Advanced Machine Learning

Lecture 3b
28th February, 2019

Paper: V. A. Huynh-Thu and G. Sanguinetti, 2015
Combining tree-based and dynamical systems for the inference of gene regulatory networks.

Vân Anh Huynh-Thu
vahuynh@uliege.be

LIÈGE université

Outline

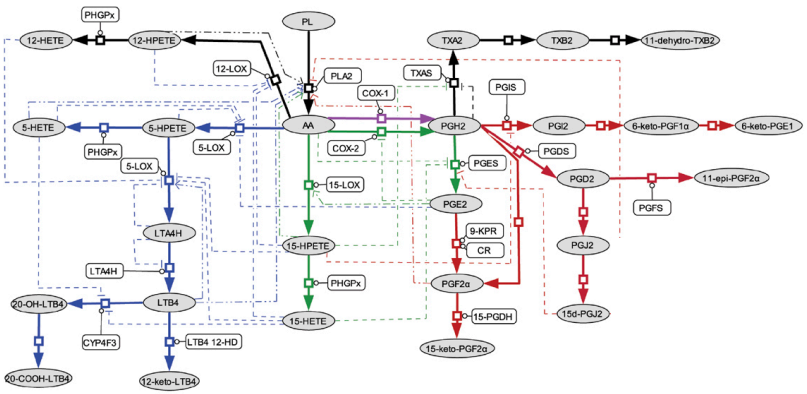
Background on gene network inference

Gene network inference with decision trees:

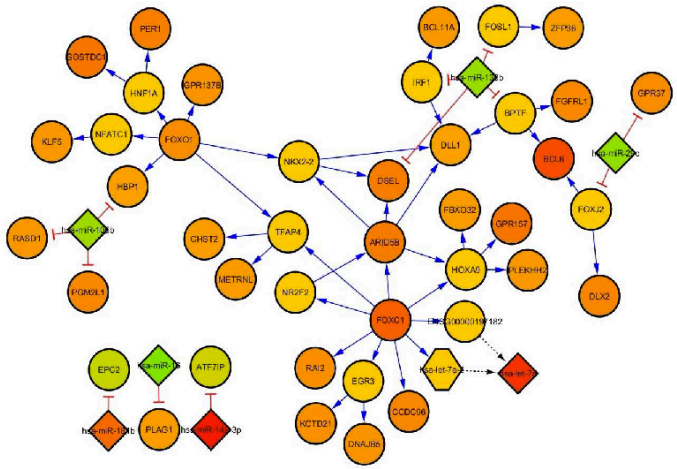
- GENIE3 (a model-free approach)
- **Jump3** (a hybrid model-free/model-based approach)

Networks are very common to represent biological information

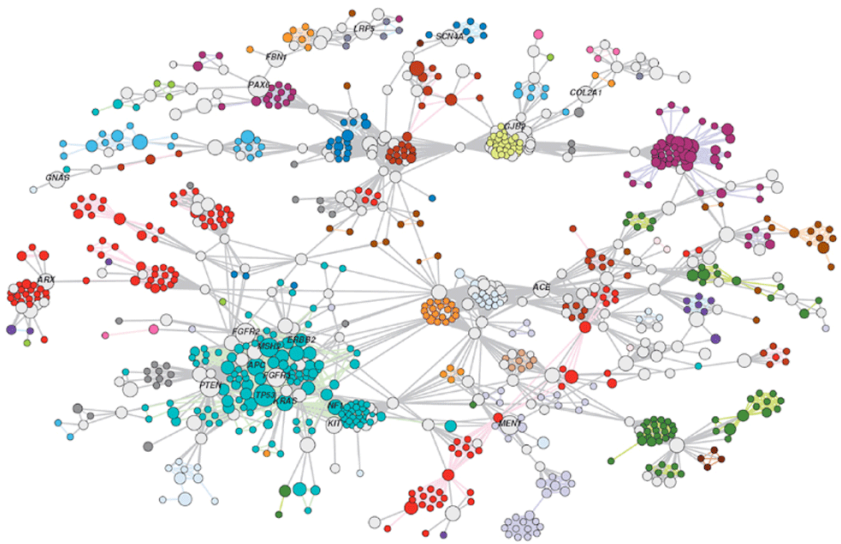
Metabolic network



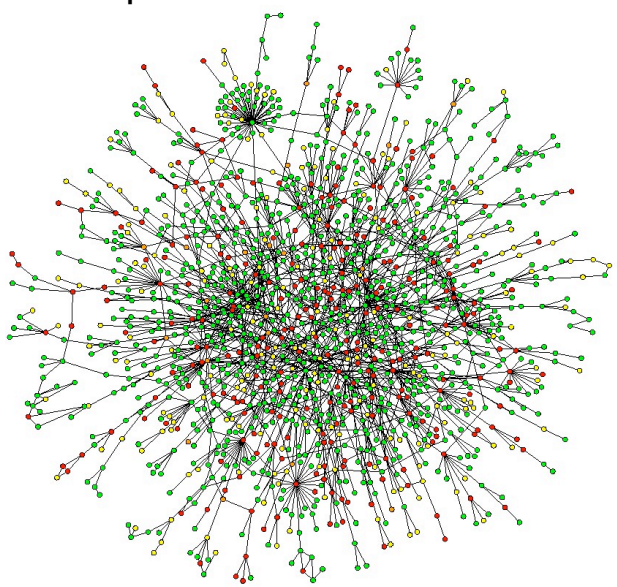
Gene regulatory network



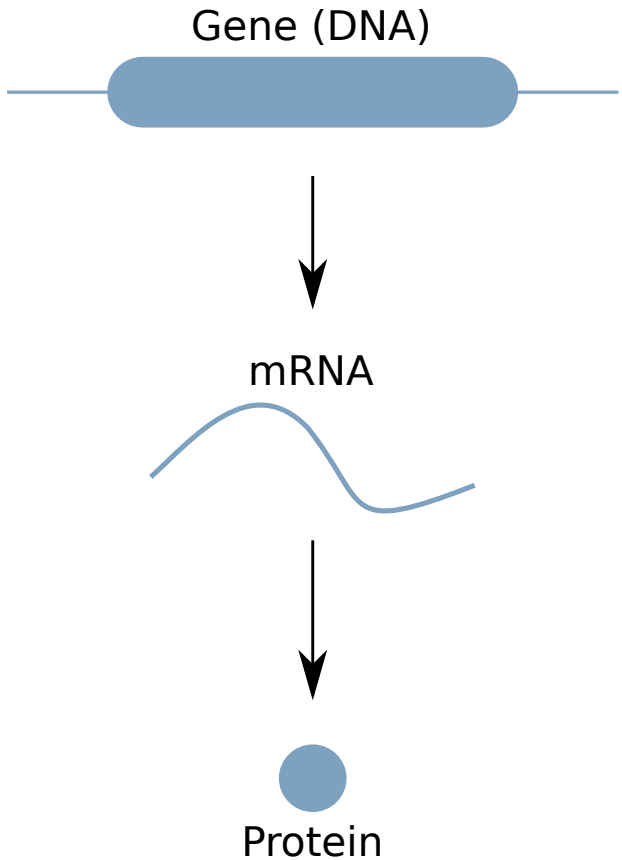
Disease gene network



Protein-protein interaction network

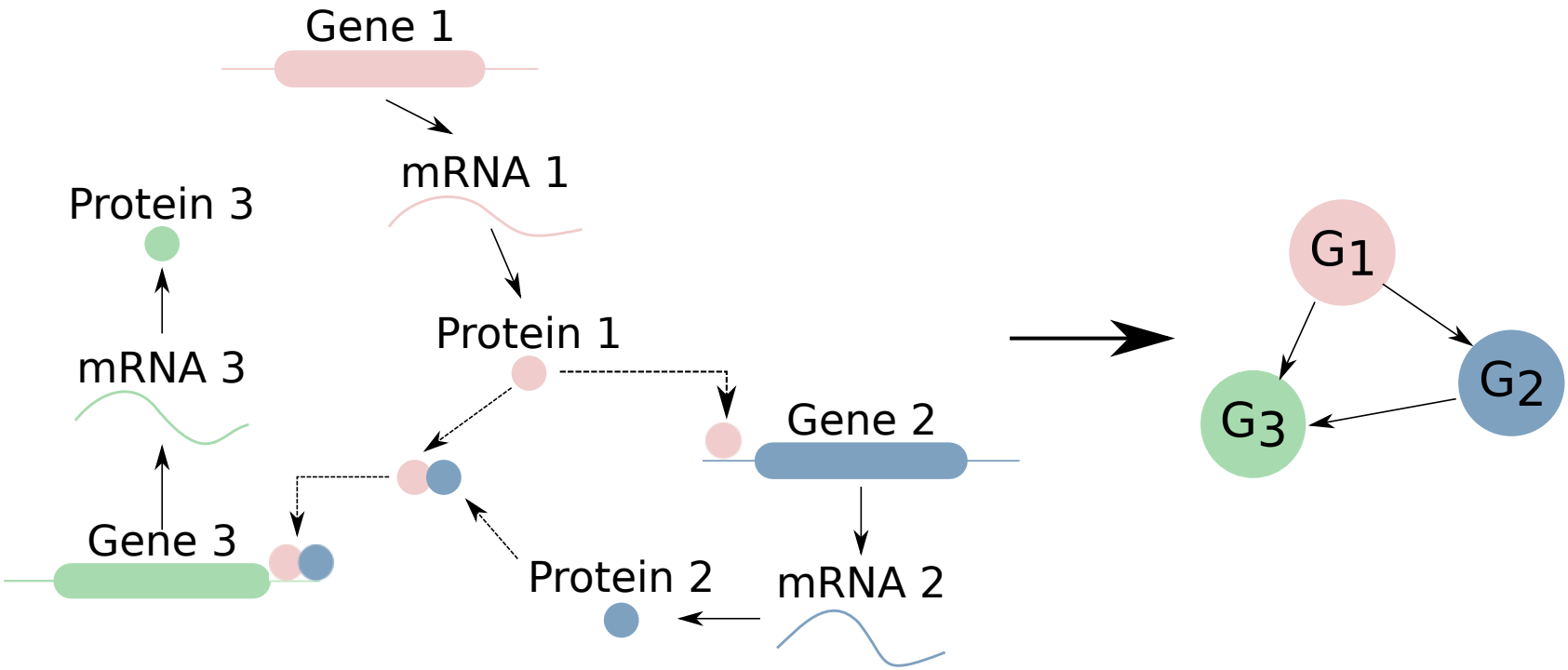


Proteins are produced from genes



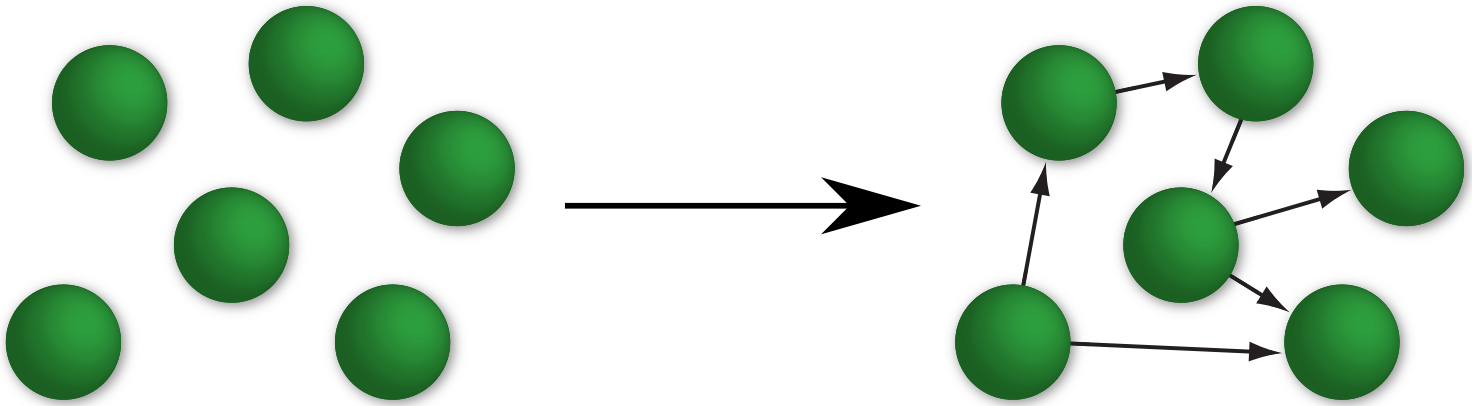
Expression of a gene = conversion of the DNA into a protein

The expression of a gene is regulated by other genes



Gene regulatory network: simplified representation of the gene regulation system.

Inferring regulatory networks is a challenging problem



unknown network

inferred network

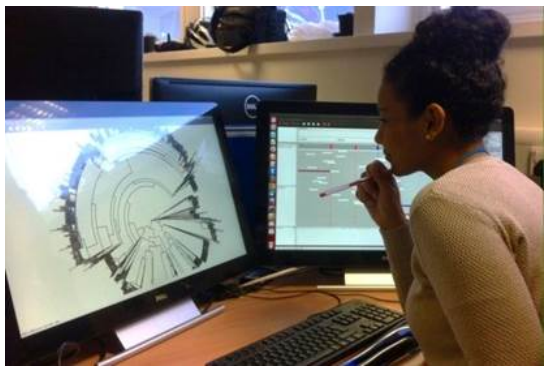
The goal is to identify the edges of network.

How to infer gene networks?

The experimental approach

- Time-consuming
- Expensive
- Not practically feasible for large networks

Bioinformatics



Development of computational methods to infer networks

High-throughput (and cheap) techniques can measure the expression levels of all the genes in a sample

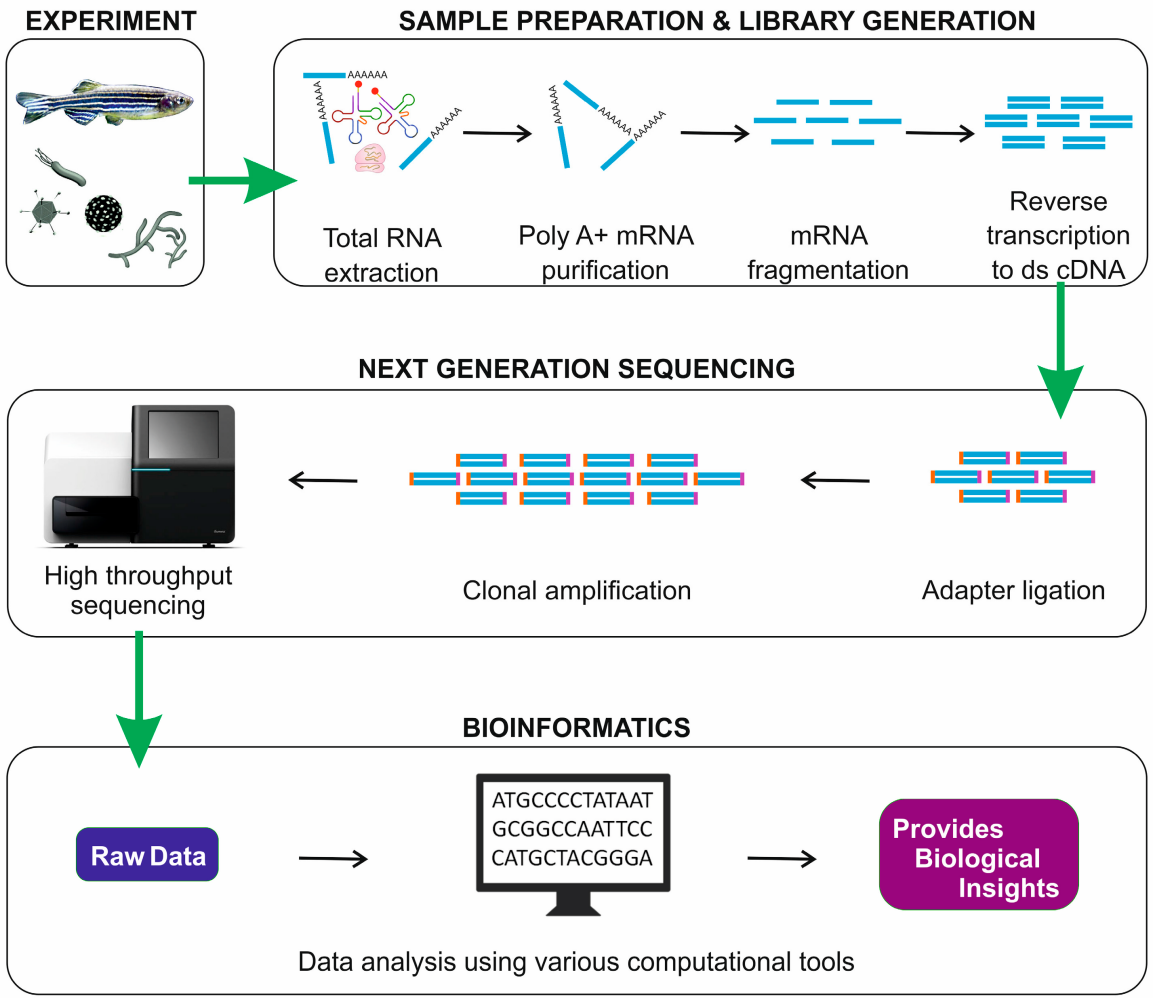


Image from https://www.rna-seqblog.com

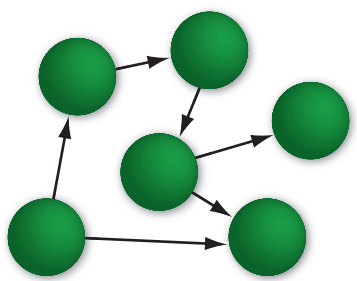
Computational methods typically infer networks from expression data

$\sim$ 10000 genes

	Gene 1	*Gene 2*	$\cdots$	*Gene p*
Sample 1	0.61	0.41	$\cdots$	0.51
Sample 2	2.30	0.10	$\cdots$	0.21
Sample 3	0.33	0.45	$\cdots$	0.30
Sample 4	0.23	0.87	$\cdots$	0.09
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Sample N	0.69	0.61	$\cdots$	0.02

$\sim$ 100 samples

Network inference algorithm



Two types of expression data

Steady-state data: expressions are measured once the system has reached some equilibrium point.
$\rightarrow$ Easy to collect, but offer limited information.

Time series data: expressions are measured at several time points following the perturbation.
$\rightarrow$ More informative about the dynamics, but are scarce.

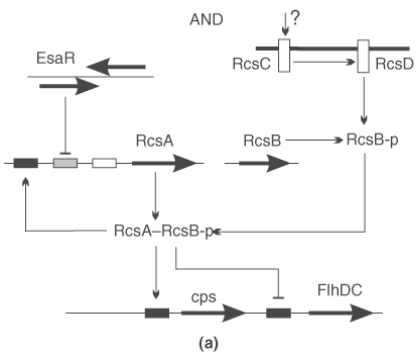
Two main families of network inference methods

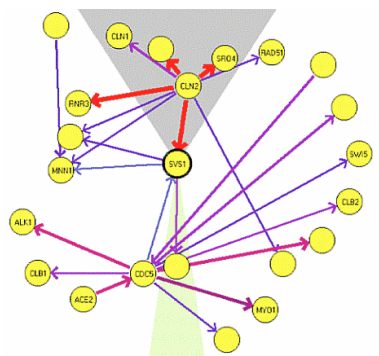
Model-free: compute statistical dependencies between pairs of expression profiles (e.g. linear correlation)

		Target gene			
		gene 1	**gene 2**	$\cdots$	**gene p**
Regulating gene	**gene 1**	-	0.05	$\cdots$	0.56
	gene 2	0.19	-	$\cdots$	0.03
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	gene p	0.11	0.42	$\cdots$	-

$\rightarrow$ Fast, but can not make predictions

Model-based: learn a model capturing the dynamics of the network (e.g. differential equations)





$$\frac{dx_i}{dt} = m_i \cdot f_i(\mathbf{y}) - \lambda_i^{\mathrm{RNA}} \cdot x_i$$

$$\frac{dy_i}{dt} = r_i \cdot x_i - \lambda_i^{\mathrm{Prot}} \cdot y_i,$$

$\rightarrow$ Realistic, but are limited to small networks

GENIE3: a model-free approach for steady-state expression data (Huynh-Thu *et al.*, 2010)

OPEN ACCESS Freely available online

PLoS one

Inferring Regulatory Networks from Expression Data Using Tree-Based Methods

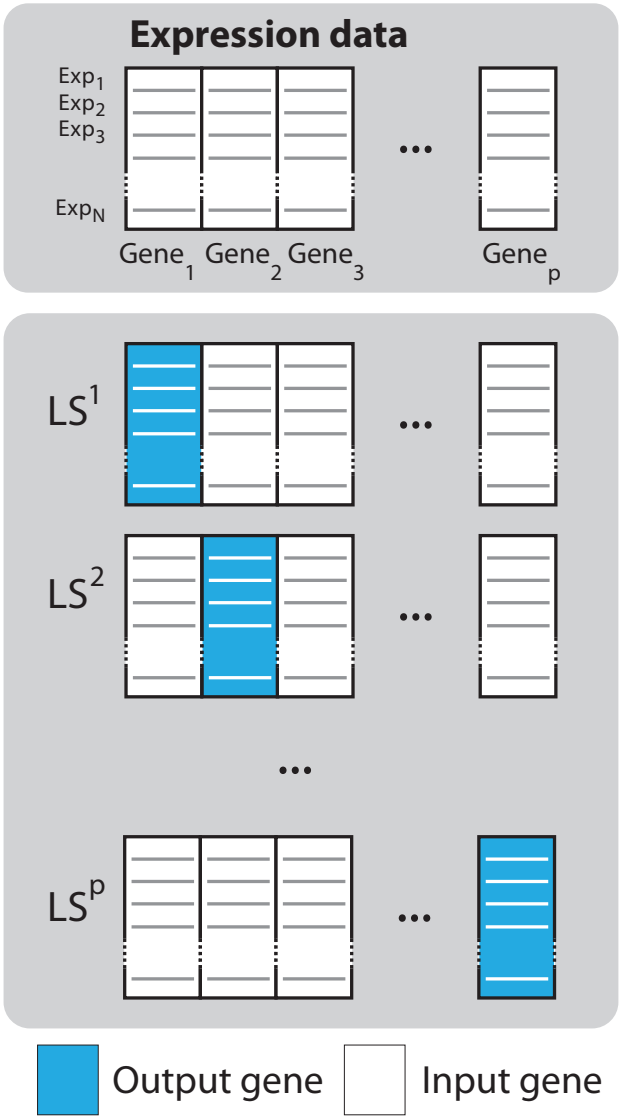
Vân Anh Huynh-Thu[1,2]*, Alexandre Irrthum[1,2], Louis Wehenkel[1,2], Pierre Geurts[1,2]

1 Department of Electrical Engineering and Computer Science, Systems and Modeling, University of Liège, Liège, Belgium, **2** GIGA-Research, Bioinformatics and Modeling, University of Liège, Liège, Belgium

Abstract

One of the pressing open problems of computational systems biology is the elucidation of the topology of genetic regulatory networks (GRNs) using high throughput genomic data, in particular microarray gene expression data. The Dialogue for Reverse Engineering Assessments and Methods (DREAM) challenge aims to evaluate the success of GRN inference algorithms on benchmarks of simulated data. In this article, we present GENIE3, a new algorithm for the inference of GRNs that was best performer in the DREAM4 *In Silico Multifactorial* challenge. GENIE3 decomposes the prediction of a regulatory network between *p* genes into *p* different regression problems. In each of the regression problems, the expression pattern of one of the genes (target gene) is predicted from the expression patterns of all the other genes (input genes), using tree-based ensemble methods Random Forests or Extra-Trees. The importance of an input gene in the prediction of the target gene expression pattern is taken as an indication of a putative regulatory link. Putative regulatory links are then aggregated over all genes to provide a ranking of interactions from which the whole network is reconstructed. In addition to performing well on the DREAM4 *In Silico Multifactorial* challenge simulated data, we show that GENIE3 compares favorably with existing algorithms to decipher the genetic regulatory network of *Escherichia coli*. It doesn't make any assumption about the nature of gene regulation, can deal with combinatorial and non-linear interactions, produces directed GRNs, and is fast and scalable. In conclusion, we propose a new algorithm for GRN inference that performs well on both synthetic and real gene expression data. The algorithm, based on feature selection with tree-based ensemble methods, is simple and generic, making it adaptable to other types of genomic data and interactions.

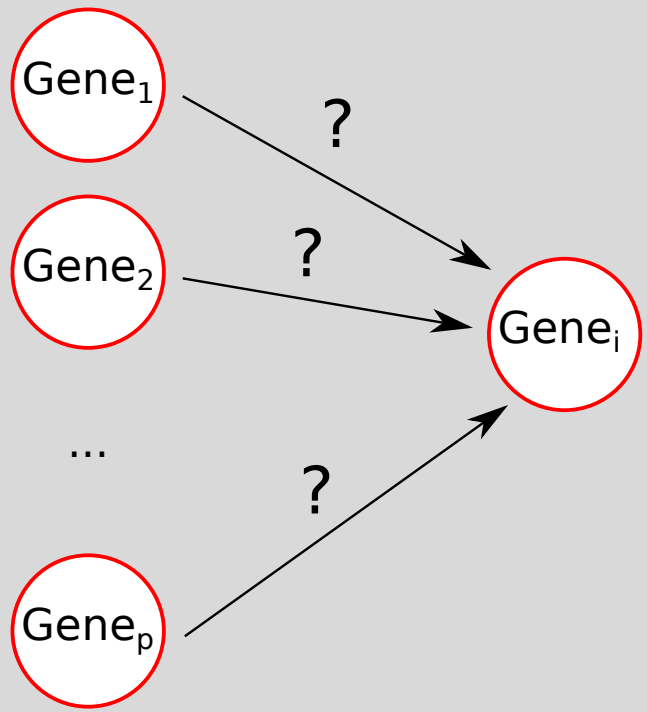
Inference is decomposed into p feature selection problems



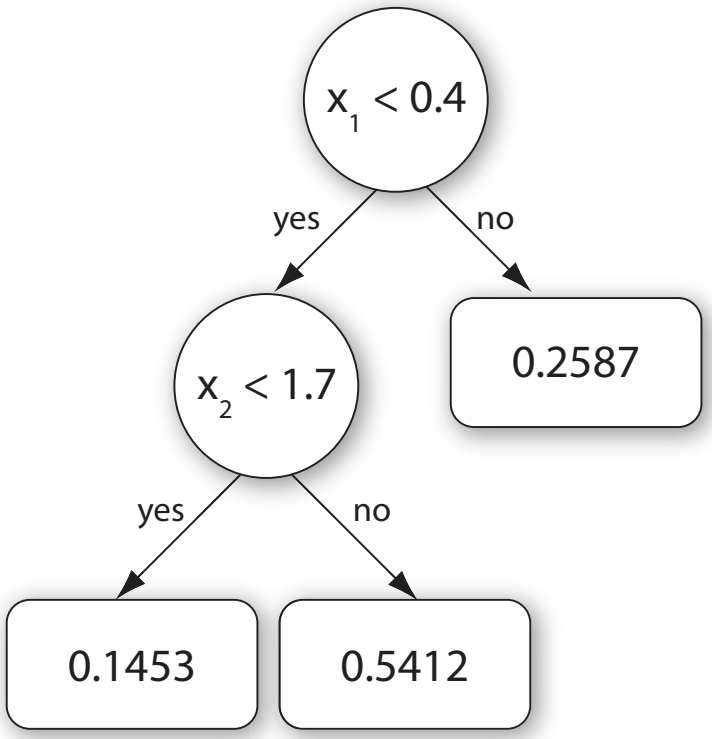
Feature selection problem i

$=$

Identify regulators of gene i



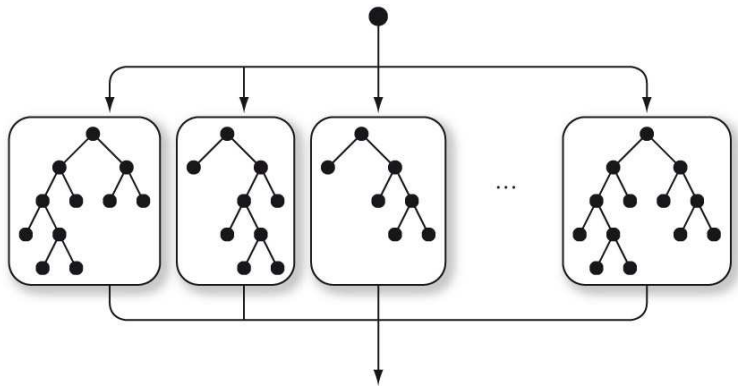
Decision trees are used to predict the gene expressions



Each interior node tests
the expression of one gene.

Each leaf predicts
the expression of the target gene.

Tree-based ensemble methods are good candidates



Bagging
Random Forests
Extra-Trees
...

Can deal with interacting features

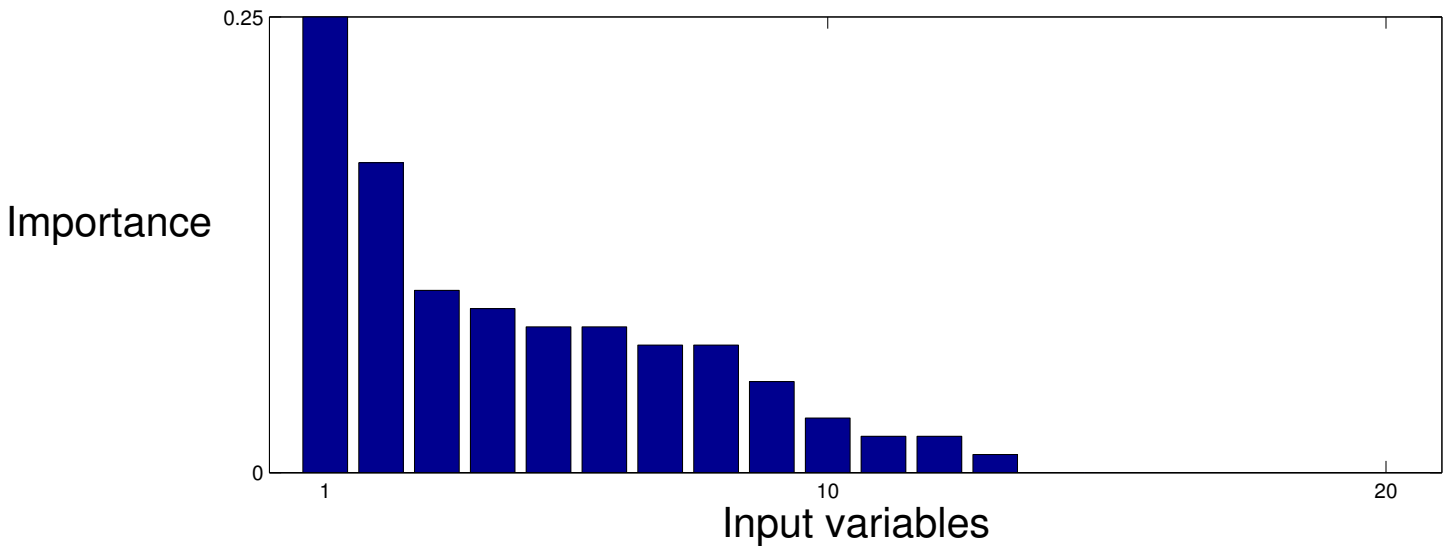
Non-parametric

Work well with high-dimensional datasets

Scalable: $O(KN \log N)$
N: number of samples
K: number of randomly selected features at each tree node

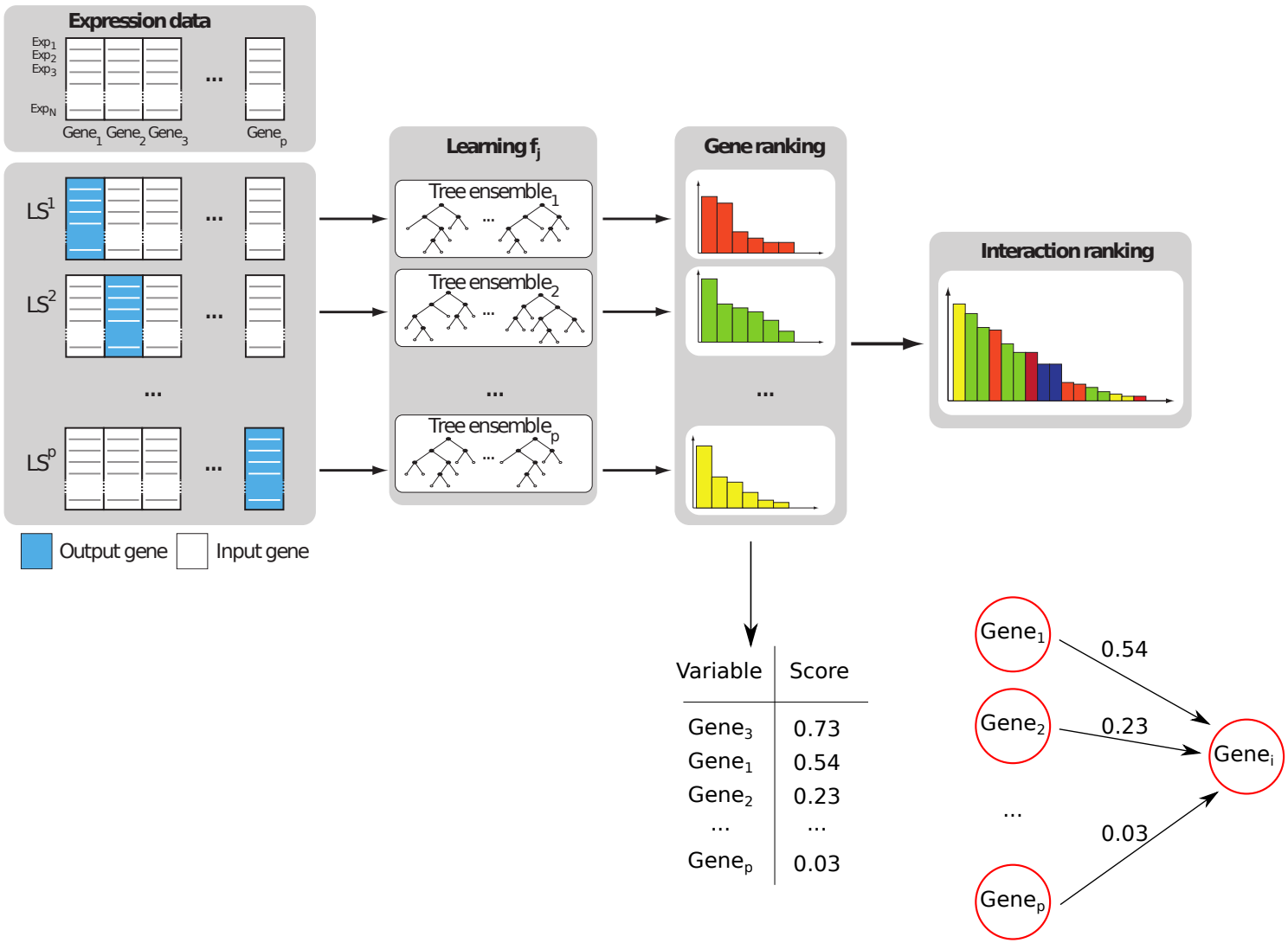
The tree-based model is informative

Feature importance scores (MDI or MDA) can be computed from the learned model.

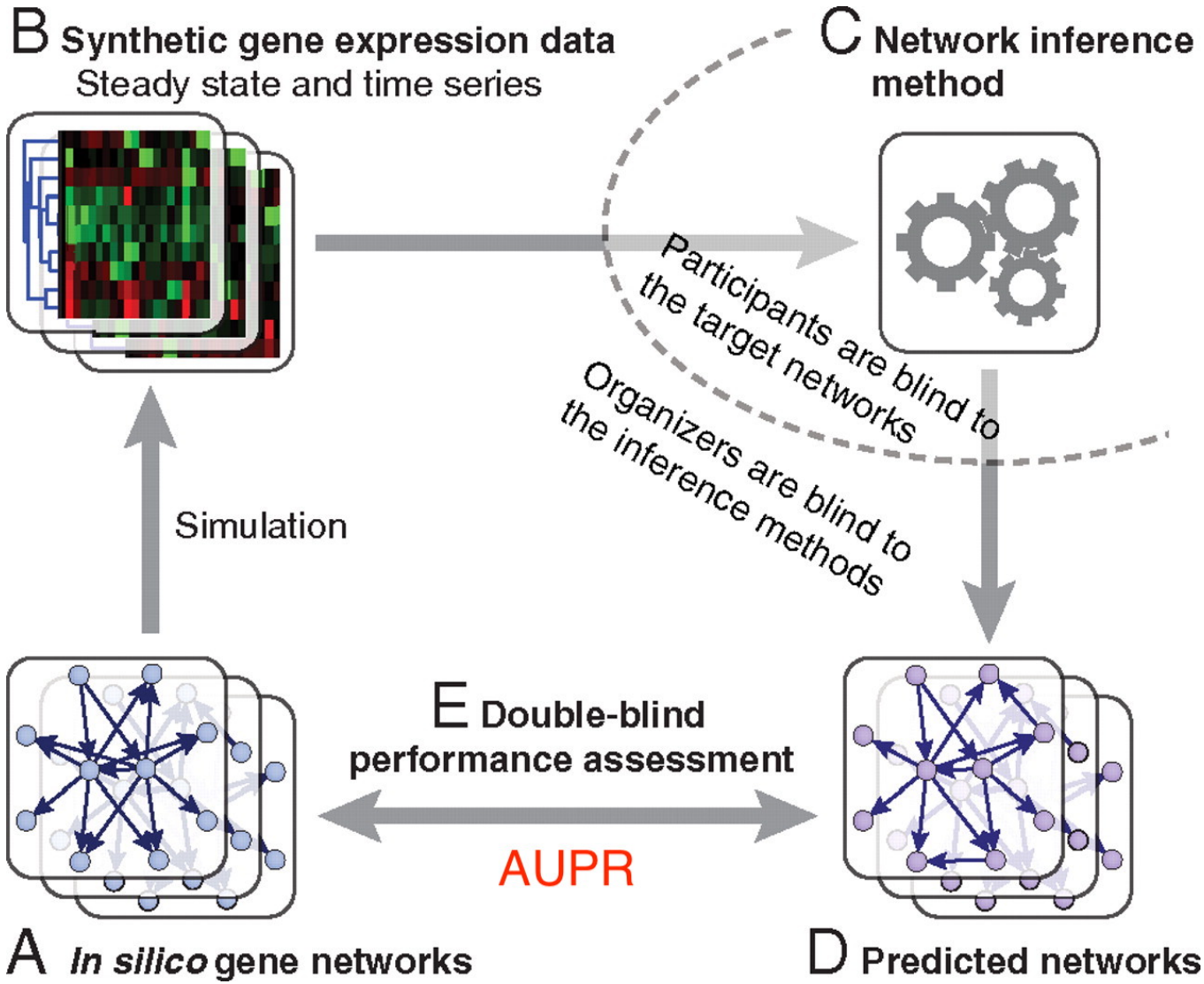


GENIE3 framework

p ensembles of trees are learned and feature importance scores are computed from each ensemble.

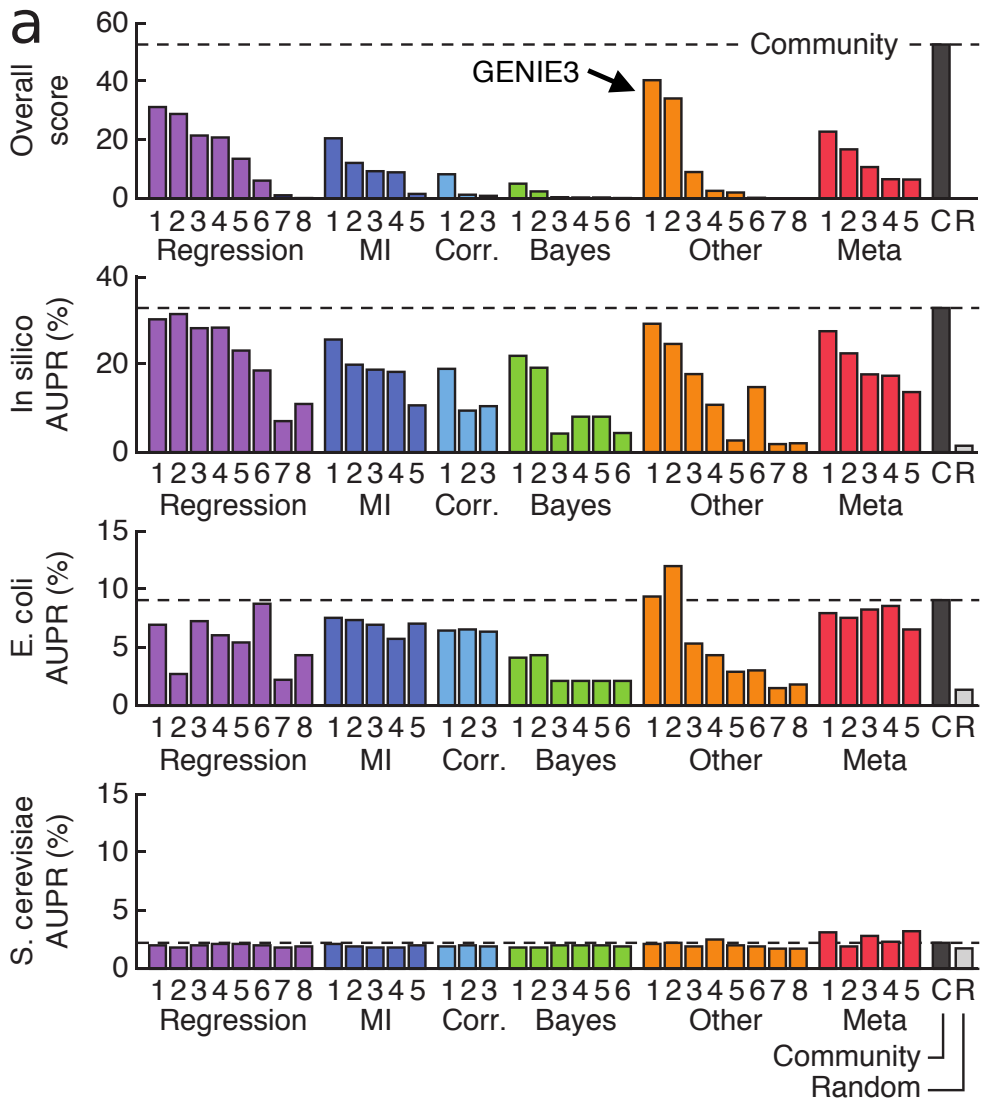


DREAM is an annual reverse engineering competition



(Marbach *et al.*, 2010)

GENIE3 was best performer in DREAM4 and DREAM5



DREAM5 results

1643 genes

805 samples

4511 genes

805 samples

5950 genes

536 samples

(Marbach *et al.*, 2012)

Advantages and limitations of GENIE3

Advantages

- Non-parametric, so very flexible
- Easy to use, only one main parameter (K)
- Good scalability compared to **some** families of methods (e.g. ODE-based methods)

Limitations:

- Too slow when applied to large datasets ($\sim$ 100K samples)
- Returns a ranking of edges, not a network
- **Lack of interpretability:** does not provide or exploit any information about the dynamics of the system

Jump3: a hybrid model-free/model-based approach for time series expression data

Combining tree-based and dynamical systems for the inference of gene regulatory networks

Vân Anh Huynh-Thu[1,*] and Guido Sanguinetti[1,2,*]

[1]School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, and [2]SynthSys - Systems and Synthetic Biology, University of Edinburgh, Edinburgh EH9 3JD, UK

*To whom correspondence should be addressed.

Associate Editor: Igor Jurisica

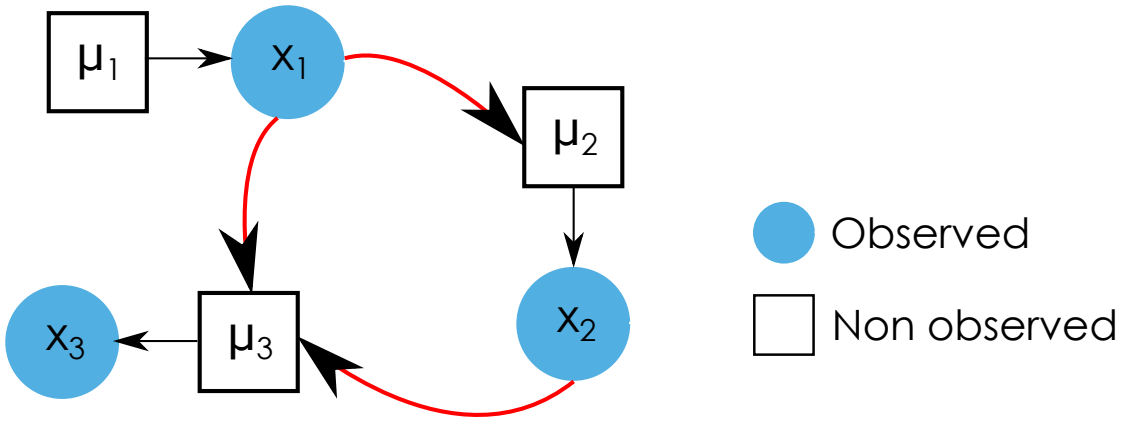
Received on October 18, 2014; revised on December 5, 2014; accepted on December 23, 2014

Abstract

Motivation: Reconstructing the topology of gene regulatory networks (GRNs) from time series of gene expression data remains an important open problem in computational systems biology. Existing GRN inference algorithms face one of two limitations: model-free methods are scalable but suffer from a lack of interpretability and cannot in general be used for out of sample predictions. On the other hand, model-based methods focus on identifying a dynamical model of the system. These are clearly interpretable and can be used for predictions; however, they rely on strong assumptions and are typically very demanding computationally.

Results: Here, we propose a new hybrid approach for GRN inference, called Jump3, exploiting time series of expression data. Jump3 is based on a formal on/off model of gene expression but uses a non-parametric procedure based on decision trees (called 'jump trees') to reconstruct the GRN topology, allowing the inference of networks of hundreds of genes. We show the good performance of Jump3 on *in silico* and synthetic networks and applied the approach to identify regulatory interactions activated in the presence of interferon gamma.

Gene regulation system representation



The system is represented by: $(i = 1, \ldots, p)$
- x_i: expression of gene i (observed at several time points)
- μ_i: state of promoter of gene i (unobserved)

Hybrid model:
- Links $\mu_i \rightarrow x_i$ are represented by a formal mathematical model.
- Links $x_i \rightarrow \mu_{j \neq i}$ are represented by a non-parametric tree model.

$\mu_i \rightarrow x_i$ is modelled with a stochastic differential equation

On/off model:

$$\mathrm{d}x_i = (A_i \mu_i(t) + b_i - \lambda_i x_i)\mathrm{d}t + \sigma \mathrm{d}w(t)$$

$\mu_i(t)$: binary (0/1)
A_i, b_i, λ_i: kinetic parameters
$\sigma \mathrm{d}w(t)$: noise process with variance σ^2

x_i is observed with Gaussian noise at N time points:

$$\begin{aligned} \hat{x}_i(t_k) &= x_i(t_k) + \epsilon \\ \epsilon &\sim \mathcal{N}(0, \sigma^2_{obs}), k = 1, \ldots, N \end{aligned}$$

Given μ_i, the expression x_i is a Gaussian process

SDE:

$$\mathrm{d}x_i = (A_i\mu_i(t) + b_i - \lambda_i x_i)\mathrm{d}t + \sigma \mathrm{d}w(t)$$

Solution:

$$x_i(t) = x_i(0)e^{-\lambda_i t} + A_i \int_0^t e^{-\lambda_i(t-\tau)}\mu_i(\tau)d\tau + \frac{b_i}{\lambda_i}(1-e^{-\lambda_i t}) + \sigma \int_0^t e^{-\lambda_i(t-\tau)}dw(\tau)$$

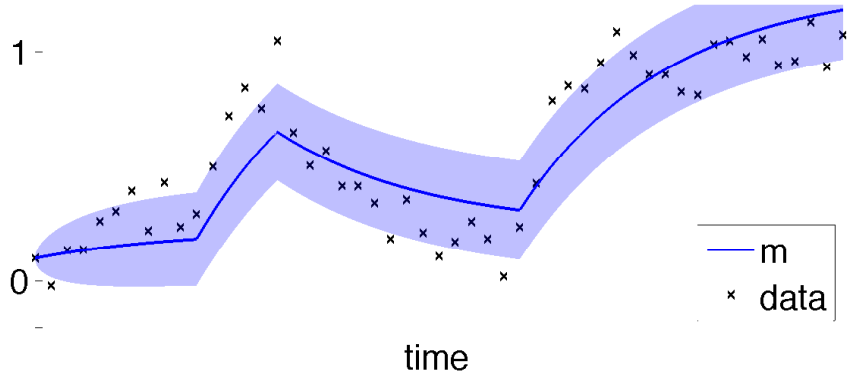
Mean:

$$m_i(t) = x_i(0)e^{-\lambda_i t} + A_i \int_0^t e^{-\lambda_i(t-\tau)}\mu_i(\tau)d\tau + \frac{b_i}{\lambda_i}(1 - e^{-\lambda_i t})$$

Covariance:

$$K_i(t, t') = \frac{\sigma^2}{2\lambda_i}(e^{-\lambda_i|t-t'|} - e^{-\lambda_i(t+t')})$$

Given μ_i, the expression x_i is a Gaussian process



- x_i is completely described by its mean m_i and covariance K_i

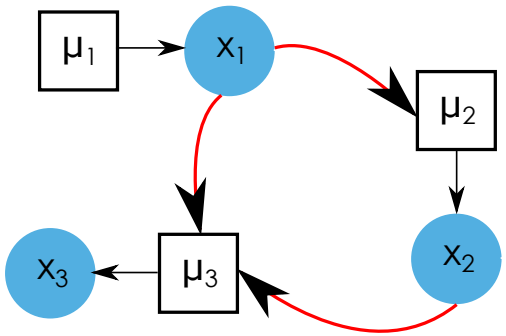
- For every finite set of time points:

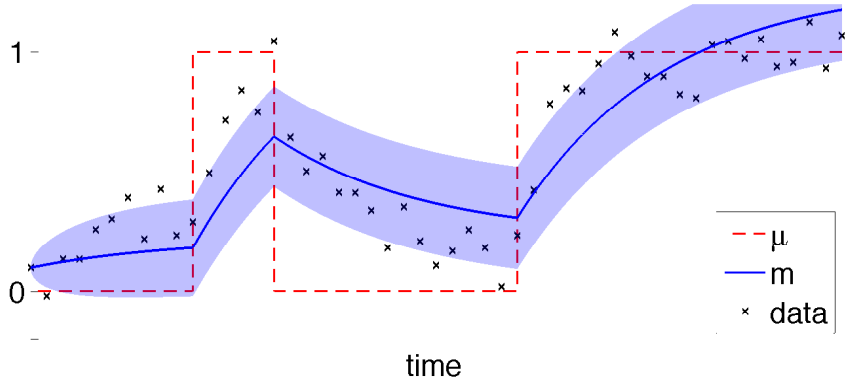
$$\begin{aligned} \mathbf{x}_i &\sim \mathcal{N}(\mathbf{m}_i, K_i) \\ \hat{\mathbf{x}}_i &\sim \mathcal{N}(\mathbf{m}_i, K_i + \sigma^2_{obs} I) \end{aligned}$$

- We can compute the likelihood:

$$\begin{aligned} \log p(\hat{\mathbf{x}}_i) = &-\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|K_i + \sigma^2_{obs} I| \\ &- \frac{1}{2}(\hat{\mathbf{x}}_i - \mathbf{m}_i)^\top (K_i + \sigma^2_{obs} I)^{-1}(\hat{\mathbf{x}}_i - \mathbf{m}_i) \end{aligned}$$

The likelihood depends on the promoter state μ





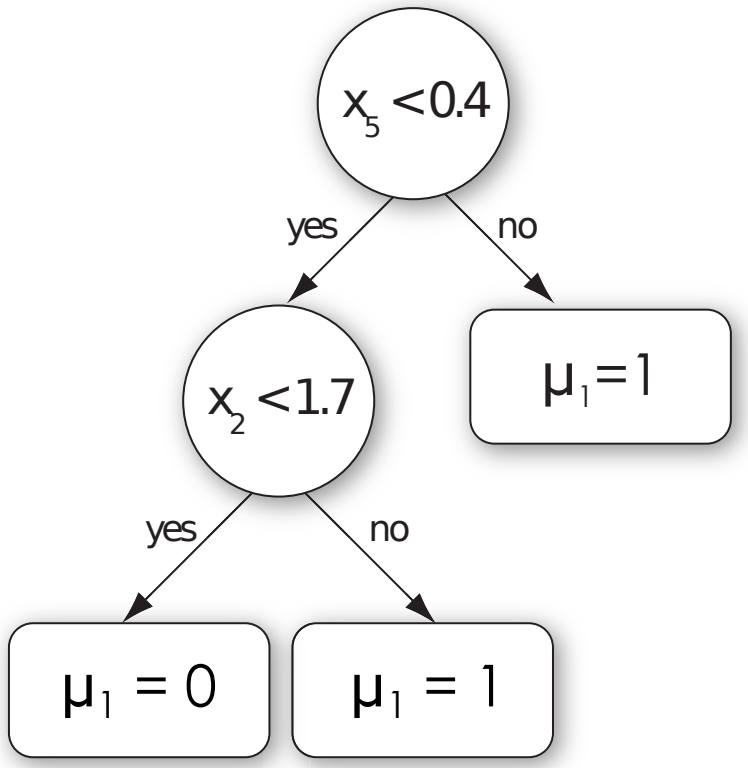
Model: $\mathrm{d}x_i = (A_i\mu_i(t) + b_i - \lambda_i x_i)\mathrm{d}t + \sigma \mathrm{d}w(t)$

Likelihood:
$\log p(\hat{\mathbf{x}}_i) = -\frac{1}{2}(\hat{\mathbf{x}}_i - \mathbf{m}_i)^\top (K_i + \sigma^2_{obs} I)^{-1}(\hat{\mathbf{x}}_i - \mathbf{m}_i) + c_i$

Goals (for each gene i):

1. Find the trajectory μ_i that maximises the likelihood
2. Find the genes that influence μ_i (network reconstruction)

$x_i \rightarrow \mu_j$ is modelled with a decision tree



Each interior node tests the expression of one gene.

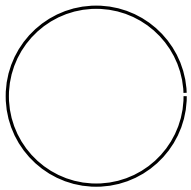
Each leaf predicts the state of the promoter of the target gene.

Jump tree = decision tree with a latent output variable

States are not observed.
$\rightarrow$ We can not use standard decision trees.
$\rightarrow$ New type of decision tree called **jump tree**

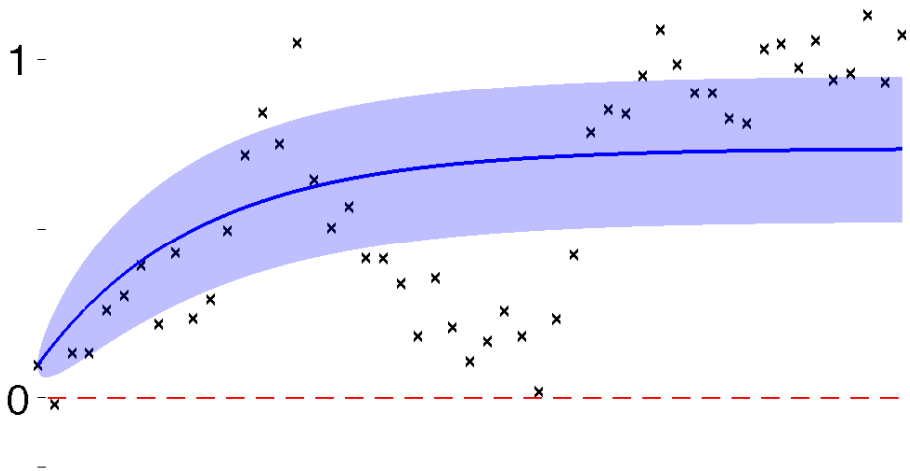
	CART	Jump tree
Inputs	Observed	Observed
Output	Observed	**Non-observed**
Induction	Top-down	Top-down
Algorithm	Greedy	Greedy
Best split	Max entropy reduction	Max **likelihood increase**

Jump trees are learned through maximisation of the likelihood

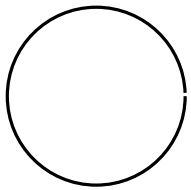


Start with $\mu_1(t) = 0, \forall t$

$$\mathcal{L} = -2.56$$

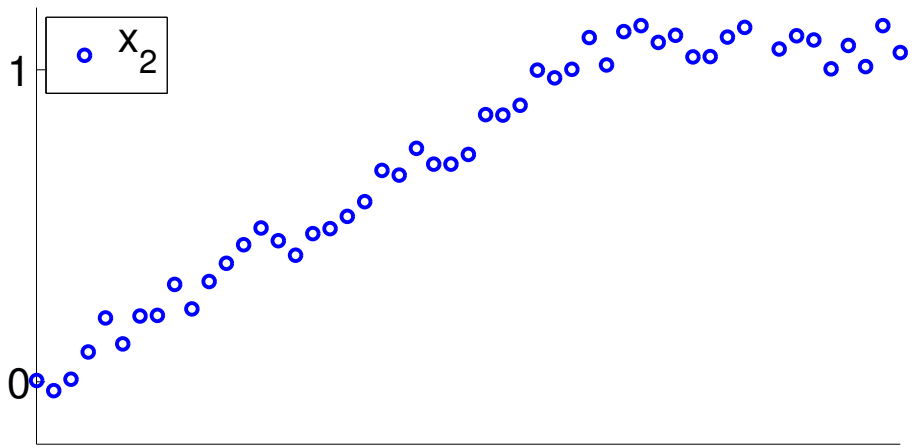


Jump trees are learned through maximisation of the likelihood

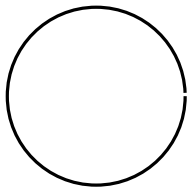


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

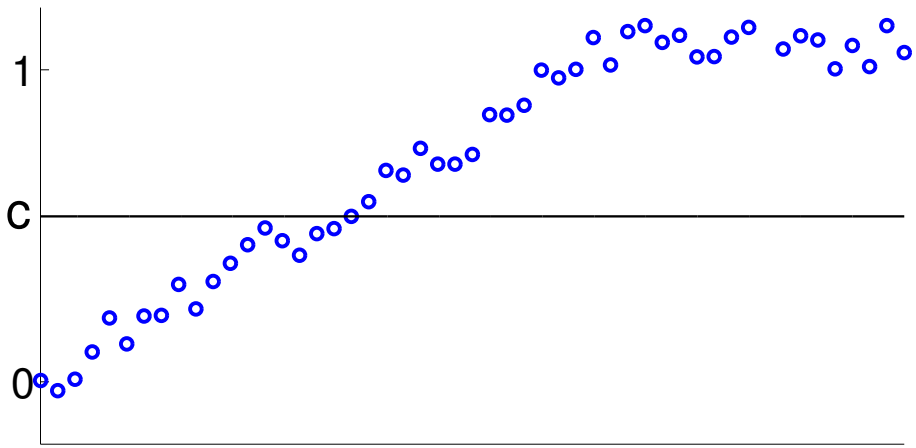


Jump trees are learned through maximisation of the likelihood

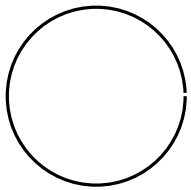


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

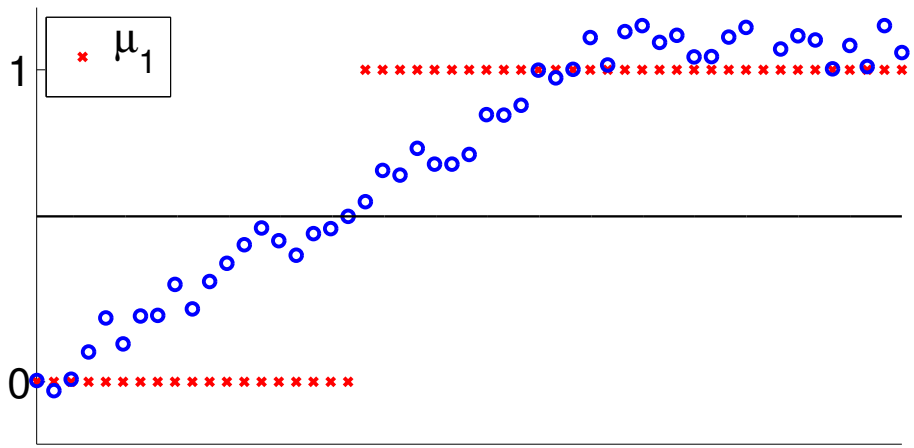


Jump trees are learned through maximisation of the likelihood

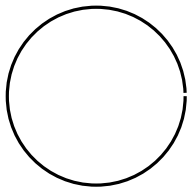


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

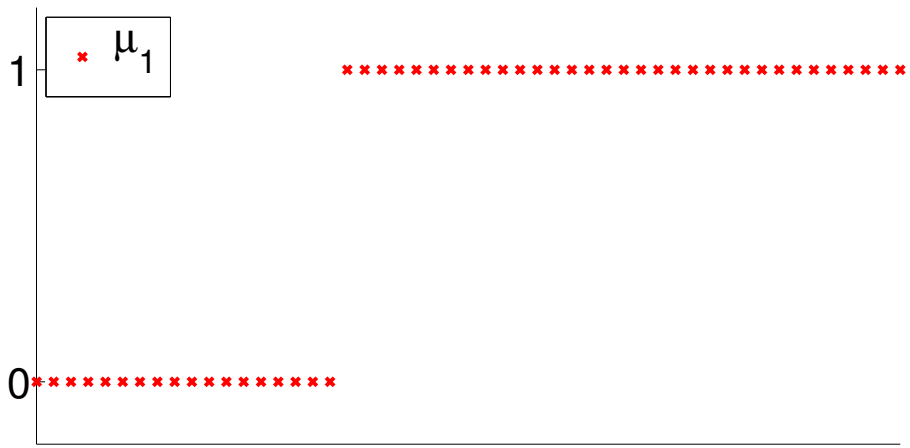


Jump trees are learned through maximisation of the likelihood

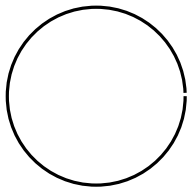


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

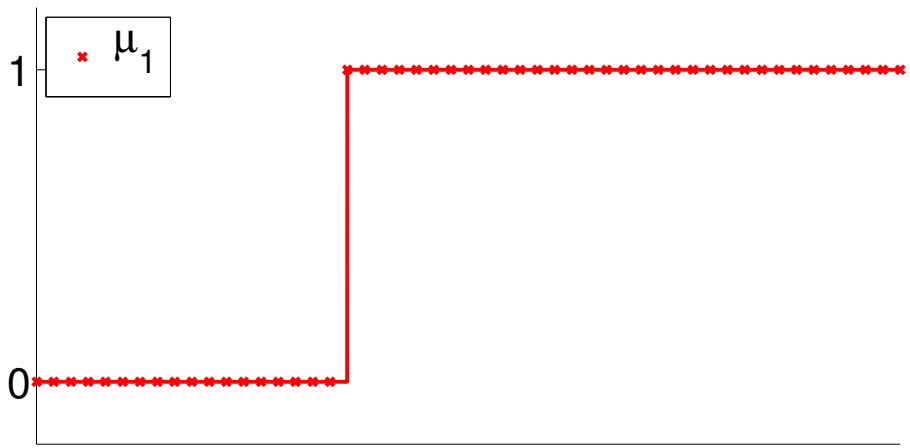


Jump trees are learned through maximisation of the likelihood

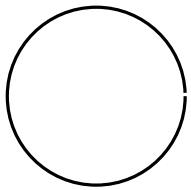


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

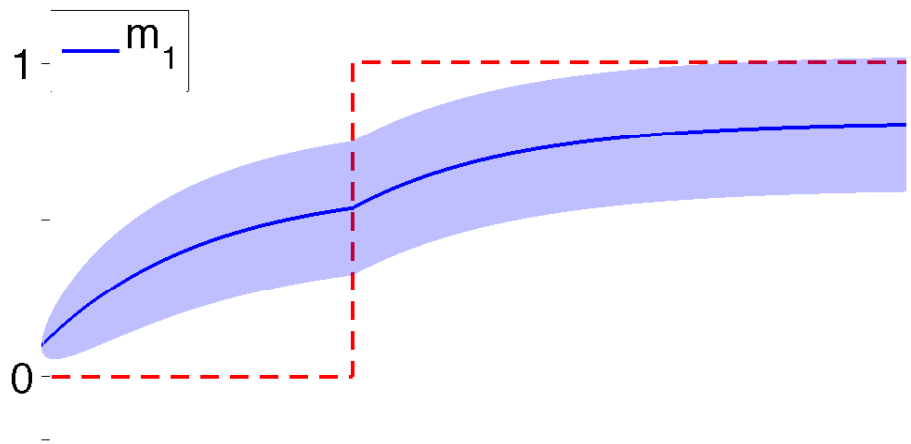


Jump trees are learned through maximisation of the likelihood

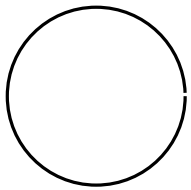


Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$



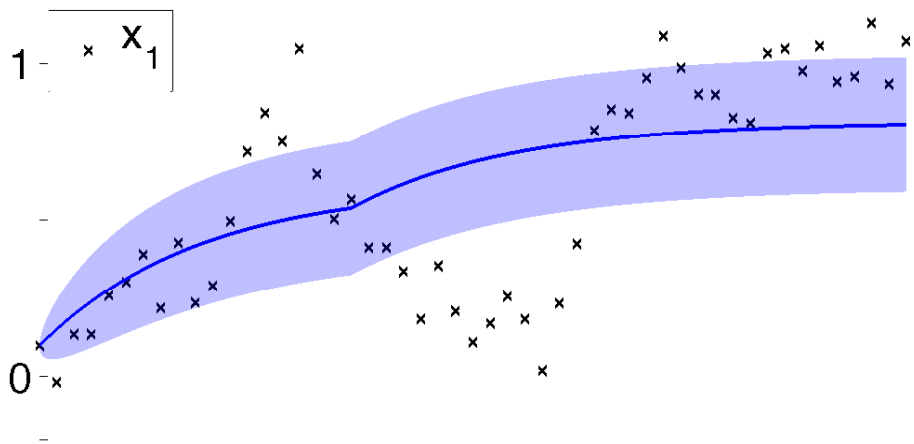
Jump trees are learned through maximisation of the likelihood



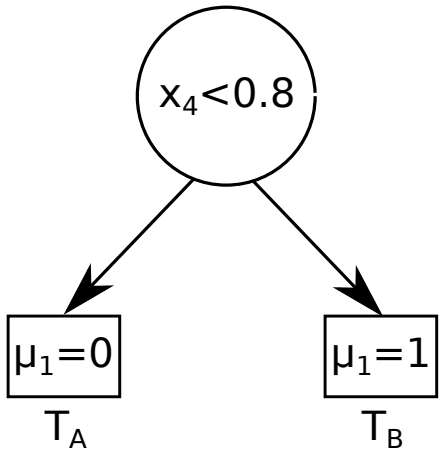
Candidate split:

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_2(t) < c \\ 1, & \text{if } \hat{x}_2(t) \geq c \end{cases}$$

$$\mathcal{L} = -2.35$$

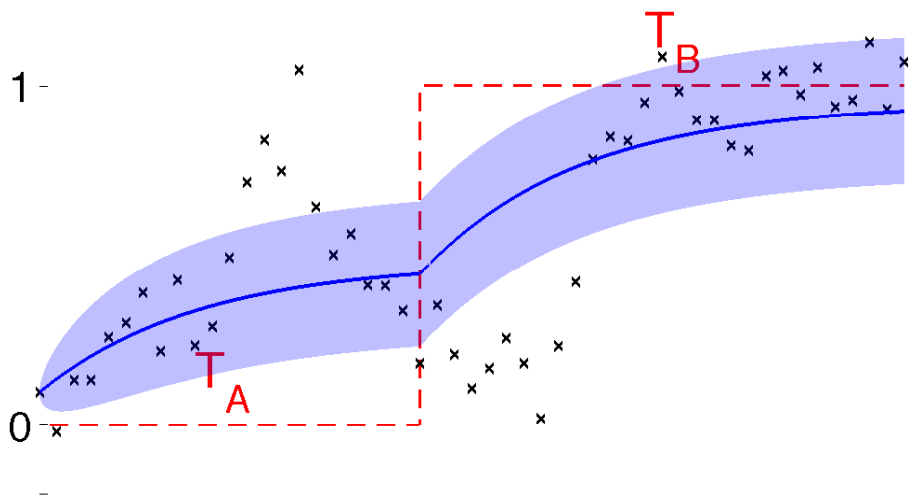


Jump trees are learned through maximisation of the likelihood

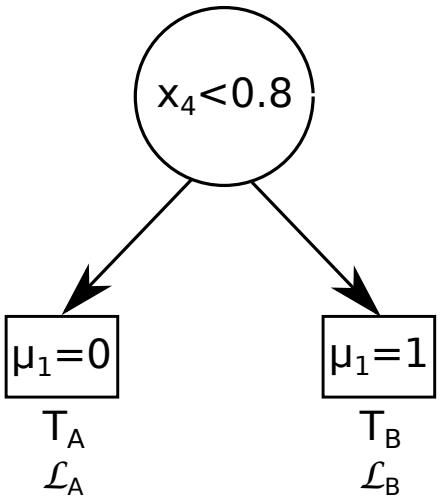


Select the split with the highest likelihood

$$\mathcal{L} = -1.39$$



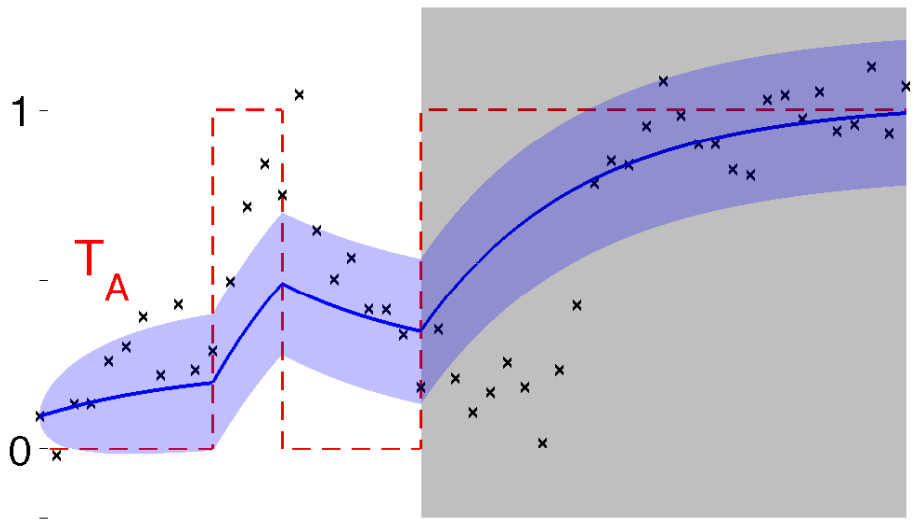
Jump trees are learned through maximisation of the likelihood



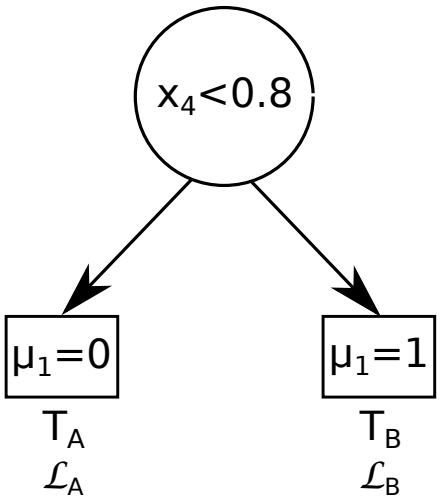
Candidate split: $\forall t \in T_A$

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_i(t) < c \\ 1, & \text{if } \hat{x}_i(t) \geq c \end{cases}$$

$$\mathcal{L}_A = -0.20$$



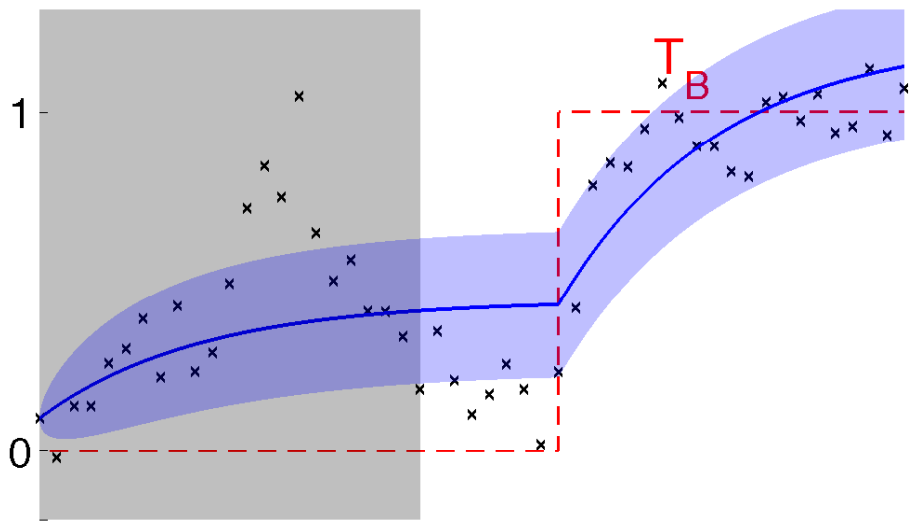
Jump trees are learned through maximisation of the likelihood



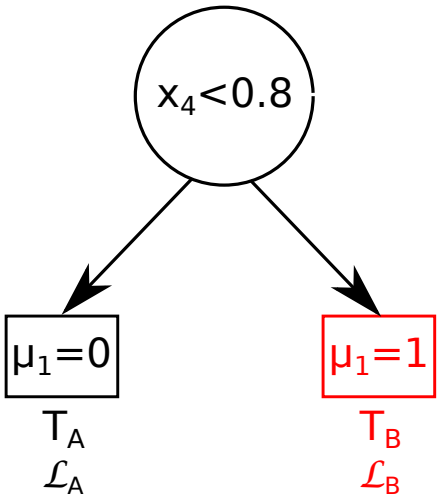
Candidate split: $\forall t \in T_B$

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_i(t) < c \\ 1, & \text{if } \hat{x}_i(t) \geq c \end{cases}$$

$$\mathcal{L}_B = 0.47$$

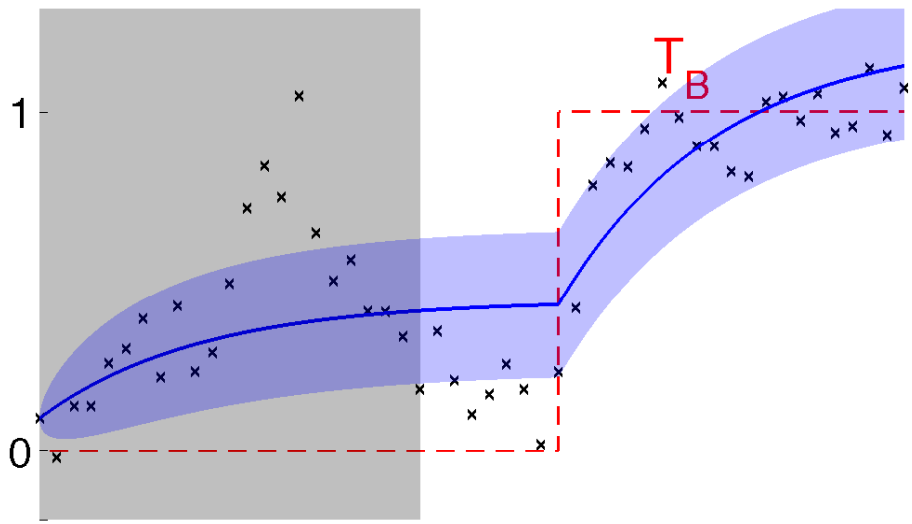


Jump trees are learned through maximisation of the likelihood

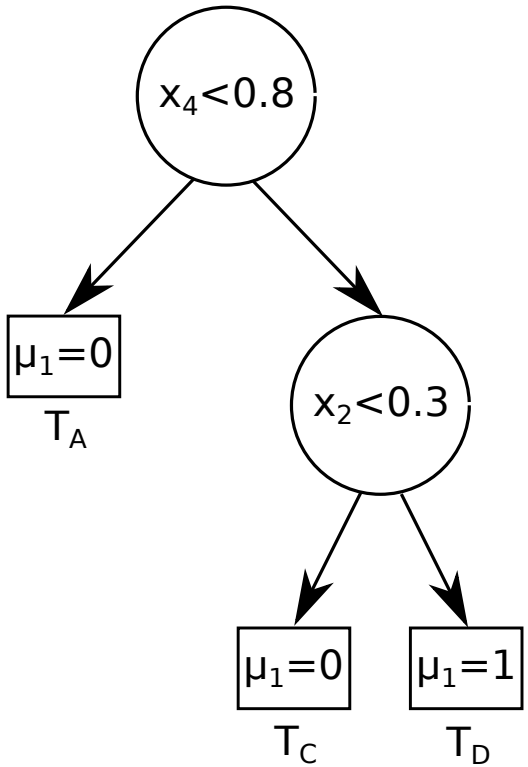


Select the leaf with the highest likelihood

$$\mathcal{L}_B = 0.47$$

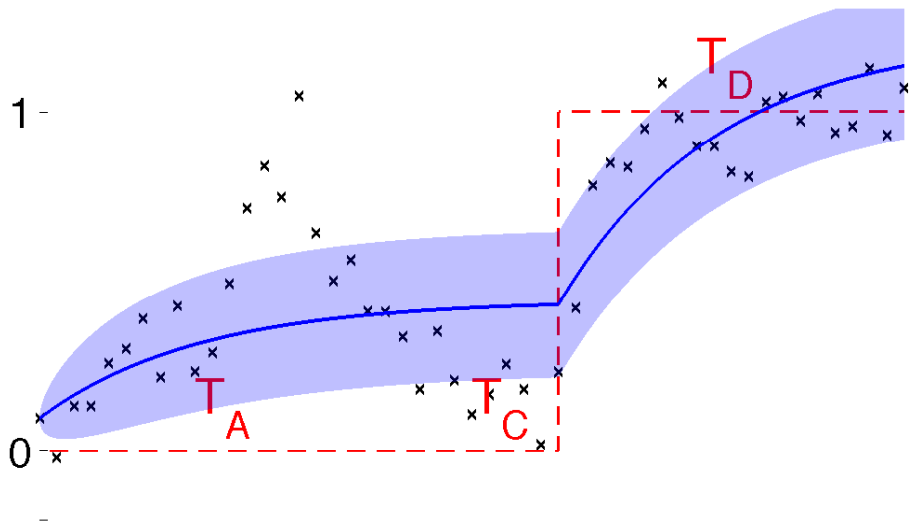


Jump trees are learned through maximisation of the likelihood

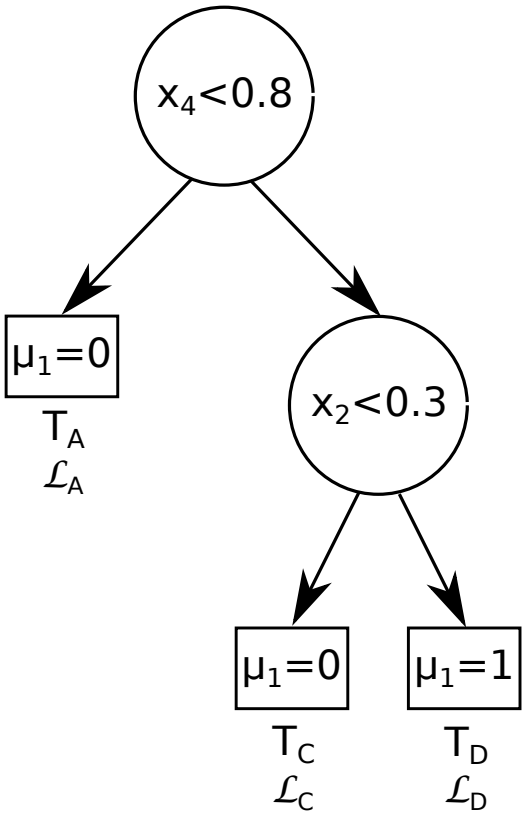


Replace the leaf with a splitting node

$$\mathcal{L} = 0.47$$

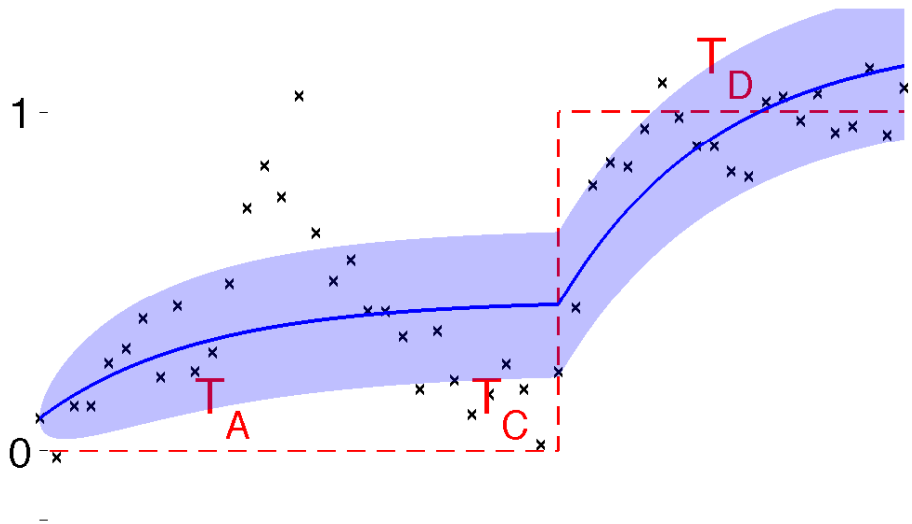


Jump trees are learned through maximisation of the likelihood

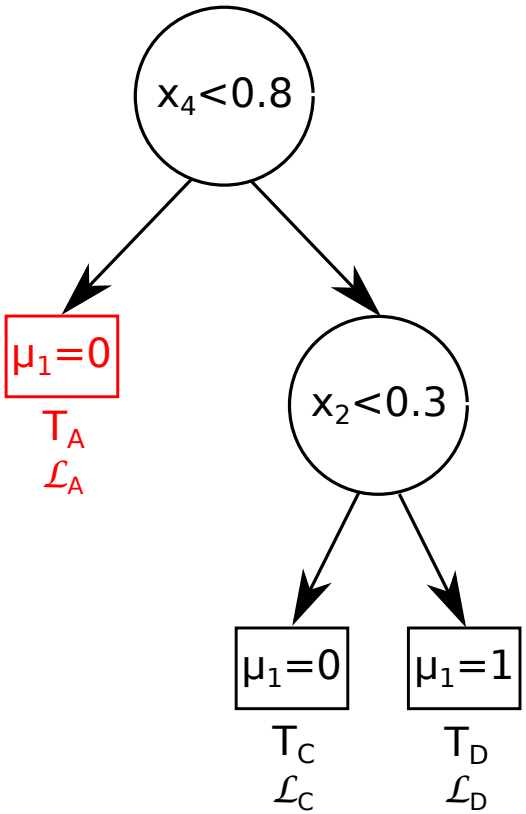


Repeat the procedure until the likelihood can not be increased

$$\mathcal{L} = 0.47$$

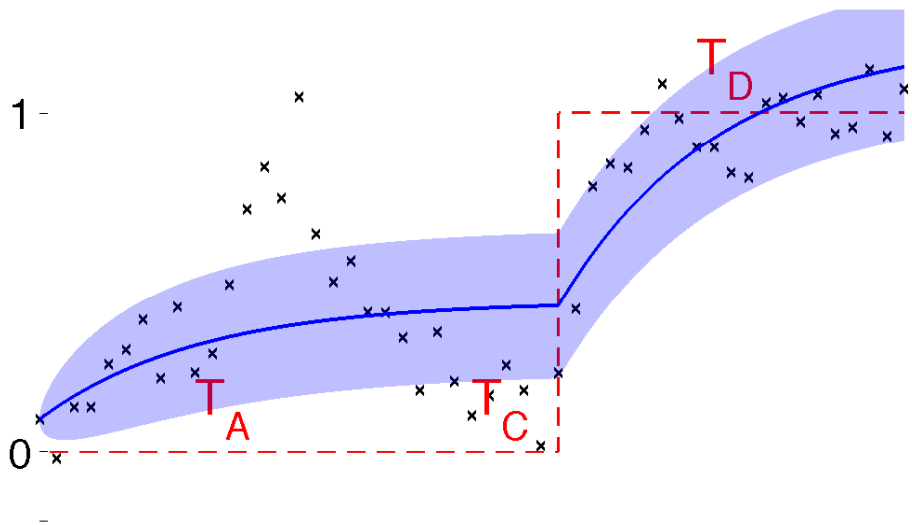


Jump trees are learned through maximisation of the likelihood

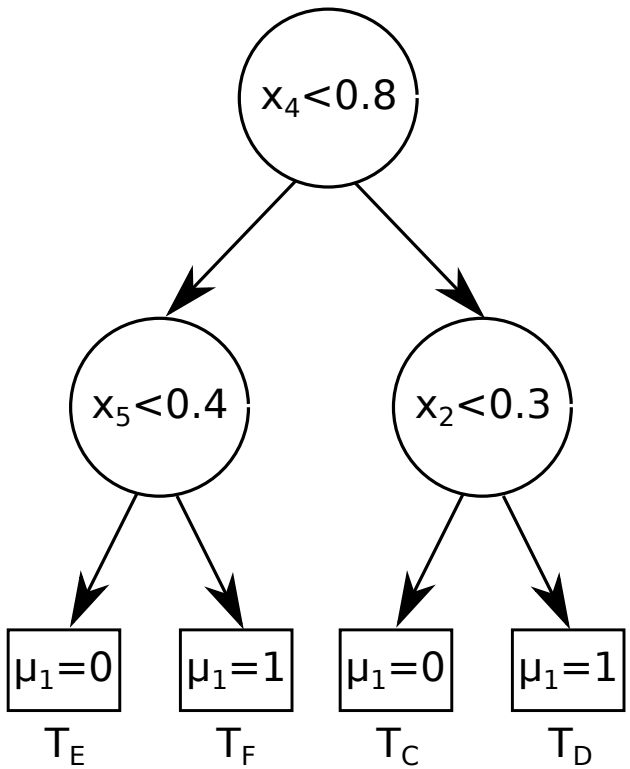


Repeat the procedure until the likelihood can not be increased

$$\mathcal{L} = 0.47$$

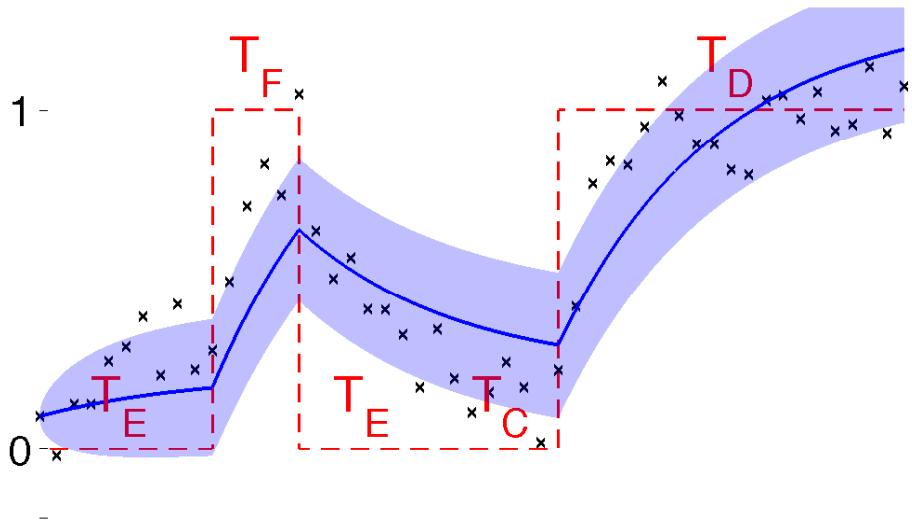


Jump trees are learned through maximisation of the likelihood

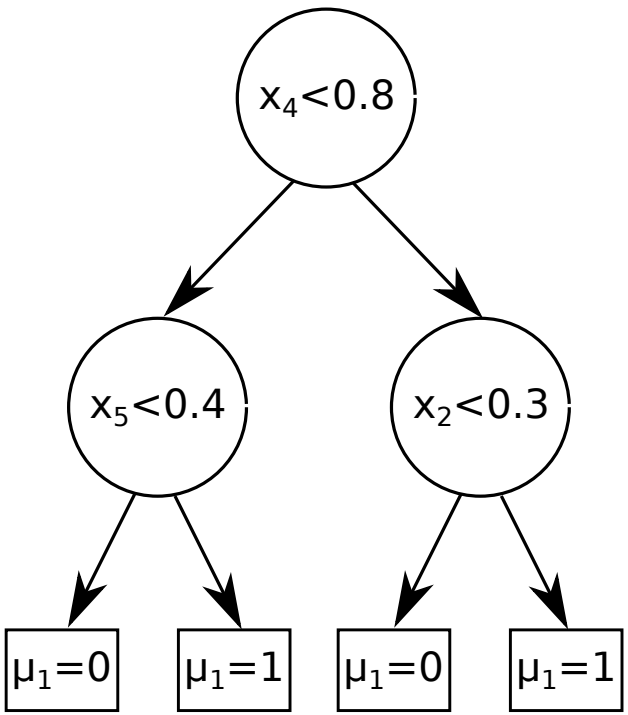


Repeat the procedure until the likelihood can not be increased

$$\mathcal{L} = 2.14$$

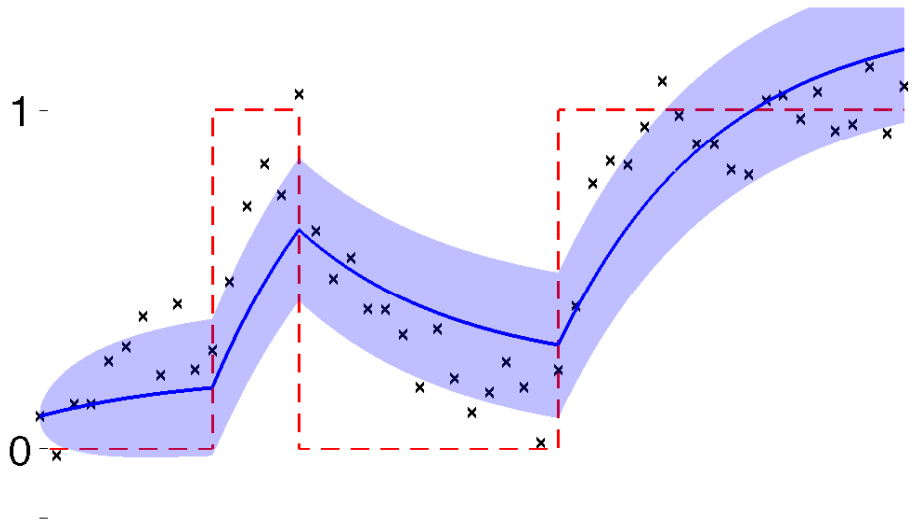


Jump trees are learned through maximisation of the likelihood

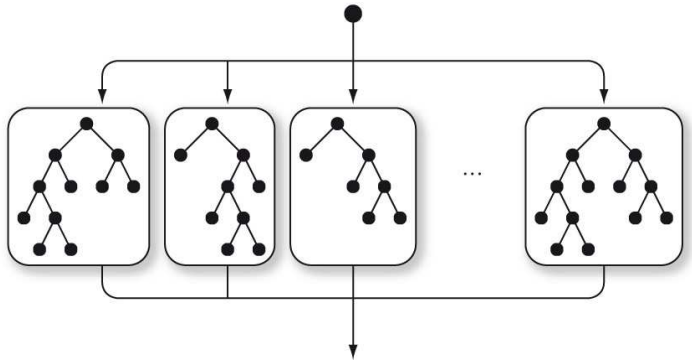


Repeat the procedure until the likelihood can not be increased

$$\mathcal{L} = 2.14$$



An ensemble of randomised trees can be constructed



Randomise x_i and c

$$\mu_1(t) = \begin{cases} 0, & \text{if } \hat{x}_i(t) < c \\ 1, & \text{if } \hat{x}_i(t) \geq c \end{cases}$$

Extra-Trees:

- At each node, the best split is chosen among K random splits.
- The prediction of $\mu(t)$ is averaged over the trees.

The variable importance is based on likelihood increase

At each tree node $\mathcal{N}$:

$$I(\mathcal{N}) = \mathcal{L}_{\text{after}} - \mathcal{L}_{\text{before}}$$

$\mathcal{L}_{\text{after}}$: likelihood after the split
$\mathcal{L}_{\text{before}}$: likelihood before the split

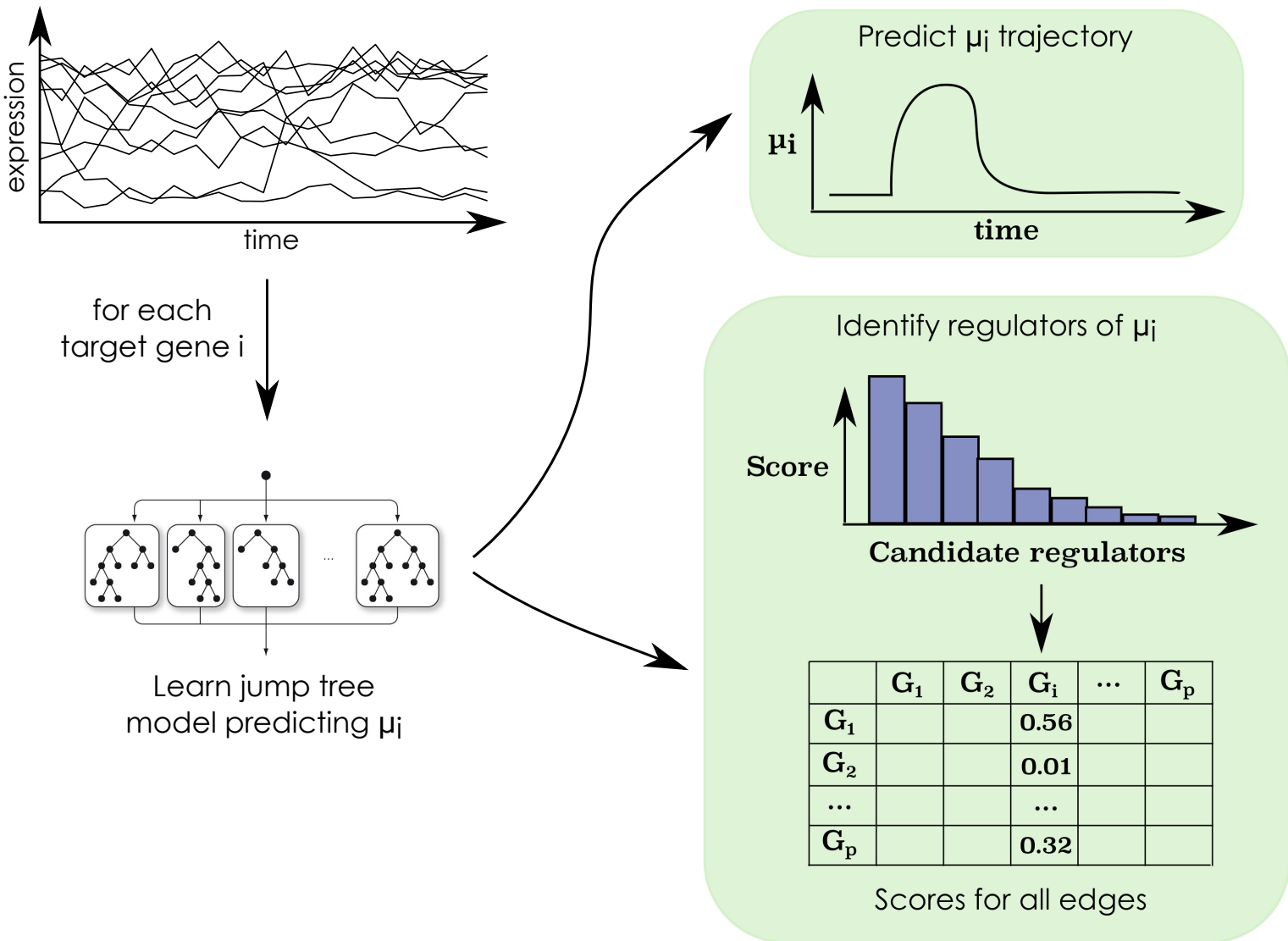
For a single tree:

w_i^t = sum of I at each node where regulator x_i appears

For an ensemble of trees:

$$w_i = \frac{1}{T} \sum_{t=1}^{T} w_i^t$$

Jump3 predicts the states and the network topology



	G_1	G_2	G_i	...	G_p
G_1			0.56		
G_2			0.01		
...			...		
G_p			0.32		

Computational complexity

Complexity of Jump3: $O(pKS^2N^2)$

p: number of genes
K: number of randomly selected variables at each node
S: number of splits
N: number of time points

Worst case: $K = p - 1$ and $S = N - 1 \rightarrow O(p^2N^4)$
But:

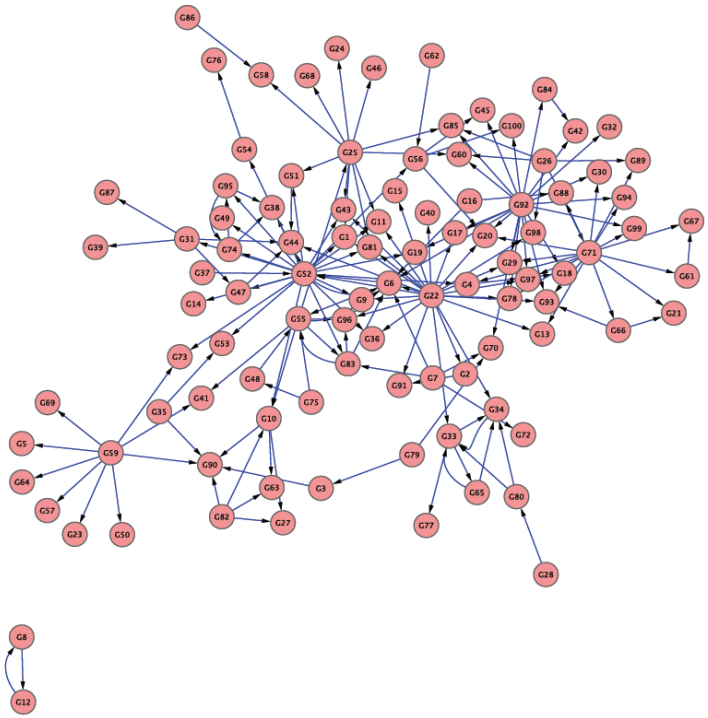
- Hopefully: $S << N$ in practice
- Jump3 can be parallelised over target genes and over trees

We validate our method using artificial networks

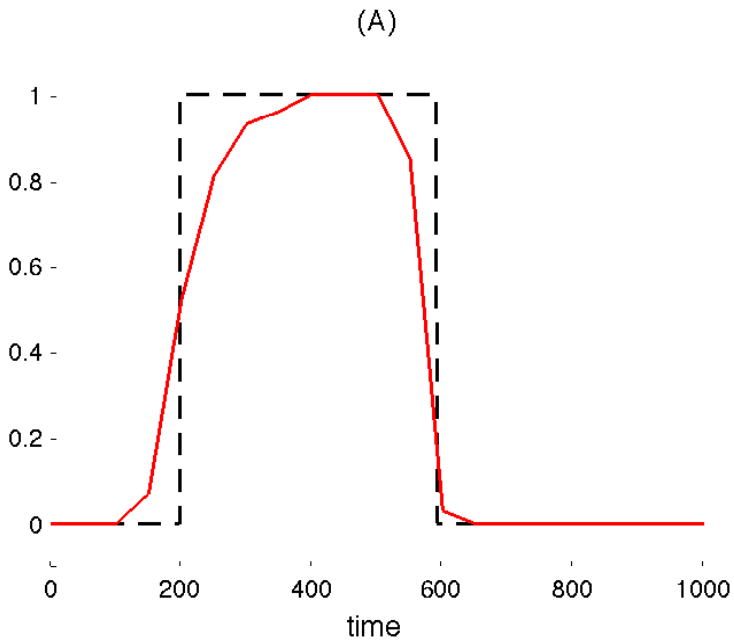
5 networks of 100 genes

Time series:
21 time points in each
perturbation experiment

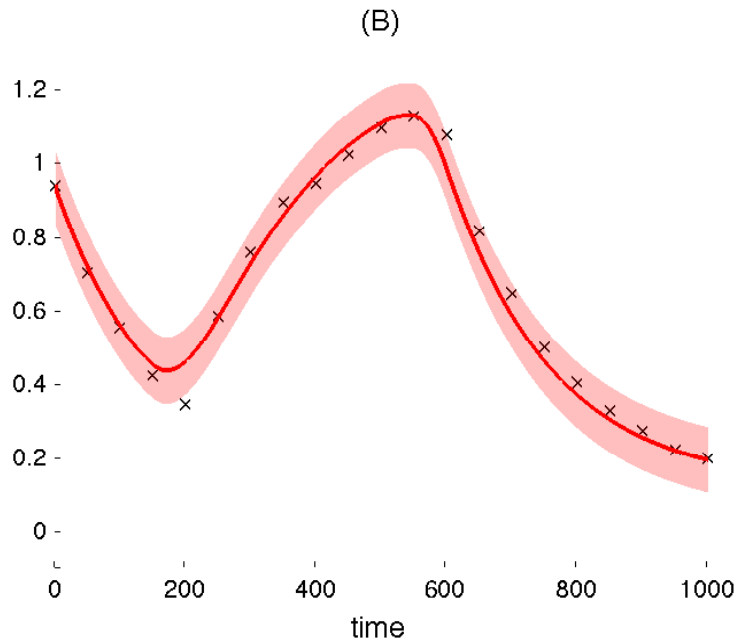
Simulation with:
- the same on/off model
- a different model (DREAM4)



Jump3 is able to correctly predict the states

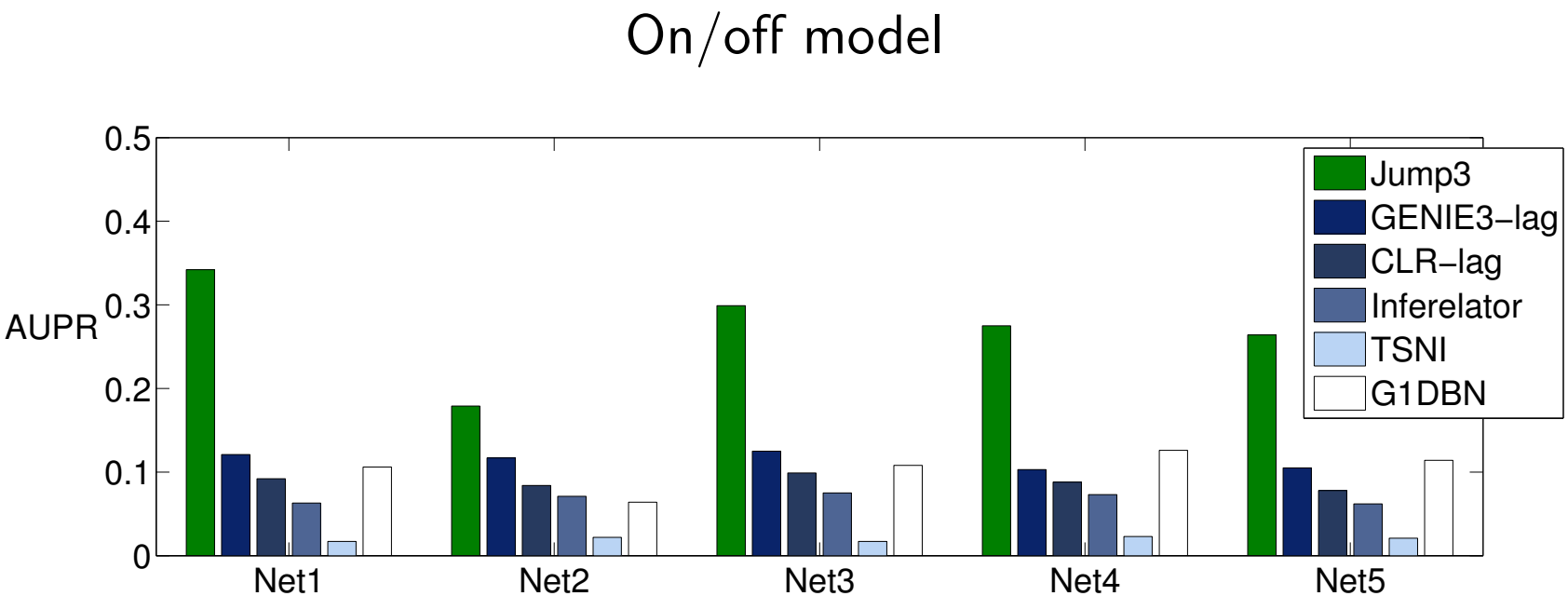


Promoter state prediction

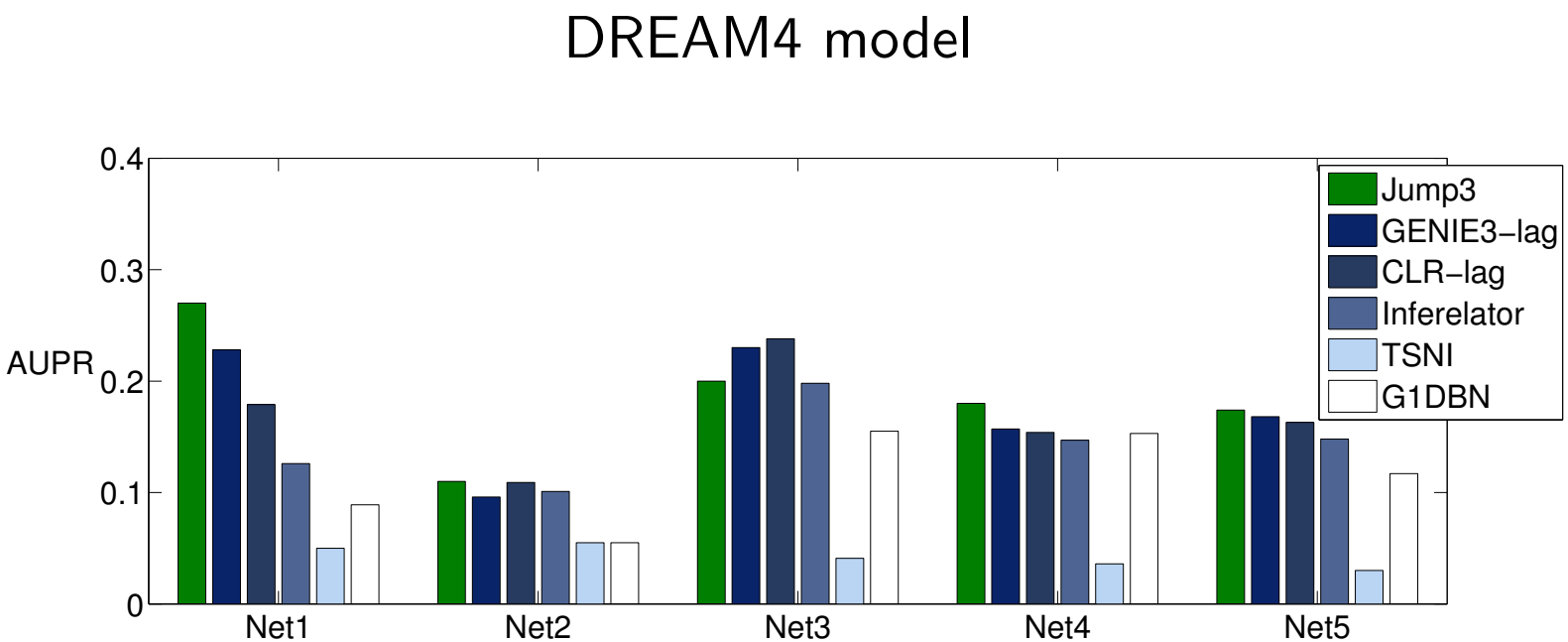


Gene expression fitting

Jump3 is competitive with existing methods



Jump3 is competitive with existing methods



We used Jump3 to infer the IFNγ network

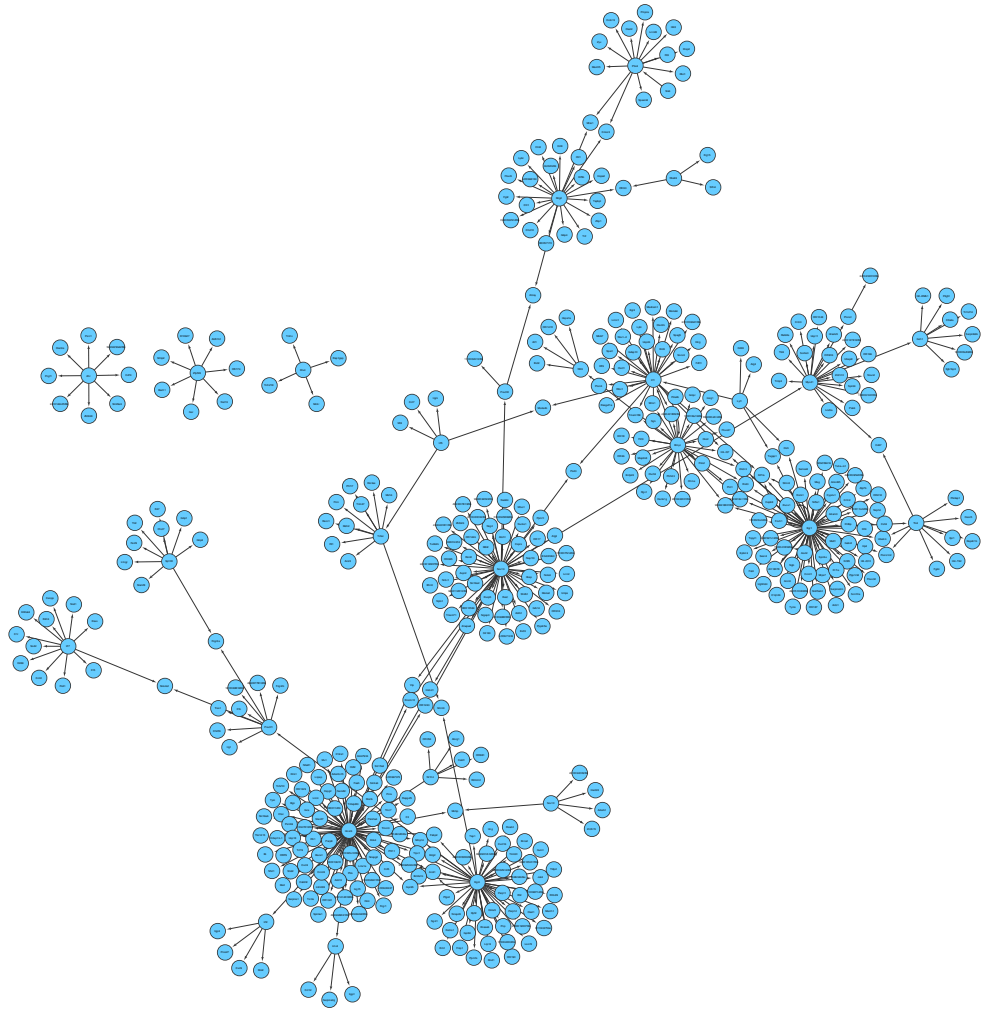
Macrophages and IFNγ: cells and protein that are involved in the immune response.

Macrophages were extracted from murine bone marrow and treated with IFNγ.

Gene expression levels were measured every 30 min over 12 hours.

We identified regulatory interactions between 40 known TFs and the 1000 most variable genes.

The predicted network is modular



Hubs TFs contain interferon genes, one gene associated with virus infection, and cancer-associated genes.

Summary

- Jump3 = semi-parametric model-based method for network inference and modelling
- Good performance on artificial data
- Can generate biologically meaningful hypotheses

Advantages

- Formal model of known parts of the system
- Flexible representation of unknown parts
- More interpretable than model-free methods
- More scalable than model-based methods (w.r.t. p)

Limitations

- Exploits only expression data
- Exploits only time series
- (Too?) simple dynamical model