Sinkhorn-Knopp アルゴリズムを利用した 時刻別エリア人口データからの移動人数推定

赤木 康紀† 田中 佑典† 倉島 健† 戸田 浩之†

† 日本電信電話株式会社 NTT サービスエボリューション研究所 〒 239–0847 神奈川県横須賀市光の丘 1-1

E-mail: †{yasunori.akagi.cu,yusuke.tanaka.rh,takeshi.kurashima.uf,hiroyuki.toda.xb}@hco.ntt.co.jp

あらまし 個人のプライバシーを保護しつつ大局的な群衆の位置情報を把握するための手がかりとして，時刻別エリア人口データが注目を集めている．時刻別エリア人口データとは，携帯電話やセンサなどから得られた位置情報を，各時刻・各エリアごとに人口という形で集計したデータである．本研究では，時刻別エリア人口データから，より詳細かつ活用しやすい情報である，エリア間の移動人数を推定するというタスクに取り組む．先行研究として，Collective Flow Diffusion Model (CFDM) を利用した推定手法が挙げられる．CFDM では，エリア間の移動人数を確率モデルで表現し，最大事後確率 (MAP) 推定に基づいて移動人数の推定を実現する．しかし，この手法には問題のサイズが大きくなった際に計算コストが膨大になるという問題点があった．本研究では，CFDM における MAP 推定問題が，最適輸送理論において重要な概念である Sinkhorn 距離を求める問題と近似的に等価であることを示す．この関係に基づき，Sinkhorn 距離の計算に利用される高速なアルゴリズムである Sinkhorn-Knopp アルゴリズムを応用し，CFDM の推定を効率的に行う新しい手法を提案する．シミュレーションデータを用いた実験を通して，提案手法が既存手法と比較して高速かつ高精度な推定を実現できることを確認した．

キーワード 時刻別エリア人口データ，移動人数推定，最適輸送，Sinkhorn-Knopp アルゴリズム

1 はじめに

GPS や Wi-Fi，センサなどの発達により，位置情報の重要性は大きくなり様々な形で利用されるようになっている．しかし，プライバシーへの懸念や個人を追跡し続けることの難しさから，個人の移動軌跡のデータを取得・利用することは依然として難しい．それに対し，時刻別エリア人口データは個人の詳細な移動の情報を含まないため比較的入手・利用しやすいことから注目を集めている．例えば，携帯電話の通信データから計算される，一定時間ごとの特定のサイズの正方形の領域における人口のデータであるモバイル空間統計 [1] はこのようなデータの一例である．別の例として，自動車などの通行データはそれぞれの車が追跡された形のデータよりも，道路に設置されたカメラやセンサによって得られる通行数という形式のデータで得られることが多い [2] [3]．このようなデータも時刻別エリア人口データの一種として捉えることができる．

時刻別エリア人口データには様々な使い道があるものの，移動に関する情報を陽に含まないため，その応用可能性は限られている．このようなデータを有効活用するために，時刻別エリア人口データから時刻間における移動人数を推定する研究が盛んに行われている．その代表的な手法の一つとして，Collective Flow Diffusion Model (CFDM) [4] と呼ばれる確率モデルを利用したものが挙げられる．これは，人々の移動をマルコフ連鎖によってモデリングし，集計化された各エリアの人口のみからモデルの学習を可能にする手法であり，交通ネットワーク [4] や都市空間 [5] [6]，アミューズメントパーク [7] や展示会場 [8] における人流の推定に応用されている．

この手法は強力であるが，問題の規模（エリアの数，エリア内の総人口など）が大きくなった際に推定にかかる計算コストが膨大になるという問題点がある．推定は移動人数を隠れ変数とみなした Expectation-Maximization (EM) アルゴリズムによって行われる．EM アルゴリズムは，確率モデルのパラメータを固定したもとで隠れ変数の期待値をとる E ステップと，隠れ変数を固定したもとで最尤推定によって確率モデルのパラメータを求める M ステップからなり，解が収束するまで E ステップと M ステップを交互に繰り返すことで隠れ変数と確率モデルのパラメータを同時に求める．この中で計算のボトルネックとなるのが E ステップであり，MCMC を用いた手法 [9] や凸最適化法を用いた手法 [10]，ネットワークフロー [11] を用いた手法などが提案されているがいずれも計算コストが高く，問題サイズが大きい場合には実用的ではない．

この問題を解決するため，本論文では新しい推定手法を提案する．提案手法のアイデアは，最適輸送理論と呼ばれる分野において重要な概念である Sinkhorn 距離 [12] の計算と CFDM における E ステップの関係を利用することにある．Sinkhorn 距離とは，2 つの確率分布の近さを測る距離の一種であり，機械学習の様々な分野で利用されるようになってきている．Sinkhorn 距離が注目を集めている理由の 1 つとして，Sinkhorn-Knopp アルゴリズムと呼ばれるアルゴリズムを利用して高速に計算することが可能であることが挙げられる．本論文では，Sinkhorn 距離を計算するための最適化問題と，CFDM における E ステップを近似的に行うための最適化問題が本質的に等価な問題であることを示す．この関係性により，CFDM における E ス

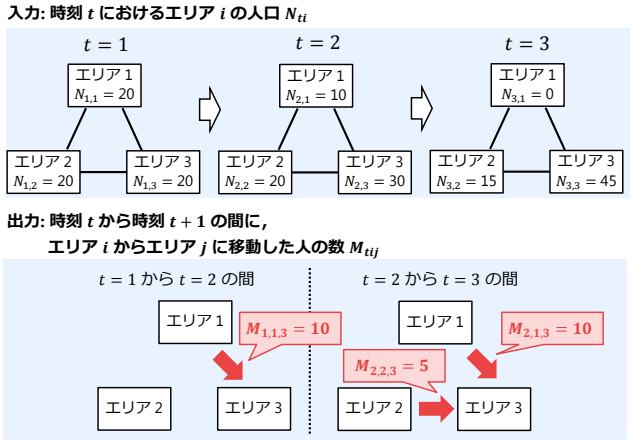


図 1　問題設定の一例．エリア数 n が 3，タイムステップの総数 T が 3 の場合を示している．

テップを高速な Sinkhorn-Knopp アルゴリズムを用いて効率的に実行することが可能になり，計算コストの大幅な削減が可能になる．本論文では，CFDM における E ステップに対して Sinkhorn-Knopp アルゴリズムを適用した場合のアルゴリズムを導出し，2 通りの実装方法（行列を用いた実装，グラフを用いた実装）を提案する．前者は操作が単純な行列の繰り返しからなるため GPGPU との相性が良く，複数の問題を並列して解くことが可能であるというメリットがあるのに対し，後者は移動確率行列が疎である場合に時間計算量及び空間計算量を大きく削減できるというメリットがある．

最後に，提案手法の有効性を確認するため，人の移動を模倣したシミュレーションデータを用いた実験を行った．その結果，様々なパラメータ設定において，提案手法は既存手法と比較して高速に解を求められると同時に，高い推定精度も達成することが確認された．

本論文の構成は以下である．2 章で本論文が扱う問題を数学的に定式化する．3 章では Collective Flow Diffusion Model (CFDM) 及び最適輸送理論に関する必要知識を述べる．4 章で提案手法について説明し，5 章で実験結果を報告する．6 章で関連研究について紹介し，7 章でまとめを述べる．

2　問 題 設 定

正の整数 k について，$[k] := \{1, \ldots, k\}$ とする．対象となる空間は n 個のエリアに分割されているとする．エリア i に時刻 t にいた人は，時刻 $t+1$ には i にとどまるか別のエリア $j \in [n] \setminus \{i\}$ に移動するとする．この手続きは T を観測時刻の総数として $t \in [T-1]$ について繰り返されるとする．

本論文で取り組む問題の入力と出力は以下である．

- 入力

時刻 t におけるエリア i の人口 $N_{t,i} (i \in [n], t \in [T])$.

- 出力

時刻 t から $t+1$ にかけてエリア i からエリア j に移動した人数 $M_{tij} (i \in [n], j \in [n], t \in [T-1])$.

図 1 は問題設定の一例を表している．

3　準　　備

3.1　Collective Flow Diffusion Model (CFDM)

$\boldsymbol{\theta}_i = \{\theta_{ij}\}_{j \in [n]}$ $(\sum_{j \in [n]} \theta_{ij} = 1)$ をエリア i からエリア $j \in [n]$ に移動する確率とする．ここで，移動確率 $\boldsymbol{\theta}_i$ は時刻 t に依存しないものとする．各時刻各エリアにおける人口 $N_{t,i}$ と移動確率 $\boldsymbol{\theta}_i$ が与えられたとき，移動人数 $\boldsymbol{M}_{ti} = \{M_{tij}\}_{j \in [n]}$ $(t \in [T-1], i \in [n])$ は $\boldsymbol{M}_{ti} \sim \mathrm{Multi}(N_{t,i}, \boldsymbol{\theta}_i)$ という多項分布に従って決定されると仮定する．$\boldsymbol{N} = \{N_{t,i} \mid t \in [T], i \in [n]\}$ と $\boldsymbol{M} = \{\boldsymbol{M}_{ti} \mid t \in [T-1], i \in [n]\}$ が与えられたとしたとき，$\boldsymbol{\theta} = \{\boldsymbol{\theta}_i \mid i \in [n]\}$ の尤度関数は

$$P(\boldsymbol{M} \mid \boldsymbol{N}, \boldsymbol{\theta}) \propto \prod_{t=1}^{T-1} \prod_{i \in [n]} \left(\frac{N_{t,i}!}{\prod_{j \in [n]} M_{tij}!} \prod_{j \in [n]} \theta_{ij}^{M_{tij}} \right) \quad (1)$$

となる．さらに，$N_{t,i}$ と $\boldsymbol{M}_{ti}$ の間には

$$N_{t,i} = \sum_{j \in [n]} M_{tij} \ (t \in [T-1], i \in [n]), \quad (2)$$

$$N_{t+1,i} = \sum_{j \in [n]} M_{tji} \ (t \in [T-1], i \in [n]) \quad (3)$$

という関係が成り立つ．等式 (2)(3) は観測された人口と移動人数の間の整合性を表す．

我々の目的は，$\boldsymbol{N}$ が与えられたもとで $\boldsymbol{M}$ を推定することである．これを実現するために，CFDM を用いた手法では $\boldsymbol{M}$ を隠れ変数とみなし，EM アルゴリズムを適用することによって $\boldsymbol{M}$ を $\boldsymbol{\theta}$ と同時に推定する．EM アルゴリズムは，パラメータ $\boldsymbol{\theta}$ を固定したもとで隠れ変数 $\boldsymbol{M}$ の期待値をとる E ステップと，$\boldsymbol{M}$ を固定したもとで最尤推定によって $\boldsymbol{\theta}$ を求める M ステップからなり，解が収束するまで E ステップと M ステップを交互に繰り返すことで $\boldsymbol{M}$ と $\boldsymbol{\theta}$ を求める．

この推定プロセスにおいて，計算のボトルネックとなるのは E ステップである．最も基本的な E ステップの実現方法は，MCMC による近似的な $\boldsymbol{M}$ の期待値計算 [9] であるが，十分な精度で近似を行うに大量のサンプルの生成が必要であり，スケーラビリティに問題がある．この問題を解決するため，通常の期待値計算を最大事後確率 (MAP) 推定によって代替する手法が提案され [10]，以後の研究において広く用いられている [5] [6] [8].

3.2　最 適 輸 送

最適輸送理論とは,「ある確率分布を別の確率分布に最小のコストで輸送する」方法に関する理論である [13].「最適な輸送をした場合のコスト」は二つの確率分布の間の距離として用いることができ，最適輸送距離と呼ばれる．最適輸送距離は，KL ダイバージェンスや全変動距離といった伝統的な確率分布間の距離尺度と比較して良い性質を持っていることが近年明らかになっており，機械学習の様々な分野において用いられるようになってきている [14] [15] [16].

（状態空間が離散的な場合の）最適輸送の数学的な定式化について説明する．$\boldsymbol{p}, \boldsymbol{q} \in \mathbb{R}^d$ を確率ベクトル（各要素が非負で

あり，要素和が 1 であるベクトル）とする．行列 $\boldsymbol{c} \in \mathbb{R}^{d\times d}$ について，以下の最適化問題を考える：

$$\begin{aligned} \max_{\boldsymbol{m}}. \quad & \sum_{i\in[d]}\sum_{j\in[d]} c_{ij}\cdot m_{ij} \\ \text{s.t.} \quad & p_i = \sum_{j\in[d]} m_{ij} \quad (i\in[d]), \\ & q_j = \sum_{i\in[d]} m_{ij} \quad (j\in[d]), \\ & m_{ij} \geqq 0 \quad (i\in[d], j\in[d]). \end{aligned} \tag{4}$$

この最適化問題の最適値を $\boldsymbol{p}$ と $\boldsymbol{q}$ の最適輸送距離，最適解 $\boldsymbol{m}^*$ を最適輸送行列と呼ぶ．$\boldsymbol{c}$ が距離の公理を満たす行列になっている場合，最適輸送距離も距離の公理を満たすことが知られている．最適化問題 (4) は線形計画問題であるため，LP ソルバーなどを用いることによって多項式時間で最適解を求めることができる．しかし，$O(d^3)$ の計算量を必要とするため，d が大きくなった場合に計算量が非常に大きくなってしまうという問題があった．

この問題を解決するために，Sinkhorn 距離と呼ばれる概念が近年提案され [12]，最適輸送理論の応用のための大きなブレークスルーとなった．Sinkhorn 距離は，最適輸送距離の良い近似でありつつも非常に高速に計算することが可能であり，また距離の公理を満たすなどの良い性質を持つ．Sinkhorn 距離は

$$\begin{aligned} \min_{\boldsymbol{m}}. \quad & \sum_{i\in[d]}\sum_{j\in[d]} (c_{ij}\cdot m_{ij} + \lambda\cdot m_{ij}\log m_{ij}) \\ \text{s.t.} \quad & p_i = \sum_{j\in[d]} m_{ij} \quad (i\in[d]), \\ & q_j = \sum_{i\in[d]} m_{ij} \quad (j\in[d]), \\ & m_{ij} \geqq 0 \quad (i\in[d], j\in[d]) \end{aligned} \tag{5}$$

という最適化問題の最適値として定義される．ただし，$\lambda > 0$ ハイパーパラメータである．もとの最適化問題 (4) との違いは，目的関数の各項に $\lambda\cdot m_{ij}\log m_{ij}$ という値（これは $\{m_{ij}\}_{i,j}$ を確率分布としてみたときのエントロピーを表す）が足されている点である．λ が 0 に近いとき Sinkhorn 距離は最適輸送距離に近くなり，$\lambda \to 0$ のとき両者は一致する．

Sinkhorn 距離は，Sinkhorn-Knopp アルゴリズムと呼ばれるアルゴリズムによって非常に高速に計算することが可能である．Sinkhorn-Knopp アルゴリズムの概略を Algorithm 1 に示す．このアルゴリズムは単純な行列計算の繰り返しからなるため非常に高速に動作することに加え，並列化が容易であり GPGPU との相性も良いという利点がある．実験的にも，Sinkhorn 距離は最適輸送距離の良い近似になりつつ非常に高速に求められることが示されている [12].

Algorithm 1 Sinkhorn-Knopp アルゴリズム

Input: 確率ベクトル $\boldsymbol{p}, \boldsymbol{q}$, コスト行列 $\boldsymbol{c}$, ハイパーパラメータ $\lambda > 0$
Output: 最適化問題 (1) の最適解 $\boldsymbol{m}^*$

```
行列 K ∈ R^{d×d} を K_ij ← exp(−λ·c_ij) で計算
ベクトル u ∈ R^d を [1, 1, ..., 1] で初期化
while not convergence do
    v ← q./(K^⊤ u)  （ただし ./ は要素ごとの除算を意味する）
    u ← p./(K v)
end while
U ← diag(u), V ← diag(v)
m* ← UKV
```

4 提 案 手 法

4.1 提案手法のアイデア

提案手法では，CFDM における EM アルゴリズムの E ステップの期待値計算を MAP 推定によって置き換える．これは，多項分布の期待値と最頻値が近似的に一致するため，E ステップにおける期待値計算を MAP 推定値で置き換えることが可能であるという観察に基づいた方法であり [10]，E ステップの計算量を大幅に削減することができる．この置き換えを行った場合は，EM アルゴリズムの実現のためには MAP 推定問題を繰り返し解く必要があり，この部分が全体の計算のボトルネックとなる．この MAP 推定問題を解くための手法として，連続緩和及び近似を適用して凸計画問題に変形して解く方法や [10]，ネットワークフローを用いて解く手法が提案されている [11].

本論文では，MAP 推定問題を解くための新しい手法を提案する．提案手法の主なアイデアは，この MAP 推定のための最適化問題と，最適輸送理論における Sinkhorn 距離を求めるための最適化問題が非常によく似ていることに着目することにある．本論文では，適切な近似及び変数の置き換えのもと，2 つの最適化問題が実は等価な問題であることを示した．この結果，Sikhorn 距離を求めるためのアルゴリズムであり，非常に高速に動作することが知られている Sinkhorn-Knopp アルゴリズムを CFDM の MAP 推定問題にも適用できることになり，高速な MAP 推定を実現することが可能になる．

4.2 問題の変形

MAP 推定問題は

$$\begin{aligned} \max_{\boldsymbol{M}}. \quad & \log P(\boldsymbol{M} \mid \boldsymbol{N}, \boldsymbol{\theta}) \\ \text{s.t.} \quad & N_{t,i} = \sum_{j\in[n]} M_{tij} \quad (t\in[T-1], i\in[n]), \\ & N_{t+1,i} = \sum_{j\in[n]} M_{tji} \quad (t\in[T-1], i\in[n]), \\ & M_{tij} \in \mathbb{Z}_{\geqq 0} \quad (t\in[T-1], i\in[n], j\in[n]) \end{aligned} \tag{6}$$

という問題になる．この問題は，直感的には $\boldsymbol{N}$ と $\boldsymbol{\theta}$ が与えられた際に，最も生じた可能性が高い $\boldsymbol{M}$ を求める問題になっている．ここで目的関数に (1) を代入し，-1 を掛けた上で $\boldsymbol{M}$ に依存しない項を省略すると $\sum_{t\in[T-1]}\sum_{i\in[n]}\sum_{j\in[n]}(-\log M_{tij}! + M_{tij}\log\theta_{ij})$ となる．これにより問題 (6) は t ごとに独立な問題に分割できることが分かるので，最適化問題

$$\begin{aligned}\min_{\boldsymbol{M}_t}.\quad & \sum_{i\in[n]}\sum_{j\in[n]}(\log M_{tij}! - M_{tij}\log\theta_{ij})\\ \text{s.t.}\quad & N_{t,i} = \sum_{j\in[n]} M_{tij}\quad (i\in[n]),\\ & N_{t+1,i} = \sum_{j\in[n]} M_{tji}\quad (i\in[n]),\\ & M_{tij}\in\mathbb{Z}_{\geqq 0}\quad (i\in[n], j\in[n])\end{aligned}\tag{7}$$

を t ごとに解けばよい．ここで，この $\boldsymbol{M}$ を連続緩和しスターリングの近似 $\log x! \approx x\log x - x$ を適用すると，目的関数は $\sum_{i\in[n]}\sum_{j\in[n]}[M_{tij}\log M_{tij} - M_{tij}\log\theta_{ij}] - \sum_{i\in[n]}\sum_{j\in[n]} M_{tij}$ と近似できる．さらに，制約 $N_{t,i} = \sum_{j\in[n]} M_{tij}$ $(i\in[n])$ より $\sum_{i\in[n]}\sum_{j\in[n]} M_{tij} = \sum_{i\in[n]} N_{t,i}$ であることから第 2 項は $\boldsymbol{M}_t$ に依存しないため，第 1 項のみ考えればよいことがわかる．以上により，MAP 推定問題は近似的に

$$\begin{aligned}\min_{\boldsymbol{M}_t}.\quad & \sum_{i\in[n]}\sum_{j\in[n]}(M_{tij}\log M_{tij} - M_{tij}\log\theta_{ij})\\ \text{s.t.}\quad & N_{t,i} = \sum_{j\in[n]} M_{tij}\quad (i\in[n]),\\ & N_{t+1,i} = \sum_{j\in[n]} M_{tji}\quad (i\in[n]),\\ & M_{tij}\geq 0\quad (i\in[n], j\in[n])\end{aligned}\tag{8}$$

という問題に変形できる．

4.3 2 つの問題の等価性

ここでは，最適化問題 (8) が Sinkhorn 距離を求める問題 (5) と等価であることを示す．全体の総人口 $F := \sum_{i\in[n]}\sum_{j\in[n]} M_{tij}$ を用いて

$$\nu_{t,i} := \frac{N_{t,i}}{F},\quad \nu_{t+1,i} := \frac{N_{t+1,i}}{F},\quad \mu_{tij} := \frac{M_{tij}}{F}\tag{9}$$

とおくと，最適化問題 (8) は

$$\begin{aligned}\min_{\boldsymbol{\mu}_t}.\quad & \sum_{i\in[n]}\sum_{j\in[n]}[(-\log\theta_{ij})\cdot\mu_{tij} + \mu_{tij}\log\mu_{tij}]\\ \text{s.t.}\quad & \nu_{t,i} = \sum_{j\in[n]}\mu_{tij}\quad (i\in[n]),\\ & \nu_{t+1,i} = \sum_{j\in[n]}\mu_{tji}\quad (i\in[n]),\\ & \mu_{tij}\geq 0\quad (i\in[n], j\in[n]).\end{aligned}\tag{10}$$

という形に変形することができる．ただし，$\boldsymbol{\mu}_t$ に依存しない定数部分に関しては省略している．このとき，$\boldsymbol{\nu}_t$ および $\boldsymbol{\nu}_{t+1}$ は定義より確率ベクトルになっている．

ここで，Sinkhorn 距離を求めるための最適化問題 (5) において $\lambda = 1$ としたものと最適化問題 (10) とを比較すると，

$$\boldsymbol{p}\leftrightarrow\boldsymbol{\nu}_t,\ \boldsymbol{q}\leftrightarrow\boldsymbol{\nu}_{t+1},\ c_{ij}\leftrightarrow -\log\theta_{ij},\ \boldsymbol{m}\leftrightarrow\boldsymbol{\mu}_t\tag{11}$$

という対応関係のもと，2 つの最適化問題は全く同じ形になっていることがわかる．このことから，Sinkhorn 距離を求めるための Sinkhorn-Knopp アルゴリズムを，近似的な MAP 推定問題 (8) を解くために利用できることがわかる．

Algorithm 2 近似的な MAP 推定問題 (8) のための Sinkhorn-Knopp アルゴリズム

Input: 確率された人口 $\boldsymbol{N}_t, \boldsymbol{N}_{t+1}$, 移動確率行列 $\boldsymbol{\theta}$
Output: 最適化問題 (8) の最適解 $\boldsymbol{M}_t^*$
1: ベクトル $\boldsymbol{u}\in\mathbb{R}^d$ を $[1, 1, \ldots, 1]$ で初期化
2: **while** not convergence **do**
3: $\quad\boldsymbol{v}\leftarrow\boldsymbol{N}_t./\left(\boldsymbol{\theta}^\top\boldsymbol{u}\right)$（ただし ./ は要素ごとの除算を意味する）
4: $\quad\boldsymbol{u}\leftarrow\boldsymbol{N}_{t+1}./\left(\boldsymbol{\theta}\boldsymbol{v}\right)$
5: **end while**
6: $U\leftarrow\mathrm{diag}(\boldsymbol{u}), V\leftarrow\mathrm{diag}(\boldsymbol{v})$
7: $\boldsymbol{M}^*\leftarrow U\boldsymbol{\theta}V$

4.4 アルゴリズム

MAP 推定問題のための最適化問題 (7) に対して Sinkhorn-Knopp アルゴリズムを適用した場合の計算手順を Algorithm 2 に示す．計算のボトルネックとなるのは while 文の中 (2-5 行目) であり，$\boldsymbol{u}, \boldsymbol{v}$ が収束するまで行列とベクトルの乗算及びベクトル同士の要素ごとの除算を繰り返すことになる．

4.5 実　　装

Algorithm 2 の主要部は，収束するまで 3 行目と 4 行目の操作を繰り返す部分である．この操作は以下のような 2 通りの実装方法が考えられ，それぞれに長所と短所がある．

4.5.1 行列を用いた実装

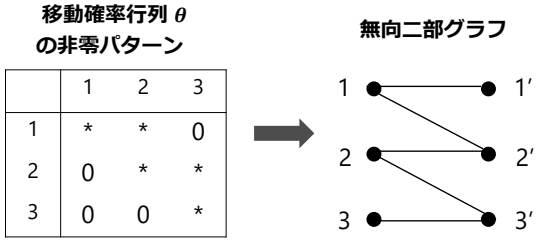
Algorithm 2 の 3 行目と 4 行目は行列演算によって素直に実装することができる．1 反復に必要な時間計算量は $O(n^2)$ であり，空間計算量も $O(n^2)$ である．この実装のメリットとして，単純な行列演算のみによって実現できるため実装が容易であり，また GPGPU を用いた高速化とも非常に相性が良い点が挙げられる．

4.5.2 グラフを用いた実装

Algorithm 2 の 3 行目と 4 行目における $\boldsymbol{\theta}^\top\boldsymbol{u}$ および $\boldsymbol{\theta}\boldsymbol{v}$ の計算について，$\boldsymbol{\theta}$ が 0 の部分に関しては計算結果に影響を与えないため計算する必要がない．このことに注目すると，$\boldsymbol{\theta}$ が疎行列であること，そして非零要素の出現位置が分かっている場合には，その非零パターンに着目した実装を行うことで効率的な計算を行うことができる．

このような状況は，現実の問題において頻繁に起こりうる．例えば広域の地理空間を扱っている場合や 1 つ 1 つのエリアのサイズが小さい場合，観測時刻の間隔が短い場合はこのような状況が実現しやすい．人間の移動の速度には限界があるため，一定の距離以上離れたエリア間での移動は発生しない（もしくは発生したとしても非常に少ない）と仮定できるためである．また，交通ネットワークや展示会場などにおいても，センサの配置と導線の関係によっては移動が発生し得ないエリアの組が存在しうる．これらの場合，$\boldsymbol{\theta}$ が疎行列になり，さらに非零要素の出現位置も前もって知ることができる．

前もって分かっている移動確率行列 $\boldsymbol{\theta}$ の非零パターンを，$n\times n$ の行列 S で表すことにする．すなわち S の (i, j) 成分は，θ_{ij} が 0 であることが分かっている場合は 0，それ以外の場



	1	2	3
1	*	*	0
2	0	*	*
3	0	0	*

図 2　移動確率行列 $\boldsymbol{\theta}$ の非零パターンから無向二部グラフ $G=(V,E)$ を作成する例.

合は 1 をとる．無向二部グラフ $G=(V,E)$ を以下のように作る．まず，頂点集合を $V=\{1,2,\ldots,n\}\cup\{1',2',\ldots,n'\}$ とする．さらに，辺 E を $(i,j')\in E \Leftrightarrow S_{ij}=1$ で定める．グラフの作成例を図 2 に示す．G は $\boldsymbol{\theta}$ の非零成分の位置を表現しているグラフと考えることができる．このグラフを用いると，$\boldsymbol{\theta}^\top\boldsymbol{u}$ および $\boldsymbol{\theta v}$ はそれぞれ

$$(\boldsymbol{\theta}^\top\boldsymbol{u})_i=\sum_{j\in\mathcal{N}(i')}u_j \tag{12}$$

$$(\boldsymbol{\theta v})_i=\sum_{j'\in\mathcal{N}(i)}v_j \tag{13}$$

で計算できる．ただし，$\mathcal{N}(i)$ はグラフ G において頂点 i に隣接する頂点集合を表す.

この方法で計算を行った場合，$\boldsymbol{\theta}$ の非零成分の数（G の辺の数に等しい）を m として，1 反復に必要な時間計算量は $O(m)$ であり，入出力を隣接リストなどの形式で行うことで空間計算量も $O(m)$ にすることができる．例えばあるエリアから移動する可能性のあるエリアの数が k で抑えられる場合，$\boldsymbol{\theta}$ の非零要素の数 m は $O(kn)$ 程度になるため，k が小さい場合には高速かつ省メモリに推定を行うことが可能になる．ただし，$\boldsymbol{\theta}$ が密行列である場合には行列を用いた実装と実質的に同じことを計算することになり，GPGPU との相性も良くない.

5 実　　験

提案手法の有効性を確認するため，人工データを用いた実験を行った.

5.1 実験設定

5.1.1 データ

人工データとして，サイズ $L\times L$ のグリッド空間における人々の移動を想定した移動データを生成した ($L=10$)．グリッド空間における一つのセルが一つのエリアに対応する．真の移動確率 $\boldsymbol{\theta}^{\text{true}}$ を $\theta_{ij}\propto s_i^{\text{true}}\cdot\exp(-\beta^{\text{true}}\cdot\text{dist}(i,j))$ に従って設定した．ただし，$s_i^{\text{true}}>0$ ($i\in[n]$) はエリア i への人の集まりやすさを表すパラメータであり，β^{true} はエリア間の距離による移動確率の減衰を表すパラメータである．また，$\text{dist}(i,j)$ はエリア i とエリア j の距離を表し，$\text{dist}(i,j)=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}$ で計算される．ただし，(x_i,y_i) は左上隅のエリアを基準としたエリア i の位置を表す（左上隅のエリアが $(1,1)$，左下隅のエリアが $(L,1)$，右下隅のエリアが (L,L)）．この移動確率行列は [6] において利用されたものの亜種になっている．実験では全てのエリアの中からランダムに 6 エリアを選択し，選択されたエリアについては $s_i^{\text{true}}=10$，それ以外のエリアについては $s_i^{\text{true}}=1$ とした．また，$\beta^{\text{true}}=1.0$ と設定した．そして，$\boldsymbol{\theta}^{\text{true}}$ に基づいて，3.1 で説明した過程に従って時刻 $T=10$ まで人の移動をシミュレートし，各時刻，各エリアの人口 $\boldsymbol{N}$ 及び各エリア間の移動人数 $\boldsymbol{M}^{\text{true}}$ を生成した．各エリアの初期人口を N_{init} とし，$N_{\text{init}}=10^3,10^4,10^5$ とした.

5.1.2 手　　法

我々の目的は $\boldsymbol{N}$ のみから EM アルゴリズムを用いて $\boldsymbol{M}^{\text{true}}$ を推定することである．E ステップに提案手法及び比較手法を適用し，EM アルゴリズム全体の性能を比較した.

提案手法は 4.5 章における「行列を用いた実装」によって実装し，GPU は使用せず CPU を用いて実験を行った．また，比較手法として，2 つの手法を用いた.

- MAP 推定問題を最小凸費用流問題と呼ばれる組合せ最適化問題に帰着して解く手法 (Flow). [11] において提案された手法であり，近似を用いることのない厳密かつ高速な推定が可能であることが報告されている.
- 最適化問題 (8) の制約をペナルティとして目的関数に加算し，矩形制約下の凸最適化問題として L-BFGS-B 法 [17] で解く手法 (L-BFGS-B). [5] [6] などで用いられている．この手法ではペナルティ項の影響の強さをコントロールするハイパーパラメータを決める必要があるが，本実験では 1.0 とした.

提案手法は python の numpy ライブラリを利用し，L-BFGS-B は python の scipy に実装されているものを利用した．また，Flow に関しては python で実装すると実行速度が非常に遅くなりフェアな比較にならないことが想定されたため， C++ (gcc 4.8.5, -O3 option) で実装を行った．EM アルゴリズム全体の詳細，特に M ステップの処理に関しては [6] を参照されたい.

5.1.3 評価指標

推定の正しさの評価指標として，Normalized Absolute Error (NAE) を用いた．$\boldsymbol{M}^{\text{true}}$ を真の移動人数，$\boldsymbol{M}^{\text{estimated}}$ を推定された移動人数として，NAE は

$$\frac{\sum_{t,i,j}|M_{tij}^{\text{true}}-M_{tij}^{\text{estimated}}|}{\sum_{t,i,j}M_{tij}^{\text{true}}} \tag{14}$$

によって計算される量で，$\boldsymbol{M}^{\text{true}}$ と $\boldsymbol{M}^{\text{estimated}}$ の近さを表す．完全に一致していれば 0 をとり，最大で 2 までの値をとる.

5.2 実験結果

5.2.1 経過時間と NAE の関係についての実験結果

図 3 は，$\beta^{\text{init}}=0.1,2.0$, $N^{\text{init}}=10^3,10^4,10^5$ の場合に，経過時間と達成された NAE の関係を表すグラフである．EM アルゴリズムの各反復（200 回目まで）の終了時点における，経過時間と NAE がプロットされている.

まず，計算時間の意味で提案手法が優れていることがわかる．いずれの場合においても，提案手法は非常に高速に NAE が小さくなり，収束している．例えば，$\beta^{\text{init}}=0.1, N^{\text{init}}=10^4$ の場合，NAE の値 0.2 を初めて達成するために必要な計算時間

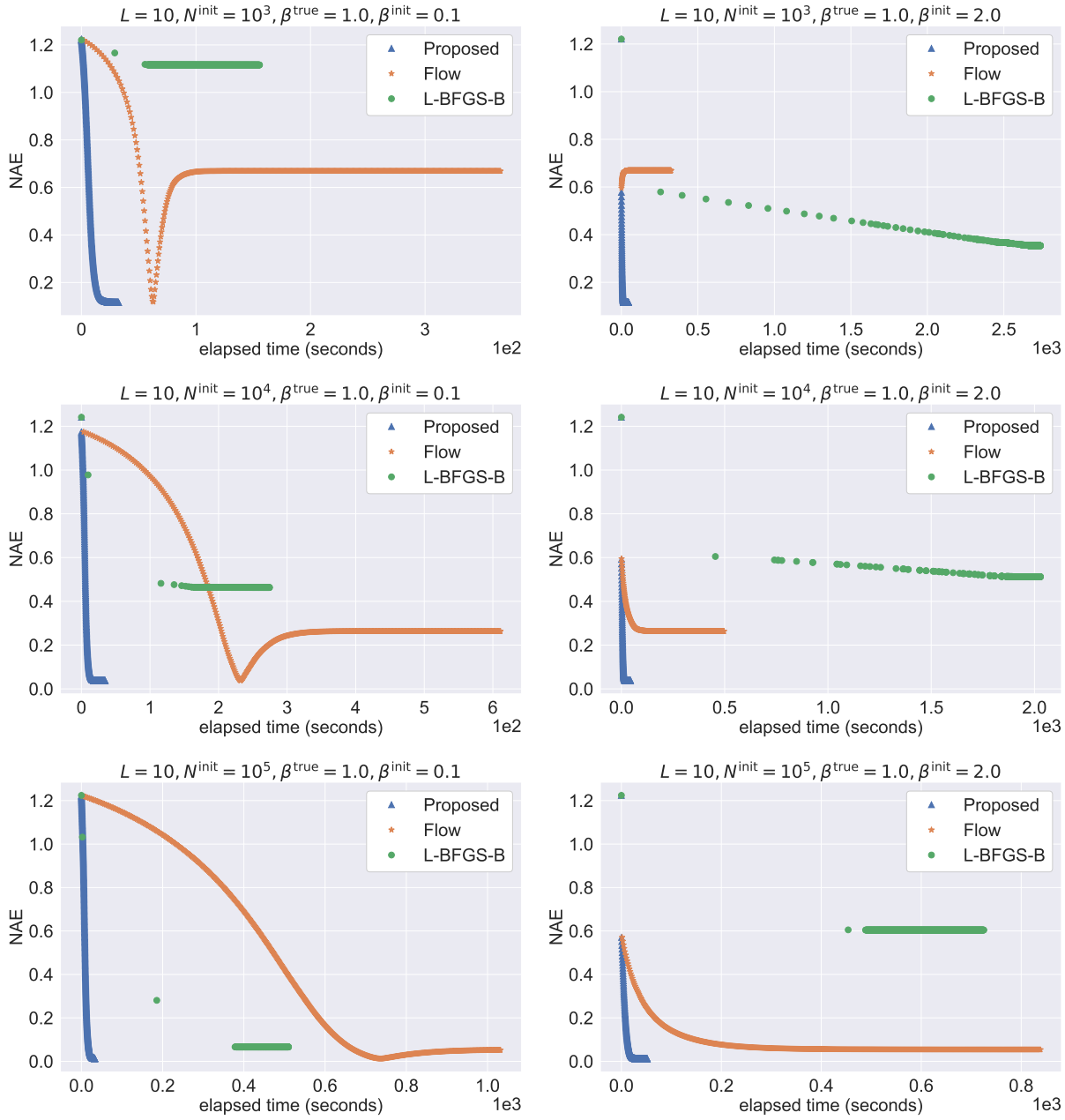


図 3　それぞれの手法を用いた EM アルゴリズムにおける，NAE (Normalized Absolute Error) と経過時間の関係．Proposed が提案手法を，Flow が最小凸費用流問題への帰着を用いた手法 [11] を，L-BFGS-B が L-BFGS-B 法を用いた手法 [5] [6] を表す．

は，既存手法 (Flow) では 210 秒程度であるのに対し，提案手法では 8.27 秒となっており，約 25 倍の高速化が達成されていることが分かる．この理由としては，1 反復において Flow では最小費用流問題を，L-BFGS-B では準ニュートン法で凸計画問題を解く必要があるのに対し，提案手法は単純な行列演算の繰り返しを行うだけで良いことが理由であると考えられる．

また，特に N^{init} が小さい場合に，手法 Flow は一度誤差が非常に小さくなった後，誤差が大きくなってしまうという現象が起こっている．これは，推定アルゴリズムが出力する解が目的関数値の悪い局所解に収束してしまっていることを表していると考えられる．それに対し，提案手法は一度誤差が小さくなった後はそこで収束しており，良い解に収束していることが見て取れる．Flow は MAP 推定を近似を用いず厳密に行うことのできる手法である．この実験結果は厳密な MAP 推定が EM アルゴリズム全体の性能という意味では必ずしも良いとは限らないことを示していると考えられる．特に，推定結果が整数値しか取らないという Flow の性質が，サンプル数が少ない場合（すなわち N^{init} が小さい場合）の M ステップにおける β の推定に悪影響を与えている可能性がある（逆に，提案手法や L-BFGS-B は連続緩和を行っているため，サンプル数が少ない場合でも $\boldsymbol{M}$ の分布をうまく表現し β をうまく推定できている可能性がある）．この部分についてはより詳細な検証が必要である．また，このような推定における誤差の挙動の違いが，今回のように $\boldsymbol{\theta}$ を β と $\boldsymbol{s}$ によってパラメタライズした場合に特

表 1　500 回 EM アルゴリズムの反復を行った後の NAE の値．各設定において 10 回推定を行った際の NAE の平均値及び標準偏差（括弧内）を示している．各試行において β^{init} を [0.0, 2.0] から一様分布に従ってサンプリングし推定を行った（β^{init} の値は各設定間で共通）．

N_{init}	10^3	10^4	10^5	10^6
Proposed	**0.117 (0.00)**	**0.038 (0.00)**	**0.013 (0.00)**	**0.003 (0.00)**
Flow	0.686 (0.00)	0.257 (0.00)	0.058 (0.00)	0.006 (0.00)
L-BFGS-B	0.305 (0.686)	0.339 (0.164)	0.293 (0.243)	0.426 (0.123)

有の現象なのか，それともパラメタライズの方法に依存しない現象なのかについても検証が必要である．

5.2.2　最終的な精度についての実験結果

表 1 は，500 回の EM アルゴリズムの反復を行った後の NAE の値を示している．区間 [0.0, 2.0] から一様分布に従って 10 個の値をサンプリングして β^{init} として利用した（β^{init} の値は各設定間で共通）．表示されている値は 10 個の β^{init} に関する NAE の平均値であり，括弧内には標準偏差を示している．まず，提案手法がいずれの設定においても NAE の値が最も小さくなっていることが分かる．Flow は N^{init} が小さい場合に NAE が大きく，L-BFGS-B はいずれの場合もあまり NAE が小さくない．また，初期値による最終的な NAE のばらつきは，提案手法と Flow では非常に小さいのに対し L-BFGS-B では大きく，結果が初期値に大きく依存していることが分かる．特に β^{init} と β^{true} が大きく離れている際に L-BFGS-B は精度が低下する傾向があった．これらの結果から，提案手法は安定して高精度な推定が行えることが確認された．

6　関 連 研 究

CFDM を含む，集計化されたデータを活用するためのフレームワークである Collective Graphical Model における推定を効率的に行うための手法は様々なものが提案されている [10] [18] が，CFDM に特化したものではないため非効率的であったり適用できない場合がある．CFDM に特化した手法としては，[6] において効率的な最適化手法が提案されているが，人間の移動確率モデル $\boldsymbol{\theta}$ が特殊な形に分解可能である場合にしか適用することができず，汎用的ではない．また，[11] では，MAP 推定問題を最小凸費用流問題と呼ばれる組合せ最適化に帰着することで推定を行う手法が提案されている．この手法は近似を用いない厳密な推定が可能であるという点で優れているが，計算時間や実装の簡便さという点では本論文の提案手法の方が優れているといえる．

CFDM を用いた時刻別エリア人口データからの移動の推定については様々な研究がなされている．例えば，[5] [19] では，変分ベイズやニューラルネットワークを用いて都市空間における人流の推定が行われている．また，[4] や [8] では，人口ではなく各エリアの入人数と出人数が入力として与えられる状況における人流の推定を扱っている．このように，観測されるデータや人間の移動を表現する確率モデルには様々なバリエーションがあるが，本論文の提案手法はいずれの手法のサブルーチンとしても利用することができ，推定の効率化を行うことができる．

また，CFDM とは異なる方法を用いてカウントデータから人々の移動を推定する手法も注目を集めている．特に本論文に関係する研究として，[20] では集計化されたデータを公開する際のセキュリティリスクを評価するために，集計されたカウントデータから個人の移動軌跡を復元するアルゴリズムを提案している．また，[21] では鳥の移動パターンを分析する目的で，部分的な観測からマルコフ連鎖のサンプルパスを再現する手法が提案されている．これらの手法と本研究の手法の違いは，(i) これらの手法の目的は個別の移動軌跡を復元することにあり，集団的な移動を傾向を再現することに無い点 (ii) これらの手法では移動モデルのパラメータを手動で決める必要がある点である．

また，関連する別の方向性の研究として，都市における人口や人流の予測する手法も活発に研究されている [22] [23] [24]．これらの手法は，それぞれのエリアにおける都市の未来のダイナミクスを，過去のデータや他の特徴量を用いて古典的な回帰モデルや深層学習モデルなどを用いて教師あり学習によって予測するものである．これに対し，我々の研究の目的は人口のスナップショットのみから人の移動の様子を教師なし学習の枠組みで推定するものであり，未来予測とは全く異なるタスクに取り組んでいる．

7　ま　と　め

本論文では，時刻別エリア人口データからエリア間の移動人数を推定するための新しい手法を提案した．提案手法は Collective Flow Diffusion Model (CFDM) と呼ばれる確率モデルに基づいており，これまでの計算のボトルネックであった E ステップの計算を，Sinkhorn-Knopp アルゴリズムによって実行することで大幅な高速化を可能にした．シミュレーションデータを用いた実験によって，提案手法が既存手法と比較して高速かつ高精度にエリア間の移動人数を推定できることを確認した．

文　献

[1] Masayuki Terada, Tomohiro Nagata, and Motonari Kobayashi. Population estimation technology for mobile spatial statistics. *NTT DOCOMO Technical Journal*, Vol. 14, No. 3, pp. 10–15, 2013.

[2] Hai Yang and Jing Zhou. Optimal traffic counting locations for origin–destination matrix estimation. *Transportation Research Part B: Methodological*, Vol. 32, No. 2, pp. 109–126, 1998.

[3] Tetsuro Morimura, Takayuki Osogami, and Tsuyoshi Idé. Solving inverse problem of markov chain with partial observations. In *NIPS*, pp. 1655–1663, 2013.

[4] Akshat Kumar, Daniel Sheldon, and Biplav Srivastava. Collective diffusion over networks: Models and inference. In *UAI*, 2013.

[5] Tomoharu Iwata, Hitoshi Shimizu, Futoshi Naya, and Naonori Ueda. Estimating people flow from spatiotemporal population data via collective graphical mixture models. *ACM Transactions on Spatial Algorithms and Systems*, Vol. 3, No. 1, pp. 1–18, 2017.

[6] Yasunori Akagi, Takuya Nishimura, Takeshi Kurashima, and Hiroyuki Toda. A fast and accurate method for estimating people flow from spatiotemporal population data. In *IJCAI*, pp. 3293–3300, 2018.

[7] Jiali Du, Akshat Kumar, and Pradeep Varakantham. On understanding diffusion dynamics of patrons at a theme park. In *AAMAS*, pp. 1501–1502, 2014.

[8] Yusuke Tanaka, Tomoharu Iwata, Takeshi Kurashima, Hiroyuki Toda, and Naonori Ueda. Estimating latent people flow without tracking individuals. In *IJCAI*, pp. 3556–3563, 2018.

[9] Daniel R. Sheldon and Thomas G. Dietterich. Collective graphical models. In *NIPS*, pp. 1161–1169, 2011.

[10] Daniel Sheldon, Tao Sun, Akshat Kumar, and Tom Dietterich. Approximate inference in collective graphical models. In *ICML*, pp. 1004–1012, 2013.

[11] Yasunori Akagi, Takuya Nishimura, Tanaka Yusuke, Takeshi Kurashima, and Hiroyuki Toda. Exact and efficient inference for collective flow diffusion model via minimum convex cost flow algorithm. In *AAAI*, 2020.

[12] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NIPS*, pp. 2292–2300, 2013.

[13] Cédric Villani. *Optimal Transport: Old and New*, Vol. 338. Springer Science & Business Media, 2008.

[14] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 9, pp. 1853–1865, 2017.

[15] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pp. 214–223, 2017.

[16] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. Learning with a Wasserstein loss. In *NIPS*, pp. 2053–2061, 2015.

[17] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, Vol. 16, No. 5, pp. 1190–1208, 1995.

[18] Tao Sun, Daniel Sheldon, and Akshat Kumar. Message passing for collective graphical models. In *ICML*, pp. 853–861, 2015.

[19] Tomoharu Iwata and Hitoshi Shimizu. Neural collective graphical models for estimating spatio-temporal population flow from aggregated data. In *AAAI*, pp. 3935–3942, 2019.

[20] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. Trajectory recovery from ash: User privacy is not preserved in aggregated mobility data. In *WWW*, pp. 1241–1250, 2017.

[21] Daniel Sheldon, MaS Elmohamed, and Dexte Kozen. Collective inference on markov models for modeling bird migration. In *NIPS*, pp. 1321–1328, 2008.

[22] Tatsuya Konishi, Mikiya Maruyama, Kota Tsubouchi, and Masamichi Shimosaka. Cityprophet: City-scale irregularity prediction using transit app logs. In *Ubicomp*, pp. 752–757. ACM, 2016.

[23] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. Flow prediction in spatio-temporal networks based on multitask deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[24] Renhe Jiang, Xuan Song, Dou Huang, Xiaoya Song, Tianqi Xia, Zekun Cai, Zhaonan Wang, Kyoung-Sook Kim, and Ryosuke Shibasaki. Deepurbanevent: A system for predicting citywide crowd dynamics at big events. In *KDD*, pp. 2114–2122. ACM, 2019.