Normalization of Life Science Data for Shape-based Similarity Querying

Dina Q. Goldin
University of Connecticut

ABSTRACT

In this paper, we focus on *shape-based similarity querying* over life science data, and discuss the key role played by normalization in this context. Our treatment of normalization is based on the semantics of *similarity transformations*, which are mathematical groups. We make an explicit association between normalization and equivalence classes over data, contaning data items with the same shape. Normalization obtains (a) the *normal form* of each data item, which serves as a representative of its equivalence class, together with (b) the *normalization parameters* that allow us to map the normal form back to the original item. Normalization-based notions of similarity distance are also treated.

1. INTRODUCTION

Life Sciences include biology, medicine, and related areas (such as plant science, virology, immunology, etc.). Research in data mining and information retrieval for life sciences data has involved medical image data [PC03, PF97, KSF98] as well as time-series data [GK95, CW99, Aach01, KK02]. Specific examples of time-series data for the life sciences include:

- blood glucose levels
- ECG
- soil temperatures
- water levels

For these examples, as well as for medical images of tumors, it is often the *shape* of the data that is of interest, rather than (or in addition to) the actual values.

In this paper, we focus on *shape-based similarity querying*, which is concerned with the shape of the data items (such as time-series sequences, or tumor outlines), and looks for data items whose shape, rather than values, is similar to that of the query item. For example:

- Blood glucose levels of a fat and a skinny patient may be different, but they may follow the same pattern that indicates hypoglycemia.
- Soil temperatures on two different days may start at different values, but may rise and fall in the same way.

Data mining techniques for life-sciences data, such as similarity querying, often involve the preprocessing of data; the various techniques for data preprocessing for life-sciences data include data smoothing, and normalization.

Normalization is preprocessing technique that "standardizes" life-sciences data by transforming it to a normal form. This transformation also generates *normalization parameters*, that allow a mapping from the normal form back to the original data. There are several types of normalization, such as:

- min-max normalization
- z score normalization
- decimal scaling

Normalization plays a key role in efficient and flexible shape-based similarity querying systems, such as [GK95, CW99]. The present paper is concerned with analyzing and formalizing the role of normalization in shape-based similarity querying; it is a technical treatment, providing formal semantics for the normalization. Every attempt is made to keep the discussion as general as possible, confining to examples all discussion of specific types of data, or specific transformations, or specific normal forms.

Unlike other treatments of this topic, ours is based on the semantics of *similarity transformations*, which are mathematical groups. We make an explicit association between normalization and equivalence classes over data, contaning data items with the same shape. Normalization obtains (a) the *normal form* of each data item, which serves as a representative of its equivalence class, together with (b) the *normalization parameters* that allow us to map the normal form back to the original item. Normalization-based notions of similarity distance are also treated.

Other types of data mining for life-sciences data call for normalization as well, notably microarray analysis. Normalization is a used in mircroarray analysis to remove non biological variations introduced by the experimental process. Several techniques for normalization of microarrays have been developed, including [SS03, Qua02, CVFB03, SD00]. These will not be discussed in this article.

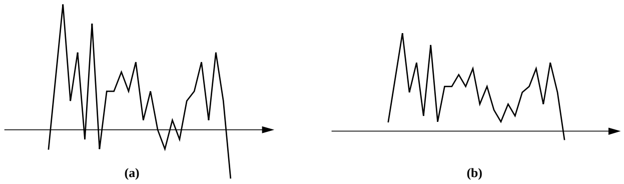

Figure 1: (b) is a similarity transformation of (a).

2. SEMANTICS OF SHAPE-BASED SIMILARITY

This section provides the transformation-based semantics and the definitions for similarity querying, which are based on normal forms.

2.1 Similarity Transformations

Our formalization of the notion of *shape-based similarity* is based on *transformations* between data items (either sequences of time-series data, or tumor/shape outlines), from some set of transformations G. Data items are defined *similar* if there exists a transformation in G which maps one to the other:

DEFINITION 2.1. **(Similarity)** *Let D be a distance metric between data items and $\epsilon \geq 0$ a tolerance. Query sequence Q is* approximately similar *within tolerance ϵ to data sequence S when there exists a similarity transformation T in G so that $D(Q, T(S)) \leq \epsilon$. When ϵ is set to 0, we say that Q and S are* exactly similar*, or just* similar.

In the case of time-series sequences, we consider two *basic* classes of transformations:

- *shift transformations*, where the value of all members in a sequences is increased or decreased by the same constraint, and
- *scale transformations*, where the all the values are multiplied by some constant.

In the case of tumor/shape outlines, the *basic* transformations are:

- *rotation* (around a fixed point, such as centroid),
- *scaling* (same shape, different area),
- *x-translation*, and
- *y-translation*.

Compositions of two or more basic transformations yield new classes of transformations; all of these transformation classes form *groups*; The mathematical field of *Transformational Geometry* [MP65] is a theory of such transformations.

EXAMPLE 2.1. *Compositions of shift- and scale- transformations are transformations that are, coincidentally, known in Transformational Geometry as* similarity transformations. *Figure 1 illustrates a time-series sequence before and after a similarity transformation.*

Note: this example, as well as all subsequent ones, is based on [GK95], which pioneered the use of normalization for shape-based similarity queries. However, the treatment of normalization for shape-based queries in the present paper is more general and more extensive than elsewhere in the literature.

The group property of transformations implies that each transformation T has an *inverse transformation* T^{-1}. For any data item X, $T(T^{-1}(X)) = T^{-1}(T(X)) = X$.

Also, given a class of transformations $\mathcal{T}$, any transformation T_P in $\mathcal{T}$ can be characterized by a tuple of real-valued *transformation parameters* P, with one parameter for each basic transformation involved.

EXAMPLE 2.2. *For similarity transformations over time-series sequences (Example 2.1), the transformation parameters are a pair of reals (a, b) (where $a > 0$); a is the* scale *parameter, and b is the* shift *parameter. The* similarity *transformation $T_{a,b}$ maps each element x_i in the sequence to $a * x_i + b$. The inverse transformation of $T_{a,b}$ is $T_{a,b}^{-1} = T_{1/a,-b/a}$.*

2.2 Similarity Classes and their Normal Forms

Every class of transformations T_P induces the corresponding similarity relation over the data items; by Definition 2.1 we have that X is exactly similar to Y iff there exist values for transformation parameters P such that $X = T_P(Y)$. It can be shown for every class of transformations, the corresponding *similarity relation* S is *reflexive*, *symmetric*, and *transitive*.

EXAMPLE 2.3. *Consider composition of shift- and scale-transformations defined in Example 2.2.*

Reflexivity: *for any sequence X, $X = T_{1,0}(X)$ (the* identity *transformation);*

Symmetry: *if $X = T_{a,b}(Y)$ then $Y = T_{1/a,-b/a}(X) = T_{a,b}^{-1}(X)$ (the* inverse *of $T_{a,b}$);*

Transitivity: *if $X = T_{a,b}(Y)$ and $Y = T_{c,d}(Z)$, then $X = T_{ac,ad+b}(Z) = (T_{a,b} * T_{c,d})(Z)$ (the non-commutative* product *of $T_{a,b}$ and $T_{c,d}$).*

Therefore, every class of transformations $\mathcal{T}$ partitions all data elements into equivalence classes, which we call *similarity classes.* We shall denote the similarity class of X by $\mathcal{T}(X)$; it consists of all data items Y such that $S(X, Y)$.

The *normal form* of any data item X is another member of $\mathcal{T}(X)$, which serves as a unique representative for that class:

DEFINITION 2.2. *For any data item X, its* normal form *is a data item $\nu(X)$ such that:*

- $S(X, \nu(X))$
- *if $S(X, Y)$, then $\nu(X) = \nu(Y)$*

We say that X is normal *if $\nu(X) = X$.*

This normal form is critical to our evaluation strategy for shape-based similarity queries.

The *normalization parameters* of any data item X allow us to obtain X from its normal form:

DEFINITION 2.3. *Given a data item X, its* normalization parameters *are the unique transformation parameters P such that $X = T_P(\nu(X))$.*

We will make the assumption that given any data item X, its normalization parameters are efficiently computable. If $X = T_P(\nu(X))$, then $\nu(X) = T_P^{-1}(X)$; hence, these normalization parameters allow us to compute both transformations (from and to the normal form).

EXAMPLE 2.4. *Given a time-series sequence X (of length n), let us denote its* average *by $\alpha(X)$, and its* standard deviation *by $\sigma(X)$:*

$$\alpha(X) = (1/n)\sum_{1\le i\le n} x_i;$$
$$\sigma(X) = ((1/n)\sum_{1\le i\le n}(x_i - \alpha(X))^2)^{1/2}.$$

We say that X is normal *if $\sigma(X) = 1$ and $\alpha(X) = 0$; that is, the normal form of X has the standard deviation of 1 and the average value of 0. Then, $(\sigma(X), \alpha(X))$ serve as normalization parameters for X:*

$$X = T_{\sigma(X),\alpha(X)}(\nu(X)).$$

An alternate approach to normalizing time-series data is to normalize its *bounding box*. In this case, the scale and shift factors depend on the interval between the *minimum* and the *maximum* values, rather than on the standard deviation and the average as in Example 2.4. This method is more sensitive to outliers, and not as effective for shape-based similarity querying.

Normalization is a technique that obtains, for any data item X, both its normal form $\nu(X)$ and its normalization parameters P. Normalization-based data mining systems rely on normalization for data preprocessing.

2.3 Similarity Distance

Similarity distance D_S tells us how similar pairs of data items are; given three items X, Y, Z, we say that X is more similar to Y than to Z if $D_S(X,Y) < D_S(X,Z)$.

DEFINITION 2.4. *Given two data items X and Y, and a distance metric D between data items, the* similarity distance *between X and Y, denoted $D_S(X,Y)$, is the distance between their normal forms:*

$$D_S(X,Y) = D(\nu(X), \nu(Y))$$

Note that the similarity distance between any pair of sequences from $\mathcal{T}(X)$ and $\mathcal{T}(Y)$ is the same.

D_S is a generalization of a similarity relation S:

$S(X,Y)$ is true iff $D_S(X,Y) = 0$.

EXAMPLE 2.5. *The similarity distance between the time-series sequences (a) and (b) of Figure 1 is 0.*

It is easy to see that, whenever D is a distance metric, then so is D_S:

- it is non-negative and symmetric;
- it obeys the triangle inequality;
- it is effectively computable.

3. NORMALIZATION FOR SIMILARITY QUERIES

Having defined the semantics of similarity and of normal forms, we are ready to discuss the role that normalization plays in shape-based similarity queries.

3.1 Exact Similarity Queries

The *Exact Similarity* query is defined as follows:

DEFINITION 3.1. **(Exact Similarity Query)** *Given a query item X and a similarity relation S, find all data items Y such that $S(X,Y)$; i.e., $D_S(X,Y) = 0$.*

Exact Similarity queries are to be contrasted with Exact Match queries, where we look for data items that are the same as (rather than similar to) the query item:

DEFINITION 3.2. **(Exact Match Query)** *Given a query item X, find all data items Y such that $X = Y$; i.e., $D(X,Y) = 0$.*

Note: While these queries do not seem very useful, especially the latter, their distance-based generalizations in the next section are more useful.

Normalization data allows one to answer both of the above queries without the need to access the original data, nor to perform any transformations during query evaluation. This is due to the following observations:

- for any data items X and Y, $S(X,Y)$ iff $\nu(X) = \nu(Y)$;
- for any data items X and Y, $X = Y$ iff $\nu(X) = \nu(Y)$ and the normalization parameters of X match those of Y.

In essense, for normalization-based systems, both of these queries are just special cases of look-up queries:

PROPOSITION 3.1. *The output of an exact similarity query consists of all data sequences whose normal form is the same as for the query sequence; the output of an exact match query consists of all data sequences whose normal form and normalization parameters are the same as for the query sequence.*

Proof: Follows from observations above.

There exist very efficient strategies for implementing look-up queries; normalization therefore serves as an optimization technique for similarity queries. By performing normalization up front, these systems avoid data transformations during similarity queries. Since the same dataset is typically subjected to multiple similarity queries, the time needed for normalization is more than offset by the time saved during the queries.

The advantages of normalization become even more apparent when one considers composite similarity transformations, consisting of two or more basic ones (Section 2.1). In this case, *all* similarity queries over the dataset – not only for similarity relation that corresponds to the composite similarity transformation, but for the similarity relation the corresponds to all the basic transformations as well – can be implemented as look-up queries over the *same* normalized data.

EXAMPLE 3.1. *The normalization data for time-series sequences consists of their normal form, and the shift and scale parameters (Example 2.4). Consider two time-series sequences X and Y, whose normalization data is $(\nu(X), a_X, b_X)$ and $(\nu(Y), a_Y, b_Y)$, respectively. Then:*

- *X and Y have the same shape if $\nu(X) = \nu(Y)$;*
- *X and Y are the same up to a shift transformation if $(\nu(X), a_X) = (\nu(Y), a_Y)$;*
- *X and Y are the same up to a scale transformation if $(\nu(X), b_X) = (\nu(Y), b_Y)$;*
- *X and Y are the same if $(\nu(X), a_X, b_X) = (\nu(Y), a_Y, b_Y)$.*

3.2 Similarity Queries with Distance

In this section, we generalize Exact Similarity and Exact Match queries to include distance; normalization continues playing a key role in the efficient evaluation of the resulting queries.

Similarity distance provides a basis for defining *Approximate Similarity* queries:

DEFINITION 3.3. **(Approximate Similarity Query)** *Given a query item X, a tolerance $\epsilon \geq 0$, a distance metric D, and a similarity relation S, find all data items Y such that $D_S(X, Y) \leq \epsilon$.*

By definition, the similarity distance between two data items is the distance between their normal forms. Just as for Exact Similarity queries, normalization allows us to precompute the normal forms, making query evaluation more efficient.

Analogously, we can define *Approximate Match* queries:

DEFINITION 3.4. **(Approximate Match Query)** *Given a query item X, a tolerance $\epsilon \geq 0$, and a distance metric D, find all data items Y such that $D(X, Y) \leq \epsilon$.*

Again, the goal is to make use of a single set of normalization data for efficient evaluation of both of the above queries, as well as for the evaluation of the various Approximate Similarity queries, for various notions of similarity. But now, query evaluation no longer consists of simple look-up, and the correspondence between Approximate Similarity and Approximate Match is not as straightforward as for their exact versions.

We view these two queries as special cases of *general similarity queries*, discussed below.

3.3 General Similarity Queries

The most general type of shape-based similarity query is as follows:

DEFINITION 3.5. **(General Similarity Query)** *Given a query item X, a tolerance $\epsilon \geq 0$, and bounds (lower and upper) for each transformation parameter, find all $[Y, P]$, where Y is a data item and P is transformation parameters within the specified bounds, such that $D(X, T_P(Y)) \leq \epsilon$.*

Note that the bounds on transformation parameters may have the value of ∞.

General queries may be specialized by omitting the bounds for one or more transformation parameter. When all bounds are omitted, we obtain an *approximate similarity query* as a special case of the general similarity query.

EXAMPLE 3.2. *For shift- and scale- similarity, the bounds form two intevals, for the scale and shift parameters respectively:*

$$(l_a, u_a); (l_b, u_b).$$

The resulting query is as follows:

> *Given X and ϵ, we are looking for all (Y, a, b) such that $D(X, aY + b) \leq \epsilon$, where $l_a \leq a \leq u_a$ and $l_b \leq b \leq u_b$.*

The bounds on a and/or b can be omitted to specialize the query. For example, if want to query for scaling transformations only, we omit the bound on the shift parameter. The resulting query is as follows:

> *Given X and ϵ, we are looking for all (Y, a) such that $D(X, aY) \leq \epsilon$, where $l_a \leq a \leq u_a$.*

The normalization techniques for general similarity queries, in the case of time-series data, were pioneered in [GK95] and implemented in [Mill96]. The approach is to translate the parameters of the general query (i.e., bounds on the transformation parameters and ϵ) to corresponding parameters of a range query over normalization data (i.e., bounds on the normalization parameters and ϵ_i, the *internal epsilon*). The latter parameters are then used in a range query against an index structure (such as an R^*-tree) that holds the normalization data.

EXAMPLE 3.3. *The general query in Example 3.2 translates to the following range query:*

Given a query sequence X, an internal tolerance $\epsilon_i \geq 0$, bounds $l_\sigma \leq u_\sigma$ and $l_\alpha \leq u_\alpha$, find all items Y in the dataset such that $D(\nu(X), \nu(Y)) \leq \epsilon_i$, $l_\alpha \leq \alpha(Y) \leq u_\alpha$, and $l_\sigma \leq \sigma(Y) \leq u_\sigma$.

The (internal) range query allows us to determine which sequences might be a potential match for our (external) similarity query. A post-processing *filtering* step is needed to weed out *false alarms* that match the range query but not the similarity query. The efficiency of the index-based range query more than compensates for the inefficiency of the extra filtering step.

Note that while the system allows such false alarms, it does not allow *false dismissals*; all sequences that satisfy the similarity query are retrieved by the range query.

4. DISCUSSION

Normalization is a data preprocessing technique that plays an important role in data mining of life-sciences data. The present paper was concerned with analyzing and formalizing the role of normalization in shape-based similarity querying over life-science data, specifically time-series data and tumor/shape data. Shift- and scale- normalization of time-series data was pioneered in [GK95], leading to an efficient yet flexible approach to similarity querying; a prototype system based on this approach was developed in [Mill96]. Other works have followed, including [CW99, ZS03].

Unlike other treatments of this topic, ours was based on the semantics of *similarity transformations*, which can be *basic* or *composite*. We made an explicit association between normalization and equivalence classes over data, contaning data items with the same shape. Every attempt was made to keep our discussion as general as possible, confining to examples all discussion of specific types of data, or specific transformations, or specific normal forms.

Normalization obtains (a) the *normal form* of each data item, which serves as a representative of its equivalence class, together with (b) the *normalization parameters* that allow us to transform the normal form back to the original data item. Normalization-based notions of similarity distance are also treated.

We considered *exact* as well as *approximate* notions of similarity, for basic as well as composite similarity transformations. Note that other extensions of these queries are possible, such as *subsequence search* for time-series data [AFS93]. However, these extensions do not offer any additional insights into the role of normalization, which is the focus of our paper.

5. REFERENCES

[Aach01] J. Aach. Aligning gene expression time series with time warping algorithms, *Bioinformatics* 17(6), pp. 459–508, 2001

[SS03] G. K. Smyth, T. Speed. Normalization of cDNA Microarray Data. In *METHODS: Selecting Candidate Genes from DNA Array Screens*, Ed. D. Carter, April 2003

[Qua02] J. Quackenbush. Microarray data normalization and transformation. *Nature Genetics* 32, pp. 496–501, 2002

[CVFB03] C. Cheadle, M. P. Vawter, W. J. Freed, K. G. Becker Analysis of Microarray Data Using Z Score Transformation. *J. Molecular Diagnostics* 5:2, May 2003.

[Mill96] T. D. Millstein. Design and Implementation of Similarity Querying for Time-Series Databases. *Undergraduate Honors Thesis, Computer Science Department*, Brown Univ., May 1996.

[MP65] Modenov and Pakhomenko. *Geometric Transformations*, Academic Press, 1965

[SD00] J. Schuchhardt, D. Beule et al. Normalization strategies for cDNA microarrays, *Nucleic Acids Research*, 28:10:E47, 2000.

[AFS93] R. Agrawal, C. Faloutsos, A. Swami. Efficient Similarity Search in Sequence Databases. *FODO Conf.*, Evanston, Ill., Oct. 1993

[CW99] K. Chu, M. Wong, Fast time-seres searching with scaling and shifting. In *Proc. ACM PODS* (Symp. on Principles of Database Systems). Philadelphia PA, 1999. pp 237–248.

[GK95] D. Goldin, P. C. Kanellakis. On Similarity Queries for Time-Series Data: Constraint Specification and Implementation. *Proc. 1st Int'l Conf. on the Principles and Practice of Constraint Programming*, LNCS 976, pp. 137–153, Cassis France, Sep. 1995.

[KK02] E. Keogh, S. Kasetty. On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. In *Prof. 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining.* Edmonton Canada, 2002. pp 102–111.

[KSG02] T. Kahveci, A. Singh, A. Gurel. An efficient index structure for shift and scale invariant search of multi-attribute time sequences. In *Proc. of the 18'th lnt'l Conf. on Data Engineering* (ICDE). San Jose, CA, Feb 26-Mar 1 2002.

[PC03] S. Park, W. Chu. Similarity-Based Subsequence Search in Image Sequence Databases. Int'l J. of Image and Graphics, 3:1 (2003), pp. 31–53

[PF97] E. Petrakis, C. Faloutsos, Similarity Searching in Medical Image Databases. *IEEE TKDE*, 9(3) pp. 435–447, 1997.

[KSF98] P. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, Z. Protopapas. Fast and Effective Retrieval of Medical Tumor Shapes. *IEEE TKDE*, 10(6), 1998

[ZS03] Y. Zhu and D. Shasha. Query by humming: a time series database approach. In *Proc. ACM SIGMOD* (Int'l Conf. on Management of Data), San Diego, CA, June 2003.