TNO innovation for life

Bayesian Parameter Inference for Industrial Printer Models

Paving the way for probabilistic nozzle diagnostics

TNO Public › TNO 2025 S11563
30 juli 2025

ICT, Strategy & Policy
www.tno.nl
+31 88 866 50 00
info@tno.nl

TNO 2025 S11563 – 30 juli 2025

Bayesian Parameter Inference for Industrial Printer Models

Paving the way for probabilistic nozzle diagnostics

Auteurs	Robert Jan Speksnijder
Rubricering rapport	TNO Public
Titel	TNO Public
Rapporttekst	TNO Public
Aantal pagina's	56 (excl. voor- en achterblad)
Aantal bijlagen	0
Opdrachtgever	Alvaro Piedrafita

Bayesian Parameter Inference for Industrial Printer Models

Paving the Way for Probabilistic Nozzle Diagnostics

by

Robert Jan Speksnijder

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Thursday July 3, 2025 at 11:00 AM.

Student number:	5410258	
Project duration:	October 31, 2024 – July 3, 2025	
Thesis committee:	Dr. ir. G. N. J. C. Bierkens,	TU Delft, responsible supervisor
	Prof. dr. ir. M. Verlaan,	TU Delft
	Dr. Á. Piedrafita Postigo,	TNO-ESI, external supervisor

An electronic version of this thesis is available at `https://repository.tudelft.nl/`. Code will be made available via `https://repository.tno.nl/`.

TU Delft

TNO

Acknowledgements

I would like to express my sincere thanks to my supervisors, Joris Bierkens and Álvaro Piedrafita Postigo, for their guidance and advice throughout this project. Our regular meetings were invaluable in shaping the work. My thanks also go to Martin Verlaan for serving on my thesis committee, to my parents for their support throughout my studies, and to TNO for providing the opportunity and resources to carry out this research.

In writing this thesis, Microsoft Copilot was applied to passages judged to need better flow and clarity; its suggestions were reviewed and incorporated where helpful. Copilot was also used to assist with debugging code.

Abstract

This thesis considers extensions of the standard independent hidden Markov model approach previously used by TNO for modelling printer nozzles. These extensions introduce parametrised transition probabilities and incorporate interactions between neighbouring nozzles to better capture the real-world printing process. The aim of this thesis is to investigate Bayesian methods to infer their model parameters. Although not the goal of this thesis, accurate parameter estimation paves the way for diagnosing nozzle malfunctions, ultimately improving printer performance.

The first method is a sampling algorithm with adaptive proposal distributions. We then present two variational inference (VI) algorithms, which aim to minimise the divergence between an approximate and true posterior of the unknown parameters. The first VI method makes use of model-specific approximations, while the latter is more flexible. The adaptive sampler, however, remains the most general of the three algorithms in addition to being asymptotically unbiased.

On synthetic data, all methods were capable of producing good estimates of the model parameters. When a good initialisation is available, the sampling method may be faster than the VI approaches; however, if no good start is known, the other methods may be preferred. The runtimes of the algorithms appeared to grow linearly with the number of nozzles. In addition, a linear scaling with the number of time steps appeared plausible.

While the methods performed well on the toy models studied here, their efficacy must be confirmed on more complex systems and demonstrated in real-world applications. If runtimes prove prohibitive, sub-sampling approaches or stochastic variational inference methods could be investigated.

Contents

1

Introduction

Large-scale industrial machines, such as high-speed printers, operate under demanding conditions where subtle, unobservable malfunctions may occur. In these printers, the faults are nozzle-specific and become evident as misprints at their respective locations on test sheets. Revealing these latent issues requires a probabilistic model that links each nozzle's unobservable state to its measurable output. Such models are defined by state-transition probabilities governing the evolution of the hidden states—which emit observed outcomes—and inferring these probabilities from the observed data is the main focus of this thesis.

At TNO, researchers focus on the diagnostic challenge of high-speed inkjet printers. These printers pull in paper and draw them through four colour-specific printhead arrays, see figure 1.1. Each printhead contains thousands of nozzles arranged on a two-dimensional grid: when projected onto the vertical axis (perpendicular to the paper flow) they are evenly spaced, while a staggered horizontal layout prevents interference. Nozzles can fail for various reasons, but this thesis focuses on two primary mechanisms. First, dried ink can accumulate in a nozzle, causing a blockage that may spread laterally to neighbouring nozzles. Second, dust drawn in with the paper can obstruct nozzles, with those closest to the paper entrance being most at risk. Nozzle functionality is assessed via regular test prints that are scanned at high resolution to confirm whether each nozzle printed. This thesis considers the nozzles of a single printhead of one colour only, since faults occur independently across the different printheads.

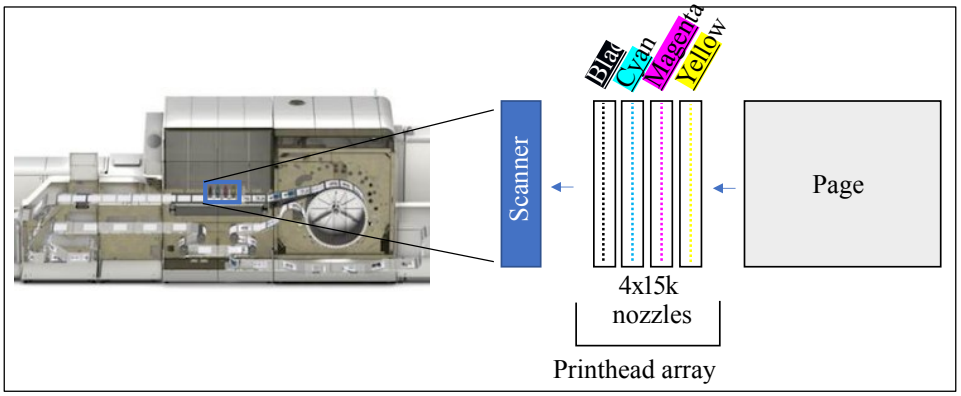

Figure 1.1: Overview of a high-speed industrial printer. Sheets of paper are pulled into the machine and led through four colour-specific arrays of printheads. Every printhead consists of thousands of nozzles arranged on a two-dimensional grid: uniformly spaced vertically and horizontally staggered to prevent interference between adjacent nozzles. Test prints are made regularly and scanned at high resolution to assess the functionality of every nozzle. This figure was provided by TNO.

Diagnosing nozzle faults requires both a probabilistic model of the printing process and an inference algorithm to recover hidden states. Previously, TNO modelled each nozzle as an independent hidden Markov model (HMM), comprising a sequence of latent states (nozzle conditions) and matching observations (print outcomes). In this framework, the state at time t evolves solely from the state at $t-1$ via the transition probabilities, and each latent state generates an observation. Because the transition probabilities fully characterise the temporal dynamics,

estimating these probabilities is a prerequisite for any subsequent diagnostic inference. TNO did so using a variational message passing algorithm [5][p. 491].

The variational message passing scheme used by TNO is an example of a Bayesian method, which bring several benefits over classical techniques such as expectation maximisation (EM), which maximises the likelihood [5][see sec. 9.4]. Firstly, Bayesian methods allow the incorporation of prior knowledge: for example, expert beliefs about the probability of specific nozzle faults can be encoded directly. Secondly, they deliver natural uncertainty quantification by producing posterior distributions over the parameters instead of single-point estimates. Thirdly, while the variational message passing algorithm exploits particular conjugate-exponential relationships [5][see p. 492], Bayesian frameworks, in general, are more flexible than classical methods and readily extend to a wider variety of models and dependency structures.

Unfortunately, standard independent HMMs omit key aspects of the real printing process. By parametrising the transition probabilities with a small set of interpretable parameters, we can embed expert knowledge about how nozzle failures develop while reducing the number of free parameters. We also model spatial interactions, such as ink spreading, between neighbouring nozzles instead of assuming they operate independently. However, combining low-dimensional parametrisations with neighbour-dependent transitions breaks the conjugate–exponential structure that variational message passing relies on, making it infeasible for our extended models [5][p. 492].

To address these shortcomings, this thesis develops Bayesian inference methods for two enhanced models. The first model retains independent HMMs but parametrises the transition probabilities with a small number of parameters. The second model, in addition to a low-dimensional parametrisation, adds spatial interactions, letting each nozzle's transition rates depend on the states of its neighbours. Considering these models separately lets us assess the impact of the extra complexity. Despite the potentially large number of nozzles, this low-dimensional parametrisation renders transition probability estimation tractable. We apply two inference frameworks to one or both of the models:

- Markov chain Monte Carlo (MCMC), generally applicable but sensitive to initialisation;
- Variational inference (VI), less sensitive to initialisation but less flexible.

Additionally, we consider two unparametrised models with the same independent HMM or coupled neighbour structure as above, serving as fallback alternatives.

All experiments in this thesis use generated data only. We emphasise that both main models in this thesis serve as simple toy examples; in practice, one may readily extend them to capture richer dynamics and more complex interactions.

The remainder of the thesis is organised as follows. In chapter 2, we review mathematical preliminaries—HMMs, Bayesian inference, MCMC and VI. In chapter 3, we formalise the printer-inspired models and associated inference problems. Chapter 4 details the development of MCMC and VI algorithms for parameter estimation. Finally, chapter 5 evaluates these methods empirically, comparing accuracy and convergence speed.

2

Preliminary Mathematics

In this thesis, we investigate Bayesian methods for inferring parameters of particular parametrised models representing the printing process of industrial printers. In the next chapter, we introduce the two models that will be studied in this thesis. To support these models, this chapter begins with an introduction to the hidden Markov model (HMM), a key building block for both models. We then provide an overview of Bayesian inference—a statistical framework that combines prior information with observed data to estimate unknown model parameters. Finally, we present the two primary techniques for parameter estimation in Bayesian statistics: Markov chain Monte Carlo (MCMC) and variational inference (VI) [8, p. 1306].

2.1 Hidden Markov Models

The following section is largely based on chapter 13 from [5].

Suppose we want to model a sequence of discrete observations $(x_1, \ldots, x_T)$ as random variables, where every variable x_t depends on past observations $x_1, \ldots, x_{t-1}$. We assume that for every $t \in \{1, \ldots, T\}$, x_t takes values in $\{0, \ldots, D_x - 1\}$. For example, $(x_1, \ldots, x_T)$ could represent the print observations from an industrial printer. One of the simplest probabilistic models to describe such a sequence is a Markov chain of random variables.

In a Markov chain, $p(x_t|x_{t-1}, \ldots, x_1) = p(x_t|x_{t-1})$. That is, given x_{t-1}, every random variable x_t is independent of all prior random variables. Now, one can always express the joint probability of $(x_1, \ldots, x_T)$ as

$$p(x_1, \ldots, x_T) = p(x_1) \prod_{t=2}^{T} p(x_t|x_{t-1}, \ldots, x_1), \tag{2.1}$$

which follows directly from the product rule of probability [5, p. 14]. If we combine this expression with the fact that $p(x_t|x_{t-1}, \ldots, x_1) = p(x_t|x_{t-1})$ for a Markov chain, we obtain

$$p(x_1, \ldots, x_T) = p(x_1) \prod_{t=2}^{T} p(x_t|x_{t-1}). \tag{2.2}$$

That is, for a given initial value x_1, the model is defined by the transition probabilities $p(x_t|x_{t-1})$. If we assume that these transition probabilities are independent of t, this model is known as a *homogeneous* Markov chain. We then write $\mathcal{T}$ for the time-independent transition matrix, where entry ij of $\mathcal{T}$ equals $\mathcal{T}_{ij} = p(x_t = j|x_{t-1} = i)$. By learning only the D_x^2 entries of $\mathcal{T}$, the model is effectively characterised. A homogeneous Markov chain therefore provides a simple framework, defined by a small number of parameters, to describe sequential observations.

The structure of a Markov chain can also be captured through a directed graphical model, see figure 2.1. Directed graphical models can additionally reveal conditional independence between sets of random variables through the concept of d-separation. See appendix A for an overview of directed graphical models and d-separation.

Although this model is simple, it fails to capture the latent dynamics inherent in many real-world processes. In numerous physical systems, evolution is driven by unobserved (hidden or latent) states that generate the observable outcomes. For instance, in DNA sequencing, the observable nucleotide bases (A, C, G, T) are produced by

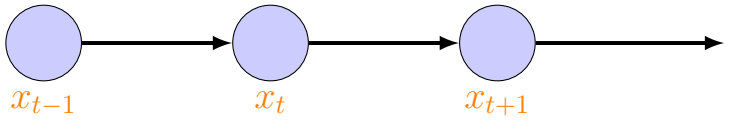

Figure 2.1: A directed graphical model illustrating a Markov chain over time. Each node represents an observation (e.g. x_{t-1}, x_t, and x_{t+1}), and the directed edges indicate that each observation given all prior observations is conditionally dependent only on its immediate predecessor. This figure visualises the temporal factorisation of the joint probability, as described in equation (2.2).

underlying hidden states [1, chapter 7]; similarly, in printing technology, the final output of print nozzles reflects latent operational factors. Because the observations do not correspond one-to-one with the underlying physical states, it is essential to explicitly model both latent and observed states to accurately capture the system dynamics. Hidden Markov models (HMM) exactly achieve this objective.

In HMMs, each observed variable x_t is paired with a latent variable z_t. We assume that for every $t \in \{1, \dots, T\}$, z_t is discrete and takes values in $\{0, \dots, D_z - 1\}$. In the case of an industrial printer, z_t represents a nozzle state at time t, while x_t represents the corresponding outcome of the test print. We then let the latent variables $(z_1, \dots, z_T)$ form a Markov chain and let each observation x_t solely depend on its corresponding latent variable z_t. That is, for every $t \in \{1, \dots, T\}$, x_t given z_t is independent of all other random variables. Hence, $p(z_t|z_{t-1}, \dots, z_1) = p(z_t|z_{t-1})$ and $p(x_1, \dots, x_T|z_1, \dots, z_T) = \prod_{t=1}^{T} p(x_t|z_t)$. It follows that the joint probability factors into

$$\begin{aligned} p(z_1, \dots, z_T, x_1, \dots, x_T) &= p(z_1, \dots, z_T) p(x_1, \dots, x_T|z_1, \dots, z_T) \\ &= p(z_1) \prod_{t=2}^{T} p(z_t|z_{t-1}) \prod_{t=1}^{T} p(x_t|z_t). \end{aligned} \tag{2.3}$$

We see that for a given initial latent z_1, the model is entirely characterised by the transition probabilities $p(z_t|z_{t-1})$ and the emission probabilities $p(x_t|z_t)$. We again assume that $p(z_t|z_{t-1})$ and $p(x_t|z_t)$ are independent of t. As a result, we have that the model is characterised by only a small number of parameters. This type of model is known as a hidden Markov model (HMM).

The corresponding graphical model for HMMs is given in figure 2.2. One may observe that applying the rules of d-separation to the graphical model does not give us that x_t given x_{t-1} is independent of prior observations. This implies that, in contrast to the Markov chain, $p(x_t|x_{t-1}, \dots, x_1)$ may depend on all previous observations. Therefore, the hidden Markov model gives us more sophisticated relations between $x_t, x_{t-1}, \dots, x_1$, while preserving simplicity in the model characterisation as in the homogeneous Markov chain.

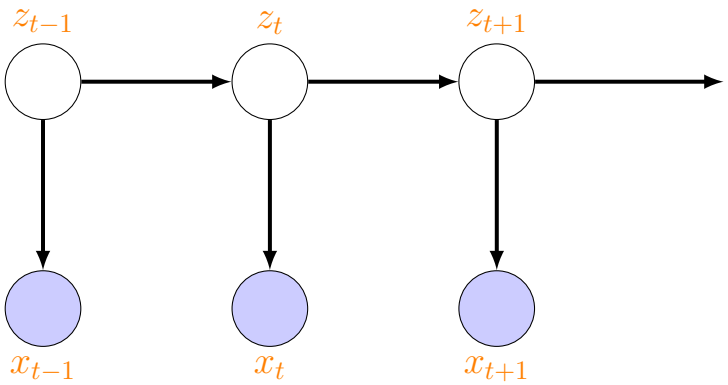

Figure 2.2: A directed graphical model depicting a Hidden Markov Model (HMM). In this diagram, the latent variables z_{t-1}, z_t, and z_{t+1} form a Markov chain, and each latent variable emits an observation (i.e. x_{t-1}, x_t, and x_{t+1}) that depends solely on its corresponding state. This representation highlights the temporal and conditional independence structure inherent in HMMs, as formulated in equation (2.3).

2.2 Bayesian Inference

Consider a probabilistic model comprising observed random variables $\{x_t\}$, latent random variables $\{z_t\}$, and model parameters θ. The joint probability of the model is defined as $p(\{x_t\}, \{z_t\}|\theta)$. When the model parameters θ are unknown, the goal is to infer them using the observed data $\{x_t\}$.

In Bayesian inference, this problem is approached by treating the model parameters θ as random variables and imposing a prior distribution $p(\theta)$ on them. The prior distribution should reflect the information known a priori about the parameters, but in practice, it is often selected based on practical considerations. After specification of the prior, the posterior distribution $p(\theta, \{z_t\}|\{x_t\})$ is given by the following relation

$$p(\theta, \{z_t\}|\{x_t\}) \propto p(\{x_t\}, \{z_t\}|\theta)p(\theta), \tag{2.4}$$

which follows from Bayes' theorem

$$p(B|A) = \frac{p(A|B)p(B)}{p(A)}. \tag{2.5}$$

[6, section 13.1.1]

Using this relation, the posterior—and consequently an estimate of θ—can, in principle, be obtained by normalising the right-hand side of equation (2.4). In practice, however, the normalising constant is often intractable to compute. Thus, we require approximate methods that leverage equation (2.4) to estimate θ from the observed data $\{x_t\}$. In the following two sections, we present two classes of algorithms that fulfil this need.

2.3 Markov Chain Monte Carlo

In this section, we discuss three sampling algorithms that will be useful for solving the parameter estimation problems treated in this thesis. These algorithms belong to a broader family of techniques known as Markov chain Monte Carlo (MCMC), which are used to generate samples from probability distributions. As the name implies, MCMC algorithms generate a Markov chain of samples that, after sufficient iterations, converge to the target distribution.

2.3.1 The Metropolis-Hastings Algorithm

Consider a target distribution $p(z)$. Suppose that one is only able to evaluate the target distribution up to a multiplicative constant $\tilde{p}(z)/Z_p = p(z)$. The Metropolis-Hastings algorithm overcomes this limitation to approximate $p(z)$.

The algorithm draws samples z^τ sequentially. In the first step, a sample z^1 is drawn according to an initial distribution. In step $\tau + 1$, if z^τ is the current sample, a new sample $z^\star$ is drawn from a proposal distribution $q(z|z^\tau)$ chosen by the user. This new sample is then accepted with probability

$$A = \min\left(1, \frac{\tilde{p}(z^\star)q(z^\tau|z^\star)}{\tilde{p}(z^\tau)q(z^\star|z^\tau)}\right). \tag{2.6}$$

If the new sample is accepted, $z^{\tau+1} = z^\star$, otherwise $z^{\tau+1} = z^\tau$. [5]

In [13, section 7.3.2], it is shown that the distribution of the samples generated by the Metropolis-Hastings algorithm converges to the target distribution under certain conditions. This holds regardless of the initial distribution. We describe this more formally. Let $K^n(z, \cdot)$, denote the probability measure of the sample generated by the Metropolis-Hastings Markov chain after n transitions, when starting from initial state $z^1 = z$. Let $P(\cdot)$ denote the probability measure of the target distribution and $\mu(\cdot)$ the probability measure of the initial distribution. Suppose the Metropolis-Hastings chain satisfies the following two conditions. (i) $q(z|z') > 0$ for all z, z' in z-space. (ii) The probability of $\{z^{\tau+1} = z^\tau\}$ is greater than zero. Then the Metropolis-Hastings algorithm converges in the following sense

$$\lim_{n\to\infty} \sup_{A\in\mathcal{F}} \left| \int K^n(z, A)\mu(dz) - P(A) \right| = 0, \tag{2.7}$$

where A are elements of the event space. Importantly, the above two conditions are sufficient, but not necessary. See [13, chapters 6 and 7] and [15] for other and more general conditions.

The above convergence implies that if τ is large enough, z^τ is drawn approximately from the target distribution. Note, however, that consecutive samples, e.g. $\{z^{\tau-1}, z^\tau, z^{\tau+1}\}$, are not independent draws from the target distribution; many additional steps are required to obtain a sample that is independent of z^τ.

2.3.2 Gibbs Sampling

A second MCMC algorithm is the Gibbs sampler. When z is multi-dimensional, the Gibbs sampler provides a coordinate-wise alternative to the Metropolis-Hastings algorithm by iteratively sampling each component from its full conditional $p(z_i|z_{\setminus i})$.

Suppose one wants to sample $z = (z_1, \ldots, z_m)$ from target distribution $p(z)$. As before, we start from a sample z^1 drawn from an initial distribution. If z^τ is the sample resulting from step τ, we go through the following sweep to generate $z^{\tau+1}$.

$$\begin{aligned}
&- \text{ Sample } z_1^{\tau+1} \sim p(z_1|z_2^\tau, z_3^\tau, \ldots, z_m^\tau).\\
&- \text{ Sample } z_2^{\tau+1} \sim p(z_2|z_1^{\tau+1}, z_3^\tau, \ldots, z_m^\tau).\\
&\quad\vdots\\
&- \text{ Sample } z_i^{\tau+1} \sim p(z_i|z_1^{\tau+1}, \ldots, z_{i-1}^{\tau+1}, z_{i+1}^\tau, \ldots, z_m^\tau).\\
&\quad\vdots\\
&- \text{ Sample } z_m^{\tau+1} \sim p(z_m|z_1^{\tau+1}, \ldots, z_{m-1}^{\tau+1}).
\end{aligned} \tag{2.8}$$

That is, we generate $z_i^{\tau+1}$ by conditioning on all updated random variables $z_1^{\tau+1}, \ldots, z_{i-1}^{\tau+1}$ and all yet to be updated random variables $z_{i+1}^\tau, \ldots z_m^\tau$. One may see this algorithm as a special case of Metropolis-Hastings, where the proposal distribution $q_k(z|z')$ for the kth component equals the conditional distribution $p(z_k|z'_{\setminus k})$. [5]

In [13, section 10.2.1], it is shown that the Gibbs sampler converges in the sense of equation (2.7) under certain conditions. We again consider the probability measure $K^n(z, \cdot)$ of the distribution of the sample generated by the chain after n Gibbs sweep, starting from z. The conjunction of the following two conditions is sufficient to establish convergence. (i) Probability measure $K^1(z, \cdot)$ can be described by a probability density. (ii) The probability of $\{z^{\tau+1} = z^\tau\}$ is greater than zero. As before, it is important to stress that these conditions are sufficient, but not necessary; see [13, chapters 6 and 10] and [15] for more details.

As for the Metropolis-Hastings algorithm, the convergence of the Gibbs sampler implies that for τ large, z^τ is drawn from the target distribution. As before, independent samples from the chain are obtained only when they are separated by a large number of intermediate steps.

2.3.3 Metropolis-within-Gibbs

When z is multi-dimensional, the Gibbs sampler provides a one-dimensional alternative to the standard Metropolis-Hastings algorithm by updating every component individually. If it is infeasible to sample from the conditional distributions, however, one can replace each step in the Gibbs sweep by one or more steps from the standard Metropolis-Hastings algorithm. We call the resulting algorithm a Metropolis-within-Gibbs algorithm. The algorithm has the following form.

1. Initialise $z^1 = (z_1^1, \ldots, z_m^1)$
2. For $\tau = 1, \ldots, N_{\text{MwG}}$:

   For $i = 1, \ldots, m$:

   $$\begin{aligned}
   &- \text{ Sample } z_i^\star \sim q_i(z_i|z_i^\tau).\\
   &- \text{ Accept with probability } A = \min\left(1, \frac{\tilde{p}(z_i^\star|z_1^{\tau+1}, \ldots, z_{i-1}^{\tau+1}, z_{i+1}^\tau, \ldots, z_m^\tau) q_i(z_i^\tau|z_i^\star)}{\tilde{p}(z_i^\tau|z_1^{\tau+1}, \ldots, z_{i-1}^{\tau+1}, z_{i+1}^\tau, \ldots, z_m^\tau) q_i(z_i^\star|z_i^\tau)}\right).\\
   &- \text{ If accepted } z_i^{\tau+1} = z_i^\star, \text{ otherwise } z_i^{\tau+1} = z_i^\star.
   \end{aligned} \tag{2.9}$$

Here $q_i(z_i|z_i^\tau)$ is the proposal distribution for component i of z. $\tilde{p}(z_i|z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m)$ is proportional to $p(z_i|z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m)$. [15, p. 2128]

To achieve convergence to the target distribution in the sense of equation (2.7), the chain needs to satisfy several conditions, see [13, chapter 6] [15]. These conditions are slightly more sophisticated than those of the previous algorithm.

2.4 Variational Inference

Consider again a Bayesian model with observed random variables $\{x_t\}$, latent random variables $\{z_t\}$ and model parameters θ, equipped with a prior distribution $p(\theta)$. As mentioned in section 2.2, the posterior distribution $p(\theta, \{z_t\}|\{x_t\})$ is often intractable to compute. As described in [8, section 2.2] and [5], variational inference (VI) provides a framework for approximating the posterior distribution by introducing a variational distribution $q(\theta, \{z_t\})$ and reformulating the inference task as an optimisation problem. In this framework, the variational distribution is selected from a predefined family of distributions, and the goal is to identify the member that best approximates the true posterior. The choice of distribution is made by optimising an objective function.

In variational inference, one aims to minimise the Kullback-Leibler (KL) divergence of $q(\theta, \{z_t\})$ from $p(\theta, \{z_t\}|\{x_t\})$,

$$\mathrm{KL}(q(\theta, \{z_t\})||p(\theta, \{z_t\}|\{x_t\})) = \mathbb{E}_q[\log(q(\theta, \{z_t\}))] - \mathbb{E}_q[\log(p(\theta, \{z_t\}|\{x_t\}))], \tag{2.10}$$

by choosing the optimal $q(\theta, \{z_t\})$ from a predefined family of distributions [5]. Although not a formal mathematical distance, the Kullback-Leibler divergence quantifies the dissimilarity between $q(\theta, \{z_t\})$ and $p(\theta, \{z_t\}|\{x_t\})$, as stated in [11]. In particular, $\mathrm{KL}(q||p) \geq 0$ and $\mathrm{KL}(q||p) = 0 \iff q = p$. That is, the optimal variational distribution equals the posterior, if the posterior is a member of the predefined family of distributions.

Equivalently, variational inference can be formulated as the maximisation of the evidence lower bound (ELBO). As the name suggests, the ELBO gives a lower bound on the logarithm of the probability of the observations. I.e.,

$$\begin{aligned}
\log(p(\{x_t\})) &= \log\left(\int p(\theta, \{z_t\}, \{x_t\}) dz d\theta\right) \\
&= \log\left(\int p(\theta, \{z_t\}, \{x_t\}) \frac{q(\theta, \{z_t\})}{q(\theta, \{z_t\})} dz d\theta\right) \\
&= \log\left(\mathbb{E}_q\left[\frac{p(\theta, \{z_t\}, \{x_t\})}{q(\theta, \{z_t\})}\right]\right) \\
&\geq \mathbb{E}_q[\log(p(\theta, \{z_t\}, \{x_t\}))] - \mathbb{E}_q[\log(q(\theta, \{z_t\}))] \\
&=: \mathcal{L}(q),
\end{aligned} \tag{2.11}$$

where we used Jensen's inequality in step 4. The ELBO is related to the Kullback-Leibler divergence in the following way:

$$\begin{aligned}
\mathrm{KL}(q(\theta, \{z_t\})||p(\theta, \{z_t\}|\{x_t\})) &= \mathbb{E}_q[\log(q(\theta, \{z_t\}))] - \mathbb{E}_q[\log(p(\theta, \{z_t\}|\{x_t\}))] \\
&= \mathbb{E}_q[\log(q(\theta, \{z_t\}))] - \mathbb{E}_q[\log(p(\theta, \{z_t\}, \{x_t\}))] + \log(p(\{x_t\}) \\
&= -\mathcal{L}(q) + \text{constant}.
\end{aligned} \tag{2.12}$$

That is, the ELBO is equal to minus the KL-divergence up to an additive constant independent of q. Hence, maximising the ELBO with respect to q, results in minimising the Kullback-Leibler divergence. [8, section 2.2]

To render a variational inference optimisation problem tractable, we typically restrict the family of distributions that q may represent. One can, for example, restrict q to be a member of the multivariate normal distribution. In this case, we maximise the ELBO with respect to the variational parameters μ and Σ. [5]

Another assumption we may impose on the variational distribution is that it factorises. This assumption is called the mean-field approximation. One may, for example, assume that $q(\theta, \{z_t\}) = q(\theta)q(\{z_t\})$. We then optimise the ELBO with respect to each factor in turn. For this purpose, we need to consider the ELBO for every factor with all other factors treated as constants. E.g., for the factor $q(\{z_t\})$, we assume $q(\theta)$ to be fixed and write

$$\begin{aligned}
\mathcal{L}(q(\{z_t\})) &= \mathbb{E}_{q(\{z_t\})}[\mathbb{E}_{q(\theta)}[\log(p(\theta, \{z_t\}, \{x_t\}))]] - \mathbb{E}_{q(\theta)}[\log(q(\theta))] - \mathbb{E}_{q(\{z_t\})}[\log(q(\{z_t\}))] \\
&= \mathbb{E}_{q(\{z_t\})}[\mathbb{E}_{q(\theta)}[\log(p(\theta, \{z_t\}, \{x_t\}))]] - \mathbb{E}_{q(\{z_t\})}[\log(q(\{z_t\}))] + \text{constant},
\end{aligned} \tag{2.13}$$

where the constant does not depend on $q(\{z_t\})$. If we define $\log(\tilde{p}(\{z_t\}, \{x_t\})) = \mathbb{E}_{q(\theta)}[\log(p(\theta, \{z_t\}, \{x_t\}))] + \text{constant}$, we obtain

$$\mathcal{L}(q(\{z_t\})) = \mathbb{E}_{q(\{z_t\})}[\log(\tilde{p}(\{z_t\}, \{x_t\}))] - \mathbb{E}_{q(\{z_t\})}[\log(q(\{z_t\}))] + \text{constant}. \tag{2.14}$$

One may recognise this expression as minus the KL-divergence of $q(\{z_t\})$ from $\tilde{p}(\{z_t\}, \{x_t\})$ up to an additive constant independent of $q(\{z_t\})$. Therefore, the ELBO is maximised when both distributions are equal, that is,

when

$$\begin{aligned} q(\{z_t\}) = \tilde{p}(\{z_t\}, \{x_t\}) &\propto \exp(\mathbb{E}_{q(\theta)}[\log(p(\theta, \{z_t\}, \{x_t\}))]) \\ &\propto \exp(\mathbb{E}_{q(\theta)}[\log(p(\{z_t\}, \{x_t\}|\theta))]), \end{aligned} \tag{2.15}$$

if there are no further restrictions on $q(\{z_t\})$. [5]

When the likelihood $p(\{z_t\}, \{x_t\}|\theta)$ has a convenient form, equation (2.15) allows us to compute $q(\{z_t\})$ [5]. For example, if the likelihood factorises into local terms–each depending on only a small subset of the latent variables $\{z_t\}$–then the right-hand side of equation (2.15) inherits this factorisation. In such cases, message passing algorithms can be used to compute the (joint) marginals of $q(\{z_t\})$ [7, Section 2.2].

Although one can derive an analogous optimality condition for $q(\theta)$, the resulting update may be intractable. In that case, we may restrict $q(\theta)$ to a specific parametric family. We write $q(\theta|\lambda)$, where λ parametrises the variational distribution. We then must optimise

$$\begin{aligned} \mathcal{L}(\lambda) &= \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}[\log(p(\theta, \{z_t\}, \{x_t\}))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta|\lambda))] \\ &= \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}[\log(p(\{z_t\}, \{x_t\}|\theta)) + \log(p(\theta))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta|\lambda))] \\ &= \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}[\log(p(\{z_t\}, \{x_t\}|\theta))] + \mathbb{E}_{q(\theta|\lambda)}[\log(p(\theta))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta|\lambda))], \end{aligned} \tag{2.16}$$

with respect to the variational parameters λ. Gradient based optimisation routines can be used to find the optimal λ [10, section 2.6].

3

Model Specification and Problem Formulation

In this chapter, we present two toy models of the printing process described in the introduction, along with their corresponding problem formulations. Both models build on the standard independent HMM approach for print nozzles, but introduce one change at a time. In the first, we introduce a low-dimensional parametrisation of the transition probabilities—drawing on our real-world insights—while keeping each nozzle's HMM independent. In the second, we add spatial coupling by linking the parametrised HMMs of neighbouring nozzles. By structuring the work this way, we gain two advantages: the simpler, first model is easier to implement and analyse, and adding coupling in the second lets us isolate and inspect its specific impact. After presenting the two main models, we also briefly discuss a more conventional approach without parametrisations for the transition probabilities.

For both main models, we first provide an in-depth derivation, followed by a compact formal mathematical formulation. In chapter 4, we present methodologies designed to solve the inference problems.

3.1 HMM Model Derivation

In this section, we derive a parametrised model of the printing process without inter-nozzle interactions and present its inference problem formulation. For brevity, we omit the explicit dependence on model parameters θ in the transition probability notation. That is, we write $p(z_t^n|z_{t-1}^n)$ instead of $p(z_t^n|z_{t-1}^n, \theta)$.

3.1.1 Latent and Observation States

We consider the printing process described in the introduction. In this process, printer nozzles can be in different conditions at any given time, which influences their ability to print. Test prints are then used to determine whether a nozzle is functioning properly.

The printer nozzles and their corresponding test-print outcomes can be modelled by hidden Markov models (HMM), see figure 3.1. For every nozzle, we observe at each time step from the test print whether it prints or not. In figure 3.1, this is denoted by x_t^n for nozzle $n \in \{1, \dots, N\}$ at time step $t \in \{1, \dots, T\}$, where N is the number of nozzles and T the number of time steps. If x_t^n equals 0, we say that the observation state is *OK*, which means that the nozzle printed. If x_t^n equals 1, the observation state is *not OK*, which means that the nozzle did not print.

Whether a nozzle prints or does not print at a given time step is determined by the latent (or hidden) state of that nozzle. In our toy model, a nozzle can either function normally, be stuck due to dust, or be stuck due to dried ink. We say that the corresponding nozzle states are *normal*, *dusty*, *dry-in*. When a nozzle state is normal, we observe that the nozzle prints; in the other two cases, we observe that it does not print. Note that this implies that not every latent state is hidden: if the nozzle prints, we know its latent state is normal.

The latent state of nozzle n at time t is given by the random variable z_t^n. Similar to x_t^n, z_t^n takes the values $0, 1, 2$ for the three states *normal*, *dusty*, *dry-in*. We assume $z_1^n = 0$ for all $n \in \{1, \dots, N\}$, i.e., the initial nozzle states are all normal. An overview of the terminology for the states and the random variables x_t^n and z_t^n is provided in

table 3.1.

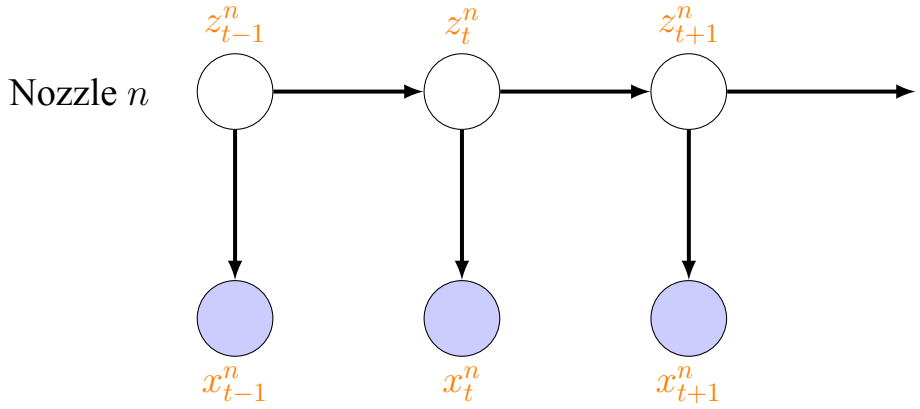

Figure 3.1: Graphical representation of the hidden Markov model for printer nozzles. For each nozzle, the latent states z_t^n (shown at time steps $t-1$, t, and $t+1$) evolve over time, and each latent state emits an observation x_t^n. This schematic illustrates the temporal factorisation of the model, where the evolution of the latent state is governed by a Markov process and each state determines a corresponding observation.

Random Variables	State Numbers	States	Explanation
x_t^n	0	OK	The nozzle prints
	1	not OK	The nozzle does not print
z_t^n	0	normal	The nozzle is functioning normally
	1	dusty	The nozzle is stuck due to dust
	2	dry-in	The nozzle is stuck due to dried ink

Table 3.1: Overview of the random variables and their corresponding states. The table lists the observation variable x_t^n, with states "OK" (indicating a successful print) and "not OK" (indicating a printing failure), and the latent variable z_t^n, with the states "normal", "dusty", and "dry-in". Each entry includes a brief explanation of the meaning behind the values.

3.1.2 Transition and Emission Probabilities

Since the initial nozzle states are known, we only need the transition probabilities $p(z_t^n|z_{t-1}^n)$ and the emission probabilities $p(x_t^n|z_t^n)$ to describe the model completely. We assume independence of t, but not of n. The emission probabilities were already described above and can be given by the emission matrix

$$E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad E_{ij} = p(x_t^n = j | z_t^n = i), \tag{3.1}$$

for all $n \in \{1, \dots, N\}$ and $t \in \{1, \dots, T\}$.

The transition probabilities, on the other hand, are unknown and have to be estimated. Based on the real-world problem, we can write down a parametrised model for the transition matrices, which additionally depend on the nozzle number n. That is, a small number of parameters contained in θ determine the entries of the transition matrices $\mathcal{T}(\theta, n)$, together with n. As mentioned before, it is the goal of our research to find methodologies to estimate these parameters.

We assume that the probability of a transition from nozzle state normal to dusty, $p(z_t^n = 1 | z_{t-1}^n = 0)$, will depend on n. We write

$$p(z_t^n = 1 | z_{t-1}^n = 0) = p_{nd}(n) = A \exp(-\gamma \cdot n), \qquad A \in [0, 1], \gamma \in [0, \infty), \tag{3.2}$$

where p_{nd} stands for the probability of transitioning from the normal to the dusty state. A and γ are two model parameters that have to be estimated.

The form of this probability is motivated by the real-world printing process. When the printer pulls in blank sheets, it also sucks in air. Along with the air, dust is carried in. Over time, some functioning nozzles become

stuck due to the dust, which corresponds to our state transition $(z_t^n = 1, z_{t-1}^n = 0)$. As there is more dust closer to the entrance of the printer, the probability of transitioning from normal to dusty should be larger for nozzles closer to the entrance than for nozzles further away. Here we assume that a larger nozzle index n implies greater distance to the entrance, hence $p_{nd}(n)$ should decrease as n increases. This is a simplification; in practice, p_{nd} would depend on each nozzle's actual spatial coordinates rather than its index. In our toy model, we have chosen an exponential decrease for convenience, however, other relations could be considered, if they make the model more realistic.

We assume that the probability of a transition from the normal state to the dry-in state is independent of n. That is

$$p(z_t^n = 2 | z_{t-1}^n = 0) = p_{di}, \tag{3.3}$$

where di in the model parameter p_{di} stands for dry-in. This gives us, together with equation (3.2), the following expression for the probability of remaining in the normal state:

$$p(z_t^n = 0 | z_{t-1}^n = 0) = 1 - p_{di} - A\exp(-\gamma \cdot n). \tag{3.4}$$

We now consider the transition probabilities from the dusty state $p(z_t^n | z_{t-1}^n = 1)$. We assume that the probability of transitioning to the dry-in state is again equal to p_{di}, as the condition of the nozzle, whether dusty or normal, is expected to have minimal impact on the ink drying process. For the transition from the dusty state to the normal state $(z_t^n = 0, z_{t-1}^n = 1)$, we also assume the probability to be constant. We let it be determined by the model parameter p_{dn}, i.e.,

$$p(z_t^n = 0 | z_{t-1}^n = 1) = p_{dn}. \tag{3.5}$$

It follows that the probability of staying in the dusty state is given by

$$p(z_t^n = 1 | z_{t-1}^n = 1) = 1 - p_{dn} - p_{di}. \tag{3.6}$$

If a nozzle is stuck due to dried ink, the transition probabilities are known, since a nozzle in this condition will remain stuck. Therefore

$$p(z_t^n | z_{t-1}^n = 2) = \begin{cases} 0 & \text{if } z_t^n = 0 \text{ or } z_t^n = 1, \\ 1 & \text{if } z_{t-1}^n = 2. \end{cases} \tag{3.7}$$

In practice, routine cleaning unblocks nozzles clogged by dried ink; this recovery process is not included in our model.

We can now give the full transition matrix:

$$\mathcal{T}(\theta, n) = \begin{pmatrix} 1 - p_{nd}(n) - p_{di} & p_{nd}(n) & p_{di} \\ p_{dn} & 1 - p_{dn} - p_{di} & p_{di} \\ 0 & 0 & 1 \end{pmatrix}, \qquad \text{with } p_{nd}(n) = A\exp(-\gamma \cdot n). \tag{3.8}$$

The model parameters θ are summarised in table 3.2, together with their parameter space. Note that since state changes are rare, we can constrain the relevant parameters in θ to lie within a small range.

Parameters θ	Parameter Space	Prior Distribution
A	$[0, 1/4]$	$4A \sim \text{Beta}(\alpha_{4A}, \beta_{4A})$
γ	$[0, \infty)$	$\gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma)$
p_{di}	$[0, 1/4]$	$4p_{di} \sim \text{Beta}(\alpha_{4pdi}, \beta_{4pdi})$
p_{dn}	$[0, 1/4]$	$4p_{dn} \sim \text{Beta}(\alpha_{4pdn}, \beta_{4pdn})$

Table 3.2: Summary of the model parameters for the HMM-based printing process model. The table details the parameters $\theta = (A, \gamma, p_{di}, p_{dn})$, the associated parameter spaces, and the prior distributions assigned to each parameter.

3.1.3 Prior Distributions

To estimate the model parameters, we put a prior distribution $p(\theta)$ on θ. We would like this prior to factorise, that is, we want $p(\theta) = p(A)p(\gamma)p(p_{di})p(p_{dn})$. However, since the rows of each transition matrix must sum to one, the values for A, p_{di}, p_{dn} cannot be large simultaneously. This implies a dependency between the model

parameters. Fortunately, from the real-world process, it is clear that the values of these parameters will never be larger than 0.25. Therefore, we can assume that A, p_{di}, p_{dn} are independent a-priori and are distributed as follows

$$4A \sim \text{Beta}(\alpha_{4A}, \beta_{4A}), \quad 4p_{di} \sim \text{Beta}(\alpha_{4pdi}, \beta_{4pdi}), \quad 4p_{dn} \sim \text{Beta}(\alpha_{4pdn}, \beta_{4pdn}), \tag{3.9}$$

since a Beta distribution takes values in $(0, 1)$. We also assume that γ is independent of the other parameters and has the following distribution

$$\gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma). \tag{3.10}$$

An overview of the prior distributions put on the parameters is given in table 3.2. It is then the aim of this research to describe methodologies to infer these parameters.

3.2 HMM Model Problem Formulation

Inference Problem Formulation

Consider the random variables representing observations

$$\{x_t^n \in \{0,1\} : \text{for } t = 1, \dots, T,\ n = 1, \dots, N\}$$

and latent states

$$\{z_t^n \in \{0,1,2\} : \text{for } t = 1, \dots, T,\ n = 1, \dots, N\}.$$

We assume that $z_1^n = 0$ for all $n = 1, \dots, N$.

The emission probabilities $p(x_t^n = j | z_t^n = i)$ are given by the emission matrix

$$E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad \text{where } E_{ij} = p(x_t^n = j | z_t^n = i).$$

Consider the model parameters

$$\theta = (A, \gamma, p_{di}, p_{dn}).$$

The transition probabilities $p(z_t^n = j | z_{t-1}^n = i, \theta)$ are described by the transition matrices $\mathcal{T}(\theta, n)$, which depend on n and θ in the following way:

$$\mathcal{T}(\theta, n) = \begin{pmatrix} 1 - p_{nd}(n) - p_{di} & p_{nd}(n) & p_{di} \\ p_{dn} & 1 - p_{dn} - p_{di} & p_{di} \\ 0 & 0 & 1 \end{pmatrix}, \quad \text{with } p_{nd}(n) = A \exp(-\gamma \cdot n).$$

The joint probability of the model factorises as follows

$$\begin{aligned} p(\{x_t^n\}, \{z_t^n\} | \theta) &= \prod_{n=1}^{N} \left(p(z_1^n) \prod_{t=2}^{T} p(z_t^n | z_{t-1}^n, \theta) \prod_{t=1}^{T} p(x_t^n | z_t^n) \right) \\ &= \prod_{n=1}^{N} \left(p(z_1^n) \prod_{t=2}^{T} \mathcal{T}_{z_{t-1}^n z_t^n}(\theta, n) \prod_{t=1}^{T} E_{z_t^n x_t^n} \right). \end{aligned} \tag{3.11}$$

We give θ a factorised prior

$$p(\theta) = p(A)p(\gamma)p(p_{di})p(p_{dn}),$$

where the factors are defined by the following

$$\begin{aligned} &4A \sim \text{Beta}(\alpha_{4A}, \beta_{4A}), \qquad \gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma), \\ &4p_{di} \sim \text{Beta}(\alpha_{4pdi}, \beta_{4pdi}), \quad 4p_{dn} \sim \text{Beta}(\alpha_{4pdn}, \beta_{4pdn}). \end{aligned}$$

Goal: estimate the model parameters θ given the observations $\{x_t^n\}$.

3.3 Nearest Neighbour Model Derivation

Mirroring the previous model's development, this section derives the nearest neighbour model and its problem formulation. The next section then gives a concise mathematical formulation.

Before elaborating on the model details, we first present a slice of the graphical model in figure 3.2. The hidden Markov model structure for the nozzle states, as was present in the previous model, is clearly identifiable. However, it is now observed that the transition of a nozzle state to the next time step also depends on the neighbouring nozzle states.

As before, we model the latent nozzle states using the random variables z_t^n, which take values in $\{0, 1, 2\}$ for $t = 1, \ldots, T,\ n = 1, \ldots, N$. Similarly, the print observations are modelled by the random variables x_t^n, assuming values in $\{0, 1\}$. See table 3.1 for an overview of the random variables, their values, the corresponding states and their interpretation.

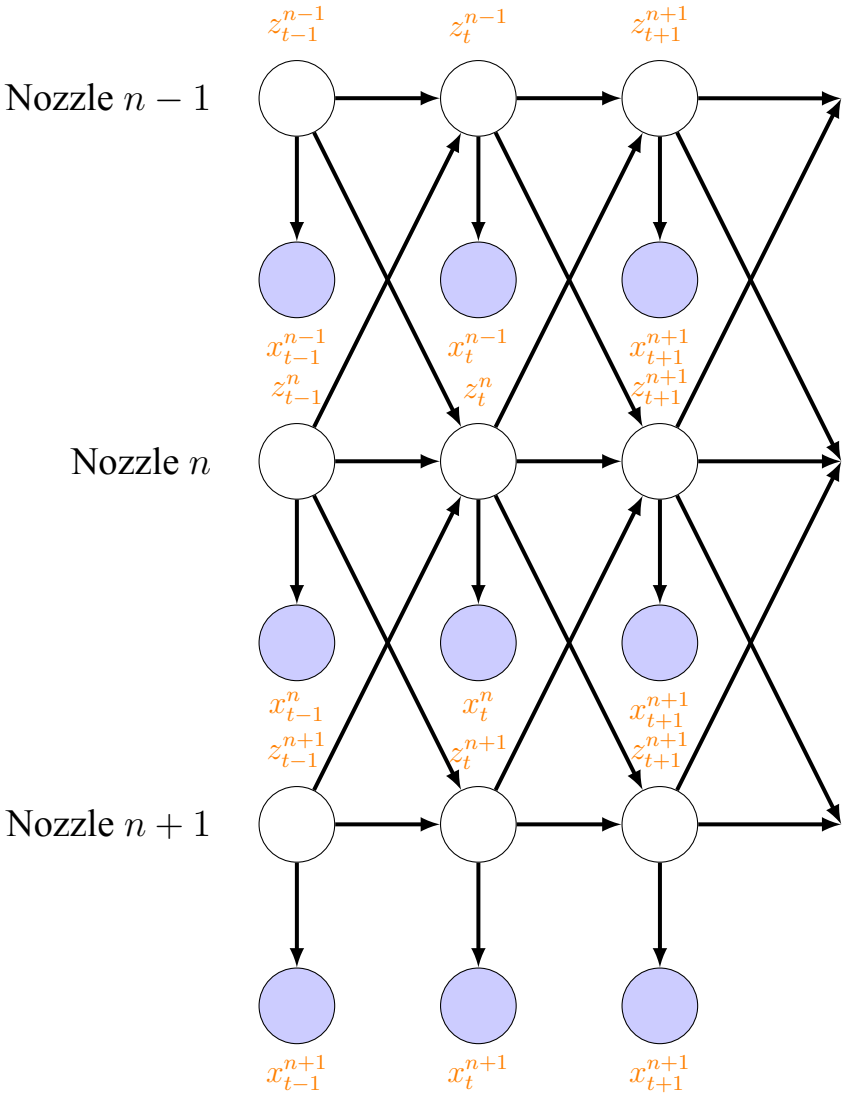

Figure 3.2: Graphical model for the nearest neighbour model. This diagram extends the standard HMM structure by illustrating that the latent state of nozzle n at time t depends not only on its previous state z_{t-1}^n, but also on the previous states of its neighbouring nozzles z_{t-1}^{n-1} and z_{t-1}^{n+1}.

3.3.1 Transition and Emission Probabilities

As in the previous model, we assume that if a nozzle is in state normal at a given time step, it produces the observation state OK. If the nozzle is in the other two states, it produces the observation state not OK. Hence we

have the emission matrix

$$E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad E_{ij} = p(x_t^n = j | z_t^n = i). \tag{3.12}$$

In the previous model, we assumed that every nozzle was independent of all other nozzles. In the real-world printing process, however, it is possible for dried ink to spread to neighbouring nozzles. Therefore, we assume that the transition from z_{t-1}^n to z_t^n additionally depends on the previous neighbouring states z_{t-1}^{n-1} and z_{t-1}^{n+1}. This is again a simplification; although nozzles are evenly spaced when projected onto the axis perpendicular to the paper's motion, their position might differ along the flow direction, resulting in varying distances between adjacent nozzle pairs. The proposed method might still apply in practice, if we consider an extended neighbourhood. As before, we assume the transition probabilities to be independent of time, but dependent on the nozzle number n.

Mathematically, we will formulate this as follows. Let θ denote the model parameters

$$\theta = (A, \gamma, p_{di0}, p_{di1}, p_{di2}, p_{dn}). \tag{3.13}$$

We consider the four-dimensional tensor $\mathcal{T}(\theta, n)$ of transition probabilities, where

$$\mathcal{T}_{ijkl}(\theta, n) = p(z_t^n = l | z_{t-1}^{n-1} = i, z_{t-1}^n = j, z_{t-1}^{n+1} = k, \theta), \tag{3.14}$$

for $n = 2, \ldots, N-1$. For $n = 1$ and $n = N$, we let the transition probabilities equal

$$\begin{aligned} p(z_t^1 = l | z_{t-1}^1 = j, z_{t-1}^2 = k, \theta) &= \mathcal{T}_{0jkl}(\theta, 1), \\ p(z_t^N = l | z_{t-1}^{N-1} = i, z_{t-1}^N = j, \theta) &= \mathcal{T}_{ij0l}(\theta, N). \end{aligned} \tag{3.15}$$

That is, we treat the transition probabilities as if there were an extra nozzle in the normal state. This is not an unusual assumption, since a well-functioning nozzle is expected to have little impact on its neighbours and, accordingly, could just as well be absent. Now, let $M^{(i,k)}(\theta, n)$ for $i, k = 0, 1, 2$ be matrices such that $M_{j,l}^{(i,k)}(\theta, n) = \mathcal{T}_{ijkl}(\theta, n)$. Then we can define the tensors $\mathcal{T}(\theta, n)$ as follows

$$M^{(i,k)}(\theta, n) = \begin{cases} \begin{pmatrix} 1 - p_{nd}(\theta, n) - p_{di0} & p_{nd}(\theta, n) & p_{di0} \\ p_{dn} & 1 - p_{dn} - p_{di0} & p_{di0} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i, k \neq 2, \\ \begin{pmatrix} 1 - p_{nd}(\theta, n) - p_{di1} & p_{nd}(\theta, n) & p_{di1} \\ p_{dn} & 1 - p_{dn} - p_{di1} & p_{di1} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i = 2, k \neq 2 \text{ or } i \neq 2, k = 2, \\ \begin{pmatrix} 1 - p_{nd}(\theta, n) - p_{di2} & p_{nd}(\theta, n) & p_{di2} \\ p_{dn} & 1 - p_{dn} - p_{di2} & p_{di2} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i, k = 2, \end{cases}$$

$$\text{with } p_{nd}(\theta, n) = A \exp(-\gamma \cdot n). \tag{3.16}$$

An overview of the model parameters and their parameter spaces is given in table 3.3.

We motivate this parametrisation of the transition probabilities in the following manner. As before, once a nozzle becomes clogged by dried ink, it remains clogged indefinitely. Furthermore, in the real-world printing process, the probability of a nozzle transitioning from the normal state to the dusty state appears to be independent of the states of its neighbours. Consequently, this transition probability is always equal to $p_{nd}(\theta, n)$, maintaining the same form as in the previous model. Similarly, the probability of transitioning from the dusty state to the normal state seems independent of the neighbouring nozzles' states. Therefore, this probability will always equal p_{dn}. However, the probability of a nozzle transitioning from the normal state to the dry-in state depends on the number of neighbouring nozzles that are already in the dry-in state. In this process, it is irrelevant whether a neighbouring

nozzle is in the normal or dusty state. Therefore, the probability that a nozzle in state normal transitions to the state dry-in depends solely on the count of its neighbours in state dry-in. Consequently, we define the transition probability from the normal to the dry-in state as p_{di0}, p_{di1} or p_{di2} when $0, 1$, or 2 neighbouring nozzles are in the dry-in state, respectively. We assume that the same probabilities and relations hold for the transition from the dusty state to the dry-in state, as indicated by inspection of the real-world printing process.

Given the transition and emission probabilities described above, the joint probability of the model factorises as follows

$$\begin{aligned} p(\{x_t^n\},\{z_t^n\}|\theta) = \prod_{n=2}^{N-1} \Bigg(p(z_1^n) \prod_{t=2}^{T} p(z_t^n|z_{t-1}^{n-1}, z_{t-1}^n, z_{t-1}^{n+1}, \theta) \prod_{t=1}^{T} p(x_t^n|z_t^n) \Bigg) \cdot \\ p(z_1^1) \prod_{t=2}^{T} p(z_t^1|z_{t-1}^1, z_{t-1}^2, \theta) \prod_{t=1}^{T} p(x_t^1|z_t^1) \cdot \\ p(z_1^N) \prod_{t=2}^{T} p(z_t^N|z_{t-1}^{N-1}, z_{t-1}^N, \theta) \prod_{t=1}^{T} p(x_t^N|z_t^N) \\ = \prod_{n=2}^{N-1} \Bigg(p(z_1^n) \prod_{t=2}^{T} \mathcal{T}_{z_{t-1}^{n-1} z_{t-1}^n z_{t-1}^{n+1} z_t^n}(\theta, n) \prod_{t=1}^{T} E_{z_t^n x_t^n} \Bigg) \cdot \\ p(z_1^1) \prod_{t=2}^{T} \mathcal{T}_{0 z_{t-1}^1 z_{t-1}^2 z_t^1}(\theta, 1) \prod_{t=1}^{T} E_{z_t^n x_t^n} \cdot \\ p(z_1^N) \prod_{t=2}^{T} \mathcal{T}_{z_{t-1}^{N-1} z_{t-1}^N 0 z_t^N}(\theta, N) \prod_{t=1}^{T} E_{z_t^N x_t^N} . \end{aligned} \tag{3.17}$$

This corresponds to the graphical model in figure 3.2.

3.3.2 Prior Distributions

As before, we put a prior on the model parameters θ. Again, we assume that the prior factorises into $p(\theta) = p(A)p(\gamma)p(p_{di0})p(p_{di1})p(p_{di2})p(p_{dn})$ and give the factors the following distributions

$$\begin{aligned} &4A \sim \mathrm{Beta}(\alpha_{4A}, \beta_{4A}), && \gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma), \\ &4p_{di0} \sim \mathrm{Beta}(\alpha_{4pdi0}, \beta_{4pdi0}), && 4p_{di1} \sim \mathrm{Beta}(\alpha_{4pdi1}, \beta_{4pdi1}), \\ &4p_{di2} \sim \mathrm{Beta}(\alpha_{4pdi2}, \beta_{4pdi2}), && 4p_{dn} \sim \mathrm{Beta}(\alpha_{4pdn}, \beta_{4pdn}), \end{aligned} \tag{3.18}$$

following the same reasoning as for the previous model. The methodologies presented in chapter 4 should enable one to infer the model parameters given the observations $\{x_t^n\}$.

Parameters θ	Parameter Space	Prior Distribution
A	$[0, 1/4]$	$4A \sim \mathrm{Beta}(\alpha_{4A}, \beta_{4A})$
γ	$[0, \infty)$	$\gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma)$
p_{di0}	$[0, 1/4]$	$4p_{di0} \sim \mathrm{Beta}(\alpha_{4pdi0}, \beta_{4pdi0})$
p_{di1}	$[0, 1/4]$	$4p_{di1} \sim \mathrm{Beta}(\alpha_{4pdi1}, \beta_{4pdi1})$
p_{di2}	$[0, 1/4]$	$4p_{di2} \sim \mathrm{Beta}(\alpha_{4pdi2}, \beta_{4pdi2})$
p_{dn}	$[0, 1/4]$	$4p_{dn} \sim \mathrm{Beta}(\alpha_{4pdn}, \beta_{4pdn})$

Table 3.3: Summary of the model parameters for the nearest neighbour model. The table lists the parameters $\theta = (A, \gamma, p_{di0}, p_{di1}, p_{di2}, p_{dn})$, their respective domains, and the corresponding prior distributions used to capture the extended dependencies in the transition dynamics.

3.4 Nearest Neighbour Model Problem Formulation

Inference Problem Formulation

Consider the random variables representing observations

$$\{x_t^n \in \{0,1\} : \text{for } t = 1,\dots,T,\ n = 1,\dots,N\}$$

and latent states

$$\{z_t^n \in \{0,1,2\} : \text{for } t = 1,\dots,T,\ n = 1,\dots,N\}.$$

We assume that $z_1^n = 0$ for all $n = 1,\dots,N$.

The emission probabilities $p(x_t^n = j | z_t^n = i)$ are given by the emission matrix

$$E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad \text{where } E_{ij} = p(x_t^n = j | z_t^n = i).$$

Consider the model parameters

$$\theta = (A, \gamma, p_{di0}, p_{di1}, p_{di2}, p_{dn}).$$

The transition probabilities $p(z_t^n = l | z_{t-1}^{n-1} = i, z_{t-1}^n = j, z_{t-1}^{n+1} = k, \theta)$ are assumed to be independent of t, but dependent on n. We describe the transition probabilities by the four-dimensional transition tensors $\mathcal{T}(\theta, n)$. We write

$$\mathcal{T}_{ijkl}(\theta, n) = p(z_t^n = l | z_{t-1}^{n-1} = i, z_{t-1}^n = j, z_{t-1}^{n+1} = k, \theta).$$

Let $M^{(i,k)}(\theta, n)$ for $i, k = 0, 1, 2$ be matrices such that $M_{jl}^{(i,k)}(\theta, n) = \mathcal{T}_{ijkl}(\theta, n)$. Then we define $\mathcal{T}$ through matrices $\{M^{(i,k)}(\theta, n)\}$:

$$M^{(i,k)}(\theta, n) = \begin{cases} \begin{pmatrix} 1 - p_{nd}(n) - p_{di0} & p_{nd}(n) & p_{di0} \\ p_{dn} & 1 - p_{dn} - p_{di0} & p_{di0} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i, k \neq 2, \\ \begin{pmatrix} 1 - p_{nd}(n) - p_{di1} & p_{nd}(n) & p_{di1} \\ p_{dn} & 1 - p_{dn} - p_{di1} & p_{di1} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i = 2, k \neq 2 \text{ or } i \neq 2, k = 2, \\ \begin{pmatrix} 1 - p_{nd}(n) - p_{di2} & p_{nd}(n) & p_{di2} \\ p_{dn} & 1 - p_{dn} - p_{di2} & p_{di2} \\ 0 & 0 & 1 \end{pmatrix} & \text{if } i, k = 2, \end{cases}$$

$$\text{with } p_{nd}(n) = A \exp(-\gamma \cdot n).$$

For $n = 1$, we let the transition probability be

$$p(z_t^1 = l | z_{t-1}^1 = j, z_{t-1}^2 = k, \theta) = \mathcal{T}_{0jkl}(\theta, 1).$$

For $n = N$, we define the transition probability

$$p(z_t^N = l | z_{t-1}^{N-1} = i, z_{t-1}^N = j, \theta) = \mathcal{T}_{ij0l}(\theta, N).$$

The joint probability of the model factorises as follows

$$
\begin{aligned}
p(\{x_t^n\},\{z_t^n\}|\theta) = \prod_{n=2}^{N-1}\Bigg(p(z_1^n)\prod_{t=2}^{T}p(z_t^n|z_{t-1}^{n-1},z_{t-1}^{n},z_{t-1}^{n+1},\theta)\prod_{t=1}^{T}p(x_t^n|z_t^n)\Bigg)\cdot \\
p(z_1^1)\prod_{t=2}^{T}p(z_t^1|z_{t-1}^1,z_{t-1}^2,\theta)\prod_{t=1}^{T}p(x_t^1|z_t^1)\cdot \\
p(z_1^N)\prod_{t=2}^{T}p(z_t^N|z_{t-1}^{N-1},z_{t-1}^{N},\theta)\prod_{t=1}^{T}p(x_t^N|z_t^N) \\
= \prod_{n=2}^{N-1}\Bigg(p(z_1^n)\prod_{t=2}^{T}\mathcal{T}_{z_{t-1}^{n-1}z_{t-1}^{n}z_{t-1}^{n+1}z_t^n}(\theta,n)\prod_{t=1}^{T}E_{z_t^n x_t^n}\Bigg)\cdot \\
p(z_1^1)\prod_{t=2}^{T}\mathcal{T}_{0z_{t-1}^1z_{t-1}^2z_t^1}(\theta,1)\prod_{t=1}^{T}E_{z_t^n x_t^n}\cdot \\
p(z_1^N)\prod_{t=2}^{T}\mathcal{T}_{z_{t-1}^{N-1}z_{t-1}^{N}0z_t^N}(\theta,N)\prod_{t=1}^{T}E_{z_t^N x_t^N}.
\end{aligned}
$$

We give θ a factorised prior

$$p(\theta) = p(A)p(\gamma)p(p_{di0})p(p_{di1})p(p_{di2})p(p_{dn}),$$

where the factors are defined by the following

$$
\begin{aligned}
4A &\sim \mathrm{Beta}(\alpha_{4A},\beta_{4A}), & \gamma &\sim \Gamma(\alpha_\gamma,\theta_\gamma),\\
4p_{di0} &\sim \mathrm{Beta}(\alpha_{4pdi0},\beta_{4pdi0}), & 4p_{di1} &\sim \mathrm{Beta}(\alpha_{4pdi1},\beta_{4pdi1}),\\
4p_{di2} &\sim \mathrm{Beta}(\alpha_{4pdi2},\beta_{4pdi2}), & 4p_{dn} &\sim \mathrm{Beta}(\alpha_{4pdn},\beta_{4pdn}).
\end{aligned}
$$

Goal: estimate the model parameters θ given the observations $\{x_t^n\}$.

3.5 More Conventional Modelling Approaches

When available knowledge is too sparse to parametrise the transition probabilities, or when simplicity is preferred, one can fall back on more conventional printing-process models. Again we may consider independent HMMs as in figure 3.1 or connect the HMMs to their nearest neighbours as in figure 3.2. Unlike before, the transition probabilities do not depend on the nozzle number n and are not governed by a specific parametrisation.

For the HMM model, one may follow the approach of [6][p. 476] and put an independent Dirichlet prior $\mathrm{Dir}(\delta_1^i,\ldots,\delta_s^i)$ on every row $(\mathcal{T}_{i1},\ldots,\mathcal{T}_{is})$ of the transition matrix. By conjugacy, each row remains independent in the posterior, with $(\mathcal{T}_{i1},\ldots,\mathcal{T}_{is})|\{z_t^n\},\{x_t^n\} \sim \mathrm{Dir}(\delta_1^i+n_{i1},\ldots,\delta_s^i+n_{is})$, where n_{ij} represents the number of transitions from state i to state j. One can then use a standard Markov chain Monte Carlo approach as described in chapter 4 to estimate the transition probabilities given the observations $\{x_t^n\}$.

We can propose a similar model if nearest neighbour interactions are present. As before, let $\mathcal{T}_{ijkl} = p\big(z_t^n = l|z_{t-1}^{n-1}=i,z_{t-1}^n=j,z_{t-1}^{n+1}=k\big)$. For each (i,k), define the matrix $M^{(i,k)}$ via $M_{jl}^{(i,k)} = \mathcal{T}_{ijkl}$. If we put independent Dirichlet priors $\mathrm{Dir}(\delta_1^{(ijk)},\ldots,\delta_s^{(ijk)})$ on the rows $M_j^{(i,k)}$, as before, a-posteriori these rows remain independent and $M_j^{(i,k)}|\{z_t^n\},\{x_t^n\}$ is distributed according to $\mathrm{Dir}(\delta_1^{(ijk)}+n_{ijk1},\ldots,\delta_s^{(ijk)}+n_{ijks})$. Here n_{ijks} is the number of transitions from states i,j and k to s. These posterior distributions can then be used to estimate the transition probabilities through an MCMC algorithm as described in chapter 4.

4

Parameter Inference Methodologies

In this chapter, we consider the two models outlined in chapter 3 and present Bayesian methodologies for inferring their parameters. The first approach relies on sampling methods, while the others employ variational inference. Although the sampling algorithm is more general and asymptotically unbiased, the variational approaches are less sensitive to initialisation. We will also briefly touch upon sampling for the more conventional models from section 3.5 and on stochastic variational inference (SVI) as a means to reduce the computational complexity of the variational methods. In chapter 5, the methods described in this chapter will be applied with the exception of SVI.

4.1 MCMC Method

In this section, we explore a Markov chain Monte Carlo approach for estimating model parameters. First, we outline the general algorithm. Next, we detail the implementation of the Gibbs sweeps. Finally, we present an adaptive strategy that fine-tunes the acceptance probabilities in the Metropolis-within-Gibbs steps to enhance the chain's mixing performance. In appendix B, we establish the convergence of the algorithm. Its convergence speed, however, is highly sensitive to the initialisation of $\{z_t^n\}$. While a good initialisation is often dictated by the specifics of the real-world problem, the variational methods outlined in section 4.2 offer a less sensitive alternative if required.

At the end of this section, we will also touch upon sampling for the more conventional models from section 3.5.

4.1.1 Algorithm Overview

Consider the finite-valued latent random variables $\{z_t^n\}$, observed random variables $\{x_t^n\}$ and model parameters θ as defined in chapter 3, where the index n runs over $\{1, \dots, N\}$ and t over $\{1, \dots, T\}$. $p(\{z_t^n\}, \{x_t^n\}|\theta)$ is defined through the transition and emission probabilities and the prior $p(\theta)$ is chosen appropriately. To sample θ and $\{z_t^n\}$ from posterior $p(\theta, \{z_t^n\}|\{x_t^n\})$, one may use two-step Gibbs sampling [6, section 13.1.4].

$$
\begin{aligned}
&1.\quad \text{Initialise } \{z_t^n\} \text{ and } \theta.\\
&2.\quad \text{For } i = 1, \dots, M:\\
&\qquad \text{- Sample } \{z_t^n\} \sim p(\{z_t^n\}|\{x_t^n\}, \theta).\\
&\qquad \text{- Sample } \theta \sim p(\theta|\{z_t^n\}, \{x_t^n\}).
\end{aligned}
\tag{4.1}
$$

Sampling from these distributions is challenging because $p(\theta|\{z_t^n\}, \{x_t^n\})$ lacks a closed-form analytical solution, while $p(\{z_t^n\}|\{x_t^n\}, \theta)$ is too high-dimensional for direct numerical computation. Therefore, we will replace the first sampling step by N_{Gibbs} Gibbs sweeps and the second by N_{MwG} Metropolis-within-Gibbs sweeps [6, section 13.1.4]. The algorithm is then described as follows.

1. Initialise $\{z_t^n\}$ and θ.
2. For $i = 1, \dots, M$:
   For $j = 1, \dots, N_{\text{Gibbs}}$:

- Sample $z_1^1 \sim p(z_1^1|\{z_1^1\}^c, \{x_t^n\}, \theta)$.

$\vdots$

- Sample $z_T^1 \sim p(z_T^1|\{z_T^1\}^c, \{x_t^n\}, \theta)$.
- Sample $z_1^2 \sim p(z_1^2|\{z_1^2\}^c, \{x_t^n\}, \theta)$.

$\vdots$

- Sample $z_T^2 \sim p(z_T^2|\{z_T^2\}^c, \{x_t^n\}, \theta)$.

$\vdots$

- Sample $z_1^N \sim p(z_1^N|\{z_1^N\}^c, \{x_t^n\}, \theta)$.

$\vdots$

- Sample $z_T^N \sim p(z_T^N|\{z_T^N\}^c, \{x_t^n\}, \theta)$.

For $k = 1, \ldots, N_{\text{MwG}}$:

For $l = 1, \ldots, |\theta|$:

- Sample $\theta_l^\star \sim q_l(\theta_l'|\theta_l)$.
- Accept with probability $A = \min\Big(1, \frac{\tilde{p}(\theta_l^\star)\, q_l(\theta_l|\theta_l^\star)}{\tilde{p}(\theta_l)\, q_l(\theta_l^\star|\theta_l)}\Big)$.
- If accepted $\theta_l = \theta_l^\star$, otherwise $\theta_l = \theta_l$. (4.2)

Here $q_l(\theta_l'|\theta_l)$ is a normal distribution with mean θ and variance $\exp(2ls_l)$. That is, ls_l is the logarithm of the standard deviation of q_l, which will be tuned to optimise the proposal distributions as described in section 4.1.3. Note that since $q_l(\theta_l|\theta_l^\star) = q_l(\theta_l^\star|\theta_l)$ for a normal distribution, the acceptance probability becomes $A = \min\left(1, \frac{\tilde{p}(\theta_l^\star)}{\tilde{p}(\theta_l)}\right)$. We set $\tilde{p}(\theta_l) = p(\{z_t^n\}, \{x_t^n\}|\theta)p(\theta)$, which is proportional to $p(\theta_l|\theta_{\setminus l}, \{z_t^n\}, \{x_t^n\})$ as explained by Bayes' theorem.

4.1.2 Gibbs Steps Procedure

It is clear that the Metropolis-within-Gibbs steps where the variables in θ are sampled can be executed; however, to perform the Gibbs steps, the conditional distributions $p(z_\tau^\nu|\{z_\tau^\nu\}^c, \{x_t^n\}, \theta)$ have to be computed. To compute these distributions, we consider the graphical models corresponding to the probabilistic models. In a graphical model, each node represents a random variable. A key property states that a random variable is conditionally independent of all other variables given its Markov blanket; that is, its parents, its children, and the parents of its children [18, p. 65]. This fact follows from applying the concept of d-separation to directed graphical models[1]. We find that $p(z_\tau^\nu|\{z_\tau^\nu\}^c, \{x_t^n\}, \theta) = p(z_\tau^\nu|\mathrm{pa}(z_\tau^\nu), \mathrm{ch}(z_\tau^\nu), \mathrm{pa}(\mathrm{ch}(z_\tau^\nu)), \theta)$, where θ is treated as a constant.

In the case of the HMM model, it follows that

$$\begin{aligned} p(z_\tau^\nu|\{z_\tau^\nu\}^c, \{x_t^n\}, \theta) = p(z_\tau^\nu|z_{\tau-1}^\nu, x_\tau^\nu, z_{\tau+1}^\nu, \theta) &\propto p(z_\tau^\nu|z_{\tau-1}^\nu, \theta)p(x_\tau^\nu|z_\tau^\nu)p(z_{\tau+1}^\nu|z_\tau^\nu, \theta), \qquad \text{for } \tau \neq 1, T, \\ p(z_1^\nu|\{z_1^\nu\}^c, \{x_1^n\}, \theta) = p(z_1^\nu|x_1^\nu, z_2^\nu, \theta) &\propto p(x_1^\nu|z_1^\nu)p(z_2^\nu|z_1^\nu, \theta), \\ p(z_T^\nu|\{z_T^\nu\}^c, \{x_t^n\}, \theta) = p(z_T^\nu|z_{T-1}^\nu, x_T^\nu, \theta) &\propto p(z_T^\nu|z_{T-1}^\nu, \theta)p(x_T^\nu|z_T^\nu). \end{aligned}$$

That is, $p(z_\tau^\nu|\{z_\tau^\nu\}^c, \{x_t^n\}, \theta)$ is proportional to the product of all transition and emission probabilities containing both z_τ^ν and elements of its Markov blanket. It is clear that these products can be normalised, since the latent variables have finite range. Hence, the Gibbs steps are viable. For the nearest neighbour model and more general models represented by directed graphical models with finite-valued latents, the same derivation can be applied.

4.1.3 Adaptive Proposal Distributions

For the proposals $q_l(\theta_l'|\theta_l) \sim \mathcal{N}(\theta_l, \exp(2ls_l))$, it is unclear what the values of ls_l should be. When ls_l is small, the proposed samples remain very close to the current sample, resulting in a high acceptance rate. However, because only minor changes in θ_l occur, the chain may mix slowly. Conversely, when ls_l is large, the proposals tend to be much further from the current sample, which can improve exploration but often leads to an extremely

[1] See appendix A for an overview of directed graphical models and d-separation.

low acceptance rate—again causing slow mixing. It has been shown that values of ls_l that result in an acceptance rate away from both 0 and 1 lead to fast mixing. Specifically, an acceptance rate close to 0.44 appears to optimal for the Metropolis-within-Gibbs algorithm. [16]

In practice, effective values for ls_l, $l = 1, \ldots, |\theta|$, are often determined through trial and error; however, when the dimension of θ is large, manual tuning becomes impractical, and automated methods are required. In [16, section 3.3], an adaptive algorithm is presented, which proceeds as follows. The above algorithm is divided into batches of sweeps of size bs. E.g., bs can be chosen to equal 50. After the n-th batch, the algorithm adjusts every ls_l by a quantity $\delta(n)$ to drive the acceptance rate towards 0.44. Specifically, after the n-th batch, the average acceptance rate over the last bs sweeps is calculated for each component l. If the average rate is below 0.44, $\delta(n)$ is subtracted from ls_l, reducing the step size to increase acceptance; if the rate exceeds 0.44, $\delta(n)$ is added, increasing the step size to encourage more exploratory moves.

To maintain convergence of the Markov chain, it is sufficient to restrict ls_l to a finite interval $[-L, L]$ and let $\delta(n) \to 0$. We can set $\delta(n) = \min(c, n^{-1/2})$, where c is a constant. In appendix B, it is shown that these conditions make the adaptive algorithm satisfy the convergence criteria given in [16].

4.1.4 MCMC for Conventional Models

In section 3.5, we presented more conventional models for the printing process that do not posses parametrisations of the transition probabilities. To estimate the transition probabilities, we present a sampling scheme analogous to that shown for the parametrised models.

$$\begin{aligned} &1.\ \text{Initialise } \{z_t^n\} \text{ and } \mathcal{T}. \\ &2.\ \text{For } i = 1, \ldots, M: \\ &\quad \text{- Sample } \{z_t^n\} \sim p(\{z_t^n\}|\{x_t^n\}, \mathcal{T}). \\ &\quad \text{- Sample } \mathcal{T} \sim p(\mathcal{T}|\{z_t^n\}, \{x_t^n\}). \end{aligned} \tag{4.3}$$

The samples of $\{z_t^n\}$ can be drawn through Gibbs sweeps as described previously. As mentioned in section 3.5, the posterior $p(\mathcal{T}|\{z_t^n\}, \{x_t^n\})$ factorises into independent Dirichlet distributions, from which we can readily draw samples. Updating the transition probabilities in this way is far simpler than the Metropolis-within-Gibbs procedure described previously. Thus, not only is the model simpler, but its sampling algorithm is too.

4.2 Variational Inference Method

While the MCMC method is very general and can be adapted to a wide range of models with only minor modifications, its convergence speed is highly sensitive to the initialisation of $\{z_t^n\}$. Consequently, we introduce variational inference methods that are less sensitive to initialisation, albeit at the expense of some generality. Notably, VI algorithms lack the asymptotically unbiased property typical of MCMC methods—a shortcoming that may limit their accuracy in certain scenarios. First, we present a method tailored specifically to the models described in chapter 3. Next, we generalise this approach. Furthermore, we explore strategies in the form of stochastic variational inference (SVI), which might enhance the scalability of these methods for some models. However, it should be noted that for this work only non-stochastic algorithms have been implemented, and exclusively for the parametrised HMM model, as will be demonstrated in the next chapter.

4.2.1 Model Specific VI Approach

We again consider the finite-valued latent random variables $\{z_t^n\}$, observed random variables $\{x_t^n\}$ and model parameters θ as defined in chapter 3, where the index n runs over $\{1, \ldots, N\}$ and t over $\{1, \ldots, T\}$. $p(\{z_t^n\}, \{x_t^n\}|\theta)$ is defined through the transition and emission probabilities and the prior $p(\theta)$ is chosen appropriately. It is our aim to find a variational distribution $q(\theta, \{z_t^n\})$ approximating the posterior $p(\theta, \{z_t^n\}|\{x_t^n\})$ such that θ can be estimated.

We consider a mean field approximation for $q(\theta, \{z_t^n\})$. That is,

$$q(\theta, \{z_t^n\}) = q(\theta)q(\{z_t^n\}), \tag{4.4}$$

where $q(\theta)$ factorises further. E.g., for the HMM model we obtain

$$q(\theta, \{z_t^n\}) = q(A)q(\gamma)q(p_{di})q(p_{dn})q(\{z_t^n\}). \tag{4.5}$$

$q(A), q(\gamma), q(p_{di}), q(p_{dn})$ are further restricted to be distributed in the same way as their prior. I.e., for $q(\theta) = q(A)q(\gamma)q(p_{di})q(p_{dn})$,

$$\begin{aligned} &4A \sim \text{Beta}(\alpha_{4A}, \beta_{4A}), \qquad \gamma \sim \Gamma(\alpha_\gamma, \theta_\gamma), \\ &4p_{di} \sim \text{Beta}(\alpha_{4pdi}, \beta_{4pdi}), \quad 4p_{dn} \sim \text{Beta}(\alpha_{4pdn}, \beta_{4pdn}). \end{aligned} \tag{4.6}$$

We write $q(\theta|\lambda)$ to indicate the dependence on the variational parameters. That is, λ is a vector of the variational parameters defined in equation (4.6). For the nearest neighbour model, similar assumptions can be imposed on $q(\theta)$.

Based on the assumptions above, an iterative algorithm where $q(\theta|\lambda)$ and $q(\{z_t^n\})$ are updated in turn, similar to the algorithm described in [7, section 2.2], will be presented below.

Suppose we have an initialisation for λ. Remember that since no further assumptions are imposed on $q(\{z_t^n\})$, the distribution that minimises the ELBO with respect to $q(\{z_t^n\})$ is that from equation (2.15).

$$\begin{aligned} q(\{z_t^n\}) &\propto \exp(\mathbb{E}_{q(\theta|\lambda)}\left[\log(p(\{z_t^n\}, \{x_t^n\}|\theta))\right]) \\ &= \exp\left(\mathbb{E}_{q(\theta|\lambda)}\left[\sum_{n=1}^{N}\sum_{t=2}^{T}\log(\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)) + \log(E_{z_t^n x_t^n})\right]\right) \\ &= \exp\left(\sum_{n=1}^{N}\sum_{t=2}^{T}\mathbb{E}_{q(\theta|\lambda)}\left[\log(\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n))\right] + \log(E_{z_t^n x_t^n})\right), \end{aligned} \tag{4.7}$$

where $\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)$ is the probability of transitioning to z_t^n given θ, n and the relevant previous states. For the HMM model this equals $\mathcal{T}_{z_{t-1}^n z_t^n}(\theta, n)$ with $\mathcal{T}(\theta, n)$ defined in equation (3.8). Observe here the similarity with the form of the joint probability distribution of the model in equation (3.11). It is apparent that the distribution factorises into a product of functions, each depending on only a few of the finite-valued random variables z_t^n. Hence, if expectations $\mathbb{E}_{q(\theta|\lambda)}[\log(\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)]$ for the different configurations of the relevant subsets of $\{z_t^n\}$ can be computed, (joint) marginals of $q(\{z_t^n\})$ can be obtained through a message passing algorithm. These (joint) marginals can then be used to optimise λ as described below. In practice, we compute each singleton marginal $q(z_t^n)$ and, for every transition term $\mathcal{T}\big(z_t^n|\{z_t^n\}^c, \theta, n\big)$, its corresponding joint marginal. In the nearest-neighbour case this means evaluating $q(z_t^n,\, z_{t-1}^{n-1},\, z_{t-1}^n,\, z_{t-1}^{n+1})$.

Given $q(\{z_t^n\})$, we optimise the portion of the ELBO that depends solely on λ to determine its optimal value. That is, we aim to maximise

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}[\log(p(\{z_t^n\}, \{x_t^n\}|\theta))] + \mathbb{E}_{q(\theta|\lambda)}[\log(p(\theta))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta|\lambda))], \tag{4.8}$$

with respect to λ. It is clear that $q(\{z_t^n\})$ is required for the first term in equation (4.8), which equals

$$\begin{aligned} \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}[\log(p(\{z_t^n\}, \{x_t^n\}|\theta))] &= \mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}\left[\sum_{n=1}^{N}\sum_{t=2}^{T}\log(\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)) + \log(E_{z_t^n x_t^n})\right] \\ &= \sum_{n=1}^{N}\sum_{t=2}^{T}\mathbb{E}_{q(\theta|\lambda)q(\{z_t^n\})}\left[\log(\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)) + \log(E_{z_t^n x_t^n})\right], \end{aligned} \tag{4.9}$$

where, as before, $\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)$ is the probability of transitioning to z_t^n given θ, n and the relevant previous states. For the two parametrised models from chapter 3, $\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)$ equals $\mathcal{T}_{z_{t-1}^n z_t^n}(\theta, n)$ or $\mathcal{T}_{z_{t-1}^{n-1} z_{t-1}^n z_{t-1}^{n+1} z_t^n}(\theta, n)$. It is clear that only the singleton marginals $q(z_t^n)$ and joint marginals for $\mathcal{T}(z_t^n|\{z_t^n\}^c, \theta, n)$ are required to evaluate this term. If we can then compute the derivatives of equation (4.8) with respect to the variational parameters, gradient based optimisation routines can be implemented to optimise λ.

We can summarise this algorithm as follows.

1. Initialise λ.
2. For $i = 1, \ldots, M$:
   - Compute marginals of $q(\{z_t^n\}) \propto \exp(\mathbb{E}_{q(\theta|\lambda)}\left[\log(p(\{z_t^n\}, \{x_t^n\}|\theta))\right])$.
   - Maximise $\mathcal{L}(\lambda)$ through gradient based optimisation.

(4.10)

Note that this algorithm is initialised solely through its variational parameters—unlike the MCMC method, no latent variable initialisation is required. Additionally, the variational distribution can undergo substantial variation in a single iteration of the algorithm, whereas the updates in the MCMC method are much more incremental. This makes the VI method less sensitive to its initialisation than the MCMC method.

Model Specific Approximation

Unfortunately, the models in chapter 3 admit no analytical expressions for $\mathbb{E}_{q(\theta|\lambda)}\left[\log \mathcal{T}_{ij}(\theta, n)\right]$ or $\mathbb{E}_{q(\theta|\lambda)}\left[\log \mathcal{T}_{ijkl}(\theta, n)\right]$, i.e. for the HMM and nearest neighbour model respectively. For example, the expectation $\mathbb{E}_{q(\theta|\lambda)}[\log(1 - A\exp(-\gamma \cdot n) - p_{di})]$ is analytically intractable. In equation (4.7), this issue can be resolved using Monte Carlo estimates. However, to compute derivatives of equation (4.8), these Monte Carlo estimates cannot be used.

A potential solution is to approximate the analytically intractable terms by differentiable expressions. For our specific models, we can use the approximation $\log(1 - x) \approx -x - \frac{x^2}{2}$ for small x and apply it to expectations similar to $\mathbb{E}_{q(\theta|\lambda)}[\log(1 - A\exp(-\gamma \cdot n) - p_{di})]$. It is then possible to apply gradient based optimisation techniques to the approximation of the ELBO. Note, however, that this approach is highly specific to our models. For example, a different parametrisation of the transition probabilities might render the approximation analytically intractable as well. In the following section we will present a more general method.

4.2.2 Automatic Differentiation Variational Inference Approach

In the previous section, we examined the algorithm outlined in equation (4.10), where $\mathcal{L}(\lambda)$ was approximated using a model-specific differentiable expression. In this section, the same outline will be followed, however, instead of approximating $\mathcal{L}(\lambda)$, a more general approach will be taken. Specifically, automatic differentiation variational inference (ADVI) will be employed to optimise $q(\theta)$ in every iteration, following the description provided in [10].

Consider the same models as in the last section. The model parameters in θ are transformed to the real line $\mathbb{R}$. E.g., parameters restricted to the positive real line $\mathbb{R}_{>0}$ can be transformed by $\log(\exp(x) - 1)$ and parameters restricted to $[0, u]$ can be transformed by $\log(\frac{x}{u-x})$. We write $\zeta = T(\theta)$ for the transformed parameters, where $T(\cdot)$ is invertible and differentiable. The joint probability becomes

$$p\left(\{z_t^n\}, \{x_t^n\}, \zeta\right) = p(\{z_t^n\}, \{x_t^n\}, T^{-1}(\zeta)) \left|\det J_{T^{-1}}(\zeta)\right|. \tag{4.11}$$

We assume that the variational distribution factorises, i.e., $q(\zeta, \{z_t^n\}) = q(\zeta)q(\{z_t^n\})$. Then we posit a Gaussian mean-field approximation on $q(\zeta)$,

$$q(\zeta|\lambda) = \prod_{k=1}^{K} \mathcal{N}(\zeta_k|\mu_k, \exp(2\omega_k)), \tag{4.12}$$

where K is the number of model parameters and $\lambda = (\mu_1, \ldots, \mu_K, \omega_1, \ldots, \omega_K)$. Here $\exp(\omega_k)$ is the standard deviation of the Gaussian corresponding to ζ_k.

As before, we first initialise λ. Then, $q(\{z_t^n\})$ and $q(\zeta|\lambda)$ are optimised in turn as in equation (4.10). The computation of $q(\{z_t^n\})$ remains unchanged. However, the maximisation of $\mathcal{L}(\lambda)$ follows a different approach.

Based on equation (4.11), we can write that

$$\begin{aligned}
\mathcal{L}(\lambda) &= \mathbb{E}_{q(\zeta|\lambda)q(\{z_t^n\})}\left[\log p(\{z_t^n\}, \{x_t^n\}, T^{-1}(\zeta)) + \log|\det J_{T^{-1}}(\zeta)|\right] - \mathbb{E}_{q(\zeta|\lambda)}[\log q(\zeta|\lambda)] \\
&= \mathbb{E}_{q(\zeta|\lambda)}\left[\mathbb{E}_{q(\{z_t^n\})}[\log p(\{z_t^n\}, \{x_t^n\}, T^{-1}(\zeta))] + \log|\det J_{T^{-1}}(\zeta)|\right] - \mathbb{E}_{q(\zeta|\lambda)}[\log q(\zeta|\lambda)] \\
&= \mathbb{E}_{q(\zeta|\lambda)}\left[\log \tilde{p}(\{x_t^n\}, T^{-1}(\zeta)) + \log|\det J_{T^{-1}}(\zeta)|\right] - \mathbb{E}_{q(\zeta|\lambda)}[\log q(\zeta|\lambda)],
\end{aligned} \tag{4.13}$$

for $\log \tilde{p}(\{x_t^n\}, \theta) = \mathbb{E}_{q(\{z_t^n\})}[\log p(\{z_t^n\}, \{x_t^n\}, T^{-1}(\zeta))]$. Given $q(\{z_t^n\})$, our goal is to compute the derivatives of $\mathcal{L}(\lambda)$ with respect to the variational parameters, enabling gradient-based optimization. By employing the reparametrisation trick—which transforms the variational distribution into a standard normal—we can express these derivatives as expectations of derivatives. The derivatives within the expectations can then be calculated using automatic differentiation, while the expectations themselves are estimated via Monte Carlo sampling.

Consider a transformation $S_\lambda(\cdot)$, which standardises the variational distribution on ζ to a standard normal. That is, $\eta = S_\lambda(\zeta) = \text{diag}(\exp(\omega))^{-1}(\zeta - \mu)$ has variational distribution

$$q(\eta) = \prod_{k=1}^{K} \mathcal{N}(\eta_k|0,1). \tag{4.14}$$

One may then write

$$\mathcal{L}(\lambda) = \mathbb{E}_{\mathcal{N}(\eta|0,I)}\left[\log \tilde{p}(\{x_t^n\}, T^{-1}(S_\lambda^{-1}(\eta))) + \log|\det J_{T^{-1}}(S_\lambda^{-1}(\eta))|\right] - \mathbb{E}_{q(\zeta|\lambda)}[\log q(\zeta|\lambda)]. \tag{4.15}$$

If we wish to compute the derivatives of equation (4.15) with respect to μ and ω, note that the expectation with respect to $\mathcal{N}(\eta|0,I)$ is independent of these parameters. Consequently, we can interchange differentiation and expectation, moving the derivatives with respect to μ and ω inside. The following expressions are then obtained,

$$\nabla_\mu \mathcal{L}(\lambda) = \mathbb{E}_{\mathcal{N}(\eta)}\left[J_{T^{-1}}(\zeta)\nabla_\theta \log \tilde{p}(\{x_t^n\}, \theta) + \nabla_\zeta \log|\det J_{T^{-1}}(\zeta)|\right], \tag{4.16}$$

$$\nabla_\omega \mathcal{L}(\lambda) = \mathbb{E}_{\mathcal{N}(\eta)}\left[(J_{T^{-1}}(\zeta)\nabla_\theta \log \tilde{p}(\{x_t^n\}, \theta) + \nabla_\zeta \log|\det J_{T^{-1}}(\zeta)|)\,\text{diag}(\exp(\omega))\eta\right] + 1. \tag{4.17}$$

The derivatives inside these expectations can be computed using automatic differentiation, a technique that applies the chain rule to evaluate the derivatives of programmed functions at specified input values [3]. By combining automatic differentiation with sampling, the expressions in equation (4.16) and equation (4.17) can be estimated. These unbiased estimates can then be used in a gradient based optimisation algorithm to find optimal values of λ.

In conclusion, in this section, we described a VI algorithm with the same outline as before, see equation (4.10). However, in this approach, the parameters θ are transformed to $\mathbb{R}^K$ and equipped with independent normals, such that $\mathcal{L}(\lambda)$ can be maximised through automatic differentiation and Monte Carlo approximation.

4.2.3 Stochastic Variational Inference

As is clear from the previous sections, variational inference casts statistical inference problems as optimisation problems. These problems can often be high-dimensional. E.g., in our models there are a large number of latents z_t^n and observables x_t^n. To address the challenges posed by high-dimensional variational inference problems, stochastic optimisation can be employed. This approach leads to a method known as stochastic variational inference (SVI) [8].

The basic SVI algorithm assumes that the data is independent. That is, we have N independent pairs (z^n, x^n). If we make the same assumptions as before on the variational distributions, i.e, $q(\theta, \{z^n\}) = q(\theta|\lambda)q(\{z^n\})$, we can write the ELBO for the variational parameters, $\mathcal{L}(\lambda)$, as follows.

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\theta|\lambda)}[\log(p(\theta))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta))] + \sum_{n=1}^{N} \mathbb{E}_{q(\theta|\lambda)}[\mathbb{E}_{q(z^n)}[\log(p(z^n, x^n|\theta))]]. \tag{4.18}$$

To estimate the ELBO and its gradient, we consider the random variable $I \sim \text{Unif}(1, \ldots, N)$. We then define

$$\mathcal{L}_I(\lambda) := \mathbb{E}_{q(\theta|\lambda)}[\log(p(\theta))] - \mathbb{E}_{q(\theta|\lambda)}[\log(q(\theta))] + N\,\mathbb{E}_{q(\theta|\lambda)}[\mathbb{E}_{q(z^I)}[\log(p(z^I, x^I|\theta))]]. \tag{4.19}$$

The expectation of $\mathcal{L}_I(\lambda)$ with respect to I equals $\mathcal{L}(\lambda)$. Consequently, when sampling i from I, the gradient of $\mathcal{L}_i(\lambda)$ serves as an unbiased estimate of $\nabla\mathcal{L}(\lambda)$. These gradient estimates can be applied in Robbins-Monro optimisation algorithms, which guarantee convergence to an optimum provided the estimates remain unbiased. [8]

As noted in [9], SVI is applicable to models composed of independent HMMs, including our model from section 3.1. In this case, the independent pairs are $(z^n, x^n) = (\{z_t^n : t = 1, \ldots, T\}, \{x_t^n : t = 1, \ldots, T\})$. However, when independence is absent, applying SVI becomes challenging, requiring modifications to the approach. For instance, [7] proposes an SVI algorithm for a stationary HMM with a large number of time steps. This method samples sub-chains from the long HMM to compute unbiased ELBO gradients, while buffer mechanisms help mitigate the effects of non-independent data. Although our models do not follow a stationary distribution, this technique may still be applicable to slightly different models.

5

Results

In this chapter, we report numerical experiments that apply the algorithms introduced in chapter 4 to the models of chapter 3. Note that all experiments were performed on synthetic data. In section 5.1, we analyse the MCMC sampler on both the HMM and nearest-neighbour models. In section 5.2, we assess the model-specific variational inference approach alongside the ADVI algorithm, applied exclusively to the HMM model. For each configuration, we illustrate the algorithm's estimation accuracy and convergence speed. We also briefly look at the non-parametrised models from section 3.5 in section 5.1.

5.1 MCMC Results

In section 5.1, we apply the MCMC sampler to both the HMM and nearest-neighbour models using the same approach. First, we examine a representative run—reporting iterations to convergence and estimate accuracy. Then, we inspect the acceptance probabilities of the Metropolis-within-Gibbs steps to asses the quality of the adaptive algorithm from section 4.1.3. Next, we analyse correlations among the sampled parameters. Finally, we compare convergence speed across different algorithm settings.

Concluding this section, we will also briefly present the sampling results for the more conventional models from section 3.5.

5.1.1 HMM Model

Run Analysis: Convergence and Outcomes

We employ the MCMC algorithm with adaptive proposal distributions from section 4.1, for the HMM model outlined in section 3.1. In this model, we set $N = 5000$ and $T = 30$, with N and T as defined in chapter 3. For the algorithm, 1200 sweeps are performed with $N_{\text{Gibbs}} = 10$ and $N_{\text{MwG}} = 50$. After every sweep, the proposal distributions are modified according to the algorithm in section 4.1.3 with $\delta(n) = \min(0.1, n^{-1/2})$. Note that the Gibbs sweeps are considerably more computationally intensive than the Metropolis-within-Gibbs sweeps because the former sample from the high-dimensional discrete latents $\{z_t^n\}$ while the latter sample from the low-dimensional parameters θ. Therefore, striking the right balance between N_{Gibbs} and N_{MwG} is crucial for maintaining computational efficiency.

In figure 5.1, the logarithm of the joint probability $p(\theta, \{z_t^n\}, \{x_t^n\})$ up to an additive constant is plotted against the algorithm's iterations as a measure of convergence. We see that after approximately 250 sweeps, the change in value slows down indicating that the algorithm is heading towards a steady state. Note, however, that a bad initialisation of $\{z_t^n\}$ might require the algorithm to perform many more sweeps before approaching convergence.

In figure 5.2, the sampled values of the parameter p_{dn} are shown alongside its true value. Following a burn-in period of 250 sweeps, the samples settle into a stable region near the true value, as expected. Figure 5.3 illustrates the running averages for p_{dn} computed both with and without burn-in. Although both running averages converge closely to the true value, the influence of the burn-in period remains evident for an extended duration. A similar pattern can be shown for the other parameters. In table 5.1, the parameter estimates—calculated as the averages

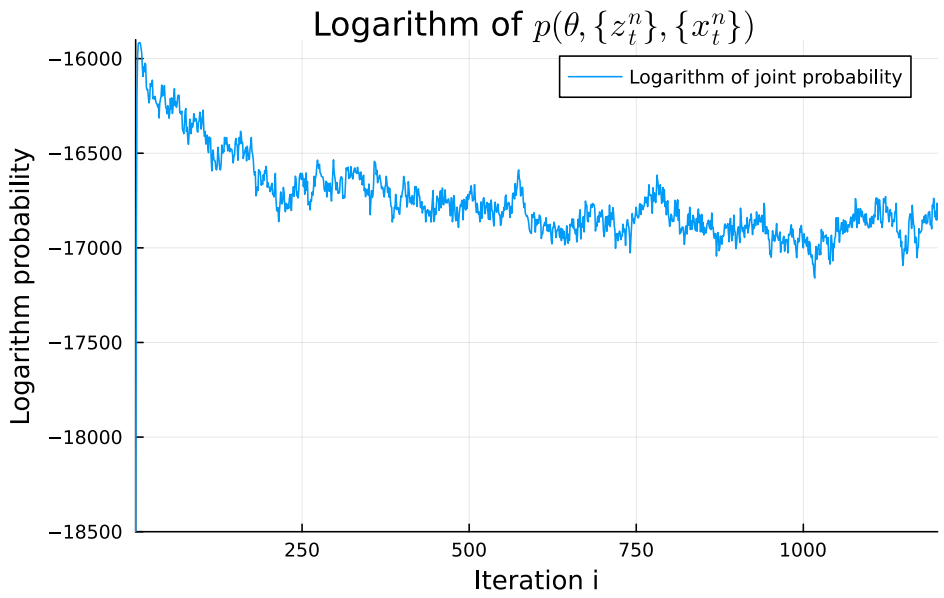

Figure 5.1: Plot of the logarithm of the joint probability $p(\theta, \{z_t^n\}, \{x_t^n\})$ up to an additive constant versus the number of MCMC iterations for the HMM model. Convergence is indicated by a plateau after approximately 250 sweeps, suggesting that the chain is approaching a steady state.

of the final 600 samples along with their sample standard deviations—are provided. Overall, the estimates closely approximate the true values, with the low standard deviations underscoring their precision.

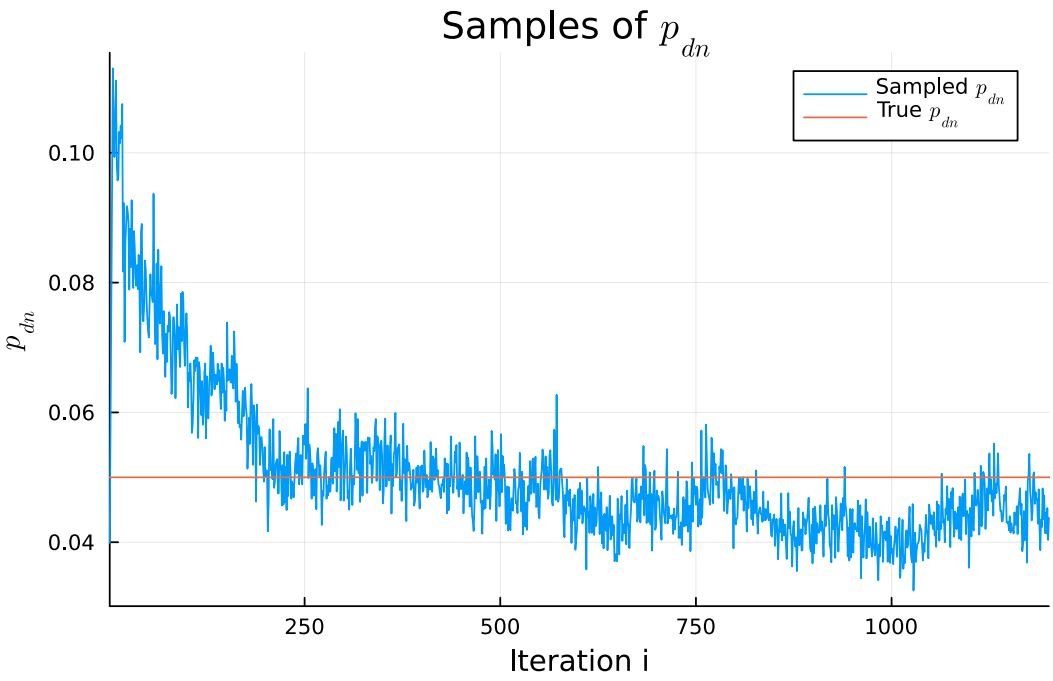

Figure 5.2: Sampled values of the parameter p_{dn} along with its true value. We observe convergence of the samples toward the true value after approximately 250 iterations, consistent with the evolution of $\log(p(\theta, \{z_t^n\}, \{x_t^n\}))$.

In section 4.1.3, an adaptive algorithm was introduced that adjusts the proposal distributions to achieve optimal acceptance probabilities. Figure 5.4 shows the average acceptance probability for p_{dn} computed every 50 steps.

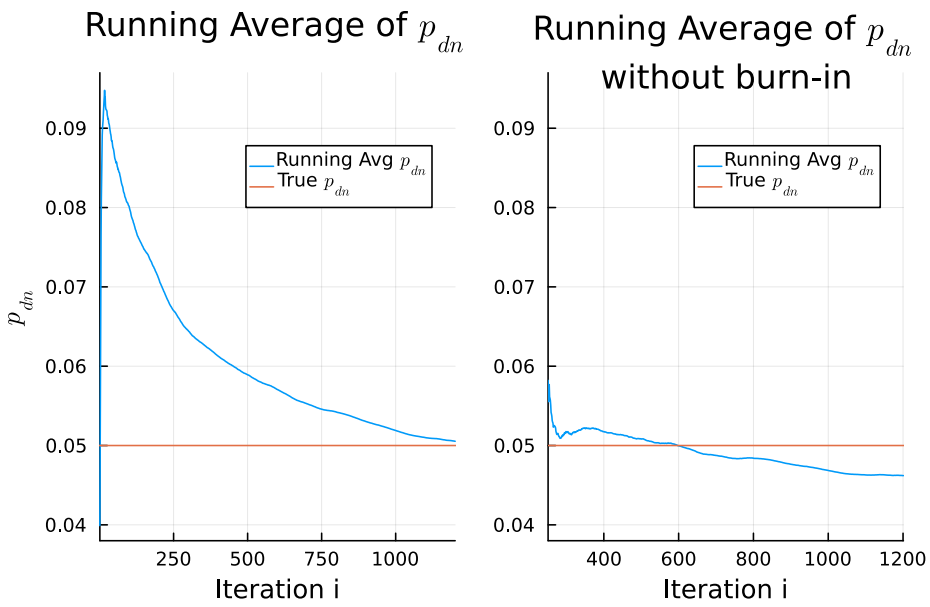

Figure 5.3: Comparison of running averages for p_{dn} computed with and without burn-in. Both averages converge to values close to the true parameter, though the influence of burn-in is evident for an extended duration.

Parameters θ	True Value	Sample Mean	Sample Standard Deviation
A	0.03	0.0301	0.00263
γ	0.001	$9.65 \cdot 10^{-4}$	$5.69 \cdot 10^{-5}$
p_{di}	0.03	0.0297	$6.11 \cdot 10^{-4}$
p_{dn}	0.05	0.0440	0.00395

Table 5.1: Summary of MCMC estimates for the HMM model parameters. The table lists the true value, sample mean, and sample standard deviation (computed from the final 600 samples) for each parameter, highlighting the precision of the estimates.

The figure demonstrates that after just a few sweeps, these probabilities converge near the target value. For the other parameters, similar trends hold.

Run Analysis: Correlations

To study correlations between the sampled model parameters, we consider the plots in figure 5.5, which are based on the last 600 samples of the algorithm. For every individual parameter, a histogram is shown at the upper or right edge of the figure. For each pair of parameters, we display a scatter plot with an overlaid trend line and use colour coding to indicate the correlation: blue for positive, red for negative, and yellow for neutral. Most parameter pairs exhibit only weak correlations, with the notable exception of A and γ, which show a strong positive correlation. This is consistent with the model where, for nozzle n, the probability of transitioning from a normal to a dusty state is given by $A\exp(-\gamma \cdot n)$. To keep this probability approximately constant, an increase in A must be accompanied by an increase in γ.

Additionally, one can observe a slight negative correlation between A and p_{dn}, as well as a slight positive correlation between p_{di} and p_{dn}. In practice, when A is slightly overestimated, both p_{dn} and, consequently (via their positive correlation), p_{di} tend to be underestimated. According to the model described in section 3.1, if the algorithm estimates that a greater number of nozzles are in the dusty state (i.e., high A accompanied by small p_{dn}), it predicts fewer nozzles in the dry-in state (resulting in a smaller p_{di}). This is logical since both states yield the same observational outcome.

Heuristic Analysis: Iterations & Runtime

We now examine the convergence and runtime characteristics of the MCMC algorithm applied to the HMM model over various problem sizes using a heuristic analysis. In figure 5.6, we show the running averages of $\log\big(p(\theta, \{z_t^n\}, \{x_t^n\})\big)$ computed for different values of N and T, with each series offset to begin at 0. Notably, despite variations in N and T, the algorithm converges in approximately the same number of iterations.

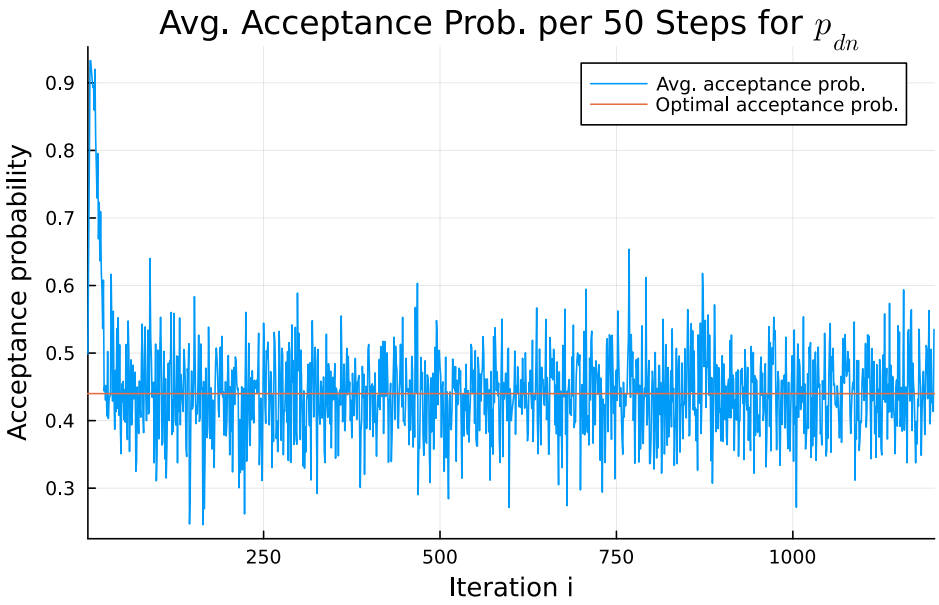

Figure 5.4: Evolution of the average acceptance probability for p_{dn}, computed every 50 steps. The plot shows that the acceptance probabilities quickly converge to the target value, ensuring efficient adaptive sampling.

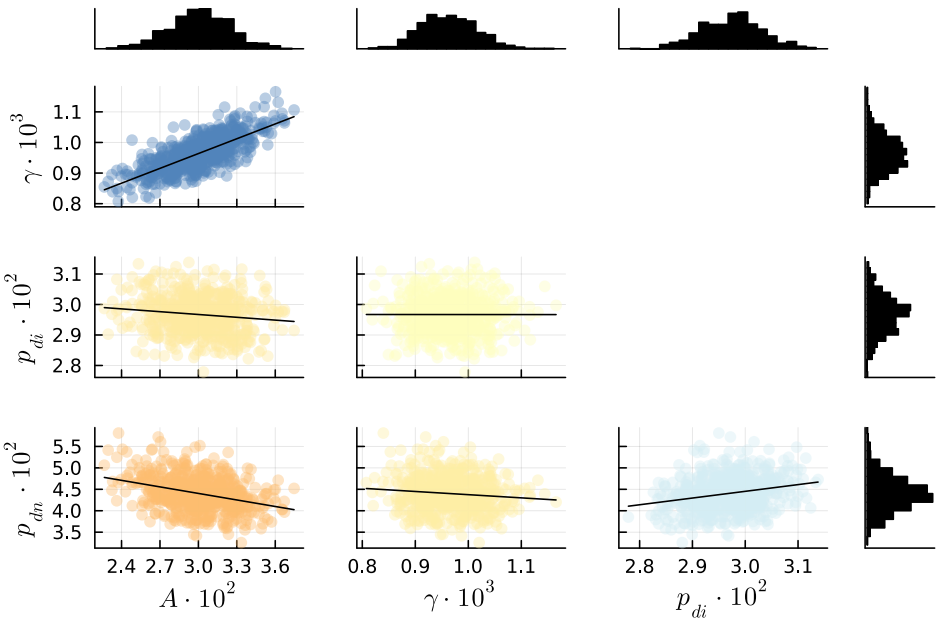

Figure 5.5: Corner plot illustrating pairwise correlations between the sampled model parameters from the HMM model. Panels on the top and right edge display histograms of individual parameters, while the other panels show scatter plots with overlaid trend lines. Blue indicates positive correlation, red indicates negative correlation, and yellow denotes neutrality. Notably, the strong positive correlation between A and γ is clearly visible. In addition, a slight negative correlation between A and p_{dn} is visible, as well as a slight positive correlation between p_{di} and p_{dn}.

In figure 5.7, we plot the algorithm's runtime for various values of N and T; the exact runtime values are listed in table 5.2. For both settings of T, the runtime scales approximately linearly with N. Although the data do not clearly reveal a difference in the slopes for $T = 15$ versus $T = 30$, it is evident that the runtime is higher when $T = 30$. However, the increase is limited, which can be attributed to the specifics of the HMM model described in chapter 3. In particular, if during the Gibbs sweeps a sample $z_t^n = 2$ is drawn, then $z_{t'}^n$ remains equal to 2 for all $t' > t$ since 2 represents an absorbing state. Consequently, there is no need to sample the later latent variables

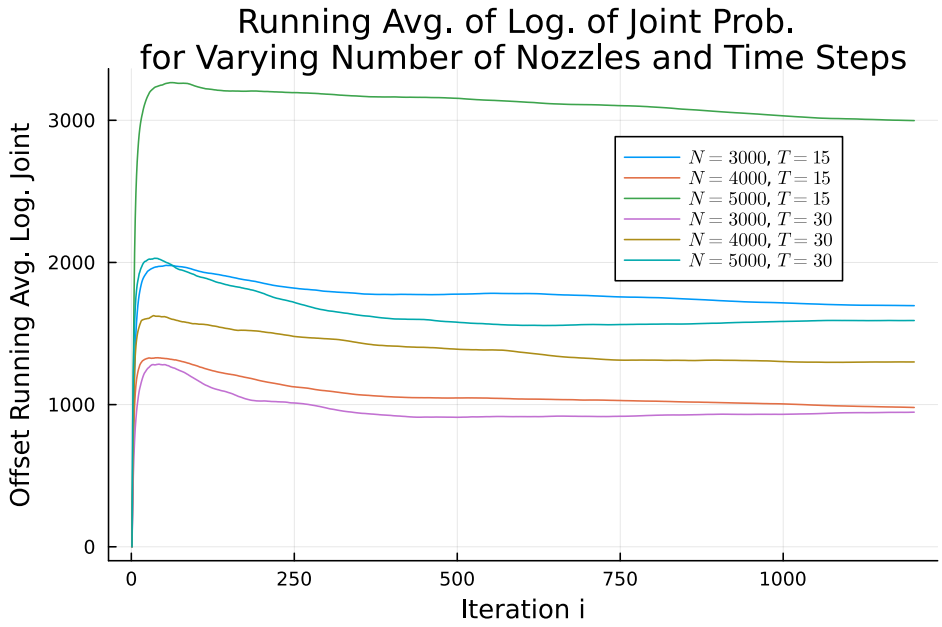

Figure 5.6: Running averages of $\log\big(p(\theta, \{z_t^n\}, \{x_t^n\})\big)$ over MCMC iterations for varying values of N and T, with each series offset to begin at 0. The convergence trends are similar across the different model sizes.

$z_{t'}^n$, which reduces the runtime. For models without absorbing states, the increase in runtime will be greater, with the Gibbs sweeps requiring twice as many steps.

N	T	Runtime (s)
3000	15	94.251
4000	15	111.845
5000	15	130.449
3000	30	102.578
4000	30	124.372
5000	30	149.877

Table 5.2: Measured runtimes (in seconds) for the MCMC algorithm applied to the HMM model, for various combinations of N and T. The table illustrates the approximately linear scaling with N and the increased computational cost for higher T.

5.1.2 Nearest Neighbour Model

Run Analysis: Convergence and Outcomes

We now consider the MCMC algorithm with adaptive proposal distributions defined in section 4.1, for the nearest neighbour model from section 3.3. As before, we set $N = 5000$ and $T = 30$. We set the number of sweeps to 1500 with $N_{\text{Gibbs}} = 10$ and $N_{\text{MwG}} = 50$. Similar to the previous section, we take $\delta(n) = \min(0.1, n^{-1/2})$ to modify the proposal distributions after every sweep as described in section 4.1.3.

In figure 5.8, we plot the logarithm of the joint probability at each sweep. Convergence is indicated by the stabilisation of this quantity after approximately 700 sweeps. This convergence period is notably longer than that observed for the previous model, reflecting the increased complexity of the nearest neighbour model. Note that a suboptimal initialisation of the latent variables $\{z_t^n\}$ may require even more sweeps to achieve convergence.

In figure 5.9 the samples of p_{dn} produced by the algorithm are displayed alongside its true value. In line with our earlier observations, the samples remain close to the true value after approximately 700 sweeps. In figure 5.10 the running averages of the p_{dn} samples are presented both with and without a 700-sweep burn-in period. Both plots clearly demonstrate convergence towards the true value, though the effect of the burn-in period is noticeable. Similar analyses can be carried out for the other parameters. Finally, table 5.3 summarises the model parameter estimates, computed as the sample means and standard deviations of the final 800 samples from the algorithm. It is evident that these estimates are very close to the true values and are obtained with good precision.

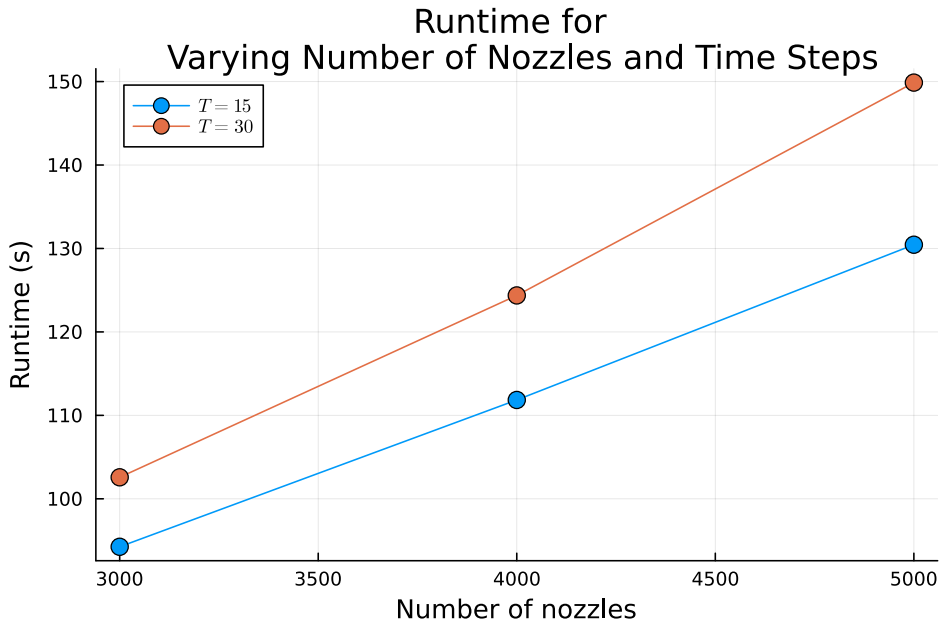

Figure 5.7: Runtime of the MCMC algorithm for the HMM model plotted against different values of N and T. For a fixed T, the runtime scales approximately linearly with N, and larger T values yield higher runtimes.

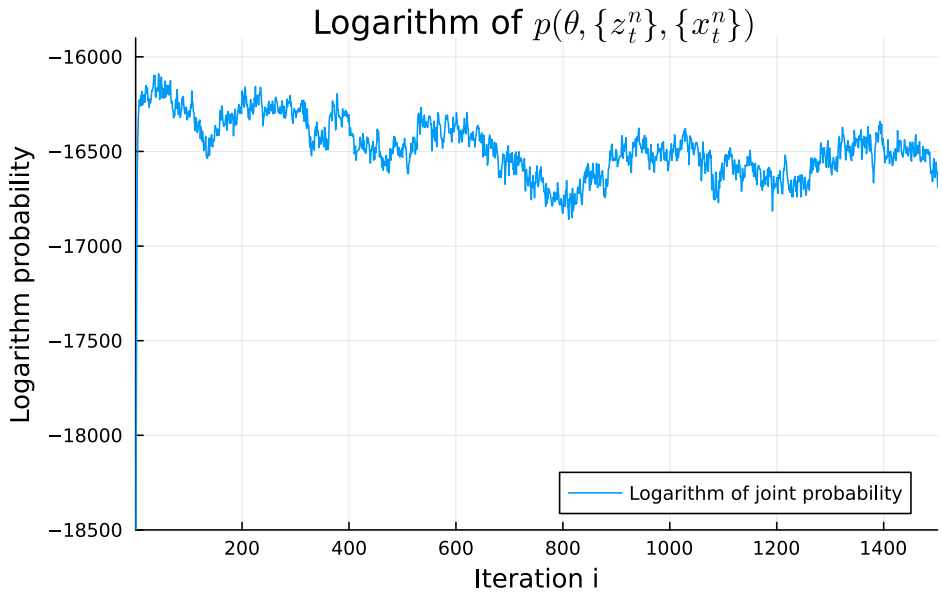

Figure 5.8: Logarithm of the joint probability versus sweep number for the nearest neighbour model. The plot shows stabilisation after approximately 700 sweeps, indicating convergence of the MCMC algorithm.

In section 4.1.3, an adaptive algorithm was introduced which adjusts the proposal distributions to achieve optimised acceptance probabilities. As with the previous model, we examine the average acceptance probability for p_{dn}, computed at intervals of 50 sweeps, as depicted in figure 5.11. The figure demonstrates that the acceptance probabilities rapidly converge to the optimal value.

Run Analysis: Correlations

To examine correlations among the sampled model parameters, we consider the plots in figure 5.12, which are based on the final 800 samples from the algorithm. All parameters have been rescaled so that they are centred around 10. For each individual parameter, a histogram is displayed along the top or right-hand edge of the figure,

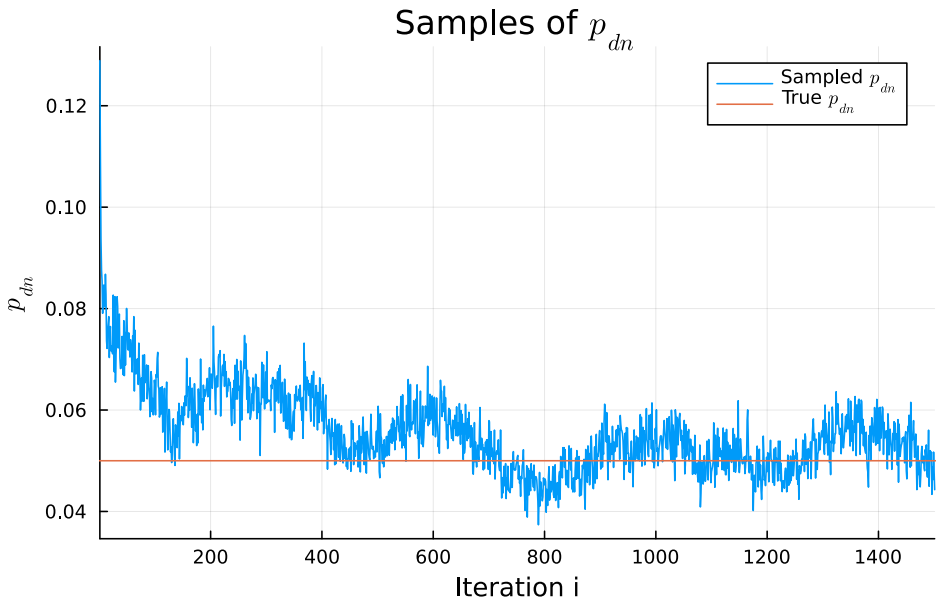

Figure 5.9: Samples of p_{dn} in the nearest neighbour model. The MCMC samples remain close to the true value after about 700 sweeps.

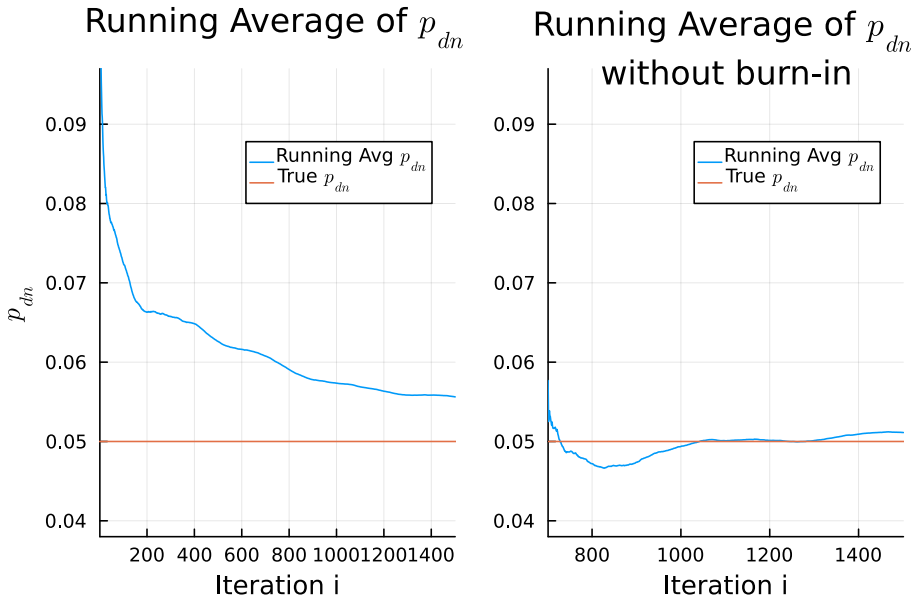

Figure 5.10: Running averages of p_{dn} in the nearest neighbour model. The graph compares the running averages computed with and without a burn-in period of 700 sweeps, highlighting the improved convergence when initial samples are discarded.

while for every pair of parameters a scatter plot with an overlaid trend line is provided. Colour coding indicates the nature of the correlation — blue for a positive correlation, red for a negative correlation, and yellow for a neutral correlation. As before, most parameter pairs exhibit only weak correlations, with the notable exception of A and γ, which display a strong positive correlation. This is consistent with the model, where, for nozzle n, the probability of transitioning from a normal to a dusty state is given by $A\exp(-\gamma \cdot n)$. To keep this probability approximately constant, an increase in A must be accompanied by a corresponding increase in γ.

Furthermore, a slight negative correlation is observed between A and p_{dn}, along with modest positive correlations between p_{di0}, p_{di1}, p_{di2} and p_{dn}. In other words, when A is marginally overestimated, p_{dn} tends to be underestimated, and consequently, so too are p_{di0}, p_{di1} and p_{di2} (owing to their positive associations). According to the

Parameters θ	True Value	Sample Mean	Sample Standard Deviation
A	0.03	0.0301	0.00236
γ	0.001	$9.71 \cdot 10^{-4}$	$6.01 \cdot 10^{-5}$
p_{di0}	0.02	0.0195	$5.79 \cdot 10^{-4}$
p_{di1}	0.035	0.0340	0.00116
p_{di2}	0.05	0.0516	0.00339
p_{dn}	0.05	0.0511	0.00465

Table 5.3: Summary of MCMC estimates for the nearest neighbour model parameters. The table lists the true value, sample mean, and sample standard deviation (computed from the final 800 samples) for each parameter, highlighting the precision of the estimates.

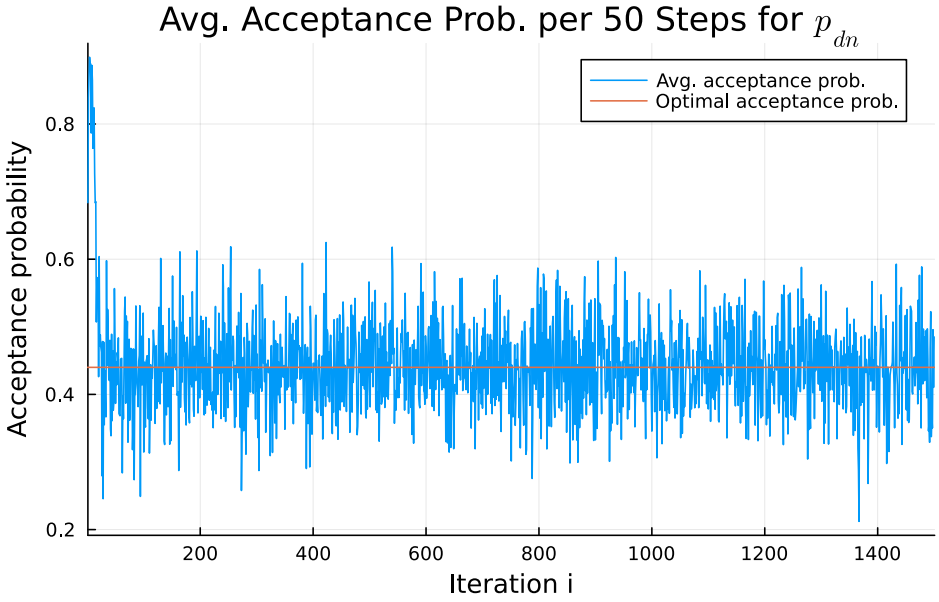

Figure 5.11: Average acceptance probability for p_{dn} in the nearest neighbour model. The plot displays the average acceptance probability computed every 50 sweeps, demonstrating that the adaptive algorithm rapidly achieves the optimised acceptance rate.

nearest neighbour model described in section 3.3, if the algorithm infers that a greater number of nozzles are in the dusty state (i.e. a high A paired with a low p_{dn}), it in turn predicts fewer nozzles in the dry-in state, leading to lower estimates for p_{di0}, p_{di1} and p_{di2}. This outcome is entirely sensible since both states produce the same observational result.

Heuristic Analysis: Iterations & Runtime

We now examine the convergence and runtime characteristics of the MCMC algorithm when applied to the nearest neighbour model across a range of model sizes using a heuristic analysis. In figure 5.13 the running averages of the logarithm of the joint probability, $\log\Big(p(\theta, \{z_t^n\}, \{x_t^n\})\Big)$, offset to start at 0, are plotted for various model configurations as an indicator of convergence. Overall, the number of iterations required to reach convergence appears to be similar across the different problem sizes.

In figure 5.14 we present the algorithm's runtime for various values of N and T; the corresponding runtime values are detailed in table 5.4. As before, the runtime scales linearly with the number of nozzles. In addition, a slight increase in runtime is observed for $T = 30$ compared to $T = 15$. This increase is only modest because the model incorporates an absorbing state (dry-in). As explained in section 4.1, once a nozzle is sampled to enter the dry-in state during the Gibbs steps at a given time step, states at subsequent time steps are fixed and do not require further sampling, thereby mitigating the overall computational cost.

Compared to the HMM model, we observe a substantially higher runtime. It should be noted that in this instance the algorithm was run for 1500 sweeps rather than 1200, which partially accounts for the difference. Nevertheless, the primary reason for the increase is that the transition probabilities now depend on two additional variables,

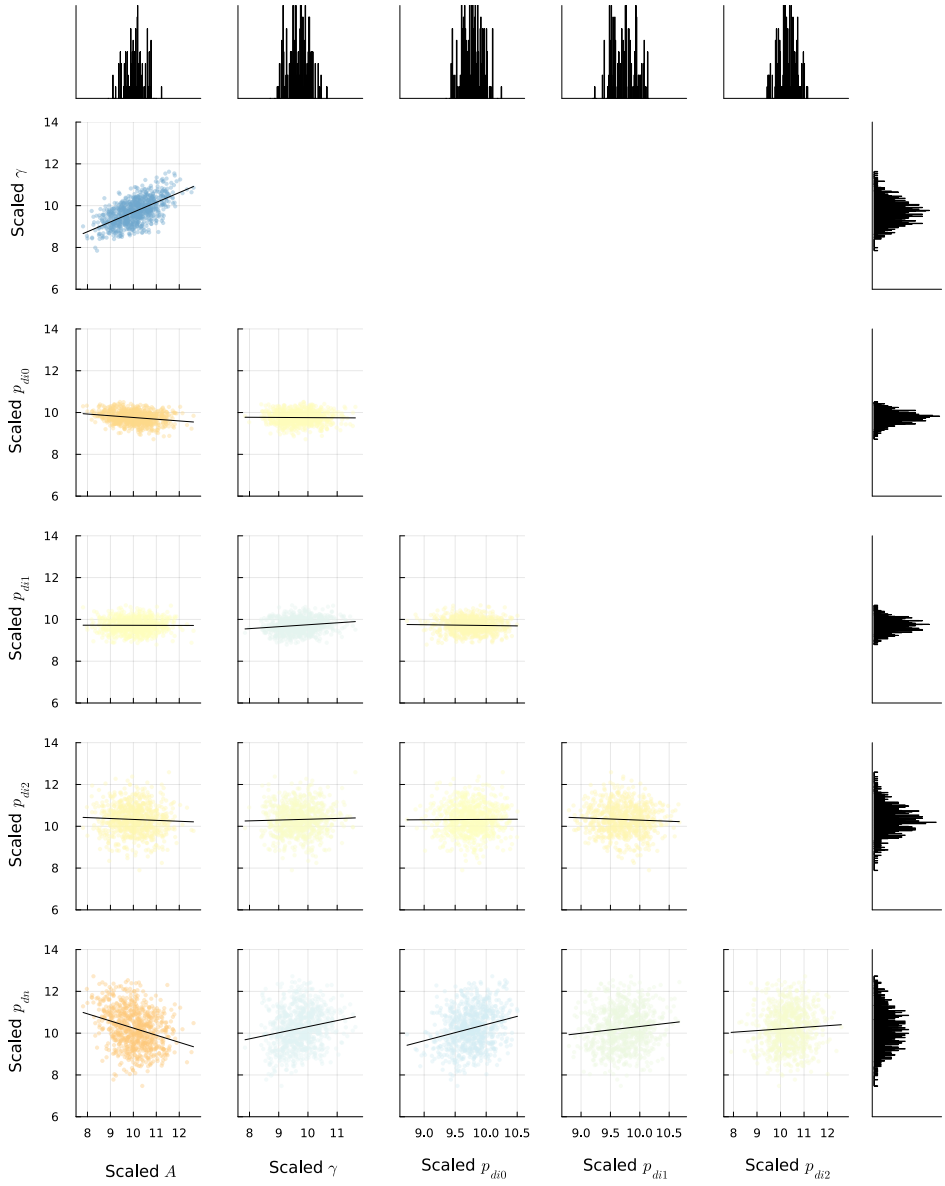

Figure 5.12: Corner plot for the model parameters of the nearest neighbour model, based on the final 800 MCMC samples. The marginal distributions (displayed as histograms along the top and right edges) correspond to parameters rescaled to be centred around 10, while each off-diagonal panel presents a scatter plot with a fitted trend line. Colour coding represents the correlation type—blue for positive, red for negative, and yellow for neutral. Notably, the strong positive correlation between A and γ aligns with the model prediction that an increase in A must be offset by a corresponding rise in γ to maintain an approximately constant transition probability.

which slows down the Gibbs steps.

5.1.3 More Conventional Models

In section 3.5, we considered more conventional models with non-parametrised transition probabilities. As before, we assumed either independent HMMs or included nearest neighbour interactions. We performed a trial run for both configurations.

In the case of independent HMMs, the true transition matrix for the trial run is defined as follows,

$$\mathcal{T} = \begin{pmatrix} 0.85 & 0.1 & 0.05 \\ 0.2 & 0.7 & 0.1 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5.1}$$

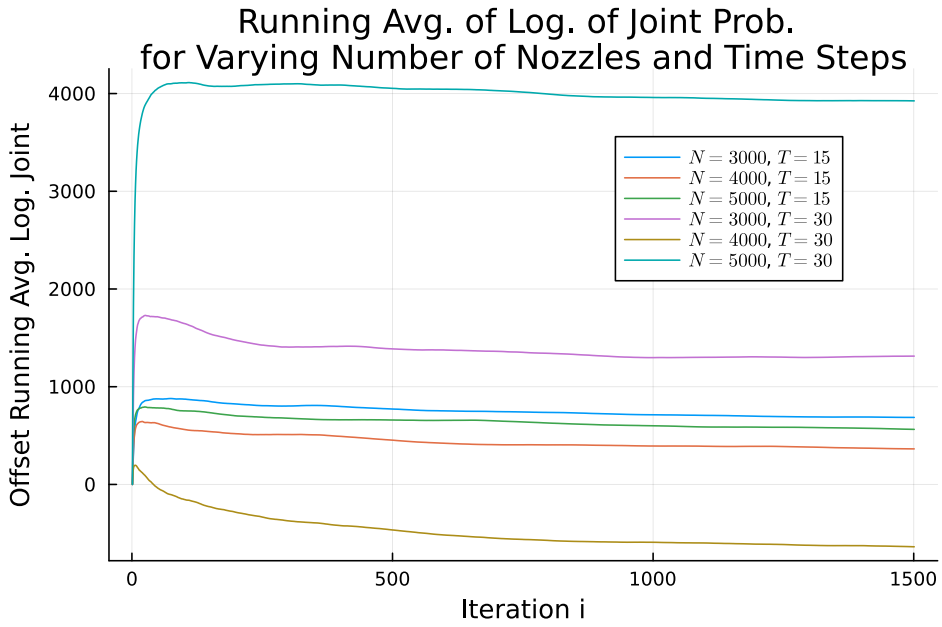

Figure 5.13: Running averages of the logarithm of the joint probability for various model configurations of the nearest neighbour model. The figure demonstrates that the number of iterations required for convergence is broadly comparable across different model sizes.

N	T	Runtime (s)
3000	15	292.027
4000	15	357.619
5000	15	428.770
3000	30	300.209
4000	30	365.901
5000	30	439.529

Table 5.4: Measured runtimes (in seconds) for the MCMC algorithm applied to the nearest neighbour model, for various combinations of N and T. The number of sweeps equals 1500; for the HMM model it was 1200. The table illustrates the approximately linear scaling with N and the slightly increased computational cost for higher T.

From our run of the sampling algorithm, outlined in section 4.1.4, the following estimates $\pm$ standard deviations were obtained,

$$\mathcal{T} = \begin{pmatrix} 0.849 \pm 9.53 \cdot 10^{-4} & 0.0981 \pm 0.0137 & 0.0528 \pm 0.0136 \\ 0.211 \pm 0.0285 & 0.699 \pm 2.61 \cdot 10^{-3} & 0.0902 \pm 0.0285 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5.2}$$

Under nearest neighbour coupling, we define the following set of transition probabilities, to which all other transition probabilities are equal.

$$\begin{aligned} M_0^{(00)} &= \begin{pmatrix} 0.85 & 0.1 & 0.05 \end{pmatrix}, & M_1^{(00)} &= \begin{pmatrix} 0.25 & 0.7 & 0.05 \end{pmatrix}, \\ M_0^{(02)} &= \begin{pmatrix} 0.8 & 0.1 & 0.1 \end{pmatrix}, & M_1^{(02)} &= \begin{pmatrix} 0.25 & 0.65 & 0.1 \end{pmatrix}, \\ M_0^{(22)} &= \begin{pmatrix} 0.75 & 0.1 & 0.15 \end{pmatrix}, & M_1^{(22)} &= \begin{pmatrix} 0.25 & 0.6 & 0.15 \end{pmatrix}. \end{aligned} \tag{5.3}$$

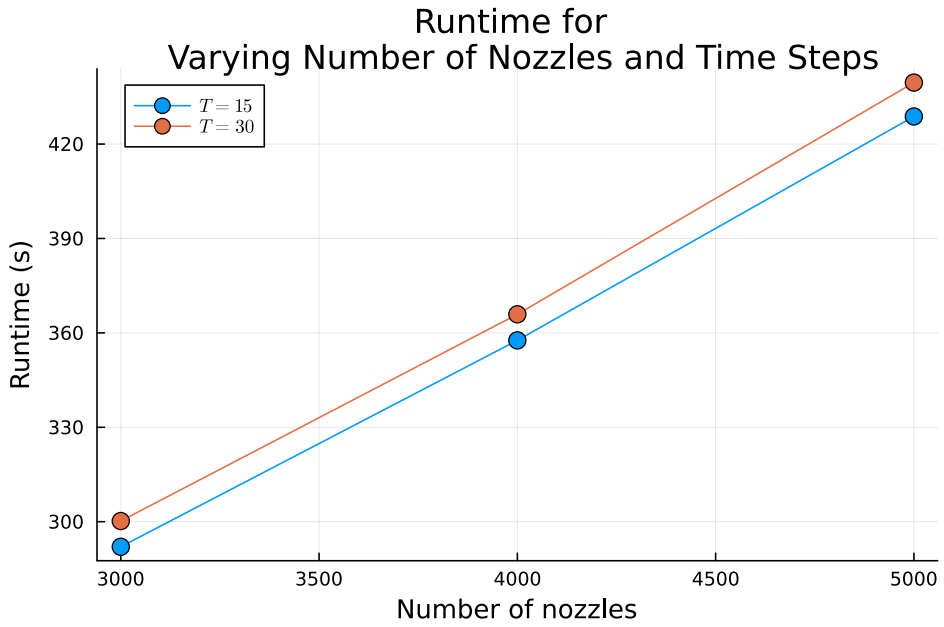

Figure 5.14: Runtimes of the MCMC algorithm for different configurations of the nearest neighbour model. The plot shows runtime versus the number of nozzles N and time steps T. The runtimes appear to be linear in N. The increase for $T = 30$ compared to $T = 15$ is slight, owing to the absorbing state. Overall, the runtimes are higher than for the HMM model. This is mainly due to the more complex transition probabilities that slow the Gibbs steps. Note, however, that 1500 sweeps were used instead of the 1200 sweeps for the HMM model.

From the samples generated by our MCMC run, we derive the following estimates.

$$\begin{aligned}
M_0^{(00)} &= \left(0.849 \pm 1.67 \cdot 10^{-3} \quad 0.101 \pm 4.94 \cdot 10^{-3} \quad 0.0505 \pm 4.74 \cdot 10^{-3}\right), \\
M_1^{(00)} &= \left(0.253 \pm 0.0124 \quad 0.697 \pm 5.37 \cdot 10^{-3} \quad 0.0502 \pm 0.0117\right), \\
M_0^{(02)} &= \left(0.801 \pm 2.85 \cdot 10^{-3} \quad 0.0986 \pm 6.62 \cdot 10^{-3} \quad 0.101 \pm 5.52 \cdot 10^{-3}\right), \\
M_1^{(02)} &= \left(0.257 \pm 0.0168 \quad 0.645 \pm 6.39 \cdot 10^{-3} \quad 0.0978 \pm 0.0165\right), \\
M_0^{(22)} &= \left(0.752 \pm 4.14 \cdot 10^{-3} \quad 0.0948 \pm 9.55 \cdot 10^{-3} \quad 0.153 \pm 9.10 \cdot 10^{-3}\right), \\
M_1^{(22)} &= \left(0.270 \pm 0.0245 \quad 0.598 \pm 9.40 \cdot 10^{-3} \quad 0.132 \pm 0.0241\right).
\end{aligned} \tag{5.4}$$

As the results illustrate, both configurations yield accurate estimates with satisfactory precision.

5.2 VI Results

In this section, we apply both the model-specific VI algorithm as well as the ADVI approach exclusively to the HMM model. For both algorithms, we adopt the same structure used in the previous section. First, we analyse a single representative run, recording iterations to convergence and parameter estimates along with their accuracy. We then assess convergence speed across various algorithm settings.

5.2.1 Model-Specific VI

Run Analysis: Convergence and Outcomes

For the HMM model described in section 3.1, we applied the model specific VI algorithm from section 4.2.1. In figure 5.15, we see the evolution of the ELBO for 25 iterations of the algorithm. Initially, the ELBO rises sharply before gradually approaching convergence. It is important to note that the number of iterations required for convergence is only weakly dependent on the initialisation.

In figure 5.16, we present the estimated model parameter p_{dn} at each iteration alongside its true value. The estimate corresponds to the mean of the variational distribution $q(p_{dn})$, while the ribbon represents its standard deviation. Over time, the estimate converges to a value close to the true parameter, accompanied by a decreasing

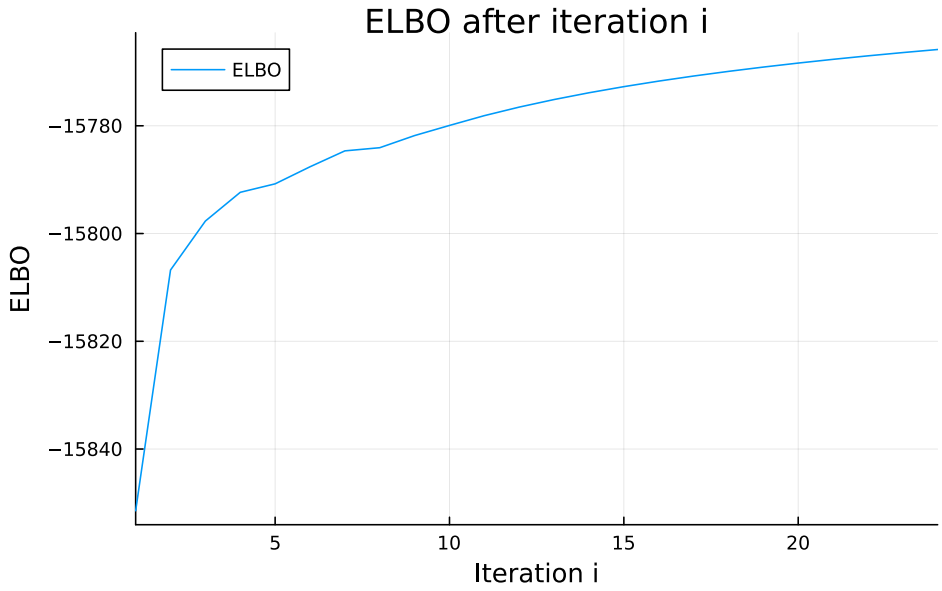

Figure 5.15: Evolution of the evidence lower bound (ELBO) over 25 iterations of the model-specific variational inference algorithm for the HMM model. The initial rapid increase followed by gradual convergence indicates that the optimiser quickly improves the variational approximation before stabilising.

standard deviation. Similar trends hold for the other parameters. The final estimates for all parameters, shown in table 5.5, demonstrate a strong agreement with their true values.

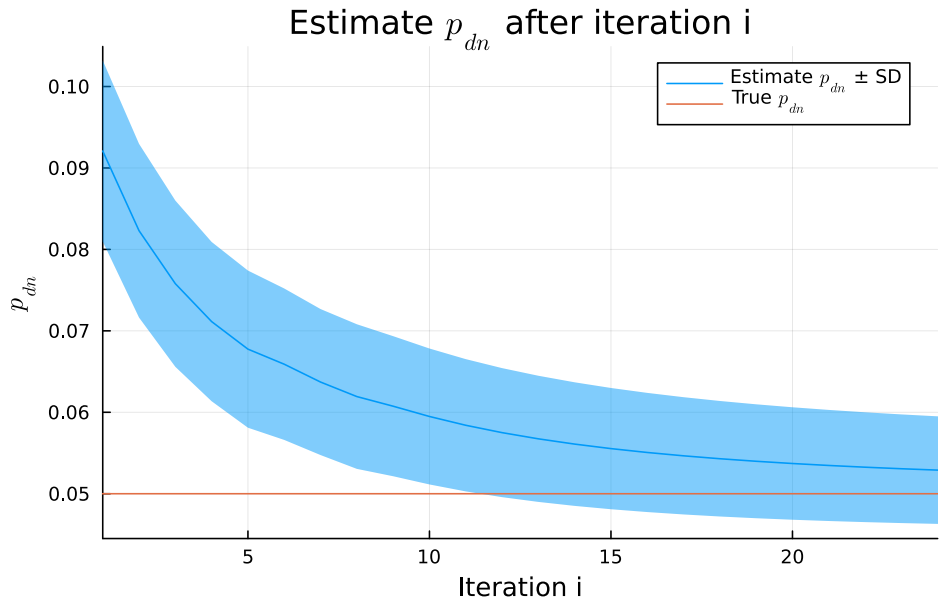

Figure 5.16: Estimates of the parameter p_{dn} at each iteration of the model-specific VI algorithm. The solid line represents the mean of the variational distribution $q(p_{dn})$, and the shaded ribbon denotes its standard deviation. The plot demonstrates convergence toward the true value with decreasing uncertainty.

Heuristic Analysis: Iterations & Runtime

We now examine the convergence and runtime characteristics of the model-specific variational inference algorithm for the HMM model. In figure 5.17, we present the evolution of the ELBO for varying values of N and T,

Parameters θ	True Value	Mean q	Standard Deviation q
A	0.03	0.0288	0.00378
γ	0.001	$9.30 \cdot 10^{-4}$	$4.62 \cdot 10^{-5}$
p_{di}	0.03	0.0314	0.00305
p_{dn}	0.05	0.0527	0.00654

Table 5.5: Final estimates and uncertainties of the HMM model parameters obtained via the model-specific variational inference algorithm. This table compares the true values with the mean and standard deviation of the variational posterior distributions, demonstrating good agreement and reasonable accuracy of this VI method.

with each series offset to start at 0. Notably, the algorithm converges in roughly the same number of iterations regardless of the model size.

In figure 5.18, we illustrate the algorithm's runtime for the different combinations of N and T; the exact runtime values are summarised in table 5.6. The data again suggest that—for a fixed T—the runtime scales linearly with N. In particular, the runtime for $T = 30$ is approximately twice that for $T = 15$. Although the runtimes given in table 5.6 are longer than those of the MCMC algorithm, the MCMC results here involve strong initialisations. In scenarios where only poor MCMC initialisations can be provided — leading to significantly longer convergence times — the rapid improvement of the ELBO in figure 5.17 indicates that this VI algorithm could make the faster option.

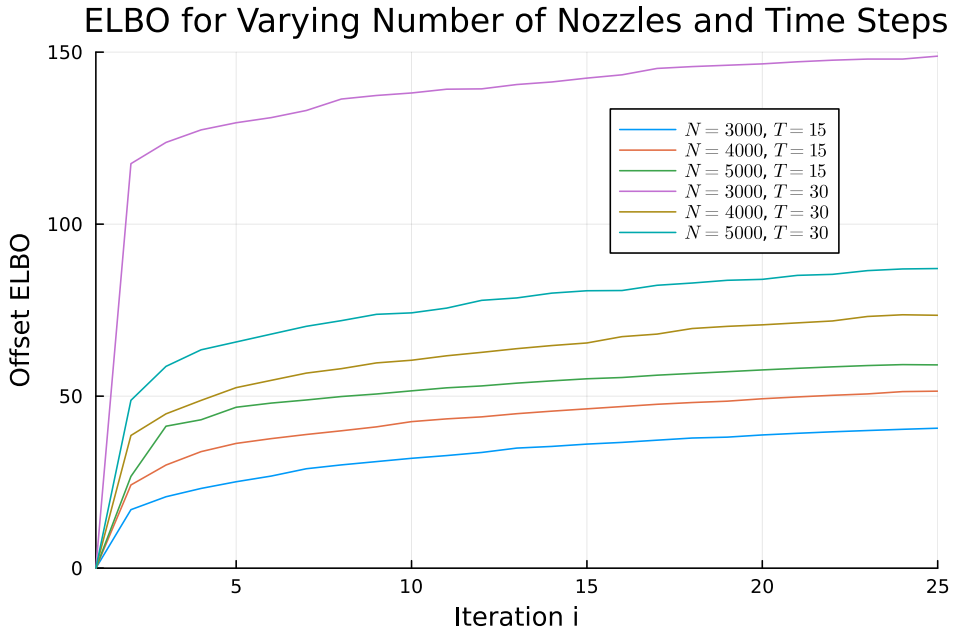

Figure 5.17: Evolution of the ELBO for the HMM model under the model-specific variational inference algorithm for different values of N and T. Despite the different model sizes, the convergence behaviour in terms of ELBO stabilisation is similar across the tested configurations.

N	T	Runtime (s)
3000	15	295.806
4000	15	405.267
5000	15	526.656
3000	30	626.801
4000	30	870.494
5000	30	1115.82

Table 5.6: Tabulated runtime (in seconds) for the model-specific variational inference algorithm for the HMM model under various values of N and T.

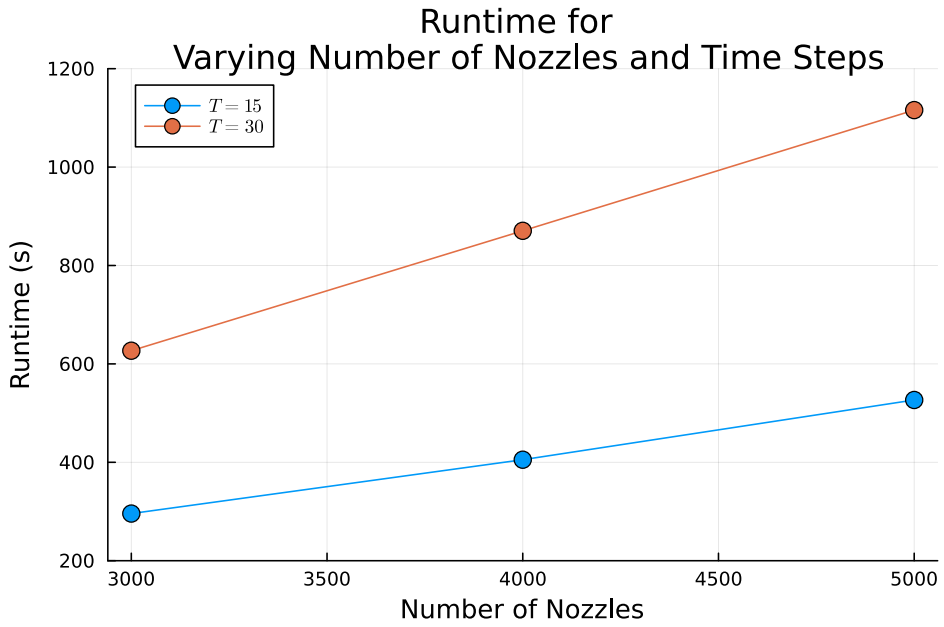

Figure 5.18: Runtime analysis of the model-specific variational inference algorithm for the HMM model. The plot suggests that for a fixed T, the runtime increases approximately linearly with N, and that doubling T roughly doubles the runtime.

5.2.2 ADVI Approach

Run Analysis: Convergence and Outcomes

For the HMM model described in section 3.1, we adopt the automatic differentiation variational inference method detailed in section 4.2.2. As before, we set $N = 5000$ and $T = 30$. figure 5.19 shows the evolution of the evidence lower bound over 25 iterations. Initially, the ELBO increases steeply before gradually approaching convergence.

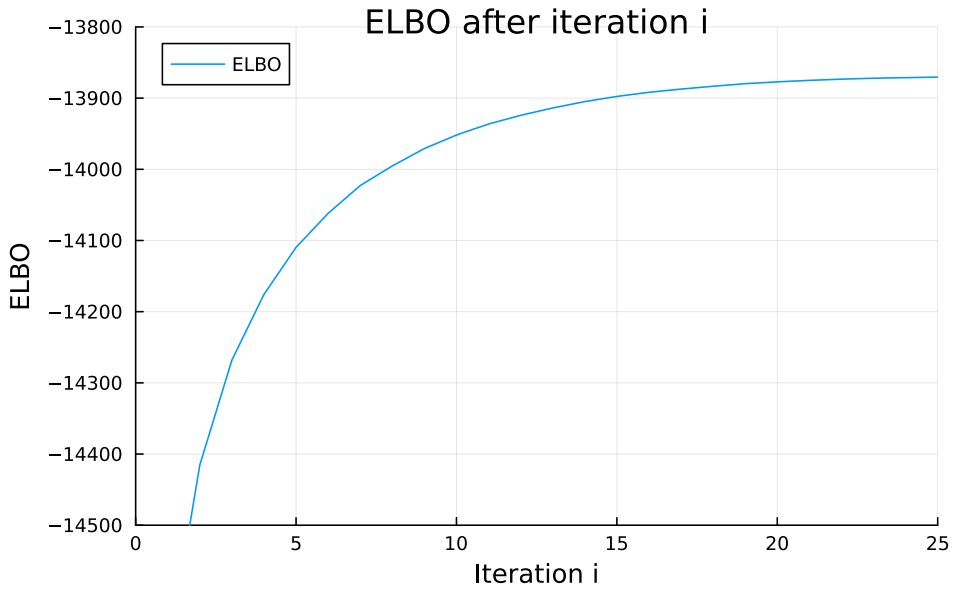

Figure 5.19: Evolution of the ELBO during ADVI for the HMM model. The figure shows a steep initial increase followed by gradual levelling off over 25 iterations, indicating the stabilisation of the variational approximation.

In figure 5.20 we present the estimated mean and standard deviation of the variational distribution $q(p_{dn})$ at each iteration. These estimates are obtained by sampling from the distribution $\zeta = T(\theta)$, see section 4.2.2. The mean converges steadily and the standard deviation decreases as the number of iterations increases. Similar trends hold

for the other parameters. Table 5.7 lists the estimates of the model parameters along with their standard deviations; overall, the estimates are in close agreement with the true values. Notably, the standard deviations produced by the ADVI algorithm are smaller than those obtained using the model-specific VI approach.

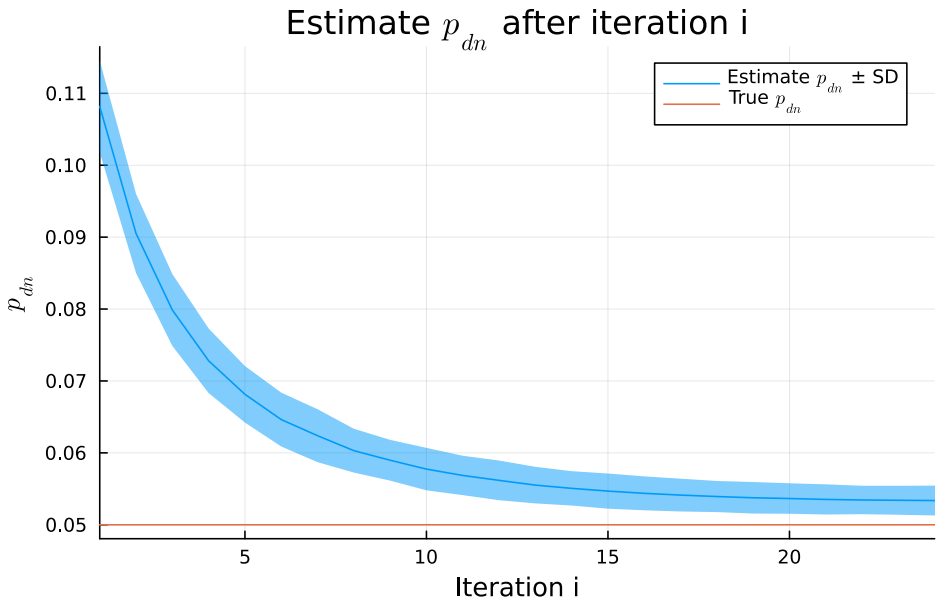

Figure 5.20: Estimated evolution of the mean and standard deviation of $q(p_{dn})$ over ADVI iterations. The estimated mean converges towards the true value, while the standard deviation decreases, indicating reduced uncertainty. These estimates are obtained by sampling from the transformed space $\zeta = T(\theta)$, see section 4.2.2.

Parameters θ	True Value	Sampled Mean q	Sampled Standard Deviation q
A	0.03	0.0347	$6.37 \cdot 10^{-4}$
γ	0.001	0.00111	$2.15 \cdot 10^{-5}$
p_{di}	0.03	0.0303	$3.14 \cdot 10^{-4}$
p_{dn}	0.05	0.0527	0.00211

Table 5.7: Final estimates and uncertainties of the HMM model parameters obtained via the automatic differentiation variational inference algorithm. This table compares the true values with the mean and standard deviation of the variational posterior distributions, demonstrating good agreement and accuracy of this VI method.

Heuristic Analysis: Iterations & Runtime

We now examine the convergence and runtime characteristics of the automatic differentiation VI algorithm for the HMM model. In figure 5.21, we present the evolution of the ELBO for varying values of N and T, with each series offset to start at 0. Notably, the algorithm converges in roughly the same number of iterations regardless of the model size.

In figure 5.22 we illustrate the algorithm's runtime for different combinations of N and T; these exact values are summarised in table 5.8. The data indicate that, for a fixed T, the runtime scales linearly with N; notably, the runtime for $T = 30$ is slightly more than double that for $T = 15$. The ADVI approach exhibits somewhat higher runtimes than the model-specific VI algorithm. As before, although the runtimes in table 5.8 exceed those of the MCMC algorithm, it should be noted that the MCMC results were obtained using good initialisations for $\{z_t^n\}$. In practice, when only poor initialisations are available—leading to much longer convergence times—the rapid improvement of the ELBO, see figure 5.21, suggests this VI algorithm may be the faster option.

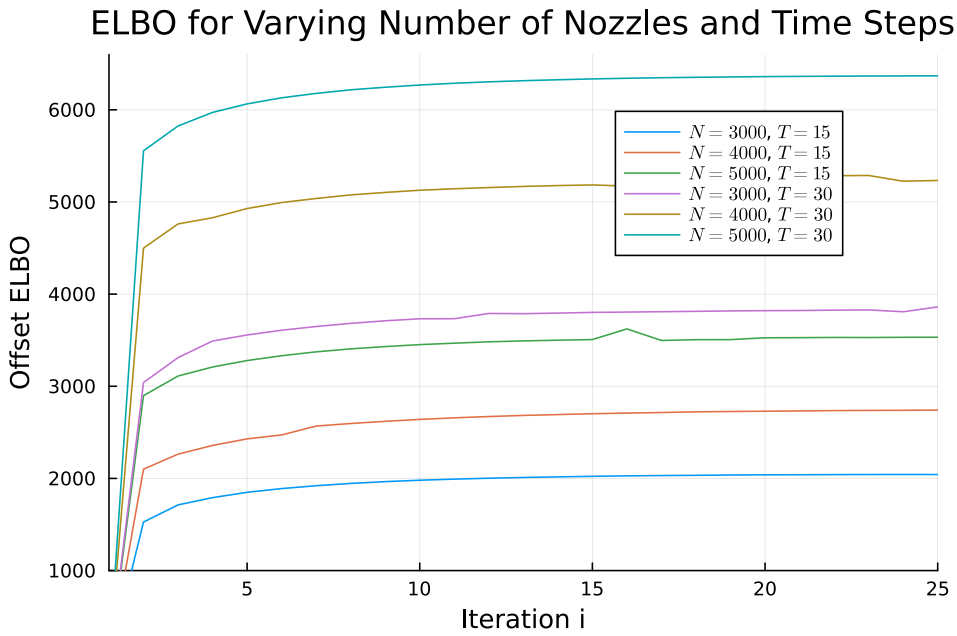

Figure 5.21: Evolution of the ELBO for the HMM model under the automatic differentiation variational inference algorithm for different values of N and T. Despite the different model sizes, the convergence behaviour in terms of ELBO stabilisation is similar across the tested configurations.

N	T	Runtime (s)
3000	15	333.996
4000	15	445.765
5000	15	567.032
3000	30	704.972
4000	30	950.163
5000	30	1189.68

Table 5.8: Tabulated runtime (in seconds) for the ADVI algorithm for the HMM model under various values of N and T.

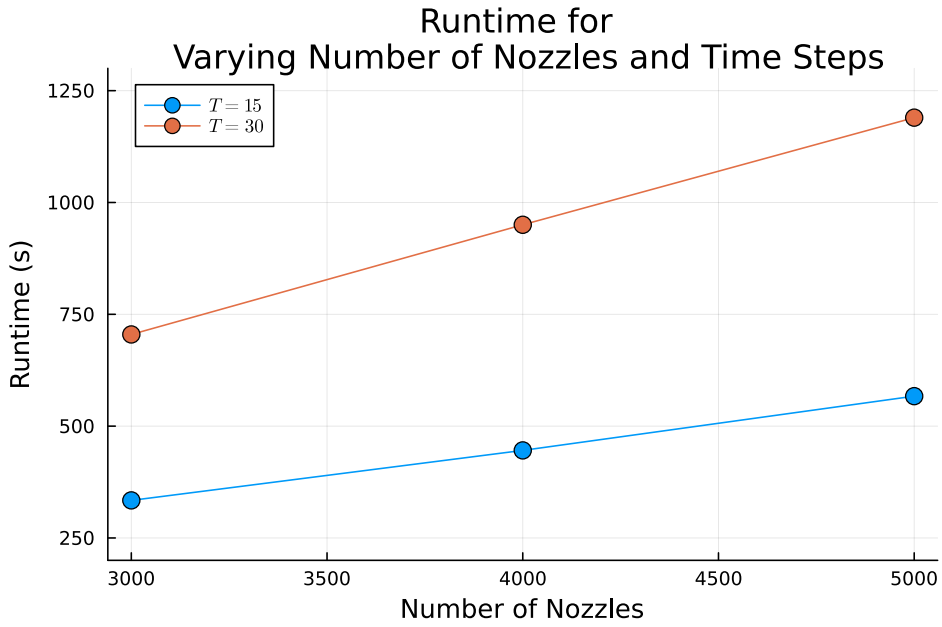

Figure 5.22: Runtime of the ADVI algorithm for the HMM model as a function of the number of nozzles N and time steps T. For a fixed T, the runtime increases linearly with N — the runtime for $T = 30$ is slightly more than double that for $T = 15$. The ADVI approach incurs marginally longer runtimes than the model-specific VI algorithm.

6

Conclusions and Discussion

In this thesis, we set out to develop Bayesian methodologies for inferring parameters of models representing the printing process of industrial printers. We explored two extensions of the standard independent HMM approach for print nozzles. The first, which we call the HMM model, preserves independent chains but replaces fixed transitions with a low-dimensional, interpretable parametrisation. The second, denoted as the nearest neighbour model, builds on this by coupling adjacent HMMs, introducing spatial interactions between neighbouring nozzles. To infer the model parameters, we formulated an MCMC approach that combines Gibbs sweeps for the latent variables, Metropolis-within-Gibbs sweeps for the parameters, and an adaptive algorithm to tune the proposal distributions. In addition, we presented two variational inference algorithms: one employing model-specific approximations to the ELBO for the parameters, and another using automatic differentiation variational inference to generalise the former approach. Notably, the VI methods are asymptotically biased in contrast to the sampling algorithm. The MCMC method was applied to both models, while the VI algorithms were only employed on the HMM model. We also briefly mentioned stochastic variational inference as a potential means to enhance scalability in some models, though it remains unimplemented in this work.

Both the MCMC algorithm and the two VI methods were implemented for the HMM model. All three approaches produced reliable estimates of the true model parameters. Remarkably, the ADVI method yielded the smallest standard deviations, especially compared to the model-specific approach, and thus demonstrates that, despite its generality, ADVI can match specialised techniques in precision. For the MCMC method, the Metropolis-within-Gibbs acceptance rates stabilised rapidly at their optimal levels, confirming the effectiveness of the adaptive proposal-tuning strategy.

In terms of computational time, the MCMC algorithm markedly outperformed both VI methods on the HMM model. However, its efficiency hinges on a well-chosen initialisation; with only a poor starting point available, the VI routines may reach convergence more swiftly. Between the two VI schemes, the model-specific method ran somewhat faster than the ADVI method. Thus, in our configuration there seems to be a trade-off between runtime and the precision of the resulting estimates.

In our experiments for the HMM model, we varied N (the number of nozzles) and T (the number of time steps) to assess runtime and convergence. Across all three methods, the iteration count required for convergence remained approximately constant, and runtimes scaled linearly with N. When T was doubled, both VI methods saw their runtimes roughly double, whereas the MCMC runtime increased only modestly—most likely reflecting a model-specific characteristic.

When applied to the nearest-neighbour model, MCMC again delivered accurate estimates with satisfactory precision. Adaptive proposal tuning remained effective, and the number of iterations to convergence was approximately invariant under changes in N and T. Runtimes scaled linearly with N and rose only moderately when T was doubled. Thus, the extra model sophistication imposes no convergence penalty, but the absolute runtime is substantially higher than for the independent HMM—so when computational efficiency is paramount, the simpler model may be preferable.

It must be noted that the above analysis was performed on synthetic data; real-world validation is needed to

confirm the effectiveness of the inference methods.

Furthermore, our current evaluation of method performance is rather informal. To obtain a more rigorous measure of divergence between the estimated and true posteriors, one can employ the Wasserstein distance [12]. This metric can be computed numerically from empirical histograms, which follow naturally from the output our algorithms produce. For the variational inference methods, only the transformed-parameter posterior is available in closed form, but we can generate samples of the original parameters and build corresponding histograms. Although the exact posterior is intractable, it can be approximated by running the MCMC sampler for a sufficiently long time and assuming convergence to its stationary distribution, thanks to MCMC's asymptotic guarantees. We can then compute the Wasserstein distance between standard runs of the MCMC and VI algorithms against this "long-chain" reference to quantify their similarity. Efficient libraries for computing the Wasserstein distance exist, for example, the Julia OptimalTransport.jl package [20].

The methodologies considered in this thesis are not confined to printing process models; in principle, they apply to any parametrised, factorisable model with discrete latent variables. Such models are commonplace in machine diagnostics, where they underpin fault detection and diagnosis in high-tech machinery.

When no parametrisation can be formulated or a simpler approach is desired, one may turn to the non-parametrised models discussed in section 3.5. In these models, we place a collection of independent Dirichlet priors on each conditional-probability vector of $\mathcal{T}$. By conjugacy, every such vector admits a Dirichlet posterior. This enables direct sampling, making the MCMC sampler, see section 4.1.4, simpler than in the parametrised case, yet still accurate in recovering the transition probabilities.

While our methods perform well on the models of chapter 3, their scalability to larger, more intricate systems remains to be fully assessed. Should computational costs prove prohibitive, algorithmic adaptations will be required. Future work could investigate sub-sampling MCMC strategies [2] or stochastic variational inference as detailed in section 4.2.3. For example, stochastic gradient Langevin dynamics blends stochastic optimisation with Langevin Monte Carlo [19], but—like most sub-sampling schemes—targets an approximate posterior. In contrast, the sub-sampled Zig-Zag process [4] can sample the true posterior exactly, albeit under stricter convergence criteria. It is important to note, however, that these methods typically assume data independence, which may limit their applicability in the presence of strongly correlated data.

Furthermore, this thesis analysed both a model-specific VI algorithm and a general ADVI algorithm applied to the HMM framework. In cases where specific approximations, see section 4.2.1, are impractical, yet the generality of ADVI is insufficiently targeted, one may turn to the generalised reparametrisation gradient. As the name implies, this method generalises the reparametrisation trick. Under that trick, an auxiliary noise variable ϵ is introduced—drawn from a distribution independent of the variational parameters λ—and the model variables $\theta = T(\epsilon, \lambda)$ are written as an invertible transformation of both λ, and the λ-independent distribution. Consequently, the ELBO can be written as an expectation over ϵ, decoupling the randomness from the variational parameters. Hence, we can interchange differentiation and expectation when computing the gradient of the ELBO with respect to λ. In gradient-based optimisation, we may then approximate the gradient via Monte Carlo sampling, provided the inner derivatives can be evaluated pointwise. Unfortunately, only a handful of distribution families admit a reparametrisation that decouples the variational parameters λ from a λ-independent distribution. The generalised reparametrisation gradient approach instead considers transformations $\theta = T(\epsilon(\lambda), \lambda)$, where ϵ only weakly depends on the variational parameters. As before, this lets us write the derivatives of the ELBO as expectations over derivatives. In doing so, the generalised reparametrisation gradient expands the class of variational distributions for θ, while maintaining a feasible optimisation procedure. In terms of accuracy, the generalised reparametrisation gradient often outperforms ADVI. [17]

In summary, in this thesis we investigated Bayesian parameter inference methods for industrial-printer models that extend the standard independent-HMM framework previously used by TNO. The greater complexity of these extensions makes the previous inference technique used by TNO infeasible, motivating the algorithms presented here. Tested on the relatively simple models of chapter 3, the presented methods recover parameters accurately and run in practical time. A more formal analysis—e.g. via Wasserstein-distance comparisons—would offer an improved assessment of the algorithms. Crucially, validation on larger models with more latent states and on real-world data remains to be done. Should runtime become prohibitive, sub-sampling strategies for MCMC or stochastic variational inference offer potential workarounds. And when models outgrow both the model-specific variational method as well as the ADVI approach, one may invoke the generalised reparametrisation gradient to expand the family of tractable variational distributions. If instead a simpler model is desired or no parametrisa-

tion is available, non-parametrised methods exploiting conjugacy of the Dirichlet distribution on the transition probabilities provide a reliable fallback. In closing, we hope that the concepts presented in this thesis will inspire continued advances in Bayesian parameter inference for fault detection—ultimately raising the reliability and diagnostic power for high-tech machinery.

References

[1] Pierre Baldi and Søren Brunak. *Bioinformatics: The Machine Learning Approach*. Jan. 2001.

[2] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. "On Markov chain Monte Carlo methods for tall data". In: *Journal of Machine Learning Research* 18.47 (2017), pp. 1–43. URL: http://jmlr.org/papers/v18/15-205.html.

[3] Atilim Gunes Baydin et al. "Automatic Differentiation in Machine Learning: a Survey". In: *Journal of Machine Learning Research* 18.153 (2018), pp. 1–43. URL: http://jmlr.org/papers/v18/17-468.html.

[4] Joris Bierkens, Paul Fearnhead, and Gareth Roberts. "The Zig-Zag process and super-efficient sampling for Bayesian analysis of big data". In: *The Annals of Statistics* 47.3 (2019), pp. 1288–1320. DOI: 10.1214/18-AOS1715. URL: https://doi.org/10.1214/18-AOS1715.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science+Business Media,LLC, 2006.

[6] Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in hidden markov models*. Springer Science+Business Media, Inc, 2005.

[7] Nicholas J. Foti et al. "Stochastic variational inference for hidden Markov models". In: *Advances in neural information processing systems* 27 (2014).

[8] Matthew D. Hoffman et al. "Stochastic variational inference". In: *J. Mach. Learn. Res.* 14.1 (May 2013), pp. 1303–1347. ISSN: 1532-4435.

[9] Matthew Johnson and Alan Willsky. "Stochastic Variational Inference for Bayesian Time Series Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1854–1862. URL: https://proceedings.mlr.press/v32/johnson14.html.

[10] Alp Kucukelbir et al. "Automatic Differentiation Variational Inference". In: *Journal of Machine Learning Research* 18.14 (2017), pp. 1–45. URL: http://jmlr.org/papers/v18/16-107.html.

[11] Frank van der Meulen. *Statistical Inference*. Lecture notes for Statistical Inference course. EEMCS, TU Delft. Aug. 2022.

[12] Gabriel Peyré and Marco Cuturi. "Computational Optimal Transport". In: *Foundations and Trends in Machine Learning* 11.5-6 (2019), pp. 355–607.

[13] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer New York, 2004.

[14] Gareth O. Roberts and Jeffrey S. Rosenthal. "General state space Markov chains and MCMC algorithms". In: *Probability Surveys* 1.none (2004), pp. 20–71. DOI: 10.1214/154957804100000024. URL: https://doi.org/10.1214/154957804100000024.

[15] Gareth O. Roberts and Jeffrey S. Rosenthal. "Harris Recurrence of Metropolis-within-Gibbs and Trans-Dimensional Markov Chains". In: *The Annals of Applied Probability* 16.4 (2006), pp. 2123–2139. ISSN: 10505164. URL: http://www.jstor.org/stable/25449844 (visited on 04/29/2025).

[16] Jeffrey S. Rosenthal. *Optimising and Adapting Metropolis Algorithm Proposal Distributions*. https://probability.ca/jeff/ftpdir/handbookart.pdf. Chapter in Handbook of MCMC, 2nd ed. 2023. (Visited on 05/01/2025).

[17] Francisco J. R. Ruiz, Michalis K. Titsias, and David M. Blei. "The generalized reparameterization gradient". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 460–468. ISBN: 9781510838819.

[18] Aad van der Vaart. *Causality and Graphical Models*. Lecture notes for Advanced Topics in Statistics course. EEMCS, TU Delft. May 2024.

[19] Max Welling and Yee Whye Teh. “Bayesian learning via stochastic gradient langevin dynamics”. In: *Proceedings of the 28th International Conference on International Conference on Machine Learning*. ICML’11. Bellevue, Washington, USA: Omnipress, 2011, pp. 681–688. ISBN: 9781450306195.

[20] Stephen Zhang et al. *OptimalTransport.jl*. https://github.com/JuliaOptimalTransport/OptimalTransport.jl. 2020.

A

Directed Graphical Models and D-separation

Probabilistic models, such as the hidden Markov model discussed in section 2.1, can, in principle, be fully represented using algebraic expressions. However, to gain insight into the structure of the model, it may be beneficial to diagrammatically represent the model as a *graphical model*; we will specifically consider *directed* graphical models in this thesis. It turns out that these directed graphical models are not merely useful visualisations of the model structure, but can also reveal conditional independencies between sets of random variables through the concept of *d-separation*. [5, see p. 359]

The following two sections are based on [18, see chapter on graphical models].

A.1 Directed Graphical Models

A directed graph consists of nodes joined by directed edges, see e.g. figure A.1. This can be formalised by the pair $(V, \mathcal{E})$, where V denotes the set of nodes $\{x_1, \ldots, x_{|V|}\}$, and $\mathcal{E}$ the set of ordered pairs denoting the directed edges. That is, $\mathcal{E}$ contains the element (x_i, x_j) if the graph has a directed edge pointing from x_i to x_j. In such a case, we say that x_i is a parent to x_j and that x_j is child of x_i. We denote the set of all parents of a node x_j by $\text{pa}(x_j)$. The set of all children of a node x_i is denoted by $\text{ch}(x_i)$

When working with directed graphical models, we have to restrict our focus to directed acyclic graphs (DAGs). These are directed graphs without cycles, where a cycle is a sequence of nodes $(x, x_2, \ldots, x_k, x)$, such that $x, x_2, \ldots, x_k$ are distinct and $(x, x_2), (x_2, x_3), \ldots, (x_k, x) \in \mathcal{E}$. An overview of concepts related to directed graphs is given in box A.1.

Box A.1: Definitions for Directed Graphs

The following concepts are taken from [18, see chapter on graphical models]. We consider a directed graph $(V, \mathcal{E})$.

- A *cycle* is a sequence of nodes $(x, x_2, \ldots, x_k, x)$, where $x, x_2, \ldots, x_k$ are distinct and $(x, x_2), (x_2, x_3), \ldots, (x_k, x) \in \mathcal{E}$.
- A *parent* of x is a node y such that $(y, x) \in \mathcal{E}$. $\text{pa}(x)$ is the set of all parents of x.
- A *child* of y is a node x such that $(y, x) \in \mathcal{E}$. $\text{ch}(y)$ is the set of all children of y.
- A *path* from x to y is a sequence of distinct nodes $(x, x_2, \ldots, x_k, y)$ joined consecutively by directed edges pointing in the direction from x to y.
- A *descendant* of x is a node y such that there exists a path from x to y.
- A *trail* from x to y is a sequence of distinct nodes $(x, x_2, \ldots, x_k, y)$ joined consecutively by directed edges, which need not point in the same direction.
- A *collider* is an internal node of a trail, such that the directed edges meet head-on. I.e., if x_i is an internal node of the trail $(x_1, \ldots, x_k)$, x_i is a collider iff $(x_{i-1}, x_i), (x_{i+1}, x_i) \in \mathcal{E}$.
- A *non-collider* is an internal node of a trail that is not a collider.

A directed graphical model describes the structural properties of a random vector X, containing the random variables of the model, through a directed graph. The nodes of the directed graph correspond to the random variables in X, while the directed edges correspond to conditional distributions. More specifically, for every node

x, the incoming edges induce the conditional distribution $p(x|\text{pa}(x))$. The joint probability of the model should then equal the factorisation of these conditional distributions; we call this property the *factorisation property*.

We illustrate the above statements using an example inspired by an example from [5, see p. 360-361]. Suppose we have three random variables x_1, x_2, x_3, which correspond to the nodes in figure A.1. The distribution of these variables can be factorised as follows

$$p(x_1, x_2, x_3) = p(x_3|x_2, x_1)p(x_2|x_1)p(x_1). \tag{A.1}$$

If for every node x_i in figure A.1, we consider the distribution $p(x_i|\text{pa}(x_i))$, we see that we obtain the distributions $p(x_3|x_2, x_1), p(x_2|x_1), p(x_1)$ for the nodes x_3, x_2, x_1, respectively. We observe that these distributions exactly factorise into the likelihood, as required. Now, if we know that $x_3 \perp x_1|x_2$, the factorisation of the likelihood simplifies and becomes

$$p(x_1, x_2, x_3) = p(x_3|x_2)p(x_2|x_1)p(x_1). \tag{A.2}$$

We can then construct an equally valid directed graphical model given in figure A.2. From this graph, we indeed derive the distributions $p(x_3|x_2), p(x_2|x_1), p(x_1)$ for the nodes x_3, x_2, x_1, respectively, and find the simplified factorisation given in equation (A.2).

In the above example, we derived a graphical model from known independence properties. However, it turns out that often the reverse holds. One can derive independence relationships between sets of random variables if a corresponding graphical model is known through the concept of d-separation.

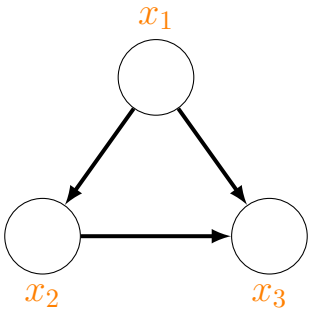

Figure A.1: Example of a directed acyclic graph (DAG) illustrating a graphical model with three random variables x_1, x_2, and x_3. The directed edges from x_1 to x_2, from x_2 to x_3, and from x_1 to x_3 correspond to the factorisation $p(x_1, x_2, x_3) = p(x_3|x_2, x_1)p(x_2|x_1)p(x_1)$, as described in equation (A.1).

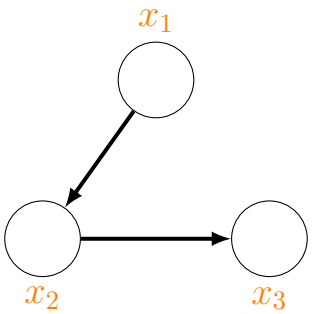

Figure A.2: Simplified DAG derived under the conditional independence assumption $x_3 \perp x_1 \mid x_2$. The graph omits the direct edge from x_1 to x_3, yielding the factorisation $p(x_1, x_2, x_3) = p(x_3|x_2)p(x_2|x_1)p(x_1)$, as shown in equation (A.2).

A.2 D-separation

As mentioned before, the concept of d-separation applied to directed graphical models allows us to deduce conditional independence relationships between sets of random variables. Before we can discuss d-separation, however, we need to introduce a few concepts related to directed graphs. See box A.1 for an overview of these concepts.

Consider a directed acyclic graph $(V, \mathcal{E})$. A *path* from node x to y is a sequence $(x, x_2, \ldots, x_k, y)$ of distinct nodes, such that $(x, x_2), (x_2, x_3), \ldots, (x_k, y) \in \mathcal{E}$. A *descendant* of a node x is a node y such that there is a path from x to y. A *trail* is a sequence of distinct nodes $(x_1, x_2, \ldots, x_k)$ joined by directed edges that need not point in the same direction. I.e., for every $i = 2, \ldots, k$, either $(x_{i-1}, x_i) \in \mathcal{E}$ or $(x_i, x_{i-1}) \in \mathcal{E}$. An internal node x_i of a trail $(x_1, \ldots, x_k)$ is a *collider* if both (x_{i-1}, x_i) and (x_{i+1}, x_i) are in $\mathcal{E}$. That is, the two directed edges joining x_{i-1} to x_i and x_i to x_{i+1} both point towards x_i as in figure A.3. An internal node of a trail that is not a collider is called a *non-collider*. The three possible configurations of a non-collider are given in figure A.4.

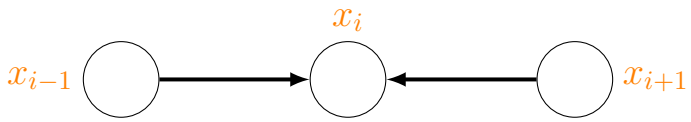

Figure A.3: Illustration of a collider configuration in a DAG. Here, the internal node x_i is a collider because both incoming edges from x_{i-1} and x_{i+1} meet at x_i. This configuration is essential for analysing conditional independence via d-separation.

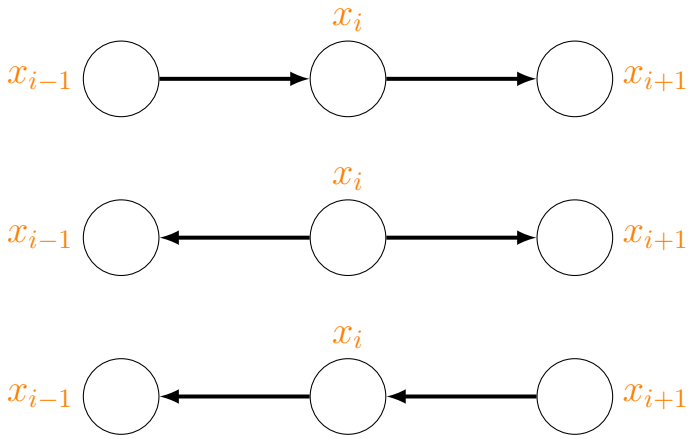

Figure A.4: Examples of non-collider configurations in a DAG. Three distinct arrangements for the internal node x_i are shown, highlighting variations in non-collider relationships used to determine whether a trail is S-open or S-closed in d-separation analysis.

To define the concept of d-separation, we need one more definition related to trails in DAGs.

Definition A.1. *(closed). A trail in a directed acyclic graph is* closed *by a set of nodes S, if for some internal node x in the trail one of the following two statements holds*

- *if $x \in S$, x is a non-collider,*
- *if $x \notin S$, x is a collider without descendants in S.*

*If the above holds, we say that the trail is S-*closed*. Otherwise, we say that the trail is S-*open*.*

We can finally introduce the concept of d-separation.

Definition A.2. *(d-separation) Let A, B, S be disjoint sets of nodes in a directed acyclic graph. We say that A and B are d-separated given S, if every trail from a node in A to a node in B is S-closed.*

By making use of d-separation we can determine whether sets of random variables corresponding to a directed graphical model are conditionally independent.

Theorem A.1. *(Global Markov property DAG). Let $(V, \mathcal{E})$ be a directed acyclic graph corresponding to a graphical model with random vector X. Let A, B, S be disjoint sets of nodes and X_A, X_B, X_S the corresponding random vectors. If A and B are d-separated given S, it follows that $X_A \perp X_B | X_S$.*

B

MCMC Method Convergence Properties

Consider the finite-valued latent random variables $\{z_t^n\}$, observed random variables $\{x_t^n\}$ and model parameters θ as described in chapter 3, where n runs over $\{1,\dots,N\}$ and t over $\{1,\dots,T\}$. $p(\{z_t^n\},\{x_t^n\}|\theta)$ is defined through the time-homogeneous transition probabilities and emission probabilities. The prior $p(\theta)$ is chosen appropriately. We consider the algorithm given in section 4.1. That is,

1. Initialise $\{z_t^n\}$ and θ.
2. For $i = 1,\dots,M$:

For $j = 1,\dots,N_{\text{Gibbs}}$:

- Sample $z_1^1 \sim p(z_1^1|\{z_1^1\}^c,\{x_t^n\},\theta)$.

$\vdots$

- Sample $z_T^1 \sim p(z_T^1|\{z_T^1\}^c,\{x_t^n\},\theta)$.
- Sample $z_1^2 \sim p(z_1^2|\{z_1^2\}^c,\{x_t^n\},\theta)$.

$\vdots$

- Sample $z_T^2 \sim p(z_T^2|\{z_T^2\}^c,\{x_t^n\},\theta)$.

$\vdots$

- Sample $z_1^N \sim p(z_1^N|\{z_1^N\}^c,\{x_t^n\},\theta)$.

$\vdots$

- Sample $z_T^N \sim p(z_T^N|\{z_T^N\}^c,\{x_t^n\},\theta)$.

For $k = 1,\dots,N_{\text{MG}}$:

For $l = 1,\dots,|\theta|$:

- Sample $\theta_l^\star \sim q_l(\theta_{l,\text{prop}}|\theta_l)$.
- Accept with probability $A = \min\Big(1, \dfrac{\tilde{p}(\theta_l^\star)}{\tilde{p}(\theta_l)}\Big)$.
- If accepted $\theta_l = \theta_l^\star$, otherwise $\theta_l = \theta_l$. (B.1)

Here $q_l(\theta_{l,\text{prop}}|\theta_l)$ is a normal distribution centred at θ_l with variance $\exp(2ls_l)$. We set $\tilde{p}(\theta_l) = p(\{z_t^n\},\{x_t^n\}|\theta)p(\theta)$, which is proportional to $p(\theta_l|\theta\setminus\theta_l,\{z_t^n\},\{x_t^n\})$.

For a Markov chain, let $K^n(z,\cdot)$ denote its probability measure after n sweeps starting from z. We attempt to establish convergence of the above algorithm by applying the following theorem from [13, theorem 6.51].

Theorem B.1. *If a Markov chain has stationary distribution $\pi(\cdot)$ and is positive, aperiodic and Harris recurrent,*

$$\lim_{n\to\infty}\sup_{A\in\mathcal{F}}\left|\int K^n(z,A)\mu(dz) - P(A)\right| = 0, \tag{B.2}$$

for every initial distribution μ.

This theorem implies that under certain conditions, the distribution of a Markov chain converges to the target distribution if the target distribution is stationary with respect to the chain, regardless of the initial distribution. We will not delve into the details of the concepts applied in theorem B.1, as doing so falls outside the scope of this thesis. Instead, we will demonstrate that our algorithm satisfies all the required conditions.

A distribution is stationary with respect to a Markov chain, if a step from the chain leaves the distribution unchanged [5, p. 540]. In [5, p. 544], it is shown that the Gibbs sweeps in the above algorithm leave $p(\{z_t^n\}|\theta, \{x_t^n\})$ invariant. We also find that, for every $l = 1, \ldots, |\theta|$, the Metropolis-Hastings steps of component θ_l leave $p(\theta_l|\theta \setminus \theta_l, \{z_t^n\}, \{x_t^n\})$ invariant [5, p. 541]. By induction, it follows that the Metropolis-within-Gibbs sweeps leave $p(\theta|\{z_t^n\}, \{x_t^n\})$ invariant. Therefore, the above chain has stationary distribution $p(\theta, \{z_t^n\}|\{x_t^n\})$ as required.

To establish positivity and aperiodicity of our algorithm, we need the chain to be ϕ-irreducible. Write $p(\{z_t^n\}, \theta|\{z_t^{n'}\}, \theta')$ for the density of $K^1((\{z_t^{n'}\}, \theta'), \cdot)$. That is, $K^1((\{z_t^{n'}\}, \theta'), A) = \int_A p(\{z_t^n\}, \theta|\{z_t^{n'}\}, \theta')dz'd\theta'$ for an event A. A sufficient condition for ϕ-irreducibility is that the density corresponding to one sweep is greater than 0 for all transitions [13, p. 213]. I.e., $p(\{z_t^n\}, \theta|\{z_t^{n'}\}, \theta') > 0$ for all allowed $\{z_t^n\}, \theta, \{z_t^{n'}\}, \theta'$. This clearly holds for the models described in chapter 3. Indeed, since the probability of reaching feasible latent variables $\{z_t^n\}$ is always greater than 0 and $q_l(\theta_{l,\text{prop}}|\theta_l')$ is normal, the density is always larger than 0. Since our chain is ϕ-irreducible and has a stationary distribution, it is positive [13, definition 6.35].

From the definition of aperiodicity [14, section 3.2], it follows that our Markov chain is aperiodic if the probability of remaining in the same state is larger than 0. This property is satisfied in our models. Specifically, after the Gibbs sweeps there is a strictly positive probability of remaining in the same state, and during the Metropolis-within-Gibbs steps, proposed samples may fall outside the parameter space and be rejected with positive probability.

To prove that the overall algorithm is Harris recurrent (and conclude that all conditions of theorem B.1 are satisfied), it suffices to show that sub-chains of the algorithm are Harris recurrent [15, proposition 15]. In particular, we consider one sub-chain consisting solely of the Gibbs sweeps and, for each parameter component, a sub-chain made up exclusively of the corresponding Metropolis–Hastings step from the Metropolis-within-Gibbs algorithm. From [13, lemma 10.9], it follows that the Gibbs sub-chain is Harris recurrent, since the transition kernel for the sub-chain $K^1(\{z_t^n\}|\cdot)$ can be described by a probability density. From [15, theorem 8], it follows that the sub-chains for the parameter components are Harris recurrent, since $q_l(\theta_{l,\text{prop}}|\theta_l) > 0$ for all feasible $\theta_{l,\text{prop}}, \theta_l$. Hence, the complete algorithm is Harris recurrent. Hence, the algorithm converges for our models.

B.1 Adaptive Method Convergence Properties

When the adaptive method from section 4.1.3 is applied to the algorithm in equation (B.1), two additional conditions have to be satisfied to guarantee convergence [16]. To describe these conditions, we need to introduce and recall a few concepts. Write $K_\gamma^n(z, \cdot)$ for the probability measure of a Markov chain, after n iterations starting from z, with parameters given by γ. Assume the Markov chains indexed by γ satisfy the conditions of theorem B.1 with stationary distribution $\pi(\cdot)$. Let $\{\Gamma_n\}$ be a sequence of random variables resulting from an adaptive algorithm, which return parameters for the transition kernel. To prove that the adaptive Markov chain (Z_n) corresponding to probability measure $K_{\Gamma_n}^n(z, \cdot)$ converges, the following two conditions are sufficient. The first condition is the diminishing adaption condition,

$$\lim_{n\to\infty} \sup_{z\in\mathcal{Z}} \sup_{A\in\mathcal{F}} \left| K_{\Gamma_{n+1}}^1(z, A) - K_{\Gamma_n}^1(z, A) \right| = 0 \quad \text{in probability.} \tag{B.3}$$

The second condition is the containment condition,

$$\{M_\epsilon(Z_n, \Gamma_n)\} \text{ is bounded in probability for all } \epsilon > 0, \tag{B.4}$$

where $M_\epsilon(z, \gamma) = \inf\{n \geq 1 : \sup_{A\in\mathcal{F}} |K_\gamma^n(z, A) - \pi(A)| \leq \epsilon\}$ is the convergence time of the kernel $K_\gamma^1(z, \cdot)$ with fixed parameters γ starting from z.

In our case, the adaptive parameters ls_l, $l = 1, \ldots, |\theta|$, restricted to $[-L, L]$ index the transition kernels. The random variables $\{\Gamma_n\}$ are described by the following procedure. After every bth batch of iterations of the overall algorithm, each ls_l is modified by a quantity $\delta(b)$ to achieve a better acceptance ratio, unless this violates the bounds on ls_l. It holds that $\delta(b) \to 0$ as $b \to \infty$.

We consider the first condition for our algorithm. If $N_{\mathrm{MG}} = 1$, it can be derived that

$$\sup_{A\in\mathcal{F}} \left| K^1_{\Gamma_{n+1}}((\{z_t^n\},\theta),A) - K^1_{\Gamma_n}((\{z_t^n\},\theta),A)\right| \leq$$
$$C\int_{\mathbb{R}^{|\theta|}} \left| q_1(\theta'_1|\theta_1,\Gamma_{n+1})\cdots q_{|\theta|}(\theta'_{|\theta|}|\theta_{|\theta|},\Gamma_{n+1}) - q_1(\theta'_1|\theta_1,\Gamma_n)\cdots q_{|\theta|}(\theta'_{|\theta|}|\theta_{|\theta|},\Gamma_n)\right| d\theta', \tag{B.5}$$

since the Gibbs sweeps for $\{z_t^n\}$ are unchanged for different ls_l. Since the proposal distributions are normals, it is clear that this bound is independent of the initial states $(\{z_t^n\},\theta)$. We get

$$\sup_{(\{z_t^n\},\theta)\in\mathcal{X}} \sup_{A\in\mathcal{F}} \left| K^1_{\Gamma_{n+1}}((\{z_t^n\},\theta),A) - K^1_{\Gamma_n}((\{z_t^n\},\theta),A)\right| \leq$$
$$C\int_{\mathbb{R}^{|\theta|}} \left| q_1(\theta'_1|0,\Gamma_{n+1})\cdots q_{|\theta|}(\theta'_{|\theta|}|0,\Gamma_{n+1}) - q_1(\theta'_1|0,\Gamma_n)\cdots q_{|\theta|}(\theta'_{|\theta|}|0,\Gamma_n)\right| d\theta', \tag{B.6}$$

with $\mathcal{X}$ the space of all latents and parameters θ. Now, since $\delta(b)\to 0$ as $n\to 0$, the integrand goes to 0 for every value of θ'. It can also be bounded by an integrable function, because $ls_l \in [-L,L]$ and the proposals are normal. By the dominated convergence theorem, this bound goes to 0 surely. Therefore, it goes to 0 in probability and equation (B.3) holds. The diminishing condition is satisfied. We can follow the same reasoning if $N_{\mathrm{MG}} > 1$.

We now consider the containment condition. We write (Z_n) for the Markov chain resulting from our adaptive algorithm with probability measure $K^n_{\Gamma_n}((\{z_t^n\},\theta),\cdot)$ after n sweeps. The density of $K^n_\gamma((\{z_t^n\},\theta),\cdot)$ is continuous with respect to the initial θ and ls_l, $l = 1,\ldots,|\theta|$ for every allowed initial $\{z_t^n\}$. Since ls_l is restricted to $[-L,L]$ and θ lives on a compact space, this density is uniformly continuous. Hence, for every $\epsilon > 0$, we can consider finitely many instances of the parameters ls_l and $(\{z_t^n\},\theta)$ and find an integer N_ϵ, such that $\inf\{n\geq 1 : \sup_{A\in\mathcal{F}} \left|K^n_\gamma((\{z_t^n\},\theta),A)\right| - \pi(A)\right| \leq \epsilon\} \leq N_\epsilon$. We can then write for $n \geq N_\epsilon$

$$\begin{aligned}\sup_{A\in\mathcal{F}} \left|K^n_{\Gamma_n}(Z_n,A) - \pi(A)\right| &\leq \sup_{A\in\mathcal{F}} \left|K^n_{\Gamma_n}(Z_n,A) - K^n_\gamma((\{z_t^n\},\theta),A)\right| + \left|K^n_\gamma((\{z_t^n\},\theta),A) - \pi(A)\right| \\ &\leq 2\epsilon \quad \text{almost surely.}\end{aligned} \tag{B.7}$$

It follows that the containment condition holds.

ICT, Strategy & Policy

High Tech Campus 25
5656 AE Eindhoven
www.tno.nl

TNO innovation for life