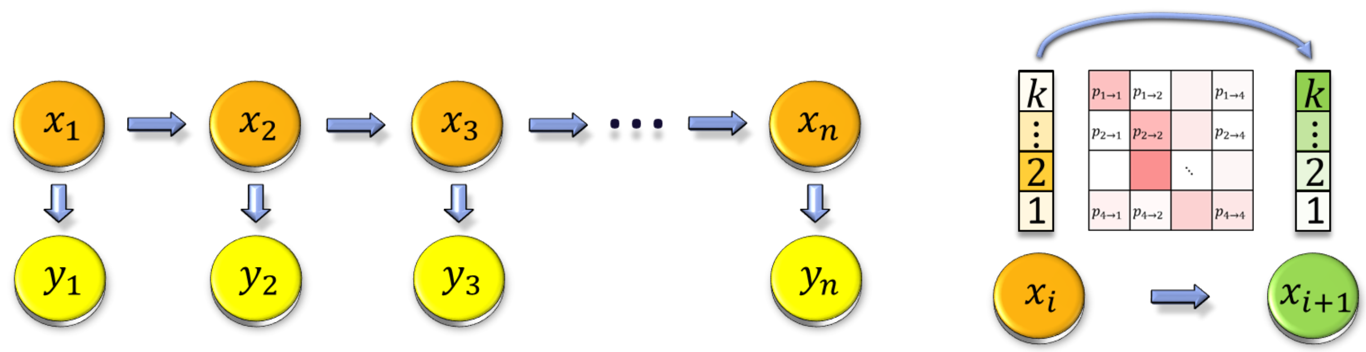
# Modelling 2

## STATISTICAL DATA MODELLING




Chapter 8

# Markovian Models

Michael Wand · Institut für Informatik, JGU Mainz · michael.wand@uni-mainz.de

Video #08

# Markovian Models

- **Markov Chains**
- **Hidden Markov Models**
- **Markov Random Fields**

## Video #08

# Graphical Models

## tl;dw

- Unrestricted statistical dependencies: intractable
- Markovian models
  - Direct dependency only on “neighbors”
  - Neighbors in time / space / something
  - Density factorizes in local terms
- Common model (e.g.: physics, natural science)

# Statistical Dependencies

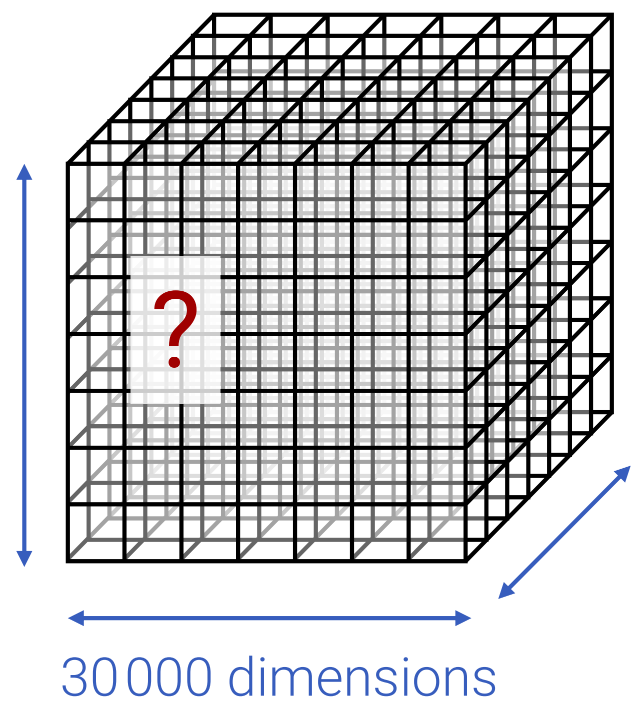
# Problem

## Parameter Estimation Problem

$$P(X|D) \sim P(D|X)\,P(X)$$

posterior — data term, likelihood — prior

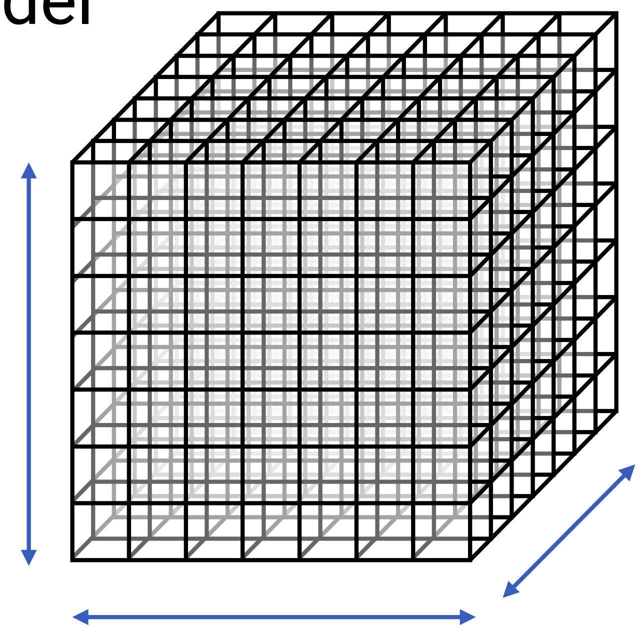
- $X$ = digital image (10K pixels)
- $D$ = noisy photo (or the like)
- Assume P($D$|$X$) is known
- Model P($X$) cannot be build
  - Not even enough training data
  - In this part of the universe :-)



# Reducing dependencies

## Problem:

- $p(x_1, x_2, \ldots, x_{30000})$ is to high-dimensional
- $k$ States, $n$ variables: $O(k^n)$ density entries
- General dependencies kill the model



## Idea

- Hand-craft dependencies
- Guess or know
  what depends on each other
- “The art of machine learning”

# Markov-Chains

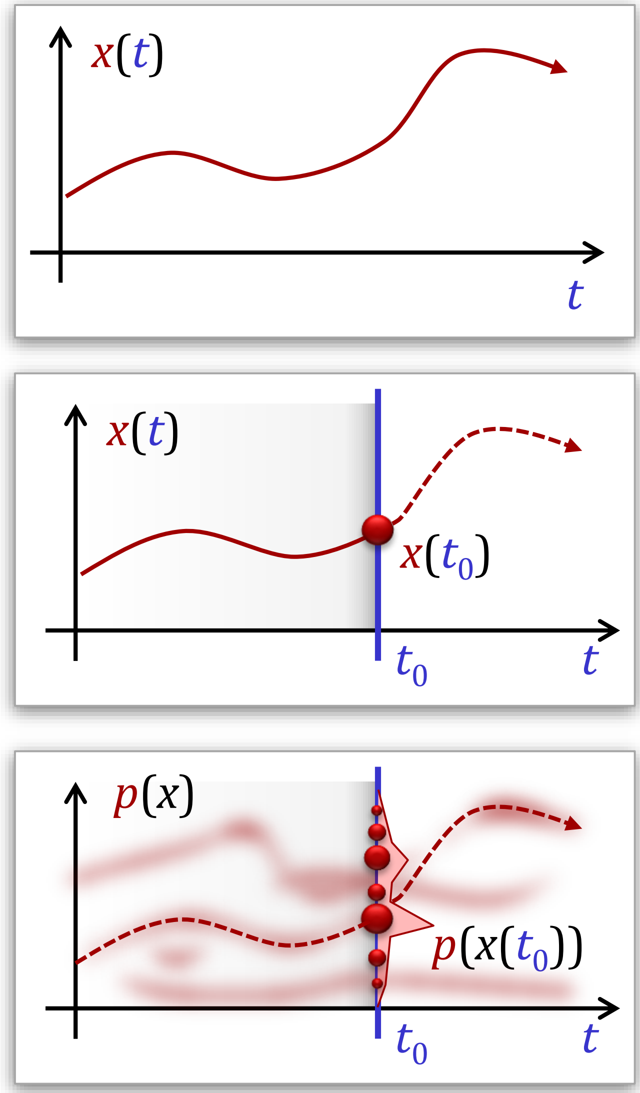
Andrei Andrejewitsch Markow (1870s)

[unknown photographer, https://commons.wikimedia.org/wiki/File:AAMarkov.jpg]

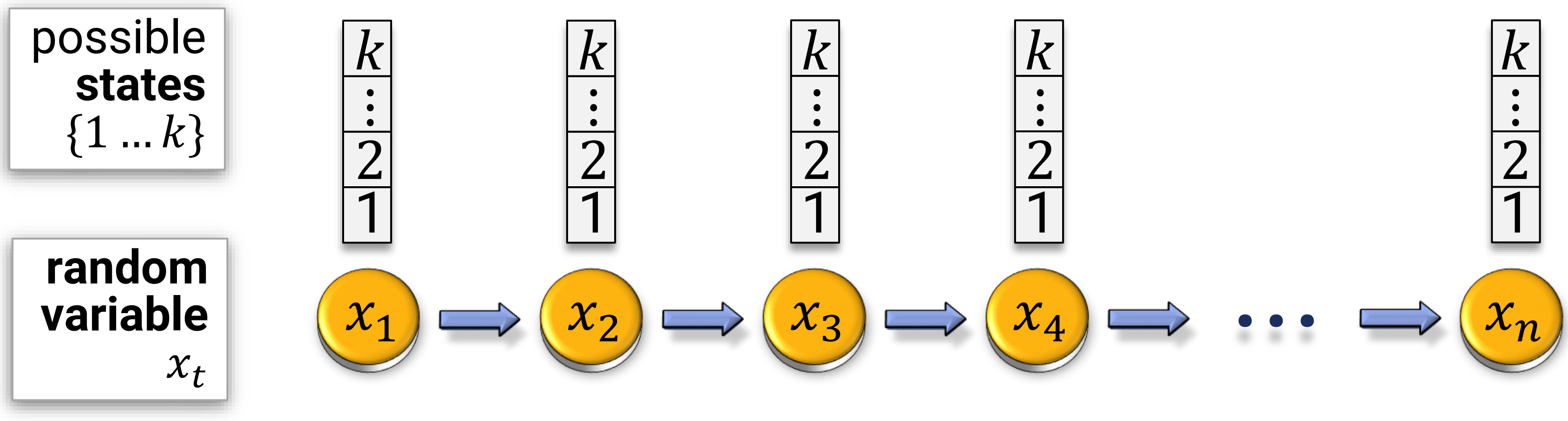
# Motivation

## Markovian Models

- Typically:
  time-dependent processes
- Physics
  - System state evolves
  - Current state determines future states
- Statistical model
  - Like an ODE with uncertainty



# Markov Chain

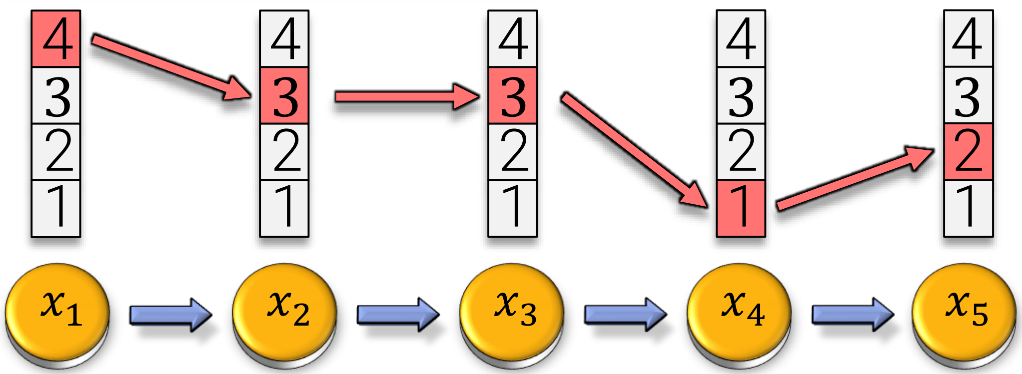



## Discrete Model

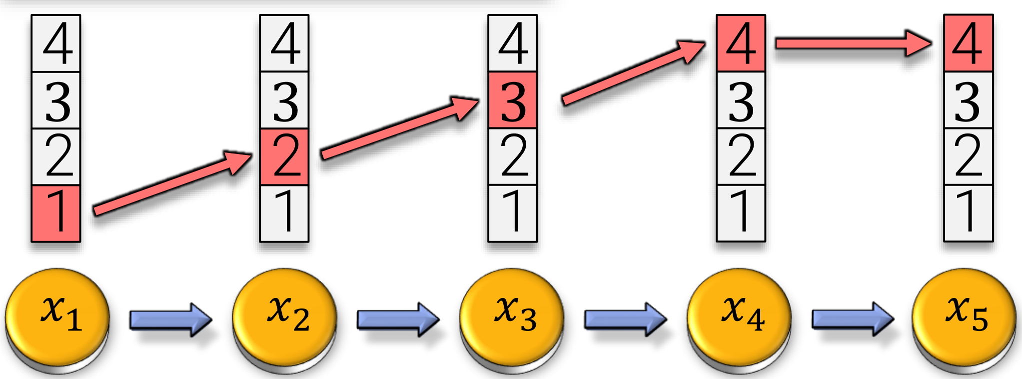
- Time steps: $t \in \{1,2,3,\ldots,n\}$
- Random variables: $x_t \in \{1,\ldots,k\}$
- Probability space: $\Omega = \mathcal{P}(\{1..k\}^n)$
- Probability measure: $p: \mathcal{P}(\{1..k\}^n) \to \mathbb{R}^+$

# $n$ Random Variables

outcome $\mathbf{x} = (4,3,3,1,2)$




outcome $\mathbf{y} = (1,2,3,4,4)$




**prob. distribution:**

$p(1,1,1,1,1) = 0.01$

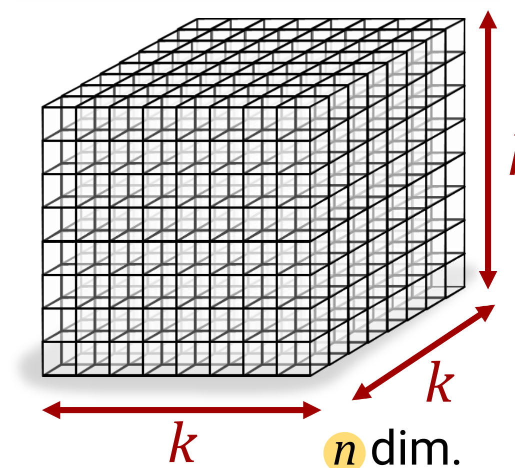
$p(1,1,1,1,2) = 0.0$

$p(1,1,1,1,3) = 0.2$

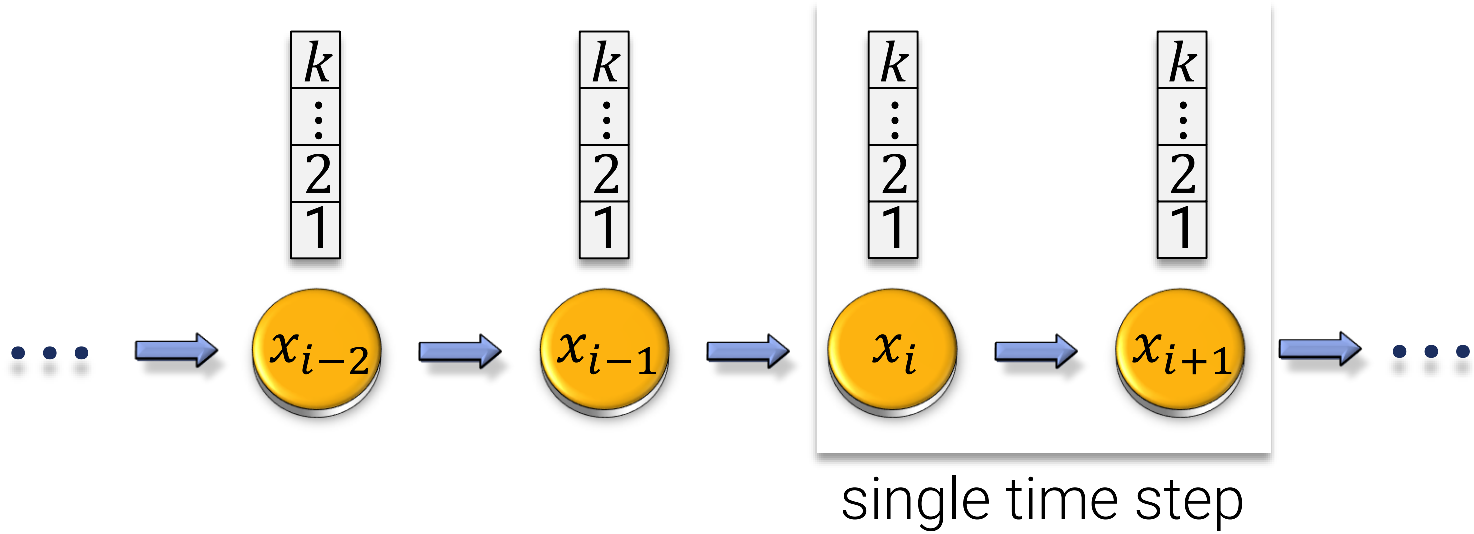
...

$p(4,4,4,4,4) = 0.05$

$\mathcal{O}(k^n)$ **outcomes**



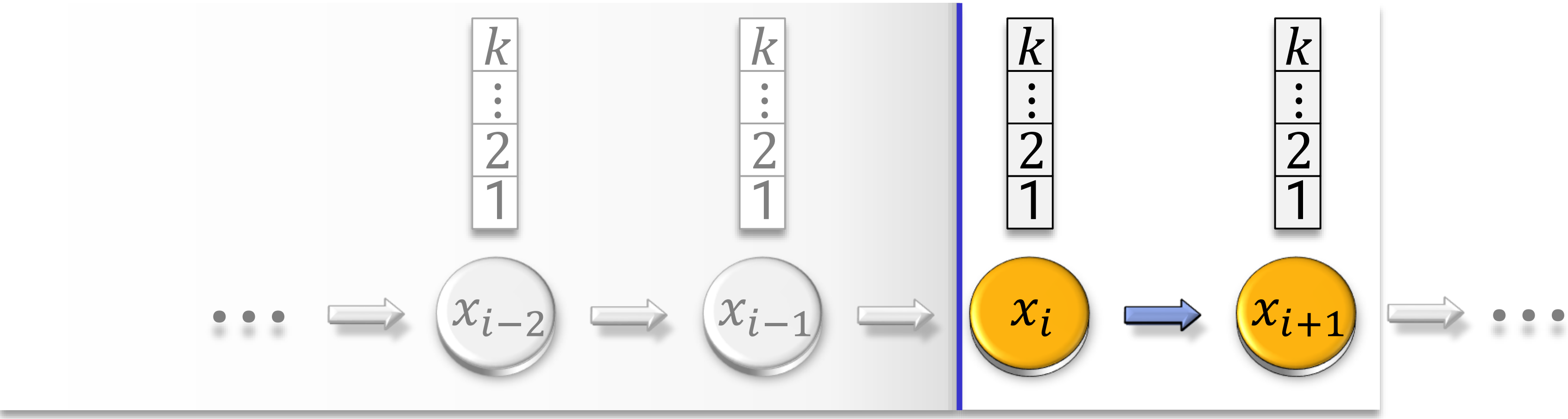
# Markov Property




## Markov property

- Simplifies model
- Present time determines/fixes the future
  - For step $i \rightarrow i+1$: We can ignore all previous steps
- Formally: $p(x_{i+1} | x_1, \ldots, x_i) = p(x_{i+1} | x_i)$

# Markov Property

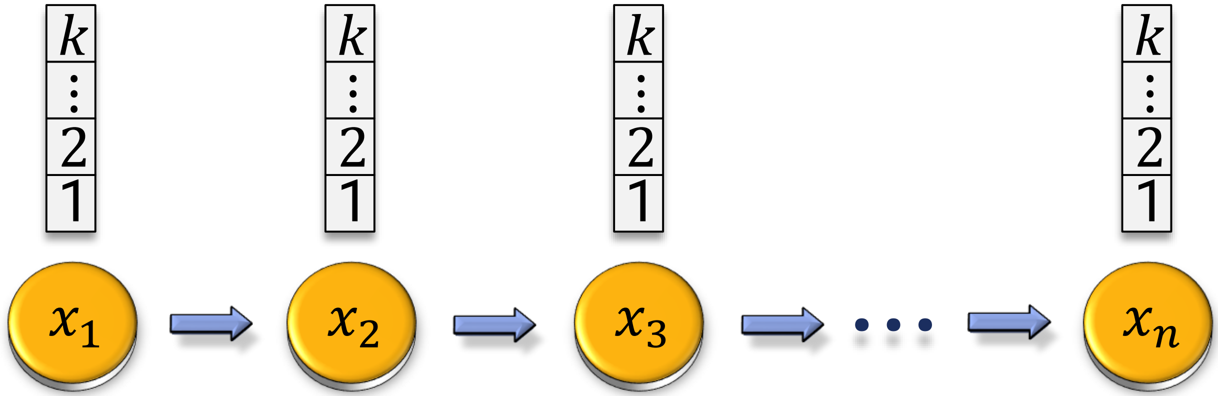



single time step

## Markov property

- Simplifies model
- Present time determines/fixes the future
  - For step $i \rightarrow i+1$: We can ignore all previous steps
- Formally: $p(x_{i+1} | x_1, \ldots, x_i) = p(x_{i+1} | x_i)$

# Markov Property



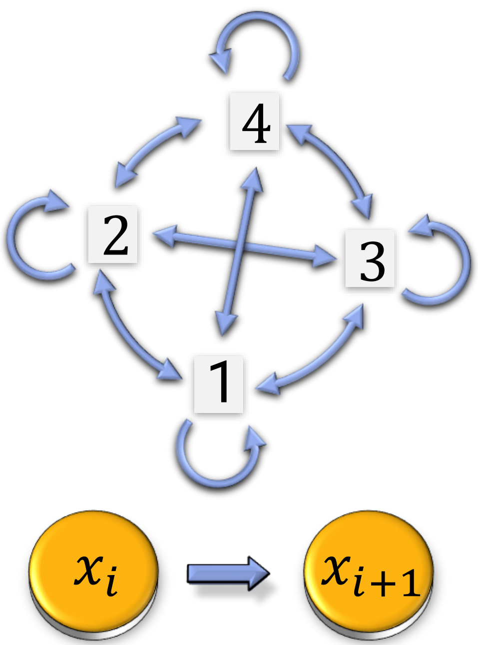

## Markov property

- Factorization

$$p(x_1, x_2, \dots, x_n) \sim p_0(x_1) \prod_{i=1}^{n-1} p(x_{i+1} | x_i)$$

- Proof: Apply (substitute) repeatedly

# Transition Matrices



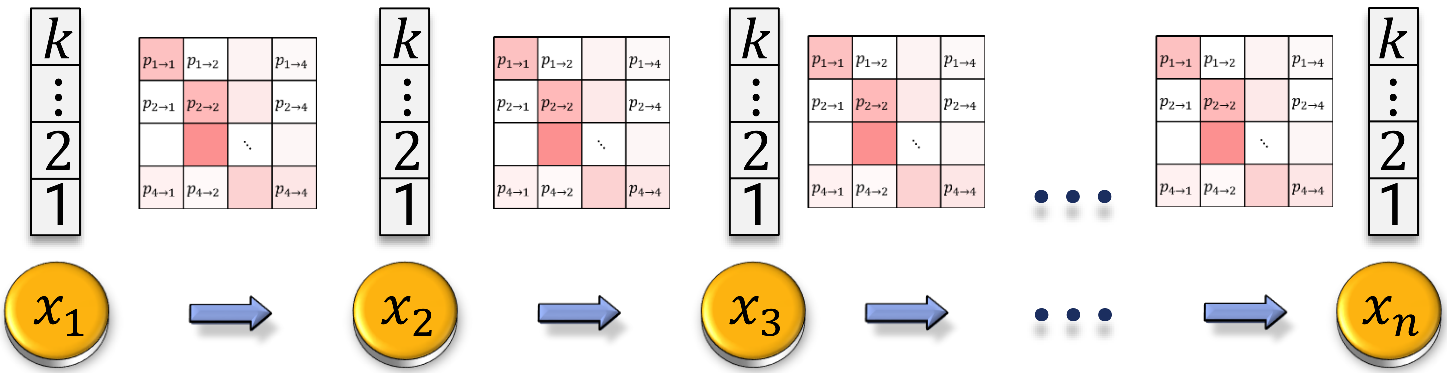

**state transitions**

| | 1 | 2 | ⋯ | $k$ |
|---|---|---|---|---|
| 1 | $p_{1\to1}$ | $p_{1\to2}$ | | $p_{1\to k}$ |
| 2 | $p_{2\to1}$ | $p_{2\to2}$ | | $p_{2\to k}$ |
| ⋮ | | | ⋱ | |
| $k$ | $p_{k\to1}$ | $p_{k\to2}$ | | $p_{k\to k}$ |

**transition matrix**

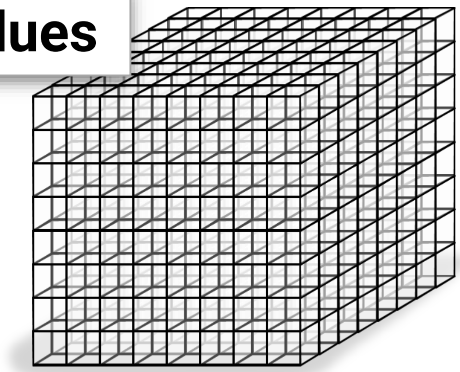
„ $p(x_{i+1}|x_i)$ "

# Transition Matrices



# Reduction of Complexity

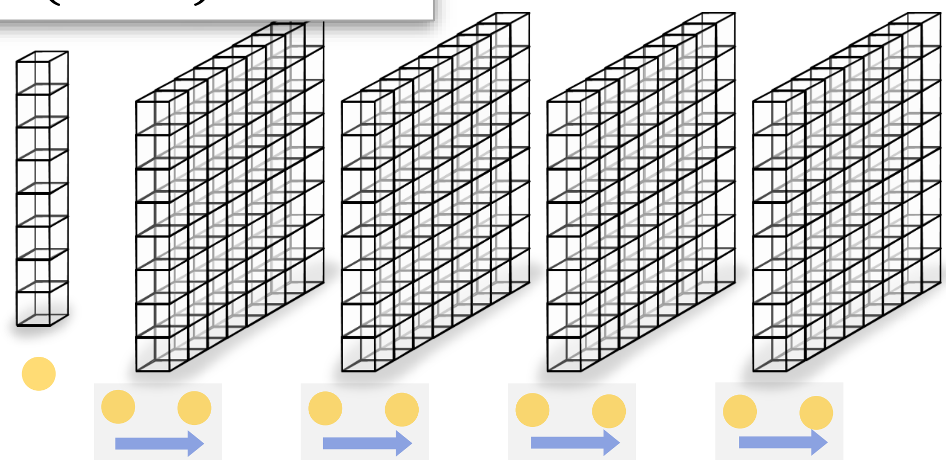
$\mathcal{O}(k^n)$ **values**



**general model**
arbitrary dependencies

$$p(x_1, x_2, \dots, x_n)$$

$\mathcal{O}(k^2 \cdot \mathrm{n})$ **values**



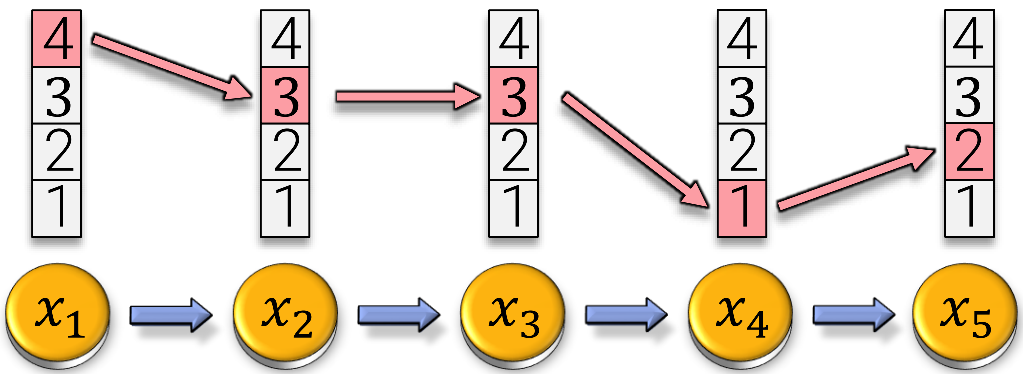
**Markov chain**
factorization in pairs

$$p(x_1, x_2, \dots, x_n)$$

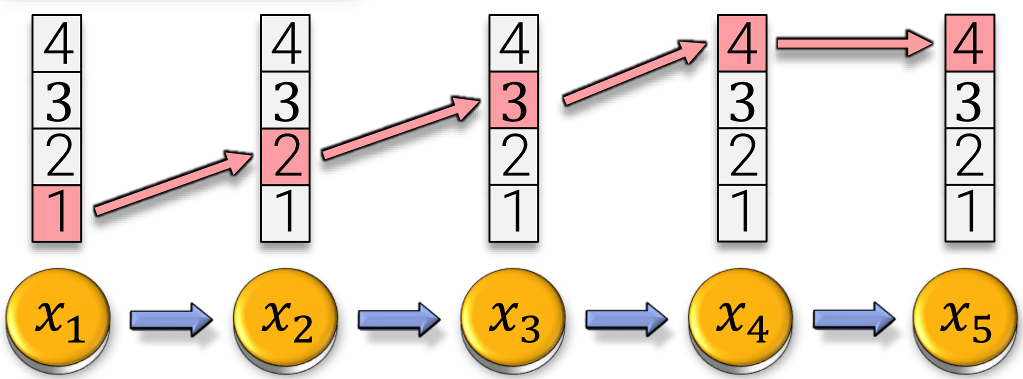
$$\sim p_0(x_1) \prod_{i=1}^{n-1} p(x_{i+1} | x_i)$$

# Markov-Chain

$\mathbf{x} = (4,3,3,1,2)$



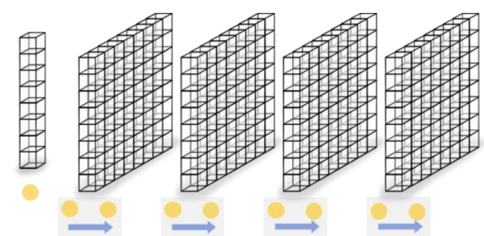

$\mathbf{y} = (1,2,3,4,4)$




**Still**

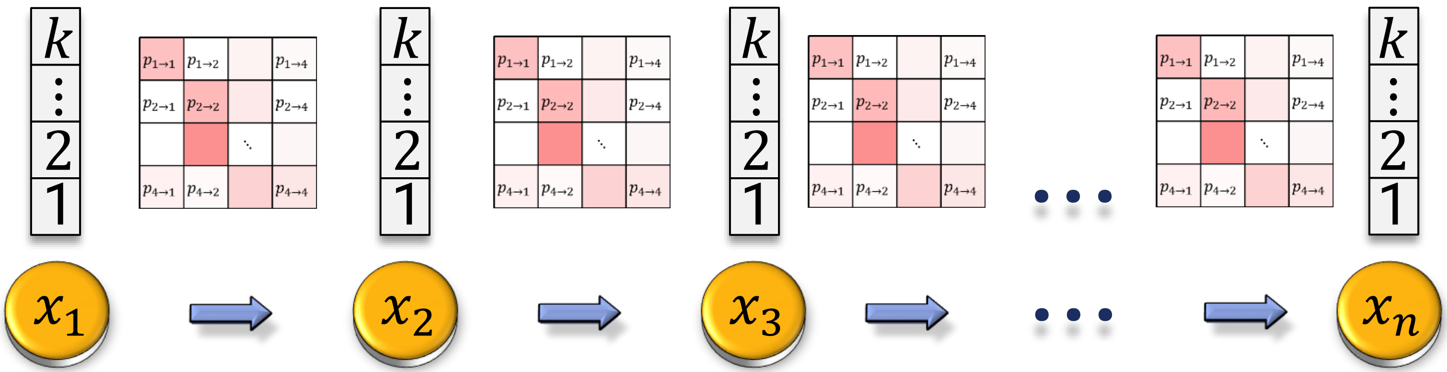
*a probability*
for *each chain*

restriction is on
expressing $p(\mathbf{x})$



$p(\mathbf{x})$ – factorized

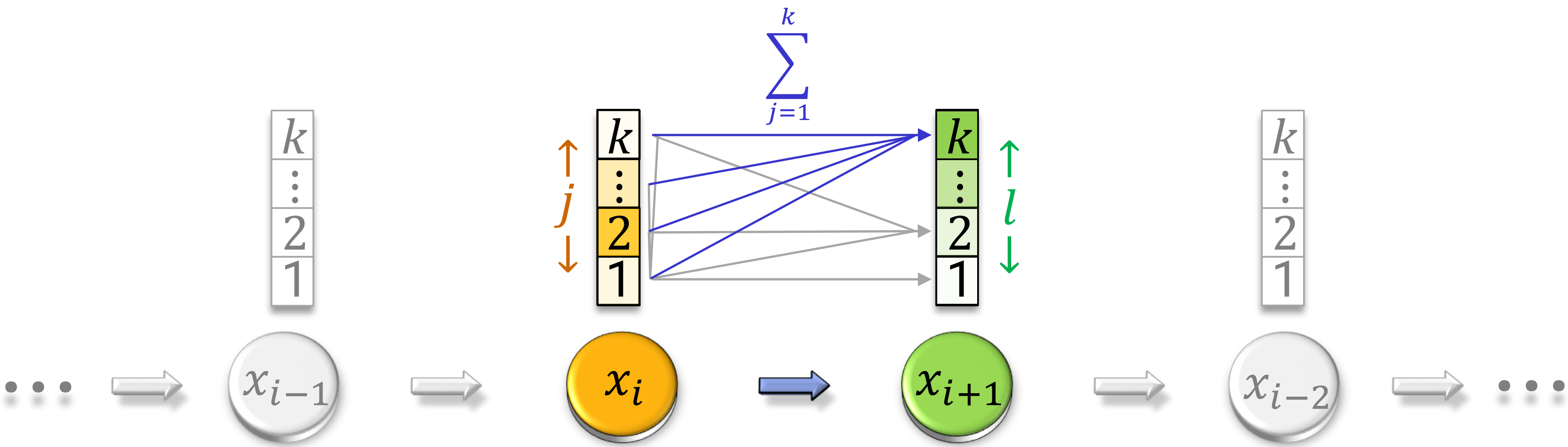
# Model Complexity




## Modell Complexity

- $k$ States, $n$ time steps
- Transition matrices determine model
  - Markov model has $\mathcal{O}(k^2 \cdot n)$ parameters
    - – Initial distribution + transition matrices
  - General model (unrestricted parameters) has $\mathcal{O}(k^n)$ parameters

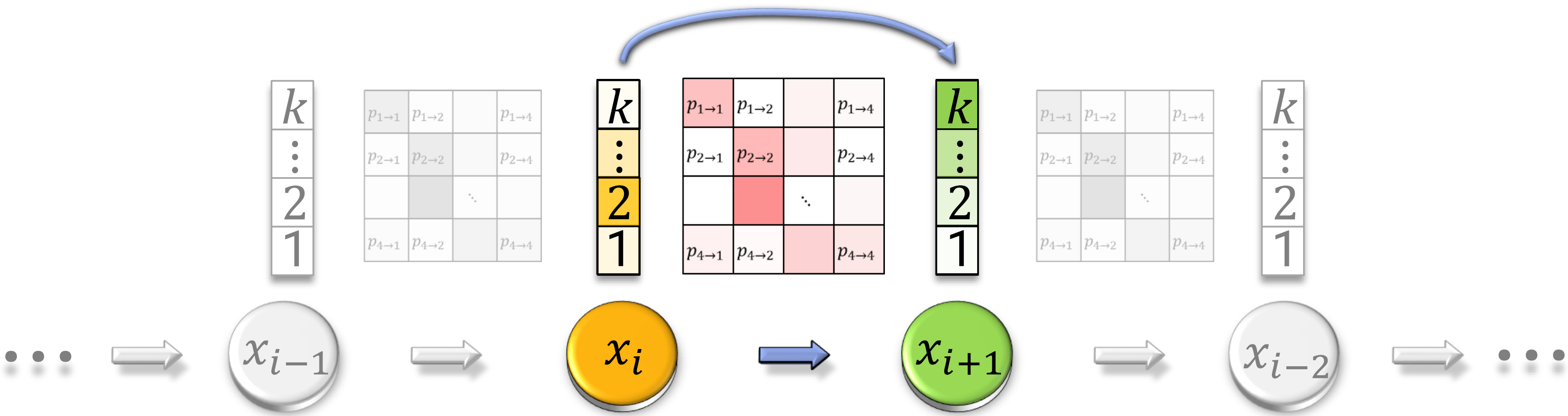
# Transition Matrices



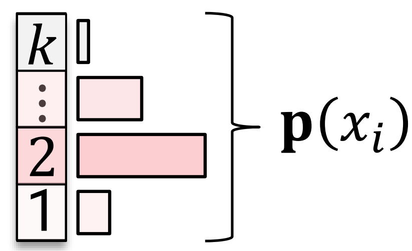

## Formally

- $p(x_{i+1} = l) = \sum_{j=1}^{k} p(x_{i+1} = l | x_i = j) p(x_i = j)$
- For each state in $x_{i+1}$:
  - Sum over states in $x_i$, weighted by:
    - Previous probability $p(x_i = j)$
    - Transition probability $p(x_{i+1} = l | x_i = j)$

# Transition Matrices




## Matrix notation




- $\mathbf{p}(x_i)$ as vector
- Transition as matrix
  $\mathbf{p}(x_{i+1}) = \mathbf{M}_{i\to i+1}\mathbf{p}(x_i)$

# Summary

# Markov Chains

**Discrete (time) series**

$$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_i, \dots \in \mathbb{R}^d$$

**Probability depends only on previous step**

$$p(\mathbf{x}_i | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{i-1}) = p(\mathbf{x}_i | \mathbf{x}_{i-1})$$

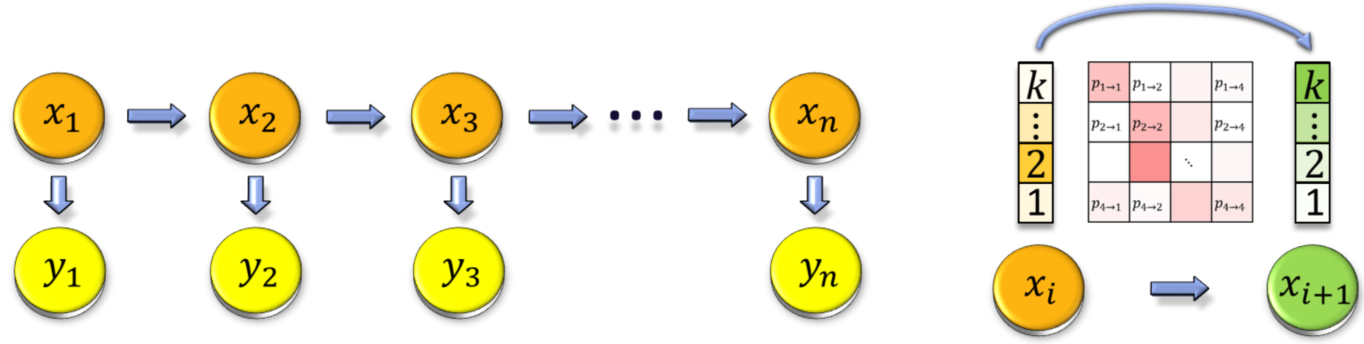
**Equivalenty:** density factorization

$$p(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) = p(\mathbf{x}_{1,}\mathbf{x}_2) \cdot p(\mathbf{x}_{2,}\mathbf{x}_3) \cdots p(\mathbf{x}_{n,}\mathbf{x}_{n-1})$$

**Simplifies probability model** (complexity)

# Modelling 2

## STATISTICAL DATA MODELLING




# Chapter 8

# Markovian Models


Michael Wand · Institut für Informatik, JGU Mainz · michael.wand@uni-mainz.de

Video #08

# Markovian Models

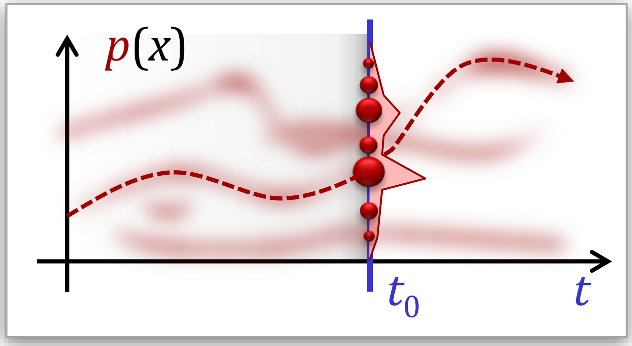
- Markov Chains
- **Hidden Markov Models**
- Markov Random Fields

# Hidden Markov Models

# Terminology

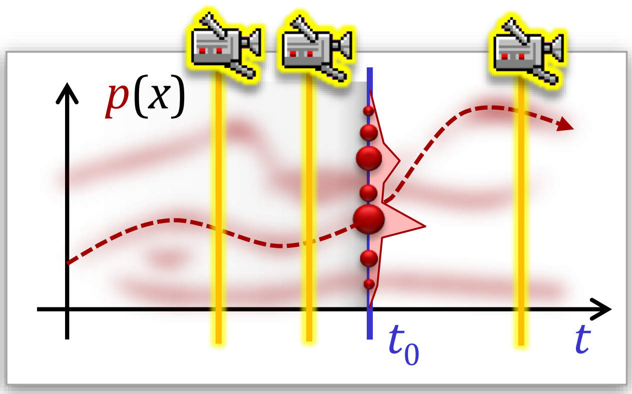
## Markov Chain

- Time dependent process
- Probabilistic model
- Present determines complete future

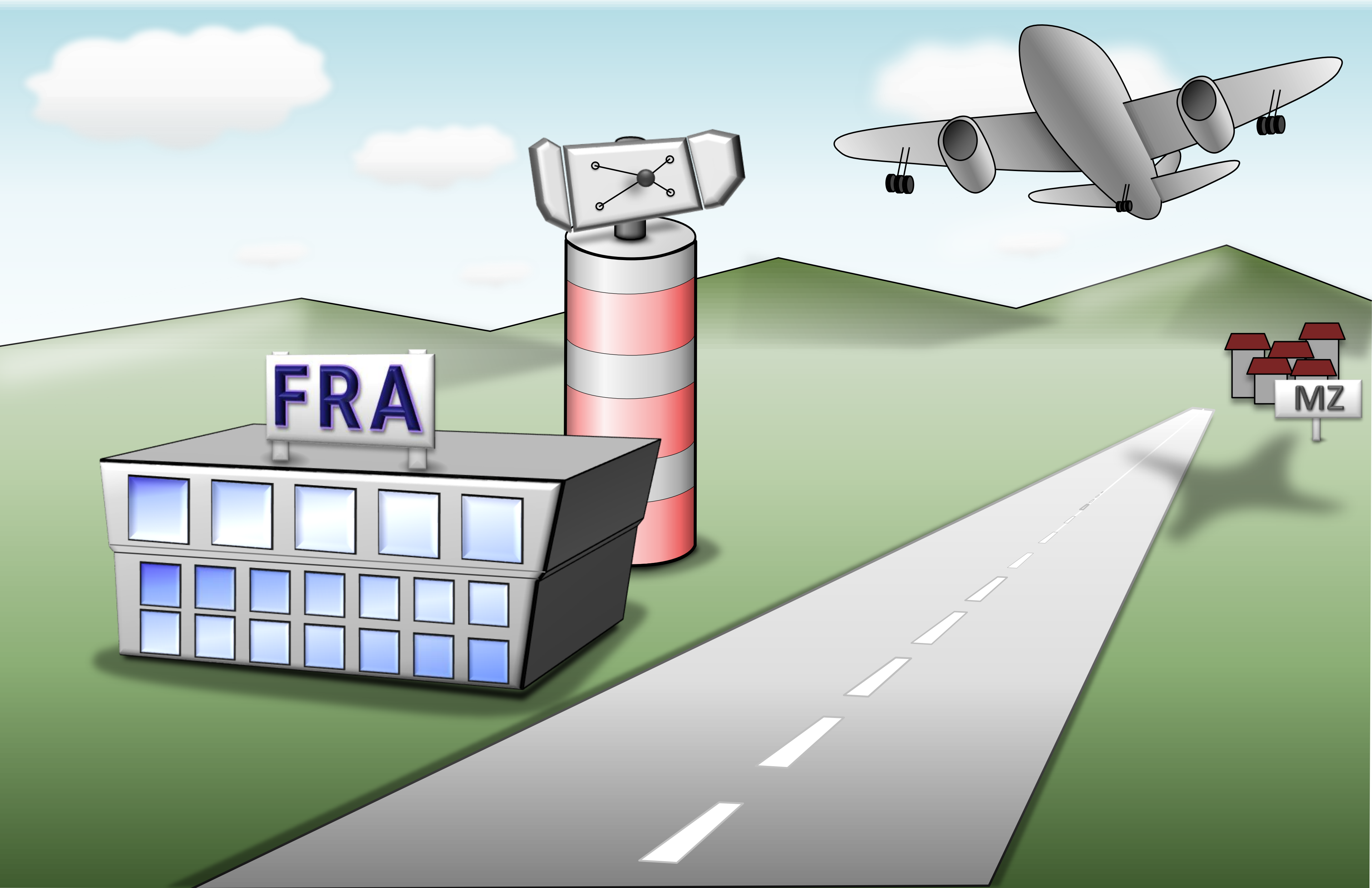



## Hidden Markov Model

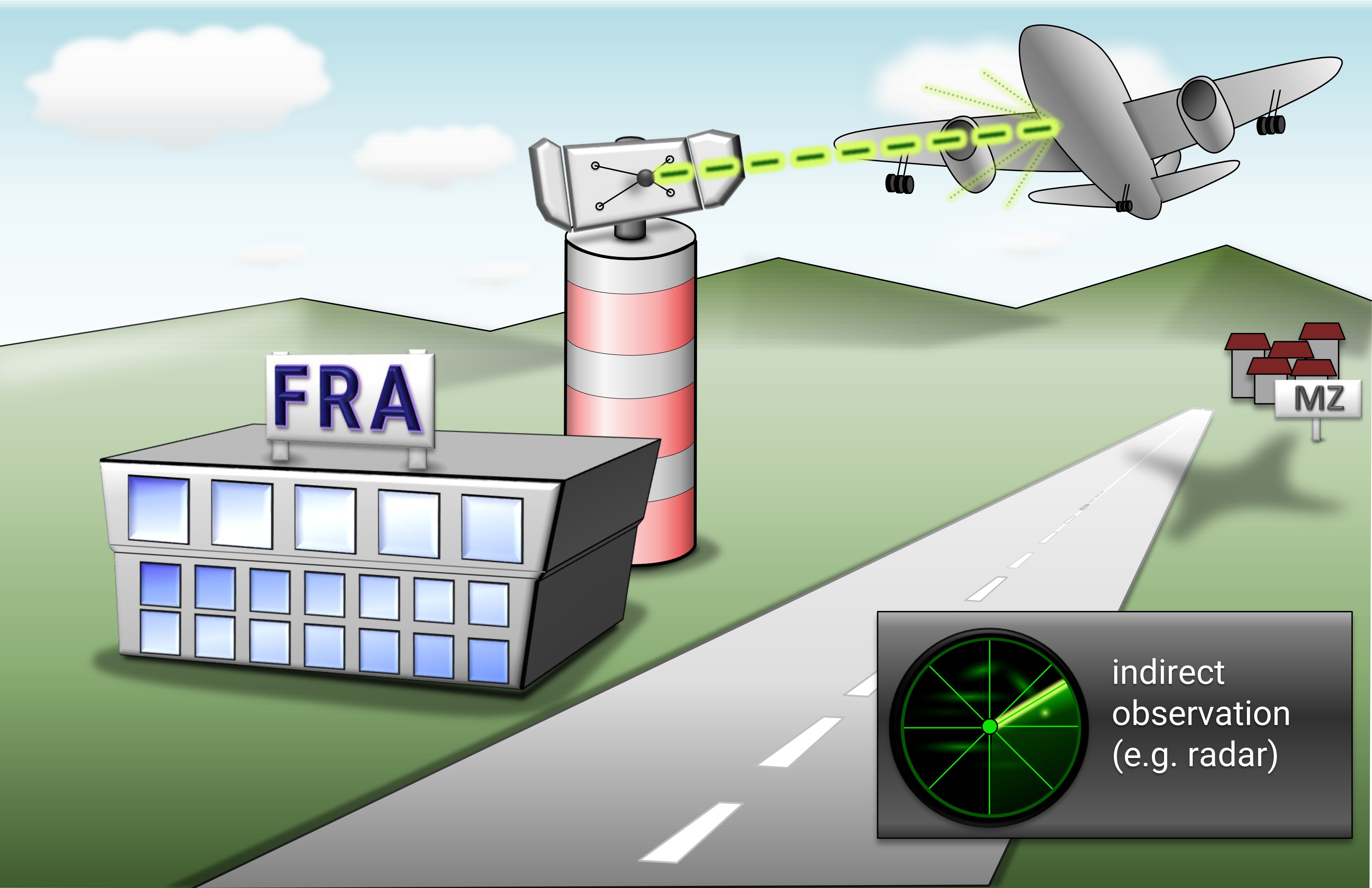
- Additionally
  - System state observed indirectly
  - For example: noisy measurement



# Example: Object Tracking



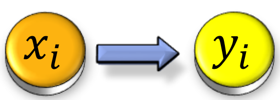
# Example: Object Tracking



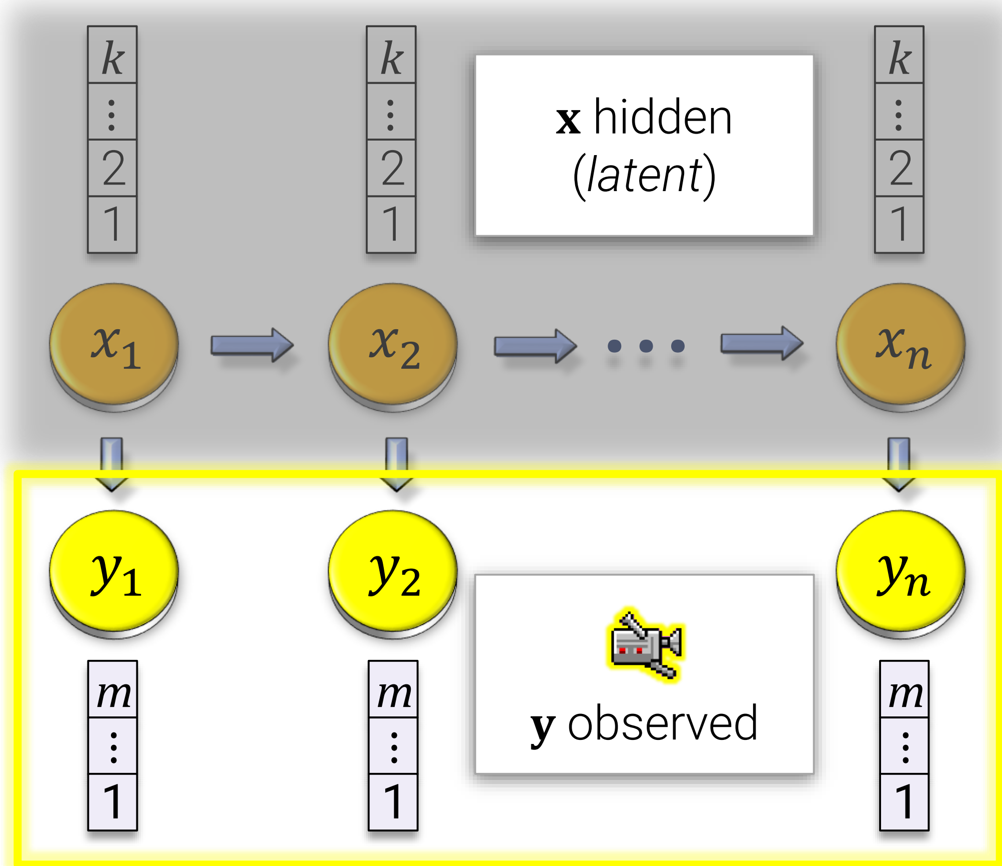
# Hidden Markov Model

## Additional items

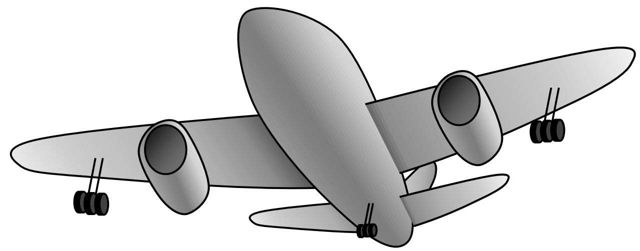
- Random variables
  $\mathbf{y} = (y_1, \dots, y_n)$
  with states
  $y_i \in \{1, \dots, m\}$
- $y_i$ depends on $x_i$ only



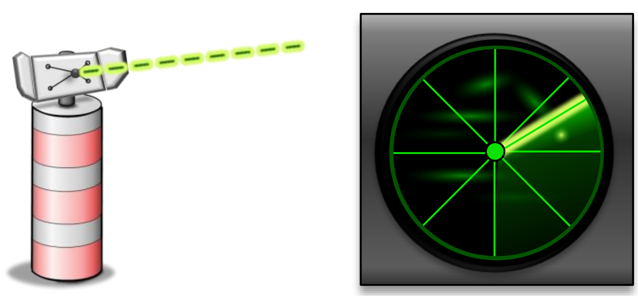

- $\mathbf{y}$ observable
  $\mathbf{x}$ hidden ("latent")



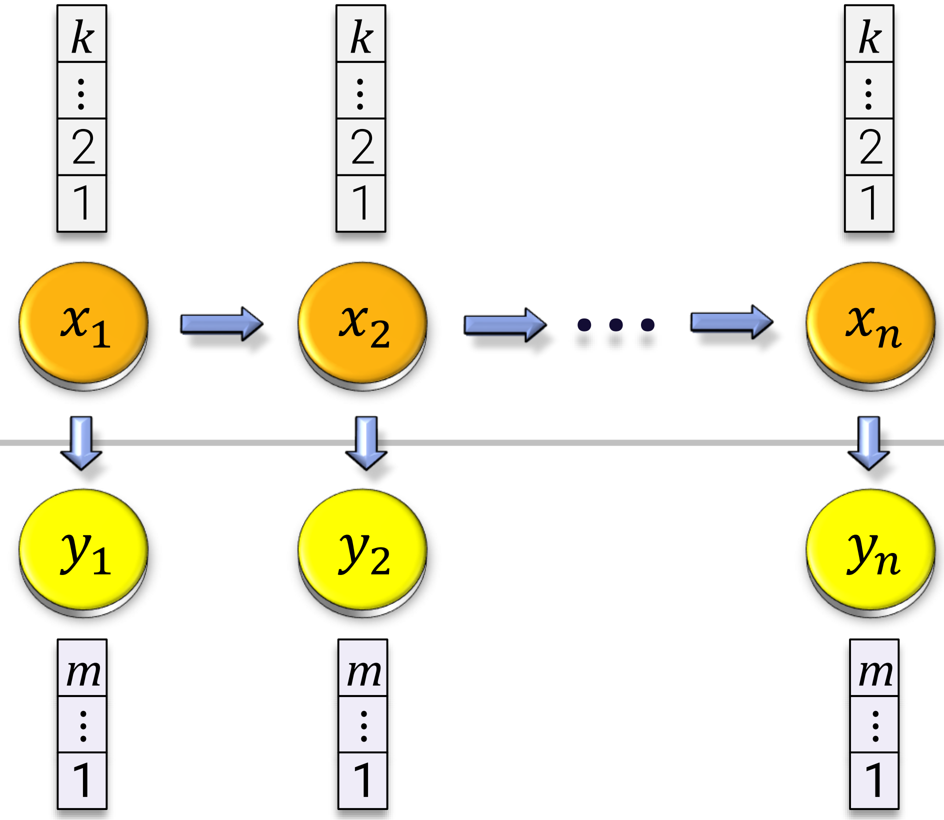
# Example



airplane position



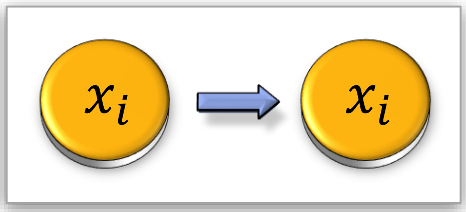
radar localization



# Components

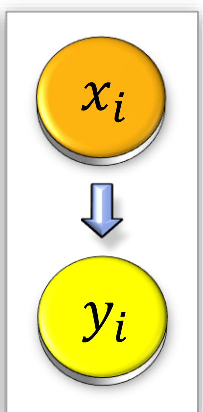
## Two model components

- Transition probabilities (hidden states)




Prior

- Observation model

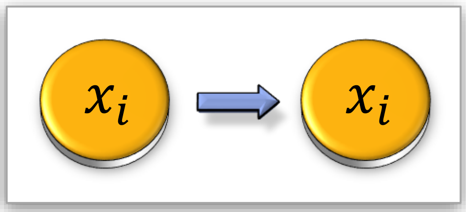



Likelihood (data term)

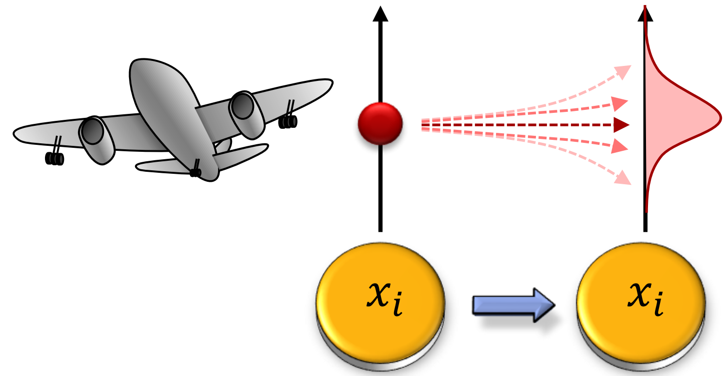
# Components

## Two model components

- Transition probabilities

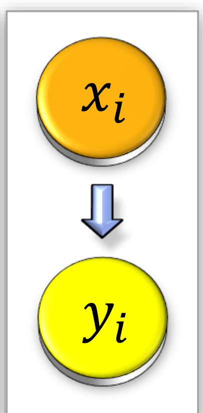



$p(x_{i+1}|x_i)$

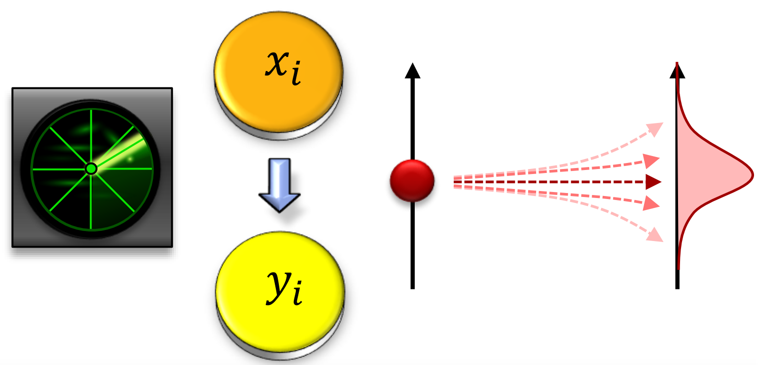



no abrupt movement

- Observation model

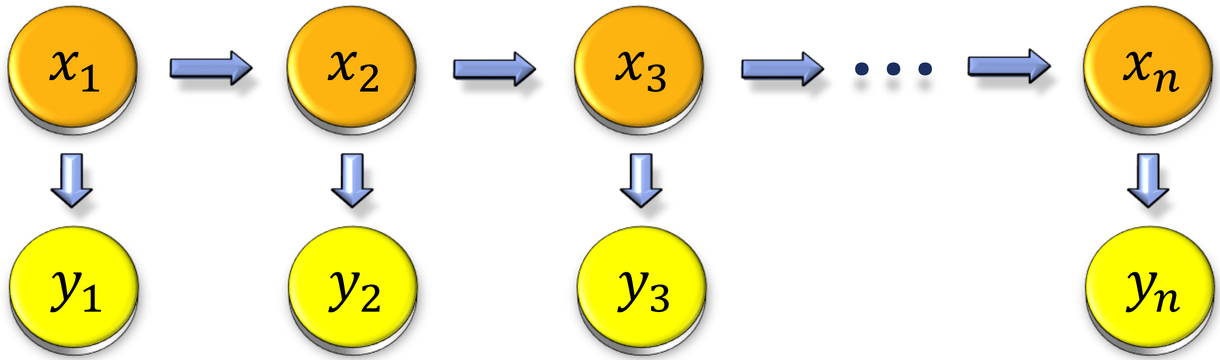



$p(y_{i+1}|x_{i+1})$




position measurement noisy

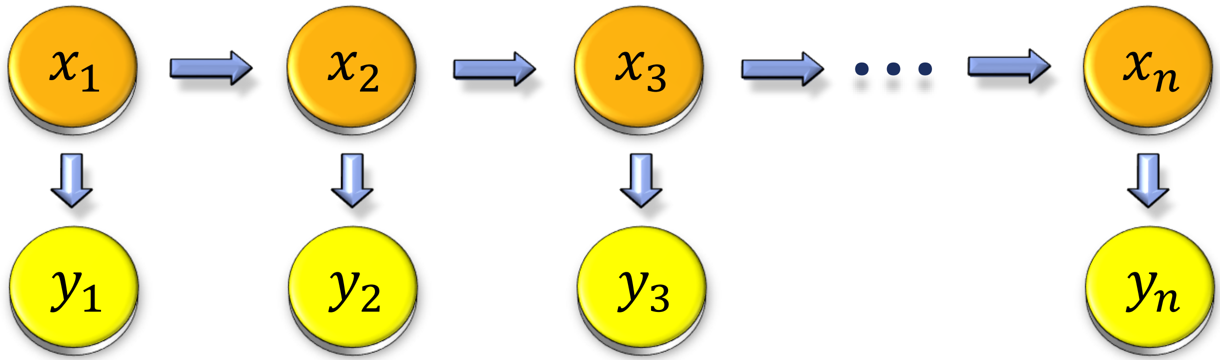
# Markov Property




## Factorization of HMMs

$$p(x_1, x_2, \ldots, x_n) \sim \prod_{i=1}^{n-1} p(x_{i+1} | x_i)$$

# Markov Property



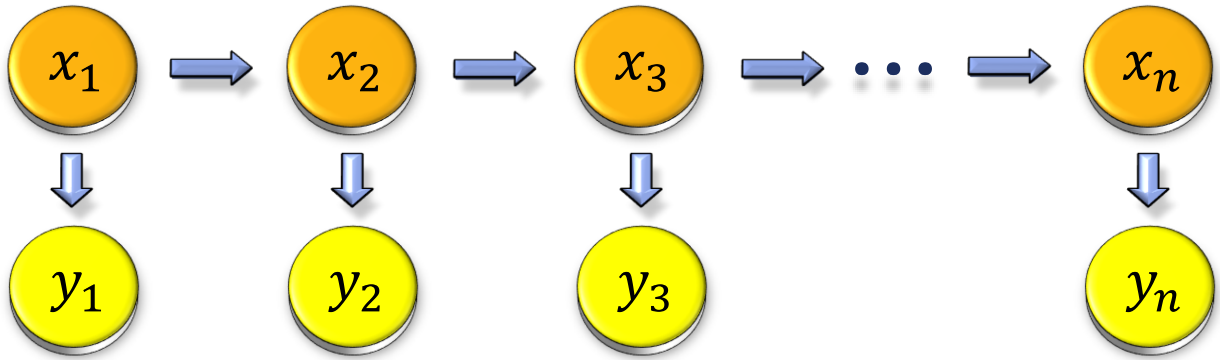

## Factorization of HMMs

$$p(x_1, x_2, \ldots, x_n | y_1, y_2, \ldots, y_n) \sim \prod_{i=1}^{n-1} p(x_{i+1}|x_i) \prod_{i=1}^{n} p(y_i|x_i)$$

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

$$P(\mathbf{x}|\mathbf{y}) \sim P(\mathbf{y}|\mathbf{x})P(\mathbf{x})$$

# Forward Decoding (Inference)

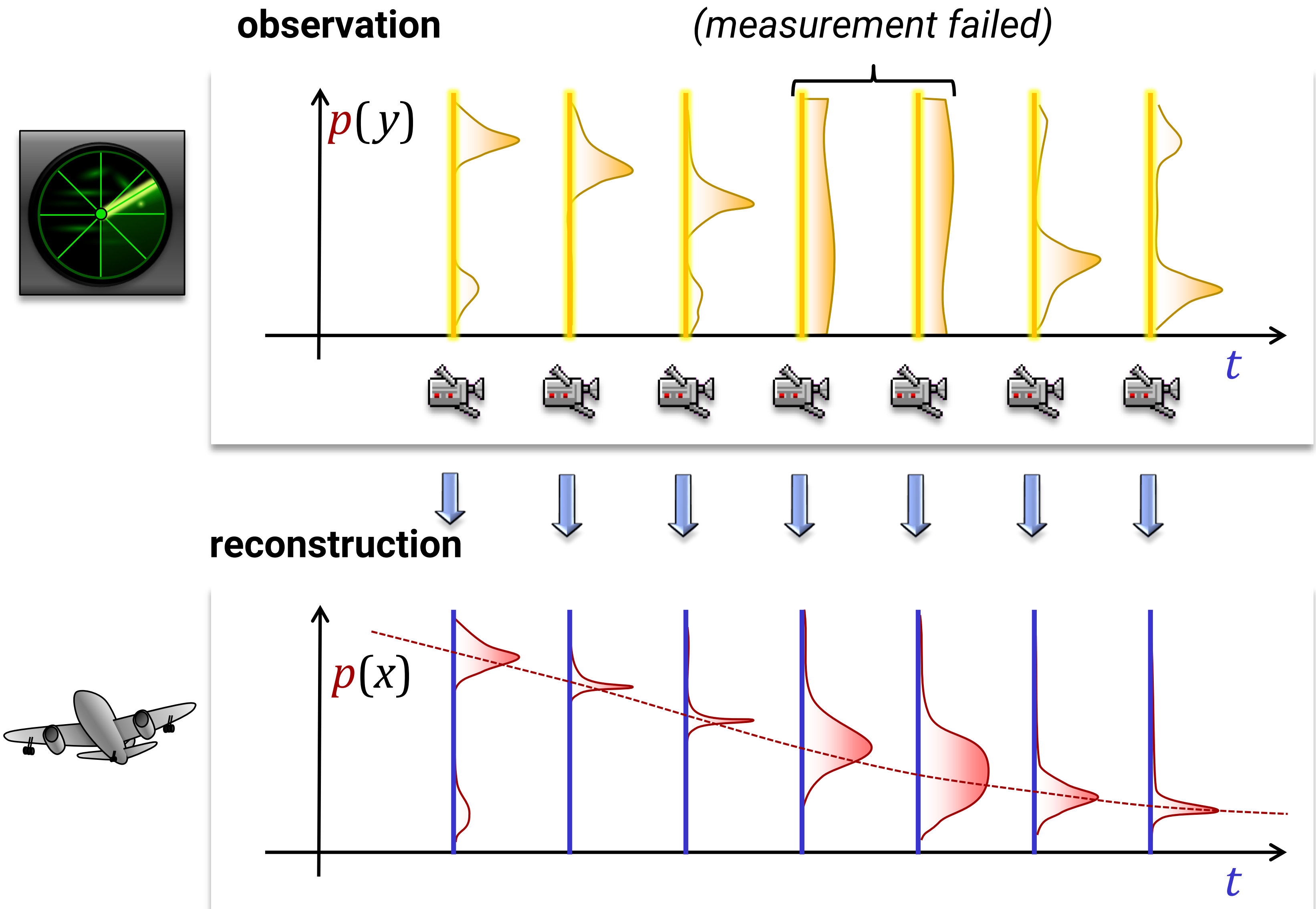



## Object Tracking

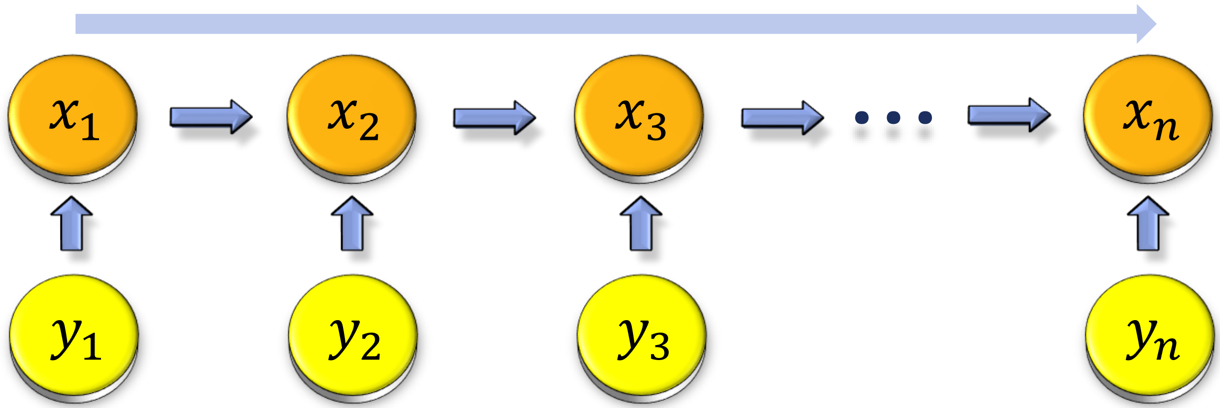
- $p(x_1 = r) = \sum_{j=1}^{m} p(y_1 = s | x_1 = r)$
- …
- $p(x_{i+1} = r) = \left[\sum_{j=1}^{k} p(x_i = j)\, p(x_{i+1} = r | x_i = j)\right]$

$$\cdot \left[\sum_{j=1}^{m} p(y_1 = s | x_1 = r)\right]$$

# Example: Object Tracking



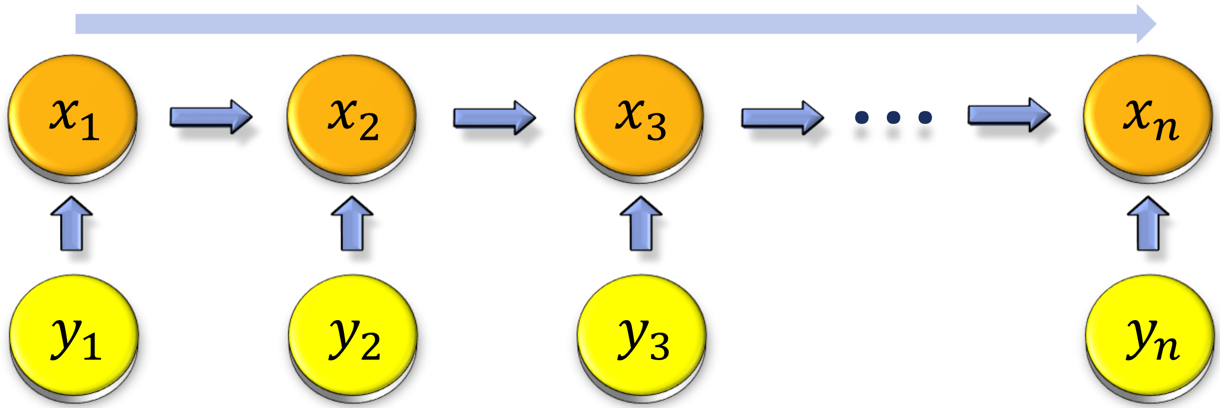
# Forward Decoding (Inference)



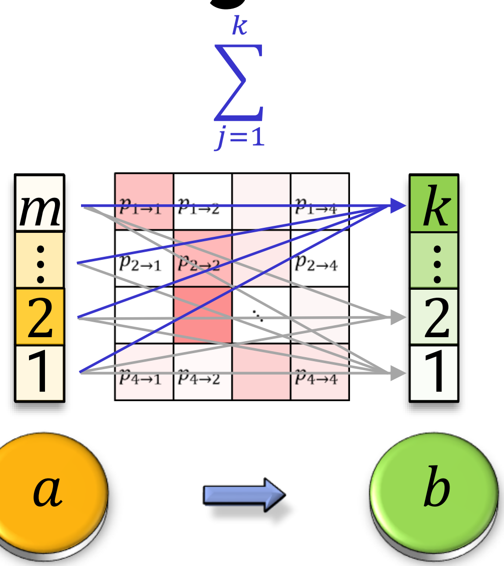

## Message Passing

- $y_1 \rightarrow x_1$
- $x_1 \rightarrow x_2, y_2 \rightarrow x_2$
- …
- $x_{i-1} \rightarrow x_i, y_i \rightarrow x_i$

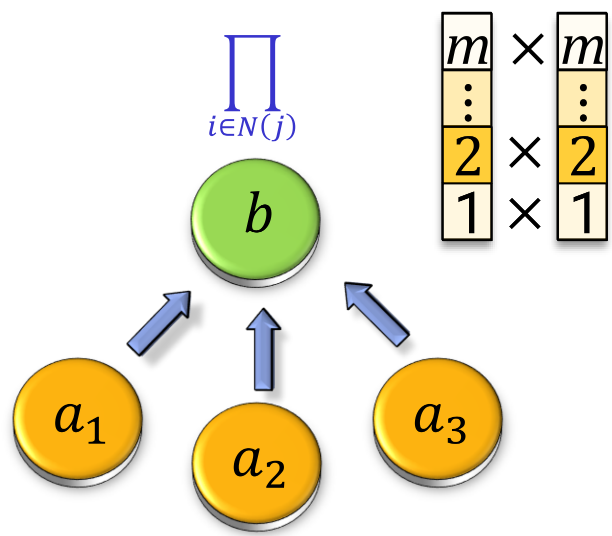
# Interpretation




## Message Passing



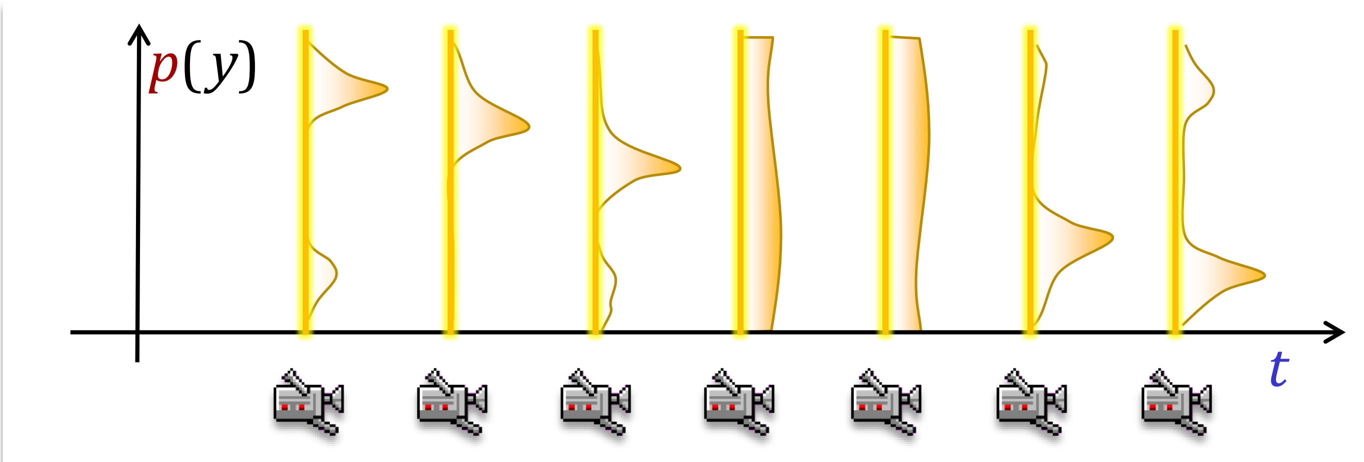

message:
transition matrix



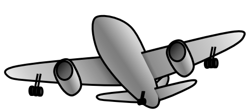

combining messages:
entry-wise product

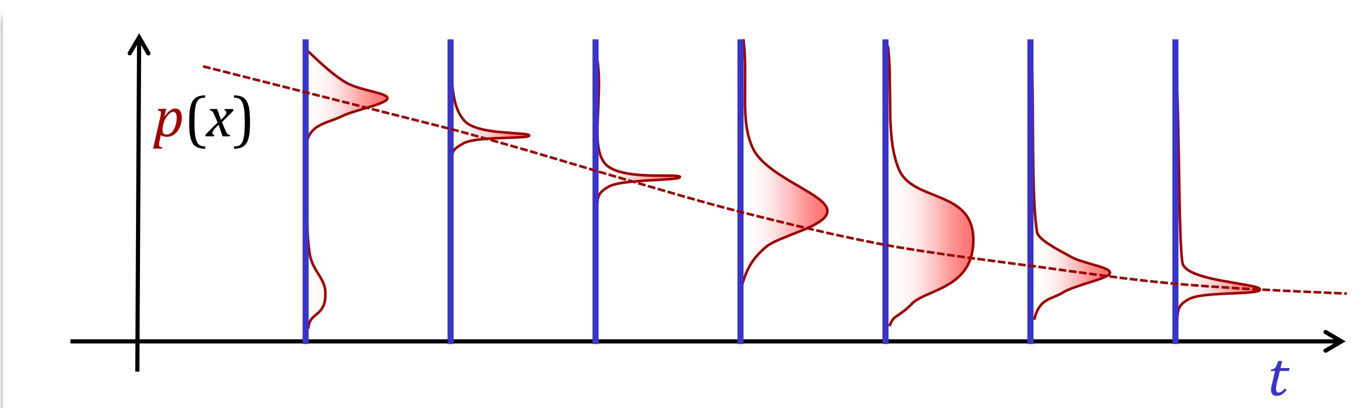
# Offline Version?

**observation** *(after all measurements are known)*

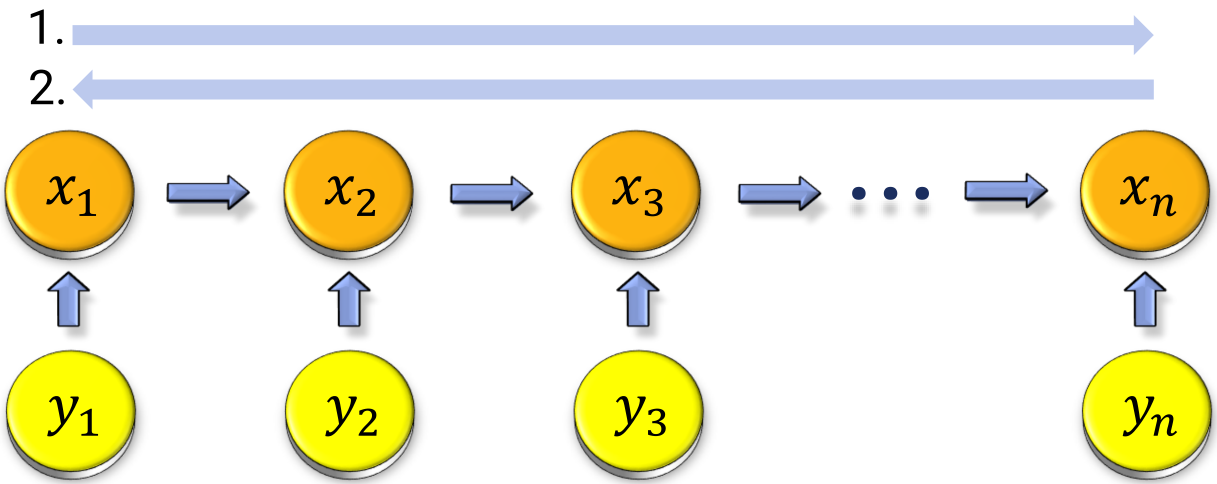



**reconstruction**





# Forward-Backward Algorithm

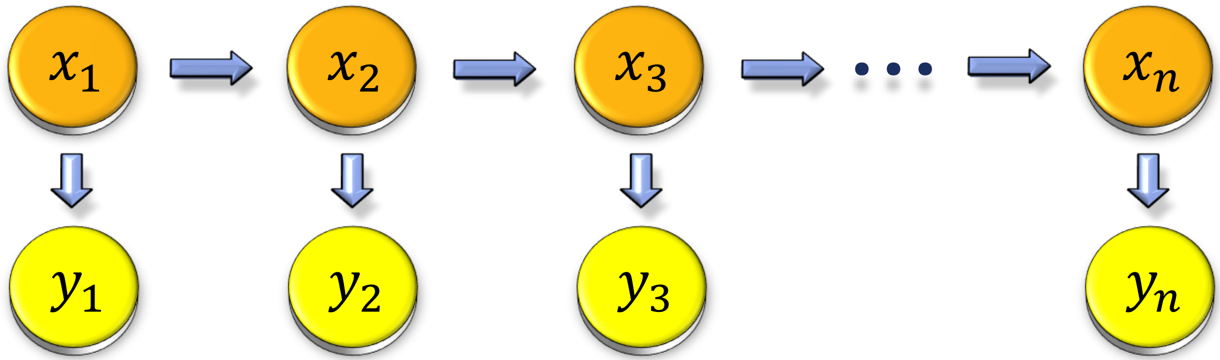



**Compute posterior marginals**

- Compute updates forwards
- Afterwards, compute updates backwards
  - Using computed values for latent states
- Exact marginals
  - „Bayesian belief propagation“

# Conditional Random Fields

# Hidden Markov Model




## Factorization of HMMs

$$p(x_1, x_2, \ldots, x_n | y_1, y_2, \ldots, \boldsymbol{y}_n) \sim \prod_{i=1}^{n-1} p(x_{i+1} | x_i) \prod_{i=1}^{n} p(y_i | x_i)$$

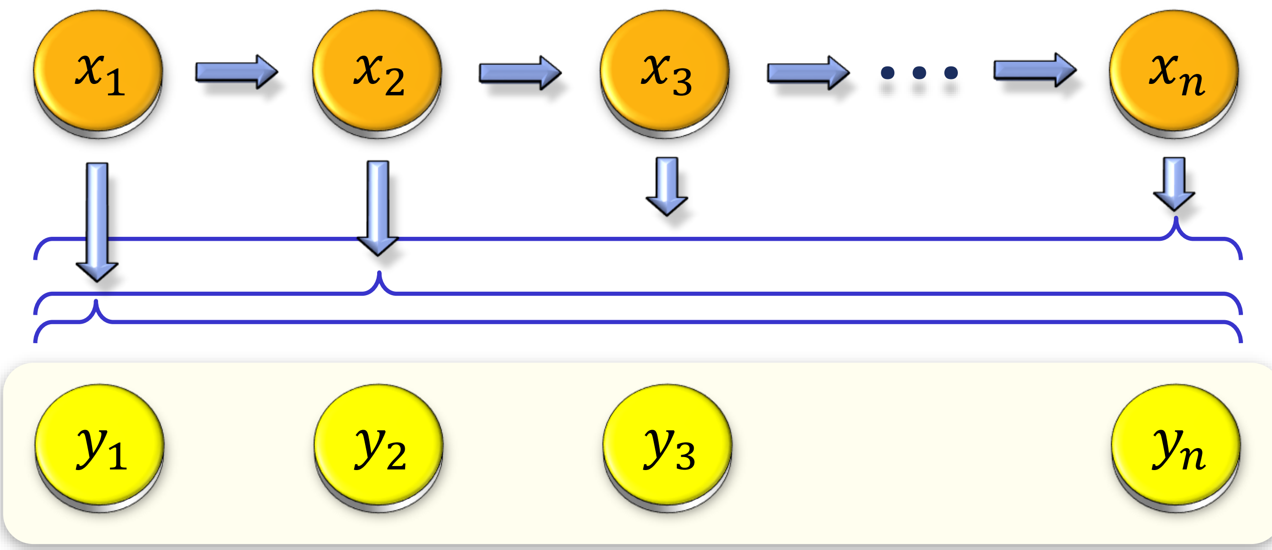
$$P(\mathbf{x}|\mathbf{y}) \sim P(\mathbf{y}|\mathbf{x}) P(\mathbf{x})$$

# Problems

**Problems**

- Every data item separate
- Information might not be enough
- Positive example
  - Radar localization – update on position
- Negative example
  - Speech recognition:
    Character in string – do not recognize words
  - [MRFs: Image recognition
    Image pixel – need to look at larger neighborhood]

# Hidden Markov Model

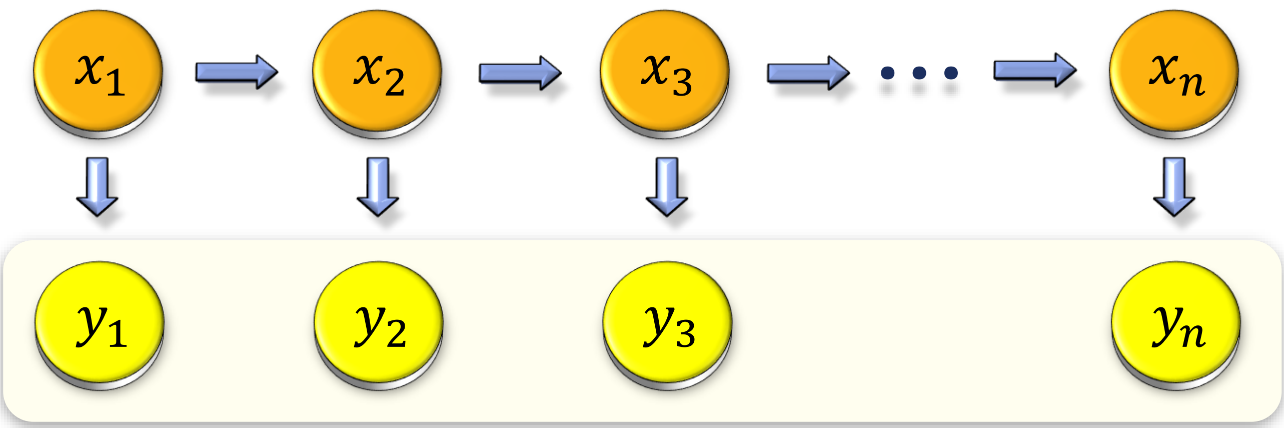



## Dependency on complete data set

$$p(x_{i+1} | x_1, \ldots, x_i, \mathbf{y}) = p(x_{i+1} | x_i, \mathbf{y})$$

- $\mathbf{x}$ Markov chain when conditioned on observation $\mathbf{y}$
- In practice: Features from $\mathbf{y}$ without size limits (e.g. HoG)

# Hidden Markov Model




## Factorization

$$p(x_1, x_2, \dots, x_n | \mathbf{y}) \sim \frac{1}{Z} \prod_{i=1}^{n-1} p_2(x_i, x_{i+1}, \mathbf{y}) \prod_{i=1}^{n} p_1(x_i, \mathbf{y})$$

- Again: Factorization in potential functions $p_1, p_2$
- Dependence on $\mathbf{y}$ unrestricted

# Learning Conditional Random Fields

## Learning HMMs / CRFs

$$\arg\max_{\theta \in \mathbb{R}^d} p(\mathbf{x}|\mathbf{y})$$

$$= \arg\max_{\theta \in \mathbb{R}^d} \frac{1}{Z} \prod_{i=1}^{n-1} p_2^{(\theta)}(x_i, x_{i+1}, \mathbf{y}) \prod_{i=1}^{n} p_1^{(\theta)}(x_i, \mathbf{y})$$

$$= \arg\max_{\theta \in \mathbb{R}^d} \frac{\prod_{i=1}^{n-1} p_2^{(\theta)}(x_i, x_{i+1}, \mathbf{y}) \prod_{i=1}^{n} p_1^{(\theta)}(x_i, \mathbf{y})}{\int_{\mathbf{x} \in \Omega(\mathbf{x})} \prod_{i=1}^{n-1} p_2^{(\theta)}(x_i, x_{i+1}, \mathbf{y}) \prod_{i=1}^{n} p_1^{(\theta)}(x_i, \mathbf{y})\, dx}$$

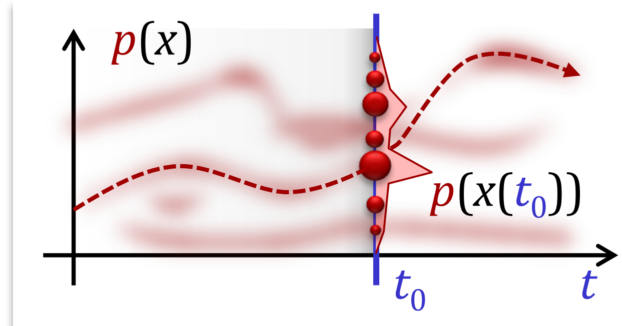
- Likelihood might depend on parameters
- Denominator: Often infeasible
  - Tractable for special cases (Gaussians, simple models)
  - Approximations possible (often - wrongly - ignored)

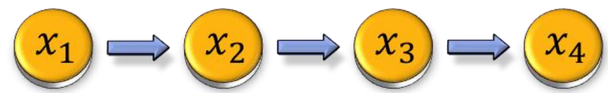
# Summary so far…

# Hidden Markov Models

## Markov-Chains

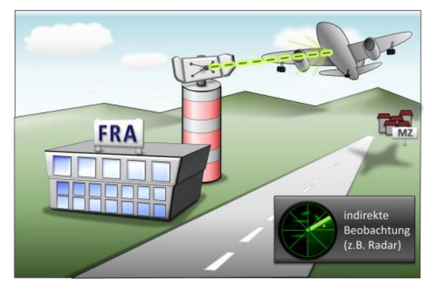
- Time-dependent processes
- Probabilistic model
- Present determines future
  - Conditional independence

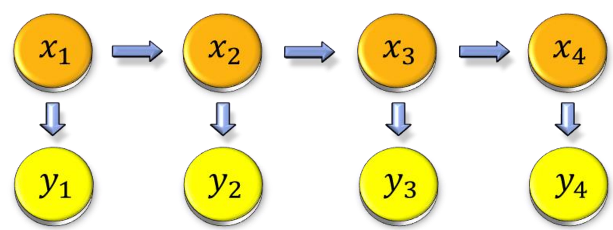



## Hidden-Markov-Models

- State cannot be directly observed
- Typical example: Tracking
- Factorization
- Forward inference



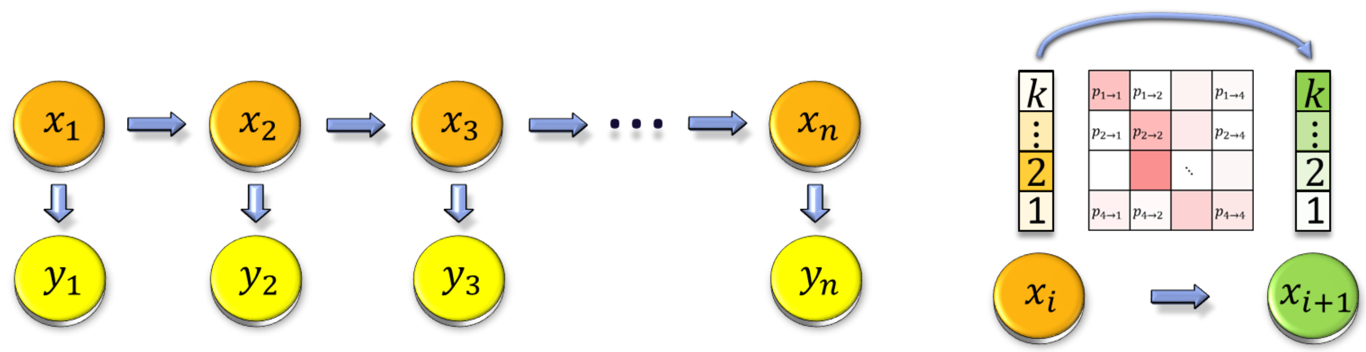



# Modelling 2

## STATISTICAL DATA MODELLING

Institut für Informatik

JG|U JOHANNES GUTENBERG UNIVERSITÄT MAINZ




## Chapter 8

# Markovian Models

Michael Wand · Institut für Informatik, JGU Mainz · michael.wand@uni-mainz.de

Video #08

# Markovian Models

- Markov Chains
- Hidden Markov Models
- **Markov Random Fields**

# Markov Random Fields and Graphical Models

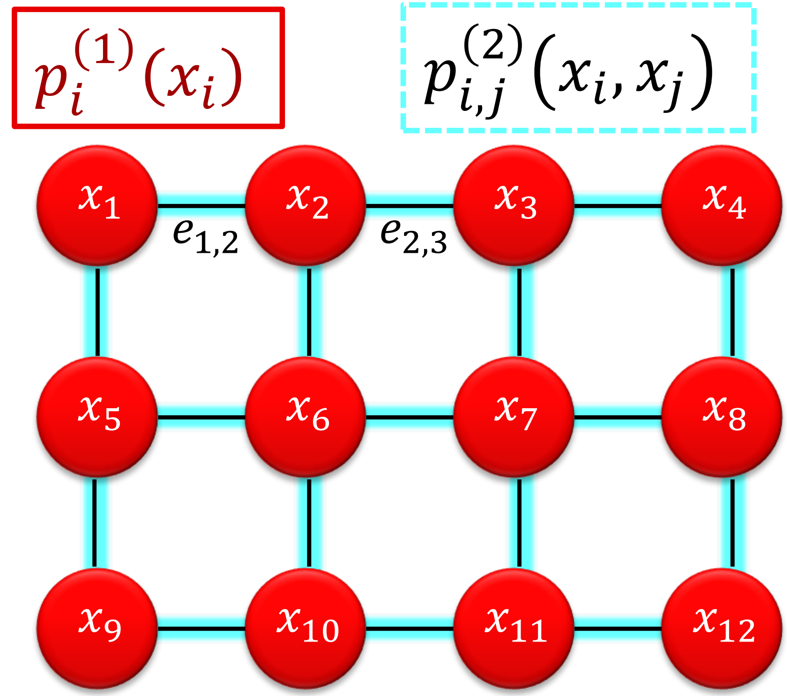
# Graphical Models

## Factorize Models

- Pairwise models:

$$p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} p_i^{(1)}(x_i) \prod_{i,j \in E} p_{i,j}^{(2)}(x_i, x_j)$$

- Model complexity:
  - $O(nk^2)$ parameters
- Higher order models:
  - Triplets, quadruples as factors
  - Local neighborhoods



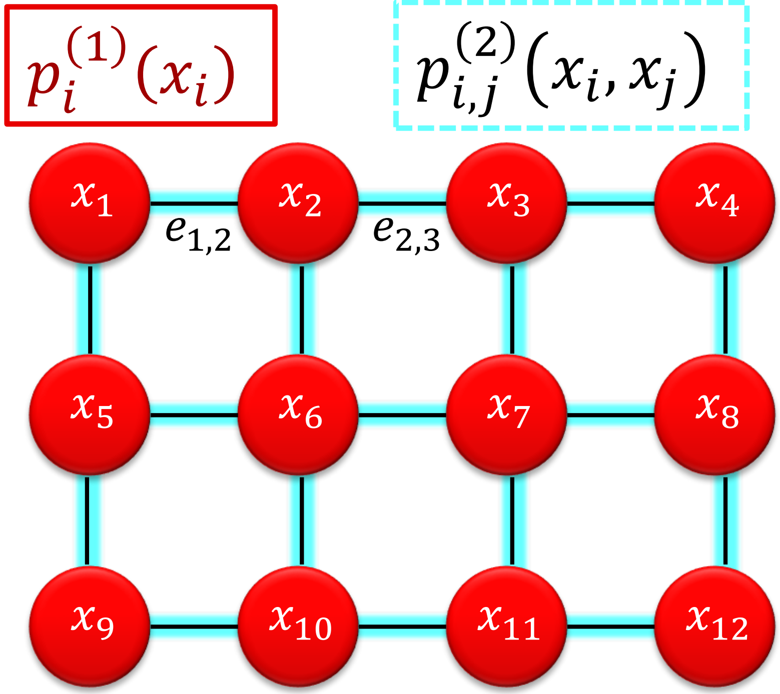
# Graphical Models

## Markov Random fields

- Factorize density in local "cliques"

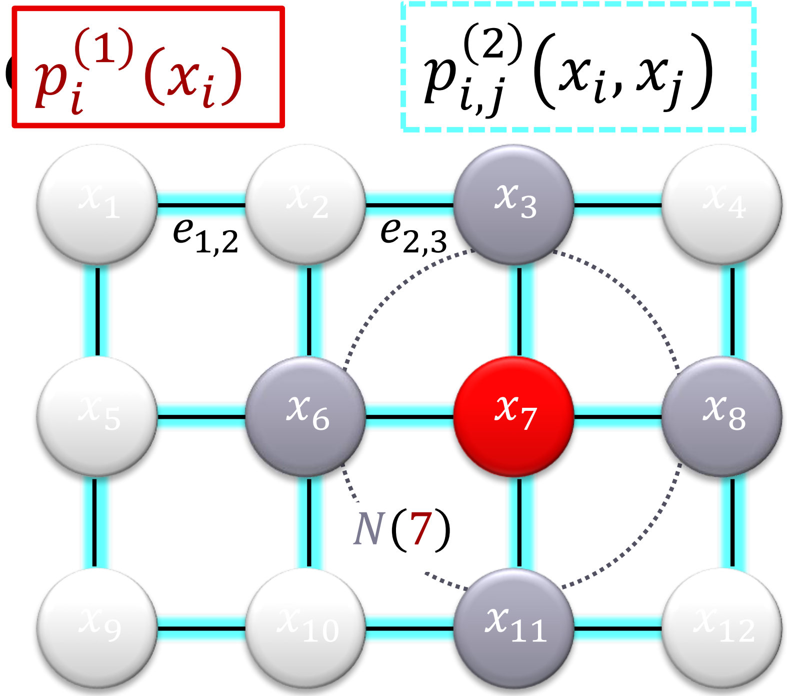
## Graphical model

- Connect variables that are directly dependent
- Formal model: Conditional independence



# Graphical Models

## Conditional Independence




- A node is conditionally independent of all others given the values of its direct neighbors
- Iff by set these values to constants, it is independent of all others

## Formally

"Markov blanket"

- $p(x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = p\left(x_i \middle| \left\{x_j \middle| j \in N(i)\right\}\right)$

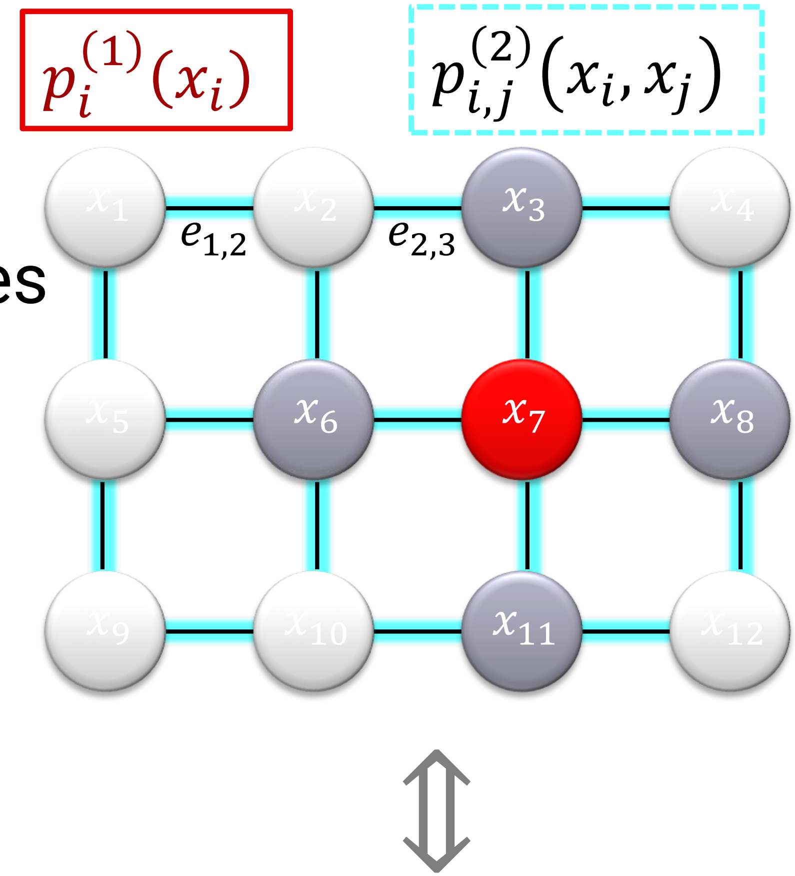
# Graphical Models

## Theorem (Hammersley–Clifford):

- Assuming positive densities $p(x_i) > 0$

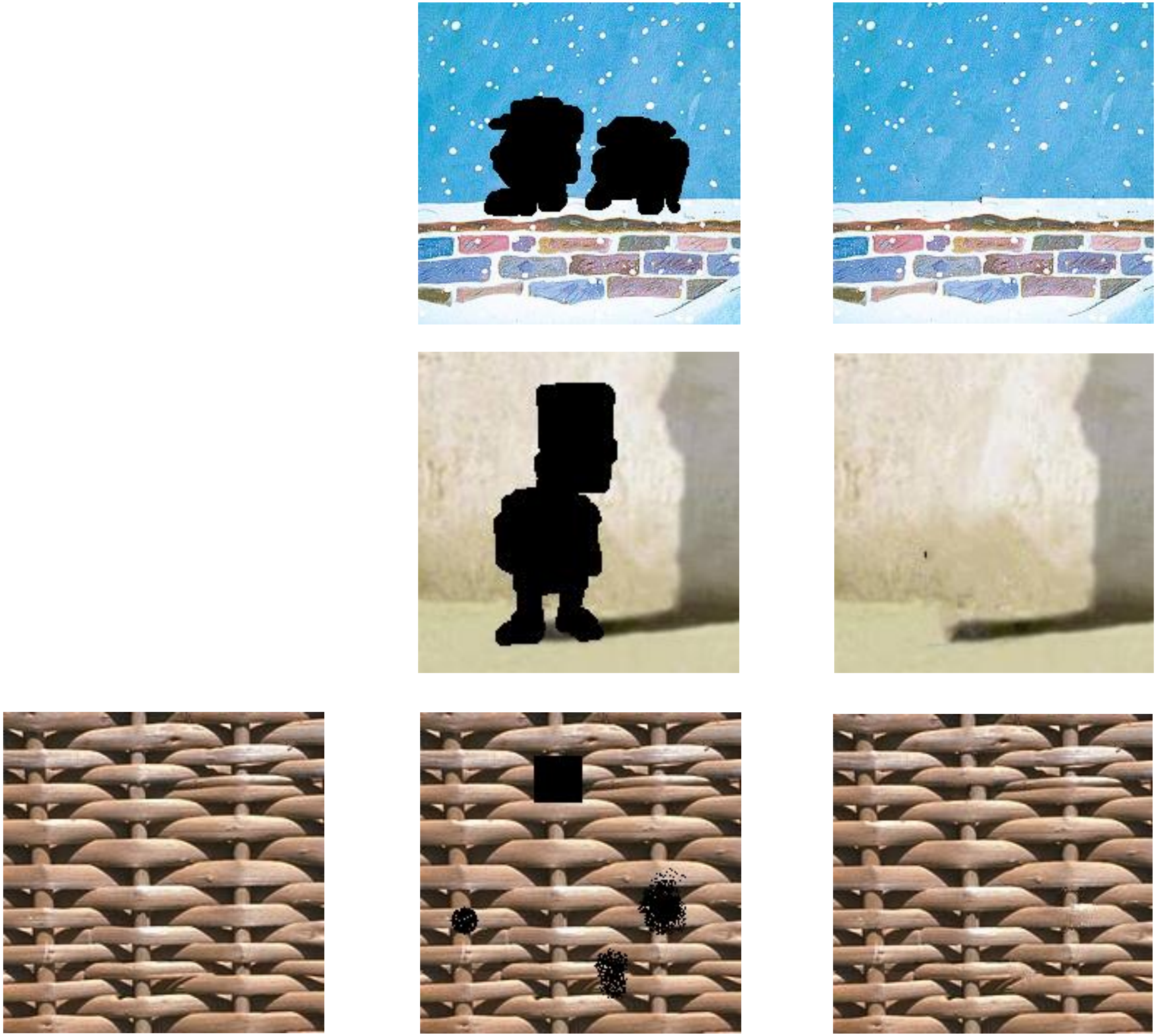
## The therorem

- Given conditional independence as graph, density factors over cliques in the graph.
- And vice versa.




$$p(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{n} \underline{p_i^{(1)}(x_i)} \prod_{i,j \in E} \underline{p_{i,j}^{(2)}(x_i, x_j)}$$

# Example: Texture Synthesis

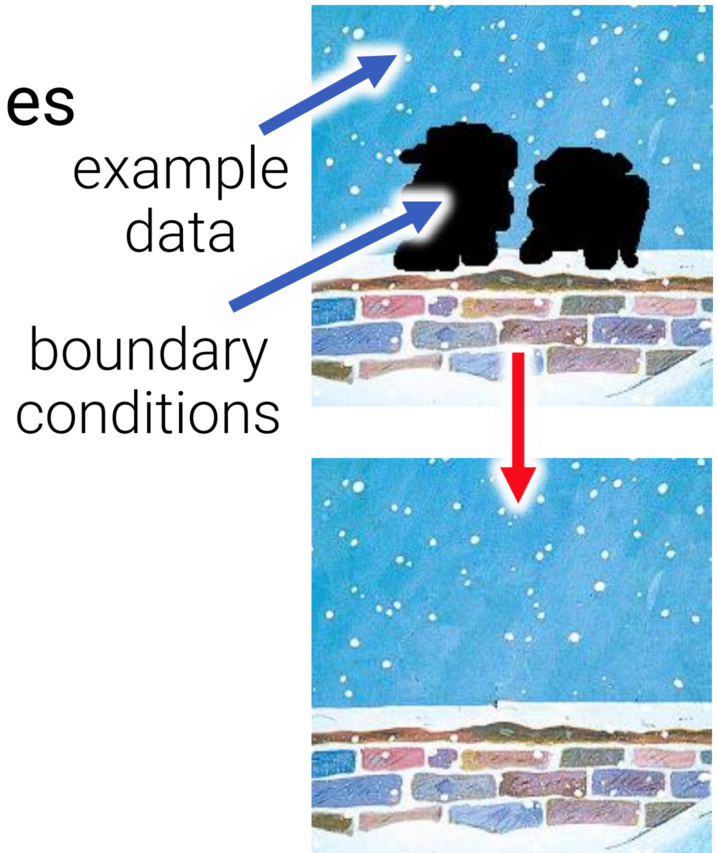


original region selected completion

# Texture Synthesis

## Idea

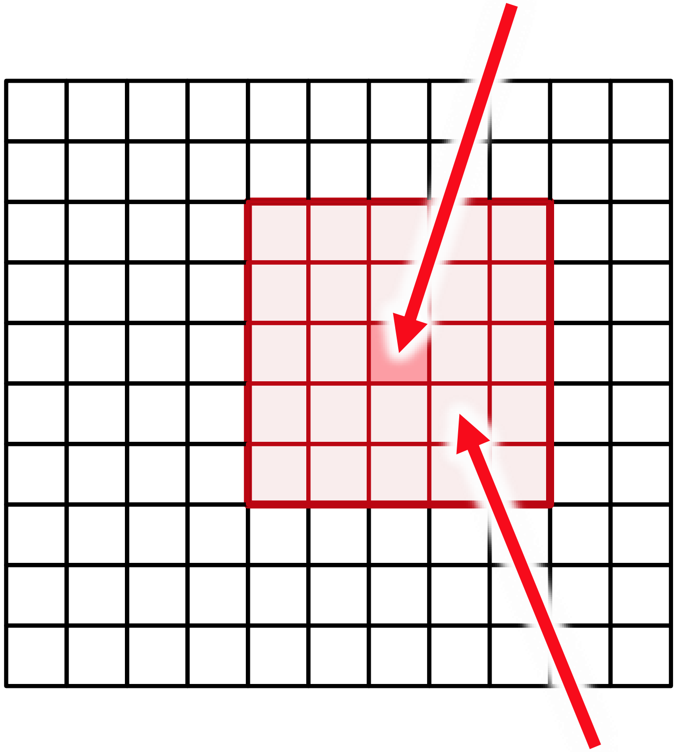
- One or more images as examples
- Learn image statistics
- Use knowledge:
  - Specify boundary conditions
  - Fill in texture



# The Basic Idea

**Markov Random Field Model**

- Image statistics
- How pixels are colored depends on local neighborhood only (Markov Random Field)
- Predict color from neighborhood



# A Little Bit of Theory...

## Image statistics:

- An image of $n \times m$ pixels
- Random variable: $\mathbf{x} = [x_{11},\ldots,x_{nm}] \in [0, 1, \ldots, 255]^{n\times m}$
- Probability distribution:
  $p(\mathbf{x}) = p(x_{11}, \ldots, x_{nm})$

256 choices

⋮

256 choices

$256^{n\times m}$ probability values

## Impossible to learn full images from examples!

# Simplification

**Problem:**

- Statistical dependencies
- Simple modell can express dependencies on all kinds of combinations

**Markov Random Field:**

- Each pixel is *conditionally independent* of the rest of the image given a small neighborhood
- In English: likelihood only depends on neighborhood, not rest of the image

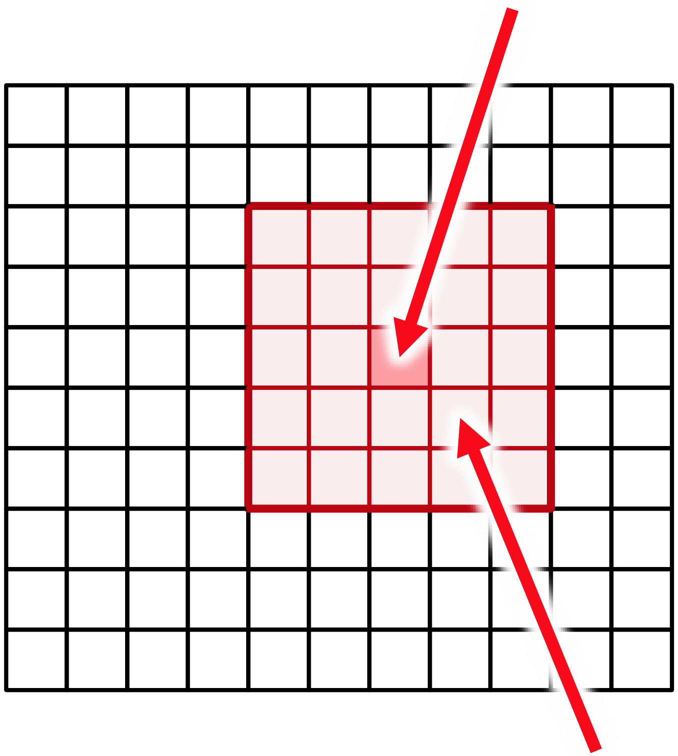
# Markov Random Field

**Example:**

- Red pixel depends on light red region
- Not on black region
- If region is known, probability is fixed and independent of the rest

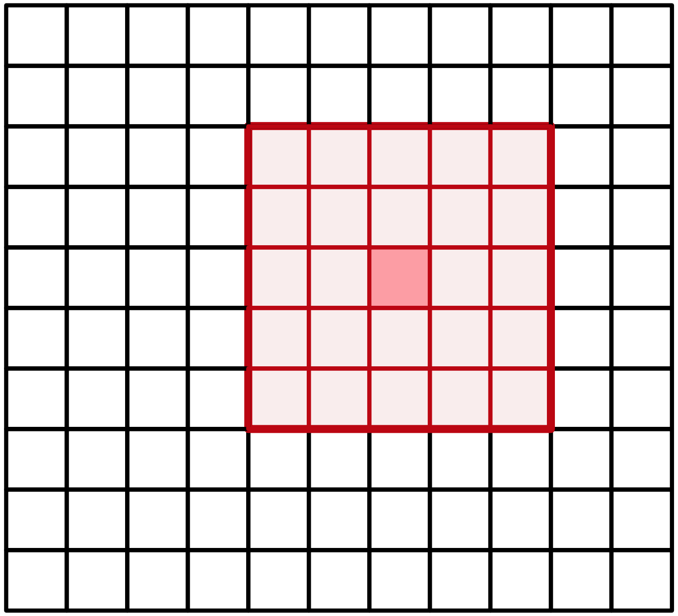
**However:**

- Regions overlap
- Indirect global dependency



# Texture Synthesis

## **Use for Texture Synthesis**



# Inference

## Inference Problem

- Computing $p(\mathbf{x})$ is trivial for known $\mathbf{x}$.
- Finding the $\mathbf{x}$ that maximizes $p(\mathbf{x})$ is very complicated.
- In general: NP-hard
- No efficient solution known (not even for images)

## In practice

- Different approximation strategies ("heuristics", strict approximation is also NP-hard)

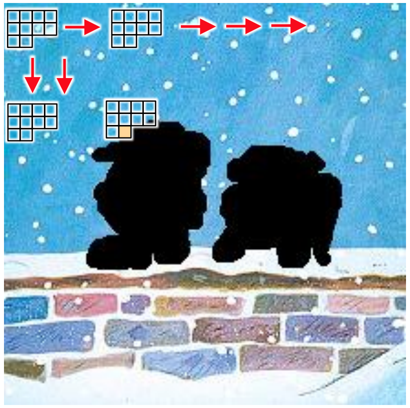
# Simple Practical Algorithm

**Here is the short story:**

- Unknown pixels:
  consider known neighborhood
- Match to all of the known data
- Copy the pixel with the best matching neighborhood
- Region growing, outside in

**Approximation only**

- Can run into bad local minima



# Example:
# Image Reconstruction


ground truth
reconstruction (Gaussian prior)
input (noisy)
reconstruction (Laplace prior)

# Gaussian MRF

## Minimize

$$E(D|X) + \mathrm{E}(X)$$

$$= \sum_{i=1}^{w} \sum_{j=1}^{h} \frac{(x_i - d_i)^2}{2\sigma_D^2} + \sum_{i=1}^{w-1} \sum_{j=1}^{h-1} \frac{\left(x_{i+1,j} - x_{i,j}\right)^2 + \left(x_{i,j+1} - x_{i,j}\right)^2}{2\sigma_X^2}$$

## Equivalent minimization objective

$$\sum_{i=1}^{w} \sum_{j=1}^{h} (x_i - d_i)^2 + \frac{\sigma_X^2}{\sigma_D^2} \sum_{i=1}^{w-1} \sum_{j=1}^{h-1} \left(x_{i+1,j} - x_{i,j}\right)^2 + \left(x_{i,j+1} - x_{i,j}\right)^2$$

## Continuous

$$\int_\Omega \left(f(\mathbf{x}) - d(\mathbf{x})\right)^2 d\mathbf{x} + \frac{\sigma_X^2}{\sigma_D^2} \int_\Omega \|\nabla f(\mathbf{x})\|^2 d\mathbf{x}$$

# Example: Weak Formulations of Differential Equations

# Differential Equations

## Example equation

$$\frac{d}{dt} f(t) = F(f(t), t)$$

## Discretization

$$\frac{y_i - y_{i-1}}{h} = F(y_i, t_i)$$

## Weak formulation (variational approach)

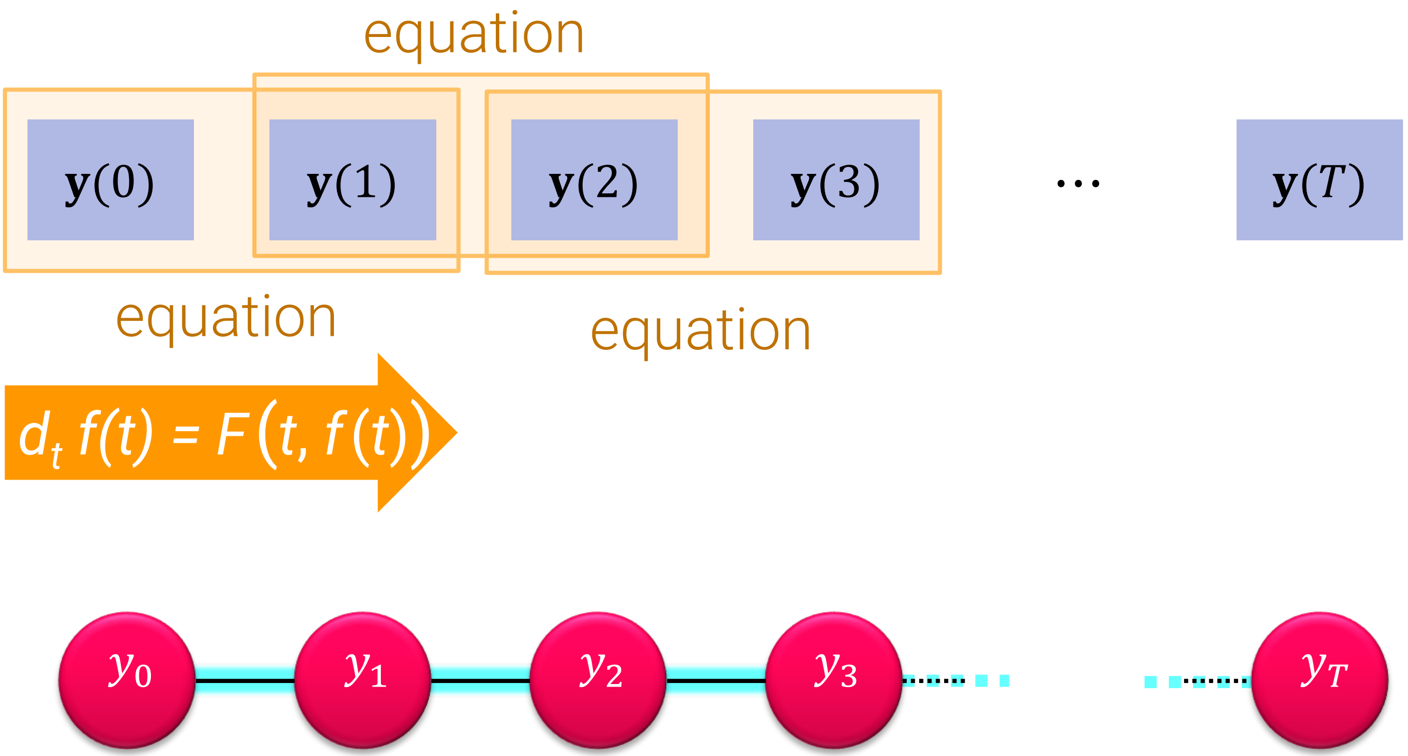
$$\left(\frac{y_i - y_{i-1}}{h} - F(y_i, t_i)\right)^2 \rightarrow min$$

$$\underset{\mathbf{y}}{\arg\max} \frac{1}{Z} \prod_{i=1}^{n-1} \exp\left(\frac{y_i - y_{i-1}}{h} - F(y_i, t_i)\right)^2$$

Gaussian MRF

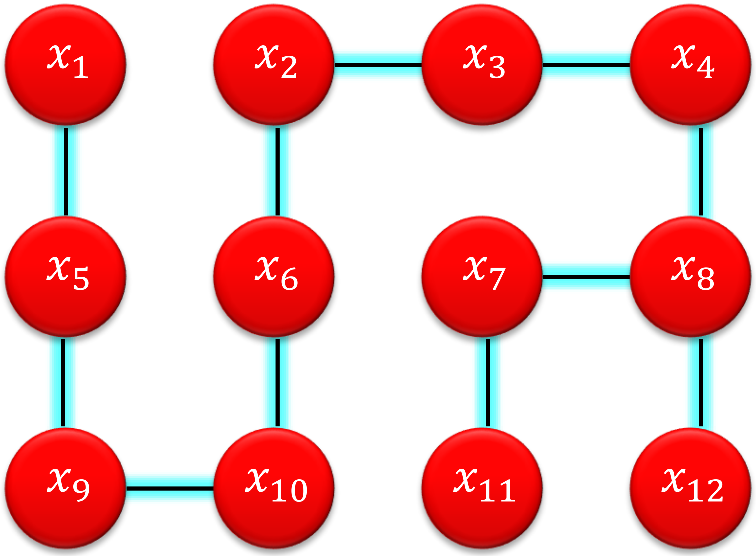
# Ordinary Differential Equations

## Causal Chain



# Inference in MRFs

# Belief Propagation




## Prerequsite

- Tree-structure graph

## Algorithm

- Extension of forward-backward algorithm for Markov chains
- Message Passing

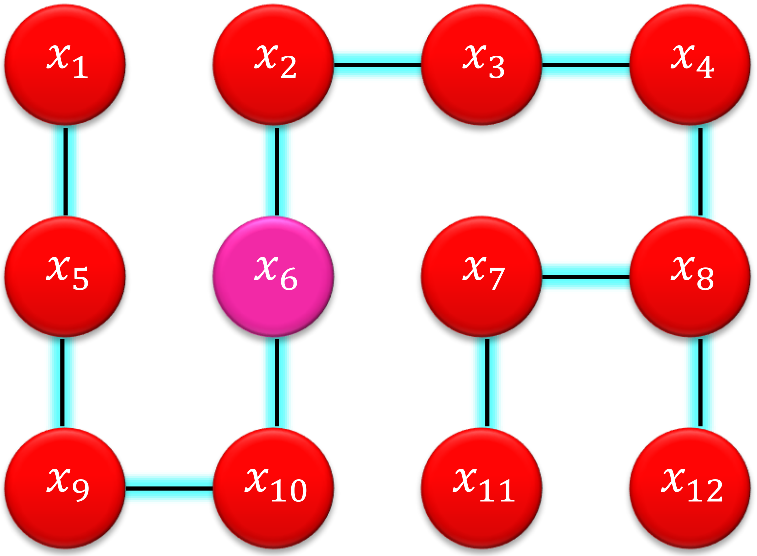
## Model

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} \underline{p_i^{(1)}(x_i)} \prod_{i,j \in E} \underline{p_{i,j}^{(2)}(x_i, x_j)}$$

# Belief Propagation

## Model

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{n} \underline{p_i^{(1)}(x_i)} \prod_{i,j \in E} \underline{p_{i,j}^{(2)}(x_i, x_j)}$$




## Looking for

- Marginal distributions
- $p(x_i) = \sum_{x_1} \sum_{x_2} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_n} p(x_1, x_2, \ldots, x_n)$

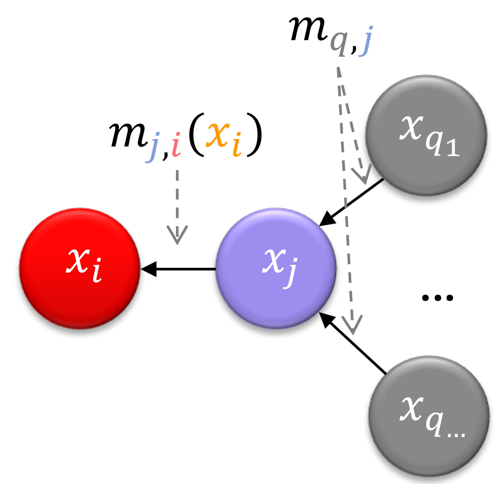
# Belief Propagation

## Model

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{n} p_i^{(1)}(x_i) \prod_{i,j\in E} p_{i,j}^{(2)}(x_i, x_j)$$

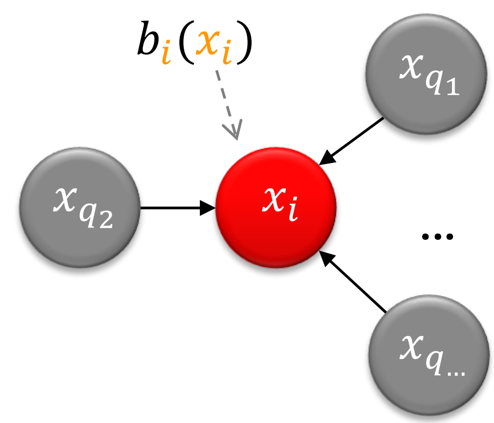
## Messages

$$m_{j,i}(x_i) = \sum_{x_j=1}^{k} p_j^{(1)}(x_j)\, p_{i,j}^{(2)}(x_i, x_j) \prod_{q\in N(j)\setminus x_i} m_{q,j}(x_q)$$




## Beliefs

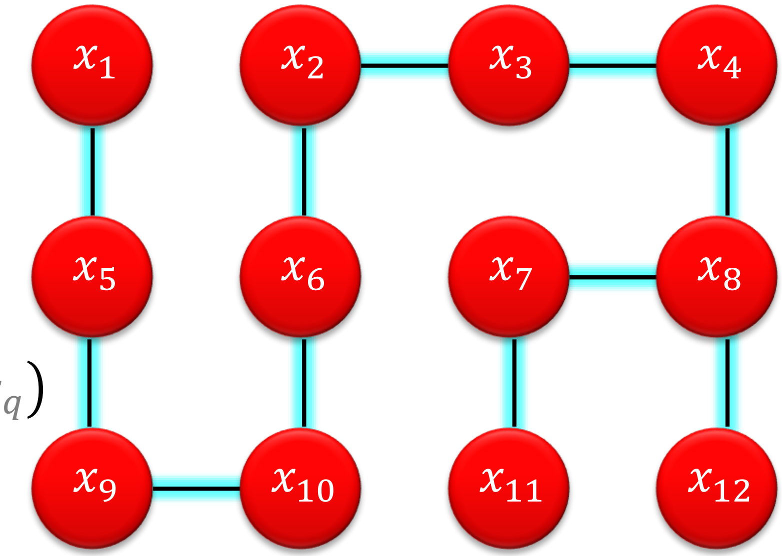
$$b_i(x_i) = \frac{1}{Z_i} p_i^{(1)}(x_i) \prod_{q\in N(i)} m_{q,i}(x_q)$$



# Belief Propagation

## Model

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{n} p_i^{(1)}(x_i) \prod_{i,j\in E} p_{i,j}^{(2)}(x_i, x_j)$$




## Messages

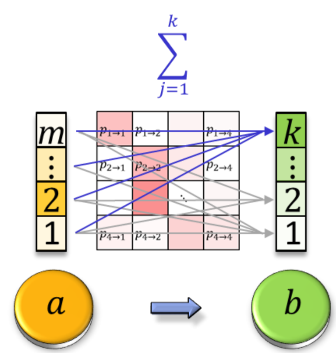
$$m_{j,i}(x_i) = \sum_{x_j=1}^{k} p_i^{(1)}(x_j)\, p_{i,j}^{(2)}(x_i, x_j) \prod_{q\in N(j)\setminus x_i} m_{q,j}(x_q)$$

## Forward & backward pass

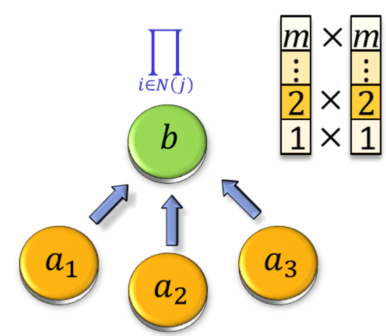
- Any start node
- Depth-first-search & back

## Beliefs

$$b_i(x_i) = \frac{1}{z_i} p_i^{(1)}(x_i) \prod_{q\in N(i)} m_{q,i}(x_q)$$




message:
transition matrix




combining messages:
entry-wise product

# Belief Propagation

**Model** $$p(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{n} p_i^{(1)}(x_i) \prod_{i,j\in E} p_{i,j}^{(2)}(x_i, x_j)$$

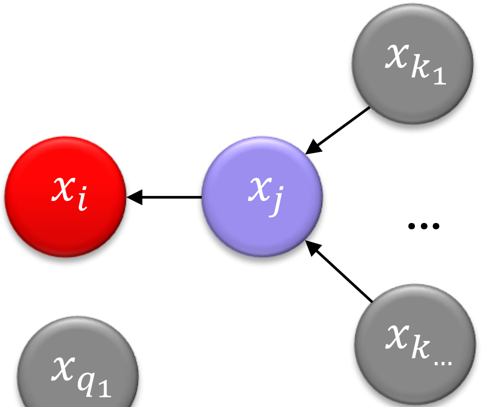
**Messages**

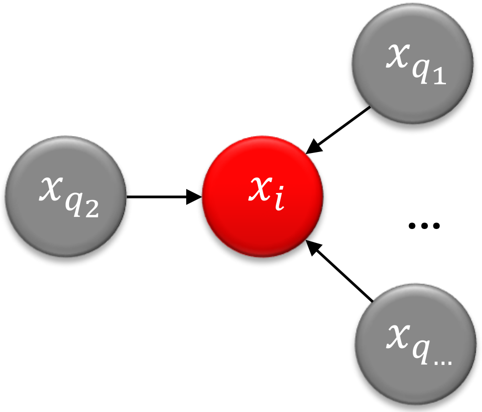
- Maximum a posteriori inference?
- Use "max-marginal":

$$m_{j,i}(x_i) = \max_{x_j=1\ldots k} p_i^{(1)}(x_j)\, p_{i,j}^{(2)}(x_i, x_j) \prod_{q\in N(j)\setminus x_i} m_{q,j}(x_q)$$

$$b_i(x_i) = \frac{1}{z_i} p_i^{(1)}(x_i) \prod_{q\in N(i)} m_{q,i}(x_q)$$

most likely state = $\arg\max_{x_i} b_i(x_i)$



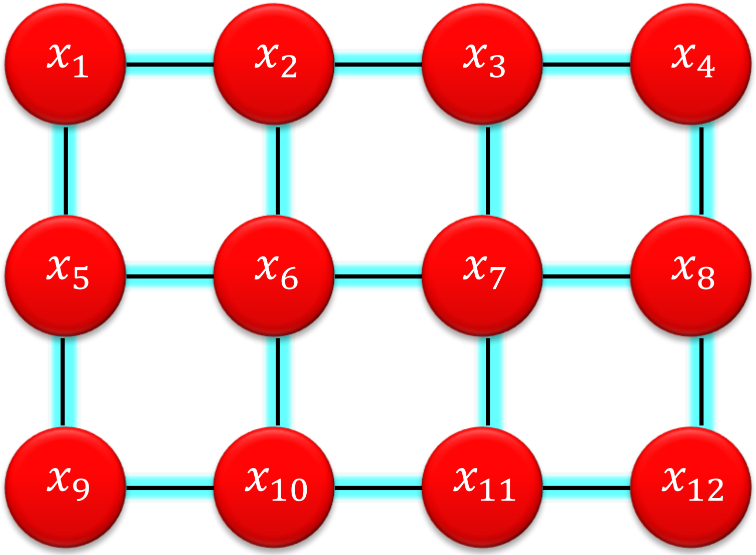



# Approximate Inference

# Loopy BP

## Loopy BP

- Loops? Which loops?
- Just run BP on loopy graph
- Arbitrary order




## Result

- No guarantees (results can be wrong)
- Most often, results will be wrong
- Frequently still a reasonable approximation
  - Problem dependent, but worth a try. Popular 20 years ago.

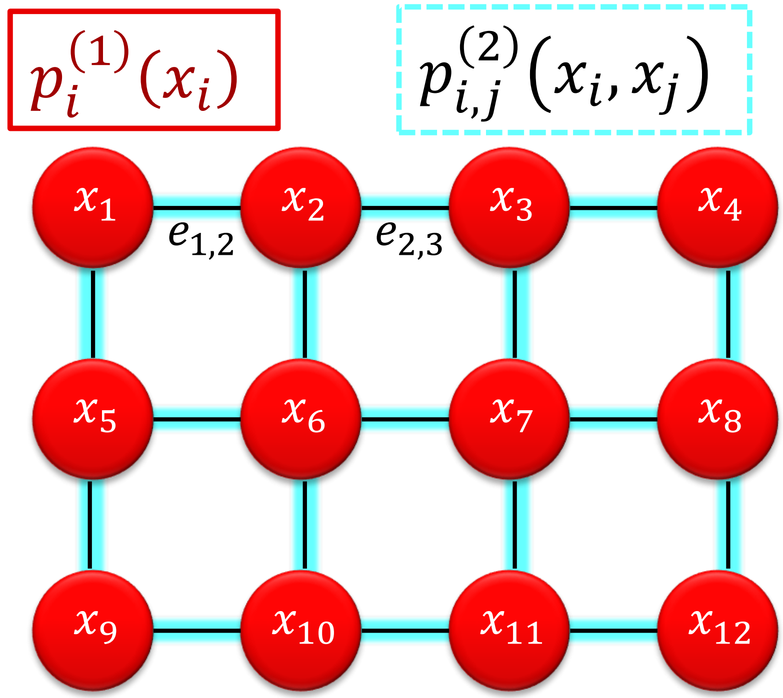
# Literature

**J.S. Yedidia, W.T. Freeman, Y. Weiss:**
Understanding Belief Propagation and its Generalizations.
In: Exploring Artificial Intelligence in the New Millenium, pp.239 – 269, Morgan Kaufmann, 2003.

**P. F. Felzenszwalb, D. P. Huttenlocher:**
Efficient Belief Propagation for Early Vision
In: International Journal of Computer Vision 70(1), 41–54, October 2006.

# Inference via Graph Cut

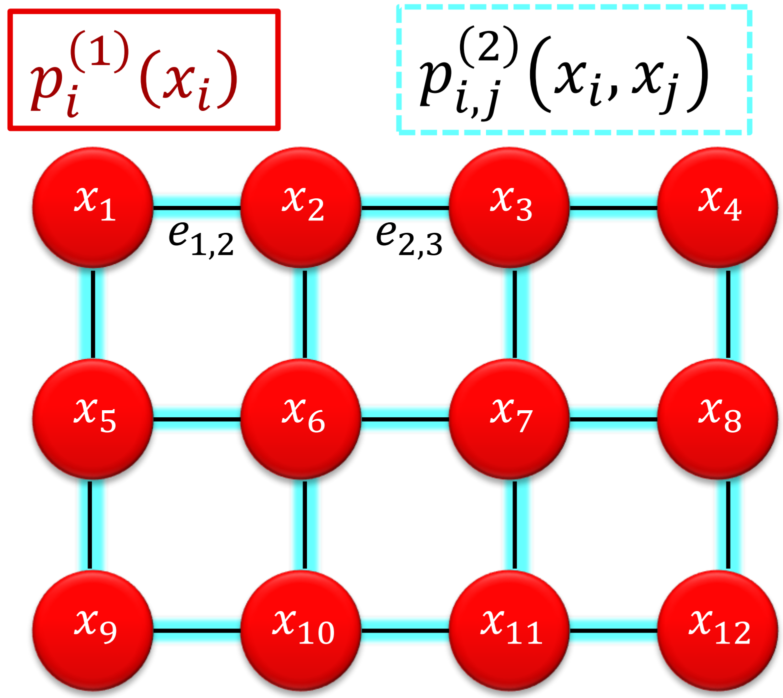
# Pairwise MRF




**Pairwise MRF**

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} \underline{p_i^{(1)}(x_i)} \prod_{i,j \in E} \underline{p_{i,j}^{(2)}(x_i, x_j)}$$

# Neg-Log Likelihood



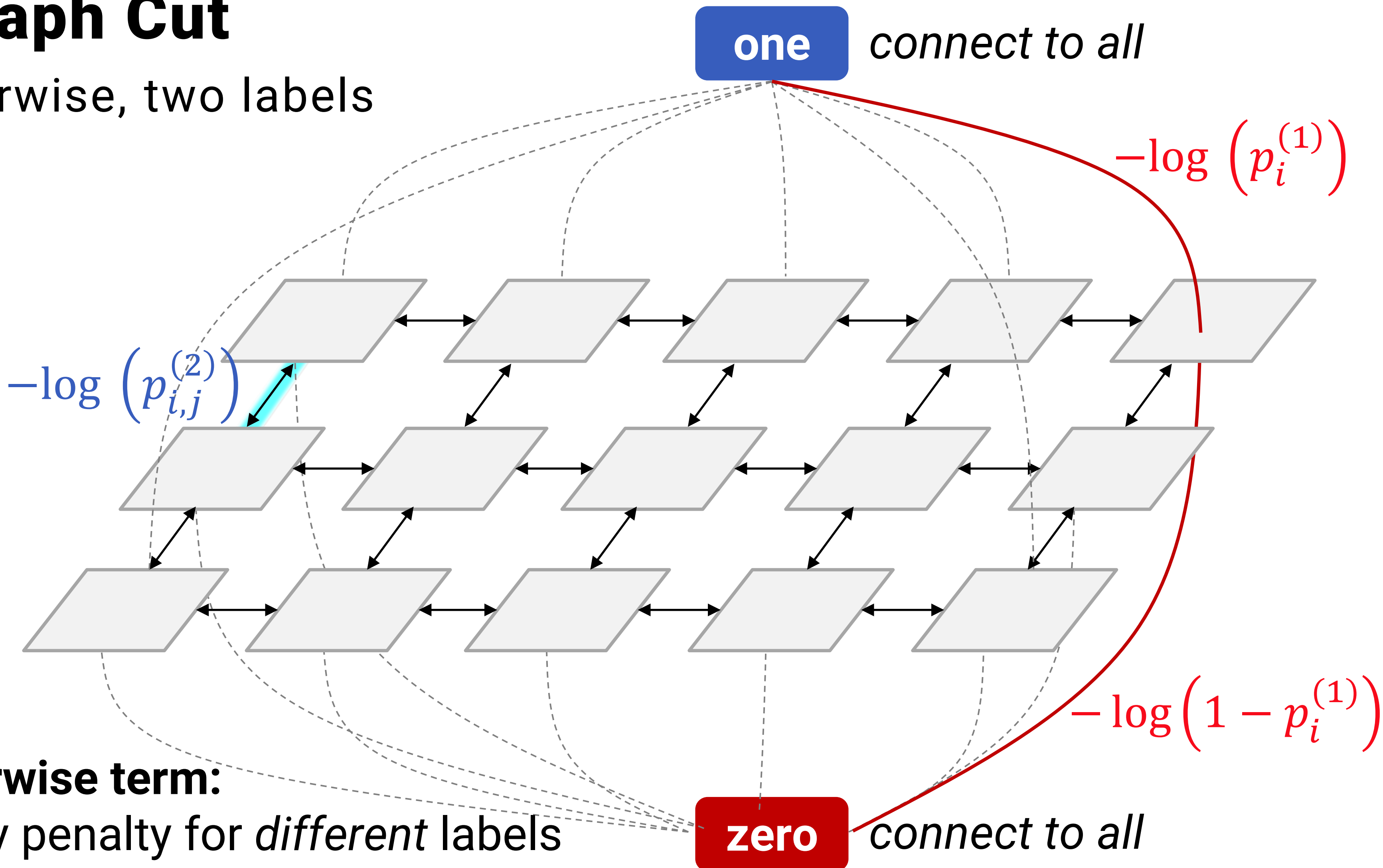

## Pairwise MRF

$$-\log\big(p(x_1, \ldots, x_n)\big) = \sum_{i=1}^{n} \left[-\log\left(p_i^{(1)}(x_i)\right)\right] \sum_{i,j \in E} \left[-\log\left(p_{i,j}^{(2)}(x_i, x_j)\right)\right]$$

# Pairwise MRF Inference

## Graph Cut

Pairwise, two labels




**Pairwise term:**
Only penalty for *different* labels possible (submodular!)

# Multi-Label

## Iterated Graph cuts

- Start with one label everywhere
- Alpha-expansion
  - Ask whether label $i$ would be better
    - Keep it → label zero
    - Switch to the other label
  - Iterate for all labels $i = 1..n$ multiple times
- Approximation guarantees depend on pairwise costs
  - Strict and “generalized” version
- Many related strategies (beta-swaps, etc.)

# Inference via MCMC

# More General Tools

**MCMC (Markov Chain Monte Carlo)**

- Gibbs Sampling
- Metropolis algorithm
- Many variants

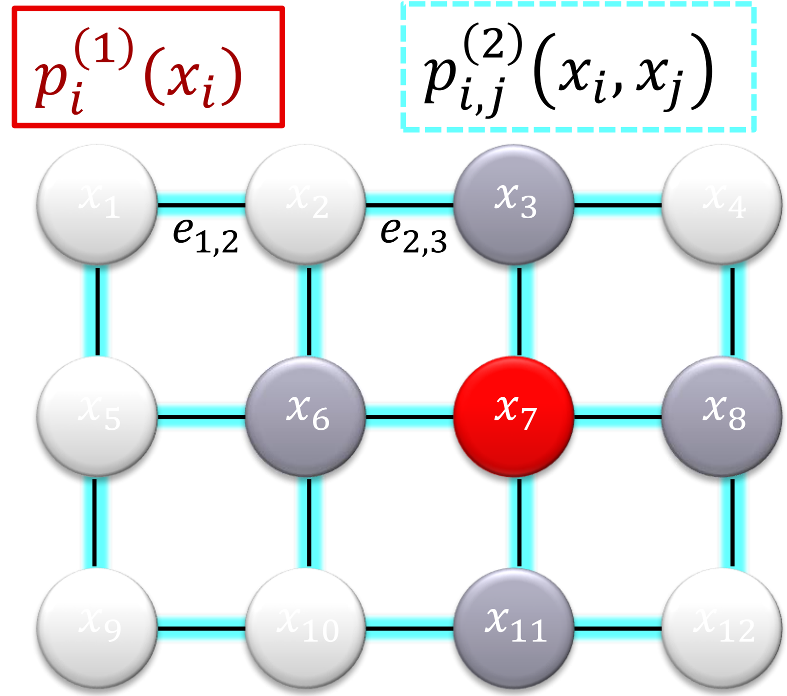
# Idea

## Naive Sampling

- Infeasible (exponential)

## Gibbs Sampling

- Random initialization
- Select random node
  - Fix neighbors
  - Compute local distribution
  - Sample
- Repeat




**Convergence:** Mixing times (hard to estimate)

# Literature

**S. Geman & D. Geman:**
Stochastic Relaxation, Gibbs Distributions,
and the Bayesian Restoration of Images.
In: IEEE Transactions on Pattern Analysis and
Machine Intelligence (PAMI) 6(6), Nov. 1984.

# Inference via Numerical Optimization

# Tractable Cases

**Computationally tractable**

- Gaussian MRFs $\rightarrow$ linear Systems
- Convex MRFs $\rightarrow$ gradient Decent, $l$-BFGS, Newton,…
- Tree-structure approximations

# Learning CRFs

# Supervised MAP Learning (general)

**Inferring parameters** $\theta$

$$P(\theta|X,D) = \frac{P(X,D|\theta)P(\theta)}{P(X,D)}$$

$$\overset{i.i.d.}{=} P(\theta)\prod_{i=1}^{n}\frac{P(x_i,d_i|\theta)}{P(x_i,d_i)}$$

$$= \prod_{i=1}^{n}\frac{P(d_i|x_i,\theta)P(x_i,\theta)P(\theta)}{P(x_i,d_i)}$$

$$= P(\theta)\prod_{i=1}^{n}\frac{P(d_i|x_i,\theta)P(x_i|\theta)}{P(d_i|x_i)P(x_i)}$$

# Supervised MAP Learning (general)

## Learning a pairwise MRF/CRF

- Minimize over $\theta$:

$$= E(\theta) + \sum_{\substack{\text{samples}\\ i}} \sum_{\substack{\text{nodes}\\ k}} E_k^{(1)}\left(d_k^{(i)} \middle| x_k^{(i)}\right) + \sum_{\substack{\text{pairs}\\ l,m}} E_k^{(2)}\left(x_l^{(i)}, x_m^{(i)}\right)$$

$$- \log \sum_{x \in \Omega_x} \prod_{\substack{\text{nodes}\\ k}} p_k^{(1)}\left(d_k^{(i)} \middle| x_k^{(i)}\right) \prod_{\substack{\text{pairs}\\ l,m}} p_k^{(2)}\left(x_l^{(i)}, x_m^{(i)}\right)$$

- "$E$" again denotes neg-log-likelihood
- Problem: Need to evaluate partition function!

# Solution Strategies

## Analytic Computation

- Z is Infeasible in general
- Tree-structured CRFs
  - Exact integration feasible
  - Dynamic programming algorithm, similar to belief propagation
  - Then use gradient descent on objective
- Approximate Marginal Likelihood by MAP Solution
  - Good approximation if density is sharply peaked

# Solution Strategies

## Piecewise Training

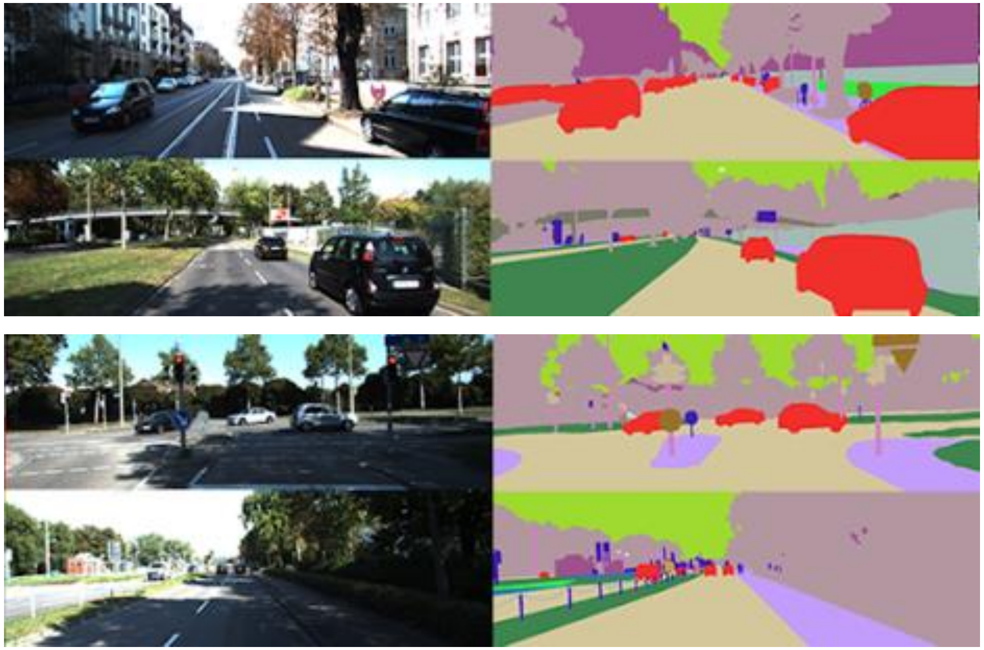
- Learn models for
  - unary terms $p_k^{(1)}(d_k|x_k)$,
  - pairwise terms $p_k^{(2)}(x_l, x_m)$

  separately.
- Problems
  - Uses partition function for separate terms, not the joint model
  - Approximates true $Z(d, x)$ only

**Completely ad-hoc** (e.g. Texture Synthesis)

# Applications

# CRFs for Image Segmentation



## CRF Image Segmentation

- Features per Pixel (SIFT/HOG, also CNNs)
- Neighborhood: likelihood for pairs

Example data from KITTI-Vision Benchmark Suite [Alhaija et al. 2018]
http://www.cvlibs.net/datasets/kitti/eval_semseg.php?benchmark=semantics2015

# Summary

# Markov Random Fields

**Generalization of Markov Chains**

- General Graphs
- Often spatial / tempo-spatial

**Same structure / idea**

- Conditional independence
- Density factorizes

**“Local interactions”**

- Common model in physics
- Common approximation in computer vision

**Complexity more problematic** (inference hard)