COMP90051 **Statistical Machine Learning**

Semester 2, 2015

Lecturer: Ben Rubinstein

9. PGM Statistical Inference

THE UNIVERSITY OF MELBOURNE

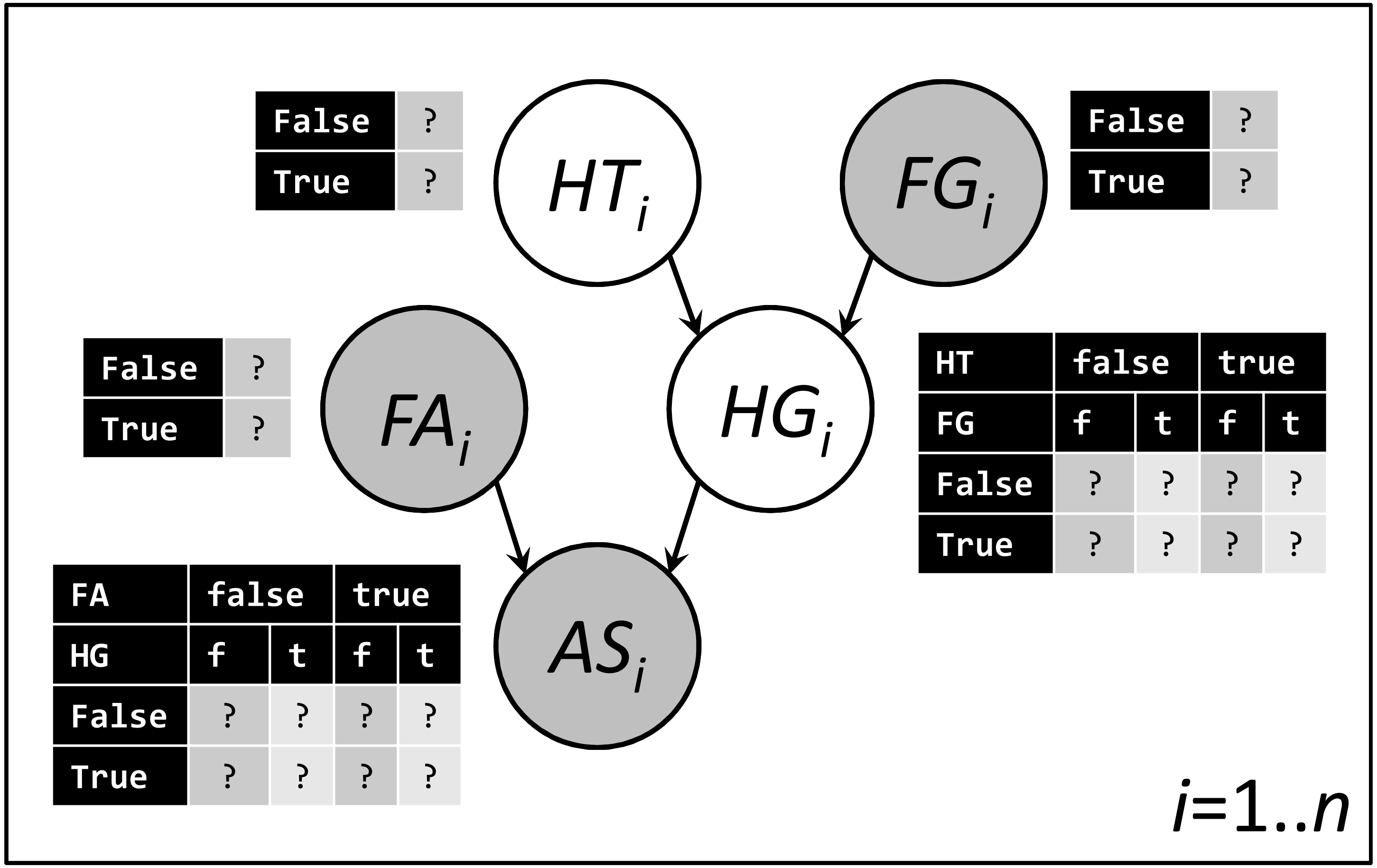
Statistical inference on PGMs

Learning from data – fitting probability tables to observations (eg as a frequentist; a Bayesian would just use probabilistic inference to update prior to posterior)

Where are we?

- Representation of joint distributions
  * PGMs encode conditional independence
- Examples
- Probabilistic inference
  * Computing other distributions from joint
  * Elimination, sampling algorithms
- Statistical inference
  * Learn parameters from data

Have PGM, Some observations, No tables…



False	?
True	?

False	?
True	?

False	?
True	?

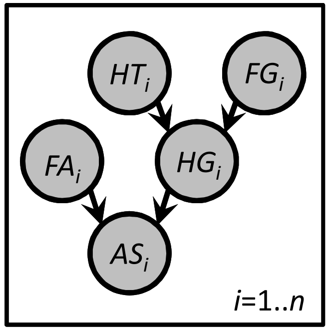
HT	false		true	
FG	f	t	f	t
False	?	?	?	?
True	?	?	?	?

FA	false		true	
HG	f	t	f	t
False	?	?	?	?
True	?	?	?	?

Fully-observed case is "easy"

- Max-Likelihood Estimator (MLE) says
  - If we observe *all* r.v.'s $\boldsymbol{X}$ in a PGM independently n times $\boldsymbol{x}_i$
  - Then maximise the *full* joint

$$\arg\max_{\theta\in\Theta} \prod_{i=1}^{n} \prod_j p\left(X^j = x_i{}^j | X^{parents(j)} = x_i{}^{parents(j)}\right)$$



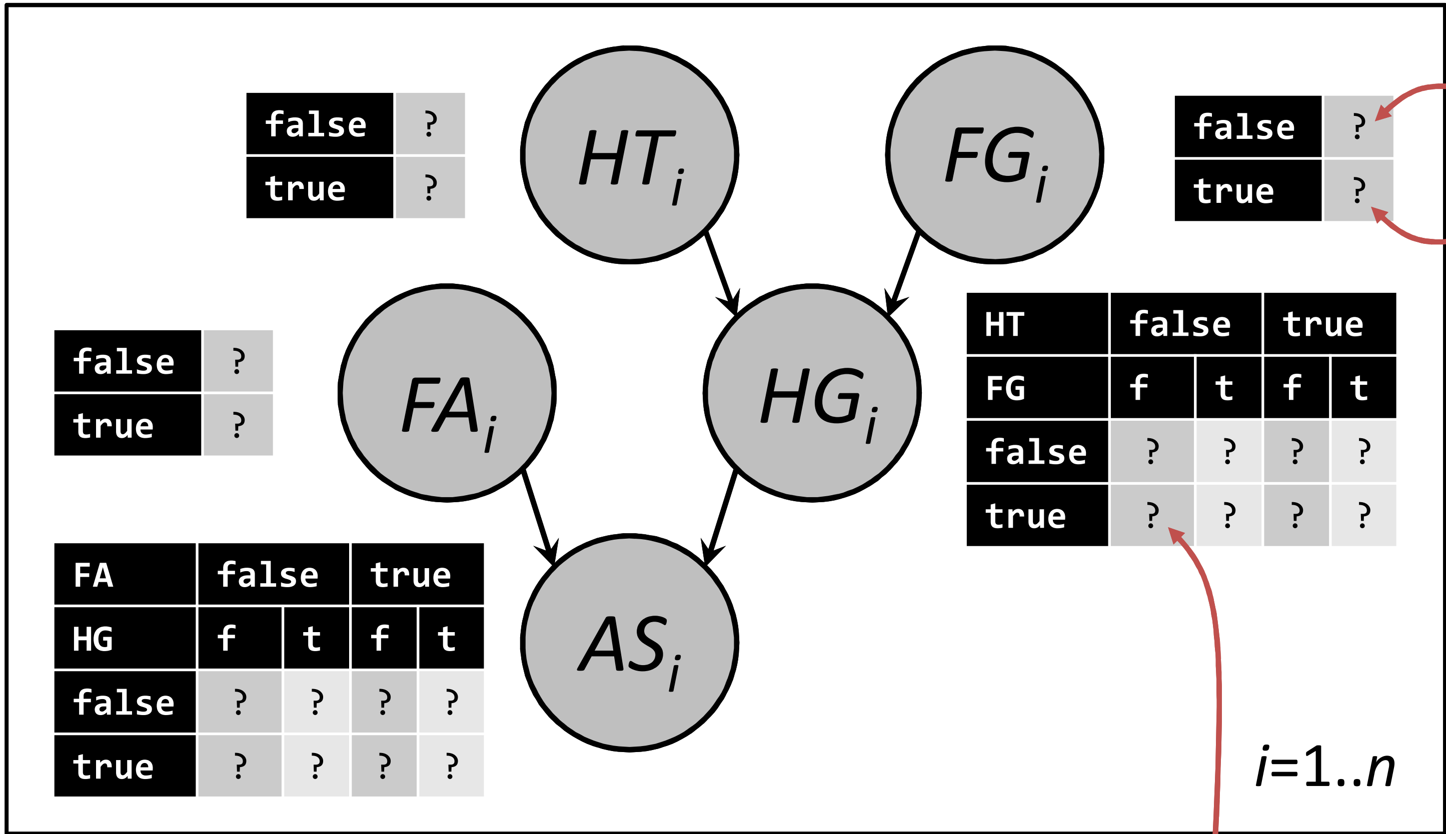
- Decomposes easily, leads to counts-based estimates
  - Maximise log-likelihood instead; becomes sum of logs

$$\arg\max_{\theta\in\Theta} \sum_{i=1}^{n} \sum_j \log p\left(X^j = xij | X^{parents(j)} = x_i{}^{parents(j)}\right)$$

  - Big maximisation of all parameters together, decouples into small independent problems

- Example is training a naïve Bayes classifier

Example: Fully-observed case



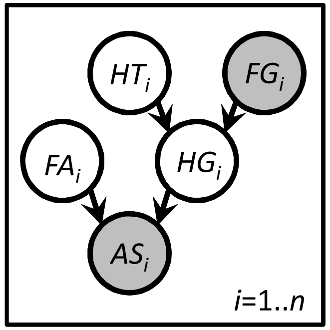
$$\frac{\#\{x_i | FG_i = false\}}{n}$$

$$\frac{\#\{x_i | FG_i = true\}}{n}$$

$$\frac{\#\{x_i | HG_i = true, HT_i = false, FG_i = false\}}{\#\{x_i | HT_i = false, FG_i = false\}}$$

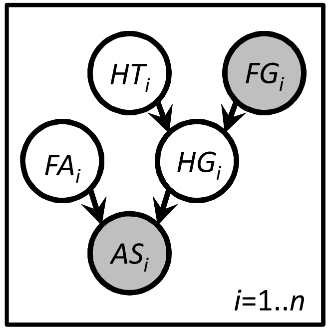
Presence of unobserved variables trickier

- But most PGMs you'll encounter will have latent, or unobserved, variables



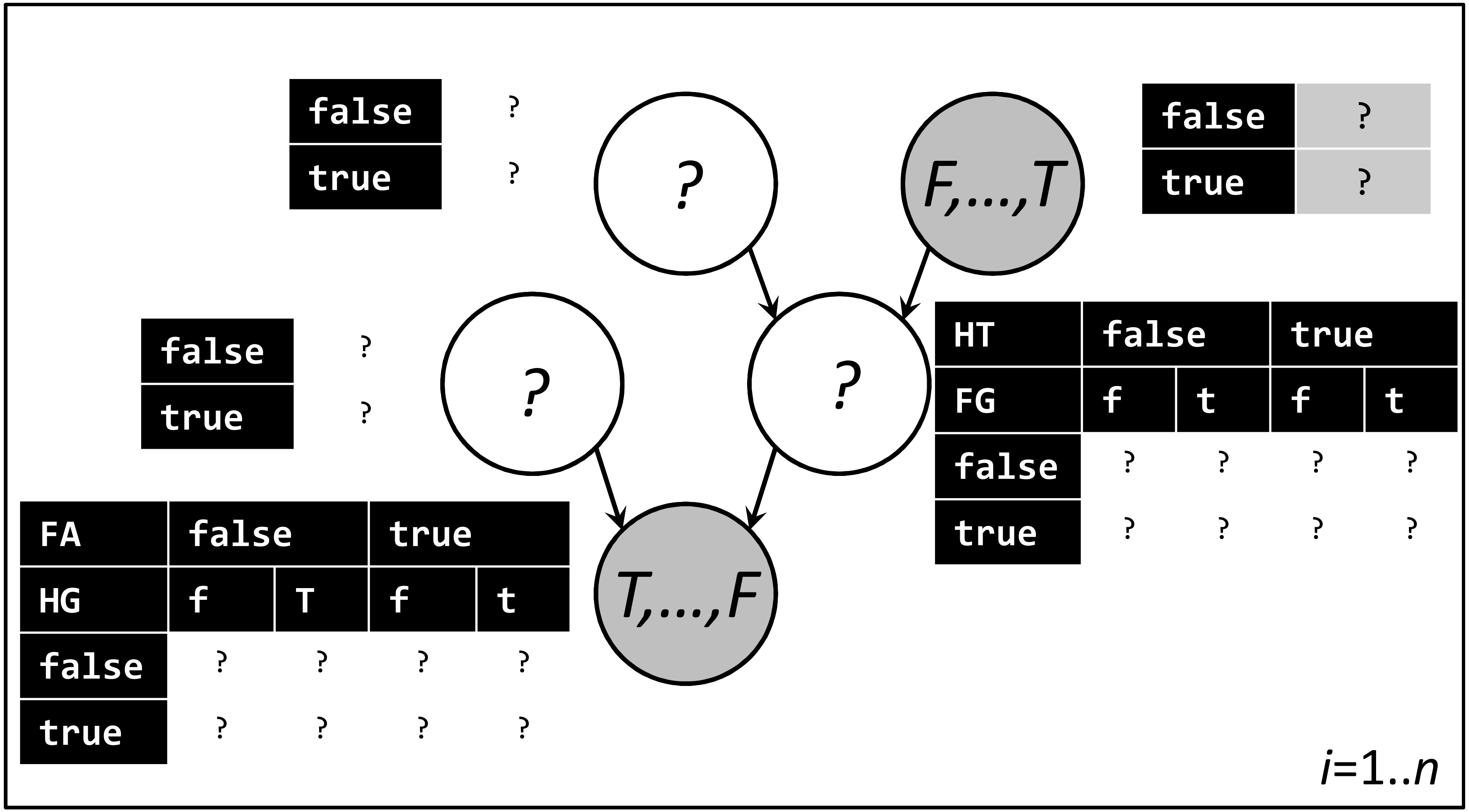
- What happens to the MLE?
  - Maximise likelihood of observed data only
  - Marginalise full joint to get to desired "partial" joint
  - $\arg\max_{\theta\in\Theta} \prod_{i=1}^{n} \sum_{\text{latent } j} \prod_j p\left(X^j = x_i{}^j | X^{parents(j)} = x_i{}^{parents(j)}\right)$
  - This won't decouple – oh-no's*!!*

Can we reduce partially-observed to fully?



- Rough idea
  - If we had guesses for the missing variables
  - We could employ MLE on fully-observed data
- With a bit more thought, could alternate between
  - Updating missing data
  - Updating probability tables/parameters
- This is the basis for training PGMs

Example: Partially-observed case



false	?
true	?

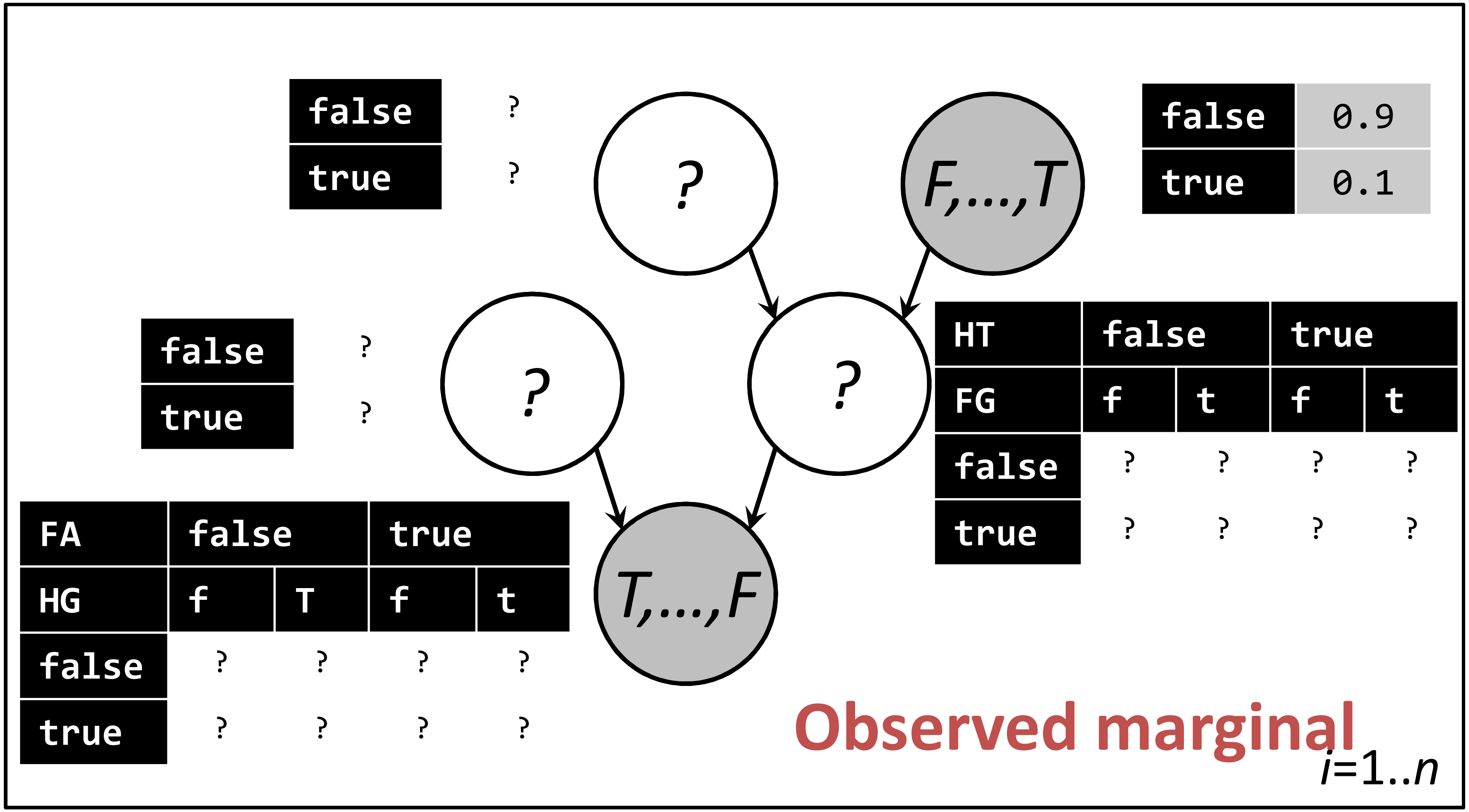
false	?
true	?

false	?
true	?

HT	false		true	
FG	f	t	f	t
false	?	?	?	?
true	?	?	?	?

FA	false		true	
HG	f	T	f	t
false	?	?	?	?
true	?	?	?	?

Example: Partially-observed case

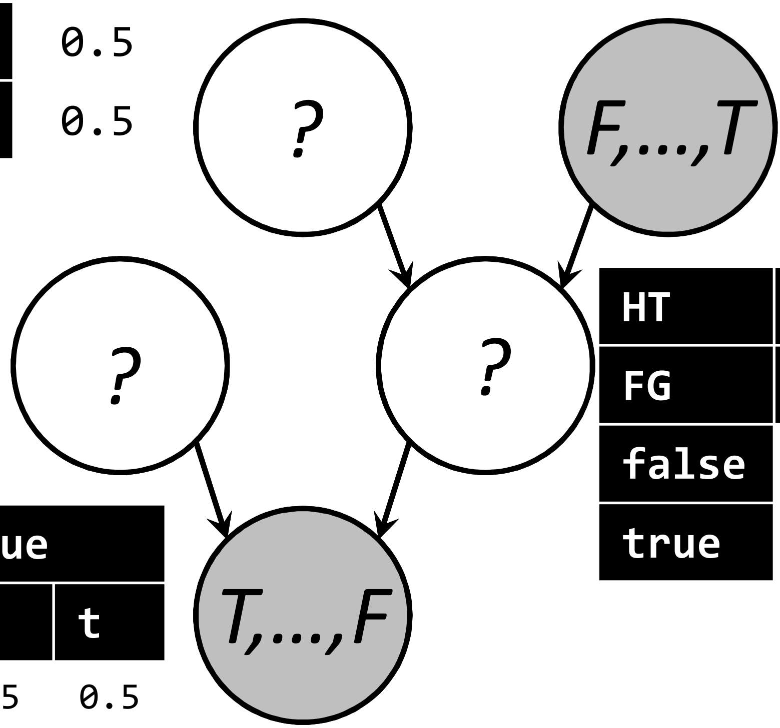


Example: Partially-observed case

false	0.5
true	0.5

false	0.5
true	0.5

false	0.9
true	0.1



HT	false		true	
FG	f	t	f	t
false	0.5	0.5	0.5	0.5
true	0.5	0.5	0.5	0.5

FA	false		true	
HG	f	t	f	t
false	0.5	0.5	0.5	0.5
true	0.5	0.5	0.5	0.5

Seed

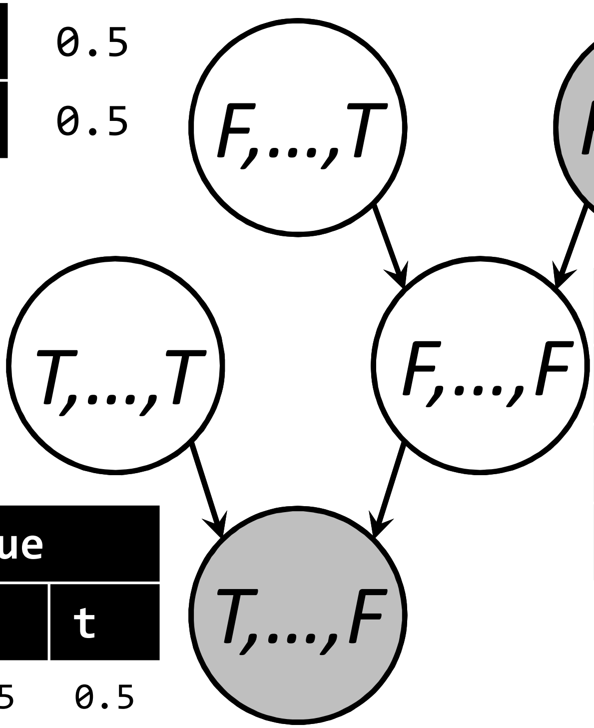
i=1..n

Example: Partially-observed case

false	0.5
true	0.5

false	0.9
true	0.1

false	0.5
true	0.5



HT	false		true	
FG	f	t	f	t
false	0.5	0.5	0.5	0.5
true	0.5	0.5	0.5	0.5

FA	false		true	
HG	f	t	f	t
false	0.5	0.5	0.5	0.5
true	0.5	0.5	0.5	0.5

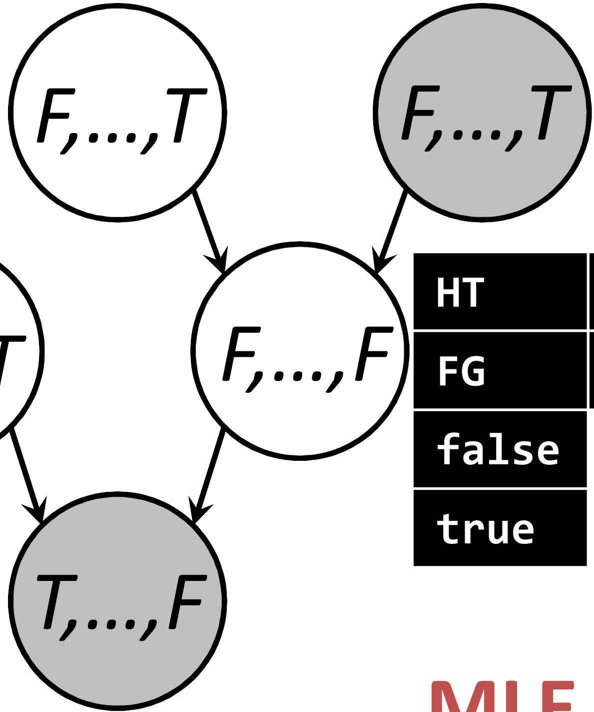
Missing data as expectation $i=1..n$

Example: Partially-observed case

false	0.7
true	0.3

false	0.9
true	0.1

false	0.6
true	0.4



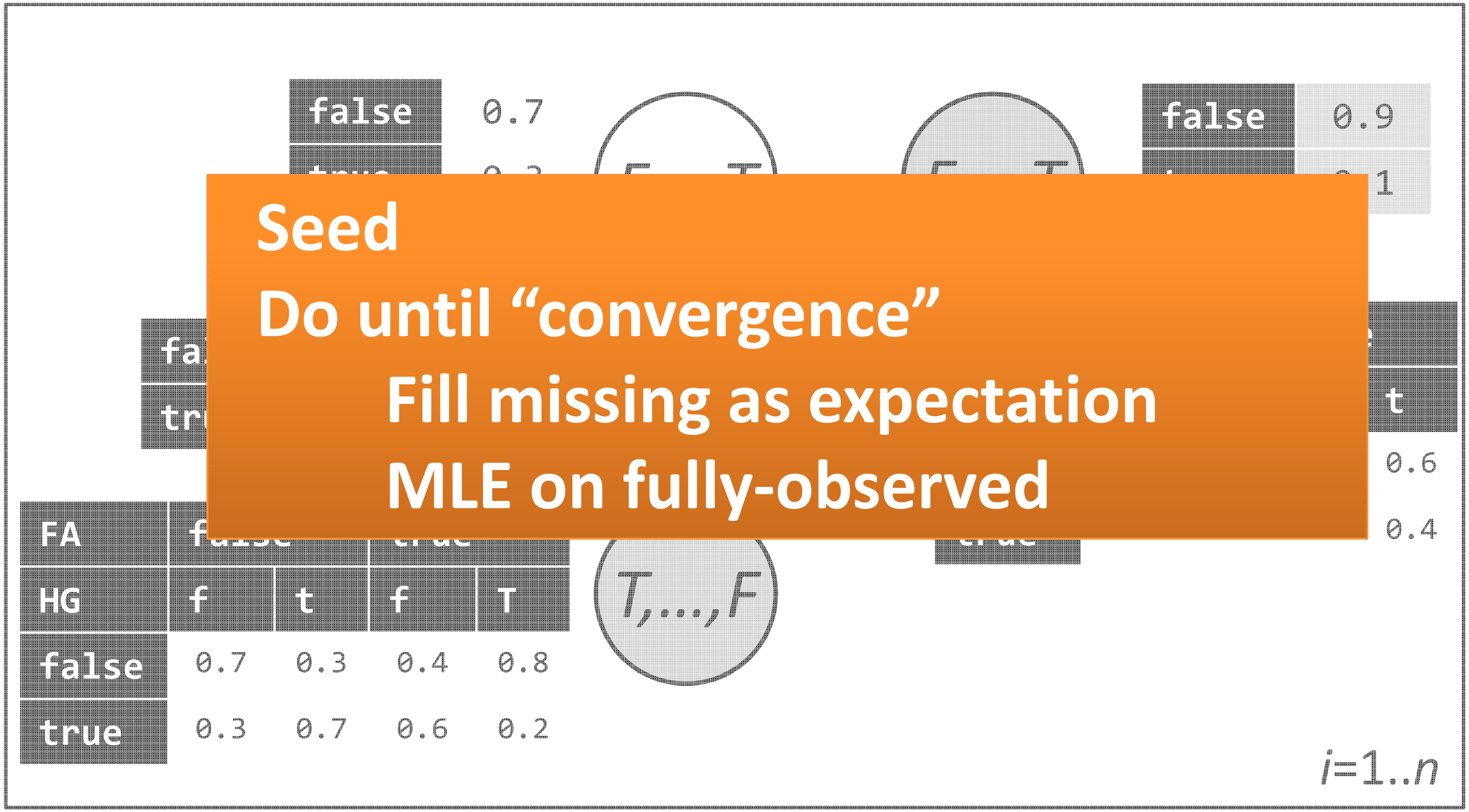
HT	false		true	
FG	f	t	f	t
false	0.7	0.4	0.3	0.6
true	0.3	0.6	0.7	0.4

FA	false		true	
HG	f	t	f	T
false	0.7	0.3	0.4	0.8
true	0.3	0.7	0.6	0.2

MLE on fully-observed

i=1..n

Example: Partially-observed case



Seed

Do until "convergence"

- **Fill missing as expectation**
- **MLE on fully-observed**

Expectation-Maximisation Algorithm

Seed parameters randomly

E-step: Complete unobserved data as ~~expectations (point estimates)~~ posterior distributions (prob inference)

M-step: Update parameters with MLE on the fully-observed data

Déjà vu?

Hard E-step

- K-means clustering
  - Randomly assign cluster centres
  - Repeat
    - Assign points to nearest clusters
    - Update cluster centres

- EM learning
  - Randomly seed parameters
  - Repeat
    - Expectations for missing variables
    - Update parameters via MLE

Soft E-step

- Assign distribution of point belonging to each cluster (e.g., 10% C1 20% C2 70% C3)

- Posteriors for missing variables given observed, current parameters

Summary

- Statistical inference on PGMs
  - What is it and why do we care?
  - Straight MLE for fully-observed data
  - EM algorithm for mixed latent/observed data