**Related work**

**Paper 2 – Robles et al. (2019)**

Robles et al. (2019) apply training data from the EAGLE suite to train a Generative Adversarial Network (GAN) to generate merger trees that are well-constructed and consistent in terms of the original merger trees from the large volume simulations. The GAN learns to recognize and generate the patterns, structure, and features of the merger trees, which makes it possible to generate realistic merger trees. The advantage of a GAN compared to other merger tree generation methods is that it is computational inexpensive, while it manages to produce realistic and consistent merger trees [2]. Robles et al. construct an encoder-decoder deep convolutional GAN (DCGAN) architecture that is based on (Dosovitskiy et al., 2015; Radford et al., 2015), which utilize row- and column-wise convolutional filters that are supposed to capture behavior within and between branches [103], [22], [101]. Treating halo merger trees as a matrix generation problem, the CGAN learns the merger trees matrix representation and how to generate new samples.

The data which Robles et al. (2019) experiment with consists of halo merger trees with only 6 branches, which is relatively uncomplex merger trees, but they are the most represented of merger trees given the number of branches. However, the paper experiment with different combinations of variables used in the training data to obtain the best results. The best results are found by evaluating the progenitor masses of the merger trees, regardless of how many and which variables are in the training data. This is done by comparing the distributions of the mass gain and loss for all configurations to the original data, which show that using all three variables in the training process yields the best results.

Furthermore, Robles et al. (2019) evaluate the probability distribution of the reconstruction of the distance between merging progenitors several snapshots before the merger happens given 2 and 3 variable configurations. The KS test is then utilized to compare the distributions to the original data. Both 2 variables configurations (mass/distance and mass/progenitor type) produce equally good results. However, the 3 variable configuration produces an even better results in terms of agreement with the original data.

The generated merger trees of Robles et al. (2019) manage to capture the structure of the original merger trees and have fairly good complexity. However, they report that there are some consistency issues regarding reconstruction in the case of absence of progenitors, where the mass variables reproduce a very small number instead of exactly zero.

**Paper 3**

Robles et al. (2022) extend the methods, experiments, and results from Robles et al. (2019) in two main directions. The first one is that they experiment with different merger tree sizes, in terms of number of branches in the trees, as training input compared to Robled et al. (2019) which only utilized merger trees with 6 branches. This require the GAN model to be modified to handle the input, but apart from that, the GAN model architecture is similar to the 2019 version. The second extension is that Robles et al. (2022) experiment with different halo finder – tree builder algorithms, namely the SUBFIND halo finder with D-TREES tree constructer combination and the ROCKSTAR halo finder with *consistenttrees* merger tree builder combination [3]. The reason for the experiment with the two combinations of halo finder and tree builders are described in the merger tree section. Different halo finders and merger tree builders yield different results as described in the merger tree section, and that is the case in this paper as well. The SUBFIND combination outperforms the ROCKSTAR combination in most of the experiments, however, this is reasoned to be because of more data points in the SUBFIND combination.

The paper continues the effort from Robles et al. (2019) in terms of experimenting and testing different input variables and combination of the variables. They obtain the same results, namely that using all three variables, that is, mass, distance and subhalo type, performs the best. To evaluate this, the KS test is applied, which is the same procedure as in the 2019 paper, however, here, the evaluation focuses on the mass assembly history only. That means the performance regarding variables is evaluated only in terms of how well the generated mass matches the original mass distribution.

Both papers provide good results in terms of reproduction and generation of merger trees. The results of this are a new method to simulate and “create” merger trees by using deep learning algorithms like GANs. The only limitation of this method is that it requires a large amount of already existing merger trees as training data and memory resources. The main advantage of using deep learning methods to generate merger trees is that it is computational inexpensive and relatively fast compared to other methods, while being able to generate realistic trees. The results from these papers open up to explore other deep learning methods in this specific topic, which is what I aim to do in this paper by applying diffusion methods to the same task.

**Improvement from gan to diffusion**

* **Paper 10?**