Supplementary Note 1: comparing two trajectories

Contents

Metrics to compare two trajectories

A trajectory, as defined in our evaluation, is a model with multiple abstractions. The top abstraction is the topology which contains information about the paths each cell can take from their starting point. Deeper abstractions involve the mapping of each cell to a particular branch within this network, and the position (or ordering) of each cells within these branches. Internally, the topology is represented by the milestone network and regions of delayed commitment, the branch assignment and cellular positions are represented by the milestone percentages (**Figure 1**).

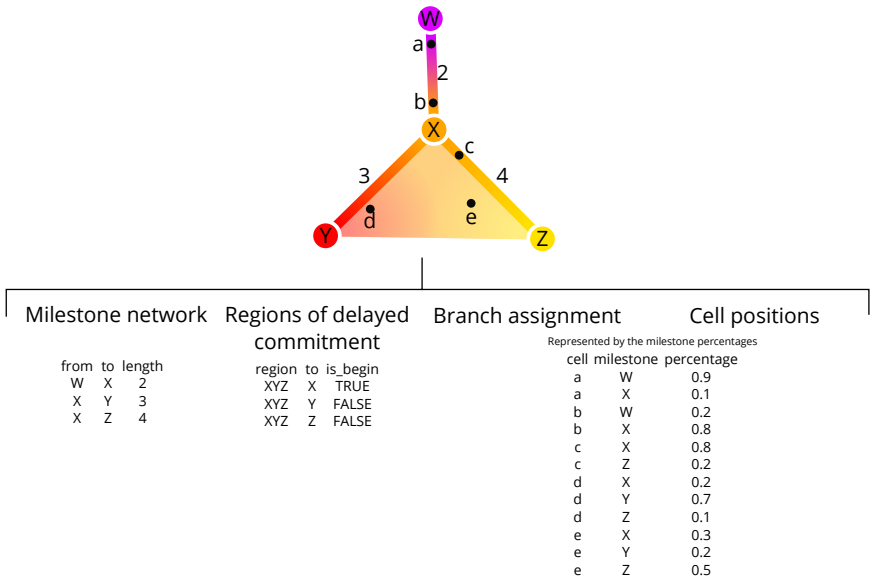


Figure 1: An example trajectory that will be used throughout this section. It contains contains four milestones (W to Z) and five cells (a to e).

Given the multilayered complexity of a trajectory model, it is not trivial to compare the similarity of two trajectory models using only one metric. We therefore sought to use different comparison metrics, each serving a different purpose:

- **Specific metrics** investigate one particular aspect of the trajectory. Such metrics make it possible to find particular weak points for methods, e.g. that a method is very good at ordering but does not frequently find the correct topology. Moreover, having multiple individual metrics allow personalised rankings of methods, for example for users which are primarily interested in using the method correct topology.
- **Application metrics** focus on the quality of a downstream analysis using the trajectory. For example, it measures whether the trajectory can be used to find accurate differentially expressed genes.
- **Overall metrics** should capture all the different abstractions, in other words such metrics measure whether the resulting trajectory has a good topology, that the cells belong to similar branches *and* that they are ordered correctly.

Here, we first describe and illustrate several possible specific, application and overall metrics. Next, we test these metrics on several test cases, to make sure they robustly identify "wrong" trajectory predictions.

All metrics described here were implemented within the *dyneval* R package (https://github.com/dynverse/dyneval).

Metric characterisation and testing

Specific metrics

isomorphic, edgeflip and HIM: Edit distance between two trajectory topologies

We used three different scores to assess the similarity in the topology between two trajectories, iregardless of where the cells were positioned.

For all three scores, we first simplified the topology of the trajectory to make both graph structures comparable:

- As we are only interested in the main structure of the topology without start or end, the graph was made undirected.
- All milestones with degree 2 were removed. For example in the topology A ⇨ B ⇨ C ⇨ D, C ⇨ D, the B milestone was removed
- A linear topology was converted to A ⇨ B ⇨ C
- A cyclical topology such as A ⇨ B ⇨ C ⇨ D or A ⇨ B ⇨ A were all simplified to A ⇨ B ⇨ C ⇨ A
- Duplicated edges such as A ⇨ B, A ⇨ B were decoupled to A ⇨ B, A ⇨ C ⇨ B

The isomorphic score returns 1 if two graphs are isomorphic, and 0 if they were not. For this, we used the used the BLISS algorithm[1], as implemented in the R *igraph* package.

The edgeflip score was defined as the minimal number of edges which should be added or removed to convert one network into the other, divided by the total number of edges in both networks. This problem is equivalent to the maximum common edge subgraph problem, a known NP-hard problem without a scalable solution[2]. We implemented a branch and bound approach for this problem, using several heuristics to speed up the search:

- First check all possible edge additions and removals corresponding to the number of different edges between the two graphs.
- For each possible solution, first check whether:
  - The maximal degree is the same
  - The minimal degree is the same
  - All degrees are the same after sorting
- Only then check if the two graphs are isomorphic as described earlier.
- If no solution is found, check all possible solutions with two extra edge additions/removals.

The HIM metric (Hamming-Ipsen-Mikhailov distance)[3] which was adopted from the R nettools package (https://github.com/filosi/nettools). It uses an adjacency matrix which was weighted according to the lengths of each edges within the milestone network. Conceptually, HIM is a linear combination of:

- The normalised Hamming distance[4], which calculates the distance between two graphs by matching individual edges in the adjacency matrix, but disregards overall structural similarity.
- The normalised Ipsen-Mikhailov distance[5], which calculates the overall distance of two graphs based on matches between its degree and adjacency matrix, while disregarding local structural similarities. It requires a γ parameter, which is usually estimated based on the number of nodes in the graph, but which we fixed at 0.1 so as to make the score comparable across different graph sizes.

We compared the three scores on several common topologies (**Figure 2**). While conceptually very different, the edgeflip and HIM still produce similar scores (**Figure 2b**). The HIM tends to punish the detection of cycles, while the edgeflip is more harsh for differences in the number of bifurcations (**Figure 2b**). The main difference however is that the HIM takes into account edge lengths when comparing two trajectories, as illustrated in (**Figure 2c**). Short “extra” edges in the topology are less punished by the HIM than by the edgeflip.

To summarise, the different topology based scores are useful for different scenarios:

- If the two trajectories should only be compared when the topology is exactly the same, the isomorphic should be used.
- If it is important that the topologies are similar, but not necessarily isomorphic, the edgeflip is most appropriate.
- If the topologies should be similar, but shorter edges should not be punished as hard as longer edges, the HIM is most appropriate.

$F1_{branches}$ and $F1_{milestones}$: Comparing how well the cells are clustered in the trajectory

Perhaps one of the simplest ways to calculate the similarity between the cellular positions of two topologies is by mapping each cell to its closest milestone *or* branch (**Figure 3**). These clusters of cells can then be compared using

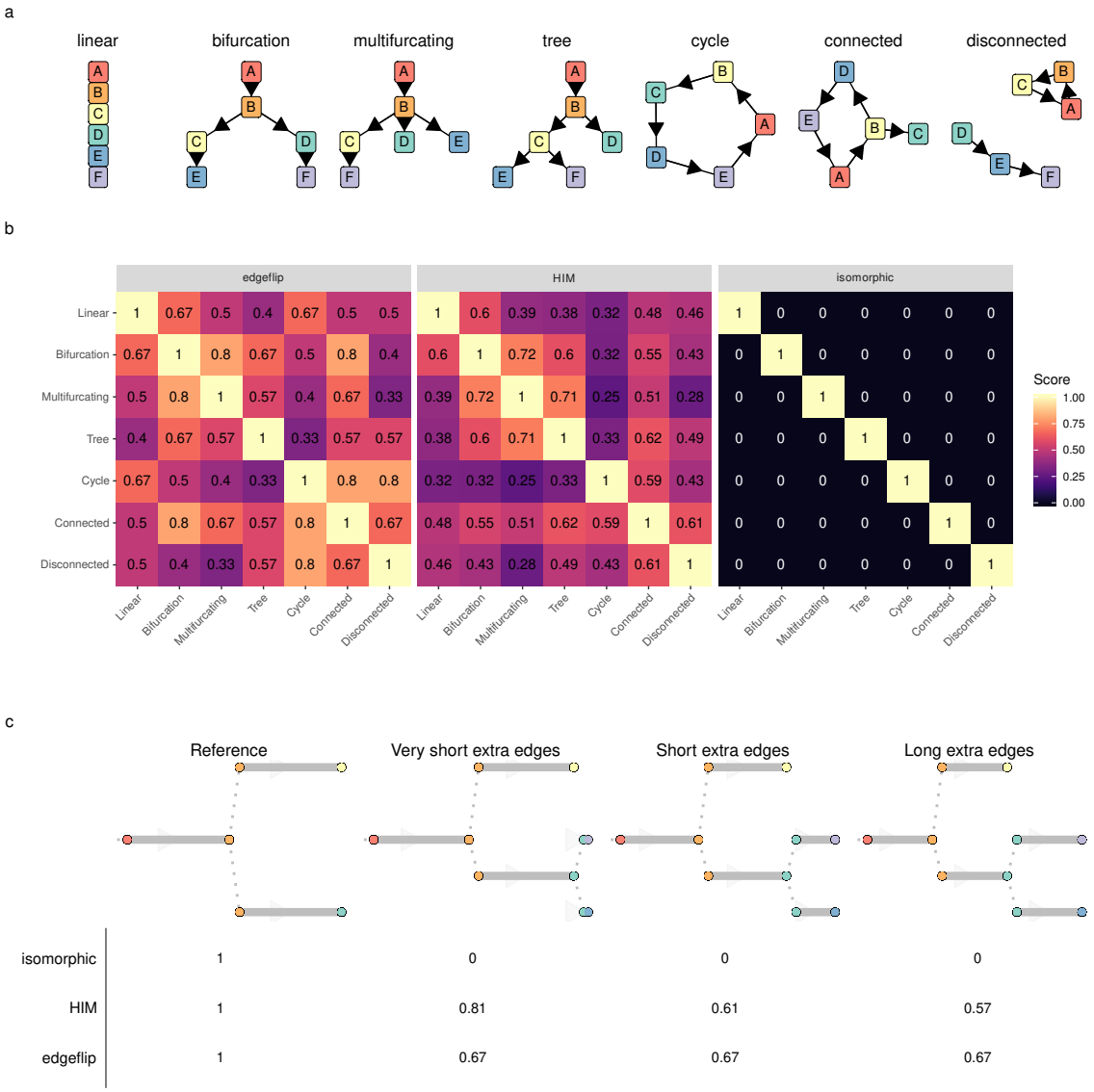


Figure 2: Showcase of three metrics to evaluate topologies: isomorphic, edgeflip and HIM. (a) The used topologies. (b) The scores when comparing each pair of trajectory types. (c) Four datasets in which aan extra edge is added and made progressively longer. This shows how the HIM can take into account edge lengths.

one of the many external cluster evaluation measures[6]. When selecting a cluster evaluation metric, we had two main conditions:

- Because we allow methods to filter cells in the trajectory, the metric should be able to handle "non-exhaustive assignment", where some cells are not assigned to any cluster.
- The metric should give each cluster equal weight, so that rare cell stages are equally important as large stages.

The F1 score between the Recovery and Relevance is a metric which conforms to both these conditions. This metric will map two clustersets by using their shared members based on the Jaccard similarity. It then calculates the Recovery as the average maximal Jaccard for every cluster in the first set of clusters (in our case the reference trajectory). Conversely, the Relevance is calculated based on the average maximal similarity in the second set of clusters (in our case the prediction). Both the Recovery and Relevance are then given equal weight in a harmonic mean (F1). Formally, if C and C' are two cell clusters:

$\text{Jaccard}(c, c') = \frac{|c \cap c'|}{|c \cup c'|}$

$\text{Recovery} = \frac{1}{|C|} \sum_{c \in C} \max_{c' \in C'} \text{Jaccard(c, c')}$

$\text{Relevance} = \frac{1}{|C'|} \sum_{c' \in C'} \max_{c \in C} \text{Jaccard(c, c')}$

$\text{F1} = \frac{2}{\frac{1}{\text{Recovery}} + \frac{1}{\text{Relevance}}}$

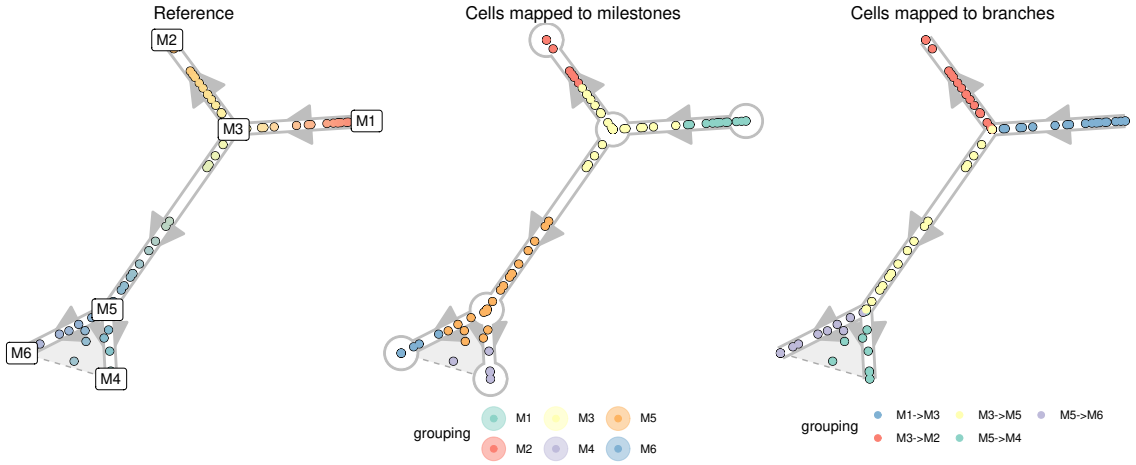


Figure 3: Mapping cells to their closest milestone or branch for the calculation of the $F1_{milestones}$ and $F1_{branches}$. To calculate the $F1_{milestones}$, cells are mapped towards the nearest milestone, i.e. the milestone with the highest milestone percentage. For the $F1_{branches}$, the cells are mapped to the closest edge.

cor_{dist}: Correlation between geodesic distances

When the position of a cell is the same in both the reference and the prediction, its *relative* distances to all other cells in the trajectory should also be the same. This observation is the basis for the cor_{dist} metric.

The geodesic distance is the distance a cell has to go through the trajectory space to get from one position to another. The way this distance is calculated depends on how two cells are positioned, showcased by an example in **Figure 4**:

- **Both cells are on the same edge in the milestone network.** In this case, the geodesic distance is defined as the product of the difference in milestone percentages and the length of their shared edge. For cells a and b in the example, $d(a, b)$ is equal to $1 \times (0.9 - 0.2) = 0.7$.
- **Cells reside on different edges in the milestone network.** First, the distance of the cell to all its nearby milestones is calculated, based on its percentage within the edge and the length of the edge. These distances in combination with the milestone network are used to calculate the shortest path distance between the two cells. For cells a and c in the example, $d(a, X) = 1 \times 0.9$ and $d(c, X) = 3 \times 0.2$, and therefore $d(a, c) = 1 \times 0.9 + 3 \times 0.2$.

The geodesic distance can be easily extended towards cells within regions of delayed commitment. When both cells are part of the same region of delayed commitment, the geodesic distance was defined as the manhattan distances between the milestone percentages weighted by the lengths from the milestone network. For cells d and e in the example, $d(d, e)$ is equal to $0 \times (0.3 - 0.2) + 2 \times (0.7 - 0.2) + 3 \times (0.4 - 0.1) = 1.9$. The distance between two

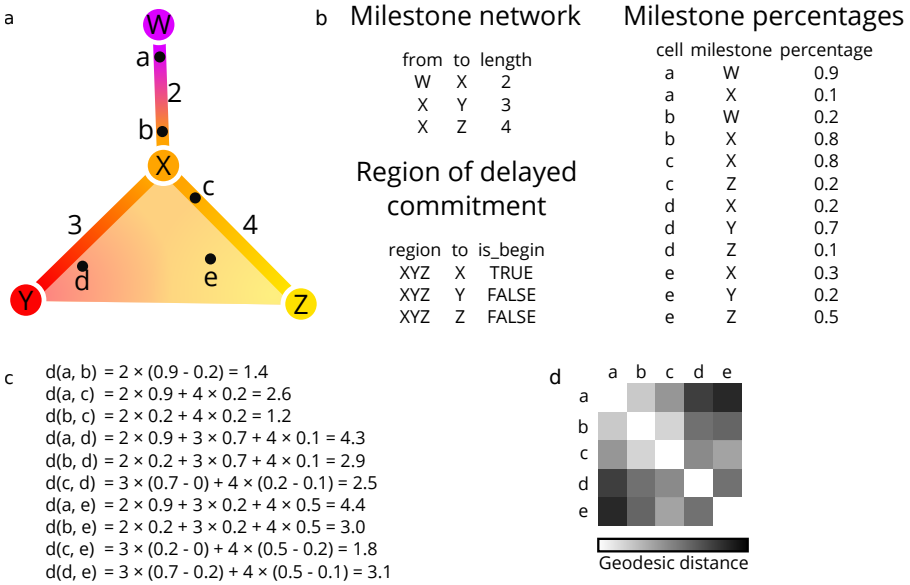


Figure 4: The calculation of geodesic distances on a small example trajectory. a) A toy example containing four milestones (W to Z) and five cells (a to e). b) The corresponding milestone network, milestone percentages and regions of delayed commitment, when the toy trajectory is converted to the common trajectory model. c) The calculations made for calculating the pairwise geodesic distances. d) A heatmap representation of the pairwise geodesic distances.

cells where only one is part of a region of delayed commitment is calculated similarly to the previous paragraph, by first calculating the distance between the cells and their neighbouring milestones first, then calculating the shortest path distances between the two.

Calculating the pairwise distances between cells scales quadratically with the number of cells, and would therefore not be scaleable for large datasets. For this reason, a set of waypoint cells are defined *a priori*, and only the distances between the waypoint cells and all other cells is calculated, in order to calculate the correlation of geodesic distances of two trajectories (**Figure 5a**). These cell waypoints are determined by viewing each milestone, edge and region of delayed commitment as a collection of cells. We do stratified sampling from each collection of cells by weighing them by the total number of cells within that collection. For calculating the cor_{dist} between two trajectories, the distances between all cells and the union of both waypoint sets is computed.

To select the number of cell waypoints, we need to find a trade-off between the accuracy versus the time to calculate cor_{dist}. To select an optimal number of cell waypoints, we used the synthetic dataset with the most complex topology, and determined the cor_{dist} at different levels of both cell shuffling and number of cell waypoints (**Figure 5b**). We found that using cell waypoints does not induce a systematic bias in the cor_{dist}, and that its variability was relatively minimal when compared to the variability between different levels of cell shuffling when using 100 or more cell waypoints.

Although the cor_{dist}'s main characteristic is that it looks at the positions of the cells, other features of the trajectory are also (partly) captured. To illustrate this, we used the geodesic distances themselves as input for dimensionality reduction (**Figure 6**) with varying topologies. This reduced space captures the original trajectory structure quite well, including the overall topology and branch lengths. Only some structures, not easily visualisabe

$NMSE_{rf}$ and $NMSE_{lm}$: Using the positions of the cells within one trajectory to predict the cellular positions in the other trajectory

An alternative approach to detect whether the positions of cells are similar between two trajectories, is to use the positions of one trajectory to predict the positions within the other trajectory. If the cells are at similar positions in the trajectory (relative to its nearby cells), the prediction error should be low.

Specifically, we implemented two metrics which predict the milestone percentages from the reference by using the predicted milestone percentages as features (**Figure 7**). We did this with two regression methods, linear regression (*lm*, using the R `lm` function) and Random Forest (*rf*, implemented in the *ranger* package[7]). In both cases, the accuracy of the prediction was measured using the Mean Squared error (MSE), in the case of Random forest we used the out-of-bag mean-squared error. Next, we calculated MSE_{worst} equal to the MSE when predicting all milestone percentages as the average. We used this to calculate the normalised mean squared error as $NMSE = 1 - \frac{MSE}{MSE_{worst}}$. We created a regression model for every milestone in the gold standard, and averaged the $NMSE$ values to finally obtain the

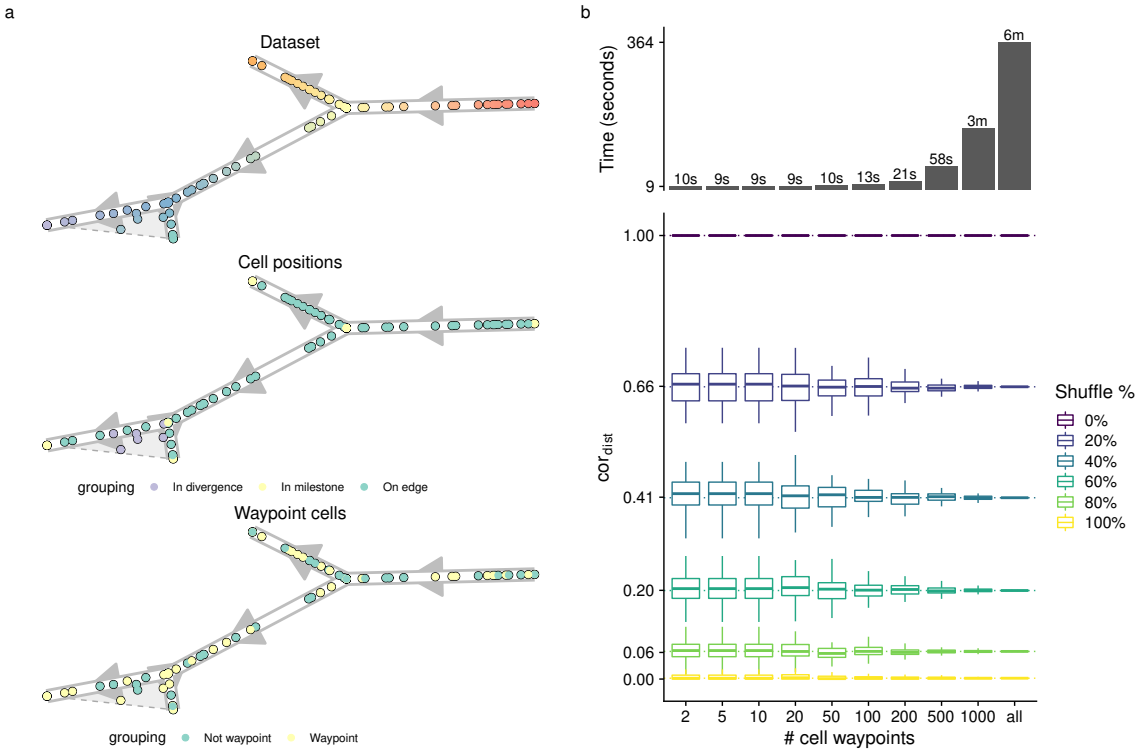


Figure 5: Determination of cell waypoints a) Illustration of the stratified cell sampling using an example dataset (top). Each milestone, edge between two milestones and region of delayed commitment is seen as a collection of cells (middle), and the number of waypoints (100 in this case) are divided over each of these collection of cells (bottom). b) Accuracy versus time to calculate cor_{dist}

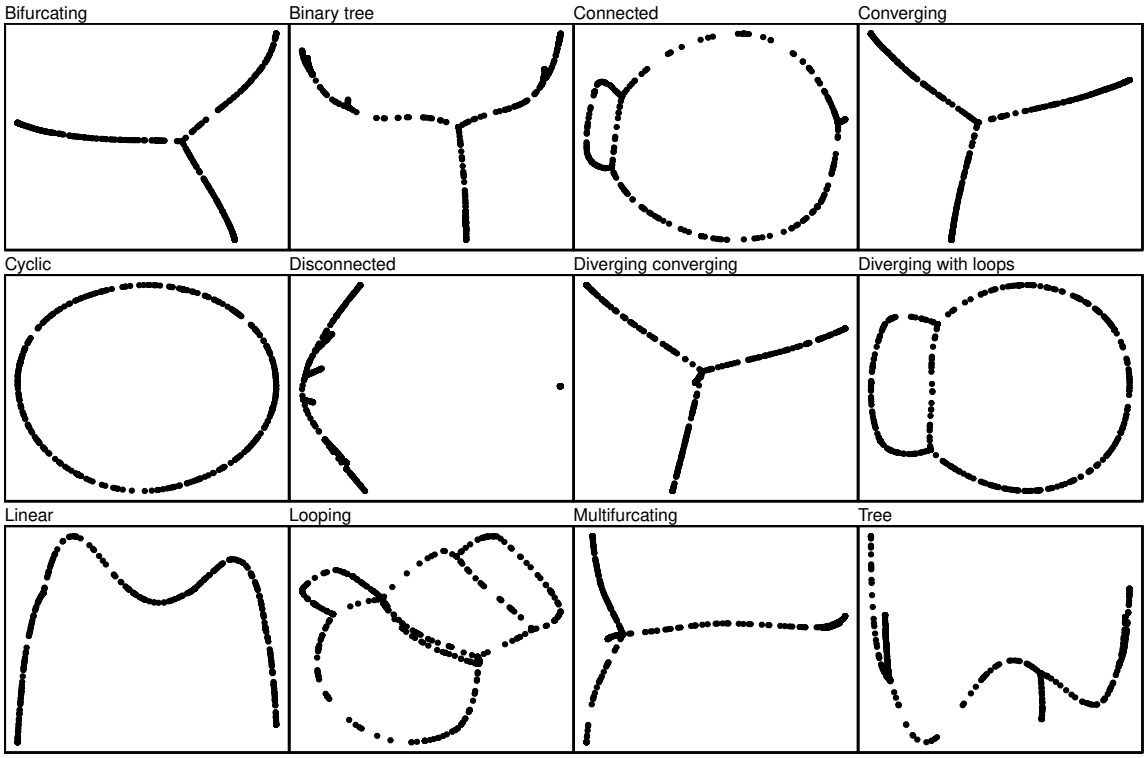


Figure 6: The geodesic distances can be used to reconstruct the original trajectory structure We generated different toy trajectory datasets with varying topologies and calculated the geodesic distances between all cells within the trajectory. We then used these distances as input for classical multidimensional scaling. This shows that the geodesic distances do not only contain information regarding the cell's positions, but also information on the lengths and wiring of the topology.

$NMSE_{rf}$ and $NMSE_{lm}$ scores.

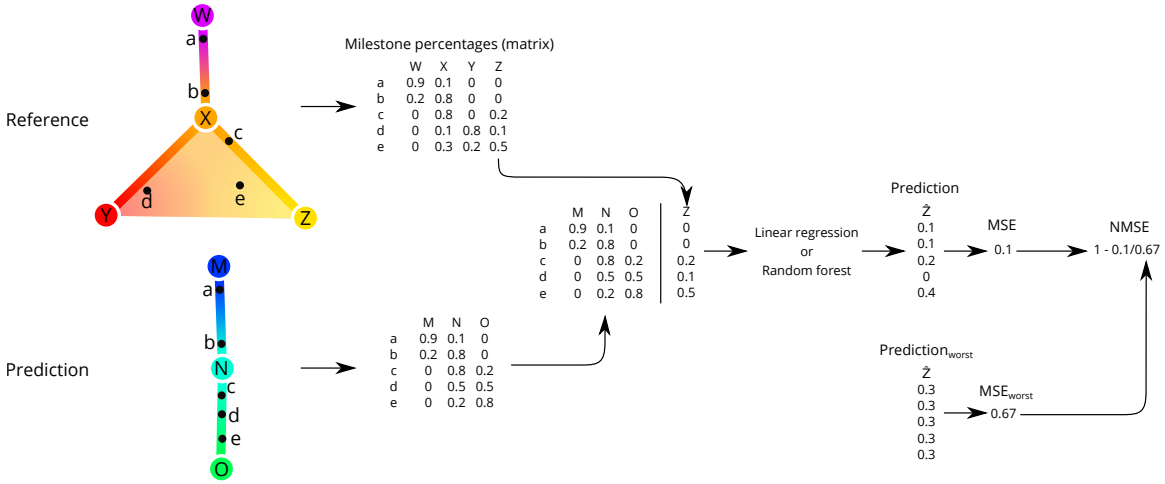


Figure 7: The calculation of $NMSE_{lm}$ distances on a small example trajectory. The milestone percentages of the reference are predicted based on the milestone percentages of the prediction, using regression models such as linear regression or random forests. The predicted trajectory is then scored by comparing the mean-squared error (MSE° of this regression model with the baseline MSE where the prediction is the average milestone percentage

Application metrics

Although most metrics described above already assess some aspects directly relevant to the user, such as whether the method is good at finding the right topology, these metrics do not assess the quality of downstream analyses and hypotheses which can be generated from these models.

cor_{features} and $wcor_{\text{features}}$: The accuracy of dynamical differentially expressed features/genes.

Perhaps the main advantage of studying cellular dynamic processes using single-cell -omics data is that the dynamics of gene expression can be studied for the whole transcriptome. This can be used to construct other models such as dynamic regulatory networks and gene expression modules. Such analyses rely on a "good-enough" cellular ordering, so that it can be used to identify dynamical differentially expressed genes.

To calculate the cor_{features} we used Random forest regression to rank all the features according to their importance in predicting the positions of cells in the trajectory. More specifically, we first calculated the geodesic distances for each cell to all milestones in the trajectory. Next, we trained a Random Forest regression model (implemented in the R *ranger* package[7], https://github.com/imbs-hl/ranger) to predict these distances for each milestone, based on the expression of genes within each cell. We then extracted feature importances using the Mean Decrease in Impurity (`importance = 'impurity'` parameter of the `ranger` function), as illustrated in (**Figure 8**). The overall importance of a feature (gene) was then equal to the mean importance over all milestones. Finally, we compared the two rankings by calculating the Pearson correlation, with values between -1 and 0 clipped to 0.

Random forest regression has two main hyperparameters. The number of trees to be fitted (`num_trees` parameter) was fixed to 10000 to provide accurate and stable estimates of the feature importance (**Figure 9**). The number of features on which can be split (`mtry` parameter) was set to 1% of all available features (instead of the default square-root of the number of features), as to make sure that predictive but highly correlated features, omnipresent in transcriptomics data, are not suppressed in the ranking.

For most datasets, only a limited number of features will be differentially expressed in the trajectory. For example, in the dataset used in **Figure 9** only the top 10%-20% show a clear pattern of differential expression. The correlation will weight each of these features equally, and will therefore give more weight to the bottom, irrelevant features. To prioritise the top differentially expressed features, we also implemented the $wcor_{\text{features}}$, which will weight the correlation using the feature importance scores in the reference so that the top features have relatively more impact on the score (**Figure 10**).

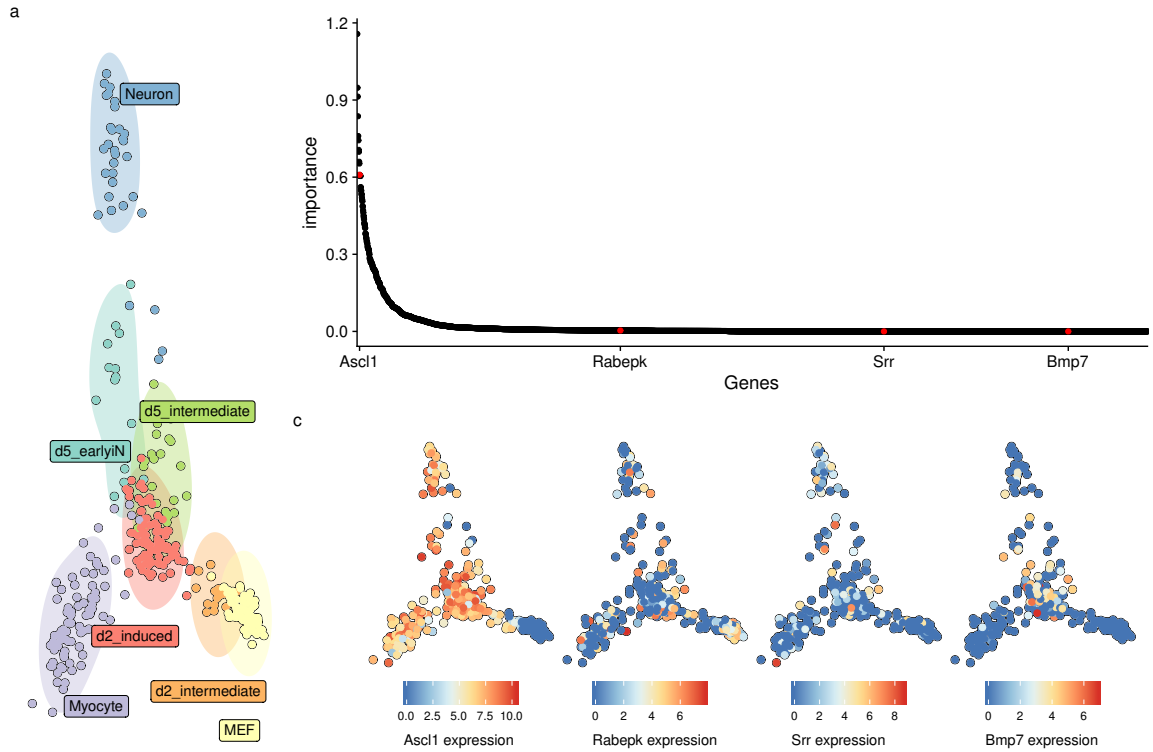


Figure 8: An illustration of ranking features based on their importance in a trajectory. (a) A MDS dimensionality reudction of a real dataset in which mouse embryonic fibroblasts (MEF) differentiate into Neurons and Myocytes. (b) The ranking of feature importances from high to low. The majority of features have a very low importance. (c) Some examples, which were also highlighted in b. Higher features in the ranking are clearly specific to certain parts of the trajectory, while features lower on the ranking have a more dispersed expression pattern.

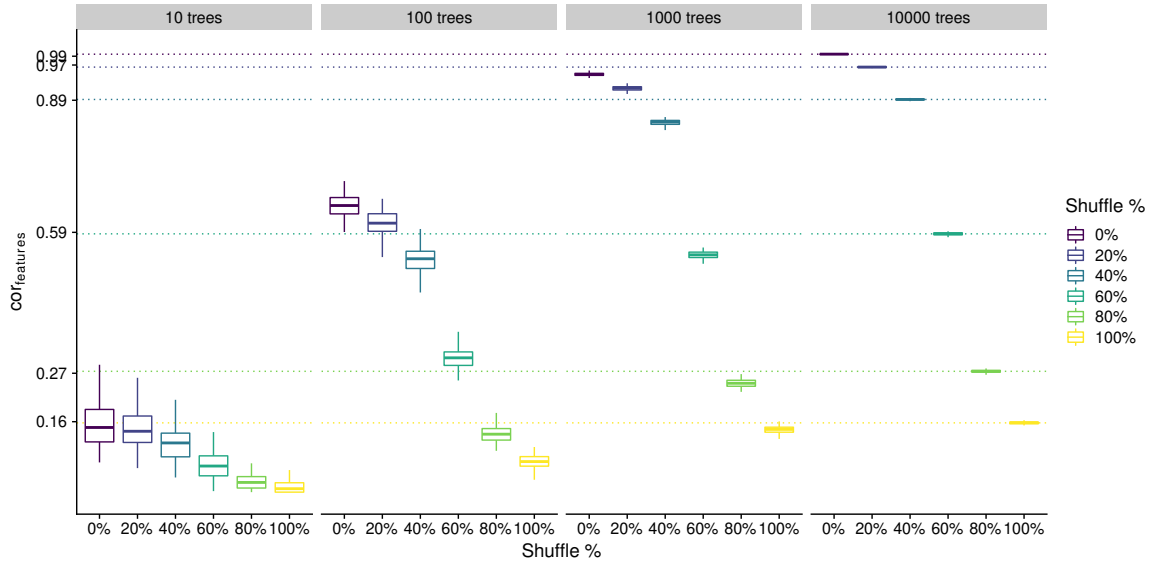


Figure 9: Effect of the number of trees parameter on the accuracy and variability of the cor_{features}. We used the dataset from **Figure 8** and calculated the cor_{features} after shuffling a percentage of cells.

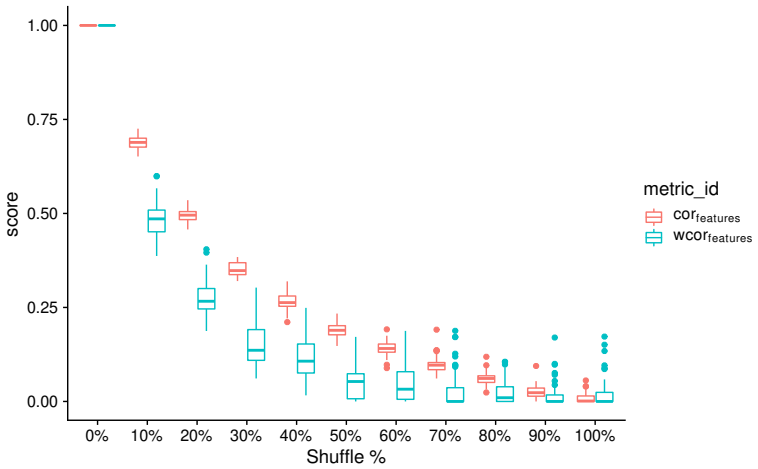


Figure 10: Effect of weighting the features based on their feature importance in the reference. We used the same dataset as in **Figure 8**, and calculated the cor_{features} after shuffling a percentage of cells.

Metric conformity

Although most metrics described in the previous section make sense intuitively, this does not necessarily mean that these metrics are robust and will generate reasonable results when used for benchmarking. This is because different methods and datasets will all lead to a varied set of trajectory models:

- Real datasets have all cells grouped onto milestones
- Some methods place all cells in a region of delayed commitment, others never generate a region of delayed commitment
- Some methods always return a linear trajectory, even if a bifurcation is present in the data
- Some methods filter cells

A good metric, especially a good overall metric, should work in all these circumstances. To test this, we designed a set of rules to which a good metric should conform, and assessed empirically whether a metric conforms to these rules.

We generated a panel of toy datasets (using our *dyntoy* package, https://github.com/dynverse/dyntoy) with all possible combinations of:

- # cells: 10, 20, 50, 100, 200 and 500
- # features: 200
- topologies: linear, bifurcation, multifurcating, tree, cycle, connected graph and disconnected graph
- Whether cells are placed on the milestones (as in real data) or on the edges/regions of delayed commitment between the milestones (as in synthetic data)

We then perturbed the trajectories in these datasets in certain ways, and tested whether the scores follow an expected pattern. An overview of the conformity of every metric is first given in **Table 1**. The individual rules and metric behaviour are discussed more into detail after that.

name	cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
Same score on identity	✔	✖	✔	✔	✔	✔	✖	✖	✔	✔	✔
Local cell shuffling	✔	✔	✔	✖	✖	✖	✔	✔	✖	✖	✔
Edge shuffling	✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔
Local and global cell shuffling	✔	✖	✖	✖	✖	✖	✔	✔	✔	✔	✔
Changing positions locally and/or globally	✔	✔	✔	✖	✖	✖	✔	✔	✖	✖	✔
Cell filtering	✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔
Removing divergence regions	✔	✔	✔	✖	✖	✖	✔	✔	✖	✔	✔
Move cells to start milestone	✔	✔	✔	✖	✖	✖	✔	✔	✖	✔	✔
Move cells to closest milestone	✔	✖	✔	✖	✖	✖	✔	✔	✖	✔	✔
Length shuffling	✔	✖	✖	✖	✖	✖	✖	✖	✖	✖	✔
Cells into small subedges	✖	✔	✖	✔	✔	✔	✖	✖	✔	✔	✔
New leaf edges	✖	✖	✖	✔	✔	✔	✔	✖	✔	✔	✔
New connecting edges	✔	✖	✖	✔	✔	✔	✔	✔	✔	✔	✔
Changing topology and cell position	✔	✖	✖	✖	✖	✖	✖	✖	✖	✖	✔
Bifurcation merging	✔	✖	✔	✔	✔	✔	✔	✔	✔	✔	✔
Bifurcation merging and changing cell positions	✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔
Bifurcation concatentation	✔	✖	✔	✔	✔	✔	✔	✔	✔	✔	✔
Cycle breaking	✔	✖	✔	✔	✔	✔	✔	✔	✖	✔	✔
Linear joining	✔	✔	✖	✔	✔	✔	✔	✔	✖	✔	✔
Linear splitting	✔	✔	✖	✔	✔	✔	✔	✔	✔	✔	✔
Change of topology	✔	✖	✖	✔	✔	✔	✔	✔	✖	✔	✔
Cells on milestones vs edges	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

Table 1: Overview of whether a particular metric conforms to a particular rule

Same score on identity

The score should be approximately the same when comparing the trajectory to itself

A metric conforms to this rule if: $0.99 \leqslant score \leqslant 1$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✔	✔	✔	✔	✖	✖	✔	✔	✔

Metrics which contain some stochasticity (random forest based metrics in particular), usually do not conform to this rule, even though their scores are still consistently high.

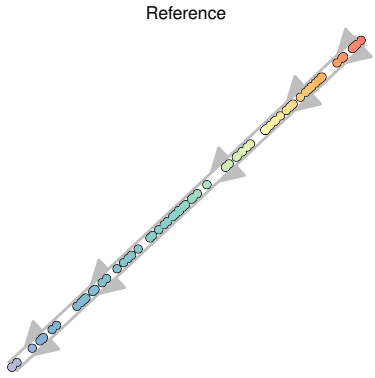


Figure 11: Example dataset(s) for this rule

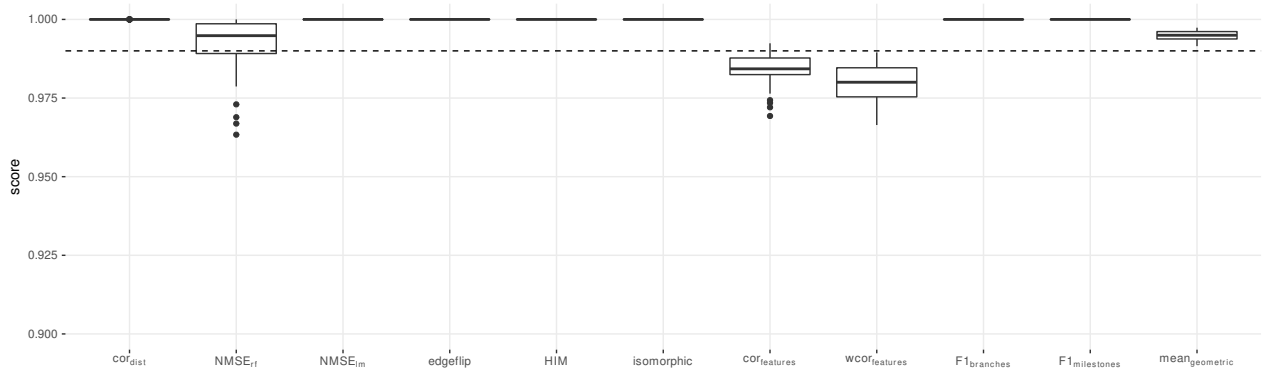
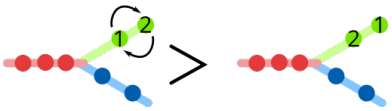


Figure 12: Scores for this rule.

Local cell shuffling



Shuffling the positions of cells within each edge should lower the score. This is equivalent to changing the cellular position locally.

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✖	✖	✔

Metrics which do not look at the cellular positioning, or group the cells within branches or milestones, do not conform to this rule.

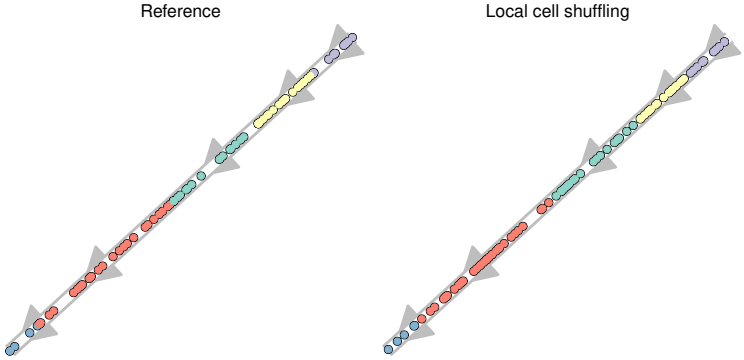


Figure 13: Example dataset(s) for this rule

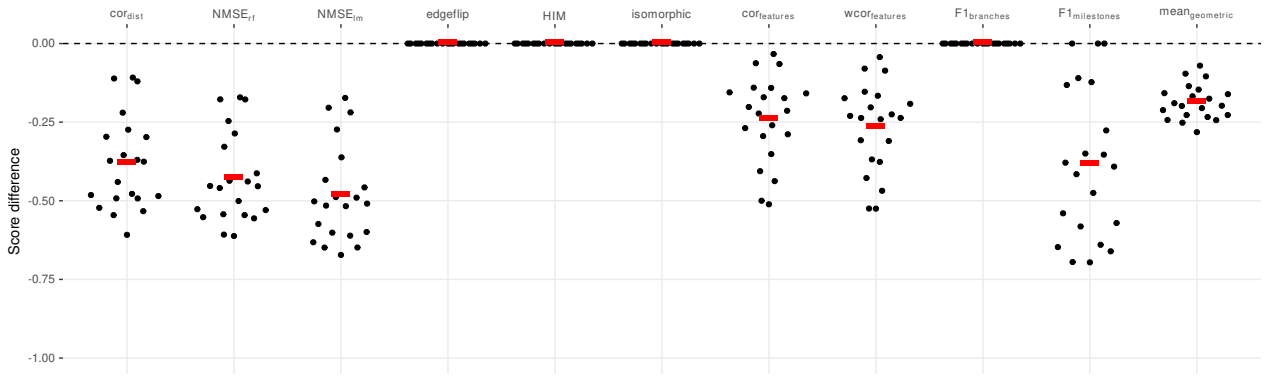
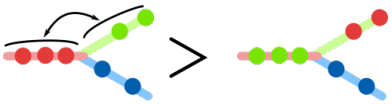


Figure 14: Scores for this rule.

Edge shuffling



Shuffling the edges in the milestone network should lower the score. This is equivalent to changing the cellular positions only globally.

A metric conforms to this rule if: $monotonic\left(\textit{shuffled edges}, \overline{score}_{\textit{shuffled edges}}\right)$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔

Metrics which only look at the topology do not conform to this rule.

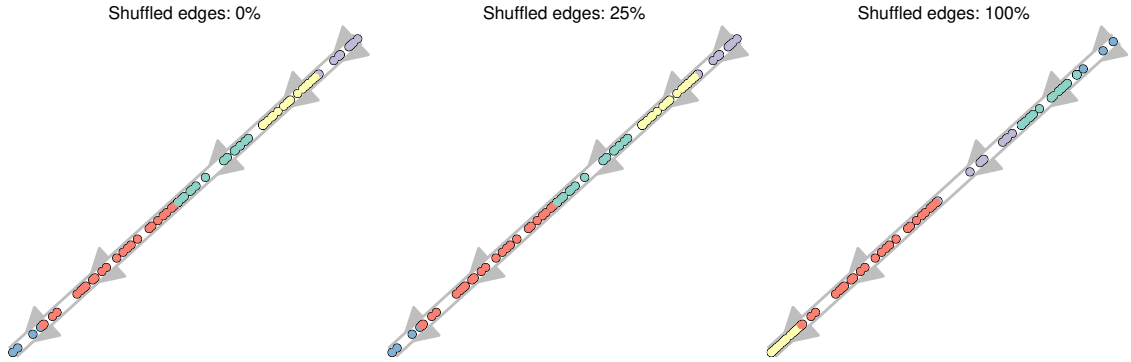


Figure 15: Example dataset(s) for this rule

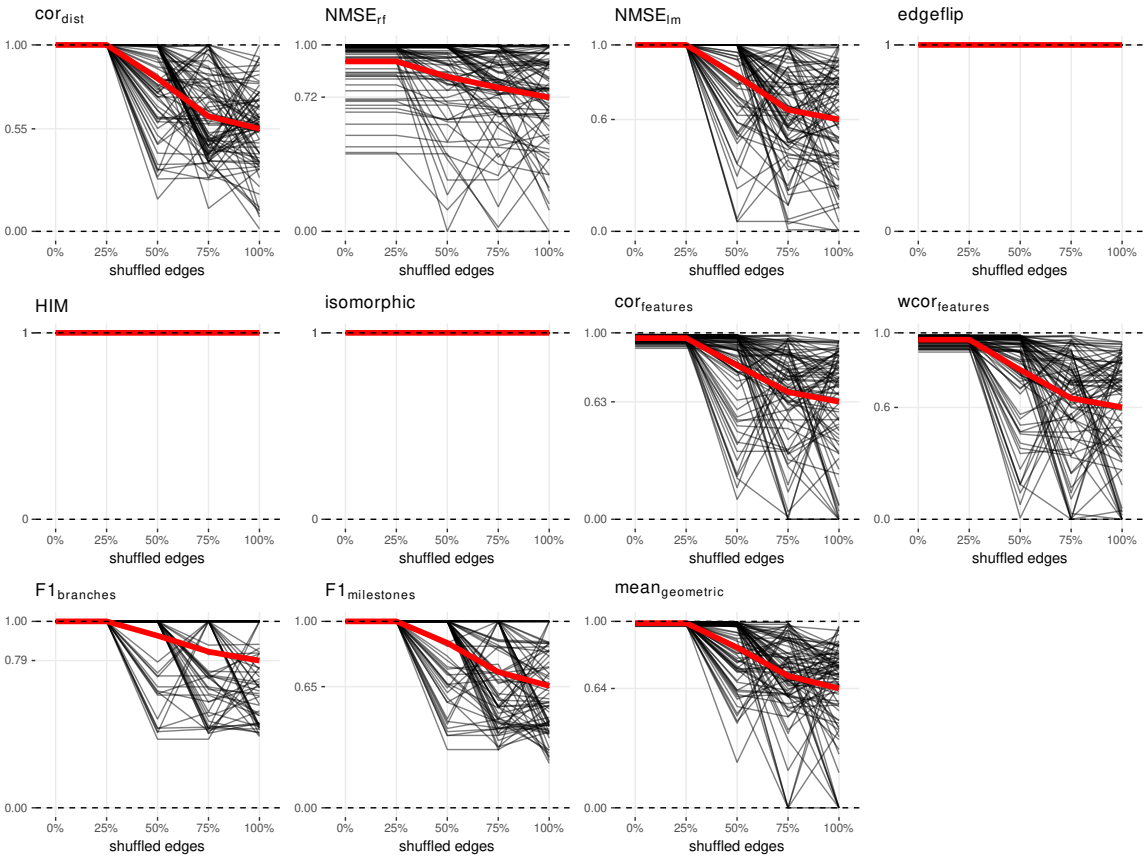


Figure 16: Scores for this rule.

Local and global cell shuffling

Shuffling the positions of cells should lower the score. This is equivalent to changing the cellular position both locally and globally.

A metric conforms to this rule if: $monotonic\left(\textit{shuffled cells}, \overline{score}_{\textit{shuffled cells}}\right)$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✖	✖	✖	✖	✖	✔	✔	✔	✔	✔

Most metrics that look at the position of each cell conform to this rule.

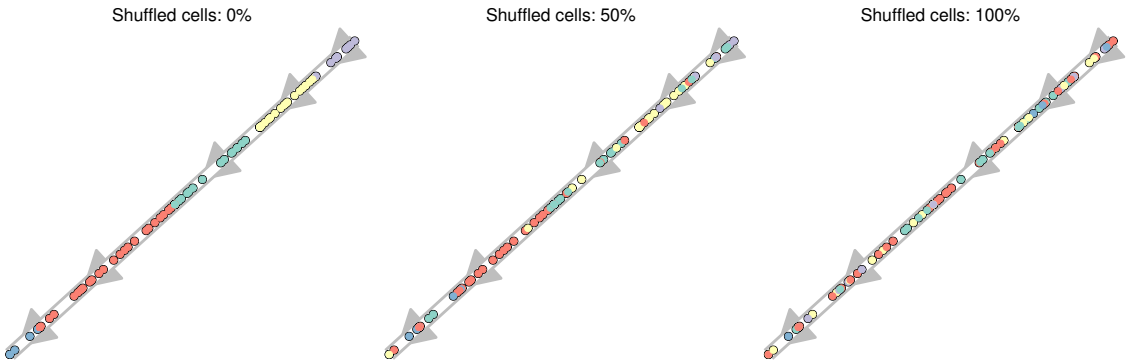


Figure 17: Example dataset(s) for this rule

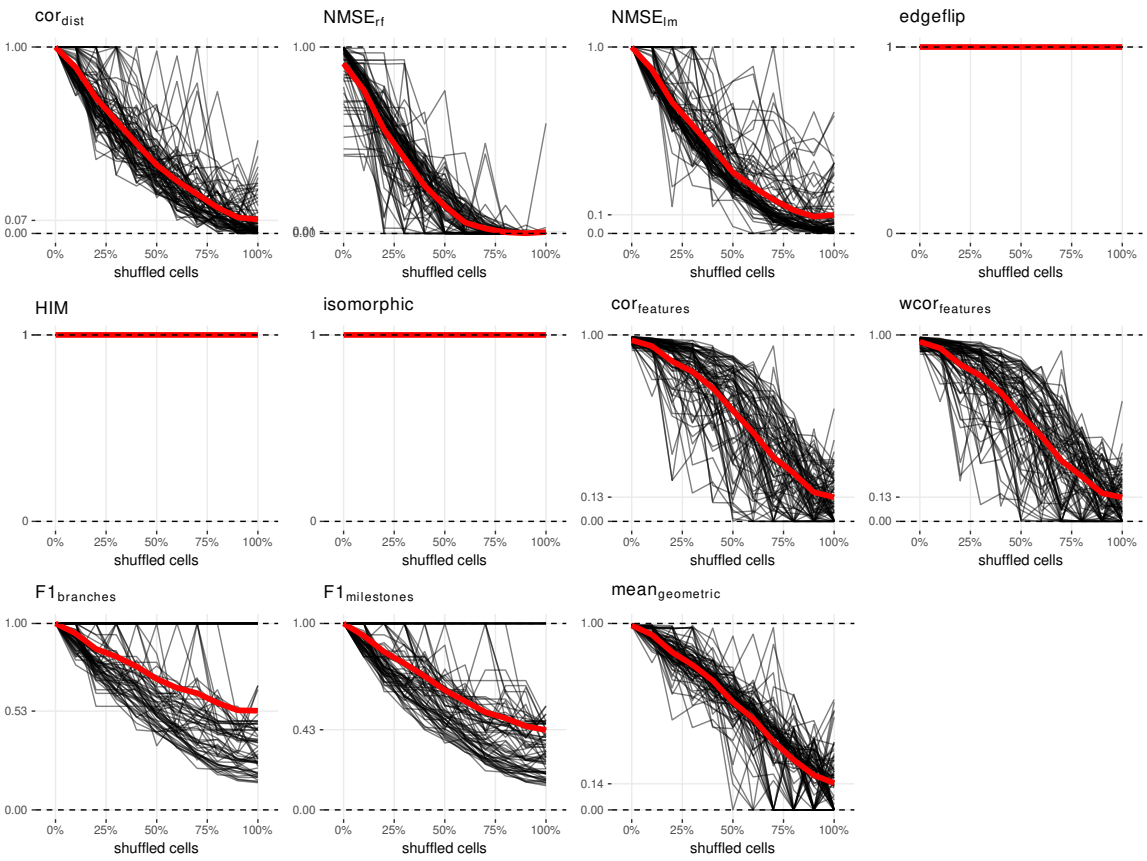
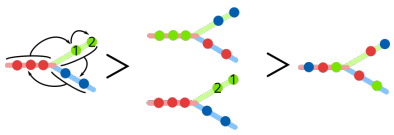


Figure 18: Scores for this rule.

Changing positions locally and/or globally



Changing the cellular position locally AND globally should lower the score more than any of the two individually.

A metric conforms to this rule if: $score_{identity} > score_a \wedge score_{identity} > score_b \wedge score_a > score_{a+b} \wedge score_b > score_{a+b}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✖	✖	✔

Because the topology remains the same, the topology scores do not conform to this rule. Also the clustering based scores have some difficulties with this rule.

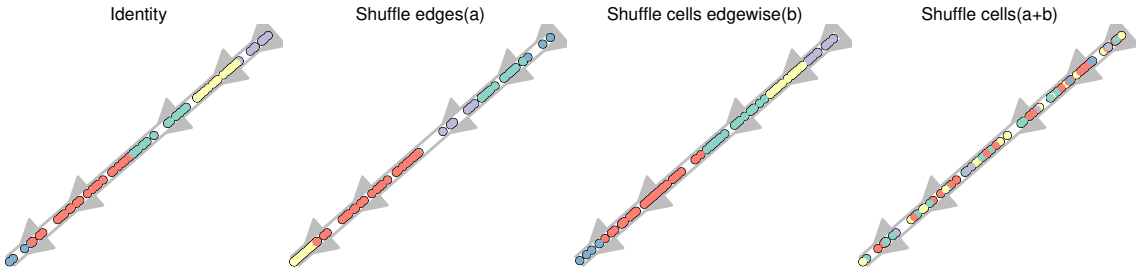


Figure 19: Example dataset(s) for this rule

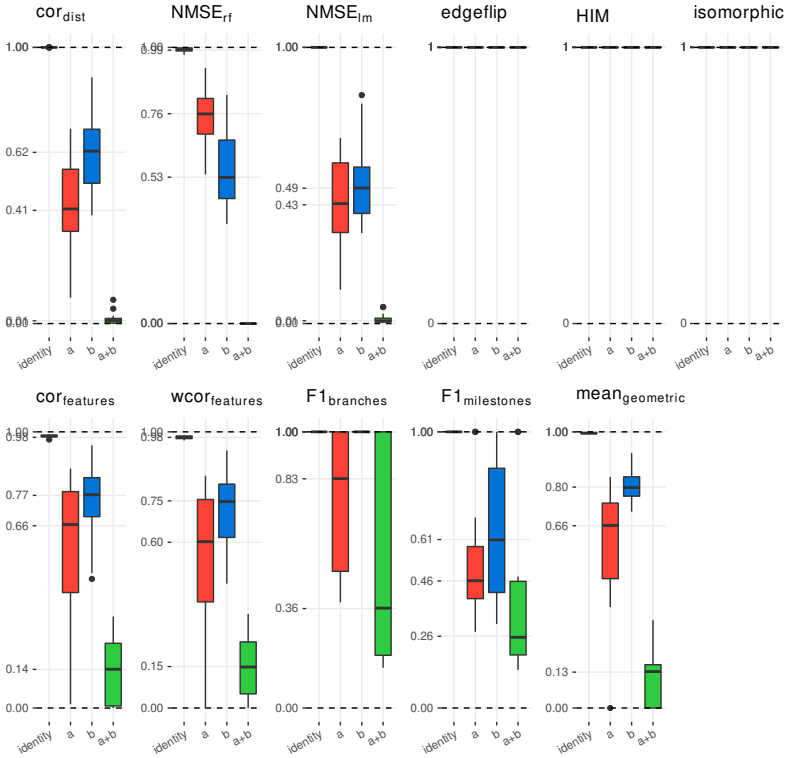


Figure 20: Scores for this rule.

Cell filtering

Removing cells from the trajectory should lower the score

A metric conforms to this rule if: $monotonic\,(\textit{Filtered cells}, \overline{score}_{\textit{Filtered cells}})$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔

Metrics which look at the topology do not conform to this rule.

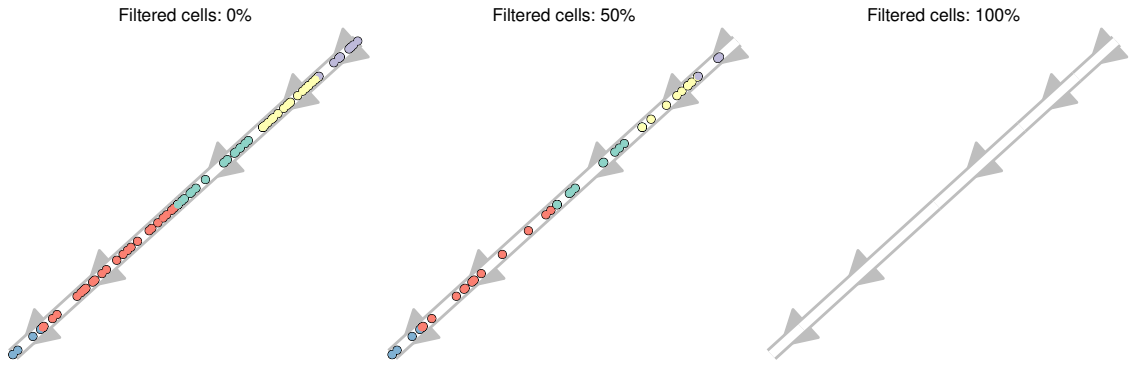


Figure 21: Example dataset(s) for this rule

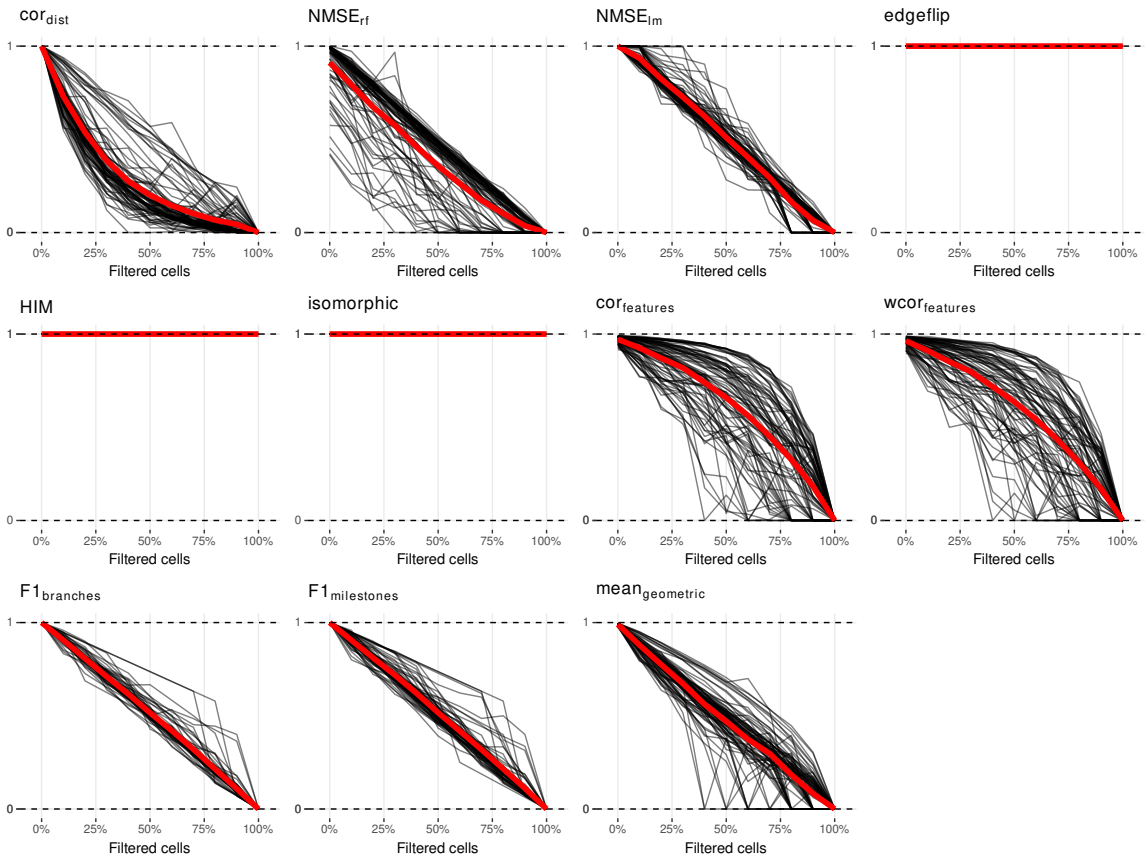


Figure 22: Scores for this rule.

Removing divergence regions

Removing divergence regions should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✖	✔	✔

Both $F1_{branches}$ and edgeflip fail here because neither the topology nor the branch assignment changes. Moreover, the decreases in score are relatively minor for all metrics, given that the impact of the positions of the cells is only minimal.

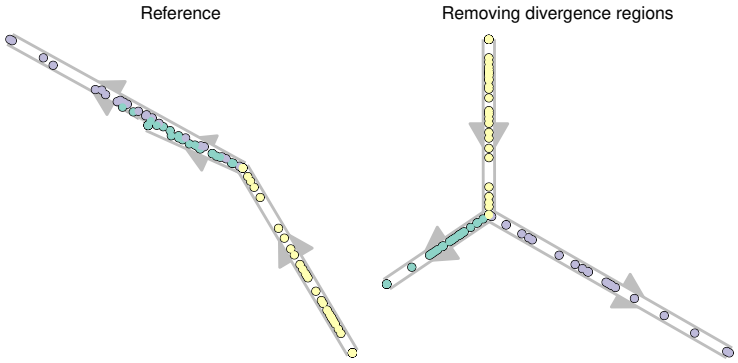


Figure 23: Example dataset(s) for this rule

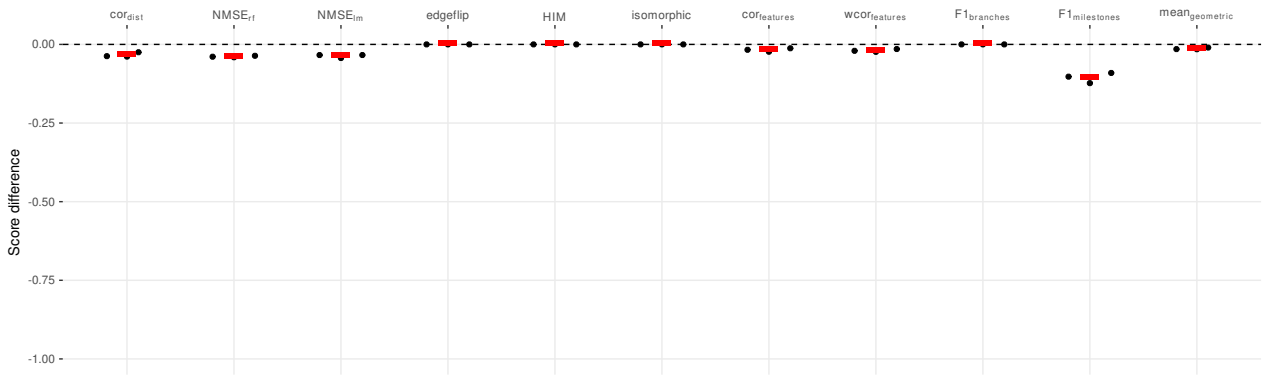
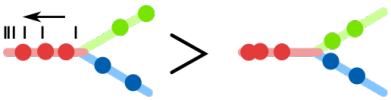


Figure 24: Scores for this rule.

Move cells to start milestone



Moving the cells closer to their start milestone should lower the score. Cells were moved closer to the start milestone using $percentage_{new} = percentage^{warp\ magnitude}$

A metric conforms to this rule if: $monotonic\,(Warp\ magnitude, \overline{score}_{Warp\ magnitude})$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✖	✔	✔

Both $F1_{branches}$ and topology scores fail here because neither the topology nor the branch assignment changes. The score decreases only slightly for all the other metrics, given that only the relative distances change between cells, but not their actual ordering.

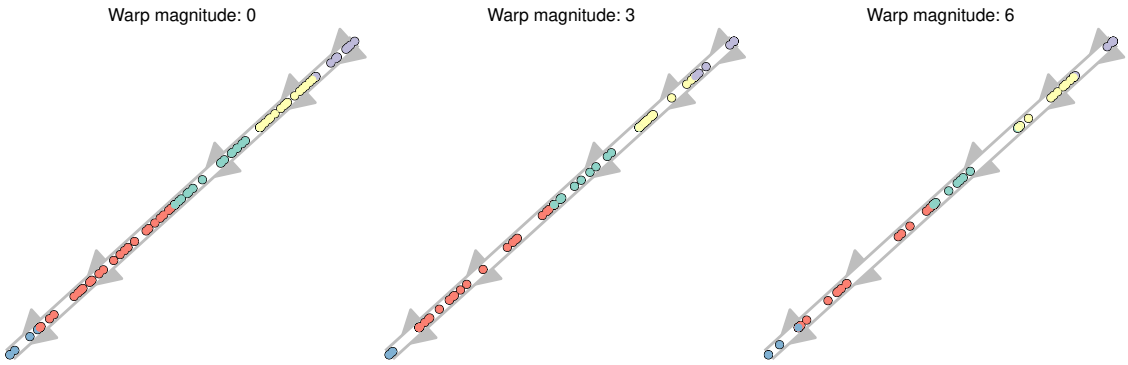


Figure 25: Example dataset(s) for this rule

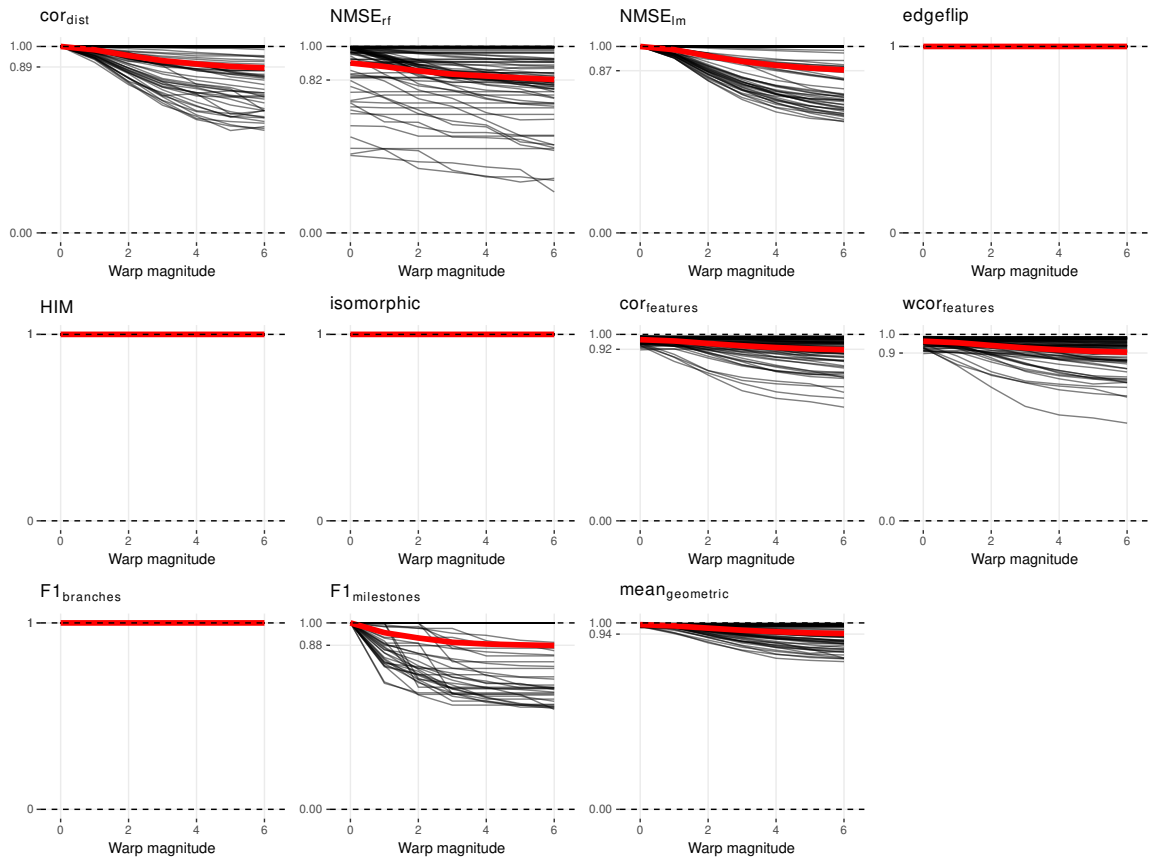


Figure 26: Scores for this rule.

Move cells to closest milestone

Moving the cells closer to their nearest milestone should lower the score

A metric conforms to this rule if: $monotonic\left(\textit{Warp magnitude}, \overline{score}_{\textit{Warp magnitude}}\right)$

cor_{dist}	$NMSE_{\text{rf}}$	$NMSE_{\text{lm}}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✖	✔	✖	✖	✖	✔	✔	✖	✔	✔

Both $F1_{\textit{branches}}$ and topology scores fail here because neither the topology nor the branch assignment changes. The score decreases only slightly for all the other metrics, given that only the relative distances change between cells, but not their actual ordering.

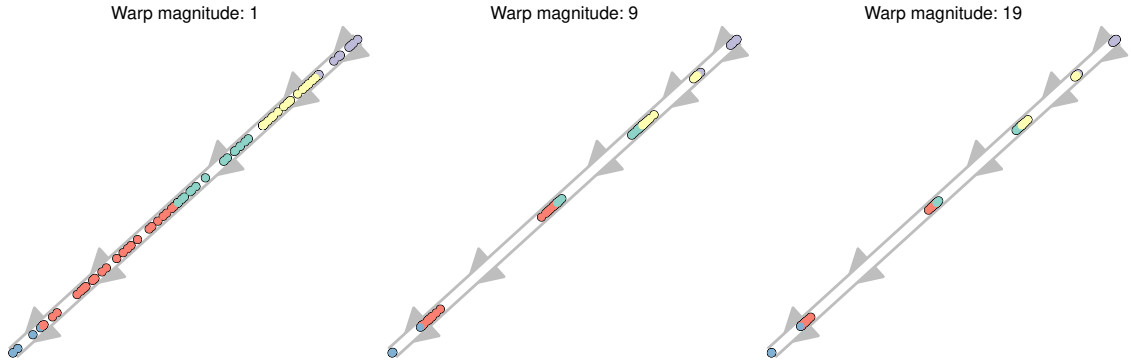


Figure 27: Example dataset(s) for this rule

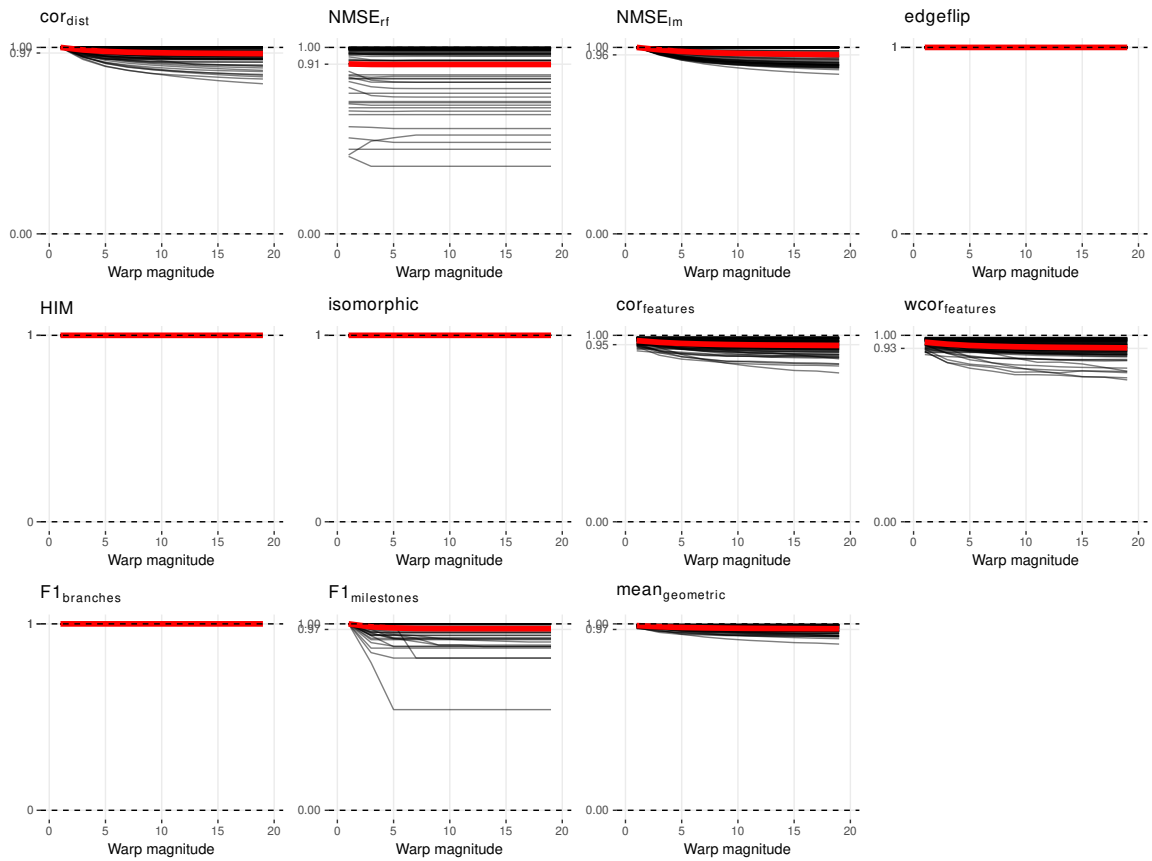
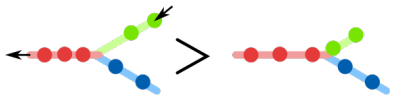


Figure 28: Scores for this rule.

Length shuffling



Shuffling the lengths of the edges of the milestone network should lower the score.

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✖	✖	✖	✖	✖	✖	✖	✖	✔

Only the correlation between geodesic distances is consistently decreases when the lengths of the edges change.

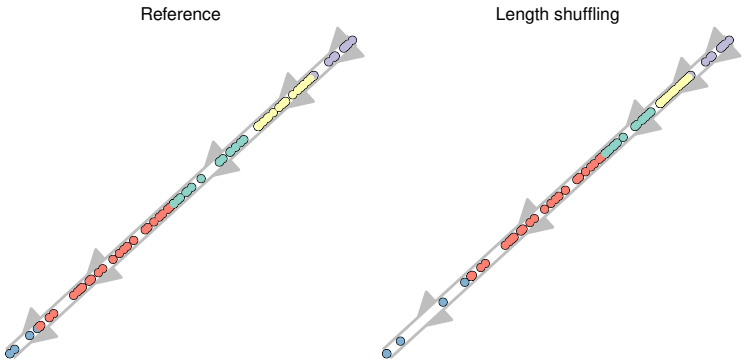


Figure 29: Example dataset(s) for this rule

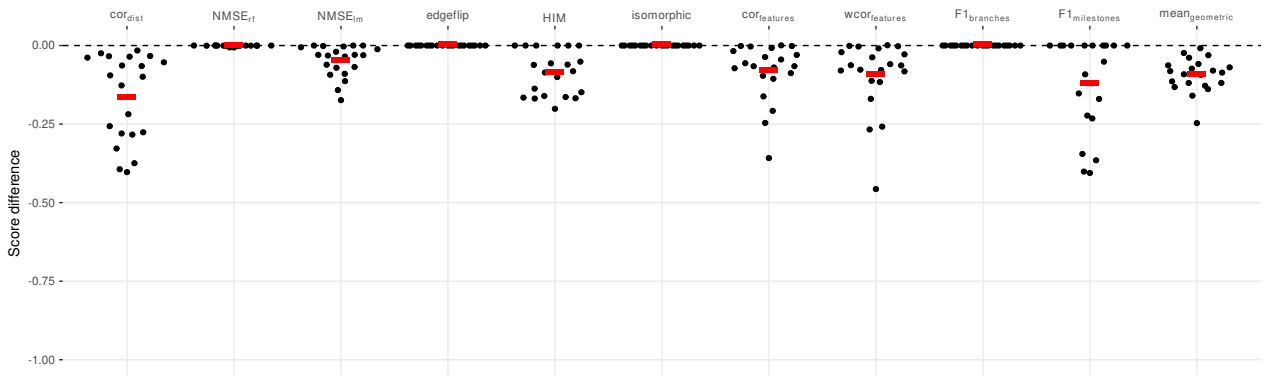
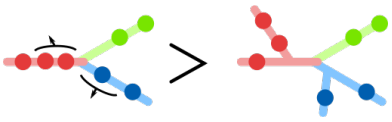


Figure 30: Scores for this rule.

Cells into small subedges



Moving some cells into short subedges should lower the score

A metric conforms to this rule if: $monotonic\left(\textit{Number of added edges}, \overline{score}_{\textit{Number of added edges}}\right)$

cor_{dist}	$NMSE_{\text{rf}}$	$NMSE_{\text{lm}}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✖	✔	✖	✔	✔	✔	✖	✖	✔	✔	✔

This rule is primarily captured by the scores looking at the topology and clustering quality.

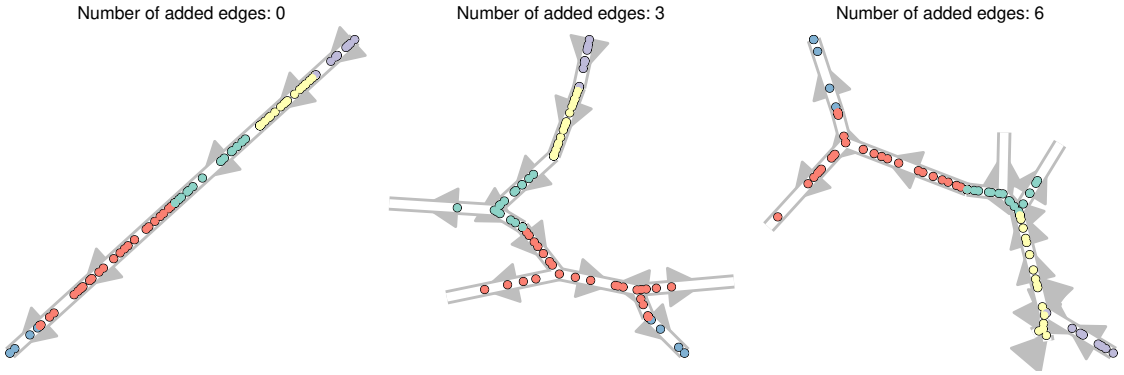


Figure 31: Example dataset(s) for this rule

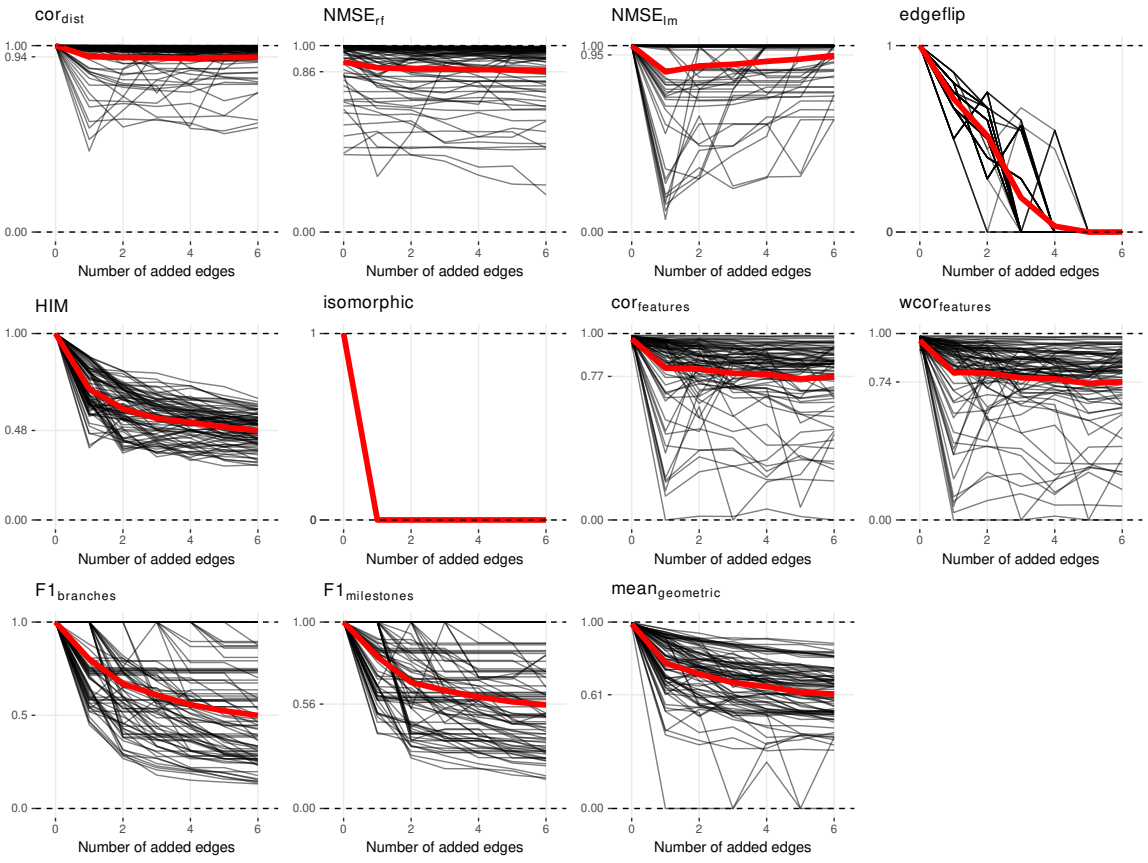


Figure 32: Scores for this rule.

New leaf edges

Adding new edges only connected to one existing milestone should lower the score

A metric conforms to this rule if: $monotonic\left(\textit{Number of edges}, \overline{score}_{\textit{Number of edges}}\right)$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✖	✖	✖	✔	✔	✔	✔	✖	✔	✔	✔

As the positions of the cells are not affected, only metrics which investigate the clustering quality and topology conform to this rule.

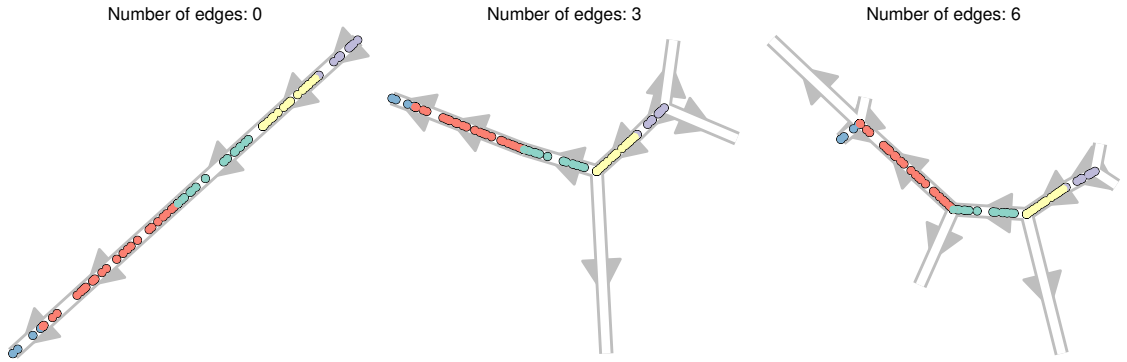


Figure 33: Example dataset(s) for this rule

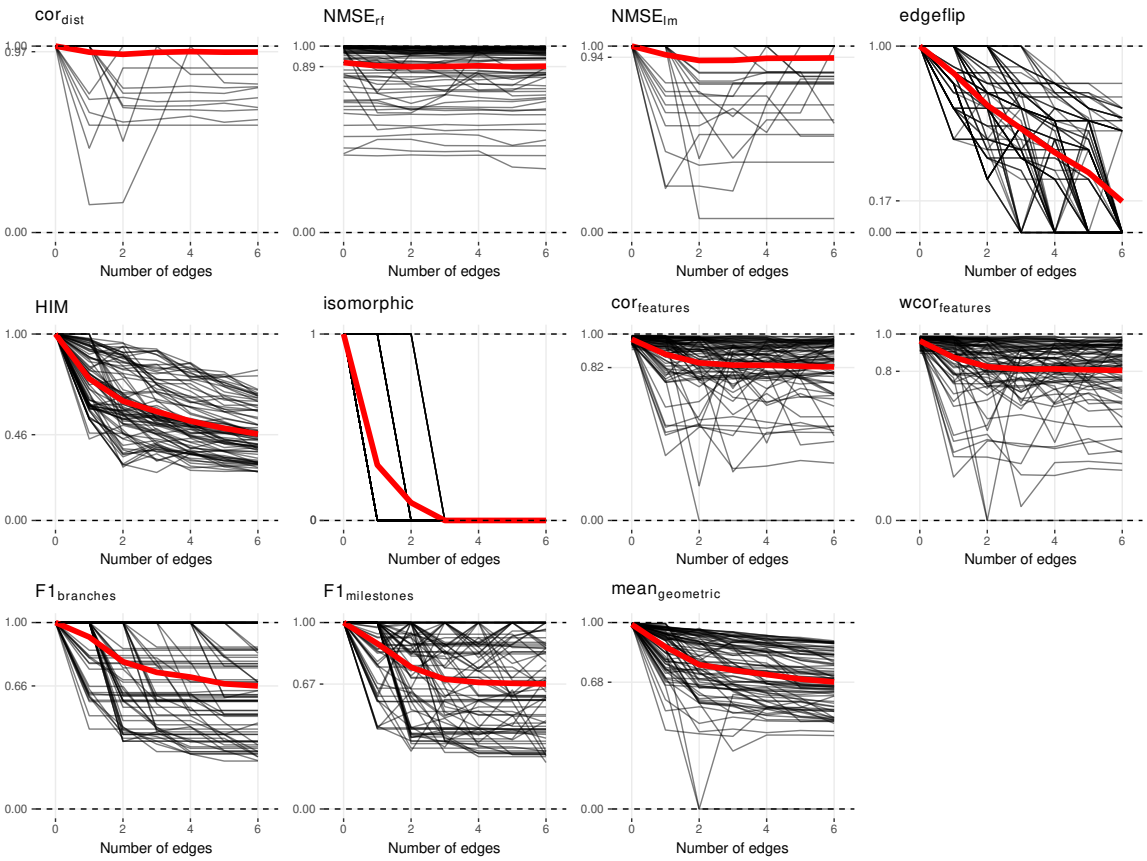
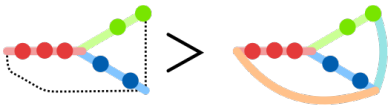


Figure 34: Scores for this rule.

New connecting edges



Adding new edges between existing milestones should lower the score

A metric conforms to this rule if: $monotonic\left(\textit{Number of edges}, \overline{score}_{\textit{Number of edges}}\right)$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✖	✔	✔	✔	✔	✔	✔	✔	✔

Even though the positions of the cells do not change, the cor_{dist} still conforms to this rule because new edges can create shortcuts which will affect the geodesic distances between cells. Apart from this, metrics which investigate the clustering quality and topology also conform to this rule.

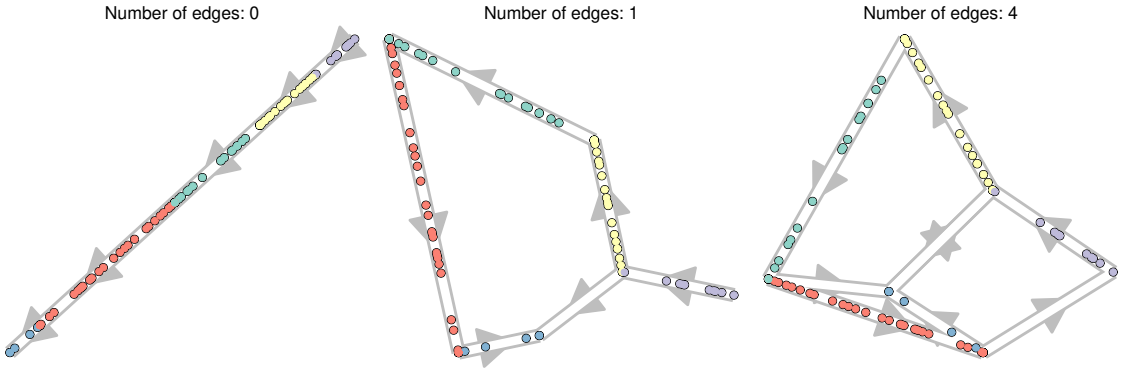


Figure 35: Example dataset(s) for this rule

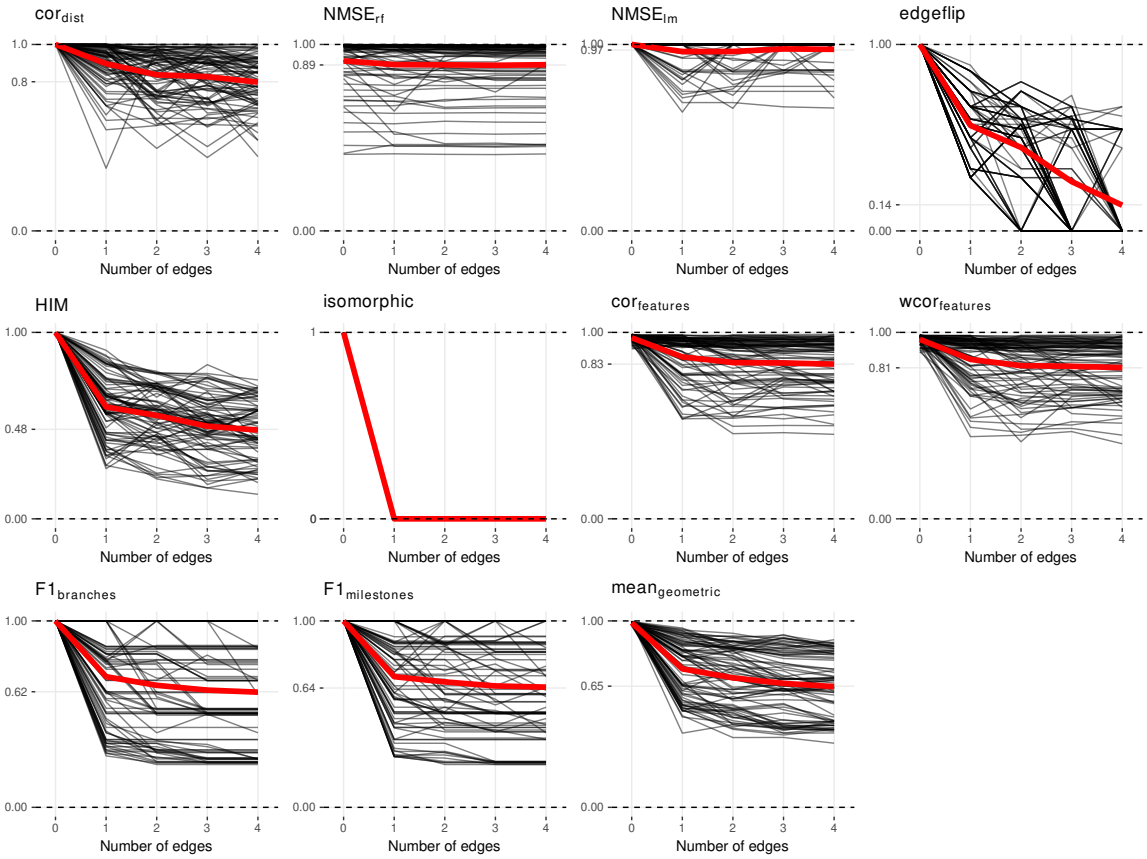
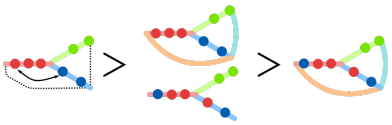


Figure 36: Scores for this rule.

Changing topology and cell position



Changing both the topology and the cell positions should lower the score more than any of the two individually

A metric conforms to this rule if: $score_{identity} > score_a \wedge score_{identity} > score_b \wedge score_a > score_{a+b} \wedge score_b > score_{a+b}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✖	✖	✖	✖	✖	✖	✖	✖	✖	✔

Most metrics have problems with this rule as they focus on either the cellular positions or the topology individually. Only the cor[dist] and $mean_{geometric}$ consistently conform to this rule.

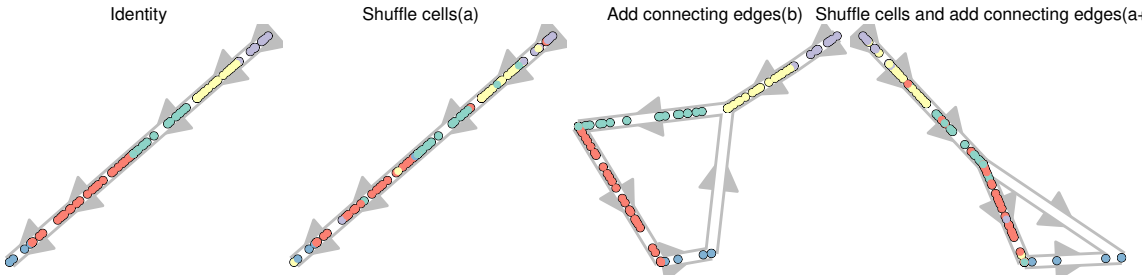


Figure 37: Example dataset(s) for this rule

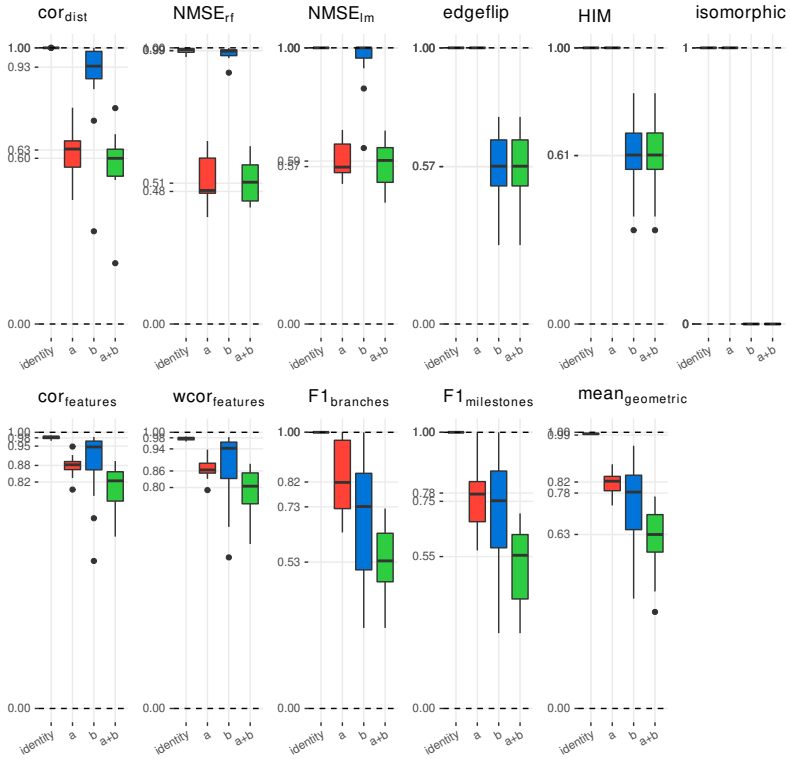
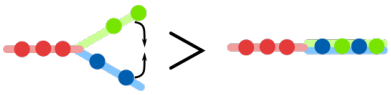


Figure 38: Scores for this rule.

Bifurcation merging



Merging the two branches after a bifurcation point should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✔	✔	✔	✔	✔	✔	✔	✔	✔

This changes both the cellular ordering and the topology so most metrics are affected.

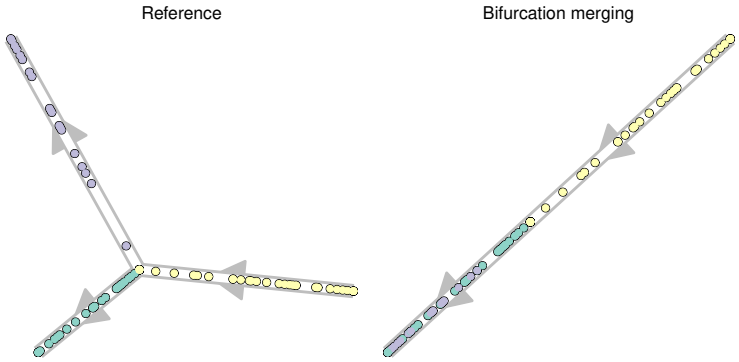


Figure 39: Example dataset(s) for this rule

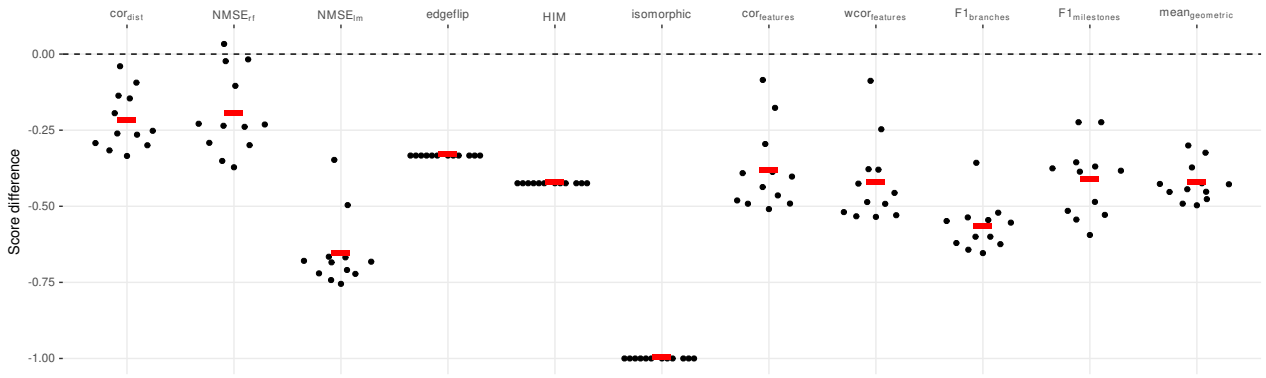
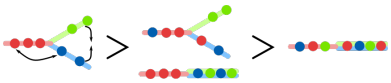


Figure 40: Scores for this rule.

Bifurcation merging and changing cell positions



Merging the two branches of a bifurcation and changing the cells positions should lower the score more than any of the two individually

A metric conforms to this rule if: $score_{identity} > score_a \wedge score_{identity} > score_b \wedge score_a > score_{a+b} \wedge score_b > score_{a+b}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✔	✖	✖	✖	✔	✔	✔	✔	✔

Only metrics which look at the topology do not conform to this rule.

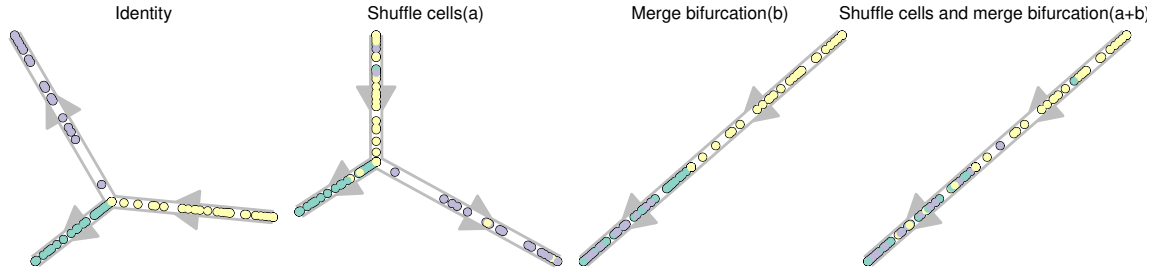


Figure 41: Example dataset(s) for this rule

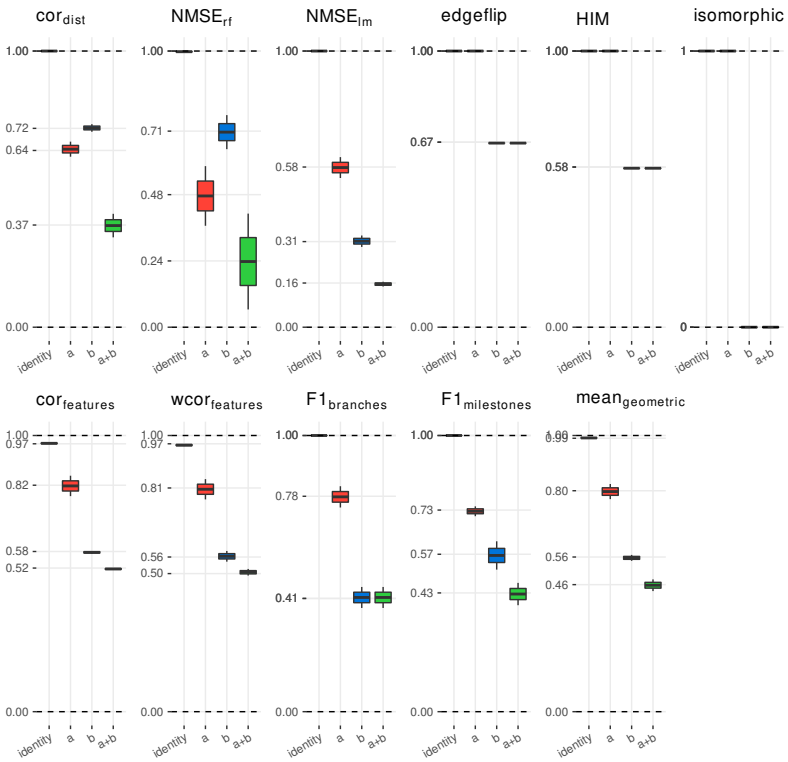


Figure 42: Scores for this rule.

Bifurcation concatentation

Concatenating one branch of a bifurcation to the other bifurcation branch should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✔	✔	✔	✔	✔	✔	✔	✔	✔

This changes both the cellular ordering and the topology so most metrics conform to this rule.

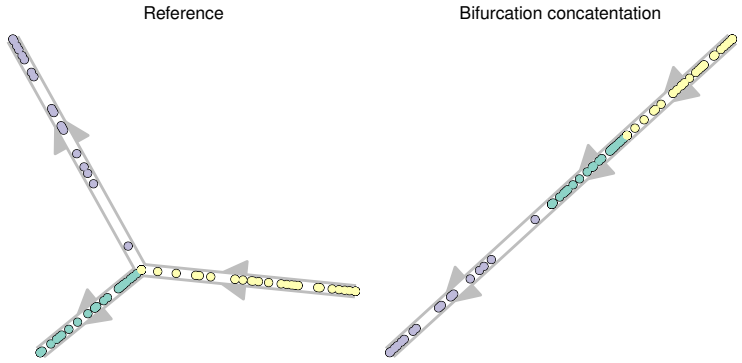


Figure 43: Example dataset(s) for this rule

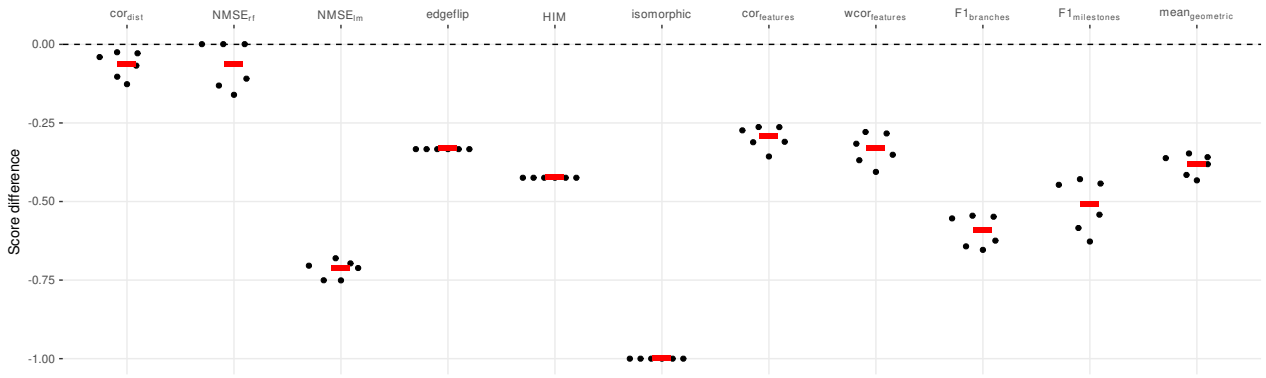


Figure 44: Scores for this rule.

Cycle breaking

Breaking a cyclic trajectory should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✖	✔	✔	✔	✔	✔	✔	✖	✔	✔

Because the actual positions of the cells nor the branch assignment change, both the MSE metrics and the $F1_{branches}$ do not conform to this rule.

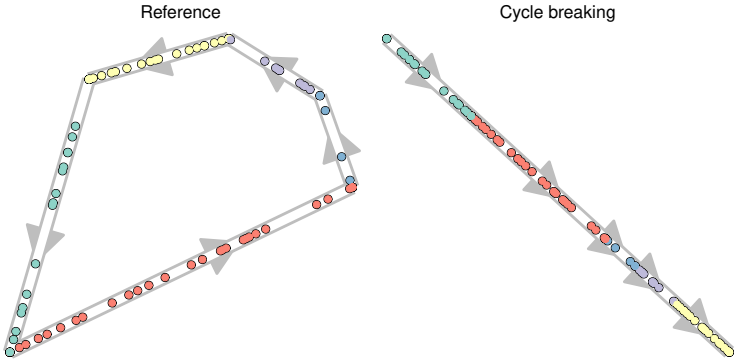


Figure 45: Example dataset(s) for this rule

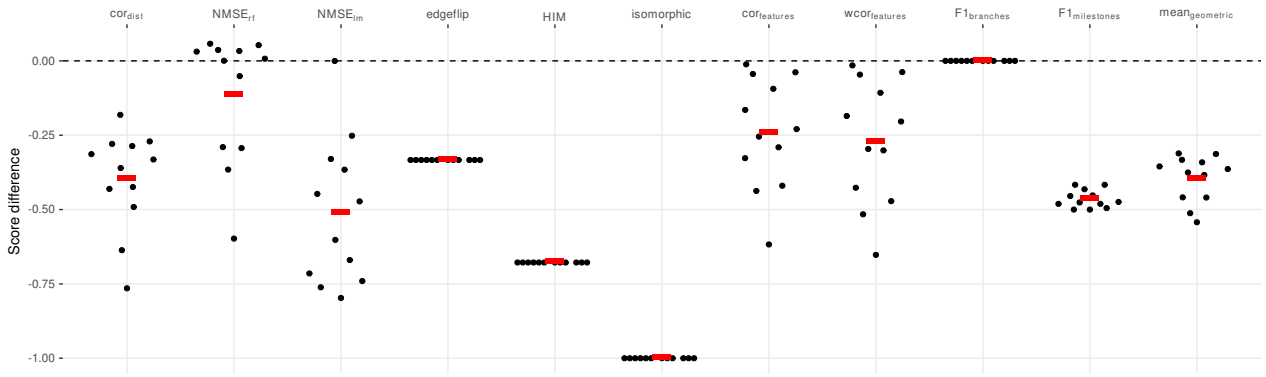
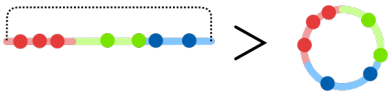


Figure 46: Scores for this rule.

Linear joining



Joining the two ends of a linear trajectory should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✖	✔	✔	✔	✔	✔	✖	✔	✔

Because the positions of the cells can be perfectly predicted, the MSE metrics do not conform to this rule. Furthermore, because the branch assignment change stays the same, the $F1_{branches}$ also does not conform to this rule.

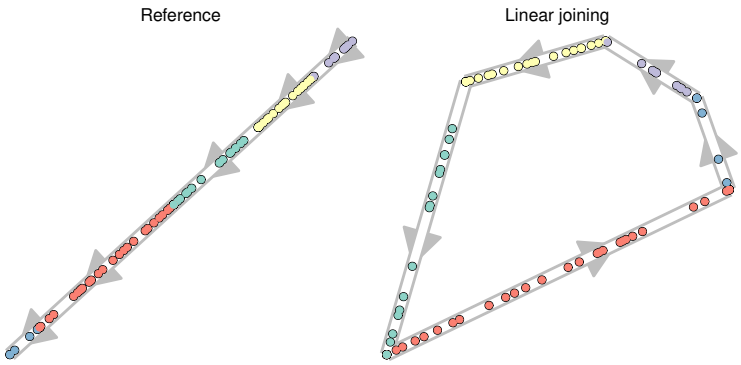


Figure 47: Example dataset(s) for this rule

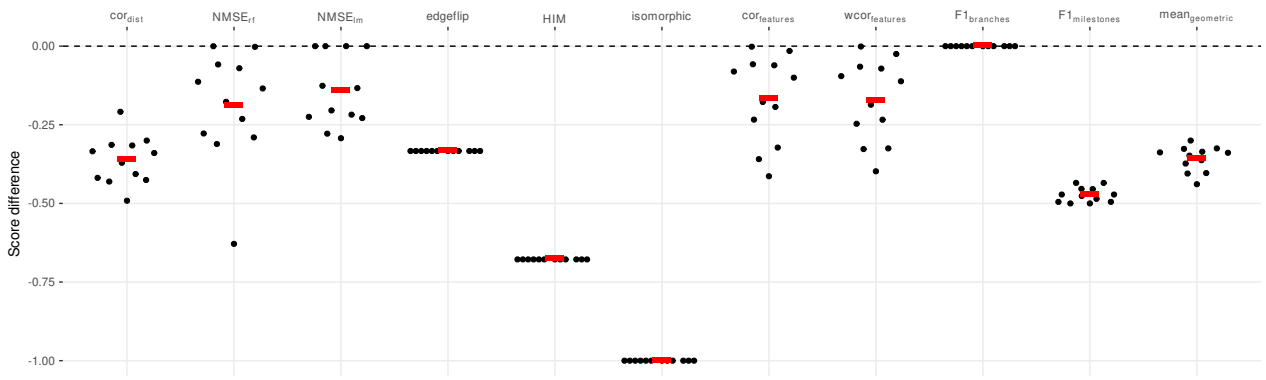
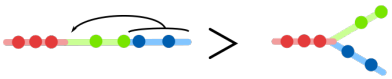


Figure 48: Scores for this rule.

Linear splitting



Splitting a linear trajectory into a bifurcation should lower the score

A metric conforms to this rule if: $score_{identity} > score_{prediction}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	$cor_{features}$	$wcor_{features}$	$F1_{branches}$	$F1_{milestones}$	$mean_{geometric}$
✔	✔	✖	✔	✔	✔	✔	✔	✔	✔	✔

Only the MSE metrics do not conform to this rule as the positions of the cells can be perfectly predicted in the gold standard given the prediction.

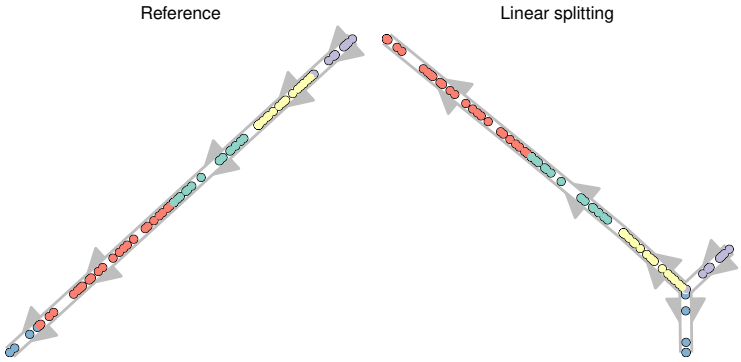


Figure 49: Example dataset(s) for this rule

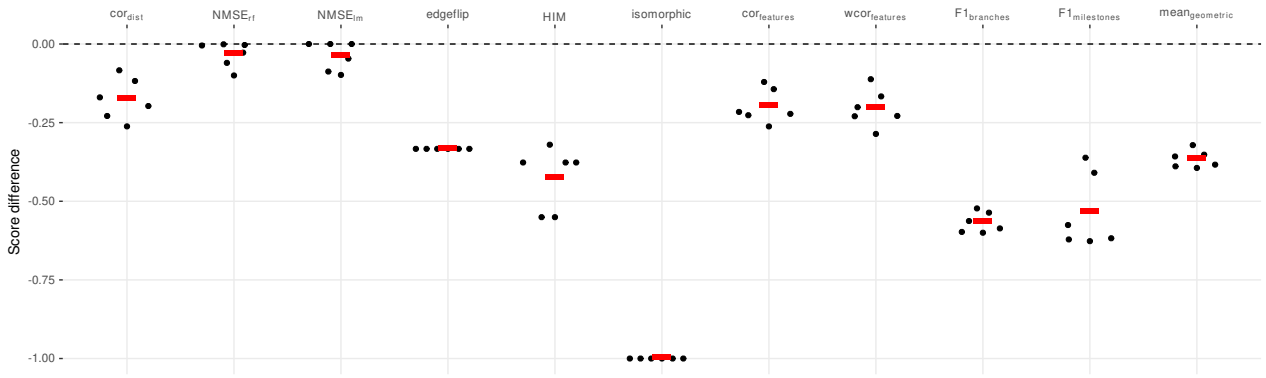


Figure 50: Scores for this rule.

Change of topology

Changing the topology of the trajectory should lower the score

A metric conforms to this rule if: $score_{\textit{same topology}} > score_{\textit{different topology}}$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✖	✖	✔	✔	✔	✔	✔	✖	✔	✔

Because the positions of the cells can be perfectly predicted, the MSE metrics do not conform to this rule. Furthermore, because the branch assignment change stays the same, the $F1_{\textit{branches}}$ also does not conform to this rule.

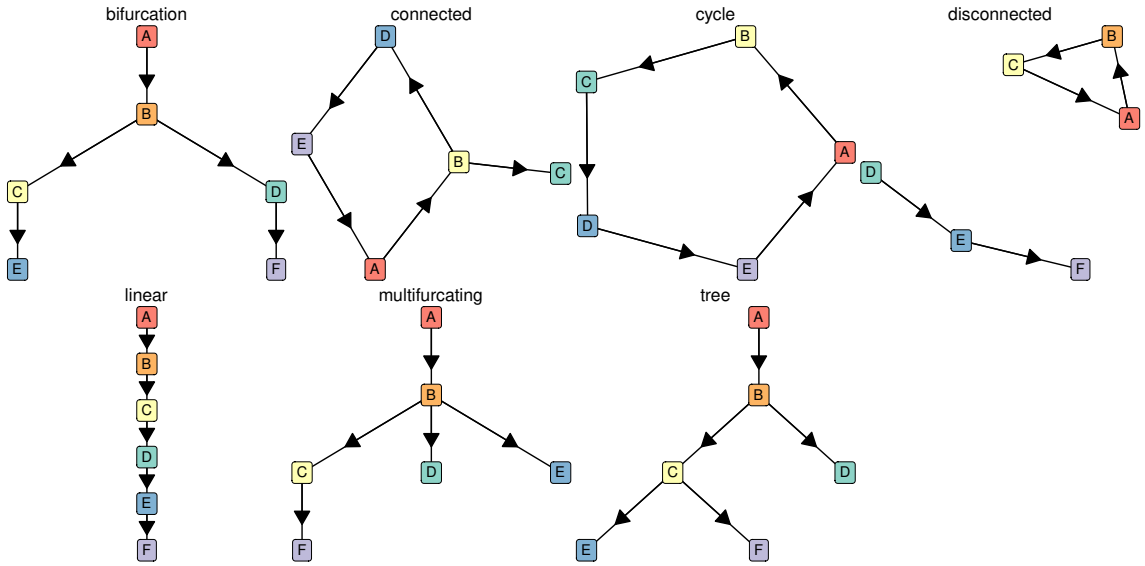


Figure 51: Example dataset(s) for this rule

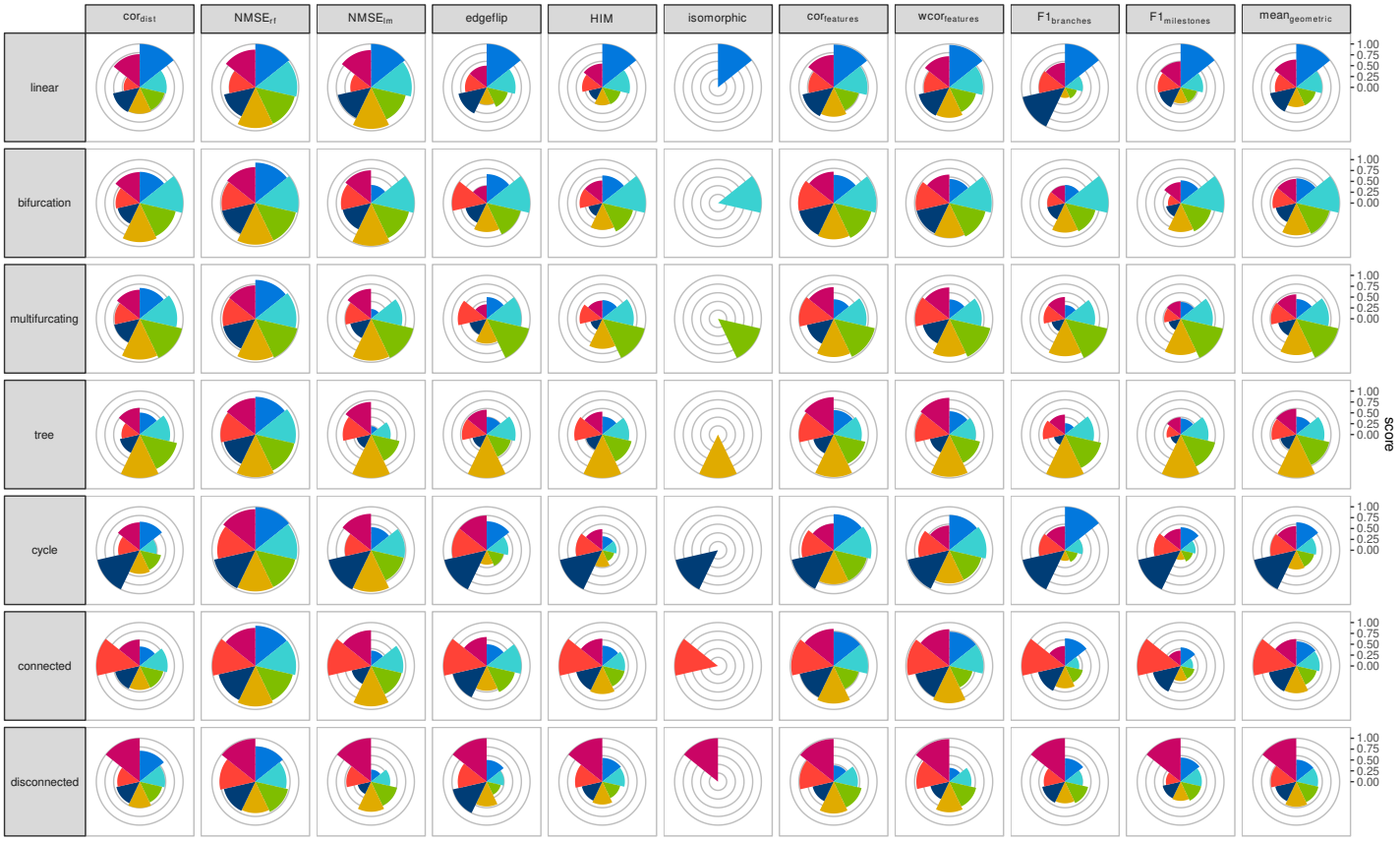


Figure 52: Scores for this rule.

Cells on milestones vs edges

A score should behave similarly both when cells are located on the milestones (as is the case in real datasets) or on the edges between milestones (as is the case in synthetic datasets).

A metric conforms to this rule if: $corr\left(score_{edges}, score_{milestones}\right) > 0.8$

cor_{dist}	$NMSE_{rf}$	$NMSE_{lm}$	edgeflip	HIM	isomorphic	cor_{features}	$wcor_{\text{features}}$	$F1_{\text{branches}}$	$F1_{\text{milestones}}$	$mean_{\text{geometric}}$
✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

All scores conform to this rule.

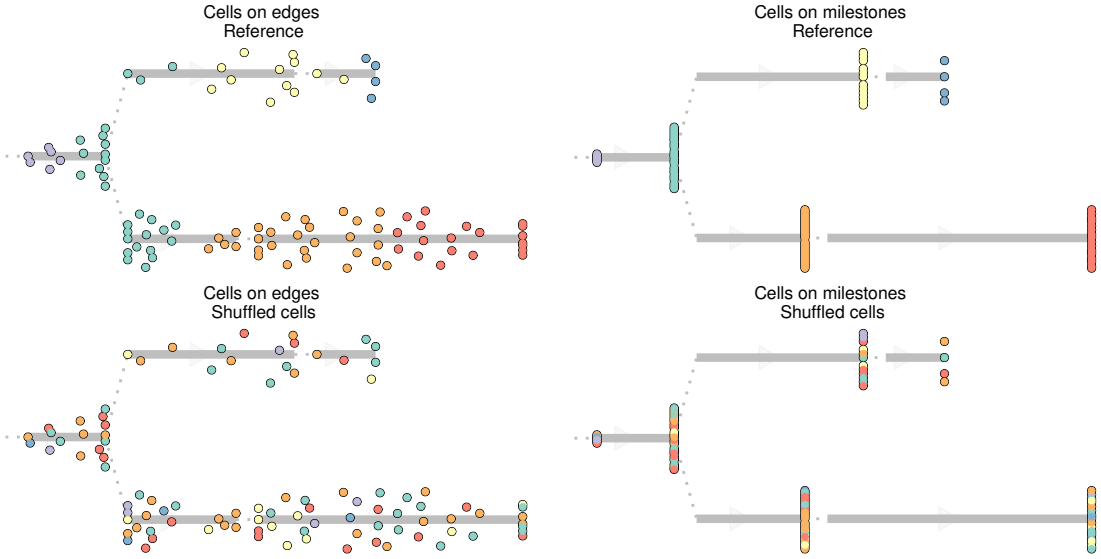


Figure 53: Example dataset(s) for this rule

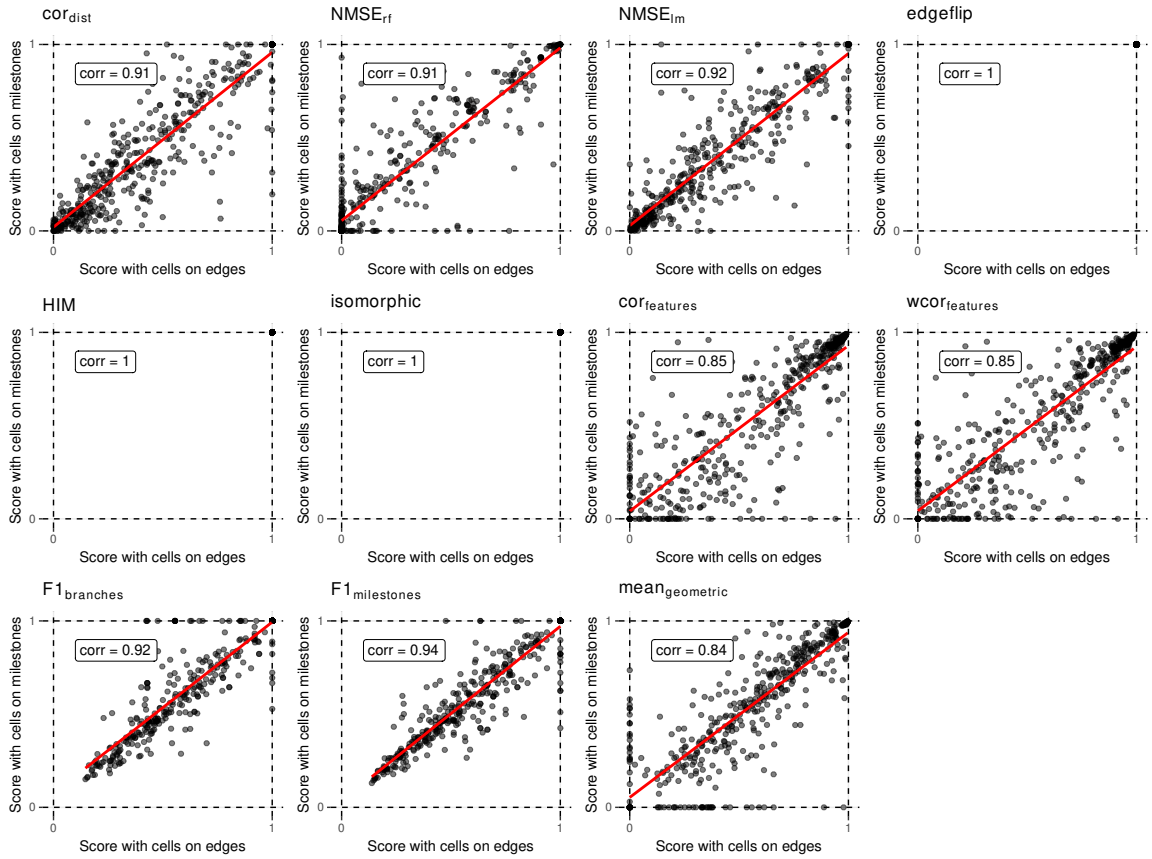


Figure 54: Scores for this rule.

Score aggregation

To rank the methods, we need to aggregate on two levels: across **datasets** and across specific metrics to calculate an **overall metric**.

Aggregating over datasets

When combining different datasets, it is important that the biases in the datasets does not influence the overall score. In our study, we define three such biases, although there are potentially many more:

- **Difficulty of the datasets**: Some datasets are more difficult than others. This can have various reasons, such as the complexity of the topology, the amount of biological and technical noise, or the dimensions of the data. It is important that a small increase in performance on a more difficult dataset has an equal impact on the final score as a large increase in performance on easier datasets.
- **Dataset sources**: It is much easier to generate synthetic datasets than real datasets, and this bias is reflected in our set of datasets. However, given their higher biological relevance, real datasets should be given at least equal importance than synthetic datasets.
- **Trajectory types**: There are many more linear and disconnected real datasets, and only a limited number of tree or graph datasets. This imbalance is there because historically most datasets have been linear datasets, and because it is easy to create disconnected datasets by combining different datasets. However, this imbalance in trajectory types does not necessarily reflect the general importance of that trajectory type.

We designed an aggregation scheme which tries to prevent these biases from influencing the ranking of the methods.

The difficulty of a dataset can easily have an impact on how much weight the dataset gets in an overall ranking. We illustrate this with a simple example in **Figure 55**. One method consistently performs well on both the easy and the difficult datasets. But because the differences are small in the difficult datasets, the mean would not give this method a high score. Meanwhile, a variable method which does not perform well on the difficult dataset gets the highest score, because it scored so high on the easier dataset.

To avoid this bias, we normalise the scores of each dataset by first scaling and centering to $\mu = 0$ and $\sigma = 1$, and then moving the score values back to $[0, 1]$ by applying the unit normal density distribution function. This results in scores which are comparable across different datasets (**Figure 55**). In contrast to other possible normalisation techniques, this will still retain some information on the relative difference between the scores, which would have been lost when using the ranks for normalisation. An example of this normalisation, which will also be used in the subsequent aggregation steps, can be seen in **Figure 56**.

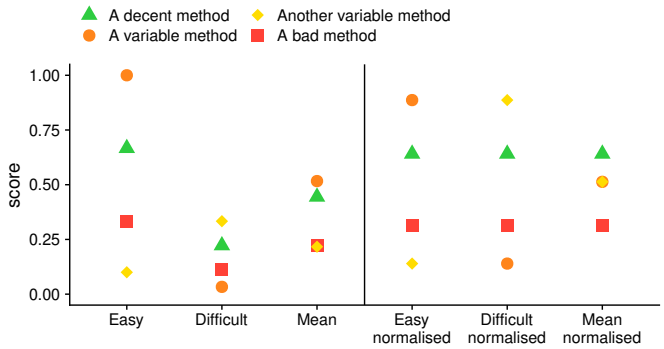


Figure 55: An illustration of how the difficulty of a dataset can influence the overall ranking. A decent method, which consistently ranks high on an easy and difficult dataset, does not get a high score when averaging. On the other hand, a method which ranks high on the easy dataset, but very low on the difficult dataset does get a high score on average. After normalising the scores (right), this problem dissapears.

After normalisation, we aggregate step by step the scores from different datasets. We first aggregate the datasets with the same dataset source and trajectory type using an arithmetic mean of their scores **Figure 57a**. Next, the scores are averaged over different dataset sources, using a arithmetic mean which was weighted based on how much the synthetic and silver scores correlated with the real gold scores **Figure 57b**. Finally, the scores were aggregated over the different trajectory types again using a arithmetic mean **Figure 57c**.

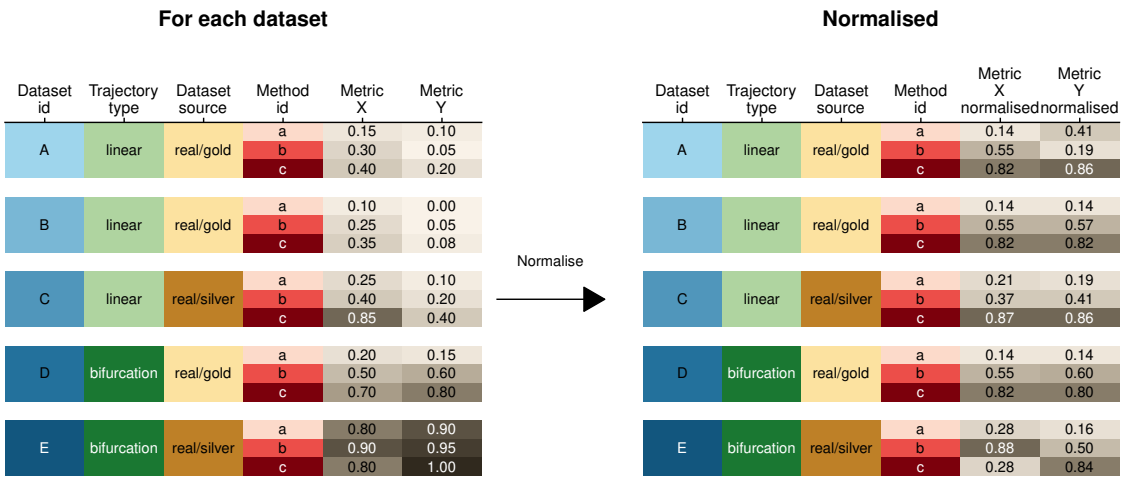


Figure 56: An example of the normalisation procedure. Shown are some results of a benchmarking procedure, where every row contains the scores of a particular method (red shading) on a particular dataset (blue shading), with a trajectory type (green shading) and dataset source (orange shading). In this example, we first split the datasets

Overall metrics

Undoubtedly, a single optimal overall metric does not exist for trajectories, as different users may have different priorities:

- A user may be primarily interested in defining the correct topology, and only use the cellular ordering when the topology is correct
- A user may be less interested in how the cells are ordered within a branch, but primarily in which cells are in which branches
- A user may already know the topology, and may be primarily interested in finding good features related to a particular branching point
- ...

Each of these scenarios would require a combinations of *specific* and *application* metrics with different weights. To provide an "overall" ranking of the metrics, which is impartial for the scenarios described above, we therefore chose a metric which weighs every aspect of the trajectory equally:

- Its **ordering**, using the cor_{dist}
- Its **branch assignment**, using the $F1_{branches}$
- Its **topology**, using the HIM
- The accuracy of **differentially expressed features**, using the $wcor_{\text{features}}$

Next, we considered three different ways of averaging different scores: the arithmetic mean, geometric mean and harmonic mean. Each of these types of mean have different use cases. The harmonic mean is most appropriate when the scores would all have a common denominator (as is the case for the Recovery and Relevance described earlier). The arithmetic mean would be most appropriate when all the metrics have the same range. For our use case, the geometric mean is the most appropriate, because it is low if one of the values is low. For example, this means that if a method is not good at inferring the correct topology, it will get a low overall score, even if it performs better at all other scores. This ensures that a high score will only be reached if a prediction has a good ordering, branch assignment, topology, and set of differentially expressed features.

The final overall score (**Figure 58**) for a method was thus defined as:

$$Overall = mean_{geometric} = \sqrt[4]{cor_{\text{dist}} \times \text{HIM} \times wcor_{\text{features}} \times F1_{branches}}$$

We do however want to stress that different use cases will require a different overall score to order the methods. Such a context-dependent ranking of all methods is provided through the dynguidelines app (guidelines.dynverse.org).

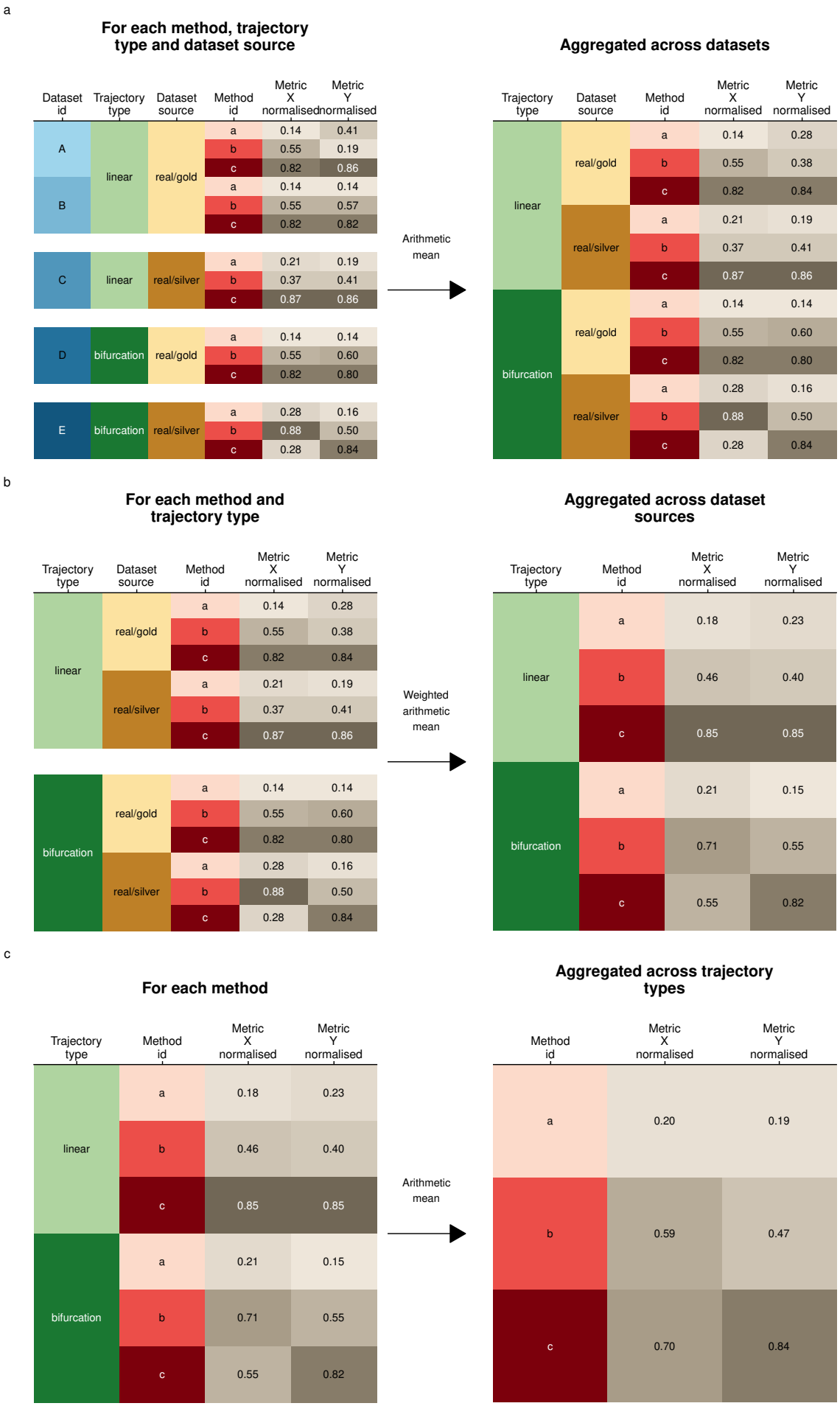


Figure 57: An example of the aggregation procedure. In consecutive steps we aggregated across (a) different datasets with the same source and trajectory type, (b) different dataset sources with the same trajectory type (weighted for the correlation of the dataset source with the real gold dataset source) and (c) all trajectory types.

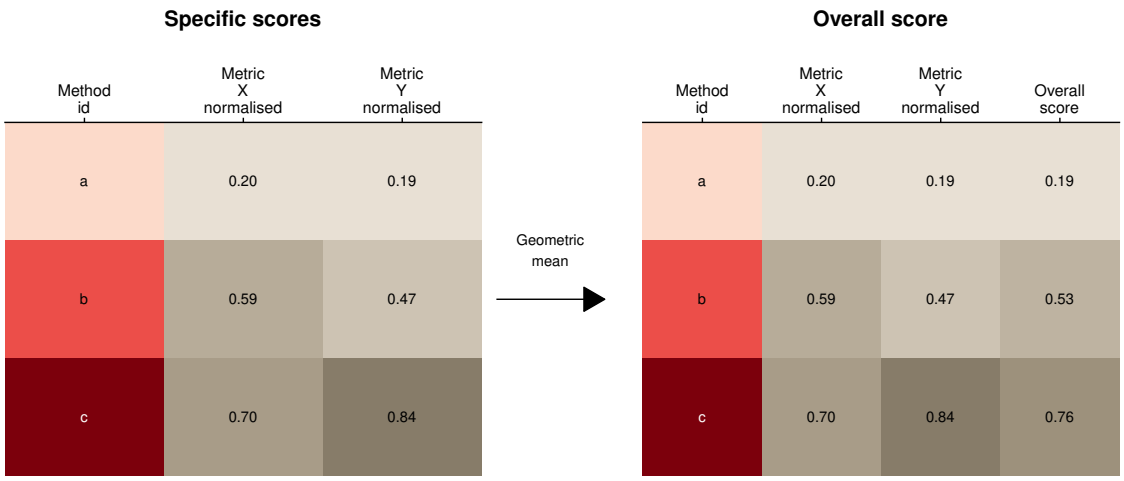


Figure 58: An example of the averaging procedure. For each method, we calculated the geometric mean between its normalised and aggregated scores

References

1. Junttila, T. & Kaski, P. Engineering an Efficient Canonical Labeling Tool for Large and Sparse Graphs. in *2007 Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments (ALENEX)* 135–149 (Society for Industrial and Applied Mathematics, 2007). doi:10.1137/1.9781611972870.13

2. Bahiense, L., Manić, G., Piva, B. & de Souza, C. C. The maximum common edge subgraph problem: A polyhedral investigation. *Discrete Applied Mathematics* **160,** 2523–2541 (2012).

3. Jurman, G., Visintainer, R., Filosi, M., Riccadonna, S. & Furlanello, C. The HIM glocal metric and kernel for network comparison and classification. in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* 1–10 (2015). doi:10.1109/DSAA.2015.7344816

4. Dougherty, E. R. Validation of gene regulatory networks: Scientific and inferential. *Briefings in Bioinformatics* **12,** 245–252 (2011).

5. Ipsen, M. & Mikhailov, A. S. Evolutionary reconstruction of networks. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics* **66,** 046109 (2002).

6. Saelens, W., Cannoodt, R. & Saeys, Y. A comprehensive evaluation of module detection methods for gene expression data. *Nature Communications* **9,** 1090 (2018).

7. Wright, M. N. & Ziegler, A. Ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R | Wright | Journal of Statistical Software. *Journal of Statistical Software* **77,** (2017).