Simulation and reverse-engineering of mechanistic GRN-driven models of gene expression

Hands-on session 1: modeling and simulation

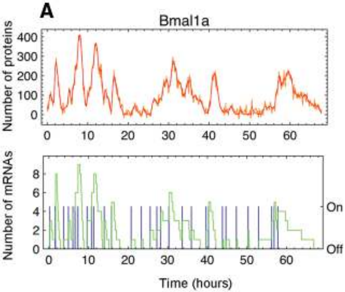
Ulysse Herbach & Elias Ventre

Inria (Nancy & Marseille)

CompSysBio 2025 - Aussois

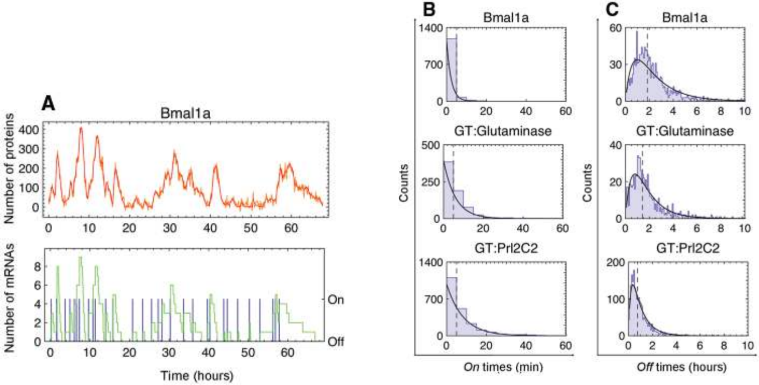
6 October 2025

The transcriptional bursting phenomenon



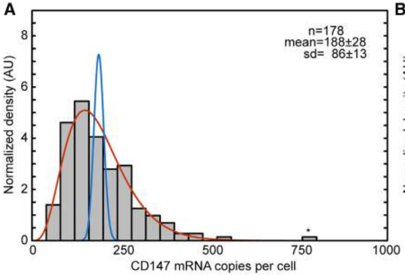
D. Suter, N. Molina *et al.*, *Science*, 2011

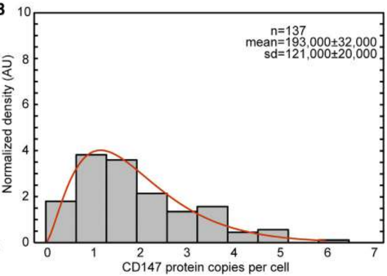
The transcriptional bursting phenomenon



D. Suter, N. Molina *et al.*, *Science*, 2011

Confirmation of biological variability





C. Albayrak, C. Jordi *et al.*, *Molecular Cell*, 2016

1. Modeling

Simplification steps

$$E \underset{k_{\text{off}}}{\overset{k_{\text{on}}}{\rightleftharpoons}} E^*$$

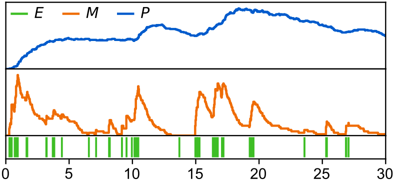
$$E^* \xrightarrow{s_0} E^* + M$$

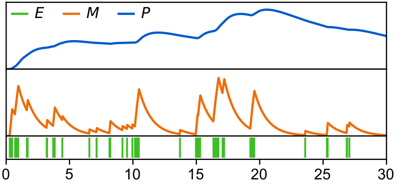
$$M \xrightarrow{s_1} M + P$$

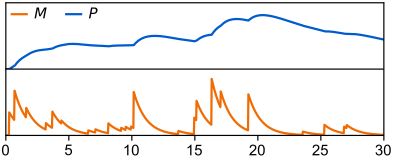
$$M \xrightarrow{d_0} \varnothing$$

$$P \xrightarrow{d_1} \varnothing$$

Simplification steps







$$E \underset{k_{\text{off}}}{\overset{k_{\text{on}}}{\rightleftharpoons}} E^*$$

$$E^* \xrightarrow{s_0} E^* + M$$

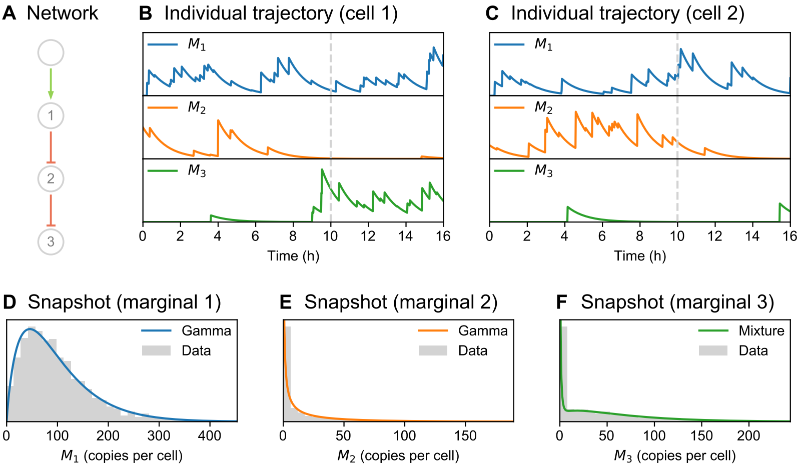
$$M \xrightarrow{s_1} M + P$$

$$M \xrightarrow{d_0} \varnothing$$

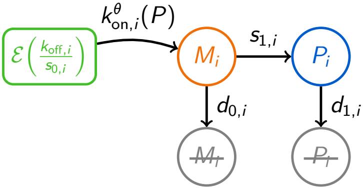
$$P \xrightarrow{d_1} \varnothing$$

Gamma distributions: **yes** ✔

Trajectories vs. distributions

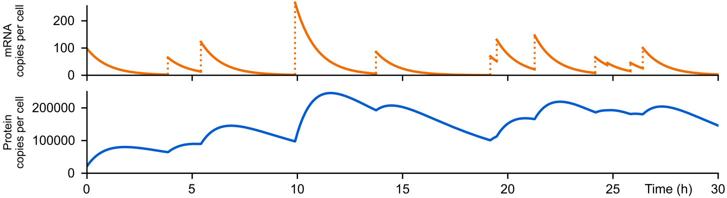


Dynamical GRN model



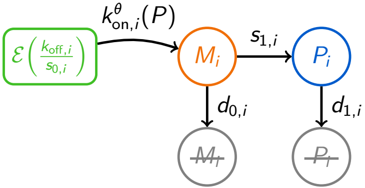
$k^{\theta}_{\text{on},i}(P)$
$\mathcal{E}\left(\frac{k_{\text{off},i}}{s_{0,i}}\right)$
M_i
$s_{1,i}$
P_i
$d_{0,i}$
$d_{1,i}$
$\cancel{M_i}$
$\cancel{P_i}$

$$\begin{cases} M_i(t) \xrightarrow{k^{\theta}_{\text{on},i}(P(t))} M_i(t) + \mathcal{E}\left(\frac{k_{\text{off},i}}{s_{0,i}}\right) \\ M_i'(t) = -d_{0,i} M_i(t) \\ P_i'(t) = s_{1,i} M_i(t) - d_{1,i} P_i(t) \end{cases}$$

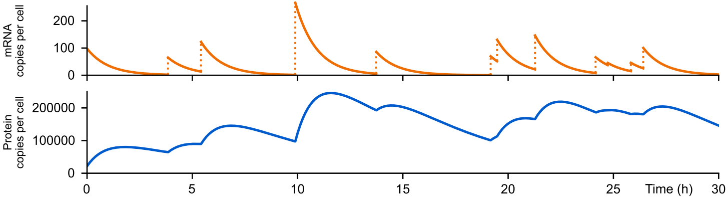


E. Ventre, U. Herbach *et al.*, *PLOS Computational Biology*, 2023

Dynamical GRN model


$k^{\theta}_{\text{on},i}(P)$
$\mathcal{E}\left(\frac{k_{\text{off},i}}{s_{0,i}}\right)$
M_i
$s_{1,i}$
P_i
$d_{0,i}$
$d_{1,i}$
~~M_i~~
~~P_i~~

$$\begin{cases} M_i(t) \xrightarrow{k^{\theta}_{\text{on},i}(P(t))} M_i(t) + \mathcal{E}\left(\frac{k_{\text{off},i}}{s_{0,i}}\right) \\ M_i'(t) = -d_{0,i} M_i(t) \\ P_i'(t) = s_{1,i} M_i(t) - d_{1,i} P_i(t) \end{cases}$$



Interaction function (burst frequency of gene *i*)

$$k_{\text{on},i}(P_1, \ldots, P_n) = \frac{k_{1,i} \exp(\beta_i + \sum_{j=1}^{n} \theta_{ji} P_j)}{1 + \exp(\beta_i + \sum_{j=1}^{n} \theta_{ji} P_j)}$$

E. Ventre, U. Herbach *et al.*, *PLOS Computational Biology*, 2023

2. Simulation

Rigorous definitions

The time-dependent multivariate distribution $p(t, y, z)$ of mRNA y and proteins z follows a **continuous master equation**:

Complete model (used for simulation)

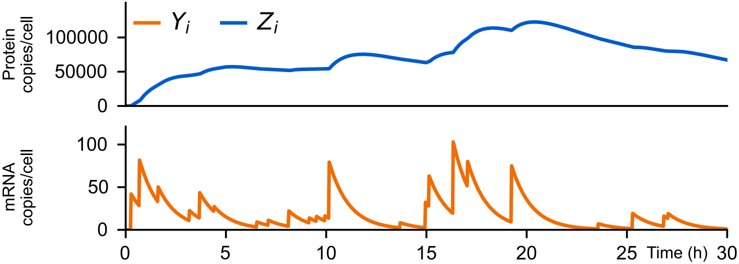
$$\frac{\partial}{\partial t} p(t, y, z) = \sum_{i=1}^{n} \Bigg[d_{0,i} \frac{\partial}{\partial y_i} \big\{ y_i p(t, y, z) \big\} + d_{1,i} \frac{\partial}{\partial z_i} \big\{ (z_i - y_i) p(t, y, z) \big\} \\ + k_{\mathrm{on},i}(z) \left(\int_0^{y_i} p(t, y - h e_i, z) b_i e^{-b_i h} \mathrm{d}h - p(t, y, z) \right) \Bigg]$$

Rigorous definitions

The time-dependent multivariate distribution $p(t, y, z)$ of mRNA y and proteins z follows a **continuous master equation**:

Complete model (used for simulation)

$$\frac{\partial}{\partial t} p(t,y,z) = \sum_{i=1}^{n} \Bigg[d_{0,i} \frac{\partial}{\partial y_i}\{y_i p(t,y,z)\} + d_{1,i} \frac{\partial}{\partial z_i}\{(z_i - y_i) p(t,y,z)\} \\ + k_{\mathrm{on},i}(z) \left(\int_0^{y_i} p(t, y - h e_i, z) b_i e^{-b_i h} \mathrm{d}h - p(t,y,z) \right) \Bigg]$$



Rigorous definitions

The time-dependent multivariate distribution $p(t,x)$ of proteins x follows a **continuous master equation**:

Reduced model (used for inference)

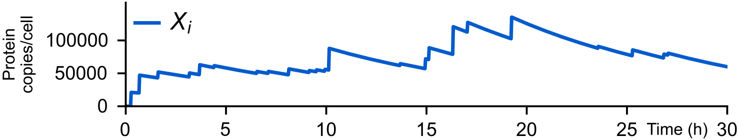
$$\frac{\partial}{\partial t}p(x,t) = \sum_{i=1}^{n}\Bigg[d_{1,i}\frac{\partial}{\partial x_i}\big\{x_i p(x,t)\big\} - k_{\mathrm{on},i}(x)p(x,t) + \int_0^{x_i} k_{\mathrm{on},i}(x-he_i)p(x-he_i,t)c_i e^{-c_i h}\mathrm{d}h\Bigg]$$

Rigorous definitions

The time-dependent multivariate distribution $p(t, x)$ of proteins x follows a **continuous master equation**:

Reduced model (used for inference)

$$\frac{\partial}{\partial t} p(x,t) = \sum_{i=1}^{n} \left[d_{1,i} \frac{\partial}{\partial x_i} \{ x_i p(x,t) \} - k_{\mathrm{on},i}(x) p(x,t) \right. \\ \left. + \int_0^{x_i} k_{\mathrm{on},i}(x - h e_i) p(x - h e_i, t) c_i e^{-c_i h} \mathrm{d}h \right]$$



Mathematical setting

Waiting time distribution

$$\mathbb{P}_{y,z}(T_1 > t) = \exp\left(-\int_0^t \sum_{i=1}^n k_{\mathrm{on},i}(\varphi_{\mathrm{P}}(y,z,\tau))\mathrm{d}\tau\right)$$

Problem: *numerical integration would be inefficient!*

Mathematical setting

Waiting time distribution

$$\mathbb{P}_{y,z}(T_1 > t) = \exp\left(- \int_0^t \sum_{i=1}^{n} k_{\mathrm{on},i}(\varphi_{\mathrm{P}}(y,z,\tau)) \mathrm{d}\tau \right)$$

Problem: *numerical integration would be inefficient!*

Main assumption: $\exists\, \lambda \geqslant \sup_{z \in \mathbb{R}_+^n} \left\{ \sum_{i=1}^{n} k_{\mathrm{on},i}(z) \right\}$

Mathematical setting

Waiting time distribution

$$\mathbb{P}_{y,z}(T_1 > t) = \exp\left(- \int_0^t \sum_{i=1}^n k_{\mathrm{on},i}(\varphi_{\mathrm{P}}(y,z,\tau))\mathrm{d}\tau \right)$$

Problem: *numerical integration would be inefficient!*

Main assumption: $\exists\, \lambda \geqslant \sup_{z \in \mathbb{R}_+^n} \left\{ \sum_{i=1}^n k_{\mathrm{on},i}(z) \right\}$

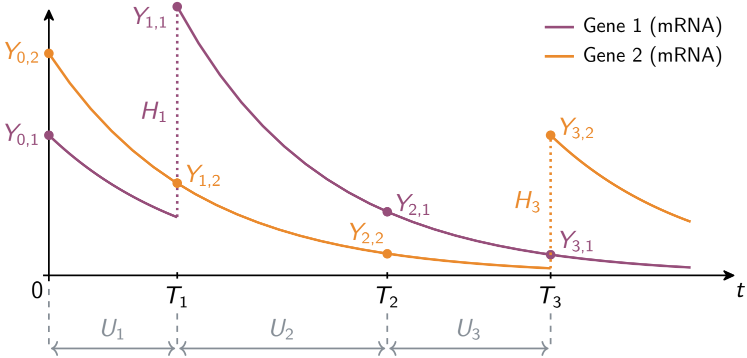
Acceptance-rejection (aka thinning) method

$$p_i(z) = \begin{cases} 1 - \dfrac{1}{\lambda} \displaystyle\sum_{i=1}^n k_{\mathrm{on},i}(z) & \text{if } i = 0 \\ \dfrac{k_{\mathrm{on},i}(z)}{\lambda} & \text{if } 1 \leqslant i \leqslant n \end{cases}$$

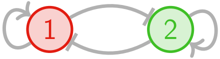
Basic algorithm (SSA-like)

```
Require: initial state $(Y_0, Z_0)$ and final time $t > 0$
$Y, Z \leftarrow Y_0, Z_0$                          ▷ Initialize current state
$T \leftarrow 0$                                    ▷ Initialize current jump time
while $T < t$ do
    $Y_{\text{old}}, Z_{\text{old}} \leftarrow Y, Z$
    $T_{\text{old}} \leftarrow T$
    $U \leftarrow \text{Exp}(\lambda)$              ▷ Draw waiting time
    $Y, Z \leftarrow \varphi(Y_{\text{old}}, Z_{\text{old}}, U)$    ▷ Apply the deterministic flow
    $i \leftarrow \mathcal{P}(Z)$                   ▷ Draw gene i
    if $i \neq 0$ then
        $Y[i] \leftarrow Y[i] + \text{Exp}(b_i)$    ▷ Apply jump
    end if
    $T \leftarrow T + U$                            ▷ Update current jump time
end while
return $\varphi(Y_{\text{old}}, Z_{\text{old}}, t - T_{\text{old}})$    ▷ Extend to final time
```

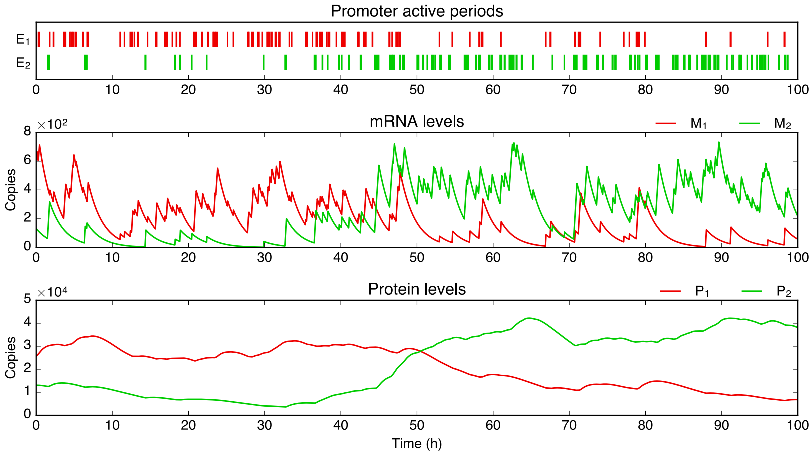
Illustration of the algorithm



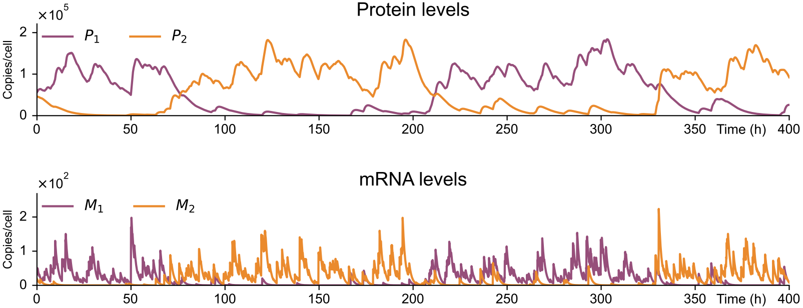
Example 1: toggle switch (two-state model)



$$\theta = \begin{pmatrix} + & - \\ - & + \end{pmatrix}$$

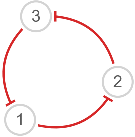


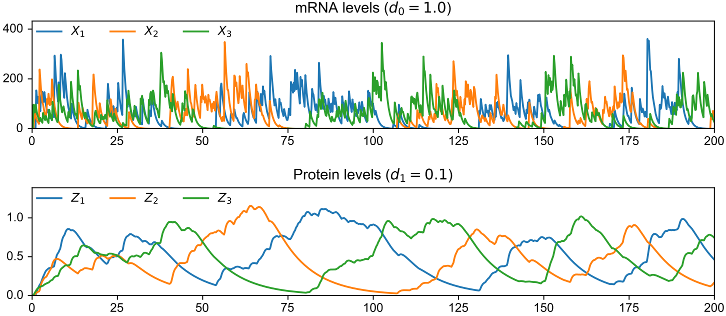
Example 1: toggle switch (bursty model)



Example 2: repressilator

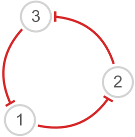
$$\beta_1 = \beta_2 = \beta_3 = 5, \quad \theta_{12} = \theta_{23} = \theta_{31} = -10$$

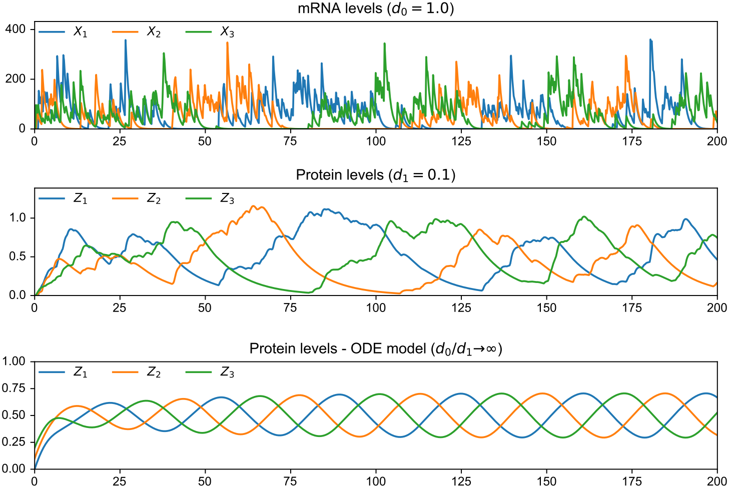




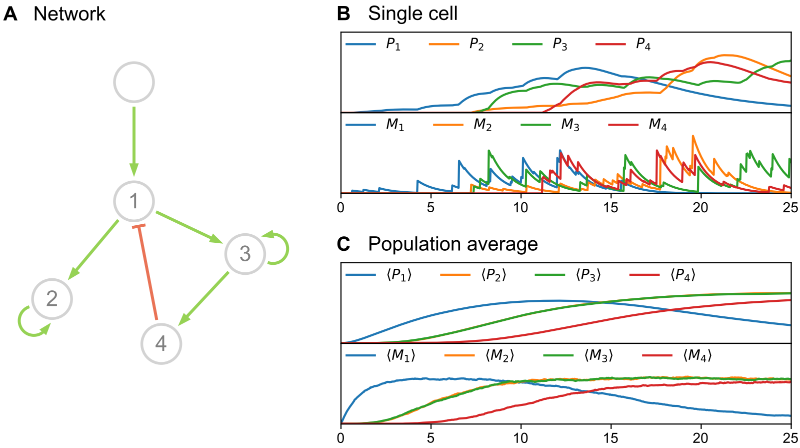
Example 2: repressilator

$$\beta_1 = \beta_2 = \beta_3 = 5, \quad \theta_{12} = \theta_{23} = \theta_{31} = -10$$





Single-cell vs. bulk (average) trajectories



A Network

B Single cell

C Population average

3. Inference

How do we switch to statistical learning?

Main idea

- We consider the invariant distribution $p(x)$ as a **statistical likelihood** parametrized by $\theta = (\theta_{ij})_{1 \leqslant i,j \leqslant n}$
- Proteins $X = (X_1, \ldots, X_n)$ are interpreted as a **latent space**, with mRNA levels $Y = (Y_1, \ldots, Y_n)$ being sampled from

$$Y \sim \bigotimes_{i=1}^{n} \mathrm{Gamma}(k_{\mathrm{on},i}(X)/d_{0,i}, b_i)$$

i.e. the *quasi-steady-state* distribution of the complete model.

How do we switch to statistical learning?

Main idea

- We consider the invariant distribution $p(x)$ as a **statistical likelihood** parametrized by $\theta = (\theta_{ij})_{1 \leqslant i,j \leqslant n}$
- Proteins $X = (X_1, \dots, X_n)$ are interpreted as a **latent space**, with mRNA levels $Y = (Y_1, \dots, Y_n)$ being sampled from

$$Y \sim \bigotimes_{i=1}^{n} \mathrm{Gamma}(k_{\mathrm{on},i}(X)/d_{0,i}, b_i)$$

i.e. the *quasi-steady-state* distribution of the complete model.

Two possible strategies

1. Use analytically tractable solutions
2. Use self-consistent approximation (*pseudo-likelihood*)

1. Using a class of very nice models

Analytical solution

Assume that there exists a function $V : (\mathbb{R}_+)^n \to \mathbb{R}$ such that for all $i = 1, \ldots, n$,

$$\frac{k_{\text{on},i}(x)}{d_{1,i} x_i} = -\frac{\partial V}{\partial x_i}(x)$$

Then the protein distribution is

$$p(x) \propto e^{-V(x)} \prod_{i=1}^{n} x_i^{-1} e^{-c_i x_i}$$

1. Using a class of very nice models

Analytical solution

Assume that there exists a function $V : (\mathbb{R}_+)^n \to \mathbb{R}$ such that for all $i = 1, \ldots, n$,

$$\frac{k_{\mathrm{on},i}(x)}{d_{1,i} x_i} = -\frac{\partial V}{\partial x_i}(x)$$

Then the protein distribution is

$$p(x) \propto e^{-V(x)} \prod_{i=1}^{n} x_i^{-1} e^{-c_i x_i}$$

Corollary 1: “GRN-informed MCMC”

We can sample from virtually any distribution using a GRN!

Corollary 2: “GRN-informed autoencoder”

Choose a **relevant parametric class** (V_θ) and learn θ from data

2. Besag's pseudo-likelihood

Definition

Besag's *pseudo-likelihood* associated with $p(x)$ is the **product of conditional densities**

$$\widetilde{p}(x) = \prod_{i=1}^{n} p^{(i)}(x) \quad \text{where} \quad p^{(i)}(x) = p(x_i | \{x_j\}_{j \neq i})$$

2. Besag's pseudo-likelihood

Definition

Besag's *pseudo-likelihood* associated with $p(x)$ is the **product of conditional densities**

$$\widetilde{p}(x) = \prod_{i=1}^{n} p^{(i)}(x) \quad \text{where} \quad p^{(i)}(x) = p(x_i | \{x_j\}_{j \neq i})$$

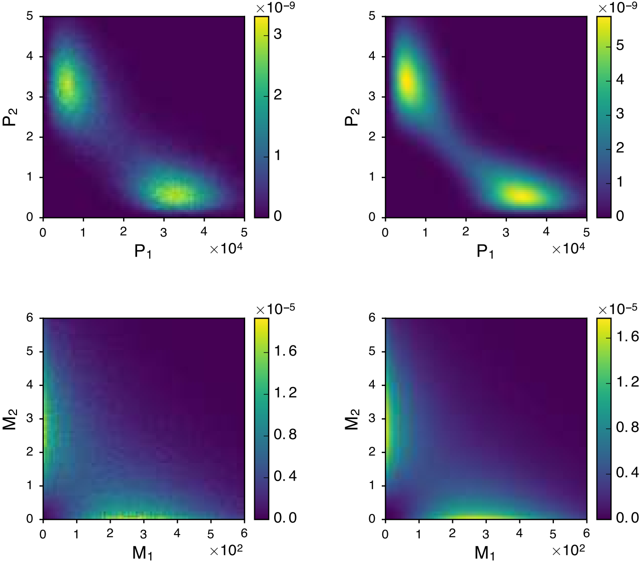
Mechanistic interpretation

$\widetilde{p}(x)$ turns out to be the **product of "frozen" steady-state solutions of the master equation**:

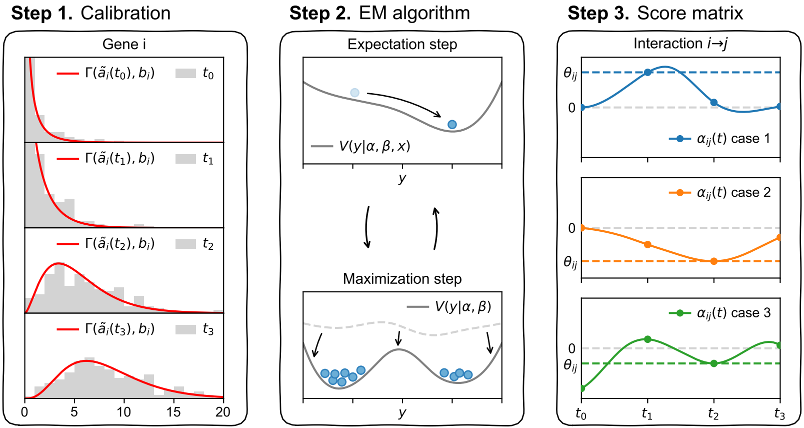
$$-d_{1,i}\partial_{x_i}\{x_i p^{(i)}(x)\} = -k_{\mathrm{on},i}(x)p^{(i)}(x) + \int_0^{x_i} k_{\mathrm{on},i}(x - he_i)p^{(i)}(x - he_i)c_i e^{-c_i h}\mathrm{d}h$$

Similar as "**self-consistent field approximation**" in physics

Exact distribution vs. pseudo-likelihood



Inference in practice - version 0.1 (2017)



Step 1. Calibration

Step 2. EM algorithm

Step 3. Score matrix

Current inference procedure - Harissa

Step 1: estimate the frequency modes $\alpha_k \in \{0,1\}^n$ in each cell k

Current inference procedure - Harissa

Step 1: estimate the frequency modes $\alpha_k \in \{0,1\}^n$ in each cell k

Step 2: *match the observed modes α_k with model fixed points*

1. Likelihood-based cost:

$$R(\theta, \alpha) = -\sum_k \sum_{i=1}^{n} L_i(\theta; \alpha_k)$$

Current inference procedure - Harissa

Step 1: estimate the frequency modes $\alpha_k \in \{0,1\}^n$ in each cell k

Step 2: *match the observed modes* α_k *with model fixed points*

1. Likelihood-based cost:

$$R(\theta, \alpha) = -\sum_k \sum_{i=1}^n L_i(\theta; \alpha_k)$$

2. Sequential optimization (*update* $\widehat{\theta}$ *at each time point*):

$$\widehat{\theta}(t_0) \leftarrow \arg\min_{\theta \in \Theta} \{R(\theta, \widehat{\alpha}(t_0)) + \lambda \|\theta\|_1\}$$

$$\widehat{\theta}(t_1) \leftarrow \arg\min_{\theta \in \Theta} \left\{R(\theta, \widehat{\alpha}(t_1)) + \lambda \|\theta - \widehat{\theta}(t_0)\|_1\right\}$$

$$\widehat{\theta}(t_2) \leftarrow \arg\min_{\theta \in \Theta} \left\{R(\theta, \widehat{\alpha}(t_2)) + \lambda \|\theta - \widehat{\theta}(t_1)\|_1\right\}$$

$$\dots$$

Current inference procedure - Cardamom

Step 1: estimate the frequency modes $\alpha_k \in \{0,1\}^n$ in each cell k

Step 2: *match the observed modes α_k with model fixed points*

1. Quadratic cost:

$$R(\theta,\alpha) = \sum_k \sum_{i=1}^{n} \left(\sigma_i^{\theta}(\alpha_k) - \alpha_{k,i}\right)^2$$

2. Sequential optimization (*update $\widehat{\theta}$ at each time point*):

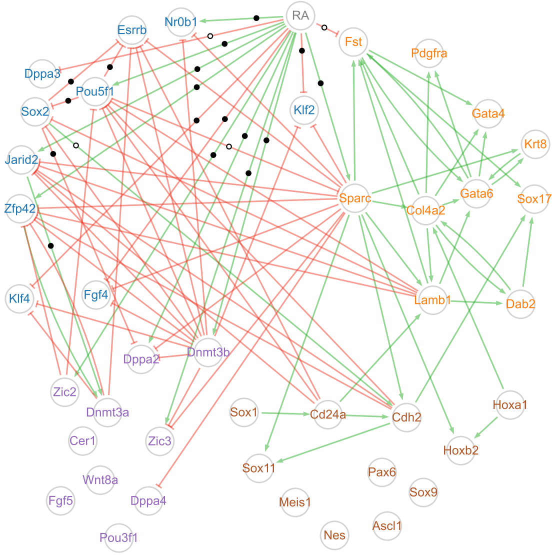
$$\widehat{\theta}(t_0) \leftarrow \operatorname*{arg\,min}_{\theta\in\Theta} \{R(\theta, \widehat{\alpha}(t_0)) + \lambda\|\theta\|_1\}$$

$$\widehat{\theta}(t_1) \leftarrow \operatorname*{arg\,min}_{\theta\in\Theta} \left\{R(\theta, \widehat{\alpha}(t_1)) + \lambda\|\theta - \widehat{\theta}(t_0)\|_1\right\}$$

$$\widehat{\theta}(t_2) \leftarrow \operatorname*{arg\,min}_{\theta\in\Theta} \left\{R(\theta, \widehat{\alpha}(t_2)) + \lambda\|\theta - \widehat{\theta}(t_1)\|_1\right\}$$

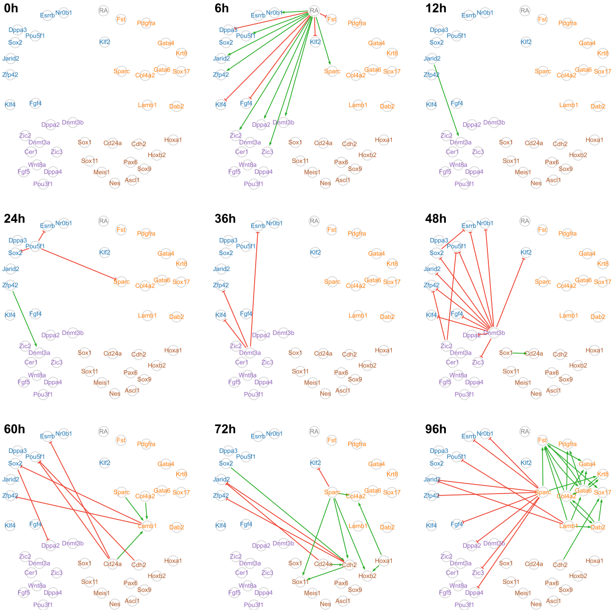
$$\dots$$

Real data example: inferred network (data from Semrau *et al.*, 2017)

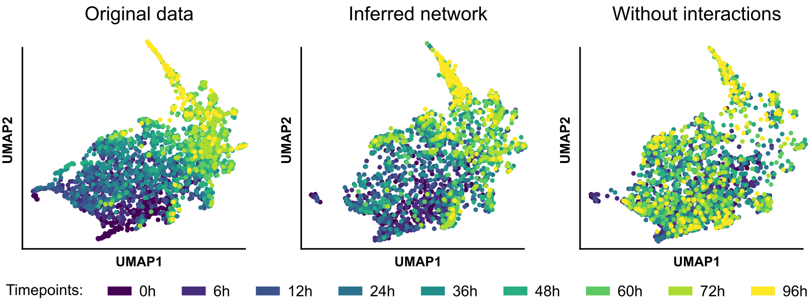


E. Ventre, U. Herbach *et al.*, *PLOS Computational Biology*, 2023

Dynamical viewpoint

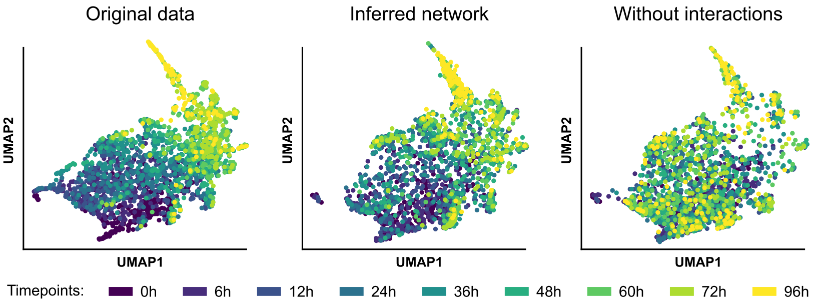


Simulation of the inferred network



E. Ventre, U. Herbach *et al.*, *PLOS Computational Biology*, 2023

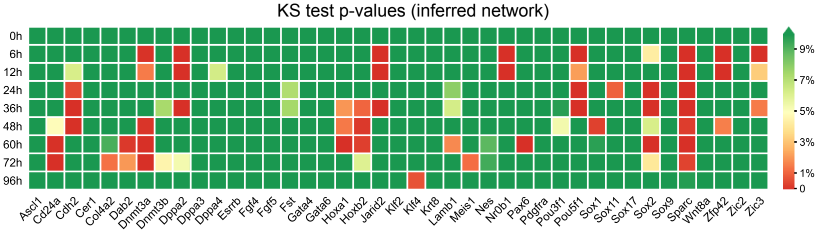
Simulation of the inferred network



Original data

Inferred network

Without interactions



KS test p-values (inferred network)

E. Ventre, U. Herbach *et al.*, *PLOS Computational Biology*, 2023

References

Gaillard, M. and Herbach, U. (2025).
Efficient stochastic simulation of gene regulatory networks using hybrid models of transcriptional bursting.
Accepted in CMSB 2025 conference in Lyon (sept. 10-12).

Ventre, E., Herbach, U., Espinasse, T., Benoit, G., and Gandrillon, O. (2023).
One model fits all: Combining inference and simulation of gene regulatory networks.
PLOS Computational Biology, 19(3):e1010962.

Ventre, E. (2021).
Reverse engineering of a mechanistic model of gene expression using metastability and temporal dynamics.
In Silico Biology, 14(3-4):89–113.

Herbach, U., Bonnaffoux, A., Espinasse, T., and Gandrillon, O. (2017).
Inferring gene regulatory networks from single-cell data: a mechanistic approach.
BMC Systems Biology, 11(1):105.