UNSUPERVISED FEATURES LEARNING FOR SAMPLED VECTOR FIELDS

MATEUSZ JUDA

Abstract. In this paper we introduce a new approach to computing hidden features of sampled vector fields. The basic idea is to convert the vector field data to a graph structure and use tools designed for automatic, unsupervised analysis of graphs. Using a few data sets we show that the collected features of the vector fields are correlated with the dynamics known for analytic models which generate the data. In particular the method may be useful in analysis of data sets where the analytic model is poorly understood or not known.

1. Introduction

Continuous mathematical models are useful to analyze and draw conclusions about complicated physical systems, where values of system states are assumed to be real numbers. However, nowadays we have countless possibilities of data collection, so scientific and industrial challenges are mostly data driven. We have only finite amount of information, so models should be well-fitted to the observed data and also adapt properly to new, previously unseen data. Usually it means that we have to create highly parameterized models in an automatic way.

We propose a new method for automatic modeling of vector field data sets. In particular, our method takes as an input a finite collection of vectors and creates a low dimensional description of the data. The description encodes features of the observed vector field and allows us to further analyze the physical system using a smaller amount of information. We only require a point cloud with vectors attached at each point. There is no need to manually model the system, e.g. via differential equations. Thanks to this we can analyze data generated by processes for which it is hard to create a traditional model, for instance magnetic field on the Sun surface [27].

The goal of this paper is to present the new method and to validate it on well understood data sets. We show examples based mostly, but not only, on simulated dynamical systems. We compare the automatically learned features with well known, analytically calculated, properties of the systems. We also extend the analysis to a series of dynamical systems, either given as parameterized equations or vector fields constructed from data. The presented examples justify the usefulness of the methods. The learned features may be used as an input to other machine learning tasks, where the original data may be viewed as vector fields, e.g. solar flares predictions, classification of data from particle image velocimetry (PIV),

The author is supported by the Polish National Science Center under Maestro Grant 2014/14/A/ST1/00453.

turbulences detection. Details of the applications are beyond the scope of this paper, are in progress, and are going to be presented elsewhere.

The paper is organized as follows: Section 1 contains the introduction, a description of related work, and it recalls the theoretical model. In Section 2 we describe the problem and model it as the problem of word embedding in Natural Language Processing (NLP). In Section 3 we describe the details of the algorithm. Finally, Section 4 contains examples and Section 5 finalizes the paper.

1.1. **Related work.** The computation of global dynamical information is a challenging problem for applications. Characterization of the global dynamical structure and its changes are fundamental in many disciplines, e.g. computational biology and engineering. Combinatorial dynamics and computational topology are powerful tools for the task. In particular, the tools are useful in classification of the qualitative properties of parameterized models. The database approach [2] helps to understand models where it is difficult to measure parameters. For sampled parameters an outer approximation of the dynamics is computed using rigorous numerical methods. The dynamics is then represented as a directed graph and classified using the Morse decomposition and Conley indices. The method is useful when we know the parameterized dynamical system model. Then, by rigorous simulations, we can find all possible dynamics and match them with collected data. Recent application of the method is applied to understand the global dynamics of gene regulatory networks [9, 15].

In experiments very often one quantity is measured - a time series - while in numerical simulations the full state of a system is an observable. A powerful tool to reason about the unknown system using only a partial information is the Takens embedding theorem [29]. The theorem has been used in [5, 22], together with the Conley index theory of multivalued maps, to identify dynamics from sampled data.

A different approach to sampled dynamical systems is presented in [11]. The method analyze data points given by an unknown self-map. The results presented in this paper suggest that the persistent homology of eigenspaces picks up the important dynamics from small data sample.

Problems similar to our work are considered in [28] where the input vector field is converted to a piecewise constant vector field. Then the Morse decomposition is computed for trajectories obtained using geometrical rules. A main difference, when compared to our work, is that the method is limited to triangulated manifold surfaces and considers only the Morse decomposition while we show a more general framework.

1.2. **A combinatorial dynamical system from a sampled vector field.** In this section we recall some definitions and results from [10, 23]. As mentioned in [10] also here the presented results may be generalized to arbitrary finite T_0 topological spaces [4]. From the viewpoint of applications, a finite topological space may be a collection of cells of a simplicial, cubical, or general cellular complex approximating a cloud of sampled points. For the sake of this paper we use simplicial complexes only.

Let K be a finite simplicial complex, either a geometric simplicial complex in $\mathbb{R}^d$ or an abstract simplicial complex (see [24, Section 1.2, 1.3]). We consider K as a poset $(K, \preceq)$

with $\sigma \preceq \tau$ if and only if σ is a face of τ (also phrased τ is a *coface* of σ). The poset structure of K provides, via the Alexandrov Theorem [1], a T_0 topology on K. We say that $A \subseteq K$ is *orderly convex* if for any $\sigma_1, \sigma_2 \in A$ and $\tau \in K$ the relations $\sigma_1 \preceq \tau$ and $\tau \preceq \sigma_2$ imply $\tau \in A$. We remark that orderly convex sets in K may be characterized in the language of the associated Alexandrov topology. Namely, $A \subseteq K$ is orderly convex if and only if it is locally closed (see [12, Sec. 2.7.1, p. 112]) in the Alexandrov topology $\mathcal{T}_K$.

We define a *multivector* as an orderly convex subset of K and a *combinatorial multivector field* (`cmf` in short) on K as a partition $\mathcal{V}$ of K into multivectors. A multivector is *critical* if its Lefschetz homology is non zero (for details see [23]). The definition encompass the combinatorial vector field of Forman [13, 14] as a special case.

Given a `cmf` $\mathcal{V}$, we denote by $[\sigma]_{\mathcal{V}}$ the unique V in $\mathcal{V}$ such that $\sigma \in V$. We associate with $\mathcal{V}$ a *combinatorial dynamical system* $F_{\mathcal{V}} : K \multimap K$ (a multi-map) given by $F_{\mathcal{V}}(\sigma) := \operatorname{cl}\sigma \cup [\sigma]_{\mathcal{V}}$, where $\operatorname{cl}\sigma$ is the *clousure* of σ defined by $\operatorname{cl}\sigma := \{\tau \mid \tau \preceq \sigma\}$.

When the dynamics which is sampled constitutes of a flow, that is, when time is continuous as in the case of a differential equation, the sampled data may consist of a cloud of points with a vector attached to every point. We call this a *cloud of vectors*. In this case the construction of a combinatorial dynamical system is done in two steps. In the first step, the cloud of vectors is transformed into a combinatorial vector field in the sense of Forman [13, 14] or its generalized version of a combinatorial multivector field [23]. In the second step, the combinatorial multivector field is transformed into a combinatorial dynamical system. Intuitively, cells of K fill the gaps between the points from the input data. The multivectors reflect the vector field behavior from lower to higher dimensional cells. The value of $F_{\mathcal{V}}(\sigma)$ allows us to travel between the points. We can use either the fillings $[\sigma]_{\mathcal{V}}$ or jump into an area influenced by another vector using $\operatorname{cl}\sigma$.

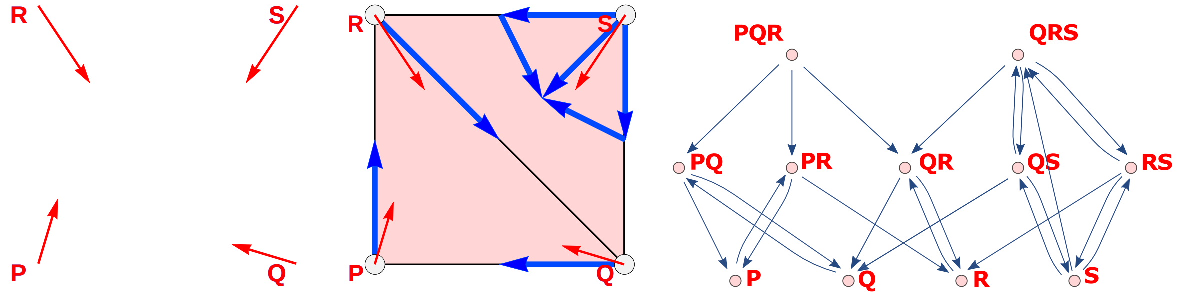


FIGURE 1. Left: A cloud of vectors. Middle: A possible combinatorial multivector field representation of the cloud of vectors. Right: The associated combinatorial dynamical system represented as a digraph.

Figure 1(left) recall a toy example of a cloud of vectors [10]. It consists of four vectors marked red at four points **P**, **Q**, **R**, **S**. One of possible geometric simplicial complexes with vertices at points **P**, **Q**, **R**, **S** is the simplicial complex K consisting of triangles **PQR**, **QRS** and its faces. A possible multivector field $\mathcal{V}$ on K constructed from the cloud of vectors consists of multivectors $\{\mathbf{P}, \mathbf{PR}\}$, $\{\mathbf{R}, \mathbf{QR}\}$, $\{\mathbf{Q}, \mathbf{PQ}\}$, $\{\mathbf{PQR}\}$,$\{\mathbf{S}, \mathbf{RS}, \mathbf{QS}, \mathbf{QRS}\}$. This multivector field is indicated in Figure 1(middle) by blue arrows between centers of mass of simplices. Note that in order to keep the figure legible, only arrows in the direction increasing the dimension are marked. The associated combinatorial dynamical system $F_{\mathcal{V}}$ presented as a digraph is in Figure 1(right). Note that in general K and $\mathcal{V}$ are not uniquely determined by the cloud of vectors. One possible method for constructing combinatorial multivector fields from a cloud of vectors is discussed in [10, Section 7.2].

Given a combinatorial multivector field $\mathcal{V}$, we define a *combinatorial multivector field graph* (`cmf graph` in short), denoted by $G_{\mathcal{V}}$, as a directed graph with the set of vertices $V(G_{\mathcal{V}}) := \mathcal{V}$ and the set of edges $E(G_{\mathcal{V}}) := \{([u]_{\mathcal{V}}, [v]_{\mathcal{V}}) \mid v \in F_{\mathcal{V}}(u)\})\}$. For a graph G by G^T we denote the *transpose* of G, i.e. G^T is a graph, such that $V(G^T) := V(G)$ and $E(G^T) := \{([v]_{\mathcal{V}}, [u]_{\mathcal{V}}) \mid ([u]_{\mathcal{V}}, [v]_{\mathcal{V}}) \in E(G)\}$. We define the *forward distance* (resp. *backward distance*) between two vertices as the number of edges in a shortest path connecting them in $G_{\mathcal{V}}$ (resp. $G^T_{\mathcal{V}}$).

2. Problem description

Our goal is to explore combinatorial multivector field graphs structure. We do it, by finding features of multivectors, as fixed length sequences of real numbers, such that multivectors with a similar local structure in a `cmf graph` have similar features. We also want to extend the features to whole graphs, in such a way that `cmf graph`s representing similar combinatorial dynamical systems have similar features. At this point we skip formal definitions of the similarities mentioned above and we use experimental justifications of the presented methods. A theory of a similarity for combinatorial dynamical systems is not developed yet and the topic is still under research. However, as long as a combinatorial dynamical system $F_{\mathcal{V}}$ reflects the dynamics of the sampled system, tools designed for graphs can be extended to study combinatorial dynamical systems.

Let $\mathcal{G}$ be a collection of labeled graphs, namely each vertex v of a graph $G \in \mathcal{G}$ has a label l_v from some set of labels $L \subseteq \mathbb{N}$. In Section 3.1 we propose an assignment of labels which uses topological properties of the multivectors. Our main goal is to *learn* latent representations of labels of a `cmf graph` $G_{\mathcal{V}}$. Namely, we want to find an *encoding function* $\Phi : L \to \mathbb{R}^D$, for a fixed dimension D, such that the codes $\Phi(l_u)$ and $\Phi(l_v)$ are close whenever there is a similar local structure in $G_{\mathcal{V}}$ around two multivectors u and v. We want to extend the encoding function to graphs, namely for two `cmf` $\mathcal{V}_1$ and $\mathcal{V}_2$ the codes of $\Phi(G_{\mathcal{V}_1})$ and $\Phi(G_{\mathcal{V}_2})$ are close whenever the dynamical systems which generates $\mathcal{V}_1$ and $\mathcal{V}_2$ are similar. The values of Φ should depend on qualitative local features of the vertices and do not depend on the vertices enumeration.

Let $W_v = \{w_1, w_2, \ldots, w_k\} \subseteq G$ be a *random walk* rooted at $v \in G \in \mathcal{G}$, i.e. $v = w_1$ and w_{i+1} is a randomly selected neighbor of w_i. We explore the graph $\mathcal{G}$ using the *DeepWalk* technique, namely short random walks, as described in [25]. The idea behind the approach is to generalize Natural Language Processing (NLP) methods to explore graphs. In this setting we treat labels of vertices as words and the random walks as sentences in an artificial language. Afterwards, we use the *Continuous Skip-gram Model* [19] to analyze the text documents structure and to find Φ, such that it minimize the following log probability:

$$\underset{\Phi}{\text{minimize}} - \log P(\{l_{w_{i-w}}, \dots, l_{w_{i+w}}\} \setminus \{l_{w_i}\} \mid \Phi(l_{w_i})), \tag{1}$$

where w is a parameter called the *window size*. As we can see in (3) the goal is to predict context based on a word without taking into account the order of words. A relaxation scheme described in [19, 20] provides efficient algorithms to compute the latent representation of words, the encoding Φ.

3. Algorithm

In the context of this paper we assume a family $\mathfrak{V}$ of combinatorial multivector fields is given. Having a `cmf` $\mathcal{V} \in \mathfrak{V}$ we transform it to the NLP data using the following steps:

(1) assign labels (words) to multivectors,
(2) using short random walks encode the structure of $\mathcal{V}$ as a text document,
(3) using NLP techniques analyze the document and extract information about the multivector fields.

Before we present detailed description of the steps we want to recall a few programming tools.

For a set $A \subseteq \mathbb{N}$ by sorted(A) we mean a sequence $\{a_1, a_2, \dots, a_n\}$ such that a_i in A and $a_i \leq a_j$ for each $i < j$. We recall that a *hash function* is any function $\mathfrak{h}$ that projects data of an arbitrary size to a value from a set with a fixed number of members [8, 18]. A *good* hash function satisfies the following properties: 1) it is fast to compute; 2) it minimize *collisions*, i.e. duplication of the function values. In practice programming languages (or additional libraries) implement a hash function for each built-in data type. For a user-defined data type a hash function may be easily defined using hashes of the data type members, e.g. for a pair of numbers (a, b) we may define $\mathfrak{h}((a, b)) := \mathfrak{h}(a) \,\mathbf{xor}\, \mathfrak{h}(b)$, where **xor** is the bitwise exclusive OR operation. To simplify the notation we use the symbol $\mathfrak{h}$ regardless of the function domain. We assume that a good hash function $\mathfrak{h}$ with 64-bits values is given for the following data types: natural numbers, tuples of natural numbers, list of natural numbers, and lists of lists of natural numbers.

3.1. **Topological vocabulary.**

The NLP procedure requires a vocabulary in order to assign labels to the vertices of a graph. We construct labels which grasp some local, topological properties of the vertices in the vector field. Intuitively, we obtain a labeling which is universal, i.e. can be computed using only a multivector and its neighbors and does not depend on the global dynamics. It is only an example of many possible labelings.

For a multivector $v \in \mathcal{V}$, that is a vertex in $G_{\mathcal{V}}$, we first define the *label of v at level (0,0)*, denoted by $\mathcal{L}^{0,0}(v)$, in the following way:

$$\mathcal{L}^{0,0}(v) := \mathfrak{h}(\max_{\sigma \in v} \dim \sigma, |v|, \chi(v)),$$

where $\dim\sigma$ denotes the dimension of a cell σ, $|v|$ stands for the cardinality of v, and $\chi(v)$ is the Euler characteristic of v (i.e. $\sum_{\sigma\in v}(-1)^{\dim\sigma}$). We use the $\mathfrak{h}$ function in the above definition to obtain a number as the label of v. Formally it is not required, however it simplified the computations and the notation.

We define the *label of v at level (b,f)* , denoted by $\mathcal{L}^{b,f}(v)$, in the following way:

$$\mathcal{L}^{b,f}(v) := \mathfrak{h}(\mathcal{L}^{0,0}(v), \mathrm{sorted}(\{\mathcal{L}^{0,0}(u) \mid u \in N_f^+(v)\}), \mathrm{sorted}(\{\mathcal{L}^{0,0}(u) \mid u \in N_b^-(v)\})),$$

where $N_f^+(v)$ (resp. $N_b^-(v)$) are sets of vertices in the forward (resp. backward) distance from v not bigger than f (resp. b). We use the sorted function to unify the order of the neighbors in the sets $N_f^+(v)$ and $N_b^-(v)$. We emphasize that the values of $\mathcal{L}^{0,0}$ are in $\mathbb{N}$ because of the hash function, so the input for the sorted function is a set of numbers. Also the value of $\mathcal{L}^{b,f}$ is always a number because of the hash function.

Currently we use NLP methods which do not use the words (labels) structure. It means that we can arbitrary map the tuples to numbers using a reasonable good hashing function $\mathfrak{h}$. However, there are NLP methods which operates on sub-words (n-grams) [7] and more sophisticated labelings may take advantage of a multivector neighborhood structure.

As an example we consider the multivector field $\mathcal{V}$ and its combinatorial dynamical system $F_{\mathcal{V}}$ presented in Figure 1. Figure 2 presents the associated graph on multivectors $G_{\mathcal{V}}$. Table 1 presents step by step calculations of the values of $\mathcal{L}^{1,1}$.

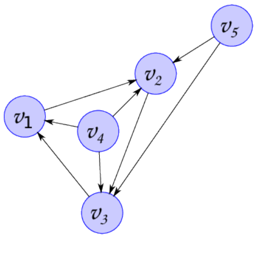


FIGURE 2. $G_{\mathcal{V}}$ graph for the example presented in Figure 1. The nodes represent multivectors in the following order: $v_1 = \{\mathbf{P}, \mathbf{PR}\}$, $v_2 = \{\mathbf{R}, \mathbf{QR}\}$, $v_3 = \{\mathbf{Q}, \mathbf{PQ}\}$, $v_4 = \{\mathbf{PQR}\}$, $v_5 = \{\mathbf{S}, \mathbf{RS}, \mathbf{QS}, \mathbf{QRS}\}$.

v	simplices of v	$\mathcal{L}^{0,0}(v)$	$N_1^+(v)$	$N_1^-(v)$	$\mathcal{L}^{1,1}(v)$
v_1	$\{\mathbf{P}, \mathbf{PR}\}$	$\mathfrak{h}(1,2,0) = 5095$	$\{v_2\}$	$\{v_3, v_4\}$	$\mathfrak{h}($ $\mathfrak{h}(1,2,0),$ $\text{sorted}(\mathfrak{h}(1,2,0)),$ $\text{sorted}(\mathfrak{h}(1,2,0), \mathfrak{h}(2,1,1))$ $) =$ 4901
v_2	$\{\mathbf{R}, \mathbf{QR}\}$	$\mathfrak{h}(1,2,0) = 5095$	$\{v_3\}$	$\{v_1, v_4, v_5\}$	$\mathfrak{h}($ $\mathfrak{h}(1,2,0),$ $\text{sorted}(\mathfrak{h}(1,2,0)),$ $\text{sorted}(\mathfrak{h}(1,2,0), \mathfrak{h}(2,1,1), \mathfrak{h}(2,4,0))$ $) =$ 7355
v_3	$\{\mathbf{Q}, \mathbf{QP}\}$	$\mathfrak{h}(1,2,0) = 5095$	$\{v_1\}$	$\{v_2, v_4, v_5\}$	$\mathfrak{h}($ $\mathfrak{h}(1,2,0),$ $\text{sorted}(\mathfrak{h}(1,2,0)),$ $\text{sorted}(\mathfrak{h}(1,2,0), \mathfrak{h}(2,1,1), \mathfrak{h}(2,4,0))$ $) =$ 7355
v_4	$\{\mathbf{PQR}\}$	$\mathfrak{h}(2,1,1) = 6161$	$\{v_1, v_2, v_3\}$	$\varnothing$	$\mathfrak{h}($ $\mathfrak{h}(2,1,1),$ $\text{sorted}(\mathfrak{h}(1,2,0), \mathfrak{h}(1,2,0), \mathfrak{h}(1,2,0)),$ $\varnothing$ $) =$ 5836
v_5	$\{\mathbf{S}, \mathbf{RS}, \mathbf{QS}, \mathbf{QRS}\}$	$\mathfrak{h}(2,4,0) = 6275$	$\{v_2, v_3\}$	$\varnothing$	$\mathfrak{h}($ $\mathfrak{h}(2,4,0),$ $\text{sorted}(\mathfrak{h}(1,2,0), \mathfrak{h}(1,2,0)),$ $\varnothing$ $) =$ 5382

TABLE 1. Step by step calculation of the labels at level 1 for the example presented in Figure 1 and Figure 2. We use the standard implementation of `hash` and `sorted` functions from the Python programming language (for simplicity the values are taken modulo 10^4). We notice that the labels at level $(0,0)$ cannot distinguish multivectors v_1, v_2, v_2. However, for the labels at level $(1,1)$ the labeling of v_1 is different from the labelings of v_2 and v_3.

3.2. Corpus.

A *d-random multivector walk on $G_{\mathcal{V}}$ from s*, denoted by $\mathcal{W}_{\mathcal{V}}^{d+}(s)$, is a stochastic process with random variables $\{W_1, W_2, \ldots, W_d\}$ such that $W_1 = s$, and W_{i+1} is a vertex chosen at random from the set $N_1^+(W_i) \cup \{W_i\}$. The probability $P(W_{i+1} = u)$ is defined as $p_u / \sum_v p_v$, where

$$p_v = \begin{cases} 1, & \text{if } v \in N_1^+(W_i) \text{ and } v \neq W_i \\ 1, & \text{if } v = W_i \text{ and } W_i \text{ is critical} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In the above definition we can replace N^+ with N^- and reverse the order of the random multivector walk. This way we define a *d-random multivector walk on $G_{\mathcal{V}}$ to t*, denoted by $\mathcal{W}_{\mathcal{V}}^{d-}(t)$.

Let $\mathfrak{V}$ be a collection of combinatorial multivector fields. A (c,d)*-random multivector corpus* of $\mathfrak{V}$ is the stochastic process:

$$\boldsymbol{\mathfrak{W}}_{\mathfrak{V}}^{c,d} := \bigoplus_{\substack{\mathcal{V}\in\mathfrak{V}\\ u\in\mathcal{V}\\ \bullet\in\{+,-\}\\ i\in 1,\ldots,c}} \mathcal{W}_{\mathcal{V}}^{d\bullet}(u),$$

where $\bigoplus$ denotes concatenation of sequences of the random variables. Note that in the above definition we repeat c times the random multivector walk. We define a *multivector* (f,b,c,d)*-corpus* of $\mathfrak{V}$, denoted by $\mathcal{L}^{f,b}\boldsymbol{\mathfrak{W}}_{\mathfrak{V}}^{c,d}$, as $\{\mathcal{L}^{f,b}(W) \mid W \in \boldsymbol{\mathfrak{W}}_{\mathfrak{V}}^{c,d}\}$, where $\mathcal{L}^{f,b}(W)$ is a label of W at level (f,b) (as defined in Section 3.1). We skip the parameters f,b,c,d if they are clear from the context.

We treat a multivector corpus as a text document for which the labels are words and the walks are sentences. Next, we apply NLP methods to the document. We emphasize that the methods we use does not depend on the order of sentences, so we can take any order in the $\bigoplus$ notation. On the other hand, the skip-gram model generates word contexts using the sliding window technique. Hence, the order of words in a sequence is important.

3.3. **Encoding.** The *distributional hypothesis* [17] in linguistics says that it is possible to state a linguistic structure in terms of patterns of co-occurrences, i.e. words with similar meaning occur in the same context. It is the main idea behind representing words as elements of a vector space. Neural network models [6, 7, 19, 20] allow us to find an encoding of words from a large text corpus. The intuition behind the vector space elements is that the distance between similar words is small, and the norm of a word encoding is proportional to its importance in the corpus. In particular, we apply the methods to our artificial multivector corpus introduced in Section 3.2.

Recall that we use the *Continuous Skip-gram Model* [19] to analyze text documents structure and to find Φ, such that it minimize the following log probability:

$$\underset{\Phi}{\text{minimize}} - \log P(\{l_{w_{i-w}}, \ldots, l_{w_{i+w}}\} \setminus \{l_{w_i}\} \mid \Phi(l_{w_i})). \tag{3}$$

The model is a shallow neural network trained to predict words within a range before and after the current word. The main parameters for the algorithm are: *window size* w - number of words around current word, *encoding dimension* D. Intuitively, the parameter w carries examined influence of a word to the meaning of a sentence. The parameter D controls the number of linguistic features learned by the model. From a trained network we extract a D dimensional representation of words, denoted by Φ^D. For natural languages the parameters values typically are: $w = 5$, $D = 300$. In our applications usually much smaller dimension is enough. In the sequel we show, that low dimensional encoding may contain useful information.

In our context $\{l_{w_i}\}_i$ is a sequence of labels in a random walk in a graph $G_{\mathcal{V}}$. It is worth to note that, intuitively, labels represent local structures of the dynamics around multivectors and the random walks represent trajectories. For instance we can consider random walks generated by a constant flow and a spiral flow close to a fixed point. We expect that the encodings of multivectors from the two groups should be distinguishable. It is because the local structures of multivectors on the spiral are different and richer than these on the constant flow. It means that for a randomly chosen label a trained Skip-gram model should be able to predict to which group the label belongs.

The goal of recently developed NLP methods is to find meaningful words encoding. Currently, there is no similar method designed directly for documents. As a workaround a common trick is to use a weighted mean of words encodings as the encoding of a document. In our context, for a multivector field graph $G_{\mathcal{V}}$, we define $\Phi^D_w(G_{\mathcal{V}})$ as $\sum_{v\in V(G_{\mathcal{V}})} w_v \Phi^D(\mathcal{L}^{f,b}(v))$, where f and b are fixed and w_v is a weight of v, e.g. $\frac{1}{|V(G_{\mathcal{V}})|}$ or TFIDF of v (term frequency-inverse document frequency [26]).

3.4. **Implementation details.** We compute the Skip-gram encoding Φ using the FastText [7] library. As we mentioned earlier, we cannot use sub-words (n-grams) in the training phase, so the parameters `minn` and `maxn` are set to 0. For the examples presented in this paper, if not stated otherwise, we set: learning rate to 0.01, size of the context window to 5, dimension of word vectors to 2. The number of epochs used for training depends on the size of the corpus. We use 1000 epoches for small corpuses in Sections 4.1 and 4.2, and 5 epochs for large corpuses in Sections 4.3 and 4.4.

We also noticed that rare words appear close to the boundary of the sampled region. It is because in that area the neighborhood of a multivector depends more on its location than on the vector field structure. We skip such multivectors by checking their distance to the region boundary. The outcome of this simplification is shown in Figure 3c, where the area close to the boundary is white, because labels of the multivectors are not in the corpus, so Φ maps them to zero.

4. Examples

In this section we present the methods in action. First we show toy examples which illustrate a qualitative properties of the encodings. Later we present applications to a series of dynamical systems analysis. Our examples are simple enough to visualize the encodings. However, in real world applications higher values of the encoding dimension (D) and the labeling level ($\mathcal{L}^{b,f}$)may be required. Then, one may need to create a pipeline, where the output of our method is only an intermediate step. We want to keep the paper simple and more complicated applications, e.g. turbulences analysis or solar flare classifications, we are going to present in a sequel paper. The presented examples illustrate that the methods are able to automatically extract meaningful features from dynamical systems.

We begin with a setup for computations. For a product of k intervals $I = [I_1^-, I_1^+] \times \ldots \times [I_k^-, I_k^+] \subseteq \mathbb{R}^k$ and a set of natural numbers $N = \{N_1, \ldots, N_k\} \subseteq \mathbb{N}_+$, we define an (I, N) *regular grid of points*, denoted by $\mathcal{R}(I, N)$, as a set of points $\{(x_1, \ldots, x_k) \in I \mid x_i = I_i^- + \frac{(I_i^+ - I_i^-)j}{N_i}$ for $j \in [0, N_i] \subseteq \mathbb{N}\}$.

In the context of this paper we assume a family $\mathfrak{V}$ of combinatorial multivector fields is given. In order to present the examples we recall a possible way to construct a combinatorial multivector field from a cloud of vectors. Let K be a simplicial complex with vertices in a cloud of points $\{p_i \mid i = 1, 2, \ldots, n\} \subseteq \mathbb{R}^d$ and the associated cloud of vectors $\{\vec{v_i} \mid i = 1, 2, \ldots, n\} \subseteq \mathbb{R}^d$ such that vector $\vec{v_i}$ originates from point p_i. To construct a `cmf` on K one can use the algorithm CVCMF [10, Table 1] (called here CVCMF$v1$). The algorithm requires an angular parameter α. We do not analyze the impact of the parameter here. We also have a parameter-less version of the algorithm (called here CVCMF$v2$). We show results obtained with the old and the new algorithms, however the new one is not published yet.

In the below examples we plot D dimensional encodings of the words obtained with the FastText [7] implementation. On the plots we observe cone-shaped points distribution. It suggest that it should be enough to use the encoding dimension parameter value equal to $D - 1$. However, this is an outcome of the negative sampling training algorithm and this is a phenomenon described in [21].

4.1. **Example: Orbit.** The main goal of this example is to show features extracted by our method on a simple dynamical system. Consider a system given by the following equation:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -y + x(4 - x^2 - y^2), \qquad \frac{\mathrm{d}y}{\mathrm{d}t} = -x + y(4 - x^2 - y^2). \tag{4}$$

In the system phase space we observe a repelling stationary point at $(0, 0)$ and an attracting periodic orbit with center at $(0, 0)$ and radius 2. We can observe this in a combinatorial dynamical system constructed from a finite sample of the vector field using methods described in [10, 23]. To achieve that we build a Delaunay triangulation K of a regular grid $\mathcal{R}([-4, 4] \times [-4, 4], \{30, 30\})$. The triangulation contains 1682 triangles and 5163 simplices. To construct a `cmf` $\mathcal{V}$ of the triangulation and vectors originates from its vertices we use the algorithm CVCMF$v1$ [10, Table 1] with the parameter $\alpha = 0$. The triangulation K with the discretized periodic orbit and the repelling point of the system are presented in Figure 3a.

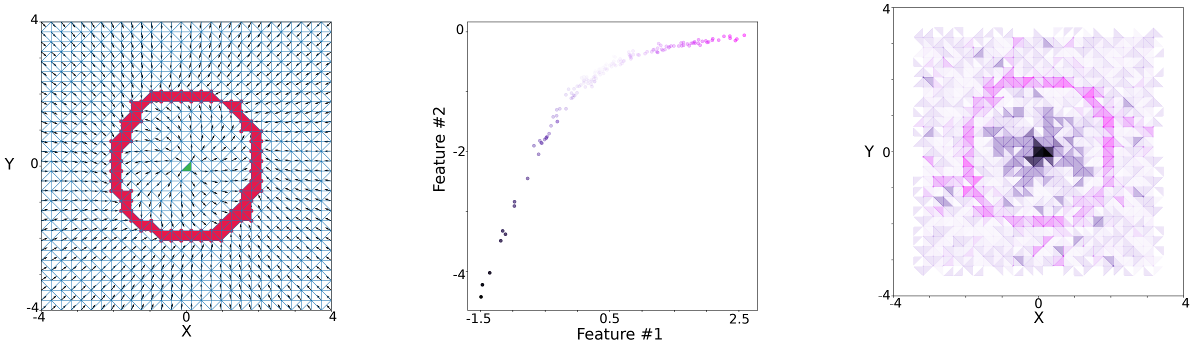
In the next step we label vertices of the graph $G_{\mathcal{V}}$ using labels at level $(1, 1)$, getting 153 distinct labels. Using the labeled graph we generate $(1, 1, 2, 10)$-corpus for which we obtain the Skip-gram encoding Φ with parameters $w = 5$ and $D = 2$. The encodings are plotted in Figure 3b, where each dot represents an unique word in the corpus (label in the graph). We use RGBA color space, where the alpha channel of a point is proportional to its norm.

In Figure 3c, we show colored multivectors, where each multivector v gets color of $\Phi(\mathcal{L}(v))$. We notice that the encoding Φ distinguish three regions of the phase space with following behaviors: close to the the orbit, around the repelling point, outside the orbit.

4.2. **Example: Nested orbits.** Consider a system given by the following equation:

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= -0.3y((x^2 + y^2 - 1) - (x^2 + y^2 - 1)^2) - x(3 - 6(x^2 + y^2 - 1) + (x^2 + y^2 - 1)^2), \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= 0.3x((x^2 + y^2 - 1) - (x^2 + y^2 - 1)^2) - y(3 - 6(x^2 + y^2 - 1) + (x^2 + y^2 - 1)^2). \end{aligned} \tag{5}$$

In the system phase space we observe an attracting stationary point at $(0, 0)$ and two periodic orbits with centers at $(0, 0)$.



(A) Vector field with the periodic orbit and the stationary point marked.

(B) 2D encoding of the multivectors.

(C) Multivectors colored using colors of the encodings.

FIGURE 3. An example of a multivector field encoding for a single orbit and a repelling point.

In this section we show features extracted by our method from a sampled dynamical system given by (5). As in Section 4.1 we build a combinatorial dynamical system using a Delaunay triangulation K of a regular grid $\mathcal{R}([-4,4]\times[-4,4],\{50,50\})$. To construct a `cmf` $\mathcal{V}$ of the triangulation and vectors originates from its vertices we use the algorithm CVCMF$v2$. Here we use the labels at level $(2,2)$ and the encoding dimension $D=3$. With lower values of the parameters we cannot distinguish multivectors on the orbits and the stationary point.

In Figure 4 we show the triangulation K, and the `cmf` $\mathcal{V}$ colored according to the encoding of the multivectors. Afterwards, in Figure 5 we show the influence of w and d parameters. In conclusion, for values $d\geq 20$ and $w\geq 5$ we can observe a clear difference between the encodings for orbits and the stationary point.

4.3. **Example: Prey-predator.** In this section we use the presented methods to analyze a series of dynamical systems. We also show the influence of the labeling levels and the CVCMF algorithm variants. We consider a prey-predator model [16] given in the following form:

$$\frac{\mathrm{d}x}{\mathrm{d}t}=x(1-\frac{x}{\gamma})-\frac{(1-c)xy}{1+\alpha\zeta+x},\qquad \frac{\mathrm{d}y}{\mathrm{d}t}=\frac{\beta[(1-c)x+\zeta]y}{1+\alpha\zeta+x}-\delta y \tag{6}$$

where x and y denote the biomass of prey and predator respectively, and $\alpha,\beta,c,\delta,\gamma,\zeta$ are parameters. We investigate sampled vector spaces obtained from simulations of the model given by (6). In particular, we are interested in qualitative behavior of the model, where $\alpha\in[0,2]$, $c\in[0,0.45]$, $\beta=0.15,\delta=0.08,\gamma=4,\zeta=0.2$. For varying α and c we denote the model by $P(\alpha,c)$.

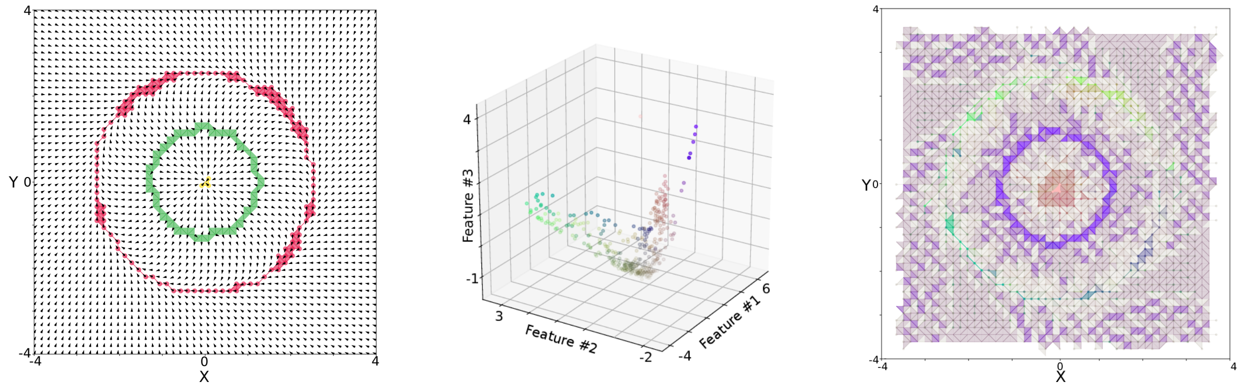


FIGURE 4. A visualization for the system generated by (5) using $(2, 2, 1, 30)$-corpus and $w = 10$: (left) the vector field with two periodic orbits and an attracting point marked; (middle) 3D encodings of the multivectors (RGB colors given by a point coordinates and the alpha channel is proportional to the point norm); (right) the multivectors field colored using the encodings.

In Figure 6 we show a decomposition of the parameter plane (α, c) obtained analytically in [16]. Our goal is to show a correlation between the model dynamics and multivectors encoding Φ.

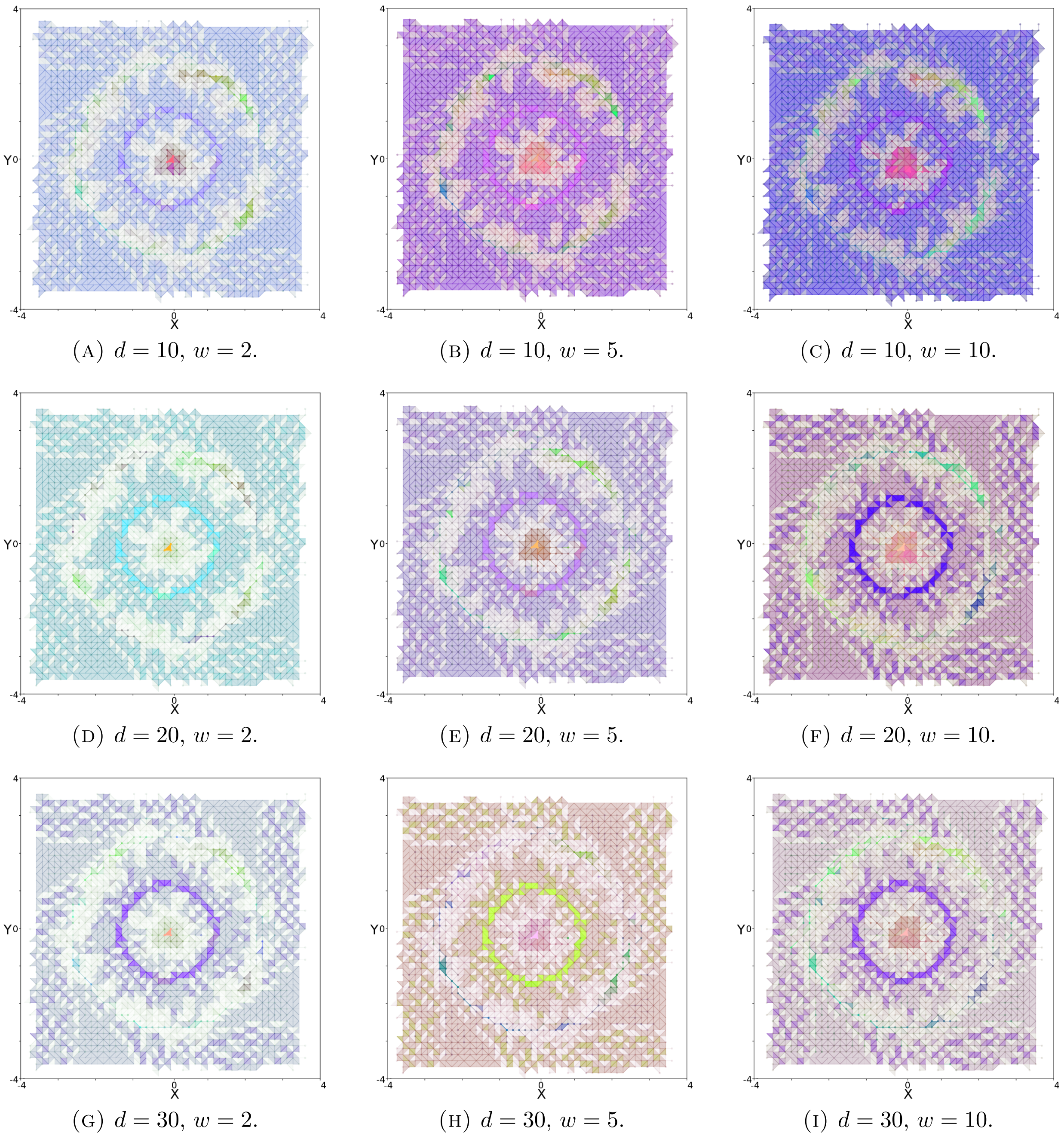
Let K be a Delaunay triangulation of a regular grid $\mathcal{R}([0, 6] \times [0.4, 2.4], \{100, 100\})$. Let $\mathcal{V}^{\theta}_{\alpha,c}$ be a `cmf` of K computed using the CVCMF algorithm (see beginning of Section 4) called with K and vector field sampled from the prey-predator model $P(\alpha, c)$. The value of θ controls the CVCMF algorithm version, namely:

- if $\theta \in [0, 2\pi]$, then we use CVCMF$v1$ with its angular parameter equals to θ,
- if $\theta = \varnothing$, then we use CVCMF$v2$.

Let $\mathfrak{V}^{\theta}$ be a family of combinatorial multivector fields

$$\{\mathcal{V}^{\theta}_{\alpha,c} \mid (\alpha, c) \in \mathcal{R}([0, 2] \times [0, 0.45], \{50, 50\})\}, \text{ for some fixed } \theta.$$

We train the Skip-gram neural network on four corpuses described in Table 2 using the FastText [7] library (see Section 3.4) and obtain 2-dimensional encodings for each of them. For each corpus we present in Figure 7 the encoding of each word and the encoding of each multivector field graph. The pictures suggest that it should be possible to distinguish more types of the prey-predator model dynamics, i.e. the structure of the green subspace does not arrange into a blob of points. We also observe that there is no big difference between corpuses $\mathcal{C}_{II}$ and $\mathcal{C}_{III}$. The best separation between green and yellow points is visible for $\mathcal{C}_{IV}$, which suggest that our new non-parameterized algorithm CVCMF$v2$ finds a better partition of a given complex into multivectors. Further research using machine learning classifiers and higher dimensional encodings are in progress.



(A) $d = 10$, $w = 2$.

(B) $d = 10$, $w = 5$.

(C) $d = 10$, $w = 10$.

(D) $d = 20$, $w = 2$.

(E) $d = 20$, $w = 5$.

(F) $d = 20$, $w = 10$.

(G) $d = 30$, $w = 2$.

(H) $d = 30$, $w = 5$.

(I) $d = 30$, $w = 10$.

FIGURE 5. Visual evaluation of the method for the system generated by (5). We use following settings: $D = 3$, $w \in \{2, 5, 10\}$, $(2, 2, 1, d)$-corpuses for $d \in \{10, 20, 30\}$. For each test case we compute its own Skip-gram model and the 3D encodings. We use a multivector encoding as its RGBA color (see Figure 4). We observe that the triangle with the attracting point at $(0, 0)$ is not clearly visible for $d = 10$, and the outer orbit merges with its neighbors for $w = 2$.

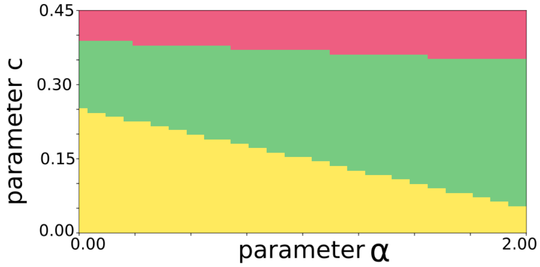


FIGURE 6. Qualitative behavior of the model (6) according to [16]. The parameter plane $(\alpha, c) \subseteq [0,2] \times [0, 0.45]$ is divided into three regions: oscillatory coexistence (yellow), stable coexistence (green), predator extinction (red).

coprus variant	# of distinct words	# of words in the corpus
$\mathcal{C}_I = \mathcal{L}^{1,1}\boldsymbol{\mathfrak{W}}^{5,10}_{\mathfrak{V}^{36°}}$	965	$3877 \cdot 10^6$
$\mathcal{C}_{II} = \mathcal{L}^{2,2}\boldsymbol{\mathfrak{W}}^{5,10}_{\mathfrak{V}^{36°}}$	7979	$3665 \cdot 10^6$
$\mathcal{C}_{III} = \mathcal{L}^{3,3}\boldsymbol{\mathfrak{W}}^{5,10}_{\mathfrak{V}^{36°}}$	28124	$3635 \cdot 10^6$
$\mathcal{C}_{IV} = \mathcal{L}^{3,3}\boldsymbol{\mathfrak{W}}^{5,10}_{\mathfrak{V}^{\varnothing}}$	96589	$4825 \cdot 10^6$

TABLE 2. Multivector corpuses tested for the prey-predator model. Columns: the number of distinct words in a corpus and the total number of words in a corpus.

4.4. **Example: Time series data.** In this section we present a not obvious application of the methods. Namely, we automatically extract features from time series data sets using an approach inspired by [30]. The key idea is to transform a time series data into a sampled vector field.

We assume that values of a time series lie in the range $[-1, 1]$. Otherwise we can re-scale the values, e.g. using min-max scaling. Let $\mathcal{T} = \{t_i\}_{i=0}^{n-1} \subseteq [-1, 1]$ be a time series. In [30] the authors introduced the *Gramian Summation Angular Field* which is a transformation of $\mathcal{T}$ into a gray-scale 2D image. A (i, j)-th pixel color is defined as $\cos(\arccos(t_i) + \arccos(t_j))$. We propose to use similar technique, but to create vectors. Let $K(\mathcal{T})$ be a Delaunay triangulation of a regular grid $\mathcal{R}([0, n] \times [0, n], \{n, n\})$, where n is the length of $\mathcal{T}$. Let $\alpha_{i,j} := \arccos(t_i) + \arccos(t_j)$ for each vertex (i, j) of the grid vertex. At each vertex (i, j) we attach a vector $v_{i,j} := (\cos \alpha_{i,j}, \sin \alpha_{i,j})$. Of course this is an arbitrary choice, but a further investigation of different possibilities is beyond the scope of this paper. In Figure 8 we show a toy example.

As a first validation of the method we use *Symbols* dataset from the UEA & UCR Time Series Classification Repository [3]. The dataset contains a collection of time series representing the motion on the x-axis of a hand drawing of a specified shape. We have 6 classes of shapes and each time series has length 398.

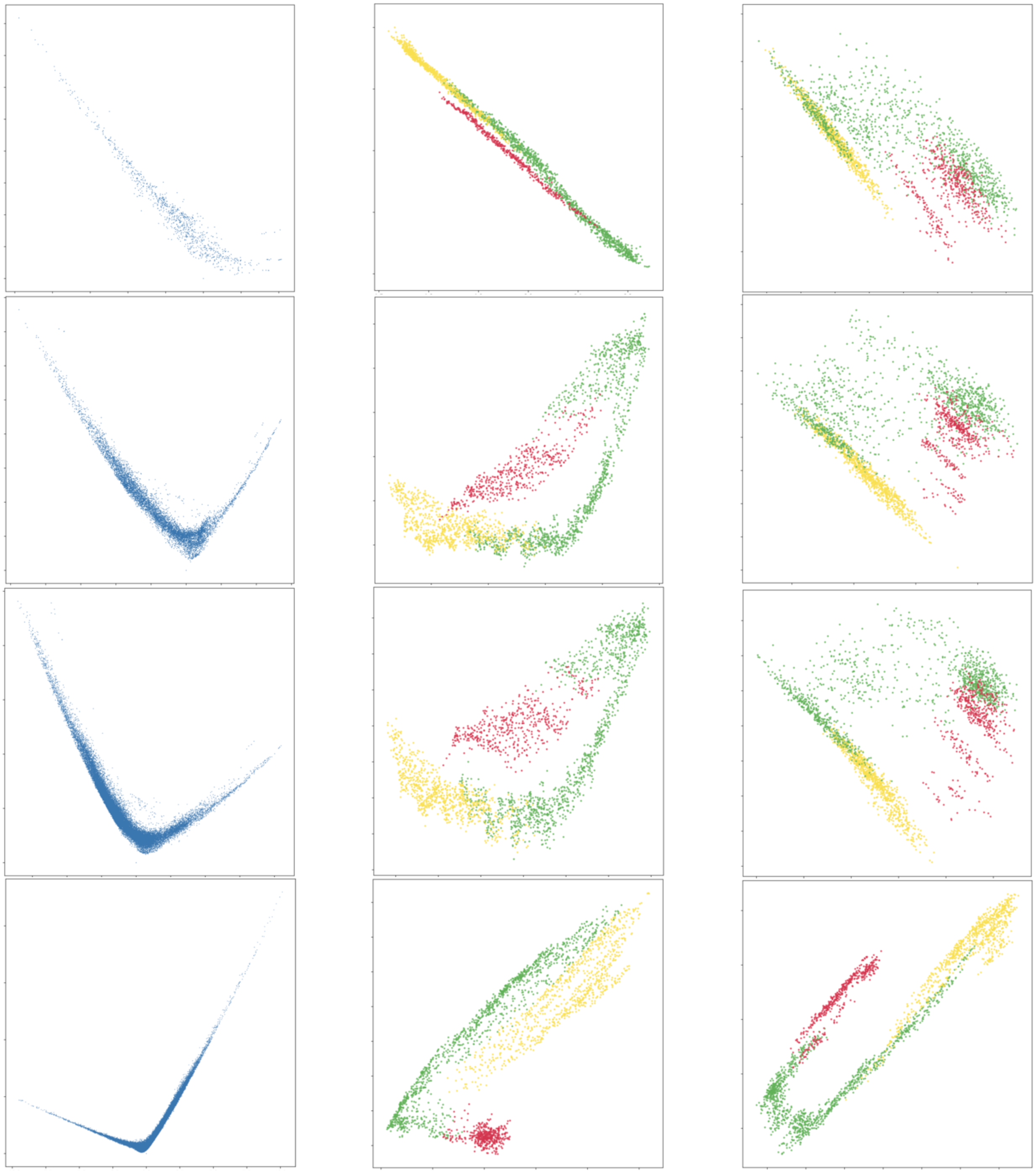


FIGURE 7. 2-dimensional encodings for the prey-predator multivector corpuses. Each row represents a corpus: $\mathcal{C}_I, \mathcal{C}_{II}, \mathcal{C}_{III}, \mathcal{C}_{IV}$ (top-down). First column represents encodings of labels: for each distinct label l in a multivector corpus there is a dot at position $\Phi^2(l)$. Second and third columns represents encodings of graphs: for each $\mathcal{V} = \mathcal{V}^{\theta}_{\alpha,c} \in \mathfrak{V}^{\theta}$ there is a dot at position $\Phi^2_w(G_{\mathcal{V}})$, where w is the mean (second column) or TFIDF (third column) weighting. Colors of the dots correspond to the colors of (α, c) in Figure 6. We are interested in the points distributions, so we skip values on the plots axes.

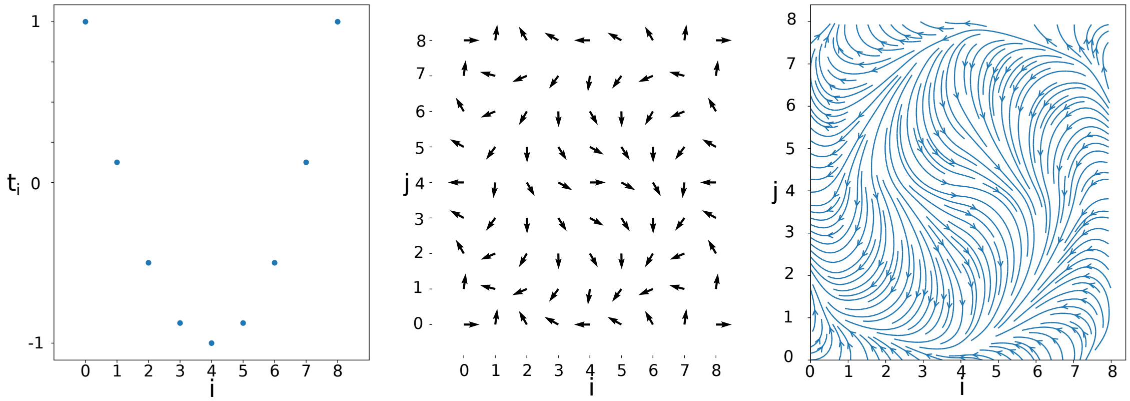


FIGURE 8. Left: min-max scaled time series $\mathcal{T} = \{(i-4)^2\}_{i=0}^{8}$. Middle: on the axes are two copies of the temporal coordinate and on the plot are the (i,j)-vectors of $\mathcal{T}$, for $i,j \in \{1,\ldots,8\}$. Right: a stream plot visualization of the (i,j)-vectors. It is worth to note that the time series data is rather sparse but yet the vector field looks smooth and it is possible to create an expressive stream plot.

The dataset is splitted into training set of size 25 and test set of size 995. In Figure 9 we show a few examples from the dataset. Note, that the presented vector fields do not have critical points and only the curvature of the flows distinguish them.

We use the methods presented in the previous sections to obtain encodings of the vector fields generated from the *Symbols* time series data set. To train the Skip-gram model we use only the training set. In Figure 10 we show the encodings for the test set, where each dot represents a time series and colors represent classes. This visualization suggest that the method can be used in time series classification problem. We may treat an encoding as a low dimensional vectorization of a time series and use it as an input to a classifier, e.g. SVM. This is a work in progress and we are going to publish more research results in the future.

5. Conclusion and future work

We introduced an efficient algorithm to compute hidden features of sampled vector fields. Our method utilizes modern machine learning techniques and provide new possibilities to study dynamical systems. By examples we show that the technique is able to find good characteristics of a given sampled vector field as well as collection of such fields.

We show applications of the method for a synthetic and experimental data sets. Our next step is to use the method to analyze bigger data sets, in particular: the magnetic field at the solar surface, time series data transformed into vector fields, velocity fields of a fluid flow measured by particle image velocimetry. Our main goal is to use the extracted features of vector fields as an input to other machine learning methods, e.g. SVM, decision trees, etc.

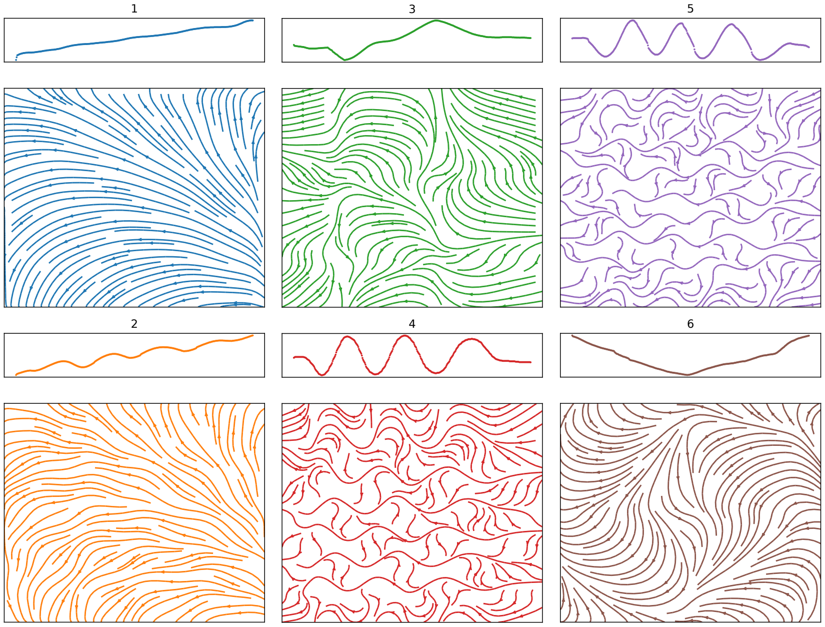


Figure 9. Time series from the *Symbols* data set: one example of a series per class and the stream plots of their vector fields.

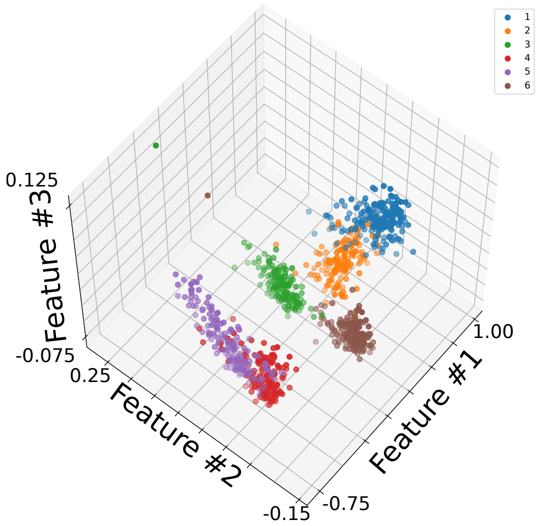


Figure 10. Encodings of the vector fields obtained from the *Symbols* time series data set.

We believe it will become a complementary description of a data set, useful in automatic analysis pipelines.

We would like to thank the anonymous reviewers for their comments and valuable suggestions.

References

[1] P. Alexandroff. "Diskrete Raume". *Recueil Mathematique. Nouvelle Serie* 2 (1937), pp. 501–519.

[2] Z. Arai, W. Kalies, H. Kokubu, K. Mischaikow, H. Oka, and P. Pilarczyk. "A Database Schema for the Analysis of Global Dynamics of Multiparameter Systems". *SIAM Journal on Applied Dynamical Systems* 8.3 (2009), pp. 757–789. ISSN: 1536-0040. DOI: 10.1137/080734935.

[3] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. "The Great Time Series Classification Bake Off: a Review and Experimental Evaluation of Recent Algorithmic Advances". *Data Mining and Knowledge Discovery* 31 (3 2017), pp. 606–660.

[4] J. A. Barmak. *Algebraic Topology of Finite Topological Spaces and Applications.* Lecture Notes in Mathematics. Berlin Heidelberg: Springer-Verlag, 2011. ISBN: 978-3-642-22002-9.

[5] B. Batko, K. Mischaikow, M. Mrozek, and M. Przybylski. "Conley index approach to sampled dynamics". *arXiv:1904.03757 [math]* (2019). arXiv: 1904.03757.

[6] Y. Bengio, H. Schwenk, J.-S. Senecal, F. Morin, and J.-L. Gauvain. "Neural Probabilistic Language Models". In: *Innovations in Machine Learning: Theory and Applications.* Ed. by D. E. Holmes and L. C. Jain. Studies in Fuzziness and Soft Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 137–186. ISBN: 978-3-540-33486-6. DOI: 10.1007/3-540-33486-6_6.

[7] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. "Enriching Word Vectors with Subword Information". *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. DOI: 10.1162/tacl_a_00051.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition.* 3rd. The MIT Press, 2009. 1312 pp. ISBN: 978-0-262-03384-8.

[9] B. Cummins, T. Gedeon, S. Harker, K. Mischaikow, and K. Mok. "Combinatorial Representation of Parameter Space for Switching Networks". *SIAM Journal on Applied Dynamical Systems* 15.4 (2016), pp. 2176–2212. ISSN: 1536-0040. DOI: 10.1137/15M1052743.

[10] T. Dey, M. Juda, T. Kapela, J. Kubica, M. Lipiński, and M. Mrozek. "Persistent Homology of Morse Decompositions in Combinatorial Dynamics". *SIAM Journal on Applied Dynamical Systems* 18.1 (2019), pp. 510–530. DOI: 10.1137/18M1198946.

[11] H. Edelsbrunner, G. Jablonski, and M. Mrozek. "The Persistent Homology of a Self-Map". *Foundations of Computational Mathematics* 15.5 (2015), pp. 1213–1244. ISSN: 1615-3383. DOI: 10.1007/s10208-014-9223-y.

[12] R. Engelking. *General topology.* Heldermann Verlag, 1989. 552 pp. ISBN: 978-3-88538-006-1.

[13] R. Forman. "Combinatorial vector fields and dynamical systems". *Mathematische Zeitschrift* 228.4 (1998), pp. 629–681. ISSN: 1432-1823. DOI: 10.1007/PL00004638.

[14] R. Forman. "Morse Theory for Cell Complexes". *Advances in Mathematics* 134.1 (1998), pp. 90–145. ISSN: 0001-8708. DOI: 10.1006/aima.1997.1650.

[15] T. Gedeon, B. Cummins, S. Harker, and K. Mischaikow. "Identifying robust hysteresis in networks". *PLOS Computational Biology* 14.4 (2018), e1006121. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1006121.

[16] J. Ghosh, B. Sahoo, and S. Poria. "Prey-predator dynamics with prey refuge providing additional food to predator". *Chaos, Solitons & Fractals* 96 (2017), pp. 110–119. ISSN: 09600779. DOI: 10.1016/j.chaos.2017.01.010.

[17] Z. S. Harris. "Distributional Structure". *WORD* 10.2 (1954), pp. 146–162. ISSN: 0043-7956, 2373-5112. DOI: 10.1080/00437956.1954.11659520.

[18] D. E. Knuth. *The art of computer programming, volume 3: (2nd ed.) sorting and searching.* USA: Addison Wesley Longman Publishing Co., Inc., 1998. ISBN: 978-0-201-89685-5.

[19] T. Mikolov, K. Chen, G. Corrado, and J. Dean. "Efficient Estimation of Word Representations in Vector Space". *arXiv:1301.3781 [cs]* (2013). arXiv: 1301.3781.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems 26.* Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Curran Associates, Inc., 2013, pp. 3111–3119.

[21] D. Mimno and L. Thompson. "The strange geometry of skip-gram with negative sampling". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.* Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 2873–2878. DOI: 10.18653/v1/D17-1308.

[22] K. Mischaikow, M. Mrozek, J. Reiss, and A. Szymczak. "Construction of Symbolic Dynamics from Experimental Time Series". *Physical Review Letters* 82.6 (1999), pp. 1144–1147. DOI: 10.1103/PhysRevLett.82.1144.

[23] M. Mrozek. "Conley–Morse–Forman Theory for Combinatorial Multivector Fields on Lefschetz Complexes". *Foundations of Computational Mathematics* 17.6 (2017), pp. 1585–1633. ISSN: 1615-3375, 1615-3383. DOI: 10.1007/s10208-016-9330-z.

[24] J. R. Munkres. *Elements Of Algebraic Topology.* CRC Press, 2018. ISBN: 978-0-429-49391-1. DOI: 10.1201/9780429493911.

[25] B. Perozzi, R. Al-Rfou, and S. Skiena. "DeepWalk: online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14.* New York, New York, USA: ACM Press, 2014, pp. 701–710. ISBN: 978-1-4503-2956-9. DOI: 10.1145/2623330.2623732.

[26] A. Rajaraman and J. D. Ullman. *Mining of Massive Datasets.* New York, NY, USA: Cambridge University Press, 2011. ISBN: 978-1-107-01535-7.

[27] P. H. Scherrer, J. Schou, R. I. Bush, A. G. Kosovichev, R. S. Bogart, J. T. Hoeksema, Y. Liu, T. L. Duvall, J. Zhao, A. M. Title, C. J. Schrijver, T. D. Tarbell, and S. Tomczyk. "The Helioseismic and Magnetic Imager (HMI)Investigation for the Solar Dynamics Observatory (SDO)". *Solar Physics* 275.1 (2012), pp. 207–227. ISSN: 1573-093X. DOI: 10.1007/s11207-011-9834-2.

[28] A. Szymczak and E. Zhang. "Robust Morse Decompositions of Piecewise Constant Vector Fields". *IEEE Transactions on Visualization and Computer Graphics* 18.6 (2012), pp. 938–951. DOI: 10.1109/TVCG.2011.88.

[29] F. Takens. "Detecting strange attractors in turbulence". In: *Dynamical Systems and Turbulence, Warwick 1980.* Ed. by D. Rand and L.-S. Young. Lecture Notes in Mathematics. Berlin, Heidelberg: Springer, 1981, pp. 366–381. ISBN: 978-3-540-38945-3. DOI: 10.1007/BFb0091924.

[30] Z. Wang and T. Oates. "Imaging Time-Series to Improve Classification and Imputation". In: *Twenty-Fourth International Joint Conference on Artificial Intelligence.* Twenty-Fourth International Joint Conference on Artificial Intelligence. 2015.

Mateusz Juda, Division of Computational Mathematics, Institute of Computer Science and Computational Mathematics, Faculty of Mathematics and Computer Science, Jagiellonian University, ul. St. Łojasiewicza 6, 30-348 Kraków, Poland.

Email address: mateusz.juda@ii.uj.edu.pl