a prior $p(\boldsymbol{v})$ reconstructs $\boldsymbol{x}$.

With this formulation, it becomes apparent that training the decoder by directly maximizing the $p_\theta(\boldsymbol{x})$ is unfeasible due to the intractable integrals in Equation (3). Thus VAEs introduce a third component – encoder $q_\phi(\boldsymbol{v}|\boldsymbol{x})$ – which, as $p_\theta(\boldsymbol{x}|\boldsymbol{v})$, is a DNN parametrized by $\phi$, whose objective is to approximate the intractable posterior $p_\theta(\boldsymbol{v}|\boldsymbol{x}) \approx q_\phi(\boldsymbol{v}|\boldsymbol{x})$. Then, the log evidence can be written as

$$\begin{aligned}
\log p_\theta(\boldsymbol{x}) &= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x})\right] && \text{(4a)}\\
&= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{v})}{p_\theta(\boldsymbol{v}|\boldsymbol{x})}\right]\right] && \text{(4b)}\\
&= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{v})}{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\frac{q_\phi(\boldsymbol{v}|\boldsymbol{x})}{p_\theta(\boldsymbol{v}|\boldsymbol{x})}\right]\right] && \text{(4c)}\\
&= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{v})}{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\right]\right] + \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{q_\phi(\boldsymbol{v}|\boldsymbol{x})}{p_\theta(\boldsymbol{v}|\boldsymbol{x})}\right]\right] && \text{(4d)}\\
&= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{v})}{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\right]\right] + D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p_\theta(\boldsymbol{v}|\boldsymbol{x})\right), && \text{(4e)}
\end{aligned}$$

where the second term in the last equation is the Kullback-Leibler (KL) divergence between $q_\phi(\boldsymbol{v}|\boldsymbol{x})$ and $p_\theta(\boldsymbol{v}|\boldsymbol{x})$, while the first is referred to as Evidence Lower Bound (ELBO), since $D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p_\theta(\boldsymbol{v}|\boldsymbol{x})\right) \geq 0$. Thus, we can see that $D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p_\theta(\boldsymbol{v}|\boldsymbol{x})\right)$ measures two quantities: by definition, the divergence between approximate and true posterior, but also, the gap between the ELBO and true log-evidence value. Since the posterior is intractable, only ELBO constitutes the loss function, and if $D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p(\boldsymbol{v})\right)$ of $q_\phi(\boldsymbol{v}|\boldsymbol{x})$ and $p(\boldsymbol{v})$ has an analytical closed-form expression, then ELBO can be rewritten as

$$\begin{aligned}
\mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{v})}{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\right]\right] &= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{v})\right] - \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log\left[\frac{q_\phi(\boldsymbol{v}|\boldsymbol{x})}{p(\boldsymbol{v})}\right]\right]\\
&= \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{v})\right] - D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p(\boldsymbol{v})\right),
\end{aligned}$$

and by approximating $\mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{v})\right]$ with $L$ Monte Carlo samples $\boldsymbol{v} \sim q_\phi(\boldsymbol{v}|\boldsymbol{x})$, we obtain the final form of the loss function

$$\widetilde{\mathcal{L}}_{\text{VAE}}(\boldsymbol{x};\theta,\phi) = -D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x})\,||p(\boldsymbol{v})\right) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{x}|\boldsymbol{v}^{(l)}\right).$$

Therefore, by looking at equation (4e), we can see that maximization of $\widetilde{\mathcal{L}}_{\text{VAE}}(\boldsymbol{x};\theta,\phi)$ will simultaneously maximize the log evidence and minimize the KL divergence between $q_\phi$ and the true posterior.

To finalize the principal formulation of the VAE architecture, it is necessary to present another essential proposition of Kingma and Welling [2022], which stems from the fact that the approximation of $\mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x})}\left[\log p_\theta\left(\boldsymbol{x}|\boldsymbol{v}\right)\right]$ with $\boldsymbol{v}$ sampled from $q_\phi\left(\boldsymbol{v}|\boldsymbol{x}\right)$ makes it impossible for gradients to directly backpropagate through the random variable $\boldsymbol{v}$. As a solution, the authors introduced the reparametrization trick, which moves the randomness from $\boldsymbol{v}$ to the newly introduced $\boldsymbol{\epsilon}$. Specifically, instead of generating samples directly through a decoder, they propose to do so with a deterministic differentiable transformation $g_\phi\left(\boldsymbol{\epsilon}, \boldsymbol{x}\right)$, where $\boldsymbol{\epsilon} \sim p\left(\boldsymbol{\epsilon}\right)$ is a noise variable, with $p\left(\boldsymbol{\epsilon}\right)$ usually set as a fixed distribution, such as $\mathcal{N}\left(\mathbf{0}, \boldsymbol{I}\right)$. Any $g_\phi$ that is differentiable mapping of $\boldsymbol{x}$ and $\boldsymbol{\epsilon}$ suffices, and to provide an example, if for a given $i$-th datapoint $\boldsymbol{x}^{(i)}$ decoder predicts the mean and variances of multivariate Gaussian with mutually independent variables

$$q_\phi\left(\boldsymbol{v}|\boldsymbol{x}^{(i)}\right) = \mathcal{N}\left(\boldsymbol{\mu}^{(i)}, \operatorname{diag}\left(\boldsymbol{\sigma}^{2(i)}\right)\right),$$

a valid transformation could be

$$\boldsymbol{v} = g_\phi\left(\boldsymbol{\epsilon}, \boldsymbol{x}^{(i)}\right) = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{2(i)} \odot \boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim p\left(\boldsymbol{\epsilon}\right).$$

Then, the mapping is deterministic and differentiable from the perspective of the decoder's parameters, while the randomness is retained through the samples from $p\left(\boldsymbol{\epsilon}\right)$.

### 2.3.2 Conditional Variational Autoencoders

The CVAE architecture follows the same core principles as in the case of VAEs. However, since it is suited for supervised learning, the generative process is now done for the labels $\boldsymbol{y}$, which introduces additional conditioning on inputs $\boldsymbol{x}$ alongside the latent variables $\boldsymbol{v}$. Thus, the decoder becomes $p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}\right)$, the encoder is defined as $q_\phi\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)$, and the prior is now trainable, as it is conditioned on $p_\theta\left(\boldsymbol{v}|\boldsymbol{x}\right)$. Yet, this requirement can be relaxed by assuming the independence of $\boldsymbol{v}$ from $\boldsymbol{x}$, giving the fixed a prior $p\left(\boldsymbol{v}\right)$ as before. Then, by applying the same logic as in derivations of equation (4), we arrive at

$$\log p_\theta\left(\boldsymbol{y}|\boldsymbol{x}\right) = \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y})}\left[\log\left[\frac{p_\theta\left(\boldsymbol{y}, \boldsymbol{v}|\boldsymbol{x}\right)}{q_\phi\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)}\right]\right] + D_{KL}\left(q_\phi\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)||p_\theta\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)\right),$$

implying the following reformulation of ELBO

$$\log p_\theta\left(\boldsymbol{y}|\boldsymbol{x}\right) \geq \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y})}\left[\log p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}\right)\right] - D_{KL}\left(q_\phi\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)||p\left(\boldsymbol{v}|\boldsymbol{x}\right)\right),$$

and the loss function

$$\widetilde{\mathcal{L}}_{\text{CVAE}}\left(\boldsymbol{x}; \theta, \phi\right) = -D_{KL}\left(q_\phi\left(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}\right)||p_\theta\left(\boldsymbol{v}, |\boldsymbol{x}\right)\right) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^{(l)}\right),$$

where $\boldsymbol{v}^{(l)} \sim q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y})$, which requires the reparametrization trick like in the case of VAEs. Thus, the basis of CVAE is again formed by three NNs: prior $p_\theta(\boldsymbol{v}|\boldsymbol{x})$, recognition network $q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y})$, and generation network $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{y})$, that from the latent samples produces the distribution of outputs $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v})$. Sohn et al. [2015] in the paper introduce the fourth network, on top of which the forenamed NNs are built, but since I adapt their approach to my case, the details of the architecture are described in the Methods section.

When the model is trained, the predictions can be inferred in multiple ways, for example, with deterministic inference

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^*), \ \boldsymbol{v}^* = \mathbb{E}_{p_\theta(\boldsymbol{v}|\boldsymbol{x})}[\boldsymbol{v}]$$

or by averaging the posteriors

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} \frac{1}{L} \sum_{l=1}^{L} p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^{(l)}\right), \ \boldsymbol{v}^{(l)} \sim p_\theta(\boldsymbol{v}|\boldsymbol{x}).$$

The final important aspect of CVAEs from the perspective of this thesis is the evaluation of the conditional $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ required for MCMC sampling. Two straightforward ways to evaluate this quantity are Monte Carlo sampling

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) \approx \frac{1}{S} \sum_{s=1}^{S} p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^{(s)}\right), \text{ where } \boldsymbol{v}^{(s)} \sim p_\theta(\boldsymbol{v}|\boldsymbol{x})$$

or, more efficiently, with importance sampling using the encoder

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) \approx \frac{1}{S} \sum_{s=1}^{S} \approx \frac{p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^{(s)}\right) p_\theta\left(\boldsymbol{v}^{(s)}|\boldsymbol{x}\right)}{q_\phi\left(\boldsymbol{v}^{(s)}|\boldsymbol{x}, \boldsymbol{y}\right)}, \text{ where } \boldsymbol{z}^{(s)} \sim q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}).$$

# 3 Dataset

In this thesis, I incorporate an artificially generated dataset in a similar way as done by Ansari et al. [2022]. Specifically, the data was generated using fully three-dimensional photoionization and dust radiative transfer code `MOCASSIN` [Ercolano et al., 2006], which simulated the spectral energy distributions (SEDs) of different core-collapse supernovae (CCSNe) and their surrounding dust heated by the explosion. Then, the SEDs were processed to simulate how they would be observed with JWST placed at different distances from the event, which resulted in the dataset of possible JWST observations of dust surrounding the CCSNe. To be more precise, each data point consists of:

1. Observations $\boldsymbol{x} \in \mathcal{X} = \mathbb{R}^{37}$ of 37 filters available in JWST, where each filter measures the magnitude of infrared (IR) radiation at different wavelengths.

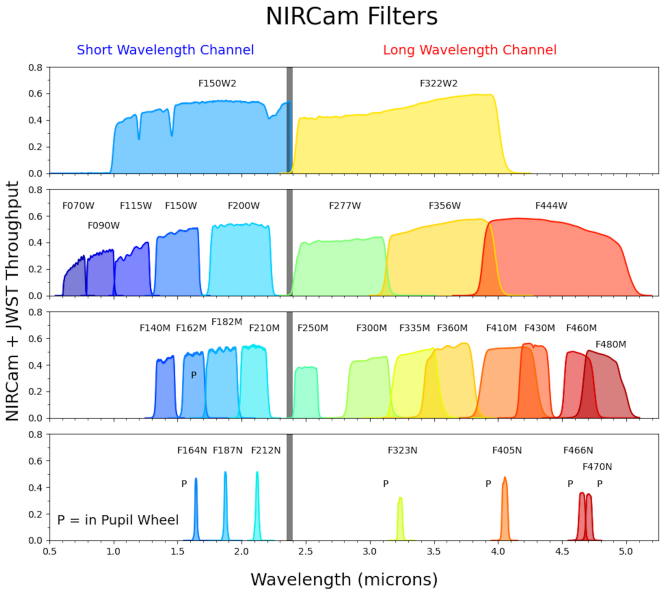
2. Features $\boldsymbol{y} \in \mathcal{Y} = \mathbb{R}^{7}$, consisting of five parameters describing the dust and two used to parametrize CCSNe.

3. The redshift constant $z \in \mathbb{R}$, indirectly measuring how distant the CCSNe is from the observer.

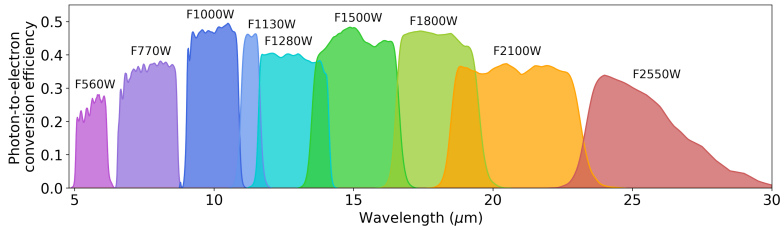
The following section provides a detailed description of each element listed above. Additionally, the last subsection outlines the data preprocessing measures taken for all experiments conducted in this project.

## 3.1 JWST filters

The James Webb Space Telescope (JWST) was specifically designed to observe the infrared (IR) portion of the electromagnetic spectrum, and thus, it is equipped with a wide range of instruments designed to detect light at these longer wavelengths. The dataset used in this thesis simulates observations from two main groups of these instruments: twenty-eight Near Infrared Camera (NIRCam) filters and nine Mid-Infrared Instrument (MIRI) filters. Each filter covers a different wavelength range, which is reflected in the following naming convention: each name starts with the letter 'F' standing for a filter, followed by an integer denoting the central wavelength of the filter in nanometers, ending with a single letter marking whether the filter covers narrow (N), medium (M), wide (W) or extra-wide (W2) range of the spectrum. Figure 4 shows a visualization of those filters, where graphs were taken from the online JWST User Documentation [Space Telescope Science Institute, 2023a]. It is expected that MIRI filters would capture more of the space dust's electromagnetic radiation (EMR) since the dust is much colder when compared to the temperature of a supernova (SN), and thus its wavelengths are shorter.




(a) NIRCam filters. Figure taken from Space Telescope Science Institute [2023c].




(b) MIRI filters. Figure taken from Space Telescope Science Institute [2023b].

Figure 4: Depiction of NIRCam and MIRI bandpass filters available in JWST with wavelength ranges that they cover and their throughputs that measure the photon-to-electron conversion efficiency. All filters above were used to generate the dataset, except for *F150W2*.

Therefore, it also means that the emissions of SN would be more visible in the NIRCam filters.

Each filter was convolved with the SEDs simulated using `MOCASSIN` to generate the dataset, resulting in a single value per filter – the synthesized magnitude of the covered wavelength. In Figure 5, I present the basic aggregate statistics of observed magnitudes. For most of the filters, about 90% of the values fall within the range of 10-30, and most of the outliers in the lower end span a relatively small range of 10-20, while on the higher end, the values often fall as high as 60. What is immediately alarming is the abnormal be-

havior of filters *F164N*, *F323N*, and *F466N*. In the case of *F164N* and *F466N*, more than 90% have an extremely high AB magnitude of 60, and even though values of F323N span a broader range, the median still falls at the higher end. As AB magnitude inversely relates to the energy of IR radiation captured by the telescope, it suggests that these filters, and generally outliers in other cases, captured weak signals. As a result of constraints on the simulation's runtime and memory usage, it is possible to generate only a finite and incomplete number of photons. Thus, considering that all abnormal filters are narrow, they might be especially susceptible to missing information, which poses a possible explanation for observed deviations. This issue is also related to the redshift constant $z$, described in the The redshift constant section.

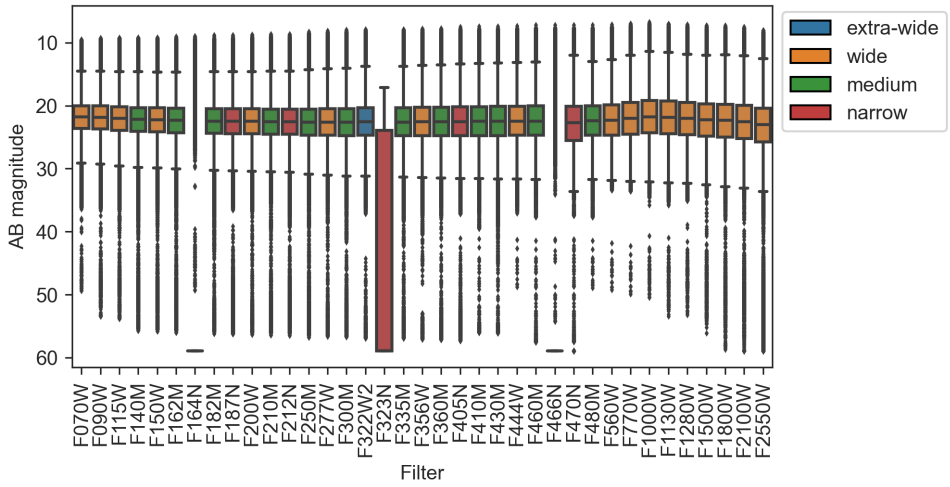



Figure 5: Box plot of per-filter magnitude distributions. Boxes span between the ends of the first and third quartile, while whiskers extend to 1.5 of the inner quartile range in both ends or up to maximal/minimal value. The solid line inside the box denotes the median, while the diamond-shaped marks represent the outliers calculated by the `boxplot` function from the seaborn (v0.12.2) [Waskom, 2021] library, which was used to generate this plot. The coloration of the boxes indicates the width of the wavelength range that the specific filter covers.

## 3.2 Dust and supernovae

The dataset contains five parameters describing the dust's properties, but four of them are of main interest. The first three characterize the fundamental physical properties of the dust – its mass, temperature, and grain size – while the last one describes the grain specie. The most common elements found in the surroundings of the stars are oxygen and carbon, each producing one

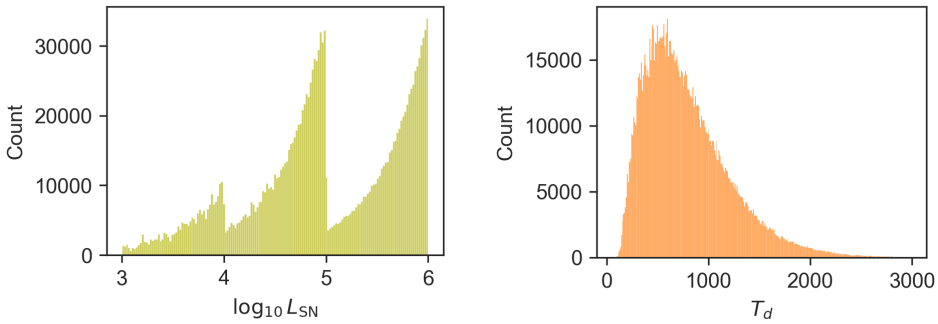
of the two main species of dust, carbonaceous dust and silicates, respectively [Gall et al., 2011]. Therefore, the simulation considered the dust consisting of either 100% carbon, 100% silicates, or a mixture of the two dust species with varying ratios. In the dataset, the dust specie is represented as a fraction of silicates composing the measured dust.

As a difference to the dataset of Ansari et al. [2022], the dataset generated for thesis included an additional dust parameter, the clump filling factor. It refers to the fraction of space occupied by dust clumps or dense regions within a given volume, essentially measuring whether dust is distributed uniformly around the SNe or if it condensates more around certain areas, leaving others more sparse. The motivation behind including this parameter in the dataset was that even though it is not crucial from the viewpoint of final prediction, it may contain valuable information for the Bayesian inference model predicting $\boldsymbol{x}$ from $\boldsymbol{y}$. For the same reason, I also included two features describing the simulated SNe – their luminosities and temperatures. Solar luminosity measures power emitted by celestial objects in the form of photons that, in the case of our simulation, heat up the surrounding dust, which causes the emission of the infrared radiation detected by JWST. Table 1 summarizes basic information about all $\boldsymbol{y}$ features described above, along with notation used further in the thesis.

| Notation | Range | Unit | Description |
|---|---|---|---|
| $M_d$ | $10^{-6} - 10^{-1}$ | $M_\odot$ (solar mass) | Dust mass |
| $S_g$ | $0.005 - 5$ | µm (micrometer) | Grain size |
| $F_s$ | $0 - 1$ | - | Silicate fraction |
| $T_d$ | $65 - 2995$ | K (kelvin) | Dust temperature |
| $C_f$ | $0 - 0.5$ | - | Clump filling factor |
| $L_{\text{SN}}$ | $10^3 - 10^6$ | $L_\odot$ (solar luminosity) | SN luminosity |
| $T_{\text{SN}}$ | $(4 - 14) \times 10^3$ | K (kelvin) | SN temperature |

Table 1: Variables constituting the feature space $\mathcal{Y}$.

All of these features span uniformly the specified ranges (where mass and grain sizes uniformly span their ranges on logarithmic scale), except for dust's temperature and SN luminosity. As Figure 6a shows, $L_{\text{SN}} \in \left[10^4, 10^5\right]$ and $L_{\text{SN}} \in \left[10^5, 10^6\right]$ contain approximately the same number of samples, while $L_{\text{SN}} \in \left[10^3, 10^4\right]$ is underrepresented by a factor of $\approx 3.3$, and they all span these ranges uniformly. In the case of $T_d$, in Figure 6b, we can see that from $\approx$500 K, the number of samples starts decreasing due to the sublimation of the heated dust.




(a) (log-10) SN luminosity. (b) Dust temperature.

Figure 6: Distributions of dust temperature $T_d$ and log SN luminosity $\log_{10}(L_{\text{SN}})$ across the training data.

### 3.3 The redshift constant

In astrophysics, redshift (commonly denoted as $z$) refers to stretching or shifting the waves of electromagnetic radiation (EMR) toward longer wavelengths. When applied to visible light, the wavelengths are moved towards the red end of the electromagnetic spectrum, thus earning its name. Multiple factors can cause the redshift of ERM traveling through the universe, but the primary one, also embodied in this thesis, is the cosmological redshift caused by the expansion of the universe. As galaxies move away from each other, the light they emit gets stretched out during their journey through expanding space, resulting in a redshift. The greater the distance of a radiation source, the more pronounced the redshift of emitted EMR tends to be, providing a way to estimate its distance from the observer. Therefore, samples in the dataset with higher values of $z$ not only have their SEDs shifted towards longer wavelengths (although the visible effect is relatively small), but also the magnitudes of the signal are decreased, since simultaneously, the larger the distance, the fainter the EMR.

Figure 7 showcases how the redshift influences the signal captured by JWST filters, where we can see both of the forenamed effects on the filter responses: change in the magnitudes is clearly visible, while the shift towards longer wavelengths can explain the abnormal behavior of the three narrow-band filters *F164N*, *F323N*, and *F466N*. It goes back to the discussion in the JWST filters section, where we could observe the abnormal behavior of these filters because they cover a very narrow range of the electromagnetic spectrum compared to other filters. Here, SEDs get shifted, resulting in the different coverage of photons, producing very different results depending on the $z$ value.

Multiple redshift values have been applied to each SED generated by the `MOCASSIN`, meaning that each $\boldsymbol{y}$ occurs numerous times in the dataset with different variants of $\boldsymbol{x}$ and accompanying $z$. However, not all samples have

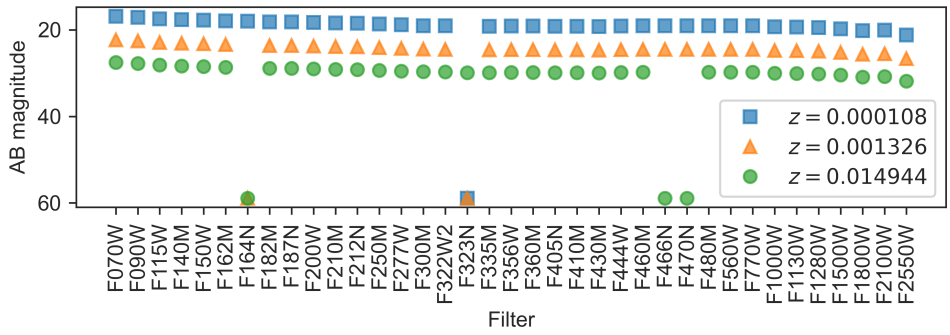
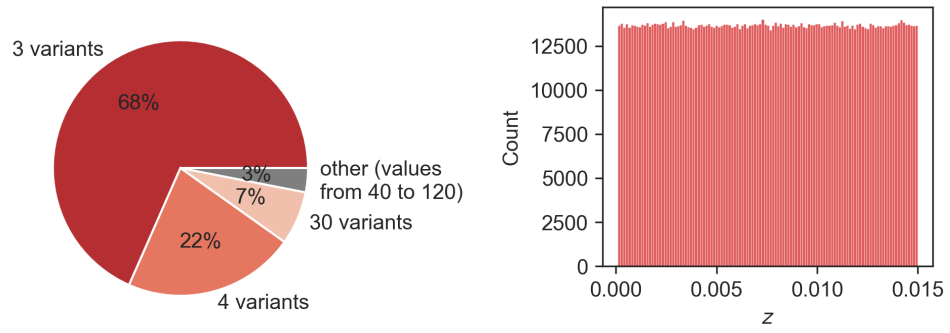



Figure 7: Synthesized magnitudes of JWST filters $\boldsymbol{x}$ of three samples from the dataset with identical $\boldsymbol{y}$ features but at different redshifts $z$.

an equal number of variants (Figure 8a). Part of the reason for this is that resampling with $z$ was used to balance the distribution of SNe luminosities (Figure 6a) since, in the simulation, values of luminosity were picked uniformly, meaning that on log-10 scale values in the range 3-4 were even more underrepresented than they are now when compared to higher exponents. Nonetheless, across the whole dataset, redshift is uniformly distributed in the range 0.0001–0.015 (Figure 8b), where the lower bound is motivated by the fact that there are no observations of extragalactic SN closer than that, while the upper bound is limited by the sensitivity and saturation limits of the JWST filters [Ansari et al., 2022].




(a) Distribution of different counts of the sample's redshift variants.

(b) Distribution of redshift values in the dataset.

Figure 8: Pie chart showing the percentage of samples in the dataset with a specified number of $\boldsymbol{x}$ and $z$ variants, along with the histogram plot presenting a uniform distribution of $z$ in the dataset.

### 3.4 Preprocessing

The total of 2 million samples constituting the dataset is split into training (64%), validation (16%), and test (20%) sets, and using the element-wise mean and variance calculated over the training data, all data points across all subsets are standardized to zero mean and unit variance. However, it is important to note that during dataset generation, the data was divided into the test set and initial training data before creating redshift variants. Then, redshift variants were generated for both, and only after this process were samples from the initial training data split into final training and validation datasets. Consequently, there could be considerable overlap in $\boldsymbol{x}$ samples between the training and validation datasets, only with different redshift variants. Naturally, this poses a limitation. Unfortunately, I identified the issue too late in the thesis project to re-run all the experiments before the submission deadline.

The subsequent important consideration comes from the fact that each JWST filter has its range of detectable EMR magnitude values, outside of which the signal would either be too bright or too faint to be observed by the telescope. For simplicity, in this thesis, I only focus on the higher end of AB magnitude values by considering a filter response as undetectable if its captured magnitude is greater or equal to 42.5, which is still a mild assumption since, for many filters, this limit is as low as $\approx 30$ [Ansari et al., 2022]. Given this threshold, many of the samples in the dataset would contain missing features, especially in the narrow filters. For example, a substantial 95% of the data points would include undetectable magnitudes in *F164N*. Therefore, I removed all narrow and extra-wide filters, additionally motivated by the fact that their wavelength ranges are already covered by medium and wide filters, as shown in Figure 4. Nevertheless, a considerable portion of the samples contains undetectable magnitudes in the kept filters (specific percentages are shown in Figure 9), which I consider during the training process, elaborated further in the next section.

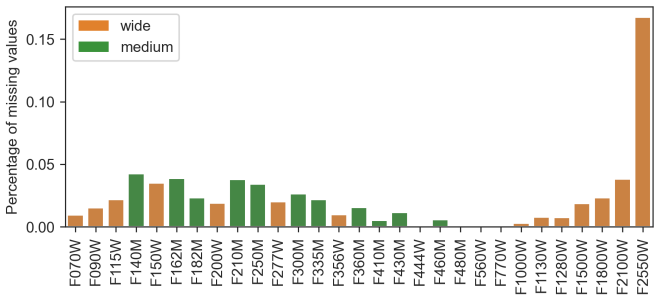



Figure 9: Percent of samples in the dataset with undetectable magnitudes, when assuming that the higher limit is 42.5, calculated for each filter (excluding narrow and extra-wide filters).

# 4 Methods

The central objective of this study is to improve upon existing methodologies for estimating cosmic dust properties based on EMR captured by the JWST. Ansari et al. [2022] showed that applying ML to this task can lead to prominent results aligned with the standard methods used in astrophysics. However, given that the number of JWST filters used to collect the data can vary depending on the task, the presented solution faced some limitations. The models were trained in a standard fashion by directly generating the predictions based on EMR, and since ML models usually require a fixed set of inputs to generate the outputs, the authors could not provide a universal solution that would work for all possible subsets of filters. Moreover, the prediction error for many dust features was too high to expect the model to generate reliable outcomes. Thus, there arises a need for robust estimates of prediction uncertainties. Although the authors address this by training the model to provide prediction variances, it may be too simplistic compared to the actual distribution of dust properties given the EMR. Therefore, this thesis aims to introduce a comprehensive framework that addresses these limitations by integrating Bayesian approach into the problem.

Given the simulated dataset $\mathcal{D} = \{(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}, z^{(1)}), \ldots, (\boldsymbol{x}^{(N)}, \boldsymbol{y}^{(N)}, z^{(N)})\}$, I try to model the distribution $p(\boldsymbol{y}|\boldsymbol{x}, z)$ of space dust properties $\boldsymbol{y}$ conditioned on their corresponding filter magnitudes $\boldsymbol{x}$ and the redshift constant $z$ using the Bayesian inference

$$p(\boldsymbol{y}|\boldsymbol{x}, z) = \frac{p(\boldsymbol{y}, \boldsymbol{x}, z)}{p(\boldsymbol{x}, z)} = \frac{p(\boldsymbol{x}|\boldsymbol{y}, z)\, p(\boldsymbol{y}|z)\, p(z)}{p(\boldsymbol{x}|z)\, p(z)} = \frac{p(\boldsymbol{x}|\boldsymbol{y}, z)\, p(\boldsymbol{y}|z)}{p(\boldsymbol{x}|z)}.$$

Here, the likelihood density $p(\boldsymbol{x}|\boldsymbol{y}, z)$ is represented with a deep neural network while the prior is set to a standard Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, where the condition on $z$ is dropped since dust's properties are independent from the redshift. This choice of $p(\boldsymbol{y})$ is rooted in the observation that $\boldsymbol{y}$ features span almost uniformly their corresponding boundaries in the dataset and because the data has been standardized to zero mean and unit variance, as detailed in the Preprocessing section. Then, I employ the MCMC method, which allows sampling from the posterior $p(\boldsymbol{y}|\boldsymbol{x}, z)$ while avoiding the need to compute the intractable evidence $p(\boldsymbol{x}|z)$. Generating enough samples enables estimation of the intricate posterior density, from which a final prediction can be derived, along with a robust and detailed assessment of prediction uncertainties. Furthermore, if $p(\boldsymbol{x}|\boldsymbol{y}, z)$ is a distribution allowing simple marginalization of $\boldsymbol{x}$ features, such as multivariate Gaussian, one can easily accommodate the method to a variable number of filters.

The description above presents the core idea behind the approach. However, its effectiveness hinges on several critical factors, including the design and training routine of the models, appropriate convergence assessment of the MCMC runs, and thorough evaluation of the results. Therefore, the fol-

lowing sections delve into the specifics of training the models that represent $p(\boldsymbol{x}|\boldsymbol{y}, z)$ (further referred to as likelihood models, following the typical naming convention where the density $p(\boldsymbol{x}|\mathrm{y})$ is referred to as the likelihood in the Bayesian framework), including their architecture, evaluation methods, and experimental setup.

First, I focus on using a regular deep neural network for that task, described in section Likelihood model. Then, I try replacing deep neural network (DNN) with a more expressive model architecture, the CVAE [Sohn et al., 2015], to assess whether it can better capture the inherent randomness of the simulation and account for confounding variables (section Replacing likelihood model with CVAE). Finally, section Sampling from the posterior provides an in-depth overview of the MCMC routine used to generate the final results and the evaluation and diagnostic measures employed.

## 4.1 Likelihood model

Similar to the prior $p(\boldsymbol{y})$, the likelihood model is represented using a multivariate normal distribution. However, for each entry $(\boldsymbol{y}, z)$, the mean $\boldsymbol{\mu}_x$ and covariance matrix $\boldsymbol{\Sigma}_x$ are predicted by deep learning model $\mathcal{M}_\theta$ parameterized by $\theta$:

$$\mathcal{M}_\theta(\boldsymbol{y}, z) \to \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = p_\theta(\boldsymbol{x}|\boldsymbol{y}, z).$$

The network is trained to minimize a negative log-likelihood (NLL) of true samples under the generated distribution. Thus, the objective can be formalized as minimizing

$$\begin{aligned}\mathcal{L}_{\mathrm{LM}}(\mathcal{D}) &= \frac{1}{N}\sum_{i=1}^{N} -\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{y}^{(i)}, z^{(i)}\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\left[d_{\boldsymbol{x}}\log(2\pi) + \log|\boldsymbol{\Sigma}_{\boldsymbol{x}^{(i)}}| + \mathrm{MD}\left(\boldsymbol{x}^{(i)}; \boldsymbol{\Sigma}_{\boldsymbol{x}^{(i)}}, \boldsymbol{\mu}_{\boldsymbol{x}^{(i)}}\right)^2\right],\end{aligned} \tag{5}$$

where $d_{\boldsymbol{x}}$ is the dimensionality of $\boldsymbol{x}$, and $\mathrm{MD}(\boldsymbol{x}; \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mu}_{\boldsymbol{x}})$ is the Mahalanobis distance between sample $\boldsymbol{x}$ and the predicted

$$\mathrm{MD}(\boldsymbol{x}; \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mu}_{\boldsymbol{x}}) = \sqrt{(\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{x}})^T \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{x}})}.$$

The training scheme and all models are implemented using the PyTorch (v1.13) library [Meta AI, 2023a] . Optimization is performed using the Adam [Kingma and Ba, 2014] optimizer with minibatches of data randomly sampled without replacement from the dataset. However, considering that certain samples lack information in specific filters, as explained in the Preprocessing section, I divide the minibatches into subsets, where each shares the same missing filters. Then, for each of them, I calculate the NLL separately by

marginalizing out the missing filters. In the case of the multivariate normal distribution, the process reduces to simply removing the corresponding indexes from the mean vector and rows/columns from the covariance matrix. Then, based on the average NLL over all samples in the minibatch, the weights are updated through backpropagation.

All models presented in this project generate the covariance matrices $\boldsymbol{\Sigma}$ by predicting the lower triangular matrix $\boldsymbol{L}$ of their corresponding Cholesky decomposition $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$. The softplus function parametrized by $\beta$

$$\text{Softplus}\left(\boldsymbol{x}\right) = \frac{1}{\beta}\log\left(1 + e^{(\beta\boldsymbol{x})}\right) \tag{6}$$

is applied to all diagonal elements of $\boldsymbol{L}$, ensuring that reconstructed matrices are positive semidefinite. The choice is motivated by the fact that softplus grows linearly in the positive values, while the rate of decrease towards zero values can be controlled by $\beta$, providing more stable training, for example, when compared to the exponential function. At first, I tried to enforce that the resulting covariance is positive definite by adding a small constant to the diagonal entries of $\boldsymbol{L}$, which resulted in an unstable training process. As it turned out, even though the approach is correct in the analytical sense, it often resulted in eigenvalues small enough to be considered equal to zero from the numerical standpoint. Therefore, to additionally set the lower bound on eigenvalues, in the final implementation, models reconstruct the covariance and perturb the diagonal by adding the minimal variance $\sigma^2_{\texttt{min}}$:

$$\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T + \sigma^2_{\texttt{min}} \cdot \boldsymbol{I}.$$

Therefore, to efficiently compute the log determinant of $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^{-1}$ for loss calculation, the matrix must be decomposed again using Cholesky factorization.

Another issue, which added to the problems with instability of the training process, comes from the fact that fully defining matrix $\boldsymbol{L}$ requires $\left(d^2 + d\right)/2$ values, meaning that the number of predicted parameters grows quadratically with the dimensionality of $\boldsymbol{\mu}$. It is especially problematic in the case of $p_\theta\left(\boldsymbol{x}|\boldsymbol{y}, z\right)$ since we have $d_{\boldsymbol{y}} = 7$ and $d_{\boldsymbol{x}} = 29$, meaning that from 8 values (+1 for $z$), the model needs to predict 435. Therefore, the models may require deep architecture capable of modeling such complex outputs while being resilient to numerical stability issues of predicting properly defined $\boldsymbol{\Sigma}$.

### 4.1.1 Architecture of the models

Considering the above, I chose residual network (ResNet) architecture [He et al., 2015] for all models I evaluate in this thesis based on the advantages described in the Residual Neural Networks section. The model consists of three residual subnetworks – first, the input is passed through the shared network, whose output is further processed by two separate ResNets, where

one predicts $\boldsymbol{\mu}$ and the other models $\boldsymbol{\Sigma}$. Each residual subnetwork consists of $K$ blocks containing two weight layers, each followed by a batch normalization and a ReLU activation function, where right before the last activation shortcut connection is added to the processed data point. Weight layers are represented either by dense connections or one-dimensional convolutions, and experimentally, I verify which combination achieves the best results.

Since there is a mismatch between the dimensionality of the input and the output, samples flowing through the network require rescaling. This rescaling always happens within the residual blocks, and in these cases, the identity mappings are replaced by linear ones or simple $1 \times 1$ convolutions, depending on the variant of the layer used. Figure 10 shows a graphical overview of the architecture, while the specific values of the layers' parameters used in all conducted experiments are described in the Results section.

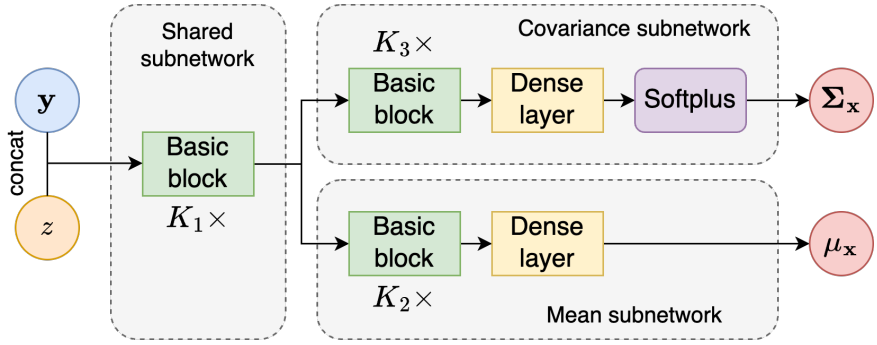



Figure 10: Graph showing the architecture of the likelihood model. Each basic block has an identical inner structure to the one presented in Figure 3. Each $K$ marks the number of times a block is repeated.

### 4.1.2 Evaluation methods

Throughout the training process, I monitor per-epoch loss values (i.e. NLL of the true sample under predicted normal distribution) on the train and validation datasets, along with the mean absolute error (MAE) between predicted $\boldsymbol{x}^* = \boldsymbol{\mu}_{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}, z)$ and sample $\boldsymbol{x}$. Then, I pick the architecture that achieves the lowest values of both metrics on the validation set. Once the best configuration of parameters out of the ones tested is established, I perform a more thorough analysis of the predictions on the test set. A similar setting of parameters is then used in the CVAE models, and the best models are used to sample from the posterior.

I begin the evaluation by taking a closer look at the distributions of loss and error of predicted $\boldsymbol{\mu}_{\boldsymbol{x}}$. To further evaluate the accuracy of the $\boldsymbol{\mu}_{\boldsymbol{x}}$ prediction, I include per-feature root mean squared error (RMSE). The change from MAE to RMSE is motivated by the fact that since the network is trained to optimize NLL, it contains mahalanobis distance, which is squared metric

from the viepoint of mean prediction errro, and thus is more biased towards outliers and RMSE is better depiction of it. MAE was used before becausue I did not realize that, and it was too late to re train all the models with tracking of RMSE instead of MAE.

Then, I proceed to the analysis of predicted covariances. Even though I train $\mathcal{M}_\theta$ to return the multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{x}})$ from which I would sample, it can also be viewed as predicting the uncertainty $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ around the model's best prediction $\boldsymbol{\mu}_{\boldsymbol{x}}$, which is often applied this way in the literature [Gawlikowski et al., 2021]. Naturally, we would want the predicted uncertainty to reflect the probability of the true sample falling in its region – $P\%$ of the true data points should fall in the $P\%$ of the confidence interval of their predicted uncertainties. For the one-dimensional case $\mathcal{Y} = \mathbb{R}$, Kuleshov et al. [2018] formalize this notion for regression task in the following way: given a cumulative density function (CDF) $F_i = \mathbb{P}\left(Y < y^{(i)}\right)$, predicted for the true value $y^{(i)}$, the predictor is calibrated if for the quantile function $F_i^{-1}$ we have

$$\frac{\sum_{i=1}^{N} \mathbb{1}\left\{y^{(i)} < F_i^{-1}(P)\right\}}{N} \to P \text{ for all } P \in [0, 1],$$

as $N$ approaches infinity.

In the case of the multivariate normal distribution, there is no closed form for $F_i$. However, Bensimhoun [2013] derives that if we define $F_i$ as the probability that a sample lies inside the ellipsoid determined by its Mahalanobis distance $(\boldsymbol{y} - \boldsymbol{\mu}_{\boldsymbol{x}})^T \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} (\boldsymbol{y} - \boldsymbol{\mu}_{\boldsymbol{x}})$ from the Gaussian, then it forms a confidence region

$$(\boldsymbol{y} - \boldsymbol{\mu}_{\boldsymbol{x}})^T \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} (\boldsymbol{y} - \boldsymbol{\mu}_{\boldsymbol{x}}) \leq \chi_k^2(P),$$

where $\chi_k^2(P)$ is the quantile function for probability $P$ of the chi-squared distribution with $k$ degrees of freedom. Therefore, I evaluate the uncertainty of the likelihood model $\mathcal{M}_\theta$ with

$$\frac{\sum_{i=1}^{N} \mathbb{1}\left\{\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{\boldsymbol{x}}^{(i)}\right)^T \left(\boldsymbol{\Sigma}_{\boldsymbol{x}}^{(i)}\right)^{-1} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{\boldsymbol{x}}^{(i)}\right) < \chi^2_{\dim(\boldsymbol{x})}(P)\right\}}{N}$$

for $P \in \{0.1, 0.2, \ldots, 0.9\}$, and I visualize it, inspired by Gawlikowski et al. [2021], using a reliability diagram.

Finally, I also present the averaged covariance matrix composed of the mean of all predicted $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ on the test set to evaluate whether the correlations estimated by the model match the real-world relations.

## 4.2 Replacing likelihood model with CVAE

In the initial steps of the project, I ran the models only using five cosmic dust features – mass, grain size, silicate fraction, temperature, and clump filling factor – as they are of primary interest when inferring the properties. However, since the simulation used to generate the dataset utilizes many more

parameters, such as supernovae (SNe) luminosity and temperature, I also ran experiments with these included, which considerably improved the performance of the likelihood model $p_\theta(\boldsymbol{x}|\boldsymbol{y}, z)$. Consequently, it suggests that there are significant cofounders that might be unavailable for the model, and considering that even the simulator used to generate the dataset incorporates a certain degree of randomness when generating the results, there might be a need to represent these uncertainties with the latent variables. To do so, I replace the standard architecture of the likelihood model described above with CVAE, which I train and evaluate both with and without the $\boldsymbol{y}$ features concerning SNe, to validate the concerns raised above. In turn, this introduces the new recognition network $q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}, z)$ and prior $p(\boldsymbol{v})$. I choose to keep the $p(\boldsymbol{v})$ fixed as standard Gaussian $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, effectively dropping the conditioning on $\boldsymbol{x}$.

As presented in the Conditional Variational Autoencoders section, since fully marginalizing out the latent space is unfeasible, the objective of maximizing the likelihood of data $p_\theta(\boldsymbol{x}|\boldsymbol{y}, z)$ is done through its lower bound

$$\log p_\theta(\boldsymbol{x}|\boldsymbol{y}, z) \geq \mathbb{E}_{q_\phi(\boldsymbol{v}|\boldsymbol{x},\boldsymbol{y},z)}\left[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}, z)\right] - D_{KL}\left(q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}, z)\,||p(\boldsymbol{v})\right). \quad (7)$$

The expansion of the KL divergence hinges on the definitions of $q_\phi$ and the prior. In this context, I chose the recognition network to predict both the $\boldsymbol{\mu}$ and variances $\boldsymbol{\sigma}^2$ as parameters of a multivariate Gaussian distribution with mutually independent variables

$$q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}, z) = \mathcal{N}\left(\boldsymbol{\mu}, \operatorname{diag}\left(\boldsymbol{\sigma}^2\right)\right),$$

motivated by the fact that KL divergence between two Gaussians with diagonal covariances has simple closed form consisting of basic algebraic operations between corresponding mean and variance indexes predicted by $q_\phi$ and prior, making it efficiently computable. Therefore, the objective of the training considering the dataset can be formulated as the minimization of the negative of the above lower bound

$$\mathcal{L}_{\text{LM-CVAE}}(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N}\left[D_{KL}\left(q_\phi\left(\boldsymbol{v}|\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}, z^{(i)}\right)||p_\theta(\boldsymbol{v})\right) - \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{y}^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{v}^{(l)}, z^{(i)}\right)\right]. \quad (8)$$

where $\boldsymbol{v}^{(l)} \sim p(\boldsymbol{v})$. As in the case of the regular likelihood model, $\log p_\theta\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{v}^{(l)}, z\right)$ is represented using predicted mean and covariance matrix of multivariate normal distribution, and thus its expansion follows the same rules as in equation (5).

I use the same training strategy as for the regular likelihood model. The objective is trained using an Adam optimizer with random minibatches of data, where the covariance matrices are reconstructed in the same fashion as described in the Likelihood section, with missing filter magnitudes marginalized in each minibatch. To allow the gradient flow through the network with backpropagation when sampling from $q_\phi$, I employ the reparametrization trick using the Pytorch `rsample` function of the `torch.distributions` class [Meta AI, 2023b]. However, even though most of the operation of the CVAE variant is shared with the standard likelihood model, its architecture follows a more complex structure.

### 4.2.1 CVAE architecture

When designing the architecture of the CVAE variant of the likelihood model, I took inspiration from Sohn et al. [2015]. Nonetheless, I tailored it to suit my specific case, introducing certain modifications. The architecture is composed of four main components:

1. The recognition network $q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}, z)$, which generates the distribution of latent samples based on the input $\boldsymbol{x}$, $\boldsymbol{y}$, and $z$, and it is primarily used during the training routine to approximate the expectation in equation (7). All arguments are concatenated before being processed by the network.

2. Prior $p(\boldsymbol{v})$, which, same as $q_\phi$, generates the latent samples, but now with condition dropped on $\boldsymbol{y}$, and thus it is used during the inference. To simplify, I also drop the conditioning on $\boldsymbol{x}$ and fix it to standard Gaussian.

3. The generation network, responsible for reconstructing the distribution of $\boldsymbol{x}$ from the latent samples $\boldsymbol{v}$, and depending on the intended usage, the samples come from either prior or $q_\phi(\boldsymbol{v}|\boldsymbol{x}, \boldsymbol{y}, z)$. No matter which density is the source of the latent samples, the generation network always takes redshift constant $z$ as an input along with $\boldsymbol{v}$.

4. The *direct network* $p(\boldsymbol{x}|\boldsymbol{y}, z)$, which, similar to the network described in the Likelihood model section, based on $\boldsymbol{y}$ and $z$, generates the distribution of sample $\boldsymbol{x}$.

In terms of composition, each network follows the same core principles as described in the Architecture of the models section and shown in Figure 10: First, the input is processed by the shared subnetwork, whose output is further processed by mean and covariance subnetworks, generating the corresponding components of predicted multivariate normal distribution, or in the case of $q_\phi$, only simplified diagonal. As before, each subnetwork is composed of residual

blocks, following the same rescaling routines of the samples flowing through the network.

Upon closer examination of the formulation above, a significant issue becomes evident. In cases where the prior remains fixed, the reconstructed outputs lack the necessary conditioning on $\boldsymbol{y}$, thus rendering the model formulation incomplete. Therefore, the architecture introduces a primary network, which plays a guiding role in predictions. In this design, each distribution of $\boldsymbol{x}$ produced by the generation network from multiple latent samples is combined with a single output of the direct network. Thus, the direct network serves as the core component upon which predictions based on latent space introduce modifications. Specifically, as predicted means and covariances by both networks are returned by their corresponding subnetworks, these outputs are combined by summation immediately before passing through their final dense layers, where they are linearly combined. This process happens regardless of whether the latent samples come from recognition or a prior network. Thus, both of these networks share the same last layer. For a visual representation of the above description, please refer to Figure 11.

### 4.2.2 Adapting evaluation to CVAE

I aim to evaluate CVAE similarly to the regular likelihood model. However, now, instead of having a single prediction per $(\boldsymbol{y}, z)$, there are multiple based on the latent $\boldsymbol{v}$, which also means that there's no direct access to the likelihood $p_\theta(\boldsymbol{x}|\boldsymbol{y}, z)$ since it requires marginalization of $\boldsymbol{v}$ in the case of CVAE. Therefore, below I present in short how I adapt evaluation methods presented in Likelihood model section to the predictions of CVAE.

Naturally, the first aspect I monitor is the value of the loss function from equation (8), calculated on both training and validation sets. To infer the single output $\boldsymbol{x}^*$ from the model, I follow a method presented in the Conditional Variational Autoencoders section, where

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \frac{1}{L} \sum_{l=1}^{L} p_\theta\left(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{v}^{(l)}, z\right), \ \boldsymbol{v}^{(l)} \sim p(\boldsymbol{v}).$$

In the case of $p_\theta(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{v}, z)$ being multivariate Gaussian, it is simply the average of predicted means. Based on $\boldsymbol{x}^*$, I also track per epoch MAE between it and the ground-truth $\boldsymbol{x}$ on train and validation sets, and based on these metrics, I follow with the best setting of the model to further evaluation on the test set. Naturally, the same settings are then used for final sampling with MCMC.

Computing the likelihood $p_\theta(\boldsymbol{x}|\boldsymbol{y}, z)$ is essential to compare the regular likelihood model with the CVAE variant. Therefore, I estimate these values using importance sampling from the $q_\phi$ density. By performing this for all samples in the dataset, I can derive the average NLL and its distribution

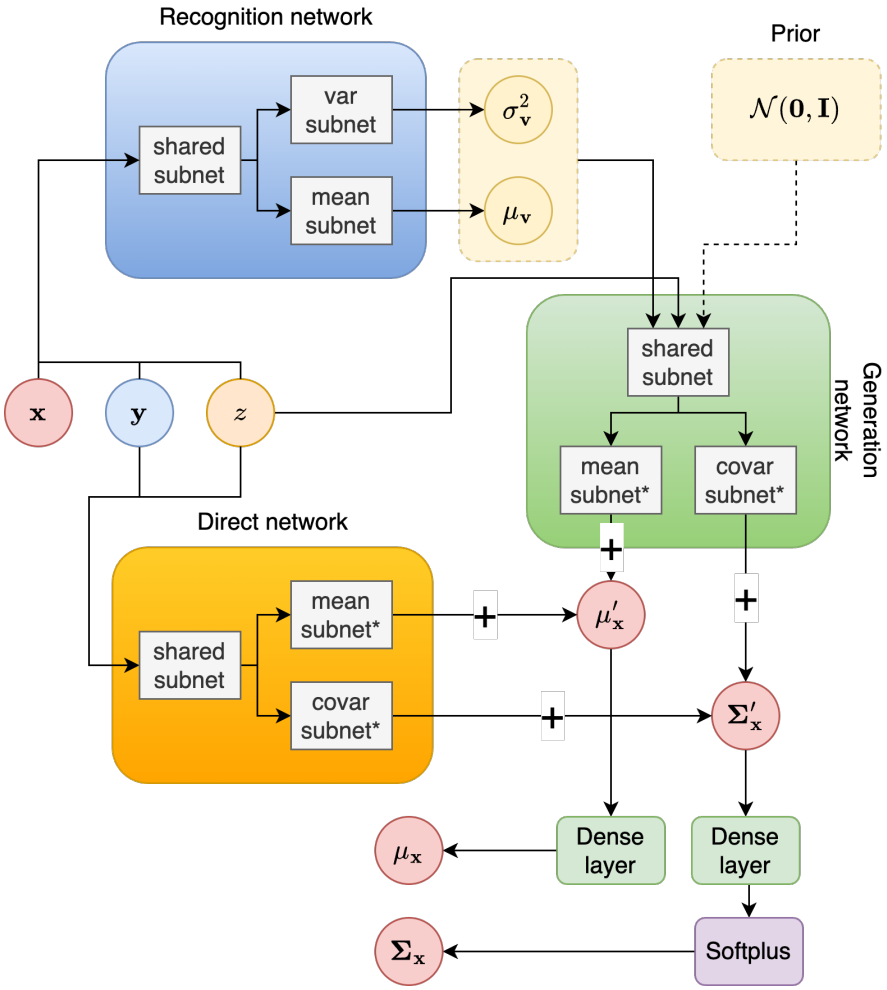



Figure 11: Graph showing the architecture of the CVAE model. Each subnetwork is defined as in the case of the likelihood model (Figure 10), with a minor exception in the subnetworks marked with an asterisk, where in their case, the last dense layer is omitted, and instead, it is applied after summation of the outputs of generation and direct networks. Depending on the usage, latent samples come either from the distribution predicted by the recognition network or from prior.

across the data. To do the same for the distribution of RMSE, I utilize the $\boldsymbol{x}^*$ prediction. Finally, to obtain the calibration plot for the model, I combine the $\boldsymbol{x}^*$ with the estimate of $\boldsymbol{\Sigma}^*_{\boldsymbol{x}}$. To do so, I compute the average covariance matrix per dataset sample across all latent samples. However, it is crucial to account for the variation in the means to obtain accurate result, thus I add the variance of the means to the diagonal of the average covariance. Subsequently, relying on $\boldsymbol{x}^*$ and $\boldsymbol{\Sigma}^*_{\boldsymbol{x}}$, I calculate the calibration using the same procedure as for the

regular model, thus concluding the evaluation of CVAE. The process described above is then repeated for the model variant trained without utilizing the SNe features.

### 4.3 Sampling from the posterior

With the trained likelihood model $p_\theta(\boldsymbol{x}|\boldsymbol{y}, z)$, I proceed to the final step of the thesis, which is obtaining the samples from the posterior density $p_\theta(\boldsymbol{y}|\boldsymbol{x}, z)$. To do so, I incorporate a variant of the MCMC algorithm, the No-U-Turn Sampler (NUTS) Hoffman and Gelman [2011]. NUTS is an extension to one of the cornerstone MCMC algorithms – Hamiltonian Monte Carlo (HMC) Neal [2010] – that self-tunes an important HMC parameter: the number of integration steps. By employing an existing implementation of NUTS from the Pyro (v1.8.4) library [Uber AI Labs, 2023], I let the algorithm adapt the step size and full mass matrix during the warm-up phase, avoiding the need to tune them myself.

When applied to CVAE models, the computation of $\log p(\boldsymbol{y}|\boldsymbol{x}, z)$ requires marginalizing out the latent variables, which I do with Monte Carlo sampling with the prior $p(\boldsymbol{v})$. In both cases, I marginalize out the filters with missing magnitudes in the ground-truth $\boldsymbol{x}$ when calculating the likelihood density. Then, the method's performance hinges on the following arguments: the number of 'warm-up' and retained samples and the number of chains, which I determine based on the minimal diagnostic requirements proposed by Vehtari et al. [2021]. Specifically, I consider that the posterior density was sufficiently explored by the NUTS algorithm run with four chains if, for every $\boldsymbol{y}$ feature, it achieved ESS-bulk and ESS-tail bigger than 400, per-chain ESS bigger than 50 (for all chains), and if $\hat{R} \leq 1.01$ (which is a minor relaxation compared to $\hat{R} < 1.01$, suggested by Vehtari et al. [2021]). All mentioned metrics are calculated using the ArviZ library [NumFOCUS, 2023] .

To obtain the final evaluation of my method, ideally, I would want to run NUTS on all data points in the test set. Unfortunately, it was unfeasible due to challenges in parallelizing multiple chains and the inherent computational complexity associated with MCMC sampling. Therefore, I limited the experiments to just a few randomly picked samples from the test set. For each evaluated datapoint $(\boldsymbol{x}, \boldsymbol{y}, z)$, I run NUTS with 5000 warm-up and sampling steps each. Rather than replicating the real-world scenario of estimating a single input, the primary objective of this evaluation is to demonstrate the method's effectiveness for multiple samples. Therefore, to accelerate the exploration of the posterior density, I pick the true $\boldsymbol{y}$ as a sampling starting point for the Markov chains. Upon completion, I check whether the aforementioned minimal requirements for ESS and $\hat{R}$ are fulfilled. After running MCMC multiple times, I discovered that in some cases, the method would encounter stagnation, leading to exceedingly long runtimes, some lasting up to 10 hours. Moreover, in multiple instances, these prolonged runs failed to

meet the minimum requirements for the mentioned statistics. Thus, I set a runtime limit of 3 hours per sample from the dataset on all MCMC runs. After filtering the results, I move on to the evaluation.

First, per each evaluated data point from the test set, I compute the average of the posterior samples $\boldsymbol{y}^*$, based on which I retrieve per-feature mean absolute error (MAE) between $\boldsymbol{y}^*$ and ground-truth $\boldsymbol{y}$. Also, to compare my approach with the standard one, I train a model equivalent in terms of architecture and training to the one presented in the Likelihood model section but designed to directly predict multivariate Gaussian distribution $p\left(\boldsymbol{y}|\boldsymbol{y}, z\right) = \mathcal{N}\left(\boldsymbol{\mu_y}|\boldsymbol{\Sigma_y}\right)$ of y of $\boldsymbol{y}$ based on $\boldsymbol{x}$ and $z$. Thus, I compare its mean prediction error with MAE of the mean of the posterior samples from my approach. Further, for each datapoint, I compute the fraction of its corresponding samples with lower likelihood under the model $p_\theta\left(\boldsymbol{y}, \boldsymbol{x}\right)$ and aggregate all metrics in ten equally spaced bins on the interval $[0, 1]$. Then, if MCMC successfully simulated sampling from the posterior, the bin counts should be uniformly distributed. Further in the thesis I refer to this metric as MCMC calibration. I replicate all the forenamed steps for the top-performing regular likelihood model and CVAE variant for both cases trained with and without SNe features.

Then, for the best-performing model, I conduct sampling again for the same data points and identical expectations for convergence but only using a subset of the available JWST filters. The only change that needs to be applied is in the potential energy function, where in $p\left(\boldsymbol{x}|\boldsymbol{y}, z\right)$, I need to marginalize the discarded filters. As for choosing which filters to keep, I turned to the results presented by Ansari et al. [2022], where authors tried to determine the minimal filter settings that still resulted in an acceptable performance of a neural network trained to directly predict the space dust features from the JWST filter magnitudes. They considered multiple sampling scenarios of the artificially generated dataset, and I picked the suggested filters for *Scenario 2* from the paper since it is the most similar to the dataset used in this project. Therefore, the following set contains the kept filters

$$\begin{aligned}\mathcal{F} = \{&F140M, F150W, F277W, F300M, F356W, F360M, F410M,\\ &F560W, F770W, F1000W, F1280W, F1500W, and F1800W\},\end{aligned} \quad (9)$$

meaning that out of 29 filters in total, only 13 are used in this experiment.

# 5 Results

The structure of this section follows natural dependencies between components of the thesis. First, I present the training results and assess the performance of the likelihood models since the architecture of the most successful setup in this context will serve as the foundation for the subsequent adaptation to Conditional Variational Autoencoder (CVAE). Following that, I delve into the evaluation of experiments regarding CVAE. Finally, the section concludes with an evaluation of sampling from the posterior using all these models, including variants trained without supernovae (SNe) features, providing the final results of the thesis.

## 5.1 Performance of the likelihood models

The proposed architecture of the likelihood model involves numerous free parameters, including the number of neurons in dense layers, convolutional layer parameters, $\sigma^2_{\texttt{min}}$, and training parameters such as learning rate and batch size. For conciseness, this section only presents results for two variants: one entirely linear ($\mathcal{M}_{\texttt{lin}}$) and one partially convolutional ($\mathcal{M}_{\texttt{cov}}$), since both demonstrated the best performance out of all configurations, while their outcomes were quite similar. Initially, I outline the specific parameter values used for these variants, followed by a comparative evaluation of their performance.

### 5.1.1 Experimental setup

Consistently throughout all conducted experiments, the variants featuring entirely dense shared and mean subnetworks resulted in the best performance. As a result, both $\mathcal{M}_{\texttt{cov}}$ and $\mathcal{M}_{\texttt{lin}}$ share identical configurations for these subnetworks. Effectively, the part where they differ becomes the covariance subnetwork, where $\mathcal{M}_{\texttt{cov}}$ employs a fully convolutional approach, while $\mathcal{M}_{\texttt{lin}}$ relies on a fully dense structure.

Notably, increasing the number of residual blocks within these subnetworks did not produce significant performance improvements. Consequently, each subnetwork consists of a single residual block. Given that the input data has a shape of $(B \times L)$, where $B$ represents the batch size, and $L$ corresponds to the length of the sample's signal, to be processed by the convolutional layer, it needs to be converted to a three-dimensional format $(B \times C \times L)$, where $C$ denotes the number of channels. Thus, before entering the convolutional part of the network, the input is expanded to $(B \times 1 \times L)$. The specific numbers of neurons in dense layers and parameter values of the convolutional layers are presented in Table 2.

Immediately, I resize the input from a size of 8 (+1 for $z$) to $d_{\boldsymbol{x}} = \dim(\boldsymbol{x}) = 29$, allowing the network the maximum learning capacity. The same resizing applies to the covariance subnetwork, which requires precisely

| Subnetwork | Layer | Number of neurons | |
|---|---|---|---|
| | | in | out |
| shared | #1 | 8 | 29 |
| | #2 | 29 | 29 |
| mean | #1 | 29 | 29 |
| | #2 | 29 | 29 |
| covariance | #1 | 29 | 435 |
| | #2 | 435 | 435 |

(a) Number of neurons in the dense layers of $\mathcal{M}_{\texttt{lin}}$, where $\mathcal{M}_{\texttt{cov}}$ shares the same setting in shared and mean subnetworks.

| Layer | Channels | | Kernel size | Stride | Padding |
|---|---|---|---|---|---|
| | in | out | | | |
| #1 | 1 | 15 | 7 | 1 | 3 |
| #1 | 15 | 15 | 7 | 1 | 3 |

(b) Parameters of the convolutional layers from the covariance subnetwork of the $\mathcal{M}_{\texttt{cov}}$ model.

Table 2: Specific values of the parameters of different layers in the subnetworks of models $\mathcal{M}_{\texttt{cov}}$ and $\mathcal{M}_{\texttt{lin}}$.

$d_{\boldsymbol{x}}\left(d_{\boldsymbol{x}}+1\right)/2 = 435$ values to represent the predicted covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$. This resizing principle also applies to the convolutional model, where the number of channels multiplied by the input's dimensionality equals 435. Whenever there's a change in input shape, the residual identity mapping is replaced with a projection having the same layer configuration as the non-residual part. The number of parameters in the last dense layers is naturally determined by the shapes of the outputs of the final layers and the size of the final prediction. For $\mathcal{M}_{\texttt{cov}}$, I picked the values of stride and padding so that the dimensionality of the input remains the same. Even though the kernel size of 7 seems relatively small, I experimentally verified that increasing its value did not yield any notable improvements.

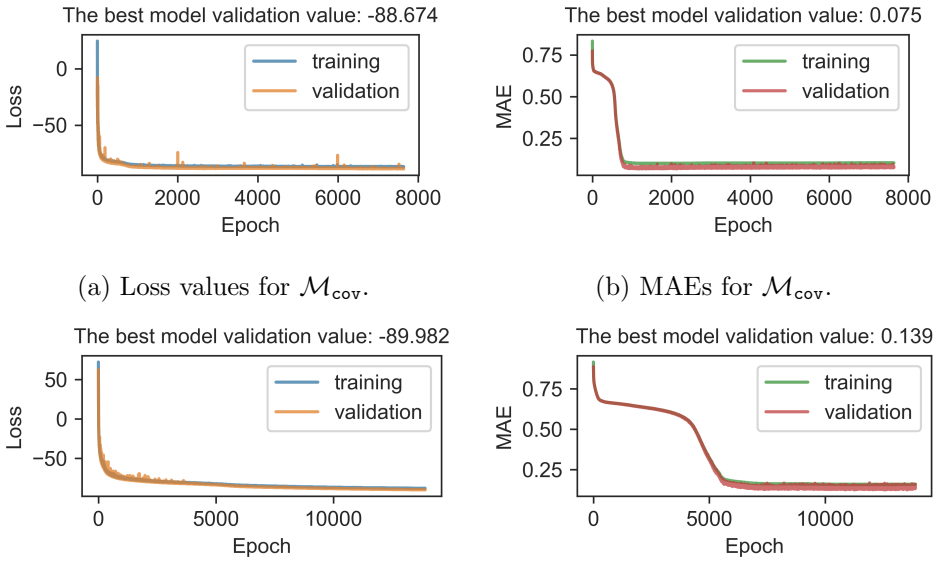
Both models are trained with $\sigma^2_{\texttt{min}} = 10^{-5}$ and a batch size of 256. Lower values of $\sigma^2_{\texttt{min}}$ led to unstable training and validation loss values. Since $\mathcal{M}_{\texttt{lin}}$ has more than double the parameters of $\mathcal{M}_{\texttt{cov}}$ (411,858 vs. 195,828), it proved significantly more challenging to optimize. Thus, the $\mathcal{M}_{\texttt{lin}}$ model had to be trained with a learning rate as small as $5 \cdot 10^{-6}$, while the convolutional variant performed well with a learning rate set to $10^{-4}$.

A crucial factor that allowed for the stable training of the dense-layered variant was the choice of the $\beta$ parameter in the softplus function (as defined in equation 6). In both $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cov}}$, I set the $\beta$ to 0.1, which reduces the rate at which softplus values decrease as the arguments approach $-\infty$. Thus,

it ensures that small changes in the argument value do not disproportionately affect the softplus value. After deciding between the linear and convolutional variants, the best-performing setup with identical parameters is used to train the model without SNe features.

### 5.1.2 Convolutional vs. Dense models

To determine which architecture, $\mathcal{M}_{\texttt{lin}}$ or $\mathcal{M}_{\texttt{cov}}$, should be pursued for the subsequent experiments, I examine the losses and MAEs computed over the training and validation datasets, as illustrated in Figure 12. Since $\mathcal{M}_{\texttt{lin}}$ had a much lower learning rate, it was trained for more epochs.




(a) Loss values for $\mathcal{M}_{\texttt{cov}}$.

(b) MAEs for $\mathcal{M}_{\texttt{cov}}$.

(c) Loss values for $\mathcal{M}_{\texttt{lin}}$.

(d) MAEs for $\mathcal{M}_{\texttt{lin}}$.

Figure 12: Comparison of loss and MAE values between $\mathcal{M}_{\texttt{cov}}$ and $\mathcal{M}_{\texttt{lin}}$, calculated throughout training on the training and validation datasets.

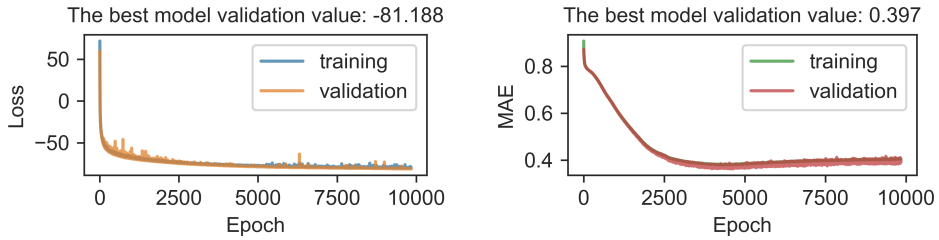
We can see that both architectures show similar trends in their curves. There are two noteworthy observations here. First, upon closer scrutiny, it becomes apparent that, in both cases, the loss and MAE values computed over the validation set consistently remain lower than those calculated on the training set. Secondly, during the most rapid reduction in loss values, the MAE tends to plateau, and then the reverse occurs — the loss decreases slowly while the MAE drops rapidly.

$\mathcal{M}_{\texttt{cov}}$ achieved nearly half the MAE compared to $\mathcal{M}_{\texttt{lin}}$, though with marginally worse NLL. As the primary goal of this optimization is to minimize NLL, and considering that $\mathcal{M}_{\texttt{lin}}$ is generally expected to be more expressive than $\mathcal{M}_{\texttt{cov}}$, I proceed with it for further experiments, and its architecture will

serve as the foundation for the CVAE setting. Consequently, in the next section, I present the comparative evaluation of $\mathcal{M}_{\texttt{lin}}$ and its counterpart trained without SNe features, denoted as $\mathcal{M}^{\star}_{\texttt{lin}}$.

### 5.1.3 With vs. without supernovae features

First, I look at the training statistics of the fully dense model trained without SNe features, presented in Figure 13. Compared to $\mathcal{M}_{\texttt{lin}}$, it achieves much




(a) Loss values.

(b) MAEs.

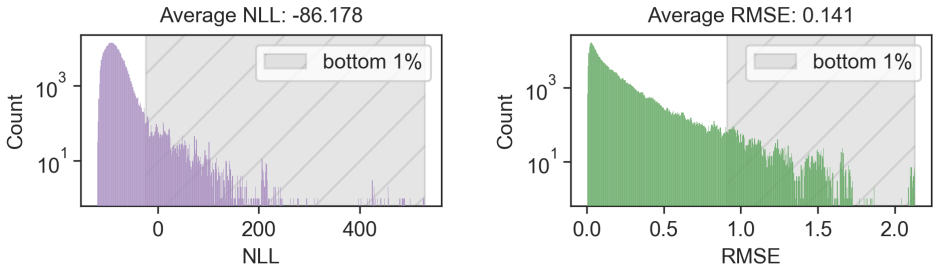
Figure 13: Loss and MAE values achieved by the $\mathcal{M}^{\star}_{\texttt{lin}}$ model, calculated throughout training on the train and validation datasets.

lower NLL and MAE scores, as there is a reduced amount of information available to the model. The loss curve follows the same trajectory as in the case of $\mathcal{M}_{\texttt{lin}}$. However, in this instance, we can observe in Figure 13b that the MAE values begin to increase, even as the loss continues to decrease.

Considering these observations, I look into the distributions of these statistics, but this time on the test set (Figure 14). For model $\mathcal{M}_{\texttt{lin}}$, there is only a slight increase in the loss value on the test set when compared to that calculated over the validation set (as shown in Figure 12c). However, in the case of $\mathcal{M}^{\star}_{\texttt{lin}}$, the difference is significant, with a substantial increase of 6. In both cases, numerous extreme outliers are noticeable, and this effect is more pronounced in the model trained without SNe features.

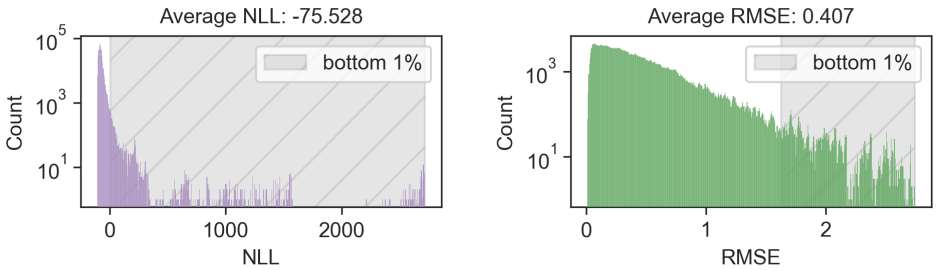
To identify which filters are particularly challenging to predict and might contribute to the outliers in the distribution plots, in Figure 15 I examine the per-feature RMSEs of both models. The filters that cover lower wavelength ranges exhibit the highest RMSE values. Similarly to the results shown in Figure 14, $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ display similar trends, with $\mathcal{M}^{\star}_{\texttt{lin}}$ being inferior in almost all aspects. Interestingly, this shift in performance doesn't seem to significantly impact the filters in the mid-infrared (MIR) part of the spectrum, which cover higher wavelength ranges.

Now, in Figure 16, I inspect the calibration of the model. Across nearly all confidence intervals, both models appear to exhibit significant underconfidence, particularly in the lower percentiles. This trend gradually diminishes as confidence levels increase, eventually achieving near-perfect calibration at




(a) Loss distribution of $\mathcal{M}_{\texttt{lin}}$.

(b) RMSE distribution of $\mathcal{M}_{\texttt{lin}}$.




(c) Loss distribution of $\mathcal{M}^{\star}_{\texttt{lin}}$.

(d) RMSE distribution of $\mathcal{M}^{\star}_{\texttt{lin}}$.

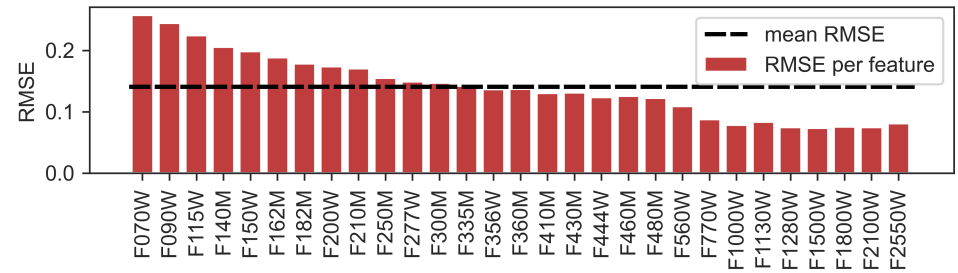
Figure 14: comparison of loss and error distributions between models $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$, calculated over the test set.

around 0.8 for $\mathcal{M}_{\texttt{lin}}$ and slightly earlier, at 0.7, for $\mathcal{M}^{\star}_{\texttt{lin}}$. Subsequently, both models transition into a mild state of overconfidence. While $\mathcal{M}_{\texttt{lin}}$ appears to achieve somewhat better calibration than $\mathcal{M}^{\star}_{\texttt{lin}}$ across all percentiles, the difference is relatively small.

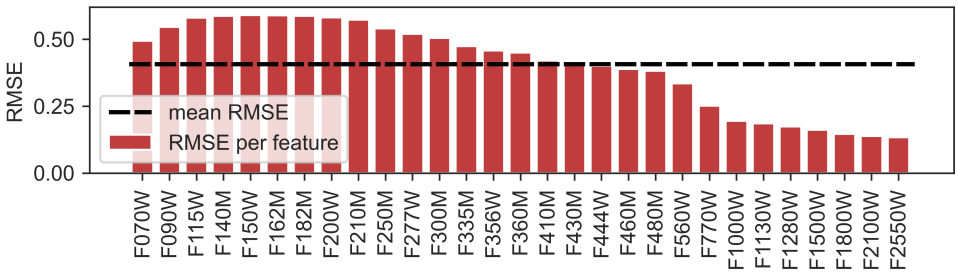
Lastly, I examine the average covariances generated by the models (Figure 17), which tend to reflect the trends observed in the pre-filter RMSE data displayed in Figure 15. Which, in essence, provides an overview of the model's uncertainty around the predicted $\boldsymbol{\mu}_{\boldsymbol{x}}$. Notably, there is an apparent correlation among the near-infrared (NIR) filters, which also assume the highest values. While the uncertainties for mid-infrared (MIR) filters are remarkably low, they still show a positive correlation. Interestingly, there appears to be a more pronounced correlation between NIR and MIR filters than among the MIR filters themselves. Nevertheless, all covariance values for the $\mathcal{M}_{\texttt{lin}}$ model are notably lower than those of the $\mathcal{M}^{\star}_{\texttt{lin}}$ model.

## 5.2 Performance of CVAE

Considering that the best design for the regular likelihood model has been established, this section only contains the specification of the experimental setup regarding the additional components introduced in CVAE, followed by




(a) Model $\mathcal{M}_{\texttt{lin}}$.




(b) Model $\mathcal{M}^{\star}_{\texttt{lin}}$.

Figure 15: Per-feature RMSE of $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ calculated over the test set.

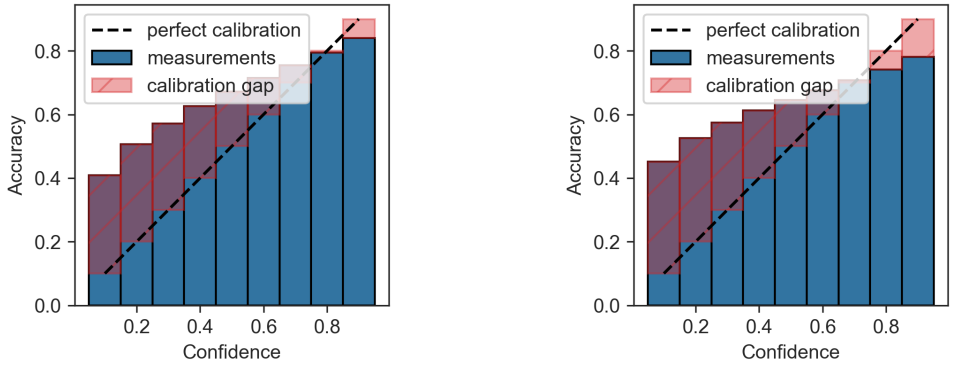
the comparative study of CVAE variant trained with and without SNe features, as done in the previous section for the regular model.

### 5.2.1 Experimental setup

Similar to the section on the regular likelihood model, here I provide the experimental setup only for the CVAE variant that demonstrated the best performance among all the models evaluated.

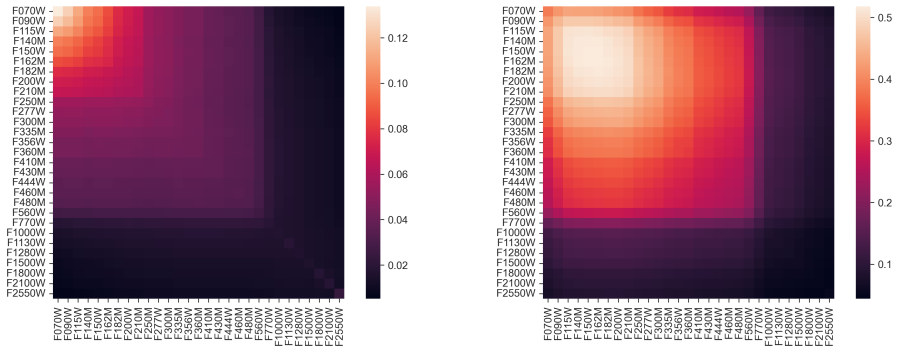
First, I configure all the parameters of the direct network of CVAE (point (4) of the list outlined in the CVAE architecture section) identically as in the setup of $\mathcal{M}_{\texttt{lin}}$. Then, for the sake of simplicity, I adopt a very similar configuration for the generation network. However, as the generation network generates predictions based on the latent samples, its input shape undergoes changes, resulting in minor adjustments to the shared subnetwork of the generation network. I chose the latent space size of 32, as increasing this value did not cause significant improvements in the model's performance. Thus, all dense layers of the shared subnetwork of the generation network consist of 32 neurons, while the parameters of the mean and covariance subnetworks remain consistent with those in Table 2.

For the recognition network, I apply a similar reasoning behind choosing the network parameters, aiming to maximize its learning capacity. Precisely,




(a) Model $\mathcal{M}_{\texttt{lin}}$. (b) Model $\mathcal{M}^{\star}_{\texttt{lin}}$.

Figure 16: Comparison of calibration plots between $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ calculated over test set.



(a) Model $\mathcal{M}_{\texttt{lin}}$. (b) Model $\mathcal{M}^{\star}_{\texttt{lin}}$.

Figure 17: Comparison of average covariances between $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ calculated over test set.

as it takes in 37 features (29 for $\boldsymbol{x}$, 7 for $\boldsymbol{y}$ and 1 for $z$), the shared network consists of 37 neurons across all layers of the shared subnetwork. These are subsequently reduced to 32 in both the mean and covariance subnetworks, as the covariance subnetwork now solely predicts the variances ($\boldsymbol{\sigma}^2_{\boldsymbol{v}}$) instead of the full covariance matrix. As detailed in the Methods section, the prior remains fixed as a standard Gaussian, with its size matching that of the latent space.

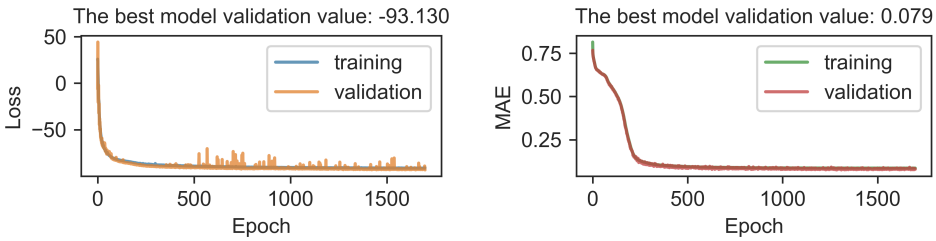
As in the case of $\mathcal{M}_{\texttt{lin}}$, the model is trained with $\sigma^2_{\texttt{min}} = 10^{-5}$ and a batch size of 256. Regarding the direct and generation networks, I maintained $\beta$ at 0.1, as in the case of $\mathcal{M}_{\texttt{lin}}$. However, for the generation network, I increased its value to 0.5. The motivation behind this adjustment is that predicting variances alone, rather than full covariances, is a relatively simpler task, which

reduces instabilities. Moreover, increasing $\beta$ in the generation network could potentially speed up the convergence to the optimum while training.

Throughout the training, I used 100 samples from the recognition network to estimate the expectation in the loss function (equations (7) and (8)). This configuration led to stable training with a learning rate set to $10^{-4}$, which is much smaller when compared to $\mathcal{M}_{\texttt{lin}}$ trained with $5 \cdot 10^{-6}$. To estimate $-\log p(\boldsymbol{x}|\boldsymbol{y}, z)$ during evaluation, I employ importance sampling using the recognition network $q_\phi$, with 100 samples used. Identical training and network setup are used for the model trained without SNe features.

### 5.2.2 CVAE evaluation

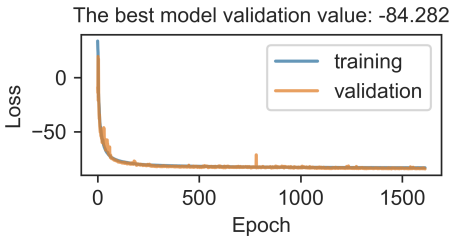
As before, I start by analyzing the training statistics of the two models. In the rest of the thesis, the regular CVAE model is denoted as $\mathcal{M}_{\texttt{cvae}}$, while the one trained without SNe features is marked as $\mathcal{M}^{\star}_{\texttt{cvae}}$, following the naming convention used for the regular likelihood model. As before, I start by analyzing the training statistics of the two models. Both variants of CVAE



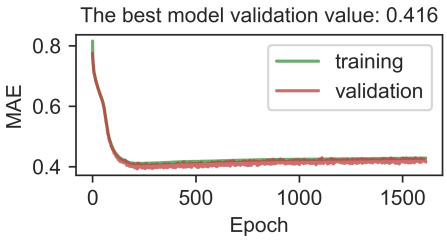

(a) Loss values for $\mathcal{M}_{\texttt{cvae}}$.




(b) MAEs for $\mathcal{M}_{\texttt{cvae}}$.




(c) Loss values for $\mathcal{M}^{\star}_{\texttt{cvae}}$.



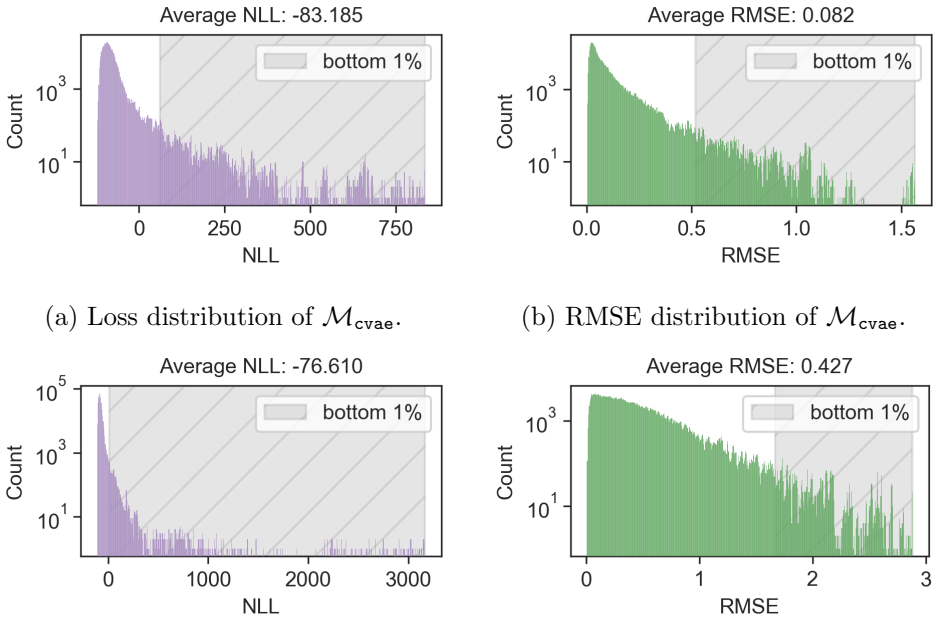

(d) MAEs for $\mathcal{M}^{\star}_{\texttt{cvae}}$.

Figure 18: Comparison of loss and MAE values between $\mathcal{M}_{\texttt{cvae}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$, calculated throughout training on the training and validation datasets.

models achieved lower loss values compared to their corresponding counterparts, $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$, even though the CVAE loss is only an upper bound for $-\log p(\boldsymbol{x}|\boldsymbol{y}, z)$. Regarding the MAE, $\mathcal{M}_{\texttt{cvae}}$ records significantly lower values than $\mathcal{M}_{\texttt{lin}}$, while the variants trained without SNe features show an inverse relation. However, in the latter case, the difference is marginal. Nonetheless,

the curves follow the same trajectories seen in the regular likelihood models, showing a distinct delay in optimizing MAE and a slight increase in MAE from its minimal value in the case of $\mathcal{M}^{\star}_{\texttt{cvae}}$ as training progresses.

Notably, the performance of CVAE models experiences a significant decline when assessed on the test set. The NLLs are considerably higher compared to the regular likelihood models, and this discrepancy is also evident in the extremity of the outliers depicted in Figure 19. A similar can be said about the RMSE of $\mathcal{M}^{\star}_{\texttt{cvae}}$ compared to that of $\mathcal{M}^{\star}_{\texttt{lin}}$, while for $\mathcal{M}_{\texttt{cvae}}$, it is still much lower than for $\mathcal{M}_{\texttt{lin}}$. As expected, $\mathcal{M}^{\star}_{\texttt{cvae}}$ performs much worse than




(a) Loss distribution of $\mathcal{M}_{\texttt{cvae}}$.

(b) RMSE distribution of $\mathcal{M}_{\texttt{cvae}}$.

(c) Loss distribution of $\mathcal{M}^{\star}_{\texttt{cvae}}$.

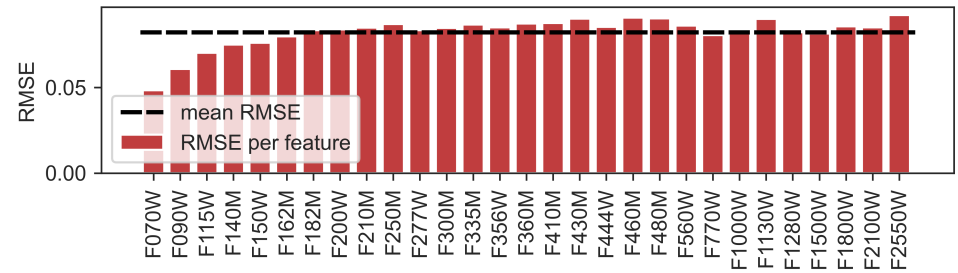
(d) RMSE distribution of $\mathcal{M}^{\star}_{\texttt{cvae}}$.

Figure 19: Comparison of loss and error distributions between models $\mathcal{M}_{\texttt{cvae}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$, calculated over the test set.

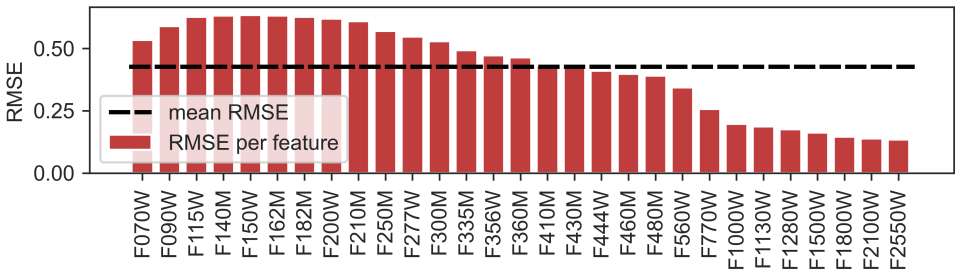
$\mathcal{M}_{\texttt{cvae}}$ on both RMSE and NLL.

In alignment with the training statistics, Figure 20 shows that $\mathcal{M}_{\texttt{cvae}}$ has achieved distinctly low RMSE values across all filters. Surprisingly, the error is much lower for low-wavelength filters, which stands in contrast to the behavior observed in $\mathcal{M}_{\texttt{lin}}$, where we can see an inverse trend. Conversely, $\mathcal{M}^{\star}_{\texttt{cvae}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ show nearly identical results.

Another intriguing difference becomes apparent when examining the calibration plots of the models, as depicted in Figure 21. Specifically, the $\mathcal{M}_{\texttt{cvae}}$ model exhibits greater confidence compared to $\mathcal{M}_{\texttt{lin}}$, leading to improved calibration at lower confidence percentiles but declining performance at higher ones. Remarkably, the CVAE variant trained without SNe features displays




(a) Model $\mathcal{M}_{\texttt{cvae}}$.




(b) Model $\mathcal{M}^{\star}_{\texttt{cvae}}$.

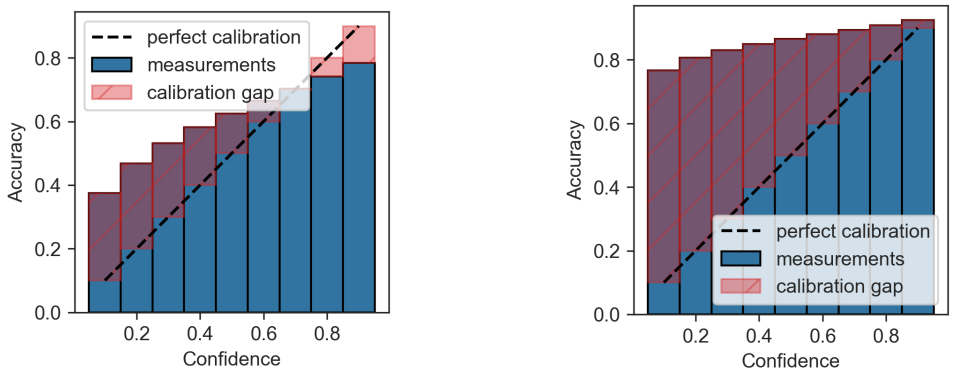
Figure 20: Per-feature RMSE of $\mathcal{M}_{\texttt{cvae}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$ calculated over the test set.

a unique behavior among all the models, as it is significantly underconfident across all confidence intervals.

Finally, in Figure 22, I examine the average covariances generated by the models. What instantly stands out is the seemingly more sophisticated representation obtained from model $\mathcal{M}_{\texttt{cvae}}$, where we can see the strongest correlations within MIR and NIR groups of filters. Interestingly, the correlations between these groups of filters seem relatively low. Conversely, the average covariance of $\mathcal{M}^{\star}_{\texttt{cvae}}$ closely resembles that of $\mathcal{M}^{\star}_{\texttt{lin}}$, further aligning with the distribution of RMSE values shown in Figure 20b, where the highest errors were observed for NIR filters, for which we can also see the amplest uncertainties.

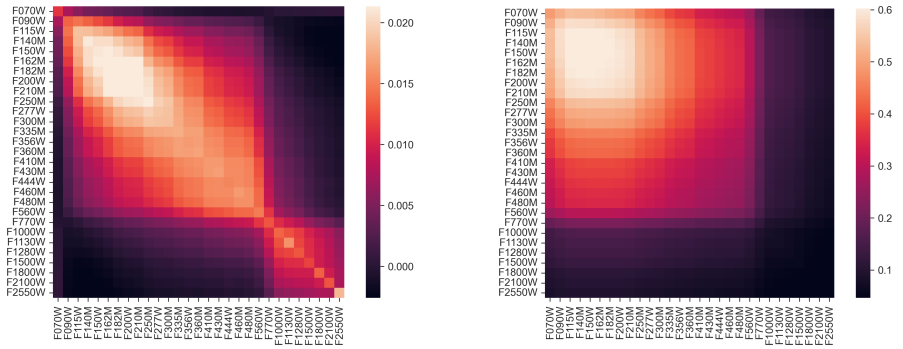
## 5.3 Posterior samples

This section serves as the culmination of the thesis, presenting the final results. Here, I showcase the outcomes of posterior sampling using all previously mentioned models: $\mathcal{M}_{\texttt{lin}}$, $\mathcal{M}^{\star}_{\texttt{lin}}$, $\mathcal{M}_{\texttt{cvae}}$, and $\mathcal{M}^{\star}_{\texttt{cvae}}$. These results are compared against a standard model, $\mathcal{M}_{\texttt{stan}}$, which is optimized to directly predict multivariate Gaussian distribution of $\boldsymbol{y}$ features based on input $\boldsymbol{x}$ and $z$. I start by detailing the experimental setup for training $\mathcal{M}_{\texttt{stan}}$. Then, I compare




(a) Model $\mathcal{M}_{\texttt{cvae}}$. (b) Model $\mathcal{M}^{\star}_{\texttt{cvae}}$.

Figure 21: Comparison of calibration plots between $\mathcal{M}_{\texttt{cvae}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$ calculated over test set.



(a) Model $\mathcal{M}_{\texttt{cvae}}$. (b) Model $\mathcal{M}^{\star}_{\texttt{cvae}}$.

Figure 22: Comparison of average covariances between $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$ calculated over test set.

sampling results across all likelihood models, including their CVAE variants. Subsequently, I pick one and employ it for sampling with a subset $\mathcal{F}$ of filters (equation 9), as described in the Sampling from the posterior section, to evaluate a more realistic scenario in which only a few filters are available for analysis.

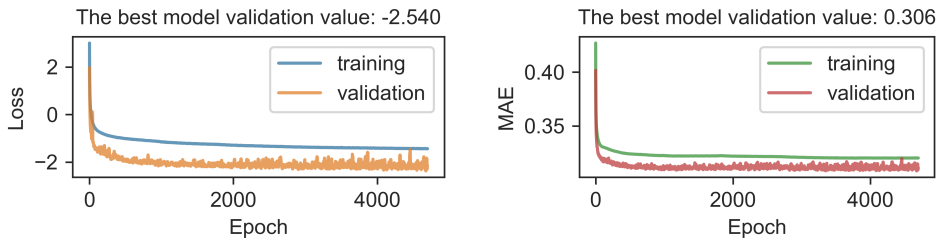
### 5.3.1 Experimental setup

The architecture of the standard model follows a straightforward design. It contains only dense layers, each featuring just one residual block per subnetwork. Each linear layer of the shared subnetwork contains 30 neurons, which aligns with the input size (29 for $\boldsymbol{x}$ and 1 for $z$). The layers of the covariance

network maintain the same number of neurons, reflecting the quantity needed to parametrize $\mathbf{\Sigma}_{\boldsymbol{y}}$. The mean network's layers contain 15 neurons each, as it falls in the middle between the size of the final prediction and the size of the input. As in the case of other models, it is trained with a batch size of 256, $\sigma^2_{\texttt{min}} = 10^{-5}$, and $\beta$ of softplus equal to 0.1. It proved to have stable training with a learning rate as high as $10^{-3}$.

### 5.3.2 Evaluation of the standard model

Training and evaluation of the standard model is not the primary focus of this thesis, as it is solely used to assess whether the main contribution of this thesis is potentially competitive with it. Therefore, I only evaluate the most critical statistics of the standard model. These include the ones computed during training, the per-feature RMSE, and the model's calibration.




(a) Loss values.

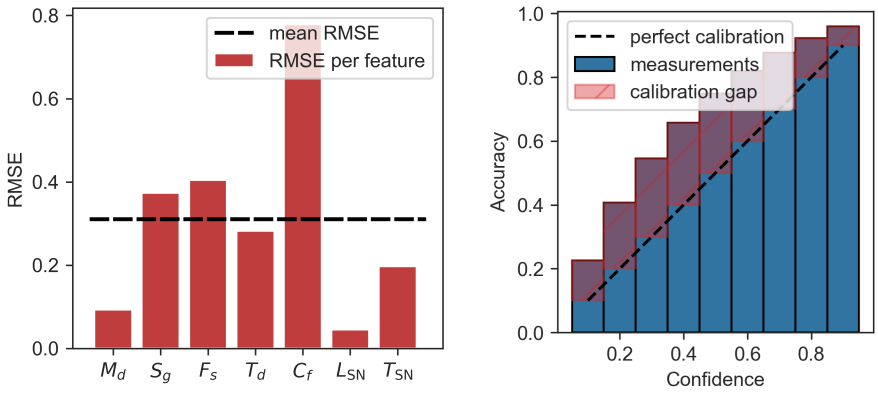
(b) MAEs.

Figure 23: Loss and MAE values achieved by the $\mathcal{M}_{\texttt{reg}}$ model, calculated throughout training on the train and validation datasets.

Compared to the loss of other models, this one has the most standard curves. What stands out, however, is again the consistently lower loss values on the validation set than on the training one. This time, MAE and loss follow almost identical trajectories.

Moving on to the RMSE plot depicted in Figure 24a, a striking observation is a substantial error in predicting the clump filling factor. Conversely, the dust mass and supernova luminosity appear reliably predictable, with remarkably low errors. SNe temperature proved to be somewhat more challenging to predict, yet it still exhibited an error of only 0.2. Grain size, silicate fraction, and dust temperature achieved similar error values, falling around the overall mean RMSE. Among these, $T_d$ turned out to be the easiest to estimate, while $F_s$ posed the biggest challenge.

Looking at the calibration plot in Figure 24b, we again observe that the model tends to be underconfident. However, compared to all the other models evaluated thus far, its degree of underconfidence is the lowest of all.




(a) Per-feature RMSE. (b) Calibration plot.

Figure 24: Per-feature RMSE and calibration plot of $\mathcal{M}_{\texttt{stan}}$, calculated over the validation set.

### 5.3.3 Comparison of the posterior samples

I start this section by presenting a comparative analysis of posterior sampling results obtained from the regular likelihood model and its CVAE variant, utilizing all 29 narrow and wide JWST filters. Then, I repeat this comparison for the models trained without SNe features. However, it's important to note that posterior sampling presented several challenges, including issues related to chain parallelization, long computational runtimes, instances of NUTS getting stuck on specific samples, problems with achieving convergence, and low ESS values. Thus, the number of successful sampling runs is relatively limited and varies heavily between the models. The specific values are summarized in Table 3.

Furthermore, due to technical issues, MCMC sampling for some models did not reach all data points picked for evaluation. Thus, when we consider sets of dataset entries for which different models managed to successfully estimate the posterior, their intersection might be much smaller than their sizes alone.

Model $\mathcal{M}_{\texttt{cvae}^\star}$ was the most affected by the mentioned issues, as it was run for only 266 samples from the dataset. $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$ managed to finish about 350 runs, while $\mathcal{M}^\star_{\texttt{lin}}$ stood out with 461 MCMC runs overall, the highest among all models. Furthermore, $\mathcal{M}^\star_{\texttt{lin}}$ retained the highest number of posterior samples for subsequent evaluation, with 76% of the total kept.

In contrast, the CVAE models retained the lowest percentage of samples, likely due to the time limits set for sampling runtimes, which is likewise evident in their reported average runtime durations. It also resulted in a significant number of incomplete runs, leading to the lowest percentage of samples meeting the convergence requirements defined by the ESS and $\hat{R}$ statistics. It is also worth noting that $\mathcal{M}_{\texttt{lin}}$ had twice as many samples filtered out compared

| Model | Total | Retained | Unfinished | Filtered out | Avg. runtime (min) |
|---|---|---|---|---|---|
| $\mathcal{M}_{\texttt{lin}}$ | 358 | 200 (56%) | 13 (4%) | 145 (40%) | 92 |
| $\mathcal{M}^{\star}_{\texttt{lin}}$ | 461 | 334 (73%) | 33 (7%) | 94 (20%) | 86 |
| $\mathcal{M}_{\texttt{cvae}}$ | 333 | 133 (40%) | 110 (33%) | 90 (27%) | 137 |
| $\mathcal{M}^{\star}_{\texttt{cvae}}$ | 266 | 121 (45%) | 111 (42%) | 34 (13%) | 126 |

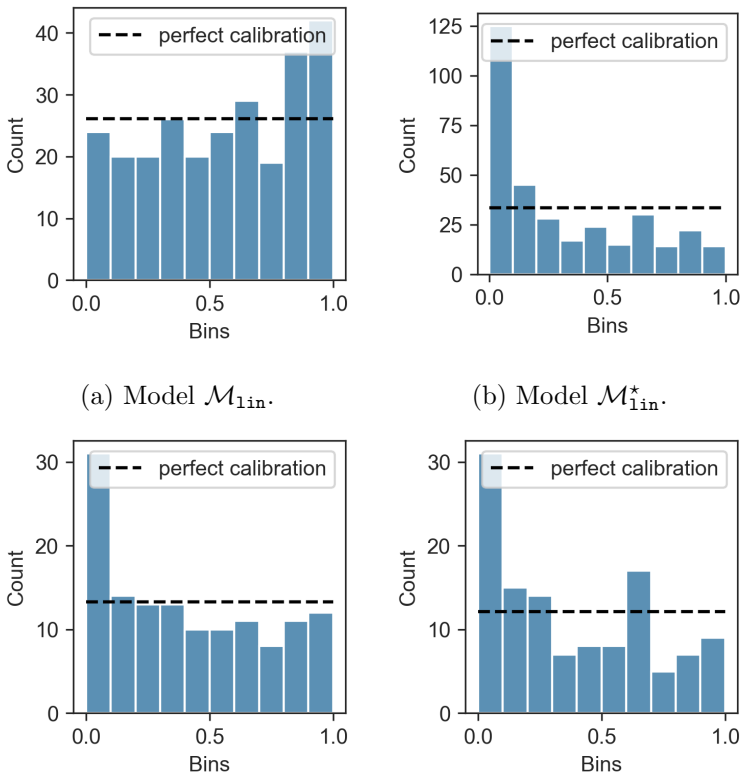
Table 3: Statistics of MCMC sampling from the posterior $p(\boldsymbol{y}|\boldsymbol{x}, z)$ using all previously evaluated likelihood models. Column 'Retained' refers to the number of samples that satisfied the MCMC diagnostic, while 'Filtered out' displays the ones that failed. The 'Unfinished' column shows a count of terminated runs due to exceeded time limit. Average time is computed by also including the time spend on unfinished samples.

to its counterpart trained without SNe features.

Figure 25 shows the calibration of posterior sampling for all models computed using their respective sets of retained samples. Models trained with all features generally display correct calibration across most bin intervals. However, in the case of $\mathcal{M}_{\texttt{lin}}$, there is a slight bias towards the 0.9-1 interval, indicating that more often than expected, the ground-truth sample had a higher likelihood under the density $p(\boldsymbol{y}|\boldsymbol{x}, z)$ than 90% of the MCMC samples. Conversely, the CVAE variant shows the opposite trend, which is even more pronounced, as the 0-0.1 bin contains roughly twice as many counts as the others. This trend is even more noticeable for the models trained without SNe features, where the bin counts show a pronounced tendency towards lower intervals, resulting in a significant imbalance.

In light of the variations in the sampling processes among the models, the collection of samples available for their comparison is significantly smaller when compared to the total number of sampling runs. For $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$, there are 71 data entries for which both models successfully estimated the posterior with adequate MCMC diagnostics. In the case of $\mathcal{M}^{\star}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$, this number slightly increases to 72. Nevertheless, I proceed with comparing MAEs between the average of posterior samples and the ground-truth Y values using these subsets. Additionally, I include the error in the mean prediction of $\mathcal{M}_{\texttt{stan}}$, computed on the matching data entries. These results are depicted in Figures 26 and 27.

When comparing $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$, their performance shows no significant difference, with one slightly outperforming the other depending on the specific feature. However, in most cases, the CVAE variant of the likelihood model demonstrates superior performance. In general, the errors of the standard model closely align with those calculated across the entire validation dataset, as illustrated in Figure 24a. Nevertheless, the approach proposed in this thesis proves more effective for all features except those with the lowest errors,




(a) Model $\mathcal{M}_{\texttt{lin}}$.

(b) Model $\mathcal{M}^{\star}_{\texttt{lin}}$.

(c) Model $\mathcal{M}_{\texttt{cvae}}$.

(d) Model $\mathcal{M}^{\star}_{\texttt{cvae}}$.

Figure 25: Calibration statistics, computed as described in the Sampling from the posterior section, of sampling form the posterior using all previously evaluated models.

namely dust mass and supernova luminosity.

Similar trends appear in the comparative study of $\mathcal{M}^{\star}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$. Both models outperform $\mathcal{M}_{\texttt{stan}}$, but this time by an even smaller margin. However, in this case, the CVAE variant performs worse than $\mathcal{M}^{\star}_{\texttt{lin}}$ across all features. It's also noteworthy that, this time, the standard model exhibits lower errors than those calculated over the entire validation dataset.

Considering that both $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$ achieved such a similar performance, it is challenging to draw definite conclusions about which performed the best. However, considering that the regular likelihood model $\mathcal{M}_{\texttt{lin}}$ showed the most balanced bin distribution in Figure 25, and it proved to have much faster runtimes than the CVAE variant, I follow with it to the final evaluation, where I marginalize out most of the filers, leaving the ones contained in the set $\mathcal{F}$.

I was able to generate a considerably larger number of samples for the routine with multiple filters marginalized out, as shown in Table 4. However, this

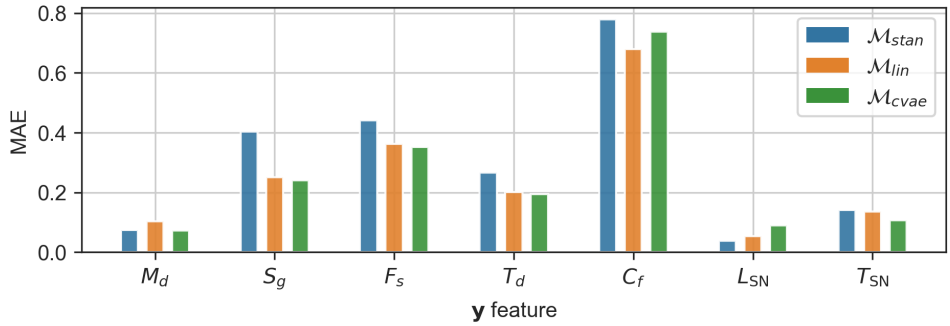



Figure 26: Comparison of MAE between the ground-truth $\boldsymbol{y}$ and the average of the posterior samples obtained from MCMC sampling with models $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$, along with MAE obtained through directly predicting $\boldsymbol{y}$ with $\mathcal{M}_{\texttt{stan}}$.

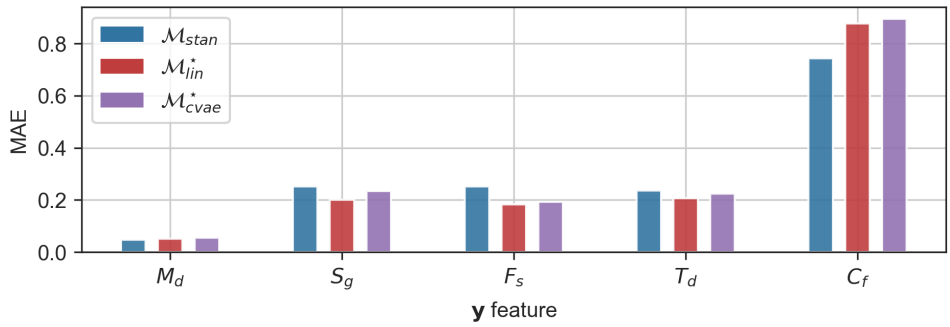



Figure 27: Comparison of MAE between the ground-truth $\boldsymbol{y}$ and the average of the posterior samples obtained from MCMC sampling with models $\mathcal{M}^{\star}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$, along with MAE obtained through directly predicting $\boldsymbol{y}$ with $\mathcal{M}_{\texttt{stan}}$.

| Total | Retained | Unfinished | Filtered out | Avg. runtime (min) |
|---|---|---|---|---|
| 585 | 277 (47%) | 197 (34%) | 111 (19%) | 123 |

Table 4: Statistics of MCMC sampling from the posterior using $\mathcal{M}_{\texttt{lin}}$, but done only with filters from the set $\mathcal{F}$.

process took much more time on average per sample. Furthermore, a greater percentage of MCMC routines were unfinished, and there was a corresponding increase in the number of rejected sampling routines due to inadequate MCMC diagnostic metrics. As a result, although I ran NUTS for more data entries in this scenario, the total number of retained samples is relatively similar to the

one obtained when sampling with all narrow and wide JWST filters.

Examining the confidence distribution in Figure 28 reveals a pattern similar to the one observed with the full filter set, where more samples tend to fall into higher-value bins. However, this effect is much more pronounced in this case. The bins within the range of 0.3-0.6 show relatively good calibration. Thus, the extreme counts on the higher end come at the cost of miscalibration on the lower one, where the 0.1-0.3 bins are heavily underrepresented in terms of counts.

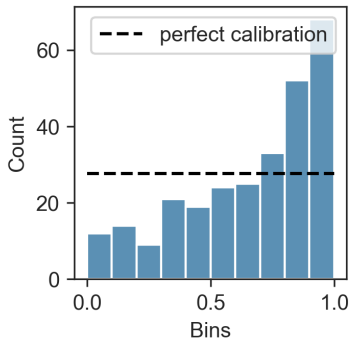



Figure 28: MCMC calibration of the sampling done using the $\mathcal{M}_{\texttt{lin}}$ model with only a subset of JWST filters used coming from set $\mathcal{F}$.

Thanks to the quantities of generated samples with both $\mathcal{M}_{\texttt{lin}}$ run with all filters and some marginalized, the intersection of successful MCMC runs was 124, which allows for a much better assessment of their prediction MAE (Figure 29). As expected, the sampling routine with only a subset of filters performs worse than one with a complete set of filters. Yet, both configurations outperformed $\mathcal{M}_{\texttt{stan}}$ on most features. The only exceptions were the features that proved to be the easiest to predict, such as dust mass and supernova luminosity, where $\mathcal{M}_{\texttt{stan}}$ exhibited slightly better performance. However, given the already low errors for these features, any potential improvement margin is relatively small.

It is again worth noting that standard model's errors closely align with those calculated across the entire validation dataset. The most significant difference between the sampling with complete filters and the one with only subset $\mathcal{F}$ is visible in the features related to SNe.

Finally, to evaluate whether reformulating the representation of $p(\boldsymbol{y}|\boldsymbol{x}, z)$ using the Bayesian framework provides a better estimate of the true distribution than multivariate Gaussian, as in the case of $\mathcal{M}_{\texttt{stan}}$, I present two examples of the distribution of MCMC samples.

In Figure 30, we can see a multimodal posterior distribution for grain size. Furthermore, when examining the plot showing dust temperature versus dust mass, we see an inversely proportional relation between dust mass and dust

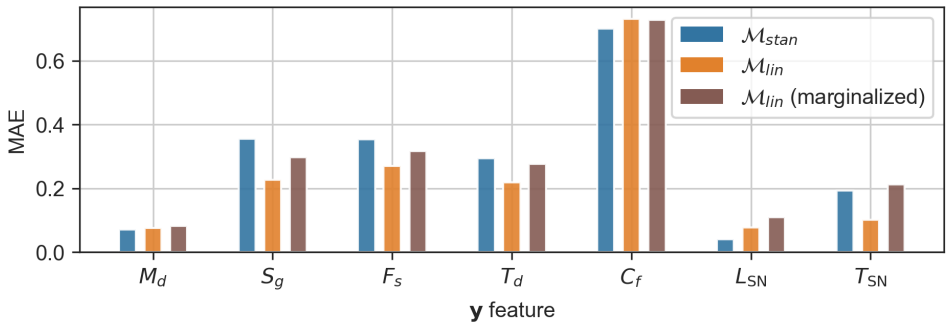



Figure 29: Comparison of MAE between the ground-truth $\boldsymbol{y}$ and the average of the posterior samples obtained from MCMC sampling with model $\mathcal{M}_{\texttt{lin}}$, where in one case, all JWST filters were used, and in the second, only a subset F was employed. Additionally, these are presented along the MAE, obtained through directly predicting $\boldsymbol{y}$ with $\mathcal{M}_{\texttt{stan}}$.

temperature. The model produces relatively accurate predictions with narrow uncertainty intervals, except for the clump filling factor and silicate fraction, where both span nearly the entire range of values present in the dataset. Often, the models produced much more narrow distributions, as shown in Figure 31.

In the case of sampling with a subset of filters, we continue to observe similar trends. However, here, we can see multimodality in the SN temperature distribution. Therefore, Figure 32 concludes the final results of the thesis.

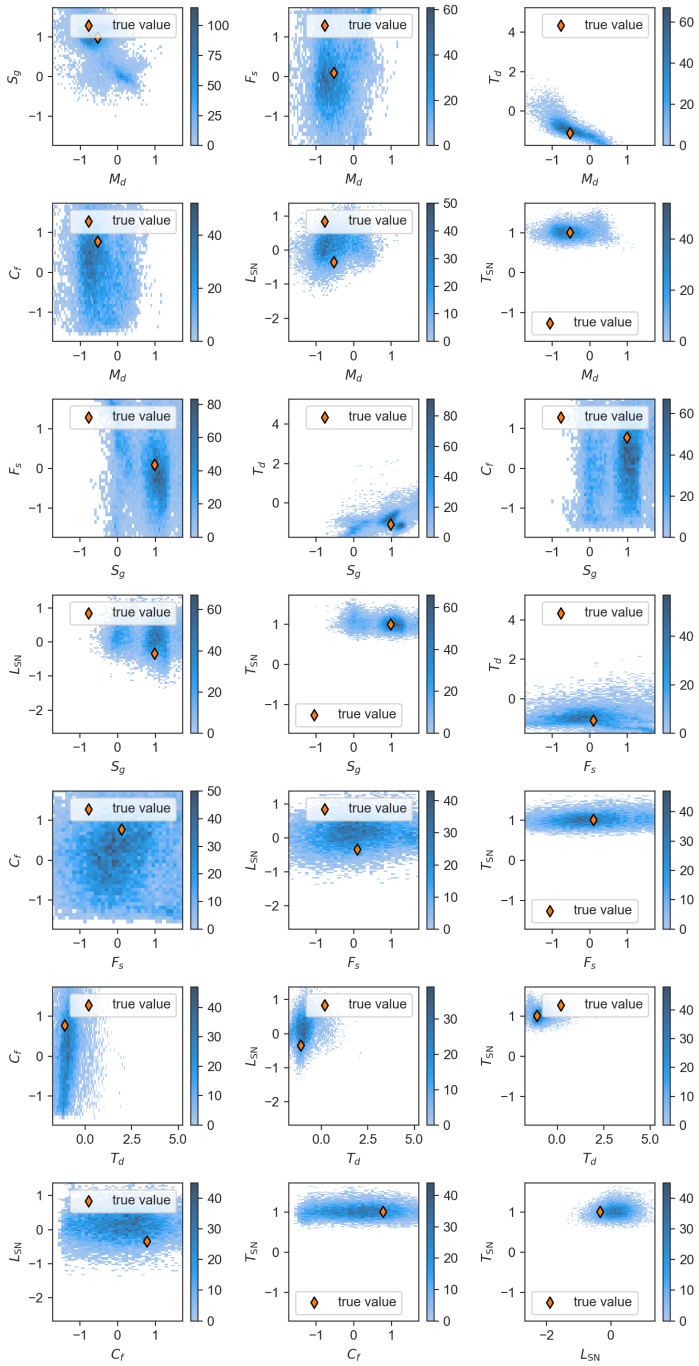



Figure 30: Example estimate of posterior distribution with MCMC samples obtained using model $\mathcal{M}_{\texttt{lin}}$. The figure shows a histogram plot, and the bars on the right represent the number of samples each square represents.

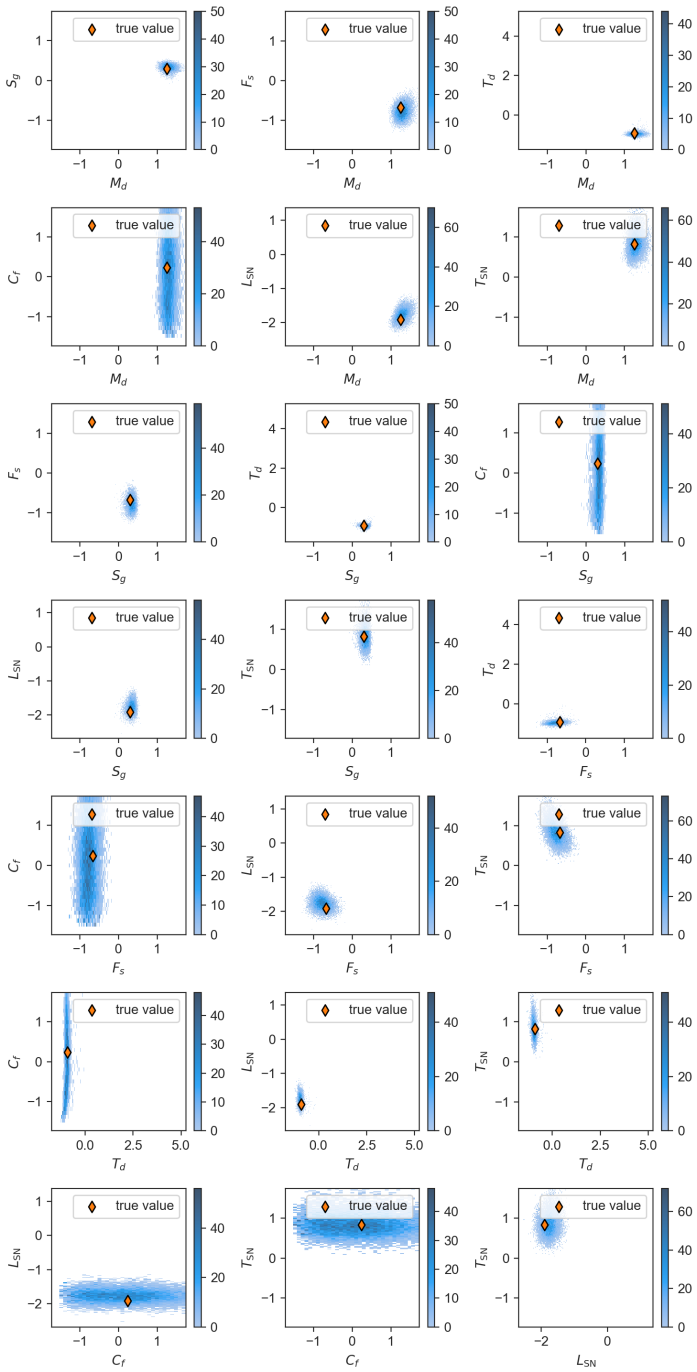



Figure 31: Example estimate of posterior distribution with MCMC samples obtained using model $\mathcal{M}_{\texttt{lin}}$. The figure shows a histogram plot, and the bars on the right represent the number of samples each square represents.

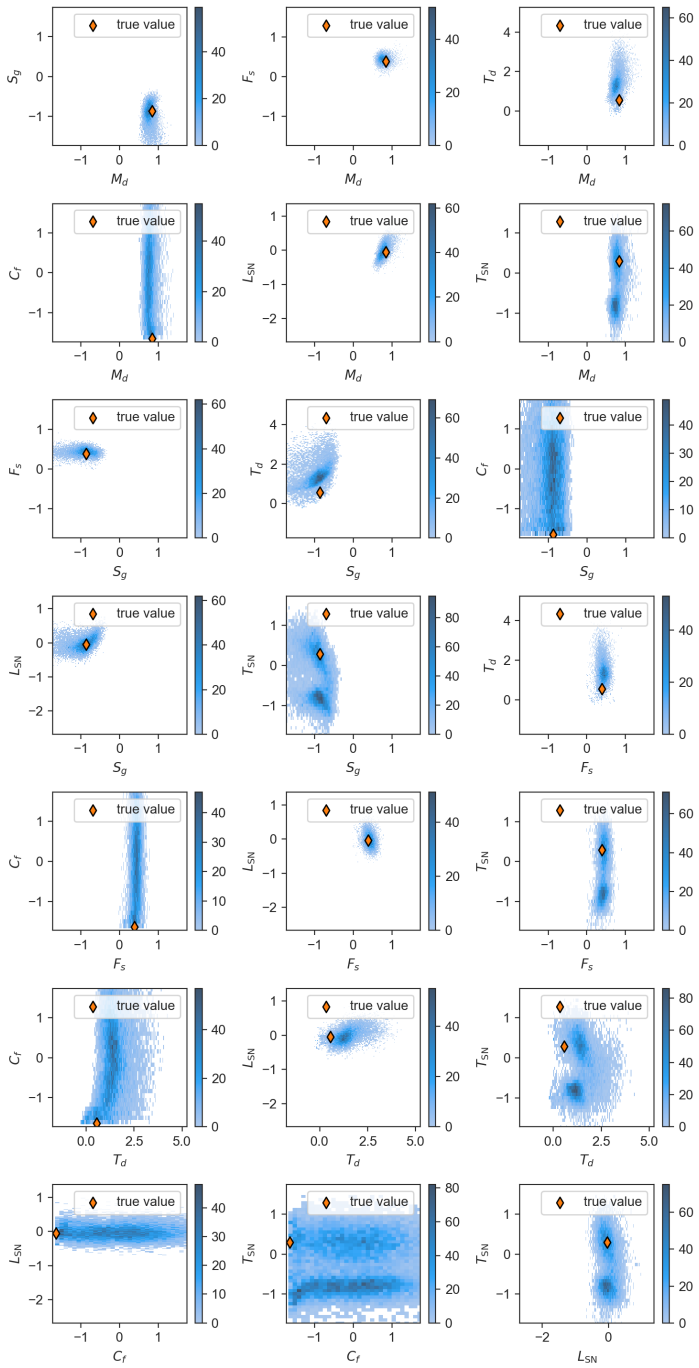



Figure 32: Example estimate of posterior distribution with MCMC samples obtained using model $\mathcal{M}_{\texttt{lin}}$, where only filters from set $\mathcal{F}$ were used. The figure shows a histogram plot, and the bars on the right represent the number of samples each square represents.

# 6 Discussion

As the primary achievement of this thesis, I managed to successfully represent the distribution of the cosmic dust properties $p\{\boldsymbol{y}|\boldsymbol{x}, z\}$ using a Bayesian inference, which required training the DL models representing density $p\{\boldsymbol{x}|\boldsymbol{y}, z\}$ and employing a robust MCMC technique. The results suggest that the limitations of current approaches to the problem of inferring dust properties based on the EMR it emits using machine learning are successfully improved upon by the presented solution. The example plots of approximated posterior distribution in Figures 30 and 32 seem to confirm a claim that the $\boldsymbol{y}$ samples are likely not normally distributed, as we can see a more complex structure of the density, sometimes even containing more than one mode in certain features.

Furthermore, comparing the reported MAEs of predictions obtained through MCMC sampling and ones generated by the standard model $\mathcal{M}_{\texttt{stan}}$, we can see that the method proved to work even better than the $\mathcal{M}_{\texttt{stan}}$, further solidifying the previous claim. I also successfully demonstrated that the introduced framework is easily adjustable for the setting with a limited number of JWST filters available and that it is competitive with sampling where all filters were used.

Also, I successfully managed to evaluate the second objective of this thesis, which involved comparing a standard variant of the likelihood model with a more complex architecture, the Conditional Variational Autoencoder. The evaluations of $\mathcal{M}_{\texttt{cvae}}$ closely align with those of the regular model, suggesting that a multivariate Gaussian distribution is a proper choice for modeling the distribution of filter responses $p(\boldsymbol{x}|\boldsymbol{y}, z)$, as employing a more complex representation did not yield significant improvements.

## 6.1 Interpretation of the results

To provide a broader context to these findings, I proceed with the analysis of the evaluation of the likelihood models, both regular and CVAE variants. The first important observation required to address is the unusual fact that consistently, across all experiments (including training of $\mathcal{M}_{\texttt{stan}}$), the validation loss was much lower than the training one. This observation can be the result of the way the dataset was split since the initial training set was partitioned into smaller subsets, including the validation set, after applying redshift variations to the samples. Consequently, both the final training and validation subsets could share similar $\boldsymbol{x}$ variants, rendering the validation set potentially easier to optimize. Additionally, the training set likely contained more outliers, resulting in higher loss values.

Another point regarding training statistics relates to the substantial increase in loss values when computed on the test set, a trend that does not apply as strongly only to the linear model trained with SNe features. Significant fluctuations in validation loss values are evident as training progresses in

most models, except $\mathcal{M}_{\texttt{lin}}$, which may provide some insights. Given that the final model parameters are selected based on the lowest loss values achieved on the validation set, it is possible that these models overfitted to the validation set. In contrast, the $\mathcal{M}_{\texttt{lin}}$ model exhibited more stable validation losses, possibly explaining better generalization capabilities on the test set.

The final observation from training concerns the characteristic bump in the MAE calculated over the training and validation data. Interestingly, this phenomenon is not as prominent when training $\mathcal{M}_{\texttt{stan}}$. This difference might relate to the fact that the number of parameters needed to parametrize the covariance matrix grows quadratically with the dimensionality of the mean/input. As a result, the optimizer might prioritize optimizing the covariance initially, leading to a delay in the reduction of MAE during the early stages of training. Since the difference between the number of parameters required to describe covariance and mean in the case of $\mathcal{M}_{\texttt{stan}}$ is smaller, this phenomena is not observed in its training.

While the regular and CVAE variants of the likelihood models generally produced similar results, it is interesting to note that $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$ showed a noticeable difference in per-feature RMSE when it comes to lower wavelength filters. Thus, it suggests that when models have access to information about supernova luminosity and temperature, it results in a more intricate distribution in lower wavelengths, as they are most affected by the EMR of SNe. $\mathcal{M}_{\texttt{lin}}$ addresses this by assigning high uncertainties in those cases as it is not capable of representing this more complex distribution of low wavelength filter responses, while CVAE manages to capture it, resulting in lower RMSE and uncertainties. On the other hand, in the case of models trained without SNe features, the absence of this information leads to a general uncertainty in predictions without the emergence of this complexity. Otherwise, in this case, the CVAE variant would perform better than a regular likelihood model instead of displaying identically high RMSE errors.

The aspect at which all models struggled was the general underconfidence in the predictions. Likely, it is caused by the $\sigma^2_{\texttt{min}}$, as it sets a hard limit on the generated variances. Considering that the RMSE is extremely small for many samples if the $\sigma^2_{\texttt{min}}$ is too high, it may cause the predicted uncertainties to be unsuitably large, resulting in an underconfident model. As this trend likely also transfers to the sampling from the posterior, it should be addressed, for example, by the more sophisticated search for suitable $\sigma^2_{\texttt{min}}$. Rethinking how to generate covariances in a more numerically stable way could also provide aid, as this is the primary factor constraining the value of minimal variance.

Even though most of the presented average covariances show uncertainties around NIR filters, it is because they are the most difficult to predict, as reflected by the per-feature RMSE. However, it doesn't mean that MIR filters are uncorrelated since, under closer inspection, they still show more correlation between each other than when correlated with NIR filters, which is most visible in Figure 22a. It is in line with the wavelengths that the filters cover shown in

Figure 4, as the NIR filters often overlap each other, and the same can be said about MIR filters, while there is no overlap between NIR and MIR filters.

Before discussing the quality of posterior samples in more detail, it is important to first provide some insights into the per-feature RMSE of $\mathcal{M}_{\texttt{stan}}$. Results show that mass and temperature are the easiest to predict when we exclude SNe features. Conversely, grain size and silicate fraction exhibit notably high errors, which aligns with the findings presented by Ansari et al. [2022]. Evidently, the clump filling factor stands out the most with the highest error. In preliminary training of the models, I used only the first four $\boldsymbol{y}$ features, leaving out SNe features and the clump filling factor. After adding the clump filling factor, there was only a marginal improvement in the performance of the likelihood models. Thus, there is likely no strong connection between EMR and the $C_f$ since it is almost impossible to predict accurately from $\boldsymbol{x}$, while also it does not convey much information in the opposite prediction direction (from $\boldsymbol{y}$ to $\boldsymbol{x}$).

Nevertheless, the most crucial aspect to analyze is the performance of the MCMC routine. When we compare the percentage of retained samples between $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{lin}}$, we observe that the former had a higher percentage of successful runs, likely because it had fewer features based on which it could get rejected. The same trend was not visible in the $\mathcal{M}_{\texttt{cvae}}$, likely because of the time limit on sampling. However, as I mentioned in the Methods section, preliminary experiments without the time limit set showed that sampling routines using CVAE models that had runtimes longer than the set limit failed to meet the minimum MCMC convergence diagnostic requirements.

It is difficult to say why sampling in the case of models $\mathcal{M}_{\texttt{cvae}}$, $\mathcal{M}^{\star}_{\texttt{lin}}$, and $\mathcal{M}^{\star}_{\texttt{cvae}}$ showed such disproportional distributions of bin counts in the MCMC calibration plot. This outcome suggests that the resulting estimate of posterior distribution tended to be overconfident since the ground truth $\boldsymbol{y}$ was consistently less likely under the $p\left(\boldsymbol{y}|\boldsymbol{x}, z\right)$ than 90% of the MCMC samples. Interestingly, such a trend was not observed in the case of the $\mathcal{M}_{\texttt{lin}}$ model. A possible explanation could be related to overfitting. Going back to the analysis of NLL computed on the test set, from all models, $\mathcal{M}_{\texttt{lin}}$ exhibited the lowest decrease in this metric compared to the validation set. This observation hints that these models might have overfitted to the training data, which could have contributed to their overconfidence during sampling.

As mentioned in the 5, $\mathcal{M}_{\texttt{lin}}$ and $\mathcal{M}_{\texttt{cvae}}$ achieved competitive, if not better, performance on the MAE plots (Figure 26) compared to the standard model. Considering that the distribution of errors of $\mathcal{M}_{\texttt{stan}}$ closely matches the corresponding distribution of errors calculated over the whole validation dataset, it can suggest that even though the results were computed using only a small set of samples, they might be still somewhat representative of the whole validation dataset. The same cannot be said about the performance of sampling with models $\mathcal{M}^{\star}_{\texttt{lin}}$ and $\mathcal{M}^{\star}_{\texttt{cvae}}$, as all reported errors (including ones of $\mathcal{M}_{\texttt{stan}}$) were lower when compared to those of $\mathcal{M}_{\texttt{stan}}$ calculted on

the whole validation dataset. Nevertheless, they showed a competitive performance when compared to standard model as well.

The outcome of running MCMC with only a subset of filters was a success, as the MAE of these predictions was only marginally worse compared to the full filter setting. Moreover, they often achieved lower errors than $\mathcal{M}_{\texttt{stan}}$. Nevertheless, sampling with a reduced number of filters does come with trade-offs. First, in this case, the results turned out to be underconfident. Thus, it suggests that the ground-truth samples were frequently more likely under the model than the ones obtained through the MCMC routine. This underconfidence might occur because, with less information available for sampling, the estimated $p(\boldsymbol{y}|\boldsymbol{x}, z)$ distribution becomes broader. Furthermore, we can also see that generally, MCMC run with $\mathcal{F}$ filters marginalized was more often unfinished when compared to the full filter setting, suggesting that it is more resource-costly and difficult to sample from the posterior under limited information.

The final samples solidify the argument that the method presented in this thesis often manages to approximate the posterior distribution accurately. By looking at example plots of estimated posterior density, we can see that the posterior density is often not best described by Gaussian distribution, as we have seen multimodal distributions for grain size, as well as SNe temperature. The former is expected since silicate fraction is the most challenging feature to predict, as shown by MAEs of $\mathcal{M}_{\texttt{stan}}$, and thus likely it has an intricate distribution. In the latter case, it may be caused by the fact that the samples were obtained with most NIR filters marginalized, which usually are most affected by the EMR of SNe. Nevertheless, models often manage to produce very narrow and accurate distributions, as in Figure 31. Hence, these models have the capacity to generate precise predictions with tight uncertainties while also providing intricate posterior estimates for more challenging data points.

It also appears that the modeled distributions reflect the anticipated real-world relationships between the features. For instance, the inverse correlation between dust mass and temperature arises from the fact that as dust becomes heavier, its emitted EMR becomes more intense, and the same principle applies to temperature. Consequently, to maintain a similar EMR, they must be inversely proportional since if both were to increase, the signal would generally become stronger.

## 6.2 Limitations

Even though the implemented solution proved successful, it is the first time a framework like this has been applied to the problem of inferring space dust properties. Thus, certain limitations arise as addressing all the shortcomings would be unfeasible in the time limit set for the work on this thesis.

At a fundamental conceptual level, one significant constraint applies to the entire proposed approach. Specifically, the primary objective of the thesis is to

estimate as accurately as possible the true distribution of cosmic dust features $p(\boldsymbol{y}|\boldsymbol{x}, z)$. In the proposed Bayesian framework, I accomplish it indirectly by maximizing the likelihood of samples of JWST filter responses from the dataset under the density $p(\boldsymbol{x}|\boldsymbol{y}, z)$ by optimizing the model through stochastic gradient descent. However, the optimal solution under $p(\boldsymbol{x}|\boldsymbol{y}, z)$ does not necessarily have to lead to optimal $p(\boldsymbol{y}|\boldsymbol{x}, z)$, as the posterior density also contains evidence of $p(\boldsymbol{x}|z)$. Thus, the models representing likelihood density $p(\boldsymbol{x}|\boldsymbol{y}, z)$ should be trained ideally by directly maximizing the likelihood of data under $p(\boldsymbol{y}|\boldsymbol{x}, z)$, which, of course, is a challenging task since the evidence is intractable, and thus addressing it was outside the scope of this thesis.

As mentioned in the Preprocessing section, it's essential to revisit the dataset splitting process into training, validation, and test sets, as it should be performed before generating the redshift variants, which was not done in this work. This adjustment seems crucial as it likely could have a significant influence on training outcomes, potentially alleviating issues related to overfitting. Furthermore, as described in the Dataset section, considering a filter as non-existing at the threshold of reported magnitude at 42.5 is somewhat arbitrary. A more sophisticated approach could be obtained by accounting for JWST detection limits.

Regarding training and the model's design, there is a chance that the chosen architecture and training parameters could be suboptimal. Even though I did preliminary experiments to derive the best setups, considering that the thesis already encompasses the evaluation of many models with two different architectures, the complex parameter search was impossible to conduct in the timeframe given for this work. It might be especially applicable to the CVAE model, as there are many ways of expanding the specific design chosen for this work. For instance, training the model with the prior condition on $\boldsymbol{x}$ could provide a more robust model. It could also be worthwhile to check if a change in architecture would give better results. For example, given that the prior is conditioned on $\boldsymbol{x}$, one could remove the direct network. Nevertheless, as experiments show, the Gaussian distribution is likely the right choice to represent $p(\boldsymbol{x}|\boldsymbol{y}, z)$. Thus, these adjustments might not yield drastic improvements.

As mentioned in the Results section, the training and evaluation of MSTAN was not done with as much care as in the case of likelihood models. Consequently, its comparatively weaker performance when compared against the framework introduced in this thesis may be attributed to these limitations. However, it's worth noting that this thesis serves as an initial exploration of the Bayesian approach to the problem, and thus, even the suboptimal performance of $\mathcal{M}_{\texttt{stan}}$ is sufficient to prove its competitiveness.

Naturally, this moves us to the limitations regarding the MCMC sampling, and considering the complexity of the method, several factors could be improved upon. While the MAE and calibration plots, derived from posterior samples, offer valuable insights and indicate that the number of successful

MCMC runs was adequate to match the method's performance on the whole test set, it is crucial to acknowledge that the evaluation was performed on a limited number of dataset entries. Thus, to solidify the validity of the findings, the count of successful MCMC runs should be expanded.

Additionally, the time limit imposed on the sampling process remained a limiting factor during the evaluation. Since I use a ground-truth $\boldsymbol{y}$ as a sampling starting point, exceeding that constraint could mean that the estimated $p(\boldsymbol{y}|\boldsymbol{x}, z)$ is just far from $\boldsymbol{y}$. Thus, errors for these samples could be higher, meaning that presented results might be biased towards lower errors. The same can be said about the runs that were completed before the time limit but did not meet the minimal MCMC convergence requirements, as it may indicate that the number of samples for which the method was run was insufficient to reach a density distant from ground-truth $\boldsymbol{y}$.

Also, even though I proposed some reasons explaining the observed miscalibration of MCMC sampling, the topic should be explored to confirm them decisively, as there might be some other causes of this result.

Finally, this work is only limited to the simulated dataset generated specifically for this thesis. Therefore, to decisively check if the method is competitive with current standard approaches of inferring space dust properties, it is imperative to evaluate the method using real-world data.

# 7 Conclusion

In this study, I managed to comprehensively address two of its primary research objectives: firstly, the effective representation of the posterior $p(\boldsymbol{y}|\boldsymbol{x}, z)$, and secondly, the evaluation of whether CVAE can provide a better representation of the likelihood density $p(\boldsymbol{x}|\boldsymbol{y}, z)$. The approach proposed for representing $p(\boldsymbol{y}|\boldsymbol{x}, z)$ through Bayesian inference has demonstrated its robustness, consistently delivering highly accurate predictions. Moreover, in situations where predictions were burdened with higher errors, it provided intricate estimates of uncertainties.

Furthermore, I have showcased the adaptability of this method in scenarios where multiple JWST filter responses are compromised. I achieved this by assessing the sampling routine in cases where only a subset of JWST filters was employed, drawing from the work of Ansari et al. [2022]. The authors have indicated that this particular filter set represents a minimal requirement to gain results comparable to those obtained with all filters available. The findings presented in this thesis are in line with these results.

Interestingly, there was no significant difference in the performance of the regular likelihood model and variants where it was represented using CVAE. Thus, the multivariate Gaussian distribution is a good fit for representing the likelihood density $p(\boldsymbol{x}|\boldsymbol{y}, z)$. What was even more striking is that the difference in performance between these two variants was even lower when trained only on the $\boldsymbol{y}$ features concerning the dust. It is likely because this missing information is easily represented by just broad Gaussian density without showcasing intricate patterns. Importantly, all models achieved competitive performance when compared to the standard approach of training the DL model to directly predict the distribution of dust features based on reported JWST filter responses.

The faced limitations where manifold. From very conceptual, such as the weakness of not training the likelihood models by directly optimizing the posterior, to more technical ones, such as the MCMC method getting stuck on some samples. A very crucial one, the incorrect separation of samples into train test and validation sets, could be easily avoided at the beginning of the work on this project. Unfortunately, it was discovered very late, and thus remained throughout all conducted experiments. It likely impacted all the steps of the project, from high loss values of likelihood models on the test set to improper calibration of MCMC routine. Nevertheless, even with these considered, the method presented in this thesis proved successful and can serve as a strong basis for further exploration.

Overall, the results of this research not only emphasize the method's effectiveness in representing posterior distribution accurately but also highlight that multivariate Gaussian is likely a good choice for representing the likelihood density. Even though it is the first attempt at employing such an approach to the problem of inferring dust properties, it already shows promis-

ing results. Considering that despite numerous limitations faced in this work, the method still proved effective, improving on them might yield results that bring us closer to finding out whether CCSNe are largely responsible for producing considerable quantities of the space dust observed, a subject so crucial to the understanding of the future and past evolution of interstellar objects.

# Bibliography

Zoe Ansari, Christa Gall, Roger Wesson, and Oswin Krause. Inferring properties of dust in supernovae with neural networks. *Astronomy & Astrophysics*, 666:A176, oct 2022. doi: 10.1051/0004-6361/202243078. URL https://doi.org/10.1051%2F0004-6361%2F202243078.

Michaël Bensimhoun. N-dimensional cumulative function, and other useful facts about gaussians and normal densities by michaël bensimhoun (reviewed by david arnon). 2013.

Hao Chen, Zhanfeng Mo, Zhouwang Yang, and Xiao Wang. Theoretical investigation of generalization bound for residual networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 2081–2087. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/288. URL https://doi.org/10.24963/ijcai.2019/288.

Barbara Ercolano, M.J. Barlow, P.J. Storey, and X.-W. Liu. Mocassin: 3d photoionisation and dust radiative transfer modelling of pne. In L. Stanghellini, J. R. Walsh, and N. G. Douglas, editors, *Planetary Nebulae Beyond the Milky Way*, pages 196–198, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-34270-0.

C. Gall, J. Hjorth, and A. C. Andersen. Production of dust by massive stars at high redshift. *The Astronomy and Astrophysics Review*, 19(1), sep 2011. doi: 10.1007/s00159-011-0043-7. URL https://doi.org/10.1007%2Fs00159-011-0043-7.

Jonathan P. Gardner, John C. Mather, Mark Clampin, Rene Doyon, Matthew A. Greenhouse, Heidi B. Hammel, John B. Hutchings, Peter Jakobsen, Simon J. Lilly, Knox S. Long, Jonathan I. Lunine, Mark J. Mccaughrean, Matt Mountain, John Nella, George H. Rieke, Marcia J. Rieke, Hans-Walter Rix, Eric P. Smith, George Sonneborn, Massimo Stiavelli, H. S. Stockman, Rogier A. Windhorst, and Gillian S. Wright. The james webb space telescope. *Space Science Reviews*, 123(4):485–606, apr 2006. doi: 10.1007/s11214-006-8315-7. URL https://doi.org/10.1007%2Fs11214-006-8315-7.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna M. Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A survey of uncertainty in deep neural networks. *CoRR*, abs/2107.03342, 2021. URL https://arxiv.org/abs/2107.03342.

Fengxiang He, Tongliang Liu, and Dacheng Tao. Why resnet works? residuals generalize, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo, 2011.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. *CoRR*, abs/1807.00263, 2018. URL http://arxiv.org/abs/1807.00263.

Meta AI. PyTorch (v1.13), 2023a. URL https://pytorch.org/docs/1.13/. Accessed on May 30, 2023.

Meta AI. PyTorch Distributions (v1.13), 2023b. URL https://pytorch.org/docs/1.13/distributions.html. Accessed on May 30, 2023.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114. URL http://link.aip.org/link/?JCP/21/1087/1.

Radford M. Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54:113–162, 2010.

NumFOCUS. ArviZ (v0.15.1), 2023. URL https://python.arviz.org/en/v0.15.1/api/index.html. Accessed on May 30, 2023.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek V. Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. Neural paraphrase generation with stacked residual LSTM networks. *CoRR*, abs/1610.03098, 2016. URL http://arxiv.org/abs/1610.03098.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

Space Telescope Science Institute. JWST User Documentation, 2023a. URL https://jwst-docs.stsci.edu. Accessed on May 30, 2023.

Space Telescope Science Institute. JWST User Documentation: MIRI Filters and Dispersers, 2023b. URL https://jwst-docs.stsci.edu/jwst-mid-infrared-instrument/miri-instrumentation/miri-filters-and-dispersers#MIRIFiltersandDispersers-Imagingfilters. Accessed on May 30, 2023.

Space Telescope Science Institute. JWST User Documentation: NIRCam Filters, 2023c. URL https://jwst-docs.stsci.edu/jwst-near-infrared-camera/nircam-instrumentation/nircam-filters#NIRCamFilters-Filtertransmissions. Accessed on May 30, 2023.

Uber AI Labs. Pyro (v1.8.4), 2023. URL https://docs.pyro.ai/en/1.8.4/. Accessed on May 30, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Aki Vehtari, Andrew Gelman, Daniel Simpson, Bob Carpenter, and Paul-Christian Bürkner. Rank-normalization, folding, and localization: An improved r^ for assessing convergence of MCMC (with discussion). *Bayesian Analysis*, 16(2), jun 2021. doi: 10.1214/20-ba1221. URL https://doi.org/10.1214%2F20-ba1221.

Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.