Chapter 13

Mining of Single-Cell Signaling Time-Series for Dynamic Phenotypes with Clustering

Maciej Dobrzyński, Marc-Antoine Jacques, and Olivier Pertz

Abstract

Fluorescent live cell time-lapse microscopy is steadily contributing to our better understanding of the relationship between cell signaling and fate. However, large volumes of time-series data generated in these experiments and the heterogenous nature of signaling responses due to cell-cell variability hinder the exploration of such datasets. The population averages insufficiently describe the dynamics, yet finding prototypic dynamic patterns that relate to different cell fates is difficult when mining thousands of time-series. Here we demonstrate a protocol where we identify such dynamic phenotypes in a population of PC-12 cells that respond to a range of sustained growth factor perturbations. We use Time-Course Inspector, a free R/Shiny web application to explore and cluster single-cell time-series.

Key words Clustering, Time-series, Single-cell data, Cell-cell heterogeneity, Signaling dynamics, Data analysis, Computational biology, Signal processing

1 Introduction

Thanks to advancements in modern time-lapse microscopy, measurements of time-resolved protein activities in individual cells routinely yield hundreds or even thousands of single-cell time-series. A number of recent studies have demonstrated that the dynamics of protein activity in response to an external perturbation such as a growth factor, an inhibitor, or irradiation can vary significantly between cells even in an isogenic population [1, 2] (Fig. 1a). Such experiments are often used to gain information about timescales and connectivity of the signaling pathway that activates the measured protein [4, 5]. However, due to heterogeneous responses of individual cells, the population averages such as those obtained with classic biochemical methods are insufficient to gain mechanistic insights about cell signaling or to investigate the relationship between the signaling and cell fate (Fig. 1b). Therefore, a common approach to identify distinct dynamics in a population of

Zhike Zi and Xuedong Liu (eds.), *TGF-Beta Signaling: Methods and Protocols*, Methods in Molecular Biology, vol. 2488,
https://doi.org/10.1007/978-1-0716-2277-3_13,
© The Author(s), under exclusive license to Springer Science+Business Media, LLC, part of Springer Nature 2022

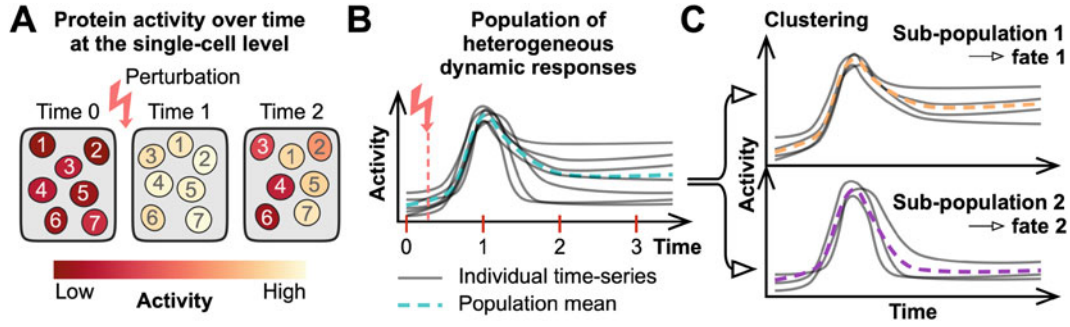

Fig. 1 Schematic representation of a single-cell time-lapse perturbation experiment that measures protein activity with a biosensor. (**a**) An isogenic cell population with a uniform protein activity at time point 0 is treated with a growth factor, which induces a population-wide response at time 1. The response becomes heterogeneous at time 2. (**b**) Heterogeneous time-series of protein activity in individual cells and the population mean. (**c**) Clustering of time-series reveals distinct dynamic phenotypes, which induce different cell fates. (Reproduced from [3] with permission from Oxford University Press)

heterogeneous single-cell time-series is to use clustering. Such dynamic patterns can be then attributed to distinct cell fates induced by the signaling (Fig. 1c) [5–7].

This chapter outlines a protocol for the identification of distinct dynamic patterns in time-series data using an open-source, freely downloadable software, Time-Course Inspector (TCI) (Fig. 2) [3]. TCI is a web application written in the R/Shiny framework [8–10] and can be run locally from within an R environment or can be deployed on a server. TCI provides a simple, yet flexible graphical user interface (GUI) to analyze, visualize, and cluster time-series without any programming knowledge.

This protocol is illustrated with an example dataset from a previously published microfluidic experiment in which growth-medium-starved PC-12 cells were treated with an epidermal growth factor (EGF) at four different concentrations ranging from 0.25 to 250 ng/ml [4]. The readout was the activity of extracellular signal-regulated kinase (ERK) measured at 2′ intervals using a fluorescence resonance energy transfer (FRET)-based biosensor [11]. We used CellProfiler [12] for image segmentation and cell tracking. The dataset is one of the test cases included in the GitHub repository with the source code of TCI (https://github.com/pertzlab/shiny-timecourse-inspector).

The EGF treatment induces transient responses of ERK activity in PC-12 cells [4, 5] (Fig. 2). After a steep increase in response to the stimulation, the ERK activity decreases over time with decay rates that reflect the strength of the negative feedback in the MAPK network. The increase in the EGF stimulation leads to a stronger activation of the negative feedback which, in turn, leads to a faster decay of the response. However, due to the inherent cell-cell variability, the response dynamics, and hence the decay rate, varies

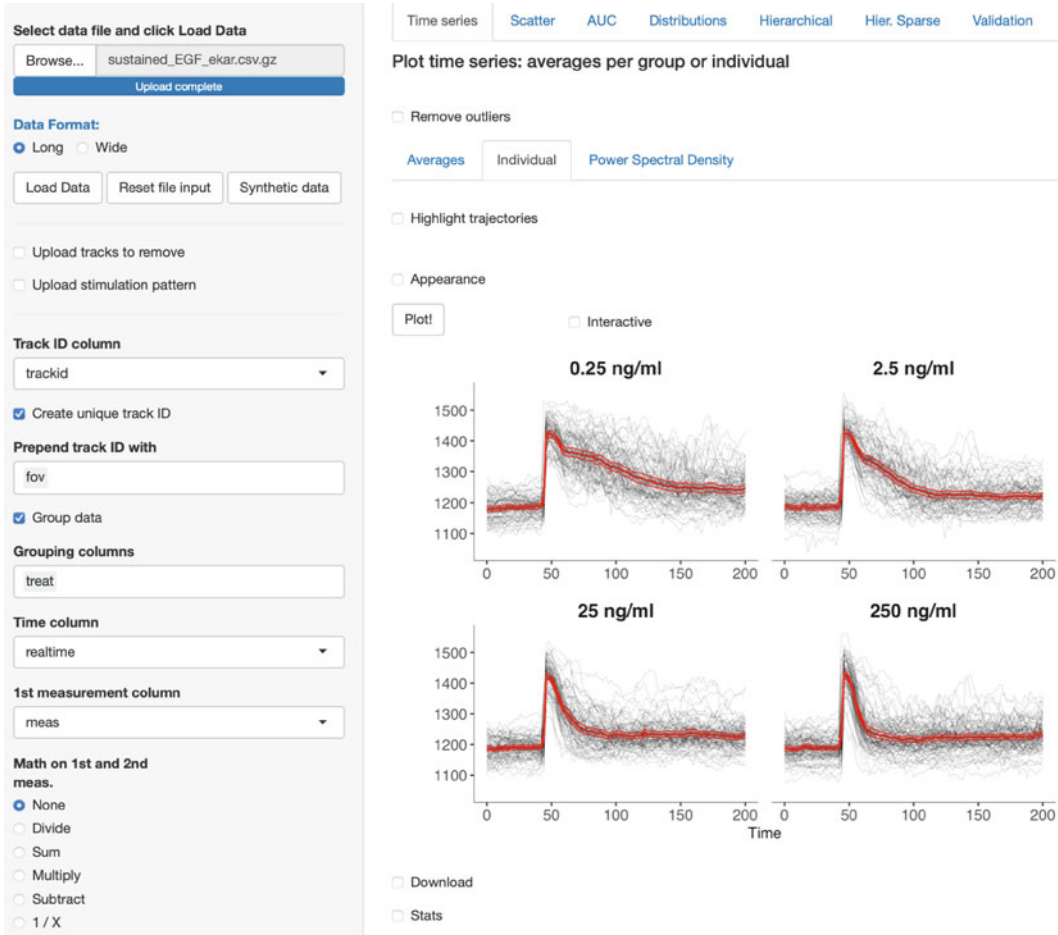

Fig. 2 Screenshot of TCI's interface with plots of single-cell ERK activity dynamics in PC-12 cells in response to EGF treatments (from the demo dataset)

significantly between cells. Therefore, for some cells the ERK activity dynamics in response to a low GF concentration may look like the response to a higher concentration. To dissect this continuum of responses and to identify prototypic dynamic patterns we pool data from four EGF treatments with increasing concentrations, and we use TCI to cluster the time-series and explore the results.

2 Materials

2.1 A Desktop or Laptop Computer

1. TCI was tested on major operating systems including macOS 11, Windows 10, Ubuntu 16, 18 and 20. There are no specific hardware requirements, as long as they meet the requirements of the R programming language interpreter. Working with larger datasets might require more RAM (>4 GB) and a

relatively modern CPU for acceptable performance. In the example presented in the following sections, each operation should take under a couple of seconds on a mid-range personal computer.

2. TCI requires a modern web browser to run. To this day, the list of the browsers that are officially supported by the R/Shiny framework includes Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge. RStudio's built-in browser can be also used when running TCI from this environment.

2.2 R Installation

1. TCI is written in a statistical programming language R [10] and will work on any operating system that has a working installation of a recent R environment (recommended version >4.0). Currently all major operating systems such as Windows, macOS, or Linux/Raspbian distributions are supported. Installation instructions can be found on the official R-project website (https://www.r-project.org).
2. As a minimum requirement, install the R *shiny* package [9] by typing in the R's console: install.packages("shiny").
3. For new R users, once you have installed R, we recommend using R from RStudio. It provides an excellent integrated development environment (IDE) and makes it straightforward to run TCI with a single click. Installation instructions can be found on the RStudio website (https://rstudio.com).
4. R is a free software distributed under GNU General Public License and is free to use. RStudio is distributed under AGPL v3 License and offers a free (but complete) version for non-commercial use.

2.3 TCI Installation

1. Download the latest version of the app directly from the GitHub repository (https://github.com/pertzlab/shiny-timecourse-inspector). Click the green button *Clone or download* to download a compressed ZIP file containing the whole directory. Unzip the file and place the resulting folder in your favorite location.
2. TCI relies on several additional R packages, which are installed automatically upon the first run of the app. For the complete list of dependencies, please refer to the TCI GitHub repository. Should the automatic installation of packages upon the app's first run fail, all missing dependencies can be installed manually using the install.packages() command in the R console.

2.4 Time-Series Data

1. The app recognizes CSV (comma-separated values) files where data columns are separated by a comma and floating-point numbers use a dot (full-stop). Compressed CSV files in .gz or .bz2 format can be uploaded directly without decompression.

Table 1
A table in long format, as expected by TCI

Group	Cell_ID	Time	Measurement_1	Measurement_2
A	1	1	3.3	0.2
A	1	2	2.1	1.1
A	1	3	4.3	0.3
A	2	1	2.8	0.1
A	2	2	1.9	0.6
A	2	3	1.7	1.0
A	2	4	2.2	0.9
B	1	1	5.1	0.5
B	1	2	5.4	0.8
B	1	3	5.3	1.1

In this example, the time-series are bivariate, *i.e.*, two measurements are performed in parallel in single cells. More measurements can be added as separate columns. The *Group* column indicates the treatment, the *Cell_ID* column identifies cells within the group, and *Time* indicates time-lapse frames

Table 2
A table in wide format, as expected by TCI

Group	Cell_ID	Time_point_1	Time_point_2	Time_point_3	...
A	1	3.3	2.1	4.3	...
A	2	2.8	1.9	1.7	...
B	1	5.1	5.4	5.3	...
B	2	4.3	4.9	4.0	...
B	3	6.7	6.1	6.6	...

The column names corresponding to time points must be numeric. The first two columns must indicate the grouping and time-series IDs, respectively

Both long and wide data formats are accepted but we highly recommend using the former, for it is more flexible, allows for multiple groupings and multivariate measurements. Each row in the long format corresponds to a single time point (*cf.* Table 1).

2. In wide format, entire univariate time-series are stored as rows, with columns treated as time points. The first two columns should contain a grouping and the identifier of time-series (*cf.* Table 2).

Table 3
Prefix the ID column to make track IDs unique across the entire dataset

Column to prepend	Non-unique ID	=>	Unique ID
A	1		A_1
A	2		A_2
B	1		B_1

Note that before this operation the ID 1 is unique only within the group. Typically, the column to prepend may correspond to the field of view or the treatment group

3. Wide format may seem more intuitive to store time-series and this would likely be the format of choice when storing data in a spreadsheet. However, multivariate time-series can be easily stored in a long format by simply adding additional columns with measurements. TCI can then be used to perform simple operations on any two chosen columns before further analysis. Additional grouping columns can be also easily included in the long format.
4. Track labels need to be unique across the entire dataset to plot, analyze, and cluster time-series correctly in TCI (*see* **Note 1**). If track identifiers (IDs) are not unique in the uploaded dataset, the user interface (UI) offers an option to create a unique track ID. Check the *Create unique track ID* checkbox in the left panel of the app and choose grouping columns to prepend the existing non-unique track label. Table 3 shows an example of creating new unique IDs.

2.5 Optional Data

1. In addition to the main dataset, a list can be uploaded with identifiers of time-series to remove from the dataset. It is important to use the same identifiers as in the main data set. If a unique identifier is created in the app's UI, for example by combining a grouping column and the identifier, the uploaded list should contain identifiers in the same format.
2. An additional file that can be uploaded into TCI is the CSV file with markers to plot underneath time-series. Markers are plotted as short line segments; XY coordinates of the beginning and end of the segment need to be specified as a single row in the file. A grouping column can be used to plot different segments according to grouping in the main dataset. Table 4 shows an example of such a dataset with markers.

Table 4
An example table to add stimulation markers under the plot of time-series

tstart	tend	ystart	yend	group	id
1	10	0	0	A	1
1	10	0	0	B	1
12	17	0	0	B	2
7	21	0	0	C	1

Each row corresponds to a marker, which is represented as a horizontal segment. The columns *tstart* and *tend* indicate the beginning and the end times (*i.e.*, the *X*-axis coordinates) of a marker, respectively. The columns *ystart* and *yend* indicate the start and end levels (*i.e.*, the *Y*-axis coordinates) of a marker, respectively. The column *group* is used along the grouping option of the plots to match the markers to different groups. The *id* column indicates the marker *id* within a group. The column names must be respected

2.6 Demo Data

1. The folder example-data/test-case-2 in the GitHub repository contains a compressed CSV file, sustained_EGF_ekar.csv.gz, with single-cell time-series data in long format. This dataset will be used throughout this protocol to demonstrate the clustering of ERK activity dynamics. The file consists of the following columns:
   - *treat*, which corresponds to four treatments with different EGF concentrations applied in four separate channels (i.e., wells) of the microfluidic device.
   - *fov*, which holds the number of the field of view; for every treatment, images were acquired at four different locations, fields of view (FOVs), per microfluidic channel. The *fov* 1–4 corresponds to the first treatment, 5–8 to the second, etc.
   - *frame* and *realtime*, which hold consecutive frame numbers and the real time of the experiment, respectively.
   - *trackid* with the identifier of a time-series, which is unique only within a FOV.
   - *meas* with the measurement of the FRET ratio.
2. The folder also contains the following files:
   - sustained_EGF_badTraj.csv with a single column of time-series identifiers of outlier cells. This file will be used later to remove such tracks from the dataset.
   - sustained_EGF_y1100.csv and sustained_EGF_y0-97.csv with segments to plot underneath the time-series and to indicate the treatment duration. The two files contain, respectively, segments positioned at the *y*-axis level 1100 to plot along unprocessed data and at 0.97 to plot together with normalized time-series.
   - tCoursesProcessed.csv.gz with trimmed, normalized time-series without outliers. This dataset is the result of the

pre-processing using TCI's built-in functions as demonstrated further in this protocol.

3 Methods

The starting point of the app is a plain spreadsheet in comma-separated values format (CSV) that contains the dataset to analyze. TCI embeds a module for simple pre-processing of the data (trimming, normalization, interpolation of missing values), various visualizations, common statistics reports, spectral decomposition, a flexible module for hierarchical clustering and cluster validation. All modules are documented with tooltips and *Learn More* sections to guide users through the UI and assist them with the analysis.

3.1 Starting the App

1. If you have installed RStudio, launch it and go to *File* > *Open Project*. In the contextual menu navigate to the location where you placed the app and open the file tcourse-inspector.Rproj. This will load the app in the current RStudio session. To start the app, either:
   (a) type shiny::runApp() in the R console, or
   (b) open the server.R or the ui.R file in the RStudio session, then click the *Run App* button with a green triangle in the upper right corner of the window with the code.

   In either of the cases, a browser window with the app will open.
2. If you did not install RStudio, or do not wish to use it, you can also start TCI directly from your OS's command line with: R -e "shiny::runApp('path-to-application-folder')".

   Then, open your web browser and point to the address provided by the output of that command.

3.2 The Interface

The app's interface has a classic outline with a panel to load and pre-process data on the left and tabs to switch between modules on the top (Fig. 2):

1. The top left panel deals with uploading of: the main data file, identifiers of outlier time-series to remove, and positions of segments to draw under the time-series. Such segments can be used to indicate the timing of experimental perturbations.
2. Essential columns for further processing such as time-series identifiers, the time and measurement variables can be selected in the middle-left panel. Some column names are automatically recognized in the dataset. Simple arithmetic operations can be performed after selecting two measurement columns.

3. Pre-processing options for data trimming, interpolation, and normalization are available in the lower left panel. An additional pre-processing to remove point outliers is available in the *Time-series* tab. Data modified using these options can be downloaded as a long-format CSV file.
4. Plots are displayed in the middle region and the information about time-series can be displayed as a tooltip in gray hover box when the interactive mode is activated. The interactive mode uses the R *plotly* package, which allows for panning and zooming in the regions of interest.
5. Plots can be downloaded using the UI at the bottom as a PDF file with dimensions specified in the option fields. Plots can also be exported as RDS files that specify an R *ggplot2* data object used to create the plot. Such an object can be later loaded in an R session with the command:

   plotObject <- readRDS('path/to/file')

   This option is useful to further modify the plot, for example using an interactive plot editor available from the R *ggedit* package.

3.3 Loading Data

1. Click the *Browse...* button in the upper left corner to select the file with the input dataset, then press the *Load Data* button. The demo file sustained_EGF_ekar.csv.gz is already in long format. Upon loading the file, the app recognizes several common column names and prepopulates the corresponding fields in the UI.
2. The *Track ID column* field requires a time-series identifier, which should be unique across the entire dataset. If an identifier is not unique, the existing identifier can be prepended with additional columns. In the demonstration dataset, the *trackid* column holds identifiers that are unique only within a field of view. To make it unique, click the *Create unique track ID* checkbox and select the *fov* column to add as a prefix.
3. The *Group data* checkbox and the associated *Grouping columns* field specify grouping columns, which is useful to display statistics or plots per treatment group. Select the *treat* column in the demo dataset to group the time-series per GF concentration.
4. The *Time column* field expects the name of the time variable. Select the *realtime* column.
5. The *first* and *second measurement column* fields are for specifying the measurement variable. The field for the second measurement appears when one of the arithmetic operations is selected. The demo dataset contains only a single measurement variable. Select the *meas* column.

3.4 Plotting Data

1. Click the *Plot!* button in the central part of the UI to plot group averages or individual time-series per group (Fig. 2). The *Plot!* button needs to be clicked to refresh the plot upon changes in the left-hand-side panel. The *Interactive* checkbox switches to the interactive plot mode in which the information about individual trajectories is displayed when hovering over the plot with a mouse. This option becomes handy during identification of outlier time-series. The time-series identifier displayed in the tooltip can be stored manually in a single-column CSV file and uploaded via the *Upload tracks to remove* option in the left-hand-side panel.
2. The *Stats* checkbox underneath the plot displays summary tables with the information about the tracks, the measurement, and duplicated time-series identifiers. If records appear in this tab, unique identifiers need to be created either in the input dataset or using the UI in the left-hand-side panel as described above (*see* **Note 1**).

3.5 Outliers

Typically, there are two types of outliers in longitudinal datasets: point and dynamic outliers.

Point outliers are individual time points that strongly exceed the range of the remaining data. They typically arise due to technical issues with image acquisition, e.g., debris in the field of view, failed image segmentation, or object tracking. Point outliers can be removed in TCI by setting a threshold to remove tails from the distribution of the measurement pooled from all groups. The size of the tail(s) is determined by a user-defined percentage (Fig. 3a). Once the outlier time points are removed, the user can determine the maximum allowed gap duration due to removal of point outliers. Tracks with gaps longer than the threshold will be removed from the dataset. Gaps smaller or equal to the user-defined threshold will be left in the dataset or can be interpolated. By choosing the latter, the user must provide an interval between time points in the UI. This procedure will also interpolate any pre-existing NAs and missing data, not only those due to removed outliers. The list of removed track identifiers can be downloaded as a CSV file.

The demo dataset contains one very strong point outlier with ERK activity FRET ratio equal to ~4060. The outlier time point is in the track number "1_22" (EGF 250 ng/ml) at a 44-min time mark. To remove this point and interpolate the resulting gap:

1. Click the *Remove outliers* checkbox and set the percentage of data to 0.003%.
2. Set the slider in the *Max allowed gap duration* to 1 time point. This will ensure that tracks with single-time-point gaps will be kept but tracks with gaps longer than 1 time frame will be removed from the dataset.

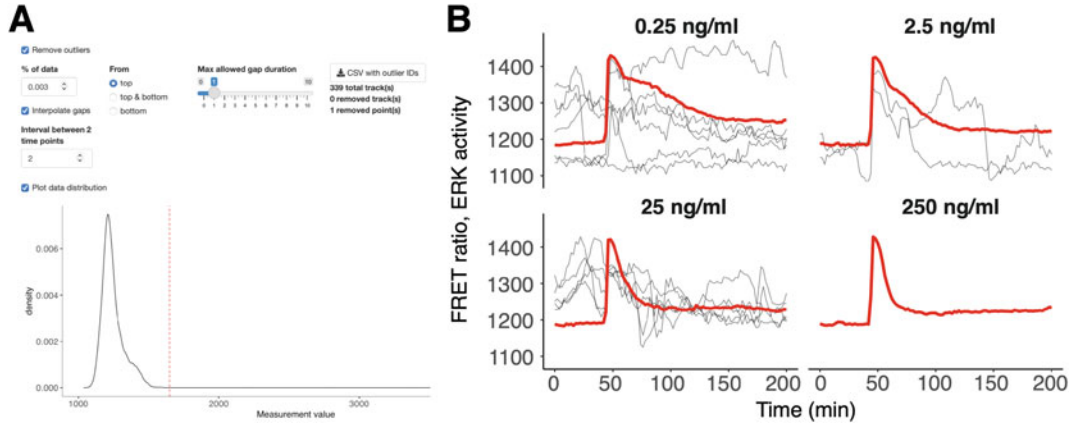

Fig. 3 Outlier detection and removal in the demo dataset. (**a**) TCI's module for the removal of point outliers based on the distribution of the measurement pooled from all time points and time-series. By setting the percentage of data points to remove the user can trim tails of the distribution. Gaps in time-series that arise from the removal of outlier points can be interpolated. A download button can be used to obtain a single-column table with outlier track IDs. (**b**) Manually selected dynamic outliers available in the sustained_EGF_-badTraj.csv file of the demo dataset. Red lines indicate the average of all time-series in the corresponding groups

3. Gaps that arise due to the removal of point outliers can be interpolated. Click the *Interpolate gaps* checkbox and set the interval between time points to 2 because we are using the *realtime* column with **step 2** for the time axis. Press the *Plot!* button to redraw the figure.

   Dynamic outliers are entire trajectories that behave differently from the rest of the population. For example, when their trend is different, or their average amplitude is much higher. Dynamic outliers can also emerge due to failed object tracking, e.g., when track identifiers of two neighboring objects are flipped during the experiment. A sudden change in the measured properties is then readily visible and can be identified using TCI's interactive plots. Such a change can also stem from the underlying biology, and we advise extreme caution before identifying data as outliers. By activating the interactive plot, the user can hover the mouse pointer over the time-series to identify track IDs of dynamic outliers. A user-created CSV file with identifiers of outlier tracks can be then uploaded into the app to remove such tracks from further analysis.

   Figure 3b shows outlier time-series that were selected manually using the interactive plotting mode of the TCI to obtain unique track IDs. These IDs were then saved to a single-column CSV file and uploaded into the TCI. By cross-checking against raw images, we confirmed that sharp signal fluctuations and spurious peaks resulted from significant image

segmentation errors. A very low signal of several trajectories in the right panel results from cells undergoing cell division.

4. To remove outliers stored in a CSV file check the *Upload tracks to remove* checkbox and select the sustained_EGF_badTraj.csv file provided in the example-data/test-case-2 folder of the GitHub repository. Press the *Load Data* button to finalize the upload and the *Plot!* button to redraw the figure.

3.6 Trimming

1. For further processing and to expose interesting features of the dynamics to the clustering algorithm trim the time-series to the interval 20′–150′. Check the *Trim X-axis* checkbox and select the appropriate range with the slider (*see* **Note 2**).

3.7 Missing Data

1. There are two types of missing data to be aware of when working with TCI: (1) explicit NAs in the measurement column, (2) missing rows in the long-format data (*see* **Notes 3** and **4**). The UI has an option to linearly interpolate both, explicit NAs, and missing data rows. For the latter, the UI requires the user to provide the interval between time points. Since we are using the *realtime* column as the time variable, set the interval to 2. Press the *Plot!* button to redraw the figure.

3.8 Normalization

The baseline before the stimulation exhibits large variability, which may come from technical and/or biological noise. Such a variability creates an additional confounding factor in our pursuit of clustering ERK activity dynamics and identifying prototypical dynamic patterns. In this analysis, we do not wish to cluster the time-series based on the baseline, but rather to focus on the peak height and the relaxation dynamics after the peak.

To verify whether normalization with respect to the baseline is permissible in this dataset, switch to the *Scatter* tab to explore the correlation of the measurement between any two time points. We are interested in finding out whether the peak height is related to the baseline. For example, if cells with low baseline ERK activity also had a lower peak amplitude, or vice versa. If there is no correlation between the two, we can safely normalize every time-series to its baseline.

1. Switch to the *Scatter* tab to correlate the measurement between any two time points.
2. Choose time points 30′ (baseline) and 48′ (peak) for the x- and y-axes in the scatter plot, respectively.
3. Set the *Smoothing* window to ±2 frames (equivalent to ±4′) to smooth spurious frame-to-frame fluctuations of the measurement.
4. Instead of plotting the actual peak value on the y-axis, switch to plotting the "Y–X" difference. The y-axis will then display the

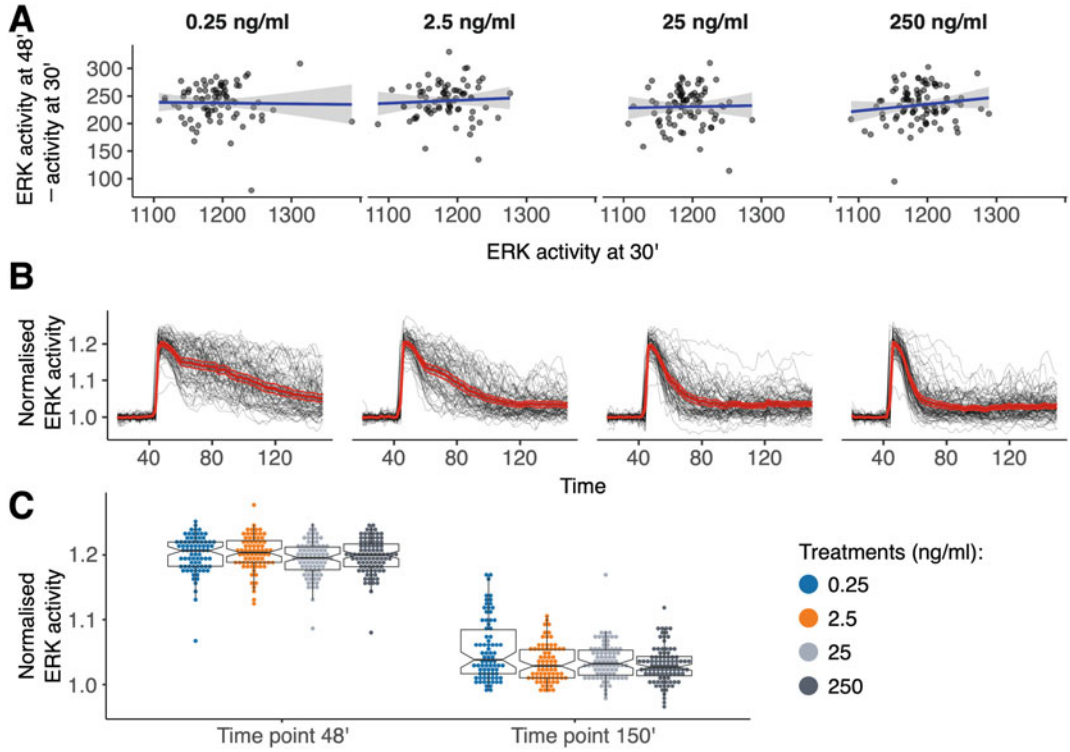

Fig. 4 Visualizations of the demo dataset. (**a**) Scatter plot between the baseline measurement at 30′ and the peak amplitude at 48′ calculated as the measurement difference between points at 48′ and 30′. The blue line indicates a linear regression fit and the gray shading the 95% confidence interval (CI). The linear regression demonstrates that the peak amplitude does not depend on the baseline magnitude. (**b**) Final dataset after trimming, removal of outliers, and normalization available in the tCoursesProcessed.csv.gz file of the demo dataset. The red line indicates the average of all time-series in the corresponding groups. The envelope indicates the 95% CI for the mean. (**c**) Distributions of normalized ERK activity at the peak at 48′ and at the late decay at 150′. Notably, the peak distribution is the same across four EGF concentrations. The box-plot notches represent approximately 95% CI for the median [13]

difference between the measurements at 48′ and 30′, which corresponds to the peak height relative to the baseline.

5. The scatter plots for every treatment group show that there is no statistically significant correlation between the baseline magnitude and the peak amplitude (i.e., peak magnitude relative to the baseline) (Fig. 4a). This implies that the peak amplitude due to the GF stimulation is independent of the baseline and we can normalize every time-series to its own baseline.

Measurements can be divided by the average of data points in a selected interval to calculate the fold-change with respect to the entire dataset, a group, or a single time-series. The latter would normalize every time-series to the mean of its own selected interval.

Instead of the mean, a z-score, $\frac{x-\mu}{\sigma}$ (where μ—the population mean, σ—population standard deviation) is also available. When

Robust stats option is active, the fold-change and *z*-score normalization is calculated using the median and Median Absolute Deviation (MAD) instead of the mean and standard deviation, respectively.

1. To normalize, activate the *Normalization* checkbox, choose the *fold-change* method, and select the time window 20′–40′ using the slider. This normalization interval is right before the peak. For the *Grouping*, choose the *Per trajectory* option.

Figure 4b shows single-cell time-series of the final dataset without point and dynamic outliers, with time-series trimmed to 20′–150′ and individually normalized to the 20′–40′ interval. This dataset is provided in the example-data/test-case-2/tCoursesProcessed.csv.gz file of the GitHub repository.

3.9 Distributions

An interesting feature of the MAPK signaling network is its ultrasensitive character due the cascaded, three-tier architecture [14]. Consequently, small changes in the input—the GF perturbation, can induce large changes in the output—the ERK activity. Due to this steep dose response, the initial response phase after the GF perturbation results in a full peak, whose amplitude is largely independent of the GF concentration.

The negative feedback present in the MAPK network [7] facilitates rapid, in the range of 30′ min, attenuation of the ERK response. This is an important mechanism to prevent persistent ERK activation, which would prompt starved PC-12 cells to differentiate. By limiting this signal, the cell collective can maintain an equilibrium between cell proliferation and differentiation [5]. Four different EGF concentrations used in the demo dataset activate the negative feedback to a different degree and result in a weak to strong attenuation of the ERK response. The strength of the negative feedback also affects the attenuation of the cell-cell variability. Therefore, we expect that when the negative feedback is weak, the variability of ERK responses will be the highest.

1. To inspect the two above-mentioned phenomena, switch to the *Distributions* module. Select two time points, time point 48′ (at the peak) and 150′ (at the late decay) and click *Box-plot* and *Dot-plot* checkboxes to plot the distributions of ERK activity. Adjust the number of bins in the dot-plot as desired and the space between groups slider if necessary.
2. The medians and distributions of ERK activity are almost the same for all GF concentrations at the peak (Fig. 4c). However, at the late decay point the distribution for the lowest concentration, EGF 0.25 ng/ml, is wider than the rest. To quantify this observation, activate the *Stats* checkbox. The robust coefficient of variation (rCV) in the summary table is around 0.07

for 0.25 ng/ml but oscillates around 0.03 for all remaining concentrations.

3.10 Clustering

TCI performs agglomerative hierarchical clustering [15] on data pooled from all groups (experimental conditions). The result of clustering is represented by a heatmap with rows corresponding to individual time-series and columns corresponding to time points. The rows are arranged based on the relative distance between the time-series. TCI offers several distance metrics available in R. For high-dimensional data such as time-series, we recommend using Manhattan distance. It is less likely to be influenced by outliers compared to Euclidean or other higher dimensional norms [16]. Dynamic time warping (DTW) [17] is a particularly useful metric to compute similarities between asynchronous time-series that have features relevant for clustering but shifted in time. Conventional metrics such as Euclidean distance calculate the distance independently for every time point. Thus, for example two time-series with an identical peak that occurs at different time points will have a large relative Euclidean distance. DTW takes a different approach by trying to align such shapes (*see* **Note 5**), hence reducing the relative distance between asynchronous trajectories.

The dendrogram on the left of the heatmap illustrates the distance hierarchy between the time-series. A useful feature is a slider to interactively *cut* the dendrogram at a desired level and to highlight major dendrogram branches. Cutting a dendrogram is effectively performing a clustering, where the isolated branches form the clusters. After cutting the dendrogram, cluster averages and time-series within such clusters can be plotted in other tabs. If grouping is present in the dataset, it is possible to display a stacked bar plot with the fraction of time-series from different clusters in every treatment group.

Sparse hierarchical clustering [18] is available in a separate module and is suitable for *sparse* datasets, where the number of time points is much larger than the number of time-series. In such cases, time-series may differ only with respect to a small fraction of time points. Consequently, clusters may not be distinguished properly if all time points are included, especially those that have similar measurement values across the time-series. The sparse clustering algorithm circumvents this issue by weighing the time points and discarding those with low weights that do not contribute to clustering. In TCI, time point weights are indicated in column labels of the heatmap plot.

Interactive exploration of various distance and linkage methods was one of the main drivers behind the development of TCI. There is no single clustering method that applies to all datasets, nor all datasets have clusters. Nonetheless, the aim should be to choose a distance/linkage combination such that the dendrogram is possibly symmetric and balanced. For example, outliers can skew the

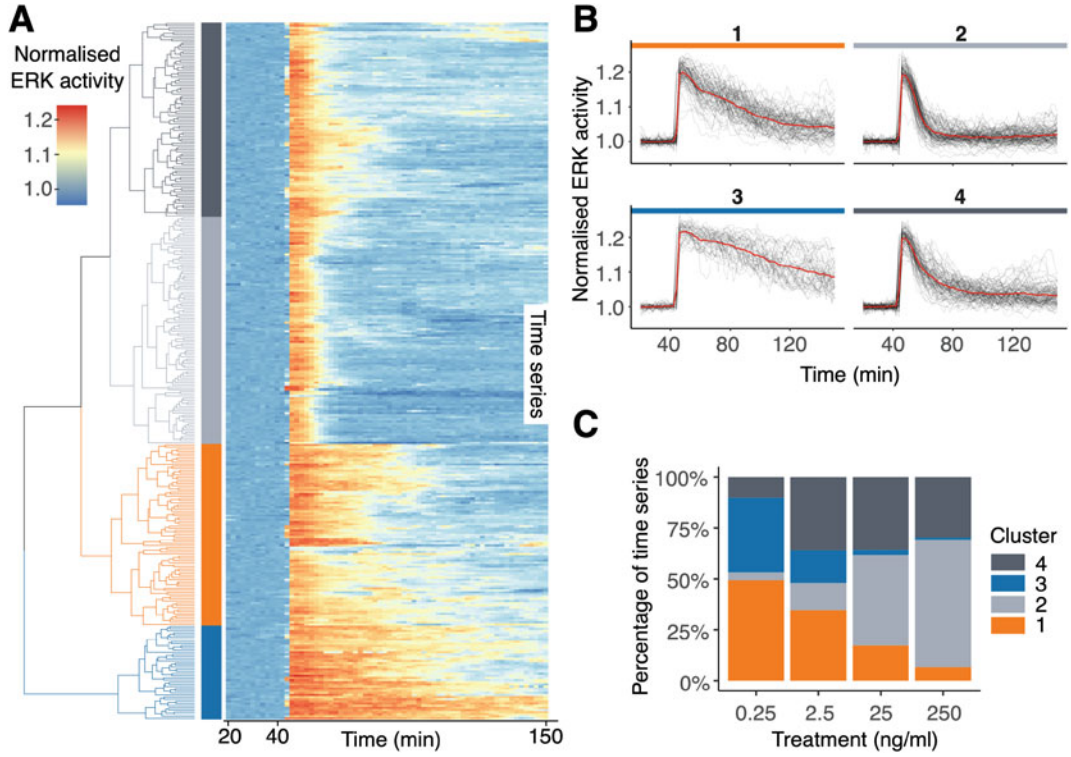

Fig. 5 Hierarchical clustering of the demo dataset. (**a**) Heatmap with dendrogram of clustering with Euclidean distance and complete linkage method. For visualization and plotting of cluster groups, the dendrogram has been cut to highlight four main branches. The colors of the four resulting clusters are matching the colors in (**b**) and (**c**). (**b**) Single-cell time-series from different clusters. The red line indicates the average of all time-series in the corresponding cluster. Note that the time-series in a cluster may come from different treatments. (**c**) Contribution of clusters across treatments. Low EGF treatments comprise mainly slowly decaying responses, while higher concentrations consist of transient dynamics

dendrogram and can form a small but distinct cluster. To avoid that, consider removing the outliers discovered in the heatmap from the dataset (*see* **Note 6**). A helpful visual cue for choosing the cut level for the dendrogram is trying to ensure that the mean dynamics of the resulting clusters corresponds well to the dynamics of individual time-series (*see* **Note 7**).

To cluster the demo dataset:

1. Go to the *Hierarchical* module of the TCI, choose the *Complete* linkage and the *Euclidean* distance, and click the *Plot!* button.
2. To highlight major branches of the dendrogram, change the *Number of dendrogram branches to cut* slider to 4 (Fig. 5a).
3. Cluster averages and individual time-series within clusters are available in *Cluster averages* and *Time-series in clusters* tabs. We recognize that the four major clusters correspond to distinct

ERK activity patterns that differ with respect to the decay rate after an initial peak (Fig. 5b). The cluster averages represent rather faithfully the behavior of individual time-series within clusters. We will quantify the quality of this clustering in the section below.

4. The *Cluster distribution* tab calculates fractions of time-series that belong to clusters displayed as a stacked bar plot for every treatment condition (Fig. 5c). We observe that ERK dynamics in response to the lowest treatment, EGF 0.25 ng/ml, consists of time-series from clusters 1 and 3 which have a slow decay rate. This implies that the negative feedback in this treatment was weak and the sudden ERK activation was attenuated slowly. With the increase in the EGF concentration, clusters 2 and 4 with more transient dynamic patterns increase their presence. This reflects the biological reality when high EGF concentrations activate the negative feedback. The cluster distribution plot offers a tangible demonstration that signaling responses to perturbations are highly heterogeneous and that population averages are insufficient to describe the dynamic behavior of the signaling network. A population-wide response to a particular GF concentration can in fact consist of subpopulations that respond in starkly different ways. Clustering as demonstrated in this protocol can reveal these hidden behaviors.

3.11 Cluster Validation

Hierarchical clustering produces a dendrogram with a hierarchy of distances between individual time-series. A common procedure is to cut the dendrogram at a manually chosen level to highlight the main branches that correspond to main clusters in the dataset. Though the process of finding the right cut is empirical and should be done in an interactive, iterative fashion, some metrics can provide support for a given clustering (*see* **Note 8**). TCI integrates two types of cluster validation implemented in the R package *factoextra*: relative and internal.

The relative cluster validation sweeps through a range of possible cluster numbers and reports global metrics about the goodness of clustering. TCI returns two such metrics: the average silhouette width [19] and the within cluster sum of squares (WSS). To perform relative validation of clustering:

1. Choose the distance metric to *Euclidean* and the linkage method to *Complete* from drop-down menus and set the slider to the maximum number of clusters to consider, e.g., 10.
2. Figure 6a summarizes the average silhouette analysis, which computes how close each trajectory is to other time-series in its own cluster. This is then divided by the distance to trajectories in other clusters. This ratio is called the silhouette width of a trajectory. A positive silhouette width indicates that a

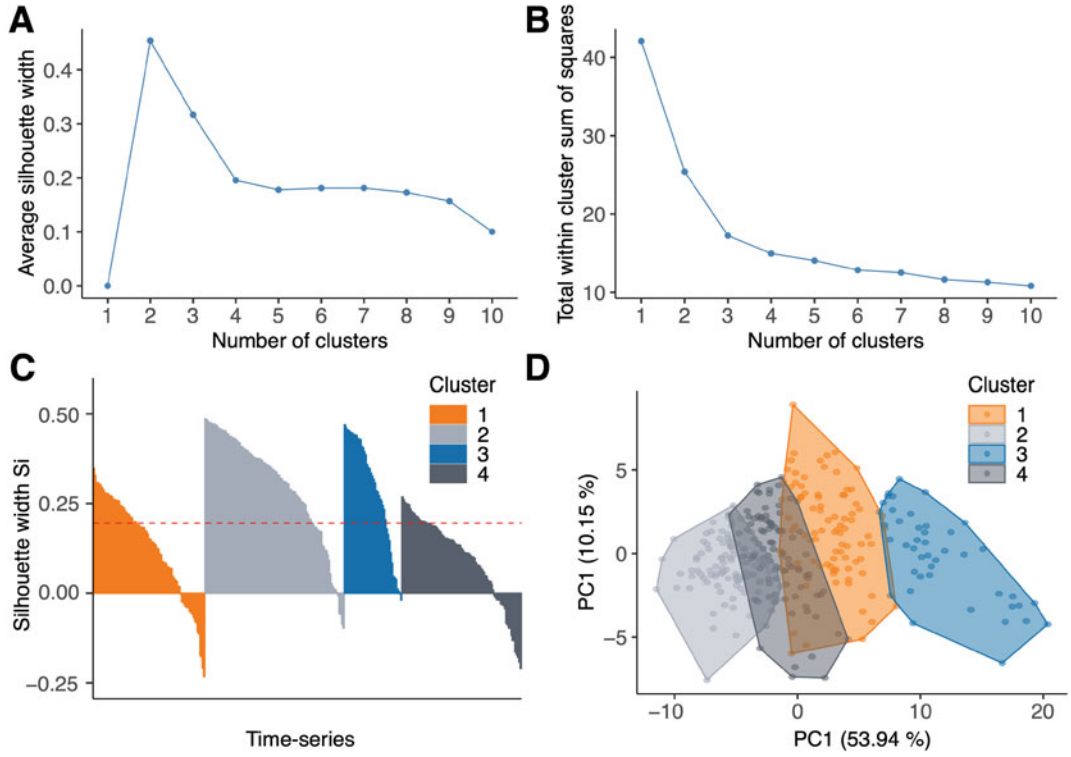

Fig. 6 Quantification of cluster validation using inter- and intra-cluster distances. Clustering of the example dataset from Fig. 5 using Euclidean distance and complete linkage. (**a**, **b**) are external clustering quality indicators. (**c**, **d**) are internal clustering quality indicators for four clusters. (**a**) Average silhouette width for a range of cluster numbers. Higher values indicate better clustering. (**b**) Within sum of squares (WSS) for a range of cluster numbers. Lower values indicate better clustering. (**c**) Silhouette plot of the clustering, where vertical bars indicate the silhouette width of single trajectories. Positive values indicate that the corresponding trajectory is on average closer to trajectories in its own cluster than to trajectories in other clusters. The red dashed line indicates the average silhouette width for four clusters as in (**a**). (**d**) Scatterplot with the two first principal components of the dataset. Each point represents a single trajectory, colored according to the clustering. Convex hulls are overlaid for each cluster

trajectory is globally closer to trajectories from its own cluster than to trajectories in other clusters. All silhouette widths of a given clustering can be averaged as an indicator of clustering quality. Hence, larger average silhouette widths usually indicate better clustering. Even though in our example the average silhouette width peaks at two clusters, further metrics must be analyzed to conclude about the optimal cluster number. For example, to make sure the averaging does not hide a locally bad clustering, the result should be crosschecked against the silhouette plot for a particular number of clusters using internal validation available in another tab (*see* **Note 9**).

3. Figure 6b shows the within cluster sum of squares (WSS), or variance, which evaluates the compactness of clusters. Compact

clusters achieve low WSS values. When plotted as function of the number of clusters, WSS would typically decrease and level off. This inflection point, i.e., *the elbow*, beyond which further increase in the number of clusters does not confer a significant decrease in WSS is considered an optimum. In our case we observe that increasing the dendrogram cut beyond four clusters does not confer a significant reduction in the total WSS.

In the second tab, *Internal*, the internal cluster validation can be calculated for a particular number of clusters. TCI offers two visualizations: a silhouette plot [19], and a visualization of the clusters using first two principal components.

1. Choose the *Euclidean* distance metric and the *Complete* linkage method from the drop-down menus and set the number of clusters to validate to 4.
2. The dendrogram in Fig. 5a shows that this partition is defining well-balanced clusters with short branches within the clusters, and that the clusters are separated with long branches. These are indications that the clustering is separating distinct and homogeneous groups.
3. The silhouette plot in Fig. 6c shows the silhouette width of each trajectory. The red dashed line indicates the average silhouette width, the same as reported for four clusters in Fig. 6a. Many silhouette widths are positive, which means that, on average, each trajectory is closer to trajectories in their own clusters than to trajectories in other clusters. This result further supports the chosen distance metric, linkage algorithm, and the number of clusters.
4. In the scatter plot in Fig. 6d, each point corresponds to a single trajectory in the space spanned by the first two principal components of the dataset. It shows the formation of compact clusters, which group trajectories from contiguous regions of the space. This plot confirms that the clustering is indeed grouping similar trajectories. One can also notice that the cluster 4 overlaps partially with clusters 1 and 2, which indicates that this part of the clustering is weaker. Consistently, in Fig. 6c, there are more negative silhouette widths for cluster 4, which are likely linked to trajectories that overlap with clusters 1 and 2.

3.12 Conclusions

This protocol describes how to explore subpopulations of cells with distinct signaling dynamics in time-series from live single-cell fluorescent measurements. The workflow and methods implemented in TCI reflect our daily experience with such datasets, which are often plagued by technical noise such as missing values or outliers. Thanks to visualizations such as scatter plots between two time

points, or measurement distributions at selected time points, we can quickly inspect the data and decide about further analysis.

We believe that the strongest feature of TCI is the interactive clustering module with access to several distance and clustering methods, which is crucial for choosing the best partitioning of the data that fits the focus of analysis. For example, in the demo dataset discussed in this protocol we were interested in isolating subpopulations of time-series that differed with respect to the decay time after the initial peak. To this end we trimmed the time-series by removing the initial baseline run-in time and the last 50′. This, together with normalization of each time-series to its own baseline helped to emphasize the important part of the dataset, which is the peak and the decay. Only then we clustered the time-series, which led us to choose the partitioning into four clusters. As demonstrated by the scatter plot of first two principal components in Fig. 6d, there is no definite grouping in the dataset. However, here the two components explain only ~65% of the total variance. Further analysis of the inter- and intra-cluster distance in the validation module confirmed that the four-cluster partitioning is a good compromise, which yields human-interpretable clusters. We could clearly recognize subpopulations with strongly transient and more sustained ERK responses to the EGF treatment. Interestingly, we find that each EGF concentration induces very heterogeneous responses with the same peak amplitude but strikingly different decay rates. The effect is the strongest for the two lowest EGF concentrations, which consist of responses that belong to clusters with high and low decay rates.

In the original study [4], we observed that the lowest EGF treatment that induces slow, adaptative, sustained-like ERK activity is sufficient to differentiate a part of the cell population. Strongly transient responses due to the negative feedback in the MAPK network resulted in very low differentiation rates comparable to the untreated case. This is in line with the fact that PC-12 cells strongly differentiate in response to the sustained ERK activation induced by the nerve growth factor (NGF). In this case the sustained dynamics is possible due to activation of a positive feedback that overpowers signal attenuation by the negative feedback. Thus, the cell fate is determined by the signaling dynamics rather than by the steady state. Modulation of the signaling dynamics by titrating the GF as shown in this protocol or the induction of synthetic signaling patterns as previously demonstrated [4, 5] determines the proportion of PC-12 cells that differentiate. Thanks to clustering we could isolate the dynamic phenotype with slowly decaying ERK responses that contribute to higher cell differentiation.

It is worth mentioning that coding TCI by programmers without prior experience in web app development was largely enabled by the convenience of the R/Shiny framework. Despite a slight learning curve in understanding reactive programming and

familiarizing oneself with the GUI design, R/Shiny is a powerful tool to turn the existing R data analysis code into an interactive, platform-independent application. We strongly believe that such tools have a potential to empower future life-scientists to quantitatively explore their data and encourage them to turn their existing code base into interactive data exploration tools.

4 Notes

1. Non-unique time-series identifier. The most common pitfall when plotting time-series data is the lack of a data-wide unique time-series identifier. This will result in a characteristic jagged, sawtooth pattern in single-cell time-series plots because the plotting function groups different trajectories together. Data from a single fluorescent microscopy experiment often contains acquisitions performed in multiple, independent wells or fields of view (FOVs). Image segmentation and tracking algorithms often assign a track identifier that is unique only within a group, i.e., a well or a FOV. One solution is to create a unique combination of the well, FOV and the track ID before uploading the data into TCI. Alternatively, TCI has an option to combine several fields in the app. To check for the presence of non-unique IDs, activate the *Stats* section underneath the plot in the *Time-series* module. The subsection *Duplicated IDs* displays trajectories with duplicated track IDs, which can also result from the absence of a unique ID.
2. Bad clustering can emerge when a distance measure puts too much weight on irrelevant parts of the data. For example, in the EGF-stimulation example, we excluded the pre-treatment phase from the clustering analysis. This is because we do not want the distance between resting baselines to influence the distance between trajectories. Additionally, the data range also matters for computing the distances. If you want to compare the relative variations in trajectories (*i.e.,* the shape) without regard to the absolute measurement levels, consider normalizing the data before computing the distances.
3. The module for outlier removal also includes an option to interpolate gaps due to removed point outliers. If activated, the pre-existing NAs and missing time points will be also interpolated using a time interval provided in the UI of the outlier removal module.
4. Clustering and cluster validation methods such as the Dynamic Time Warping (DTW) distance or the principal component analysis (PCA) will not work when the missing data are present.

5. If the dynamics you are trying to capture are asynchronous across the trajectories, try using the dynamic time warping (DTW) distance. This distance will try to align the trajectories based on their shape, while other distances such as Euclidean or Manhattan compute distances at fixed time points.
6. Outlier time points with values largely exceeding the rest of the dataset, can heavily skew the color scale of the heatmap in the clustering module. This results in homogenous heatmaps where only the outlier points have distinctive colors, and all fine variations in the remaining data are indiscernible. To solve this issue directly from within the TCI, consider removing the outlier time points or normalizing the data. Another solution could be to transform the measurement values (*e.g.*, log-transform for right-skewed data) prior to uploading in TCI. Tip: The outliers will often form separate branches in the clustering dendrogram. This property can be used to identify and trim the outliers as we have demonstrated in another freely available R/Shiny web application for outlier removal developed in our lab (https://github.com/pertzlab/Outlier_app).
7. Make sure that the apparent incoherence of clustering is not linked to its visualization. For example, if you use the DTW distance to capture asynchronous dynamics across the trajectories, the cluster averages might appear flat for all clusters. Instead, try inspecting individual trajectories in the clusters and see if you can identify a common pattern. You can also try to visualize the power spectrum densities (PSD) of the clusters. This decomposition is frequency-based and is insensitive to the actual time point at which specific dynamics appear. We recommend this approach for oscillating trajectories.
8. Clustering is an empirical approach and what defines a "good" clustering depends on the aspects of the data you would like the clustering to capture. Nevertheless, you can get some support for a given clustering with the clustering validation measures. These measures can also help you to narrow down the range of parameters you should explore.
9. How to choose the right number of clusters, fix bad clustering, and obtain insightful clusters? Frequently, clustering will look seemingly "bad." For example, it can group dissimilar trajectories together or miss obvious groups in the data. This most likely stems from inadequate choice of parameters: the number of clusters, a distance metric, or a linkage method. In general, before trying other distances or linkages, first try to change the number of clusters to better separate distinct dynamics. The choice of the number of clusters is an essential part of the analysis and many metrics have been developed to determine

the optimal number of clusters. In TCI, the external cluster validation tab provides useful indications for a reasonable range of the number of clusters: the average silhouette width and the within cluster sum of squares (WSS). In an ideal scenario, finding the optimal number of clusters should be as easy as maximizing the former and minimizing the latter. However, each metric captures a different aspect of cluster solidity and, together, they can even lead to incoherent indications. For example, in the example discussed above, the average silhouette width peaks at two clusters (Fig. 6a) while we retained four clusters. Also, for the WSS method the exact location of the elbow point can be difficult to spot. Rather than using these metrics as an absolute proof of a better clustering, use them as global indications to guide your analysis. In practice, determine a range of numbers that is common across the external indicators, try to use several clusters around the "optimal" ones, inspect the result, and see if you can get additional support with the internal validation metrics.

Acknowledgments

We acknowledge the financial support of the Swiss National Science Foundation (grants 31003A-163061 and 51PHPO-163583), and the Swiss Cancer League.

References

1. Niepel M, Spencer SL, Sorger PK (2009) Non-genetic cell-to-cell variability and the consequences for pharmacology. Curr Opin Chem Biol 13:556–561
2. Lee TK, Covert MW (2010) High-throughput, single-cell NF-κB dynamics. Curr Opin Genet Dev 20:677–683
3. Dobrzyński M, Jacques M-A, Pertz O (2020) Mining single-cell time-series datasets with Time Course Inspector. Bioinformatics 36: 1968–1969
4. Blum Y, Mikelson J, Dobrzyński M et al (2019) Temporal perturbation of ERK dynamics reveals network architecture of FGF2/MAPK signaling. Mol Syst Biol 15:e8947
5. Ryu H, Chung M, Dobrzyński M et al (2015) Frequency modulation of ERK activation dynamics rewires cell fate. Mol Syst Biol 11: 838
6. Purvis JE, Lahav G (2013) Encoding and decoding cellular information through signaling dynamics. Cell 152:945–956
7. Kholodenko BN, Hancock JF, Kolch W (2010) Signalling ballet in space and time. Nat Rev Mol Cell Biol 11:414–426
8. Sievert C (2020) Interactive web-based data visualization with R, plotly, and shiny. Chapman and Hall/CRC, New York
9. Chang W, Cheng J, Allaire JJ, Sievert C, Schloerke B, Xie Y, Allen J, McPherson J, Dipert A, Borges B (2021) shiny: Web Application Framework for R. R package version 1.7.1. https://CRAN.R-project.org/package=shiny
10. R Core Team (2020) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna
11. Fritz RD, Letzelter M, Reimann A et al (2013) A versatile toolkit to produce sensitive FRET biosensors to visualize signaling in time and space. Sci Signal 6:rs12
12. McQuin C, Goodman A, Chernyshev V et al (2018) CellProfiler 3.0: next-generation image processing for biology. PLoS Biol 16: e2005970

13. Mcgill R, Tukey JW, Larsen WA (1978) Variations of box plots. Am Stat 32:12–16
14. Huang CY, Ferrell JE (1996) Ultrasensitivity in the mitogen-activated protein kinase cascade. Proc Natl Acad Sci 93:10078–10083
15. Hastie T, Tibshirani R, Friedman J (2009) Unsupervised learning. In: Hastie T, Tibshirani R, Friedman J (eds) The elements of statistical learning: data mining, inference, and prediction. Springer, New York, pp 485–585
16. Aggarwal CC, Hinneburg A, Keim DA (2001) On the surprising behavior of distance metrics in high dimensional space. In: Van den Bussche J, Vianu V (eds) Database theory — ICDT 2001. Springer, Berlin, pp 420–434
17. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: Proceedings of the third international conference on knowledge discovery and data mining. AAAI Press, Seattle, pp 359–370
18. Witten DM, Tibshirani R (2010) A framework for feature selection in clustering. J Am Stat Assoc 105:713–726
19. Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53–65