Supporting Information

Reconstructing gene regulatory dynamics from high-dimensional single-cell snapshot data

Andrea Ocone, Laleh Haghverdi, Nikola S. Müller, Fabian J. Theis

Contents

1 Models and methods

We present our framework to reconstruct gene regulatory dynamics and refine GRN structure from high-dimensional single-cell snapshot data. A block diagram of the framework is shown in Figure 1.

The first step represents a dimensionality reduction, obtained through diffusion map [1]. When snapshot data embedded in a low-dimensional space presents multiple branching paths, then an *ad hoc* clustering algorithm is used to separate branches. In each single branch, cells are used to reconstruct pseudo time-series by means of Wanderlust [2], a cell time-ordering algorithm. Pseudo time-series are then used to estimate kinetic parameter in ODE-based transcriptional models and do model comparison. Different transcriptional models are based on a coarse network structure obtained through network inference algorithm GENIE3 [3]. Details of each module composing our framework are reported in the following sections.

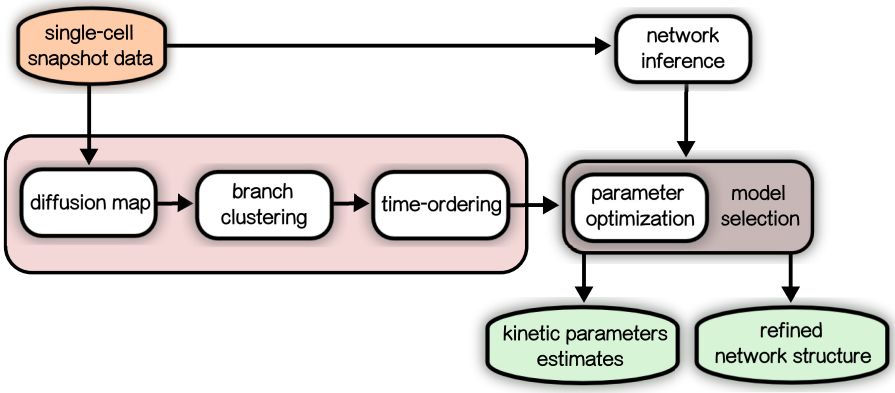

Figure 1: Diagram of our framework.

1.1 Dimensionality reduction

Single-cell gene expression snapshot data consist of gene expression values for a set G of genes in a number N of cells at a certain time point t. In other words, for each of the N cells, gene expression for all of the G genes is measured at time t. Since gene expression is essentially a stochastic process [4], the expression of a gene is different in each cell and can span on a wide range, depending on the role of the gene in the GRN. Therefore, snapshot data reflect in some way the GRN state at multiple stages during the network's temporal evolution.

So defined, single-cell gene expression snapshot data represent high-dimensional data, where the dimension is given by the number G of genes. A dimensional reduction method allows to embed this high-dimensional data into a low-dimensional space. In such a way, N cells can be visualised in a two or three dimensional space, where cell clusters can be identified according to their gene expression value. As it will be clear later on, this step is required in order to learn GRN dynamics, since it facilitates the process of cell clustering (e.g. separation of branches along differentiation paths) and subsequent cell time-ordering.

Many dimensional reduction techniques are available, among which PCA, its probabilistic version [5] and its nonlinear version [6]. Here we use a non-linear method known as diffusion map [1] which has the advantage of preserving the global geometrical structure of the data as a continuum as well as being remarkably robust to noise. In diffusion maps, the similarity between cells (in terms of expression of their genes) is indicated by their Euclidean distance in the lower dimensional space. Here, the coordinates of cells are given by the first few eigenvectors (i.e. diffusion components) of a properly built $N \times N$ Markovian transition matrix (with N as the total number of cells). Figure 2 shows an application of diffusion map to snapshot data generated from the toggle switch network described in main paper.

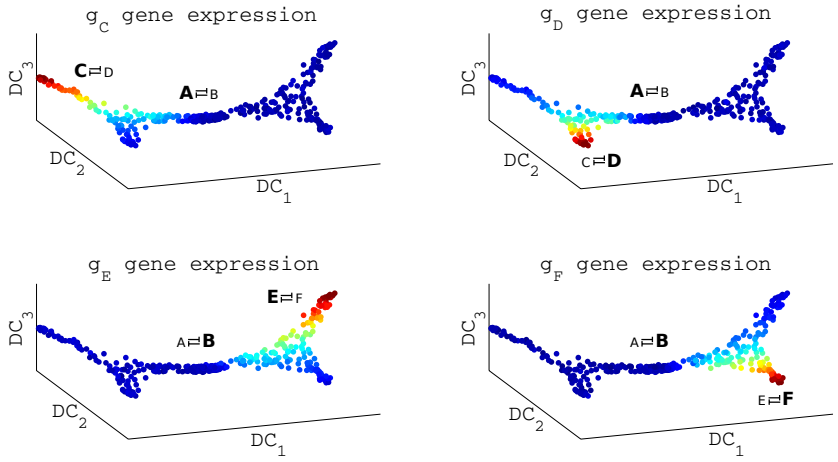

Figure 2: Embedding of single-cell snapshot data generated from toggle switch network, after application of diffusion maps. Colours encode gene expression levels for gene g_C (top left), g_D (top right), g_E (bottom left) and g_F (bottom right). Bold letters indicate genes which are activated in each of the branches, e.g. top left branch is generated by activations of genes g_A and g_C.

1.2 Branch clustering

Single-cell snapshot data can represent multiple cellular processes, e.g. both toggle switch synthetic dataset and hematopoietic snapshot data, include multiple differentiation pathways. Diffusion map provides a mean to identify and visualize developmental trajectories in single-cell gene expression data [7]. For example, Figure 2 clearly shows the presence of multiple branches in the low-dimensional space, each corresponding to a specific differentiation pathway.

In order to separate cells associated with different processes, we use an ad hoc clustering method to separate an a priori defined number B of branches in low-dimensional space.

As different branches in the diffusion map can also share same cells, k-means or similar algorithms could not work properly. We adopt a different strategy based on shortest paths. For each branch we select a starting cell C_S and a final cell C_F and we run Dijkstra algorithm [9] to find the shortest path between C_S and C_F along the diffusion map. Cells belonging to the shortest path are considered part of the branch, together with their N_n nearest neighbours. For each branch b, the algorithm is summarised as follows:

```
select starting cell C_S and final cell C_F
find shortest path using Dijkstra algorithm → [C_S, C_s1, C_s2, ..., C_sN, C_F]
for C ∈ {C_S, C_s1, C_s2, ..., C_sN, C_F}
    find N_n neighbours
branch b is defined by cells C_S, C_s1, C_s2, ..., C_sN, C_F and their N_n neighbours
```

Figure 3 shows application of the clustering algorithm to the 3D embedding obtained from diffusion map on the toggle switch simulated snapshot data.

The number N_n of neighbours represents a user-defined parameter. We tested the performance of our approach in terms of parameter estimation accuracy and inferred GRN structure on six simulated pseudo time-series datasets obtained by using six different number N_n of neighbours, in the range $N_n = [5, 30]$. Results reported in Section 2.2.4 show that no significant change occurs for our simulated toggle switch network. We could not test on the FFL network, as branch clustering was not used for that.

The number and selection of branches crucially depend on the choice of the initial cell C_S. Prior information is required to select an approximate position for C_S, which is generally not available. Visual inspection of diffusion map geometry together with prior knowledge about gene expression profile of key genes, can be sufficient to approximately locate C_S.

In the toggle switch network example, we might know that gene expression of g_C, g_D, g_E and

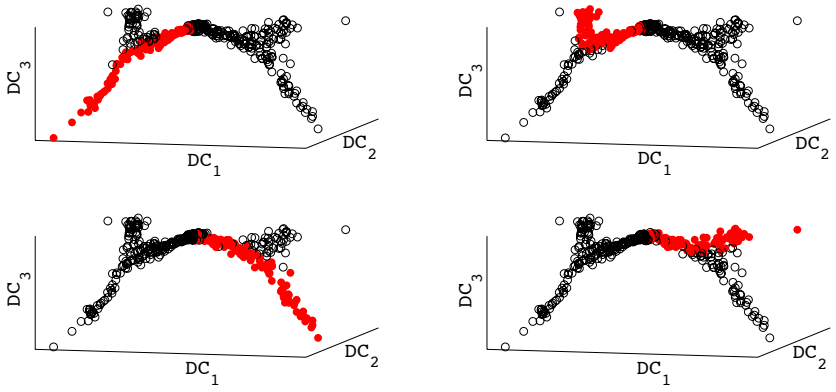

Figure 3: Application of clustering algorithm on 3D embedded snapshot data simulated from toggle switch network. Axis labels DC_i (with $i = [1, 2, 3]$) denote diffusion components, correspondent to diffusion map eigenvectors. Plots represent application of clustering algorithm to four different branches.

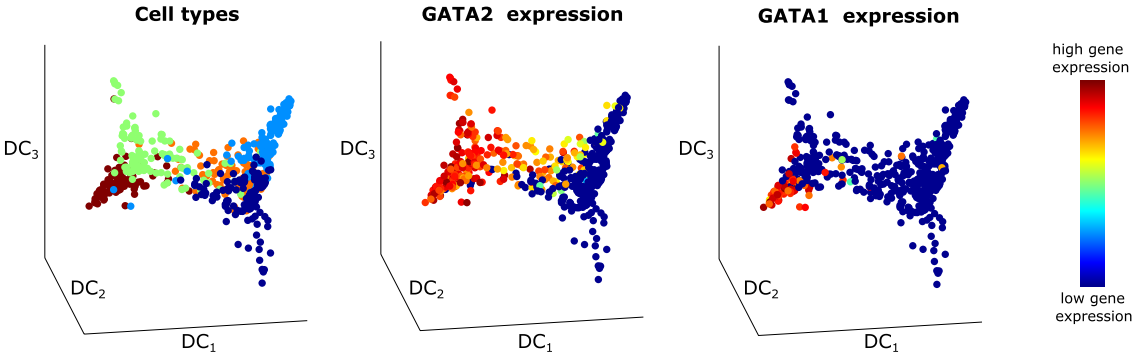

Figure 4: Diffusion map of hematopoiesis single-cell snapshot data set. Colours represent cell types (left), GATA2 expression (centre) and GATA1 expression (right).

g_F change from low gene expression values to high gene expression values in order to generate different cell fates. As a consequence, by looking at the four plots in Figure 2, the only possible initial cell position has to be approximately at the centre of the X shape geometry. This would allow to select four possible branches, with final cells at the four different endings of the X shape geometry, such that in each of the branches the expression of one single key gene (g_C, g_D, g_E and g_F) would change from low to high values.

For the hematopoietic data, assuming that we do not have information about cell types given by cell surface markers, the reasoning is similar. From literature, we have information about gene expression of key genes, e.g. GATA1 is known to be expressed at high levels in PreMegE lineage, but not in HSC [10]. In addition we know that GATA2 is known to be expressed both in HSC and in PreMegE lineage [11, 12]. By looking at gene expression patterns of GATA1 and GATA2 on diffusion map embedding (Fig. 4), we can identify HSC region (i.e. where GATA1 is not expressed and GATA2 is expressed). Initial cell C_S location will be in this case set approximately at the top-left ending of the X shape geometry, such that the developmental trajectory from HSC to the other cell lineages is maximal.

Once the initial cell location is set, final cells (C_F) to define different branches are located at the sharp endings of diffusion map geometry. Different geometries are also possible, e.g. a two-branch structure was found through diffusion map in another dataset from blood precursors cells [7].

1.2.1 Clustering with/without diffusion map

Here we test if the use of diffusion map improves results of the branch clustering algorithm. In other words we test if diffusion map embedding is only useful for determining position of initial and final cells in the clustering algorithm or also if the application of the algorithm on the diffusion map low-dimensional space provides better results.

We do the comparison on toggle switch dataset, simulated with different observation noise levels. Clusters are defined during realisation of stochastic simulation with Euler-Maruyama algorithm, before adding observation noise and before generating snapshot data. In particular, 6 regions are defined, according to expression of the genes, as showed in Figure 5. These six regions are used as prior labels to define 4 true clusters. In particular, the 4 clusters are so defined: cluster 1 includes blue and brown cells, cluster 2 includes blue and red cells, cluster 3 includes dark blue and cyan cells, cluster 4 includes dark blue and yellow cells. The number of regions is 6 (and not 4 as the number of clusters), because at early stage of stochastic simulation is not possible to distinguish the exact cell fates.

We simulate three snapshot data with observation noise $\sigma_\epsilon = 20$ and three snapshot data with higher observation noise $\sigma_\epsilon = 80$. For each snapshot data we run diffusion map algorithm and on the low-dimensional diffusion map embedding we select initial cell and final cells for the four branches. Using these initial and final cells, we run the clustering algorithm, once directly on high-dimensional data and once on the embedded low-dimensional space.

Results reported in Table 1 show a misclassification error rate, calculated as the percentage of false clustered cells on the total number of clustered cells (for each branch). In particular, results represent average over all four branches and over the three datasets. Results confirm that clustering algorithm performs better on the low-dimensional embedding, therefore diffusion map is necessary to obtain a more precise clustering of the differentiation pathways and consequently more reliable pseudo time-series through Wanderlust algorithm.

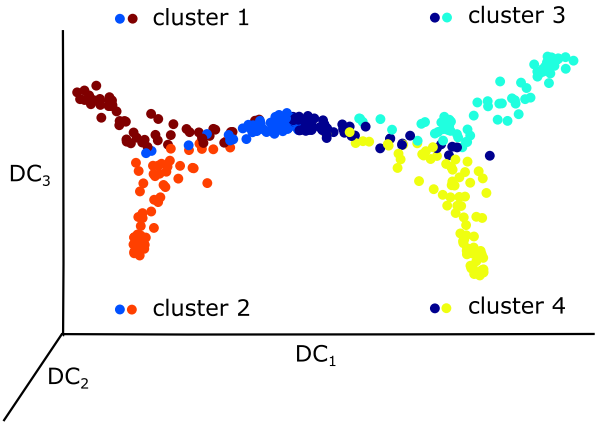

Figure 5: Diffusion map embedding of toggle switch snapshot data (with observation noise $\sigma_\epsilon = 20$). Colours indicate six regions defined during stochastic simulations, used as labels to quantitatively evaluate the performance of clustering algorithm.

	low-dimensional	high-dimensional
$\sigma_\epsilon = 20$	7.4 ± 1.8	9.9 ± 1.7
$\sigma_\epsilon = 80$	8.7 ± 2.5	14.1 ± 1.6

Table 1: Misclassification error rate obtained by running clustering algorithm on low-dimensional data (on diffusion map embedding) and on original high-dimensional data, with different observation noise levels. Values represent average over four branches, over three different datasets.

1.3 Cell time-ordering

As the input to our framework is represented by high-dimensional static data and the output by knowledge about GRN dynamics, an initial requirement for the method is the extraction of dynamic information from static one. Since gene expression is a function of time, we assume that cells can be ordered by time along paths in branches identified in the embedded data space.

Here, we order the cells by time using Wanderlust algorithm [2]. Wanderlust is able to robustly map single cells from multidimensional data onto a one-dimensional developmental trajectory. By assuming cells represent nodes in a k-nearest neighbour graph, it computes distances between cells and their k neighbours, as edge weights defined by a similarity measure. The trajectory is finally computed by following the shortest-path distances between nodes in the graph, which are defined by the minimum sum of graph edge weights. In order for the algorithm to work correctly, a starting cell from which shortest-path distances are computed has to be defined a priori. This information is generally not available, but, as described above, diffusion map embedding is useful to recover an approximate location of the starting cell. While an approximate starting cell is required as prior information, small variation in the position of the starting cell do not affect much Wanderlust's performance [2]. A detailed description is beyond the scope of this work and can be found in [2].

One drawback of Wanderlust algorithm is that it assumes a non-branching developmental trajectory. As in both toggle switch simulated dataset and hematopoietic snapshot data we are dealing with multiple differentiation branches, we overcome the non-branching assumption by combining Wanderlust algorithm with the branch clustering algorithm. In other words, we first use the clustering to separate an a priori number of branches on the diffusion map embedding. Within each branch, cells are ordered by time using Wanderlust algorithm. Note that Wanderlust is applied directly on high-dimensional single-cell data, after single cells have been clustered (in different branches) in the low-dimensional embedding. In the case of real hematopoietic data, the clustering step was not necessary, as information about different cell populations was already available through cell surface markers [8].

The clustering algorithm allows to include in the Wanderlust algorithm only cells belonging to a single (i.e. non-branching) differentiation path, therefore it prevents outliers in reconstructed pseudo time-series.

In order to measure quantitatively how pseudo time-series would differ in case Wanderlust was applied on low-dimensional data instead of high-dimensional data, we follow this procedure. We run Wanderlust on high-dimensional data, as usual, and on low-dimensional data (after diffusion map), to produce two different sets of pseudo time-series. Then we compute the correlation between the two sets of pseudo time-series. As an example we use snapshot data simulated from toggle switch network and we apply Wanderlust on two different branches, producing ten replicates for each branch. Note that when applying Wanderlust on the low-dimensional data, we consider a number of dimensions equal to the number of diffusion map significative eigenvalues[1]. Results of correlation averaged over all replicates is 92%, showing that in this case pseudo time-series obtained after diffusion map are very similar to pseudo time-series obtained by applying Wanderlust directly on high-dimensional data.

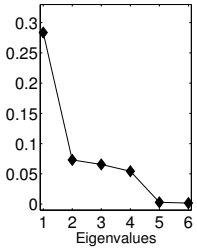

Figure 6: Diffusion map eigenvalues for toggle switch network.

[1]Eigenvalues are considered significative before a cutoff value which is detected by visual inspection (Fig. 6). In the case of toggle switch snapshot data, this cutoff occurs after the first four eigenvalues, therefore Wanderlust in low-dimension is applied on the low-dimensional space defined by the first four diffusion components.

1.4 Network inference

Once pseudo time-series have been reconstructed, they can be used to infer kinetic parameters in ODE-based transcriptional models. ODE models include network structure knowledge in form of: different numbers and combinations of inputs for a given target gene; types of regulation (i.e. activating/inhibiting input); logical functions in the transcriptional activation function (i.e. AND/OR gates or their mixtures). For this reason, the number of possible ODE models increases combinatorially with the number of genes in the network. In order to limit model selection step to a relatively low number of models to compare, we use a network inference method. This method predicts a coarse GRN representing a basis for which we build our transcriptional ODE models.

Here we use GENIE3 [3], an algorithm which decomposes the structural inference problem of a network with N genes into N regression problems. In each regression problem, the expression of target gene is represented as a nonlinear function of expressions of all the other $N-1$ genes. The single regression problem is solved using a method based on random forest. A regulatory edge for a given target gene is predicted when the weight assigned to that edge (representing input's importance for the target gene) is greater than a given threshold.

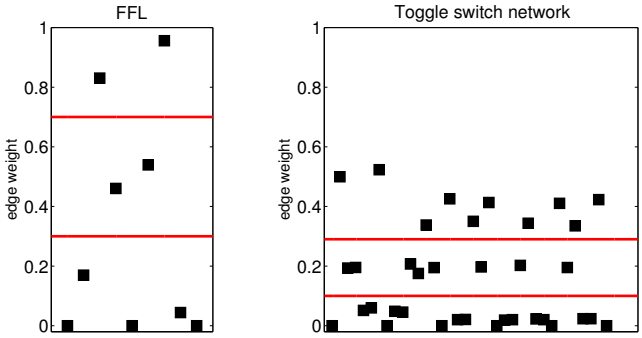

Figure 7: Edge weights obtained from GENIE3 network inference method on FFL dataset (left) and toggle switch dataset (right). Solid red lines represent possible threshold values: 0.3 and 0.7 for FFL network; 0.1 and 0.3 for toggle switch network. Order of points on x-axis is arbitrary.

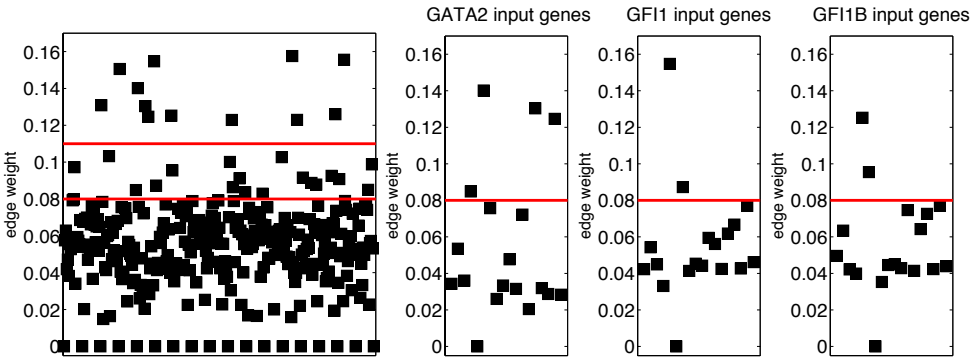

Figure 8: Left plot: edge weights obtained from GENIE3 network inference method on real hematopoietic data. Solid red lines represent possible threshold values: 0.08 and 0.11. Remaining plots: edge weights for specific target gene (GATA2, left, GFI1, centre, GFI1B, right). Order of points on x-axis is arbitrary.

Definition of the threshold for this method represents an open problem [3]. In this work we chose a threshold in an empirical way, which depends on the resulting set of edge weights and on sparsity constraints. In other words, we look at resulting edge weight values and we set the

threshold where a gap between the values is present. Figure 7 reports edge weight values obtained by running GENIE3 on the FFL and toggle switch network snapshot data. Each plot shows a number of N^2 values (where N is the number of genes in the network), representing all possible network edges. For both datasets we can identify two gaps between the edge weights, represented by solid red lines in Figure 7. Of course, a lower threshold value means that we are considering more ODE models to compare in model selection. However, in order to avoid a combinatorial explosion of the number of ODE models to compare, we keep the number of input edges to each target network node around 2-3. Figure 7 does not show how many input edges are present for each target gene, but only the total number of edges with correspondent weight. This sparsity constraint represents a requisite for our approach. In order to cope with a larger number of dynamical models (i.e. a lower threshold value in the network inference), a greedy-type strategy could be adopted in model selection, but this is not taken into account in this work.

In main paper, while for the FFL we have reported results using the lower threshold value at 0.3, for the toggle switch network we have used the higher threshold value, still at 0.3. In Section 2.2.5 we also report results for toggle switch network using the lower threshold value at 0.1. In this case, the final inferred network still represents the correct one; however, results have been obtained comparing a total of 94 ODE models, for a single subnetwork including target gene g_C.

For real data, it is harder to decide for a threshold value based on a gap: the only visible gap is given at a value 0.11 (Fig. 8, left), which results in an inferred network where the number of edges is smaller than the number of network nodes. Therefore, we select a threshold value by following only the sparsity constraint. The resulting threshold (0.08) represents the lowest possible value before having combinatorial explosion in model comparison.

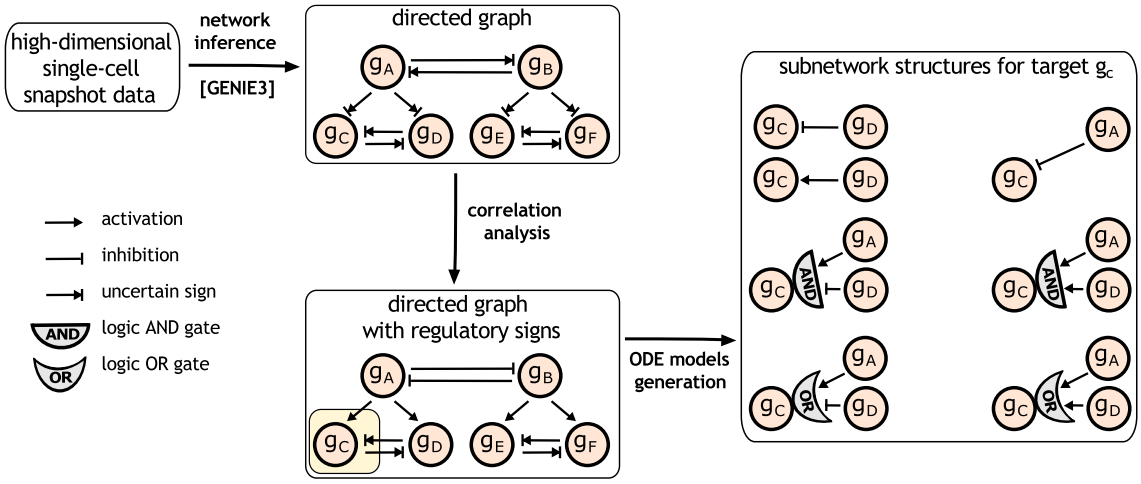

Figure 9: Application of GENIE3 on snapshot data generated from toggle switch network. Inferred network (top centre) represents a directed graph. After correlation analysis on expression snapshot data, regulatory signs are associated to the inferred network (bottom centre). Generated subnetwork structures for target gene g_C (right) are based on the inferred network.

In addition to be very fast and scalable, the method was best performer in one of the well known crowdsourcing challenges for network inference [13]. Furthermore, as the method does not take into account of temporal information in gene expression data, it suits to high-dimensional single-cell snapshot data.

Figure 9 shows application of GENIE3 to snapshot data simulated from the 6-gene toggle switch network described in main paper. In order to further reduce the number of ODE models for model selection, after network inference we perform a correlation analysis on gene expression snapshot data. As a result, a regulatory sign (e.g. activation/inhibition) is associated to each directed edge in the inferred network, when a Pearson correlation coefficient is significative.

In synthetic data, we consider a correlation coefficient significative when its absolute value is larger than a threshold 0.5. We also evaluate the framework performance by considering higher

threshold values. For example, in the synthetic toggle switch network, we tried to increase this threshold value to 1 (i.e. no correlation analysis) and considered a larger quantity of ODE models to be compared (see Section 2.2.5). Results show that even in this case the correct ODE model is selected. On the other hand, for real data we follow [8]: we do not set a threshold value for the correlation coefficient, but we compute a p-value and consider the correlation significative when its p-value is less than 0.01.

1.5 Parameter estimation

Pseudo time-series resulting from application of the time-ordering algorithm on diffusion map embeddings, are used to learn GRN dynamics. A GRN is defined by a set of nodes (i.e. genes), connected by a number of edges, representing interactions between nodes. Interactions between nodes, representing the gene expression process, involve two main mechanisms: transcription of gene into mRNA and translation of mRNA into protein. Subsequent fast post-transcriptional modifications convert the protein into an active transcription factor (TF), controlling transcription of its target gene(s).

1.5.1 Mathematical models

When describing gene expression mechanisms through mathematical models, the choice of right level of model abstraction is crucial to achieve a certain task. Our application requires a mathematical model which is flexible enough to explain nonlinear gene expression dynamics and allows for an efficient and accurate solution of the parameter estimation problem. To this aim, we use the following ODE-based model to describe gene-gene interaction:

$$\begin{aligned} \dot{y}(t) = &\ \alpha f(x(t),\theta) - \lambda y\,, \\ f(x(t),\theta) = &\begin{cases} \dfrac{x^h}{x^h+\kappa^h}, & \text{if } x \text{ is activating} \\ \dfrac{\kappa^h}{x^h+\kappa^h}, & \text{if } x \text{ is inhibiting}\,, \end{cases} \end{aligned} \tag{1}$$

where x and y represent mRNA concentrations of input and target gene, respectively. Kinetic parameter α represents production rate and λ decay constant of target gene expression; $\theta \equiv (\kappa, h)$ are parameters of a nonlinear Hill-type function $f(x(t),\theta)$ (κ and h are dissociation constant and Hill coefficient, respectively).

With Equation 1, we are assuming that mRNA concentration y of target gene can be used as a proxy for concentration level of its active TF. This is valid by considering that post-transcriptional modifications occur on a faster timescale with respect to transcription and translation process. In case this condition is not verified, alternative methods could be used where TF activity is modelled as a latent variable [14]. The combination of transcription and translation in the same process is also reasonable in absence of protein expression data; the presence of a translation model would require an inference step, which would be nontrivial due to the model nonlinearity. As a direct consequence, kinetics parameters in Equation 1 will take into account of both transcription and translation mechanisms.

Generally, in GRNs the expression of a target gene y is regulated by the activity of a number M of inputs x_i (with $i = 1,\dots,M$). These can be combined according to different logical expressions, e.g. by combining logical conjunction and disjunction operations (i.e. AND and OR gate, respectively), with identity and negation operation (i.e. activating and inhibiting inputs). Therefore, a wide range of possible models can be used to describe interactions between M inputs on the target y. The following models:

$$\dot{y}(t) = \alpha \prod_{m=1}^{M} f_m(x_m(t),\theta_m) - \lambda y\,, \tag{2}$$

$$\dot{y}(t) = \alpha \sum_{m=1}^{M} f_m(x_m(t),\theta_m) - \lambda y\,, \tag{3}$$

encode different logical expressions to combine M input genes, respectively AND and OR logic gates. Mixtures of different logical operations are also possible, when the number of inputs is greater than two. For example, with $M = 3$ inputs, the following combinations are possible:

$$\dot{y}(t) = \alpha\left[f_1(x_1(t),\theta_1) + f_2(x_2(t),\theta_2)\right]\cdot f_3(x_3(t),\theta_3) - \lambda y\,, \tag{4}$$

$$\dot{y}(t) = \alpha_1\left[f_1(x_1(t),\theta_1)\cdot f_2(x_2(t),\theta_2)\right] + \alpha_2\, f_3(x_3(t),\theta_3) - \lambda y\,. \tag{5}$$

In Equation 4, two inputs are combined through an OR gate function, which is in turn combined through an AND gate with the third input. In a similar way, two inputs in Equation 5 are combined through an AND gate function, which is in turn combined through an OR gate with the third input.

1.5.2 Optimisation

In the following sections we describe how we optimize kinetic parameters for a GRN. Here we assume that ODE transcriptional models for all target genes in the network are known. Model selection for different ODE models is treated in Section 1.6.

1.5.2.1 Metropolis MCMC

For the parameter estimation we use an approximation method based on Markov chain Monte Carlo (MCMC). Monte Carlo methods make use of numerical sampling to explore efficiently the parameter space. The idea is to obtain a set of independent samples, from a simpler proposal distribution, which we can use to reconstruct the intractable posterior density $p(\Theta)$ over the parameter space. Compared with Monte Carlo methods, such as rejection and importance sampling, MCMC methods can easily cope with high dimensional sample spaces [15]. However, since samples are drawn from a Markov chain, they are correlated; this means that long simulations are needed to obtain independent samples [16]. There are several possible MCMC methods [17]; here we use a Metropolis MCMC method [18], as summarised below:

```
initialise parameters set Θ
solve model ODEs → y(Θ)
compute initial likelihood L(y(Θ))
for t = 1 to (max iteration number)
   draw new sample Θ* ~ N(Θ^t, Σ)
   solve model ODEs → y(Θ*)
   accept Θ* with probability min(1, α)
   where α = L(y(Θ*)) [L(y(Θ^t))]^-1
   if Θ* is accepted
      Θ^{t+1} = Θ*
   else
      Θ^{t+1} = Θ^t
```

The proposal distribution is a multivariate Gaussian, with mean at the current state Θ^t and diagonal covariance matrix Σ. The Gaussian likelihood function $\mathcal{L}(y(\Theta))$ is computed using the solution $y(\Theta)$ of model ODEs at observation pseudo times:

$$\mathcal{L}(y(\Theta)) \propto \prod_{i=1}^{T} \exp\left[-\frac{\left(D_i - y(t_i,\Theta)\right)^2}{2\sigma_i^2}\right]\,, \tag{6}$$

where D_i and σ_i^2 represent observation and likelihood variance at pseudo times t_i, respectively; $y(t_i,\Theta)$ is the solution of ODE model for parameter set Θ at pseudo time t_i. It is possible to show that drawing samples from the proposal distribution and accepting them with probability $\min(1,\alpha)$, ensures that we are taking samples from a Markov transition density whose stationary distribution is $p(\Theta)$ [19].

In order to efficiently explore the parameter space, in our MCMC we use a Gaussian random walk on the log scale. At every iterations, ODEs are solved using a classical Runge-Kutta method.

As in our application all parameters must have positive values, we use a positive uniform prior over Θ. In particular we use a uniform distribution $\mathcal{U}(0, 2000)$ for synthesis rates and dissociation constants and a uniform distribution $\mathcal{U}(0, 30)$ for Hill coefficients. For decay rate λ, we also use a uniform distribution $\mathcal{U}(0, 50)$ or, where explicitly stated, an informative Gaussian prior.

1.5.2.2 Gaussian processes emulators

Computation of the likelihood function for target gene y requires the solution of ODE for $y(t)$. In order to solve ODE for $y(t)$, the values of input genes $x_i(t)$ (with $i = 1, \ldots, M$) at all time points are needed, which can in turn be obtained by solving ODEs for $x_i(t)$. The optimisation process so defined represents a recursive system, which requires many model runs at every MCMC iteration. This makes the sampling algorithm impractical for our purpose.

We bypass the problem by using emulators for input genes functions $x_i(t)$ [20]. An emulator of a function $x(t)$ is a statistical model which provides a probability distribution over the function $x(t)$ [21]. The mean of the distribution represents the approximation to the function $x(t)$, while its covariance function describes the error introduced by the approximation with respect to the true function $x(t)$. To obtain emulators for input genes $x_i(t)$, we use a nonparametric regression method based on Gaussian processes (GPs), which provide a natural way to describe probability distribution over functions.

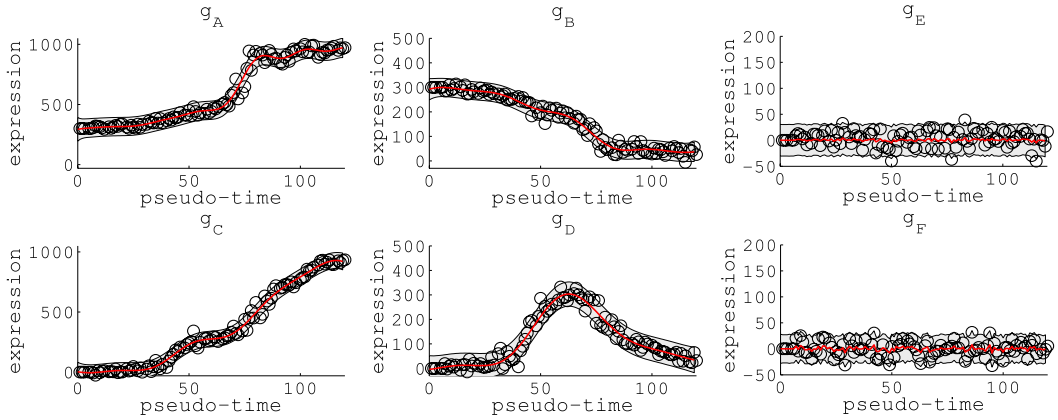

Figure 10: Emulators obtained from pseudo time-series reconstructed from a single diffusion map's branch in toggle switch simulated data set. GP means (solid red) are used as interpolators of observations at pseudo times (black circles). Shaded areas represent 95% confidence intervals.

A GP is a stochastic process, whose any finite number of samples is distributed according to a multivariate Gaussian distribution [22]. In general, it can be seen as a probability distribution over functions $f(x)$, such that any finite subset of function values, $[f(x_1), f(x_2), \ldots, f(x_N)]$, have a joint Gaussian distribution. As a Gaussian distribution is specified by its mean and covariance matrix, a GP is completely defined by its mean $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$. To denote that a function $f(\mathbf{x})$ is distributed according to a GP, we use the following notation

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right) ,$$

where

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}\left[f(\mathbf{x})\right] , \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}\left[\left(f(\mathbf{x}) - m(\mathbf{x})\right)\left(f(\mathbf{x}') - m(\mathbf{x}')\right)\right] , \end{aligned}$$

Given a set of data points $\mathbf{y} = f(\mathbf{x})$ corresponding to inputs $\mathbf{x}$, and assuming a zero-mean Gaussian noise model

$$p(\mathbf{y}|\mathbf{f}) \sim \mathcal{N}(\mathbf{f}, \sigma^2 I) , \tag{7}$$

we can compute a posterior distribution over the function values corresponding at inputs $\mathbf{x}$ using Bayes' theorem:

$$p(\mathbf{f}|\mathbf{y},\mathbf{x}) = \frac{p(\mathbf{y}|\mathbf{f},\mathbf{x})\,p(\mathbf{f}|\mathbf{x})}{p(\mathbf{y}|\mathbf{x})}\,, \tag{8}$$

where the prior distribution over the latent function $\mathbf{f}$ is given by a GP. Generalisation to new inputs $\mathbf{x}^\star$ is done by computing the predictive distribution of the latent function at the new inputs

$$p(\mathbf{f}^\star|\mathbf{x},\mathbf{y},\mathbf{x}^\star) = \int p(\mathbf{f}^\star|\mathbf{f},\mathbf{x},\mathbf{x}^\star)\,p(\mathbf{f}|\mathbf{y},\mathbf{x})\,\mathrm{d}\mathbf{f}\,, \tag{9}$$

whose mean represents our interpolation curve. By using emulators for all M inputs x_i (with $i = 1,\ldots,M$) of a target gene y, we are able to solve the ODE for a given logical model (e.g. AND or OR gate) and compute the likelihood given by Equation 6 in the Metropolis MCMC algorithm.

Figure 10 shows GP emulators obtained from pseudo time-series in a single branch of the toggle switch network. GP regressions have been performed using GPML Code [23]. Although strong smoothness assumptions are not very realistic for biological processes [24], we adopt the commonly used squared-exponential kernel as pseudo time-series data can be highly noisy.

1.5.2.3 Subsystem optimization approach

As described in main paper, optimisation of GRN kinetic parameters is done by using a subnetwork approach [20]. Considering a GRN with N genes, estimation of kinetic parameters is obtained by decomposing the network into N different subnetworks and solving N optimization problems. Each subnetwork includes one of the N target gene y and its regulators (i.e. inputs x_i, with $i = 1,\ldots,M$). Given emulators for input genes x_i of a given target y, each subnetwork is *conditionally independent* on the others. At a second stage, emulators can be replaced by fittings obtained for target genes y_i (with $i = 1,\ldots,N$) during a first stage, until kinetic parameter for the full network converge.

As the dimensionality of parameter space is reduced in each subnetwork, this strategy turns to be very efficient and scalable with GRN size. Figure 11 shows how a given GRN with $N = 6$ genes is decomposed in N different subnetworks.

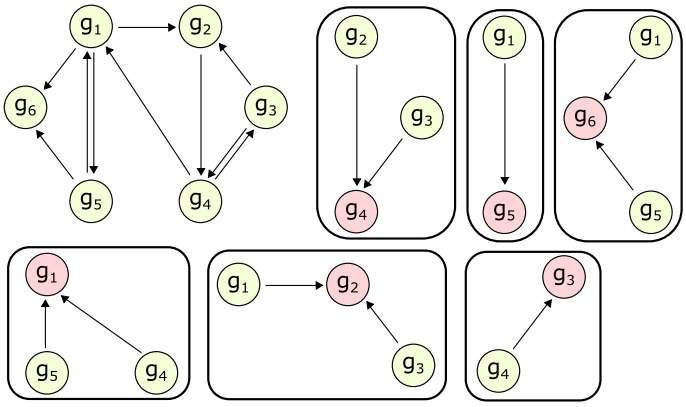

Figure 11: Decomposition of a GRN with $N = 6$ genes into N subnetworks. Each subnetwork is composed of a target gene y (pink circle) and a number of inputs x_i (green circles).

1.6 Model selection

In the last step of our framework we select which logical model explains better the pseudo time-series data. This is done for every subnetwork. In order to get a rough idea of which ODE models best fit pseudo time-series we first compute AIC and BIC statistics as follows [16]:

$$\begin{aligned} AIC &= \log p(D|\theta) - W \\ BIC &= \log p(D|\theta) - \frac{1}{2}\,W\,log(N_{obs}) \end{aligned}$$

where $log(p(D|\theta))$ is the log-likelihood for the best fit (i.e. with optimized parameters θ), W is the number of model parameters and N_{obs} the number of observations.

When results from AIC and BIC are not enough to select a model rather than another, we do Bayesian model comparison by computing the ratio between posterior distribution over different models. Considering models M_1 and M_2 (e.g. AND and OR gate), ratio between their posterior distribution is given by:

$$\frac{p(M_1|D)}{p(M_2|D)} = \frac{p(D|M_1)}{p(D|M_2)} \frac{p(M_1)}{p(M_2)} \tag{10}$$

where D represent the data. The ratio between likelihoods of data given the models is called Bayes' factor R. Generally, prior distributions over different models are the same (i.e. $p(M_1) = p(M_2)$); in this case, R represents the ratio of posterior probabilities over the models.

In order to compute R, we need to compute the marginal likelihood $p(D|M)$, given by an integral over all parameter space

$$p(D|M) = \int_\theta p(D|\theta, M)p(\theta|M)\mathrm{d}\theta \tag{11}$$

which is intractable except in simple cases. Therefore, in ordered to compute it, we need to resort to numerical methods.

1.6.1 Thermodynamic integration

In order to compute the integral in Equation 11, different numerical methods can be used. Here we adopt thermodynamic integration, which provides a better approximation compared to other methods [25].

We define the power posterior as

$$p_t(\theta) = \frac{p(D|\theta, M)^t p(\theta|M)}{Z_t} \tag{12}$$

where t is the temperature and the partition function Z_t is given by

$$Z_t = \int_\theta p(D|\theta, M)^t p(\theta|M)\mathrm{d}\theta\,. \tag{13}$$

At temperature $t = 0$, we have that $Z_0 = 1$; on the other hand, when $t = 1$, the partition function is exactly the marginal likelihood. As a consequence, the following equivalence is valid

$$\log \frac{Z_1}{Z_0} = \log Z_1 - \log Z_0 = \log p(D) \tag{14}$$

and the marginal likelihood can be computed as[2]

$$\log p(D) = \log \frac{Z_1}{Z_0} = \int_0^1 \mathrm{E}_{\theta|D,t} \log p(D|\theta)\mathrm{d}t\,. \tag{15}$$

In practice, we need to run the Metropolis MCMC for a range of different temperatures, compute the expectation of log-likelihood using samples from the posterior and finally calculate numerically the integral. We follow the algorithm below [27]:

1: `discretise` $t \in [0, 1]$: $0 = t_0 < t_1, \ldots, t_n = 1$
2: `for each` t_i $(i = 0, \ldots, n)$
3: `draw samples from power-posterior:` $\theta_1, \theta_2, \ldots, \theta_N \sim p(\theta|D, t)$
4: `compute elements` $\mathrm{E}_i = \mathrm{E}_{\theta|D,t_i} \log p(D|\theta)$ `as` $\frac{1}{N}\sum_{n=1}^{N} \log p(D|\theta_n)$
5: `estimate numerically marginal likelihood using trapezoidal rule:`

[2]Proof can be found in [26].

$$\log p(D) = \sum_{i=1}^{n} (t_i - t_{i-1}) \left(\frac{E_{i-1} + E_i}{2} \right)$$

Once marginal likelihoods are available, it is trivial to compute ratio R between them and interpret the result according to Jeffrey's scale [28] (Tab. 2).

R	$\log_{10}(R)$	**Jeffrey's interpretation**
1 to 3.2	0 to 0.5	Not worth more than a bare mention
3.2 to 10	0.5 to 1	Substantial
10 to 100	1 to 2	Strong
>100	>2	Decisive

Table 2: Jeffrey's interpretation of Bayes' factor. R represents the ratio between marginal likelihood of model M_1, $p(D|M_1)$, and marginal likelihood of model M_2, $p(D|M_2)$. Higher is R, higher is the evidence that model M_1 is better that model M_2.

2 Simulation details and additional results

2.1 Feed-forward loop network motifs

2.1.1 Single-cell snapshot data generation

Simulations for OR-gate I1-FFL have been obtained using Euler-Maruyama method on $N = 300$ cells (or realisations), using the following recursive system of equations:

$$\begin{aligned} g_Y(t+1) &= g_Y(t) + \Delta t \left[\alpha \frac{g_X(t)^{h_+}}{g_X(t)^{h_+} + \kappa_+^{h_+}} - \lambda g_Y(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\ g_Z(t+1) &= g_Z(t) + \Delta t \left[\alpha \left(\frac{g_X(t)^{h_+}}{g_X(t)^{h_+} + \kappa_+^{h_+}} + \frac{\kappa_-^{h_-}}{g_Y(t)^{h_-} + \kappa_-^{h_-}} \right) - \lambda g_Z(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \end{aligned}$$

where input $g_X(t)$ is a sigmoid-shape stochastic function. Δt is the simulation time step, $\Delta w(t) \sim \mathcal{N}(0,1)$ are i.i.d. normal random variables and σ_s represents the system noise variance. True kinetic parameters are reported in Table 3. Final time t_{stop} of simulations is set when $g_X(t) > 960$. Snapshot data were then created by selecting arbitrary times t_a and collecting expression values for all genes at those times (Fig. 12).

As FFLs consist of only three genes, dimensionality reduction is not needed. Time-ordering through Wanderlust algorithm produces pseudo time-series which can be compared with time-series (Fig. 2, main paper). As FFL model is stochastic, realisations differ from each other, therefore we compare reconstructed pseudo time-series with time-series resulting from a deterministic FFL model (by simply setting system noise to zero). Final time for the deterministic simulation is set by averaging times t_{stop} for a large number of stochastic realisations.

synthesis rate	α	100
decay rate	λ	0.25
dissociation constant (activation)	κ_+	400
Hill coefficient (activation)	h_+	20
dissociation constant (inhibition)	κ_-	200
Hill coefficient (inhibition)	h_-	10

Table 3: Kinetic parameters used to simulate OR-gate I1-FFL data.

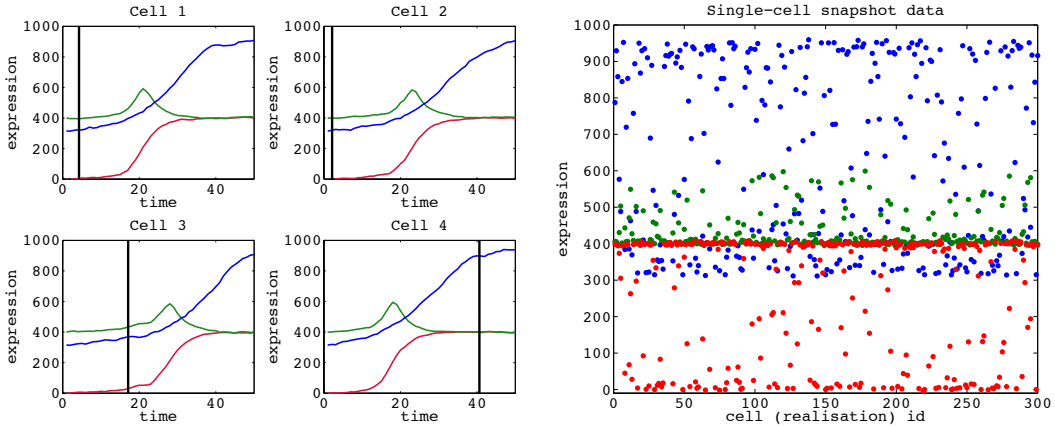

Figure 12: Simulation of snapshot data for OR-gate I1-FFL. Left: gene expression activity in four different cells (i.e. four realisations) for gene g_X (blue), g_Y (red) and g_Z (green). Vertical black lines represent arbitrary times t_a. Right: expression of all genes at times t_a in all $N = 300$ cells are collected to produce single-cell snapshot data.

2.1.2 Model selection results

By applying GENIE3 algorithm on single-cell snapshot data generated from OR-gate I1-FFL, we obtain the GRN in Figure 13 (centre), where each edge is associated with a weight w. Regulatory links with weight $w < 0.3$ are not considered. Correlation analysis between expression of genes in the network, reveals high positive correlation (correlation coefficient $r = 0.9$) between master TF g_X and slave gene g_Y. However, coefficients $|r| < 0.5$ are not informative for the other two regulatory edges.

Figure 13 (right) shows the GRN after correlation analysis. From this GRN, multiple ODE models are generated for target gene g_Z and compared through model selection (Fig. 14). Computation of AIC and BIC statistics for all ODE models are reported in Figure 14. The favoured model is the same from which data are generated, where activating input g_X and inhibiting input g_Y are combined through a logic OR gate into g_Z activation function.

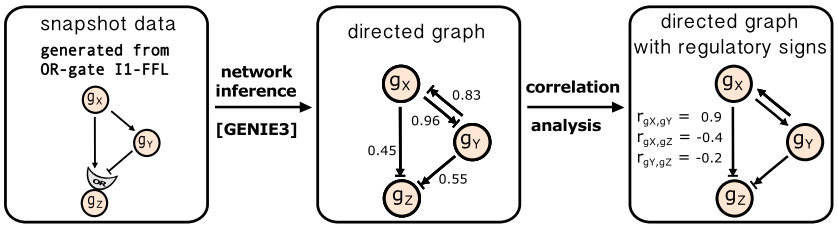

Figure 13: Application of GENIE3 network inference algorithm and correlation analysis to single-cell snapshot data generated from OR-gate I1-FFL model.

Also model with inhibiting input g_X and activating input g_Y (combined through OR gate) presents high values for AIC and BIC. Fitting of this model to pseudo time-series are also good (Fig. 15). For this reason, we compute Bayes' factor between models $g_X \rightarrow g_Z \vdash g_Y$ (OR gate) and $g_X \dashv g_Z \leftarrow g_Y$ (OR gate), to check which one best explains pseudo time-series. Log-evidence computed through thermodynamics integration is $\log p(D|M_A) = -400$ for model $g_X \dashv g_Z \leftarrow g_Y$ (OR-gate) and $\log p(D|M_B) = -385$ for model $g_X \rightarrow g_Z \vdash g_Y$ (OR-gate). As a consequence, Bayes' factor $R = \exp\left[\log p(D|M_B) - \log p(D|M_A)\right] > 100$ decisively favours model M_B.

Another competitive model is $g_X \rightarrow g_Z \vdash g_Y$ (AND gate). As described in main paper, also for this model a Bayes' factor $R > 100$ favours model $g_X \rightarrow g_Z \vdash g_Y$ (OR gate).

Note that GENIE3 predicts a false positive edge between slave TF g_Y and master TF g_X.

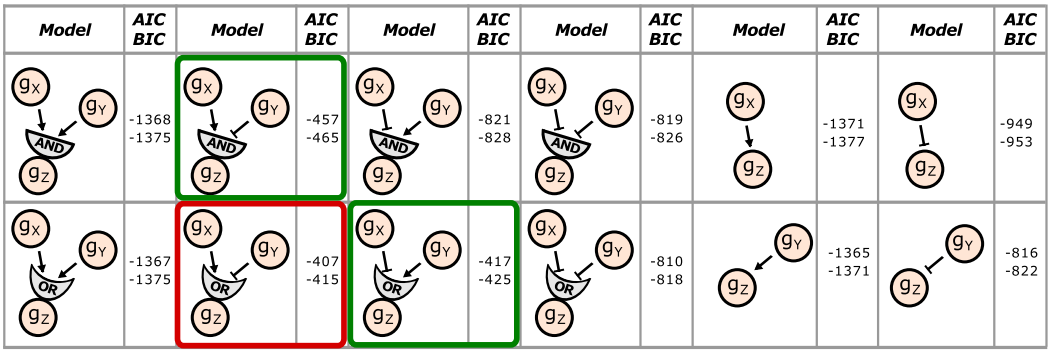

Figure 14: Transcriptional ODE models used in model selection for OR-gate I1-FFL data. Marginal likelihoods are computed for best model (red square) and competitive models (green squares).

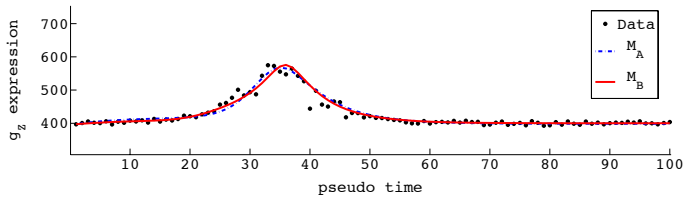

Figure 15: Fitting of models $g_X \dashv g_Z \leftarrow g_Y$ (dashed blue, M_A) and $g_X \rightarrow g_Z \vdash g_Y$ (solid, redM_B) to pseudo time-series data (black circles).

Through model selection we can compare ODE models where a single input or multiple inputs are present, but not ODE models without any input genes. For such a comparison, a latent variable model could be used to infer the presence of an unknown external input [14] followed eventually by model selection, but this lies beyond the scope of this work. For this reason, in the final network we retain regulatory link $g_Y \rightarrow g_Y$. However, this does not affect our model selection results on target gene g_Z or our model predictions.

2.1.3 Model predictions

Once we have inferred a refined GRN structure and estimates of kinetic parameters, this knowledge is used to generate predictions under perturbation conditions. We use the following procedure:

1: run a large number of stochastic simulations from true perturbed system and store final times t_{stop}
2: compute average final simulation time T_f, by averaging times t_{stop}
3: run simulations until T_f using deterministic version of true perturbed system to produce time-series data
4: produce observations $D = \{D_{g_X}, D_{g_Y}, D_{g_Z}\}$ by adding measurement noise to time series data
5: generate GP emulator GP_X for input gene g_X using noisy data D_{g_X}
6: generate prediction using inferred GRN, estimated parameters and input GP_X, under perturbation
7: compare generated prediction for target gene g_Z to observations D_{gz}

To simulate the whole GRN dynamics, also parameters for g_Y's activation function have been estimated, by considering input gene g_X as activating regulator (Fig. 13, right). Table 4 reports estimated parameters for gene g_Y and g_Z used to generate predictions.

		g_Y	g_Z
synthesis rate	α	99.1890	96.9701
decay rate	λ	0.2489	0.2490
dissociation constant (activation)	κ_+	420.0862	417.1350
Hill coefficient (activation)	h_+	14.7256	14.1570
dissociation constant (inhibition)	κ_-	264.2697	
Hill coefficient (inhibition)	h_-	8.1481	

Table 4: Mode of estimated kinetic parameters posteriors, used to generate predictions. Parameters for g_Z are obtained using an informative prior over λ, so that average of relative errors is 14.5%.

2.2 Toggle switch network

2.2.1 Single-cell snapshot data generation

Single-cell snapshot data for the toggle switch network are obtained in the same way as for the FFL case. Realisations through Euler-Maruyama method have been obtained for $N = 400$ cells from the following recursive system of equations:

$$
\begin{aligned}
g_A(t+1) &= g_A(t) + \Delta t \left[\alpha \frac{\kappa_-^{h_-}}{g_B(t)^{h_-} + \kappa_-^{h_-}} - \lambda g_A(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\
g_B(t+1) &= g_B(t) + \Delta t \left[\alpha \frac{\kappa_-^{h_-}}{g_A(t)^{h_-} + \kappa_-^{h_-}} - \lambda g_B(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\
g_C(t+1) &= g_C(t) + \Delta t \left[\alpha \left(\frac{g_A(t)^{h_+}}{g_A(t)^{h_+} + \kappa_+^{h_+}} \cdot \frac{\kappa_-^{h_-}}{g_D(t)^{h_-} + \kappa_-^{h_-}} \right) - \lambda g_C(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\
g_D(t+1) &= g_D(t) + \Delta t \left[\alpha \left(\frac{g_A(t)^{h_+}}{g_A(t)^{h_+} + \kappa_+^{h_+}} \cdot \frac{\kappa_-^{h_-}}{g_C(t)^{h_-} + \kappa_-^{h_-}} \right) - \lambda g_D(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\
g_E(t+1) &= g_E(t) + \Delta t \left[\alpha \left(\frac{g_B(t)^{h_+}}{g_B(t)^{h_+} + \kappa_+^{h_+}} \cdot \frac{\kappa_-^{h_-}}{g_F(t)^{h_-} + \kappa_-^{h_-}} \right) - \lambda g_E(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,, \\
g_F(t+1) &= g_F(t) + \Delta t \left[\alpha \left(\frac{g_B(t)^{h_+}}{g_B(t)^{h_+} + \kappa_+^{h_+}} \cdot \frac{\kappa_-^{h_-}}{g_E(t)^{h_-} + \kappa_-^{h_-}} \right) - \lambda g_F(t) \right] + \sigma_s \sqrt{\Delta t}\, \Delta w(t)\,,
\end{aligned}
$$

where Δt is the simulation time step, $\Delta w(t) \sim \mathcal{N}(0,1)$ are i.i.d. normal random variables and σ_s represents the system noise variance. Kinetic parameters used in the simulation are reported in Table 5. Figure 16 shows realisations of toggle switch network correspondent to the four different cell fates. Note that, due to stochasticity, final times change in different realisations.

synthesis rate	α	250
decay rate	λ	0.25
dissociation constant (activation)	κ_+	400
Hill coefficient (activation)	h_+	20
dissociation constant (inhibition)	κ_-	200
Hill coefficient (inhibition)	h_-	2

Table 5: Kinetic parameters used to simulate toggle switch data.

Due to bistability of toggle switches, we cannot directly compare pseudo time-series with deterministic simulations. In fact, in order to get a dynamics in gene expression, the initial state must be perturbed from its stationary level. An initial perturbation can only be achieved by system noise or, in a deterministic case, by setting an initial gene expression value which differs from its stationary one. By setting a fixed initial value, deterministic simulations will evolve according to this initial gene expression state. Therefore we cannot compare deterministic simulations (for a single specific initial value) with reconstructed pseudo time-series.

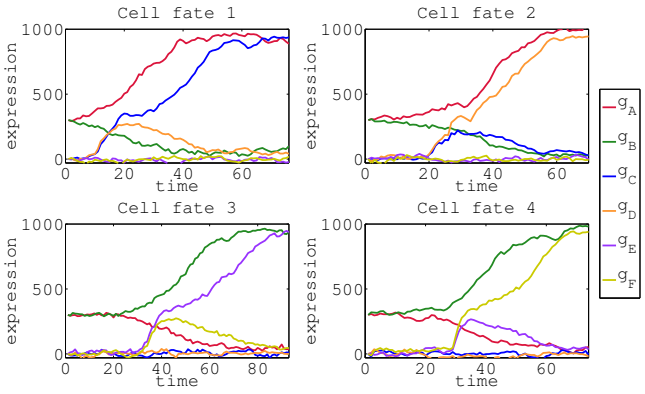

Figure 16: Different cell fates in toggle switch network.

2.2.2 Model selection results

Application of GENIE3 algorithm on single-cell snapshot data generated from toggle switch network, produces the GRN shown in Figure 17 (centre), where again we consider only regulatory links with weight $w \geq 0.3$. After correlation analysis, we obtain GRN shown in Figure 17 (right), from which we can generate transcriptional ODE models for model selection. These models, together with model selection results, are reported in Figure 3 in main paper for gene g_C and in Figure 18 for gene g_D.

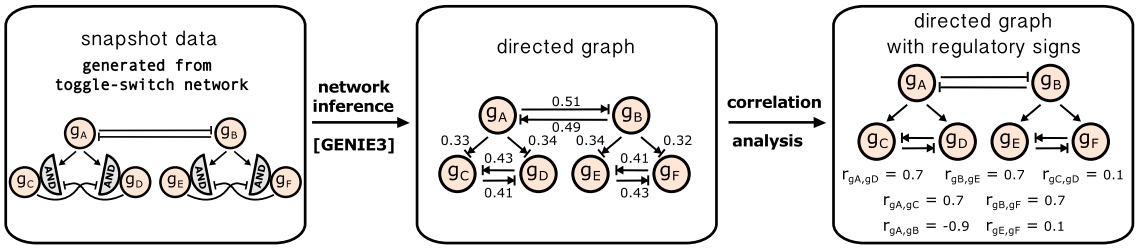

Figure 17: Application of GENIE3 network inference algorithm and correlation analysis to single-cell snapshot data generated from toggle switch model.

2.2.3 Robustness to noise

Here we test how well our framework performs in presence of different levels of measurement noise. We use the same system of stochastic differential equation to generate single-cell realisations and then add observation noise before producing snapshot data. We test robustness to two different observation noise models, additive and multiplicative, mathematically described as

$$\begin{aligned} \hat{g}(t) &= g(t) + \epsilon_+(t)\,, \\ \hat{g}(t) &= g(t) \cdot \epsilon_\times(t)\,, \end{aligned}$$

where variables $\hat{g}(t)$ and $g(t)$ represent gene expression values with and without observation noise, respectively. Observation noise terms $\epsilon_+(t)$ and $\epsilon_\times(t)$ represent Gaussian random variables with mean 0 and 1, respectively, and variance σ_ϵ^2:

$$\begin{aligned} \epsilon_+(t) &\sim \mathcal{N}\left(0, \sigma_\epsilon^2\right)\,, \\ \epsilon_\times(t) &\sim \mathcal{N}\left(1, \sigma_\epsilon^2\right)\,. \end{aligned}$$

In particular, we generate single-cell snapshot data using two different σ_ϵ values for additive noise terms (i.e. $\sigma_\epsilon = 20$ and $\sigma_\epsilon = 40$) and two different values for multiplicative noise terms (i.e.

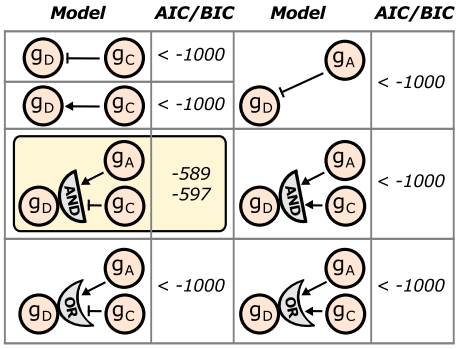

Figure 18: Model selection results for gene g_D in toggle switch network.

$\sigma_\epsilon = 0.04$ and $\sigma_\epsilon = 0.12$). Figure 19 (upper plots) shows realisations for the six genes in toggle switch network with different levels of additive or multiplicative observation noise. Snapshot data are then generated using $N = 400$ cells. Diffusion maps applied to snapshot data for different observation noise levels are showed in Figure 19 (bottom plots). Through clustering of diffusion maps branches and application of Wanderlust algorithm, we generate pseudo time-series, which we use to learn kinetic parameters.

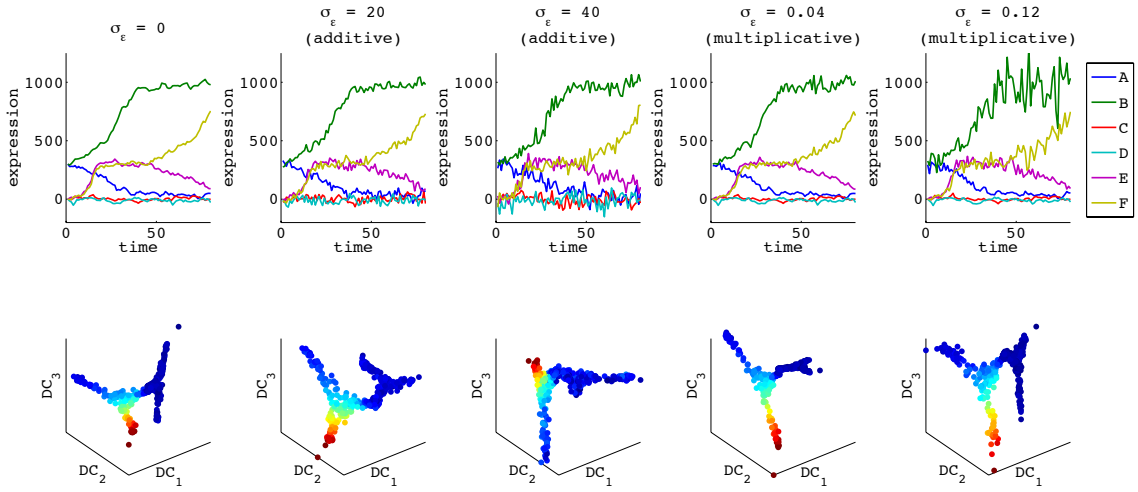

Figure 19: Upper plots: stochastic realisations of toggle switch network, using different levels of observation noise. Bottom plots: 3D embedding (after diffusion map) of snapshot data generated from toggle switch model, using different levels of observation noise. Orientations for 3D embedding is the same in all plots and colours encode for gene g_C expression levels.

In order to test the performance of our framework to recover gene expression dynamics, we assume the gene regulatory network structure is known. We learn kinetic parameters for gene g_C activation function, considering the correct logic gate model (i.e. AND-gate model) and using pseudo time-series from two branches where g_C expression exhibits a dynamical behaviour. Fittings of transcriptional models are good for pseudo time-series data from both branches and for different noise levels and noise models (Fig. 20).

To quantitatively evaluate parameter estimation results, we compute relative errors averaged over all kinetic parameters (i.e. α, λ, κ_+, h_+, κ_-, h_-) for the different data sets (Tab. 6). Error in parameter estimation increases only slightly with the increasing of observation noise, both additive and multiplicative. In some cases it can also slightly decrease with the increasing of noise level. This may be due to the fact that we do not use enough pseudo time-series replicates or also it may be due to different performances of the time-ordering algorithm.

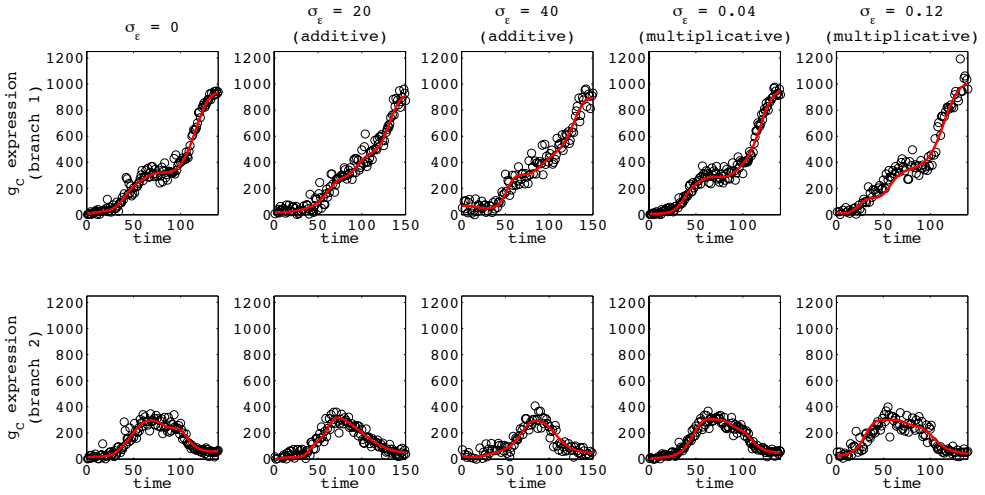

Figure 20: Fitting of toggle switch model to pseudo time-series obtained from snapshot data generated with different observation noise levels. Pseudo time-series data (black circles) compared with model fitting (solid red line), for branch 1 (top) and branch 2 (bottom).

	$\sigma_\epsilon = 0$	$\sigma_\epsilon = 20$ (+)	$\sigma_\epsilon = 40$ (+)	$\sigma_\epsilon = 0.04$ (×)	$\sigma_\epsilon = 0.12$ (×)
Average relative error (%)	6.8	7.6	9.1	5.1	8.2

Table 6: Average relative errors for parameter estimation, obtained during optimisation with an informative prior distribution over λ. Relative errors are computed as $|\hat{\theta} - \theta| \cdot \theta^{-1}$, where $\hat{\theta}$ and θ are estimated and true kinetic parameter values, respectively.

As expected, increasing observation noise to large values, we observe progressive failures at different stages of the framework. With an additive observation noise $\sigma_\epsilon = 200$, model selection is still able to select the right model through AIC/BIC statistics (considering a threshold 0.3 for the network inference, as in main paper), but parameter estimation completely fails. By increasing additive observation noise to $\sigma_\epsilon = 300$, also model selection fails. In this case, the diffusion map can still produce a definite geometrical structure but where the four branches are not recognisable anymore (Fig. 21).

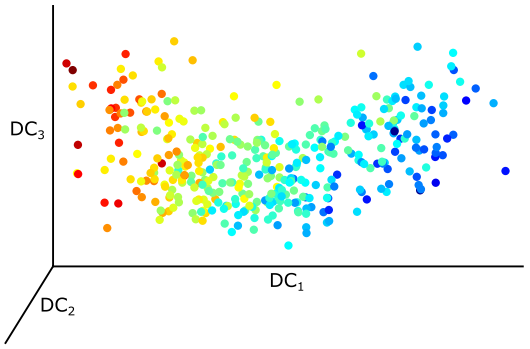

Figure 21: Diffusion map embedding of toggle switch snapshot data generated with very large additive noise ($\sigma_\epsilon = 300$).

2.2.4 Robustness to parameters of clustering algorithm

Here we test our framework performance by changing the settings of the clustering algorithm. Branch clustering in the diffusion map embedding is essential in order to separate single cells belonging to different differentiation pathways, prior to running of Wanderlust algorithm. We take into account the following two fundamental user-defined settings: the number N_n of nearest neighbours along the shortest path and the location of the starting cell C_S.

Changing N_n

In order to test the number N_n of nearest neighbours along the shortest path, we run the clustering algorithm by using six different N_n values: $N_n = [5\,, 10\,, 15\,, 20\,, 25\,, 30]$. In this case, the initial cell C_S is set at the correct approximate position. Results show that the correct ODE model is still selected in all cases. Accuracy of parameter estimation is not affected and does not show a monotonic improvement with the increasing or decreasing of N_n. Average of relative errors (considering all N_n cases) is around 10%.

To obtain these results, we have used snapshot data generated with an additive noise observation model ($\sigma_\epsilon = 20$). Parameter estimation has been performed using uniform prior distribution over all kinetic parameters, including decay rate λ.

Changing C_S position

Given an approximate correct location of C_S, we consider two different cases: one, where C_S is taken ahead along the correct branch which has to be selected, and another, where C_S is taken before the correct branch starts. In the second case, C_S is located on another branch, such that the final selected branch will include also some cells belonging to another differentiation pathway. We refer to the first case as *forward* and to the second case as *backward* (Fig. 22). For each case we repeat three times the whole process (i.e. clustering, time-ordering, parameter estimation and model selection), for three different choices of initial cell (at different distances from the approximate correct C_S location), in order to select an ODE model for the subnetwork with target gene g_C, as described in the main paper.

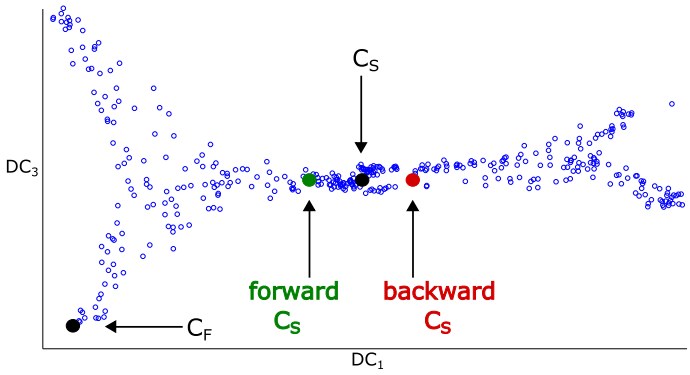

Figure 22: Single-cell snapshot data from toggle switch network after diffusion map embedding. Black cells represent the approximate correct C_S position and C_F position for the bottom-left branch. Considering this branch we select different initial cell positions: forward C_S, not taking initial branch cells, and backward C_S, including also cells from another branch.

While the framework is still able to select the correct model in both *forward* and *backward* case (i.e. activator g_A and inhibitor g_D both regulate g_C through an AND logic gate), parameter estimation performs slightly different. Average of relative errors, computed using maximum a posteriori estimates, is around 12% and 17%, respectively in *forward* and *backward* case. As expected, in both cases it is slightly larger compared to the error obtained using the correct approximate initial cell position (i.e. around 10%). Figure 23 and 24 show fitting results to pseudo time-series from *forward* and *backward* case, respectively, using the correct AND gate model. Note

how expression of genes g_A and g_B is slightly cut in the *forward* case, while it includes also spurious initial cells from another branch in the *backward* case.

In this robustness analysis, we have not considered different number of branches as varying parameter. As described above, information about the number of branches is required by the algorithm as prior information, together with an approximate position of starting cell C_S. We already described how diffusion map helps to visually locate an approximate C_S position and consequently the number of branches. We also described how final cells C_F are located, once the branches have been defined.

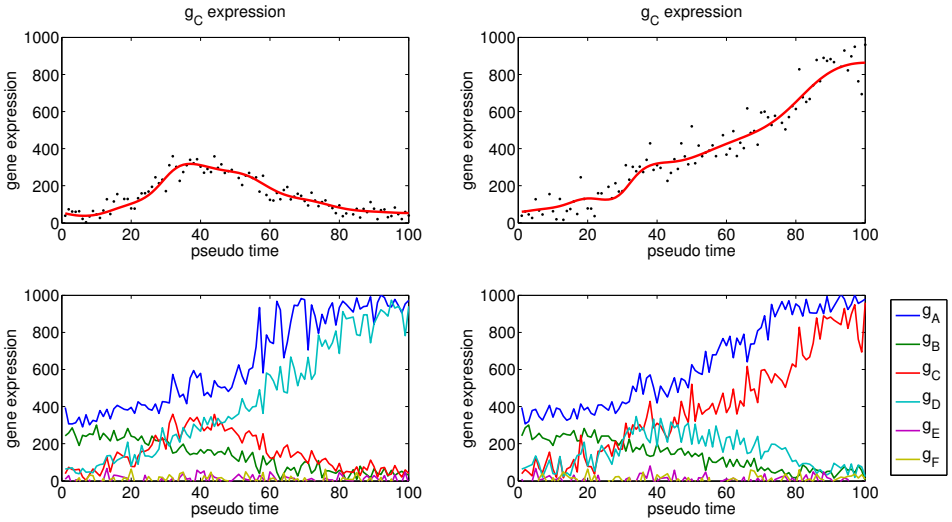

Figure 23: Top: fitting of g_C with AND gate model, for two different branches, in *forward* case. Bottom: pseudo time-series obtained after clustering with initial cell C_S in *forward* case.

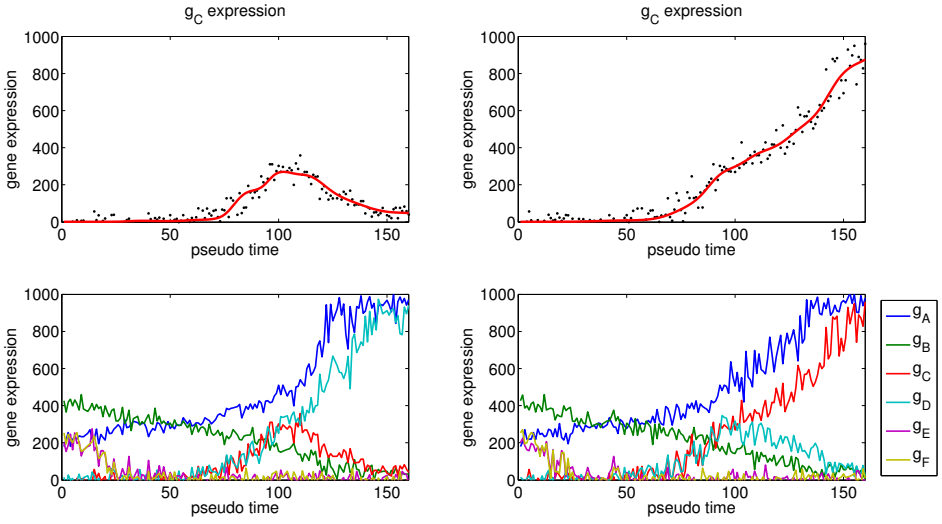

Figure 24: Top: fitting of g_C with AND gate model, for two different branches, in *backward* case. Bottom: pseudo time-series obtained after clustering with initial cell C_S in *backward* case.

2.2.5 Robustness to threshold for network inference

In previous analyses we have considered a single threshold value at 0.3 when applying network inference method GENIE3 on toggle switch dataset. By using this particular threshold value we infer the initial rough network structure in Figure 17 (centre). After correlation analysis and selection of significative correlations with threshold 0.5 we finally obtain the network structure in Figure 17 (right).

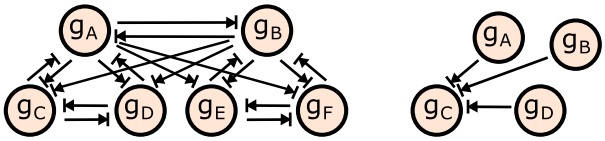

Figure 25: Toggle switch network inferred using GENIE3 with weight threshold 0.1 (left). Subnetwork for target gene g_C, obtained after network inference with GENIE3 (right).

ODE model	|BIC| / |AIC|	ODE model	|BIC| / |AIC|	ODE model	|BIC| / |AIC|
and(A+,D-)	590.0 / 581.8	and(or(A-,D-),B+)	3764 / 3753	or(and(A+,B-),D+)	6297 / 6284
and(or(A+,B-),D-)	592.1 / 581.3	and(or(A-,B+),D-)	3765 / 3754	or(and(A-,D-),B-)	6298 / 6285
and(or(B+,D-),A+)	594.7 / 583.8	or(and(A-,B+),D-)	3766 / 3754	and(A+,D+)	6401 / 6396
and(A+,B+D-)	594.8 / 583.9	or(and(B+,D-),A-)	3767 / 3754	and(A+,B+,D+)	6411 / 6400
and(or(B-,D-),A+)	594.9 / 584.0	and(or(A+,B-),D+)	3843 / 3832	or(B+,D+)	6426 / 6418
and(A+,B-,D-)	596.5 / 585.7	and(A-,B+,D-)	3949 / 3938	and(or(B+,D+),A-)	6457 / 6447
or(and(A+,D-),B+)	596.9 / 584.7	or(A-,B+,D-)	4227 / 4217	and(A-,D+)	6481 / 6473
or(and(A+,D-),B-)	597.2 / 584.9	and(or(B-,D+),A+)	4478 / 4467	or(and(A-,D+),B+)	6487 / 6475
and(or(A-,D-),B-)	599.9 / 589.0	and(A-,B+,D+)	4478 / 4467	D+	6492 / 6487
and(or(A+,D-),B-)	599.9 / 589.0	or(A+,B-)	4549 / 4541	or(A-,D+)	6497 / 6489
or(and(B+,D-),A-)	602.2 / 589.9	or(B-,D+)	4706 / 4698	and(B+,D+)	6497 / 6489
and(A-,B-,D-)	603.5 / 592.6	or(A+,B-,D+)	4710 / 4699	and(or(A-,D+),B+)	6501 / 6491
and(B-,D-)	614.8 / 606.6	or(A-,B-,D+)	4710 / 4700	or(and(A+,B+),D+)	6504 / 6492
and(or(A-,B-),D-)	619.6 / 608.7	and(or(B-,D+),A-)	4713 / 4702	or(and(A-,B+),D+)	6504 / 6492
or(and(B-,D-),A+)	622.0 / 609.7	or(B+,D-)	5185 / 5177	or(and(B+,D+),A-)	6504 / 6492
and(or(A+,B+),D-)	2025 / 2014	and(or(B+,D-),A-)	5208 / 5197	and(or(B+,D+),A+)	6555 / 6544
or(and(A+,B-),D-)	2556 / 2543	and(or(A-,D+),B-)	5531 / 5520	A+	6572 / 6567
or(and(A-,B-),D-)	2565 / 2553	and(A-,B-,D+)	5828 / 5817	and(A+,B+)	6577 / 6569
or(and(A-,D+),B-)	2865 / 2853	or(and(B-,D+),A-)	5881 / 5869	or(A-,B+,D+)	6580 / 6569
and(or(A-,B-),D+)	3007 / 2996	and(B-,D+)	5932 / 5924	or(and(A+,D+),B+)	6584 / 6572
or(B-,D-)	3189 / 3181	and(A+,B-,D+)	5937 / 5926	or(and(B+,D+),A+)	6584 / 6572
or(A-,B-,D-)	3194 / 3183	and(or(A+,D+),B-)	6006 / 5995	or(and(B-,D+),A+)	6584 / 6572
and(or(B-,D-),A-)	3194 / 3183	and(or(A-,B+),D+)	6131 / 6120	or(A+,B+)	6631 / 6623
or(A+,B-,D-)	3566 / 3555	and(A-,B-)	6131 / 6123	D-	6666 / 6661
or(and(A+,B+),D-)	3616 / 3604	or(and(A-,B-),D+)	6139 / 6126	B-	6666 / 6661
or(A+,D-)	3648 / 3639	or(A+,D+)	6188 / 6180	B+	6666 / 6661
or(and(B+,D-),A+)	3649 / 3637	and(or(A+,D+),B+)	6193 / 6182	and(A-,B+)	6671 / 6663
or(A+,B+,D-)	3652 / 3641	or(A+,B+,D+)	6193 / 6182	or(A-,B+)	6671 / 6663
and(or(A+,D-),B+)	3664 / 3653	and(or(A+,B+),D+)	6194 / 6183	or(and(A-,D-),B+)	6678 / 6666
or(A-,D-)	3759 / 3751	or(A-,B-)	6288 / 6280	A-	6722 / 6717
and(B+,D-)	3759 / 3751	and(A+,B-)	6289 / 6280		
and(A-,D-)	3759 / 3751	or(and(A+,D+),B-)	6296 / 6284		

Table 7: Model selection results in toggle switch network. First, third and fifth columns represent ODE models, with corresponding absolute values of BIC/AIC statistics in second, fourth and sixth columns, respectively.

Here we show that by decreasing GENIE3 threshold value and considering all possible regulation signs (i.e. no correlation analysis), our framework is still able to select the correct logical model. By selecting a threshold value 0.1 in GENIE3, we obtain network structure in Figure 25 (left). As the structure is symmetrical, we consider only the subnetwork including g_C target gene, whose input genes are g_A, g_B and g_D. We consider all possible 94 ODE models and do model selection. Results are reported in Table 7, where ODE models are given with BIC values in descending order. The model which can best explain pseudo time-series is $and(A+, D-)$, where g_A (as activator) and g_D (as inhibitor) combine through an AND gate to regulate g_C expression. AIC statistics show that another ODE model, $and(or(A+, B-), D-)$, is also able to explain well the data. By computing log marginal likelihood for the two models with thermodynamic integration, we obtain $\log(D|M_1) = -566.195$ for model $and(A+, D-)$ and $\log(D|M_2) = -567.455$ for model $and(or(A+, B-), D-)$.

Computation of Bayes' factor, $R = \exp(\log(D|M_1) - \log(D|M_2)) = 3.53$, reveals that model $and(A+, D-)$ can explain data substantially better than the competitive model.

Furthermore, analysis of parameters (not shown) of other models with large AIC/BIC values, also reveals that kinetic parameters for g_B in these models are such that gene g_B has not influence on g_C regulation. In other words, by removing gene g_B, the fit to g_C pseudo time-series does not change, and the models collapse to model $and(A+, D-)$.

As the network is symmetric (Fig, 25, left), similar results can be obtained for gene g_D, g_E and g_F. On the other hand, inference for subnetwork of target g_A (not reported) has trivial results. In Figure 25 (left), g_A is regulated by three inputs: g_B, g_C and g_D. As g_A shows a proper dynamics in all of the four diffusion map branches (Fig. 16), we can use pseudo time-series from all branches for model selection. However, as in two out of four branches, g_C and g_D have no dynamics, it is easy to predict that the only possible input gene for g_A will result g_B.

As described above, our approach still requires a certain sparsity of the GRN in order to do model selection. For this reason, in the case of toggle switch network we do not attempt to decrease the threshold value below 0.1, as this would generate a combinatorial explosion in the number of ODE models to compare.

2.2.6 Robustness to parameters of Wanderlust algorithm

We do not test here robustness of Wanderlust algorithm to user-defined parameters as extensive tests have been previously performed [2]. Wanderlust has been proved to be a robust algorithm, able to provide a consistent output over multiple runs on the same dataset.

As mentioned above, Wanderlust requires an absolute starting point for the initial cell, which is obtained through prior knowledge. However, the performance of the algorithm has been evaluated with a wide range of starting cell choices and it has been shown that only an approximate starting cell is needed.

More importantly, in order to reduce uncertainty due to the choice of the initial starting cell, in our analysis we always run Wanderlust three times to produce three replicates where initial cell position slightly change. All replicates are simultaneously used in parameter estimation and model selection steps.

Other user-defined parameters are the number of nearest neighbours k and the neighbour subset size I. Each cell (i.e. graph node) in the network is connected to a number of initial neighbours k. However, for each node, only a number I of random neighbours is retained. Additional tests showed that Wanderlust output was consistent over multiple choices of these two parameters. In particular, the algorithm provides accurate trajectories as long as k is greater than I. More details can be found in [2] (supplementary information).

In our framework we have always used combinations of k and I values, such as $I = k/3$.

2.2.7 Robustness to different sampling strategies

So far we have generated synthetic single-cell snapshot data using a uniform sampling distribution. In other words, arbitrary times t_a (Fig. 12) were selected along the time axis with a uniform distribution.

Here we test the performance of our framework by generating snapshot data using different sampling strategy. In particular, we sample single cells by using three different sampling functions, obtained from mixtures of Gaussian distributions (Fig. 26, top). The first function generates high cell density from three different areas along the time axis and low cell density from other two areas; the second function generates high cell density only in two areas along the time axis and low cell density in the remaining areas. Finally, the last sampling function generates high cell density in three different areas and no samples along two areas of time axis.

Figure 26 (bottom) shows diffusion map embeddings for datasets generated with the three different sampling functions. Note that diffusion maps are not affected by different sampling strategies. This is due to a density normalisation in diffusion map algorithm, such that density heterogeneities in data sampling do not affect how cells are close in the diffusion metric [29].

We create three datasets for each of the three different sampling functions, using an additive observation noise model ($\sigma_\epsilon = 20$), and use our framework to do parameter estimation in the correct ODE model, $and(A+, D+)$, for gene g_C subnetwork. Results of parameter estimation, measured

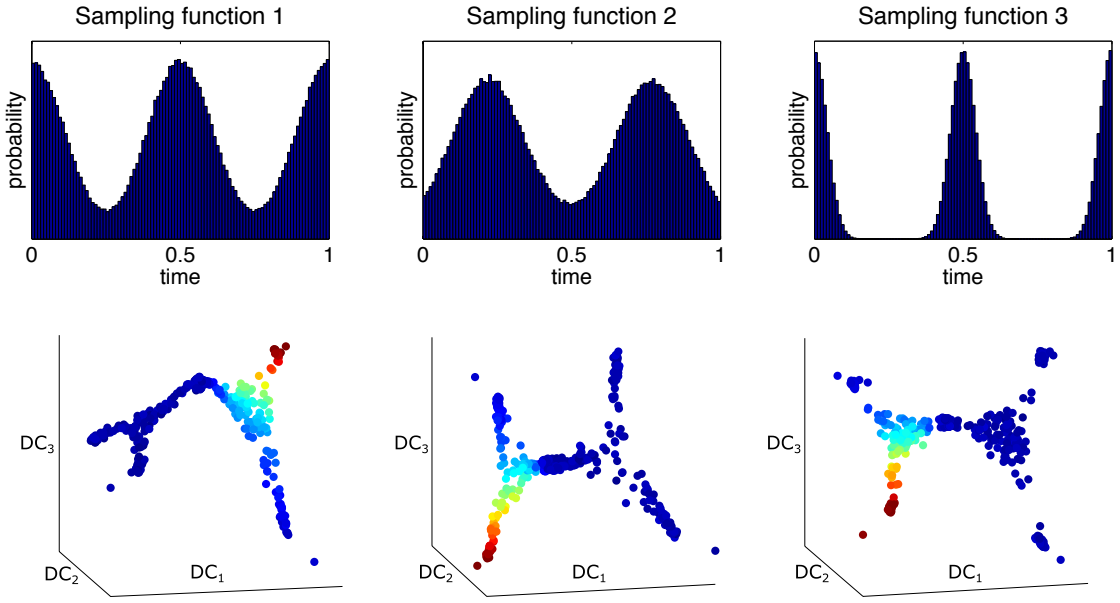

Figure 26: Top: sampling functions. X-axis values represent arbitrary measures to be scaled to every single stochastic realisation when generating snapshot data. E.g. if final time of a given realisation is T_f, then the sampling function will span in range $[0, T_f]$. Bottom: diffusion map embedding of toggle switch single-cell snapshot data generated with respective sampling functions.

as average of relative errors (among all datasets) is $(9.8 \pm 4.2)\%$, for both first and second sampling strategy. This is the same obtained with a dataset generated with uniform sampling.

However, for the third sampling strategy, parameter estimation fails and the average of relative errors increase to $(92.5 \pm 9.2)\%$. The reason may be due to cell time-ordering, through Wanderlust, which is not able to take into account absolute distances along diffusion map branches, but only relative distances between cells. For example, by looking at the embedding in Figure 26 (bottom, right), we note that groups of cells at the end of top-right and top-left branches maintain a certain distance to the main diffusion map part. Wanderlust is not able to detect this feature and therefore time-ordering results biased. This problem can be of course reduced by correcting Wanderlust algorithm, but we do not do it in this work.

We can conclude that our framework is robust to different sampling strategies (i.e. with different non-uniform densities), but, at the present state, it still requires that samples are taken from all parts of the state-space.

Finally we evaluate the framework performance to a reduced number of cells. We use snapshot data simulated from toggle switch network, generated with the second sampling strategy (Fig. 26, top-centre), by using 200 cells (i.e. realisations) instead of the usual 400 cells as in the rest of this work. Results on parameter estimation show that average of relative errors (among three datasets) is $(32.3 \pm 8.6)\%$, that is around three times worse compared to inference on snapshot dataset with 400 cells. This represents a limitation of the model. However, we have to consider that in toggle switch we also use the clustering algorithm to separate four branches; this means that the effective number of cells for each branch is usually not more than $200/4$, which is a relative small number.

2.3 Blood stem cells data

2.3.1 Branch clustering

As described in main paper we do not need the branch clustering algorithm for hematopoietic real data. As cell surface markers have been used to distinguish cell types [8], branches are simply separated by selecting only cells of interests belonging to a particular differentiation pathway.

Here we try to use the clustering algorithm to separate branches without information about cell types. As described above, using diffusion map embedding and prior information about dynamics of some key genes, we are able to locate the approximate position of starting cell C_S (top-left corner of left diffusion map in Figure 4).

Using this starting cell C_S we run the clustering algorithm to separate three branches corresponding to the three differentiation pathways HSC → PreMegE, HSC → LMPP → GMP and HSC → LMPP → CLP. Final cell positions for the three branches is set at the three remaining sharp corners of diffusion map. By using now information about cell types, for each clustered branch we calculate how many cells belongs to the experimentally measured pathway. For example, once cells along pathway HSC → LMPP → GMP have been separated in one of the branch, we calculate how many of these cells have been associated with surface markers for HSC, LMPP or GMP. Results (obtained by using $N_n = 15$ nearest neighbours) show that our algorithm is able to correctly cluster cells with error 1.3%, 7.4% and 10.3%, respectively for differentiation pathways PreMegE, GMP and CLP.

2.3.2 Data pre-processing

After cell time-ordering through Wanderlust algorithm, we obtain pseudo time-series as showed in Figure 27. Values on y-axis represent CT values (i.e. number of qPCR cycles needed to detect a signal), which are inversely proportional to the quantity of transcribed mRNAs for a given gene. For example, a value CT = 5 for a given gene at a given pseudo time (i.e. in a given cell) means that 5 qPCR cycles were necessary in order to obtain a sufficient amount of mRNAs to produce a detectable signal. In particular, since the number of mRNA transcripts increases exponentially with number of cycles, the quantity of gene expression in proportional to 2 to the power of $C_{max}-$ CT, where C_{max} is the maximum number of possible cycles (e.g. $C_{max} = 15$ in Figure 27).

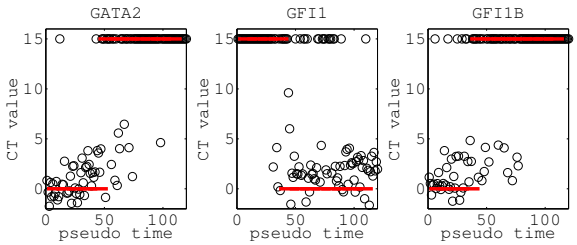

Figure 27: Pseudo time-series for genes GATA2, GFI1 and GFI1B obtained from differentiation branch HSC → LMPP → GMP. Black circles represent cells; thick red lines show outputs obtained through a moving window of length $L = 18$.

As mentioned in main paper, we focus on the triad composed of GATA2, GFI1 and GFI1B. After time-ordering with Wanderlust, we obtain pseudo time-series as showed in Figure 27. Here, it is possible to observe not only noise in expression profiles but especially a switching behaviour between cells with very small expression values (CT=15) and high expression values (small CT value). This switching behaviour does not appear at all pseudo times but it is dominant at border regions where the average gene expression behaviour changes from high to low values (or vice versa). The switching may be due to two main reasons: 1) cell ordering errors due to Wanderlust algorithm and 2) errors due to the cell labelling process (through cell surface markers). In particular, some of the cells belong to plasticity regions, where cells are not fully committed: in this case, they are assigned from the biologists to one or another cell lineage, with a certain error. Therefore, when we order cells through Wanderlust along a certain differentiation pathway, we also erroneously consider misclassified cells (i.e. belonging to other pathways). Hill-function based ODE models we use in this work do not allow to fit these regions with highly dense switching. Therefore we do some data pre-processing in order to remove outliers present in such regions.

To remove these outliers, we simply use a sample window, which moves along the pseudo time axis to count for cells with small/large CT value. The output, showed as thick red lines in Figure 27, represents areas where most cells have small or large CT value, respectively with lines at ordinates

CT= 0 and CT= 15. Based on this result, we define pseudo times at the intersection between the two discontinuous red lines parts to identify transition times from high to low (or low to high) expression. In this way, cells with low CT value at pseudo times after transition from high to low gene expression will be filtered out, and vice versa. We have run this pre-processing step by using different sample window sizes in a range between 22 and 28 cells (such that the pseudo time axis results divided in only two areas, without further fragmentation). Different sample window sizes can enlarge or reduce the intersection area of 2-3 cells, therefore final pseudo time-series change only slightly without affecting model selection results.

After remotion of outliers, we replace values at CT=15 with samples from a Gaussian density. The rationale is that $C_{max} = 15$ represents a technical threshold, therefore genes in some cells may need a larger number of cycles to have a detectable fluorescent signal in qPCR. The variance of the Gaussian density is chosen to be similar to data variance in areas with small CT values.

Finally, CT values data are transformed in gene expression data. We work with logarithm of expression, obtained through the following transformation: $\log(\text{expression}) = 20 - \text{CT}$. A $C_{max} = 20$ (instead of $C_{max} = 15$) prevents values below zero (Fig. 28).

Differently from FFL and toggle switch networks, we do not use GP emulators as they do not provide a good interpolation for this data set. Instead we use interpolators based on simple piecewise linear regression.

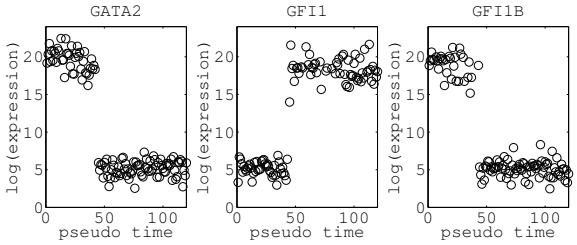

Figure 28: Pseudo time-series for genes GATA2, GFI1 and GFI1B obtained from differentiation branch HSC → LMPP → GMP, after remotion of outliers and transformation from CT values to log(expression).

2.3.3 Parameter estimation and model selection results

By applying GENIE3 network inference method to the hematopoietic single-cell snapshot data, we obtain the GRN in Figure 29 (centre), where only regulatory edges with weight $w > 0.08$ have been retained[3]. Further application of correlation analysis gives GRN in Figure 29 (right), from which we then generate transcriptional ODE models for model comparison.

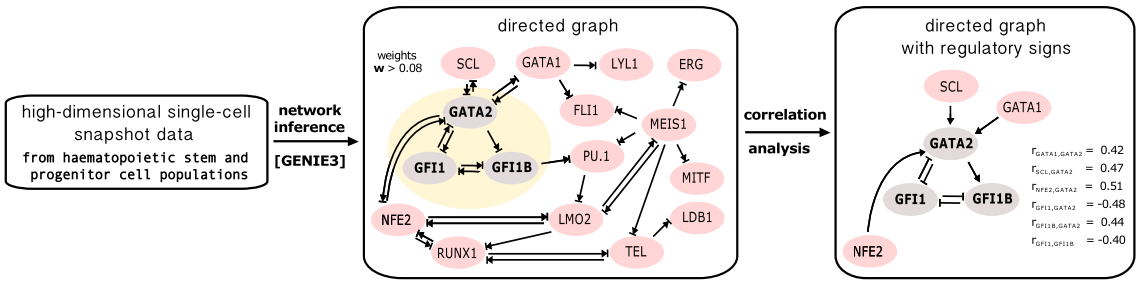

Figure 29: Application of GENIE3 network inference algorithm and correlation analysis to hematopoietic single-cell snapshot data.

[3]As explained above, the threshold weight has been chosen to keep the GRN sparse, such that model selection is still possible without a combinatorial explosion problem.

Tables 8 and 9 show ODE models compared for each of the target gene (i.e. GATA2, GFI1 and GFI1B). ODE models have been interpreted by looking at median of estimated posterior parameter distributions. In particular, most of Hill coefficient and dissociation constant collapse to zero, so that one or more genes in the activation function represent just constant values in time (i.e. they cannot be considered regulating inputs). In other words, by replacing Hill function for those genes with a constant (e.g. 0.5 if both Hill coefficient and dissociation constant are zero), fittings to pseudo time-series data does not change. Last columns of Tables 8 and 9 report effective ODE models, where only non-collapsing regulating genes are considered.

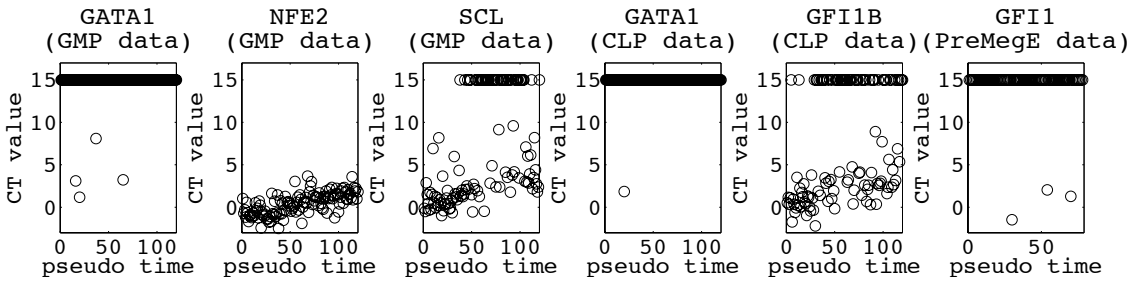

Figure 30: Pseudo time-series of TFs in our target network, obtained by using Wanderlust algorithm along different differentiation pathways.

For target GATA2 we compare seventeen models (Tab. 8), using data from differentiation pathway HSC → LMPP → CLP . As explained in main paper, GATA1 is not considered in the analysis as its expression is flat at low values (Fig. 30). A graphical visualisation of model selection results in given in Figure 31, which shows the four effective interactions with a score based on AIC/BIC statistics. Among these interactions, three interactions (i.e. direct activation by SCL, direct inhibition by GFI1, combinatorial interaction (through AND gate) between GFI1 and NFE2) have a comparable score and all of them can explain well pseudo time-series data for GATA2. The remaining model has not been reported in Figure 5 (main paper), because it has lower AIC/BIC values with respect to the other three interactions.

Model selection results for target GFI1 (Tab. 8, top) show that GFI1 is likely to be regulated by GFI1B but not to GATA2. Three ODE models with comparable AIC/BIC values collapse to the simple model where GFI1 is regulated by GFI1B . On the other hand, model with GATA2 as a positive regulator results with a lower AIC/BIC score.

Finally, results for target GFI1B show that inputs GFI1 and GATA , can separately influence GFI1B expression. All of combinatorial ODE models collapse to a model with single input, therefore no logical AND/OR gate are predicted.

Optimization was performed in logarithmic expression scale as well as diffusion map algorithm. For this reason, kinetic parameters parameters for hematopoietic data cannot be interpreted in the

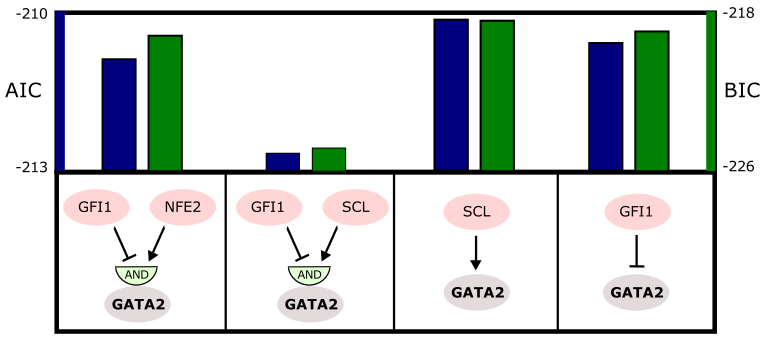

Figure 31: Model selection results for target GATA2 . As many ODE models collapse to the same model, we take those with highest AIC/BIC values.

same way as for the synthetic data examples.

Using pseudo time-series from multiple branches simultaneously, as we have done for the toggle switch network, is nontrivial for real data. In fact, while in simulated data we know exactly that length of different branches is the same in terms of time, for real data we do not know a priori if one of the differentiation pathway is longer (i.e. it lasts more time) than another pathway. A solution to the problem could be given by looking at densities of cells along branches, which in simulated data are directly proportional to time intervals. However, this hypothesis should be tested experimentally, therefore in this work we do not attempt to combine information from multiple branches during optimization.

ODE model	AIC / BIC	α	λ	κ	h	κ	h	κ	h	Effective
GFI1	-214 / -219	12	0.61	13	3.6					GFI1
NFE2	-302 / -308	1.5	0.03	21	29.4					NFE2
SCL	-223 / -229	424.8	1.77	172	1.1					SCL
and(GFI1,NFE2)	-211 / -219	751.3	0.70	8.0	1.7	144	1.9			and(GFI1,NFE2)
or(and(GFI1,NFE2),SCL)	-215 / -227	0/425	1.77	0.0	0.0	0.0	0.0	352	1.1	SCL
and(GFI1,SCL)	-228 / -236	460.8	2.33	0.0	0.0	74.1	1.1			SCL
or(and(GFI1,SCL),NFE2)	-213 / -225	72.3/0	2.61	559	19.1	44.2	1.2	0.0	3.4	and(GFI1,SCL)
and(GFI1,NFE2,SCL)	-217 / -228	1864.5	2.29	0.0	0.0	0.0	0.0	158	1.1	SCL
and(NFE2,SCL)	-210 / -218	1525.4	2.65	0.0	0.4	486	1.0			SCL
or(and(NFE2,SCL),GFI1)	-213 / -225	106/0	2.74	0.0	8.4	72.1	1.2	0.0	0.0	SCL
or(GFI1,NFE2)	-211 / -219	26.1	2.78	13.0	23.8	0.0	0.0			GFI1
and(or(GFI1,NFE2),SCL)	-212 / -223	195.5	1.71	0.0	0.1	0.0	0.0	65.4	1.1	SCL
or(GFI1,SCL)	-211 / -219	368.4	1.92	0.0	1.2	148	1.1			SCL
and(or(GFI1,SCL),NFE2)	-212 / -223	515.7	1.85	0.0	21.6	203	1.1	0.0	0.0	SCL
or(GFI1,NFE2,SCL)	-213 / -224	72	2.03	0.0	29.8	641	0.7	21.5	2.2	SCL
or(NFE2,SCL)	-210 / -218	26.6	2.78	0.0	0.0	13.1	15.6			SCL
and(or(NFE2,SCL),GFI1)	-212 / -223	151.9	1.84	27	26.7	48.0	1.2	0.0	0.0	SCL

Table 8: Model selection and parameter estimation for GATA. Values represent mode of posterior distributions for estimated parameters. Parameters from left to right are: synthesis (or production) rate (α), decay constant (λ), dissociation constant (κ) and Hill coefficient (h). When multiple dissociation constants and Hill coefficients are present, they refer to different TFs in the order they are given in the model (first column). When multiple synthesis rates are presents, the first refers to the AND part of the model, whereas the second to the third gene, according to model given by Equation 5. Sign of regulatory edges are the same as reported in Figure 29 (right).

ODE model	AIC	BIC	α	λ	κ	h	κ	h	Effective
and(GATA2,GFI1B)	-198	-206	555.6	0.70	0.02	0.0	0.24	1.0	GFI1B
or(GATA2,GFI1B)	-191	-199	11.0	0.80	0.00	0.0	6.17	16.6	GFI1B
GATA2	-213	-218	1940.2	0.54	0.01	0.8			GATA2
GFI1B	-193	-199	379.2	0.54	0.13	1.0			GFI1B

ODE model	AIC	BIC	α	λ	κ	h	κ	h	Effective
and(GATA2,GFI1)	-220	-228	47.5	2.77	0.0	2.4	17.5	28.0	GFI1
or(GATA2,GFI1)	-216	-224	20.4	1.44	6.5	22.8	0.0	0.0	GATA2
GATA2	-212	-218	19.0	0.99	6.2	4.6			GATA2
GFI1	-219	-224	48.2	2.76	17.5	29.0			GFI1

Table 9: Model selection and parameter estimation for GFI1 (top) and GFI1B (bottom). Values represent mode of posterior distributions for estimated parameters. Parameters from left to right are: synthesis (or production) rate (α), decay constant (λ), dissociation constant (κ) and Hill coefficient (h). When multiple dissociation constants and Hill coefficients are present, they refer to different TFs in the order they are given in the model (first column). Sign of regulatory edges are the same as reported in Figure 29 (right).

2.4 User-defined parameters and time constraints

In this section we report user-defined parameters used for the two synthetic dataset and for the real hematopoietic data (Tab. 10).

For diffusion map, the Gaussian kernel width (σ) is the only parameter used in the method. For optimal determination of this parameter we use heuristics as described in [29]. The argument is that since cell differentiation is generally a highly nonlinear process, the Euclidean distances

between cells in the Gaussian kernel are only valid within small neighbourhoods in the high-dimensional space and the radius of this neighbourhood is defined by the characteristic length scales of the manifold. According to [29], the local maxima of the average intrinsic dimensionality curve $\langle d \rangle$ indicate the characteristic length scales of the data manifold and hence the optimal σ. Figure 32 and 33 show that, close to the characteristic length scales, the diffusion map is robust to σ variations for real (qPCR) and synthetic data, respectively. Consequently we choose σ in this robustness regions.

As described in previous sections, clustering algorithm is only used for the toggle switch network data, since FFL represents a non-branching process and in real data we have information from cell surface markers. The main parameter used in clustering algorithm is the number of neighbours. For the toggle switch dataset we have used a value $N_n = 15$; however, as we have described in Section 2.2.4, the algorithm is robust to a wide range of values. In order to run the clustering algorithm, the position of a starting cell is also necessary. In Section 1.2 we described how an approximate starting cell position can be retrieved by using prior information. Once the starting cell position has been decided, the number of branches (with their final cell position) is decided by direct visual inspection of diffusion map geometry, as explained in Section 1.2.

Once the branches have been defined from a starting cell position, the same approximate starting cell position is used also in Wanderlust algorithm. Together with the starting cell position, Wanderlust relies on three main parameters: the number of nearest neighbours k, the neighbour subset size I and the number of dimensions of the dataset. As described in Section 2.2.6, Wanderlust is robust to different choices of the number of nearest neighbours k. Here we report parameters which we have used to produce results in main paper. In FFL and toggle switch data we use a larger value $k = 18$ compared to the one used in real data ($k = 15$), otherwise the k-nearest neighbour graph results disconnected with a consequent error of the algorithm [2]. Neighbour subset size I is simply set to $k/3$, such that $I < k$, as explained in Section 2.2.6. The number of dimensions is given by the dimensionality of the snapshot data (i.e. the number of genes).

Threshold for edge weights in GENIE3 depends on the specific dataset. In Section 1.4 we clarified how this threshold can be chosen empirically, by directly looking at the values of all the resulting network edge weights.

Finally, threshold for correlation analysis has been only used for simulated data with a default value 0.5. As we showed in Section 2.2.5, correlation analysis is used to reduce the number of ODE models to compare in model selection: model selection in toggle switch data was also performed without using correlation analysis (i.e. using a threshold value equal to 1), without affecting the results. In real data, we have followed [8] and computed a p-value for the significance of correlation coefficient, instead of using a threshold value 0.5.

Diffusion maps parameters	Feed-forward loop	Toggle switch network	Hematopoietic data
Gaussian kernel width σ		1000	$10^{0.9}$

Clustering parameters	Feed-forward loop	Toggle switch network	Hematopoietic data
N_n		15	
number of branches		4	

Wanderlust parameters	Feed-forward loop	Toggle switch network	Hematopoietic data
k (nearest neighbours)	18	18	15
I (neighbour subset size)	$k/3$	$k/3$	$k/3$
number of dimensions	3	6	18

GENIE3 parameters	Feed-forward loop	Toggle switch network	Hematopoietic data
weights threshold	0.3	0.1 - 0.3	0.08

Correlation analysis parameters	Feed-forward loop	Toggle switch network	Hematopoietic data
threshold	0.5	0.5	
p-value significance level			1%

Table 10: User-defined parameters used in our framework

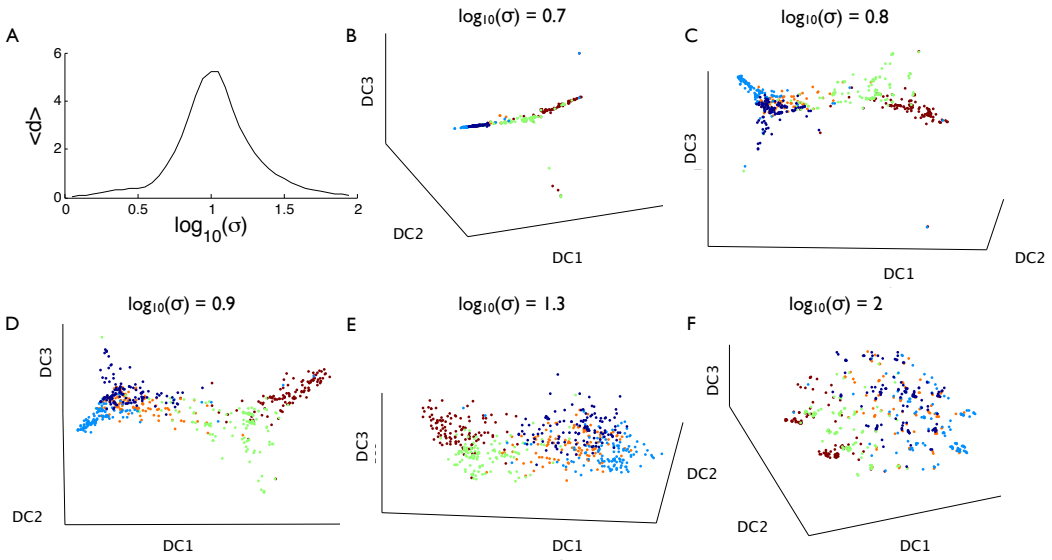

Figure 32: (A) Average intrinsic dimensionality $\langle d \rangle$ for real (qPCR) data as a function of $log_{10}(\sigma)$ shows maximum at $log_{10}(\sigma) = 0.9$. (B-F) Diffusion map for different values of σ.

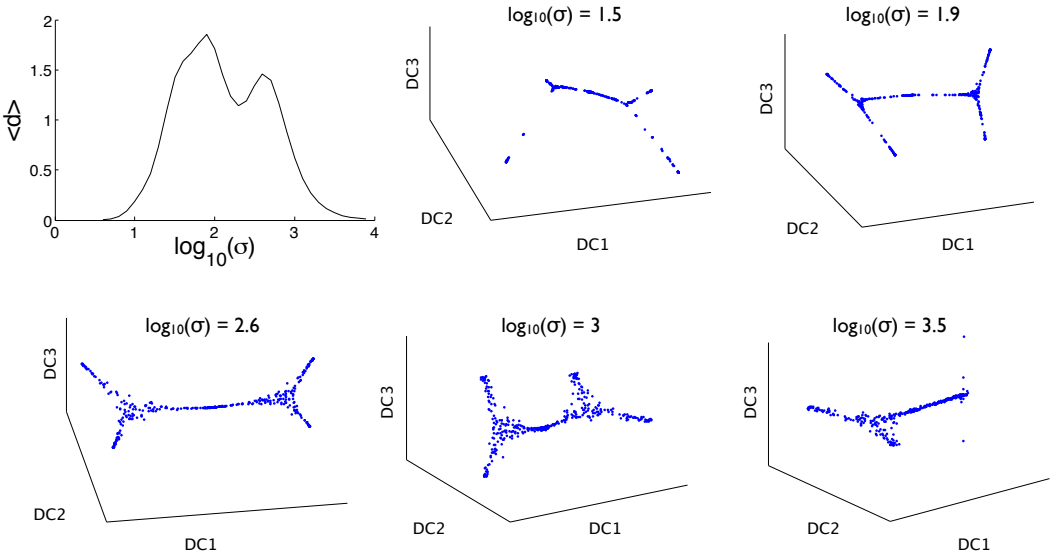

Figure 33: (A) Average intrinsic dimensionality $\langle d \rangle$ for toggle switch synthetic data as a function of $log_{10}(\sigma)$ shows two maxima at $log_{10}(\sigma) = 1.9$ and $log_{10}(\sigma) = 2.6$. (B-F) Diffusion map for different values of σ.

Data generation for simulated data, diffusion maps, branch clustering, network inference with GENIE3, Wanderlust algorithm are all performed very fast (each of them runs within 30 sec on a standard 2,5 GHz Intel Core i5, 4 GB RAM). Parameter estimation and model selection represent the most time consuming, especially when thermodynamic integration is needed. For a subsystem including a target and two input genes, parameter estimation needs around 2h with 6 pseudo time-series (i.e. 3 replicates $\times$ 2 branches), each with around 100 cells. However, as the subsystems optimisation approach can be parallelised, in 2h we can compare as many ODE models as possible (depending on computational resources). For real data, the process took several hours ($\sim$10h); in this case, the largest subnetwork is composed of a target and three input genes and the number of cells in each branch is larger.

References

[1] Coifman RR, et al. Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. PNAS, 102(21): 7426–31, 2005.

[2] Bendall SC, et al. Single-cell trajectory detection uncovers progression and regulatory coordination in human B cell development. Cell, 157(3):714–25, 2014.

[3] Huynh-Thu VA, Irrthum A, Wehenkel L, Geurts P. Inferring regulatory networks from expression data using tree-based methods. PLoS One, 5(9): e12776, 2010.

[4] Elowitz MB1, Levine AJ, Siggia ED, Swain PS. Stochastic gene expression in a single cell. Science, 297(5584): 1183–6, 2002.

[5] Tipping ME, Bishop CM. Probabilistic principal component analysis. J R Statistic Soc B, 61(3), 611–622, 1999.

[6] Lawrence ND, Gaussian process latent variable models for visualisation of high dimensional data. NIPS 2003.

[7] Moignard V, et al. Decoding the regulatory network of early blood development from single-cell gene expression measurements. Nat Biotechnol, 10.1038/nbt.3154, 2015.

[8] Moignard V, et al. Characterization of transcriptional networks in blood stem and progenitor cells using high-throughput single-cell gene expression analysis. Nat Cell Biol, 15(4): 363–72, 2013.

[9] Dijkstra, EW. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

[10] Yamamoto M, Takahashi S, Onodera K, Muraosa Y, Engel JD. Upstream and downstream of erythroid transcription factor GATA-1. Genes Cells, 2, 107?115, 1997.

[11] Mouthon MA, et al. Expression of tal-1 and GATA-binding proteins during human hematopoiesis. Blood 81, 647?655, 1993.

[12] Orlic D, Anderson S, Biesecker LG, Sorrentino BP, Bodine DM. Pluripotent hematopoietic stem cells contain high levels of mRNA for c-kit, GATA-2, p45 NF-E2, and c-myb and low levels or no mRNA for c-fms and the receptors for granulocyte colony-stimulating factor and interleukins 5 and 7. Proc Natl Acad Sci USA, 92, 4601-05, 1995.

[13] Marbach D, et al. Wisdom of crowds for robust gene network inference. Nat Methods. 9(8): 796–804, 2012.

[14] Ocone A, Millar AJ, Sanguinetti G. Hybrid regulatory models: a statistically tractable approach to model regulatory network dynamics. Bioinformatics. 29(7): 910–6, 2013.

[15] MacKay, DJC. Information Theory, Inference, and Learning Algorithms. Cambridge University Press, 2003.

[16] Bishop, CM. Pattern recognition and machine learning. Springer, 2006.

[17] Brooks, S, Gelman, A, Jones, G, Meng, XL (eds). Handbook of Markov Chain Monte Carlo. Chapman & Hall/CRC, 2011.

[18] Metropolis, N, Ulam, S. The Monte Carlo Method. Journal of the American Statistical Association, 44(247), 335–341, 1949.

[19] Barber, D. Bayesian Reasoning and Machine Learning. Cambridge University Press, 2012.

[20] Georgoulas A, Clark A, Ocone A, Gilmore S, Sanguinetti G. A subsystems approach for parameter estimation of ODE models of hybrid systems. HSB 2012: 30-41.

[21] O'Hagan, A. Bayesian analysis of computer code outputs: a tutorial. Reliability Engineering and System Safety, 91, 1290–1300, 2006.

[22] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. The MIT Press, Cambridge, MA, 2006.

[23] Rasmussen CE, Nickisch H. Gaussian Processes for Machine Learning (GPML) Toolbox. Journal of Machine Learning Research, 11, 3011–3015, 2010.

[24] Stein ML. Interpolation of spatial data. Springer-Verlag, New York, 1999.

[25] Vyshemirsky, V, Girolami, M. Bayesian ranking of biochemical system models. Bioinformatics, 24(6), 833–9, 2008.

[26] Calderhead B, Girolami M. Estimating Bayes factors via thermodynamic integration and population MCMC. Computational Statistics & Data Analysis, 53(12): 4028–4045, 2009.

[27] Friel N, Pettitt AN. Marginal likelihood estimation via power posteriors. J R Statist Soc B, 70(3): 589–607, 2008.

[28] Jeffreys H. Theory of Probability. 3rd edition Oxford, Clarendon Press, 1961.

[29] Haghverdi L, Büttner F, Theis FJ. Diffusion maps for high-dimensional single-cell analysis of differentiation data. Bioinformatics (accepted), 2015.