Some context-specific graphical models for discrete longitudinal data

David Edwards and Smitha Ankinakatte

Center for Quantitative Genetics and Genomics
Department of Molecular Biology and Genetics
Aarhus University
Denmark

Wirtschaftsuniversität, Wien, November 2013

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

APFA equivalent to conventional Markov models

Summary and conclusion

Introduction

- Acyclic probabilistic finite automata[1] (APFA) are a rich family of models for discrete longitudinal data.
- An APFA
  - embodies a set of context-specific conditional independence relations
  - may be represented as a directed multigraph.
  - and is a context-specific graphical model.
- The methodology is highly scalable and is routinely used for high-dimensional genomic data in the Beagle software[2].
- Here we describe the models and methods from a statistical perspective.

---

[1] Ron, Singer and Tishby (1998). On the learnability and usage of acyclic finite automata. J. Comp. Syst. Sci, 56, 133-52.

[2] Browning and Browning (2007). Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. Am. J. Hum. Gen., 81, 1084-1097

Outline

- Introduction
- **Automata**
- Maximum likelihood estimation
- State merging
- Hypothesis tests
- Model selection
- APFA equivalent to conventional Markov models
- Summary and conclusion

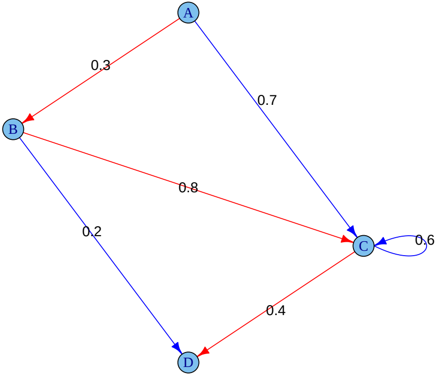
Automata

- Automata are devices that either input strings (as in parsing) or output strings.
- They are used in computer science and machine learning:
  - to represent formal languages and regular expressions;
  - for speech recognition,
  - natural language processing,
  - machine translation.
- We first consider the more general probabilistic finite automata (PFA) before focussing on the subclass of APFA.

Probabilistic Finite Automata

- A PFA is a device to generate random strings of symbols.
- It may be displayed as a directed multigraph, in which
  - nodes are called states,
  - there is one initial or root state with only outgoing edges, and one final or sink state with only incoming edges,
  - self-loops (edges from a state to itself) are allowed,
  - each edge e has a symbol $\sigma(e)$ and a probability $\pi(e)$, and
  - outgoing edges from each state have distinct symbols and the sum of their probabilities is unity.

A PFA



(a)

red='1'; blue='2'

How a PFA generates strings

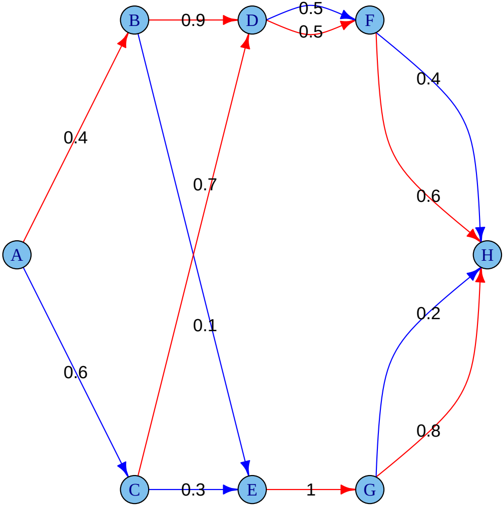
It starts at the root, then repeatedly

- chooses an outgoing edge at random according to the edge probabilities,
- emits the edge symbol,
- traverses the edge to the next state,

until it reaches the sink.

This generates symbol strings of possibly variable length.

An APFA



Acyclic Probabilistic Finite Automata

- An APFA $\mathcal{A}$ is a PFA that generates strings of constant length.
- So all root-to-sink paths have the same length p.
- So all paths from the root to any specific state have the same length, called the level of the state.
- Regard the strings as realizations of a random p-vector $\mathbf{X} = (X_1, X_2, \dots X_p)$.
- Distinct root-to-sink paths $\mathbf{e} = (e_1, e_2, \dots, e_p)$ generate distinct realizations of $\mathbf{X} = \sigma(\mathbf{e}) = (\sigma(e_1), \sigma(e_2), \dots, \sigma(e_p))$.
- The sample space of $\mathbf{X}$ is $\mathbb{X}(\mathcal{A}) = \{\sigma(\mathbf{e}) : \mathbf{e} \in \mathcal{E}(\mathcal{A})\}$, where $\mathcal{E}(\mathcal{A})$ is the set of root-to-sink paths in $\mathcal{A}$.
- For any $\mathbf{x} \in \mathbb{X}(\mathcal{A})$ there exists a unique root-to-sink path $\mathbf{e}$ such that $\mathbf{x} = \sigma(\mathbf{e})$: we write this as $\mathbf{e} = \sigma^{-1}(\mathbf{x})$.

A little theory

- The sample space of X_i, $\mathbb{X}_i$, is the set of symbols on edges incoming to a level i state.
- The parameters are the edge probabilities $\pi = \{\pi(e) : e \in E(\mathcal{A})\}$.
- The $\pi(\mathbf{e})$ specify the right-hand side of

$$\Pr(\mathbf{X} = \mathbf{x}) = \Pr(X_1 = x_1) \prod_{i=2\ldots p} \Pr(X_i = x_i | X_{<i} = x_{<i}) \quad (1)$$

where $\mathbf{X}_{<i} = (X_1, \ldots, X_{i-1})$, $\mathbf{x}_{\geq i} = (x_i, \ldots, x_p)$, $\mathbf{Y}_{\geq i; \leq j} = (Y_i, \ldots, Y_j)$ etc.

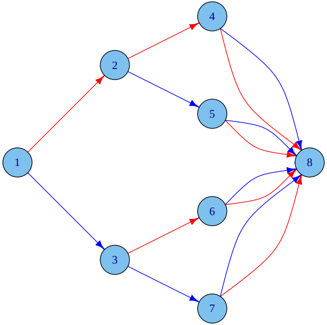
Context-specific conditional independences

- When the data generating process arrives at a level i state w, the distribution of $\mathbf{X}_{>i}$ does not depend on the path the process took to arrive at w. So

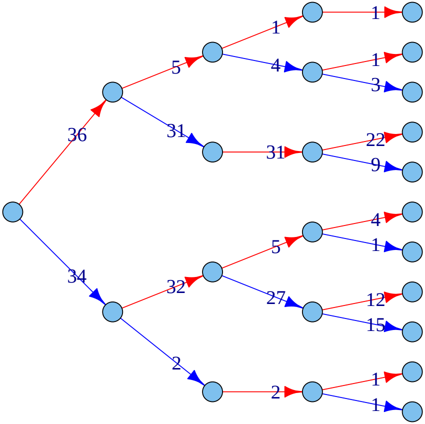
$$\mathbf{X}_{>i} \perp\!\!\!\perp \mathbf{X}_{\leq i} | \mathbf{X}_{\leq i} \in \mathcal{C}(w) \tag{2}$$

  where $\mathcal{C}(w) = \{\sigma(\mathbf{e}) : \mathbf{e} \in \mathcal{P}(w)\}$, and $\mathcal{P}(w)$ is the set of paths from the root to w.
- Thus an APFA expresses a set of context-specific conditional independence constraints on the distribution of $\mathbf{X}$.

Maximal and minimal APFA for three binary variables



A sample tree for $N = 70$ observations of 4 binary variables



To derive the maximal (unrestricted) APFA, contract the states at the last level.

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

APFA equivalent to conventional Markov models

Summary and conclusion

Likelihood

We draw independent samples $\mathbf{x}^{(v)}$ for $v = 1 \dots N$ from $\mathcal{A}$, and want to estimate the $\pi(e)$. We have

$$\Pr(\mathbf{x}) = \prod_{i=1 \dots p} \pi(e_i)$$

where $\mathbf{e} = \sigma^{-1}(\mathbf{x})$ so that the likelihood of the sample is

$$\prod_{v=1 \dots N} \prod_{i=1 \dots p} \pi(e_i^{(v)})$$

where $\mathbf{e}^{(v)} = \sigma^{-1}(\mathbf{x}^{(v)})$. This can be re-written as

$$\prod_{e \in E(\mathcal{A})} \pi(e)^{n(e)}$$

where $n(e)$ is the edge count, i.e. the number of observations in the sample whose root-to-sink path traverses the edge e.

Maximum likelihood

- So the log-likelihood is:

$$\ell(\mathcal{A}) = \sum_{e \in E(\mathcal{A})} n(e) \log \pi(e).$$

- which is easy to maximize:

$$\hat{\pi}(e) = \frac{n(e)}{n(v)}, \tag{3}$$

where $n(v)$ is the node count of v, the source node of e.

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

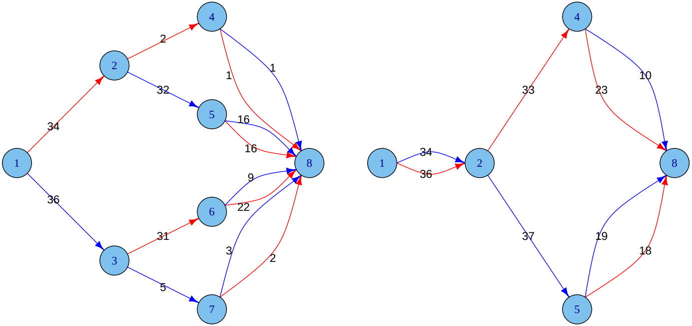
APFA equivalent to conventional Markov models

Summary and conclusion

State merging

- Simplifying APFA involves state merging.
- Only states at the same level may be merged.
- Suppose we wish to merge state w into state v.
- That is, redirect all incoming edges to w to v instead, and all outgoing edges from w to outgo from v instead.
- This can lead to the existence of outgoing edges from v with duplicate symbols.
- Any such edges must therefore also be merged, and if their target nodes are distinct, these must also be merged.
- So the operation is recursive.
- Write $\mathcal{L}(s)$ for the merge-list induced by merging s. E.g. $\mathcal{L}(\{2,3\}) = \{2,3\}, \{5,7\}, \{4,6\}$.

An example of state merging



(a)

(b)

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

APFA equivalent to conventional Markov models

Summary and conclusion

Likelihood ratio tests

- We can construct likelihood ratio tests of nested hypotheses, that is of $\mathcal{A}_0$ versus $\mathcal{A}$, where $\mathcal{A}_0$ is a submodel of $\mathcal{A}$.
- For example, for the APFA shown 3 slides back, the deviance is

$$\begin{aligned} G^2 &= -2[\hat{\ell}(\mathcal{A}) - \hat{\ell}(\mathcal{A}_0)] \qquad (4) \\ &= 53.1228 \qquad (5) \end{aligned}$$

- Under $\mathcal{A}_0$, $G^2 \sim \chi^2(k)$ where k is the difference in model dimension (number of free parameters) between the models.
- By inspection we see that $\mathcal{A}$ has 7 free parameters and $\mathcal{A}_0$ has 4, so $k = 3$, and clearly $\mathcal{A}_0$ fits very poorly.

Likelihood ratio tests continued

- The same test can be computed by applying a standard contingency table test of independence to the table

source	(1,1)	(1,2)	(2,1)	(2,2)
2	2	3	22	9
3	16	16	1	1

- Recall that for an $r \times c$ table of counts $\{n_{ij}\}_{i=1\ldots r; j=1\ldots c}$ the deviance is

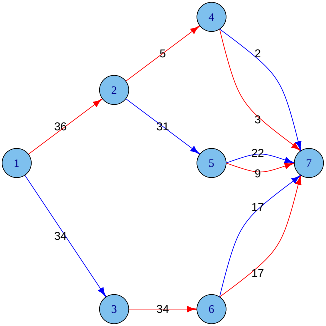
$$G^2 = 2 \sum_{i,j} n_{ij} \log \frac{n_{ij} n_{++}}{n_{i+} n_{+j}} \tag{6}$$

with degrees of freedom given as

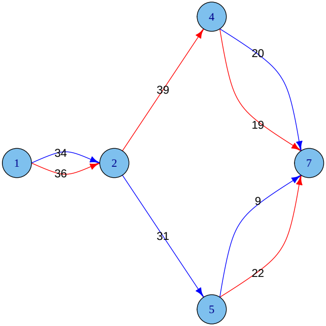
$$k = (\#\{i : n_{i+} > 0\} - 1)(\#\{j : n_{+j} > 0\} - 1) \tag{7}$$

where n_{i+} and n_{+j} are the row and column totals, respectively.

Another example



(a)



(b)

Adjusted degrees of freedom

- As before we can construct the contingency table

source	(1,1)	(1,2)	(2,1)	(2,2)
2	2	3	22	9
3	17	17	0	0

and find $G^2 = 67.288$ on 3 d.f.

- or we can decompose the test

element of $\mathcal{L}(2,3)$	2×2 table		G^2	df
(2,3)	5 34	31 0	67.112	1
(4,6)	2 17	3 17	0.176	1
sum			67.288	2

and find $G^2 = 67.288$ on 2 d.f.

- This is a sharper result that takes account of inestimability.
- We call these the adjusted degrees of freedom.
- For large APFA the unadjusted and adjusted degrees of freedom can differ considerably.

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

APFA equivalent to conventional Markov models

Summary and conclusion

The model selection algorithm of Ron et al (1998)

- The sample tree is constructed and then simplified in a series of state merging operations.
- Two nodes v and w at level i are merged

  $$\Pr(\text{future}|\mathbf{X}_{\leq i} \text{ goes through } v) = \Pr(\text{future}|\mathbf{X}_{\leq i} \text{ goes through } w).$$

  or in other words if $\forall\ \mathbf{x}_{>i}$,

  $$\Pr(\mathbf{X}_{>i} = \mathbf{x}_{>i}|\mathbf{X}_{\leq i} \in \mathcal{C}(v)) = \Pr(\mathbf{X}_{>i} = \mathbf{x}_{>i}|\mathbf{X}_{\leq i} \in \mathcal{C}(w)).$$

- The decision is based on a measure of similarity $\delta(v, w)$ between nodes v and w, and a fixed threshold, μ.
- v and w are called similar if $\delta(v, w) < \mu$: otherwise they are called dissimilar. Dissimilar nodes are not merged.

The algorithm

1. Start with the sample tree.
2. From level 1 to $p-1$:
   Repeatedly merge similar nodes until all the resulting nodes are pairwise dissimilar.
3. Merge all nodes at level p.

Similarity scores

- Ron et al proposed the similarity score

$$\delta_R(v,w) = \max_{k=i+1\dots p} \max_{\mathbf{x}_{i+1,\dots,k}}$$
$$|\hat{\Pr}(\mathbf{X}_{i+1,\dots,k} = \mathbf{x}_{i+1,\dots,k} | \mathbf{X}_{\leq i} \in \mathcal{C}(v)) - \hat{\Pr}(\mathbf{X}_{i+1,\dots,k} = \mathbf{x}_{i+1,\dots,k} | \mathbf{X}_{\leq i} \in \mathcal{C}(w))|$$

- We propose instead a score based on the penalized likelihood criterion

$$IC(\mathcal{A}) = -2\hat{\ell}(\mathcal{A}) + \alpha \dim(\mathcal{A}) \tag{8}$$

namely

$$\begin{aligned} \delta_{IC}(v,w) &= IC(\mathcal{A}_0) - IC(\mathcal{A}) \\ &= G^2 - \alpha k \end{aligned} \tag{9}$$

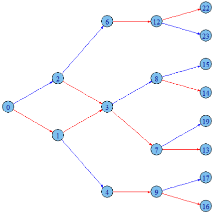
We set $\mu = 0$, so that two nodes are similar whenever merging them decreases the IC.

- Thus the selection algorithm seeks to minimize the IC.
- We are currently comparing the performance of this algorithm with the one in Beagle.

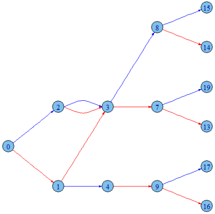
An example

Level	Node pair	G^2	k	δ_{IC}	Action
1	1,2	53.98	5	32.74	go to next level
2	3,4	20.78	3	8.03	
2	3,5	1.03	3	-11.71	
2	3,6	5.60	3	-7.14	
2	4,5	58.49	3	45.74	
2	4,6	0.36	1	-3.89	
2	5,6	7.43	3	-5.31	merge 5 into 3
2	3,4	61.36	3	48.62	
2	3,6	7.60	3	-5.15	
2	4,6	0.36	1	-3.89	merge 6 into 3
2	3,4	56.60	3	43.85	go to next level
3	7,8	2.88	1	-1.37	
3	7,9	0.05	1	-4.19	
3	8,9	5.40	1	1.15	merge 9 into 7
3	7,8	6.41	1	2.16	stop

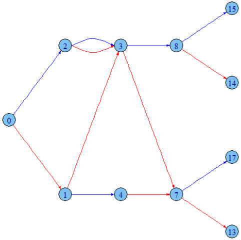
An example



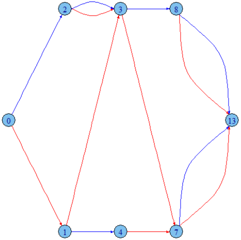
(a)



(b)



(c)



(d)

Outline

Introduction

Automata

Maximum likelihood estimation

State merging

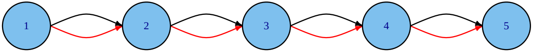
Hypothesis tests

Model selection

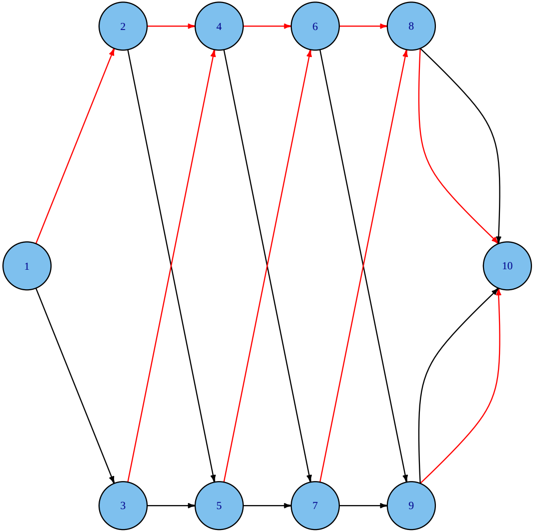
APFA equivalent to conventional Markov models

Summary and conclusion

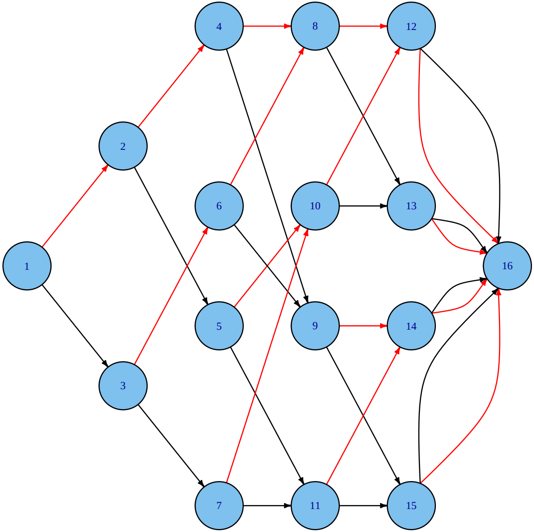
Independence



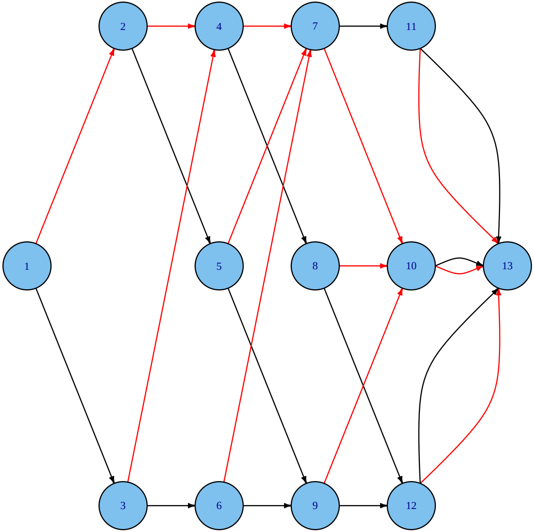
First order Markov



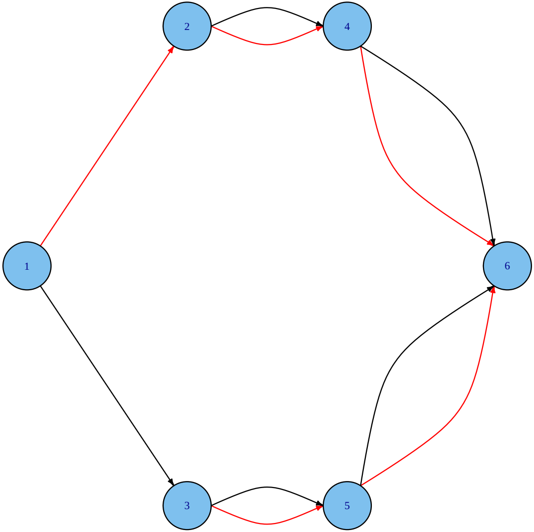
Second order Markov



Variable order Markov



Memory gap Markov



Outline

Introduction

Automata

Maximum likelihood estimation

State merging

Hypothesis tests

Model selection

APFA equivalent to conventional Markov models

Summary and conclusion

Summary and conclusion

- This talk has tried to describe APFA as statistical models.
- An APFA embodies a set of context-specific conditional independence relations, and may be represented as a directed multigraph.
- So it may be called a context-specific graphical model.
- APFA form a very rich class of models for discrete longitudinal data.
- We have shown how likelihood ratio tests may be constructed, and used this to modify the selection algorithm of Ron et al. (1998).
- We are preparing an R package to work with the models.