Input Output HMMs for modeling network dynamics

Sushmita Roy

sroy@biostat.wisc.edu

Computational Network Biology
Biostatistics & Medical Informatics 826
https://compnetbiocourse.discovery.wisc.edu

Oct 17th, 18th 2018

The style of IOHMM is adapted from Prof. Craven's lectures on HMMs

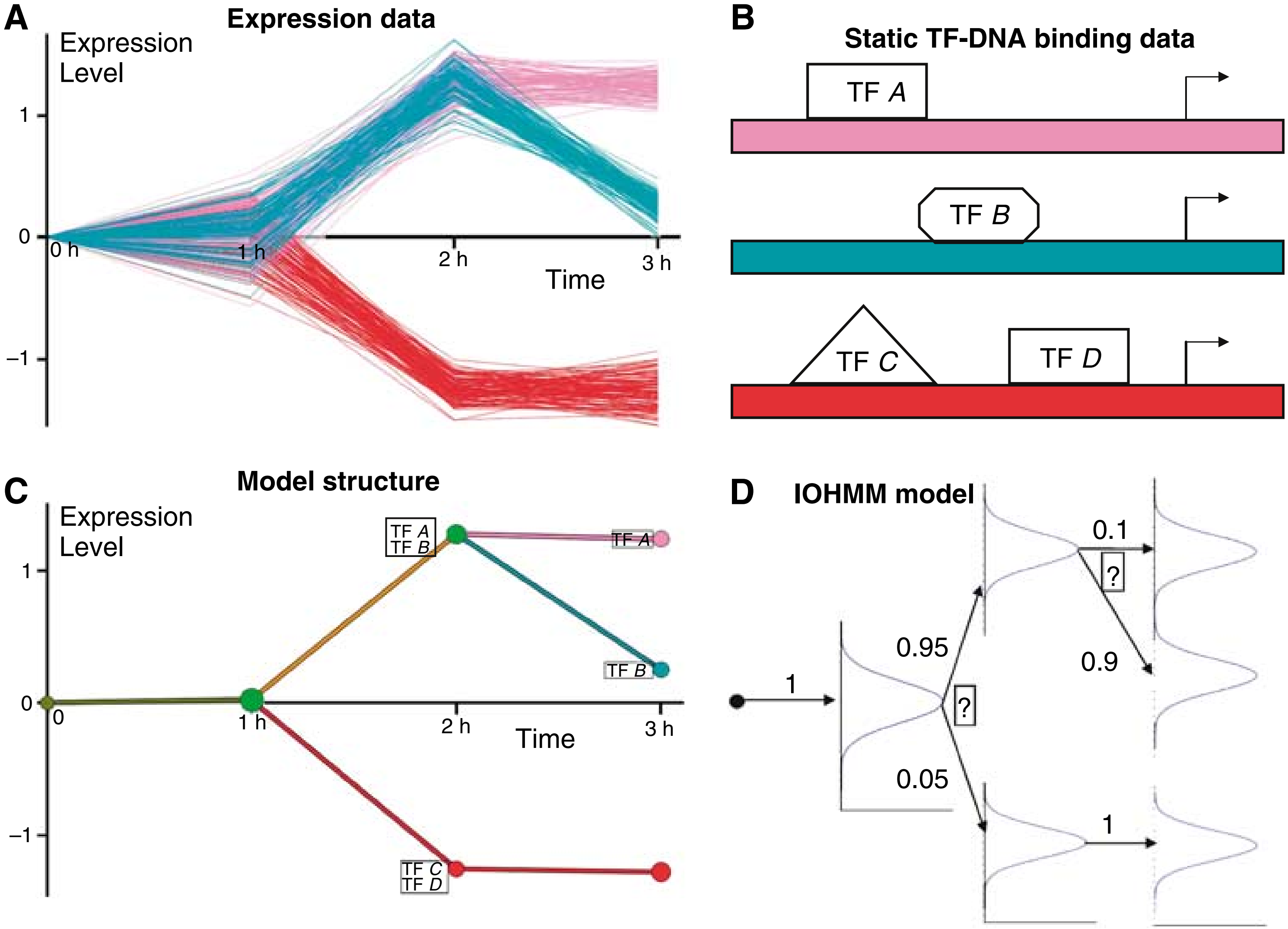
Goals for today

- What are Hidden Markov Models (HMMs)?
  - How do they relate to Dynamic Bayesian Networks?
- What are Input/Output HMMs (IOHMMs)?
- EM algorithm for learning IOHMMs
- Application of IOHMMs to examine regulatory network dynamics

Motivation

- Suppose we are given time series expression profiles
- We wish to find key regulators that are associated with changes in expression levels over time
- We have seen a simple approach to do this
  - Activity subgraph/skeleton network-based approaches
- Can we more explicitly take time into account?

DREM: Dynamic Regulatory Events Miner

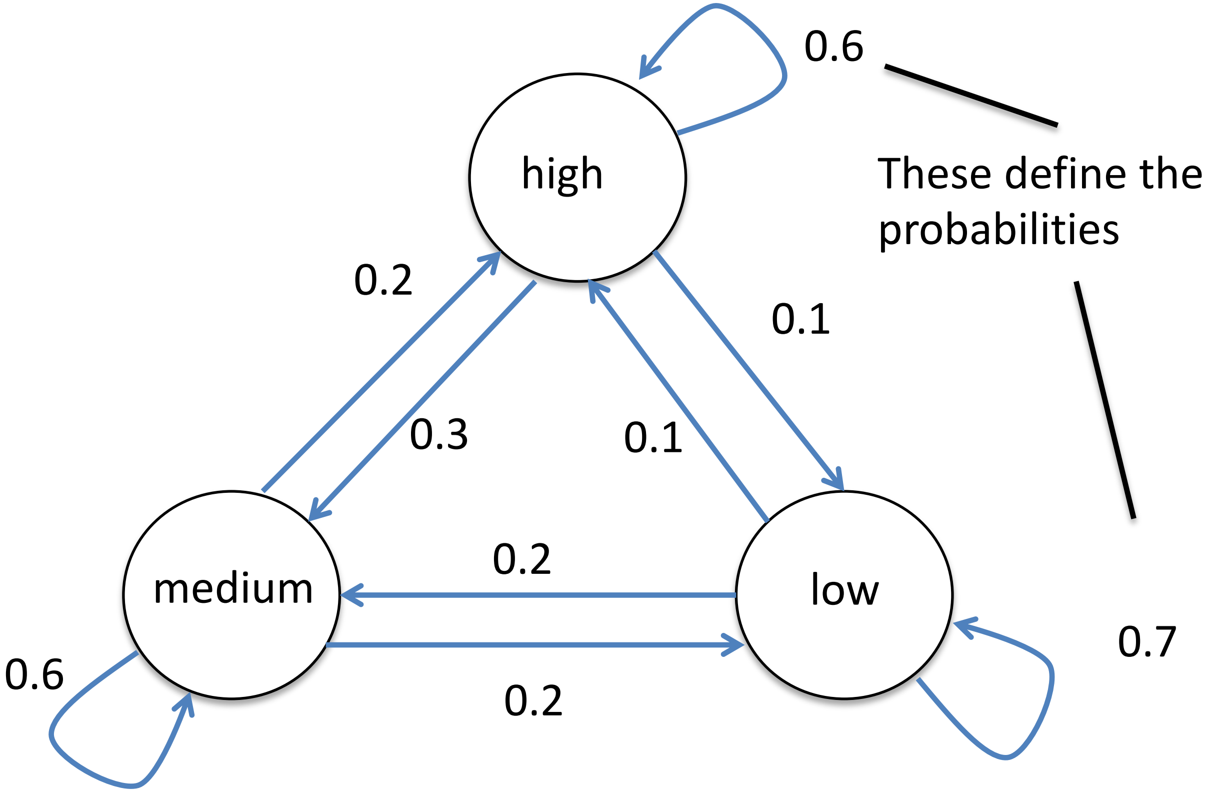


Ernst et al., 2007, Mol Sys Biol

Recall Markov chain

- A Markov chain is a probabilistic model for sequential observations where there is a dependency between the current and the previous state
- It is defined by a graph of possible states and a transition probability matrix defining transitions between each pair of state
- The states correspond to the possible assignments a variable can state
- One can think of a Markov chain as doing a random walk on a graph with nodes corresponding to each state

A three state Markov chain



These define the transition probabilities

$P(X_{t+1}=high|X_t=low)=0.1$

Hidden Markov Models

- Hidden Markov models are also probabilistic models used to model sequential data about a dynamical system
- At each time point the system is a hidden state that is dependent upon the previous states (history)
- The observation sequence is the output of a hidden state
- HMMs are defined by observation models and transition models

Murphy 2000

Notation

- States are numbered from 1 to K
  - – observed character at position t
- $x = \{x_1, \cdots, x_T\}$ Observed sequence
- $\pi = \{\pi_1, \cdots, \pi_T\}$ Hidden state sequence or path
- Transition probabilities

$$a_{kl} = P(\pi_{t+1} = l | \pi_t = k)$$

- Emission probabilities: Probability of emitting symbol b from state k

$$e_k(b) = P(x_t = b | \pi_t = k)$$

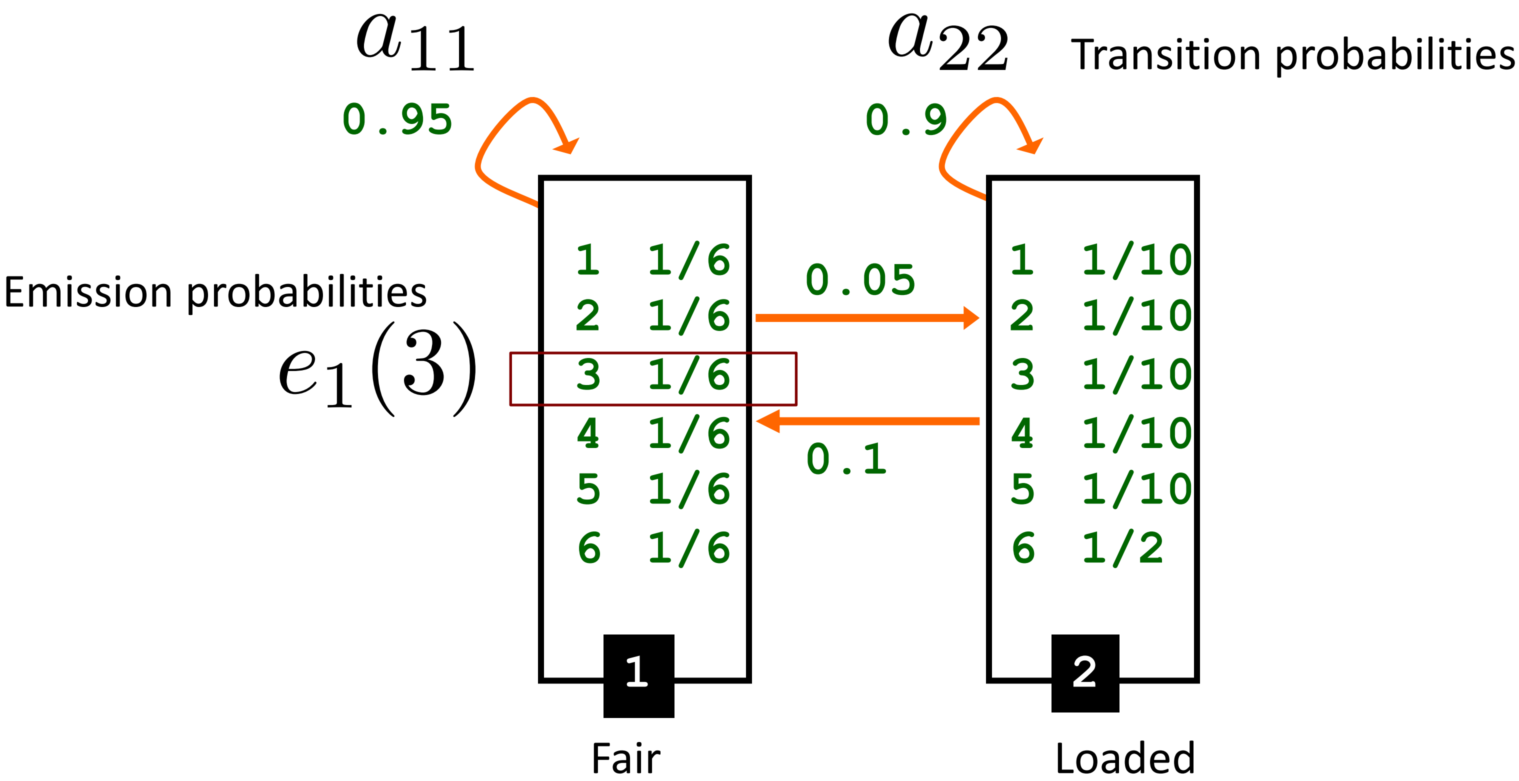
What does an HMM do?

- Enables us to model observed sequences of characters generated by a hidden dynamic system
- The system can exist in a fixed number of “hidden” states
- The system *probabilistically transitions* between states and at each state it *emits* a symbol/character

Defining an HMM

- States
- Emission alphabet
- Parameters
  - State transition probabilities for probabilistic transitions from state at time t to state at time $t+1$
  - Emission probabilities for probabilistically emitting symbols from a state

An HMM for an occasionally dishonest casino



a_{11} 0.95

a_{22} 0.9

Transition probabilities

Emission probabilities

$e_1(3)$

1	1/6
2	1/6
3	1/6
4	1/6
5	1/6
6	1/6

0.05

0.1

1	1/10
2	1/10
3	1/10
4	1/10
5	1/10
6	1/2

1

Fair

2

Loaded

What is hidden? Which dice is rolled

What is observed? Number (1-6) on the die

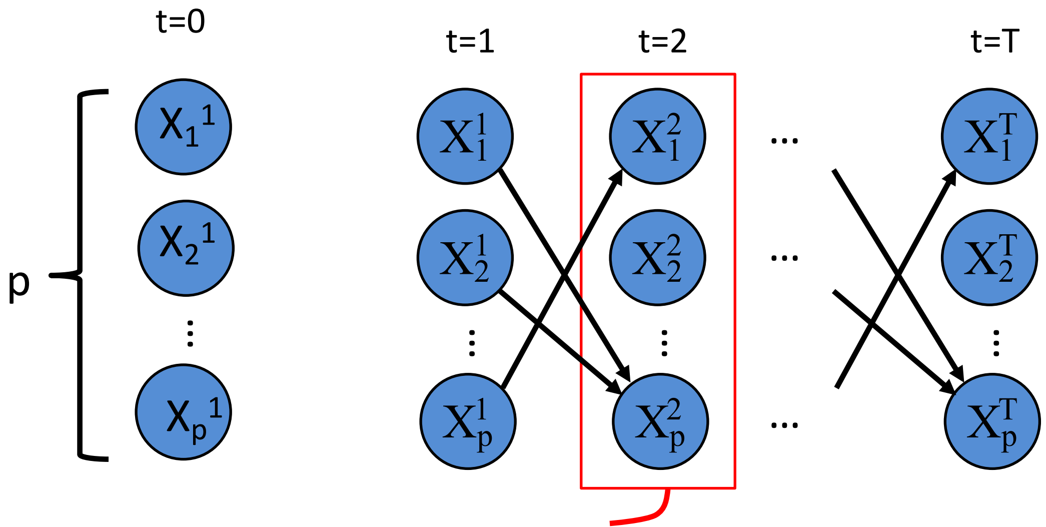
Formally defining a HMM

- States
- Emission alphabet
- Parameters
  - State transition probabilities for probabilistic transitions from state at time t to state at time $t+1$
  - Emission probabilities for probabilistically emitting symbols from a state

Goals for today

- What are Hidden Markov Models (HMMs)?
  - How do they relate to Dynamic Bayesian Networks?
- What are Input/Output HMMs (IOHMMs)?
- Learning problems of IOHMMs
- Application of IOHMMs to examine regulatory network dynamics

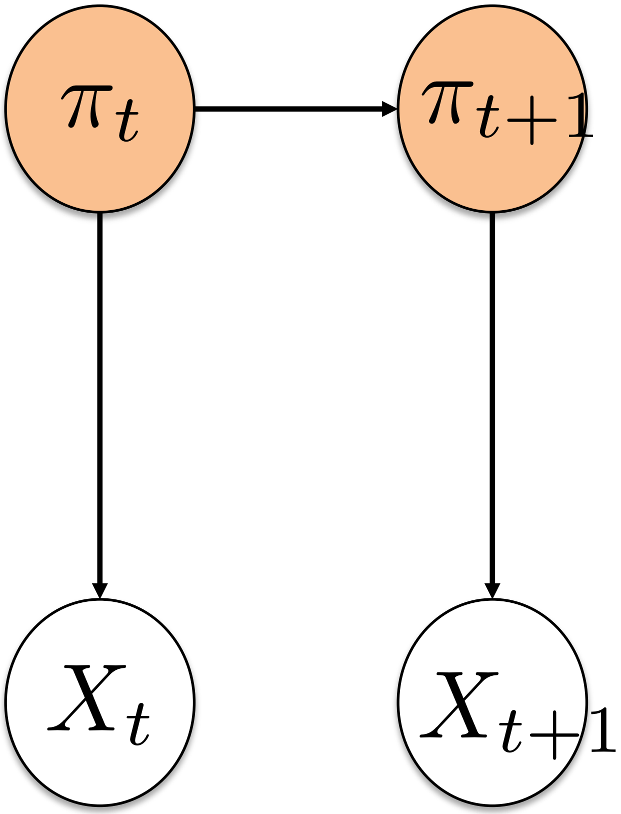
Recall a DBN for *p* variables and *T* time points



$X^{2:}$ Variables at time t=2

Dependency at the first time point

HMM represented as a DBN



DBN

- A DBN could be used to represent the transition probabilities more compactly.

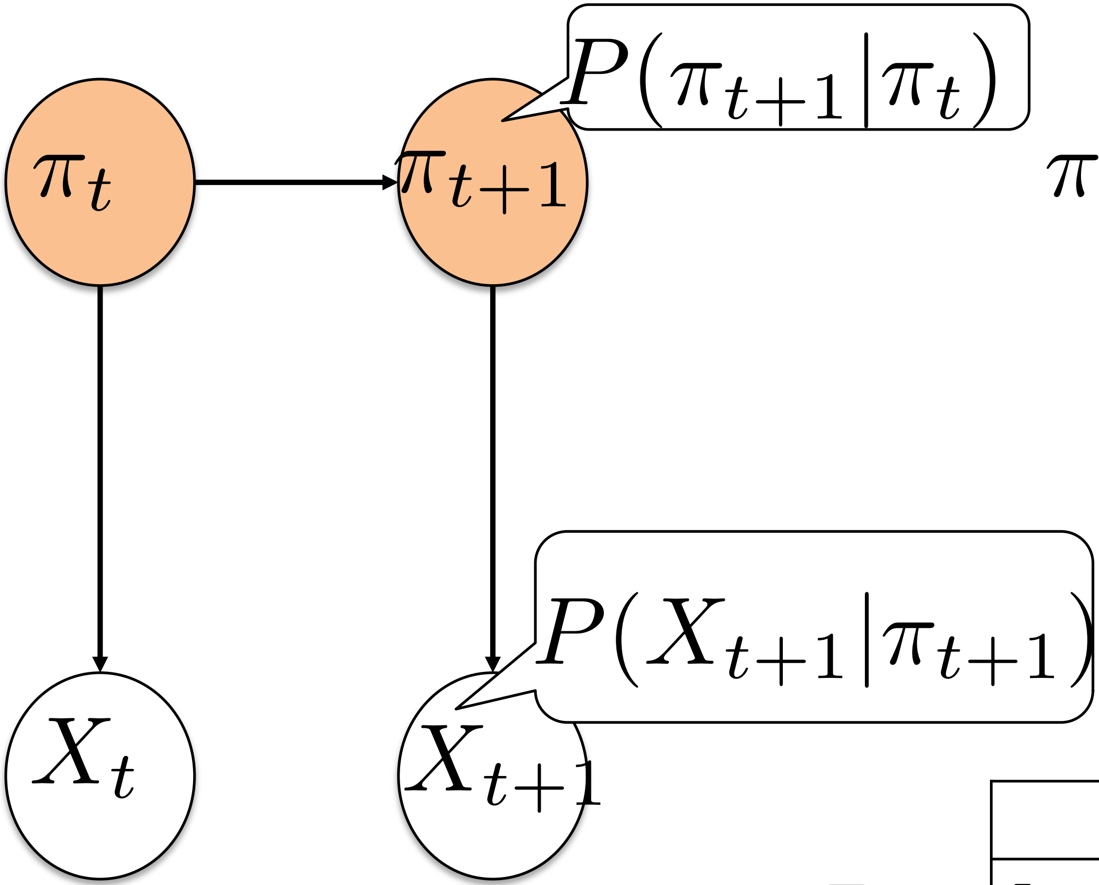
- For example, consider the state variable to be D-dimensional each with K possible values.
  - For example we are tracking D objects and each object can have K possible settings
  - The state variable can have K^D possible values

- An HMM will attempt to model the transition probabilities between all state combinations.

- In other words, the DBN will look fully connected.

Kevin Murphy, 2000

DBN version of the occasional dishonest casino



π_{t+1}

		F	L
π_t	F	0.95	0.05
	L	0.1	0.9

X_{t+1}

		1	2	3	4	5	6
π_{t+1}	F	1/6	1/6	1/6	1/6	1/6	1/6
	L	0.1	0.1	0.1	0.1	0.1	0.5

Three important questions in HMMs

- What is the probability of a sequence from an HMM?
  - *Forward algorithm*
- What is the most likely sequence of states for generating a sequence of observations
  - *Viterbi algorithm*
- How can we learn an HMM from a set of sequences?
  - *Forward-backward or Baum-Welch (an EM algorithm)*

Computing the probability of a sequence from an HMM

$$P(x_1,\cdots,x_T,\pi_1\cdots,\pi_T) =$$

$$a_{0\pi_1}\prod_{t=1}^{T} e_{\pi_t}(x_t)a_{\pi_t\pi_{t+1}}$$

Initial transition

Emitting symbol x_t

State transition between consecutive time points

Computing the probability of a sequence from an HMM

- But we don’t know what the sequence of states (path) is
- So we need to sum over all paths
- The probability over *all* paths is:

$$P(x_1, \cdots, x_T) = \underbrace{\sum_{\pi_1 \cdots, \pi_T}}_{\text{Sum over all paths}} a_{0\pi_1} \prod_{t=1}^{T} e_{\pi_t}(x_t) a_{\pi_t \pi_{t+1}}$$

- The forward algorithm gives an efficient way to compute this probability
- It is based on the concept of dynamic programming

Goals for today

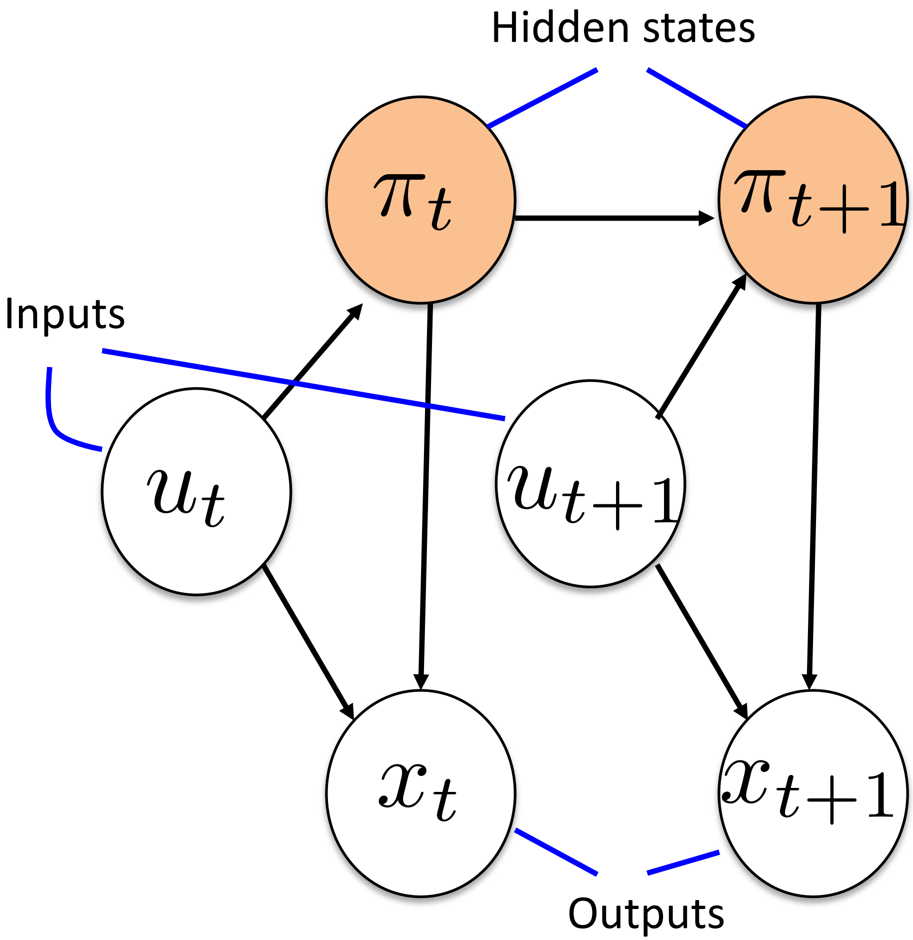
- What are Hidden Markov Models (HMMs)?
  - How do they relate to Dynamic Bayesian Networks?
- What are Input/Output HMMs (IOHMMs)?
- Learning problems of IOHMMs
- Application of IOHMMs to examine regulatory network dynamics

Input output Hidden Markov Models (IOHMM)

- As in the HMM we have
  - States, emissions and transitions
- In addition we have a sequence of inputs
  - The transitions and emissions can depend on inputs ($u_1,..,u_T$)
- In a way, IOHMMs map inputs to outputs
  - This is different from HMMs
- HMMs aim to define $P(x_1..x_T)$ while IOHMMs define $P(x_1..x_T|u_1..u_T)$

Bengio & Frasconi, IEEE Trans on Neural Networks 1996

Input output Hidden Markov Models (IOHMM)



Transition and emissions are dependent upon a set of external stimuli

Formally defining an IOHMM

- The set of *K* hidden states
- Emission characters/symbols/values
- Transition probabilities conditioned on the input
  - Unlike HMMS where we had $a_{kl} = P(\pi_{t+1} = l | \pi_t = k)$
  - Here we have $a_{kl} = P(\pi_{t+1} = l | \pi_t = k, u_{t+1})$
- Similarly for emission probabilities on the input and state

$$e_k(x_t) = P(x_t | \pi_t = k, u_t)$$

Three important questions in IOHMMs

- What is the probability of a sequence from an IOHMM?
  - *Forward algorithm*
- What is the most likely sequence of states for generating a sequence of observations
  - *Viterbi algorithm*
- How can we learn an IOHMM from a set of sequences?
  - *Forward-backward algorithm (an EM algorithm)*

Computing the probability of a sequence from an IOHMM

$$P(x_1, \cdots, x_T, \pi_1, \cdots, \pi_T | u_1, \cdots, u_T)$$

$$= \prod_{t=1}^{T} P(x_t | \pi_t, u_t) P(\pi_t | \pi_{t-1}, u_t)$$

Emitting symbol x_t

State transition between consecutive time points

As in the case of HMMs

- We would need to sum over the possible state configurations

$$P(x_1, \cdots, x_T | u_1 \cdots, u_T) = \sum_{\pi_1, \cdots, \pi_T} \prod_{t=1}^{T} P(x_t | \pi_t, u_t) P(\pi_t | \pi_{t-1}, u_t)$$

Sum over all paths

- We will use the forward algorithm for this problem

How likely is a given sequence: Forward algorithm

- Define $f_k(t)$ as the probability of observing $x_1, \cdots, x_t$ and ending in state k at time t given inputs $u_1..u_t$

$$f_k(t) = P(x_1, \cdots, x_t, \pi_t = k | u_1, \cdots, u_t)$$

- This can be written as follows

$$f_k(t+1) = P(x_{t+1}|\pi_{t+1}, u_{t+1}) \sum_{l=1}^{K} f_l(t) P(\pi_{t+1} = k | \pi_t = l, u_{t+1})$$

$$f_k(t+1) = e_k(x_{t+1}) \sum_{l=1}^{k} f_l(t) a_{lk}$$

Steps of the Forward algorithm

- Initialization

$$f_k(1) = e_k(x_1)P(\pi_1 = k|u_1)$$

- Recursion: for t=2 to T

$$f_k(t) = e_k(x_t)\sum_l a_{lk} f_l(t-1)$$

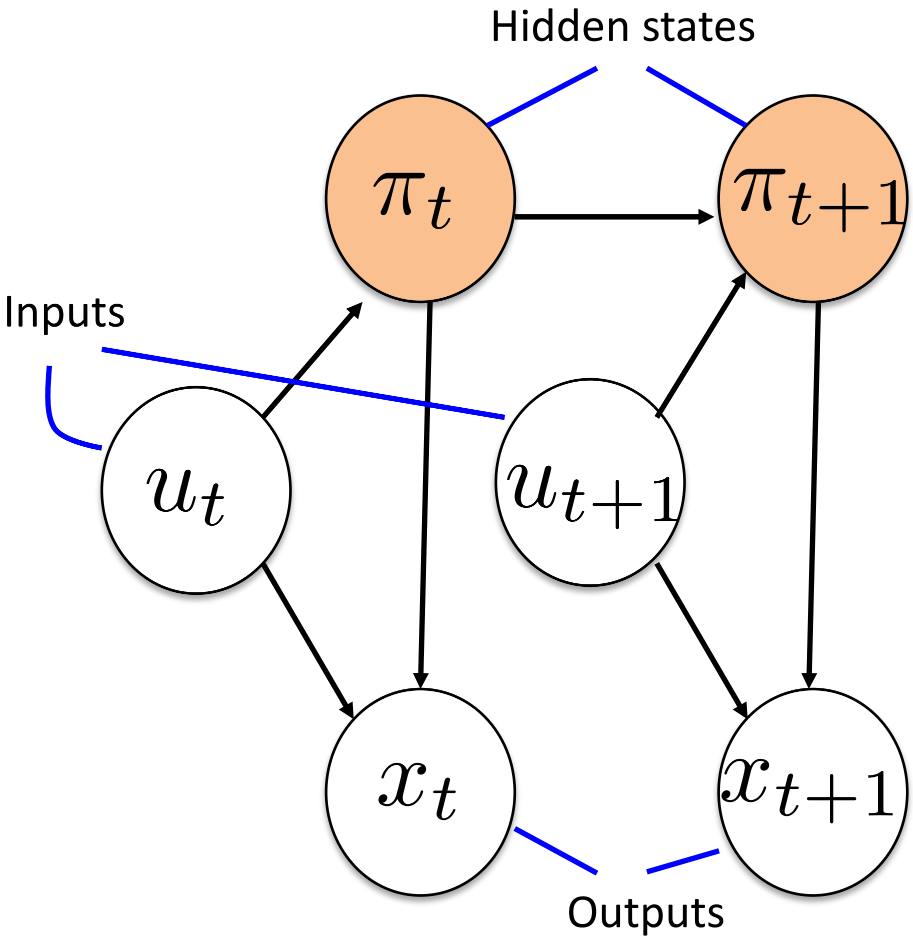
- Termination

$$P(x_1, \cdots, x_T|u_1, \cdots, u_T) = \sum_{l=1}^{K} f_l(T)$$

Working through an example

- Suppose we are able to measure three reporter molecules whose values are dependent upon input chemical stimulus and whether one of four possible hidden pathways are triggered.
- Chemical stimulus: {0,1}
- Hidden Pathways: {A, B, C, D}
- Reporter molecules: $\{r_1, r_2, r_3\}$
- Given a sequence of reporter molecule measurements, and chemical stimuli, infer which hidden pathway was likely triggered

Mapping to an IOHMM



$$\pi_t \in \{A, B, C, D\}$$
$$u_t \in \{0, 1\}$$
$$x_t \in \{r_1, r_2, r_3\}$$

We need to specify three CPTs

$$P(\pi_1 | u_1)$$
$$P(\pi_t | \pi_{t-1}, u_t)$$
$$P(x_t | \pi_t, u_t)$$

The CPTs that we will use

π_1

u_1

	A	B	C	D
0	0.5	0.5	0	0
1	0.5	0.5	0	0

x_t

π_t, u_t

		r_1	r_2	r_3
0	**A**	0.8	0.1	0.1
0	**B**	0.2	0.6	0.2
0	**C**	0.25	0.5	0.25
0	**D**	0.2	0.2	0.6
1	**A**	0.2	0.6	0.2
1	**B**	0.25	0.5	0.25
1	**C**	0.2	0.2	0.6
1	**D**	0.5	0.25	0.25

π_{t+1}

π_t, u_{t+1}

		A	B	C	D
0	**A**	0.6	0.2	0.2	0
0	**B**	0.2	0.6	0	0.2
0	**C**	0.1	0	0.8	0.1
0	**D**	0	0.25	0.25	0.5
1	**A**	0.8	0.1	0.1	0
1	**B**	0.8	0.1	0	0.1
1	**C**	0.1	0	0.8	0.1
1	**D**	0	0.1	0.8	0.1

Suppose we observed the following sequences

Input: 0 1 1 0

Output: r_1 r_1 r_2 r_3

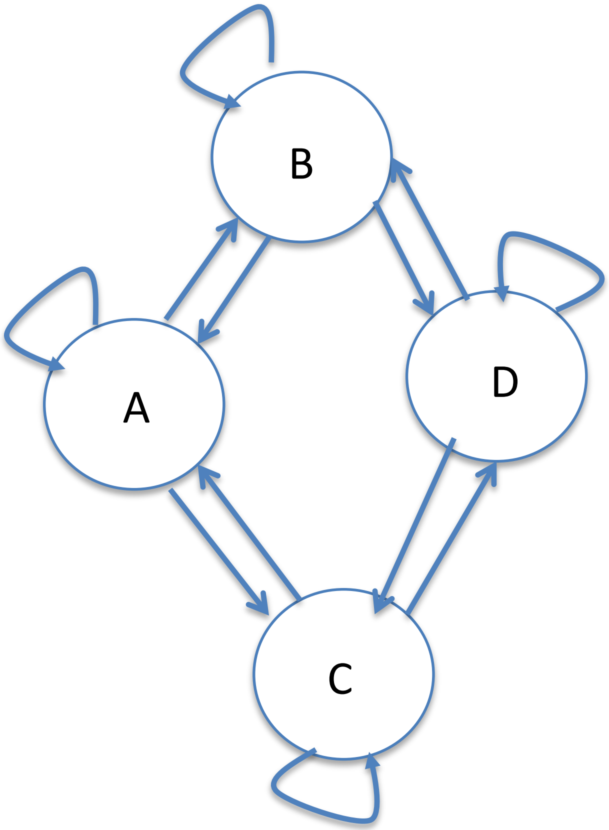
How likely is this observation from our IOHMM?

Transition probabilities encode some independencies

π_{t+1}

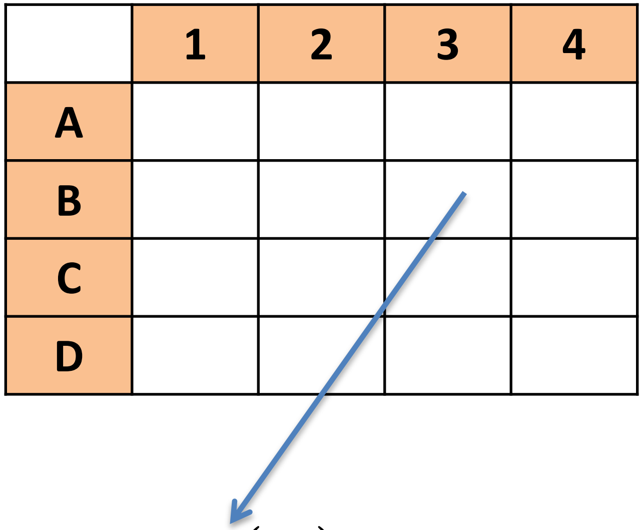
π_t, u_{t+1}

		A	B	C	D
0	A	0.6	0.2	0.2	0
0	B	0.2	0.6	0	0.2
0	C	0.1	0	0.8	0.1
0	D	0	0.25	0.25	0.5
1	A	0.8	0.1	0.1	0
1	B	0.8	0.1	0	0.1
1	C	0.1	0	0.8	0.1
1	D	0	0.1	0.8	0.1



Applying the forward algorithm

Input:	**0**	**1**	**1**	**0**
Output:	r_1	r_1	r_2	r_3



	1	**2**	**3**	**4**
A				
B				
C				
D				

$$f_B(3) = P(r_2|B,1) * (f_A(2)P(B|A,1) + f_B(2)P(B|B,1) + f_D(2)P(D|C,1))$$

Result of applying the forward algorithm

Input:	**0**	**1**	**1**	**0**
Output:	r_1	r_1	r_2	r_3

	1	**2**	**3**	**4**
A	0.4	0.08	0.04488	0.0033861
B	0.1	0.0125	0.033125	0.021860825
C	0	0.008	0.00308	0.0028790625
D	0	0.01	0.0007625	0.00438855

$$P(r_1, r_1, r_2, r_3 | 0, 1, 1, 0) = f_A(4) + f_B(4) + f_C(4) + f_D(4) = 0.0325$$

Learning an IOHMM from data

- Given J paired sequences
$$\{(\boldsymbol{x}_{1:T_1}, \boldsymbol{u}_{1:T_1}), .., (\boldsymbol{x}_{1:T_J}, \boldsymbol{u}_{1:T_J})\}$$
- Parameter estimation:
  - Learn the transition and emission probability distributions
  - This is very similar to what is done in HMMs
- Structure learning:
  - Learn the number of states and the dependencies among the states
  - Because states are hidden variables and we do not how many there are, this adds another level of complexity in learning
  - We will first assume that we know the number of states

The expectation maximization algorithm

- Expectation Maximization (EM) is a widely used when there are hidden variables
- It is an iterative algorithm that maximizes the likelihood of the data
- Each iteration is made up of two steps
  - Expectation step (E): estimate the expected values of hidden variables given the data and previous parameter settings
  - Maximization step (M): estimate the parameters using the expected counts

Learning without hidden information

- Transition probabilities

$$a_{kl} = P(\pi_t = l|\pi_{t-1} = k, u_t = p)$$

Number of transitions from state k to state l given input p

$$= \frac{n_{k\to l|u_t=p}}{\sum_{l'} n_{k\to l'|u_t=p}}$$

- Emission probabilities

$$e_k(c) = P(x_t = c|\pi_t = k, u_t = p)$$

Number of times c is emitted from k given input p

$$= \frac{n_{k,c|u_t=p}}{\sum_{c'} n_{k,c'|u_t=p}}$$

The expectation step

- We need to know the probability of the symbol at *t* being produced by state *i*, given the entire observation and input sequence $u_{1:T}, x_{1:T}$

$$P(\pi_t = k | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T})$$

- We also need to know the probability of observations at *t* and *(t+1)* being produced by state *i*, and *l* respectively given sequence *x*

$$P(\pi_t = i, \pi_{t-1} = j | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T})$$

- Given these we can compute our expected counts for state transitions, character emissions

Bengio & Frasconi, IEEE Trans on Neural Networks 1996

Computing $P(\pi_t = k | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T})$

- First we compute the probability of the entire observed sequence with the t^{th} symbol being generated by state k

$$P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})$$

- Then our quantity of interest is computed as

$$P(\pi_t = k | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T}) = \frac{P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}{P(\boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}$$

Obtained from the forward algorithm

Computing $P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})$

- To compute

$$P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})$$

- We need the forward and backward algorithm

$$= P(\boldsymbol{x}_{1:t}, \pi_t = k | \boldsymbol{u}_{1:t}) P(\boldsymbol{x}_{t+1:T} | \pi_t = k, \boldsymbol{x}_{1:t}, \boldsymbol{u}_{1:T})$$

$$= P(\underbrace{\boldsymbol{x}_{1:t}, \pi_t = k | \boldsymbol{u}_{1:t})}_{\text{Forward algorithm } f_k(t)} \underbrace{P(\boldsymbol{x}_{t+1:T} | \pi_t = k, \boldsymbol{u}_{t:T})}_{\text{Backward algorithm } b_k(t)}$$

Steps of the backward algorithm

- Initialization (*t*=*T*)

$$b_k(t) = 1$$

- Recursion (*t*=*T*-*1 to 1*)

$$b_k(t) = \Sigma_l a_{kl} e_l(x_{t+1}) b_l(t+1)$$

Trying out an example with backward algorithm

- Again assume we have the same CPTs as those associated with the forward algorithm demo
- Assume we observe the following

Input: 0 1 1

Output: r_1 r_2 r_2

- What are computations for the backward algorithm?

Results from applying the backward algorithm

Input: 0 1 1

Output: r_1 r_2 r_2

	1	2	3
A		0.19	1
B		0.09	1
C			1
D			1

π_{t+1}

π_t, u_{t+1}

		A	**B**	**C**	**D**
0	**A**	0.6	0.2	0.2	0
0	**B**	0.2	0.6	0	0.2
0	**C**	0.1	0	0.8	0.1
0	**D**	0	0.25	0.25	0.5
1	**A**	0.8	0.1	0.1	0
1	**B**	0.8	0.1	0	0.1
1	**C**	0.1	0	0.8	0.1
1	**D**	0	0.1	0.8	0.1

$$b_B(2) = P(A|B,1)P(r_2|A,1)b_A(3) + P(B|B,1)P(r_2|B,1)b_B(3)$$
$$+P(D|B,1)P(r_2|D,1)b_D(3)$$

Computing $P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})$

- Using the forward and backward variables, this is computed as

$$P(\pi_t = k | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T}) = \frac{P(\pi_t = k, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}{P(\boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}$$

$$P(\pi_t = k | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T}) = \frac{f_k(t) b_k(t)}{P(\boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}$$

Computing $P(\pi_t = i, \pi_{t-1} = j | \boldsymbol{u}_{1:T}, \boldsymbol{x}_{1:T})$

- This is the probability of symbols at t and $t+1$ emitted from states k and l given the entire observed and sequence $x_{1:T}$ and input sequence $u_{1:T}$

$$= \frac{P(\pi_t = i, \pi_{t-1} = j, \boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}{P(\boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}$$

$$= \frac{f_j(t-1) a_{ji} e_i(x_t) b_i(t)}{P(\boldsymbol{x}_{1:T} | \boldsymbol{u}_{1:T})}$$

Putting it all together

- Assume we are given J training instances
$$\{(\boldsymbol{x}_{1:T_1}, \boldsymbol{u}_{1:T_1}), .., (\boldsymbol{x}_{1:T_J}, \boldsymbol{u}_{1:T_J})\}$$
- Expectation step
  - Using current parameter values compute for each $(\boldsymbol{x}_{1:T_J}, \boldsymbol{u}_{1:T_j})$
    - Apply the forward and backward algorithms
    - Compute
      - expected number of transitions between all pairs of states
      - expected number of emissions for all states
- Maximization step
  - Using current expected counts
    - Compute the transition and emission probabilities

Baum-Welch one iteration

- Let's assume we have J=2 training instances

$$\{(0,1,1),(r_1,r_2,r_2)\}$$

$$\{(1,0,0),(r_2,r_1,r_1)\}$$

- Each training example will contribute to the expected counts of transition and emission probabilities
- Expectation step:
  - Compute the forward and backward variables for both training samples, for all time points

Baum-Welch one iteration M step

- Suppose we are updating the transition probability of A to B given input u=1

$$\{(0,1,1),(r_1,r_2,r_2)\}$$

$$\{(1,0,0)(r_2,r_1,r_1)\}$$

$$n_{A\to B|u=1}$$

$$= \frac{f_A(1)a_{AB}e_B(r_2)b_B(2) + f_A(2)a_{AB}e_B(r_2)b_B(3)}{P(r_1,r_2,r_2|0,1,1)}$$

Contribution from sample 1

Sample 2 will not contribute as there is no relevant configuration

Baum-Welch one iteration M step

- Suppose we are updating the expected counts for observing r$_2$ from state B given input *u=1*

$$n_{r_2,B|u=1}$$

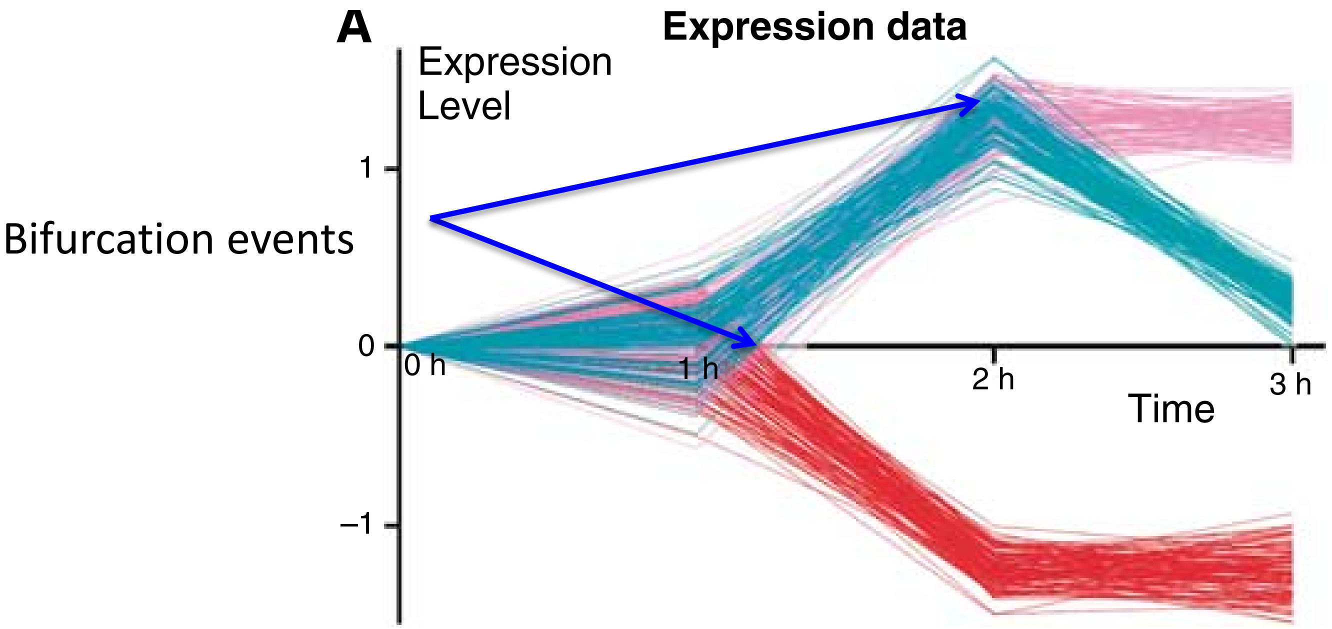
$$= \underbrace{\frac{f_B(2)b_B(2)+f_B(3)b_B(3)}{P(r_1,r_2,r_2|0,0,1)}}_{\text{Contribution from sample 1}} + \underbrace{\frac{f_B(1)b_B(1)}{P(r_2,r_1,r_1|1,0,0)}}_{\text{Contribution from sample 2}}$$

Goals for today

- What are Hidden Markov Models (HMMs)?
  - How do they relate to Dynamic Bayesian Networks?
- What are Input/Output HMMs (IOHMMs)?
- Learning problems of IOHMMs
- Application of IOHMMs to examine regulatory network dynamics

Bifurcation events

- Bifurcation events occur when sets of genes that have roughly the same expression level up until some time point diverge



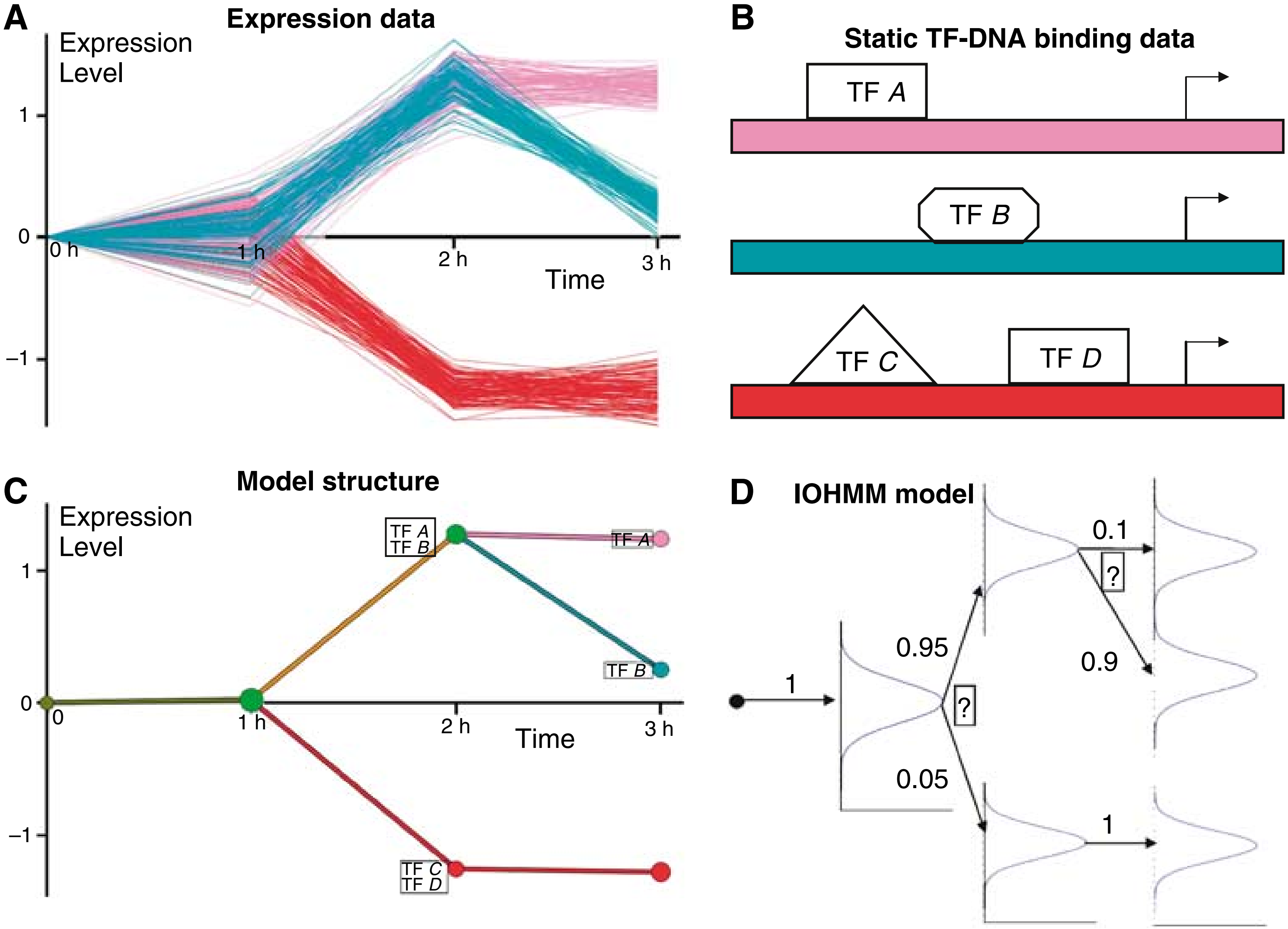
Dynamic Regulatory Events Miner (DREM)

- Given
  - a gene expression time course
  - Static TF binding data or signaling networks
- Do
  - Identifies important regulators for interesting temporal changes

- DREM is suited for short time courses
- DREM is based on an Input-Output HMM

Ernst et al., 2007 Mol Sys Biol

DREM key idea



Ernst et al., 2007, Mol Sys Biol

IOHMM model in DREM

- The output distributions were modeled as Gaussians
  - Enabled modeling continuous expression values
- State transitions depended on static input and the current state
  - A binary classifier was trained in the M step for each state with two children, to discriminate between genes assigned to the bifurcating states

Defining the transition probability in DREM

- DREM uses a binary classifier (logistic regression) to define transition probabilities
- Assume we are state h, which has two child states a and b

$$P(x_{t+1} = a | x_t = h, \boldsymbol{u}^i) = \frac{1}{1 + \exp(-\beta_0^h - \sum_f \beta_f^h \boldsymbol{u}^i(f))}$$

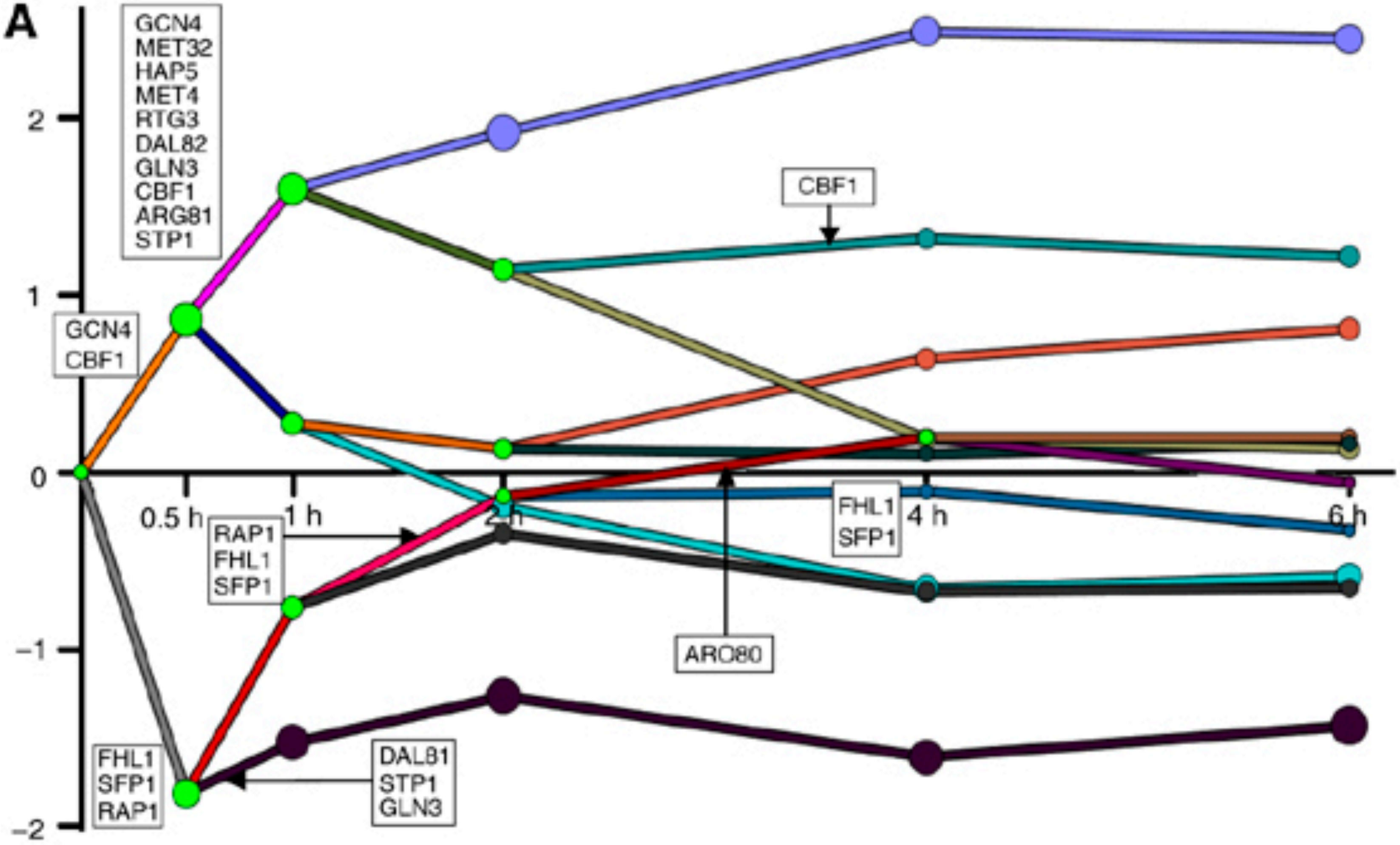
Input associated with the i^{th} gene: collection of binding sites on gene i's promoter

State-specific parameters

Results

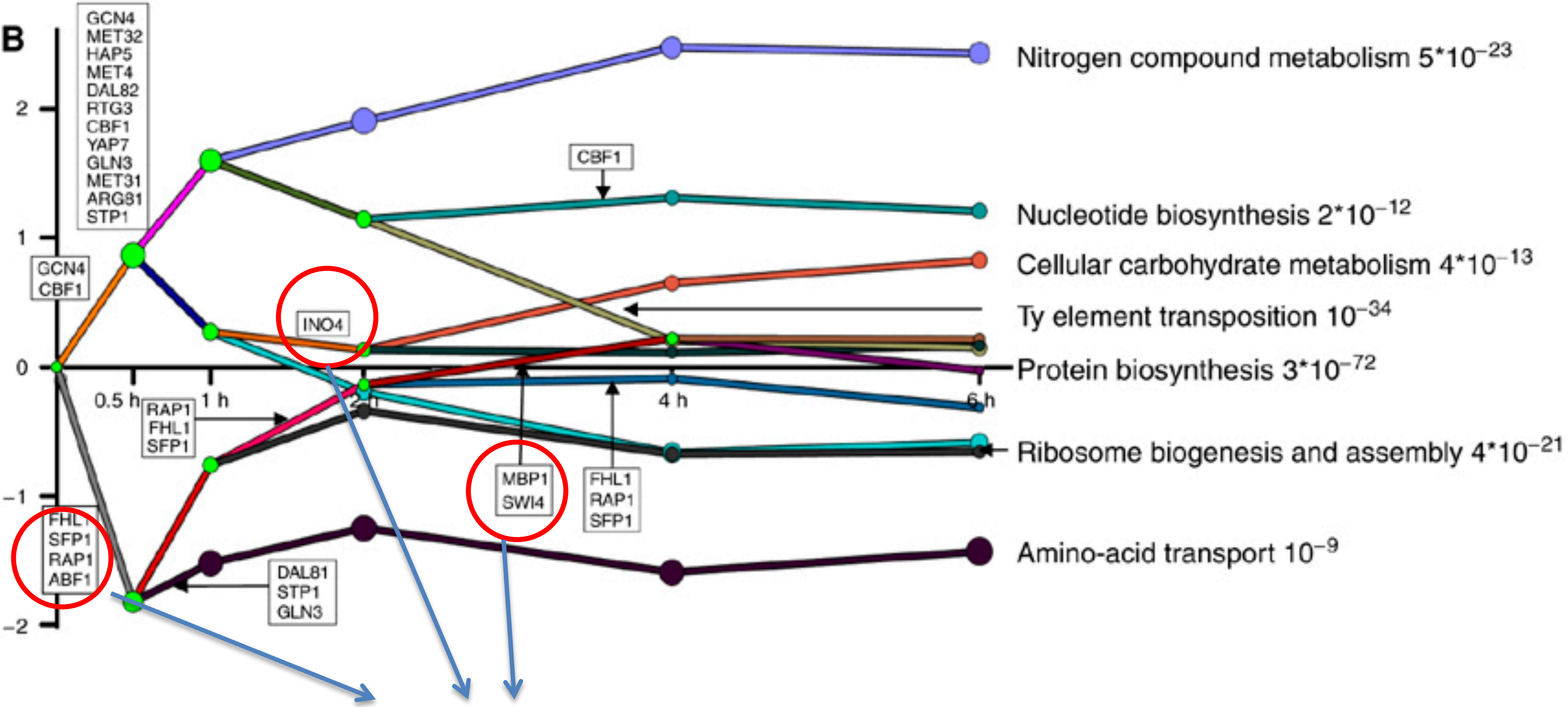
- Application of DREM to yeast expression data
  - Amino acid (AA) starvation
  - One time point ChIP binding in AA starvation
- Analysis of condition-specific binding
- Application to multiple stress and normal conditions

DREM application in yeast amino acid starvation



DREM identified 11 paths, and associated important AA related TFs for each split

Does condition non-specific data help?



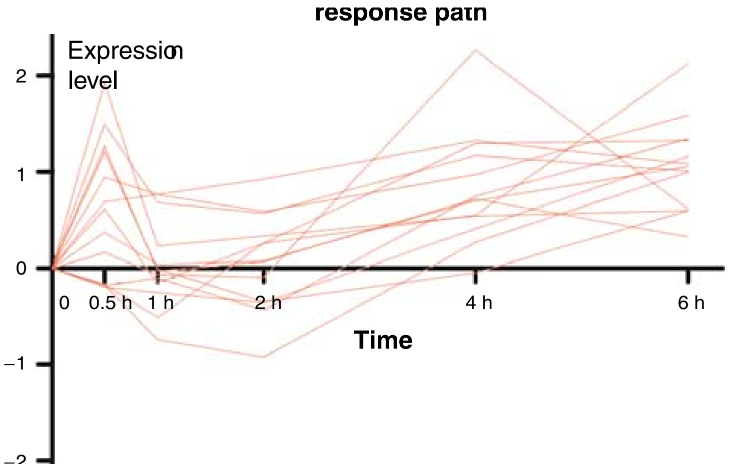
Yes, adding additional non-condition specific data helped explain more splits or found more TFs per split

Validation of INO4 binding

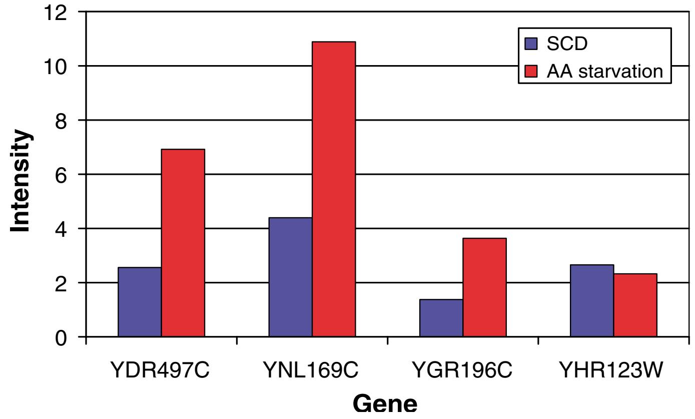
- INO4 was a novel prediction by the method
- Using a small scale experiment, test binding in 4 gene promoters after AA starvation
- Measure genome-wide binding profile of INO4 in AA starvation and SCD and compare relative binding

Validation of INO4 binding

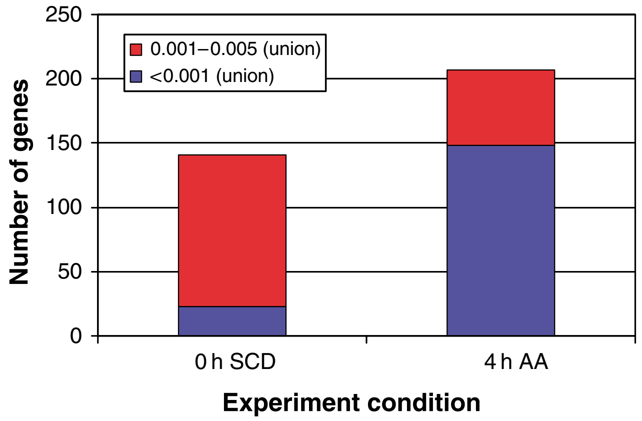
Genes associated with INO4 split profile



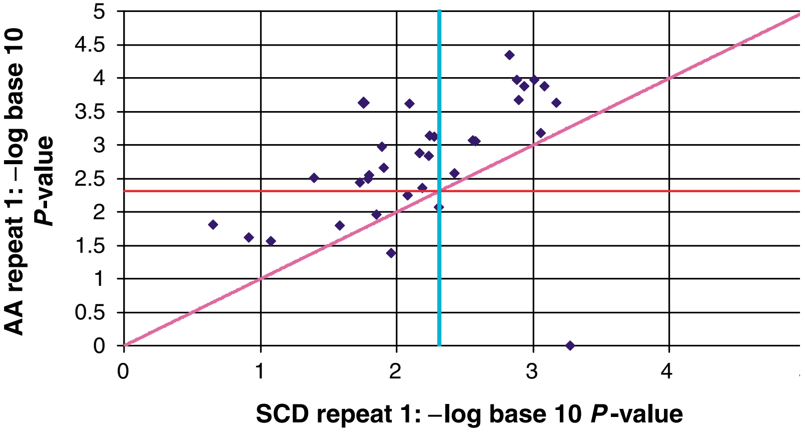
INO4 occupancy is much higher in AA starvation compared to normal (SCD)



More genes are bound genome-wide in AA starvation



Stronger binding in AA starvation of genes in this path



Does integration help?

- Randomize ChIP data and ask if enriched TFs with paths were identified
  - Fewer TFs were identified
- Compare IOHMM vs HMM
  - Lesser enrichment of Gene Ontology processes in HMMs paths compared to IOHMMs

Take away points

- Network dynamics can be defined in multiple ways
- Skeleton network-based approaches
  - + The universe of networks is fixed, nodes become on or off
  - + Simple to implement, and does not need lot of data
  - + No assumption of how the network changes over time
  - – No model of how the network changes over time
  - – Requires the skeleton network to be complete
- Dynamic Bayesian network
  - + Can learn new edges
  - + Describes how the system transitions from one state to another
  - + Can incorporate prior knowledge
  - – Assumes that the dependency between t-1 and t is the same for all time points
  - – Requires sufficient number of timepoints
- IOHMMS (DREM approach)
  - + Integrates static TF-DNA and dynamic gene expression responses
  - + Works at the level of groups of genes
  - + Focus on bifurcation points in the time course
  - − Tree structure might be restrictive (although possible extensions are discussed)
  - − Depends upon the completeness of the TF binding data