MICA: Multiple Interval-based Curve Alignment

Martin Mann[a,b,*], Hans-Peter Kahle[a], Matthias Beck[b], Bela Johannes Bender[a], Heinrich Spiecker[a], Rolf Backofen[b,c,d]

[a] *Chair of Forest Growth and Dendroecology, University of Freiburg, Tennenbacher Str. 4, 79106 Freiburg, Germany*
[b] *Bioinformatics Group, Department of Computer Science, University of Freiburg, Georges-Köhler-Allee 106, 79110 Freiburg, Germany*
[c] *Center for Biological Signaling Studies (BIOSS), University of Freiburg, Schänzlestr. 18, 79104 Freiburg, Germany*
[d] *Center for Biological Systems Analysis (ZBSA), University of Freiburg, Habsburgerstr. 49, 79104 Freiburg, Germany*

Abstract

MICA enables the automatic synchronization of discrete data curves. To this end, characteristic points of the curves' shapes are identified. These landmarks are used within a heuristic curve registration approach to align profile pairs by mapping similar characteristics onto each other. In combination with a progressive alignment scheme, this enables the computation of multiple curve alignments.

Multiple curve alignments are needed to derive meaningful representative consensus data of measured time or data series. MICA was already successfully applied to generate representative profiles of tree growth data based on intra-annual wood density profiles or cell formation data.

The MICA package provides a command-line and graphical user interface. The `R` interface enables the direct embedding of multiple curve alignment computation into larger analyses pipelines. Source code, binaries and documentation are freely available at https://github.com/BackofenLab/MICA

Keywords: curve alignment, landmark registration, global alignment,

*Corresponding author
Email addresses: `mmann@informatik.uni-freiburg.de` (Martin Mann), `hans-peter.kahle@iww.uni-freiburg.de` (Hans-Peter Kahle), `instww@uni-freiburg.de` (Heinrich Spiecker), `backofen@informatik.uni-freiburg.de` (Rolf Backofen)

Background

Biological data is often derived and represented in the form of time series, curves or profiles. To enable generalization, data of multiple measurements or for different instances has to be aggregated to derive e.g. statistics or extract common curve shapes. This is straightforward, if a clear correspondence between data points is available. However, in many situations such a correspondence is unknown or only partially available. Here, alignment or time registration techniques are required, in order to map corresponding data. This so called curve registration problem [1], also known as time warping [2] or curve alignment, has many applications in biological research like the study of gene expression profiles [3, 4], growth data [5], wood anatomy [6, 7], or medical sensoring [8, 9], to name only a few.

Here, we present MICA - Multiple Interval-based Curve Alignment - a software tool for the global alignment of curves. MICA combines a heuristic pairwise curve alignment strategy using a landmark registration approach [10, 11] with a progressive alignment scheme [12, 13] to generate a multiple curve alignment. To this end, important curve characteristics like optima and inflection points are automatically identified, filtered and the resulting curve intervals aligned. MICA was already successfully applied to study the correlation between weather data and wood density profiles [6] and to temporarily annotate wood anatomic data [7]. It comes with a graphical user interface for interactive usage, a command-line interface for high-throughput application and an `R` interface to embed curve alignment into downstream analyses. Within this manuscript, we introduce the algorithmic and implementation details of the recent MICA version and discuss the different user interfaces of MICA.

Preliminaries

In the following, we provide the mathematical background and objectives necessary to introduce the MICA approach.

Curve alignment problem

A *curve (or profile)* C is defined by the tuple (X, Y), where $X, Y \in \mathbb{R}^n$ encode a set of n 2D-coordinates (X has to be strictly monotone, i.e. $\forall_{1 \leq i < n} : X_i < X_{i+1}$).

In order to compute the alignment, we need to access interpolated curve/slope coordinates for any x-coordinate within the interval $[X_1, X_n]$. To this end, we use the *interpolation function* $y(x, C)$ that provides the y-coordinate for an $x \in [X_1, X_n]$. If x is a known x-coordinate from X, the according y-coordinate is returned. If this is not the case, i.e. $\nexists i : X_i = x$, the value is derived via linear interpolation between the enclosing coordinates identified by $\arg\max_i : X_i < x$ and $\arg\min_j : X_j > x$. Linear interpolation is applied since it is i) fast to compute and ii) preserves the min/max characteristics of the curve data. The function $s(x, C)$ is defined analogously to provide interpolated slope values, i.e. first derivatives of the curve. Since we are using a linear interpolation, the slope values are defined by the slopes of the lines connecting explicit data points of X. Thus, if x-coordinates are changed during alignment, the slope values are changing too and have to be updated. Both interpolation functions y and s are implemented by the LinearInterpolator class from the Apache commons.math3 package.

Given two curves C, C', we define the *global slope-based distance function* $d_b^s(C, C')$, which computes the arithmetic mean of absolute slope differences for $b > 0$ equidistant x-coordinates within the whole x-ranges of the two profiles, i.e.

$$\begin{aligned} d_b^s(C, C') &= b^{-1} \sum_{1 \leq j \leq b} |s(X_1 + j\delta, C) - s(X'_1 + j\delta', C')| \qquad (1) \\ \text{with } \delta &= (X_n - X_1)/(b+1) \quad \text{and } \delta' \text{ analogously.} \end{aligned}$$

A distance function based on y-coordinates is defined analogously. Note, a slope-based distance measure is invariant to general shifts of the y-coordinates e.g. due to measurement issues. Thus, the slope-based distance function from Eq. 1 is applied in the following.

In order to correct for x-coordinate shifts or to align respective data points, we have to shift the x-coordinates X of a curve C while preserving their order. Such a shift can be encoded by an injective *warping function* $a(X_i) \in [X_1, X_n]$ that maps each x-coordinate within the overall x-range. In order to keep the coordinate order, a has to be strictly monotone, i.e. $X_i < X_j \rightarrow a(X_i) < a(X_j)$. Furthermore, since we are interested in global alignments of the whole curves, the start and end coordinate have to be preserved, i.e. $a(X_1) = X_1$ and $a(X_n) = X_n$.

Given this, we can define the *global pairwise curve alignment problem* for two curves C, C' and a global distance function d as the problem to find two warping functions a and a' that minimize the distance function. Formally, this is given by

$$\arg\min_{a,a'} d\left(\left(a(X), Y\right), \left(a'(X'), Y'\right) \right). \tag{2}$$

W.l.o.g., we assume that the start and end coordinates of the curves are already aligned, i.e. $X_1 = X'_1$ and $X_n = X'_{n'}$.

The according *global multiple curve alignment problem* for k curves is to find according k warping functions that minimize the sum of all pairwise distances in analogy to Eq. 2.

Given a set of k aligned curves $\{C^1, \ldots, C^k\}$, the *representative consensus profile* $\bar{C} = (\bar{X}, \bar{Y})$ contains one coordinate for each x-coordinate present in at least one of the curves. The respective y-coordinates are defined by according mean y-values, i.e.

$$\bar{X} = \bigcup_{1 \leq l \leq k} X^l \tag{3}$$

$$\bar{Y}_i = k^{-1} \sum_{1 \leq l \leq k} y(\bar{X}_i, C^l). \tag{4}$$

An illustration is provided in Fig. 1d+e.

Landmark-based curve alignment

To reduce the computational cost of curve alignments, one way is to constrain the warping functions under consideration. To this end, one can refer to *landmark-based curve alignments*, also named curve or landmark registration [10, 11]. The general idea is to identify a subset of curve coordinates that mark important or distinct features of the curves and to find the best mapping/alignment of these landmarks only, where aligned landmarks are shifted to the same x-coordinate. All other coordinates are then shifted accordingly via linear interpolation, to preserve the strict monotone character of a warping function. Since this approach only aligns only the (small) subset of landmarks, the search space is strongly reduced to the combinatorial subspace of alignable landmarks.

To encode whether or not a curve's coordinate is to be considered as an alignable landmark, we introduce the *curve annotation* $L(C) \in \mathbb{N}^n$, or abbreviated by just L. The annotation value L_i encodes whether the i-th coordinate of C is a landmark that can be aligned ($L_i > 0$) or not ($L_i \leq 0$). The value itself encodes the type of the landmark. Note, since we do global alignment, start and end coordinate are always to be aligned and thus it holds $L_1, L_n > 0$.

To simplify presentation and since we are interested in global curve alignments, we assume in the following that all input curves show the same x-range. Given k curves $C^1, \ldots, C^k$, this can be done by a simple preprocessing as follows. First the mean start coordinate $\bar{x}_1 = k^{-1} \sum_{1 \leq l \leq k} X_1^l$ and mean overall x-range $\bar{r} = k^{-1} \sum_{1 \leq l \leq k} (X_n^l - X_1^l)$ of all curves are identified. Successively, the normalized x-coordinates X^* from X of each curve are computed by $X_i^* = \bar{r}(X_i - X_1)/(X_n - X_1) + \bar{x}_1$. An illustration is provided in Fig. 1a+b.

Given two curves C, C' with n, n' coordinates, resp., a *landmark alignment* $A : [1, n] \times [1, n'] \to [X_1, X_n]$ is a partial injective function that maps pairs of coordinate indices (i, i') to their aligned x-coordinate (assuming normalized x-ranges). Note, only landmark positions are mapped, i.e. $((i, i'), x) \in A \to (L_i > 0 \wedge L'_{i'} > 0)$. Furthermore, aligned landmarks have to be compatible, e.g. to ensure curve maxima are mapped to maxima but not to minima. Fi-

nally, each landmark can only be mapped once, i.e. $\forall_{((i,i'),x)\neq((j,j'),x')\in A} : (i \neq j \wedge i' \neq j')$, and the landmark alignment has to be ordered and monotone, i.e. $\forall_{((i,i'),x)\neq((j,j'),x')\in A} : (i < j \rightarrow i' < j') \wedge (i < j \rightarrow x < x')$. Note, start and end coordinate are per definition a landmark (see above) and have to be part of the landmark alignment, i.e. $A(1,1) = X_1 = X'_1$ and $A(n,n') = X_n = X'_n$.

Given a landmark alignment A for two curves C, C', we can derive two warping functions a_A and a'_A as follows. Coordinates that are mapped landmarks, i.e. part of A, are shifted to their aligned x-coordinate provided by A. All other coordinates are shifted according to a linear interpolation within the new (mapped) range of their flanking aligned landmarks. Note, this is always possible since start and end points are part of A. Thus, it holds

$$\begin{aligned} a_A(X)_i &= \begin{cases} A(i,i') & \text{if } \exists((i,i'),x) \in A, \text{ i.e. mapped} \\ x^A_{\text{left}} + \frac{(X_i - X_l)(x^A_{\text{right}} - x^A_{\text{left}})}{(X_r - X_l)} \end{cases} \\ \text{with } l &= \arg\max_{1 \leq j < i} (\exists((j,j'),x) \in A) \quad \text{and} \quad x^A_{\text{left}} = A((l,l')) \\ \text{and } r &= \arg\max_{i < j \leq n} (\exists((j,j'),x) \in A) \quad \text{and} \quad x^A_{\text{right}} = A((r,r')) \end{aligned} \tag{5}$$

while $a'_A(X)$ is defined analogously to Eq. 5.

Given this, we define the *global landmark-based pairwise curve alignment problem* to be the identification of the landmark alignment A for two given curves C, C' with respective annotations L, L' that minimizes a given global curve distance function d, i.e.

$$\arg\min_A d\left((a_A(X), Y), (a'_A(X'), Y') \right). \tag{6}$$

Note, since we constrain the considered warping functions, Eq. 6 will usually find only suboptimal solutions for Eq. 2. The *global landmark-based multiple curve alignment problem* is defined in analogy to the global multiple curve alignment problem.

Algorithm and Implementation

MICA is tailored to align multiple profiles of discretized curve data to derive a representative consensus profile and addresses the global landmark-based

multiple curve alignment problem. It assumes that start/end points are corresponding and thus can be aligned, i.e. the alignment is done globally and the whole curves are considered. By doing global alignment we follow the idea that all curves share a similar pattern or shape. This implies that all curves are essentially based on the same e.g. growth profile and differences are mainly distortions as the profile can vary in signal strength (amplitude), character and spatial/temporal assignment (x-axis) due to noise, measurement problems, asynchronicity, etc. A curve can be provided as y-data only (equidistant distant data points assumed) or with explicit coordinate data.

As an example, we can use intra-annual wood density profiles measured on a tree stem cross section (disk) from the tree's pith to the bark along different radial directions [6, 14]. Each curve is generated by a growth process based on the same growth conditions, which are determined for example by the prevailing weather conditions. Nevertheless, the cambium tissue around a tree stem shows different growth activities even along the circumference at the same height of an individual tree. This manifests in multifaceted differences and asynchronicity within measured data, which makes the generation of a representative consensus profile per tree via arithmetic mean impossible even for the sub-profiles of a single growth period [6]. Here, MICA provides a solution to this problem by first synchronizing the profiles before a consensus is derived.

MICA applies a progressive alignment scheme based on a heuristic pairwise interval-based curve alignment strategy (PICA) detailed below. The progressive scheme, as known from sequence alignment [12], iteratively evaluates the similarity of subsets of already aligned curves based on pairwise alignments (PICA). The pairwise alignment information is then also used to merge the curves into a joint alignment.

To simplify the presentation, we assume that all input curves $C^1, \ldots, C^k$ are normalized to a common x-range as discussed above, i.e. their start and end x-coordinate are the same. Furthermore, we partition the set of input curves into singleton sets, each containing exactly one of the input curves. These initial curve sets are denoted by $\mathcal{P} = \{P_1, \ldots, P_k\}$ with $P_i = \{C^i\}$. The representative

consensus profile of a curve set $P \in \mathcal{P}$ is denoted by $\bar{P}$. MICA will use these consensi to iteratively align and fuse the according aligned curve sets.

PICA workflow

The Pairwise Interval-based Curve Alignment - PICA - implements a greedy heuristic approach to address the global landmark-based pairwise curve alignment problem. That is, given two sets of aligned curves $P, P' \in \mathcal{P}$ and their respective consensus profiles $\bar{P}, \bar{P}'$ with according landmark annotations L, L', PICA identifies a landmark alignment A of the consensus profiles that optimizes a given curve distance function d in accordance with Eq. 6. W.l.o.g., we assume d to be the slope-based distance function from Eq. 1.

To check whether or not two coordinates of the consensi to align are landmarks and can be mapped (e.g. maxima on maxima but not on minima), we introduce the relation $comp \subseteq [1, n] \times [1, n']$, where n, n' are the number of coordinates in $\bar{P}, \bar{P}'$, respectively. The relation *comp* contains all combinations of positive landmark assignments that are compatible. A most simple relation would be based on identity, i.e. $(i, i') \in comp \leftrightarrow (L_i > 0 \wedge L_i = L'_i)$.

Given this, we can sketch the PICA workflow that follows a greedy divide-and-conquer alignment strategy. First, we initialize the landmark alignment $A = \{(1, 1,), (n, n')\}$, i.e. we map the start and end points of each consensus. This provides the initial search interval boundaries for compatible landmarks defined by $\overset{<}{m} = 1 = \overset{<}{m'}$, $\overset{>}{m} = n$, and $\overset{>}{m'} = n'$. Given this, we search for the pair of compatible landmarks within the open interval (excluding the boundaries) that minimize the distance function when mapped to their mean x-coordinate (weighted by the number of curves represented by the consensi), i.e.

$$\underset{\substack{(i,i') \in \left(\overset{<}{m}, \overset{>}{m}\right) \times \left(\overset{<}{m'}, \overset{>}{m'}\right) \\ (i,i') \in comp}}{\arg\min} \quad d\left(\left(a_{A^*}(\bar{X}), \bar{Y}\right), \left(a'_{A^*}(\bar{X}'), \bar{Y}'\right) \right) \tag{7}$$

$$\text{with } A^* = A \cup \{ ((i, i'), x^*) \} \qquad \text{and} \quad x^* = \frac{|P|\bar{X}_i + |P'|\bar{X}'_{i'}}{|P| + |P'|}. \tag{8}$$

Note, the weighting of the mean coordinate x^* in Eq. 8 is needed to reduce side

effects of the progressive alignment strategy, e.g. when aligning a single curve to a set of many curves.

If the optimal landmark pair (i, i') within the current interval identified via Eq. 7 provides a better alignment compared to no further alignment, i.e.

$$d\left(\left(a_{A^*}(\bar{X}), \bar{Y}\right), \left(a'_{A^*}(\bar{X}'), \bar{Y}'\right)\right) \quad < \quad d\left(\left(a_A(\bar{X}), \bar{Y}\right), \left(a'_A(\bar{X}'), \bar{Y}'\right)\right),$$

we fix this alignment decision and replace the current landmark alignment A with the extended alignment A^* from Eq. 8. This fixation decomposes the alignment problem into two independent subproblems, namely the optimization of the intervals to the left and right of the aligned landmark pair (i, i'). Therefore, we recursively repeat the sketched PICA workflow for the new subinterval combinations $(\overset{<}{m}, i) \times (\overset{<}{m'}, i')$ and $(i, \overset{>}{m}) \times (i', \overset{>}{m'})$, which is illustrated in Fig. 1c+d. Note, the PICA workflow does not necessarily map all available landmarks, which is essential since typically the curves to be aligned feature different numbers of landmarks.

The resulting final landmark alignment A is then used to align and merge the two curve sets P and P' into a new curve set $\hat{P}$, which is the final output of PICA. To this end, we add for each curve $C = (X, Y) \in P$ a warped curve (X^*, Y) to $\hat{P}$, with

$$X_i^* = a_A(\bar{X}, \bar{Y})_j \text{ with } \bar{X}_j = X_i. \tag{9}$$

That is, we identify for each coordinate of C the respective consensus data with equal x-coordinate and replace the x-coordinate with the aligned value based on A. The curves in P' are treated analogously such that it holds $|\hat{P}| = |P| + |P'|$.

Note, each tested individual landmark pair shifts curve coordinates (within the current search interval) and thus changes the respective slope values. Since we are optimizing a slope-based distance (Eq. 1), we can therefore not precompute or reuse distance data from previous computation steps. This prohibits the application of a dynamic programming approach, as e.g. used in the `dtw` package [15], to find an optimal solution of the problem, but can be solved by the introduced heuristic divide-and-conquer optimization strategy used within

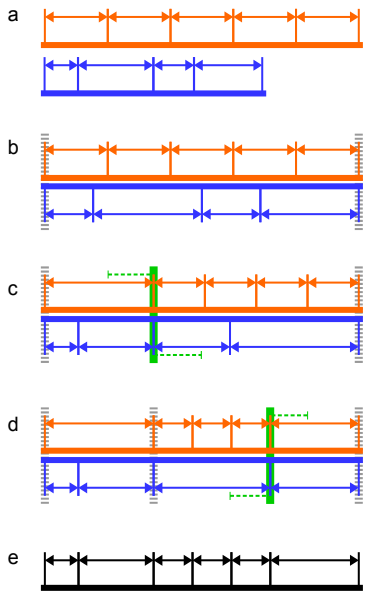

Figure 1: Depictions of the PICA workflow to align two curves (orange/blue) and to derive an according consensus profile (black). To simplify the presentation, only the x-ranges (double arrows in orange/blue) of intervals defined by alignable landmarks (vertical ticks) are shown. Here, we assume all landmarks are of same type and can be mapped. a) Input are two curves that might differ in length/x-range as well as number of landmarks. b) Initially, start and end of the curves are aligned (dashed grey bars) and the x-coordinates adapted via linear interpolation. c) For each pair of mappable landmarks, an according alignment is computed that shifts the aligned landmarks to their mean x-coordinate (green bar) and linearly interpolates the curve up to the next aligned pair (dashed grey bars) and the respective distance of the altered curves is computed. d) The best alignment from (c) is fixed (dashed grey bar), which decomposes the problem into two independent ones (left and right of last fixation up to next aligned pair). For each subproblem, (c+d) is applied until either no landmark pairs can be mapped or an alignment does not lower the curves' distance. e) Given (d) as the final alignment, the consensus curve (black) is compiled. It contains one coordinate for each present in one of the curves in the alignment from (d) where the y-coordinate is derived as the according mean of both curves.

PICA.

PICA supports various constraints to further guide the alignment process. Among them are the *minimal length/x-range of an interval* to be considered for further decomposition by alignment, a *maximally allowed x-shift* of aligned landmarks, and a *maximal interval length warping* to restrict the distortions resulting from landmark alignments.

MICA workflow

Given the PICA workflow, the progressive alignment scheme of MICA is introduced for a set of curve sets $\mathcal{P}$ that represents the input curves $C^1, \ldots, C^k$ or already aligned subsets.

First, a consensus profile $\bar{P}$ is computed for each curve set $P \in \mathcal{P}$. This is automatically annotated with landmark information $\bar{L}$ and filtered according to user given constraints. Such constraints are a minimal relative y-distance of neighbored maxima/minima (to reduce noise in the data and to focus on dominant optima) or a minimal relative slope of inflection points. Next, the two curve sets $P, P' \in \mathcal{P}$ are identified that show the lowest PICA distance $d_{\mathrm{PICA}}(P, P')$, i.e.

$$\arg\min_{P,P' \in \mathcal{P}} d_{\mathrm{PICA}}(P, P'), \tag{10}$$

where $d_{\mathrm{PICA}}(P, P')$ resembles the final distance according to the best landmark alignment produced by PICA. Finally, both curve sets are removed from $\mathcal{P}$ and replaced by the respective PICA aligned curve set $\hat{P}$. Furthermore, an according consensus profile with filtered landmark annotations is computed for $\hat{P}$.

This procedure is repeated until $\mathcal{P}$ is a singleton. The curves $C^* \in P \in \mathcal{P}$ are then reflecting the final alignment of the input curves $C^1, \ldots, C^k$ and are provided as MICA's output.

In addition to the sketched MICA workflow, which does an unguided alignment, MICA also supports the alignment of curves against a selected reference profile. In contrast to the depicted workflow, the x-coordinates of the reference are not altered and kept static during the PICA alignment. Thus, only the data points of the remaining profiles are shifted. Technically, this only changes the

computation of the shifted/aligned x-value x^* (Eq. 8), which now distinguishes whether or not the reference profile is represented by one of the curves. If so, x^* is set to the respective original value; otherwise the given equation is used. Such an alignment mode is of importance when e.g. extrapolating annotations available for a single profile to other related curves [7].

Implementation details

The MICA implementation is based on Java 8 and established libraries like *Apache Commons Math* and *Lang* as well as *JOpt Simple*. Extensive unit tests of the core algorithm and utility classes are set up via *JUnit*. The architecture follows a strict separation of algorithm classes and the different application interface implementations to enable a high level of modularity and maintainability.

User Interfaces

MICA provides different user interfaces to cover different use cases how curve alignments are generated.

Graphical user interface

The graphical user interface (GUI), shown in Fig. 2 and implemented in Java, enables an interactive usage of MICA. When loaded from the command-line, the default parameters can be altered by according command-line arguments. The GUI enables the load of profile data in the common CSV format, whereby different CSV format parameters like separator etc. can be set. The input profiles are directly visualized and can be interactively inspected using standard zoom or drag functionality. After setting the MICA parameters and starting the alignment, the GUI is automatically updated with the alignment in a second screen, to enable a comparison of the initial profiles with their aligned versions. The instant update assists in the inspection of the effects of different filter and parameter setups. The final alignment data can be exported in CSV format for

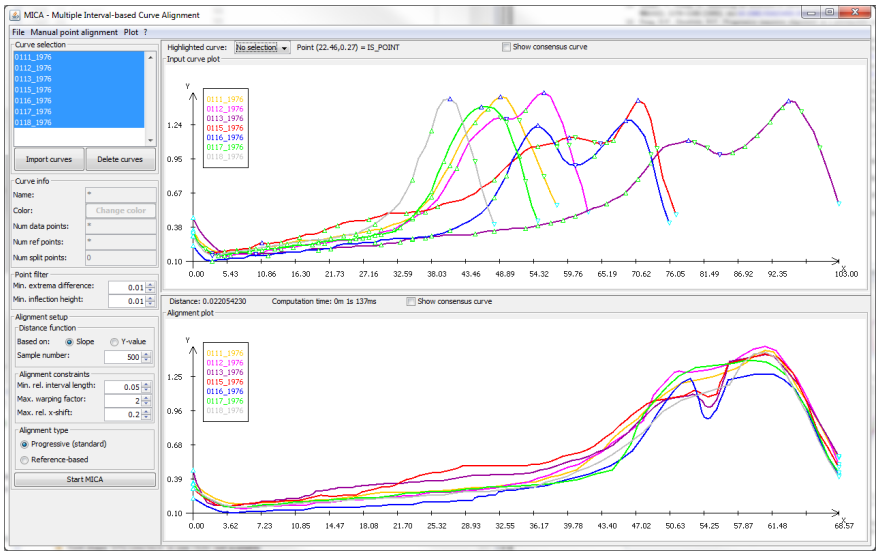

Figure 2: MICA's graphical user interface provides (top right) a depiction of the input profiles highlighting the identified and filtered alignable points, (bottom right) the final curve alignment, and (left) profile information and MICA options.

further processing and analyses. Furthermore, images of the input and output visualization can be exported.

All landmark registration approaches have in common that their performance heavily depends on the correct identification and filtering of the landmarks available for alignment. While MICA supports various constraints to control the automatic landmark identification, it still can be improved by expert knowledge where available. To this end, the GUI allows for the interactive prealignment of the curves prior to the automatic MICA alignment. The manually selected data point alignments are in the following treated the same as if aligned by MICA itself, i.e. they define boundaries for intervals that might be further decomposed. This feature is of importance if the data set contains curves that differ much from other curves. Here, manual alignment based on expert knowledge can guide the correct alignment of such outliers.

Another use case of the GUI is the manual tuning of parameters and constraints for a subsequent automated MICA application to a large number of data

sets to be aligned. That is, suitable parameters are first manually identified on a subset of the data using the GUI and later used within a high-throughput automatic application of MICA via its command-line interface for all data sets. Detailed GUI documentation is provided on MICA's github page (see Availability section).

Command-line interface

Due to the large available amount of biological data, bioinformatics analyses are often automated. To this end, different tools and filters are combined into according pipelines, e.g. using systems like Galaxy [16] or by simple scripts. MICA can also be embedded into high-throughput pipelines via its command-line interface (CLI). The CLI enables the setup of all MICA parameters including landmark filters. Input and output (CSV format) can either be read from or written to files or respective streams can be used. The latter is especially useful for the direct piping of temporary output into downstream processing without the (delaying) generation of temporary files.

R interface

Another similar use case is the integration of MICA into `R`-based analyses. The `R` framework [17] is a common platform to do semi-automated analyses and investigation of all type of data. MICA's `R` interface is based on the `rJava` package [18], which enables the use of native Java data structures from within `R`. This way, the computationally most demanding steps of MICA are done within the more efficient Java environment while the analyses and visualization can be done in the easy-to-use `R` framework [7]. The simple interface comes along with a small set of utility functions for data pre-/post-processing as well as data interpolation. According documentation is provided on the github page (see Availability section) or within the `R` interface sources.

Related packages

Within the `R` framework, other curve alignment approaches are available. The `fda` package [19, 20] offers for instance the landmark registration imple-

mentation `landmarkreg`, which enables multiple curve alignments for a given set of landmarks. The major drawbacks of `landmarkreg` are that landmarks have to be provided (no automatic annotation available as in MICA) and that for each curve the same number of landmarks are to be given (while MICA aligns only a suitable subset). Thus, the package authors refer to `register.fd`, a continuous registration function also part of the `fda` package, if the landmarks to be aligned are unknown. Unfortunately, the `register.fd` function requires a template profile to align the provided curves to. If no such profile is provided, the arithmetic mean curve of all input curves is taken as template. This might result in a poor template (showing not necessarily the common characteristics) if the input curves are heavily distorted or shifted, as e.g. the case for wood density data [6].

The `dtw` package [15] joins various "dynamic time warping" (DTW) algorithms in one generic implementation. DTW approaches [1] try to find an optimal mapping of all data points for two given profiles. One of the profiles is used as template (data points fixed). The optimization is done via dynamic programming, similar to a standard sequence alignment, while using dedicated scoring functions (e.g. without cumulative gap costs etc.) [8, 15]. Since the `dtw` package is tailored for reference-based, pairwise curve alignments, it can neither be easily used for multiple curve alignments nor for an alignment where no reference template is available. The latter is intrinsically the case, if the identification of a representative consensus profile (which would be the ideal template) is to be computed via a multiple curve alignment [6]. Wang and Gasser [21] proposed an iterative DTW-based approach for consensus profile computation, but no implementation is available.

Successful Applications

An earlier R-based implementation of the MICA approach was introduced in [6] and evaluated on a large data set of intra-annual wood density profiles similar to the data discussed above. The study investigated the effect of the MICA application in comparison to non-aligned simple mean profile derivation

of the initial curves. It was shown that the application of MICA significantly reduced the sampling error of the slopes. The resulting MICA consensus profiles well represented the common characteristics of the input curves, which were often lost when aggregating initial profiles without alignment. For instance, sharper curve peaks representing clear environmental signals are observed due to according alignment.

MICA was recently used in [7] to setup a protocol for a better understanding of the environmental control of wood formation during the growing season. The study used xylogenesis data via micro-core wood sampling and dendrometer monitoring to convert spatial scales of wood anatomical profiles to seasonal time scales. The comparison of the MICA aligned profiles in spatial annotation and the temporally annotated profiles demonstrated that MICA contributed significantly to increase the synchronicity of averaged wood anatomical characteristics.

In addition, it was shown in [7] that MICA could be used to extrapolate information. Here, temporal annotation of tree-ring development was derived based on experimental field data, which can only be collected for a few trees due to laborious and expensive methodology and time consuming sample preparation needed. Nevertheless, this data can be extrapolated to other tree samples (with similar growth characteristics) using the reference-based alignment mode of MICA. After alignment, the temporal annotation of the reference curve can be transferred to the newly aligned curves. The temporally annotated curves and the derived consensus profile eventually allowed a detailed understanding on how intra-seasonal drought periods modify intra-annual wood formation dynamics and cell-anatomical variables within tree-rings.

Discussion and Conclusions

Curve alignment is a central step to generate representative consensus data for a given set of e.g. measured discrete time series. If it is possible to identify prominent characteristics (landmarks) that are common among the curves

to be aligned, landmark registration approaches can be used. The latter identify an optimal time point (and according interval) mapping for the identified landmarks.

MICA implements a heuristic landmark registration method in combination with a progressive alignment scheme to generate multiple curve alignments and according representative consensus data. In contrast to available implementations, it automatically identifies landmarks for a given filter setup and generates an alignment without a predefined reference curve. The latter, i.e. a fixed reference, is a common prerequisite for available methods. While not mandatory, also reference-based alignment is possible using MICA. To face the varying numbers of identified landmarks for the curves to align, MICA does not enforce the mapping of all landmarks provided but identifies a subset mapping that minimizes the slope or amplitude difference between curves.

Generally, landmark registration works best if only few and very prominent characteristics are identified and used for alignment. Thus, a preprocessing of the data to smooth low amplitude fluctuations and other noise artefacts, e.g. due to measurement precision, can ease the alignment problem and increase the overall quality [6]. Since such preprocessing is non-trivial and very much depends on the data at hand, MICA does not provide any smoothing functionality.

Automatic handling of outlier curves among a data set is also a feature not part of the current MICA tool. While the use of the MICA GUI will easily help to identify curves that differ much in shape or other curve features, outliers will be part of the alignment and thus might reduce the alignment quality. An according outlier example data set is provided on github. One strategy could be to simply ignore the curve most dissimilar to all other curves based on average PICA scores. Such information is available via the `R` interface, which provides the pairwise distance table that guides the progressive alignment. Another strategy would be to investigate the guide tree (fusion order) of the alignment for outliers, which is also available via the `R` interface.

To further speedup the implementation, we are currently investigating the impact of a parallelized computation of the individual interval optimizations,

since once separated by mapped landmarks, they represent independent sub-
385 problems. Furthermore, we are investigating whether a smoothing of the warping function [10, 11] shows a significant impact on the heuristic alignment results.

So far, MICA was successfully applied to derive representative consensus profiles for wood density and cell growth data [6, 7]. Its graphical user interface enables ad hoc usage while its command-line and `R` interface are tailored for its
390 automated application in data processing pipelines of arbitrary discrete time series.

Availability of data and materials

The source code of MICA as well as precompiled binary for direct usage are freely available at https://github.com/BackofenLab/MICA. Here, also manuals
395 and the R-interface scripts are provided.

Acknowledgments

This work was supported by the German Research Foundation (DFG) with grants [*SP-437/19*] and [*BA 2168/12*] to HS and RB, resp., co-authored by HPK. The article processing charge was funded by the German Research Foun-
400 dation (DFG) and the University of Freiburg in the funding programme Open Access Publishing.

We thank D.F. Stangler for continuous testing of MICA and R. Schmitt for discussions on registration methods.

[1] J. O. Ramsay, X. Li, Curve registration, Journal of the Royal Statistical
405 Society: Series B (Statistical Methodology) 60 (2) (1998) 351–363. `doi:10.1111/1467-9868.00129`.

[2] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics, Speech, and Signal Processing 26 (1) (1978) 43–49. `doi:10.1109/TASSP.1978.1163055`.

[3] R. Tang, H.-G. Müller, Time-synchronized clustering of gene expression trajectories, Biostatistics 10 (1) (2009) 32. doi:10.1093/biostatistics/kxn011.

[4] F. Hermans, E. Tsiporkova, Merging microarray cell synchronization experiments through curve alignment, Bioinformatics 23 (2) (2007) e64. doi:10.1093/bioinformatics/btl320.

[5] L. M. Sangalli, P. Secchi, S. Vantini, V. Vitelli, k-mean alignment for curve clustering, Computational Statistics & Data Analysis 54 (5) (2010) 1219 – 1233. doi:http://doi.org/10.1016/j.csda.2009.12.008.

[6] B. Bender, M. Mann, R. Backofen, H. Spiecker, Microstructure alignment of wood density profiles: an approach to equalize radial differences in growth rate, Trees - Structure and Function 26 (4) (2012) 1267–1274. doi:10.1007/s00468-012-0702-y.

[7] D. Stangler, M. Mann, H.-P. Kahle, E. Rosskopf, S. Fink, H. Spiecker, Spatiotemporal alignment of radial tracheid diameter profiles of submontane norway spruce, Dendrochronologia 37 (2016) 33–45. doi:10.1016/j.dendro.2015.12.001.

[8] P. Tormene, T. Giorgino, S. Quaglini, M. Stefanelli, Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation, Artificial Intelligence in Medicine 45 (1) (2009) 11–34. doi:10.1016/j.artmed.2008.11.007.

[9] S. Boudaoud, H. Rix, O. Meste, Core shape modelling of a set of curves, Computational Statistics & Data Analysis 54 (2) (2010) 308 – 325. doi:http://doi.org/10.1016/j.csda.2009.08.003.

[10] A. Kneip, T. Gasser, Statistical tools to analyze data representing a sample of curves, Ann. Statist. 20 (3) (1992) 1266–1305. doi:10.1214/aos/1176348769.

[11] T. Gasser, A. Kneip, Searching for structure in curve sample, Journal of the American Statistical Association 90 (432) (1995) 1179–1188. doi:10.1080/01621459.1995.10476624.

[12] D. F. Feng, R. F. Doolittle, Progressive sequence alignment as a prerequisite to correct phylogenetic trees, Journal Mol Evol 25 (4) (1987) 351–60. doi:10.1007/BF02603120.

[13] J. D. Thompson, D. G. Higgins, T. J. Gibson, CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice, Nucleic acids research 22 (22) (1994) 4673–4680. doi:10.1093/nar/22.22.4673.

[14] M. G. Schinker, N. Hansen, H. Spiecker, High-frequency densitometry - a new method for the rapid evaluation of wood density variations, IAWA Journal 24 (3) (2003) 231–239. doi:https://doi.org/10.1163/22941932-90001592.

[15] T. Giorgino, Computing and visualizing dynamic time warping alignments in R: The dtw package, Journal of Statistical Software 31 (1) (2009) 1–24. doi:10.18637/jss.v031.i07.

[16] E. Afgan, D. Baker, M. van den Beek, D. Blankenberg, D. Bouvier, M. Cech, J. Chilton, D. Clements, N. Coraor, C. Eberhard, B. Grüning, A. Guerler, J. Hillman-Jackson, G. Von Kuster, E. Rasche, N. Soranzo, N. Turaga, J. Taylor, A. Nekrutenko, J. Goecks, The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update, Nucleic Acids Research 44 (W1) (2016) W3. doi:10.1093/nar/gkw343.

[17] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2016).
URL https://www.R-project.org/

[18] S. Urbanek, rJava: Low-Level R to Java Interface, r package version 0.9-8 (2016).
URL https://CRAN.R-project.org/package=rJava

[19] J. O. Ramsay, H. Wickham, S. Graves, G. Hooker, fda: Functional Data Analysis, r package version 2.4.4 (2014).
URL https://CRAN.R-project.org/package=fda

[20] J. O. Ramsay, B. W. Silverman, Functional Data Analysis. 2nd ed., Series in Statistics, Springer, Berlin Heidelberg, 2005. doi:10.1007/b98888.

[21] K. Wang, T. Gasser, Synchronizing sample curves nonparametrically, The Annals of Statistics 27 (2) (1999) 439–460. doi:10.1214/aos/1018031202.

Nr	**(executable) Software metadata description**	**Please fill in this column**
S1	Current software version	2.0.1
S2	Permanent link to executables of this version	https://github.com/BackofenLab/MICA/releases
S3	Legal Software License	MIT
S4	Computing platform / Operating System	Linux, OS X, Microsoft Windows, Unix-like
S5	Installation requirements & dependencies	Java 8
S6	If available Link to user manual - if formally published include a reference to the publication in the reference list	https://github.com/BackofenLab/MICA

Table 1: Table 1 - Software metadata

Nr	Code metadata description	Please fill in this column
C1	Current Code version	2.0.1
C2	Permanent link to code / repository used of this code version	`https://github.com/BackofenLab/MICA`
C3	Legal Code License	MIT
C4	Code Versioning system used	git
C5	Software Code Language used	Java, R
C6	Compilation requirements, Operating environments & dependencies	Java, R
C7	If available Link to developer documentation / manual	`https://github.com/BackofenLab/MICA`
C8	Support email for questions	`https://github.com/BackofenLab/MICA/issues`

Table 2: Table 2 - Code metadata