Learning Tumor Evolution With Diffusion Models

Siddharth Sabata
Carnegie Mellon University
ssabata@andrew.cmu.edu

Russell Schwartz
Carnegie Mellon University
russells@andrew.cmu.edu

Abstract

Reconstructing tumor clonal phylogenies is vital for understanding intra-tumor heterogeneity, anticipating therapy resistance, and advancing precision oncology. Traditional unsupervised model-based approaches infer evolutionary trees from bulk sequencing data but are constrained by lengthy runtimes and instability when applied to sparse datasets. Data-driven generative modeling offers an alternative by directly learning complex statistical distributions. Learning distributions of tumor phylogenies enables greater generalizability and faster inference. This work explores the mathematical feasibility of using generative discrete diffusion for phylogenetic reconstruction. We introduce DiPhy, an adapted discrete graph diffusion model for unconditional phylogeny generation. We develop a synthetic training dataset and train two 7.1 million parameter models on purely synthetic data, achieving 92.4% structurally valid graphs at test time. This work serves as an initial step toward leveraging generative deep learning for fast, reliable tumor phylogeny inference.

1 Introduction

Cancer is fundamentally driven by evolution: cell subpopulations, known as clones, within a tumor continuously acquire mutations, leading to diverse subpopulations that influence treatment responses, disease progression, and metastasis. Clones can be characterized by unique mutation profiles. Liquid biopsies, which detect tumor-derived circulating DNA (ctDNA) in blood samples, offer a minimally invasive, low-cost method to monitor cancer progression [4][15]. While they show great promise as a diagnostic tool, integrating noisy ctDNA data into accurate models of tumor evolution remains challenging.

Using ctDNA in a longitudinal fashion to predict a patient's cancer relapse has shown clinical promise [1]. The recent Mase-phi framework builds on this idea by optimizing the selection of informative ctDNA markers (mutations) through Bayesian inference and optimization, significantly reducing biomarker panel costs for understanding subclonal dynamics [9]. The framework operates by first analyzing bulk sequencing reads from a high-resolution tissue biopsy through a probabilistic model. This model reconstructs the tumor's evolutionary history, termed a phylogeny, hundreds of times to generate a distribution of possible phylogenies. These tissue-based phylogenies then serve as a Bayesian prior. The framework subsequently incorporates lower-resolution longitudinal ctDNA bulk sequencing reads to update this phylogenetic distribution and identify the most significant biomarkers.

Most phylogeny reconstruction, or deconvolution, methods rely on model based unsupervised machine learning approaches, which often results in long and computationally expensive runtimes. This challenge is amplified when these methods are bootstrapped, as in frameworks like Mase-phi. These models learn pre-defined parameters through inference-time optimization. Different deconvolution methods have varying data requirements, with some utilizing small nucleotide variations (SNVs), copy number alterations (CNAs), and single cell sequencing data [8][3]. While incorporating multiple data modalities improves accuracy, it substantially increases clinical costs [11]. PhyloWGS distinguishes itself from other methods by functioning solely on sparse amounts of SNV data, making it the method of choice for Mase-phi [6]. However, because PhyloWGS is an unsupervised model based on Markov Chain Monte Carlo (MCMC) sampling, processing a single SNV panel can require over eight hours on a supercomputing cluster. Additionally, the stochastic nature of the model means that runs may occasionally fail.

Unconditional-diffusion models are a class of self-supervised deep generative models that excel at learning complex statistical distributions [5]. They directly learn data distributions with neural networks through training-time optimization. With roots in statistical physics, Denoising Diffusion Probabilistic Models (DDPMs) are powerful

generative models that have revolutionized the field of data synthesis, particularly in continuous domains like image generation [10][18]. The fundamental concept behind DDPMs is to progressively add noise to real data points until they become indistinguishable from random noise, and then train neural networks (called a denoiser) to reverse this process, effectively learning the data distribution. Many consumer grade diffusion models, such as DALL-E and Stable Diffusion, are conditional, meaning they generate samples given some kind of input (i.e text) [16][17]. Conditional diffusion models are supervised machine learning algorithms, requiring clearly labeled training data. Guidance methods help steer sampling given some input, while the denoiser models the data distribution. On top of learning complex data distributions well, diffusion models are significantly faster at inference due to their training-time optimization; they do not have to learn or optimize at test-time, but require labeled training data.

While creating a diffusion-based, end-to-end replacement for PhyloWGS is a massive undertaking, this work explores the foundation of conditioning a diffusion model to generate biologically relevant graphical structures from purely synthetic training data. This foundational work paves the way for further, larger-scale initiatives in dataset creation, model development, and training strategies to eventually create a modernized, clinically viable tumor deconvolution model.

2 Methods

We train DiPhy, a 7.1 million parameter graph diffusion model, on the task of unconditional phylogeny generation using synthetic data. DiPhy is an extension of DiGress, a general discrete graph diffusion model, with a customized graph encoding designed to be compatible with the raw simulated ground truth data. Figure 1 outlines the transformation of simulator data into the graph representation used in this work.

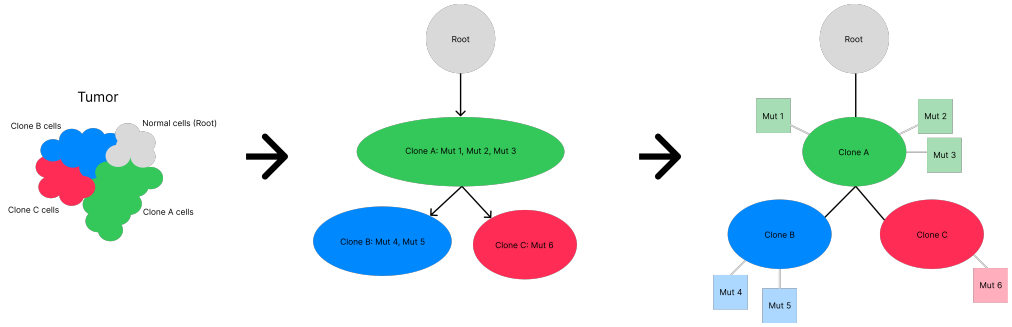

Figure 1: Understanding the dataset starting from raw simulator outputs to graph encoding. Phylogenies always have a root node (gray) that represent normal cells within a tumor, then bold colored nodes that represent clones, or tumor sub-populations. Because clones consist of multiple mutations (indicated as "Mut n"), the Newick tree (middle) contains multi-class nodes. These nodes indicate ancestry through the directed edges (black), meaning each child contains all the mutations of the parent. We unravel the clones (right) with specific undirected mutation edges (light gray) that connect to the corresponding clones, creating single class nodes. We also make the parental edges undirected (black). These changes were made for simplicity and model compatibility.

2.1 Model Selection

DiGress extends the DDPM framework specifically to discrete graphs, like molecules [21]. The process involves corrupting graph structures by progressively adding or removing nodes/edges and changing their categories, and training a graph transformer-based denoiser to reverse the process. Graph transformers are an adaptation of traditional transformer models, with the ability to learn neighborhood-level node/edge relationships instead of text relationships [20][7]. This approach has achieved state-of-the-art results in molecular graph generation, demonstrating significant improvements in generating valid graph structures even from extremely large datasets. This is significant because DiGress has already demonstrated that diffusion models can efficiently capture and reconstruct complex graph topologies. By adapting DiGress for unconditional generation of tumor phylogenies, we leverage the existing computational architecture and proven efficacy, significantly reducing the complexity of our development and enhancing confidence in our approach.

2.2 Simulator Selection

Training an unconditional generative version of DiGress requires only ground truth phylogenetic trees to which the model can iteratively add and remove noise. In cancer genomics, large and accurately labeled datasets for training models are notoriously hard to find. This challenge is particularly apparent for tumor phylogenies because generating a ground truth phylogeny from patient data is impossible. Therefore, we must turn to simulated data [13][14]. SISTEM addresses this challenge by providing realistic simulations of tumor dynamics [22]. This Python package simulates tumor growth, metastasis, and DNA sequencing under genotype-driven selection. It works in a two step process: it first simulates tumor growth until a minimum detectable population size is detected, providing a single compressed metadata file. Then it simulates sampling cells, constructs a clonal lineage, and generates synthetic bulk read counts based on input parameters. SISTEM also has the capability to generate single cell lineages and the associated data in the second step from any generated tumor. Most importantly, this tool provides the exact training data necessary for our approach: ground truth clone trees with paired bulk sequencing read counts. Both components are necessary because the sequencing read counts contain the simulated mutations associated with each clone, which form a critical element of the graph encoding. The read counts are not directly used for training in this work, but they are essential for developing a conditional generative, supervised counterpart in future work. SISTEM provides a foundational resource for training supervised models that can potentially be applied to real-world tumor data.

2.3 Synthetic Dataset

We generated a synthetic dataset of 1,967 tumor phylogenies for training. This dataset was generated by deploying a Dockerized wrapper around SISTEM on Google Cloud and a local computer. We maximized compute resources through parallel tumor generation and sampling steps in the SISTEM data generation application. The virtual machine used in Google Cloud consisted of 24 vCPUs, 96GB memory, and 400GB of storage, while the local machine was a 2024 MacBook Pro with an M4 Pro chip and 24GB of memory. Occasionally, SISTEM simulations terminated prematurely when all simulated cells died out before the minimum detectable population size, reflecting the inherent stochastic nature of the simulator. To account for this, we attempted to generate 800 unique tumors using identical parameters (see Table 2) and resample each tumor three times, providing a theoretical dataset of 2,400 trees. Due to the stochastic nature of SISTEM, a total of 728 unique tumors successfully generated, along with 1,967 successful resamples. All data were moved to the local machine (MacBook Pro) to perform further processing. Key outputs included Newick trees and tables representing clonal lineages and clone-mutation associations, respectively.

Once the phylogenies were obtained in Newick format, we converted them into a graph representation compatible with DiGress. The model requires two one-hot encoded tensors: a node tensor $X \in \mathbb{R}^{n \times a}$ and an edge tensor $E \in \mathbb{R}^{n \times n \times b}$, where n is the number of nodes, a is the number of node classes, and b is the number of edge classes. We designed a framework to encode phylogenies densely and efficiently perform one-hot encoding of the tensors in the data loader using PyTorch.

2.4 Graph Encoding

Each graph with n nodes is represented by a node feature matrix X and an edge feature tensor E. The matrix $X \in \mathbb{R}^n$ encodes categorical attributes for each node: normal root (0), tumor clone (1), or mutation event (2). The tensor $E \in \mathbb{R}^{n \times n}$ is a symmetric square matrix whose nonzero entries denote edges: clone-clone connections (1) and clone-mutation assignments (2). A label list L accompanies X, where index i in L corresponds to index i in X and to row/column i in E, storing human-readable node names. This representation meets DiGress's requirement that each node belongs to a single class while preserving mutation and clonal relationships. The dense encoding facilitates efficient one-hot encoding of tensors as required by DiGress. All training graphs were stored as a list of dictionaries, each containing a unique tree identifier and its associated tensors X, E, and L. Only X and E are directly used for training.

2.5 Model Engineering

We implemented a customized data loader into the base version of DiGress to handle the phylogenetic graph encoding. During preprocessing, each graph's E matrix is converted to a sparse edge-list representation, providing a lightweight method for storing edge information. Processed graphs are organized into small shards of 64 graphs and stored on disk. Each shard contains the sparse edge-list representation (replacing E), X, and relevant metadata for each graph. During

training, graphs are loaded on-demand from their shards rather than loading the entire dataset into memory. When a graph is requested, it is extracted from its shard, converted to the one-hot encoded format required by DiGress, and then batched together with other graphs.

2.6 Training

We trained two 7.1 million parameter models using this modified version of DiGress and our synthetic training dataset. The dataset was split to 80% train, 10% validation, 10% test. Both models were trained with cross-entropy loss for 500 epochs, with each sample going through 1000 diffusion steps. Parameters were chosen to mirror models that performed well on similar mathematical graph structures in the original DiGress paper. The noise transition was varied between models (see Table 3). Two options exist for this parameter: marginal and uniform, each corresponding to one of the trained models. We refer to these as DiPhy$_M$, and DiPhy$_U$, respectively. As noise is added to a clean graph G_0, the forward process utilizes a Markov transition matrix Q_t to progressively corrupt the graph. Uniform transitions means that Q_t makes each category (for both nodes and edges) equally likely as noise increases. Marginal transitions force Q_t to preserve the class distribution of nodes and edges from the training data to increase realism. The models were trained using the pre-built PyTorch Lightning multi-GPU training framework from DiGress on four 32GB NVIDIA V100 GPUs.

2.7 Evaluation

Because DiPhy is an unconditional, self-supervised model, we cannot evaluate the model's accuracy using standard supervised learning metrics, as there are no training labels. We developed customized test and validation metrics to assess model performance. These metrics compare the distributions of generated graphs against test and validation graphs. We collected the average and standard deviation of node and edge counts, along with node type composition metrics and validity assessments. Tree validity was assessed by requiring graphs to pass four tests: (1) no cycles, (2) exactly one root node with exactly one clone neighbor, (3) clone edges only connect clones or roots, and (4) mutation edges only connect mutations and clones. All four tests must pass for a tree to be considered valid. A well-performing model will minimize the difference between validation and test metrics. These metrics were calculated every 50 epochs, along with negative log likelihood.

3 Results

Test Metric	Test Dataset	Uniform	Marginal
num_graphs	196.0	1000.0	1000.0
mean_nodes	80.67	**80.26**	78.45
std_nodes	21.58	22.28	**21.16**
mean_edges	79.67	**79.37**	77.48
std_edges	21.58	22.40	**21.19**
mean_clone_fraction	0.0548	0.0332	**0.0375**
mean_mutation_fraction	0.9321	0.9535	**0.9489**
validity_pass_pct	100.0	88.1	**92.8**

Table 1: Test-time metrics comparing test dataset, with generated datasets from DiPhy$_U$ and DiPhy$_M$. num_graphs: number of graphs in dataset, mean/std_nodes: mean/standard deviation of nodes in dataset, mean/std_edges: mean/standard deviation of edges in dataset, mean_clone_fraction: mean fraction of nodes that are clone nodes, mean_mutation_fraction: mean fraction of nodes that are mutation nodes, validity_pass_pct: percent of graphs that pass all four validity criterion.

3.1 Generative Performance

Both models achieved strong performance, particularly in tree validity, demonstrating that this framework successfully generates structurally valid trees. The DiPhy models have been trained to remove noise at a given time step t. At test time, DiPhy begins with random noise (dependent on the transition framework of uniform or marginal) and applies

its trained denoiser from $t = 0$ to $t = 999$, corresponding to 1,000 denoising steps, to generate the final graph that is evaluated. This process is repeated 1,000 times to generate 1,000 sample graphs. It is important to note that this generation process is not guided. We provide the model with noise and evaluate its ability to denoise that input into graphs that are structurally sound. Table 1 outlines the results from the two models at test time.

The marginal transition model, DiPhy$_M$, performs best, generating 92.8% structurally valid phylogenies at inference. DiPhy$_U$ performs very closely to its marginal counterpart in nearly all metrics and still achieves a high percentage of valid samples at 88.1 percent. However, the 4.7% validity increase represents a significant difference, clearly establishing DiPhy$_M$'s marginal transitions as the superior method for learning phylogenetic tree structures. Figure 4 illustrates the differences between DiPhy$_M$ and DiPhy$_U$ at inference.

Examples of valid graphs generated by DiPhy$_M$ are shown in Figure 2a, displaying several graphs that were generated. Since only one tumor type was simulated, the general motif of all generated samples follows a pattern where the first clone node from the root contains numerous mutations and then branches outward in some cases. Figure 2b shows samples generated at test time that failed the validity check. Invalid samples tend toward errors in edge connections, meaning that replacing invalid edges would correct the graph. From manual observation, the model appears to have learned the root-clone rule, but further benchmarking is required to confirm this observation.

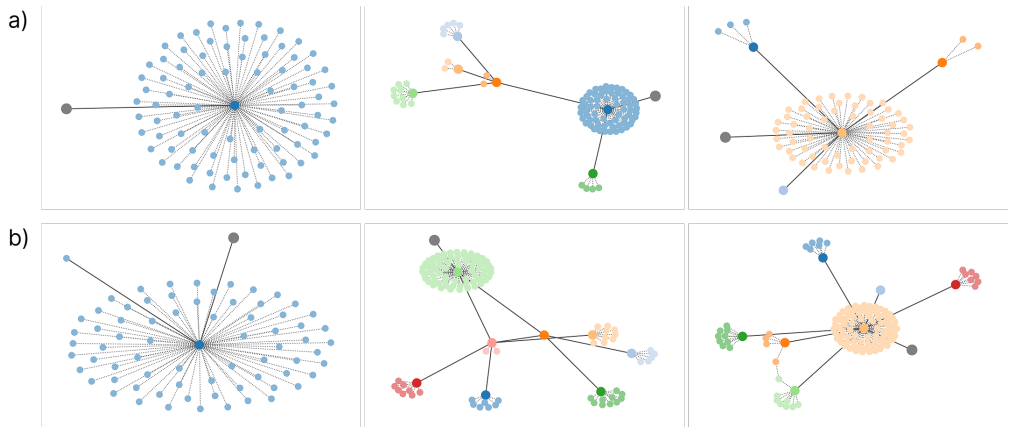

Figure 2: Examples of graphs generated by DiPhy$_M$ at test-time. Valid phylogenies are shown in (a), and invalid phylogenies are shown in (b). Gray nodes are roots, bolder colored nodes are clones, and lighter versions of the bold colors are mutations. Solid edges are parental, or clone edges, while dotted edges represent mutation edges.

3.2 Training

Training dynamics followed standard patterns. DiGress tracks the cross-entropy (CE) loss for the node, edge, and global feature tensors (X, E, y, respectively). Global features represent properties that apply to the entire graph, such as molecular weight for atomic datasets. The global tensor y is completely ignored to prioritize the structural learning of DiPhy, which explains the choice of $\lambda_{\text{train}} = \langle 5, 0 \rangle$ (see Table 3). This vector represents the weighting for the total loss, which is calculated as

$$\text{loss}_{\text{total}} = \text{loss}_X + \lambda_0 \, \text{loss}_E + \lambda_1 \, \text{loss}y$$

$$\text{loss}_{\text{total}} = \text{loss}_X + 5 \, \text{loss}_E.$$

Training and validation loss metrics initially decreased sharply, followed by clear convergence without overfitting (see Figures 5 and 6). Additional validation metrics were tracked over validation epochs, with most following a similar trend of sharp change during the initial epochs followed by largely stable dynamics thereafter. Exceptions can be seen with the average number of nodes and the validity pass rate over epochs. The average number of nodes remains stable around 75 and 82 total nodes for the two models, while the validity pass percentage gradually reaches its final values without overfitting. Validation validity pass percentage reached 92.0% and 95.0% for DiPhy$_U$ and DiPhy$_M$ respectively.

3.3 Inference Speed

A key motivation for this work was to increase inference speed, thereby increasing clinical viability for pipelines like Mase-phi. Deep generative models are significantly faster on inference tasks compared to unsupervised model methods, because parameters are optimized before inference. Though DiPhy cannot be compared to stardard methods like PhyloWGS directly, it is important to gauge the generative speed at an early stage. Inference time was not directly tracked, but though training logs, we can get a sense for the generative speed of our trained models. DiPhy$_M$ spent 4:07, and DiPhy$_U$ spend 4:22 at inference. Each model used a single NVIDIA V100 GPU with the task of generating 1,000 graphs, visualizing 100 of these graphs, and creating 20 GIF animations of the generative process, in batches of 8. Visualization probably plays a large part in the long inference time, but this is still a promising start.

4 Discussion

In this work, we generated a synthetic dataset using a tumor simulator and trained an unconditional adapted graph diffusion model, DiPhy, with this synthetic data. Two separate models, DiPhy$_M$ and DiPhy$_U$, achieved accurate generative capabilities, with DiPhy$_M$ demonstrating particularly strong performance. This work establishes the feasibility of deep generative approaches for tumor phylogeny reconstruction, potentially offering a faster and more clinically viable alternative to traditional methods. While significant challenges remain, including the need for conditional generation and enforcement of biological constraints, these results provide a promising foundation for future development of deep learning-driven phylogenetic inference tools that could accelerate clinical decision-making in precision oncology. The remainder of this section discusses future directions for this work.

4.1 Increasing Dataset Size

We generated a small-scale dataset using a single set of parameters with SISTEM. Future work should prioritize generating a biologically diverse yet relevant dataset of hundreds of thousands of samples to improve generalizability. Using a deep generative approach to this problem means that performance is directly correlated with the quality and diversity of the training data. Further benchmarking would be needed to create a comprehensive simulated dataset, especially when applying guided sampling. The same methodology can be used to generate additional raw data; all that would be required are ample CPU resources and storage. It is important to note that this process is time-intensive, requiring approximately one hour to generate a single tumor and approximately one minute to resample.

4.2 Further Adaptations

Immediate future work should focus on further testing this adapted framework. It is well established that increasing both training dataset size and model size typically improves performance [12]. This principle can be applied to DiPhy by expanding the dataset as mentioned above and incorporating additional graph-transformer layers. Furthermore, testing smaller data subsets with variable model sizes provides valuable insight into the tradeoff between data complexity and model capacity, enabling the development of resource-aware models that maximize accuracy. Additionally, phylogeny-specific graph-level metrics should be tracked to ensure proper model development and validation [19].

4.3 Guided Sampling

While the diffusion prior can extract valid graph structures from pure noise, it lacks the capability to generate graphs conditioned on specific input data. This is where guidance becomes essential, either through a secondary classifier model or through built-in classifier-free guidance. This capability is necessary to progress from annotated bulk sequencing data to a phylogeny, analogous to the approach used by PhyloWGS. The data generation framework developed in this work contains paired read count data with all corresponding trees and can be used to train the built-in guidance model included with DiGress. While some customization would be required, this represents the most promising next step following the current work. Demonstrating the viability of guided sampling is a crucial milestone before allocating time and resources to building a fully customized model from scratch.

4.4 Customized Architecture

DiGress serves as a foundation model for this work, though this approach involves some performance tradeoffs. Our efforts focused on generating a synthetic dataset and encoding it in a way that enables the model to learn phylogenetic structures. This design means that the rules governing phylogeny graphs are enforced purely through statistical learning rather than explicit constraints. Consequently, DiPhy can produce invalid graphs, which still occurs in 8-12% of cases. Now that we have demonstrated that a graph-transformer-based discrete diffusion model architecture can learn these data representations through structure alone, future work should develop a customized encoding that explicitly enforces phylogenetic and biological constraints when learning phylogeny distributions [2]. This customized model should then be conditioned on bulk read count data, ultimately achieving the goal of a cost-aware, high-performing conditional generative model for tumor phylogenies that relies on a single data source.

4.5 Use Cases

There are numerous use cases for both unconditional and conditional generative phylogeny reconstruction models. A pretrained unconditional model could be finetuned, or further trained, with a set of unverified reconstructed patient tumor phylogenies to create realistic distributions of patient data rooted in ground truth simulated data. These learned distributions can then be used to generate new synthetic datasets of completely de-identified patient data, enabling the creation of publicly available datasets for developing additional models and analyzing tumor phylogenies. This capability would significantly expand accessibility to the field, which is currently limited by many private clinical datasets. Another clear use case is to enhance clinical inference workflows, such as Mase-phi, with conditional generative models. This advancement would enable further investigation of evolution-based methods for predicting disease progression.

5 Code and Dataset Availability

All relevant code can be found at `github.com/siddsabata/repo`
Training dataset can be found at `google drive link`

References

[1] Christopher Abbosh, Alexander M. Frankell, Thomas Harrison, and et al. Tracking early lung cancer metastatic dissemination in tracerx using ctdna. *Nature*, 616:553 – 562, 2023.

[2] Guy E. Blelloch, Kedar Dhamdhere, Eran Halperin, R. Ravi, Russell Schwartz, and Srinath Sridhar. Fixed parameter tractability of binary near-perfect phylogenetic tree reconstruction. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 667–678, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[3] Nishat Anjum Bristy and Russell Schwartz. Deconvolution and phylogeny inference of diverse variant types integrating bulk dna-seq with single-cell rna-seq. *Bioinformatics Advances*, 5(1):vbaf234, 09 2025.

[4] David W. Cescon, Scott V. Bratman, Steven M. Chan, and Lillian L. Siu. Circulating tumor dna and liquid biopsy in oncology. *Nature Cancer*, 1:276 – 290, 2020.

[5] Minshuo Chen, Song Mei, Jianqing Fan, and Mengdi Wang. Opportunities and challenges of diffusion models for generative ai. *National Science Review*, 11(12):nwae348, 10 2024.

[6] Amit G. Deshwar, Shankar Vembu, Christina K. Yung, Gun Ho Jang, Lincoln D. Stein, and Quaid D. Morris. Phylowgs: Reconstructing subclonal composition and evolution from whole-genome sequencing of tumors. *Genome Biology*, 16, 2015.

[7] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. In *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021.

[8] Xuecong Fu, Haoyun Lei, Yifeng Tao, and Russell Schwartz. Reconstructing tumor clonal lineage trees incorporating single-nucleotide variants, copy number alterations and structural variations. *Bioinformatics*, 38(Supplement_1):i125–i133, 06 2022.

[9] Xuecong Fu, Zhicheng Luo, Yueqian Deng, William Laframboise, David Bartlett, and Russell Schwartz. Marker selection strategies for circulating tumor dna guided by phylogenetic inference. *bioRxiv*, 2024.

[10] Jonathan Ho, Ajay Jain, and P. Abbeel. Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239, 2020.

[11] Chenghan Jiang, Zhe Wang, Ruoyu Wang, Shanshan Liang, and Shuai Tao. Computational strategies in tumor phylogenetics: evaluating multimodal integration and methodological trade-offs across study designs. *Bioinformatics Advances*, 5(1):vbaf242, 10 2025.

[12] Jared Kaplan, Sam McCandlish, T. J. Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

[13] Jiaying Lai, Yunzhou Liu, Robert B Scharpf, and Rachel Karchin. Evaluation of simulation methods for tumor subclonal reconstruction. *ArXiv*, 2024.

[14] Jiaying Lai, Yi Yang, Yunzhou Liu, Robert B Scharpf, and Rachel Karchin. Assessing the merits: an opinion on the effectiveness of simulation techniques in tumor subclonal reconstruction. *Bioinformatics Advances*, 4(1):vbae094, 06 2024.

[15] Thomas Rachman, Patrick Wagner, William LaFramboise, David Bartlett, Russell Schwartz, and Oana Carja. Abstract 3695: A mechanistic model of ctdna shedding and its relevance for clinical interpretation. *Cancer Research*, 85(8_Supplement_1):3695–3695, 04 2025.

[16] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022.

[17] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2021.

[18] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

[19] Yifeng Tao, Ashok Rajaraman, Xiaoyue Cui, Ziyi Cui, Haoran Chen, Yuanqi Zhao, Jesse Eaton, Hannah Kim, Jian Ma, and Russell Schwartz. Assessing the contribution of tumor mutational phenotypes to cancer progression risk. *PLoS Computational Biology*, 17, 2021.

[20] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

[21] Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *ArXiv*, abs/2209.14734, 2022.

[22] Samson Weiner and Mukul S. Bansal. SISTEM: Simulation of tumor evolution, metastasis, and dna-seq data under genotype-driven selection. Under Review, 2025.

AI Usage: AI tools (GPT and Claude) used for code and writing assistance.

A Additional Tables

Parameter	Value
nsites	3
epsilon	1×10^{-9}
min_detectable	500000
capacities	10000000
focal_driver_rate	0.0005
SNV_pass_rate	0.01
ncells_prim	10000
ncells_meta	5000
ncells_normal	2000
min_mut_fraction	0.05
coverage	100

Table 2: Simulation parameters used for SISTEM. nsites: number of anatomical sites including primary tumor, epsilon: baseline site migration probability, min_detectable: number of cells required to terminate growth simulation at a site, capacities: max number of cells (carrying capacity) of each site, focal_driver_rate: probability of acquiring driver focal CNA per generation, SNV_pass_rate: probability of acquiring passenger SNV per generation, ncells_prim: number of cells to sample from primary tumor, ncells_meta: number of cells to sample from metastatic sites, ncells_normal: number of cells to dilute primary site with, min_mut_fraction: minimum clonal frequency required to keep clone, coverage: average number of reads which cover any base pair in genome.

Parameter	Value
diffusion steps	1000
λ_{train}	$\langle 5, 0 \rangle$
batch size	8
epochs	500
learning rate	0.0002
weight decay	1.00×10^{-12}
optimizer	AdamW
n_layers (graph transformer layers)	8

Table 3: Model hyperparameters and architectural configuration.

B Additional Figures

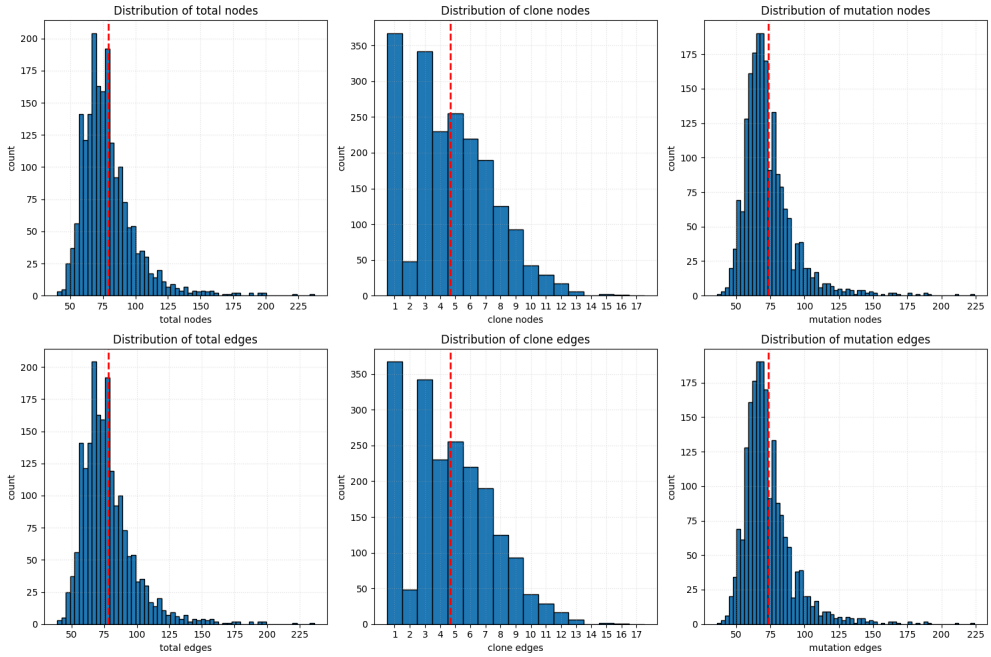

Figure 3: Distribution of entire synthetic dataset before train/val/test split. The vertical red dotted lines indicate the means of each respective distribution.

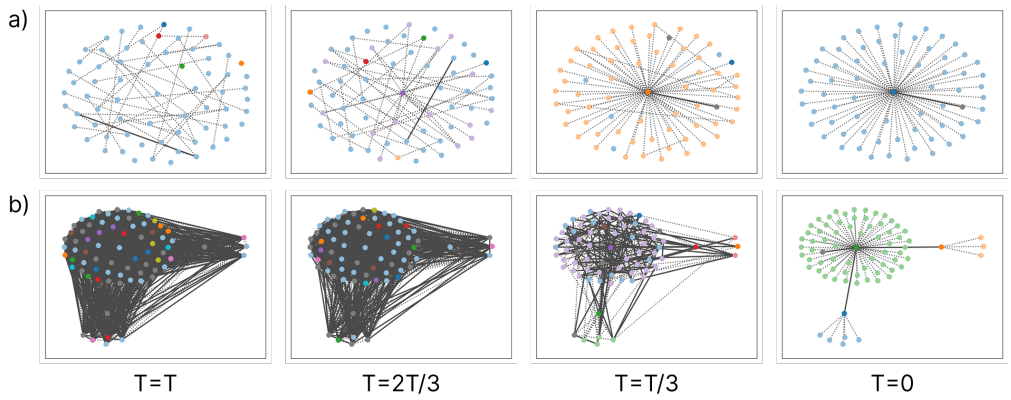

Figure 4: Showcasing generative processes of DiPhy$_M$ (a), and DiPhy$_U$ (b) over time where $T = 1000$.

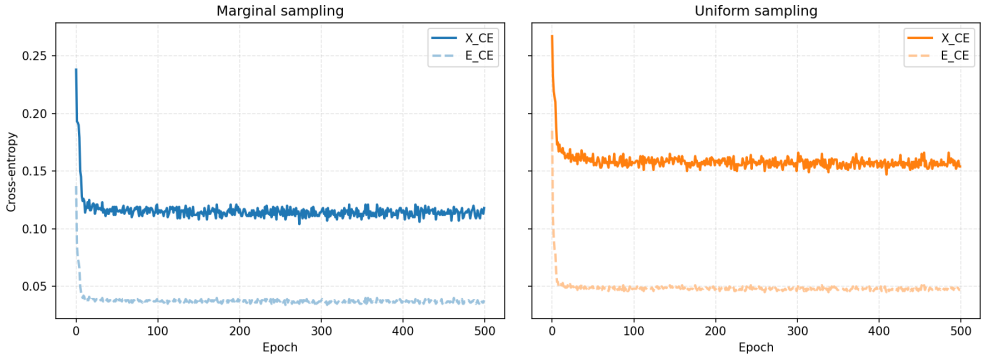

Figure 5: Cross-entropy loss over epochs for DiPhy$_M$ (left), and DiPhy$_U$ (right) for both node (X), and edge (E) tensors.

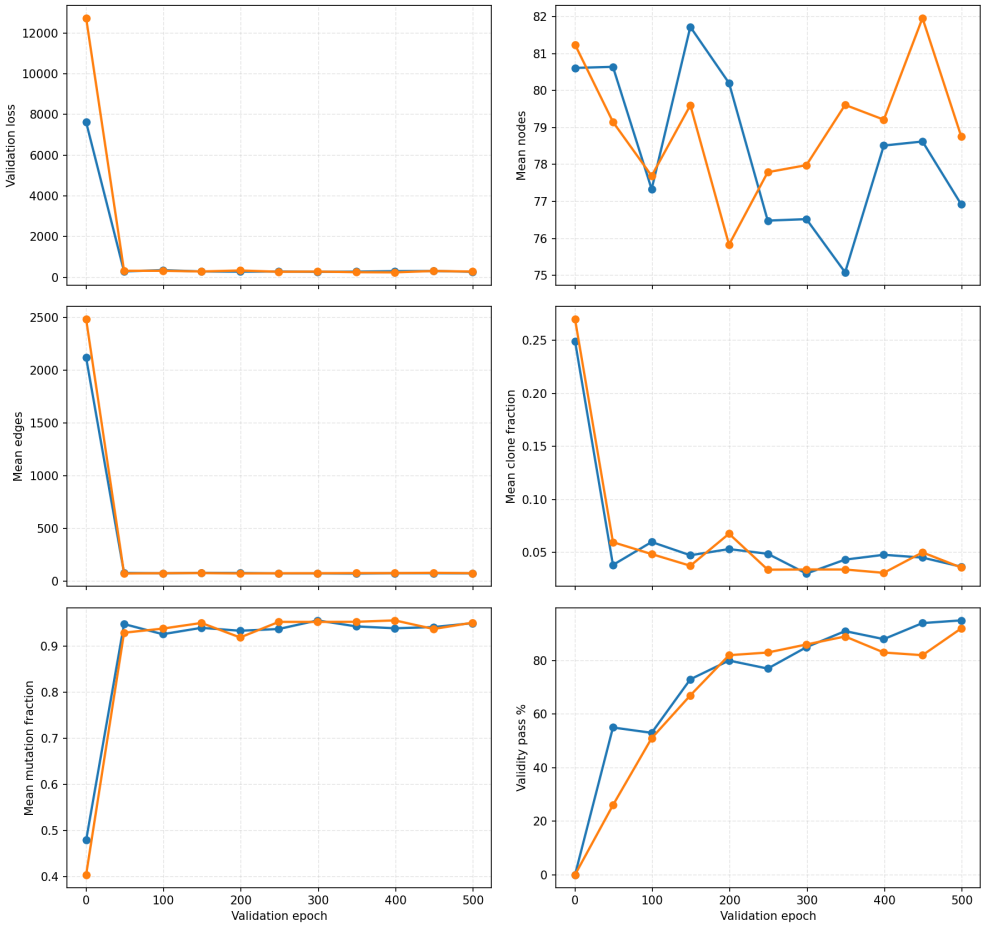

Figure 6: Validation metrics over epochs for DiPhy$_M$ (blue), and DiPhy$_U$ (orange). Validation metrics calculated every 50 epochs. Validation loss: negative log likelihood, Mean edges: average number of edges in generated validation datasets, Mean mutation fraction: mean fraction of nodes that are mutation nodes, Mean nodes: average number of nodes, Mean clone fraction: mean fraction of nodes that are clone nodes, Validity pass %: percent of generated graphs that pass the four criterion test.