



MAP inference algorithms without approximation for collective graphical models on path graphs via discrete difference of convex algorithm

Yasunori Akagi¹ · Naoki Marumo² · Hideaki Kim¹ · Takeshi Kurashima¹ · Hiroyuki Toda¹

Received: 13 April 2022 / Revised: 21 September 2022 / Accepted: 30 October 2022 /

Published online: 8 December 2022

© The Author(s) 2022

Abstract

Collective graphical model (CGM) is a probabilistic model that provides a framework for analyzing aggregated count data. Maximum a posteriori (MAP) inference of unobserved variables under given observations is one of the essential operations in CGM. Because the MAP inference problem is known to be NP-hard in general, the current mainstream approach is to solve an alternative problem obtained by approximating the objective function and applying continuous relaxation. However, this approach has two significant drawbacks. First, the quality of the solution deteriorates when the values in the count data are negligible due to the inaccuracy of Stirling's approximation. Second, the application of continuous relaxation causes the violation of integrality constraints. This paper proposes novel algorithms for MAP inference in CGMs on path graphs to overcome these problems. Our method is based on the discrete difference of convex algorithm (DCA); DCA is a general framework to minimize the sum of a convex function and a concave function by repeatedly minimizing surrogate functions. Utilizing the particular structure of path graphs, we efficiently solve the surrogate function minimization by minimum convex cost flow algorithms. Furthermore, our approach also leads to a new method of solving another important task; MAP inference of the sample size in CGM on path graphs. Our method is naturally applicable to this task, allowing us to design very efficient algorithms. Experimental results on synthetic and real-world datasets show the effectiveness of the proposed algorithms.

Keywords Collective graphical model · Network flow · Discrete difference of convex algorithm

Editor: Pradeep Ravikumar.

✉ Yasunori Akagi
yasunori.akagi.cu@hco.ntt.co.jp

Extended author information available on the last page of the article

1 Introduction

In recent years, the importance of aggregated count data, which is calculated from the data of multiple individuals, has been increasing (Tanaka et al. 2019; Zhang et al. 2020). Although technologies for acquiring individual data such as sensors and GPS have greatly advanced, it is still very difficult to handle individual data due to privacy concerns and the cost of data collection. However, there are many situations where data aggregated from multiple individuals can be obtained and utilized easily. For example, mobile spatial statistics (Terada et al. 2013), which is the hourly population data of fixed-size square grid cells calculated from cell phone network data in Japan, is available for purchase; such data is being used for disaster prevention and urban planning (Suzuki et al. 2013). In traffic networks, traffic volume data at each point can be obtained more easily by sensors or cameras than the trajectories of individual cars, and the data is useful for managing traffic congestion (Morimura et al. 2013; Zhang et al. 2017).

Collective graphical model (CGM) (Sheldon and Dietterich 2011) is a probabilistic model to describe aggregated statistics of a sample drawn from a graphical model. CGM makes it possible to conduct various practical tasks on aggregated count data, such as interpolation of unobserved aggregated count values, denoising of observed count values, and parameter learning of the underlying graphical model. Particularly, the case where the underlying graph is a path graph is important because CGMs on path graphs can treat time series data in which the states of interest follow Markov chains. In fact, most of the real-world applications of CGMs utilize CGMs on path graphs to represent the collective movement of humans and animals (Du et al. 2014; Sun et al. 2015; Akagi et al. 2018). Detailed analyses of time series of collective people movements from limited observations would be useful for controlling people flow to avoid congestion and to maintain social distancing in urban spaces.

One of the essential operations in CGM is maximum a posteriori (MAP) inference. MAP inference is the discrete (combinatorial) optimization problem of finding an assignment of unobserved variables that maximizes the posterior probability under given observations. MAP inference makes it possible to interpolate missing values of aggregated data and estimate more detailed information behind the observations. For example, suppose the population distribution of humans in each area and at each time is given as observations. In that case, the number of people moving between each time and area with the highest posterior probability can be estimated by performing MAP inference on a CGM. This allows us to obtain more detailed information about crowd movements from a series of snapshots of population distribution. Another example is population interpolation; by conducting MAP inference on a CGM with population distribution at two-time points as input, we can obtain population distribution at the time points in between. This allows us to obtain the city's high temporal resolution population dynamics from observations of population distribution at limited time points.

Unfortunately, MAP inference for general CGMs has been shown to be NP-hard (Sheldon et al. 2013) and thus is difficult to solve exactly and efficiently. Therefore, an alternative approach that solves an approximate problem, which is derived by applying Stirling's approximation and continuous relaxation, has been proposed (Sheldon et al. 2013). Subsequent studies have focused on solving this approximate problem efficiently (Sun et al. 2015; Vilnis et al. 2015; Nguyen et al. 2016; Singh et al. 2020).

However, there are inherent problems with this approach to solving the approximate problem. First, this approach tends to output a solution with a low posterior

probability when the values in the count tables are small, because Stirling's approximation, $\log x! \approx x \log x - x$, is inaccurate when x is small. Such a situation frequently occurs when the number of values that each variable in the graphical model takes is large, or when the sample size is small. Second, since continuous relaxation is applied, the integrality constraints of count table values are violated in the output. As a result, values that should be integers (e.g., the number of people) are no longer integers, which not only reduces interpretability, but also makes the output less sparse, resulting in high memory consumption to maintain the output. It is possible to obtain integer-valued results by rounding the output, but this rounding process destroys the sum constraints among the estimated counts, e.g., the sum of the count table values at each node may not match the sample size.

To resolve these issues, in this paper, we propose a new method for MAP inference for CGMs on path graphs. We first show that the objective function of the problem can be expressed as the sum of univariate discrete convex functions and discrete concave functions. Based on this expression, we utilize the idea of the difference of convex algorithm (DCA) (An Le Thi and Pham Dinh 2018). DCA is a framework to minimize a function expressed as the sum of a convex function and a concave function. In DCA, a solution is obtained by repeatedly minimizing a surrogate function that upper-bounds the objective function, and the objective function value decreases monotonically in each iteration. In addition, the algorithm terminates in a finite number of iterations in our case since the variables are discrete, not continuous.

The key to make the DCA-based algorithm efficient is a fast minimization algorithm for the surrogate function. Because the feasible region of our problem is limited to integer lattice points, continuous optimization methods such as the gradient descent, which are usually used in DCAs, cannot be applied to minimize our surrogate function. Instead, we utilize the special structure of path graphs; it enables us to formulate the minimization problem of the surrogate function as a combinatorial optimization problem called the *minimum convex cost flow* problem. Fast algorithms for the minimum convex cost flow problem are known and we can minimize the surrogate function efficiently by using these algorithms.

The proposed method has several practical advantages. First, since the proposed method does not use Stirling's approximation, it offers an accurate inference even when the values in the count tables are small. This makes it possible to output solutions with much higher posterior probability than the approximation-based approach. Second, because the proposed method does not apply continuous relaxation, the obtained solution is guaranteed to be integer-valued, which results in sparse and interpretable outputs. In Sect. 5, we show experimental results gained from synthetic and real-world datasets; they indicate that the proposed method outputs higher quality solutions than the existing approach. We show that the superiority of the proposed method is much greater when the sample size is not very large or the number of states on nodes in the graphical model is large.

Furthermore, our approach also leads to a new method of solving another important task; sample size estimation in CGM on path graphs. The sample size is the number of individuals in a sample obtained from the original graphical model before aggregation; for instance, in human flow analysis, the sample size corresponds to the total number of people in the entire space. In most existing CGM studies, the MAP inference problem is solved under the assumption that the sample size is given in advance (Sheldon et al. 2013; Sun et al. 2015; Singh et al. 2020; Nguyen et al. 2016). However, it is often difficult to know the true sample size exactly a priori. This is because the true sample size cannot be obtained from the observed aggregate values due to observation noise, or worse, some aggregate values may be missing. For example, when dealing with population distribution data of a

city, it is possible that the total number of people in the entire city at each time is not constant due to inaccurate observations. In this case, there are several options for the sample size, and we have to manually set the appropriate value. This may be a barrier to real-world applications of the MAP inference problem of CGMs.

A natural approach to tackle this task is to set a prior distribution for the sample size and conduct MAP inference of sample size and unobserved aggregate values simultaneously. However, it is not easy to construct an efficient algorithm for this new MAP inference problem; a naive method is to solve the MAP inference problem with a given sample size for all candidate sample sizes and output the solution with the highest posterior probability among them. But this method requires a lot of computation time because we have to solve many MAP inference problems.

Our approach based on DCA and the minimum convex cost flow algorithms can naturally handle the MAP inference problem even when the sample size is not given. The proposed method can estimate the sample size by solving only one MAP inference problem and thus can output a solution very efficiently. Experimental results confirm that the proposed method can achieve almost the same objective function values with significantly less computational time than the baseline method with a brute force search of sample size.

A preliminary version of this work appeared in the Proceedings of NeurIPS'21 (Akagi et al. 2021). The main difference from Akagi et al. (2021) is that the algorithm is extended to the problem setting when the sample size is unknown, and the effectiveness of the algorithm has been confirmed by experiments using synthetic and real-world datasets.

2 Related work

2.1 MAP inference for CGMs

Several methods have been proposed for the MAP inference of CGMs, but most of them take the approach of solving the approximate problem (Sheldon et al. 2013), which is derived by applying Stirling's approximation and continuous relaxation. For example, the interior point method (Sheldon et al. 2013), projected gradient descent (Vilnis et al. 2015), message passing (Sun et al. 2015) and Sinkhorn-Knopp algorithm (Singh et al. 2020) have been used to solve the approximate problem. In particular, Nguyen et al. (2016) proposes a method to use DCA to solve this approximate problem. Although this approach is similar to our proposal in that it uses DCA, the purpose of applying DCA is totally different: our focus is to solve the MAP inference problem without using any approximation or continuous relaxation.

One of the few exceptions is the method proposed in Akagi et al. (2020), which solves the original MAP inference problem directly without using approximation. Our method follows this line of research, but there are two major differences. First, their method can only be applied to CGM on a graph with two vertices, and thus applicability is very limited. Since our method is consistent with this method when applied to CGM on a graph with two vertices, our method can be regarded as a generalization of their method. Second, their work assumes accurate observations and does not handle observation noise.

Sheldon et al. (2007) solves related collective MAP inference problems on path graphs. The problems addressed in this paper are different from ours; their purpose is finding the most likely assignments of the entire variables for each individual, while our purpose is finding the most likely node and edge contingency tables. In their settings, non-linear terms

in the log posterior probability vanish, and the MAP inference problem can be solved easily by linear optimization approaches.

2.2 Difference of convex algorithm (DCA)

DCA, which is sometimes called Convex Concave Procedure (Yuille and Rangarajan 2001), is a framework to minimize a function expressed as the sum of a convex function and a concave function (An Le Thi and Pham Dinh 2018). DCA was originally proposed as a method for optimization in continuous domains. DCA has been used in various machine learning fields, such as feature selection (An Le Thi et al. 2015), reinforcement learning (Piot et al. 2014), support vector machines (Xu et al. 2017) and Boltzmann machines (Nitanda and Suzuki 2017).

Several studies have applied DCA to discrete optimization problems. This line of research is sometimes called discrete DCA (Maehara and Murota 2015). (Narasimhan and Bilmes 2005; Iyer and Bilmes 2012) propose algorithms to minimize the sum of a submodular function and a supermodular function. This algorithm is generalized to yield the minimization of the sum of an M/L-convex function and an M/L-concave function (Maehara and Murota 2015), where M-convex function and L-convex function are classes of discrete convex functions (Murota 1998). Although our work is closely related to these studies, it is not part of them. This is because our problem can be regarded as the minimization of the sum of two M-convex functions and a separable concave function, and this is not included in the class of functions dealt with in Maehara and Murota (2015)¹.

3 Collective graphical models

3.1 Collective graphical models in previous studies

Collective graphical model (CGM) is a probabilistic generative model that describes the distributions of aggregated statistics of a sample drawn from a certain graphical model (Sheldon and Dietterich 2011). Let $G = (V, E)$ be an undirected tree graph (i.e., a connected graph with no cycles). We consider a pairwise graphical model over discrete random variable $X := (X_u)_{u \in V}$ defined by

$$\Pr(X = x) = \frac{1}{Z} \prod_{(u,v) \in E} \phi_{uv}(x_u, x_v), \quad (1)$$

where $\phi_{uv}(x_u, x_v)$ is a local potential function on edge (u, v) and $Z := \sum_x \prod_{(u,v) \in E} \phi_{uv}(x_u, x_v)$ is the partition function for normalization. In this paper, we assume that x_u takes values on the set $[R]$ for all $u \in V$, where $[k]$ denotes the set $\{1, 2, \dots, k\}$ for a positive integer k .

We draw an ordered sample $(X^{(1)}, \dots, X^{(M)})$ independently from the graphical model, where M is the sample size. Let $\mathbf{n}_u := (n_u(i))_{i \in [R]}$ and $\mathbf{n}_{uv} := (n_{uv}(i, j))_{i, j \in [R]}$, where $n_u(i) := |\{m \mid X_u^{(m)} = i\}|$ and $n_{uv}(i, j) := |\{m \mid X_u^{(m)} = i, X_v^{(m)} = j\}|$. Each entry of \mathbf{n}_u and \mathbf{n}_{uv} is the number of occurrences of a particular variable setting (see Fig. 1). We

¹ A separable convex function is both L-convex and M-convex, but a sum of two M-convex functions is neither M-convex nor L-convex.

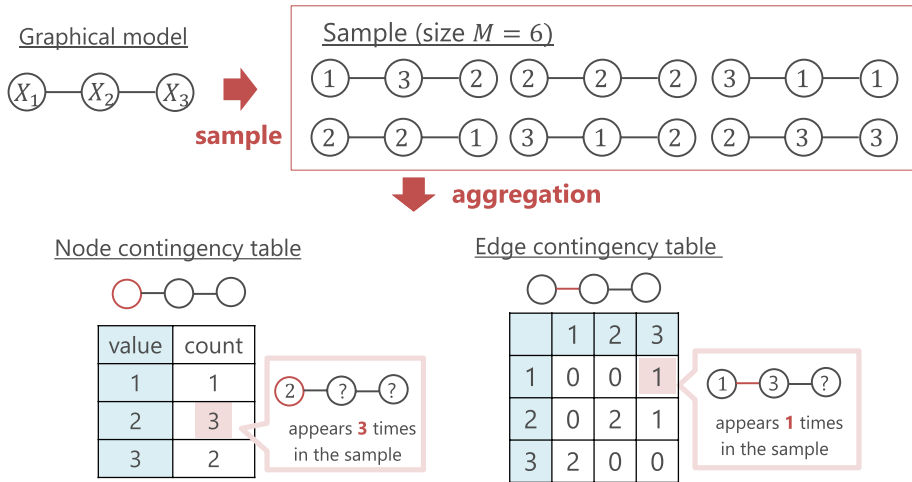


Fig. 1 An example of generation process of contingency tables in CGM on a path graph when $T = 3, R = 3, M = 6$

call $(\mathbf{n}_u)_{u \in V}$ node contingency table and $(\mathbf{n}_{uv})_{(u,v) \in E}$ edge contingency table, and denote $\mathbf{n} := ((\mathbf{n}_u)_{u \in V}, (\mathbf{n}_{uv})_{(u,v) \in E})$. We assume that observations $\mathbf{y} := (\mathbf{y}_u)_{u \in V}$ are generated by adding noise to the node contingency table $(\mathbf{n}_u)_{u \in V}$, and the distribution of \mathbf{y} is given by

$$\Pr(\mathbf{y} | \mathbf{n}) = \prod_{u \in V} \prod_{i \in [R]} p_{ui}(y_u(i) | n_u(i)),$$

where p_{ui} is the noise distribution. An additional assumption is described below.

Assumption 1 For $u \in V$ and $i \in [R]$, $\log p_{ui}(y | n)$ is a concave function in n .

Assumption 1 is a quite common assumption in CGM studies (Sheldon et al. 2013; Sun et al. 2015). Commonly used noise distributions such as Gaussian distribution $p_{ui}(y_u(i) | n_u(i)) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y_u(i) - n_u(i))^2}{2\sigma^2}\right)$ and Poisson distribution $p_{ui}(y_u(i) | n_u(i)) = \frac{n_u(i)^{y_u(i)}}{y_u(i)!} \cdot \exp(-n_u(i))$ satisfy Assumption 1.

It is also possible to consider observation models other than the one described here. For example, some observations may be missing, noiseless observations may be obtained, or some or all of the elements of the noisy edge contingency table may be observed. For the sake of simplicity, we limit ourselves to models where noisy vertex contingency tables are observed, but the following description and the proposed method can be generalized to the above cases as well.

The MAP inference problem for CGM is to find \mathbf{n} that maximizes the posterior probability $\Pr(\mathbf{n} | \mathbf{y})$. The MAP inference problem is the operation of finding the vertex/edge contingency table with the highest posterior probability from noisy observations. It is of great importance in CGM research because it allows interpolation of missing values in aggregate data and estimation of more detailed information hidden behind observations. For more specific applications, see the example in Sect. 3.3.

Since $\Pr(\mathbf{n} \mid \mathbf{y}) = \Pr(\mathbf{n}, \mathbf{y}) / \Pr(\mathbf{y})$ from Bayes' rule, it suffices to maximize the joint probability $\Pr(\mathbf{n}, \mathbf{y}) = \Pr(\mathbf{n}) \cdot \Pr(\mathbf{y} \mid \mathbf{n})$. $\Pr(\mathbf{n})$ is called CGM distribution and calculated as follows (Sun et al. 2015):

$$\Pr(\mathbf{n}) = F(\mathbf{n}) \cdot \mathbb{I}(\mathbf{n} \in \mathbb{L}_M^{\mathbb{Z}}), \quad (2)$$

$$F(\mathbf{n}) := \frac{M!}{Z^M} \cdot \frac{\prod_{u \in V} \prod_{i \in [R]} (n_u(i)!)^{v_u-1}}{\prod_{(u,v) \in E} \prod_{i,j \in [R]} n_{uv}(i,j)!} \cdot \prod_{(u,v) \in E} \prod_{i,j \in [R]} \phi_{uv}(i,j)^{n_{uv}(i,j)}, \quad (3)$$

$$\mathbb{L}_M^{\mathbb{Z}} := \left\{ \mathbf{n} \in \mathbb{Z}_{\geq 0}^{|V|R+|E|R^2} \mid M = \sum_{i \in [R]} n_u(i) \ (u \in V), \right. \\ \left. n_u(i) = \sum_{j \in [R]} n_{uv}(i,j) \ ((u,v) \in E, i \in [R]) \right\}.$$

Here, $\mathbb{I}(\cdot)$ is the indicator function, v_u is the degree of node u in G , and $\mathbb{L}_M^{\mathbb{Z}}$ is the set of possible contingency tables. Using the above notations, the MAP inference problem can be written as

$$\min_{\mathbf{n} \in \mathbb{L}_M^{\mathbb{Z}}} -\log F(\mathbf{n}) - \log \Pr(\mathbf{y} \mid \mathbf{n}). \quad (4)$$

3.2 Prior distribution for sample size

In this paper, we consider both existing problem setups where the sample size is given as input and new problem setups where the sample size is not given as input. As we will see later, the existing problem setup can be considered a particular case of the new problem setup, so we will discuss the new problem setup, i.e., when the sample size is unknown, and mention the particular case when necessary.

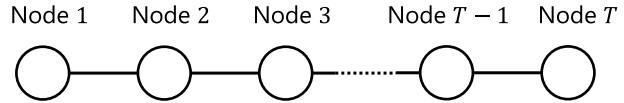
We set a prior probability distribution of the sample size, $\Pr(M) = q(M)$, where $M \geq 0$ is a random variable which represents the sample size. We make an assumption on the prior probability distribution $q(M)$.

Assumption 2 $\log q(M) + \log q(M+2) \leq \log q(M+1)$ holds for all $M \in \mathbb{Z}_{\geq 0}$.²

Many practical probability distributions satisfy Assumption 2; e.g. the discrete Gaussian distribution $q_{\text{Gauss}}(M) \propto \exp(-(M-\mu)^2/2\sigma^2)$, the Poisson distribution $q_{\text{Poisson}}(M) \propto \exp(\lambda^M/M!)$, and the uniform distribution

$$q_{\text{uniform},U}(M) = \begin{cases} \frac{1}{U} & M \in \{0, 1, \dots, U-1\} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

² This condition is equivalent to discrete concavity of $\log q(M)$, which is defined later in Definition 1. The proof of this equivalence is shown in Lemma 7.

Fig. 2 A path graph P_T 

for some positive integer U^3 .

Note that by setting

$$q_{\text{given}}(M) = \begin{cases} 1 & M = M_{\text{given}}, \\ 0 & \text{otherwise,} \end{cases}$$

we obtain the existing problem setup where the sample size M_{given} is given as input. This indicates that our new formulation is a generalization of the existing problem setup. Therefore, all subsequent discussions are also applicable to existing problem settings.

The posterior distribution of (M, \mathbf{n}) under the given observation \mathbf{y} is written as $\Pr(M, \mathbf{n} \mid \mathbf{y}) = \Pr(M, \mathbf{n}, \mathbf{y}) / \Pr(\mathbf{y}) \propto q(M) \Pr(\mathbf{n} \mid M) \Pr(\mathbf{y} \mid \mathbf{n})$. Therefore, by the same argument as the derivation of (4), the MAP inference problem can be written as

$$\min_{(M, \mathbf{n}) \in \mathbb{L}^Z} -\log q(M) - \log F(M, \mathbf{n}) - \log \Pr(\mathbf{y} \mid \mathbf{n}), \quad (6)$$

where

$$\begin{aligned} \mathbb{L}^Z := \left\{ (M, \mathbf{n}) \in \mathbb{Z}_{\geq 0}^{1+|V|R+|E|R^2} \mid M = \sum_{i \in [R]} n_u(i) \ (u \in V), \right. \\ \left. n_u(i) = \sum_{j \in [R]} n_{uv}(i, j) \ ((u, v) \in E, i \in [R]) \right\} \end{aligned} \quad (7)$$

and $F(M, \mathbf{n})$ is the same function as $F(\mathbf{n})$ in (3) with M added to the argument.

3.3 CGMs on path graphs

Hereafter, we focus on CGMs on path graphs, which is the main topic of this paper. Path graph P_T is an undirected graph whose vertex set is $V = [T]$ and edge set is $E = \{(t, t+1) \mid t \in [T-1]\}$ (see Fig. 2). A graphical model (not CGM) on path graph is the most basic graphical model that represents a time series generated by a Markov model; that is, the current state depends only on the previous state. A CGM on a path graph represents the distribution of aggregated statistics when there are many individuals whose state transition is determined by a Markov model. In the rest of this paper, we use the notation $n_{ii} := n_i(i)$, $n_{ij} := n_{t,t+1}(i, j)$, and $\phi_{ij} := \phi_{t,t+1}(i, j)$ for simplicity.

We give an example of a CGM on a path graph which models human mobility. Consider that a space is divided into R distinct areas and that M people are moving around in the space. The random variable $X_t^{(m)}$ represents the area to which person m belongs at time step t , and the time series $\mathbf{X}^{(m)} = (X_1^{(m)}, \dots, X_T^{(m)})$ is determined by the graphical model $p(\mathbf{x}) = \frac{1}{Z} \prod_{t=1}^{T-1} \phi_{t, t+1}$. Here, ϕ_{ij} is the affinity between two areas i and j at time step $t \rightarrow t+1$. n_{ii} represents the number of people in area i at time step t , and n_{ij} represents the

³ In this paper, we define $\log 0 := -\infty$.

number of people who moved from area i to j at time step $t \rightarrow t + 1$. We have noisy observations y_{ti} for $t \in [T]$ and $i \in [R]$, which are generated by adding noise to n_{ti} . The MAP inference problem we want to solve is to find the sample size M , the true number of people of each area at each time step, $(n_{ti})_{t \in [T], i \in [R]}$, and the true number of people moving between each two areas, $(n_{tij})_{t \in [T-1], i, j \in [R]}$, with the highest posterior probability given the observation $(y_{ti})_{t \in [T], i \in [R]}$.

The MAP inference problem of (M, \mathbf{n}) can be formulated as follows from (2), (3), (6), (7):

$$\begin{aligned} \min_{M, \mathbf{n}} \quad & \sum_{t=1}^{T-1} \sum_{i, j \in [R]} f_{tij}(n_{tij}) + \sum_{t=2}^{T-1} \sum_{i \in [R]} g(n_{ti}) + \sum_{t=1}^T \sum_{i \in [R]} h_{ti}(n_{ti}) \\ & + k(M) + g(M) \\ \text{s.t.} \quad & \sum_{i \in [R]} n_{ti} = M \quad t \in [T], \\ & \sum_{j \in [R]} n_{tij} = n_{ti} \quad t \in [T-1], i \in [R], \\ & \sum_{i \in [R]} n_{tij} = n_{t+1, j} \quad t \in [T-1], j \in [R], \\ & n_{tij}, n_{ti}, M \in \mathbb{Z}_{\geq 0}, \end{aligned} \quad (8)$$

where $f_{tij}(z) = \log z! - z \cdot \log \phi_{tij}$, $g(z) = -\log z!$, $h_{ti}(z) = -\log p_{ti}(y_{ti} | z)$, $k(z) = -\log q(z) + z \log Z$. Note that Z is the partition function of the original graphical model (see (1)).

We can derive the optimization problem (8) as follows; because

$$v_t = \begin{cases} 1 & \text{if } t = 1, T, \\ 2 & \text{otherwise} \end{cases}$$

holds on path graphs, we have

$$F(M, \mathbf{n}) = \frac{M!}{Z^M} \cdot \frac{\prod_{t=2}^{T-1} \prod_{i \in [R]} n_{ti}!}{\prod_{t=1}^{T-1} \prod_{i, j \in [R]} n_{tij}!} \cdot \prod_{t=1}^{T-1} \prod_{i, j \in [R]} \phi_{tij}^{n_{tij}}$$

from (3). This gives

$$\begin{aligned} & -\log q(M) - \log F(M, \mathbf{n}) - \log \Pr(\mathbf{y} | \mathbf{n}) \\ = & -\log q(M) - \log M! + M \log Z - \sum_{t=2}^{T-1} \sum_{i \in [R]} \log n_{ti}! \\ & + \sum_{t=1}^{T-1} \sum_{i, j \in [R]} \log n_{tij}! - \sum_{t=1}^{T-1} \sum_{i, j \in [R]} n_{tij} \log \phi_{tij} - \sum_{t=1}^T \sum_{i \in [R]} \log p_{ti}(y_{ti} | n) \\ = & \sum_{t=1}^{T-1} \sum_{i, j \in [R]} f_{tij}(n_{tij}) + \sum_{t=2}^{T-1} \sum_{i \in [R]} g(n_{ti}) + \sum_{t=1}^T \sum_{i \in [R]} h_{ti}(n_{ti}) + k(M) + g(M) + C, \end{aligned}$$

where C is a constant. We can verify easily that the feasible region of problem (8) is $\mathbb{L}^{\mathbb{Z}}$ defined in (7).

4 Algorithms for MAP Inference Problem for CGMs on Path Graphs

4.1 Application of DCA

To solve problem (8), we propose utilizing the idea of the Difference of Convex Algorithm (DCA). Before describing our method, we review a discrete version of DCA (Maehara and Murota 2015; Maehara et al. 2018).

DCA is a general framework to solve the minimization problem

$$\min_{\mathbf{x} \in D} \left\{ \mathcal{P}(\mathbf{x}) := \mathcal{Q}(\mathbf{x}) + \mathcal{R}(\mathbf{x}) \right\}, \quad (9)$$

where $D \subseteq \mathbb{Z}^d$, $\mathcal{Q} : D \rightarrow \mathbb{R}$ is a discrete convex function, and $\mathcal{R} : D \rightarrow \mathbb{R}$ is a discrete concave function. Here, we define the convexity for discrete functions as follows.⁴

Definition 1 Let $D \subseteq \mathbb{Z}^d$. A discrete function $f : D \rightarrow \mathbb{R}$ is called convex if for all $\mathbf{x} \in D$, there exists an affine function $\tilde{f} : \mathbb{R}^d \rightarrow \mathbb{R}$ such that

$$\begin{aligned} \tilde{f}(\mathbf{x}) &= f(\mathbf{x}), \\ \tilde{f}(\mathbf{y}) &\leq f(\mathbf{y}) \quad (\forall \mathbf{y} \in D). \end{aligned}$$

A discrete function f is called concave if $-f$ is convex.

To solve problem (9), DCA generates a solution sequence $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots$ by the following procedure:

Step 1 Choose an arbitrary feasible solution $\mathbf{x}^{(1)} \in D$ and set the iteration counter as $s \leftarrow 1$.

Step 2 Find a function $\tilde{\mathcal{R}}^{(s)} : \mathbb{R}^d \rightarrow \mathbb{R}$ such that

$$\begin{aligned} \tilde{\mathcal{R}}^{(s)}(\mathbf{x}^{(s)}) &= \mathcal{R}(\mathbf{x}^{(s)}), \\ \tilde{\mathcal{R}}^{(s)}(\mathbf{x}) &\geq \mathcal{R}(\mathbf{x}) \quad (\forall \mathbf{x} \in D). \end{aligned}$$

Note that such a function exists since \mathcal{R} is discrete concave.

Step 3 Set

$$\mathbf{x}^{(s+1)} \in \arg \min_{\mathbf{x} \in D} \left\{ \tilde{\mathcal{P}}^{(s)}(\mathbf{x}) := \mathcal{Q}(\mathbf{x}) + \tilde{\mathcal{R}}^{(s)}(\mathbf{x}) \right\} \quad (11)$$

and $s \leftarrow s + 1$. Go to Step 2.

Because

$$\mathcal{P}(\mathbf{x}^{(s+1)}) \leq \tilde{\mathcal{P}}^{(s)}(\mathbf{x}^{(s+1)}) \leq \tilde{\mathcal{P}}^{(s)}(\mathbf{x}^{(s)}) = \mathcal{P}(\mathbf{x}^{(s)}),$$

the objective function value monotonically decreases: $\mathcal{P}(\mathbf{x}^{(1)}) \geq \mathcal{P}(\mathbf{x}^{(2)}) \geq \dots$.

⁴ There are several possible definitions of discrete convexity, and the convexity in Definition 1 is equivalent to the one called *convex extensibility* in the previous literature (Murota 1998).

To apply the DCA framework, the objective function must be expressed as the sum of convex and concave functions. The following proposition shows that our MAP inference problem in (8) has such a structure.

Proposition 3 *Let D be the feasible region of problem (8), and define discrete functions $\mathcal{Q} : D \rightarrow \mathbb{R}$ and $\mathcal{R} : D \rightarrow \mathbb{R}$ by*

$$\begin{aligned}\mathcal{Q}(M, \mathbf{n}) &:= \sum_{t=1}^{T-1} \sum_{i,j \in [R]} f_{ij}(n_{ij}) + \sum_{t=1}^T \sum_{i \in [R]} h_{ti}(n_{ti}) + k(M), \\ \mathcal{R}(M, \mathbf{n}) &:= \sum_{t=2}^{T-1} \sum_{i \in [R]} g(n_{ti}) + g(M).\end{aligned}$$

Under Assumption 1, \mathcal{Q} is discrete convex. \mathcal{R} is discrete concave.

The proof is given in the "Appendix". Hereafter, we set functions \mathcal{Q} and \mathcal{R} as in Proposition 3. As the objective function of problem (8) is written as $\mathcal{P}(M, \mathbf{n}) = \mathcal{Q}(M, \mathbf{n}) + \mathcal{R}(M, \mathbf{n})$, we can apply DCA to our problem.

The following proposition explicitly provides an efficiently computable upper bound of \mathcal{R} we can use in Step 2.

Proposition 4 *Define a function $\bar{\mathcal{R}}^{(s)} : \mathbb{R}^d \rightarrow \mathbb{R}$ by*

$$\bar{\mathcal{R}}^{(s)}(M, \mathbf{n}) := \sum_{t=2}^{T-1} \sum_{i \in [R]} \bar{g}_{ti}^{(s)}(n_{ti}) + \bar{g}^{(s)}(M),$$

where

$$\begin{aligned}\bar{g}_{ti}^{(s)}(z) &:= -\log(n_{ti}^{(s)}!) + \alpha_{ti}^{(s)} \cdot (z - n_{ti}^{(s)}) \\ \bar{g}^{(s)}(z) &:= -\log(M^{(s)}!) + \gamma^{(s)} \cdot (z - M^{(s)})\end{aligned}$$

and $\alpha_{ti}^{(s)}$ is a real number which satisfies $-\log(n_{ti}^{(s)} + 1) \leq \alpha_{ti}^{(s)} \leq -\log n_{ti}^{(s)}$ and $\gamma^{(s)}$ is a real number which satisfies $-\log(M^{(s)} + 1) \leq \gamma^{(s)} \leq -\log M^{(s)}$. Then, the function $\bar{\mathcal{R}}^{(s)}$ satisfies

$$\begin{aligned}\bar{\mathcal{R}}^{(s)}(M^{(s)}, \mathbf{n}^{(s)}) &= \mathcal{R}(M^{(s)}, \mathbf{n}^{(s)}), \\ \bar{\mathcal{R}}^{(s)}(M, \mathbf{n}) &\geq \mathcal{R}(M, \mathbf{n}) \quad (\forall (M, \mathbf{n}) \in D).\end{aligned}$$

Please see the "Appendix" for the proof.

4.2 Minimum cost flow algorithm for the subroutine

The most important and difficult part to derive an efficient DCA-based algorithm is designing efficient algorithms for subproblem (11). To achieve this, we show that the subproblem can be formulated as the Minimum Convex Cost Flow Problem (C-MCFP), which is the efficiently solvable subclass of the Minimum Cost Flow Problem (MCFP). The (non-linear) MCFP is a combinatorial optimization problem on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node $i \in \mathcal{V}$ has a supply value $b_i \in \mathbb{Z}$, and each edge $(i, j) \in \mathcal{E}$ has a cost function

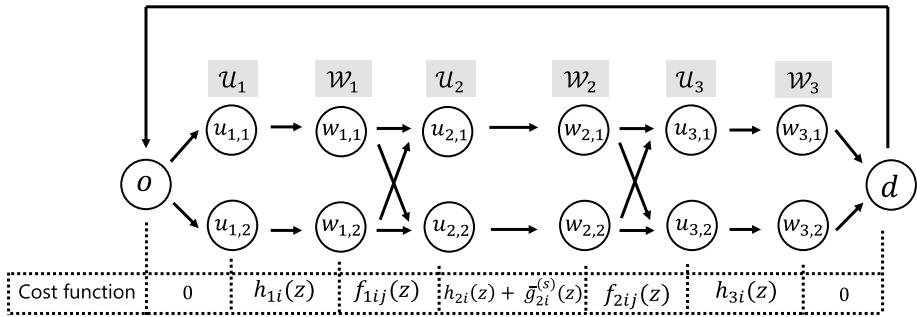


Fig. 3 An example of the MCFP instance defined in Proposition 5 when $T = 3$ and $R = 2$

$c_{ij} : \mathbb{Z}_{\geq 0} \rightarrow \mathbb{R} \cup \{+\infty\}$. MCFP is the problem of finding a minimum cost flow on \mathcal{G} that satisfies the supply constraints at all nodes. MCFP can be described as follows:

$$\min_{z \in \mathbb{Z}_{\geq 0}^{|\mathcal{E}|}} \sum_{(i,j) \in \mathcal{E}} c_{ij}(z_{ij}) \quad \text{s.t.} \quad \sum_{j: (i,j) \in \mathcal{E}} z_{ij} - \sum_{j: (j,i) \in \mathcal{E}} z_{ji} = b_i \quad i \in \mathcal{V}.$$

Note that z takes only integer values (i.e., $z \in \mathbb{Z}^{|\mathcal{E}|}$). A subclass of MCFP in which all cost functions are discrete convex functions (see Definition 1) is called the C-MCFP; it is known to be efficiently solvable (Ahuja et al. 1993).

The following proposition shows that the subproblem $\min_{(M,n) \in D} \bar{\mathcal{P}}^{(s)}(M,n)$ can be formulated as a C-MCFP.

Proposition 5 Define the MCFP instance as follows:

- the node set \mathcal{V} is defined by $\mathcal{V} := \{o, d\} \cup (\cup_{t \in [T]} (\mathcal{U}_t \cup \mathcal{W}_t))$, where $\mathcal{U}_t := (u_{t,i})_{i \in [R]}$, $\mathcal{W}_t := (w_{t,i})_{i \in [R]}$
- the edge set \mathcal{E} consists of five types of edges,
 - edges $(o, u_{1,i}, 0)$ and $(w_{T,i}, d, 0)$ for $i \in [R]$,
 - edges $(u_{t,i}, w_{t,i}, h_{ti}(z))$ for $t = 1, T$ and $i \in [R]$,
 - edges $(u_{t,i}, w_{t,i}, \bar{g}_{ti}^{(s)}(z) + h_{ti}(z))$ for $t = 2, \dots, T-1$ and $i \in [R]$,
 - edges $(w_{t,i}, u_{t+1,i}, f_{tij}(z))$ for $t \in [T-1]$ and $i, j \in [R]$,
 - an edge $(d, o, k_1(z) + \bar{g}^{(s)}(z))$,

where $(u, v, c(z))$ represents a directed edge from node u to node v with cost function $c(z)$,

- the supply values $(b_i)_{i \in \mathcal{V}}$ are defined by $b_v = 0$ for $v \in \mathcal{V}$.

Let z^* is an optimal solution of this MCFP instance, and define M^* by $M^* := z_{d,o}^*$, n^* by $n_{ii}^* := z_{u_{t,i}w_{t,i}}^*$ and $n_{tij}^* := z_{w_{t,i}u_{t+1,j}}^*$. Then, (M^*, n^*) is an optimal solution of the problem $\min_{(M,n) \in D} \bar{\mathcal{P}}^{(s)}(M,n)$. Furthermore, the MCFP instance belongs to C-MCFP.

The proof is given in the "Appendix". Figure 3 illustrates an example of the MCFP instance defined in Proposition 5. The above proposition enables us to solve the subproblem

$\min_{(M,n) \in D} \tilde{\mathcal{P}}^{(s)}(M,n)$ efficiently by applying existing algorithms for C-MCFP. A minimum cost flow problem such that $b_v = 0$ for all $v \in \mathcal{V}$, as with this problem, is called a *minimum cost circulation problem* (Tardos 1985).

Algorithm 1 DCA for problem (8)

```

1:  $(M^{(1)}, n^{(1)}) \leftarrow \mathbf{0}$ 
2: for  $s = 1, 2, \dots$  do
3:    $(M^{(s+1)}, n^{(s+1)}) \leftarrow ((M^*, n^*) \text{ defined in Proposition 5})$ 
4:   if  $\mathcal{P}(M^{(s)}, n^{(s)}) = \mathcal{P}(M^{(s+1)}, n^{(s+1)})$  then
5:     return  $(M^{(s)}, n^{(s)})$ 
6:   end if
7: end for
  
```

4.3 Overall view of the proposed method and time complexity analysis

From the above arguments, we can construct an efficient optimization algorithm, described in Algorithm 1, for the MAP inference Problem (8). Under the assumption that the support of the prior distribution $q(M)$ is finite, the algorithm is guaranteed to terminate after a finite number of iterations because $\mathcal{P}(M^{(s)}, n^{(s)})$ monotonically decreases and D is a finite set.

We analyze the time complexity of one iteration of the proposed method (Lines 3–5 in Algorithm 1). The computational bottleneck is solving C-MCFP in Line 3. There are several algorithms to solve C-MCFP and time complexity varies depending on which one is adopted. In this paper, we consider two typical methods, the successive shortest path algorithm (SSP) and the capacity scaling algorithm (CS) (Ahuja et al. 1993).

SSP is an algorithm that successively augments unit flow along the shortest path from a supply node (i.e. $b_i > 0$) to a demand node (i.e. $b_i < 0$) in the *residual graph*, which is an auxiliary graph calculated from the current flow. Given a C-MCFP instance with graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the shortest path in the residual graph can be computed in $O(|\mathcal{E}| \log |\mathcal{V}|)$ time by Dijkstra's algorithm with a binary heap, and the augmentation of the flow can be done in $O(|\mathcal{E}|)$ time. The augmentation is performed $B := (\sum_{i \in \mathcal{V}} |b_i|)/2$ times totally, so the total time complexity is $O(B|\mathcal{E}| \log |\mathcal{V}|)$. CS resembles SSP, but it differs in that it tries to push a large amount of flow, rather than a unit amount of flow, in a single augmentation. In CS, the number of shortest path calculations and flow augmentations can be bounded $O(|\mathcal{E}| \log U)$ times, where $U := \max_{i \in \mathcal{V}} |b_i|$, so the total computational complexity is $O(|\mathcal{E}|^2 \log |\mathcal{V}| \log U)$.

Although $b_v = 0, \forall v \in \mathcal{V}$ in the C-MCFP instance constructed in Proposition 5, we have to push M_{\max} flow from o to d beforehand to eliminate the negative cost edge, where M_{\max} is the maximum value M can take (i.e. the maximum value of the support of the prior distribution $q(M)$). Therefore, $B = \Theta(M_{\max})$ and $U = \Theta(M_{\max})$ holds in our problem. Because $|\mathcal{V}| = O(TR)$ and $|\mathcal{E}| = O(TR^2)$, the time complexity in one iteration is $O(M_{\max} TR^2 \log(TR))$ when SSP is applied and $O(T^2 R^4 \log(TR) \log M_{\max})$ when CS is applied. As a special case, the time complexity for the MAP inference problem with a given sample size (the problem setting in previous studies) with SSP is $O(M_{\text{given}} TR^2 \log(TR))$ and with CS is $O(T^2 R^4 \log(TR) \log M_{\text{given}})$ where M_{given} is the given sample size, because $M_{\max} = M_{\text{given}}$. These result imply that each method has its own advantages and disadvantages: SSP has

small time complexity for T and R , while CS has small time complexity for M_{\max} or M_{given} . This difference is confirmed empirically in Sect. 5.

4.4 Discussions

Here, we discuss why it is possible to construct an efficient algorithm when the graph is a path graph. The difficulty of MAP inference in CGM can be decomposed into the following two factors. The first is the non-convexity of the objective function; the objective function is the sum of convex functions and concave functions, as expressed in (8), and the objective function as a whole is not convex. The second is many constraints; as can be seen from (7), the variables M and n has a lot of complex constraints, all of which must be considered. The proposed method addresses difficulty (i) with the discrete DCA. Difficulty (ii) is addressed by restricting the graph to path graphs. An essential property of path graphs is that they only contain vertices of degree 2 or less. Because the constraints on vertices of degree 2 or less can be expressed as flow conservation laws, we can formulate the problem as a minimum convex cost flow problem by constructing an appropriate flow network.

When considering graphical models on tree graphs other than path graphs, there are always vertices of degree 3 or higher in the graph, and constraints around these vertices cannot be expressed as flow-preserving laws. Therefore, to extend our approach to graphs other than path graphs, it is necessary to develop a different method than network flow to handle constraints at vertices of degree 3 or higher. We leave this extension to future work.

5 Experiments

We perform experiments to evaluate the effectiveness of the proposed methods. All experiments are conducted on a 64-bit macOS machine with Intel Core i7 CPUs and 16 GB of RAM. All algorithms are implemented in C++ (gcc 9.1.0 with -O3 option). Experiments are conducted both in the existing problem setting where sample size is given as input and in the new problem setting where no sample size is given. The experiments are as follows:

- Experiments using synthetic instances with a given sample size in 5.1,
- Experiments using synthetic instances without a given sample size in 5.2,
- Experiments using real-world instances with and without a given sample size in 5.3,
- Experiments on histogram interpolation, one of the applications of MAP inference, with synthetic instances in 5.4.

5.1 Synthetic instances with a given sample size

5.1.1 Settings

We solve randomly generated synthetic instances of the MAP inference problem (8). We fix T to 5 and vary the values of R and M . We use two types of potential functions as follows.

1. **uniform.** ϕ_{ij} is independently drawn from a uniform distribution on the set of integers $\{1, 5, 10\}$.

2. **distance.** We set $\phi_{ij} = \frac{1}{|i-j+1|}$. This potential models the movement of individuals in one-dimensional space: the state indices i and j represent coordinates in the space, and the closer the two points are, the more likely are movements between them to occur.

The input observations \mathbf{y} are generated by the following procedure; first, we generate independent M samples by Gibbs sampling in the graphical model (Bishop and Nasrabadi 2006), then we calculate the true contingency tables \mathbf{n}^{true} by aggregating the generated samples, and finally we get \mathbf{y} according to the Gaussian observation distribution $p_{ii}(y_{ii} | n_{ii}) \propto \exp\left(-\frac{(y_{ii}-n_{ii})^2}{10}\right)$. The sample size M is given as input of the algorithms.

To construct surrogate functions in the proposed method, we can choose arbitrary $\alpha_{ii}^{(s)}$ which satisfies the condition $-\log(n_{ii}^{(s)} + 1) \leq \alpha_{ii}^{(s)} \leq -\log n_{ii}^{(s)}$ (see Proposition 4). To investigate the influence of the choice of $\alpha_{ii}^{(s)}$, we try three strategies to decide $\alpha_{ii}^{(s)}$: (1) $\alpha_{ii}^{(s)} = -\log(n_{ii}^{(s)})$, (2) $\alpha_{ii}^{(s)} = -\frac{1}{2}(\log(n_{ii}^{(s)}) + \log(n_{ii}^{(s)} + 1))$, (3) $\alpha_{ii}^{(s)} = -\log(n_{ii}^{(s)} + 1)$. We call them Proposed (L), Proposed (M), Proposed (R), respectively. Note that when the sample size is given, the term $k(M) + g(M)$ in the objective function of (8) can be ignored because M is a constant, and it is not necessary to determine $\gamma^{(s)}$.

As the compared method, we use Non-Linear Belief Propagation (NLBP) (Sun et al. 2015), which is a message-passing style algorithm to solve approximate MAP inference problem derived by applying Stirling's approximation and continuous relaxation. Because the output of NLBP is not integer-valued and $\log(z!)$ is defined only if z is an integer, we cannot calculate the objective function of (8) directly. To address this, we calculate it by replacing the term $\log(z!)$ by linear interpolation of $\log(\lfloor z \rfloor!)$ and $\log(\lceil z \rceil!)$, which is given by $(\lceil z \rceil - z) \cdot \log(\lfloor z \rfloor!) + (z - \lfloor z \rfloor) \cdot \log(\lceil z \rceil!)$. Note that although there are various algorithms to solve the approximate MAP inference problem (see Sect. 2.1), the objective function values attained by these algorithms are the same. This is because the approximate problem is a convex optimization problem (Sheldon et al. 2013).

We also calculated and compared the error between the solution output by each algorithm and the ground truth contingency table which was generated by Gibbs sampling and aggregation. The error metric we used is normalized absolute error (NAE), which is defined as

$$\frac{\sum_{t \in [T-1]} \sum_{i \in [R]} \sum_{j \in [R]} |n_{ij}^{\text{true}} - n_{ij}^{\text{est}}|}{\sum_{t \in [T-1]} \sum_{i \in [R]} \sum_{j \in [R]} n_{ij}^{\text{true}}},$$

where n_{ij}^{true} is the ground truth value of the edge contingency table and n_{ij}^{est} is the estimated value of the edge contingency table. Note that NAE is a metric that has been used frequently in previous studies of CGM (Tanaka et al. 2018; Akagi et al. 2018; Iwata and Shimizu 2019).

5.1.2 Results

First, we compare the attained objective values and NAEs. The results are shown in Table 1. We generate 10 instances for each parameter setting and determined the average of attained objective function values. Because the objective function $\mathcal{P}(\mathbf{n})$ is equal to $-\log \Pr(\mathbf{n} | \mathbf{y}) + \text{const.}$, $\mathcal{P}(\mathbf{n})$ takes both positive and negative values, and the difference of the objective function values is essential; when $\mathcal{P}(\mathbf{n}_1) - \mathcal{P}(\mathbf{n}_2) = \delta$, $\Pr(\mathbf{n}_1 | \mathbf{y}) = \exp(-\delta) \cdot \Pr(\mathbf{n}_2 | \mathbf{y})$ holds.

Table 1 Attained objective function values and NAEs in synthetic instances with a given sample size

	<i>M</i>	10 ¹			10 ²			10 ³		
		10	20	30	10	20	30	10	20	30
Obj. Vals. (uniform)	Proposed (L)	− 9.18e+01	− 7.98e+01	− 6.51e+01	− 1.16e+03	− 1.21e+03	− 1.21e+03	− 1.16e+04	− 1.39e+04	− 1.48e+04
	Proposed (M)	− 9.13e+01	− 7.89e+01	− 6.40e+01	− 1.16e+03	− 1.21e+03	− 1.21e+03	− 1.16e+04	− 1.39e+04	− 1.48e+04
	Proposed (R)	− 9.07e+01	− 7.81e+01	− 6.27e+01	− 1.16e+03	− 1.21e+03	− 1.21e+03	− 1.16e+04	− 1.39e+04	− 1.48e+04
	NLBP	− 6.13e+01	− 3.04e+01	− 5.26e+00	− 1.12e+03	− 1.05e+03	− 9.49e+02	− 1.16e+04	− 1.38e+04	− 1.44e+04
NAE (uniform)	Proposed (L)	1.430	1.800	1.825	0.612	1.136	1.461	0.198	0.427	0.623
	Proposed (M)	1.490	1.825	1.835	0.614	1.135	1.459	0.198	0.428	0.624
	Proposed (R)	1.490	1.820	1.830	0.613	1.143	1.453	0.198	0.427	0.624
	NLBP	1.636	1.897	1.949	0.654	1.288	1.630	0.198	0.434	0.660
	<i>M</i>	10 ¹			10 ²			10 ³		
	<i>R</i>	10	20	30	10	20	30	10	20	30
Obj. Vals. (distance)	Proposed (L)	1.72e+01	3.59e+01	5.29e+01	− 4.99e+01	− 6.46e+00	2.70e+01	− 6.53e+02	− 1.31e+03	− 1.29e+03
	Proposed (M)	1.77e+01	3.59e+01	5.29e+01	− 4.98e+01	− 5.49e+00	2.85e+01	− 6.53e+02	− 1.31e+03	− 1.29e+03
	Proposed (R)	1.78e+01	3.59e+01	5.29e+01	− 4.97e+01	− 4.70e+00	3.06e+01	− 6.53e+02	− 1.31e+03	− 1.29e+03
	NLBP	5.28e+01	9.47e+01	1.29e+02	− 4.25e+00	1.96e+02	3.58e+02	− 6.36e+02	− 1.19e+03	− 7.99e+02
NAE (distance)	Proposed (L)	1.495	1.715	1.825	0.666	1.141	1.324	0.209	0.436	0.648
	Proposed (M)	1.475	1.715	1.830	0.667	1.143	1.320	0.209	0.436	0.647
	Proposed (R)	1.485	1.715	1.830	0.667	1.141	1.323	0.209	0.436	0.647
	NLBP	1.633	1.845	1.916	0.678	1.254	1.501	0.208	0.431	0.653

For each setting, we generated 10 instances and average values are shown. The smallest value is highlighted for each setting. The top table shows the result of “uniform” potential and the bottom table shows the result of “distance” potential. The proposed methods attain smaller objective function values and NAEs in many cases

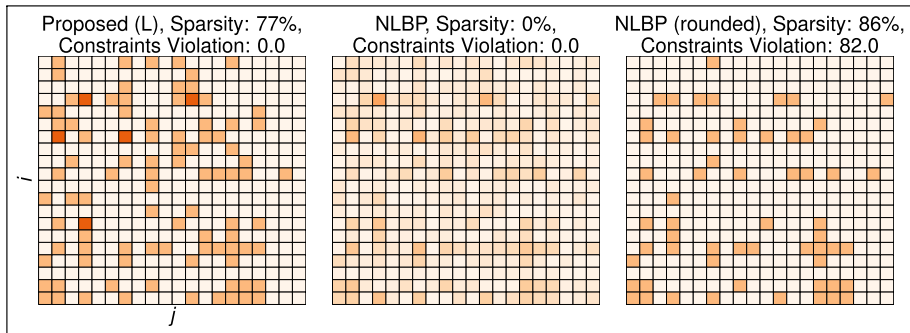


Fig. 4 Comparison of solutions yielded by proposed method (L), NLBP, and NLBP (rounded). We solve an instance with $R = 20$, $M = 10^2$ and uniform potential. The obtained edge contingency table n_{ij} is presented as a matrix heatmap with the maximum value of color map 3. Sparsity is defined by (12) and Constraints Violation is defined by (13)

All the proposed methods consistently have smaller objective function values than the compared method. The difference tends to be large when R is large and M is small. This would be because small values appear in the contingency table more frequently when R is large and M is small, and the effect of the inaccuracy of Stirling's approximation becomes larger. Among the proposed methods, proposed (L) achieves the smallest objective function values in most cases. This finding is considered to be an important guideline for determining hyperparameters $\alpha_{ii}^{(s)}$.

Furthermore, in most cases, all the proposed methods achieve smaller NAEs than the existing method. As with the objective function values, the differences tend to be larger when R is large or M is small. This confirms the superior performance of the proposed method in terms of estimation accuracy. However, NLBP achieves a slightly smaller NAE when $R = 10$, $M = 10^3$ or $R = 20$, $M = 10^3$ with the distance potential. This may be due to the fact that the values included in the contingency table are large enough that Stirling's approximation becomes accurate and NLBP is able to output a good solution. Among the three proposed methods, there was not much difference in NAEs. This indicates that the proposed method is robust with respect to the choice of hyperparameters $\alpha_{ii}^{(s)}$ in terms of estimation accuracy.

To compare the characteristics of solutions obtained by proposed (L) and NLBP, we solve an instance with $R = 20$, $M = 10^2$, and uniform potential by each method. Obtained edge contingency tables n_{ij} are shown in Fig. 4 as heat maps. We also show the edge contingency table obtained by rounding each element of the NLBP solution to the nearest integer. We observe that the proposed method outputs sparse solutions while the solutions by NLBP are blurred and contain a lot of non-zero elements. This difference is quantified by “sparsity”, which is calculated by

$$\left(1.0 - \frac{(\# \text{ of non - zero elements})}{(\# \text{ of elements})}\right) \times 100 (\%). \quad (12)$$

Sparsity of the output of proposed (L) is 77%, while the sparsity of the output of NLBP is 0%. This is caused by its application of continuous relaxation and the inaccuracy of Stirling's approximation around 0. For NLBP (rounded), many near-zero values are rounded to 0 and the solution is sparser than the Proposed (L) solution. But the constraints of the

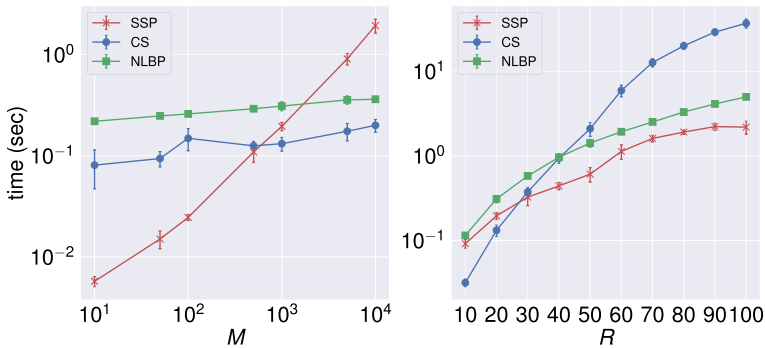


Fig. 5 The computation time of each algorithm. The values are averages of 10 synthetic instances when R is fixed to 20 (left) and M is fixed to 10^3 (right). T is set 5 and the uniform potential is used. This result indicates a trade-off between SSP and CS in terms of computation time depending on the value of M and R

problem (8) are totally violated; for example, the sum of the edge contingency table values does not match the sample size. To quantitatively evaluate the constraints violation, we compute

$$\left| \sum_{i \in [R]} n_{1i}^{\text{est}} - M \right| + \sum_{i \in [R]} \left| \sum_{j \in [R]} n_{1ij}^{\text{est}} - n_{1i}^{\text{est}} \right| + \sum_{j \in [R]} \left| \sum_{i \in [R]} n_{1ij}^{\text{est}} - n_{2j}^{\text{est}} \right| \quad (13)$$

for each solution \mathbf{n}^{est} (we call this value “constraints violation”). Each term in this formula corresponds to a constraint of the optimization problem (8), and this value measures the violation of constraints of the optimization problem (8). For the solutions output by Proposed (L) and NLBP, this value is 0, indicating that the constraints are not violated, but for NLBP (rounded), the value is 82.0, indicating that the constraints are violated drastically. In additional experiments, we observed that the outputs of the three methods become closer as M increases.

We compare the computation time of each algorithm. As explained in Sect. 4.3, we can choose an arbitrary C-MCFP algorithm as the subroutine in the proposed method and the time complexity varies depending on the choice. We compare proposed (L) with SSP, proposed (L) with CS, and NLBP.

Figure 5 shows the relationship between input size and computation time. These results are consistent with the complexity analysis results in Sect. 4.3; SSP is efficient when R is large but becomes inefficient when M is large, and the converse is true for CS. The results also suggest that it is important to choose the algorithm depending on the size parameter of the input. The proposed method is not much worse than the existing method in terms of computation time by choosing an appropriate C-MCFP algorithm according to the size of the input. Figure 6 shows the relationship between running time and objective function value of each method in instances of $R = 20, M = 10^4$ and $R = 30, M = 10^3$ with uniform potential. These results show that the properly selected proposed method reaches smaller objective function values more quickly than the existing method.

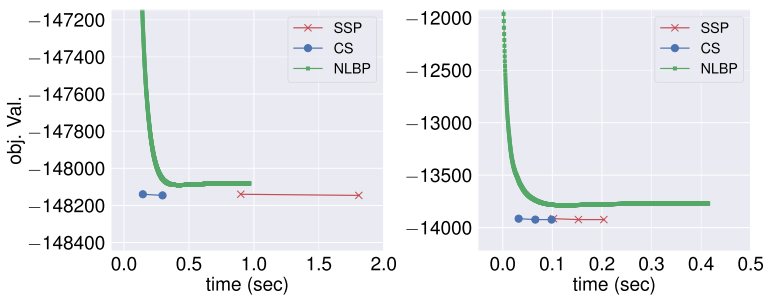


Fig. 6 The relationship between running time and objective function value. The left figure shows the result of an instance of $R = 20, M = 10^4$ and uniform potential, and right figure shows that of $R = 30, M = 10^3$ and uniform potential. This result shows that the properly selected proposed method reaches smaller objective function values more quickly than the existing method

5.2 Synthetic instances without a given sample size

5.2.1 Settings

We compared methods for MAP inference problem without a given sample size using random instances generated in the same way as in Sect. 5.1. Note that the sample size used to generate random instances is not given as input to the algorithms. We use Gaussian distribution $p_{ii}(y_{ii} | n_{ii}) \propto \exp(- (y_{ii} - n_{ii})^2)$ as the noise distribution and the uniform distribution $q_{\text{uniform}, 10^6}(M)$ defined by (5) as the prior distribution of M .

We prepare two proposed methods; one is to solve the instance of C-MCFP constructed in Proposition 5 using SSP (DCA-SSP) and the other is to solve it using CS (DCA-CS). In both methods, we set $\alpha_{ii}^{(s)} = -\log(n_{ii}^{(s)})$, $\gamma^{(s)} = -\log M^{(s)}$ when constructing the surrogate function (see Proposition 5).

Because no methods for MAP inference of the sample size have been proposed so far, we use a naive approach to solve the optimization problem (8) as baselines; solving the problem for fixed M by the algorithm for the MAP inference problem with a given sample size with a brute force search of $M \in \{0, 1, \dots, M_{\max}\}$. We prepare two baseline methods, one using SSP (BF-SSP) and the other using CS (BF-CS), for solving the problem (8) with fixed M . We set $M_{\max} = \max(10, 2 \cdot \sum_{i=1}^R y_{1i})$.

5.2.2 Results

First, we compare the attained objective values and NAEs by DCA-SSP, DCA-CS, BF-SSP, BF-CS. The results are shown in Table 2. As shown, the four methods achieve nearly identical objective function values and NAEs in all cases. BF-SSP and BF-CS are naive methods that conduct brute force searches of the sample size, so the objective function value achieved by them is considered to be the smallest among the currently available methods. The fact that the proposed methods, DCA-SSP and DCA-CS, achieve almost the same objective function values as BF-SSP and BF-CS indicates that the optimization performances of the proposed methods are sufficiently high.

Second, we compare the computation time of the four methods. Figure 7 shows the relationship between input size and computation time. It can be seen that the two proposed methods are much faster than the two baseline methods. In particular, the difference is

Table 2 Attained objective function values and NAEs by the MAP inference methods without a given sample size in synthetic instances

<i>M</i>	10 ¹			10 ²			10 ³		
	10	20	30	10	20	30	10	20	30
Obj. Vals. (uniform)	DCA-SSP	7.33e+01	1.07e+02	1.41e+02	2.93e+02	5.94e+02	7.86e+02	6.81e+02	1.81e+03
	DCA-CS	7.33e+01	1.07e+02	1.41e+02	2.93e+02	5.95e+02	7.86e+02	6.81e+02	1.81e+03
	BF-SSP	7.31e+01	1.07e+02	1.41e+02	2.93e+02	5.94e+02	7.84e+02	6.81e+02	1.81e+03
	BF-CS	7.31e+01	1.07e+02	1.41e+02	2.93e+02	5.94e+02	7.84e+02	6.81e+02	1.81e+03
NAE (uniform)	DCA-SSP	1.020	1.055	1.065	0.597	1.065	1.244	0.205	0.432
	DCA-CS	1.000	1.045	1.065	0.589	1.073	1.244	0.205	0.432
	BF-SSP	1.020	1.050	1.065	1.072	1.072	1.265	0.205	0.432
	BF-CS	0.970	1.045	1.060	0.590	1.079	1.273	0.205	0.431
<hr/>									
<i>M</i>	10 ¹			10 ²			10 ³		
	10	20	30	10	20	30	10	20	30
Obj. Vals. (distance)	DCA-SSP	7.32e+01	1.09e+02	1.39e+02	3.15e+02	5.39e+02	6.55e+02	7.13e+02	1.88e+03
	DCA-CS	7.32e+01	1.09e+02	1.39e+02	3.15e+02	5.39e+02	6.56e+02	7.13e+02	1.88e+03
	BF-SSP	7.29e+01	1.09e+02	1.39e+02	3.15e+02	5.38e+02	6.54e+02	7.13e+02	1.88e+03
	BF-CS	7.29e+01	1.09e+02	1.39e+02	3.15e+02	5.38e+02	6.54e+02	7.13e+02	1.88e+03
NAE (distance)	DCA-SSP	0.890	1.100	1.085	0.660	1.007	1.156	0.210	0.642
	DCA-CS	0.890	1.095	1.085	0.659	1.004	1.149	0.210	0.642
	BF-SSP	0.925	1.110	1.080	0.661	1.015	1.170	0.210	0.642
	BF-CS	0.915	1.110	1.080	0.660	1.013	1.171	0.210	0.642

For each setting, we generated 10 instances and average values are shown
The smallest value is highlighted for each setting. The top table shows the result of “uniform” potential and the bottom table shows the result of “distance” potential. The proposed methods (DCA-SSP and DCA-CS) attain almost the same objective function values and NAEs as baseline methods (BF-SSP and BF-CS)

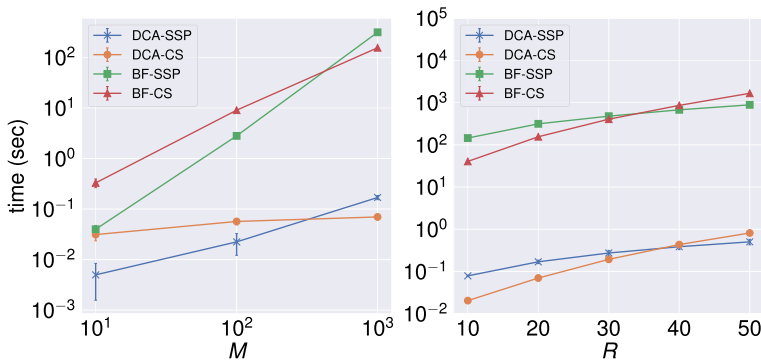


Fig. 7 The computation time of the MAP inference methods without a given sample size in synthetic instances. The values are averages of 10 synthetic instances when R is fixed to 20 (left) and M is fixed to 10^3 (right). T is set 5 and the uniform potential is used. The result indicates the proposed methods (DCA-SSP and DCA-CS) solve the problem much faster than baseline methods (BF-SSP and BF-CS)

significantly larger when M is large. This is because the proposed method needs to apply DCA only once, while the baseline method needs to apply it M_{\max} times. In the comparison between proposed methods, DCA-CS is faster when M is large, and DCA-SSP is faster when R is large. This result is consistent with the results of the time complexity analysis.

Overall, the proposed methods achieve almost the same objective function values and NAEs as the baseline methods in much smaller computation times, demonstrating the effectiveness of the proposed methods.

5.3 Real-world Instances

We conduct experiments using real-world population datasets in both settings where a sample size is given and not given.

5.3.1 Settings

The datasets are generated from 8694 car trajectories collected by a car navigation application in the Greater Tokyo area, Japan⁵. We randomly sample M ($M = 100, 500, 1000$) trajectories from this data and create aggregated population data of each area at fixed time intervals. The areas are decided by dividing the targeted geospatial space into fixed-size grid cells. The grid size is set to $10 \text{ km} \times 10 \text{ km}$ ($R = 8 \times 7 = 56$) and $5 \text{ km} \times 5 \text{ km}$ ($R = 16 \times 13 = 208$), and time interval is 60 min ($T = 24$).

Aggregated population data of humans often contain noise based on the following scenarios. The first scenario is that it is difficult to obtain accurate aggregate values due to the poor performance of the sensors and equipment used to observe location information. The second scenario is that noises are added artificially to aggregate values when published to prevent identifying individual information. To take such a situation into account, we assume noisy observation. As the noise distribution, we use Gaussian distribution $p_{ii}(y_{ii} | n_{ii}) \propto \exp(-(y_{ii} - n_{ii})^2)$.

⁵ We use the data collected by the smartphone car navigation application of NAVITIME JAPAN Co., Ltd. (<http://corporate.navitime.co.jp/en/>). The data are collected with consent and appropriately anonymized.

Table 3 Attained objective function values and NAEs for real-world instances

		<i>M</i>		100		500		1000	
		<i>R</i>		56	208	56	208	56	208
Obj. Val.	Proposed (L)			– 3.02e+02	2.97e+02	– 6.61e+03	– 2.62e+03	– 1.70e+04	– 9.76e+03
	NLBP			1.30e+03	3.24e+03	– 3.46e+03	6.93e+03	– 1.34e+04	4.70e+03
	NLBP (rounded)			–	–	–	–	–	–
NAE	Proposed (L)			0.690	0.441	1.002	0.829	1.073	0.956
	NLBP			1.38	1.544	1.241	1.438	1.209	1.381
	NLBP (rounded)			0.877	0.870	1.079	0.907	1.128	0.991
		<i>M</i>		100		500		1000	
		<i>R</i>		56	208	56	208	56	208
Obj. Val.	DCA-SSP			3.11e+03	3.44e+03	1.06e+04	1.67e+04	1.68e+04	2.94e+04
	DCA-CS			3.11e+03	3.45e+03	1.06e+04	1.67e+04	1.68e+04	2.94e+04
NAE	DCA-SSP			0.721	0.741	1.003	0.820	1.070	0.947
	DCA-CS			0.719	0.745	1.003	0.821	1.070	0.948
Estimated <i>M</i>	DCA-SSP			76.0	40.9	491.5	404.9	995.9	922.6
	DCA-CS			76.0	40.6	491.6	404.6	995.9	922.1

The top table shows the results for a problem with a given sample size, and the bottom table shows the results for a problem without a given sample size. The bottom table also shows the sample sizes estimated by each method. For each setting, we generated 10 instances and averages are shown. The smallest values in each table are highlighted

We construct the potential $\phi_{ij} = \exp(-\text{dist}(i, j))$, where $\text{dist}(i, j)$ is the Euclidean distance between the centers of cell i and cell j in the grid space. We create 10 instances by random sampling and averaged the attained objective function values for each setting. We also evaluate the estimation accuracy of the edge contingency table $(n_{ij})_{i \in [T-1], j \in [R]}$ by NAE.

We solved the MAP inference problem in both problem settings where the sample size is given and not given. In the former setting, the compared methods are proposed (L) and NLBP. In addition, we also evaluate the estimation accuracy of NLBP (rounded), which is a method that rounds the output of NLBP to integer values. Note that we do not evaluate the objective function values of NLBP (rounded). This is because the output of NLBP (rounded) completely violates the constraints on summation in the MAP inference problem (8) and the objective function value cannot evaluate whether the optimization problem is successfully solved or not. In the setting where the sample size is not given, the compared methods are DCA-SSP and DCA-CS, which is explained in Sect. 5.2. The surrogate function and the prior distribution for the sample size are the same as those depicted in Sect. 5.2.

5.3.2 Results

Table 3 shows the results. The top table shows the results for a problem with a given sample size, and the bottom table shows the results for a problem without a given sample size. The bottom table also shows the sample sizes estimated by each method.

First, in the top table, we observe that Proposed (L) consistently attain smaller objective values and NAEs than the existing method. The superiority of the proposed method increase when R is large and M is small, and this is the same trend as the results with synthetic data. The NAE of NLBP is relatively large; this is due to the fact that small values are assigned to the elements of the output that should be 0, which is the same phenomenon seen in Fig. 4. The NAE values are improved to some extent by rounding, but the proposed method is still superior.

In the setting where the sample size is not given (the bottom table in Table 3), the two proposed methods (DCA-SSP and DCA-CS) achieved similar objective function values and NAEs, suggesting that the effect of the method of solving the minimum convex cost flow problem on the estimation performance is small. For cases other than $M = 100$, the proposed method achieves the same level of NAE as the problem setting where the sample size is given. This indicates that the proposed method can successfully estimate the sample size even when the sample size is not given. Note that it is not very meaningful to compare the objective function values between the setting where the sample size is given and not because their objective functions are different. For $M = 100$, NAEs are worse than in the problem setting where the sample size is given. This may be because the estimated sample size (the value of “estimated M ” in the table) is underestimated compared to the true value. When the problem is highly sparse (i.e., M is small and R is large), the sample size tends to be underestimated (this tendency is also confirmed in experiments using synthetic data). There are possible ways to get around this, such as using a probability distribution that assigns more probability to large values as a prior distribution for sample size.

5.4 Histogram interpolation

As an application of MAP inference of CGMs on path graphs, we can interpolate the time series of histograms. This application is useful, for example, when the population distribution can only be observed at rough time intervals. By estimating the population distribution at fine time intervals through interpolation, it will be possible to understand the movement of crowds in more detail.

In this section, we show experimental results on this application and discuss the differences between the output of the proposed method and that of the existing method.

5.4.1 Settings

First, we briefly explain how to realize interpolation between two histograms by MAP inference of CGMs on path graphs. Suppose we are given histogram $\boldsymbol{\eta}_1 := [\eta_{11}, \dots, \eta_{1R}]$ at time 1 and the histogram $\boldsymbol{\eta}_T := [\eta_{T1}, \dots, \eta_{TR}]$ at time T . The interpolated histogram $\boldsymbol{\eta}_t$ at time t ($t = 2, \dots, T - 1$) is calculated by the following procedure.

1. Consider a CGM on a path graph with T vertices.
2. Let $\mathbf{y}_1 = \boldsymbol{\eta}_1$ and $\mathbf{y}_T = \boldsymbol{\eta}_T$.
3. \mathbf{y}_t ($t = 2, \dots, T - 1$) is treated as a *missing value*. This can be achieved by setting $h_{ti}(z) = 0$ ($t = 2, \dots, T - 1, i \in [R]$) in the objective function of the problem (8).

Fig. 8 Three examples of interpolation results yielded by each method. In each example, three sequences of histograms in the two-dimensional grid space are presented; the first row shows the input histograms η_1 and η_T , the second row shows the interpolation results obtained by proposed (L), and the third row shows the interpolation results obtained by NLBP

4. Find a solution n^* to the MAP inference problem under an appropriate potential ϕ .
5. Obtain an interpolation result by $\eta_{ti} = n_{ti}^* (t = 2, \dots, T - 1, i \in [R])$.

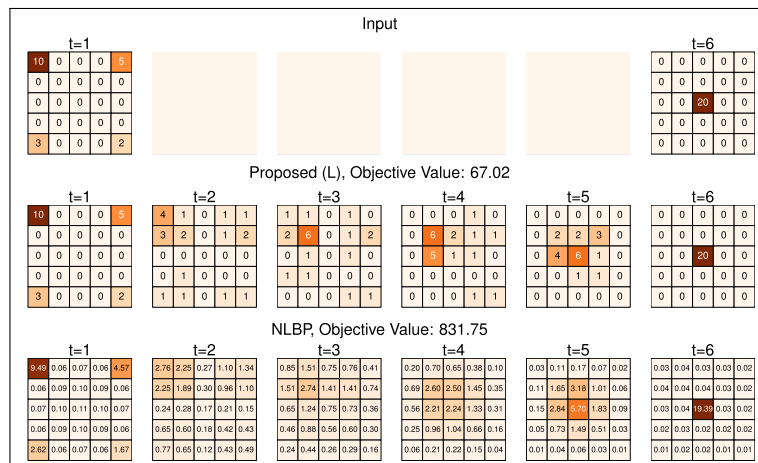
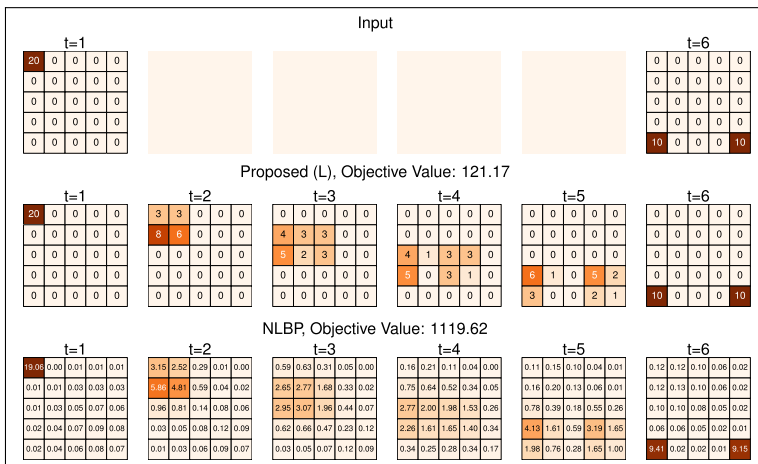
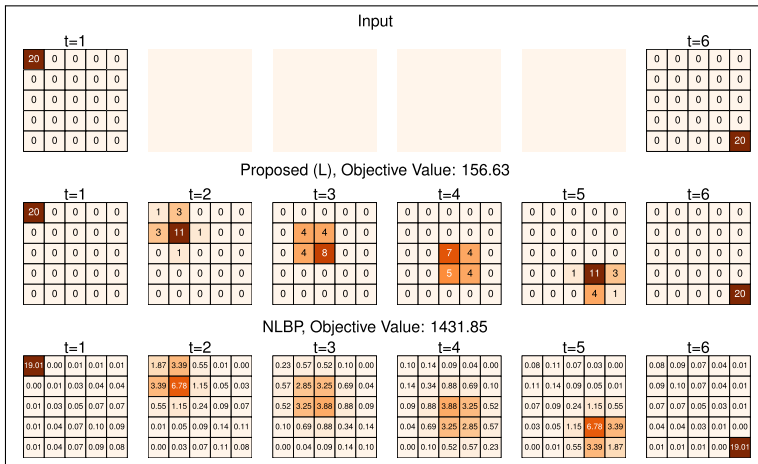
In our experiment, we consider a grid space of size $5 \times 5 = 25 (= R)$ and a histogram $\eta := [\eta_1, \dots, \eta_R]$ with a value η_i for each cell $i (= 1, \dots, R)$. To get interpolation results which consider the geometric structure defined by Euclidean distance in the grid space, we set the potential $\phi_{ij} = \exp(-(r_i - r_j)^2 - (c_i - c_j)^2)$, where (r_i, c_i) is the two-dimensional coordinate of the center of cell i in the grid space. We set $T = 6$ and use Gaussian distribution $p_{ti}(y_{ti} | n_{ti}) \propto \exp(-5(y_{ti} - n_{ti})^2)$ for the noise distributions at $t = 1, T$. The sample size M is given as input, and M is set to 20.

5.4.2 Results

The results are shown in Fig. 8. Note that Fig. 8 illustrates different objects from what is shown in Figs 4; 8 illustrates the interpolated node contingency table values n_{ti} as two-dimensional grid spaces, while Fig. 4 illustrate edge contingency table values n_{1ij} as matrices. As shown in the figure, NLBP tends to assign non-zero values to many cells, while proposed (L) assigns non-zero values to a small number of cells, resulting in sparse solutions. Moreover, the outputs of the proposed (L) are integer-valued while those of NLBP are not. This characteristic of the proposed method is beneficial for interpretability when the histogram values are the numbers of countable objects (e.g., the number of people in the area). The figure also describes the objective function values achieved. In all cases, the objective function values achieved by the proposed method are much smaller than those by NLBP, which confirms that the proposed method is able to output interpolation results with large posterior probabilities. Note that the sparse and sharp nature of the output of the proposed method may mislead the readers to believe that there is a solution with a posteriori probability significantly higher than the others; there are many solutions with almost identical posterior probabilities around the MAP solution in the MAP inference problem we are solving here. Care should be taken when interpreting the output of the proposed method.

6 Conclusion

In this paper, we propose a non-approximate method to solve the MAP inference problem for CGMs on path graphs. Our algorithm is based on an application of DCA. In the algorithm, surrogate functions can be constructed in closed-form and minimized efficiently by C-MCFP algorithms. Our method is naturally applicable to problem settings where sample size is not given as input. Experimental results show the effectiveness of our algorithms both in the quality of solutions and computation time.



Appendix A Proofs

A.1 Proof of Proposition 3

Proof First, we prove two lemmas.

Lemma 6 For $i = 1, \dots, n$, let $\phi_i : \mathbb{Z} \rightarrow \mathbb{R}$ be a discrete convex function. Then, the function $\phi : \mathbb{Z}^n \rightarrow \mathbb{R}$ defined by

$$\phi(z_1, \dots, z_n) = \sum_{i=1}^n \phi_i(z_i)$$

is also discrete convex.

Proof Fix $\mathbf{z} = (z_1, \dots, z_n)$ arbitrarily. By the assumption, we have affine functions $\bar{\phi}_1, \dots, \bar{\phi}_n : \mathbb{R} \rightarrow \mathbb{R}$ such that

$$\begin{aligned}\bar{\phi}_i(z_i) &= \phi_i(z_i), \\ \bar{\phi}_i(y) &\leq \phi_i(y) \quad (\forall y \in \mathbb{Z}).\end{aligned}$$

By defining $\bar{\phi} : \mathbb{R}^n \rightarrow \mathbb{R}$ by

$$\bar{\phi}(y_1, \dots, y_n) = \sum_{i=1}^n \bar{\phi}_i(y_i),$$

we have an affine function that shows the discrete convexity of ϕ . □

Lemma 7 Let $\phi : \mathbb{Z} \rightarrow \mathbb{R}$ be a discrete function. The function ϕ is discrete convex if and only if the following holds for all $z \in \mathbb{Z}$:

$$\phi(z+1) + \phi(z-1) \geq 2\phi(z). \quad (\text{A1})$$

Proof \implies part: Assume that ϕ is discrete convex. Fix $z \in \mathbb{Z}$ arbitrarily, and we have an affine function $\bar{\phi}$ such that

$$\begin{aligned}\bar{\phi}(z) &= \phi(z), \\ \bar{\phi}(y) &\leq \phi(y) \quad (\forall y \in \mathbb{Z}).\end{aligned}$$

We then obtain the desired results as follows:

$$\phi(z+1) + \phi(z-1) \geq \bar{\phi}(z+1) + \bar{\phi}(z-1) = 2\bar{\phi}(z) = 2\phi(z),$$

where the first equality holds since $\bar{\phi}$ is affine.

\Leftarrow part: Assume that (A1) holds for all $z \in \mathbb{Z}$. Fix $x \in \mathbb{Z}$ arbitrarily, and define an affine function $\bar{\phi} : \mathbb{R} \rightarrow \mathbb{R}$ by

$$\bar{\phi}(z) = (\phi(x+1) - \phi(x))(z-x) + \phi(x).$$

We have $\bar{\phi}(x) = \phi(x)$. To prove the discrete convexity of ϕ , it suffices to show that $\bar{\phi}(y) \leq \phi(y)$ for all $y \in \mathbb{Z}$. For $y > x$, since

$$\phi(x+1) - \phi(x) \leq \phi(x+2) - \phi(x+1) \leq \dots \leq \phi(y) - \phi(y-1)$$

by (A1), we obtain the desired inequality as follows:

$$\begin{aligned}\bar{\phi}(y) &= \phi(x) + (\phi(x+1) - \phi(x))(y-x) = \phi(x) + \sum_{i=x}^{y-1} (\phi(x+1) - \phi(x)) \\ &\leq \phi(x) + \sum_{i=x}^{y-1} (\phi(i+1) - \phi(i)) = \phi(y).\end{aligned}$$

For $y < x$, it follows from the same argument that $\bar{\phi}(y) \leq \phi(y)$. \square

From Lemma 6, it suffices to show that $f_{ij}(z)$, $-g(z)$, $h_{ii}(z)$, $k(z)$ are univariate discrete convex functions. From Lemma 7, the proof is completed by showing

$$f_{ij}(z+2) + f_{ij}(z) \geq 2f_{ij}(z+1) \quad \forall z \in \mathbb{Z}, \quad (\text{A2})$$

$$-g(z+2) - g(z) \geq -2g(z+1) \quad \forall z \in \mathbb{Z}, \quad (\text{A3})$$

$$h_{ii}(z+2) + h_{ii}(z) \geq 2h_{ii}(z+1) \quad \forall z \in \mathbb{Z}, \quad (\text{A4})$$

$$k(z+2) + k(z) \geq 2k(z+1) \quad \forall z \in \mathbb{Z}. \quad (\text{A5})$$

The inequality (A2) holds because

$$\begin{aligned}&f_{ij}(z+2) + f_{ij}(z) - 2f_{ij}(z+1) \\ &= \log(z+2)! - (z+2) \log \phi_{ij} + \log z! - z \log \phi_{ij} - 2 \log(z+1)! + 2(z+1) \log \phi_{ij} \\ &= \log(z+2) - \log(z+1) \geq 0.\end{aligned}$$

The inequality (A3) holds because

$$\begin{aligned}&-g_{ij}(z+2) - g_{ij}(z) + 2g_{ij}(z+1) \\ &= \log(z+2)! + \log z! - 2 \log(z+1)! \\ &= \log(z+2) - \log(z+1) \geq 0.\end{aligned}$$

The inequality (A4) holds because $-\log[p_{ii}(y_{ii} | z)]$ is a continuous convex function in z from Assumption 1. The inequality (A5) holds from Assumption 2. \square

A.2 Proof of Proposition 4

Proof First, we show that

$$-\log(w!) + \alpha \cdot (z-w) \geq -\log(z!), \quad \forall z \in \mathbb{Z}_{\geq 0} \quad (\text{A6})$$

holds for arbitrary $w \in \mathbb{Z}_{\geq 0}$, when $-\log(w+1) \leq \alpha \leq -\log w$. When $z \geq w$,

$$-\log(w!) + \alpha \cdot (z-w) + \log(z!) = \sum_{k=w+1}^z (\alpha + \log k) \geq 0$$

holds because $\alpha + \log(w + 1) \geq 0$. When $z < w$,

$$-\log(w!) + \alpha \cdot (z - w) + \log(z!) \sum_{k=z+1}^w (-\alpha - \log k) \geq 0$$

holds because $-\alpha - \log w \geq 0$. Thus, inequality (A6) holds.

Substituting $w = n_{ii}^{(s)}$ and $w = M^{(s)}$ in (A6), we get $\bar{g}_{ii}^{(s)}(z) \geq g(z)$ and $\bar{g}^{(s)}(z) \geq g(z)$ for all $z \in \mathbb{Z}_{\geq 0}$. This yields

$$\bar{\mathcal{R}}^{(s)}(M, \mathbf{n}) = \sum_{i=2}^{T-1} \sum_{i=1}^R \bar{g}_{ii}^{(s)}(n_{ii}) + \bar{g}^{(s)}(M) \geq \sum_{i=2}^{T-1} \sum_{i=1}^R g(n_{ii}) + g(M) = \mathcal{R}(M, \mathbf{n}).$$

Furthermore, since $\bar{g}_{ii}^{(s)}(n_{ii}^{(s)}) = g(n_{ii}^{(s)})$ and $\bar{g}^{(s)}(M^{(s)}) = g(M^{(s)})$ from simple calculation, we get $\bar{\mathcal{R}}^{(s)}(M^{(s)}, \mathbf{n}^{(s)}) = \mathcal{R}(M^{(s)}, \mathbf{n}^{(s)})$. \square

A.3 Proof of Proposition 5

Proof There is a one-to-one correspondence between a feasible solution to the problem (8), (M, \mathbf{n}) , and a feasible solution to the MCFP instance constructed, \mathbf{z} , under the relationship $M = z_{d,o}$, $n_{ii} = z_{u_{i,i}w_{t,i}}$ and $n_{ij} = z_{w_{t,i}u_{t+1,j}}$; the constraint $\sum_{i \in [R]} n_{ii} = M$ corresponds to the flow conservation rule at node o and d , the constraint $\sum_{j \in [R]} n_{ij} = n_{ii}$ corresponds to the flow conservation rule at node $w_{t,i}$ and the constraint $\sum_{i \in [R]} n_{ij} = n_{i+1,j}$ corresponds to the flow conservation rule at node $u_{t+1,j}$. Moreover, corresponding (M, \mathbf{n}) and \mathbf{z} have the same objective function value in problem (8) and the MCFP instance, respectively. These facts yield that (M^*, \mathbf{n}^*) is the optimum solution of the problem (8). Because all the cost functions are discrete convex (this can be easily verified by (A2, A3, A4, A5) and definition of $\bar{g}_{ii}^{(s)}(z)$ and $\bar{g}^{(s)}(z)$ in Proposition 4), the constructed instance belongs to C-MCFP. \square

Funding This research is funded by the Nippon Telegraph and Telephone Corporation (NTT).

Availability of data and material The authors cannot make data public because it is the confidential information of NTT, the company for which they work. Code availability The authors cannot make codes public because it is the confidential information of NTT, the company for which they work.

Declarations

Conflicts of interest All authors are employed and paid by NTT.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Authors' contributions All authors contributed to the study conception. The details of the proposed methods were developed by YA and NM. Implementation and analysis of the experimental results were conducted by YA. The first draft of the manuscript was written by YA and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.


References

- Ahuja, R.K., Magnanti, T.L., & Orlin, J.B. (1993). *Network flows: Theory, algorithms, and applications*. Prentice-Hall Inc.
- Akagi, Y., Nishimura, T., Kurashima, T., & Toda, H. (2018). A fast and accurate method for estimating people flow from spatiotemporal population data. In *Proceedings of the 27th international joint conference on artificial intelligence and the 23rd European conference on artificial intelligence* (pp. 3293–3300).
- Akagi, Y., Nishimura, T., Tanaka, Y., Kurashima, T., & Toda, H. (2020). Exact and efficient inference for collective flow diffusion model via minimum convex cost flow algorithm. In *Proceedings of the 34th AAAI conference on artificial intelligence* (pp. 3163–3170).
- Akagi, Y., Marumo, N., Kim, H., Kurashima, T., & Toda, H. (2021). Non-approximate inference for collective graphical models on path graphs via discrete difference of convex algorithm. *Advances in Neural Information Processing Systems*, 34.
- Bishop, C.M., & Nasrabadi, N.M. (2006). *Pattern recognition and machine learning* (Vol. 4). Springer.
- Du, J., Kumar, A., & Varakantham, P. (2014). On understanding diffusion dynamics of patrons at a theme park. In *Proceedings of the 13th international conference on autonomous agents and multiagent systems* (pp. 1501–1502).
- Iwata, T., & Shimizu, H. (2019). Neural collective graphical models for estimating spatio-temporal population flow from aggregated data. In *Proceedings of the 33rd AAAI conference on artificial intelligence* (pp. 3935–3942).
- Iyer, R. & Bilmes, J. (2012). Algorithms for approximate minimization of the difference between submodular functions, with applications. In *Proceedings of the 28th conference on uncertainty in artificial intelligence* (pp. 407–417).
- Le Thi, A., Hoai, L., Minh, H., & Dinh, T.P. (2015). Feature selection in machine learning: An exact penalty approach using a difference of convex function algorithm. *Machine Learning*, 101(1), 163–186.
- Le Thi, H.A., & Pham Dinh, T. (2018). DC programming and DCA: Thirty years of developments. *Mathematical Programming*, 169(1), 5–68.
- Maehara, T., & Murota, K. (2015). A framework of discrete DC programming by discrete convex analysis. *Mathematical Programming*, 152(1–2), 435–466.
- Maehara, T., Marumo, N., & Murota, K. (2018). Continuous relaxation for discrete DC programming. *Mathematical Programming*, 169(1), 199–219.
- Morimura, T., Osogami, T., & Idé, T. (2013). Solving inverse problem of Markov chain with partial observations. *Advances in neural information processing systems*, 26, 1655–1663.
- Murota, K. (1998). Discrete convex analysis. *Mathematical Programming*, 83(1), 313–371.
- Narasimhan, M., & Bilmes, J. (2005). A submodular-supermodular procedure with applications to discriminative structure learning. In *Proceedings of the 21th conference on uncertainty in artificial intelligence* (pp. 404–412).
- Nguyen, T., Kumar, A., Lau, H.C., & Sheldon, D. (2016). Approximate inference using DC programming for collective graphical models. In *Proceedings of the 19th international conference on artificial intelligence and statistics* (pp. 685–693).

- Nitanda, A., & Suzuki, T. (2017). Stochastic difference of convex algorithm and its application to training deep Boltzmann machines. In *Proceedings of the 20th international conference on artificial intelligence and statistics* (pp. 470–478).
- Piot, B., Geist, M., & Pietquin, O. (2014). Difference of convex functions programming for reinforcement learning. *Advances in Neural Information Processing Systems*, 27, 2519–2527.
- Sheldon, D., & Dietterich, T.G. (2011). Collective graphical models. *Advances in Neural Information Processing Systems*, 24, 1161–1169.
- Sheldon, D., Saleh Elmohamed, M. A., & Kozen, D. (2007). Collective inference on Markov models for modeling bird migration. *Advances in Neural Information Processing Systems*, 20, 1321–1328.
- Sheldon, D., Sun, T., Kumar, A., & Dietterich, T. (2013). Approximate inference in collective graphical models. In *Proceedings of the 30th international conference on machine learning* (pp. 1004–1012).
- Singh, R., Haasler, I., Zhang, Q., Karlsson, J., & Chen, Y. (2020). Inference with aggregate data: An optimal transport approach. *CoRR*. [arXiv:abs/2003.13933](https://arxiv.org/abs/2003.13933).
- Sun, T., Sheldon, D., & Kumar, A. (2015). Message passing for collective graphical models. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 853–861.
- Suzuki, T., Yamashita, M., & Terada, M. (2013). Using mobile spatial statistics in field of disaster prevention planning. *NTT DOCOMO Technical Journal*, 14(3), 37–45.
- Tanaka, Y., Iwata, T., Kurashima, T., Toda, H., & Ueda, N. (2018). Estimating latent people flow without tracking individuals. In *Proceedings of the 27th international joint conference on artificial intelligence and the 23rd European conference on artificial intelligence* (pp. 3556–3563).
- Tanaka, Y., Tanaka, T., Iwata, T., Kurashima, T., Okawa, M., Akagi, Y., & Toda, H. (2019). Spatially aggregated Gaussian processes with multivariate areal outputs. *Advances in Neural Information Processing Systems*, 32, 3000–3031.
- Tardos, É. (1985). A strongly polynomial minimum cost circulation algorithm. *Combinatorica*, 5(3), 247–255.
- Terada, M., Nagata, T., & Kobayashi, M. (2013). Population estimation technology for mobile spatial statistics. *NTT DOCOMO Technical Journal*, 14(3), 10–15.
- Vilnis, L., Belanger, D., Sheldon, D., & McCallum, A. (2015). Bethe projections for non-local inference. In *Proceedings of the 31st conference on uncertainty in artificial intelligence* (pp. 892–901).
- Xu, H.-M., Xue, H., Chen, X.-H., & Wang, Y.-Y. (2017). Solving indefinite kernel support vector machine with difference of convex functions programming. In *Proceedings of the 31st AAAI conference on artificial intelligence* (pp. 2782–2788).
- Yuille, A.L., & Rangarajan, A. (2001). The concave-convex procedure (CCCP). *Advances in Neural Information Processing Systems*, 14.
- Zhang, S., Wu, G., Costeira, J.P., & Moura, J.M.F. (2017). Understanding traffic density from large-scale web camera data. In *Proceedings of the 30th IEEE conference on computer vision and pattern recognition* (pp. 5898–5907).
- Zhang, Y., Charoenphakdee, N., Zhenguo, W., & Sugiyama, M. (2020). Learning from aggregate observations. *Advances in Neural Information Processing Systems*, 33, 470–478.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Yasunori Akagi¹  · Naoki Marumo² · Hideaki Kim¹ · Takeshi Kurashima¹ · Hiroyuki Toda¹

Naoki Marumo
naoki.marumo.ec@hco.ntt.co.jp

Hideaki Kim
hideaki.kin.cn@hco.ntt.co.jp

Takeshi Kurashima
takeshi.kurashima.uf@hco.ntt.co.jp

Hiroyuki Toda
hiroyuki.toda.xb@hco.ntt.co.jp

- ¹ NTT Human Informatics Laboratories, NTT Corporation, 1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa 239-0847, Japan
- ² NTT Communication Science Laboratories, NTT Corporation, 2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan