Trajectories, bifurcations and pseudotime in large clinical datasets: applications to myocardial infarction and diabetes data

Sergey E. Golovenkin[1], Jonathan Bac[2,3,4], Alexander Chervov[2,3,4], Evgeny M. Mirkes[5,6], Yuliya V. Orlova[1], Emmanuel Barillot[2,3,4], Alexander N. Gorban[5,6] and Andrei Zinovyev[2,3,4]

[1]Krasnoyarsk State Medical University, 660022, Krasnoyarsk, Russia and [2]Institut Curie, PSL Research University, F-75005 Paris, France and [3]INSERM, U900, F-75005 Paris,France and [4]CBIO-Centre for Computational Biology, Mines ParisTech, PSL Research University, 75006 Paris, France and [5]University of Leicester, LE1 7RH, Leicester, UK and [6]Lobachevsky University, 603000 Nizhny Novgorod, Russia

Abstract

Large observational clinical datasets become increasingly available for mining associations between various disease traits and administered therapy. These datasets can be considered as representations of the landscape of all possible disease conditions, in which a concrete pathology develops through a number of stereotypical routes, characterized by 'points of no return' and 'final states' (such as lethal or recovery states). Extracting this information directly from the data remains challenging, especially in the case of synchronic (with a short-term follow up) observations. Here we suggest a semi-supervised methodology for the analysis of large clinical datasets, characterized by mixed data types and missing values, through modeling the geometrical data structure as a bouquet of bifurcating clinical trajectories. The methodology is based on application of elastic principal graphs which can address simultaneously the tasks of dimensionality reduction, data visualization, clustering, feature selection and quantifying the geodesic distances (pseudotime) in partially ordered sequences of observations. The methodology allows positioning a patient on a particular clinical trajectory (pathological scenario) and characterizing the degree of progression along it with a qualitative estimate of the uncertainty of the prognosis. Overall, our pseudo-time quantification-based approach gives a possibility to apply the methods developed for dynamical disease phenotyping and illness trajectory analysis (diachronic data analysis) to synchronic observational data. We developed a tool *ClinTrajan* for clinical trajectory analysis implemented in Python programming language. We test the methodology in two large publicly available datasets: myocardial infarction complications and readmission of diabetic patients data.

Key words: Clinical data; Clinical Trajectory; Patient disease pathway; Dynamical diseases phenotyping; Data analysis; Principal trees; Dimensionality reduction; Clustering; Pseudotime; Myocardial infarction; Diabetes

Background

Large observational datasets are becoming increasingly available, reflecting physiological state of observed individuals, their lifestyles, exposure to environmental factors, received treatments and passed medical exams. From the big data point of view, each person's life can be represented as a trajectory in a multidimensional space of qualitative or quantitative traits. Simultaneous analysis of a large number of such trajectories can reveal the most informative features whose dynamics is correlated with trajectory clusters, associations between various factors and, potentially, the "points of no return", i.e. bifurcations representing important fate decisions.

The most important applications of such a framework are medical. The notion of "disease trajectory" as a person's trajectory in the data space of various diagnoses (diseases quantified by their severity) accompanying the person's life has emerged recently and became available for the large-scale analyses in certain contexts [1, 2]. For example, a dataset containing an electronic health registry collecting during 15 years and covering the whole population of Denmark, with 6.2 million individuals, have been analyzed with an objective to determine previously unreported disease co-morbidities [1]. An ambitious 'Data Health Hub' (https://www.health-data-hub.fr/) project has been recently launched in France with the aim to make available for machine learning-based analysis the collection of several decades-long population-wide anonymized health insurance records [3]. A formal review and meta-analysis of sci-

Compiled on: October 11, 2024.
Draft manuscript prepared by the author.

Key Points

- Large-scale observational clinical datasets represent landscapes of the variety of disease states
- Diachronic clinical trajectories can be approximated from the multi-dimensional geometry of synchronic data and used for disease dynamical phenotyping
- ClinTrajan: Python package for finding and analyzing clinical trajectories using elastic principal graph method

entific texts using the concept of patient trajectory (or clinical pathway) based on disease management and care but also considering medico-economic aspects with a focus on myocardial infarction has been recently published in [4].

Dynamical phenotyping is the conceptual paradigm underlying such studies which can be applied at organismal and cellular scales [5, 6, 7]. It states that distinguishing various dynamical types of progression of a disease or a cellular program is more informative than classifying biological system states at any fixed moment of time, because the type of dynamics is more closely related to the underlying hidden mechanism. From the machine learning point of view, this dictates different choices of methods, with clustering more adapted to the synchronic (snapshot) data [8] while more specific methods for trajectory analysis are needed in the case of diachronic (having important temporal aspect) data [9, 10, 11, 12, 13]. The dynamical phenotyping paradigm and accompanying data mining methodologies become even more important with wider introduction of various types of continuous health monitoring devices and apps [14].

However, examples of massive comprehensive and life-long longitudinal clinical data are still rare. Most of the existing clinical datasets correspond to relatively short periods of patients' stays inside hospitals, or during their treatment for a particular disease. In this sense, clinical datasets frequently represent detailed but rather "static snapshot" than the dynamical picture of the individuals' states. Nevertheless, one can hypothsize that such a snapshot, if sufficiently large, can sample the whole landscape of possible clinical states with certain routes and branches corresponding to some averaged illness trajectories. Then each patient can be thought to occupy a particular position along such a trajectory, where those patients following the same trajectory can be ranked accordingly to the progression along it from the hypothetical least heavy state towards some extreme state.

This situation is reminiscent of some recent studies of molecular mechanisms of several highly dynamical biological processes such as development or differentiation, at single cell level. Indeed, profiling a snapshot of a cell population can capture individual cells in the variety of different states (e.g., map their progression through the cell cycle phases). This allows reconstructing cellular trajectories through sampling the dynamics of the underlying phenomenon without necessity to follow each individual cell in time [15, 16]. In this field, a plethora of machine learning-based methods have been recently suggested in order to capture the cellular trajectories and quantify progression along them in terms of *pseudo-time*, representing the total number of molecular changes in the genome-wide profiles of individual cells [16] rather than physical time. Many of these methods are able to detect branching trajectories, where the branches can represent important bifurcations (or, cell fate decisions) in the dynamical molecular processes underlying differentiation of developmental programs.

The aim of the present study is to suggest and test a computational methodology for extracting clinical trajectories from sufficiently large synchronic clinical datasets. Clinical trajectory is a clinically relevant sequence of ordered patient phenotypes representing consecutive states of a developing disease and leading to some final state (i.e., a lethal outcome). Importantly, in our approach we do not assume that these are the same patient's states, even if this can be so in the case when there exist some longitudinal observations. Each clinical trajectory can be characterized by its proper pseudotime which allows one to quantitatively characterize the degree of progression along the trajectory. Each clinical variable can be analyzed as a function of pseudotime conditioned on a given clinical trajectory. We also assume that clinical trajectories can be characterized by branches (bifurcations), representing important decisive moments in the course of a disease.

An important methodological difference between the previously developed methodology of cell trajectory analysis in omics datasets, where the majority of the variables can be considered continuous and of similar nature (e.g., gene expression levels), the clinical datasets possess certain specifics which must be taken into account. Typical real-life clinical data are characterized by the following features: a) they contain mixed data types (continuous, binary, ordinal, categorical variables, censored data); b) they typically contain missing values with non-uniform missingness pattern across the data matrix; c) they do not have a uniquely defined labeling (part of the clinical variables can be used to define clinical groups, but this can be done in several meaningful ways). This means that an important integral part of the methodology should be procedures for quantifying and imputing missing values in mixed type datasets making them amenable for further application of machine learning methods. The last feature (c) suggests that unsupervised or semi-supervised methodology might play more important and insightful role here than purely supervised methods.

We develop a methodology of clinical data analysis, based on modeling the multi-dimensional geometry of a clinical dataset as a "bouquet" of diverging clinical trajectories, starting from one or several quasi-normal (least severe) clinical states. As a concrete approach we exploit the methodology of elastic principal trees (EPT), which is a non-linear generalization of Principal Component Analysis (PCA). Principal tree is a set of principal curves assembled in a tree-like structure, characterized by branching topology [17, 18]. Principal trees can be constructed using ElPiGraph computational tool, which has been previously exploited in determining branching trajectories in various genomics datasets (in particular, in single cell omics data) [19, 15, 20]. As an unsupervised machine learning method, estimating elastic principal graphs solves several tasks simultaneously, namely dimensionality reduction, data visualization, partitioning the data by the non-branching graph segments (analogous to clustering) and quantifying robust geodesic distances (pseudo-times) from one data point to another along the reconstructed principal graph. Unlike many other methods relying on heuristics for guessing the optimal graph topology (e.g., a tree) such as Minimal Spanning Tree (MST), elastic principal graph method optimizes the graph structure via application of topological grammars and gradient descent-like optimization in the discrete space of achievable graph structures (e.g., all possible tree-like graphs)[19].

The suggested method is implemented as Python package, *ClinTrajan*, which can be easily used in the analysis of clinical datasets. We provide several reproducible Jupyter notebooks illustrating the different steps of the methodology. The figures in this paper are directly copied from these notebooks. The methodology proved to be scalable to the datasets containing hundreds of thousands of clinical observations, using an ordinary laptop, and can be scaled up further for even larger datasets.

Data Description

In this study we apply the suggested methodology to two publicly available clinical datasets, one of moderate size (1700 patients) and one of relatively large size (>100,000 patients).

Complications of myocardial infarction database

Myocardial infarction (MI) is one of the most dangerous diseases. The wide spread of this disease over the past half century has made it one of the most acute problems of modern medicine. The incidence of myocardial infarction (MI) remains high in all countries. This is especially true of the urban population of highly developed countries, exposed to the chronic effects of stress factors, irregular and not always balanced nutrition. In the United States annually, more than one million people become ill with myocardial infarction [21].

The course of the disease in patients with MI is diverse. MI can occur without complications or with complications that do not worsen the long-term prognosis. At the same time, about half of patients in the acute and subacute periods have complications leading to a worsening of the course of the disease and even death. Even an expert can not always foresee the development of these complications. In this regard, predicting the complications of myocardial infarction in order to timely carry out the necessary preventive measures seems to be an important task.

The database analyzed here was collected in the Krasnoyarsk Interdistrict Clinical Hospital (Russia) in 1992-1995 years, but has only recently been deposited to the public domain. The original database and its description can be downloaded from [22]. It contains information about 1700 patients and 110 features characterizing the clinical phenotypes and 12 features representing possible complications of the myocardial infarction disease. Previously, the dataset was a subject of machine learning method applications, including convolutional neural networks [23] and dimensionality reduction methods [24]. We believe that introducing this dataset, which exemplifies the specificity and difficulties of analysing the real-life clinical data, to the big data and machine learning research community should contribute to developing better treatment and subtyping strategies in cardiology and in clinical research in general [25].

Diabetes readmission data set

Together with myocardial infarction, various diabetes-related clinical states such as hyperglycemia are widely spread in the modern population. The management of hyperglycemia in the hospitalized patients has a significant bearing on outcome, in terms of both morbidity and mortality [26]. An assembly and analysis of a large clinical database was undertaken to examine historical patterns of diabetes care in patients admitted to a US hospital and to inform future directions which might lead to improvements in patient safety [26]. In particular, the use of HbA1c as a marker of attention to diabetes care in a large number of individuals identified as having a diagnosis of diabetes mellitus was analyzed. A focus was on the readmission probability of a patient after leaving the hospital and its dependency on other clinical features that can be collected during hospitalization.

The dataset represents 10 years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks. It includes over 50 features representing patient and hospital outcomes. The dataset can be downloaded from UCI repository at https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008 or from Kaggle at https://www.kaggle.com/brandao/diabetes. The data contains more than 100,000 hospitalization cases with patients suffering from diabetis characterized by 55 attributes.

Analyses

ClinTrajan package for trajectory inference in large clinical datasets

We suggest computational methodology of constructing principal trees in order to extract clinical trajectories from large-scale clinical datasets which takes into account their specificity. The following steps of the analysis have been implemented:

- Univariate and multi-variate quantification of nominal variables, including an original implementation of the optimal scaling procedure for ordinal values
- Several methods for missing values imputation including two original implementations of SVD-based imputers
- Constructing principal tree for a quantified clinical dataset
- Partitioning the data accordingly to the non-branching segments of the principal tree (analogue of clustering) and associating the segments to clinical variables.
- Extracting clinical trajectories and associating the trajectories to clinical variables
- Visualization of clinical variables using principal trees and metro map data layouts [27]
- Pseudotime plots of clinical variables along clinical trajectories, visualization of their bifurcations

Myocardial infarction complications case study

Assigning data point classes

As we've mentioned above, the classes of the clinical observations can be usually defined by selecting a subset of clinical variables which represent some final read-outs of a patient state. Thus, in the myocardial infarction complications dataset, 12 clinical variables report the complications, 11 of them represent binary variables and 1 categorical variable LET_IS, whose value is 0 if there is no lethal outcome. Otherwise, LET_IS can take one of the 7 nominal values representing the death cause. Following the methodology suggested in this study, the LET_IS variable is first made a subject of dummy coding, introducing 7 binary features. The resulting 18 binary variables were characterized by 158 unique combinations of 0/1 values, which looked too many to define one class per such unique combination.

Therefore, it was decided to reduce the number of the distinct complication states to a more manageable number by clustering them. The table of 158 possible complications and 18 binary variables was analyzed by the method of elastic principal trees as described below and clustered into 11 clusters accordingly to the principal tree non-branching segments (see Figure 1). 7 of these clusters contained lethal outcomes and clearly corresponded to particular death causes, which corresponds to non-zero values of LET_IS variable. The non-lethal

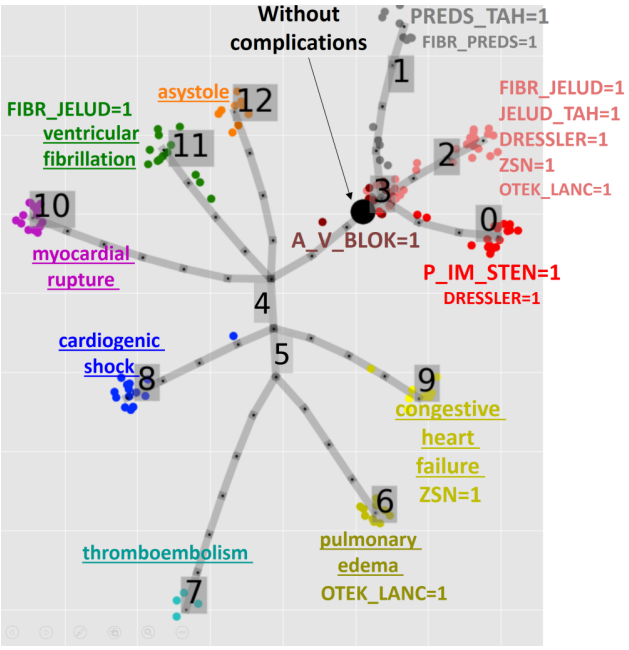

Figure 1. Defining classes of myocardial infarction complications, using principal tree-based clustering. The labeling marks either the cause of death (underlined, for lethal outcome classes) or a set of complication variables strongly overrepresented in the cluster, accordingly to the χ^2 test of independence; the size of the label reflects the significance of over-representation (the larger the label the more significant is the deviation from independence). The meaning of the complication variables here are: FIBR_PREDS - atrial fibrillation, PREDS_TAH - supraventricular tachycardia, JELUD_TAH - ventricular tachycardia, FIBR_JELUD - ventricular fibrillation, A_V_BLOK - third-degree AV block, OTEK_LANC - pulmonary edema, DRESSLER - Dressler syndrome, ZSN - chronic heart failure, REC_IM - relapse of the myocardial infarction, P_IM_STEN - post-infarction angina.

outcomes have been clustered into 4 classes ('0','1','2','3'). Classes '1' and '2' appeared to be characterized by fibrillation and tachycardia, but differed in the types ('1' corresponded to the 'atrium' fibrillation and tachycardia while '2' had a tendency to be characterized by the ventricle ones). Non-lethal class '0' was distinguished by 'P_IM_STEN' (post-infarction angina), and the class '3' – by the presence of diagnosed 'A_V_BLOCK' (third-degree atrioventricular block).

Besides this clustering, a particular non-lethal state was distinguished characterized by zero values of all complication variables. We distinguished this class as a separate 'no complications class'. In the complete dataset, it corresponded to 45% of clinical records (denoted as black points in Figure 1 and Figure 2). In the rest of the analysis, all complication variables have been analyzed together with the 112 clinical characteristics.

Quantification of nominal values and imputation of missing data values

As a first step of pre-processing, 7 variables have been removed from the initial myocardial infarction complication data table, as containing more than 30% of missing values. Afterwards, 126 records have been removed as containing more than 20% of missing values. After this step, the data table contained 2.5% of missing values with 533 rows (34% of all clinical cases) having no missing values.

After the missing value filtering step, the data table of myocardial infarction complications contained 84 binary, 9 continuous numerical, 22 ordinal and 1 categorical variables. Large number of ordinal variables requires careful quantification of them (see Methods), which is not trivial given the large number of rows with missing values.

We considered that the small number of continuous numerical variables is not enough to apply the methodology of Categorical Principal Component Analysis (CatPCA) [28]. Therefore, for all ordinal and binary variables we first applied the univariate quantification following the approach described in the 'Methods' section. This quantification allowed applying 'SVD-Complete' imputation method for imputing the missing values, as described in 'Methods'. After all missing values have been imputed, we could apply the optimal scaling approach for ordinal values, optimizing the pairwise correlations between them and between ordinal and continuous numerical variables. The 22 ordinal variables quantified in this way were further used for forming the data space. In addition, all variables were converted to z-scores.

Constructing elastic principal tree

The initial data space was formed by 123 variables. We evaluated the global intrinsic dimension of the dataset using several methods implemented in https://github.com/j-bac/scikit-dimension, and found that the majority of non-linear methods estimate the intrinsic dimension in the range 10-15 while linear methods based on PCA gives much larger intrinsic dimension values (see Supplementary Figure 1). We compared the estimations of intrinsic dimensions with and without complication variables and found them to be similar, which indicates that there exists a certain level of dependency between the complication variables and the rest of the clinical variables. We also observed that the screen plot for this dataset is characterized by elbow approximately at $n = 12$. As a result of this analysis, for further inference of the principal tree, we projected the dataset into the space of the first 12 principal components.

The elastic principal tree was computed using ElPiGraph Python implementation as documented in the Jupyter notebook provided at https://github.com/auranic/ClinTrajan and in the Section 'Method of Elastic Principal Graphs (ElPiGraph)' of this manuscript. The principal tree explained 52.4% of total variance in contrast to the first two principal components that explained 25.9% and first five PCs explaining 54%. The obtained principal tree (shown in Figure 2) was used to provide a 2D layout of the dataset which can be used for visualization of various clinical variables and the results of analyses. Globally, the principal tree defined three terminal non-branching segments populated with non-lethal clinical cases (indicated as #3,#5,#6 in Figure 2, panel 'Tree segments (branches)') and associated with younger patients (Figure 2, panel 'AGE'). Other terminal segments (#0,#7,#9,#10,#12,#14,#15) were characterized by various risks of lethality (Figure 2, panel 'Lethal cases'), with two terminal segments #12 and #15 being strongly enriched with lethal cases, caused by cardiogenic shock and myocardial rupture correspondingly.

Each node of the principal tree is connected with a subset of data points. We performed the enrichment analysis, based on application of independence χ^2 test in order to determine the node which is the most strongly associated to 'no complication' class (black points in Figure 2). The position of this node (#8) is indicated as 'Root node' in Figure 2, main panel.

Dataset partitioning (clustering) by principal tree non-branching segments

The explicitly defined structure of the computed elastic principal tree allows partitioning the dataset accordingly to the projection of the data points on various internal and terminal segments as described in the 'Methods' section and shown by color in Figure 2, panel 'Tree segments (branches)'. Such partitioning can play a role of clustering with the advantage of that the tree segments can recover non-spherical and non-linear data clusters. In addition, the data clusters, defined in such a way,

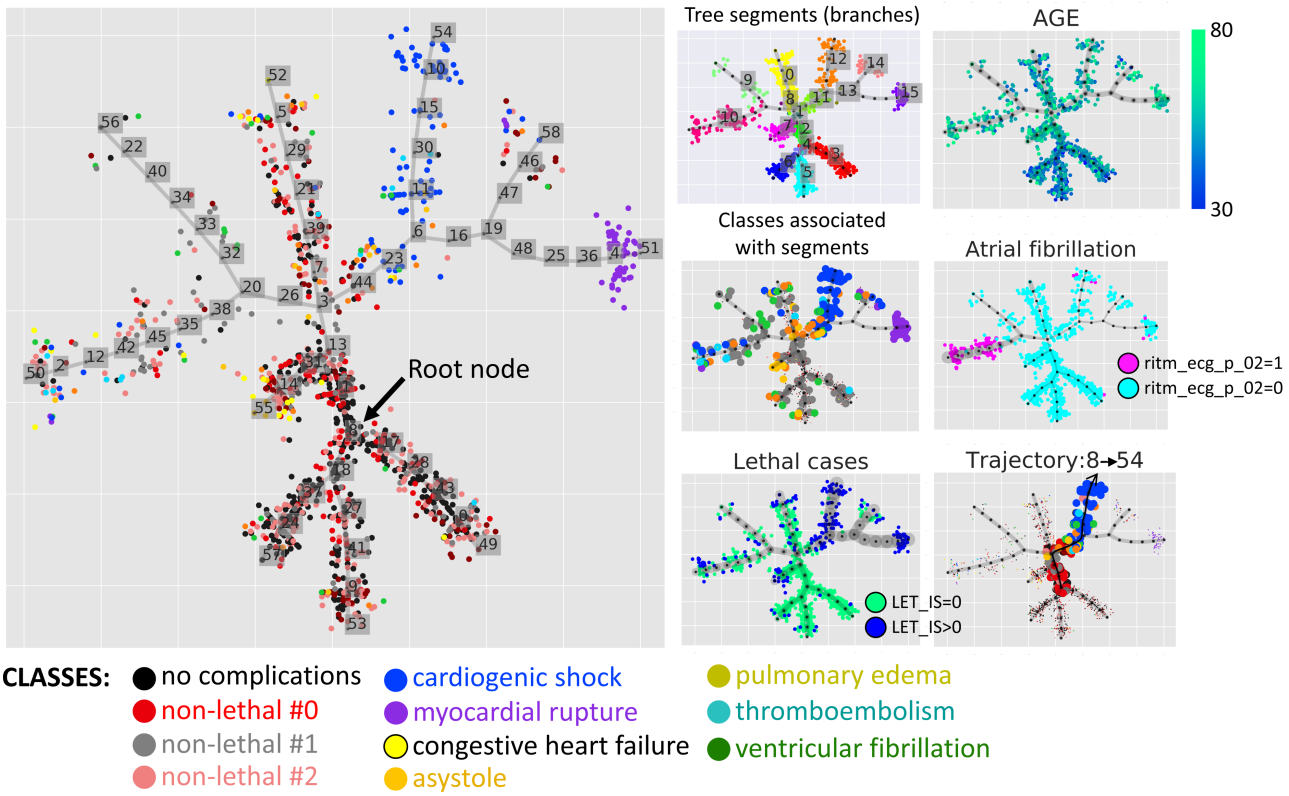

Figure 2. Principal tree recapitulating the multidimensional structure of the myocardial infarction complications dataset. Distribution of classes along the tree is visualized in the large panel. Various modes of data visualization are shown in the small panels. 'Tree segments (branches)' shows partitioning (clustering) of data points accordingly to the linear fragments connecting branching points and/or leaf nodes (called 'non-branching segments' in this work). 'AGE' is an example of a continuous variable visualization using color gradient of data points. 'Classes associated with segments' shows data points of only those classes which have statistical associations with one or more segments. 'Artrial fibrillation' and 'Lethal cases' show visualization of a binary variable, with edge width reflecting the trend (in case of 'Lethal cases' it can be interpreted as lethality risk estimate). 'Trajectory 8→ 54' a subset of data points, colored accordingly to their classes and belonging to one particular clinical trajectory, having the node with the least risk of complications as the root node (node 8) and the highest risk of cardiogenic shock as its final state (node 54).

are connected in a tree-like configuration, with junctions corresponding to the branching points which can correspond to 'points of no return' in the state of the patients.

Each non-branching segment in the tree can be associated by enrichment analysis either to a data class or to a variable. The points of the data classes which are associated to at least one tree segment are highlighted by size in Figure 2, panel 'Classes associated with segments'. The results of enrichment analysis for all clinical variables are shown in Figure 3. Briefly, we found that 44 clinical variables, including 8 complication variables, can be associated to at least one segment (Figure 3) with reasonably high thresholds either for the deviation score (8) or ANOVA linear model coefficient (provided that the results of chi-square or ANOVA tests are statistically significant).

Analysis of clinical trajectories and pseudotime

The non-branching segments of the principal tree are connected into trajectories, from the root node of the tree corresponding to the least frequency of complications to one of the leaf nodes representing some extreme states of the disease (some of which are connected with increased risk of lethality). Internal tree segments are shared between several trajectories, while the terminal segments correspond to one single trajectory. Consequently, each data point can be associated to one or more trajectories. The position of the data point on a trajectory is quantified by the value of pseudo-time characterizing the intrinsic geodesic distance from the root node, measured in the units of the number of tree edges. The value of pseudo-time is continuous since a data point can be projected on a tree edge, in between two nodes.

If a data point (clinical observation) is attributed to several trajectories then it is characterized by the same pseudo-time value on each of them. This can be interpreted as the state of uncertainty from which several clinical scenarios can be developed in the further course of the disease, following one or several bifurcation points. Those clinical observations belonging to a single trajectory correspond to less uncertainty in the prognosis, with higher chances to end up in a terminal state.

In order to determine the factors affecting the choices between alternative clinical trajectories, it is necessary to associate clinical variables to each of the trajectory and determine the trend of their changes along them. Mathematically this corresponds to the solving the regression problem connecting a clinical variable and the observation pseudo-time. Using this approach we identified 35 variables associated with pseudo-time with $R^2 > 0.3$ for at least one trajectory (Figure 4,A,B). The pseudo-dynamics of these variables is shown in Figure 4,C. This analysis allows one to conclude on the sequence of clinical variable changes leading to various complications. Thus, four trajectories $8 \rightarrow 52, 51, 54, 55$ are associated with increasing risks of four distinct lethal outcomes (progress of congestive heart failure, myocardial rupture, cardiogenic shock and pulmonary edema respectively). Three trajectories ($8 \rightarrow 49, 53, 57$) correspond to mild course of the disease associated with younger patients with the risk of ventricular tachycardia increasing along the trajectory $8 \rightarrow 53$.

In order to illustrate the picture of decreasing uncertainty while the disease progress along clinical trajectories, we focused on four trajectories $8 \rightarrow 55, 50, 56, 52$ sharing one or several internal tree segments. We selected several clinical vari-

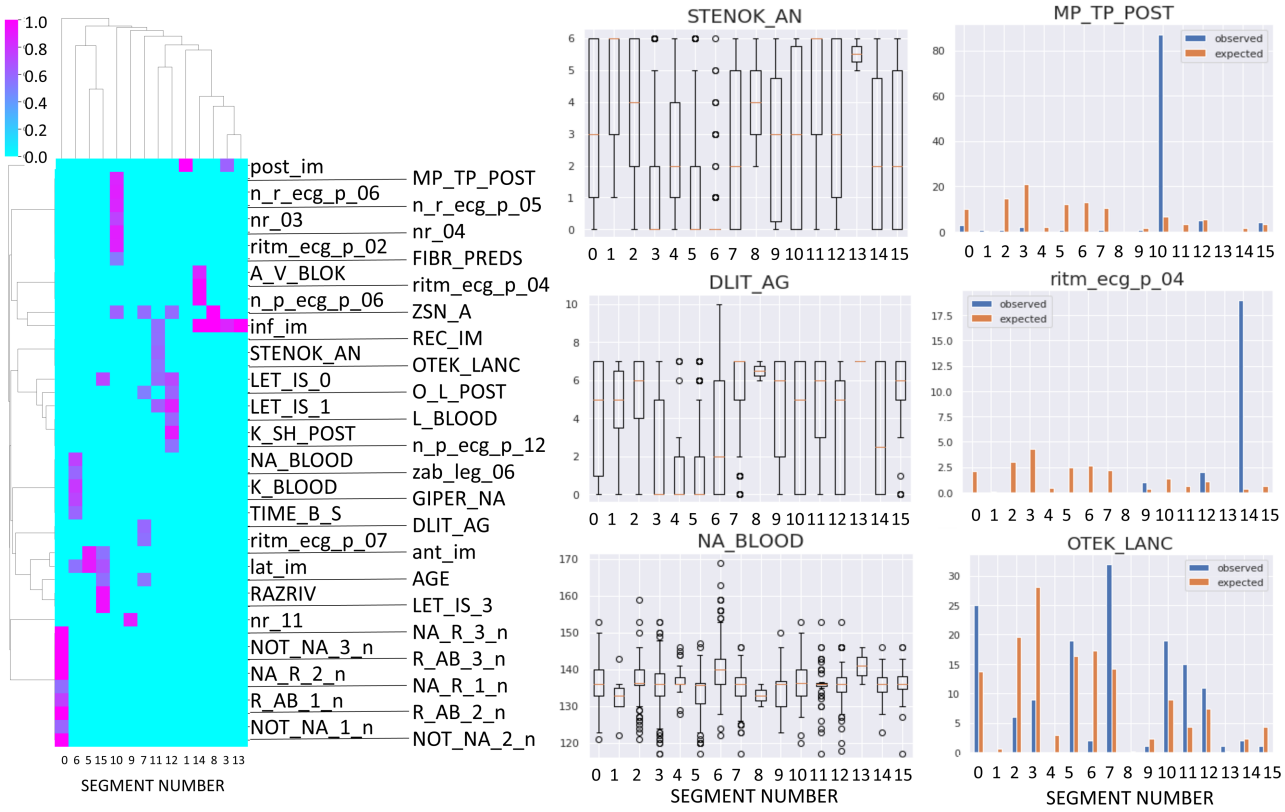

Figure 3. Association of principal tree segments (as shown in Figure 2) with data variables. On the left, a hierarchical clustering dendrogram of association scores is shown. On the right, three examples of strong associations with continuous/ordinal and binary variables are shown. Here the following variables have been shown: STENOK_AN, Exertional angina pectoris in the anamnesis, DLIT_AG, Duration of arterial hypertension (years), NA_BLOOD, Serum sodium conten, mmol/L, MP_TP_POST, Paroxysms of atrial fibrillation, ritm_ecg_p_04, ECGrhythm at the time of admission to hospital–atrial, OTEC_LANC, Pulmonary edema.

ables and two lethal outcome variables associated with pseudo-time along these trajectories and showed them all in one plot (Figure 5). One can see that the pseudo-dynamics of some clinical variables estimated by logistic regression as the probability of value '1', gradually diverge at the branching points of the principal tree.

The trajectory 8 → 55 is characterized by increasing sinus tachycardia after the bifurcation point A and increasing risk of congestive heart failure and to lesser extent the pulmonary edema. The trajectory 8 → 50 is characterized by the absence of sinus tachycardia with gradual decline in the variable 'ECG rhythm - sinus with a heart rate 60-90' after the point B, and, after the bifurcation point C, rapid increase of the probability of paroxysms of atrial fibrillation. The prognosis along this trajectory is relatively favorable as well as on the clinical trajectory 8 → 56, which is characterized by slow and incomplete decrease of the probability of 'ECG rhythm - sinus with a heart rate 60-90' after the point C.

The trajectory 8 → 52 is characterized by high risk of pulmonary edema and gradual increase of sinus tachycardia. One of the distinguishing features of this trajectory is increased use of opioid and antiinflammatory drugs in the intensive care unit at days 2 and 3 after the admission to the hospital, which is in turn connected to the pain relapse (R_AB_2_n variable).

Diabetes readmission case study

Clinical trajectories in large-scale observational diabetes data

In order to check if the *ClinTrajan* package can be applied to larger datasets, we extracted clinical trajectories using a publicly available dataset, representing 10 years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks. The dataset contains 101766 records satisfying the following conditions: (1) it is an inpatient encounter (a hospital admission), (2) it is a diabetic encounter, that is, one during which any kind of diabetes was entered to the system as a diagnosis, (3) the length of stay was at least 1 day and at most 14 days, (4) laboratory tests were performed during the encounter, (5) medications were administered during the encounter. The data contains such attributes as patient race, gender, age, admission type, time in hospital, medical specialty of admitting physician, number of lab test performed, HbA1c test result, diagnosis, number of medication, diabetic medications, number of outpatient, inpatient, and emergency visits in the year before the hospitalization. In the supervised setting, the aim of the analysis of this dataset is usually to predict the readmission event ('readmitted' variable) within 30 days after disposal from the hospital. In our analysis, we considered the readmitted variable as a part of the data space, in order to perform unsupervised analysis of the dataset with the aim to extract clinical trajectories, some of them leading to the increased readmission likelihood.

The exact protocol for encoding the diabetes dataset is provided at the *ClinTrajan* github https://github.com/auranic/ClinTrajan/. Importantly, we encoded several categorical variables as ordinal. In particular, the readmitted variable was encoded in three levels with 0 value corresponding to 'No' (absence of recorded readmission), 1 - to '>30 days' and 2 - to '<30 days'. The 'A1Cresult' feature (related to the HbA1c test) was encoded in two variables. The first one was binary indicating absence ('None' value) or presence of the measurement event. The second was the actual level of HbA1c: missing values corresponding to 'None', and three level encoding for the measured values, 0 - for 'Norm', 1 - for '>7' and 2 for '>8'. Since the A1Cresult field was not 'None' only in 17% of patient

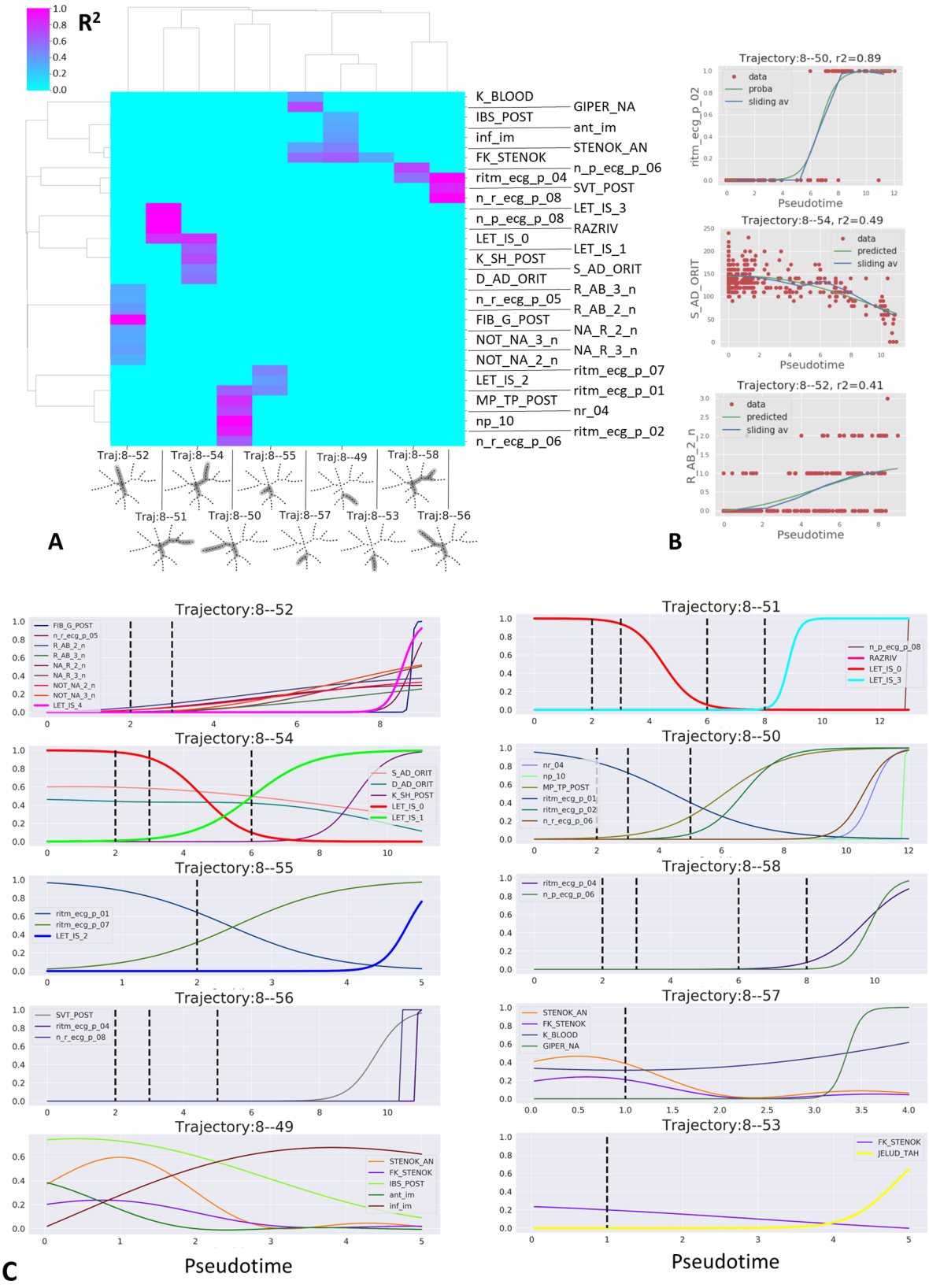

Figure 4. Clinical trajectory analysis of the myocardial infarction complications dataset. A) Visualization of R^2 values for the regression between clinical variables and the pseudo-time along 10 clinical trajectories. B) Examples of regression analysis for a binary (logistic regression), continuous and ordinal (Gaussian kernel regression) clinical variables. C) Pseudotime plots for clinical variables selected by regression analysis. For binary variables, the probability inferred by logistic regression is shown. For ordinal and continuous variables, non-linear regression line is shown. Complication variables associated to the clinical trajectories are shown with thick lines (for example, LET_IS_0 represents the survival probability.) Vertical dashed lines indicate the positions of tree branching points along pseudo-time. The abscissa in the pseudotime plots corresponds to the variable value scaled to unity for the total variable amplitude.

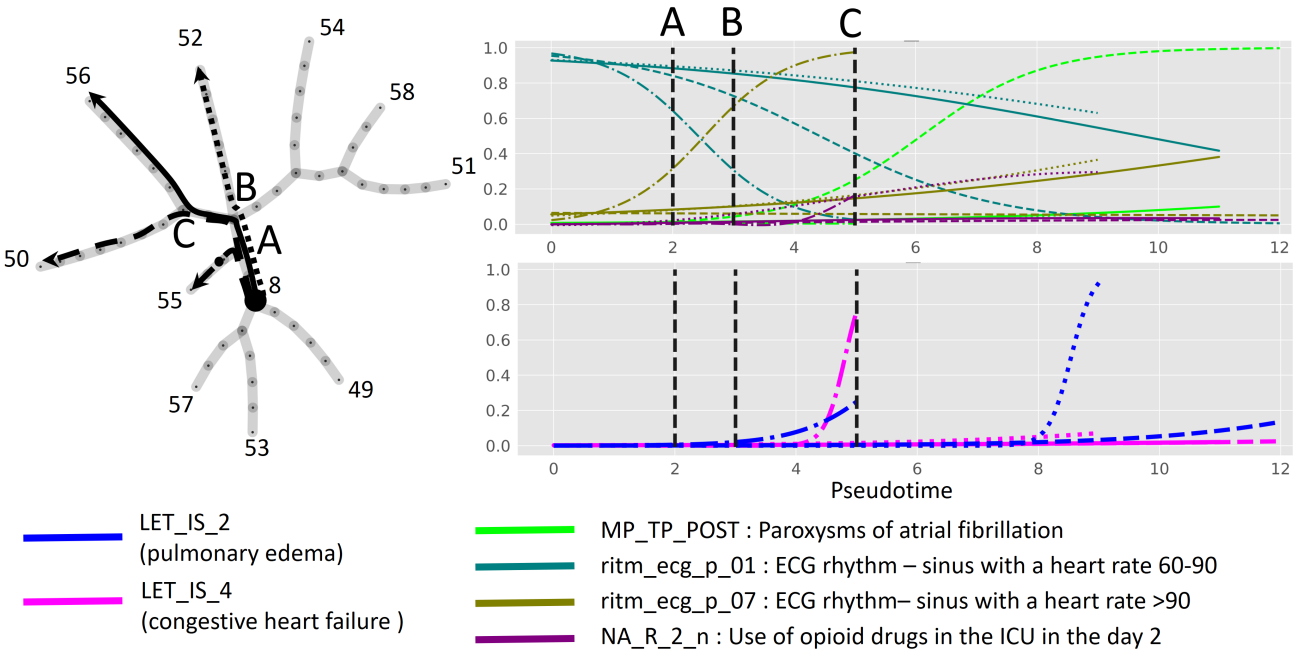

Figure 5. Example of bifurcating clinical trajectories. Four clinical trajectories out of ten are depicted for myocardial infarction complications data. The trajectories all share the internal segment 8-A and diverge at nodes A, B, C. Four selected binary clinical variables and two lethal outcome variables are shown as functions of four pseudotime measurements, one per trajectory. The abscissa in the pseudotime plots corresponds to the variable value scaled to unity for the total variable amplitude.

records, this created a column containing 83% of missing values, which were further imputed from the rest of the data. This was the only variable containing missing values. Age field was encoded as a 10-level ordinal variable accordingly to 10 age intervals provided in the initial data table.

For encoding the 23 categorical fields of the dataset describing the administered medications and change in their dosage, we used the following schema. First, we kept only four most frequently (in >10% of cases) medications: insulin (53% cases), metformin (20% cases), glipizide (12% cases) and glyburide (10.4% cases). Second, each medication field was encoded into two variables: one binary indicating the absence ('No' value) or presence ('Steady' or 'Down' or 'Up') of the treatment prescription, and one three-level with 0 corresponding to either absence or no change in the treatment dose ('No' or 'Steady'), -1 corresponding to the decreased dose ('Down') and +1 to the increased dose ('Up').

We did not include some of the categorical variables such as admission type or diagnosis in the definition of the data space, since they contained hundreds of different values, and we used them rather as annotations to be visualized on top of the constructed tree. 2.3 % of records corresponding to the elapsed states (hence, without possibility of readmission) have been excluded from the analysis, similarly to some previous analyses [29].

The resulting encoded dataset contained 22 variables (8 numerical, 7 ordinal and 7 binary). We performed the data preprocessing similar to the one done in the infarction datasets, with imputing the missing values in the 'A1Cresult_value' column and with further application of optimal scaling to ordinal values (see the corresponding Jupyter notebook at https://github.com/auranic/ClinTrajan/). The dimensionality of the dataset was reduced to 6 as it was the consensus value resulted from application of several methods of intrinsic dimension estimation (see Supplementary Figure 1), excluding outlying measurements.

The principal tree algorithm was applied with the same parameters as in the previous section. The construction of the principal tree with 50 nodes for the 6-dimensional dataset with 99343 data points took approximately 400 seconds on an ordinary laptop. The principal tree explained 64% of the total variance in contrast to 47% of the first two principal components, with 4 PCs needed to explain the same percentage of variance as the principal tree. The tree contained 8 branching points (see Figure 6,A) with one forth-order star. The principal tree-based data layout was used to visualize the values of data space variables (Figure 6,A), and some other variables from the annotation data (Figure 6,B,D), some of which did not participate in determining the structure of the principal tree.

As a root node in this case, we selected the middle node of one of the internal segments of the principal tree (segment #3 in Figure 6,A), which was characterized by the shortest times spent in the hospital, smallest number of all procedures, no history of inpatient stays or emergency calls in the preceding year, normal predicted (not measured) value of HbA1C, absence of any medication. Therefore, this area of the principal tree was considered as corresponding to quasi-normal state in terms of diabetes treatment.

Starting from this root node, the structure of the principal tree allowed us to define 8 distinct clinical trajectories. We focused on two of them, depicted in Figure 5,C as solid and dashed lines, together with pseudo-time dependence of several selected clinical variables. One of these two trajectories was the only one associated to the high readmission incidence, increasing with pseudo-time. It did not correspond, however, to the longest stays in hospital, which was the feature of the second considered clinical trajectory. Therefore, we will designate these clinical trajectories as 'readmission-associated' and 'long stay-associated'. Not surprisingly, the readmission-associated trajectory was characterized by increasing number of inpatient and outpatient stays as well as the increasing number of emergency visits in the preceding year. This association must be interpreted by clinicists in order to attribute it either to objective clinical patient state requiring frequent return to the hospital or a psychological pattern of behaviour. In favour of the objective cause, one can notice that the

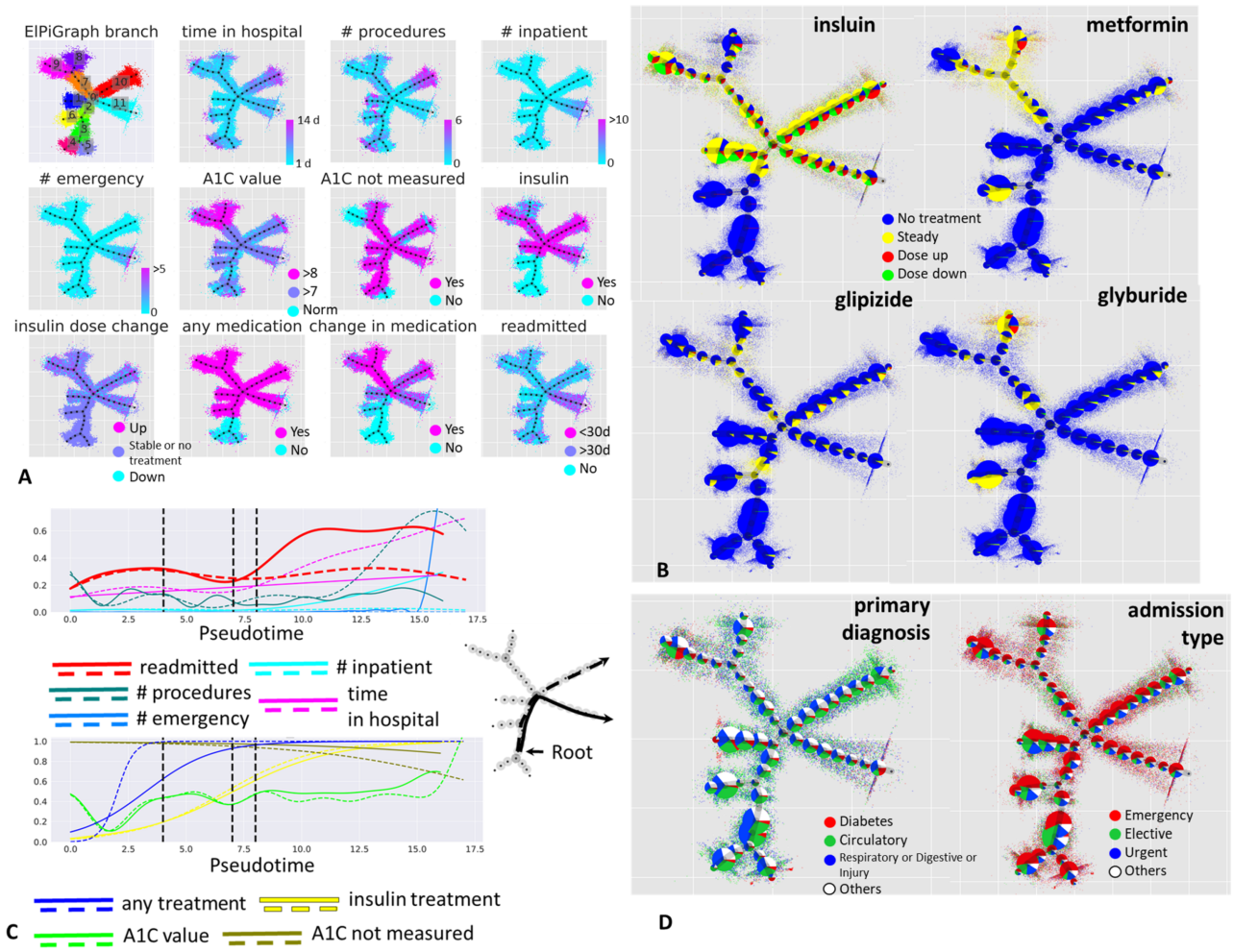

Figure 6. Analysis of clinical trajectories in large-scale diabetes dataset. A) visualization of various clinical variables on top of the metro map layout of the principal tree. Partitioning the data accordingly to the principal tree segments is also shown in the top left corner. B) Visualization of four categorical variables related to the administered drug treatments and their dose changes. Here, the data points are shown in semi-transparent background, while on top of each graph node the relative proportions of the associated (the closest) data points are shown as a pie-chart. The size of the pie-charts are proportional to the number of points associated to each node. C) Pseudotemporal dynamics of clinical variables correlated to readmission- and long stay-associated clinical trajectories (shown as solid and dashed lines on the left). D) Visualization of the data table fields not participating in the construction of the principal tree, "primary diagnosis" on the left and "admission type" on the right.

readmission-associated trajectory contains different spectrum of primary diagnoses compared to the long stay-associated trajectory where the primary diagnoses related to circulatory system are dominating (Figure 5,D,left). We can also notice the elective hospitalizations were increasingly more frequent for the long stay-associated trajectory, while the pseudo-time of the readmission-associated trajectory correlates with increasing probability of admission by emergency (Figure 5,D, right).

Readmission-associated trajectory in this analysis can be considered as undesirable clinical scenario, the main source of burden on the medical system with respect to the diabetes disease. By the trajectory-based analysis we confirmed previous conclusions from [26] that the readmission-associated trajectory was connected with almost complete absence of the measured HbA1C (Figure 5,C), unlike long stay-related trajectory where up to 40% of patients passed through HbA1C testing at the final pseudotime values. The predicted value of HbA1C along readmission-related trajectory was '>7' (moderate elevation). Both readmission- and long stay-associated trajectories were characterized by administered treatment by insulin, with slightly more metformin indications along the long stay-associated trajectory. Importantly, the long stay-associated clinical trajectory is connected to the earlier, in terms of pseudotime, "any treatment" variable dynamics (Figure 5,C,bottom panel).

Discussion

In this study we considered two rich and large publicly available observational clinical datasets from the two most challenging areas of public health: cardiology and diabetes. Both datasets contain syncronic (related to the moment of staying in the hospital) observations over a relatively large population of patients. Therefore, the traditional unsupervised machine learning approach for treating these data in order to classify clinical states is supposed to be some kind of clustering or manifold learning. We demonstrate that there exists an alternative approach which allowed us to represent these data as pseudo-diachronic, i.e., reflect to some extent the temporal aspects. This opens a possibility to classify not only the states of particular patients but their hypothetical clinical trajectories arriving from the past and projected into the future. This in turn gives a possibility to reason in terms of dynamical disease phenotyping, e.g., classifying clinical states in terms of the disease dynamics type.

Identification of clinical trajectories is made possible due to the use of the branching pseudotime approach, consisting

in modeling the geometry of the dataset as “bouquet" of diverging trajectories, starting from one or several hypothetical quasi-normal, e.g., characterized by the least severe condition, disease states. The progression along a particular clinical trajectory can be quantified in terms of pseudotime, reflecting the abstract accumulated amount of changes in the observed clinical traits. The main requirement for the possibility of such reconstruction is the existence of sufficient number of observations (thousands) such that the individual variations in the clinical states would reveal the major non-linear routes along which they progress in real physical time.

Trajectory analysis from the snapshot data is a widely used approach in modern molecular single cell studies, where the genome-wide measurements of individual cell states are inevitably destructive. Collecting the information about a large number of cell states allows reconstructing the underlying hidden cellular dynamics without following each individual cell in physical time [15]. Dynamic phenotyping of cell states is a rapidly emerging concept in this scientific field [6]. Elastic principal graphs (ElPiGraph) is an established general machine learning method which is widely used for the purpose of reconstructing cellular trajectories from the single cell data, in the form of principal trees or other more or less complex graph topologies [19]. Here we suggest to apply ElPiGraph to quantification of clinical trajectories in large clinical datasets, which requires adapting ElPiGraph to the datasets characterized by mixed data types and presence of non-randomly distributed missing values.

This effort resulted in *ClinTrajan* Python package which can be readily applied in the analysis of clinical datasets containing even millions of observations. In the real life diabetes dataset considered here and containing more than one hundred thousands of observations, the analysis by *ClinTrajan* takes few minutes on an ordinary laptop.

Similar to the cellular trajectories, the reconstructed clinical trajectories do not possess any natural orientation: therefore, orienting them requires expert-based decisions for choosing one or several root nodes in the principal tree. Also, the hypothetical dynamics of patients along the clinical trajectory does not have to be assumed irreversible. Some additional insights about orientation and reversibility can be obtained from a mix of synchronic and diachronic data, where individual patients can be represented not by simple data points but by the more or less longitudinal observations represented by short trajectories. The best practices of using such data from the machine learning perspective remains to be established [10].

It appears interesting to relate the inferred clinical trajectories with the physical time related to the aging of patients. Indeed, the chronological age represents the most basic way to rank the patients in a sequence which can potentially correlate to the clinical state (hence, define a clinical trajectory). However, the relation between ’biological’ and ’chronological’ age remains complex, especially in the pathological context [30]. In our study we exploited the chronological patient age as any other clinical variable, and observed that indeed age correlates to some clinical trajectories but not to the others. Moreover, some clinical trajectories might be characterized by decreasing chronological age, which can be interpreted as aggravating clinical picture specific to younger patients. We can imaging other ways of using the age variable: for example, for learning the structure of the principal tree in a semi-supervised fashion. How to use the chronological age in the most informative way when analyzing both longitudinal and synchronic data remains an open question [30].

Another limitation of the suggested approach is that the clinical trajectories are assumed to be diverging from some initial root state or states. In reality, convergence of clinical trajectories seems to be feasible (as in the case of the cellular trajectories). In this case, the model of the principal tree has to be generalized to some more general graph topologies (e.g., existence of few loops). In case of ElPiGraph method, such modifications are easy to introduce technically: however, introducing graph structures more complex than trees requires careful consideration in order to avoid creating data approximators whose complexity will be comparable to the complexity of the data themselves [31].

Potential implications

Quantification of clinical trajectories represents the first step in using the concept of dynamic clinical phenotyping for diagnostics and prognosis. Predicting the probabilities of future clinical states for a particular patient together with their uncertainties, using the knowledge of clinical trajectories, can be a natural next step for future studies. These approaches can consider clinical trajectories as a coarse-grained reconstruction of the state transition graph for a dynamic system, described by, for example, continuous Markov chain equations. Some methodological ideas can be borrowed from the recent omics data studies [32].

Recapitulating the multi-dimensional geometry of a clinical dataset in terms of clinical trajectories might open possibilities for more efficient applications of other methods, more oriented towards supervised machine learning. For example, it can be potentially used for learning the optimal treatment policy, based on application of reinforcement learning as in [33].

Overall, we believe that introducing trajectory-based methodology in the analysis of seemingly static datasets might change the angle of view on their use for developing prognostic and diagnostic expert systems.

Methods

Implementation of the methodology

The *ClinTrajan* methodology is implemented in Python, packaged and openly available at https://github.com/auranic/ClinTrajan/ together with Jupyter notebooks providing the exact protocols of applying the *ClinTrajan* package to several case studies. The detailed description of *ClinTrajan* functionality is provided from its web-site.

Quantification of mixed type datasets

Quantification of mixed type datasets, i.e. assigning a numerical value for nominal variables is a vast field where many solutions have been suggested [34]. In the *ClinTrajan* package we used several popular ideas adapted to the aim of finding non-linear trajectories in the data.

Firstly, for all non-binary categorical variables we suggest applying the “dummy” encoding (or “one-hot” encoding), e.g. introducing new columns, containing binary values one per each category (one of the categories might be dropped as redundant). Alternatively, if there is a sufficient number of numerical variables, Categorical Principal Component Analysis (CatPCA) can be applied [35, 28].

Secondly, for ordinal variables (including binary ones as particular case) we suggest to use either univariate or multivariate quantification. For univariate quantification we assume that the ordinal values are obtained by binning a ‘latent’ numeric continuous variable possessing the standard normal distribution (zero mean and unit variance), following the approach described in [36]. Let us consider an ordinal variable V

which takes ordered values $v_1 < v_2 \ldots < v_m$, and each v_i value has n_i counts in the dataset. We quantify V by the values

$$x_i = \Phi^{-1}\left(\sum_{j=1}^{i-1} p_j + \frac{p_i}{2}\right), \quad (1)$$

where $p_i = \frac{n_i}{N}$, N is the total number of data points, and $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{x^2}{2}} dx$.

If there exist many ordinal variables in the dataset, one can use multivariate methods for joint quantification of them. One of the most popular approaches is a particular variant of *optimal scaling*, aiming at maximizing the sum of squared pairwise correlations between all variables, including numerical and ordinal ones [34]. *ClinTrajan* package includes its own implementation of this variant of optimal scaling which can be used to quantify ordinal variables in clinical datasets.

The advantage of multivariate ordinal variable quantification with respect to the univariate one in that it can decrease the intrinsic dimensionality of the resulting data point cloud which can be beneficial for further applying of manifold learning methods, including the method of elastic principal graphs (ElPiGraph). The disandvatage of multivariate quantification of ordinal variables consists in necessity to have sufficiently large portion of data table rows without missing values. If this part is small then it might be not possible to quantify certain ordinal levels since they won't be represented in this complete part of the dataset, while this might still be possible with univariate quantification.

Thus, imputing missing values requires quantification of ordinal variables, and multivarite quantification of them requires imputation of missing values. Therefore, in practice we apply a hybrid approach consisting in application of univariate quantification with further imputation of missing values and further application of multivariate quantification using the optimal scaling approach.

Lastly, we suggest to transform all continuous numerical variables to the standard z-scores (i.e., centering and scaling), in order to make them comparable.

Imputing missing values in mixed type datasets

The real-life clinical datasets are almost always only partially complete and contain missing values. Typically, these values are not distributed uniformly across rows and columns of the data matrix but rather form some non-random patterns, which can be even constructively used for the tasks of clinical data analysis [37]. A typical pattern is existence of a column (or a row) containing abnormally large number of missing values. One can define two parameters δ_{row} and δ_{column} as maximally tolerable fraction of missing values in any row or column of the data matrix. The problem of finding the maximum size submatrix satisfying these constraints is not completely trivial but can be approximated by some simple iterative approaches. In practice, the trivial suboptimal solution consists in eliminating columns having the fraction of missing values larger than δ_{column}, and then eliminating the rows having the fraction of missing values larger than δ_{row}.

After constraining the maximum fraction of missing values in the data matrix, one can apply one of the available missing value imputation algorithms (imputers), which can be also classified into univariate and multivariate. For our purposes we advocate the use of multivariate imputers that allow us to avoid having strong data outliers destroying the manifold structure of the dataset. The standard scikit-learn collection provides two types of imputers: nearest neighbors imputation and iterative multivariate one, which can be in principle used for this purpose. In *ClinTrajan* package We add two alternative imputers based on application of Singular Value Decomposition (SVD) of order k. The first one which we will designate as "SVDComplete" is applicable if the number of rows in the data matrix having no missing values is sufficiently large (e.g., not much smaller than 50%). Then the standard SVD or order k is computed on the sub-matrix having only complete rows, and each data vector containing missing values is projected into the closest point of the hyperplane spanned by the first k principal components. The imputed value is then read out from the projected vector. For ordinal and binary variables, the imputed value can be additionally rounded to the closest discrete numerical value, in order to avoid "fuzzy values" which do not correspond to any initial nominal value. The mutually exclusivity of binary variables encoding the categorical fields can be also taken into the account. The second SVD-based imputer is called "SVDFull" and it is based on computing SVD of order k for the full matrix with missing values; for example, using the method suggested in [38, 18]. After computing the principal vectors, the imputation is performed as in "SVDComplete" imputer. Choice of k can be made either through applying cross-validation, or using a simple heuristics consisting in setting k to the value of the intrinsic dimensionality (ID) of the data. ID can be estimated through the application of full-order SVD and analyzing the scree plot, or through a number of more sophisticated approaches [39].

Method of Elastic Principal Graphs (ElPiGraph)

Computing the elastic principal graph

Elastic principal graphs are structured data approximators [17, 40, 41, 18], consisting of nodes connected by edges. The graph nodes are embedded into the space of the data, minimizing the mean squared distance (MSD) to the data points, similarly to the k-means clustering algorithm. However, unlike unstructured k-means, the edges connecting the nodes are used to define the elastic energy term. This term is used to create penalties for edge stretching and bending of segments. To find the optimal graph structure, ElPiGraph uses topological grammar (or, graph grammar) approach and gradient descent-based optimization of the graph topology, in the set of graph topologies which can be generated by a limited number of graph grammar operations.

Elastic principal graph is an undirected graph with a set of nodes $V = \{V_i\}$ and a set of edges $E = \{E^i\}$. The set of nodes V is embedded in the multidimensional space. In order to denote the position of the node in the data space, we will use the notation $\phi(V_j)$, where $\phi(V_j)$ is a map $\phi : V \to R^m$. The optimization algorithm search for such $\phi()$ that the sum of the data approximation term and the graph elastic energy is minimized. The optimization functional is defined as:

$$U^{\Phi}(X, G) = MSD^{\Phi}(X, V) + U_E^{\Phi}(G) + U_R^{\Phi}(G), \quad (2)$$

where

$$MSD^{\Phi}(X, V) = \frac{1}{|X|} \sum_{i=1}^{|X|} \min(||X_i - \phi(V_{P(i)})||^2, R_0^2), \quad (3)$$

$$U_E^{\Phi}(G) = \sum_{E^i} \lambda_{penalized}(E^i) \left(\phi(E^i(0)) - \phi(E^i(1))\right)^2, \quad (4)$$

$$U_R^{\Phi}(G) = \mu \sum_{S^j} \left(\phi(S^j(0)) - \frac{1}{deg(S^j(0))} \sum_{i=1}^{deg(S^j(0))} \phi(S^j(i)) \right)^2, \quad (5)$$

$$\lambda_{penalized}(E^i) = \lambda + \alpha \left(\max(2, deg(E^i(0)), deg(E^i(1))) - 2 \right), \quad (6)$$

where $|V|$ is the number of elements in set V, $X = \{X_i\}, i = 1, \ldots, |X|$ is the set of data points, $E^i(0)$ and $E^i(1)$ denote the two nodes of a graph edge E^i, star S^j is a subgraph with central node $S^j(0)$ and several (more than 1) connected nodes (leaves), $S^j(0), \ldots, S^j(k)$ denote the nodes of a star S^j in the graph (where $S^j(0)$ is the central node, to which all other nodes are connected), $deg(V_i)$ is a function returning the order k of the star with the central node V_i, and $P(i) = \text{argmin}_{j=1,\ldots,|V|} \|X_i - \phi(V_j)\|^2$ is a data point partitioning function associating each data point X_i to the closest graph node $V_{P(i)}$. R_0, λ, μ, and α are parameters having the following meaning: R_0 is the trimming radius such that points further than R_0 from any node do not contribute to the optimization of the graph, λ is the edge stretching elasticity modulo regularizing the total length of the graph edges and making their distribution close to equidistant in the multidimensional space, μ is the star bending elasticity modulo controlling the deviation of the graph stars from harmonic configurations (for any star S^j, if the embedding of its central node coincides with the mean of its leaves embedding, the configuration is considered harmonic). α is a coefficient of penalty for the topological complexity (existence of higher-order branchings) of the resulting graph.

Given a set of data points and a principal graph with nodes embedded into the original data space, a local minimum of $U\phi(X, G)$ can found by applying a splitting-type algorithm. Briefly, at each iteration given the initial guess of ϕ, the partitioning $P(i)$ is computed, and then, given the $P(i)$, $U\phi(X, G)$ is minimized by finding new node positions in the data space. A remarkable feature of ElPigraph is that the $U\phi(X, G)$ minimization problem is quadratic with respect to node coordinates and is reduced to solving a system of linear equations. Importantly, the convergence of this algorithm is proven [18, 42].

Topological grammar rules define a set of possible transformations of the current graph topology. The graph configuration of this set possessing the minimal energy $U\phi(X, G)$ after fitting the candidate graph structures to the data is chosen as the locally best with a given number of nodes. Topological grammars are iteratively applied to the selected graph until given conditions are met (e.g., a fixed number of grammar application, or a given number of nodes is reached, or the required approximation accuracy $MSD^{\Phi}(X, V)$ is achieved). The graph learning process is reminiscent to a gradient descent-based optimization in the space of all possible graph structures achievable by applying a set of topological grammar rules (e.g., in the set of all possible trees).

One of the simplest graph grammars consists of two operations 'add a node to node' and 'bisect an edge', which generates a discrete space of tree-like graphs [19]. The resulting elastic principal graphs are called *elastic principal trees* in this case. In *ClinTranjan* package we currently use only principal trees for quantifying trajectories and pseudotime, even though using more general graph topologies is possible. The advantages of limiting of the graph topology to trees are in that it is easy to layout the structure of the graph on a 2D plane and that any trajectory connecting two nodes of the graph is unique.

The resulting explicit tree structure can be studied independently on the data. Also, an artbitrary vector x – not necessary belonging to the dataset X – can be projected onto the tree and receive a position in its intrinsic geodesic coordinates. The projection is achieved by finding the closest point on the principal graph as a piecewise linear manifold, composed of nodes and edges as linear segments connecting nodes. Therefore, the projection can end up in a node or on an edge. In further we define a projection function $\{p, \epsilon\} = Proj(x, G)$, returning a couple containing the index of the edge which is the closest one to x and the position of the projection from the beginning of the edge $E^p(0)$ as a fraction of the edge length $\epsilon \in [0, 1]$. Therefore, if $\epsilon = 0$ then x is projected into $E^p(0)$ and if $\epsilon = 1$ then the projection is in $E^p(1)$. If $\epsilon \in (0, 1)$ then the projection is on a linear segment, connecting $E^p(0)$ and $E^p(1)$.

A detailed description of ElPiGraph and related elastic principal graph approaches is available elsewhere [19]. The ElPiGraph package implemented in Python is available from https://github.com/sysbio-curie/ElPiGraph.P. Implementations of ElPiGraph in R, Matlab, Java and Scala are also available from https://sysbio-curie.github.io/elpigraph/index.html. When analyzing the clinical datasets, the principal tree inference with ElPiGraph was performed using the following parameters: $R_0 = \infty$, $\alpha = 0.01$, $\mu = 0.1$, $\lambda = 0.05$. After the initial principal tree was constructed, it was pruned and the terminal segments were extended. The pruning consists in eliminating the final terminal segments of the tree containing only one single edge. Extending the terminal segments consists in extrapolating the segment in order to have most of the data points projected on the edges of the terminal segment and not at its terminal node. Both functions are standard principal tree post-processing choices, implemented in ElPiGraph package.

Partitioning (clustering) the data according to the principal graph segments

Embedding a graph to the data space allows us to partition (cluster) the dataset in several natural ways: for example by assigning each datapoint to the closest node or the closest edge. However, these ways do not fully suit our purposes, since they do not reflect the intuition of "trajectory". So it is natural first to decompose the graph itself into linear fragments without branching (we will call them non-branching graph segments or simply segments) and afterwards to cluster the dataset according to the closest segment. This is the idea of the data partitioning used in the paper, and it is described with more details below.

By the branching node in a graph we denote any node with connectivity degree larger than 2 ($deg(V_i) > 2$), and by the leaf or terminal node of the graph we denote a node with degree less than 2 ($deg(V_i) < 2$).

Let us call linear segment (or, simply segment) of a graph such a path which connects one branching node to another branching or a leaf node and which does not contain any other branching nodes. Internal segments connect two branching nodes and the terminal segments connect a branching node to a leaf node (Figure 7,A). As one can see such definition reflects the intuition underlying the notion of the "segment"; we only need to specify several exceptional cases. For a graph which is an isolated cycle (not containing branching or leaf nodes), the whole cycle should be considered as a "segment". The same is true if a graph contains several connected components which are cycles: then all of them are considered as separate "segments". The other exceptional case are nodes of degree zero (isolated nodes) - they also will be considered as separate "segments". These exceptional cases cannot happen for connected principal trees which are the main object of the present study, they were just mentioned for completeness.

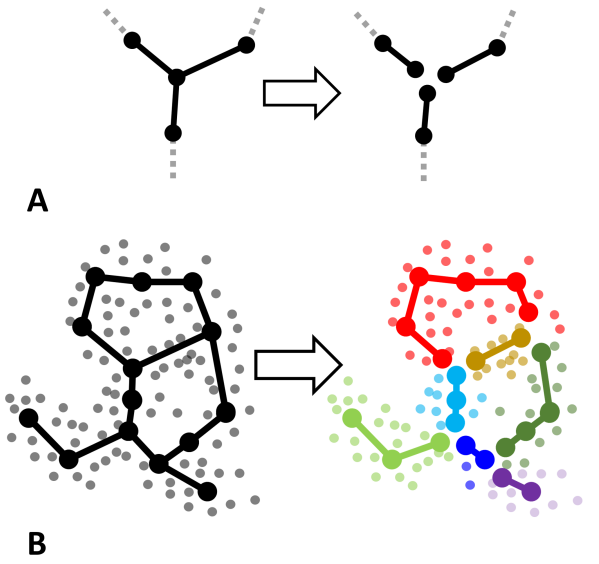

Figure 7. Decomposing a graph into non-branching segments and partitioning the data accordingly to the principal graph segments. A) The principal operation for segmenting the graph: each branching point of the graph (having degree more than two) is multiplied and attached to the end of every edge composing the branching point such that the edges of the graph star are unglued and become disconnected. B) Toy example of a principal graph approximating a cloud of data points (shown in grey), and using its decomposition into non-branching segments for partitioning the data which can be considered a kind of clustering (see text for details).

Any graph can be uniquely split into "segments", which is not difficult to prove, especially for trees.

We coded in Python a version of the depth-first search algorithm to produce a split into segments for an arbitrary graph. Main difference to the classical depth-first search is a storage of visited edges (not only nodes) of the graph to correctly process possible cycles in the graph. The algorithm starts from any branching or leaf node, and walks along edges in depth, joining them to the "current segment" until it meets a branching or a leaf node. Here, the "current segment" is terminated. In case of a leaf node one returns from the recursion, the same for already visited branching node. In case of a new branching node (not visited before) one goes into deeper level of depth-first recursive process. After all edges of a graph are partitioned into segments, one can partition (or, cluster) the dataset accordingly to the closest segment which can be done in two ways. Firstly, choose nearest edge to a given datapoint and associate the point to the segment to which that edge belongs to. However, a simpler approach is much more computationally efficient: calculate distances from datapoints to nodes and choose the segment which contains the nearest node. In case this node belongs to several segments (therefore, it is a branching node), we choose the segment which contains the second nearest node among all nodes which belong to the corresponding segments. If the number of nodes in the graph is large enough then both approaches will produce (almost) identical results (Figure 7,B).

Dimensionality reduction and data visualization using principal graphs

To visualize the principal graph, each datapoint is first associated with the closest ElPiGraph edge in full dimensional space, and the distance to the projection onto the edge is recorded.

We then embed the graph structure in 2D by computing a force-directed layout with the Kamada-Kawai algorithm [43]. Each datapoint is placed orthogonally on a random side of its associated edge, at the distance proportional to the distance to the projection in the initial space. The proportionality constant is called scattering parameter, which is adjusted by a user, or can be optimized in order to preserve, in the best way, the structure of the distances betweeen the datapoints in the initial data space.

Edge widths can also be used to visualize the values of a variable or any function of the variables defined in the nodes of the graph.

Quantifying pseudo-time and extracting trajectories using principal trees

After computing the principal tree, a root node V_{root} has to be defined by the user, accordingly to the application-specific criteria. For example, it can correspond to the node of the graph closest to a set of data points enriched with those having the least of disease severity.

The pseudotime $Pt(x)$ of an arbitrary vector x is defined as the total geodesic distance in the principal tree from V_{root} to the projection $\{p, \epsilon\} = Proj(x, G)$ of x on the graph. Algorithmically, we need to define which node of the edge E^p is the closest to the V_{root} and add the ϵ accordingly, i.e.

$$Pt(x) = \begin{cases} |V_{root} \to E^p(0)| + \epsilon, & \text{if } |V_{root} \to E^p(0)| < |V_{root} \to E^p(1)| \\ |V_{root} \to E^p(0)| - \epsilon, & \text{if } |V_{root} \to E^p(0)| > |V_{root} \to E^p(1)| \end{cases} \tag{7}$$

where $|V_i \to V_j|$ signifies the number of edges (length) of the trajectory $V_i \to V_j$.

Associating class labels and data variables and principal tree segments

Segment labeling of the data points induced by the structure of the principal graph represent a categorical label which can be associated to the dataset variables of various types.

In order to test if there is an association of the tree segments to a categorical variable (including binary as a particular case), we used the standard independence χ^2 test. If the test was significant then we identified those segments which have the most unexpected value of the variable k by considering a simple deviation score:

$$Deviation_k(\text{value } j, \text{segment} i) = \frac{E^i_{kj} - O^i_{kj}}{E^i_{kj}}, \tag{8}$$

where O^i_{kj} is the observed number of data points associated to the segment i having value j of the variable k and E^i_{kj} is the expected number of occurrences of the value j of the variable k, from the standard independence assumption. Positive values of this score correspond to the positive enrichment and negative values for negative enrichment.

In order to test statistical association between tree segments and numerical variable (including ordinal as particular case), we used the standard ANOVA test representing the independent tree segment variable through the standard one hot encoding into a set of binary variables. If the test was significant then we evaluated the significance of each of the segments by looking at the value and the p-values of the generalized linear model coefficients for each segment. Positive values of the coefficients correspond to positive enrichment, and negative for negative enrichment.

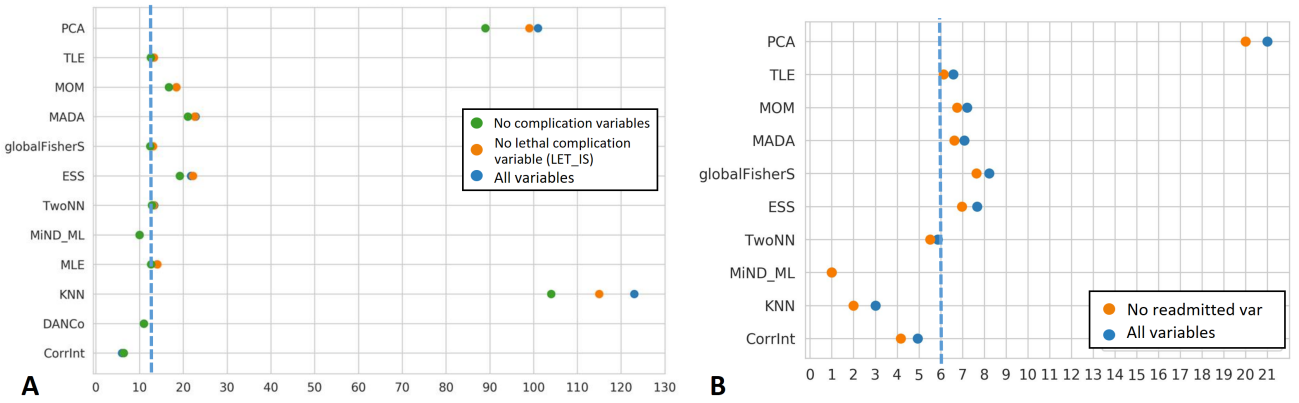

Figure 8. Supplementary Figure 1. Intrinsic dimensionality analysis of clinical datasets used in the study. The PCA-based estimation is defined here as the number of the eigenvalues of the covariance matrix exceeding λ_0/C, where λ_0 is the first (largest) eigenvalue and C is the maximal conditional number of the covariance matrix after dimensionality reduction (here $C = 10$). The computations were performed using the package https://github.com/j-bac/scikit-dimension, where one can find the complete definitions of the methods and the corresponding references.

Associating data variables and trajectories

For computing the score of association between a data variable k and a trajectory, we compute the R^2 score of the regression:

$$x^k = F(Pt(x)), \text{for } x \in X_{V_{root} \to V_j}, \tag{9}$$

where V_j is one of the leaf node in the tree and $Pt(x)$ is the pseudotime value of the data point x computed from (7). For continuous variables, $F()$ can be linear or a non-linear regression (for example, the most popular Gaussian kernel regression). For binary variables, we fit $F()$ by computing the logistic regression. We consider a variable k associated to the trajectory $X_{V_{root} \to V_j}$ if R^2 of the regression problem solution exceeds certain threshold.

Availability of source code and requirements

- Clinical trajectories (ClinTrajan)
- Project home page: https://github.com/auranic/ClinTrajan
- Operating system(s): Platform independent
- Programming language: Python 3.*
- Other requirements: none
- License: LGPL

Availability of supporting data and materials

The data set(s) supporting the results of this article is(are) available in the [repository name] repository, [cite unique persistent identifier].

Supplementary Figures

Supplementary Figure 1. Intrinsic dimensionality analysis of clinical datasets used in the study. The PCA-based estimation is defined here as the number of the eigenvalues of the covariance matrix exceeding λ_0/C, where λ_0 is the first (largest) eigenvalue and C is the maximal conditional number of the covariance matrix after dimensionality reduction (here $C = 10$). The computations were performed using the package https://github.com/j-bac/scikit-dimension, where one can find the complete definitions of the methods and the corresponding references.

Declarations

List of abbreviations

Consent for publication

Not applicable.

Competing Interests

The author(s) declare that they have no competing interests

Funding

This work has been partially supported by the Ministry of Science and Higher Education of the Russian Federation (project No. 14.Y26.31.0022), by Agence Nationale de la Recherche in the program Investissements d'Avenir (project No. ANR-19-P3IA-0001; PRAIRIE 3IA Institute), by European Union's Horizon 2020 program (grant No. 826121, iPC project), by the Association Science et Technologie, the Institut de Recherches Internationales Servier and the doctoral school Frontières de l'Innovation en Recherche et Education Programme Bettencourt.

Author's Contributions

A.Z., S.E.G., E.M.M. and A.N.G. designed the study. S.E.G., E.M.M. and Yu.O. prepared the myocardial infarction database, made it publicly available and advised on its quantification. A.Z., J.B., A.Ch., E.M.M., A.N.G., E.B. developed the methodology, based on application of elastic principal graphs and A.Z., J.B., A.Ch. implemented it in Python. J.B. packaged *ClinTrajan*. A.Z. and S.E.G. applied *ClinTrajan* to the clinical datasets. S.E.G., A.Z., A.N.G. and Yu.O. participated in the interpretation of the results. A.Z. and S.E.G. drafted the manuscript. All authors participated in editing and finalizing the text.

References

1. Jensen AB, Moseley PL, Oprea TI, Ellesøe SG, Eriksson R, Schmock H, et al. Temporal disease trajectories condensed from population-wide registry data covering 6.2 million patients. Nature Communications 2014;5(1):1–10.
2. Westergaard D, Moseley P, Sørup FKH, Baldi P, Brunak S. Population-wide analysis of differences in disease progression patterns in men and women. Nature Communications 2019;10(1):1–14.
3. Moulis G, Lapeyre-Mestre M, Palmaro A, Pugnet G, Montastruc JL, Sailler L. French health insurance databases: What interest for medical research? Rev Med Interne 2015;36(6):411–417.
4. Pinaire J, Azé J, Bringay S, Landais P. Patient healthcare trajectory. An essential monitoring tool: a systematic review. Health Information Science and Systems 2017;5(1):1.
5. Albers DJ, Tabak E, Perotte A, Hripcsak G. Dynamical Phenotyping : Using Temporal Analysis of Clinically Collected Physiologic Data to Stratify Populations. PLoS ONE 2014;9(6):e96443.
6. Ruderman D. The emergence of dynamic phenotyping. Cell Biology and Toxicology 2017;33:507–509.
7. Wang W, Zhu B, Wang X. Dynamic phenotypes : illustrating a single-cell odyssey. Cell Biology and Toxicology 2017;33:423–427.
8. Xu R, Wunsch DC. Clustering. IEEE Press, Piscataway, NJ; 2008.
9. Jung T, Wickrama KAS. An Introduction to Latent Class Growth Analysis and Growth Mixture Modeling. Social and Personality Psychology Compass 2008;2(1):302–317.
10. Nagin DS, Odgers CL. Group-Based Trajectory Modeling in Clinical Research. Annual Review of Clinical Psychology 2010;(6):109–138.
11. Rizopoulos D. Dynamic Predictions and Prospective Accuracy in Joint Models for Longitudinal and Time-to-Event Data. Biometrics 2011;67(3):819–829. https://www.jstor.org/stable/41242530.
12. Schulam P, Wigley F, Saria S. Clustering longitudinal clinical marker trajectories from electronic health data: Applications to phenotyping and endotype discovery. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence; 2015. p. 2956–2964.
13. Schulam P, Arora R. Disease trajectory maps. In: Proceedings of the Thirtieth Conference on Neural Information Processing Systems; 2016. .
14. Banaee H, Ahmed MU, Loutfi A. Data mining for wearable sensors in health monitoring systems: A review of recent trends and challenges. Sensors (Basel) 2013;13(12):17472–17500.
15. Chen H, Albergante L, Hsu JY, Lareau CA, Lo Bosco G, Guan J, et al. Single-cell trajectories reconstruction, exploration and mapping of omics data with STREAM. Nature Communications 2019;10(1903).
16. Saelens W, Cannoodt R, Todorov H, Saeys Y. A comparison of single-cell trajectory inference methods. Nature Biotechnology 2019;37(5):547–554.
17. Gorban AN, Sumner NR, Zinovyev AY. Topological grammars for data approximation. Applied Mathematics Letters 2007;20(4):382 – 386. http://www.sciencedirect.com/science/article/pii/S0893965906001856.
18. Gorban AN, Zinovyev AY. Principal Graphs and Manifolds. In: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques (ed. E.Olivas) Information Science Reference, Hershey, PA; 2008.http://arxiv.org/abs/0809.0490{%}0Ahttp://dx.doi.org/10.4018/978-1-60566-766-9.
19. Albergante L, Mirkes E, Bac J, Chen H, Martin A, Faure L, et al. Robust and scalable learning of complex intrinsic dataset geometry via ElPiGraph. Entropy 2020;22(3):296.
20. Parra RG, Papadopoulos N, Ahumada-Arranz L, Kholtei JE, Mottelson N, Horokhovsky Y, et al. Reconstructing complex lineage trees from scRNA-seq data using MERLoT. Nucleic Acids Research 2019;47(17):8961–8974. https://academic.oup.com/nar/article/47/17/8961/5552070.
21. Marso SP, Griffin BP, Topol EJ, editors. Manual of Cardiovascular Medicine. Lippincott Williams & Wilkins, Philadelphia, Pennsylvania, US; 1999.
22. Golovenkin SE, Gorban AN, Mirkes EM, Shulman V, Rossiev DA, Shesternya DA, et al., Myocardial infarction complications Database. Dataset; 2020. https://doi.org/10.25392/leicester.data.12045261.v2.
23. Gorban AN, Rossiev DA, Butakova EV, Gilev SE, Golovenkin SE, Dogadin SA, et al. Medical and Physiological Applications of MultiNeuron Neural Simulator. In: International Neural Network Society Annual Meeting; Lawrence Erlbaum Associates, vol. 1; 1995. p. 170–175. https://arxiv.org/abs/q-bio/0411034.
24. Zinovyev AY. Visualization of Multidimensional data [in Russian]. Krasnoyarsk, Russia: Krasnoyarsk State Technical Universtity; 2001.
25. Potluri R, Drozdov I, Carter P, Sarma J. Big data and cardiology: time for mass analytics? Eur Med J 2016;1(2):15–17.
26. Strack B, Deshazo J, Gennings C, Olmo Ortiz JL, Ventura S, Cios K, et al. Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records. BioMed research international 2014;2014:781670.
27. Gorban AN, Sumner NR, Zinovyev AY. Beyond the concept of manifolds: Principal trees, metro maps, and elastic cubic complexes. In: Principal manifolds for data visualization and dimension reduction (eds. Gorban A.N, Kegl, B., Wunsch D., Zinovyev A.) Lecture Notes in Computational Science and Engineering, Springer; 2008.p. 219–237.
28. Casacci S, Pareto A. Methods for quantifying ordinal variables: a comparative study. Quality and Quantity 2015;49(5):1859–1872. http://dx.doi.org/10.1007/s11135-014-0063-2.
29. Long A, Using Machine Learning to Predict Hospital Readmission for Patients with Diabetes with Scikit-Learn; 2018. https://towardsdatascience.com/predicting-hospital-readmission-for-patients-with-diabetes-using-sci
30. Whitwell HJ, Bacalini MG, Blyuss O, Chen S, Garagnani P, Gordleeva SY, et al. The Human Body as a Super Network: Digital Methods to Analyze the Propagation of Aging. Frontiers in Aging Neuroscience 2020;12:136.
31. Zinovyev A, Mirkes E. Data complexity measured by principal graphs. Computers and Mathematics with Applications 2013;65(10):1471–1482.
32. Setty M, Kiseliovas V, Levine J, Gayoso A, Mazutis L, Pe'er D. Characterization of cell fate probabilities in single-cell data with Palantir. Nature Biotechnology 2019;37:451–460.
33. Saria S. Individualized sepsis treatment using reinforcement learning. Nature Medicine 2018;24:1641–1642.
34. Young FW. Quantitative analysis of qualitative data. Psychometrika 1981;46:357–388.
35. Linting M, Van Der Kooij A. Nonlinear principal components analysis with CATPCA: A tutorial. Journal of Personality Assessment 2012;94(1):12–25.
36. Fehrman E, Egan V, Gorban AN, Levesley J, Mirkes EM, Muhammad A. Personality Traits and Drug Consumption: a story told by data. Springer Berlin / Heidelberg; 2019.
37. Mirkes EM, Coats TJ, Levesley J, Gorban AN. Handling missing data in large healthcare dataset: A case study of unknown trauma outcomes. Computers in Biology and

Medicine 2016;75:203–216. http://www.sciencedirect.com/science/article/pii/S0010482516301421.
38. Dergachev VA, Gorban AN, Rossiev AA, Karimova LM, Kuandykov EB, Makarenko NG, et al. The Filling of Gaps in Geophysical Time Series by Artificial Neural Networks. Radiocarbon 2001;43(2A):365–371.
39. Albergante L, Bac J, Zinovyev A. Estimating the effective dimension of large biological datasets using Fisher separability analysis. In: Proceedings of the International Joint Conference on Neural Networks; 2019. .
40. Gorban A, Kégl B, Wunch D, Zinovyev A, editors. Principal Manifolds for Data Visualisation and Dimension Reduction. Lecture notes in Computational Science and Engineering, Springer, Berlin; 2008.
41. Gorban AN, Zinovyev A. Principal manifolds and graphs in practice: from molecular biology to dynamical systems. International Journal of Neural Systems 2010;20(3):219–232. http://arxiv.org/abs/1001.1122.
42. Gorban AN, Mirkes E, Zinovyev AY. Robust principal graphs for data approximation. Archives of Data Science 2017;2(1):1:16.
43. Kamada T, Kawai S, et al. An algorithm for drawing general undirected graphs. Information processing letters 1989;31(1):7–15.