Parameter Estimation in Dynamic Bayesian Networks: Theory and Methods

Allen Qian

Under the direction of

Joonsoo Lee
Massachusetts Institute of Technology

Research Science Institute
July 30, 2025

Abstract

Dynamic Bayesian Networks (DBNs) have received much attention over the past few decades, with potential application in fields like medicine for the inference of conditions, environmental modeling for risk assessment, machine learning, and many other fields. However, the problem of parameter estimation within these networks remains an open problem. This paper focuses on this problem, particularly the weights defining node interactions over time. We derive Maximum Likelihood Estimators (MLEs) for the linear Gaussian case, proving that the optimal weights satisfy a system of equations involving empirical moments. We further establish non-asymptotic bounds on estimation error and prove convergence at the rate of $O(\frac{1}{\sqrt{N}})$ under sub-Gaussian assumptions. We develop a general framework for nonlinear update rules, showing that parameter estimates can be obtained for any general update function. Simulations validate these theoretical findings and show how graph structure and sample size affect estimation accuracy. Our results provide insight into the role of network structure in parameter learning and point toward future directions for extending these methods to nonlinear or non-Gaussian settings.

1 Introduction

In many complex real-world systems, such as longitudinal brain imaging [1], environmental modeling [2], and aviation [3], there are many interconnected aspects and variables. Capturing and understanding how each component affects the system is important for explanation and prediction in these domains.

Dynamic Bayesian Networks (DBNs) provide a method to model these interactions, in which each node evolves over time based on its neighbors' past states [4, 5]. However, given observations of a complex system, it is difficult to infer how strong or in what way variables will influence each other. To do so, researchers have used various statistical algorithms, such as Maximum Likelihood Estimation (MLE) or Bayesian estimation [6, 7, 8]. For example, Koller and Friedman 2009 [9] established foundational results for parameter learning in Gaussian Bayesian Networks, showing the feasibility of learning conditional dependencies from data. Building on this, Kungurtsev et al. 2024 [8] applied variational inference to dynamic models, offering scalable estimation approaches. However, they did not investigate the case of DBNs, bounding the error of estimates, or extending the models into the general case. In this paper, we study the problem of parameter influence in these networks: given certain observations of node states for a fixed graph, are we able to estimate certain parameters on each node? Understanding these parameters has significant implications for applications across various fields.

We look at a specific type of Dynamic Bayesian Networks: discrete time Gaussian networks [9]. Through numerical experiments and statistical analysis, we find that the MLE method performs similarly to Bayesian estimation in estimating parameters. Furthermore, we also demonstrate certain cases where the MLE is likely to break down due to the non-invertibility of covariance matrices.

In Section 2, we outline some necessary proofs and definitions related to Dynamic Bayesian networks. Next, in Section 3, we consider the linear gaussian case, where each node is sampled from a distribution centered at the weighted average of the node's parents, and prove a Maximum Likelihood Estimate for the weights; we also look at bounding the estimate for the weights. Then, in Section 4, we look at more general update functions for the mean and find a method for estimating weights in the general case. Then, in Section 5, we run simulations that test the accuracy of our estimators for a variety of graphs by finding how the error in our estimates changes as a function of sample size. Lastly, in Section 6, we present ideas for the future direction of our work.

2 Preliminaries

We start with some essential definitions and theorems.

2.1 Probability Foundations

A *random variable* is a variable that can take on different values, each with an associated probability. For example, a coin flip can be modeled by a random variable X where $P(X = \text{heads}) = 0.5$.

The following is an important law in probability that we use to define dependence relations between random variables in Dynamic Bayesian Networks.

Theorem 2.1 (Law of Total Probability)**.** *For random variables* $X, Y_1, \ldots, Y_n$,

$$P(X) = \sum_{y_1} \cdots \sum_{y_n} P(X|Y_1 = y_1, \ldots, Y_n = y_n) P(Y_1 = y_1, \ldots, Y_n = y_n).$$

This theorem tells us how to compute the marginal probability of a variable X by averaging over the possible values of variables $Y_1, \ldots, Y_n$ that influence it. This is a key tool in Bayesian modeling, where we often wish to determine how one variable depends on others.

To model how variables influence each other, we use a mathematical structure called a *graph*. In classical graph theory, a graph is defined as a pair (V, E) where V is the set of vertices (nodes), and $E \subseteq V \times V$ is the set of directed or undirected edges.

Definition 2.1 (Weighted Directed Graph)**.** A *weighted directed graph* with n nodes is a pair (V, E) where $V \subset \mathbb{R}^n$ is a set of vertices, and $E \in \mathbb{R}_{\geq 0}^{n \times n}$ is a non-negative matrix. The entry $E_{ij} > 0$ indicates a directed edge from node i to node j, with weight E_{ij} representing the strength of that connection.

Note that in this work, we assume that the edge connections E are known. The weights on the edges E_{ij} are parameterized with α's, which are stored alongside the adjacency matrix of the graph.

To understand Bayesian networks, we must explain Directed Acyclic Graphs.

Definition 2.2 (Directed Acyclic Graphs)**.** A *Directed Acyclic Graph (DAG)* is a graph with all directed edges, and with no cycles (any path from a node that leads back to itself).

DAGs are essential in probabilistic models because they allow us to factor joint probability distributions using conditional independence.

The main object in this study are Bayesian Networks, and we define them as follows:

Definition 2.3 (Bayesian Networks)**.** A *Bayesian Network (BN) Structure* $\mathcal{G}$ is a DAG where each node i is assigned a random variable X_i, where each $X_i \in V$ is sampled from a probability distribution $\mathcal{P}_E$. This is denoted by the notation $V \sim \mathcal{P}_E$.

A direct consequence of the Law of Total Probability is that Bayesian networks entirely contain the information of the conditional probability distributions between random variables. [9]

Example. Consider a 3-node Bayesian Network where X_1 influences X_2, and both X_1 and X_2 influence X_3. The DAG is shown in Figure 1, and the joint probability factors as:

$$P(X_1, X_2, X_3) = P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_1, X_2).$$

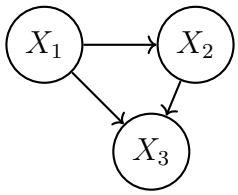


Figure 1: Example of a 3-node Bayesian Network.

Definition 2.4 (Independence)**.** Random variables X and Y are said to be *independent* if

$$P(X|Y) = P(X).$$

Bayesian networks provide a convenient way to depict these independencies: if there is no edge between two nodes, the two nodes are independent. There are also additional hidden independencies that may be present in BNs. While these independencies are not important to this study, they are still important to general Bayesian Networks nonetheless [4, 9].

2.2 Graph Theory Foundations

We now provide the key definitions and notation used throughout this paper.

We use standard graph-theoretic notation. For a directed graph $G = (V, E)$, we write $\mathrm{Pa}(i)$ for the set of parents of node i. Nodes with no incoming edges are called *roots*; nodes with no outgoing edges are called *leaves*.

Definition 2.5 (Neighbors)**.** The *neighbors* of a node X are the set of all nodes directly connected to it by an edge in either direction: that is, its parents and children.

Definition 2.6 (Dynamic Bayesian Models)**.** A *Dynamic Bayesian Model* is a BN dependent on an additional variable t:

$$V(t) \sim \mathcal{P}_E(t).$$

In a discrete case, we have $V(t_k) \sim \mathcal{P}_E(t_k)$. In this work, we specifically focus on 1st order Markovian discrete time DBNs. This means that for $A_i \in \mathbb{R}^n$,

$$P[V(t_k)|V(t_{k-1}) = A_{k-1}, V(t_{k-2}) = A_{k-2}, \dots] = P[V(t_1)|V(t_0) = A_0].$$

This Markovian assumption allows us to decompose time dependent DBNs nicely using the law of total probability: $P[V(t_k)] = \prod_{i=1}^{k} P[V(t_i)|V(t_{i-1})]P[V(t_0)]$. Of course, we are able to write each distribution on the graph as a function of the conditional probabilities of each node in the graph.

Note: In this case, Markovian DBNs are mathematically equivalent to having many samples of a Bayesian network.

Definition 2.7 (Gaussian Networks)**.** *Gaussian Networks* are Bayesian Networks where all conditional distributions are Gaussian. Mathematically:

$$P(X_i|\mathrm{Pa}(X_i)) = \mathcal{N}\left(\mu_i + \sum_{j \in \mathrm{Pa}(i)} \alpha_{ij}(X_j - \mu_j), \sigma_i^2\right)$$

for each node i.

3 The Linear Gaussian Case

Consider a Dynamic Bayesian network with n vertices and variables $X_1, \dots, X_n$. We use the following convention to refer to random variables and their values: random variables are capitalized, and the corresponding lowercase letters denote their instantaneous value. We assign each node a weight α_i that characterizes the strength of influence from parent nodes. We make the following assumptions:

1. Node x_i is updated based on the distribution

$$x_i(t) \sim \mathcal{N}\left(\frac{1}{d_i}\sum_{j\in \mathrm{Pa}(i)} \alpha_j x_j(t-1) + c_i, \sigma^2\right).$$

   d_i is the number of parents of node i, σ^2 is the predefined variance of the distribution, and c_i is a constant bias term.

2. The roots of the graph are not sampled from the same distribution because they have no parents. In this case, in order to avoid complications, we can just assume they are sampled from the standard normal distribution, or from the normal distribution centered at their corresponding weights.

3. Data that corresponds to the probability distributions in the graph is collected ahead of time and contains many particles sampled over the network's distribution.

Using these, we derive an approximation for the α's in a linear gaussian system.

Theorem 3.1. *Under the above assumptions, the maximum likelihood estimates for the parameters $\boldsymbol{\theta}_{x_i} = \{\alpha_1, \ldots, \alpha_k, c_i\}$ of node i with parents $U_1, \ldots, U_k$ are given by the solution to the linear system*

$$E(X \cdot U_j) = c_i E(U_j) + \frac{1}{d_i}\sum_{\ell=1}^{k} \alpha_\ell E(U_\ell \cdot U_j), \quad j = 1, \ldots, k.$$

Proof. There is a somewhat well-known proof of this fact [9], which we will reiterate with different techniques. In order to best approximate the parameters $\boldsymbol{\theta}_{x_i} = \{\alpha_0, \ldots \alpha_k\}$, we will use Maximum Likelihood Estimation (MLE). Firstly, we simplify the normal distribution by taking the log-likelihood function of the sum over the M samples, or particles, of a node X, which is just the logarithm of the overall probabilities for a certain sample from given parameters and a prior, which we treat as uniform for the time being. We denote the mth sample of the jth node at time t by $u_j(t)[m]$.

We use the measurements for all $U_j \in \mathrm{Pa}(X)$ at time $t = 0$, as well as the measurements for X made at $t = 1$. Because of the Markovian assumption that each node depends only on its parents, taking a large number of samples is equivalent to taking a large number of time

steps. The log-likelihood simplifies as

$$\log L_{x_i}(\boldsymbol{\theta}_{x_i} : \mathcal{D}) = \log \prod_m \mathcal{N}\left(\frac{1}{d_i}\sum_{j\in \mathrm{Pa}(i)} \alpha_j u_j(t-1)[m] + c_i, \sigma^2\right)$$
$$= \sum_m \left[\log \frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{2\sigma^2}\left(c_i + \frac{1}{d_i}\sum_j \alpha_j u_j[m] - x[m]\right)^2\right].$$

We assume that our likelihood is log-concave, to find the maximum value of the log-likelihood. First differentiating both sides with respect to c_i, then setting the equation equal to 0, we obtain

$$0 = \frac{\partial}{\partial c_i}\sum_m \left[\log \frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{2\sigma^2}\left(c_i + \frac{1}{d_i}\sum_j \alpha_j u_j[m] - x[m]\right)^2\right]$$
$$= -\frac{1}{\sigma^2}\left(Mc_i + \alpha_1 \sum_m \frac{u_1[m]}{d_i} + \cdots + \alpha_k \sum_m \frac{u_k[m]}{d_i} - \sum_m x[m]\right).$$

Rearranging and multiplying both sides by $\frac{\sigma^2}{M}$, we find

$$\frac{1}{M}\sum_m x[m] = c_i + \alpha_1 \frac{1}{Md_i}\sum_m u_1[m] + \cdots + \alpha_k \frac{1}{Md_i}\sum_m u_k[m]. \tag{1}$$

Notice that because expected value is defined as

$$E(X) = \frac{1}{M}\sum_m x[m],$$

then we may substitute into equation (1) to obtain

$$E(X) = c_i + \alpha_1 E\left(\frac{U_1}{d_i}\right) + \cdots + \alpha_k E\left(\frac{U_k}{d_i}\right).$$

If, rather than differentiating with respect to c_i, we differentiate with respect to some α_i, then similar arithmetic yields

$$E(X \cdot U_i) = c_i E(U_i) + \alpha_1 E(U_1 \cdot U_i) + \cdots + \alpha_k E(U_k \cdot U_i) \qquad \text{for } 1 \le i \le k. \tag{2}$$

At this point, we have $k+1$ linear equations for the $k+1$ unknowns α_i, and so we are able to solve for the parameters $\boldsymbol{\theta}_{x_i}$. □

We can also derive a bound on the estimation error of the vector $\alpha = \{\alpha_1, \alpha_2, ...\}$, as shown below.

Let A = $\begin{bmatrix} E(U_0U_0) & E(U_1U_0) & \dots \\ E(U_0U_1) & \ddots & \\ \vdots & & E(U_kU_k) \end{bmatrix}$, and $b = \begin{bmatrix} E(XU_1) \\ \vdots \\ E(XU_k) \end{bmatrix}$. Then, the weights satisfy the equation $A\hat{\alpha} = b$.

Proposition 3.2. *Suppose $\alpha = A^{-1}b$ are the true parameters and $\hat{\alpha} = \hat{A}^{-1}\hat{b}$ is the empirical estimates. Then, the following bound applies:*

$$\|\hat{\alpha} - \alpha\| \leq \frac{\|A^{-1}\|(\|\hat{b} - b\| + \|\hat{A} - A\|\|\alpha\|)}{1 - \|A^{-1}\|\|\hat{A} - A\|}.$$

, where we take the 2-norm.

Proof. Suppose that $\hat{A} = A + \delta A$ and $\hat{b} = \delta b$ are the empirical estimates.

Initially, we have $A\alpha = b$ and $\hat{A}\hat{\alpha} = \hat{b}$. Subtracting, we obtain

$$\begin{aligned} \hat{A}\hat{\alpha} - A\alpha &= \hat{b} - b \\ (A + \delta A)\hat{\alpha} - A\alpha &= \delta b \\ A(\hat{\alpha} - \alpha) &= \delta(b - A\hat{\alpha}) \\ \hat{\alpha} - \alpha &= A^{-1}\delta(b - A\hat{\alpha}). \end{aligned}$$

We rewrite $\delta A\hat{\alpha}$ as $\delta A(\hat{\alpha} - \alpha + \alpha) = \delta A\alpha + \delta A(\hat{\alpha} - \alpha)$ Plugging this back in, we obtain

$$\begin{aligned} \hat{\alpha} - \alpha &= A^{-1}(\delta b - \delta A\alpha - \delta A(\hat{\alpha} - \alpha)) \\ (I + A^{-1}\delta A)(\hat{\alpha} - \alpha) &= A^{-1}(\delta b - \delta A\alpha) \\ \hat{\alpha} - \alpha &= (I + A^{-1}\delta A)^{-1}A^{-1}(\delta b - \delta A\alpha). \end{aligned}$$

Because $\|(I + A^{-1}\delta A)^{-1}\| \leq \dfrac{1}{1 - \|A^{-1}\delta A\|}$ if $\|A^{-1}\delta A\| < 1$ (For more information, see [10]),

$$\|\hat{\alpha} - \alpha\| \leq \frac{\|A^{-1}\|(\|\delta b\| + \|\delta A\|\|\alpha\|)}{1 - \|A^{-1}\delta A\|}.$$

Substituting back $\delta A = \hat{A} - A$ and $\delta b = \hat{b} - b$, we conclude that

$$\|\hat{\alpha} - \alpha\| \leq \frac{\|A^{-1}\|(\|\hat{b} - b\| + \|\hat{A} - A\|\|\alpha\|)}{1 - \|A^{-1}\|\|\hat{A} - A\|}.$$

□

A significant result can be obtained by looking at the bound in relation to the particle sample size.

Theorem 3.3. *The estimation error decreases at a rate of*

$$\|\hat{\alpha} - \alpha\| = O_p\left(\frac{1}{\sqrt{N}}\right).$$

Proof. We aim to bound the estimation error in

$$\hat{\alpha} = \hat{A}^{-1}\hat{b}, \quad \alpha = A^{-1}b, \text{ where } \hat{A} = \frac{1}{N}\sum_{i=1}^{N} U_i U_i^\top, \quad \hat{b} = \frac{1}{N}\sum_{i=1}^{N} X_i U_i,$$

and $A = \mathbb{E}[UU^\top]$, $b = \mathbb{E}[XU]$.

Assume that each $U_i \in \mathbb{R}^k$ is a centered sub-Gaussian vector, and X_i is a centered sub-Gaussian scalar, independent of U_i.

We define

$$X_i = U_i U_i^\top - \mathbb{E}[U_i U_i^\top], \qquad Y_i = X_i U_i - \mathbb{E}[X_i U_i].$$

Then, we can represent the error of the expectation matrices as

$$\hat{A} - A = \frac{1}{N}\sum_{i=1}^{N} X_i, \qquad \hat{b} - b = \frac{1}{N}\sum_{i=1}^{N} Y_i.$$

We apply the Matrix Bernstein Inequality [11] to $\hat{A} - A$. The conditions are that X_i are independent, centered, symmetric random matrices, and that there exists a constant L such that $\|X_i\| \leq L$ almost surely.

Then by Tropp's matrix Bernstein inequality [11], for all $t \geq 0$,

$$\mathbb{P}\left(\left\|\sum_{i=1}^{N} X_i\right\| \geq t\right) \leq 2k \cdot \exp\left(-\frac{t^2/2}{\sigma^2 + Lt/3}\right),$$

where $\sigma^2 = \left\| \sum_{i=1}^{N} \mathbb{E}[X_i^2] \right\|$.

To translate this to the average $\hat{A} - A = \frac{1}{N} \sum X_i$, we write

$$\mathbb{P}\left(\left\| \hat{A} - A \right\| \geq t \right) = \mathbb{P}\left(\left\| \sum X_i \right\| \geq Nt \right) \leq 2k \cdot \exp\left(-\frac{(Nt)^2/2}{\sigma^2 + LNt/3} \right).$$

To simplify the exponent, we use the inequality

$$\frac{(Nt)^2}{\sigma^2 + LNt} \geq cN \cdot \min\left(\frac{t^2}{K^2}, \frac{t}{K} \right),$$

for some constant $c > 0$ and a bound $K = \max\left(\frac{\sigma}{\sqrt{N}}, L \right)$.

Hence,

$$\mathbb{P}\left(\|\hat{A} - A\| \geq t \right) \leq 2k \exp\left(-cN \min\left(\frac{t^2}{K^2}, \frac{t}{K} \right) \right).$$

A similar vector Bernstein inequality [12] applies to $\hat{b} - b$, yielding the same type of bound:

$$\mathbb{P}\left(\|\hat{b} - b\| \geq t \right) \leq 2k \exp\left(-cN \min\left(\frac{t^2}{K^2}, \frac{t}{K} \right) \right).$$

These imply that the rates of convergence of both $\|\hat{A} - A\|$ and $\|\hat{b} - b\|$ are $O_p(1/\sqrt{N})$, since for any fixed $\varepsilon > 0$, there exists a constant $C > 0$ such that:

$$\mathbb{P}\left(\|\hat{A} - A\| > \frac{C}{\sqrt{N}} \right) < \varepsilon.$$

Finally, plugging into the bound in Proposition 3.2, we conclude that $\|\hat{\alpha} - \alpha\| = O_p(1/\sqrt{N})$ as well. □

Remark 3.4. The constants c and K appearing in these inequalities depend on the sub-exponential norms of the random variables above. While these constants do not impact the convergence rate, they may impact finite-sample performance of estimation. In applications, these constants can sometimes be computed, but they rely on quantities like the sub-exponential norm $\|U_i\|_{\psi_2}$, and so may be difficult to compute.

A promising area for future work in improving this bound is through the use of total variance bounds.

Bhattacharyya et al. 2022 [13] present a general framework for bounding the error of distributions by controlling the change in total variation. In particular, Theorem 3.1 implies a bound on the total variation for a given Bayesian Network. Suppose $\mathcal{G}$ is a fixed DAG on n variables with in-degree at most d. For $\varepsilon, \delta \in (0,1)$, and given $\mathcal{O}(nd_{avg}\varepsilon^{-2} \cdot \log(n\delta^{-1})$ for an unknown Bayesian network $\mathcal{P}$ over $\mathcal{G}$, then with probability at least $1-\delta$, we recover a Bayesian network Q over $\mathcal{G}$ in $O(n^2 d_{avg}^2 d\varepsilon^{-2} \cdot \log(\delta^{-1}))$ time such that $d_{TV}(\mathcal{P}, Q) \leq \varepsilon$, where

$$d_{\mathrm{TV}}(\mathcal{P}, Q) = \frac{1}{2} \int |\mathcal{P}(x) - Q(x)| \, dx.$$

In the setting of estimators, if one can bound the change in the distribution of $\hat{\alpha}$ as the underlying distribution changes from μ to $\hat{\mu}_N$, then one may derive bounds on $\|\hat{\alpha} - \alpha\|$.

Conjecture 3.5. *For a Linear Gaussian DBN with true distribution $\mathcal{P}$ and MLE estimate Q, the parameter error is bounded by the Total Variation distance:*

$$\|\hat{\alpha} - \alpha\|_2^2 \leq C \cdot d_{\mathrm{TV}}(\mathcal{P}, Q)$$

where C depends on the spectral properties of the covariance matrix A.

4 The Non-Linear Case

A natural extension of the linear model is to consider non-linear relationships between parent and child nodes. Instead of linear combinations, we can use parameterized functions to represent dependencies.

Firstly, for simplicity, consider a node X with parents X_1 and X_2. We introduce functions $f(\alpha)$ and $g(\alpha)$ parameterized by the weight vector α, giving the update rule:

$$x = f(\alpha)x_1 + g(\alpha)x_2.$$

This is still linear in x, but gives us a start into using general functions.

We obtain a result for the estimation of the α's in this case under some key assumptions.

Theorem 4.1. *The MLE estimates for the non-linear model for 2 parents satisfy*

$$\begin{cases} E(XX_1) = fE(X_1^2) + gE(X_1X_2) \\ E(XX_2) = fE(X_1X_2) + gE(X_2^2). \end{cases}$$

Proof. Using MLE in a similar method to the linear case and taking the derivative, we obtain the equation

$$0 = \sum_m (x - f(\vec{\alpha})x_1 - g(\vec{\alpha})x_2)(\nabla f(\vec{\alpha})x_1 + \nabla g(\vec{\alpha})x_2).$$

To simplify notation, we let $f = f(\vec{\alpha})$ and $g = g(\vec{\alpha})$. Expanding this equation and rearranging, we obtain

$$\sum_m x x_1 \nabla f + x x_2 \nabla g = \sum_m f \nabla f x_1^2 + g \nabla g x_2^2 + (f \nabla g + g \nabla f) x_1 x_2.$$

Again letting $E(X) = \dfrac{1}{M} \sum_m x[m]$, we obtain

$$E(XX_1)\nabla f + E(XX_2)\nabla g = E(X_1^2) f \nabla f + E(X_2^2) g \nabla g + E(X_1 X_2)(f \nabla g + g \nabla f).$$

We can write this in matrix form as g

$$\begin{bmatrix} \nabla f & \nabla g \end{bmatrix} \begin{bmatrix} E(XX_1) \\ E(XX_2) \end{bmatrix} = \begin{bmatrix} \nabla f & \nabla g \end{bmatrix} \begin{bmatrix} E(X_1^2) & E(X_1 X_2) \\ E(X_2 X_1) & E(X_2^2) \end{bmatrix} \begin{bmatrix} f \\ g \end{bmatrix}$$

Here, we assume that $\begin{bmatrix} \nabla f & \nabla g \end{bmatrix}$ is invertible. We also assume that $\begin{bmatrix} \nabla f & \nabla g \end{bmatrix}$ is a square matrix so we can take the inverse: thus, we assume that $\dim \vec{\alpha} = \mathrm{Pa}(i)$.

Then, we can cancel the gradients from both sides, and we are left with a system of two equations:

$$\begin{cases} E(XX_1) = f E(X_1^2) + g E(X_1 X_2) \\ E(XX_2) = f E(X_1 X_2) + g E(X_2^2). \end{cases}$$

From the system of equations, we are able to solve for the values of f and g. If we know the forms of f and g as a function of α_1 and α_2, we may be able to solve for α_1 and α_2, assuming that the system of equations is solvable. □

These equations can be easily extended into higher dimensions. Consider a node with n parents $U_1, U_2, \ldots, U_n$ and functions $f_1, \ldots, f_n$. Then, there is a system of n equations with $1 \le i \le n$:

$$E(XU_i) = f_1 E(U_1 U_i) + \cdots + f_n E(U_n U_i).$$

Remark 4.2. This system may not be solvable if the functions f_i are linearly dependent upon one another, as presented in the next example.

Example 4.1. *Suppose $f = \alpha_1\alpha_2$ and $g = 2\alpha_1\alpha_2$. Then, if the MLE estimates yields $f = 1$ and $g = 2$ after solving, then f and g are dependent and the system is not solvable for α.*

Example 4.2. *Take the case when $f(\vec{\alpha}) = \alpha_1 \cdot \alpha_2$ and $g(\vec{\alpha}) = \alpha_1 + \alpha_2$. Then, we obtain $x = (\alpha_1\alpha_2)x_1 + (\alpha_1 + \alpha_2)x_2$, which is solvable. For this case, we simulated data in Figure 2 to approximate α_1 and α_2, which converged to the true values as sample size increased, similar to the established bounds.*

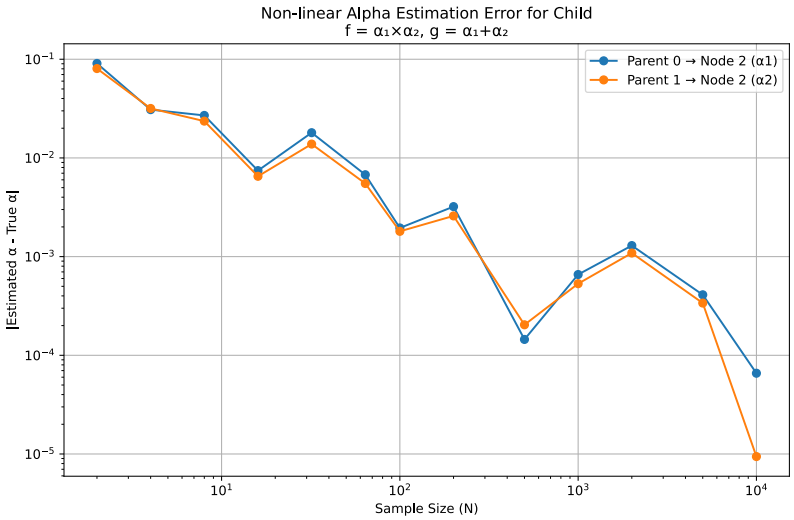


Figure 2: Logarithmic plot of the errors of α's when $f(\vec{\alpha}) = \alpha_1 \cdot \alpha_2$ and $g(\vec{\alpha}) = \alpha_1 + \alpha_2$.m

We can also consider a more general function of the form $f(\alpha_i, x_i)$, where the corresponding distribution of a node X would be sampled according to the distribution

$$x_i(t) \sim \mathcal{N}\left(f(\alpha_i, x_i), \sigma^2\right).$$

Theorem 4.3. *The Maximum Likelihood estimates for the parameters α in the general case are given by*

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i} \left(x_i - f(\alpha_i, u_i)\right)^2.$$

Proof. The Log-Likelihood function of the normal distribution with mean $f(\alpha_i, x_i)$ is simply

$$\log L_{x_i}(\boldsymbol{\theta}_{x_i} : \mathcal{D}) = \sum_{m} \left[\log \frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{2\sigma^2}\left(x[m] - f(\alpha_i, x_i)\right)^2\right].$$

In order to maximize the log-likelihood, we must minimize the residual squared loss function of $(x[m] - f(\alpha_i, x_i))^2$, and so the result follows:

$$\hat{\alpha} = \arg\min_{\alpha} \sum_i (x_i - f(\alpha_i, u_i))^2 .$$

□

For the Bayesian estimator, the prior is also introduced into the distribution.

Corollary 4.1. *Assume the prior estimates for α are sampled from the normal distribution $\mathcal{N}(\mu, s^2)$. The Bayesian estimates for the parameters α in the general case are given by*

$$\hat{\alpha} = \arg\min_{\alpha} \sum_i (x_i - f(\alpha_i, u_i))^2 + \frac{\sigma^2}{s^2}\|\alpha - \mu\|^2.$$

Proof. We start with a prior on our α, such that $\alpha \sim p(\alpha)$. Then, by Bayes' rule,

$$p(\alpha \,|\, \text{data}) \propto p(\text{data} \,|\, \alpha)\ p(\alpha).$$

In our case, $p(\text{data} \,|\, \alpha) = p(x_i \,|\, u_i, \alpha_i)$.

Taking the logarithm of both sides, the posterior becomes

$$\log p(\alpha|\text{data}) = -\frac{1}{2\sigma^2} \sum_i (x_i - f(\alpha_i, u_i))^2 + \log p(\alpha) + \text{const.}$$

If we use a gaussian prior,

$$\log p(\alpha) = -\frac{1}{2s^2}\|\alpha - \mu\|^2,$$

and so this simplifies to the desired result.

□

Remark 4.4. If the uniform distribution is used for the priors in this case, then $p(\alpha) = \dfrac{1}{\prod_i (b_i - a_i)}$ if $\alpha_i \in [a_i, b_i]$. Thus, the prior term would become

$$\log p(\alpha) = \begin{cases} -\sum_i \log (b_i - a_i), & \alpha \in [a_i, b_i] \\ -\infty & \alpha \notin [a_i, b_i] \end{cases}$$

5 Numerical Simulations and Validation

In practice, we solved the parameter estimation problem using gradient-based optimization methods, such as L-BFGS, to minimize the negative log-likelihood derived in Section 4. We worked in a *well-parameterized regime*, meaning that the global minimizer of the loss function exists within the function space considered. This assumption ensures that the optimization procedure converges to a meaningful set of parameters.

Conceptually, the implementation follows these steps:

1. Initialize parameters α for each edge in the network.
2. Define a loss function corresponding to the residual squared error between observed and predicted node values under the chosen nonlinear update rule.
3. Iteratively update α using a quasi-Newton optimization method until convergence.

We validated our theoretical results through numerical simulations on synthetic Dynamic Bayesian Networks with known parameters, and both linear and non-linear functions. We generated data through Python, based on a pre-defined graph and adjacency matrix, a weight vector, and sample sizes. For the Linear Gaussian case, we calculated the error of the alpha estimates for each sample size of data and plotted it, as shown below in Figure 3.

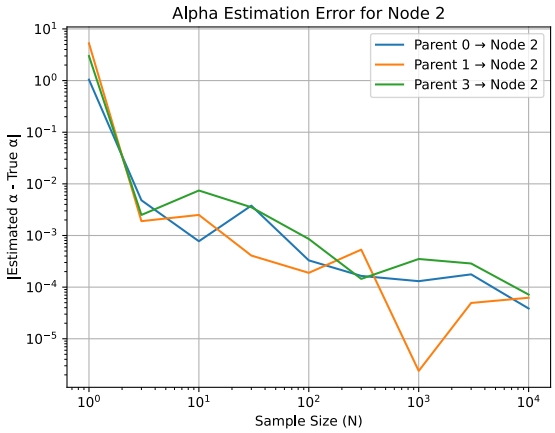
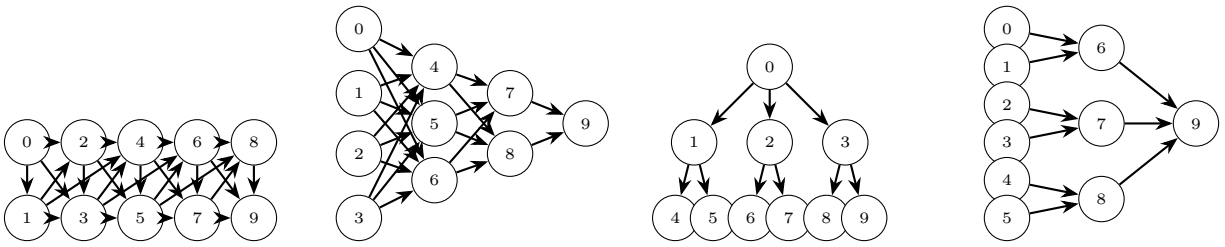


Figure 3: Estimation error versus sample size for a node with 3 parents. The error decreases at rate $O\left(\frac{1}{\sqrt{N}}\right)$, consistent with Theorem 3.3. Each curve represents a different parent's weight estimate.

We tested the models on graphs categorized by connectivity and number of root nodes. Figure 4 shows the four types of graphs that we used in our simulation, highly connected graphs with a high number of roots, highly connected graphs with a low number of roots, not highly connected graphs with a high number of roots, and not highly connected graphs with a low number of roots. These were chosen in order to test whether different graph structures and extra details would confound the parameter estimation model in any way. Figure 5 shows the graph of sample size vs the 2-norm of the difference between the vectors $\hat{\alpha}$ and α.



(a) Highly connected, 1 root

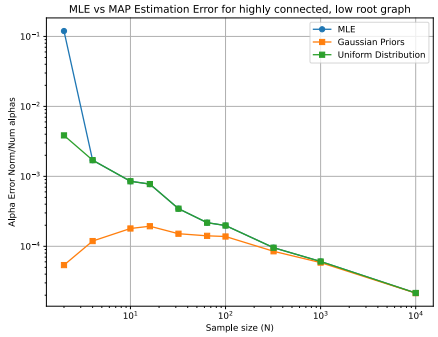
(b) Highly connected, 4 roots

(c) Low connected, 1 root (tree)

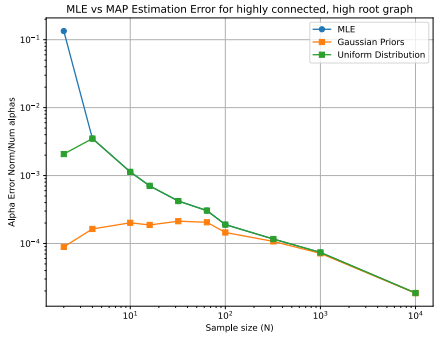
(d) Low connected, 6 roots

Figure 4: The 4 different graph topologies used, with varying connectivity and root count.

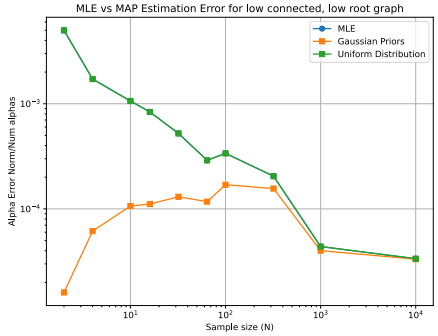
In Figure 5, we used a Gaussian prior centered very close to the true value of α, with $\sigma^2 = s^2 = 0.01^2$. However, in real-world scenarios, this may be overly optimistic. When we center the priors on the value of the MLE estimation from the previous step, we obtain a result much more similar to the MLE, as shown in Figure 6.



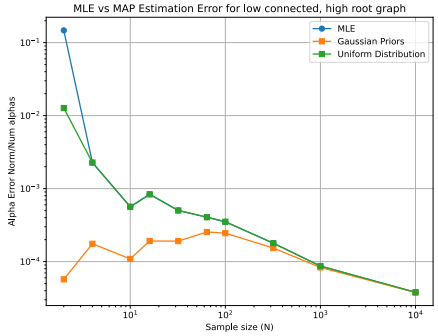
(a) Highly connected, low root



(b) Highly connected, high root



(c) Low connected, low root (tree)



(d) Low connected, high root

Figure 5: Log plots of error norm for different graph topologies used in simulations.

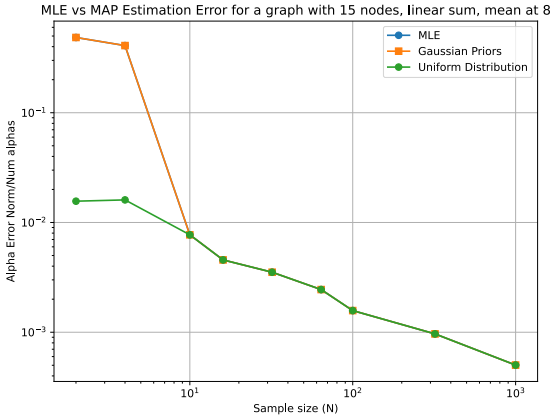
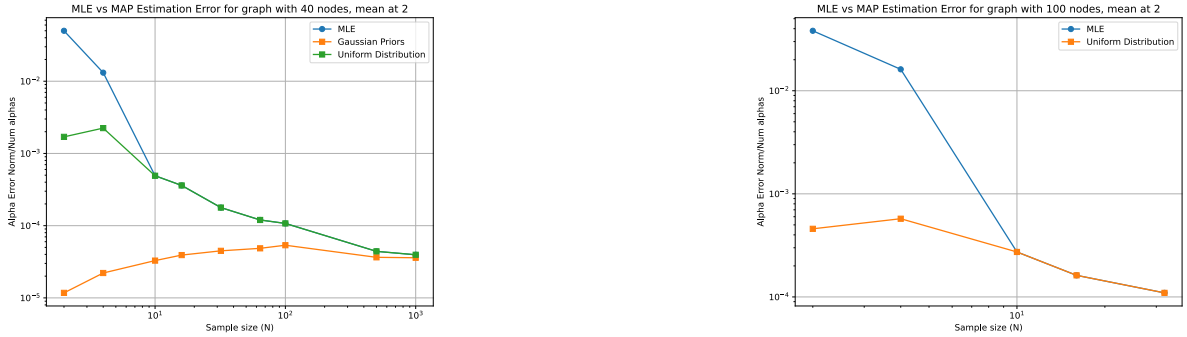


Figure 6: MAP Gaussian prior centered at the MLE estimates: note the similarity.

Figure 5 shows that the MLE and Bayesian methods of estimation for α are very similar

in accuracy, when the priors for the Bayes are from a uniform distribution. All four graphs converged at a similar rate to 0 as sample size increased, showing the effectiveness of weight estimation in accurately predicting the parameters in an asymptotically exponential time. The difference in graph structure appears to play a lesser role in

We also tested the effect of graph size on estimates, with both results for 40 and 100 node graphs showing a better performance for the uniform distribution for low sample sizes, while MLE matched its performance for higher sample sizes, as shown in Figure 7.

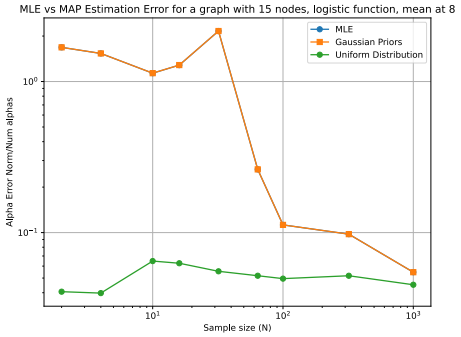


(a) Simulations on graphs with 40 Nodes

(b) Simulations on graphs with 100 Nodes

Figure 7: Results of MLE vs MAP Estimation for graphs with high node count. Note that fewer sample sizes were ran for 100 nodes, due to the computational intensive nature of the algorithm. Graphs created by the student researcher using Python, 2025.

When testing more, complicated non-linear functions, we found interesting results with the uniform prior. In particular, when updating the means on a logistic curve from the parents of a node, the uniform priors, with the priors set between 0 and 1, performed much better than the MLE, as shown in Figure 8.



.p

Figure 8: Simulation results after using a logistic update function. Graph created by the student researcher using Python, 2025.

This result is interesting, as it has implications for instances where the weights in graphs are mapped to a limited interval or finite space, such as in uses of the sigmoid function in machine learning. In these cases, implementing a uniform prior for estimation may prove more accurate than a simple MLE estimate, especially for limited sample sizes.

5.1 Real-World Application: European Temperature Data

We applied our method to daily temperature measurements from the ECA&D (European Climate Assessment & Dataset) [14] spanning 87 data points across 80 weather stations in the United Kingdom from 2015-2016. Temperature in particular was chosen due to daily mean temperature is often being close to normal.

We constructed a Dynamic Bayesian Network where each node represents a weather station, and bi-directional edges connect geographically proximate stations. For a distance of 0.5 degrees, 80 edges were drawn between stations. We also assumed a linear Gaussian model for simplicity.

After applying the MLE and MAP model to the ECA&D data, we measured the significance of these results with a p-value test. More specifically, standard errors and p-values were computed using observed Fisher information (the inverse Hessian of the negative log-likelihood), approximated with the Gauss-Newton relation

$$\widehat{\mathrm{Cov}(\hat{\alpha})} \approx \hat{\sigma}^2 (J^T J)^{-1},$$

where J is the Jacobian of model predictions with respect to α, and $\hat{\sigma}^2$ is the residual variances of the model. The z-statistic $z_k = \hat{\alpha}_k / \mathrm{SE}(\hat{\alpha}_k)$ was used to obtain 2-sided p-values.

As observed in Figure 9 larger-magnitude α values corresponded to smaller p-values, suggesting that stronger inter-station dependencies were estimated more precisely, with the Least Squared Regression Line yielding an equation of

$$p = 0.749 - 0.761\alpha.$$

The strong R^2 value of 0.850 also indicates that the correlation between $|\hat{\alpha}|$ and standard error further supports that stronger connections were estimated more precisely.

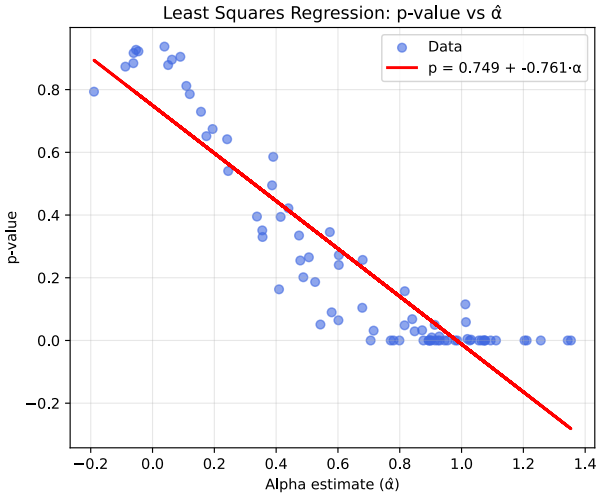


Figure 9: Relationship between $\hat{\alpha}$ and p-values. Graph created by the student researcher using Python, 2025.

To interpret the network geographically, we plotted each weather station at its latitude-longitude positions with the cartopy module and drew edges colored corresponding to the sign and magnitude of each $\hat{\alpha}_k$, as shown in Figure 10. This visualization highlights clusters of stations with strong positive coupling, particularly in densely sampled regions such as central and southern England. The accuracy of predictions of future temperatures can also be added from these $\hat{\alpha}$s, but have been omitted here.

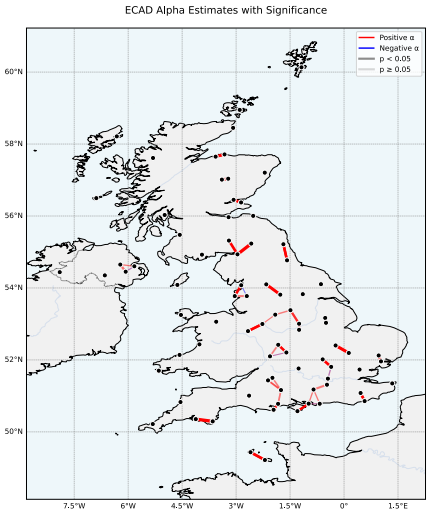


Figure 10: Geographical locations of each station with graphical edges. Graph created by the student researcher using Python, 2025.

6 Discussion and Future Work

We have developed a maximum likelihood framework for parameter estimation in Dynamic Bayesian Networks, providing closed-form solutions for the linear Gaussian case and extensions to non-linear functional relationships.

There are some key points for future research that can be explored. Firstly, the effect of the invertibility of the covariance matrix A warrants investigated. In certain cases, the true value of A can be non-invertible, but in simulations, these cases appeared to still approximate the α's well, potentially due to perturbations in the matrix due to randomness. Exploring why this is the case is a remaining open problem. Additionally, refining the bound estimate by using the definition of Total Variation present in Bhattacharyya et al. 2022 [13] to get a bound in terms of the distribution. A lower bound for the error in A can be obtained by the methods present in Devroye et al. 2023 [15], but the issue with extending this to the error with α is that there is no direct inequality relating the total variation to α.

The problem of dynamical alphas is also a potential area of research. In some real-world scenarios, the weights of the graph may change on the values in the graph in a dynamical method, and modeling this is also an open problem.

This work has applications across fields where understanding temporal dependencies is crucial. In climate modeling, our methods enable measuring strengths of spatial correlations between monitoring stations, improving extreme weather prediction. as shown through

our application to ECA&D data. In healthcare, they could identify the relative influence of risk factors on disease progression, helping with clinical treatments. In neuroscience, mapping interactions between brain regions could be useful in understanding neural circuits and detecting disorders early. More broadly, any domain with time-evolving systems, such as financial networks, ecological systems, or social dynamics, can use these methods to learn interaction strengths from data By providing both theoretical guarantees and efficient algorithms, this work makes Bayesian network parameter learning accessible across scientific disciplines.

7 Acknowledgements

First, I would like to thank my mentor, Joonsoo Lee, for the extensive introduction into the field and for providing pivotal guidance throughout the research process. I also want to thank Dr. Tanya Khovanova for her weekly guidance in writing my paper and her advice on conducting research, and my advisors, Professor Roman Bezrukavnikov and Dr. Jonathan Bloom for overseeing the progress of this research problem. I would also like to thank my tutor, AnaMaria Perez, for helping me greatly in the research process with my paper and presentation, and for her moral support. I want to thank Mircea Dan Hernest, Ghina Darazi and Stanislav Harizanov, who have all helped me with their suggestions on this paper. I would like to acknowledge all last-week TAs and especially Marina Lin, who helped edit this paper and provided helpful comments. I also acknowledge the data providers in the ECA&D project. Lastly, I would like to thank MIT, CEE, and RSI for providing me with this opportunity, and to my sponsor, Joseph V. Mandarino, who made my experience at RSI possible.

References

[1] R. Chen, S. M. Resnick, C. Davatzikos, and E. H. Herskovits. Dynamic Bayesian network modeling for longitudinal brain morphometry. *NeuroImage*, 59(3):2330–2338, 2012.

[2] J. Chang, Y. Bai, J. Xue, L. Gong, F. Zeng, H. Sun, Y. Hu, H. Huang, and Y. Ma. Dynamic Bayesian networks with application in environmental modeling and management: A review. *Environmental Modelling Software*, 170:105835, 2023.

[3] B. Matthews, S. Das, K. Bhaduri, K. Das, R. Martin, and N. Oza. Discovering Anomalous Aviation Safety Events Using Scalable Data Mining Algorithms. *Journal of Aerospace Information Systems*, 10:467–475, 10 2013.

[4] Z. Ghahramani. Learning dynamic Bayesian networks. *International School on Neural Networks, Initiated by IIASS and EMFCSC*, pages 168–197, 1997.

[5] Z. Ghahramani. An introduction to hidden Markov models and Bayesian networks. *International journal of pattern recognition and artificial intelligence*, 15(01):9–42, 2001.

[6] E. P. Xing. 6 : Learning Fully Observed Bayesian Networks, 2017.

[7] L. Schmidt-Thieme. Bayesian Networks 9. Parameter Learning / MLE and MAP, 2010.

[8] V. Kungurtsev, Apaar, A. Khandelwal, P. S. Rastogi, B. Chatterjee, and J. Mareček. Empirical Bayes for Dynamic Bayesian Networks Using Generalized Variational Inference, 2024.

[9] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. 2009.

[10] O. P. Ferreira and S. Z. Németh. Projection onto simplicial cones by a semi-smooth Newton method, 2014.

[11] J. A. Tropp. User-Friendly Tail Bounds for Sums of Random Matrices. *Foundations of Computational Mathematics*, 12(4):389–434, Aug. 2011.

[12] R. Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

[13] A. Bhattacharyya, D. Choo, R. Gajjala, S. Gayen, and Y. Wang. Learning Sparse Fixed-Structure Gaussian Bayesian Networks, 2022.

[14] A. M. G. Klein Tank et al. Daily dataset of 20th-century surface air temperature and precipitation series for the European Climate Assessment. *International Journal of Climatology*, 22(12):1441–1453, 2002. Data and metadata available at `https://www.ecad.eu`.

[15] L. Devroye, A. Mehrabian, and T. Reddad. The total variation distance between high-dimensional Gaussians with the same mean, 2023.