Fast Computation of Branching Process Transition Probabilities via ADMM

Achal Awasthi
Duke University

Jason Xu
Duke University

Abstract

Branching processes are a class of continuous-time Markov chains (CTMCs) prevalent for modeling stochastic population dynamics in ecology, biology, epidemiology, and many other fields. The transient or finite-time behavior of these systems is fully characterized by their transition probabilities. However, computing them requires marginalizing over all paths between endpoint-conditioned values, which often poses a computational bottleneck. Leveraging recent results that connect generating function methods to a compressed sensing framework, we recast this task from the lens of sparse optimization. We propose a new solution method using variable splitting; in particular, we derive closed form updates in a highly efficient ADMM algorithm. Notably, no matrix products—let alone inversions—are required at any step. This reduces computational cost by orders of magnitude over existing methods, and the resulting algorithm is easily parallelizable and fairly insensitive to tuning parameters. A comparison to prior work is carried out in two applications to models of blood cell production and transposon evolution, showing that the proposed method is orders of magnitudes more scalable than existing work.

1 INTRODUCTION

Continuous time Markov Chains (CTMCs) have been widely used in stochastic modeling of population dynamics with diverse applications in the fields including genetics, epidemiology, finance, cell biology, and nuclear fission [Renshaw, 2015, Allen, 2010, Paul and Baschnagel, 2013]. Their popularity owes to their flexibility, interpretability, and desirable mathematical properties. From the perspective of statistical inference, one of the most fundamental quantities characterizing a CTMC are its *transition probabilities*, the conditional probabilities that a chain ends at a specific state given a starting state and finite time interval, do not have closed form expressions. The set of all transition probabilities fully define a process, and form the backbone of central quantities such as the likelihood function of discretely observed data from a CTMC [Kalbfleisch and Lawless, 1985, Guttorp, 2018]. Unfortunately, obtaining these transition probabilities is often computationally intensive as it requires marginalization over an infinite set of endpoint-conditioned paths [Hobolth and Stone, 2009, Hajiaghayi et al., 2014].

Classically, this is achieved by computing the matrix exponential of the infinitesimal generator of the CTMC at a computational cost of $O(N^3)$ where N is the size of the state space. Since this procedure is not suited even for state spaces of moderate sizes, practitioners often rely on simplifying assumptions or simulation based approaches [Rao and Teh, 2011, Ross, 1987, Grassmann, 1977], each with their own potential drawbacks. Many core frequentist and Bayesian inferential methods alike require iterative evaluation of the observed data likelihood.

In certain structured classes within CTMCs, alternative approaches are possible. Transition probabilities can be calculated explicitly for simple examples such as the Poisson process, and the class of linear birth-death processes provides another special case that admits closed form solutions [Crawford et al., 2014, Karlin and Taylor, 1975, Lange, 2010, Tavaré, 2018].

In this article, we focus on the case of multi-type branching processes, for which efficient numerical techniques have only recently been developed. Xu et al. [2015] make use of a method that expresses the probability generating function (PGF) as a Fourier series expansion. The generating functions are obtained as solutions to differential equation systems, and the transition probabilities can be recovered from fast numerical series inversion techniques, enabling maximum likelihood inference and Expectation-Maximization (EM) algorithms [Doss et al., 2013]). These methods however can also become costly for large systems: for instance, approximating Fourier inversion formula using a Riemann sum requires $O(N^b)$ PGF evaluations, where b is the number of types in the branching process and N is the largest

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

population size at the end of the desired transition probabilities, which entails millions of PGF evaluations even for a two-type process with populations in the thousands. As many likelihood-based procedures are iterative, scalable alternatives to this computation are critical to avoid a bottleneck that renders these methods out of reach. When sparsity is available in the probabilities, Xu and Minin [2015] suggest using compressed sensing to reduce the number of computations at the expense of solving a sparse optimization problem using proximal gradient descent (PGD).

To make progress, this article revisits this compressed sensing framework from the lens of variable splitting methods. We show how the sparse optimization problem can be solved orders of magnitude more efficiently via a careful implementation of the Alternating Direction Method of Multipliers (ADMM) algorithm [Everett III, 1963, Eckstein and Fukushima, 1994, Boyd et al., 2011]. Our approach operates on a vectorized version of the problem, avoiding large matrix inversions that prevent the previous PGD approach from applying to large-scale settings. Surprisingly, not only does variable splitting avoid inversions, but can be accomplished without calls even to matrix multiplication by clever use of the Fast Fourier Transform. Our method inherits desirable convergence and recovery guarantees, and admits easily parallelizable implementations. Moreover, it is surprisingly inelastic to tuning parameters— we find that successful performance is robust to a broad range of reasonable penalty parameters. We validate these merits in thorough empirical examples, showcasing significant advantages to prior algorithms applied to a two-type branching model of hematopoiesis, the process of blood cell formation, and a model of transposon evolution from genetic epidemiology.

2 MARKOV BRANCHING PROCESSES

A Markov branching process is a continuous-time Markov chain comprised of a collection of particles that proliferate independently, and whose reproduction or death follows a probability distribution. We consider continuous-time, multi-type branching processes that take values over a discrete state space of non-negative integers where each particle type has its own mean lifetime and reproductive pattern [Lange, 2010, Karlin and Taylor, 1975]. For exposition, we focus on the two-type case, though results apply generally.

Let $\mathbf{X}(t)$ denote a linear, two-type branching process that takes values in a discrete state space Ω, with $X_i(t)$ representing the number of particles of type i present at time $t \geq 0$, where each particle of type i at the end of its lifespan can produce k particles of type 1 and l particles of type 2 with instantaneous rates $a_i(k,l)$. The overall rates are multiplicative in the number of particles due to rate linearity, which follows from the independence assumption.

While they are defined by the instantaneous rates, the transition probabilities provide an alternate characterization of a branching process, also completely defining its dynamics:

$$p_{(j,k),(l,m)}(t) = \mathbb{P}(\mathbf{X}(t+s) = (l,m) | \mathbf{X}(s) = (j,k)). \quad (1)$$

These transition probabilities play a central role in statistical inference either in the form of maximum likelihood estimation or within Bayesian methods where likelihood calculations often enter in schemes such as Metropolis-Hastings to sample from the posterior density. Motivated by their broad importance, we now focus our attention on computing them in the class of continuous-time branching process.

2.1 Generating Function

The probability generating function (PGF) for a two type process is

$$\begin{aligned}\phi_{jk}(t, s_1, s_2; \boldsymbol{\theta}) &:= \mathbb{E}_{\boldsymbol{\theta}}(s_1^{X_1(t)} s_2^{X_2(t)} | X_1(0) = j, X_2(0) = k) \\ &= \sum_{l=0}^{\infty} \sum_{m=0}^{\infty} p_{(j,k),(l,m)}(t; \boldsymbol{\theta}) s_1^l s_2^m, \end{aligned} \quad (2)$$

with a natural extension existing for any m-type process. Given the instantaneous rates $a_j(k,l)$, the system of differential equations governing ϕ_{jk} can be derived using Kolmogorov forward or backward equations as shown by Bailey [1991]. Once we have ϕ_{jk}, the transition probabilities can be formally obtained by differentiating ϕ_{jk} in Equation 2 and normalizing by an appropriate constant,

$$p_{(j,k),(l,m)}(t) = \frac{1}{l!m!} \frac{\partial^l}{\partial s_1} \frac{\partial^m}{\partial s_2} \phi_{jk}(t) \Big|_{s_1 = s_2 = 0}. \quad (3)$$

However, in practice repeated numerical differentiation is a computationally intensive procedure and often becomes numerically unstable, especially for large l, m. To circumvent this issue, we use a technique by Lange [1982] that allows us to use Fast Fourier transform (FFT) to calculate the transition probabilities $p_{(j,k),(l,m)}(t)$ which occur as coefficients of $s_1^l s_2^m$ in Equation 2. We can map the domain $s_1, s_2 \in [0,1] \times [0,1]$ to the boundary of the complex unit circle by setting $s_1 = e^{2\pi i \omega_1}, s_2 = e^{2\pi i \omega_2}$ and view the PGF as a Fourier series

$$\phi_{jk}(t, e^{2\pi i \omega_1}, e^{2\pi i \omega_2}) = \sum_{l,m=0}^{\infty} p_{(j,k),(l,m)}(t) e^{2\pi i l \omega_1} e^{2\pi i m \omega_2}.$$

We can now compute the transition probabilities via the Fourier inversion formula, approximating the integral by a Riemann sum:

$$\begin{aligned} p_{(j,k),(l,m)}(t) &= \int_0^1 \int_0^1 \phi_{jk}(t, e^{2\pi i \omega_1}, e^{2\pi i \omega_2}) \\ &\quad \times e^{-2\pi i l \omega_1} e^{-2\pi i m \omega_2} d\omega_1 d\omega_2 \\ &\approx \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \phi_{jk}(t, e^{2\pi i u/N}, e^{2\pi i v/N}) \\ &\quad \times e^{-2\pi i l u/N} e^{-2\pi i m v/N}. \end{aligned} \quad (4)$$

3 COMPRESSED SENSING FRAMEWORK

Compressed sensing (CS) is based on the principle that a sparse or compressible signal can be reconstructed, often perfectly, from a small number of its projections onto a certain subspace through solving a convex optimization problem. For our application, transition probabilities play the role of a target sparse signal of Fourier coefficients. CS enables us to restrict the necessary computations to a much smaller sample of PGF evaluations, which play the role of measurements used to recover the sparse signal.

In this setup, let $\mathbf{u} \in \mathbb{C}^N$ be an unknown sparse signal and let $\mathbf{\Psi} = [\psi_1, \psi_2, \ldots, \psi_N] \in \mathbb{C}^{N \times N}$ denote an orthonormal basis of $\mathbb{C}^N$. Then there exists a unique $\mathbf{s} \in \mathbb{C}^N$ such that

$$\mathbf{u} = \sum_{i=1}^{N} \psi_i s_i = \mathbf{\Psi s}. \tag{5}$$

If the number of non-zeros in $\mathbf{s}$, or $||s||_0$, is less than K, then $\mathbf{u}$ is said to be K-*sparse* under $\mathbf{\Psi}$. We are interested in cases where $K << N$ or $\mathbf{u}$ is highly compressible.

Let an integer M satisfy $K < M << N$, and $\mathbf{u}$ be observed through a measurement

$$\mathbf{b} = \mathbf{\Phi u} = \mathbf{As} \in \mathbb{C}^M, \tag{6}$$

where $\mathbf{\Phi} \in \mathbb{C}^{M \times M}$ denotes a nonadaptive sensing matrix and $\mathbf{A} = \mathbf{\Phi\Psi}$. Here nonadaptiveness means that $\mathbf{\Phi}$ does not depend on $\mathbf{u}$. As $M << N$, this system is underdetermined and the space of solutions is an infinite affine space. However, in certain sparse settings, Candès [2006] proves that reconstruction can be accurately achieved by finding the most sparse solution among all solutions of Equation 6, i.e.,

$$\hat{\mathbf{s}} = \{\arg\min_{\mathbf{s}} ||s||_0 : \mathbf{As} = \mathbf{b}\}. \tag{7}$$

Due to the combinatorial intractability of the non-convex problem, it is impractical to solve this ℓ_0 formulation directly. Instead we use the ℓ_1-relaxation as a proxy which results in an easier convex optimization problem, where we optimize the following unconstrained penalized objective

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \frac{1}{2}||\mathbf{As} - \mathbf{b}||_2^2 + \lambda||\mathbf{s}||_1, \tag{8}$$

with λ serving as the regularization parameter to enforce the sparsity of $\mathbf{s}$.

Past works indicate that when $\mathbf{A}$ satisfies the *Restricted Isometry Property* (RIP) [Candès and Tao, 2005], Candès et al. [2006], the K-sparse signal $\mathbf{u}$ or equivalently $\mathbf{s}$ can be reconstructed using only $M = CK \log N$ measurements for some constant C. The result assumes that the columns of ϕ and ψ are not just incoherent pairwise, but K-wise incoherent. The *coherence* between ϕ and ψ is defined as $\mu(\boldsymbol{\phi}, \boldsymbol{\psi}) = \sqrt{n} \max_{1 \le i,j \le n} |\langle \boldsymbol{\phi}_i, \boldsymbol{\psi}_j \rangle|$.

In practice, verifying RIP can be a demanding task; Candès et al. [2006] and Donoho [2006] show that RIP holds with high probability in some Gaussian random matrix settings. In the focus of this paper, $\mathbf{A}$ is comprised of a random "spike" basis playing the role of $\mathbf{\Phi}$ together with a Fourier domain representation $\mathbf{\Psi}$ [Rudelson and Vershynin, 2008, Candès et al., 2006], which form a maximally incoherent pair [Candès and Romberg, 2007]. We see how these components enter the derivation below, and later confirm the success this theory suggests via a thorough empirical study.

3.1 Higher Dimensions

We can easily extend this exposition focused on the vector valued case to higher-dimensional signals [Candès, 2006]. To illustrate this, consider the $2D$ case where the sparse solution $\mathbf{S} \in \mathbb{C}^{N \times N}$ and the measurement

$$\mathbf{B} = \mathbf{ASA}^T \in \mathbb{C}^{M \times M} \tag{9}$$

are matrices instead of vectors. We may solve an equivalent problem

$$\hat{\mathbf{S}} = \arg\min_{\mathbf{S}} \frac{1}{2}||\mathbf{ASA}^T - \mathbf{B}||_2^2 + \lambda||\mathbf{S}||_1. \tag{10}$$

Equivalently, this can also be represented in a vectorized framework with

$$\text{vec}(\mathbf{S}) = \tilde{\mathbf{s}} \in \mathbb{C}^{N^2}, \quad \text{vec}(\mathbf{B}) = \tilde{\mathbf{b}} \in \mathbb{C}^{M^2},$$

and we now pursue $\tilde{\mathbf{b}} = \tilde{\mathbf{A}}\tilde{\mathbf{s}}$, with $\tilde{\mathbf{A}} = \mathbf{A} \bigotimes \mathbf{A}$ being the Kronecker product of $\mathbf{A}$ with itself. It can be preferable to solve Equation 10 rather than the vectorized problem explicitly, as the number of entries in $\tilde{\mathbf{A}}$ grows rapidly. However, we will show how working with vectorized forms together with clever implementations of the Fast Fourier Transform will provide the best of both worlds, avoiding matrix multiplication and inversion entirely.

4 TRANSITION PROBABILITIES VIA ADMM

Recall that we wish to compute the transition probabilities $p_{(jk),(lm)}(t)$ for any $t > 0$ and initial $\mathbf{X}(0) = (j, k)$. Within the CS framework described in the previous section, $\mathbf{S} \in \mathbb{C}^{N \times N}$ is the matrix of transition probabilities with entries

$$\{\mathbf{S}\}_{l,m} = p_{jk,lm}(t).$$

Without the CS framework, one can directly obtain the transition probabilities using Equation 4 by first computing an equally sized matrix of PGF solutions

$$\tilde{\mathbf{B}} = \{\phi_{jk}(t, e^{2\pi iu/N}, e^{2\pi iv/N})\}_{u,v=0}^{N-1} \in \mathbb{C}^{N \times N}. \tag{11}$$

Obtaining $\tilde{\mathbf{B}}$ becomes computationally expensive for large N values, especially when each PGF must be solved, for

instance via numerically evaluating a differential equation. Given a way to compute $\tilde{\mathbf{B}}$, one recovers the transition probabilities by taking the Fast Fourier Transform (FFT). We can better understand how we will avoid this computation within the CS framework with the help of matrix notation. We have $\mathbf{S} = \mathbf{F}\tilde{\mathbf{B}}\mathbf{F}^{\mathbf{T}}$, where $\mathbf{F} \in \mathbb{C}^{N\times N}$ denotes the Discrete Fourier Transform (DFT) matrix. Thus, the Inverse Discrete Fourier Transform (IDFT) matrix $\mathbf{F}^{-\mathbf{1}}$ becomes the sparsifying basis $\mathbf{\Psi}$ mentioned in Equation 5, and we have $\tilde{\mathbf{B}} = \mathbf{\Psi}\mathbf{S}\mathbf{\Psi}^{\mathbf{T}}$.

If we expect $\mathbf{S}$ to have a sparse representation in this basis, we can employ a method to reconstruct $\mathbf{S}$ using a much smaller set of PGF evaluations arranged in the matrix $\mathbf{B} \in \mathbb{C}^{M\times M}$, corresponding to a subset of entries from $\tilde{\mathbf{B}}$ selected uniformly at random. This much smaller matrix is a projection $\mathbf{B} = \mathbf{A}\mathbf{S}\mathbf{A}^T \in \mathbb{C}^{M\times M}$ in accordance with Equation 9, where $\mathbf{A} \in \mathbb{C}^{M\times N}$ is obtained by selecting a subset of rows of $\mathbf{\Psi}$ that correspond to $\mathcal{J}$, the randomly sampled indices. This uniform sampling of rows is equivalent to multiplication by measurement matrix encoding the *spike* (or standard) basis. Mathematically, we have $\mathbf{A} = \mathbf{\Phi}\mathbf{\Psi}$, with the rows of the measurement matrix $\mathbf{\Phi}_j(l) = \delta(j-l)$. Thus, uniformly sampling the indices $\mathcal{J}$ is optimal in our setting because the spike and Fourier bases are *maximally incoherent* in any dimension [Candès and Wakin, 2008].

In the compressed sensing framework, we now require only computing this reduced matrix $\mathbf{B}$, which entails only a logarithmic proportion $|\mathbf{B}| \propto K\log|\tilde{\mathbf{B}}|$ of PGF evaluations compared to the original problem. Thus, the problem of computing the transition probabilities $\mathbf{S}$ has been reduced to a signal recovery problem that can be solved by solving the optimization problem in Equation 10.

4.1 Solving the Convex Problem Using ADMM

We use the Alternating Direction Method of Multipliers (ADMM) algorithm [Everett III, 1963, Boyd et al., 2011] to solve this problem efficiently. ADMM is useful for optimization problems where the objective function is of the form $\{f(\mathbf{x})+g(\mathbf{y}) : \mathbf{x} = \mathbf{y}\}$ with both $f(.)$ and $g(.)$ convex but not necessarily differentiable.

We first introduce some notations before detailing the optimization routine. Recall that for our application, $\mathbf{U}$ is a two-dimensional unknown sparse signal being observed through a measurement matrix $\mathbf{B}$, and $\mathbf{F_P} = \mathbf{P}\mathbf{F} \in \mathbb{C}^{M\times N}$ denotes the sensing matrix, where $\mathbf{P} \in \mathbb{R}^{M\times N}$ is a selection matrix containing M rows of the identity matrix of order N. Similarly, we will denote $\mathbf{F_P^{-1}} = \mathbf{P}\mathbf{F^{-1}} \in \mathbb{C}^{M\times N}$ where $\mathbf{F^{-1}}$ is the IDFT matrix.

To solve this problem efficiently using ADMM, we first split the optimization variable by introducing a new variable $\mathbf{Z}$ under an equality constraint with the variable of interest:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{Z}} \quad & \frac{1}{2}||\mathbf{F_P^{-1}}\mathbf{U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1 \\ \text{subject to} \quad & \mathbf{U} = \mathbf{Z}. \end{aligned} \tag{12}$$

Next, we consider the Augmented Lagrangian

$$\begin{aligned} L_\beta(\mathbf{U},\mathbf{Z},\mathbf{Y}) = \frac{1}{2}||\mathbf{F_P^{-1}}\mathbf{U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1 \\ + \mathbf{Y}^T(\mathbf{U}-\mathbf{Z}) + \frac{\beta}{2}||\mathbf{U}-\mathbf{Z}||_2^2, \end{aligned} \tag{13}$$

where $\mathbf{Y}$ serves as the *dual variable* enforcing $\mathbf{U} = \mathbf{Z}$. Now, the ADMM algorithm entails the following iteration:

$$\begin{aligned} \mathbf{U^{(k+1)}} &:= \arg\min_{\mathbf{U}} L_\beta(\mathbf{U}, \mathbf{Z^{(k)}}, \mathbf{Y^{(k)}}) \\ \mathbf{Z^{(k+1)}} &:= \arg\min_{\mathbf{Z}} L_\beta(\mathbf{U^{(k+1)}}, \mathbf{Z}, \mathbf{Y^{(k)}}) \\ \mathbf{Y^{(k+1)}} &:= \mathbf{Y^{(k)}} + \beta(\mathbf{U^{(k+1)}} - \mathbf{Z^{(k+1)}}) \end{aligned} \tag{14}$$

Deriving the explicit update for the subproblem in $\mathbf{Z}$ here is straightforward, given by the soft-thresholding operator

$$\mathbf{Z^{(k+1)}} = \text{SoftThresh}\Big(\mathbf{U^{(k+1)}} + \frac{\mathbf{Y^{(k)}}}{\beta}, \frac{\lambda}{\beta}\Big),$$

where $\text{SoftThresh}(\epsilon, \lambda/\beta) = \max\{|\epsilon| - \lambda/\beta, 0\} \cdot sign(\epsilon)$ for a constant $\epsilon \in \mathbb{R}$. To minimize $\mathbf{U}$, we set $\nabla_U\Big[L_\beta(\mathbf{U},\mathbf{Z},\mathbf{Y})\Big]$ equal to 0,

$$\begin{aligned} &\nabla_U\Big[\frac{1}{2}||\mathbf{F_P^{-1}}\mathbf{U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2\Big] + \nabla_U(\lambda||\mathbf{Z}||_1) \\ &\quad + \nabla_U(\mathbf{Y}^T(\mathbf{U}-\mathbf{Z})) + \nabla_U\Big[\frac{\beta}{2}||\mathbf{U}-\mathbf{Z}||_2^2\Big] = 0, \\ &(\mathbf{F_P^{-1}})^H(\mathbf{F_P^{-1}})\mathbf{U}(\mathbf{F_P^{-1}})^H(\mathbf{F_P^{-1}}) - (\mathbf{F_P^{-1}})^H\mathbf{B}(\mathbf{F_P^{-1}}) \\ &\quad + \mathbf{Y} + \beta(\mathbf{U}-\mathbf{Z}) = 0, \\ &(\mathbf{F_P^{-1}})^H(\mathbf{F_P^{-1}})\mathbf{U}(\mathbf{F_P^{-1}})^H(\mathbf{F_P^{-1}}) + \beta\mathbf{U} \\ &\quad = (\mathbf{F_P^{-1}})^H\mathbf{B}(\mathbf{F_P^{-1}}) + \beta\mathbf{Z} - \mathbf{Y}. \end{aligned} \tag{15}$$

Notice that Equation 15 cannot be used to obtain a closed form update for $\mathbf{U}$. However, we can make progress by vectorizing, allowing us to rewrite Equation 15 as

$$\begin{aligned} &(\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} + \beta\tilde{\mathbf{u}}^{(\mathbf{k+1})} \\ &\quad = (\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{b}} + \beta\Big(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\Big), \end{aligned}$$

where $\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} = (\mathbf{F_P^{-1}}) \bigotimes (\mathbf{F_P^{-1}}) \in \mathbb{C}^{M^2\times N^2}$ and $\tilde{\mathbf{I}} = \mathbf{I} \bigotimes \mathbf{I} \in \mathbb{R}^{N^2\times N^2}$. This yields the update for $\tilde{\mathbf{u}}$

$$\begin{aligned} \tilde{\mathbf{u}}^{(\mathbf{k+1})} = &\Big[(\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}}\Big]^{-1} \\ &\Big[(\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{b}} + \beta\Big(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\Big)\Big]. \end{aligned} \tag{16}$$

Avoiding matrix operations: Updating $\tilde{\mathbf{u}}$ naïvely according to Equation 16 requires dense matrix multiplication and inversion, steps that would significantly add to the runtime of our algorithm in large-scale settings.

To circumvent this, we may instead think of Equation 16 as the solution to the linear system

$$\mathbf{M}\tilde{\mathbf{u}}^{(\mathbf{k+1})} = \mathbf{a}, \quad \text{where}$$

$$\mathbf{M} = \left((\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H \tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}}\right) \in \mathbb{C}^{N^2 \times N^2} \quad \text{and}$$

$$\mathbf{a} = \left((\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H \tilde{\mathbf{b}} + \beta\left(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\right)\right) \in \mathbb{C}^{N^2}.$$

Multiplying $\tilde{\mathbf{F}}^{-\mathbf{1}} = \mathbf{F}^{-\mathbf{1}} \bigotimes \mathbf{F}^{-\mathbf{1}} \in \mathbb{C}^{N^2 \times N^2}$ to both sides of the above equation, we obtain

$$\hat{\mathbf{M}}\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} = \hat{\mathbf{a}}, \quad \text{where}$$

$$\hat{\mathbf{M}} = \beta\tilde{\mathbf{I}} + \tilde{\mathbf{P}}^{\mathbf{T}}\tilde{\mathbf{P}} \quad \text{and} \quad \hat{\mathbf{a}} = \tilde{\mathbf{F}}^{-\mathbf{1}}\beta\tilde{\mathbf{z}}^{(\mathbf{k})} + \tilde{\mathbf{P}}^{\mathbf{T}}\tilde{\mathbf{b}}. \quad (17)$$

The complete details of these derivations appear step-by-step in the Supplement. Note that $\hat{\mathbf{M}}$ is a diagonal matrix of order N^2 because both $\beta\tilde{\mathbf{I}}$ and $\tilde{\mathbf{P}}^{\mathbf{T}}\tilde{\mathbf{P}}$ are diagonal matrices of order N^2. Of course in practice we only compute and store the vector $\hat{\mathbf{m}}$ containing the N^2 diagonal elements of $\hat{\mathbf{M}}$. For exposition, let $N = 4, M = 2, \beta = 0.1$ and the uniformly sampled indices $\mathcal{J} = [1, 2]$, then we can compute $\hat{\mathbf{m}}$ as,

$$\hat{\mathbf{m}}_{\mathbf{1}} = \beta \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1 \\ \vdots \\ 0.1 \end{bmatrix}_{16\times 1} \qquad \tilde{\mathbf{p}} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}_{4\times 1}$$

$$\tilde{\mathbf{p}} \bigotimes \tilde{\mathbf{p}} = \left[1, 1, 0, 0, 1, 1, (\mathbf{0})_{10}\right]^T_{1\times 16},$$

$$\hat{\mathbf{m}} = \hat{\mathbf{m}}_{\mathbf{1}} + \tilde{\mathbf{p}} \bigotimes \tilde{\mathbf{p}}$$

$$= \left[1.1, 1.1, 0.1, 0.1, 1.1, 1.1, (\mathbf{0.1})_{10}\right]^T_{1\times 16}.$$

On the other hand, since $\hat{\mathbf{a}} = \text{vec}(\hat{\mathbf{A}}) = \text{vec}(\mathbf{F}^{-\mathbf{1}}\beta\mathbf{Z}^{(\mathbf{k})} + \mathbf{P}^{\mathbf{T}}\mathbf{B}\mathbf{P})$, computing $\hat{\mathbf{a}}$ is equivalent to first computing $\hat{\mathbf{A}}$ and then vectorizing it. Thus, we can compute $\hat{\mathbf{m}}$, compute $\hat{\mathbf{a}}$ and then efficiently solve the linear system

$$\mathbf{F}^{-1}\mathbf{U}^{(\mathbf{k+1})} = \text{vec}^{-1}(\hat{\mathbf{a}} \oslash \hat{\mathbf{m}}), \quad (18)$$

where $\oslash$ is the *Hadamard (elementwise) division* operator and vec^{-1} is the inverse of the vectorization operator, i.e. reshaping $\hat{\mathbf{a}} \oslash \hat{\mathbf{m}}$ back into a $N \times N$ matrix. Thereafter, to isolate the optimization variable, we may recover $\mathbf{U}^{(\mathbf{k+1})}$ by "cancelling out" $\mathbf{F}^{-\mathbf{1}}$ through applying the FFT on both sides. Doing so avoids explicitly multiplying matrices implied by 16; details are mentioned in line 8 of Algorithm 1.

Computational complexity: We now we analyze the per-iteration computational complexity of the subproblems in minimizing $\mathbf{Z}$ and $\mathbf{U}$. First, note the minimization of $\mathbf{Z}$ requires evaluating the soft-thresholding operator elementwise, and thus has a complexity of $O(kN^2)$, where k is the number of ADMM iterations and N is the maximum population size.

Next, we consider the complexity of naively minimizing $\mathbf{U}$ if one were to use Equation 16 directly. Doing so would involve dense matrix multiplications to calculate $\mathbf{M}$ and $\mathbf{a}$, as well as matrix inversion to obtain $\mathbf{U}$. Calculating $\mathbf{M}$ has a complexity of $O(N^4M^2)$ as it involves matrix multiplication of $\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} \in \mathbb{C}^{M^2\times N^2}$ with its complex conjugate. Calculating $\mathbf{a}$ has a complexity of $O(\max\{(N^2M^2, kN^2\})$ while inverting $\mathbf{M}$ has $O(N^6)$ cost, thus incurring a total computational complexity of $O(\max\{N^6, \max\{N^2M^2, kN^2\}\}) = O(N^6)$ for the $\mathbf{U}$ update.

Our discussion on reducing this complexity by avoiding matrix operations reduces this significantly, making use of $\hat{\mathbf{m}}$ and $\hat{\mathbf{A}}$ instead of $\mathbf{M}$ and $\mathbf{a}$ as described above. Forming $\hat{\mathbf{m}}$ has $O(N^2)$ cost as it involves the calculation of N^2 non-zero diagonal elements of $\hat{\mathbf{M}}$. Then, computing $\hat{\mathbf{A}}$ requires the calculation of $\mathbf{P}^{\mathbf{T}}\mathbf{B}\mathbf{P}$, $\beta\mathbf{Z}^{(\mathbf{k})}$ and its IDFT, which have complexities $O(M^2)$, $O(kM)$, and $O(kN^2\log_2 N)$ respectively. Therefore, the calculation of $\hat{\mathbf{A}}$ has a complexity of $O(\max\{M^2, kM, kN^2\log_2 N\}) = O(kN^2\log_2 N)$ which matches the cost of computing two-dimensional FFT required to obtain $\mathbf{U}^{(\mathbf{k+1})}$. Since this is the dominant complexity in solving the linear system described by Equation 18, the overall complexity of the $\mathbf{U}$ update (and in turn the $\mathbf{Z}$ update) becomes $O(kN^2\log_2 N)$, a dramatic reduction from $O(N^6)$.

It is worth mentioning that the total complexity of the algorithm can be further improved by leveraging the sparsity of the transition probability matrix and using sparse FFT (sFFT) techniques [Hassanieh et al., 2012], which can reduce the complexity of the two-dimensional IDFT from $O(kN^2\log_2 N)$ to $O(kM^2\log_2 M)$. However, for our application, we use the Fastest Fourier Transform in the West (`FFTW`) [Frigo and Johnson, 1998] to implement FFT as the differences in execution times between implementing FFT using `FFTW` and sFFT are only appreciable for $N > 2^{17}$ [Hassanieh et al., 2012], which is beyond the maximum population sizes considered in this paper.

Execution time of ADMM: We further improve the wall-clock execution time of our algorithm by vectorizing intermediate steps (lines 2, 3, and 8 of Algorithm 1) whenever possible, and making use of optimized Numpy libraries Harris et al., 2020 and parallelized FFT in `FFTW` [Frigo and Johnson, 1998].

4.2 ADMM Convergence and Stopping Criteria

Having detailed the algorithm and having emphasized careful diagonalization tricks in the Fourier domain to significantly reduce runtime, we now discuss aspects related to convergence. We establish the convergence guarantees and stopping criteria of our ADMM algorithm. It is not difficult to show that our formulation inherits powerful standard convergence results. We begin by verifying assumptions behind one of the classical convergence theorems.

Proposition 4.1. *The functions $f(\mathbf{U})$ and $g(\mathbf{Z})$ are closed, convex and proper.*

The functions $f(\mathbf{U}) = \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2$ and $g(\mathbf{Z}) = \lambda||\mathbf{Z}||_1$ are closed, convex and proper by virtue of being the square of the L_2 norm and L_1 norm, respectively [Rudin et al., 1976, Folland, 1999].

Proposition 4.2. *For $\beta = 0$, $L_0 = \lambda||\mathbf{Z}||_1 + \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^{\mathbf{H}} - \mathbf{B}||_2^2 + \mathbf{Y}^T(\mathbf{U} - \mathbf{Z})$, the unaugmented Lagrangian has a saddle point for all $\mathbf{U}, \mathbf{Z}, \mathbf{Y}$.*

Proposition 4.3. *Under propositions 4.1 and 4.2 the ADMM algorithm achieves:*

1. *Primal residual convergence: $\mathbf{R^{(k+1)}} = \mathbf{U^{(k+1)}} - \mathbf{Z^{(k+1)}} \to \mathbf{0}$ as $k \to \infty$.*
2. *Dual residual convergence: $\mathbf{S^{(k+1)}} = \beta(\mathbf{Z^{(k+1)}} - \mathbf{Z^{(k)}}) \to \mathbf{0}$ as $k \to \infty$.*

The proofs of Propositions 4.2 and 4.3 are given in the Supplement.

4.2.1 Optimality Conditions and Stopping Criteria

The necessary and sufficient optimality conditions for the optimization problem described in Equation 12 are defined by primal feasibility,

$$\mathbf{U}^* + \mathbf{Z}^* = \mathbf{0}, \tag{19}$$

and dual feasibility,

$$0 = \nabla\left[\frac{1}{2}||\mathbf{F_P^{-1}U}^*(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2\right] + \mathbf{Y}^* \tag{20}$$

$$0 \in \partial(\lambda||\mathbf{Z}^*||_1) - \mathbf{Y}^*, \tag{21}$$

where $(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*)$ is the saddle point of L_0, ∂ denotes the subdifferential operator [Rockafellar, 1970, Borwein and Lewis, 2006] due to $\lambda||\mathbf{Z}^*||_1$ being non-differentiable. Thus, Equations 19-21 constitute the three optimality conditions for the optimization problem described in Equation 12. The last condition (21) always holds for $(\mathbf{U^{(k+1)}}, \mathbf{Z^{(k+1)}}, \mathbf{Y^{(k+1)}})$ whereas the other conditions give rise to the primal $\mathbf{R^{(k+1)}}$ and dual $\mathbf{S^{(k+1)}}$ residuals with proof in the Supplement; these residuals converge to zero as ADMM iterates according to Proposition 4.3.

Algorithm 1: Fast sparse reconstruction using ADMM

function ADMM $(\mathbf{B}, \beta, \lambda, \epsilon_{abs}, \epsilon_{rel}, N_{iter}, \mathbf{P}, \mathcal{J})$;
Input : Measurements $\mathbf{B} \in \mathbb{C}^{M\times M}$, list of uniformly sampled indices $\mathcal{J}$, stepsize β, regularization parameter λ, error thresholds ϵ_{abs} and ϵ_{rel}, iteration max N_{iter}, matrix $\mathbf{P} \in \mathbb{R}^{M\times N}$.
Output : A real-valued 2D matrix $\hat{\mathbf{S}}$
Initialize $\hat{\mathbf{m}}_\mathbf{1}$ with $\mathbf{1_{N^2\times 1}}$, $\tilde{\mathbf{p}}$ with $\mathbf{0_{N\times 1}}$; $\tilde{\mathbf{p}}[\mathcal{J}] = 1$;
set $\hat{\mathbf{m}} = \hat{\mathbf{m}}_\mathbf{1} + \tilde{\mathbf{p}} \bigotimes \tilde{\mathbf{p}}$;
Initialize U, Z $\hat{\mathbf{A}}$ with $\mathbf{0_{N\times N}}$;
$\hat{\mathbf{A}}[\mathcal{J}, \mathcal{J}] = \mathbf{B}$;
while $k <$ N_{iter} **do**
  $\hat{\mathbf{a}} = \text{vec}(\hat{\mathbf{A}} + IFFT_{2D}(\beta\mathbf{Z}^{(k)}))$;
  $\mathbf{U^{(k+1)}} = FFT_{2D}[\text{vec}^{-1}(\hat{\mathbf{a}} \oslash \hat{\mathbf{m}})]$;
  $\mathbf{Z^{(k+1)}} = \text{SoftThresh}(\mathbf{U^{(k+1)}} + \mathbf{Y^{(k)}}/\beta, \lambda/\beta)$;
  $\mathbf{Y^{(k+1)}} = \mathbf{Y^{(k)}} + \beta(\mathbf{U^{(k+1)}} - \mathbf{Z^{(k+1)}})$;
  `// Stopping criteria using primal and dual tolerances`
  $\epsilon_{pri} = N\epsilon_{abs} + \epsilon_{rel}\max(||\mathbf{U^{(k)}}||_2, ||\mathbf{Z^{(k)}}||_2)$;
  $\epsilon_{dual} = N\epsilon_{abs} + \epsilon_{rel}||\mathbf{Y^{(k)}}||_2$;
  if $||\mathbf{R^{(k)}}||_2 < \epsilon_{pri}$ *and* $||\mathbf{S^{(k)}}||_2 < \epsilon_{dual}$ **then**
    return $\hat{\mathbf{S}} = \mathbf{U^{(k)}}/N^2$;
  end
end
return $\hat{\mathbf{S}} = \mathbf{U^{(k)}}/N^2$;

We can thus use these residuals to put a bound on the objective suboptimality of the current point, $\frac{1}{2}||\mathbf{F_P^{-1}U^{(k)}}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z^{(k)}}||_1 - q^*$, where q^* is the optimal value that the objective function given in Equation 12 will converge to for $k \to \infty$. The following equation appearing in the proof of convergence in the Supplement,

$$\frac{1}{2}||\tilde{\mathbf{F}}_\mathbf{P}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k})} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 - q^* \leq -(\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{r}}^{(\mathbf{k})} + (\tilde{\mathbf{u}}^{(\mathbf{k})} - \tilde{\mathbf{u}}^*)^T\tilde{\mathbf{s}}^{(\mathbf{k})},$$

suggests that when the primal and dual residuals are small, the objective suboptimality must also be small, thereby motivating the use of the following stopping criterion:

$$||\mathbf{R^{(k)}}||_2 \leq \epsilon_{pri} \quad \text{and} \quad ||\mathbf{S^{(k)}}||_2 \leq \epsilon_{dual}.$$

Here $\epsilon_{pri} > 0$ and $\epsilon_{dual} > 0$ are the tolerances for the primal (19) and dual feasibility (20) conditions respectively. We choose the following tolerances for robustness analysis,

$$\epsilon_{pri} = N\epsilon_{abs} + \epsilon_{rel}\max(||\mathbf{U^{(k)}}||_2, ||\mathbf{Z^{(k)}}||_2), \tag{22}$$

$$\epsilon_{dual} = N\epsilon_{abs} + \epsilon_{rel}, ||\mathbf{Y^{(k)}}||_2, \tag{23}$$

where ϵ_{abs} and ϵ_{rel} are the absolute and relative tolerances respectively, typically chosen between 10^{-2} to 10^{-4} depending on the application.

5 APPLICATION AND EMPIRICAL STUDY

We implement the framework described in the previous section to recover the transition probabilities for a two-type branching process; the idea and the framework can be easily extended to multi-type branching processes as well. As an outline, we calculate the full transition probability matrix $\mathbf{S}$ for a two-type branching process with known rates, randomly select a subset of the PGF evaluations $\mathbf{B}$, and use ADMM along with CS to recover the transition probabilities $\hat{\mathbf{S}}$ while also exploring the robustness of the ADMM algorithm with respect to the stepsize β, and the regularization parameter λ. Once satisfied with the robustness, we then proceed to compare the runtimes of recovering the transition probabilities using ADMM algorithm with the runtimes of PGD algorithm for similar errors.

We describe the two model applications used to generate data in our study, following Xu and Minin [2015].

5.1 Two-Compartment Hematopoiesis Model

Hematopoiesis is a process of continuous formation and turnover of blood cells in order to fulfill the everyday demands of the body. At the origin of hematopoiesis is hematopoietic stem cell (HSC) which has two fundamental features: the first being the ability to self-renew, a divisional event which results in the formation of two HSCs from one HSC and the second being the ability for multipotent differentiation into all mature blood lineages.

We can stochastically model this biological phenomenon as a two type branching process [Catlin et al., 2001]. Under such representation, X_1 and X_2, the type one and type two particle populations correspond to HSCs and progenitor cells respectively. With the same parameters as denoted in Figure 4a of the Supplement, the non-zero instantaneous rates that define the process are,

$$\begin{aligned} &a_1(2,0) = \rho, a_1(0,1) = \nu, a_1(1,0) = -(\rho+\nu), \\ &a_2(0,0) = \mu, a_2(0,1) = -\mu. \end{aligned} \tag{24}$$

Now that we have the instantaneous rates, we can derive the solutions for its PGF, defined in Equation 2 and subsequently obtain the transition probabilities using Equation 4 with the details included in the Supplement. Note that the cell population in this application can reach to the order of tens of thousands, thereby motivating the splitting methods approach to compute transition probabilities in an efficient manner.

5.2 Birth-Death-Shift Model for Transposons

For our second application, we look at the birth-death-shift (BDS) process proposed by Rosenberg et al. [2003] to model evolutionary dynamics of the genomic mobile sequence elements known as *transposons*. Each transposon can either produce a new copy which can move to a new genomic location, shift to a different genomic location, or be completely removed from the genome, independently of all other transposons with per-particle instantaneous rates γ, σ, δ and overall rates proportional to the total number of transposons. The $IS6110$ transposon in the *Mycobacterium tuberculosis* genome from a San Francisco community study dataset [Cattamanchi et al., 2006] was used to estimate these evolutionary rates by Rosenberg et al. [2003]. A two-type branching process can be used to model the BDS process over a finite observation interval, where X_1 and X_2 represent the numbers of initially occupied genomic locations and newly occupied genomic locations respectively, thereby capturing the full dynamics of the BDS process.

The two-type branching process representation of the BDS model has the following non-zero rates

$$\begin{aligned} &a_1(1,1) = \gamma, a_1(0,1) = \sigma, a_1(0,0) = \delta, \\ &a_1(1,0) = -(\gamma+\sigma+\delta), a_2(0,2) = \gamma, \\ &a_2(0,1) = -(\gamma+\delta), a_2(0,0) = \delta. \end{aligned} \tag{25}$$

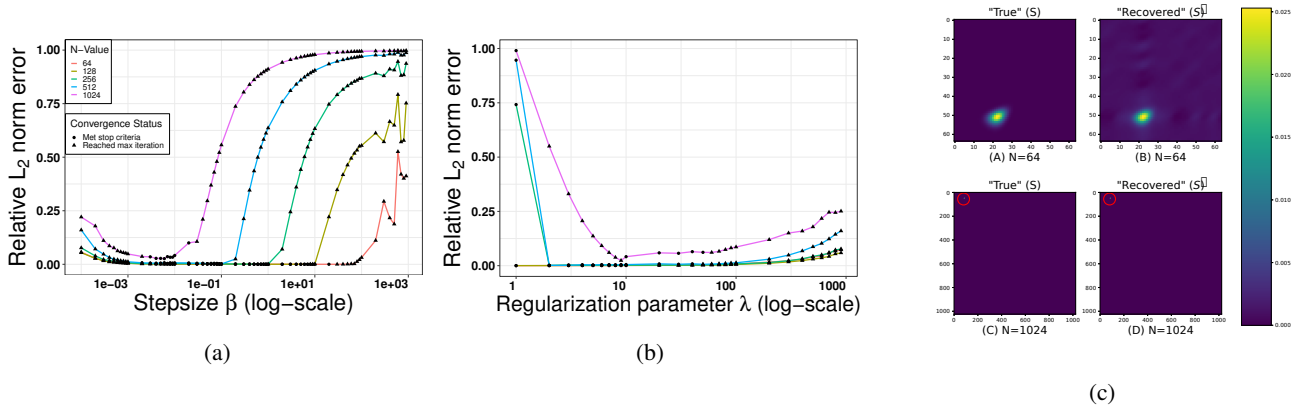
In our application, we calculate the transition probabilities $\mathbf{S}$ for maximum population sizes $N = 2^6, 2^7, \ldots, 2^{10}$, given the time intervals t, branching process' rate parameters $\boldsymbol{\theta}$ and initial population size $\mathbf{X}(\mathbf{0})$. Based on biologically sensible rates and observation times scales of data from previous hematopoiesis studies [Catlin et al., 2001, Golinelli et al., 2006, Fong et al., 2009], we set per-week branching rates and the observation time as $\boldsymbol{\theta}_{HSC} = [\rho, \nu, \mu] = [0.125, 0.104, 0.147]$ and $t = 1$ week respectively. Similarly for the BDS application, based on previously estimated rates in Xu et al. [2015] and the average length between observations in the tuberculosis dataset from San Francisco [Cattamanchi et al., 2006], we set the per year event rates and the observation time as $\boldsymbol{\theta}_{BDS} = [\gamma, \sigma, \delta] = [0.016, 0.004, 0.019]$ and $t = 0.35$ years respectively. In each case, we computed $M^2 = (\min(\lfloor\sqrt{\log NK10}\rfloor, \lfloor\sqrt{N} - \sqrt{N}/5\rfloor)^2$ total random measurements to obtain $\mathbf{B}$, which will be used to recover the transition probabilities.

5.3 Robustness of ADMM Algorithm

We begin by investigating the robustness of our ADMM algorithm under a broad range of parameter settings. We consider varying the stepsize β and the regularization parameter λ for each N, and repeat this procedure across all N values for the HSC model while examining the effects of these variations on the relative L_2 errors $\epsilon_{rel}^{L_2}$ defined

$$\epsilon_{rel}^{L_2} = ||\hat{\mathbf{S}} - \mathbf{S}||_2 / ||\mathbf{S}||_2,$$

where $||.||$ indicates the L_2 norm, $\mathbf{S}$ and $\hat{\mathbf{S}}$ represent the "true" and recovered transition probabilities respectively.



(a) (b) (c)

Figure 1: Relative L_2 norm error, $\epsilon_{rel}^{L_2}$ for (a) varying stepsizes β and (b) varying regularization parameter λ under varying settings of N, HSC model. The successful convergence of ADMM algorithm is observed over a large range of stepsizes and λ. c) The sparse "true" transition probability matrix ($\mathbf{S}$) along with the recovered transition probability matrix ($\hat{\mathbf{S}}$) for $N = 64$ and $N = 1024$ for the HSC model using ADMM, with red circles highlighting non-zero transition probabilities.

In more detail, we start by fixing $\lambda = 0.5(\log M)$ and vary β in equal intervals from 10^{-4} to 9×10^2 while recording and subsequently plotting $\epsilon_{rel}^{L_2}$ corresponding to each β. Thereafter, we fix $\beta = 0.08$, vary λ from 1 to 10^3 and plot $\epsilon_{rel}^{L_2}$ corresponding to each λ. The primal ϵ_{pri} and dual ϵ_{dual} feasibility tolerances are given by Equations 22 and 23 respectively with $\epsilon_{abs} = \epsilon_{rel} = 10^{-3}$.

Figure 1 displays these results. The algorithm is fairly inelastic to both the tuning of the stepsize β as well as the penalty parameter λ: it achieves low errors conveying successful signal recovery for a wide range of values, which is visually evident even on the log scale. This is promising as our method may enter as a subroutine in optimization frameworks when the scale of these parameters is unknown, and must be learned throughout an iterative scheme.

5.4 Comparing the Performance of PGD and ADMM

Having confirmed that the method is not too sensitive to tuning, we compare its performance to the PGD algorithm proposed in Xu and Minin [2015]. In order to compare the performance we first compute sets of transition probabilities $\mathbf{S}$ of both HSC and BDS models using the full set of PGF solution measurements $\tilde{\mathbf{B}}$ as described in Equation 11.

Following Xu and Minin [2015] we set the regularization parameters as $\lambda_{HSC}^{PGD} = \sqrt{\log M}$ and $\lambda_{BDS}^{PGD} = \log M$ for the PGD algorithm while keeping $\lambda = 0.5(\log M)$ for the ADMM algorithm for both the applications. It is promising that this simple heuristic for choosing the parameters leads to successful performance across *all settings we consider*, further validating our empirical validation of robustness to tuning. Figure 1c illustrates the sparse solution and provides a visual demonstration of the accuracy of reconstructed solutions using our ADMM approach, with the estimated matrix $\hat{\mathbf{S}}$ essentially identical to the ground truth.

For each value of N, we repeat this procedure five times under randomly sampled indices $\mathcal{J}$, and recover the transition probabilities $\hat{\mathbf{S}}$ using only a subset of PGF solution measurements $\mathbf{B}$ for both PGD and ADMM algorithms. We report median runtimes over the five trials in a fair conservative comparison, in that we ensure the measure of accuracy $\epsilon_{rel}^{L_2}$ under our proposed method is at least as good as PGD. To match errors across the algorithms, we choose

$$\epsilon_{pri} = D_1 \epsilon_{abs} + \epsilon_{rel} \max(||\mathbf{U^{(k)}}||_2, ||\mathbf{Z^{(k)}}||_2),$$

$$\epsilon_{dual} = D_2 \epsilon_{abs} + \epsilon_{rel} ||\mathbf{Y^{(k)}}||_2,$$

where D_1, D_2 are $\{N^2, N^5\}$ and $\{N^2, N^2\}$ for the HSC and BDS models respectively, complete details summarized in Table S1 of the Supplement. Since our ADMM algorithm largely depends on the FFT, a GPU based algorithm was also implemented to show that the algorithm can be accelerated drastically through straightforward parallelization, thus making it highly scalable for large data. Further details on the GPU version are included in the Supplement.

The median runtimes of PGD, ADMM (Vanilla), and ADMM (GPU) for both HSC and BDS models are reported in Figure 2 while the median relative L_2 norm errors, $\epsilon_{rel}^{L_2}$ are reported in the Supplement. As evident from Figure 2, the ADMM algorithm consistently outperforms the PGD algorithm across all N values for both the HSC and BDS models. For instance, with $N = 1024$, there is a 97.65% and 71.12% reduction in the runtimes of ADMM (Vanilla) algorithm for the HSC model and BDS models respectively while the GPU implementation of ADMM is about 21 times faster than ADMM (vanilla) and about 885 times faster than PGD algorithm for the HSC model. Similarly, for $N = 1024$ the ADMM (GPU) algorithm is about 40 times faster than ADMM (vanilla) and about 140 times faster than PGD algorithm for the BDS model.

Computing Infrastructure The experiment is conducted on Google Colaboratory[1] platform that claimed to have a RAM of 12.6 GB, a single-core Xeon CPU of 2.20 Ghz (No Turbo Boost) and a Tesla K80 with 2496 CUDA cores and 12GB GPU memory as well. The GPU version of FFT was implemented in CuPy [Okuta et al., 2017], a Numpy-like API library for CUDA with matched parameters in both the vanilla and GPU implementations of ADMM. The code and data used to implement the ADMM algorithm can be accessed from `https://github.com/awasthi-12/Transition_Prob_via_ADMM/`.

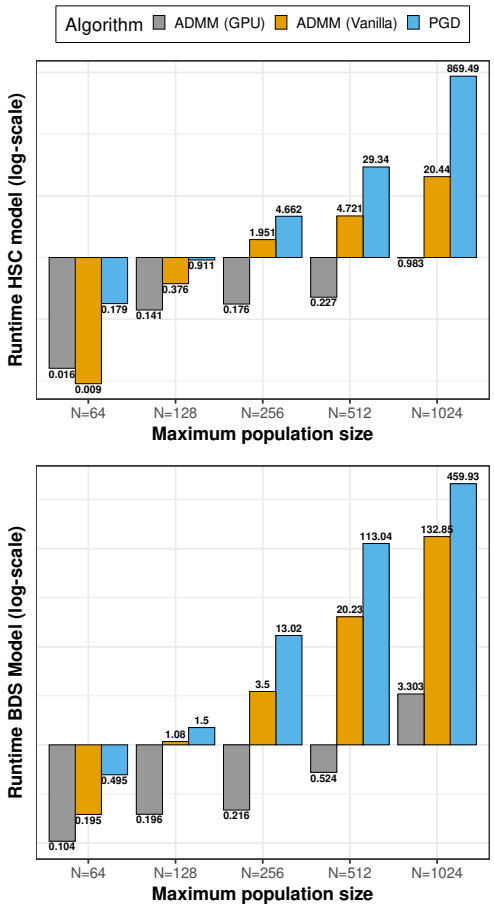


Figure 2: A comparison of median running times for different algorithms and maximum population sizes for the HSC model and BDS model respectively. For all population sizes N considered under each model, ADMM algorithm outpaces PGD algorithm under comparable or lower errors at convergence. This discrepancy becomes more pronounced as the size of the problem grows; note results are plotted on the log-scale.

[1] https://colab.research.google.com/

6 DISCUSSION

We revisit the computational challenge of computing transition probabilities of large-scale multi-type branching processes. Focusing on the two-type setting, we derive a novel application of ADMM, showing how variable splitting can significantly accelerate methods to compute transition probabilities within a compressed sensing paradigm. Through a suite of experiments, we validate not only the robustness of ADMM with respect to the stepsize β and the regularization parameter λ, but also its superior performance over previous attempts utilizing proximal gradient descent. The advantages of the proposed method become especially pronounced as the population size N increases. We also show through thoughtful algorithmic considerations that the primal variable "**U**"-update in ADMM can be carried out without matrix inversions. In particular, mathematically manipulating the expressions to leverage the FFT, the dominant complexity requires $O(N^2 \log_2 N)$ flops, dramatically increasing the scale of problems the method can consider. Lastly, a GPU-based parallel implementation further reduces the runtimes over the standard version.

In many realistic data settings where these stochastic population models apply, it is natural to expect sparsity in the support of transition probabilities at finite time-scales of interest. We have been able to achieve precise results in two such relevant examples under realistic parameter settings from scientific literature. Importantly, not only can the relevant quantities be computed efficiently, but in a way that succeeds over a wide range of tuning parameters. This robustness to tuning will be crucial when embedding these methods as subroutines within inferential schemes such as maximum likelihood estimation [Doss et al., 2013, Xu et al., 2015]. Recall that for a Markov process $\mathbf{X}(t)$ observed at discrete times $\{t_1, \dots t_T\}$, the likelihood function of observed data is a product of its transition probabilities

$$\mathcal{L}(\mathbf{X}; \boldsymbol{\theta}) = \prod_{i=1}^{T-1} p_{\mathbf{X}(t_i), \mathbf{X}(t_{i+1})}(t_{i+1} - t_i; \boldsymbol{\theta}).$$

Optimizing this likelihood numerically often entails iterative schemes, and similar bottlenecks arise under other estimators as well as Bayesian approaches where the likelihood appears in routines such as Metropolis-Hastings ratios [Guttorp, 2018, Stutz et al., 2022]. The robustness, efficiency, and accuracy of the proposed method make it well-suited to aid or enable likelihood-based inference in challenging missing data settings.

Acknowledgements

This work was partially supported by NSF grants DMS-2230074 and PIPP-2200047. We thank Galen Reeves for helpful discussions and Yiwen Wang for early contributions to the code.

References

Linda JS Allen. *An introduction to stochastic processes with applications to biology*. CRC press, 2010.

Norman TJ Bailey. *The Elements of Stochastic Processes with Applications to the Natural Sciences*, volume 25. New York: John Wiley & Sons, 1991.

Jonathan Borwein and Adrian Lewis. *Convex Analysis*. Springer, 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Emmanuel J Candès. Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Citeseer, 2006.

Emmanuel J Candès and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.

Emmanuel J Candès and Terence Tao. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

Sandra N Catlin, Janis L Abkowitz, and Peter Guttorp. Statistical inference in a two-compartment model for hematopoiesis. *Biometrics*, 57(2):546–553, 2001.

A Cattamanchi, PC Hopewell, LC Gonzalez, DH Osmond, L Masae Kawamura, CL Daley, and RM Jasmer. A 13-year molecular epidemiological analysis of tuberculosis in San Francisco. *The International Journal of Tuberculosis and Lung Disease*, 10(3):297–304, 2006.

Forrest W Crawford, Vladimir N Minin, and Marc A Suchard. Estimation for general birth-death processes. *Journal of the American Statistical Association*, 109(506): 730–747, 2014.

David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

Charles R Doss, Marc A Suchard, Ian Holmes, Midori Kato-Maeda, and Vladimir N Minin. Fitting birth-death processes to panel data with applications to bacterial DNA fingerprinting. *The annals of applied statistics*, 7(4):2315, 2013.

Jonathan Eckstein and Dimitri P Bertsekas. On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992.

Jonathan Eckstein and Masao Fukushima. Some reformulations and applications of the Alternating Direction Method of Multipliers. In *Large scale optimization*, pages 115–134. Springer, 1994.

Hugh Everett III. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations research*, 11(3):399–417, 1963.

Gerald B Folland. *Real analysis: modern techniques and their applications*, volume 40. John Wiley & Sons, 1999.

Youyi Fong, Peter Guttorp, and Janis Abkowitz. Bayesian inference and model choice in a hidden stochastic two-compartment model of hematopoietic stem cell fate decisions. *The annals of applied statistics*, 3(4):1696, 2009.

Matteo Frigo and Steven G Johnson. Fftw: An adaptive software architecture for the FFT. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 3, pages 1381–1384. IEEE, 1998.

Daniel Gabay. Applications of the method of multipliers to variational inequalities. In *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, volume 15, pages 299–331. Elsevier, 1983.

D Golinelli, P Guttorp, and JA Abkowitz. Bayesian inference in a hidden stochastic two-compartment model for feline hematopoiesis. *Mathematical Medicine and Biology*, 23(3):153–172, 2006.

Winfried K Grassmann. Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4 (1):47–53, 1977.

Peter Guttorp. *Stochastic modeling of scientific data*. Chapman and Hall/CRC, 2018.

Monir Hajiaghayi, Bonnie Kirkpatrick, Liangliang Wang, and Alexandre Bouchard-Côté. Efficient continuous-time markov chain estimation. In *International Conference on Machine Learning*, pages 638–646. PMLR, 2014.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J.

Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Simple and practical algorithm for sparse Fourier transform. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1183–1194. SIAM, 2012.

Asger Hobolth and Eric A Stone. Simulation from endpoint-conditioned, continuous-time Markov chains on a finite state space, with applications to molecular evolution. *The annals of applied statistics*, 3(3):1204, 2009.

JD Kalbfleisch and Jerald Franklin Lawless. The analysis of panel data under a markov assumption. *Journal of the american statistical association*, 80(392):863–871, 1985.

Samuel Karlin and Howard M. Taylor. *A First Course in Stochastic Processes (Second Edition)*. Academic Press, second edition edition, 1975.

Kenneth Lange. Calculation of the equilibrium distribution for a deleterious gene by the finite Fourier transform. *Biometrics*, 38:79–86, 1982.

Kenneth Lange. *Applied Probability*. Springer, New York, NY, second edition edition, 2010.

Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis. CuPy: A NumPy-compatible library for NVIDIA GPU calculations. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

Wolfgang Paul and Jörg Baschnagel. *Stochastic processes: From Physics to Finance*. Springer Cham, 2013.

V Rao and YW Teh. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence, UAI 2011*, 2011.

Eric Renshaw. *Stochastic population processes: analysis, approximations, simulations*. OUP Oxford, 2015.

R Tyrrell Rockafellar. *Convex analysis*, volume 18. Princeton university press, 1970.

Noah A Rosenberg, Anthony G Tsolaki, and Mark M Tanaka. Estimating change rates of genetic markers using serial samples: applications to the transposon IS6110 in Mycobacterium tuberculosis. *Theoretical Population Biology*, 63(4):347–363, 2003.

Sheldon M Ross. Approximating transition probabilities and mean occupation times in continuous-time Markov chains. *Probability in the Engineering and Informational Sciences*, 1(3):251–264, 1987.

Mark Rudelson and Roman Vershynin. On sparse reconstruction from Fourier and Gaussian measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 61(8):1025–1045, 2008.

Walter Rudin et al. *Principles of Mathematical Analysis*, volume 3. McGraw-hill New York, 1976.

Timothy C Stutz, Janet S Sinsheimer, Mary Sehl, and Jason Xu. Computational tools for assessing gene therapy under branching process models of mutation. *Bulletin of Mathematical Biology*, 84(1):1–17, 2022.

Simon Tavaré. The linear birth–death process: an inferential retrospective. *Advances in Applied Probability*, 50(A): 253–269, 2018.

Jason Xu and Vladimir N Minin. Efficient transition probability computation for continuous-time branching processes via compressed sensing. In *Uncertainty in Artificial Intelligence*, volume 2015, page 952, 2015.

Jason Xu, Peter Guttorp, Midori Kato-Maeda, and Vladimir N Minin. Likelihood-based inference for discretely observed birth–death-shift processes, with applications to evolution of mobile genetic elements. *Biometrics*, 71(4):1009–1021, 2015.

Supplementary Materials

A DERIVATIONS AND PROOFS

A.1 Discrete Fourier Transform Matrix

The $N \times N$ discrete Fourier transform (DFT) matrix $\mathbf{F}$ has entries $\{\mathbf{F}\}_{j,k} = \frac{1}{\sqrt{N}}(\omega)^{jk}$, where $\omega = e^{2\pi i/N}$ and $j, k = 0, 1, \ldots N-1$. As mentioned in the main text, recall that the Hermitian is given by its conjugate transpose: $\mathbf{F}^H = \left(\overline{\mathbf{F}}\right)^T$. Now let $\mathbf{x}$ be a vector with $\hat{\boldsymbol{x}} = \mathbf{F}\mathbf{x}$ and $\mathbf{x} = \mathbf{F}^H\hat{\boldsymbol{x}}$. Applying both $\mathbf{F}^H$ and $\mathbf{F}$ we get

$$\mathbf{F}^H\mathbf{F}\mathbf{x} = \mathbf{F}^H\hat{\boldsymbol{x}} = \mathbf{x} \implies \mathbf{F}^H\mathbf{F} = \boldsymbol{I}.$$

Thus, the DFT matrix $\mathbf{F}$ is unitary.

A.2 Extended Derivations

Here we provide details on obtaining $\hat{\mathbf{M}}$ and $\hat{\mathbf{a}}$ as described by Equation 17 of the main text. This enables us to update $\mathbf{u}$ efficiently, as mentioned in Section 4.2 of the main text.

We begin with the following equation

$$\mathbf{M}\tilde{\mathbf{u}}^{(\mathbf{k+1})} = \mathbf{a}, \tag{S1}$$

where $\mathbf{M} = \left((\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}}\right) \in \mathbb{C}^{N^2\times N^2}$ and $\mathbf{a} = \left((\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{b}} + \beta\left(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\right)\right) \in \mathbb{C}^{N^2}$. Multiplying both sides of Equation S1 by $\tilde{\mathbf{F}}^{-\mathbf{1}} = \mathbf{F}^{-\mathbf{1}} \bigotimes \mathbf{F}^{-\mathbf{1}} \in \mathbb{C}^{N^2\times N^2}$ and simplifying using the properties of the discrete Fourier transform matrix, the left-hand side of the equation becomes:

$$\begin{aligned}
\tilde{\mathbf{F}}^{-\mathbf{1}}\mathbf{M}\tilde{\mathbf{u}}^{(\mathbf{k+1})} &= \tilde{\mathbf{F}}^{-\mathbf{1}}\left((\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}}\right)\tilde{\mathbf{u}}^{(\mathbf{k+1})} \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}((\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}})^H\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}})\tilde{\mathbf{u}}^{(\mathbf{k+1})} \quad [\tilde{\mathbf{F}}^{-\mathbf{1}} = \tilde{\mathbf{F}}^H \text{ because DFT matrix is unitary}] \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}((\tilde{\mathbf{F}}^H)^H\tilde{\mathbf{P}}^H\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}} + \beta\tilde{\mathbf{I}})\tilde{\mathbf{u}}^{(\mathbf{k+1})} \quad [\tilde{\mathbf{P}}^H = \tilde{\mathbf{P}}^T \text{ because entries of } \tilde{\mathbf{P}} \text{ are real by construction}] \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{F}}\tilde{\mathbf{P}}^T\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} + \beta\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} \quad [\beta \in \mathbb{R} \text{ is a constant}] \\
&= \tilde{\mathbf{P}}^T\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} + \beta\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} \\
&= (\tilde{\mathbf{P}}^T\tilde{\mathbf{P}} + \beta\tilde{\mathbf{I}})\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} \\
&= \hat{\mathbf{M}}\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})}
\end{aligned}$$

Similarly, the right-hand side of Equation S1 becomes:

$$\begin{aligned}
\tilde{\mathbf{F}}^{-\mathbf{1}}\mathbf{a} &= \tilde{\mathbf{F}}^{-\mathbf{1}}\left[(\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}})^H\tilde{\mathbf{b}}\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}} + \beta\left(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\right)\right] \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}\left[(\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}})^H\tilde{\mathbf{b}}\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}} + \beta\left(\tilde{\mathbf{z}}^{(\mathbf{k})} - \frac{\tilde{\mathbf{y}}^{(\mathbf{k})}}{\beta}\right)\right] \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}(\tilde{\mathbf{F}}^H)^H\tilde{\mathbf{P}}^H\tilde{\mathbf{b}}\tilde{\mathbf{P}}\tilde{\mathbf{F}}^{-\mathbf{1}} + \beta\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{y}}^{(\mathbf{k})} \\
&= \tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{F}}\tilde{\mathbf{P}}^T\tilde{\mathbf{b}} + \beta\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{z}}^{(\mathbf{k})} \\
&= \tilde{\mathbf{P}}^T\tilde{\mathbf{b}} + \beta\tilde{\mathbf{F}}^{-\mathbf{1}}\tilde{\mathbf{z}}^{(\mathbf{k})} \\
&= \hat{\mathbf{a}}
\end{aligned}$$

□

A.3 Proofs of Propositions

We present the proofs of propositions 4.2 and 4.3 described in Section 4.2 of the main text, first recalling the statement of results:

Proposition A.1. 4.1 *The functions* $f(\mathbf{U}) = \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2$ *and* $g(\mathbf{Z}) = \lambda||\mathbf{Z}||_1$ *are closed, convex, and proper.*

The proof of Proposition 4.1 has been provided in the main text. Proposition 4.1 implies that there exist $\mathbf{U}$ and $\mathbf{Z}$, not necessarily unique, that minimize the augmented Lagrangian, thereby making the $\mathbf{U}$ and $\mathbf{Z}$ updates

$$\begin{aligned} \mathbf{U^{(k+1)}} &:= \arg\min_{\mathbf{U}} L_\beta(\mathbf{U}, \mathbf{Z^{(k)}}, \mathbf{Y^{(k)}}), \\ \mathbf{Z^{(k+1)}} &:= \arg\min_{\mathbf{Z}} L_\beta(\mathbf{U^{(k+1)}}, \mathbf{Z}, \mathbf{Y^{(k)}}), \end{aligned} \tag{S2}$$

solvable.

Proposition A.2. 4.2 *For* $\beta = 0$, $L_0 = \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1 + \mathbf{Y}^T(\mathbf{U} - \mathbf{Z})$, *the unaugmented Lagrangian has a saddle point for all* $\mathbf{U}, \mathbf{Z}, \mathbf{Y}$.

Proof. A Lagrangian L_0 is said to have a saddle point when there exist $(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*)$ not necessarily unique such that,

$$L_0(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}) \leq L_0(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*) \leq L_0(\mathbf{U}, \mathbf{Z}, \mathbf{Y}^*).$$

From Proposition 4.1 it follows that $L_0(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*)$ is finite for any saddle point $(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*)$. This implies not only that $(\mathbf{U}^*, \mathbf{Z}^*)$ is a solution to the primal objective function $\frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1$, thus $\mathbf{U}^* = \mathbf{Z}^*$ and $\frac{1}{2}||\mathbf{F_P^{-1}U}^*(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 < \infty$ and $\lambda||\mathbf{Z}^*||_1 < \infty$, but also that strong duality holds and $\mathbf{Y}^*$ is a dual optimal, i.e. it maximizes the dual function $\mathbf{Y}^T(\mathbf{U} - \mathbf{Z})$ [Boyd and Vandenberghe, 2004].

We can easily verify that

$$\begin{aligned} \frac{1}{2}||\mathbf{F_P^{-1}U}^*(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{U}^*||_1 &= \frac{1}{2}||\mathbf{F_P^{-1}U}^*(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{U}^*||_1 \\ &< \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1 + (\mathbf{Y}^*)^T(\mathbf{U} - \mathbf{Z}), \end{aligned}$$

where the last inequality follows from $(\mathbf{U}^*, \mathbf{Z}^*)$ minimizing $\frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1$ among all $(\mathbf{U}, \mathbf{Z})$ and $\mathbf{Y}^*$ maximizing the dual objective function among all $\mathbf{Y}$.

□

Proposition A.3. 4.3 *Under propositions 4.1 and 4.2, the ADMM algorithm achieves the following:*

1. *Primal residual convergence:* $\mathbf{R^{(k+1)}} = \mathbf{U^{(k+1)}} - \mathbf{Z^{(k+1)}} \to 0$ *as* $k \to \infty$.
2. *Dual residual convergence:* $\mathbf{S^{(k+1)}} = \beta(\mathbf{Z^{(k+1)}} - \mathbf{Z^{(k)}}) \to 0$ *as* $k \to \infty$.

Proof. The proof of proposition 4.3 [Gabay, 1983, Eckstein and Bertsekas, 1992, Boyd et al., 2011] is divided into three parts based on three central inequalities, all of which we prove as part of this proof.

Let $(\mathbf{U}^*, \mathbf{Z}^*, \mathbf{Y}^*)$ be a saddle point for the unaugmented Lagrangian $L_0 = \frac{1}{2}||\mathbf{F_P^{-1}U}(\mathbf{F_P^{-1}})^H - \mathbf{B}||_2^2 + \lambda||\mathbf{Z}||_1 + \mathbf{Y}^T(\mathbf{U} - \mathbf{Z})$. Equivalently let $(\tilde{\mathbf{u}}^*, \tilde{\mathbf{z}}^*, \tilde{\mathbf{y}}^*)$ be the saddle point for the unaugmented Lagrangian $L_0 = \frac{1}{2}||\tilde{\mathbf{F}}_\mathbf{P}^{-1}\tilde{\mathbf{u}} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}||_1 + \tilde{\mathbf{y}}^T(\tilde{\mathbf{u}} - \tilde{\mathbf{z}})$ and consider the first inequality

$$q^* - q^{(k+1)} \leq (\tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})}, \tag{S3}$$

where q^* is the optimal value to which the objective function converges as $k \to \infty$.

A.3.1 Proof of Inequality S3

The following inequality holds

$$L_0(\tilde{\mathbf{u}}^*, \tilde{\mathbf{z}}^*, \tilde{\mathbf{y}}^*) \leq L_0(\tilde{\mathbf{u}}^{(\mathbf{k+1})}, \tilde{\mathbf{z}}^{(\mathbf{k+1})}, \tilde{\mathbf{y}}^*) \tag{S4}$$

because $(\tilde{\mathbf{u}}^*, \tilde{\mathbf{z}}^*, \tilde{\mathbf{y}}^*)$ is a saddle point for L_0. Using the constraint of the optimization problem $\tilde{\mathbf{u}}^* = \tilde{\mathbf{z}}^*$, the left-hand side of inequality S4 reduces to q^*. The right-hand side of inequality S4 can be simplified as

$$\begin{aligned} L_0(\tilde{\mathbf{u}}^{(\mathbf{k+1})}, \tilde{\mathbf{z}}^{(\mathbf{k+1})}, \tilde{\mathbf{y}}^*) &= \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 + (\tilde{\mathbf{y}}^*)^T(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}) \\ &= q^{(k+1)} + (\tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} \quad \left[\tilde{\mathbf{r}}^{(\mathbf{k+1})} = \tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}\right] \\ &\qquad \left[q^{(k+1)} = \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1\right]. \end{aligned}$$

Thus, we have

$$\begin{aligned} q^* &\leq q^{(k+1)} + (\tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} \\ q^* - q^{(k+1)} &\leq (\tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})}. \end{aligned}$$ □

Next, we consider the second inequality

$$q^{(k+1)} - q^* \leq -(\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} - \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^*)). \tag{S5}$$

A.3.2 Proof of Inequality S5

We know that, by definition, $\tilde{\mathbf{u}}^{(\mathbf{k+1})}$ minimizes $L_\beta(\tilde{\mathbf{u}}, \tilde{\mathbf{z}}^{(\mathbf{k})}, \tilde{\mathbf{y}}^{(\mathbf{k})})$. Since $f(\tilde{\mathbf{u}}) = \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}} - \tilde{\mathbf{b}}||_2^2$ is closed, proper, and convex by Proposition **4.1**, it is differentiable, implying that L_β is subdifferentiable. The necessary and sufficient optimality condition is

$$\begin{aligned} 0 \in \partial L_\beta(\tilde{\mathbf{u}}^{(\mathbf{k+1})}, \tilde{\mathbf{z}}^{(\mathbf{k})}, \tilde{\mathbf{y}}^{(\mathbf{k})}) &= \partial\Big(\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 + (\tilde{\mathbf{y}}^{(\mathbf{k})})^T(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}) + \frac{\beta}{2}||\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}||_2^2\Big) \\ &= \partial\Big(\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2\Big) + \partial(\lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1) + \partial((\tilde{\mathbf{y}}^{(\mathbf{k})})^T(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})) + \partial\Big(\frac{\beta}{2}||\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}||_2^2\Big) \\ &= \partial\Big(\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2\Big) + \tilde{\mathbf{y}}^{(\mathbf{k})} + \beta(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}), \end{aligned} \tag{S6}$$

where we use the fact that the subdifferential of the sum of a subdifferential function and a differentiable function with domain $\mathbb{R}^N$ is the sum of the subdifferential and the gradient [Rockafellar, 1970].

Since the update of $\tilde{\mathbf{y}}$ is

$$\tilde{\mathbf{y}}^{(\mathbf{k+1})} = \tilde{\mathbf{y}}^{(\mathbf{k})} + \beta(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}),$$

we can substitute $\tilde{\mathbf{y}}^{(\mathbf{k})} = \tilde{\mathbf{y}}^{(\mathbf{k+1})} - \beta(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})})$ in S6 and rearrange the terms to obtain

$$\begin{aligned} 0 &\in \partial\Big(\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2\Big) + \tilde{\mathbf{y}}^{(\mathbf{k+1})} - \beta(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}) + \beta\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \beta\tilde{\mathbf{z}}^{(\mathbf{k})} \\ &= \partial\Big(\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2\Big) + \tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}). \end{aligned}$$

This implies that $\tilde{\mathbf{u}}^{(\mathbf{k+1})}$ minimizes

$$\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}.$$

Analogously we can show that $\tilde{\mathbf{z}}^{(\mathbf{k+1})}$ minimizes $\lambda||\tilde{\mathbf{z}}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}$ and it follows that,

$$\begin{aligned} &\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^{(\mathbf{k+1})} \\ &\qquad \leq \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^* - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^* \end{aligned} \tag{S7}$$

and

$$\lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{(\mathbf{k+1})} \leq \lambda||\tilde{\mathbf{z}}^*||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^* \tag{S8}$$

Adding the inequalities S7 and S8, we obtain

$$\begin{aligned}&\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^{(\mathbf{k+1})} + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{(\mathbf{k+1})} \\ &\leq \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^* - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^* + \lambda||\tilde{\mathbf{z}}^*||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^*. \end{aligned} \tag{S9}$$

We can rearrange and simplify the terms on the left-hand side of inequality S9,

$$\begin{aligned}&\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\mathbf{u}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^{(\mathbf{k+1})} + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{(\mathbf{k+1})} \\ &= \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T(\tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}) \\ &\qquad + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T\tilde{\mathbf{u}}^{(\mathbf{k+1})} \quad \text{[Rearranging the terms]} \\ &= q^{(k+1)} + (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{r}}^{(\mathbf{k+1})} + \tilde{\mathbf{z}}^{(\mathbf{k+1})}) \quad \text{[Because } \tilde{\mathbf{r}}^{(\mathbf{k+1})} = \tilde{\mathbf{u}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k+1})}] \end{aligned} \tag{S10}$$

Similarly, we can also simplify the terms on the right-hand side of the inequality S9,

$$\begin{aligned}&\frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^* - \tilde{\mathbf{b}}||_2^2 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})} + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}))^T\tilde{\mathbf{u}}^* + \lambda||\tilde{\mathbf{z}}^*||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^* \\ &= \frac{1}{2}||\tilde{\mathbf{F}}_{\mathbf{P}}^{-\mathbf{1}}\tilde{\mathbf{u}}^* - \tilde{\mathbf{b}}||_2^2 + \lambda||\tilde{\mathbf{z}}^*||_1 + (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T(\tilde{\mathbf{u}}^* - \tilde{\mathbf{z}}^*) + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T\tilde{\mathbf{u}}^* \quad \text{[Rearranging the terms]} \\ &= q^* + \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}^* \quad \text{[Because of the optimization constraint } \tilde{\mathbf{u}}^* = \tilde{\mathbf{z}}^*] \end{aligned} \tag{S11}$$

We can combine S10 and S11 and rearrange the terms to obtain inequality S5

$$q^{(k+1)} - q^* \leq -(\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} - \beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^*)). \quad \square$$

Now that we have proved the inequalities S4 and S5), only the last inequality

$$V^{(k)} - V^{(k+1)} \geq \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2, \tag{S12}$$

remains to be proven.

A.3.3 Proof of Inequality S12

Recall that $(\tilde{\mathbf{u}}^*, \tilde{\mathbf{z}}^*, \tilde{\mathbf{y}}^*)$ is a saddle point for the unaugmented Lagrangian L_0 and define

$$V^{(k)} := \frac{1}{\beta}||\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^*||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{z}}^*)||_2^2.$$

Adding inequalities S4 and S5, rearranging the terms, and multiplying by 2 on both sides gives us

$$2(\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + 2\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + 2\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^*) \leq 0. \tag{S13}$$

The first term of inequality S13 can be rewritten as

$$\begin{aligned}2(\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} &= 2(\tilde{\mathbf{y}}^{(\mathbf{k})} + \beta\tilde{\mathbf{r}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} \qquad \text{[Because } \tilde{\mathbf{y}}^{(\mathbf{k+1})} = \tilde{\mathbf{y}}^{(\mathbf{k})} + \beta\tilde{\mathbf{r}}^{(\mathbf{k+1})}] \\ &= 2(\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + 2\beta(\tilde{\mathbf{r}}^{(\mathbf{k+1})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} \qquad [\beta^T = \beta \text{ since } \beta \in \mathbb{R}] \\ &= 2(\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^*)^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2, \qquad [(\tilde{\mathbf{r}}^{(\mathbf{k+1})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} = ||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2]\end{aligned}$$

and plugging $\tilde{\mathbf{r}}^{(\mathbf{k+1})} = \frac{1}{\beta}(\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{(\mathbf{k})})$ in the first two terms gives

$$\begin{aligned}
&\frac{2}{\beta}(\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^{*})^T(\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{(\mathbf{k})}) + \frac{1}{\beta}||\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{(\mathbf{k})}||_2^2 + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 \\
&= \frac{1}{\beta}\Big(||\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{*}||_2^2 - ||\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^{*}||_2^2\Big) + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 \qquad \text{[Adding and subtracting } \tilde{\mathbf{y}}^{*}\text{]}. \qquad \text{(S14)}
\end{aligned}$$

We can now rewrite the remaining terms of the inequality S13

$$\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + 2\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T\tilde{\mathbf{r}}^{(\mathbf{k+1})} + 2\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{*}), \qquad \text{(S15)}$$

where the term $\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2$ is extracted from Equation S14. Adding and subtracting $\tilde{\mathbf{z}}^{(\mathbf{k})}$ from the last term of S15 gives

$$\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 + 2\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})^T(\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{z}}^{*}),$$

and by substituting $\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})} = (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{*}) - (\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{z}}^{*})$ in the last two terms, we get

$$\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 + \beta\Big(||\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{*}||_2^2 - ||\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{z}}^{*}||_2^2\Big).$$

This implies that inequality S13 can be written as

$$\begin{aligned}
&\frac{1}{\beta}\Big(||\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{*}||_2^2 - ||\tilde{\mathbf{y}}^{(\mathbf{k})} - \tilde{\mathbf{y}}^{*}||_2^2\Big) + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 + \beta\Big(||\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{*}||_2^2 - ||\tilde{\mathbf{z}}^{(\mathbf{k})} - \tilde{\mathbf{z}}^{*}||_2^2\Big) \leq 0 \\
&\qquad\qquad \implies V^{(k)} - V^{(k+1)} \geq \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2
\end{aligned}$$

We can expand $\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2$ as

$$\beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + 2\beta(\tilde{\mathbf{r}}^{(\mathbf{k+1})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}) + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2.$$

Thus, to prove the third inequality S12, it suffices to show that $2\beta(\tilde{\mathbf{r}}^{(\mathbf{k+1})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})$ is positive. Recall that $\tilde{\mathbf{z}}^{(\mathbf{k+1})}$ minimizes $\lambda||\tilde{\mathbf{z}}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}$ and similarly, $\tilde{\mathbf{z}}^{(\mathbf{k})}$ minimizes $\lambda||\tilde{\mathbf{z}}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}$ which allows us to add

$$\lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{\mathbf{k+1}} \leq \lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{\mathbf{k}}$$

and

$$\lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}^{\mathbf{k}} \leq \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}^{\mathbf{k+1}}$$

to get

$$\begin{aligned}
&\lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{\mathbf{k+1}} + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}^{\mathbf{k}} \\
&\qquad\qquad \leq \lambda||\tilde{\mathbf{z}}^{(\mathbf{k})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k+1})})^T\tilde{\mathbf{z}}^{\mathbf{k}} + \lambda||\tilde{\mathbf{z}}^{(\mathbf{k+1})}||_1 - (\tilde{\mathbf{y}}^{(\mathbf{k})})^T\tilde{\mathbf{z}}^{\mathbf{k+1}} \\
&\qquad\qquad \implies (\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{(\mathbf{k})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}) \geq 0.
\end{aligned}$$

We substitute $\tilde{\mathbf{y}}^{(\mathbf{k+1})} - \tilde{\mathbf{y}}^{(\mathbf{k})} = \beta\tilde{\mathbf{r}}^{(\mathbf{k+1})}$ to get

$$\beta(\tilde{\mathbf{r}}^{(\mathbf{k+1})})^T(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}) \geq 0,$$

because $\beta > 0$ and subsequently

$$\begin{aligned}
V^{(k)} - V^{(k+1)} &\geq \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})} + (\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \\
&\geq \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2, \quad \square
\end{aligned}$$

thus proving the third inequality. The convergence of ADMM is a consequence of the three inequalities S3, S7, and S12. Finally, the inequality S12 can be rewritten as

$$V^{(k+1)} \leq V^{(k)} - \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 - \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2,$$

which implies that $V^{(k)}$ decreases in each iteration, thus bounding $\tilde{\mathbf{y}}^{(\mathbf{k})}$ and $\tilde{\mathbf{z}}^{(\mathbf{k})}$ as $V^{(k)} < V^0$. In particular, note that inductively inequality S12 leads to the following set of inequalities

$$\begin{aligned}
V^{(0)} &\geq V^{(1)} + \beta||\tilde{\mathbf{r}}^{(\mathbf{1})}||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{1})} - \tilde{\mathbf{z}}^{(\mathbf{0})})||_2^2, \\
V^{(1)} &\geq V^{(2)} + \beta||\tilde{\mathbf{r}}^{(\mathbf{2})}||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{2})} - \tilde{\mathbf{z}}^{(\mathbf{1})})||_2^2, \\
&\vdots \\
V^{(k)} &\geq V^{(k+1)} + \beta||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + \beta||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2, \\
&\vdots
\end{aligned}$$

which, upon summation reveals

$$\begin{aligned}
\sum_{k=0}^{\infty} V^{(k)} &\geq \sum_{k=0}^{\infty} V^{(k+1)} + \beta \sum_{k=0}^{\infty} \left(||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + ||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \right) \\
\sum_{k=0}^{\infty} (V^{(k)} - V^{(k+1)}) &\geq \beta \sum_{k=0}^{\infty} \left(||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + ||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \right) \\
V^{(0)} &\geq \beta \sum_{k=0}^{\infty} \left(||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + ||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \right) \qquad \left[\sum_{k=0}^{\infty} (V^{(k)} - V^{(k+1)}) \text{ is a telescopic series} \right].
\end{aligned}$$

Thus the sum of non-negative terms on the right-hand side $\sum_{k=0}^{\infty} \left(||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + ||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \right)$ converges, as the n^{th} partial sum is bounded above by

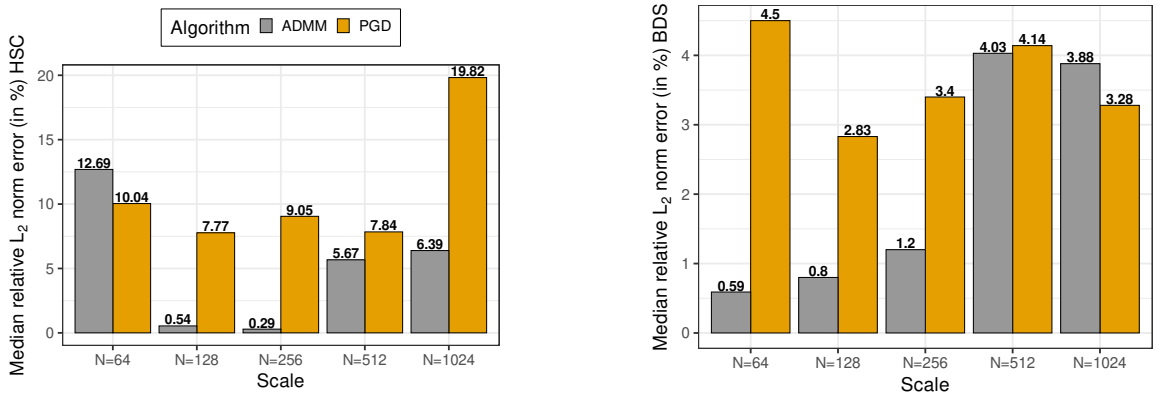
$$\sum_{k=0}^{n} \left(||\tilde{\mathbf{r}}^{(\mathbf{k+1})}||_2^2 + ||(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})})||_2^2 \right) \leq \frac{V^0}{\beta}.$$

Thus, the primary residual, $\tilde{\mathbf{r}}^{(\mathbf{k+1})} \to 0$ and the secondary residual $\beta(\tilde{\mathbf{z}}^{(\mathbf{k+1})} - \tilde{\mathbf{z}}^{(\mathbf{k})}) \to 0$ as $k \to \infty$ immediately from $\beta > 0$ which is equivalent to $\mathbf{R}^{(\mathbf{k+1})} \to 0$ and $\beta(\mathbf{Z}^{(\mathbf{k+1})} - \mathbf{Z}^{(\mathbf{k})}) \to 0$ as $k \to \infty$, completing the proof of Proposition 3. □

B Additional FIGURES AND TABLES

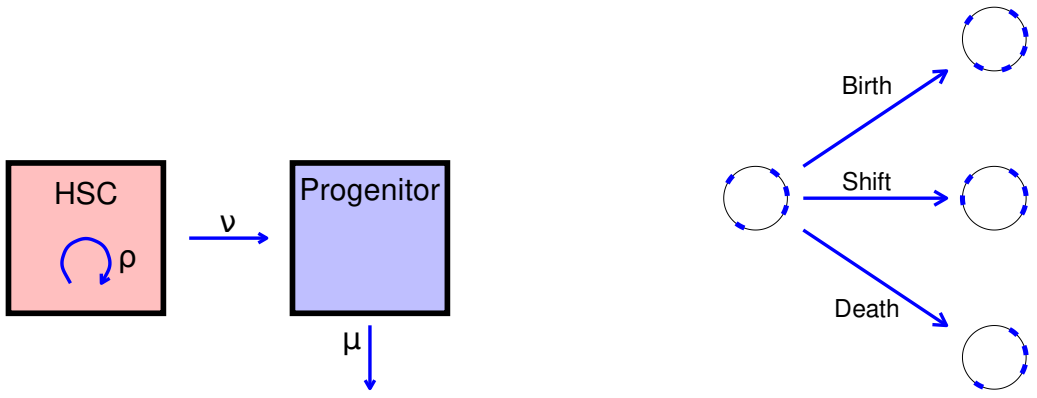
	HSC Model					**BDS Model**				
	$N=64$	128	256	512	1024	$N=64$	128	256	512	1024
Stepsize (β)	0.08	0.005	0.08	0.005	0.005	0.005	0.005	0.005	0.0005	0.0005
D_1	N^2	N^2	N^2	N^2	N^2	N^2	N^2	N^2	N^2	N
D_2	N^5	N^5	N^5	N^5	N^5	N^2	N^2	N^2	N^2	N
ϵ_{abs}	10^{-2}	10^{-2}	10^{-2}	10^{-2}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}
ϵ_{abs}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}

Table 1: This table provides complete details of the ADMM algorithm in our proposed framework, including the step size β, the tolerances for the primal and dual feasibility conditions, $\epsilon_{pri} = D_1\epsilon_{abs} + \epsilon_{rel} \max(||\mathbf{U}^{(\mathbf{k})}||_2, ||\mathbf{Z}^{(\mathbf{k})}||_2)$ and $\epsilon_{dual} = D_2\epsilon_{abs} + \epsilon_{rel}||\mathbf{Y}^{(\mathbf{k})}||_2$ respectively, and ϵ_{abs} and ϵ_{rel} are the absolute and relative tolerances respectively. The regularization parameter λ was chosen to be $0.5(\log M)$ for both models. These were chosen in a simple manner across *all* trials but our framework still yielded promising performance results as compared to the PGD algorithm.



(a) (b)

Figure 3: The median relative norm errors $\epsilon_{rel}^{L_2}$ in recovering the transition probabilities using both the PGD and ADMM algorithms for the a) HSC model and the b) BDS model. Note that, in contrast to runtimes, $\epsilon_{rel}^{L_2}$ was the same for both the vanilla and GPU implementations of the ADMM algorithm.



(a) (b)

Figure 4: a) HSCs can self-renew, producing new HSCs at rate ρ, or differentiate into progenitor cells at rate ν. Further progenitor differentiation is modeled by rate μ b) Illustration of the three types of transposition - birth, death, shift - along a genome, represented by circles [Rosenberg et al., 2003]. Transposons are depicted by rectangles occupying locations along the circles/genomes. On the right set of diagrams, a birth event keeps the number of type 1 particles intact and increments the number of type 2 particles by one, a death event changes the number of type 1 particles from five to four and keeps the number of type 2 particles at zero, and finally a shift event decreases the number of type 1 particles by one and increases the number of type 2 particles by one.

Tables 2 and 3 show the median running times for different algorithms namely, PGD, ADMM (Vanilla), and ADMM (GPU) and the percentage change in runtimes relative to the runtime of the PGD algorithm,

$$\% \text{ change} = \frac{\text{runtime}_{ADMM} - \text{runtime}_{PGD}}{\text{runtime}_{PGD}},$$

for the HSC and BDS models, respectively. The negative percent change indicates that the ADMM algorithm is faster than the PGD algorithm for the corresponding value N.

		Median runtime		
Maximum population size	M	**PGD**	**ADMM (% change)**	**ADMM (GPU) (% change)**
$N=64$	51	0.179	0.009 (-94.97%)	0.016 (-91.06%)
$N=128$	78	0.911	0.376 (-58.73%)	0.141 (-84.52%)
$N=256$	83	4.662	1.951 (-58.15%)	0.176 (-96.22%)
$N=512$	88	29.34	4.721 (-83.91%)	0.227 (-99.23%)
$N=1024$	93	869.49	20.44 (-97.65%)	0.983 (-99.89%)

Table 2: A comparison of median running times (percent change relative to the runtime of the PGD algorithm) for different algorithms and scales for the HSC model. For all maximum population sizes, our proposed framework is faster in recovering the transition probabilities than the PGD algorithm for errors at least as good as PGD.

		Median runtime		
Maximum population size	M	**PGD**	**ADMM (% change)**	**ADMM (GPU) (% change)**
$N=64$	18	0.495	0.195 (-60.60%)	0.104 (-78.99%)
$N=128$	19	1.50	1.09 (-27.33%)	0.196 (-86.93%)
$N=256$	29	13.02	3.50 (-86.93%)	0.216 (-98.34%)
$N=512$	22	113.04	20.23 (-82.10%)	0.524 (-99.54%)
$N=1024$	28	459.93	132.85 (-71.12%)	3.303 (-99.28%)

Table 3: A comparison of median running times (percent change relative to the runtime of the PGD algorithm) for different algorithms and scales for the BDS model. For all maximum population sizes, our proposed framework is faster in recovering the transition probabilities than the PGD algorithm for errors at least as good as PGD.

C DETAILS ON GENERATING FUNCTIONS AND PGD ALGORITHM

C.1 Pseudocode for PGD

Algorithm 2 summarizes the proximal gradient descent approach proposed in Xu and Minin [2015].

C.2 Derivation of the PGF for HSC Model

The details for deriving the PGF are included here only for completeness but are standard, following the "random variable technique" of [Bailey, 1991]. Given a two-type branching process with instantaneous rates $a_i(k,l)$, we can define the pseudo-generating function for $i = 1, 2$,

$$u_i(s_1, s_2) = \sum_k \sum_l a_i(k,l) s_1^k s_2^l.$$

Algorithm 2: PGD Algorithm

function PGD $(B, \beta, \tau, \gamma, \epsilon_{abs}, \epsilon_{rel}, N_{iter}, P)$;
Input : initial sizes $X_1 = j, X_2 = k$, time interval t, branching rates θ, signal size $N > j, k$, measurement size M, penalization constant $\lambda > 0$, line-search parameters L, c.
Output : A real-valued 2D matrix $\hat{\boldsymbol{S}}$
Uniformly sample M indices $\mathcal{J} \subset [0, \dots N]$
Compute $\mathbf{B} = \{\phi_{j,k}(t, e^{2\pi i u/N}, e^{2\pi i v/N})\}_{u,v \in \mathcal{J}}$
Define $\mathbf{A}$= $\psi_{\mathcal{J}}$, the $\mathcal{J}$ rows of the IDFT matrix ψ.
Initialize: $\boldsymbol{S_1 = Y_1 = 0}$
for $k = 1, 2, \dots$ **do**
  Choose L_k = line-search(L, c, Y_k)
  Update extrapolation parameter: $\omega_k = \frac{k}{k+3}$
  Update momentum: $\boldsymbol{Y_{k+1} = S_k + \omega_k(S_k - S_{k-1})}$
  Update: $S_{k+1} = \text{SoftThresh}(S_k - L_k)$
end
return $\hat{S} = S_{k+1}$

The probability generating function can be expanded as

$$\begin{aligned}
\phi_{10}(t, s_1, s_2) &= \mathbb{E}(s_1^{X_1(t)} s_2^{X_2(t)} | X_1(0) = 1, X_2(0) = 0) \\
&= \sum_k \sum_l p_{(1,0),(k,l)}(t) s_1^k s_2^l \\
&= \sum_k \sum_l (\delta_{k=1,l=0} + a_1(k,l)t + o(t)) s_1^k s_2^l \\
&= s_1 + u_1(s_1, s_2)t + o(t).
\end{aligned}$$

Similarly, starting with one particle of type 2 instead of type 1, we can obtain an analogous expression for $\phi_{01}(t, s_1, s_2)$. For brevity, we will write $\phi_{10}(t, s_1, s_2) := \phi_1(t, s_1, s_2), \phi_{01}(t, s_1, s_2) := \phi_2(t, s_1, s_2)$.

Differentiating $\phi_1(t, s_1, s_2)$ and $\phi_2(t, s_1, s_2)$ with respect to time, we get

$$\begin{aligned}
\frac{d\phi_1}{dt}(t, s_1, s_2)|_{t=0} &= u_1(s_1, s_2) \\
\frac{d\phi_2}{dt}(t, s_1, s_2)|_{t=0} &= u_2(s_1, s_2).
\end{aligned}$$

We now derive the backward and forward equations with the Chapman-Kolmogorov equations, yielding the following symmetric relations:

$$\phi_1(t + h, s_1, s_2) = \phi_1(t, \phi_1(h, s_1, s_2), \phi_2(h, s_1, s_2)) \tag{S16}$$
$$= \phi_1(h, \phi_1(t, s_1, s_2), \phi_2(t, s_1, s_2)). \tag{S17}$$

We expand around t and apply Equation S16 to derive the backward equations:

$$\begin{aligned}
\phi_1(t + h, s_1, s_2) &= \phi_1(t, s_1, s_2) + \frac{d\phi_1}{dh}(t + h, s_1, s_2)|_{h=0} h + o(h) \\
&= \phi_1(t, s_1, s_2) + \frac{d\phi_1}{dh}(h, \phi_1(t, s_1, s_2), \phi_2(t, s_1, s_2))|_{h=0} h + o(h) \\
&= \phi_1(t, s_1, s_2) + u_1(\phi_1(t, s_1, s_2), \phi_2(t, s_1, s_2)) + o(h).
\end{aligned}$$

We can apply an analogous argument for $\phi_2(t, s_1, s_2)$ to arrive at the following system of ODEs

$$\frac{d\phi_1}{dt}(t, s_1, s_2) = u_1(\phi_1(t, s_1, s_2) + \phi_2(t, s_1, s_2)),$$
$$\frac{d\phi_2}{dt}(t, s_1, s_2) = u_2(\phi_1(t, s_1, s_2) + \phi_2(t, s_1, s_2)),$$

with initial conditions $\phi_1(0, s_1, s_2) = s_1$, $\phi_2(0, s_1, s_2) = s_2$.

Recall from the main text the rates that define the two-component HSC model are given by:

$$a_1(2,0) = \rho, \quad a_1(0,1) = \nu, \quad a_1(1,0) = -(\rho + \nu), \quad a_2(0,0) = \mu, \quad a_2(0,1) = -\mu.$$

Thus, the pseudo-generating functions become

$$u_1(s_1, s_2) = \rho s_1^2 + \nu s_2 - (\rho + \nu)s_1,$$
$$u_2(s_1, s_2) = \mu - \mu s_2 = \mu(1 - s_2).$$

Plugging these into the backward equations yields,

$$\frac{d\phi_1}{dt}(t, s_1, s_2) = \rho\phi_1^2(t, s_1, s_2) + \nu\phi_2(t, s_1, s_2) - (\rho + \nu)\phi_1(t, s_1, s_2),$$
$$\frac{d\phi_2}{dt}(t, s_1, s_2) = \mu(1 - \phi_2(t, s_1, s_2)).$$

Note that the differential equation for $\phi_2(t, s_1, s_2)$ corresponds to that of a pure death process and has a closed-form solution.

$$\frac{d\phi_2}{dt}(t, s_1, s_2) = \mu(1 - \phi_2(t, s_1, s_2))$$
$$\frac{d}{dt}\left(\frac{\phi_2}{1 - \phi_2}\right)(t, s_1, s_2) = \mu$$
$$\phi_2(t, s_1, s_2) = 1 - e^{-\mu t + C}$$

Inserting $\phi_2(0, s_1, s_2) = s_2$, we obtain $C = ln(1 - s_2)$, and we get

$$\phi_2(t, s_1, s_2) = 1 + (s_2 - 1)e^{-\mu t}. \tag{S18}$$

When inserting equation S18 into the backward equation involving $\phi_1(t, s_1, s_2)$, we obtain

$$\frac{d\phi_1}{dt}(t, s_1, s_2) = \rho\phi_1^2(t, s_1, s_2) - (\rho + \nu)\phi_1(t, s_1, s_2) + \nu(1 + (s_2 - 1)e^{-\mu t}) \tag{S19}$$

Given the rates of the process and the values of the three arguments, Equation S19 can be solved numerically, allowing the computation of $\phi_{i,j}(t, s_1, s_2) = \phi_1^i(t, s_1, s_2)\phi_2^j(t, s_1, s_2)$ which holds by particle independence.