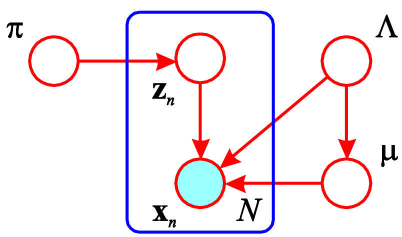
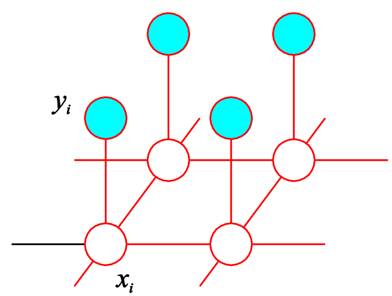
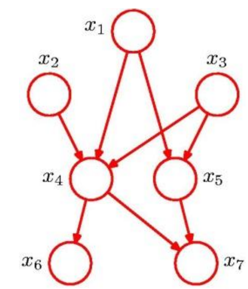
Graphical Models

Chris Bishop

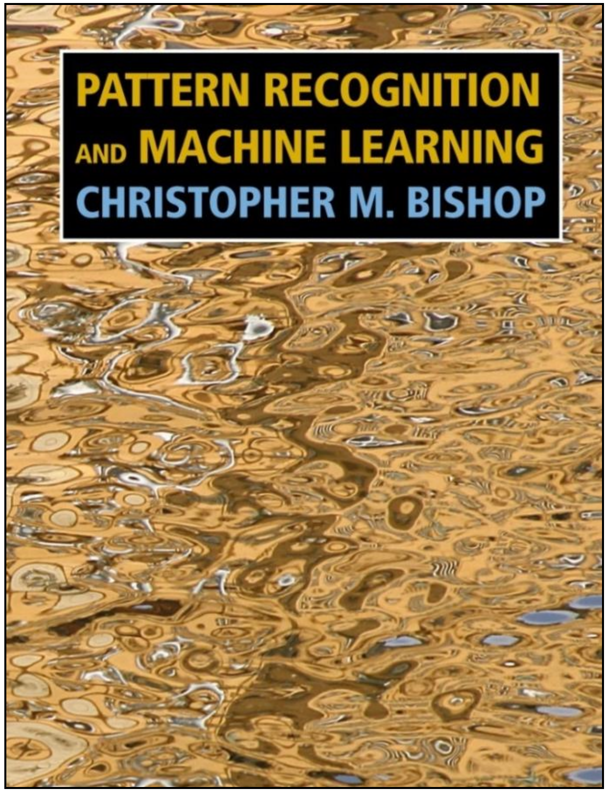
Microsoft Research Cambridge







Machine Learning Summer School 2013, Tübingen



http://research.microsoft.com/~cmbishop

Chapter 8: Graphical Models (PDF download)

PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A
MATHEMATICAL, PHYSICAL & ENGINEERING SCIENCES

Model-based machine learning

Christopher M. Bishop

Phil. Trans. R. Soc. A 2013 **371**,
doi: 10.1098/rsta.2012.0222

References This article cites 11 articles
http://rsta.royalsocietypublishing.org/content/371/1984/20120222.full.html#ref-list-1

EXiS Open Choice This article is free to access

Subject collections Articles on similar topics can be found in the following collections

artificial intelligence (7 articles)
pattern recognition (5 articles)
robotics (2 articles)

Email alerting service Receive free email alerts when new articles cite this article - sign up in the box at the top right-hand corner of the article or click here

To subscribe to *Phil. Trans. R. Soc. A* go to:
http://rsta.royalsocietypublishing.org/subscriptions

http://research.microsoft.com/~cmbishop

Please ask questions!

1. Introduction

Traditional machine learning

Markov random field

K-means clustering

RVM

Gaussian mixture

logistic regression

Kalman filter

random forest

HMM

principal components

neural networks

deep networks

support vector machines

kernel PCA

ICA

linear regression

Boltzmann machines

Radial basis functions

Gaussian process

decision trees

factor analysis

KINECT™

for XBOX 360


KINECT
for XBOX 360
Microsoft
Research

10 March 2011 Last updated at 06:09 ET

Microsoft Kinect 'fastest-selling device on record'

Microsoft has sold more than 10 million Kinect sensor systems since launch on 4 November, and - according to Guinness World Records - is the fastest-selling consumer electronics device on record.

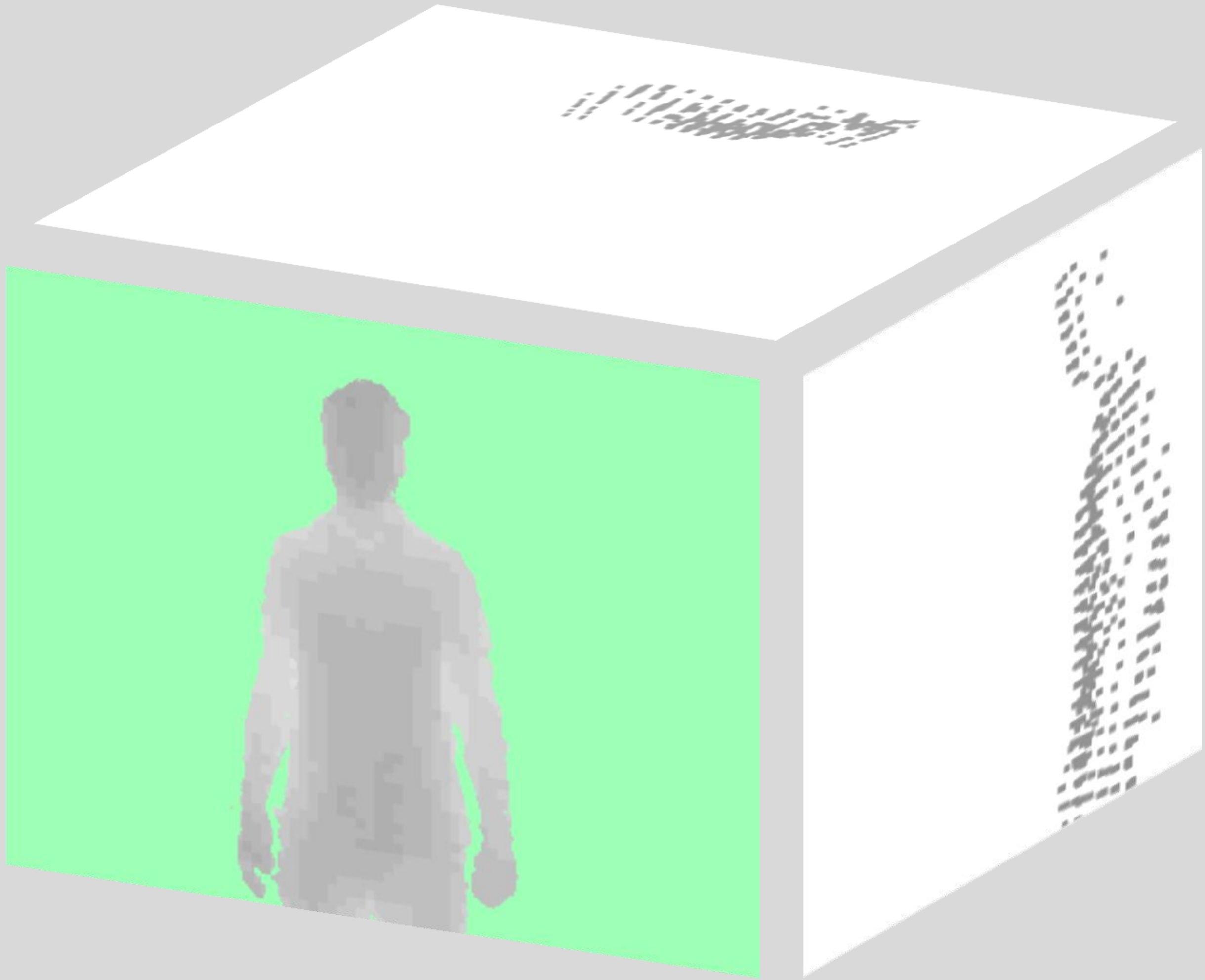
The popularity of the Kinect has helped to boost sales of games, Microsoft says

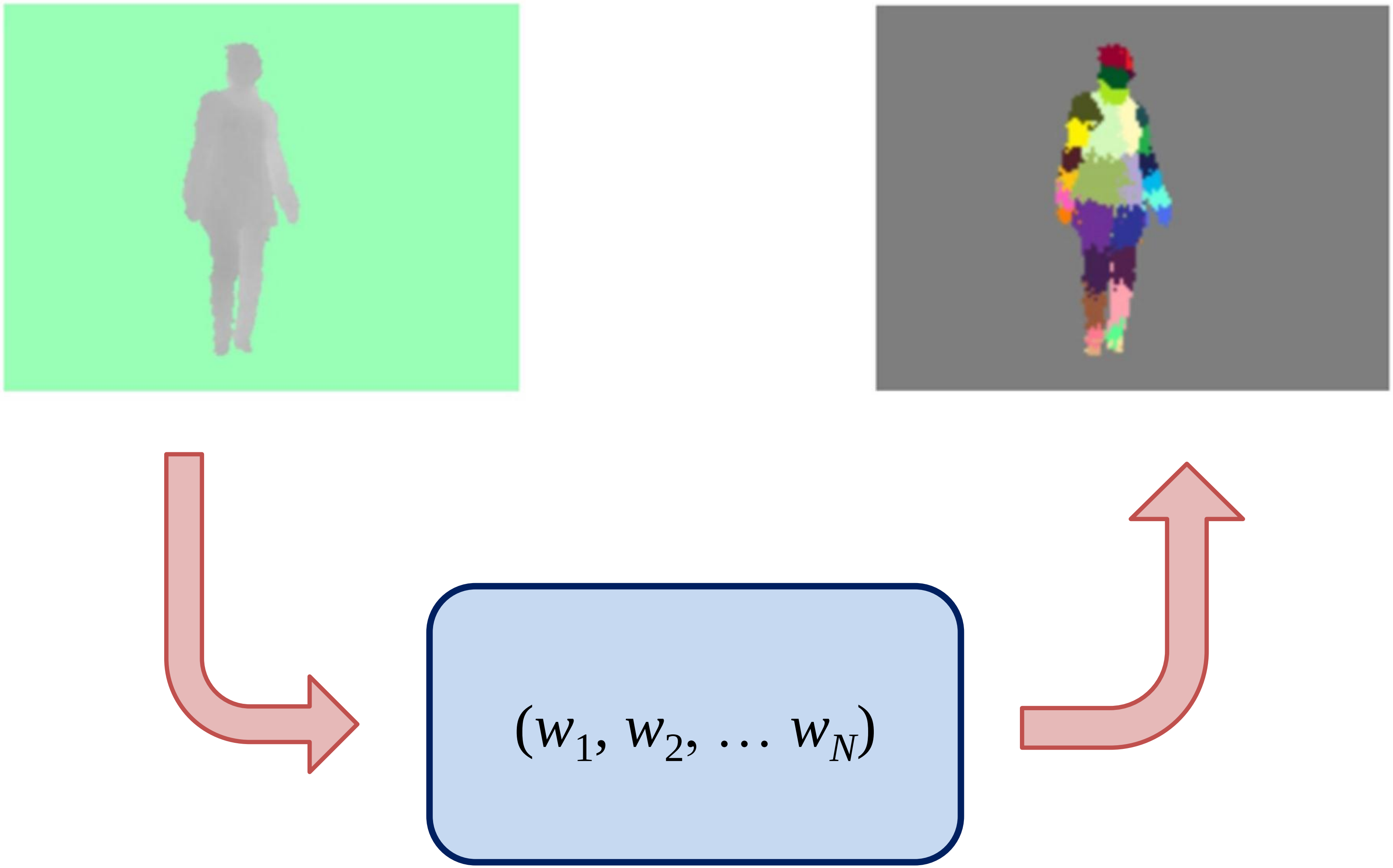
The sales figures outstrip those of both Apple's iPhone and iPad when launched, Guinness said.

Kinect is an infrared camera add-on for Microsoft's Xbox 360 games console that allows it to track body movements.

Related Stories

KINECT™ for XBOX 360

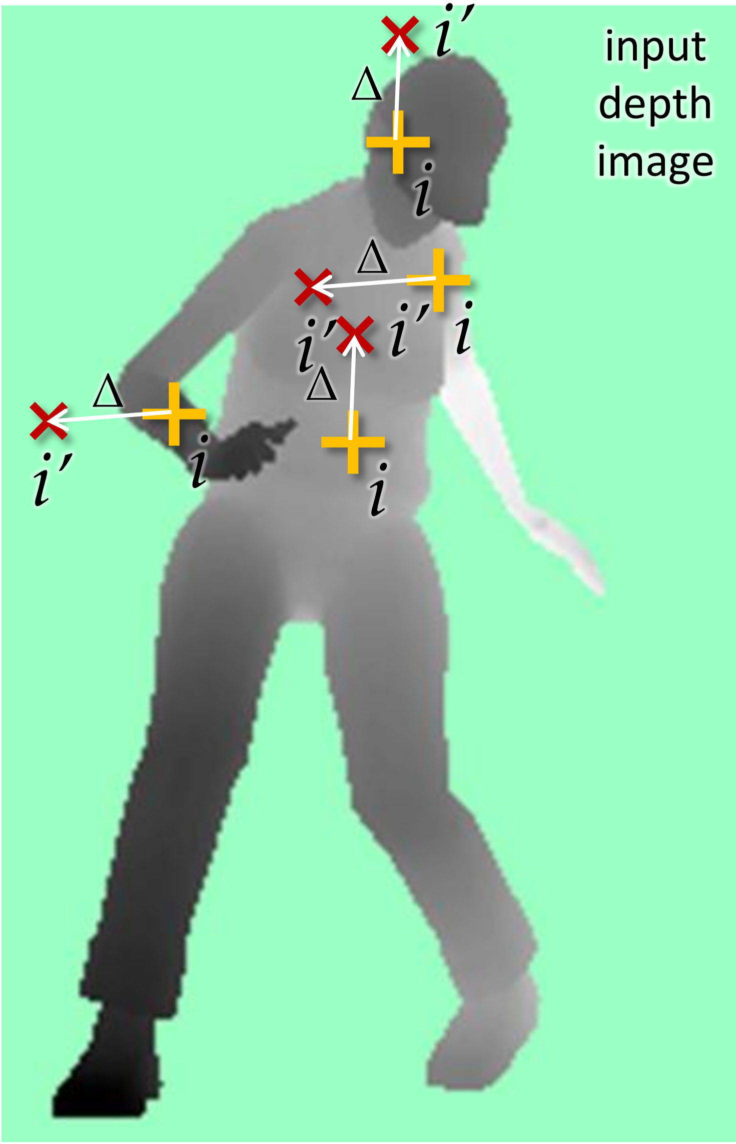


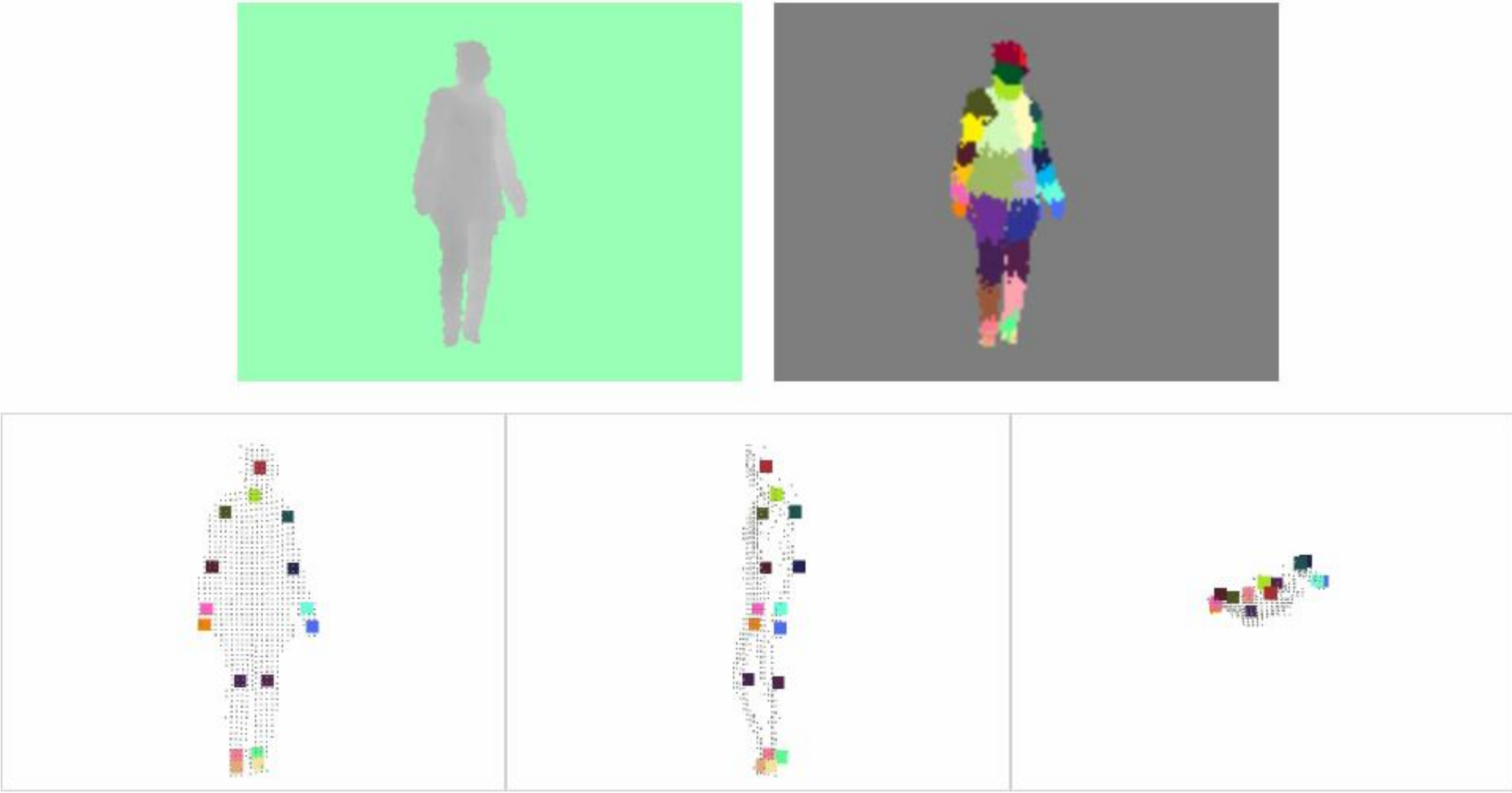

$(w_1, w_2, \ldots w_N)$

Fast depth image features

Depth comparisons:

- $f(\mathbf{x}_i ; \Delta) = d(\mathbf{x}_i) - d(\mathbf{x}_{i'})$
- where $\mathbf{x}_{i'} = \mathbf{x}_i + \Delta / d(\mathbf{x}_i)$





2. Model-based Machine Learning

Model-based machine learning

Goal:
A *single* development framework which supports the creation of a wide range of bespoke models

Traditional:

“how do I map my problem into standard tools”?

Model-based:

“what is the model that represents my problem”?

Potential benefits of MBML

Models optimised for each new application

Transparent functionality

- Models expressed as compact code
- Community of model builders

Segregate model from training/inference code

Newcomers learn one modelling environment

Does the “right thing” automatically

Intelligent software

Goal: software that can adapt, learn, and reason

Player skill → **Game result**

Movie preferences → **Ratings**

Words → **Ink**

Can be described by a *model*

Intelligent software

Goal: software that can adapt, learn, and reason

Player skill ← **Game result**

Movie preferences ← **Ratings**

Words ← **Ink**

Reasoning backwards

3. Uncertainty

Handling uncertainty

We are uncertain about a player's skill

Each result provides relevant information

But we are never completely certain

How can we compute with uncertainty in a principled way?

Uncertainty everywhere

Which movie should the user watch next?
Which word did the user write?
What did the user say?
Which web page is the user trying to find?
Which link will user click on?
What kind of product does the user wish to buy?
Which gesture is the user making?
Many others …

Probability

Limit of infinite number of trials

Quantification of uncertainty

60% **40%**

Movie Recommender Demo

Matchbox

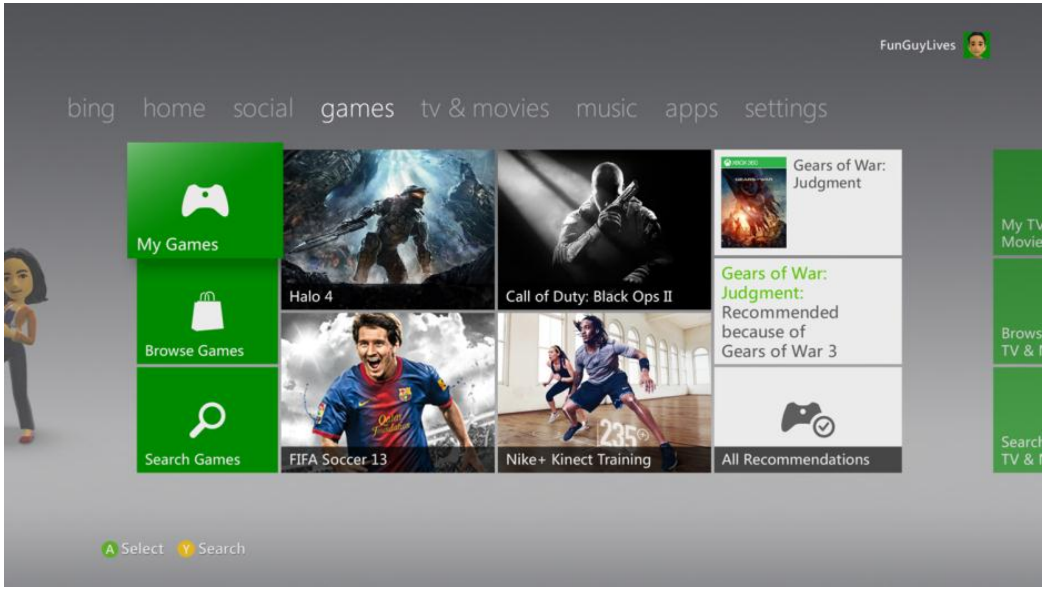
infer.net

Xbox Live Recommendation

Over 50M users

Serves more than 100M requests per day

Spans verticals: games, TV programmes, movies



4. Probabilities

A murder mystery

A fiendish murder has been committed

Whodunit?

There are two suspects:

- the **Butler**
- the **Cook**

There are three possible murder weapons:

- a butcher’s **Knife**
- a **Pistol**
- a fireplace **Poker**

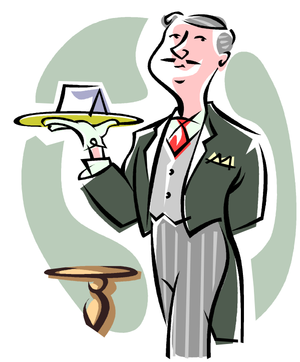
Prior distribution

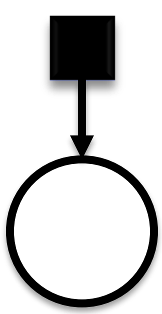
Butler has served family well for many years
Cook hired recently, rumours of dodgy history

P(Culprit = **Butler**) = 20%

P(Culprit = **Cook**) = 80%

Probabilities add to 100%





P(Culprit)

Culprit = {**Butler**, **Cook**}

This is called a *factor graph*
(we'll see why later)

Conditional distribution

Butler is ex-army, keeps a gun in a locked drawer

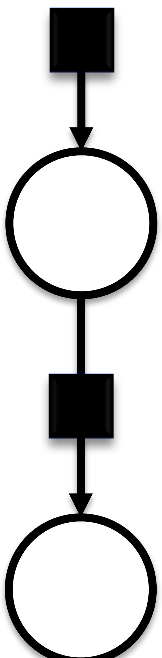
Cook has access to lots of knives

Butler is older and getting frail

	Pistol	Knife	Poker	
Cook	5%	65%	30%	**= 100%**
Butler	80%	10%	10%	**= 100%**

$P(\text{Weapon} \mid \text{Culprit})$

Factor graph



Prior distribution

P(Culprit)

Culprit = {**Butler**, **Cook**}

Conditional distribution

P(Weapon | Culprit)

Weapon = {**Pistol**, **Knife**, **Poker**}

Joint distribution

What is the probability that the **Cook** committed the murder using the **Pistol**?

P(Culprit = **Cook**) = 80%

P(Weapon = **Pistol** | Culprit = **Cook**) = 5%

P(Weapon = **Pistol** , Culprit = **Cook**) = 80% x 5% = 4%

Likewise for the other five combinations of Culprit and Weapon

Joint distribution

	Pistol	Knife	Poker
Cook	4%	52%	24%
Butler	16%	2%	2%

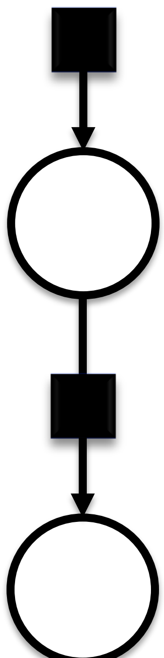
= 100%

P(Weapon, Culprit) = P(Weapon | Culprit) P(Culprit)

$$\boxed{P(x,y) = P(y|x)P(x)}$$

Product rule

Factor graphs



P(Culprit)

Culprit = {**Butler**, **Cook**}

P(Weapon | Culprit)

Weapon = {**Pistol**, **Knife**, **Poker**}

Generative model

P(Weapon, Culprit) = P(Weapon | Culprit) P(Culprit)

Generative viewpoint

Murderer	Weapon
Cook	Knife
Butler	Knife
Cook	Pistol
Cook	Poker
Cook	Knife
Butler	Pistol
Cook	Poker
Cook	Knife
Butler	Pistol
Cook	Knife
…	…

Marginal distribution of Culprit

	Pistol	Knife	Poker	
Cook	4%	52%	24%	**= 80%**
Butler	16%	2%	2%	**= 20%**

$$P(x) = \sum_{y} P(x, y)$$

Sum rule

Marginal distribution of Weapon

	Pistol	Knife	Poker
Cook	4%	52%	24%
Butler	16%	2%	2%
	= 20%	**= 54%**	**= 26%**

$$P(x) = \sum_{y} P(x, y)$$

Sum rule

Posterior distribution

We discover a **Pistol** at the scene of the crime

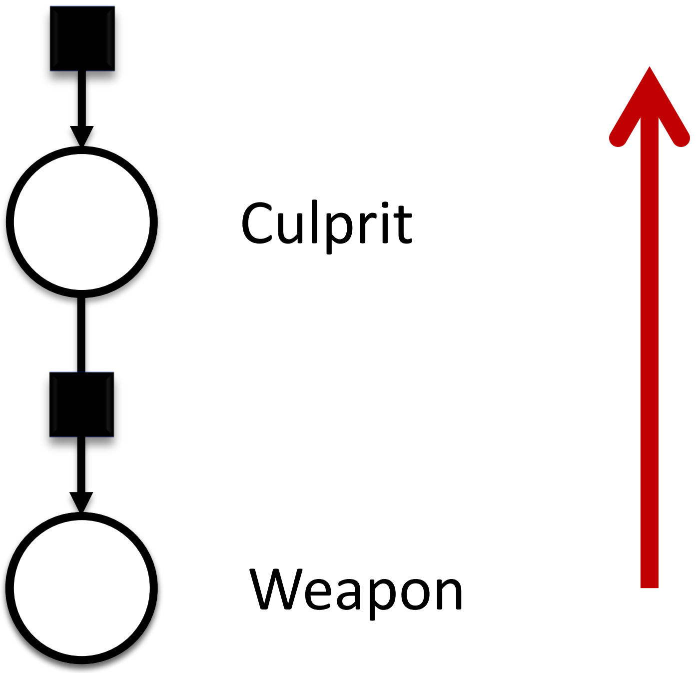
	Pistol	Knife	Poker	
Cook	4%	~~52%~~	~~24%~~	= 20%
Butler	16%	~~2%~~	~~2%~~	= 80%

This looks bad for the Butler!

Generative viewpoint

Murderer	Weapon
~~Cook~~	~~Knife~~
~~Butler~~	~~Knife~~
Cook	Pistol
~~Cook~~	~~Poker~~
~~Cook~~	~~Knife~~
Butler	Pistol
~~Cook~~	~~Poker~~
~~Cook~~	~~Knife~~
Butler	Pistol
~~Cook~~	~~Knife~~
…	…

Reasoning backwards



Bayes' theorem

$$P(x, y) = P(y|x)P(x)$$

likelihood

prior

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

posterior

Prior – belief before making a particular obs.

Posterior – belief after making the obs.

Posterior is the prior for the next observation

– Intrinsically incremental

Two views of probability

Frequency: limit of infinite number of trials

Bayesian: quantification of uncertainty

The Rules of Probability

Sum rule

$$P(x) = \sum_y P(x, y)$$

Product rule

$$P(x, y) = P(y|x)P(x)$$

Bayes' theorem

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

Denominator

$$P(x) = \sum_y P(x|y)P(y)$$

5. Directed Graphs

Probabilistic Graphical Models

Combine probability theory with graphs

- ✓ new insights into existing models
- ✓ framework for designing new models
- ✓ Graph-based algorithms for calculation and computation (c.f. Feynman diagrams in physics)
- ✓ efficient software implementation

Three types of graphical model

Directed graphs

– useful for designing models

Undirected graphs

– good for some domains, e.g. computer vision

Factor graphs

– useful for inference and learning

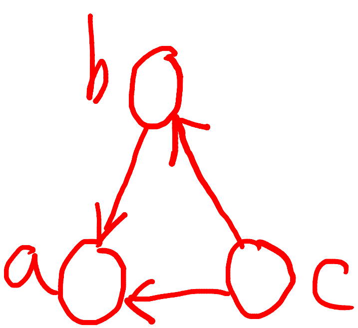
Decomposition

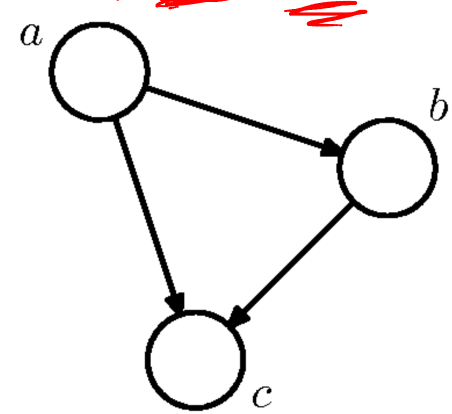
Consider an arbitrary joint distribution

$$p(a, b, c)$$

By successive application of the product rule:

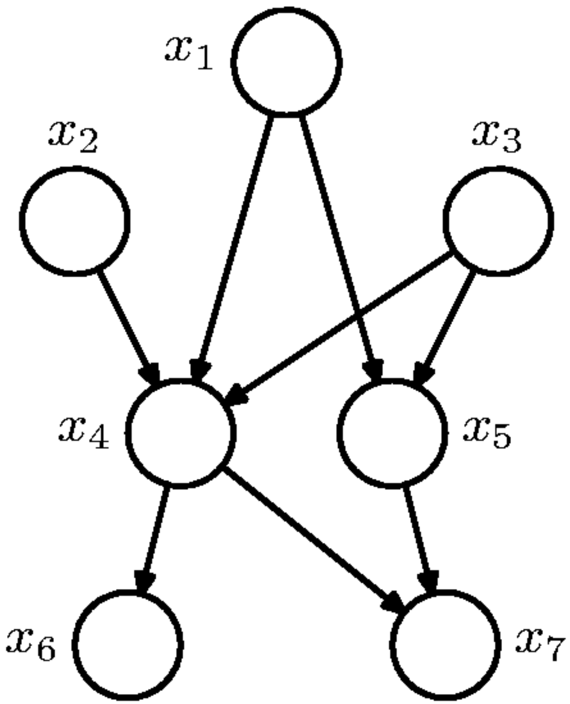
$$p(a,b,c) = p(a|b,c)\,p(b,c) = p(a|b,c)\,p(b|c)\,p(c)$$





Directed Graphs

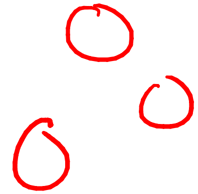
$$p(x_1, \ldots x_7) = p(x_1)p(x_2)p(x_3) \cdot p(x_4|x_1,x_2,x_3) \cdot p(x_5|x_1,x_3) \cdot p(x_6|x_4)\,p(x_7|x_4,x_5)$$

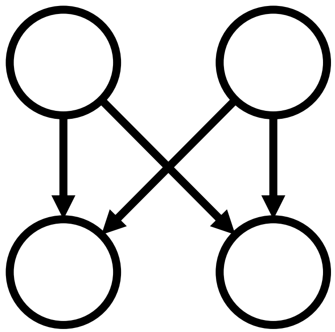


$$p(x_1 \ldots x_N) = \prod_{i=1}^{N} p(x_i|\pi_i)$$

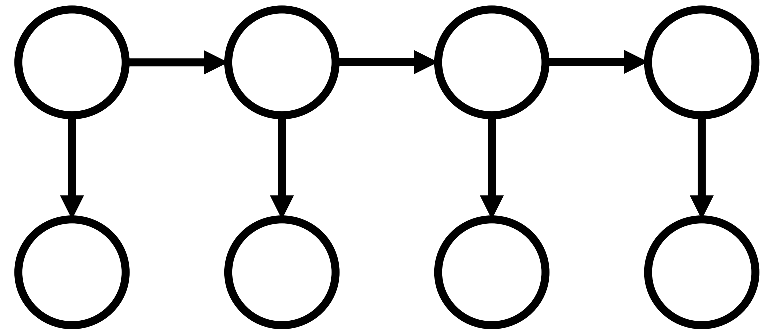
Arrows may indicate causal relationships

Special cases





PCA, ICA,
factor analysis,
linear regression,
logistic regression,
mixture models



Kalman filters,
hidden Markov models

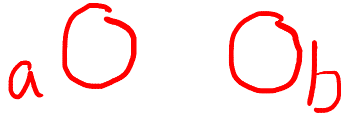
$$p(x_1, \ldots x_N) = \prod_{i=1}^{N} p(x_i)$$

We’re hiring!

Interns, Postdocs, Researchers, Developers

6. Conditional Independence

Conditional Independence



$$p(a,b) = p(a)\,p(b)$$

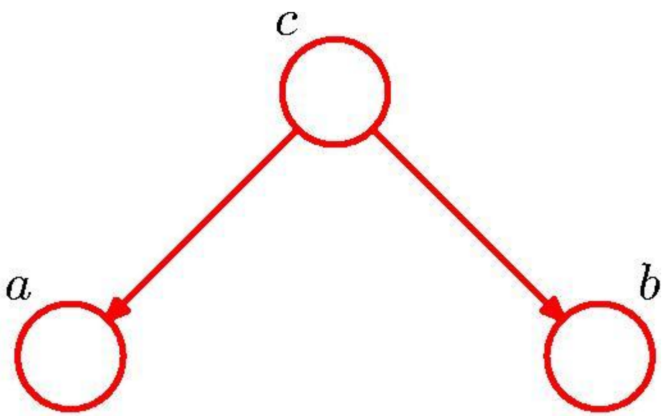
$$p(a|b) = \frac{p(a,b)}{p(b)} = p(a)$$

$$p(a,b|c) = p(a|b,c)\,p(b|c) = p(a|c)\,p(b|c)$$

$$a \perp\!\!\!\perp b \mid c$$

Conditional Independence: Example 1

$$p(a,b,c) = p(c)\,p(a|c)\,p(b|c)$$



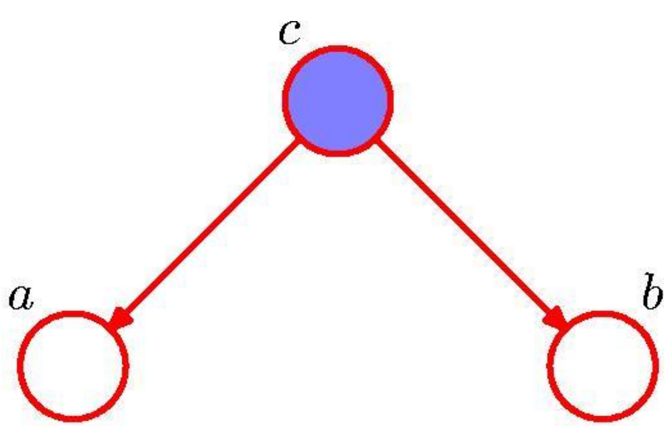
$$a \perp\!\!\!\perp b \mid \emptyset \ ?$$

$$p(a,b) = \sum_c p(a,b,c) = \sum_c p(c)\,p(a|c)\,p(b|c)$$

$$\neq p(a)\,p(b)$$

$$\boxed{a \not\!\perp\!\!\!\perp b \mid \emptyset}$$

Conditional Independence: Example 1



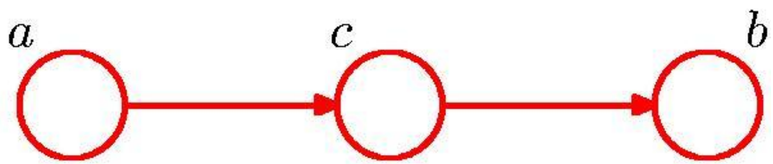
$$p(a,b \mid c) = \frac{p(a,b,c)}{p(c)}$$

$$= \frac{\cancel{p(c)}\, p(a \mid c)\, p(b \mid c)}{\cancel{p(c)}}$$

$$= p(a \mid c)\, p(b \mid c)$$

$$\boxed{a \perp\!\!\!\perp b \mid c}$$

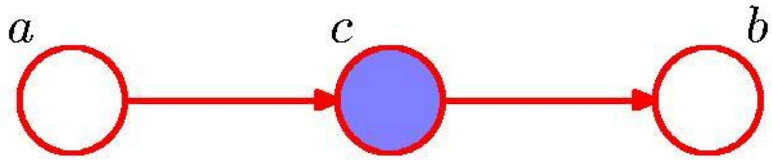
Conditional Independence: Example 2



$$p(a,b,c) = p(a)\,p(c|a)\,p(b|c)$$

$$a \not\perp\!\!\!\perp b \mid \emptyset$$

Conditional Independence: Example 2

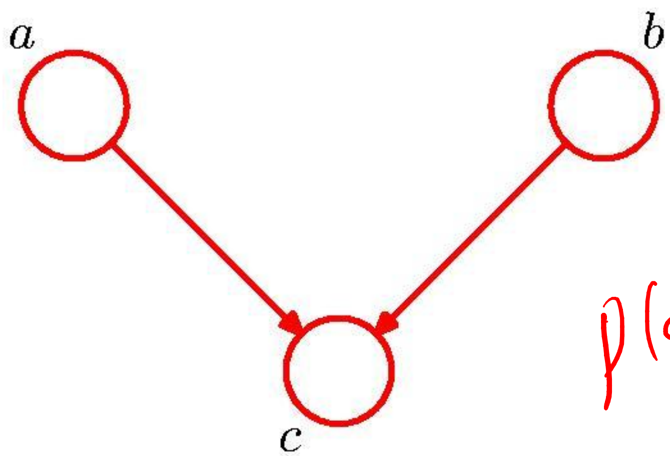


$a \perp\!\!\!\perp b \mid c$

Conditional Independence: Example 3

$$p(a,b,c) = p(a)p(b)p(c|a,b)$$

$$a \perp\!\!\!\perp b \mid \emptyset \; ?$$



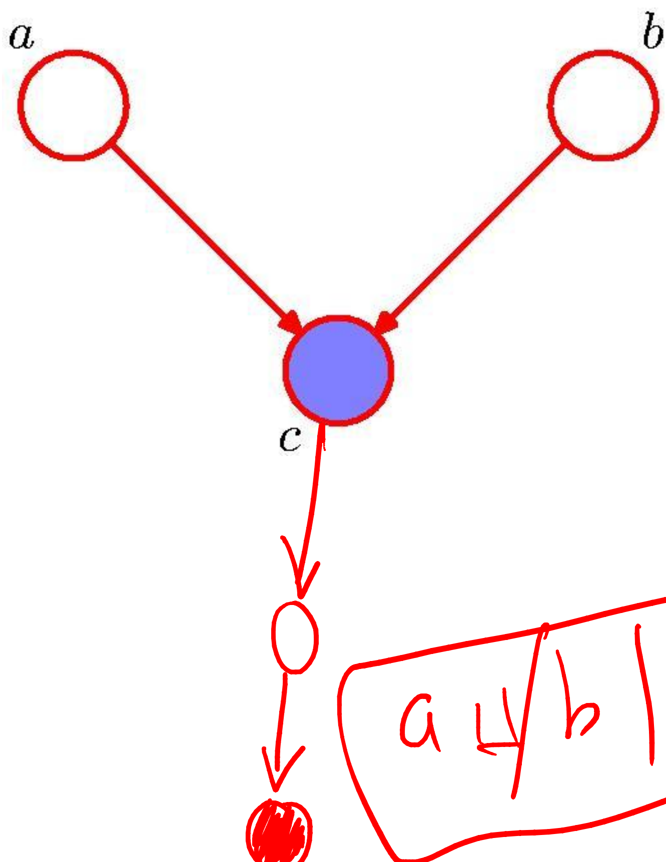
$$p(a,b) = \sum_c p(a,b,c)$$

$$= p(a)p(b)\sum_c p(c|a,b)$$

$$= p(a)p(b)$$

$$\boxed{a \perp\!\!\!\perp b \mid \emptyset}$$

Conditional Independence: Example 3



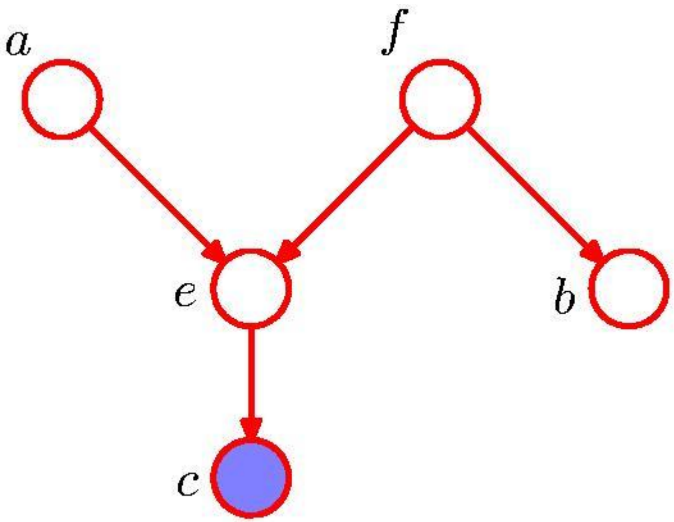
$$p(a,b|c) = \frac{p(a,b,c)}{p(c)}$$

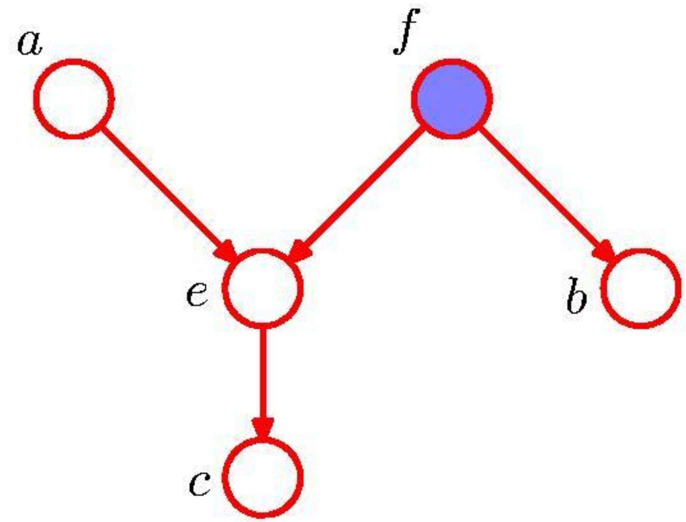
$$= \frac{p(a)\,p(b)\,p(c|a,b)}{p(c)}$$

$$\neq p(a|c)\,p(b|c)$$

$$\boxed{a \not\perp\!\!\!\perp b \mid c}$$

D-separation





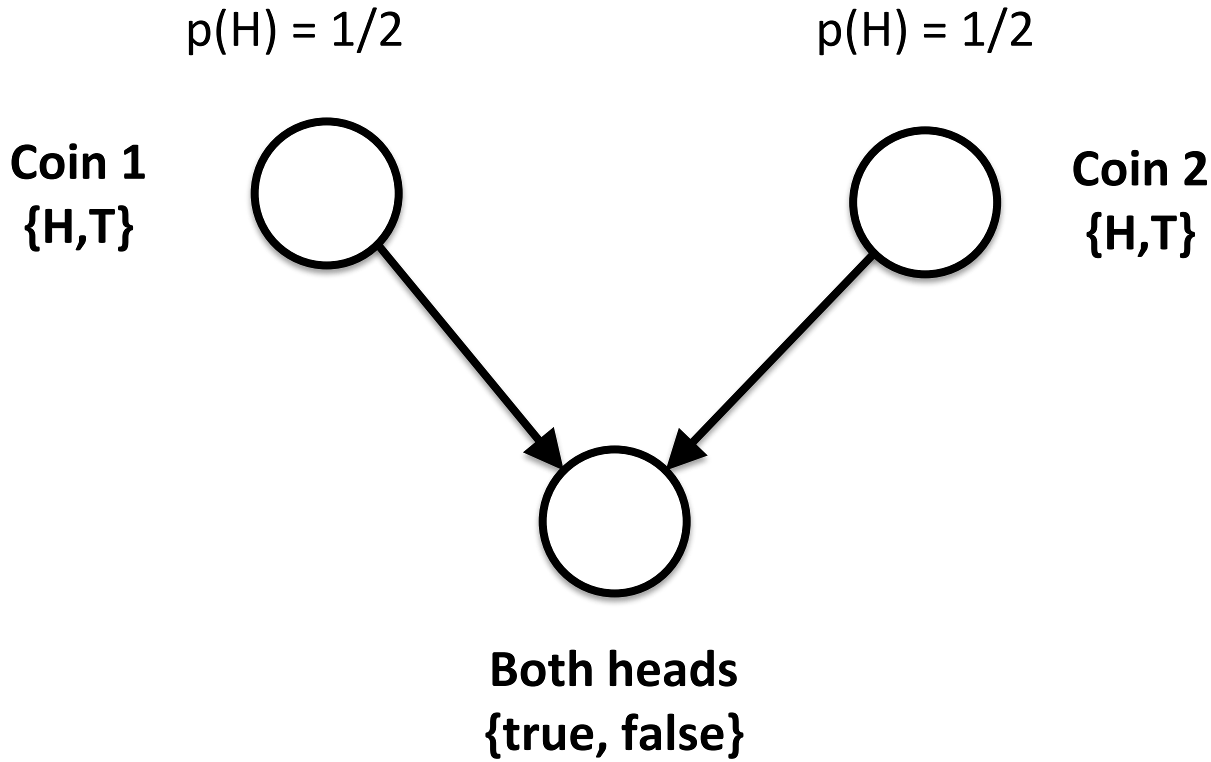
$a \perp\!\!\!\perp b \mid c$?

✗

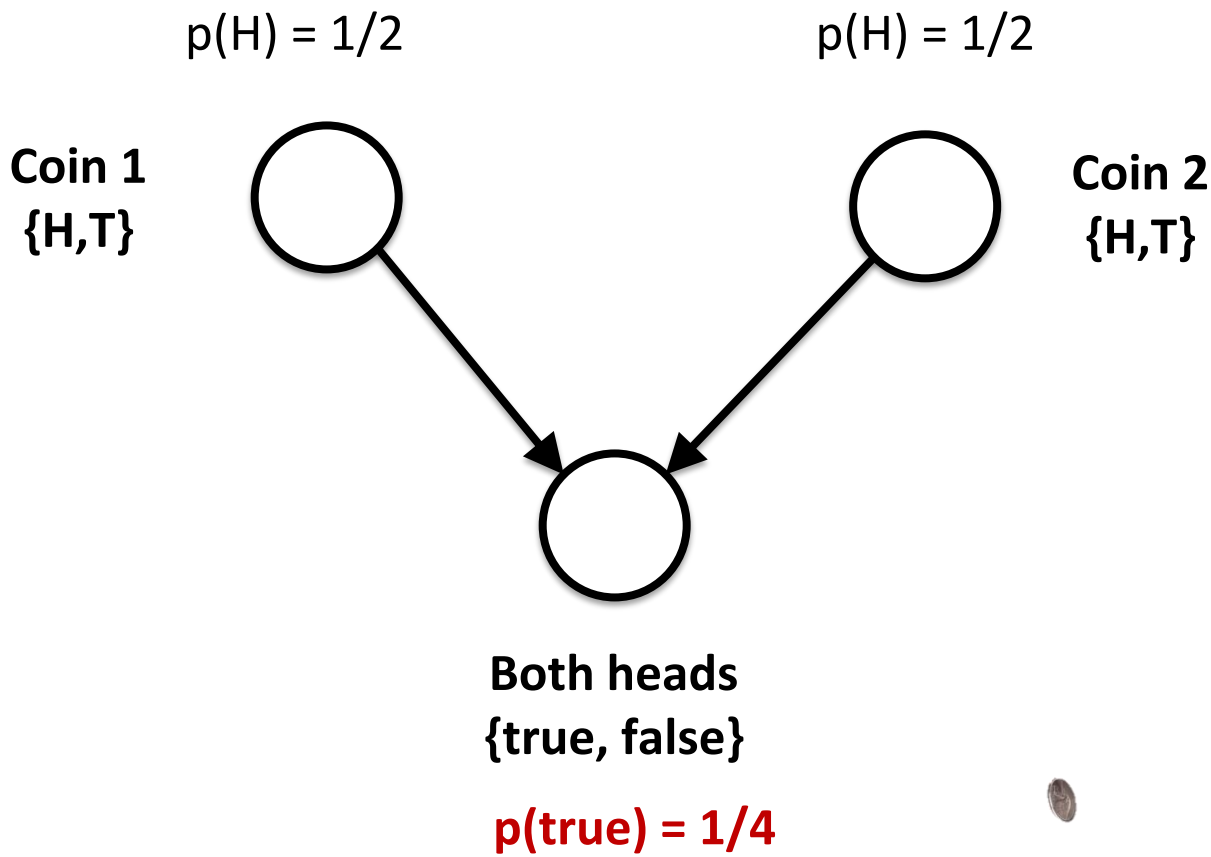
$a \perp\!\!\!\perp b \mid f$

✓

Two coins



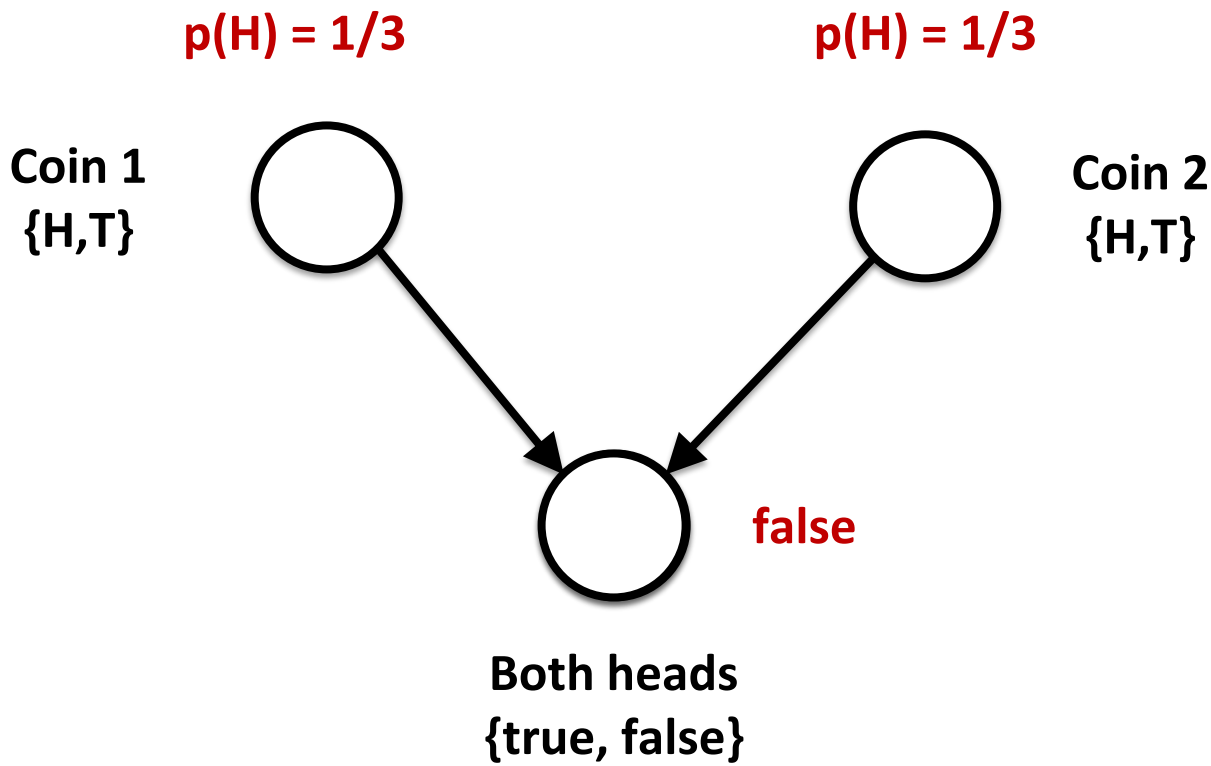
What is the probability of two heads?



Generative model

Coin 1	T	T	H	H
Coin 2	T	H	T	H
Both heads	false	false	false	true

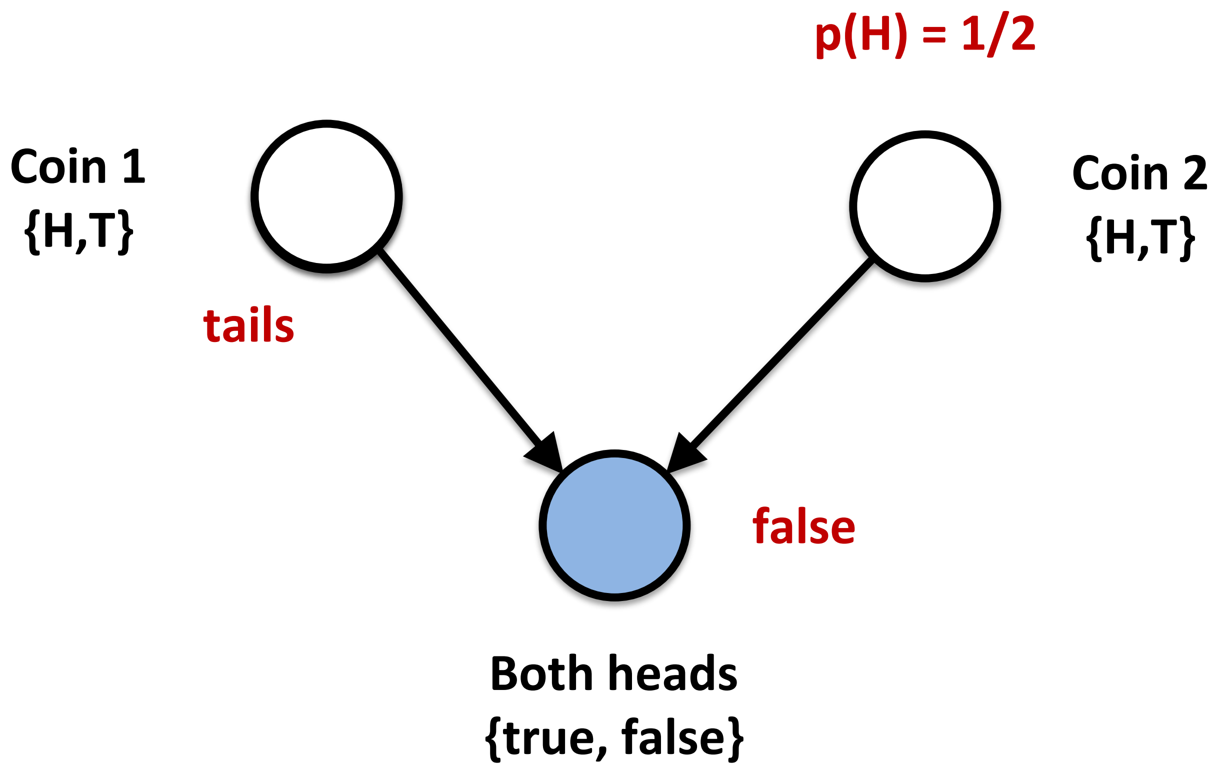
Reasoning backwards



Inference

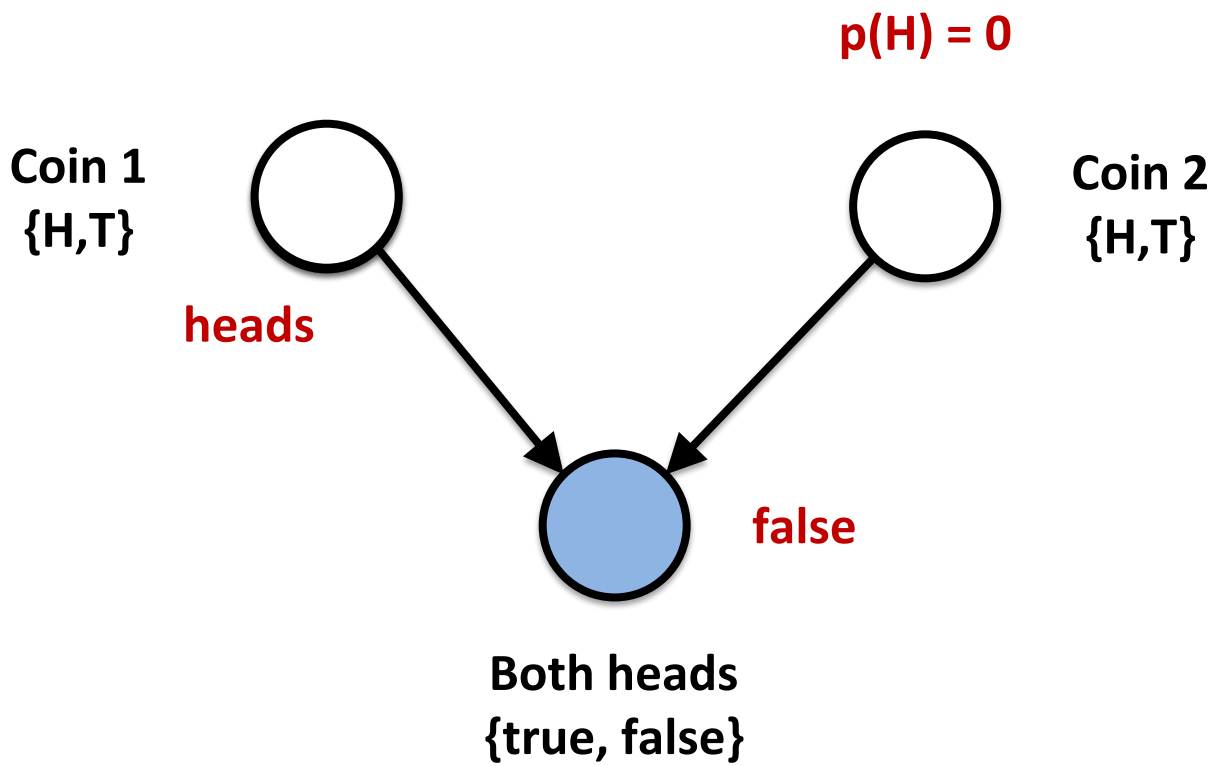
Coin 1	T	T	H	H
Coin 2	T	H	T	H
Both heads	false	false	false	true

Reasoning backwards



Coin 1	T	T	H	H
Coin 2	T	H	T	H
Both heads	false	false	false	true

Reasoning backwards

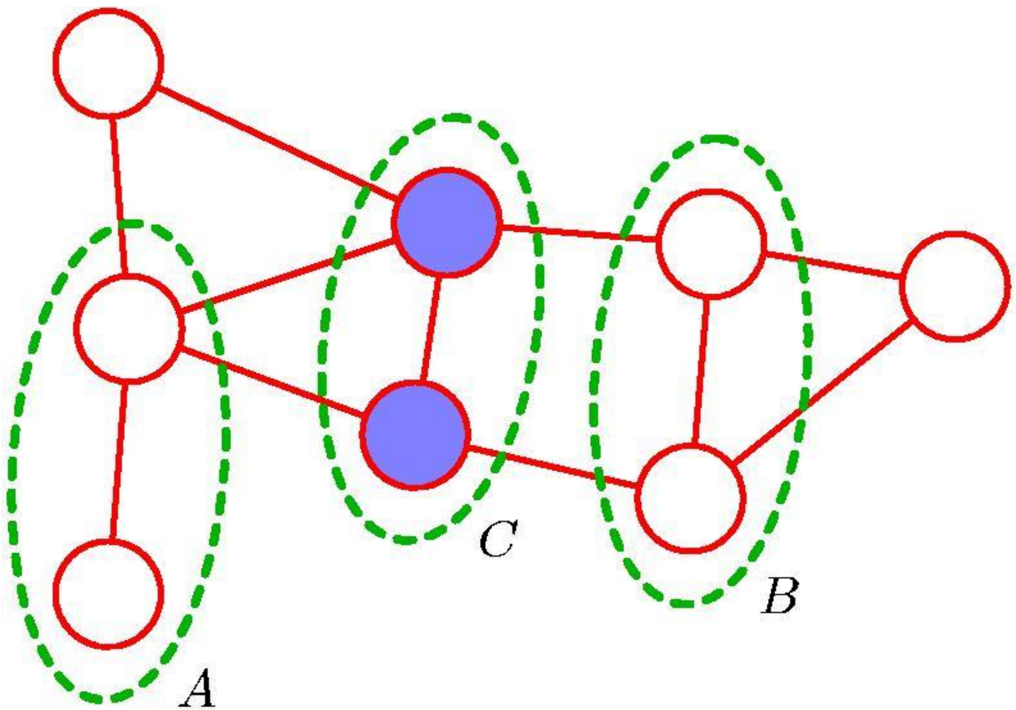


Coin 1	T	T	H	H
Coin 2	T	H	T	H
Both heads	false	false	false	true

“Explaining away”

7. Undirected Graphs

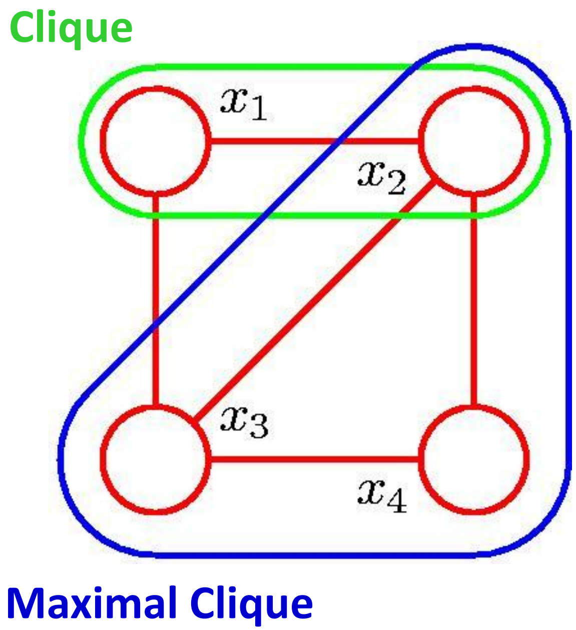
Undirected Graphs



$$A \perp\!\!\!\perp B \mid C$$

Markov random fields

Factorization

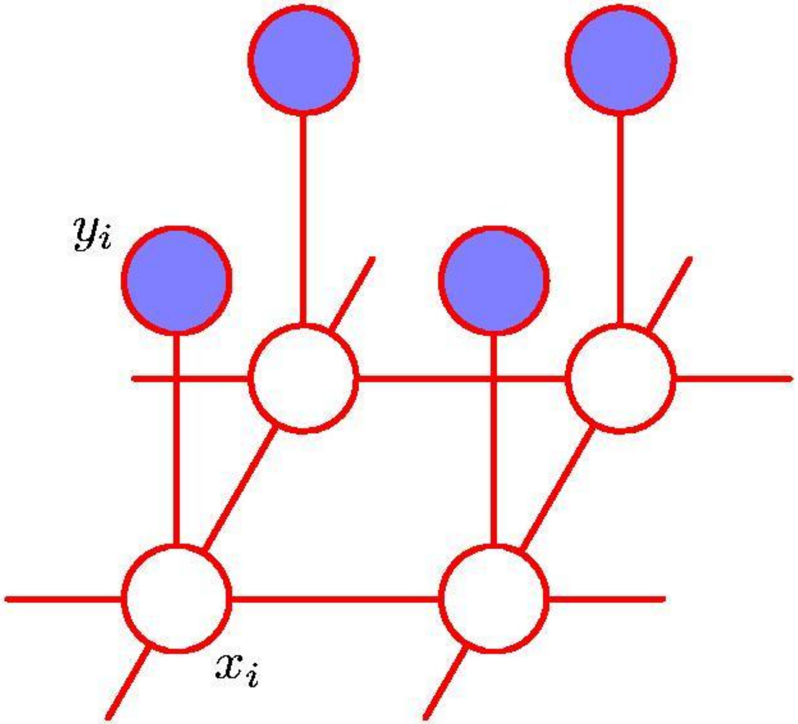


$$p(\mathbf{x}) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)$$

$$Z = \sum_{\mathbf{x}} \prod_C \psi_C(\mathbf{x}_C)$$

M K-state variables $\rightarrow K^M$ terms in Z

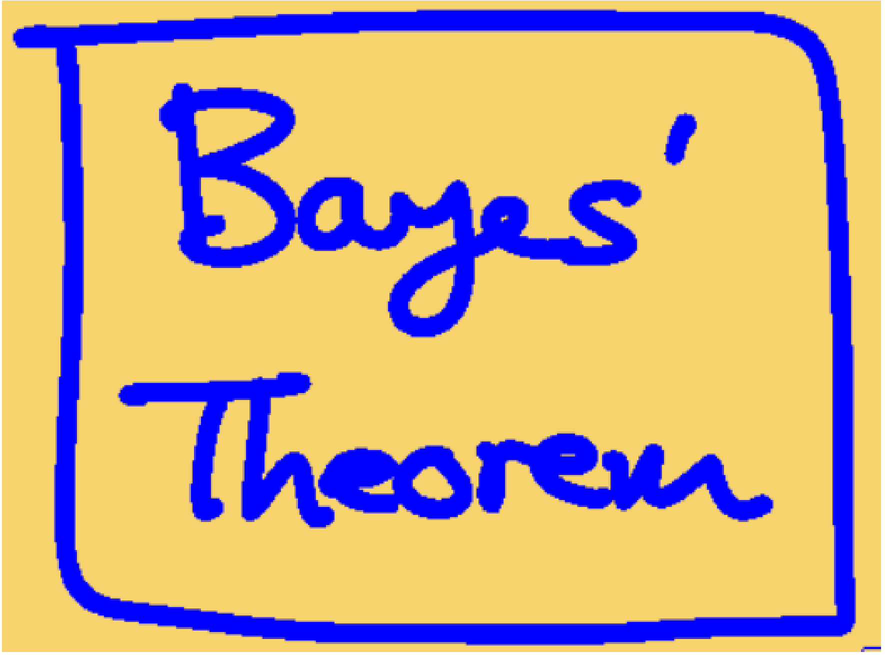
Illustration: Image De-Noising

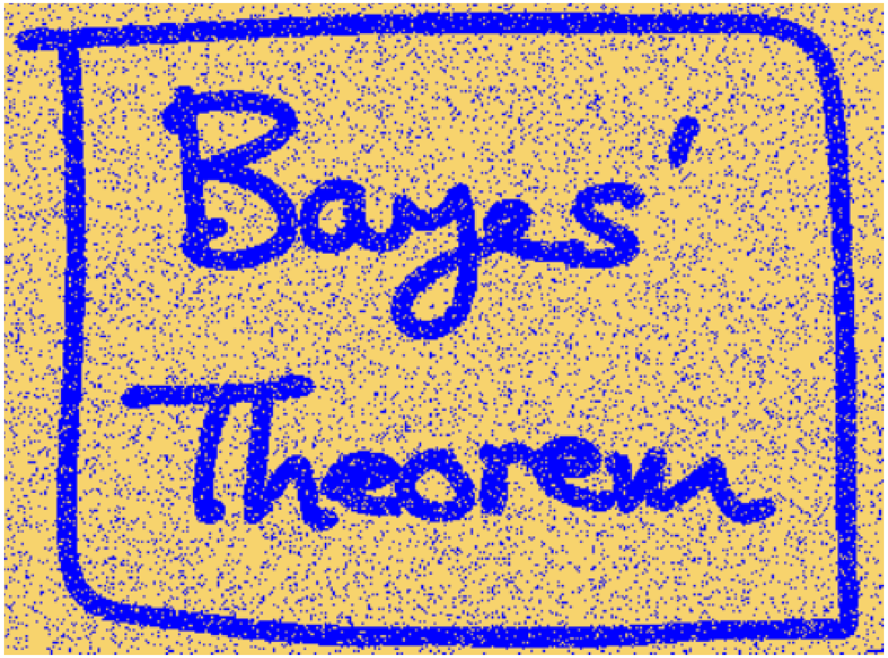


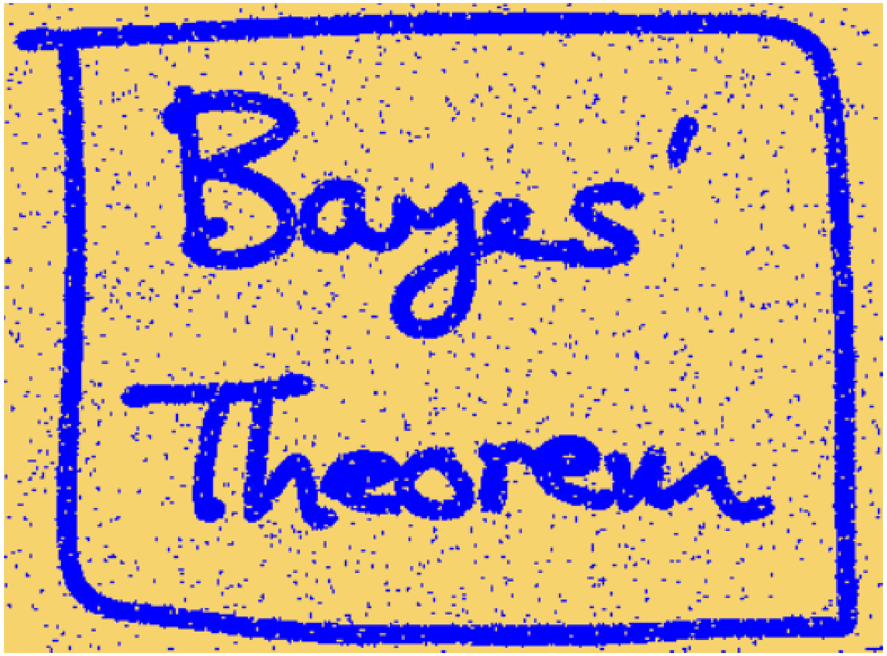
$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \exp\{-E(\mathbf{x}, \mathbf{y})\}$$

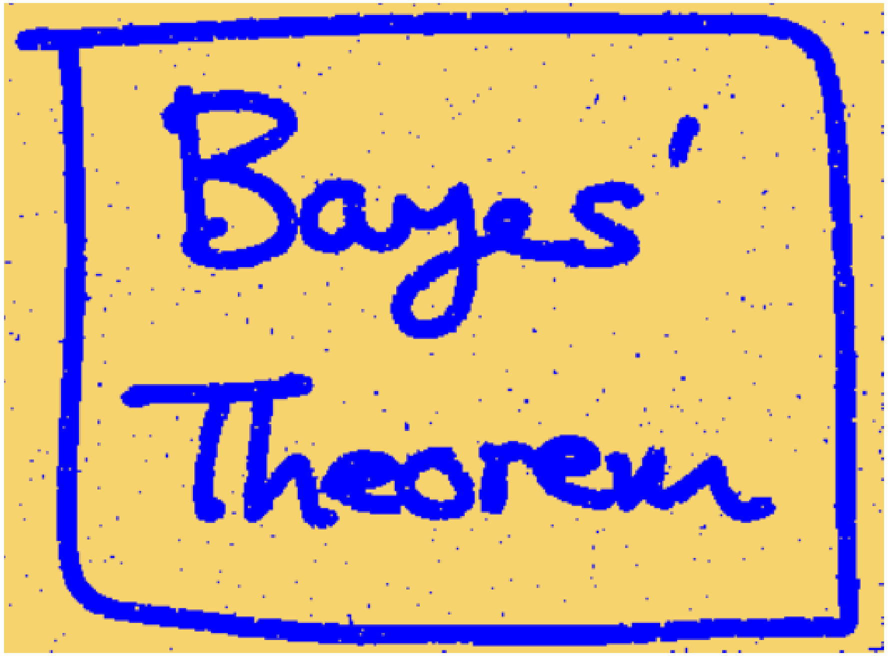
$x_i \in \{-1, +1\}$

$$E(\mathbf{x}, \mathbf{y}) = h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_i$$

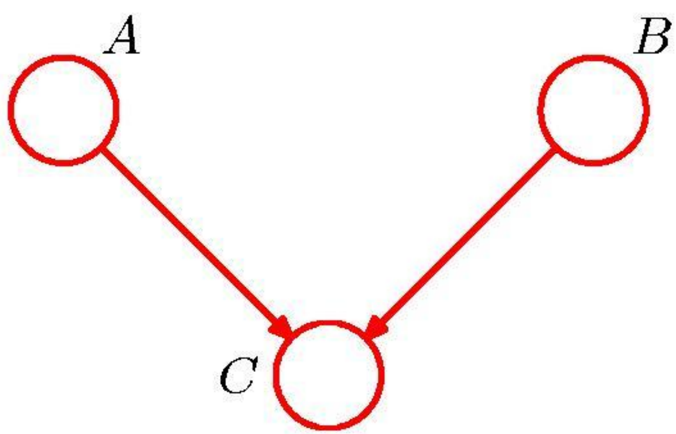

Bayes'
Theorem

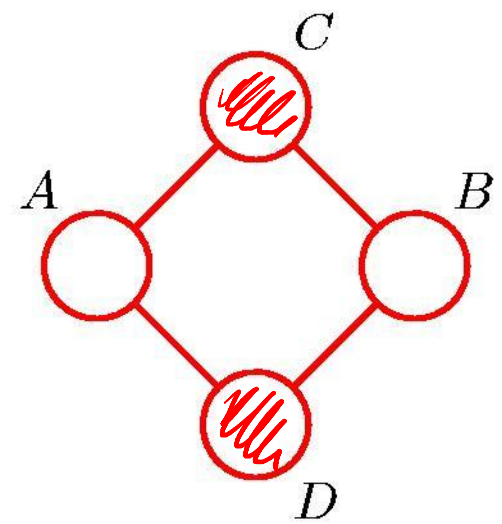

Bayes'
Theorem


Bayes'
Theorem


Bayes'
Theorem

Directed versus Undirected





8. Factor Graphs

Factorization

Directed graphs:

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

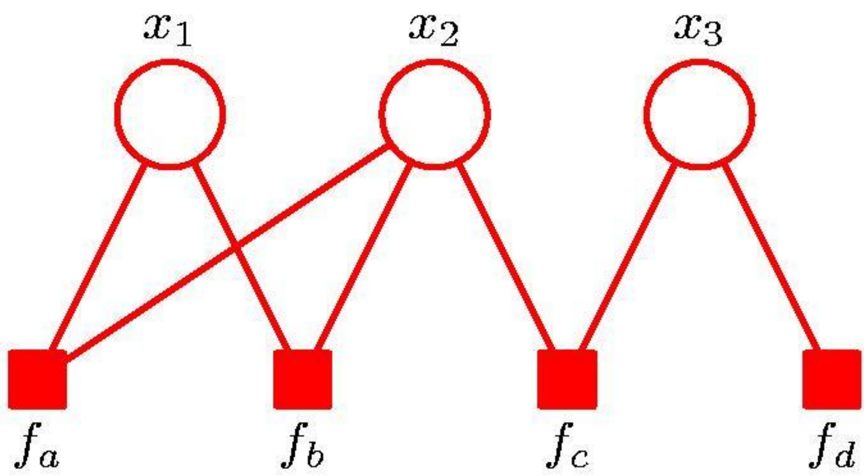
Undirected graphs:

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{C} \psi_C(\mathbf{x}_C)$$

Both have the form of products of factors:

$$p(\mathbf{x}) = \prod_{s} f_s(\mathbf{x}_s)$$

Factor Graphs

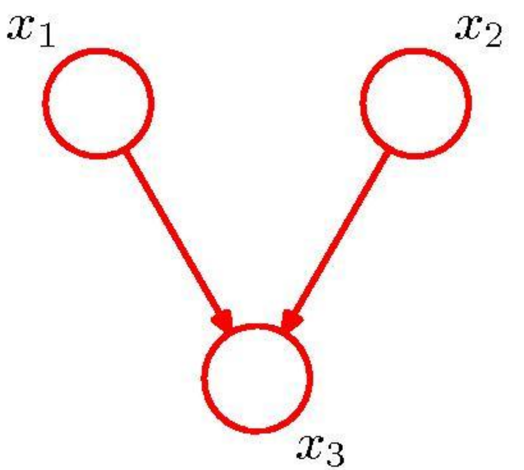


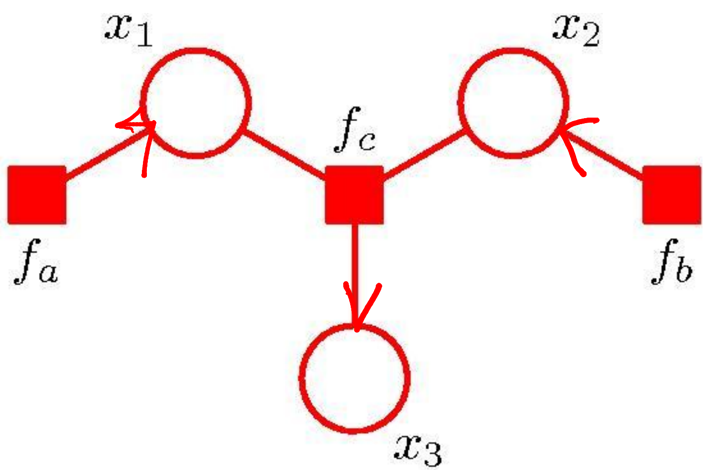
$$p(\mathbf{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

From Directed Graph to Factor Graph

$$p(x_1, x_2, x_3) = p(x_1)p(x_2)p(x_3|x_1, x_2)$$





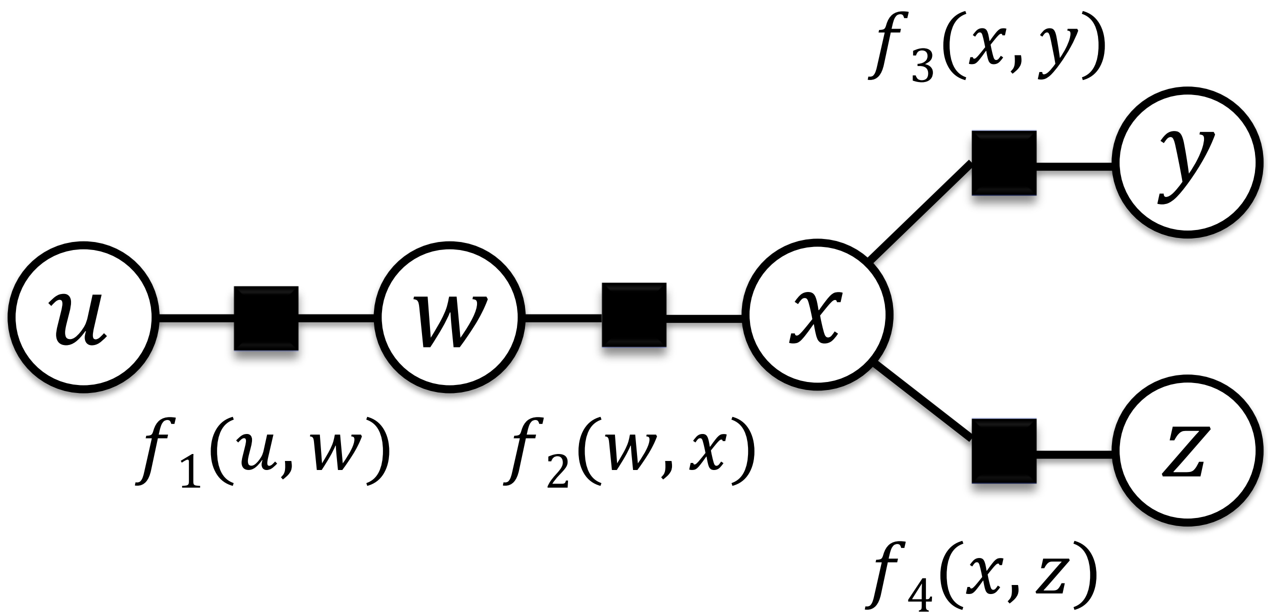
$$\begin{aligned} f_a(x_1) &= p(x_1) \\ f_b(x_2) &= p(x_2) \\ f_c(x_1, x_2, x_3) &= p(x_3|x_1, x_2) \end{aligned}$$

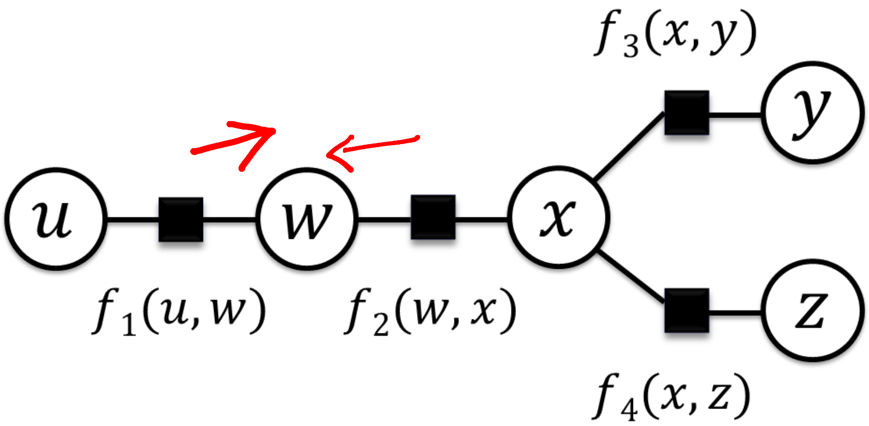
9. Inference

Efficient inference

$$
\begin{aligned}
\sum_x \sum_y xy &= x_1 y_1 + x_2 y_1 + x_1 y_2 + x_2 y_2 \\
&= (x_1 + x_2)(y_1 + y_2)
\end{aligned}
$$

The Sum-Product Algorithm





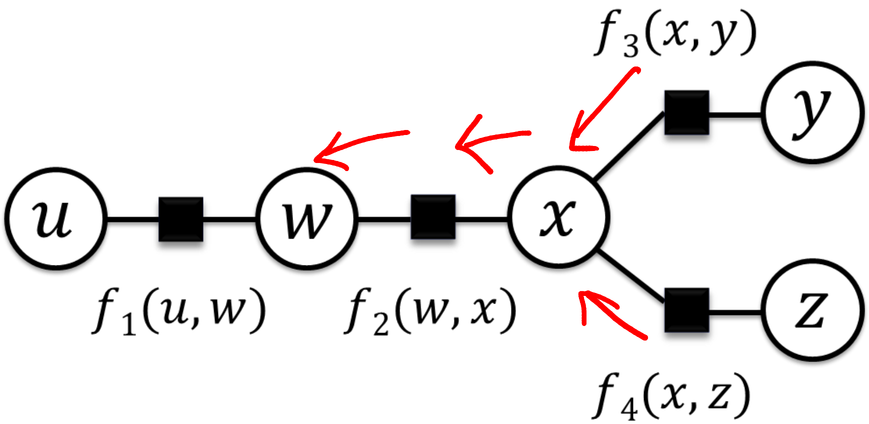
$$p(u,w,x,y,z) = f_1(u,w) \cdot f_2(w,x) \cdot f_3(x,y) \cdot f_4(y,z)$$

$$O(K^5)$$

$$p(w) = \sum_u \sum_x \sum_y \sum_z p(u,w,x,y,z)$$

$$= \left[\sum_u f_1(u,w)\right]\left[\sum_x \sum_y \sum_z f_2(w,x) f_3(x,y) f_4(x,z)\right]$$

$$= m_{f_1 \to w}(w) \cdot m_{f_2 \to w}(w)$$



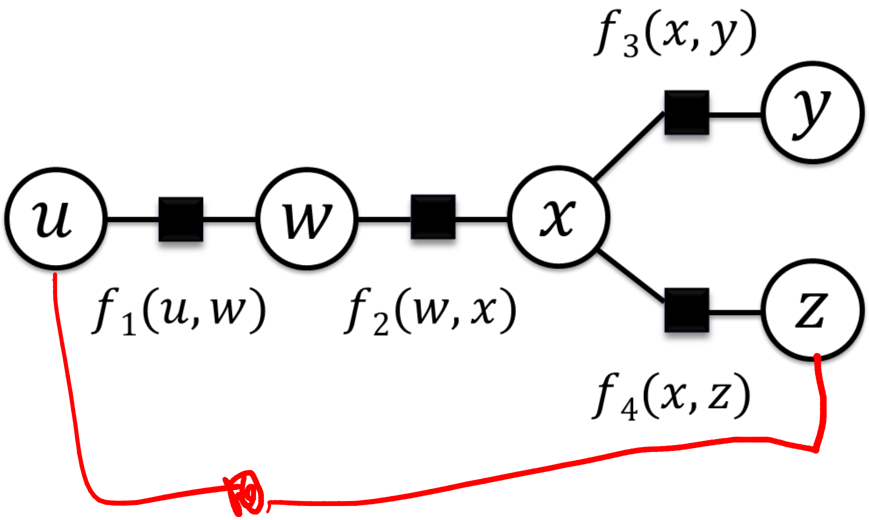
$O(K^2 L)$

$$m_{f_2 \to w}(w) = \sum_x f_2(w,x) \left[\sum_y \sum_z f_3(x,y) f_4(x,z) \right]$$

$m_{x \to f_2}(x)$

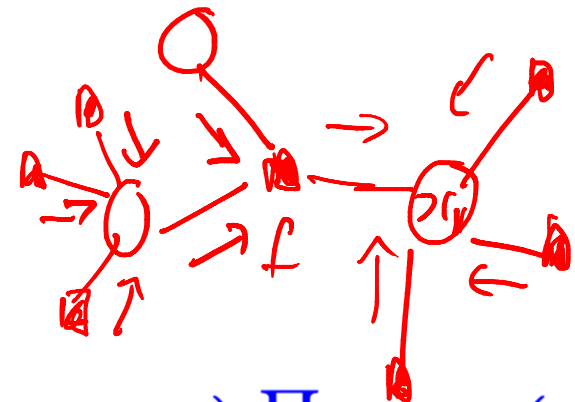
$$\left[\sum_y f_3(\cancel{y}x,y) \right] \left[\sum_z f_4(x,z) \right]$$

$m_{f_3 \to x}(x)$ $\qquad$ $m_{f_4 \to x}(x)$


$f_3(x, y)$
y
u
w
x
z
$f_1(u, w)$
$f_2(w, x)$
$f_4(x, z)$

The Sum-Product Algorithm

Three update equations



$$p(x) = \prod_{f \in F_x} m_{f \to x}(x)$$

$$m_{f \to x_1}(x_1) = \sum_{x_2} \sum_{x_3} \cdots \sum_{x_n} f(x_1, x_2, x_3, \ldots) \prod_{i>1} m_{x_i \to f}(x_i)$$

$$m_{x \to f}(x) = \prod_{f_j \in F_x \setminus \{f\}} m_{f_j \to x}(x)$$

Message schedule from root to leaves and back

One message in each direction on each link

What if the graph is not a tree?

Condition on variables to break loops

– *cut-set conditioning* (exact)

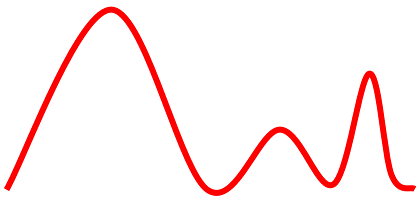
Transform graph into tree of composite nodes

– *junction tree algorithm* (exact)

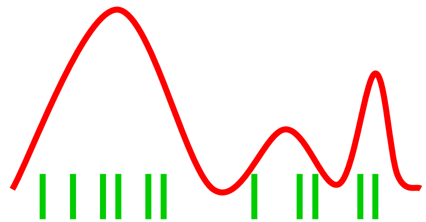
Approximate: keep iterating the messages:

– *loopy belief propagation* (approximate)

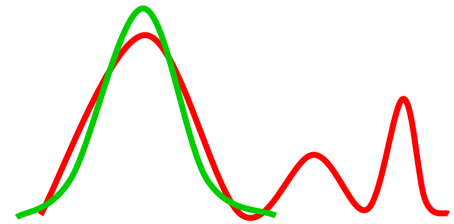
What if the messages are intractable?



True distribution



Monte Carlo

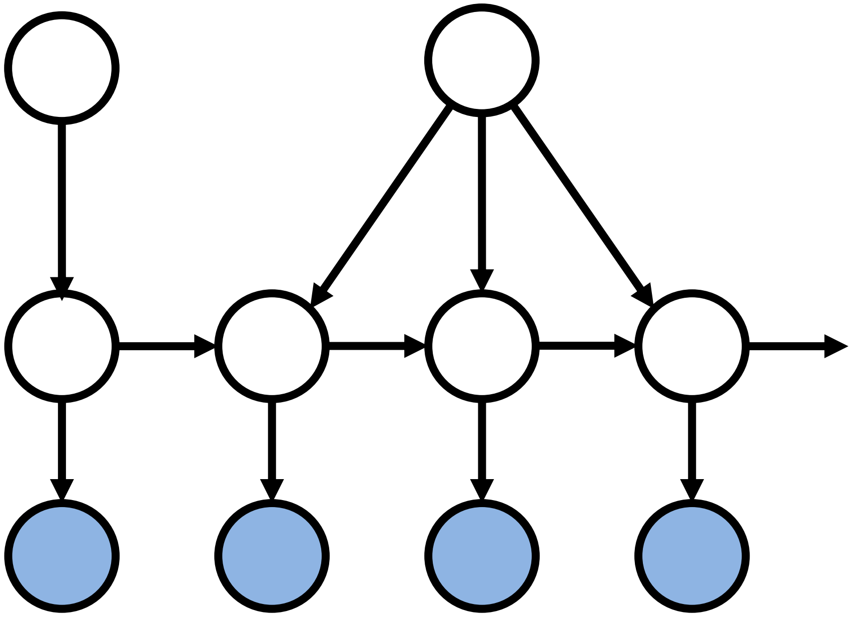


Variational Message Passing

Expectation propagation

⋮

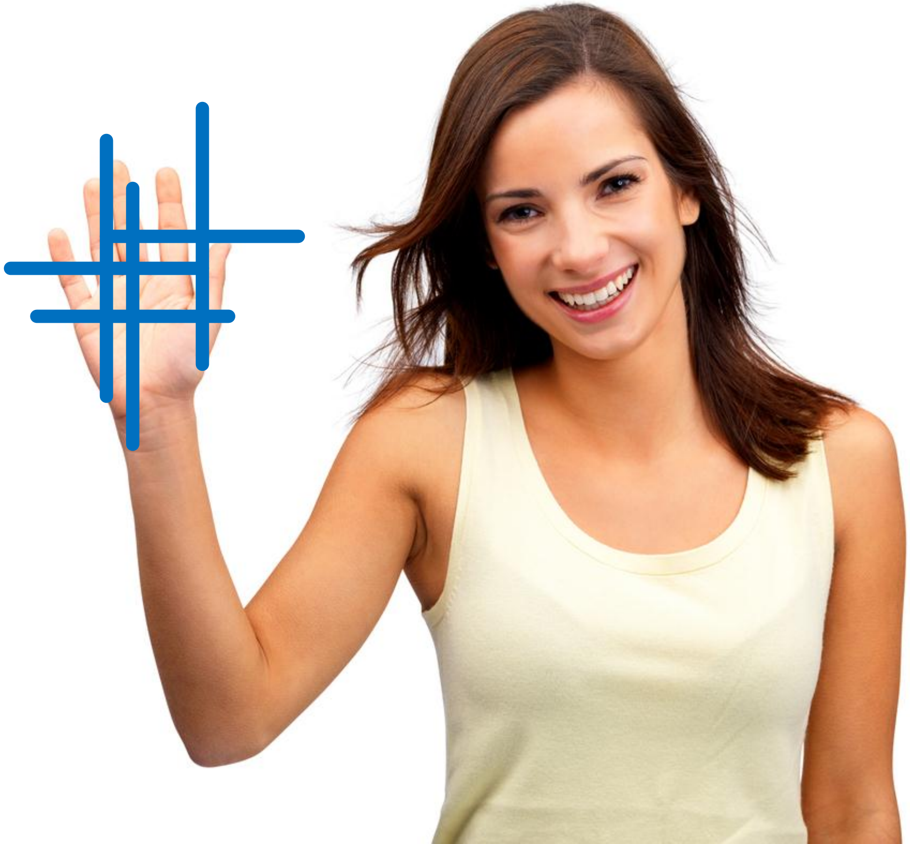
Learning is just inference!



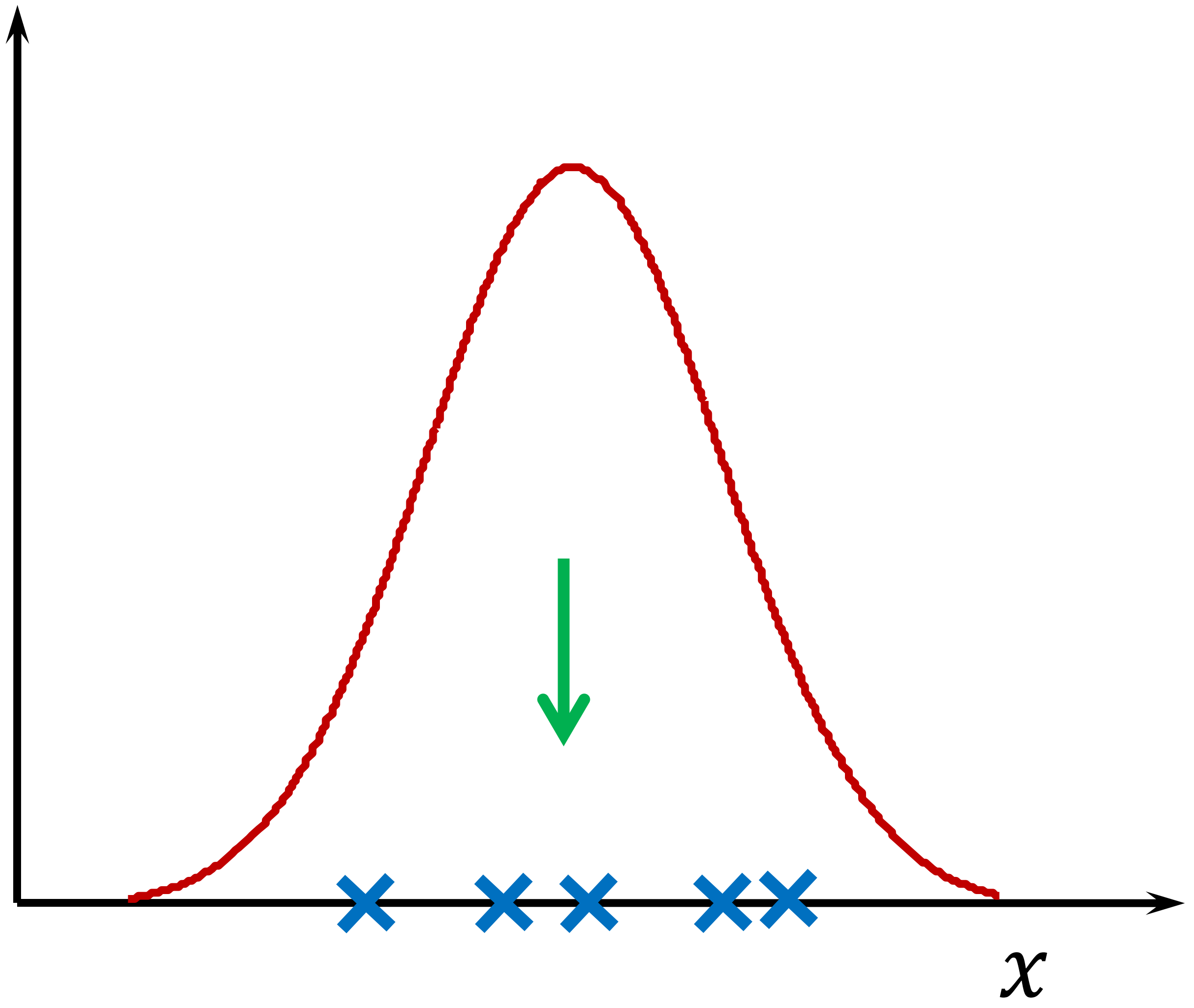
10. Example: Kalman filter

Hand location

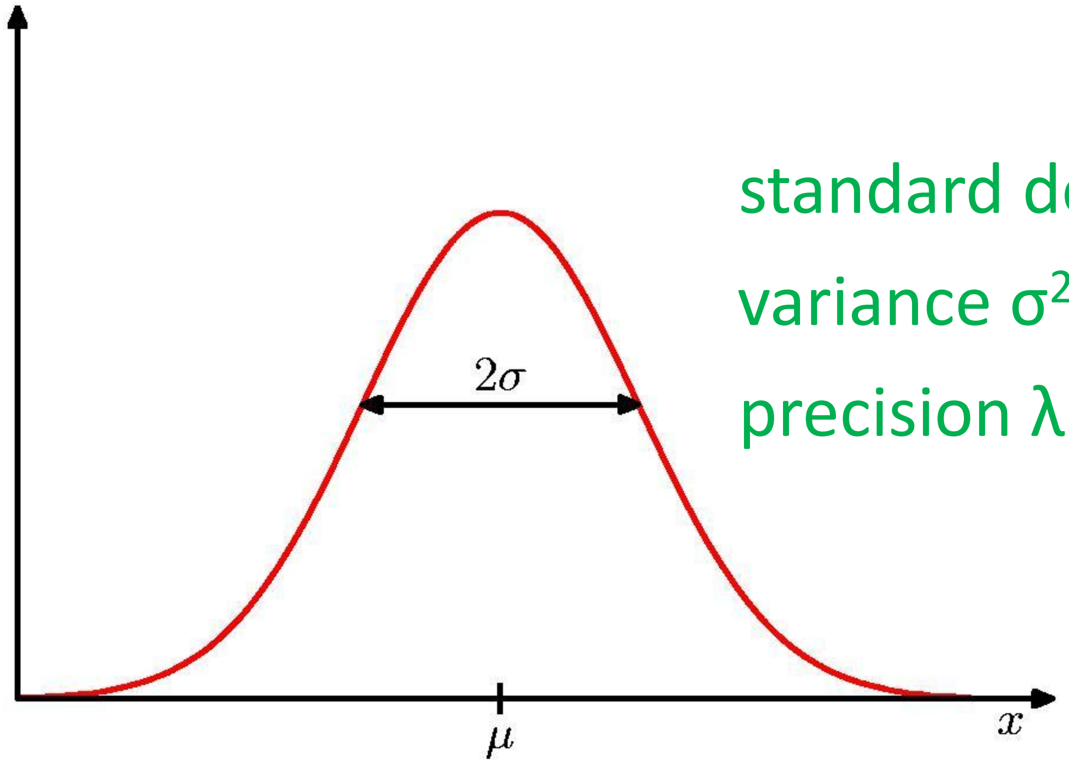
Noisy position sensor



Finding the true location



The Gaussian distribution



standard deviation σ

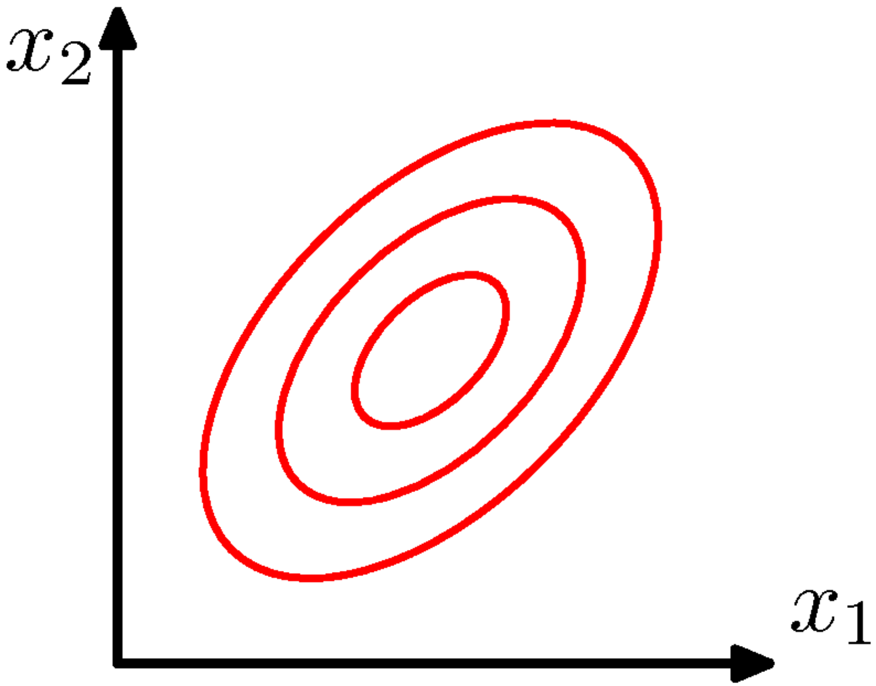
variance σ^2

precision $\lambda = 1/\sigma^2$

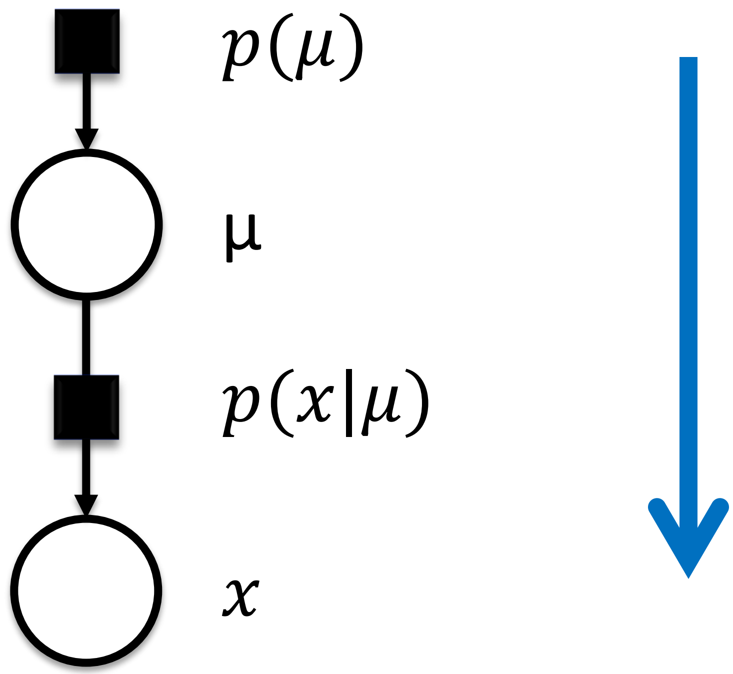
mean μ

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

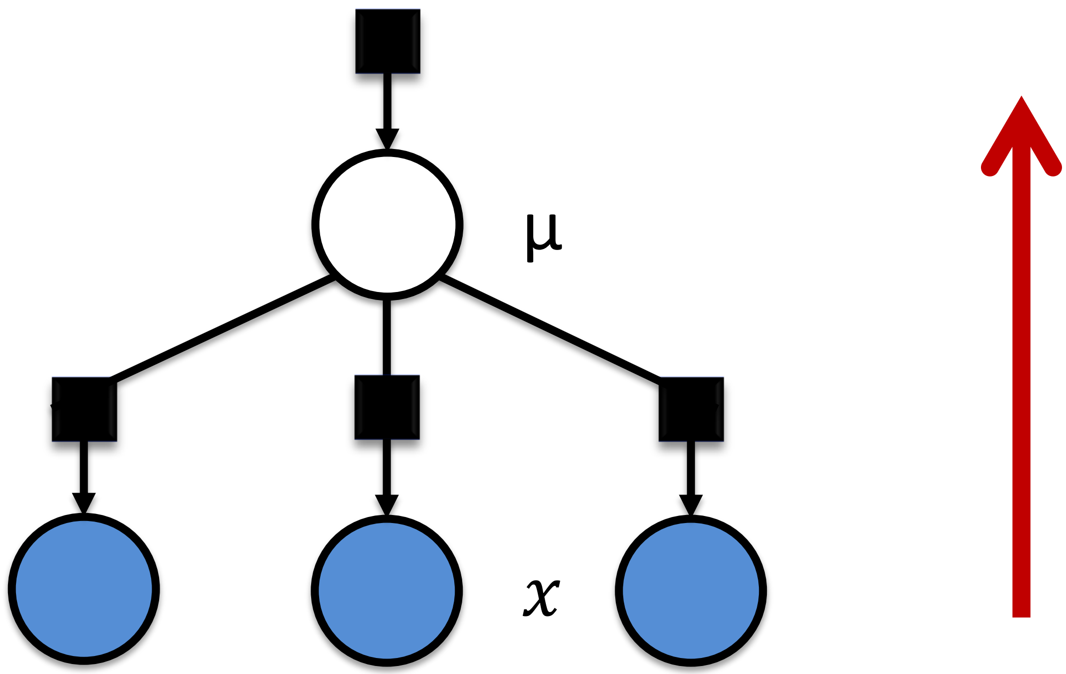
The multi-dimensional Gaussian



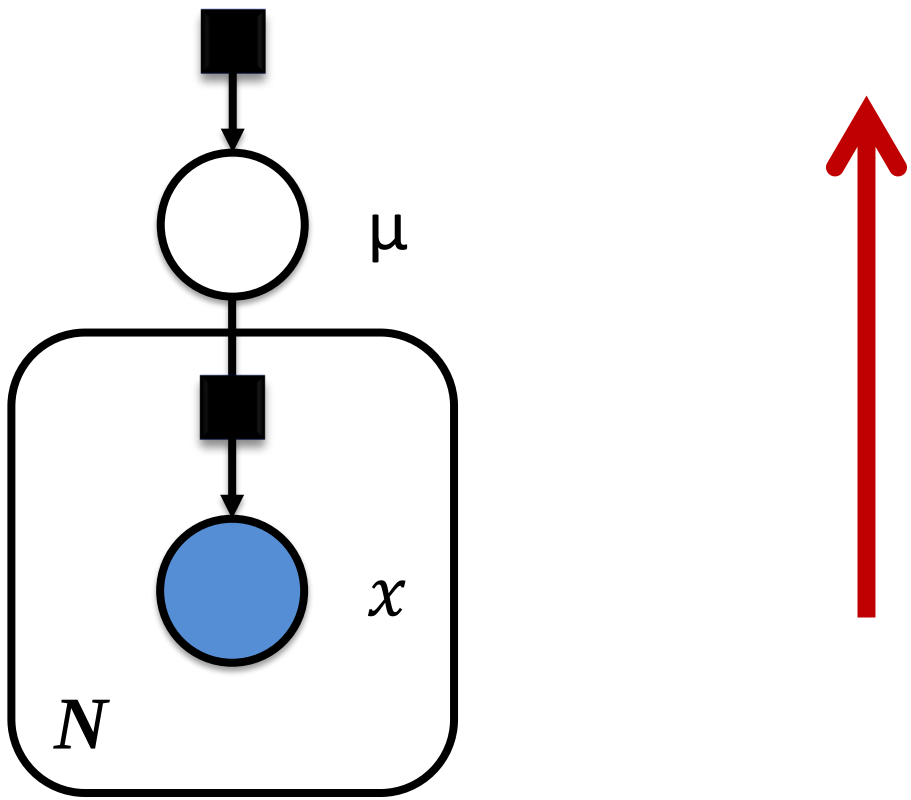
Learning the mean

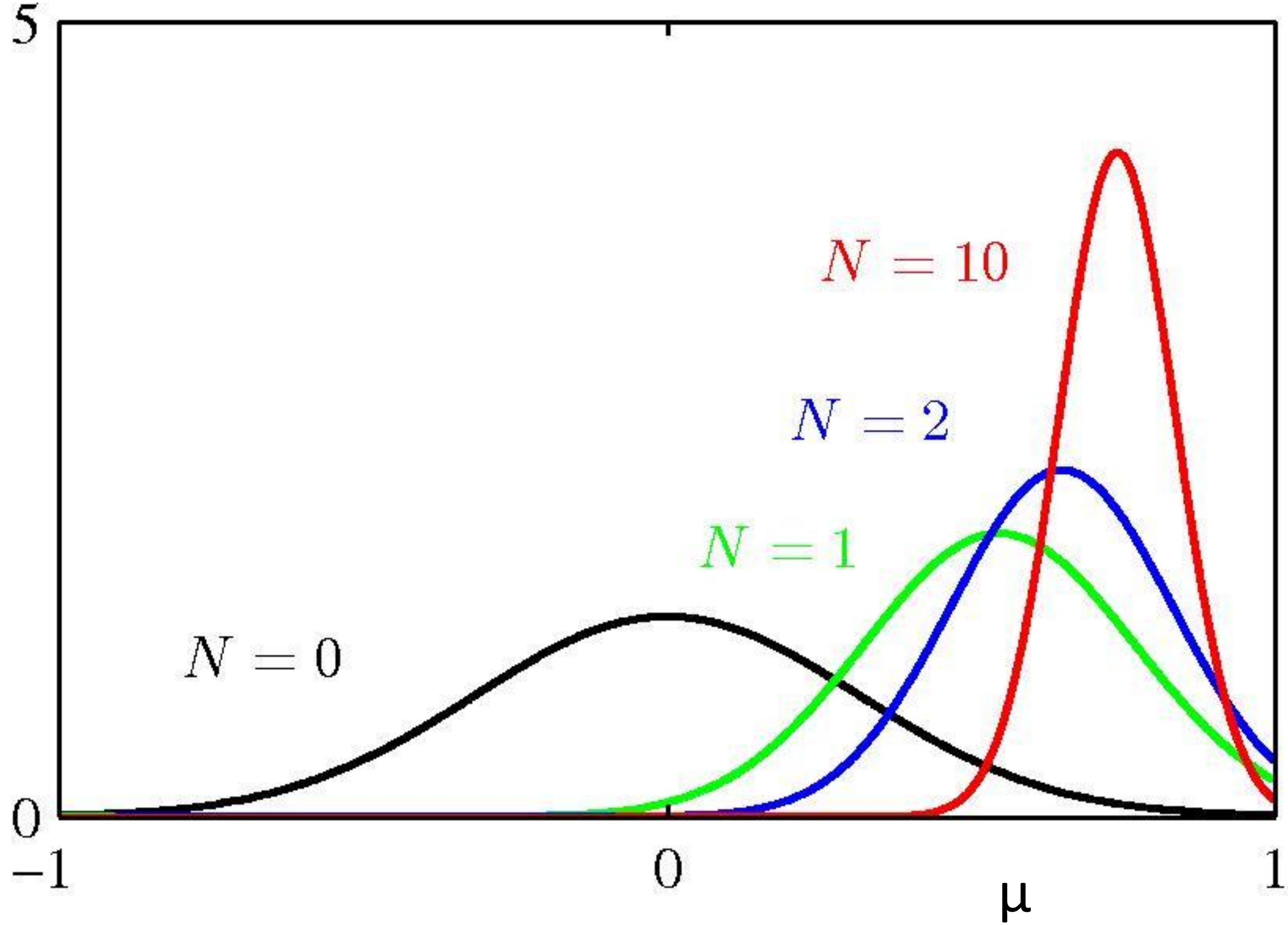


Learning the mean



Plates

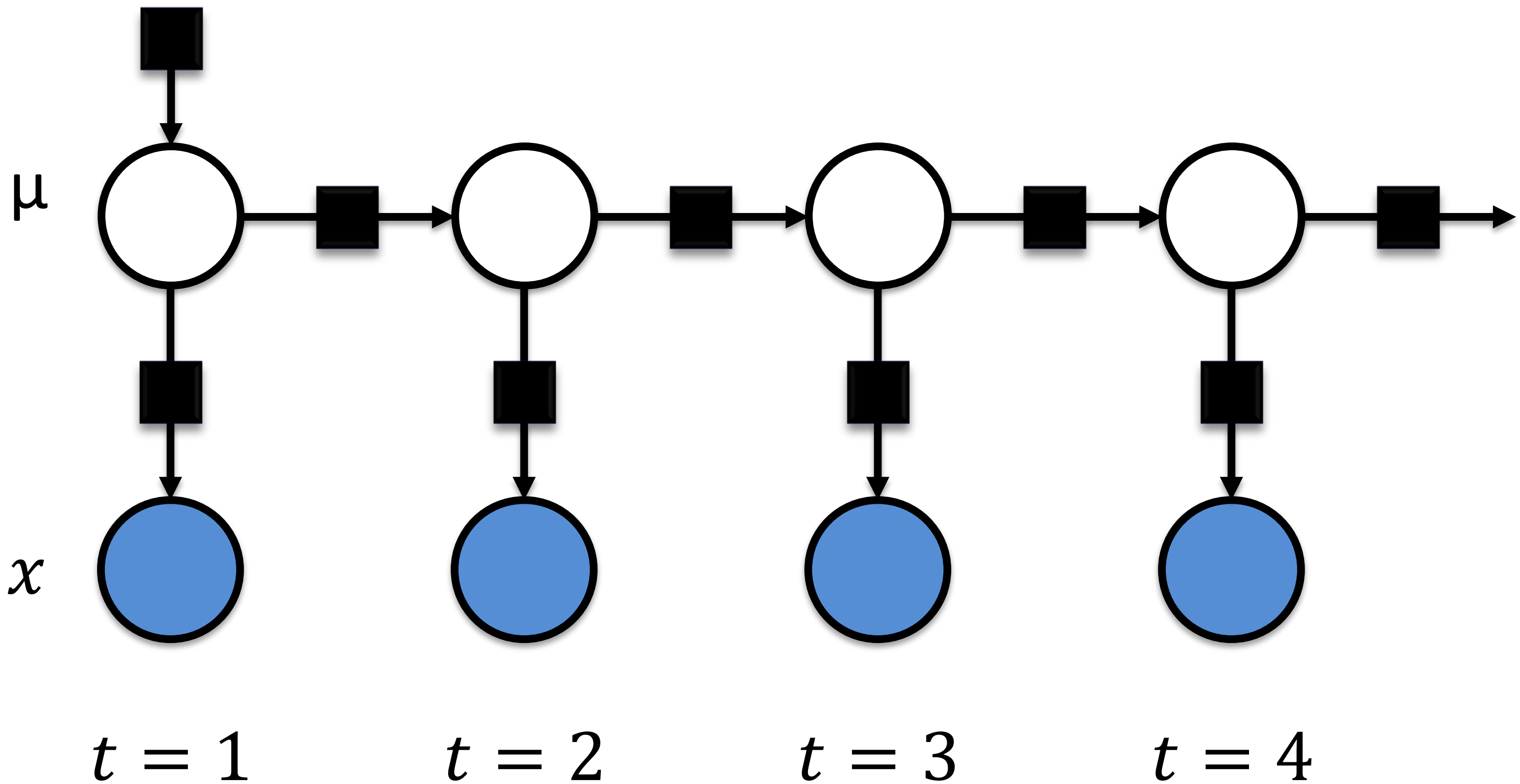




5
0
−1
0
1
μ
N = 10
N = 2
N = 1
N = 0

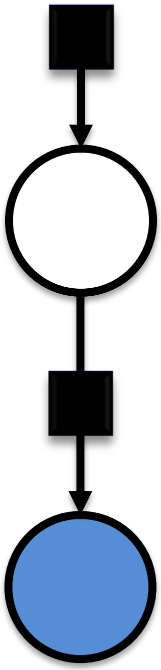
Hand tracking

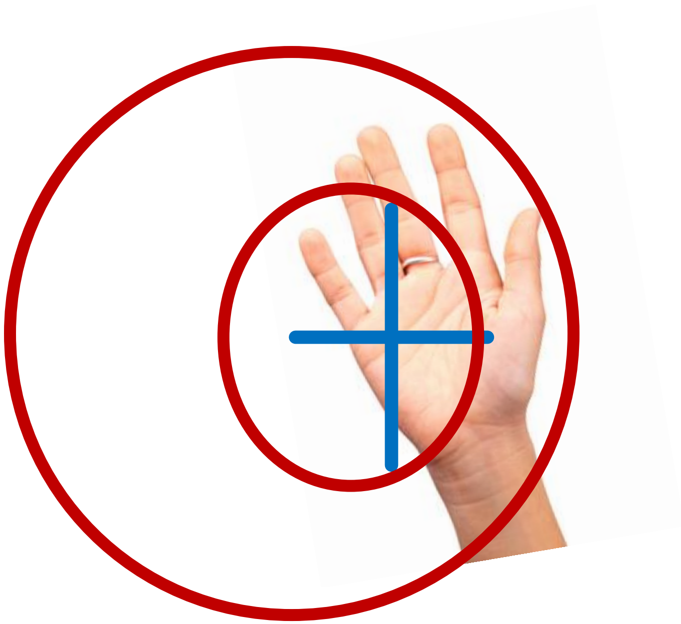
Noisy position sensor *and* moving hand

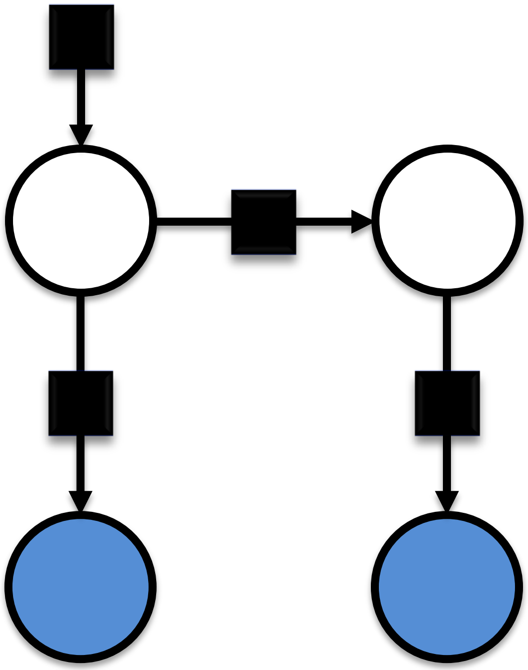


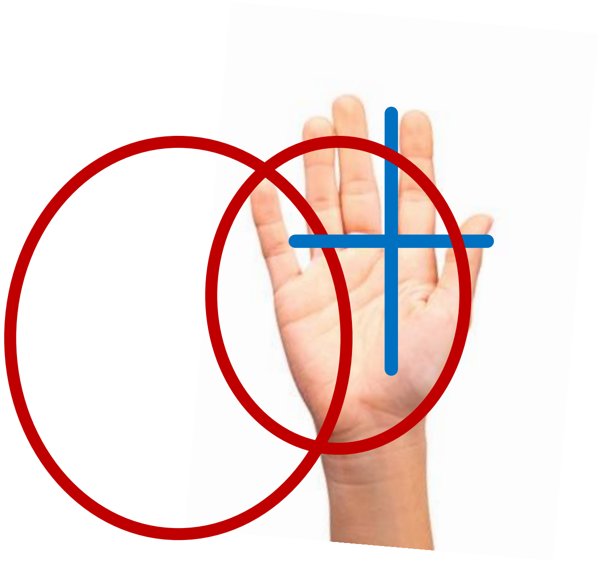
The Kalman filter

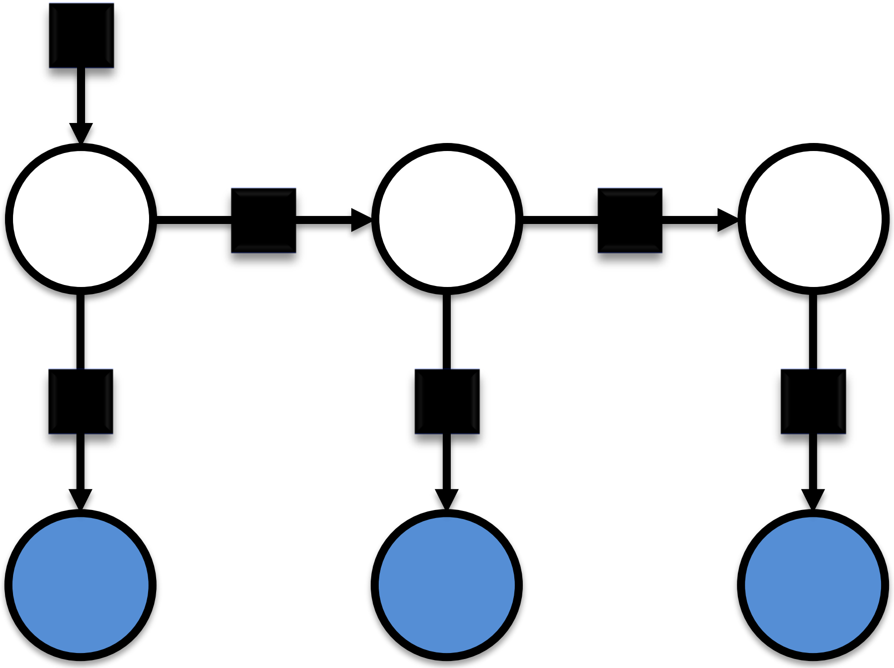
(*The hidden Markov model*)

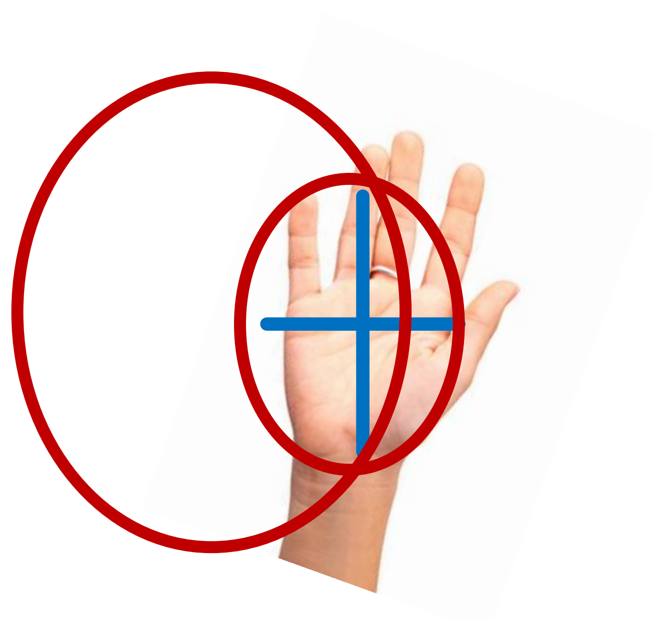




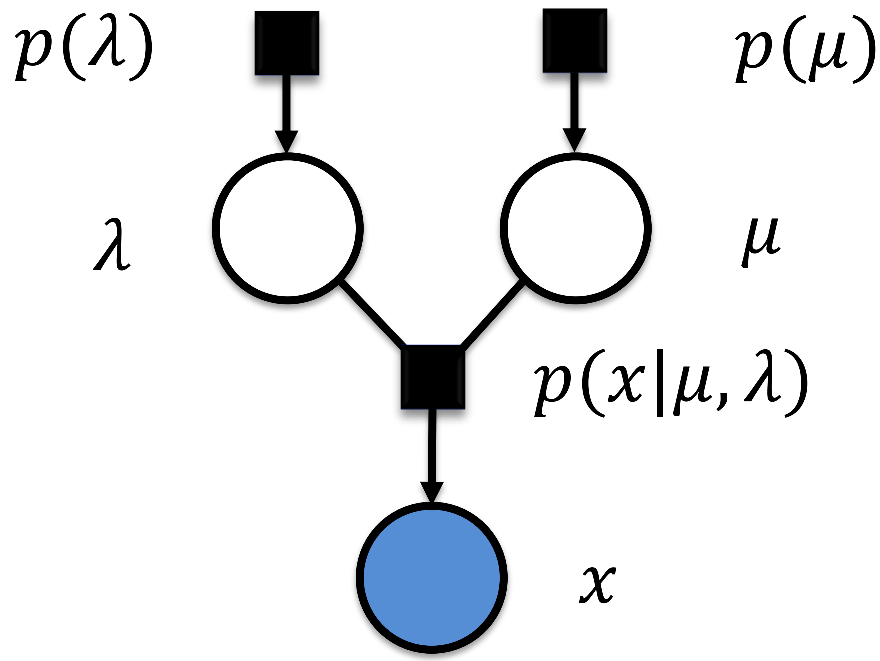


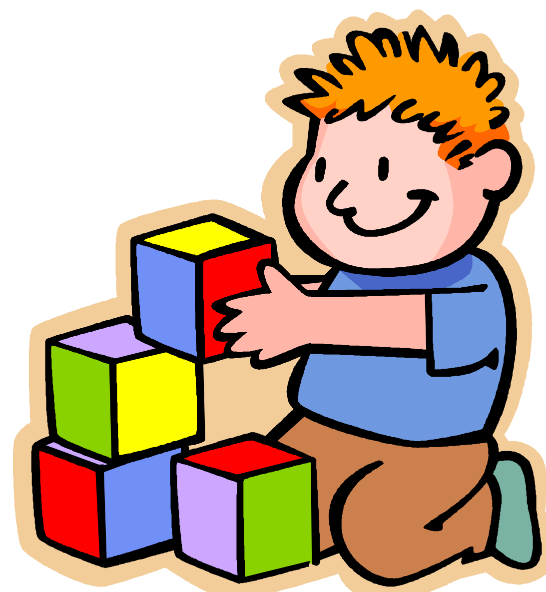






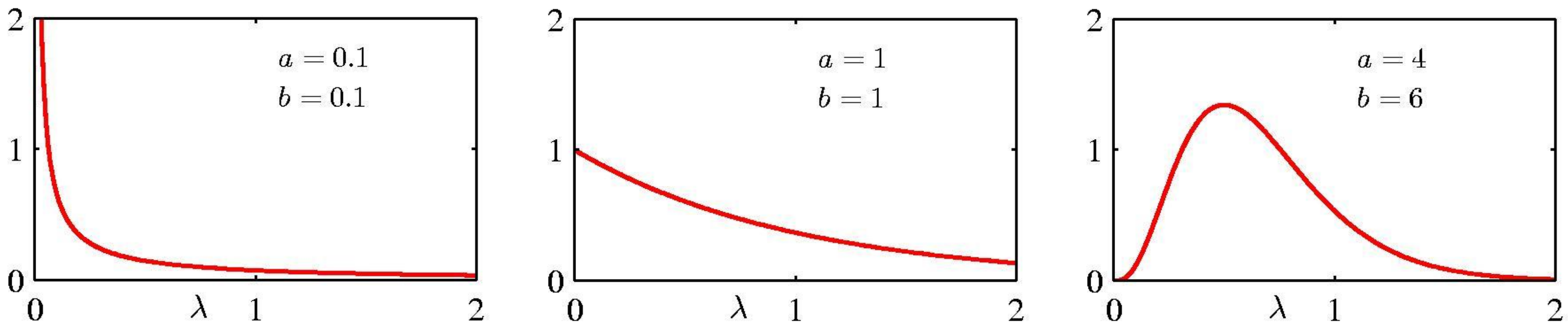
What about the noise level?





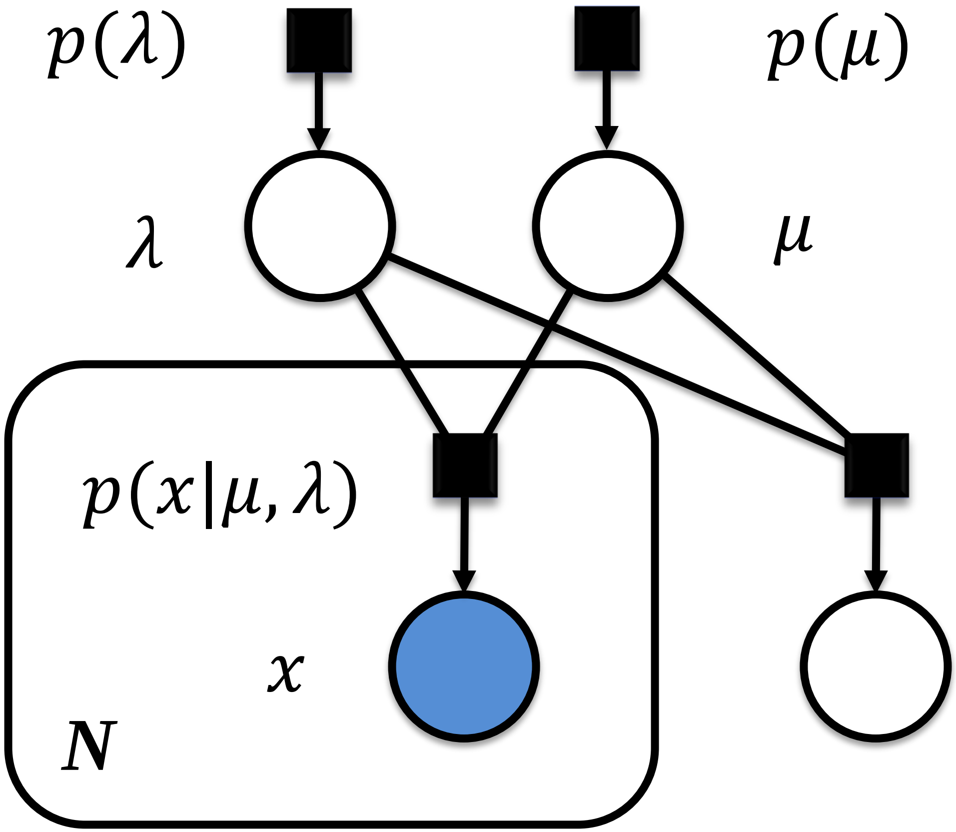
The gamma distribution

$$\mathrm{Gam}(\lambda|a,b) = \frac{1}{\Gamma(a)} b^a \lambda^{a-1} \exp(-b\lambda)$$

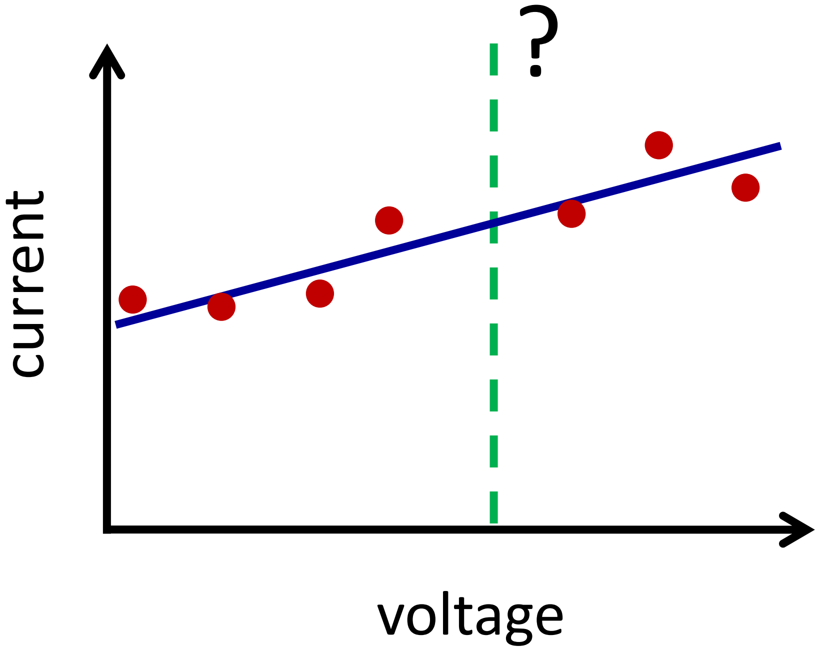


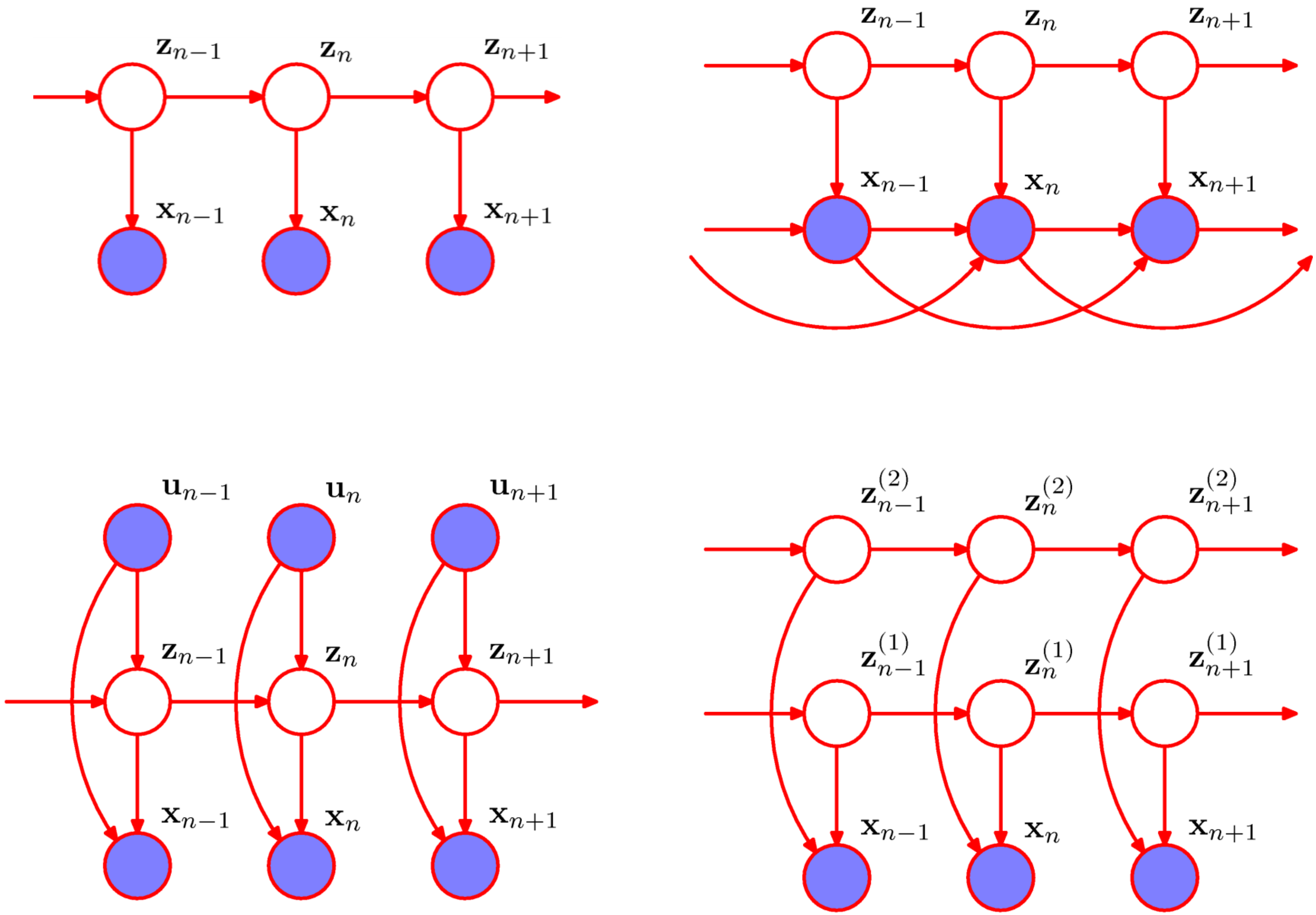
An example of a *conjugate prior*

Predictions



“Big data”




$\mathbf{z}_{n-1}$
$\mathbf{z}_{n}$
$\mathbf{z}_{n+1}$
$\mathbf{x}_{n-1}$
$\mathbf{x}_{n}$
$\mathbf{x}_{n+1}$
$\mathbf{z}_{n-1}$
$\mathbf{z}_{n}$
$\mathbf{z}_{n+1}$
$\mathbf{x}_{n-1}$
$\mathbf{x}_{n}$
$\mathbf{x}_{n+1}$
$\mathbf{u}_{n-1}$
$\mathbf{u}_{n}$
$\mathbf{u}_{n+1}$
$\mathbf{z}_{n-1}$
$\mathbf{z}_{n}$
$\mathbf{z}_{n+1}$
$\mathbf{x}_{n-1}$
$\mathbf{x}_{n}$
$\mathbf{x}_{n+1}$
$\mathbf{z}^{(2)}_{n-1}$
$\mathbf{z}^{(2)}_{n}$
$\mathbf{z}^{(2)}_{n+1}$
$\mathbf{z}^{(1)}_{n-1}$
$\mathbf{z}^{(1)}_{n}$
$\mathbf{z}^{(1)}_{n+1}$
$\mathbf{x}_{n-1}$
$\mathbf{x}_{n}$
$\mathbf{x}_{n+1}$

11. Case Study: *TrueSkill*™

TrueSkill™

Ralf Herbrich, Tom Minka, and Thore Graepel (NIPS, 2007)

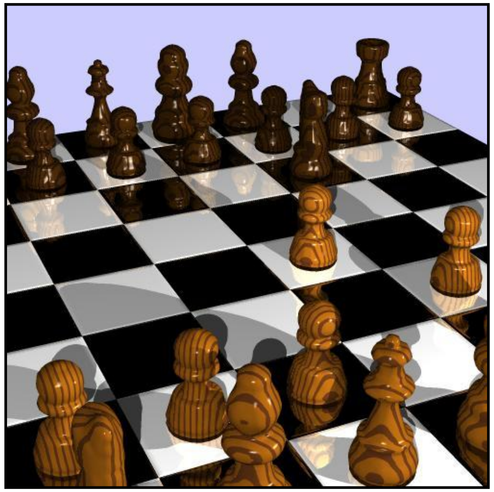
Elo

International standard for chess grading

A single rating for each player

Limitations:

- not applicable to more than two players
- not applicable to team games

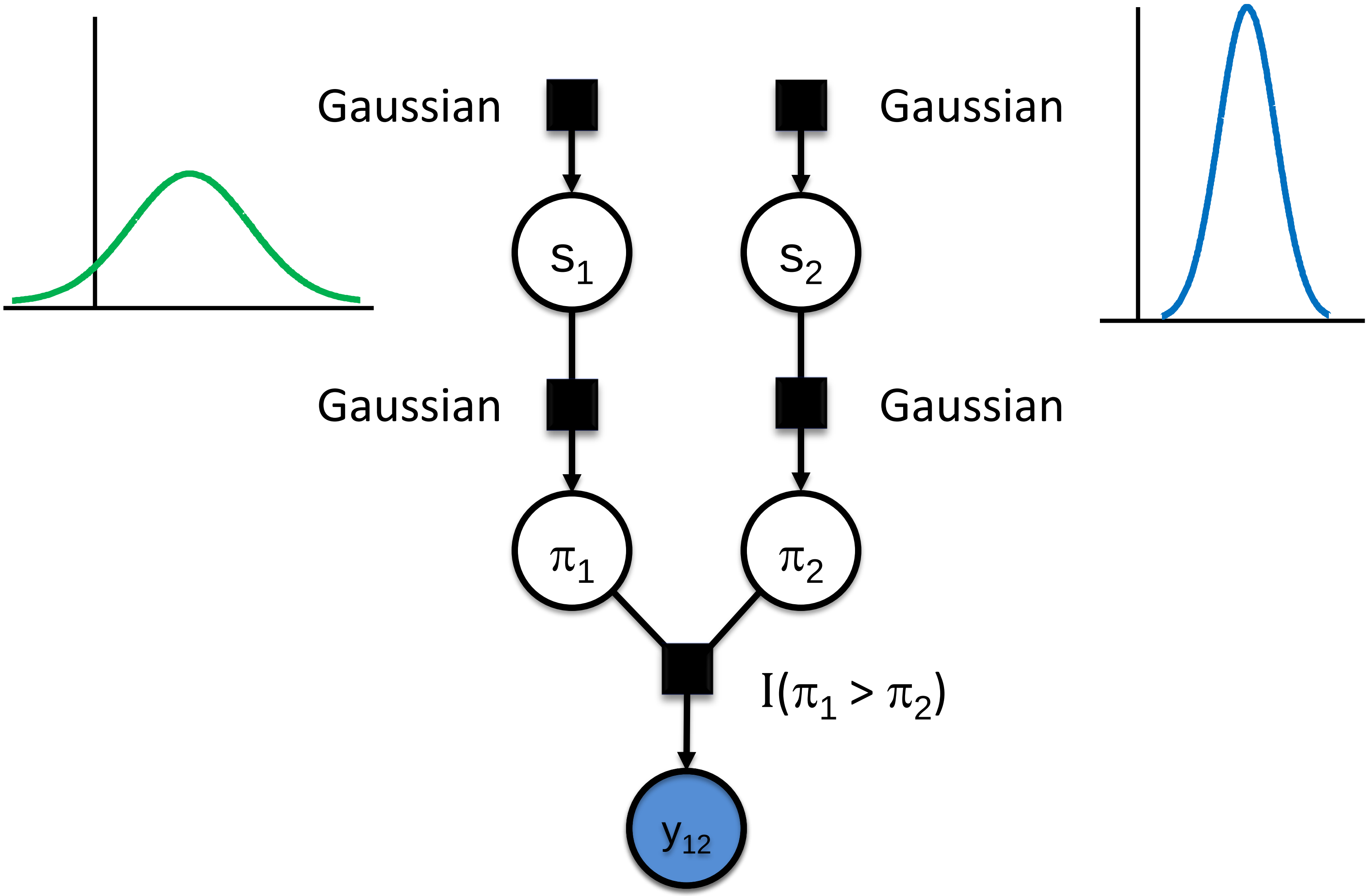


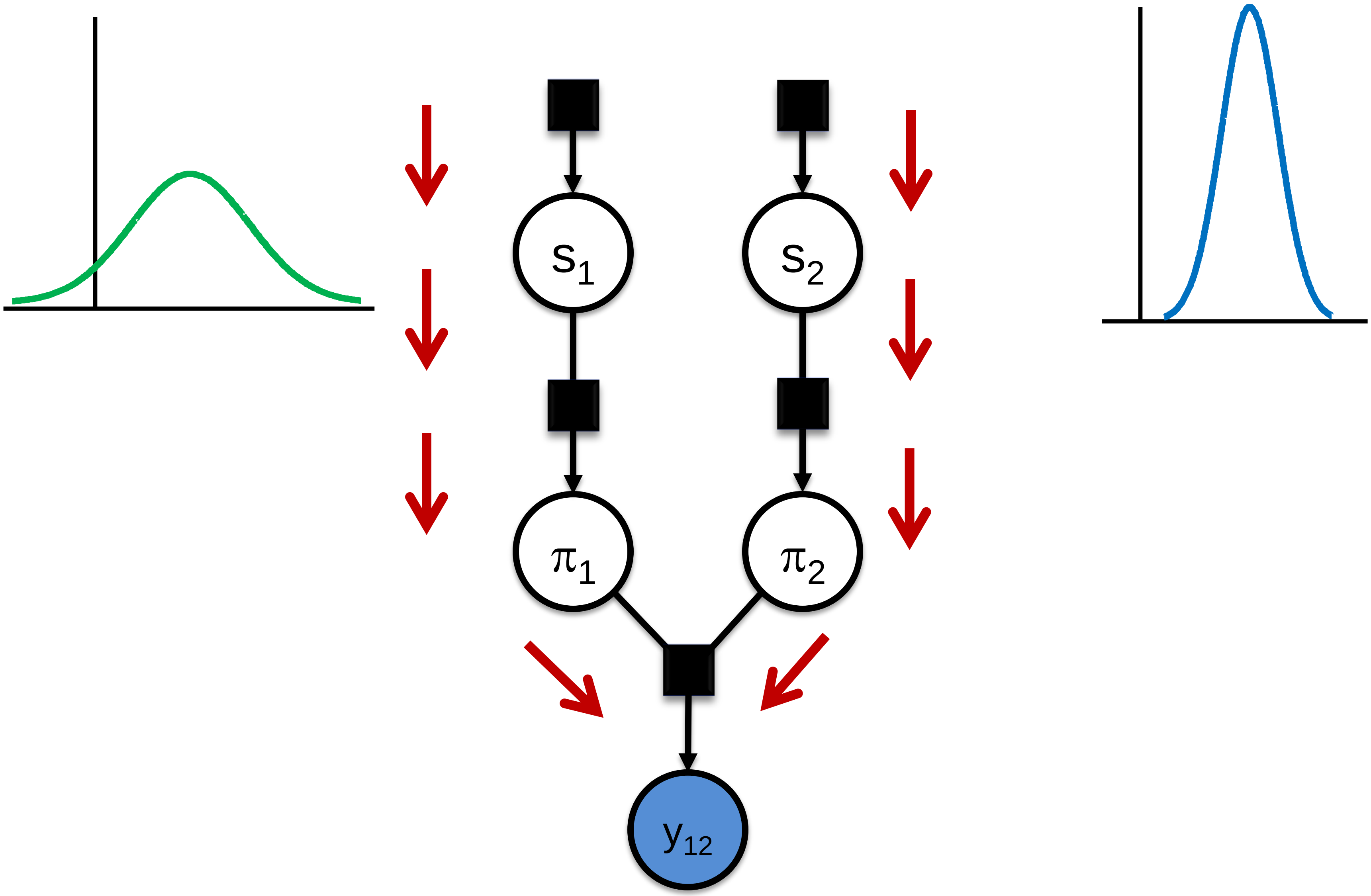
Stages of MBML

1. Build a *model*: joint probability distribution of all of the relevant variables (e.g. as a graph)
2. Incorporate the *observed* data
3. Compute the distributions over the desired variables: *inference*

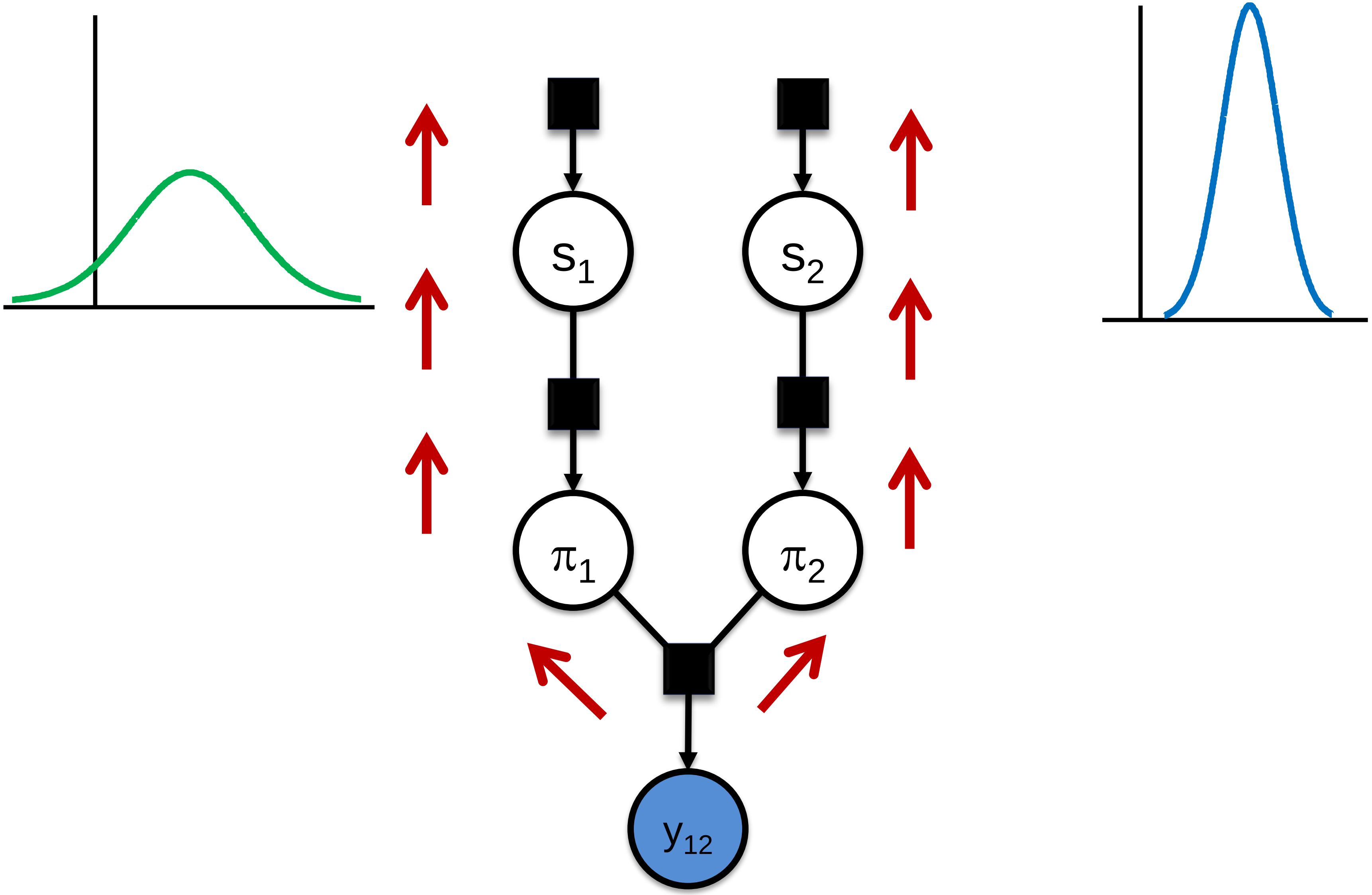
Iterate 2 and 3 in real-time applications

Extend model as required

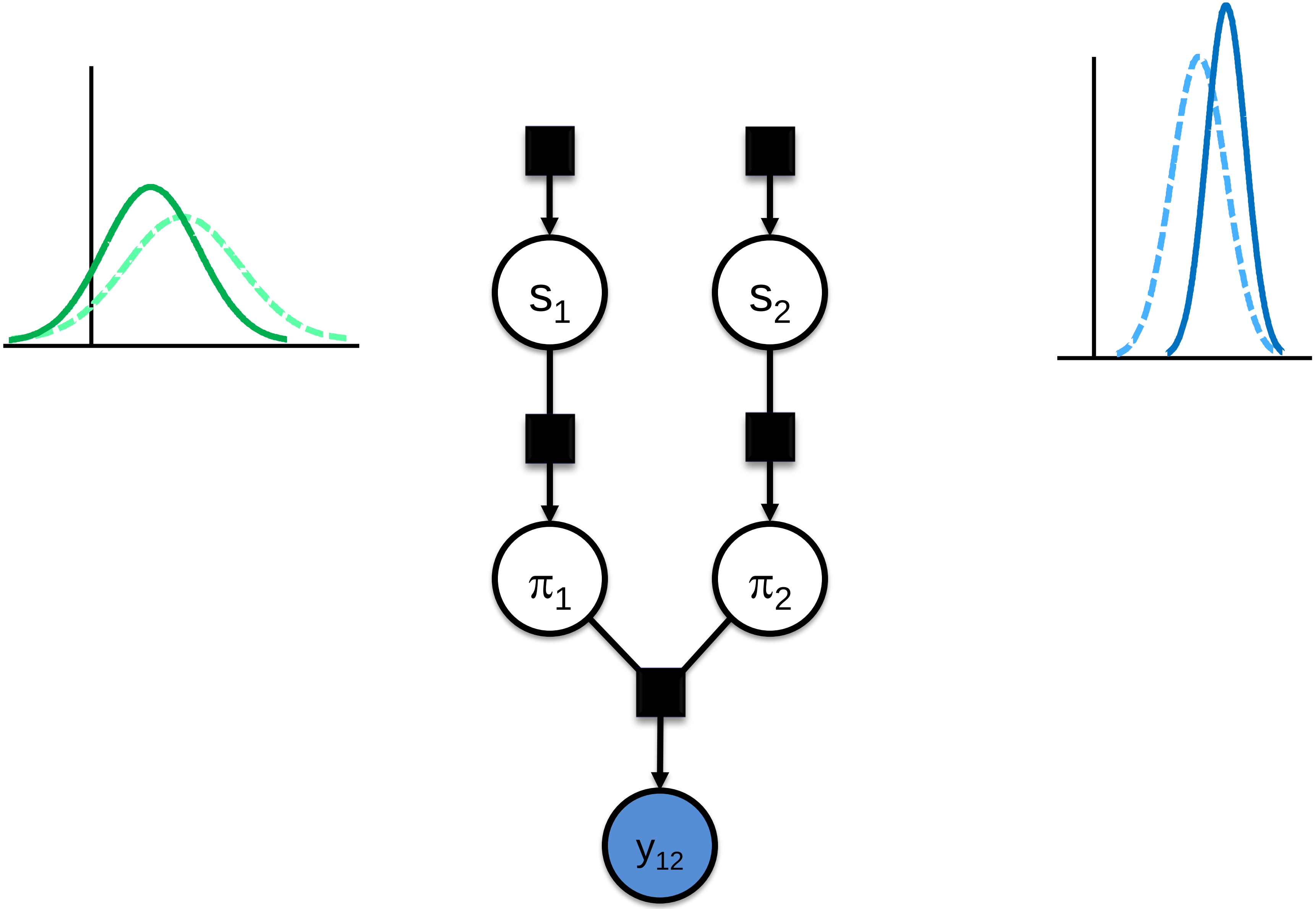

Gaussian
Gaussian
s_1
s_2
Gaussian
Gaussian
π_1
π_2
$I(\pi_1 > \pi_2)$
y_{12}



s_1
s_2
π_1
π_2
y_{12}

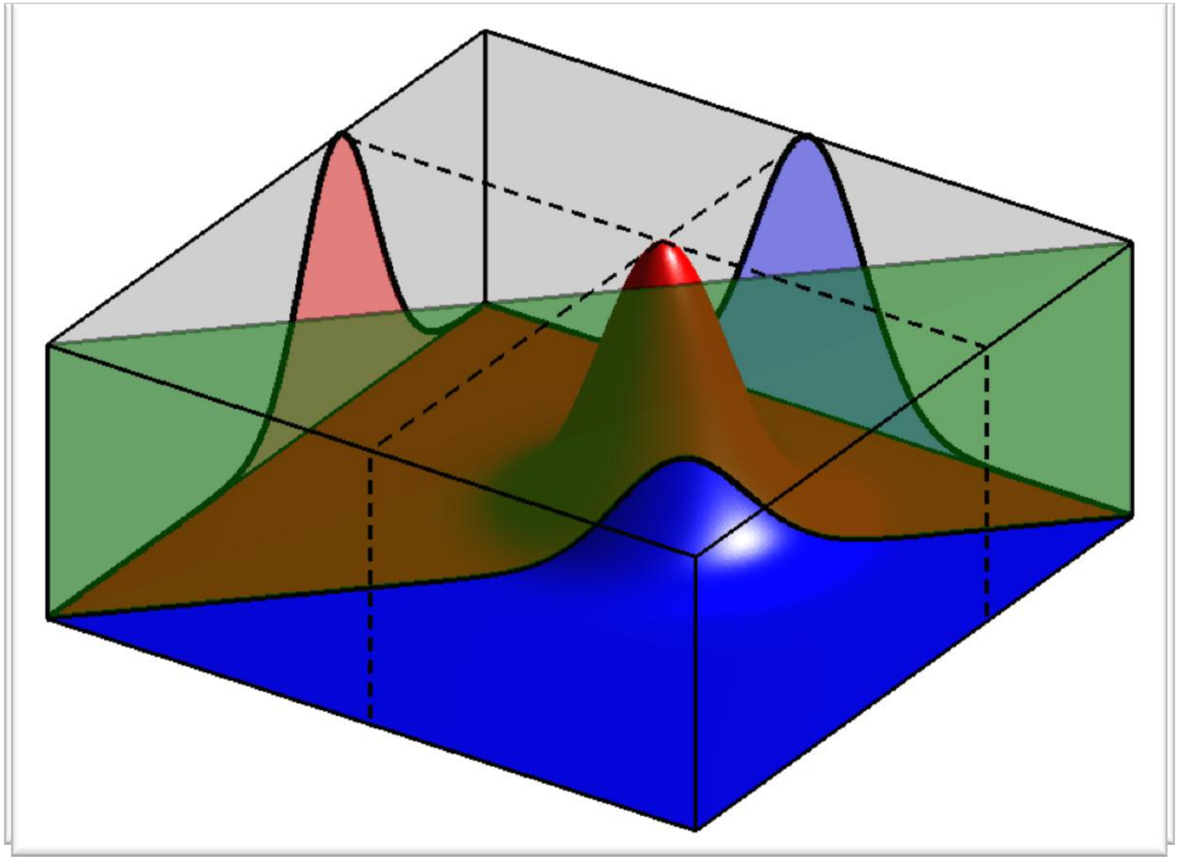


s_1
s_2
π_1
π_2
y_{12}

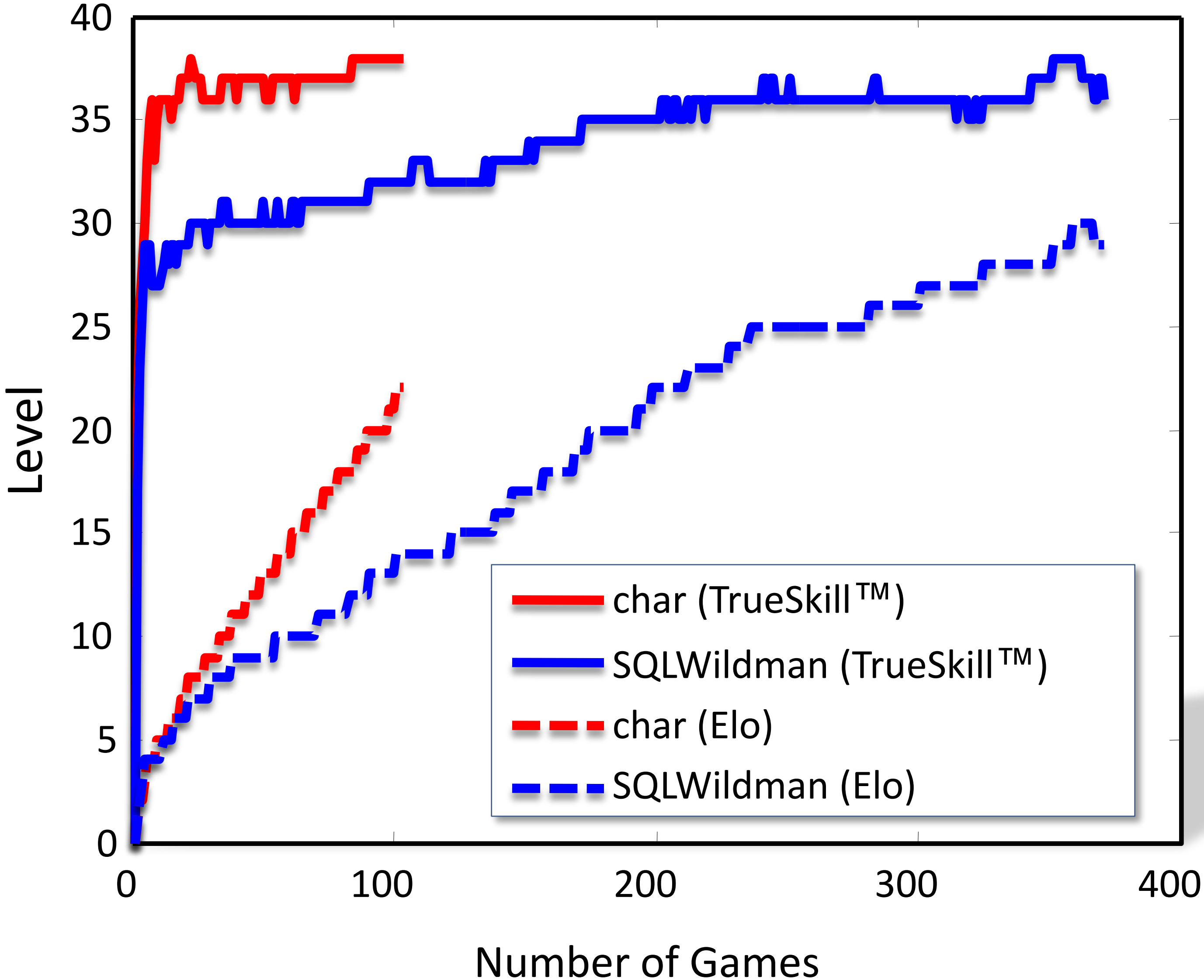


s_1
s_2
π_1
π_2
y_{12}

Expectation propagation (EP)



Convergence



13. Probabilistic Programming

CSOFT

A representation language for probabilistic models.

Takes C# and adds support for:

- **random variables**
- **constraints** on variables
- **inference**

Can be embedded in ordinary C# to allow integration of deterministic + stochastic code

Random variables

Normal variables have a fixed single value

```
int length=6
```

Random variables have a probability distribution

```
int length = random(Uniform(0,10))
```

Constraints

- Constraints on random variables

```
constrain(visible==true)

constrain(length==4)

constrain(length>0)

constrain(i==j)
```

Inference

Compute posterior distribution

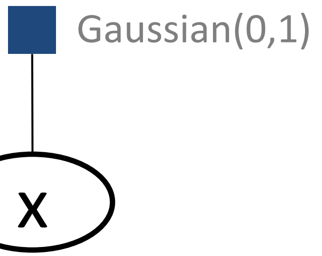
```
int i = random(Uniform(1,10));

bool b = (i*i>50);

Dist bdist = infer(b); //Bernoulli(0.3)
```

Random variables

Probabilistic program

```
double x = random(Gaussian(0,1));
```

Graphical model

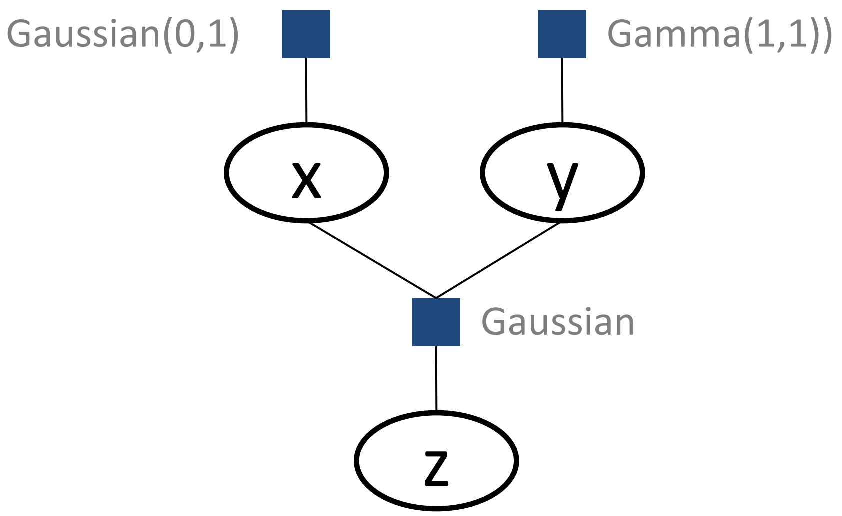


Bayesian networks

Probabilistic program

```
double x = random(Gaussian(0,1));
double y = random(Gamma(1,1));
double z = random(Gaussian(x,y));
```

Graphical model

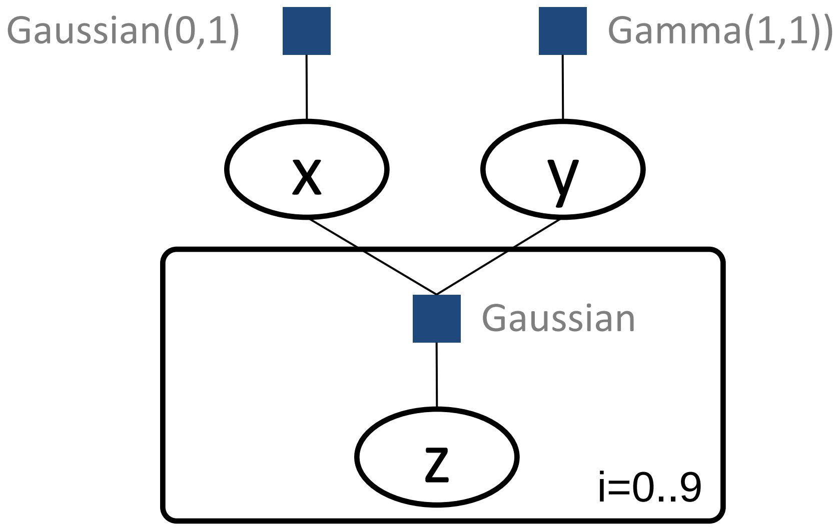


Loops → plates

Probabilistic program

```
double x = random(Gaussian(0,1));
double y = random(Gamma(1,1));
for(int i=0;i<10;i++) {
  double z = random(Gaussian(x,y));
}
```

Graphical model

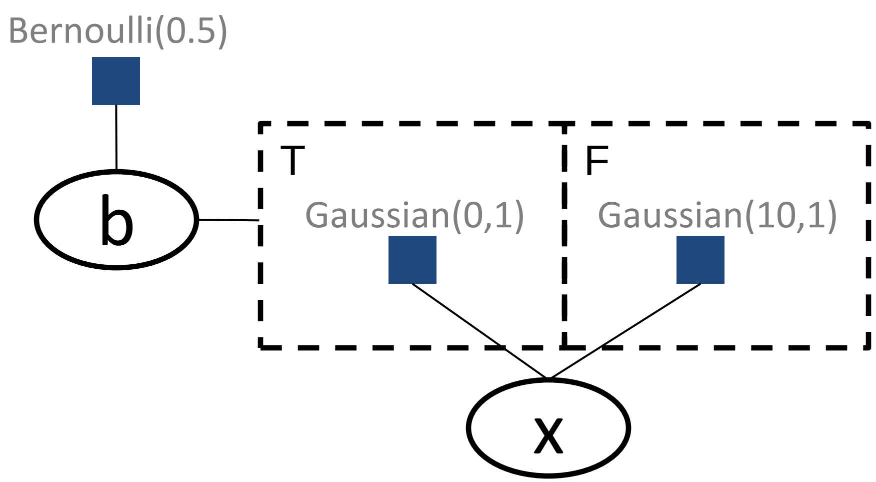


If statement → gates

Probabilistic program

```
bool b = random(Bernoulli(0.5)); double x;
if (b) {
  x = random(Gaussian(0,1));
} else {
  x = random(Gaussian(10,1));
}
```

Graphical model



Gates (Minka and Winn, NIPS 2008)

$$a \perp\!\!\!\perp b \mid c$$

Other language features

Probabilistic program

- Functions/recursion
- Indexing
- Jagged arrays
- Mutation: `x=x+1`
- Objects
- ...

Graphical model

<u>No</u> common equivalent

Sampling interpretation

Imagine running program many times, where

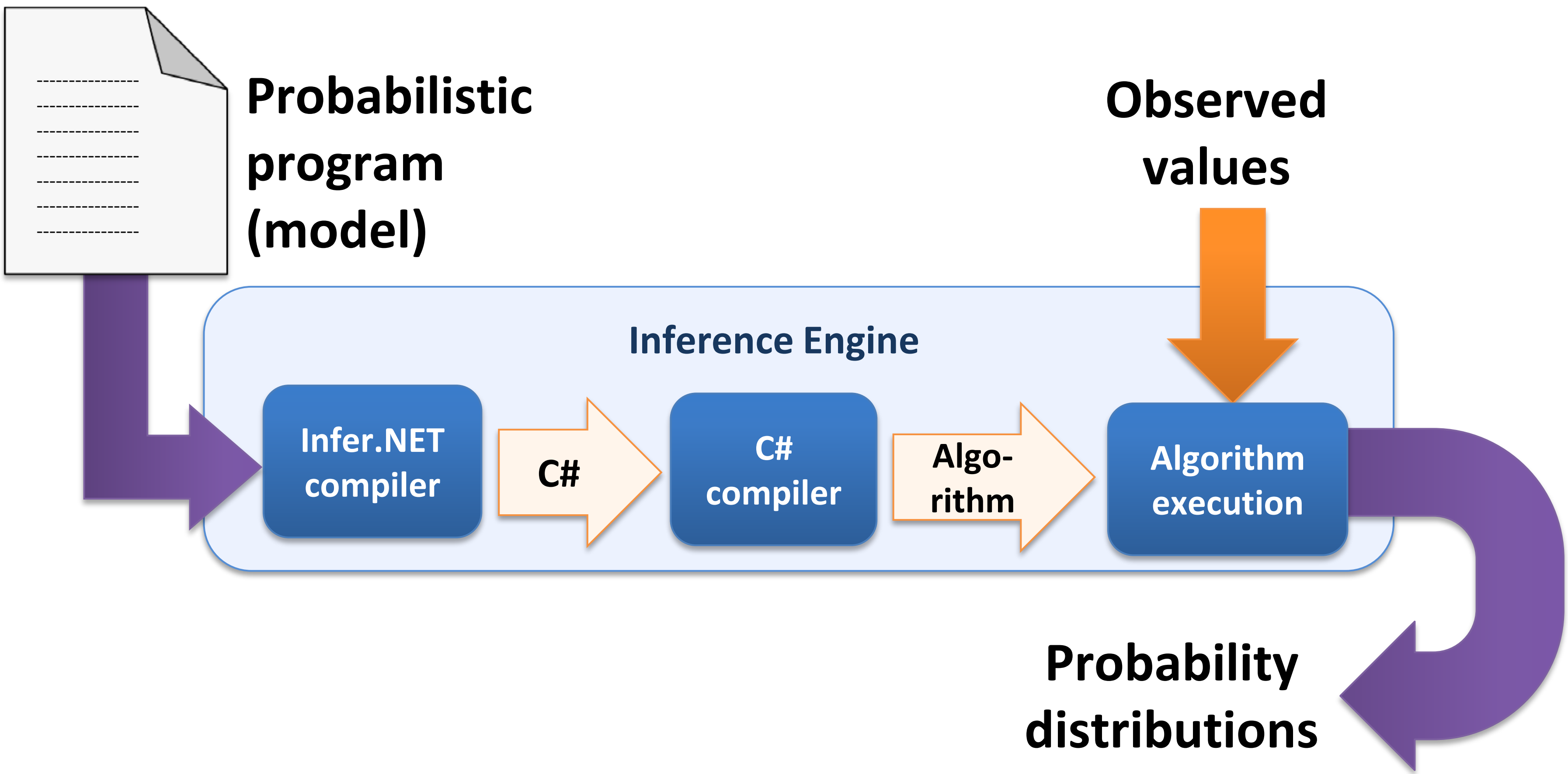
- **`random`**`(dist)` draws a random number from dist
- **`constrain`**`(b)` stops the run if b is not true
- **`infer`**`(x)` accumulates the value of x into memory

infer.net

http://research.microsoft.com/infernet

John Winn, Tom Minka, John Guiver, *et al.*

How Infer.NET works



Standard models supported

Mixture models

Factor analysis / PCA / ICA

Logistic regression

Discrete Bayesian networks

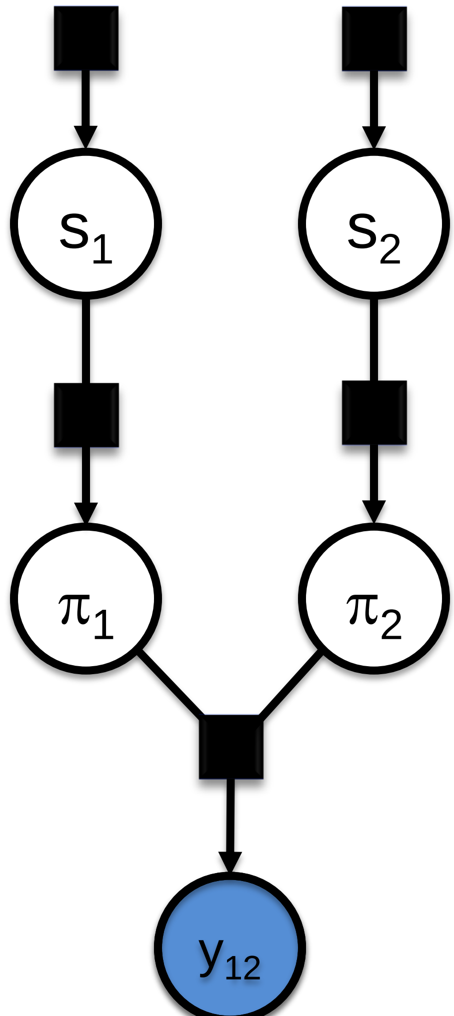
Hidden Markov models

Ranking models

Kalman filters

Hierarchical models

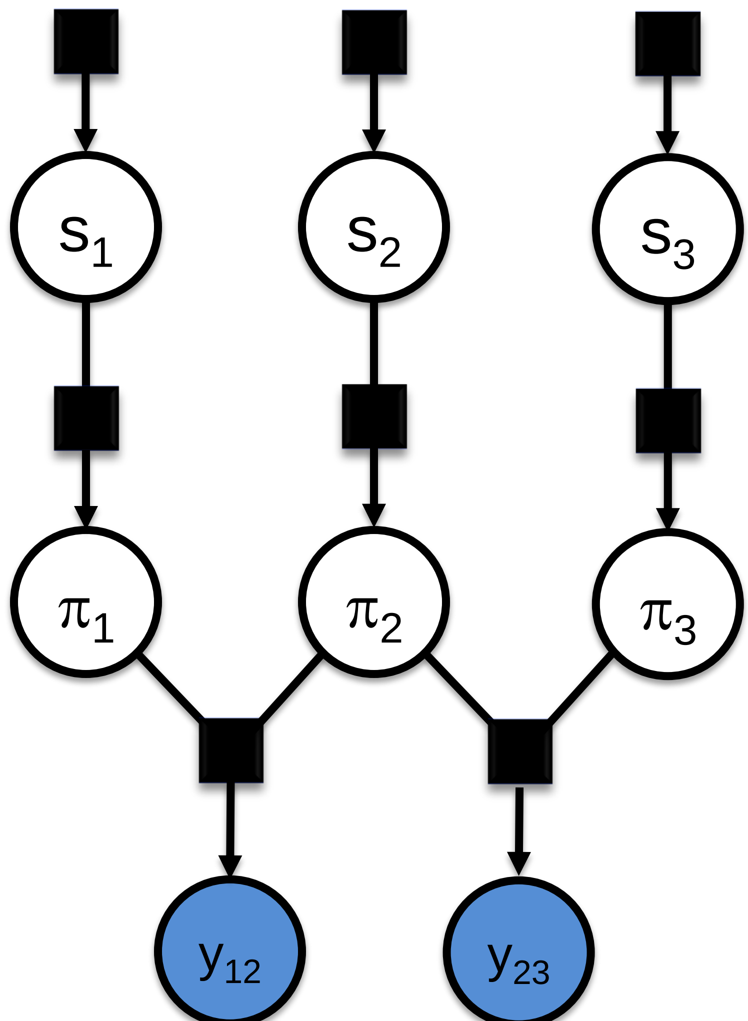
…



s_1
s_2
π_1
π_2
y_{12}

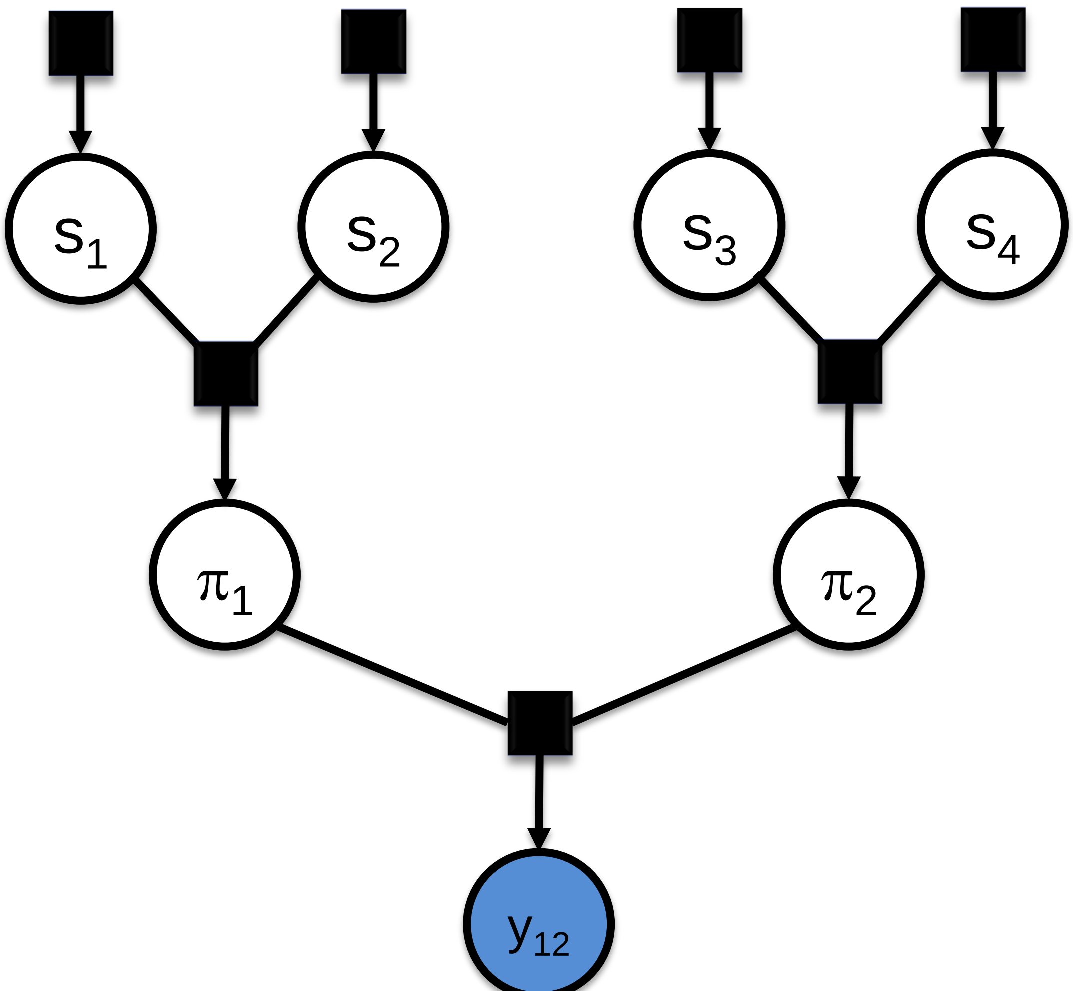
```
// model variables
Variable<double> skill1, skill2;
Variable<double> performance1, performance2;
Gaussian skillPosterior1, skillPosterior2;

// model
skill1 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill2 = Variable.GaussianFromMeanAndPrecision(0, 1);

performance1 = Variable.GaussianFromMeanAndPrecision(skill1, beta);
performance2 = Variable.GaussianFromMeanAndPrecision(skill2, beta);

Variable.ConstrainPositive(performance1 - performance2);

// infer new posterior skills
InferenceEngine engine = new InferenceEngine();

skillPosterior1 = engine.Infer<Gaussian>(skill1);
skillPosterior2 = engine.Infer<Gaussian>(skill2);
```

Extension to Multiple players



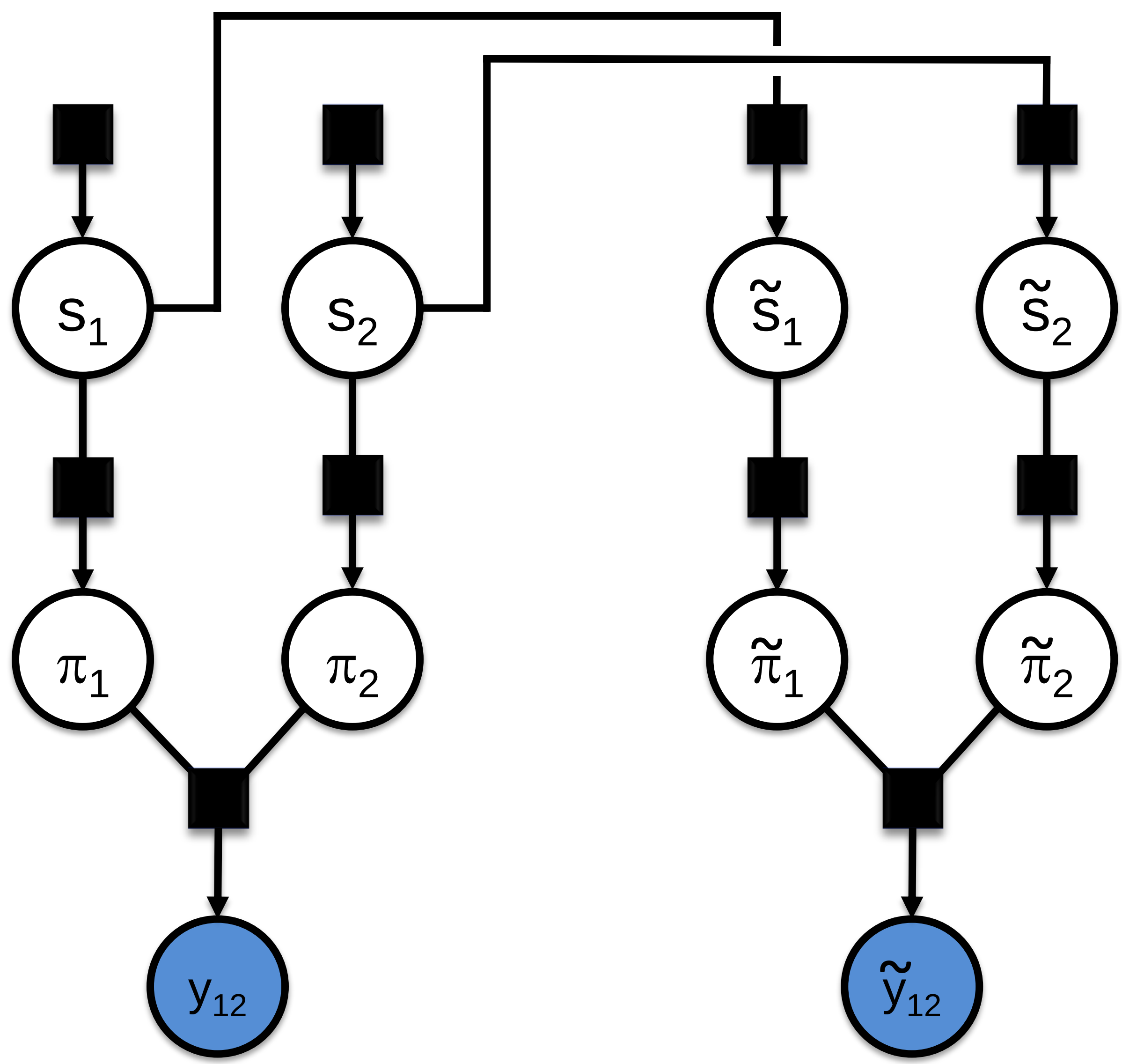
```
// model variables
Variable<double> skill1,         skill2,         skill3;
Variable<double> performance1,   performance2,   performance3;
Gaussian         skillPosterior1,skillPosterior2, skillPosterior3;

// model
skill1 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill2 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill3 = Variable.GaussianFromMeanAndPrecision(0, 1);

performance1 = Variable.GaussianFromMeanAndPrecision(skill1, beta);
performance2 = Variable.GaussianFromMeanAndPrecision(skill2, beta);
performance3 = Variable.GaussianFromMeanAndPrecision(skill3, beta);

Variable.ConstrainPositive(performance1 - performance2);
Variable.ConstrainPositive(performance2 - performance3);

// infer new posterior skills
InferenceEngine engine = new InferenceEngine();

skillPosterior1 = engine.Infer<Gaussian>(skill1);
skillPosterior2 = engine.Infer<Gaussian>(skill2);
skillPosterior3 = engine.Infer<Gaussian>(skill3);
```

Extension to Teams



```
// model variables
Variable<double> skill1, skill2, skill3, skill4;
Variable<double> performance1, performance2 , performance3, performance4;
Gaussian skillPosterior1,skillPosterior2, skillPosterior3, skillPosterior4;

// model
skill1 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill2 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill3 = Variable.GaussianFromMeanAndPrecision(0, 1);
skill4 = Variable.GaussianFromMeanAndPrecision(0, 1);

performance1 = Variable.GaussianFromMeanAndPrecision(skill1 + skill2, beta);
performance2 = Variable.GaussianFromMeanAndPrecision(skill3 + skill4, beta);

Variable.ConstrainPositive(performance1 - performance2);

// infer new posterior skills
InferenceEngine engine = new InferenceEngine();

skillPosterior1 = engine.Infer<Gaussian>(skill1);
skillPosterior2 = engine.Infer<Gaussian>(skill2);
skillPosterior3 = engine.Infer<Gaussian>(skill3);
skillPosterior4 = engine.Infer<Gaussian>(skill4);
```

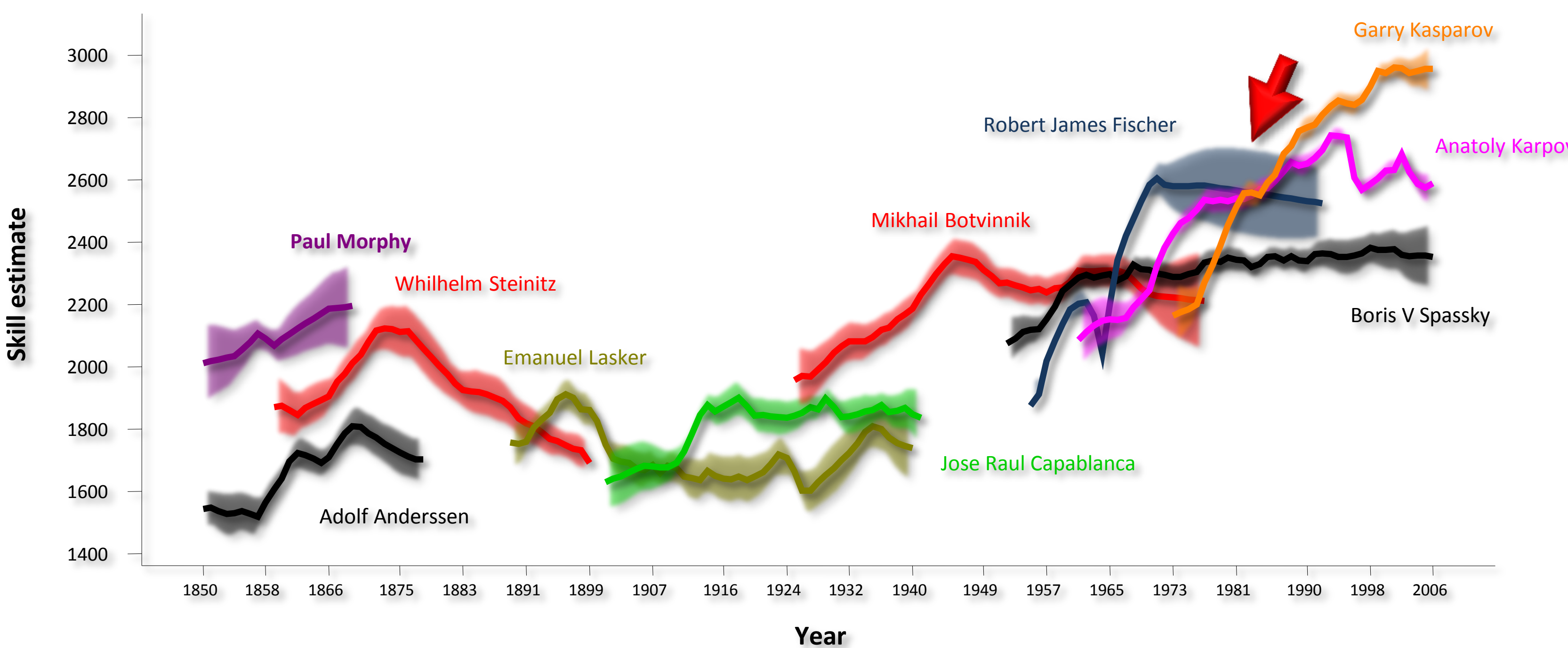
TrueSkill™ through time



```
// model variables
Variable<double> skill1, skill2;
Variable<double> performance1, performance2;
Gaussian skillPosterior1, skillPosterior2;

// model
skill1 = Variable.GaussianFromMeanAndPrecision(oldskill1, alpha);
skill2 = Variable.GaussianFromMeanAndPrecision(oldskill2, alpha);

performance1 = Variable.GaussianFromMeanAndPrecision(skill1, beta);
performance2 = Variable.GaussianFromMeanAndPrecision(skill2 ,beta);

Variable.ConstrainPositive(performance1 - performance2);

// infer new posterior skills
InferenceEngine engine = new InferenceEngine();

skillPosterior1 = engine.Infer<Gaussian>(skill1);
skillPosterior2 = engine.Infer<Gaussian>(skill2);
```

ChessBase Analysis: 1850 - 2006

3.5M game outcomes
20 million variables (200,000 players in each year of lifetime + latent variables)
40 million factors



DARPA ENVISIONS THE FUTURE OF MACHINE LEARNING

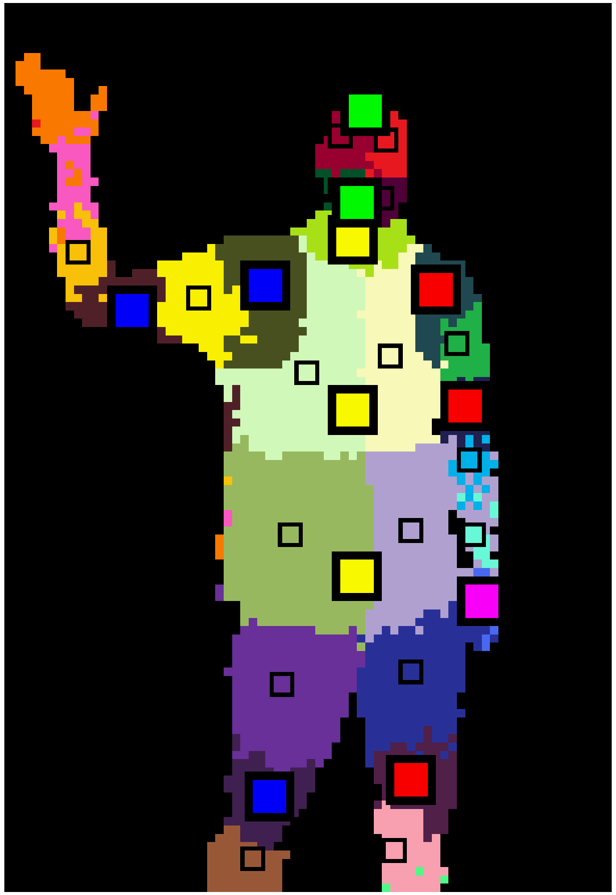
March 19, 2013

Automated tools aim to make it easier to teach a computer than to program it

Machine learning – the ability of computers to understand data, manage results, and infer insights from uncertain information – is the force behind many recent revolutions in computing. Email spam filters, smartphone personal assistants and self-driving vehicles are all based on research advances in machine learning. Unfortunately, even as the demand for these capabilities is accelerating, every new application requires a Herculean effort. Even a team of specially-trained machine learning experts makes only painfully slow progress due to the lack of tools to build these systems.

The Probabilistic Programming for Advanced Machine Learning (PPAML) program was launched to address this challenge. Probabilistic programming is a new programming paradigm for managing uncertain information. By incorporating it into machine learning, PPAML seeks to greatly increase the number of people who can successfully build machine learning applications and make machine learning experts radically more

Any questions?



Thank you!