Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

BAYESIAN OPTIMIZATION FOR CLINICAL PATHWAY DECOMPOSITION FROM AGGREGATE DATA

William Plumb[1], Alex Bottle[2], and Giuliano Casale[1]

[1]Dept. of Computing, Imperial College London, UK
[2]School of Public Health, Imperial College London, UK

ABSTRACT

Data protection rules often impose anonymization requirements on datasets by means of aggregations that hinder the exact simulation of individual subjects. For example, clinical pathways that disclose medical conditions of patients may typically need to be aggregated to preserve anonymity of the subjects. However, aggregation unavoidably results in biasing the simulation process, for example, by introducing spurious pathways that can skew the simulated trajectories. In this paper, we study this problem and develop approximate decomposition methods that mitigate its impact. Our method is shown to produce from the raw aggregates pathways with higher fidelity than sampling a Markov chain model of the aggregate data, even preserving the same length of the original pathways. In particular, we observe a relative increase in average cosine similarity of up to 52% with respect to the true pathways compared with aggregate Markov chain sampling.

1 INTRODUCTION

Data protection and privacy are vital attributes in healthcare information, regulated by laws such as GDPR in Europe (Olatunji et al. 2022). Among the essential anonymization techniques is data aggregation, which severely limits the identifiability of clinical information associated with individuals. In the context of clinical pathways, which are descriptors of the sequence of medical events associated with a clinical episode, aggregation yields the side effect of introducing ambiguities concerning the paths followed by patients within the healthcare system. This can therefore suggest that certain combinations of events are possible in real-life, when in fact they do not occur in the disaggregated data traces. Such instances, which we refer to as spurious pathways, consequently diminish the value of simulation as a predictive tool, since the system dynamics is biased by the presence of fictitious sequences that may be far statistically from the actual observations. Therefore, a scientific challenge arises on how to improve the quality of simulations based on aggregate data, reducing the incidence of spurious pathways in determining the simulation outcomes. Solving this challenge allows automating discovery of clinical pathway classes for future simulation models, without the need for time-consuming manual data processing.

In this paper, we study this problem, focusing on aggregate pathways data presented in the form of counting matrices. Our analysis reveals that the knowledge of the original pathways length and the presence of second-order statistical information concerning the aggregate, in terms of standard deviation of the counts, can lead to a substantial reduction of the biases produced by spurious pathways, resulting in higher fidelity simulation of clinical pathways. In particular, we propose a Bayesian optimization approach, called IBPD (Iterated bi-level pathway decomposition), which significantly improves the similarity of pathways generated from aggregate data compared with simple baseline methods relying on a simulation of the Markov chain associated to the aggregate counting data. We observe a relative increase in average cosine similarity of up to 54% with respect to the true pathways compared with the baseline Markov chain sampling approach.

Core to the IBPD technique is the generation of linear feasibility constraints for the simulated pathways, as well as an iterative method to tighten the feasible region based on tighter upper bounds on the decision variables. We show through ablation analysis that these mechanisms are some of the main drives of the overall accuracy of the method.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 gives the problem statement, followed in Section 4 by the definition of the proposed decomposition methods. Experiments are given in 5, followed by conclusions in Section 6.

2 RELATED WORK

Anonymization techniques are widely used to preserve privacy on individual data by removing Personally Identifiable Information (PPI), using a wide variety of techniques (Murthy et al. 2019). While keeping individual data, even anonymized, can still be potentially identified by a small number set of quasi-identifiers (QI) features such as gender, zip code and date of birth (Sweeney 2002a; Sweeney 2002b). This can follow on to linkage attacks (Fung et al. 2010), also known as a deanonymization attack. For graph-based information, anonymization techniques can be used (Majeed and Lee 2020). These methods include, adding or modifying edges, shifting, shuffling, and clustering (Casas-Roma et al. 2019; Kumar and Kumar 2017; Mohapatra and Patra 2019). In healthcare, this has been used with electronic health records, treating them as graph structures using manipulation or clustering-based anonymization (Olatunji et al. 2022).

These anonymization techniques, such as adding edges, would hinder the exact simulation of a system with the introduction of spurious or simplified graphs. Previous studies applying simulation have shown how using aggregated data affects performance and insights (Patterson et al. 2010; Hoffmann et al. 2016).

Synthetic data generation is a way to bypass some issues arising from privacy concerns (Figueira and Vaz 2022). If the raw data is accessible, it is common to train Generative Adversarial Networks (GANs) to generate a new synthetic graph structure (Hernandez et al. 2022). However, the limitations and potential bias using GANs generated data may misrepresent the raw dataset through issues such as oscillating loss, mode collapse, uninformative loss, vanishing gradients, and hyperparameter-tuning (Figueira and Vaz 2022).

To preserve as much structural information from graphs, we use an alternative method combining multiple graphs for later decomposition and reconstruction. As part of the process of graph compression, finding similar graph structures can reduce the complexity from large networks (Fan et al. 2012). These aggregated structures can hold privacy safe information, but the need for decomposition is still required. The topic of decomposition in the literature can be equivalent to the term graph disentanglement. Decomposition methods include creating a Directed Acyclic Graph (DAG) of all possible subgraphs and evaluating their isomorphism (Williams et al. 2006), using k-edge connected components to decompose large graphs (Chang et al. 2013), and compact matrix decomposition (Sun et al. 2007). Whereas, graph disentanglement uses deep learning methods such as graph neural networks to find granular relationships in large graph structures (Wang et al. 2020; Hu et al. 2020). In large graph decomposition research, the common validation approach revolves around the speed of the process. Additionally to the speed of our methods, we validate based on the similarity of the produced decompositions compared with the true graph.

3 CLINICAL PATHWAY DECOMPOSITION

Patient activities in healthcare systems give rise to sequences of medical events, such as General Practitioner (GP) visits, diagnostic tests, specialist appointments, surgeries, etc. A *clinical pathway* r that describes passage through the healthcare system by a given patient is thus an ordered sequence of nodes $(n_{r,1},\ldots,n_{r,s},\ldots,n_{r,S})$, where $n_{r,s}$ is the medical event incurred at step s of the pathway and S is the length of the pathway. The set of all possible medical events is denoted with $N=\{n_1,\ldots,n_{|N|}\}$, so that $n_{r,s}\in N$. Throughout, we assume that there exists a common *exit event* e to all pathways, i.e., $e\equiv e(r)$, $\forall r$, which may be either a real or a virtual event, the latter when this is artificially appended at the end of a pathway.

A pathway r may be readily summarized by a *counting matrix* $C_r = [C_r(i,j)]$, where $C_r(i,j)$ counts the number of transitions from node i to j. For a given pathway, we denote with $a_r = [a_r(j)]$ as its initial vector, i.e., a vector of all zeros but a 1 in the position associated with the first event of the pathway.

Note that counting matrices are a fairly flexible way of describing a pathway. On the one hand, since repeated medical events may be modeled as separate nodes (e.g., 1st GP visit, 2nd GP visit, ...), the set N may be defined to preserve in the counting matrix exactly all the sequence information available in the original pathway r. In such case, the counting matrix is a binary matrix. However, if repeated visits to the same medical event are merged as visits to the same node, the counting matrix allows compression of the representation at the expense of some loss of information on the precise sequence of the events, while exactly preserving the transition counts. In such case, the counting matrix is integer valued.

Throughout, we focus on datasets represented as counting matrices C_r and initial vectors a_r. Note that from C_r, it is straightforward to define a discrete-time Markov chain (DTMC) by normalizing rows to be transition probabilities. DTMCs of this kind are in widespread practice in clinical pathway simulation (Standfield et al. 2014).

3.1 Problem Statement

We assume that the dataset available for simulation features only aggregate data obtained by merging multiple pathways, typically having some similar characteristics, e.g., based on demographics or co-morbidities. More precisely, consider a group formed by g pathways having counting matrices $C_1, C_2, \ldots, C_g$. Henceforth, we assume that due to anonymization or data access costs, it is not possible to access directly the g counting matrices, and instead we have access to an *aggregate counting matrix* $A = [A(i,j)]$, with

$$A(i,j) = \sum_{r=1}^{g} C_r(i,j), \qquad i,j = 1,\ldots,|N| \tag{1}$$

Similarly, the initial vectors are summarized into an *aggregate initial vector* $a = [a(j)]$, with $a(j) = \sum_{r=1}^{g} a_r(j)$, $j = 1,\ldots,|N|$. Note that, according to these definitions, we also know the number of pathways $g = |a|$.

In addition to the (a,A) pair, we also assume we know values $l(k)$, k_{max}, counting the number of pathways of length $l(k)$, where l_{max} is an upper bound on the length of all the clinical pathways in the dataset. Lastly, we assume we have the bias-corrected standard deviation of the transition counts, i.e., the matrix $S = [S(i,j)]$, with

$$S(i,j) = \left(\frac{1}{g-1} \sum_{r=1}^{g} (C_r(i,j) - A(i,j)/g)^2 \right)^{1/2}, \qquad i,j = 1,\ldots,|N| \tag{2}$$

where $A(i,j)/g$ is the mean number of transitions from i to j for a patient in the group.

Our problem statement is as follows: we seek to obtain the decomposed counting matrices $C_1, C_2, \ldots, C_g$ from the knowledge of the (a,A) pair, l, and S. Let $\widehat{C}_1, \widehat{C}_2, \ldots, \widehat{C}_g$, be decomposed counting matrices. For a given algorithm, we assess the performance of the decomposition in terms of cosine similarity. For two matrices A and B, the cosine similarity is defined as

$$\theta(A,B) = \frac{\sum_{i,j} A(i,j) \cdot B(i,j)}{\sqrt{\sum_{i,j} A(i,j)^2} \sqrt{\sum_{i,j} B(i,j)^2}} \tag{3}$$

The cosine similarity is known to lie in $[-1,1]$, with $\theta(A,B) = 1$ indicating that the vectors $vec(A)$ and $vec(B)$ obtained by stacking the columns of their argument are proportional vectors, $\theta(A,B) = -1$ indicating opposite vectors, and $\theta(A,B) = 0$ orthogonal vectors. Thus, higher values of the cosine similarity indicate higher similarity.

The cosine similarity metric is widely used to compare sparse matrices in high dimensions, which is indeed typical of the counting matrices for clinical pathways. For instance, in the real-world dataset we use for validation, where data is aggregated based on demographic and other information, the counting matrix order is $|N| = 29$, but only approximately $49 \approx 2|N|$ non-zeros are used on average across the counting matrices.

Given a set of decomposed matrices $(\widehat{C}_1, \widehat{C}_2, \ldots, \widehat{C}_g)$, we define the summary value of their cosine similarity from the true matrices $(C_1, \ldots, C_g)$ as follows. First, we compute the cosine similarity matrix $\Theta = [\theta_{i,j}]$, where $\theta_{i,j} = \theta(\widehat{C}_i, C_j)$ for all $i, j = 1, \ldots, g$. Then, we solve the integer optimization program

$$\begin{aligned} \Theta = \max_{y} \; & \frac{1}{g} \sum_{i=1}^{g} \sum_{j=1}^{g} \theta_{i,j} y_{i,j} \\ s.t. \quad & \sum_{i=1}^{g} y_{i,j} = 1 \\ & \sum_{j=1}^{g} y_{i,j} = 1 \\ & y_{i,j} \in \{0,1\} \qquad i, j = 1, \ldots, g \end{aligned} \tag{4}$$

This program seeks through the integer variables $y = (y_{i,j})$ for the best possible one-to-one matching of a matrix $\widehat{C}_i$ to a counting matrix C_j, to maximize the average cosine similarity Θ across the resulting pairs. Note that this metric requires to know the true pathways, therefore it is used here only for performance evaluation purposes, and it is not part of the proposed decomposition technique.

3.2 Illustrative Example

As an example of the problem at hand, consider a toy model with $|N| = 3$ nodes and $g = 2$, pathways $r_1 = (1,1,1,2,3)$ and $r_2 = (2,1,3)$. The counting matrices are

$$C_1 = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad C_2 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Since the pathways have respectively length 5 and 3, the pmf of the length is $l = (0,0,1,0,1)$, with $kmax = 5$. The initial vectors are $a_1 = (1,0,0)$, $a_2 = (0,1,0)$, so that $a = (1,1,0)$.

As expected, the aggregation process introduces *spurious pathways* in the aggregate model. For example, pathways $s_1 = (2,1,1,1,3)$ and $s_2 = (2,1,1,3)$ are in principle feasible under knowledge only of (a,A). However, they do not map to either C_1 or C_2 and thus are both in reality spurious.

However, adding to the knowledge of (a,A) the pathway length vector l, we can readily recognize that s_2 is spurious, since $l(4) = 0$. Conversely, s_1 remains compatible with the available information, since $l(5) \neq 0$. Indeed, even assuming s_1 to be a legitimate pathway and removing its transition counts from A, we are left with the counting matrix

$$A' = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

that identifies the second pathway to be $s_3 = (1,2,3)$. Indeed, the pair of pathways $\{s_1, s_3\}$ is fully consistent with all the available information. This occurs since s_1 and s_3 are permutations of r_1 and r_3 that preserve the same initial event and 1-step transition counts. A similar conclusion is also reached if we include standard deviation information in the analysis. Let S be the standard deviation matrix for $\{r_1, r_2\}$ and let

S' be the corresponding matrix for $\{s_1, s_3\}$. We have

$$S' = \begin{bmatrix} \sqrt{2} & \sqrt{2}/2 & \sqrt{2}/2 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 \\ 0 & 0 & 0 \end{bmatrix} = S$$

That is, both the pathway groups $\{r_1, r_2\}$ and $\{s_1, s_3\}$ have the same standard deviation matrix and which pair is spurious cannot be distinguished based on all the available information.

Let us now repeat the analysis above, using the spurious pathways $s_4 = (1,1,3)$ and $s_5 = (2,1,1,2,3)$. Following a similar logic, we find that, while they are both compatible with (a,A) and l, the standard deviation matrix for these pathways would now be

$$S'' = \begin{bmatrix} 0 & \sqrt{2}/2 & \sqrt{2}/2 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 \\ 0 & 0 & 0 \end{bmatrix} \neq S$$

Thus, the $\{s_4, s_5\}$ pair can now be recognized as including at least a spurious pathway. In summary, the above example illustrates that, while the addition of l and S to an aggregate count dataset does not deanonymize in general the dataset, it enables us to recognize that several candidate pathways for the counting matrices C_1, ..., C_g are spurious. In the following, we investigate general methods to rely on these observations to identify best-effort estimators for C_1, ..., C_g that are as close as possible to the true counting matrix values.

4 DECOMPOSITION METHOD

4.1 Decision Variables

We propose a methodology based on optimization to obtain a decomposition $\widehat{C} = (\widehat{C}_1, \widehat{C}_2, \ldots, \widehat{C}_g)$. In our setting, several options are assessed to define the decision variables within the optimization program, which we compare in the next subsections. In particular, we argue that the first two proposed methods, transition probabilities and sequence variables, have shortcomings that lead us to prefer to work directly with counts as decision variables.

4.1.1 Transition Probabilities

In this formulation, we search over a continuous space of DTMCs $P_1, \ldots, P_g$, where $P_r = [p_r(i,j)]$, with real-valued $p_r(i,j) \geq 0$ and subject to $\sum_j p_r(i,j) = 1$, $\forall i$. The DTMCs all feature an absorbing state, corresponding to the exit event e. Such a method enables the use of continuous optimization, which is more efficient than integer programs. However, this type of decision variable faces two limitations.

First, for a given pair (i,j) such that $A(i,j) > 0$, a continuous solver has a limited incentive to set any of the $p_r(i,j)$ values to zero; thus, the P_r matrices are unlikely to display a similar sparsity pattern as the original C_r matrices. A second problem is that the modeling of the pathway lengths to match the l vector introduces complexity, since a DTMC can represent a path length using powers of its transition matrix P_r. This corresponds to polynomial terms of order l_{max}, resulting in a difficult and typically non-convex, continuous optimization program. Due to the above difficulties, in preliminary experiments this approach performed significantly worse than all the other methods discussed in this section, thus it is not undertaken further in the paper.

4.1.2 Sequence Variables

In this approach, we leverage the knowledge of the pathway lengths to define a collection of g integer vectors $x_1, \ldots, x_g$, of the same length as the entries in the vector l. For instance, in the illustrative example

of Section 3.2 this would correspond to two vectors $x_1 = (x_{1,1}, x_{1,2}, x_{1,3})$ and $x_2 = (x_{2,1}, x_{2,2}, x_{2,3}, x_{2,4}, x_{2,5})$, where $x_{i,j} \in \{1, \ldots, |N|\}$ represents the j-th event in the i-th decomposed clinical pathway. The formulation can be further reduced by recognizing that the final event of both pathways is $x_{1,3} = x_{2,5} = e$. From the sequence variables in x_r, it is straightforward from the definitions to obtain the decomposed counting matrices $\widetilde{C}_r$ and the decomposed initial vector a_r for each patient r in the group.

The sequence variables overcome the problems of the DTMC representation in that the length of the pathways is matched exactly by construction, while the pathways can easily provide counting matrices with different sparsity patterns. However, the method is inefficient if the counting matrix models repeated medical events as successive visits to the same node. For instance, if a patient loops 20 times at a medical event n_i, then this would result in 20 integer variables, whereas the counting matrix data in A and S describes these transition as a single coefficient for $A(i,i)$.

Another drawback of this representation is that the order of the medical events in the sequence maps into distinct values of the solution vectors x_i. While this may be appropriate in some problems, in general the counting matrices A and S do not fully preserve sequence information and in some instances can even be invariant with respect to sequence permutations. In such cases, the integer program would forcibly spend computational effort to compare permutations that can neither improve the objectives nor the feasibility of the constraints. Due to these shortcomings, we also do not consider this formulation again. Instead, we consider a related integer representation in the next section that overcomes the above limitations.

4.1.3 Count Variables

In what follows, we focus instead on the following formulation, which rely on decision variables for the counts that appear in the entries of the C_r matrices and a_r vectors. In this representation, we consider integer vectors $x_1 = (x_1(i,j)), \ldots, x_g = (x_g(i,j))$ of identical length, equal to h, where $H(A)$ is the set of index pairs (i,j) associated with the non-zero elements in A and $h = |H(A)|$. Each element $x_r(i,j) \in [0, A(i,j)]$, $(i,j) \in H(A)$, describes the number of times the transition (i,j) occurs in the decomposed counting matrix. Thus, the decomposed counting matrices are given by $\widetilde{C}_r = [\widetilde{c}_r(i,j)]$, where $\widetilde{c}_r(i,j) = x_r(i,j)$ if $(i,j) \in H(A)$ and $\widetilde{c}_r(i,j) = 0$ otherwise.

Using this representation, the matching of the aggregate counting matrix follows from the linear constraints

$$\sum_{r=1}^{g} x_r(i,j) = A(i,r), \qquad (i,j) \in H \tag{4}$$

Pathway lengths are also readily imposed by setting

$$\sum_{(i,j)\in H} x_r(i,j) = l(r), \qquad r = 1, \ldots, g \tag{5}$$

To describe the initial state of the pathway, we introduce an additional set of integer variables $u_r = [u_r(j)]$, $u_r(j) \geq 0$, where $j \in H(a)$, $H(a)$ being the set of non-zeros of the aggregate initial vector a. The decomposed initial vector is then $\widetilde{S}_r(j) = u_r(j)$ if $j \in H(a)$ or $\widetilde{S}_r(j) = 0$ otherwise. We then require that there exists a single initial event

$$\sum_{j\in H(a)} u_r(j) = 1, \qquad r = 1, \ldots, g \tag{6}$$

and that the overall sum of the initial vectors matches the aggregate initial counts

$$\sum_{r=1}^{g} u_r(j) = a(j), \qquad j \in H(a) \tag{7}$$

The above formulation generally overcomes the main issues of the previous representations. However, compared with the sequence representation, it does not ensure that the decomposed counting matrices

define a valid pathway. For example, the decomposed counting matrices

$$\widetilde{C}'_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \widetilde{C}''_r = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

with initial vector $\widetilde{v}_r = (1,0,0)$ would be both invalid: $\widetilde{C}'_r$ has no transition outgoing to the initial event; conversely, $\widetilde{C}''_r$ has no valid sequence of transitions connecting to the exit event $e = 3$. To exclude these behaviors, we need to impose *flow balance* constraints for each node, i.e., impose that all nodes (except the last one) have the same count of incoming and outgoing transitions, excluding self-loops, and accounting also for the contribution of the initial state. Formally,

$$\delta_{\{i \in H(a)\}} u_r(i) + \sum_{k:(i,k)\in H(A) \text{ and } i \neq k} x_r(i,k) = \sum_{k:(k,j)\in H(A) \text{ and } j \neq k} x_r(k,j), \qquad i = \{1,\ldots,|N|\} \setminus \{e\} \tag{8}$$

where $\delta_{\mathscr{P}} = 1$ if predicate $\mathscr{P}$ is true or zero otherwise. To prevent getting caught in a self looping node, we constrain that if a diagonal value is non-zero, it must have a transition out. Given by,

$$x_r(i,i) > 0 \implies \sum_{(i,j)\in H \text{ and } j \neq i} x_r(i,j) > 0, \quad i \in H(a) \tag{9}$$

Lastly, we require that there is a single transition to the end state, i.e.,

$$\sum_{(i,e)\in H(A)} x_r(i,e) = 1 \tag{10}$$

One limitation with the above formulation is that it does not rule out the presence of multiple connected components in the count matrix of a patient.

4.2 Optimization Programs

Leveraging the count variables, we note that the set of constraints (4)-(10) are all linear in the integer variables $x_r(i,j)$ and $u_r(j)$. Thus, any feasible solution to an integer linear program (ILP) with such constraints and also imposing the bounds on the variables $x_r(i,j)$ and $u_r(j)$ will be consistent with the information available from (a,A), and the pathway lengths in l. In general, this solution does not need to be unique, due to the aforementioned problem of the spurious pathways.

As shown in the illustrative example, to reduce the incidence of spurious pathways, we may search within the feasible set for solutions also consistent with the standard deviation matrix S. However, such formulation involves quadratic terms, making it considerably harder to find solutions than in an ILP. To address this problem, we propose two bi-level optimization strategies, described in the following subsection.

4.2.1 Integer Quadratic Program

In this formulation, we consider the Integer Quadratic Program (IQP) with integer variables to maximize the cosine similarity $\theta(\widetilde{S},S)$ of the decomposed standard deviation matrix $\widetilde{S}$ from the true standard deviation matrix S. This is given by

$$\begin{aligned} \theta^*_S = \max_{u,x} \ & \theta(\widetilde{S}(u,x),S) \\ & s.t. \text{ constraints } (4)-(10) \\ & u_r(j) \in \{0,1\} \qquad\qquad j = 1,\ldots,H(a); \\ & x_r(i,j) \in \{0,\ldots,A(i,j)\} \quad i,j = 1,\ldots,H(A). \end{aligned} \tag{11}$$

A limitation of this formulation is that IQPs are known to be significantly harder to solve than ILPs (Eiben et al. 2019), thus we propose next a set of strategies to ease the evaluation of the program by restricting the set of feasible solutions.

4.2.2 Bi-level Pathway Decomposition (BPD)

In this approach, we propose a bi-level optimization program, where we use an ILP in the *inner program*, leveraging a linear objective function with cost vector $c = (c_i)$ to control the decomposition. The vector c is then chosen optimally by means of an *outer program* that aims at matching the aggregate standard deviation matrix S. We refer to this formula as the Bi-level Pathway Decomposition (BPD) algorithm.

In BPD, the inner ILP has a feasible region described by the constraints (4)-(10) and the bounds on the variables $x_r(i,j)$ and $u_r(j)$. In the real-world parameterization we use in the validation, this problem can be typically solved in the order of seconds at most.

Conversely, the *outer program* varies the cost vector $c = (c_i)$ in the objective function of the inner ILP so as to force the inner program to return different candidate feasible solutions. Formally, the bi-level program has the outer program

$$\theta_S^* = \max_{c \in [-1,1]^h} \theta(\widetilde{S}(u_c^*, x_c^*), S), \tag{12a}$$

where $h = g|H(a)| + g|H(A)|$ is the total number of variables in the inner program, $\theta(\cdot,\cdot)$ is the cosine similarity, and $\widetilde{S}$ is the decomposed standard deviation matrix associated with the minimizer (u_c^*, x_c^*) of the inner program. To define the latter, let $c_r^x(i,j)$ define a cost coefficient associated with variable $x_r(i,j)$ and similarly let $c_r^u(i,j)$ be a cost associated with variable $u_r(i,j)$. Then the inner program is

$$\begin{aligned}
(u_c^*, x_c^*) = \arg\min_{(u,x)} \quad & \sum_{r=1}^{g} \sum_{j \in H(a)} c_r^u(j) u_r(j) + \sum_{r=1}^{g} \sum_{(i,j) \in H(A)} c_r^x(i,j) x_r(i,j) \\
s.t.\ & \text{constraints } (4)-(10) \\
& u_r(j) \in \{0,1\} && j = 1, \ldots, H(a); \\
& x_r(i,j) \in \{0, \ldots, A(i,j)\} && i,j = 1, \ldots, H(A).
\end{aligned} \tag{12b}$$

Note that since both the objective and the constraints are linear, the resulting program can be optimized using a standard integer (or mixed-integer) linear programming solver. Upon finding the optimal solution of the outer program (12a), BPD finally returns the values (u_c^*, x_c^*) associated with the optimal vector c.

It should be noted that, due to elementary properties of linear programs on convex sets, the inner program does not allow us to obtain solutions in the interior of the feasible region, as the ILP will eventually return a solution on the convex hull of the feasible set, if a finite solution exists. Thus, by varying the objective function by modifying the c vector, several vertices of the convex hull can be identified. Indeed, we show later in the experimental validation, this is sufficient to achieve significant improvements in the decomposed standard deviation matrix compared with a method that does not use the bi-level approach.

4.2.3 Iterated Bi-level Pathway Decomposition (IBPD)

In this strategy, we define a variant of BPD, called Iterated BPD (IBPD), where we iteratively adapt the upper bound of the $x_r(i,j)$ variables. This is advantageous since we know their average value to be much smaller, i.e., $A(i,j)/g$. Thus, the upper bound $A(i,j)$ can become loose as g grows.

In detail, the iterative bi-level optimization program requires at iteration $t = 0, 1, \ldots$ that we restrict the range of the decision variables as $x_r(i,j) \in \{0, \ldots, \lceil (A(i,j)+t)/g \rceil\}$. If the problem is unfeasible, we continue to the next iteration $t+1$. Conversely, we obtain and store $(u_c^*, x_c^*, \theta_S^*)$. The process is iterated, skipping any iteration t where none of the upper bounds has changed compared with the previous iteration $t-1$. Finally, we return the pair (u_c^*, x_c^*) associated with the best θ_S^* value.

5 EXPERIMENTS

5.1 Methodology

In this section, we run experiments to compare the optimization programs using two similarity metrics:

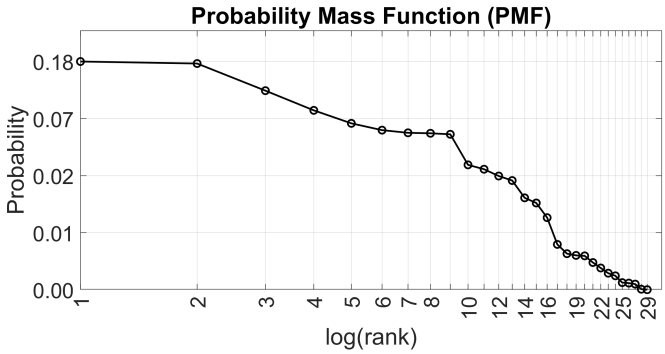


Figure 1: Node popularity in orthopedic surgery dataset.

- Θ: the average cosine similarity to the original counting matrices, as introduced in (4).
- Θ_S: the cosine similarity to the aggregate standard deviation matrices, i.e., $\Theta_S = \rho(\widehat{S}, S)$, where $\widehat{S}$ is the standard deviation matrix computed using the decomposed counting matrices $\widetilde{C}_r$.

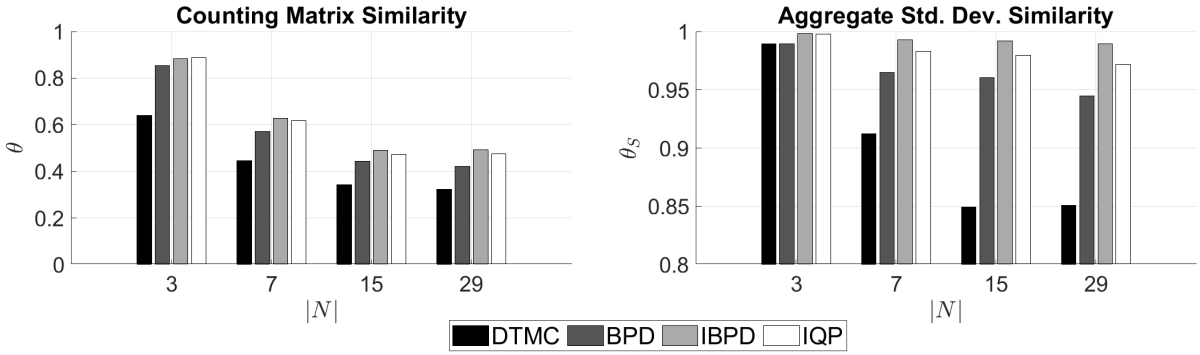
In particular, we compare the three formulations we have proposed in the last section, namely IQP, BPD, and IBPD, together with a baseline Monte Carlo method based on DTMCs. The latter is used to illustrate the value of constrained optimization approaches such as the ones we propose. All four methods are evaluated numerically using code implementations in MATLAB 2023a. Remarks about the implementation of the individual methods are as follows:

- DTMC. This is a baseline Monte Carlo method that repeatedly samples the DTMC obtained from A until obtaining a collection of g decomposed pathways of the required length. The process is repeated 30 times and the decomposition with the highest Θ_S is returned.
- IQP. The integer quadratic program is solved using a genetic algorithm with 3000 generations, as implemented in MATLAB's ga function. The default control to stop the optimization based on stall generations is disabled to ensure that all the 3000 generations are produced. The program stops early only if the objective reaches its maximum value 1.
- BPD/IBPD. The inner program evaluations rely on MATLAB's intlinprog, based on the dual simplex algorithm. Instead, for the outer program we choose a blackbox optimization method that takes into account the high cost of the inner program: we use Bayesian Optimization (BO), as implemented in MATLAB's bayesopt, configured with the *expected-improvement-plus* acquisition function[1]. The latter combines the classic expected improvement function with a correction to avoid overexploitation of an area. The BO algorithm is configured to stop upon reaching 30 evaluations of the outer objective function.

5.2 Dataset

We validate the method using artificial data generated according to realistic parameters aligned to actual clinical pathways in the UK National Healthcare System, using Clinical Practice Research Datalink (CRPD) GOLD (CPRD 2023) linked with Hospital Episode Statistics (HES). Our original dataset contains 2712 aggregated pathways related to elective orthopedic surgery, with an average of 6.11 pathways in each aggregate. Using a pathway represented from the first symptom or diagnosis event, to completed hip or knee replacement surgery. The original aggregation criteria are based on surgery type, UK region, followed by demographics. The dataset includes medical events ranging from primary care, such as family doctor visits, to secondary care, which encompasses elective procedures.

[1] https://uk.mathworks.com/help/stats/bayesian-optimization-algorithm.html



(a) (b)

Figure 2: Counting matrix distance and aggregate standard deviation matrix distance results. $|N|$ represents the number of medical events, Θ_S is the cosine similarity of the standard deviation matrices.

As the original dataset does not have decomposed pathways, we generate artificial data with similar aggregate characteristics to be able to verify the correctness of the decomposition. Each artificial aggregate matrix consists of $g = 6$ pathways. We vary the number of medical events in $|N| \in \{3, 7, 15, 29\}$, where $|N| = 29$ matches the value observed in the aggregate orthopedics dataset (29.1). The probability of sampling an event i in a pathway matches the popularity of the top-$|N|$ most popular events in the real-world trace, renormalized to have support $\{1, \ldots, |N|\}$. This distribution is illustrated in Figure 1. The pathway length is drawn randomly from the range $[2, 4|N| + 1]$. For $|N| = 29$, this has a mean of about 59.5 events, which is slightly more complex than the real-world dataset, which has a mean of 49.0 events.

5.3 Results

Experimental results are summarized in Figure 2. As expected, Figure 2a shows that, as the number of medical events $|N|$ grows, the cosine similarity generally decreases. This appears unavoidable due to the necessary increase in the number of spurious paths that meet all known constraints.

Aside from this systematic trend, the results indicate that the IQP, BPD and IBPD methods deliver systematically higher similarity values in both counting matrix distance (Θ; Figure 2a) and aggregate standard deviation similarity (Θ_S; Figure 2b). In particular, while BPD is not as effective as IQP, the additional tunings that define its IBPD variant make the bi-linear formulation generally more effective for both Θ and Θ_S. In particular, for $|N| = 3$ the DTMC method achieves $(\Theta_S, \Theta) = (0.98930, 0.63942)$, considerably lower than the IBPD values $(0.99813, 0.88379)$. For larger number of events, with $|N| = 29$, DTMC scores $(0.85084, 0.32377)$ while IBPD reaches $(0.98945, 0.49155)$. Overall, IBPD improves the DTMC method by 16% in Θ_S and by 52% in Θ. We further note that the performance of IBPD appears more stable in terms of Θ_S value across all the instances, whereas the other methods display a more marked decrease in aggregate standard deviation similarity as the number of medical events grows.

To better understand the role that each IBPD feature plays in determining the results, we present in Table 1 an ablation analysis of the method. Three variants of IBPD are considered:

- "IBPD Inner". In this variant, we obtain a solution by solving IBPD's inner program only, with $c = (1, \ldots, 1)$. The results are found to be fairly accurate, although less than the proposed IBPD method. Thus, this variant offers a good heuristic when there is a need to execute IBPD without an expensive Bayesian optimization search.
- "IBPD Monte Carlo". This method replaces Bayesian Optimization with random sampling of 30 c vectors. The results indicate that the method is dominated by IBPD, thus further corroborating the effectiveness of Bayesian Optimization for the problem at hand. In the case of $|N| = 29$, the

Table 1: Ablation analysis results. Bold highlights the best objective function value Θ_S.

	$\|N\| = 3$		$\|N\| = 7$		$\|N\| = 15$		$\|N\| = 29$	
Method	Θ_S	Θ	Θ_S	Θ	Θ_S	Θ	Θ_S	Θ
IBDP	**0.99813**	0.88379	**0.99214**	0.62809	**0.99162**	0.49119	**0.98945**	0.49155
IBDP Inner	0.99215	0.85512	0.98559	0.61599	0.98711	0.48483	0.98589	0.48654
IBDP Monte Carlo	0.99201	0.87197	0.98547	0.62054	0.98684	0.49307	0.98593	0.49324
IBDP w/o Path Length	0.95857	0.68769	0.95267	0.50158	0.94815	0.41417	0.93589	0.41544

Θ value of the variant is higher than for IBPD. However, this is a result of the fact that lower Θ_S scores do not strictly result in higher Θ values. Since the optimization does not know the original counting matrices, it is not possible to optimize directly Θ.

- "IBPD w/o Path Length". In this variant, the formulation does not impose the pathway length constraints (5). The major drop in both similarity scores indicates the significance of constraining the pathway counts.

Summarizing, the results indicate that the IBPD provides the best performance among the considered methods and delivers significantly higher similarity to the original counting matrices than sampling the DTMC associated to the aggregate matrix A. The ablation analysis further demonstrates that Bayesian optimization systematically improves over the other strategies to compare feasible solutions subject to constraints (4)-(10). Computational time for all methods exhibits a significant increase with the complexity of the system, particularly with increased $|N|$ such as BPD increasing from 15 minutes to 7 hours with 3 and 29 events respectively. IBPD, employing an additional iterative approach, incurs a threefold increase in computation time compared with BPD. Notably, IQP terminates upon reaching a predefined objective threshold, resulting in reduced computation time compared with other methods.

6 CONCLUSION

In this paper, we have considered the problem of clinical pathway decomposition from aggregate datasets. After comparing representations for the decision variables, we adopt a search style based on count variables. Then, we define an Integer Quadratic Program (IQP), a Bi-linear Pathway Decomposition (BPD), and an iterative variant of the latter (IBPD) to obtain decomposed pathways as close as possible to the original pathways. The BPD and IBPD rely on Bayesian optimization to reduce their computational cost.

Numerical results indicate that the IBPD method provides the best performance. Conversely, BPD provides slightly worse performance but is less computationally demanding. The IQP method, instead, while dominated by the proposed bi-level approach, is more customizable as it may be easily expanded with other non-linear constraints. For clinical pathway simulation modeling, the proposed methods enable a data-driven discovery of higher fidelity pathways without manual clinical input, thereby improving the accuracy of future pathway simulation models.

Future extensions of this work may consider the availability of multiple aggregate count matrices simultaneously and the possible presence of noise in count data, as suggested for example in anonymization schemes based on differential privacy (Olatunji et al. 2022).

ACKNOWLEDGMENTS

William Plumb is supported by UKRI Centre for Doctoral Training in AI for Healthcare (EP/S023283/1).

REFERENCES

Casas-Roma, J., J. Salas, F. D. Malliaros, and M. Vazirgiannis. 2019. "k-Degree Anonymity on Directed Networks". *Knowledge and Information Systems* 61:1743–1768.

Chang, L., J. X. Yu, L. Qin, X. Lin, C. Liu, and W. Liang. 2013. "Efficiently Computing K-edge Connected Components via Graph Decomposition". In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 205–216. Association for Computing Machinery.

CPRD 2023. "Clinical Practice Research Datalink (CRPD) GOLD January 2023 (Version 2023.01.001), Protocol 21 000496".

Eiben, E., R. Ganian, D. Knop, and S. Ordyniak. 2019. "Solving Integer Quadratic Programming via Explicit and Structural Restrictions". In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 33, 1477–1484. AAAI.

Fan, W., J. Li, X. Wang, and Y. Wu. 2012. "Query Preserving Graph Compression". In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, 157–168. Association for Computing Machinery.

Figueira, A. and B. Vaz. 2022. "Survey on Synthetic Data Generation, Evaluation Methods and GANs". *Mathematics* 10(15):2733.

Fung, B. C., K. Wang, R. Chen, and P. S. Yu. 2010. "Privacy-Preserving Data Publishing: A Survey of Recent Developments". *ACM Computing Surveys (CSUR)* 42(4):1–53.

Hernandez, M., G. Epelde, A. Alberdi, R. Cilla, and D. Rankin. 2022. "Synthetic Data Generation for Tabular Health Records: A Systematic Review". *Neurocomputing* 493:28–45.

Hoffmann, H., G. Zhao, S. Asseng, M. Bindi, C. Biernath, J. Constantin, *et al*. 2016. "Impact of Spatial Soil and Climate Input Data Aggregation on Regional Yield Simulations". *PLOS One* 11(4):e0151782.

Hu, L., S. Xu, C. Li, C. Yang, C. Shi, N. Duan, *et al*. 2020. "Graph Neural News Recommendation with Unsupervised Preference Disentanglement". In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4255–4264. Association for Computational Linguistics.

Kumar, S. and P. Kumar. 2017. "Upper Approximation Based Privacy Preserving in Online Social Networks". *Expert Systems with Applications* 88:276–289.

Majeed, A. and S. Lee. 2020. "Anonymization Techniques for Privacy Preserving Data Publishing: A Comprehensive Survey". *IEEE Access* 9:8512–8545.

Mohapatra, D. and M. R. Patra. 2019. "Anonymization of Attributed Social Graph using Anatomy Based Clustering". *Multimedia Tools and Applications* 78:25455–25486.

Murthy, S., A. Abu Bakar, F. Abdul Rahim, and R. Ramli. 2019. "A Comparative Study of Data Anonymization Techniques". In *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity)*, 306–309. IEEE.

Olatunji, I. E., J. Rauch, M. Katzensteiner, and M. Khosla. 2022. "A Review of Anonymization for Healthcare Data". *Big Data*.

Patterson, Z., M. Kryvobokov, F. Marchal, and M. Bierlaire. 2010. "Disaggregate Models with Aggregate Data: Two UrbanSim Applications". *Journal of Transport and Land Use* 3(2):5–37.

Standfield, L., T. Comans, and P. Scuffham. 2014. "Markov Modeling and Discrete Event Simulation in Health Care: A Systematic Comparison". *International Journal of Technology Sssessment in Health Care* 30(2):165–172.

Sun, J., Y. Xie, H. Zhang, and C. Faloutsos. 2007. "Less is More: Compact Matrix Decomposition for Large Sparse Graphs". In *Proceedings of the 2007 SIAM International Conference on Data Mining*, 366–377. SIAM.

Sweeney, L. 2002a. "Achieving K-Anonymity Privacy Protection using Generalization and Suppression". *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10(05):571–588.

Sweeney, L. 2002b. "K-Anonymity: A Model for Protecting Privacy". *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10(05):557–570.

Wang, X., H. Jin, A. Zhang, X. He, T. Xu, and T. S. Chua. 2020. "Disentangled Graph Collaborative Filtering". In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1001–1010.

Williams, D. W., J. Huan, and W. Wang. 2006. "Graph Database Indexing using Structured Graph Decomposition". In *2007 IEEE 23rd International Conference on Data Engineering*, 976–985. IEEE.

AUTHOR BIOGRAPHIES

WILLIAM PLUMB is a PhD student within the UKRI Centre for Doctoral Training in AI for Healthcare at Imperial College London. His research focuses on the use of AI methods, such as simulations, to understand and optimize clinical pathways design for better pathway care. His email address is w.plumb20@imperial.ac.uk.

ALEX BOTTLE is a Professor of Medical Statistics in the School of Public Health at Imperial College London. His research focus is on using routinely collected data to measure and explain variations in health service quality and safety with the aim of improving patient care. Having published over 400 peer reviewed papers, he serves as an Associate Editor of BMJ Quality & Safety and Deputy Statistics Editor at Thorax. His email address is robert.bottle@imperial.ac.uk.

GIULIANO CASALE joined the Department of Computing at Imperial College London in 2010, where he is currently a Reader. He teaches and does research in performance engineering and cloud computing, topics on which he has published more than 150 refereed papers. He served as Chair of ACM SIGMETRICS during 2019-2023. His email address is g.casale@imperial.ac.uk.