# On the likelihood of randomly perturbed max-solutions

Tamir Hazan
Technion

# Inference in machine learning

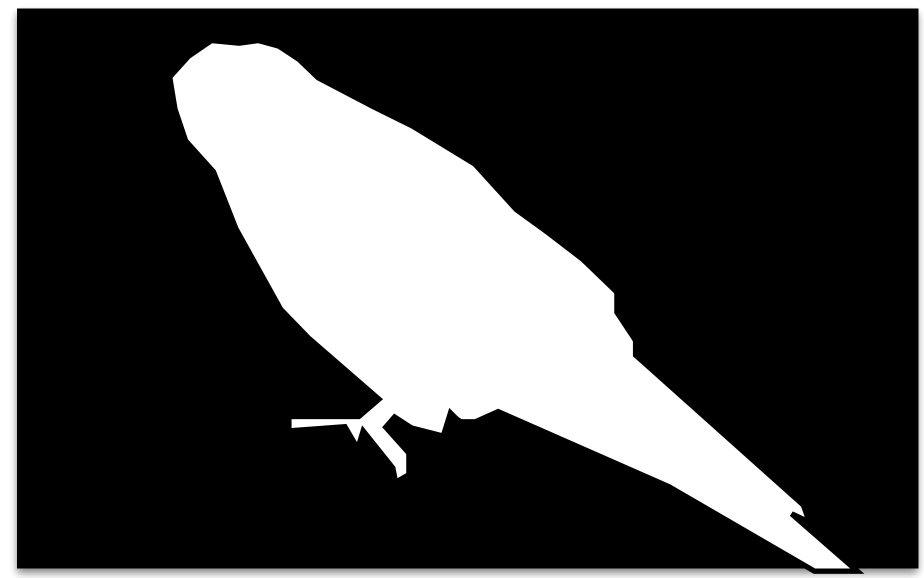
- 20 years ago: does an image contain a person?

# Inference in machine learning

- 10 years ago: which object is in the image?

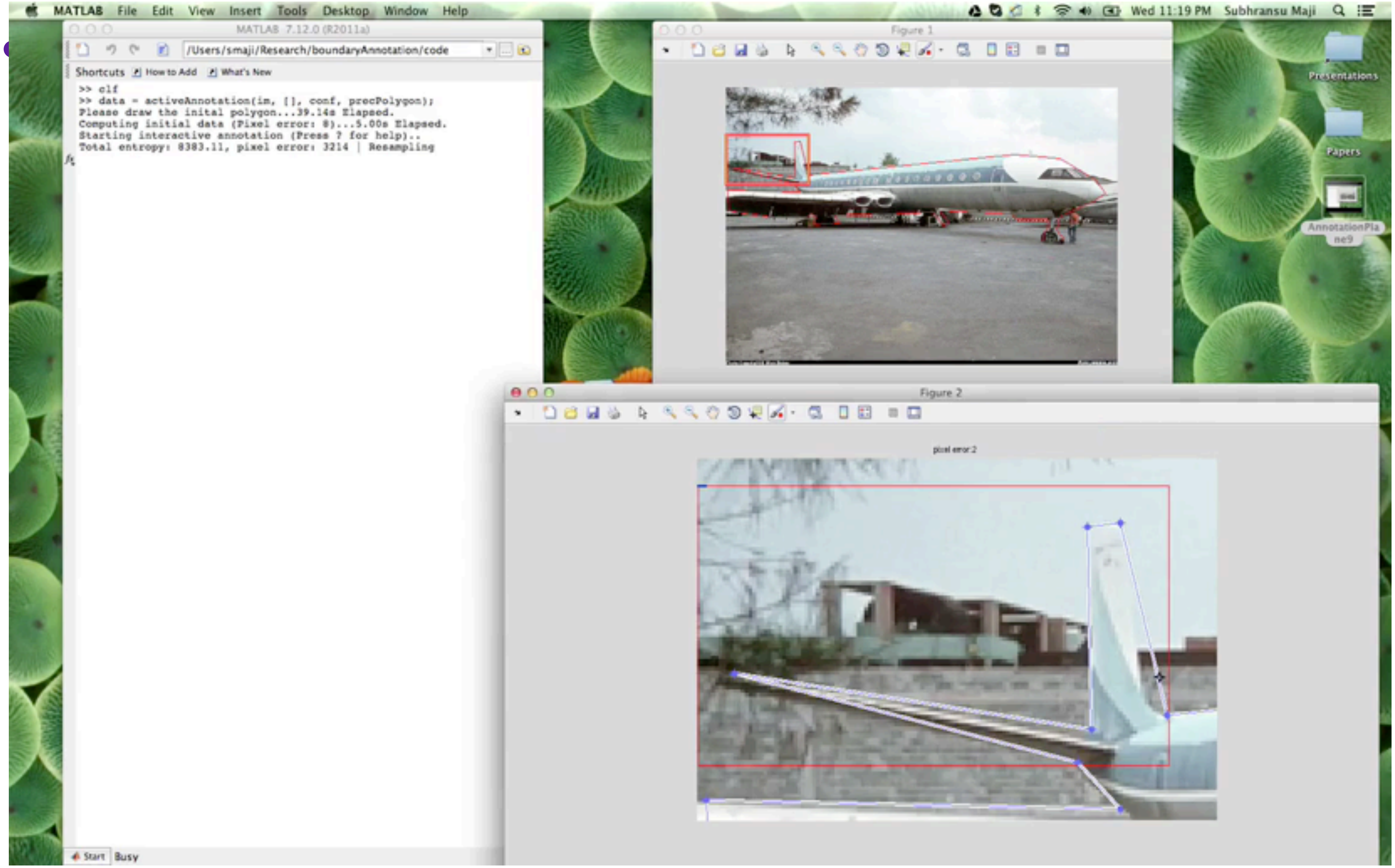
# Inference in machine learning

- Today's challenge:  exponentially many options

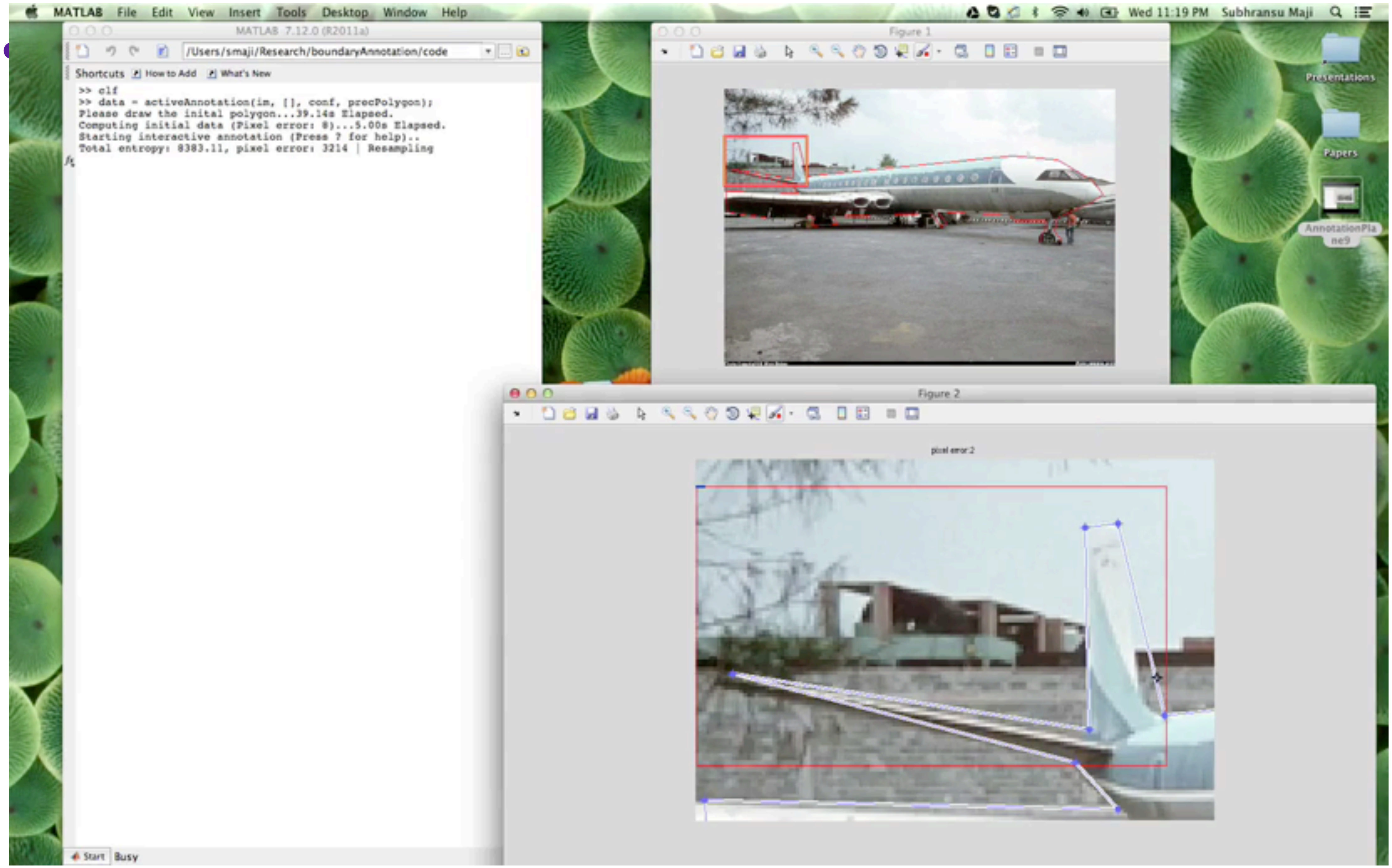


- For each pixel: decide if it is foreground or background.
- The space of possible structures is exponential

# Inference in machine learning



# Inference in machine learning

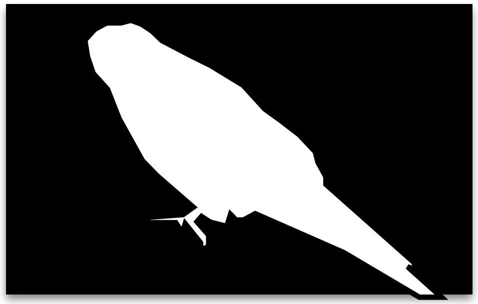


# Inference in machine learning

- Complex structures dominate machine learning applications:

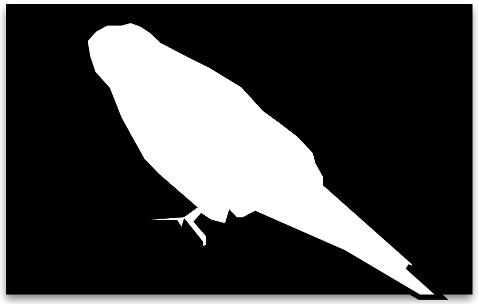
# Inference in machine learning

- Complex structures dominate machine learning applications:
  - Computer vision

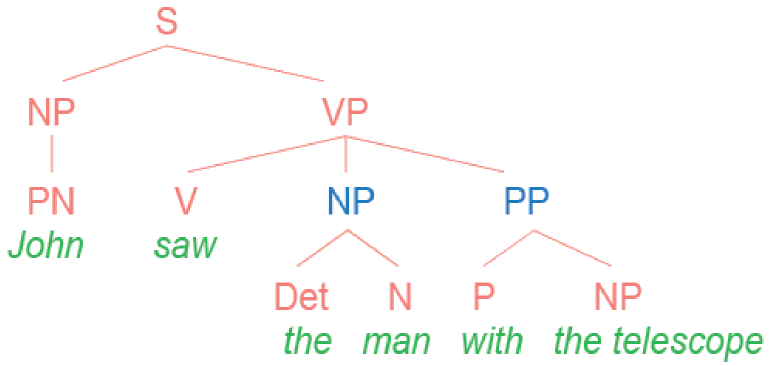


# Inference in machine learning

- Complex structures dominate machine learning applications:
  - Computer vision

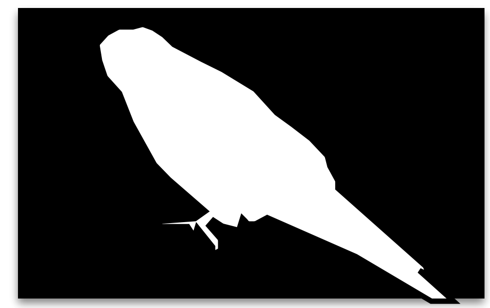


  - Natural language processing

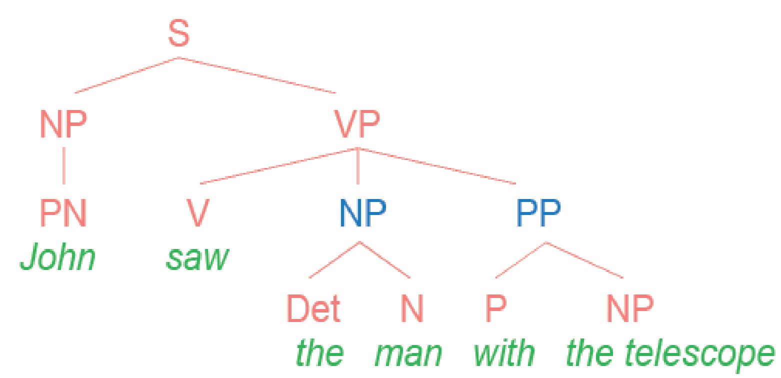


# Inference in machine learning

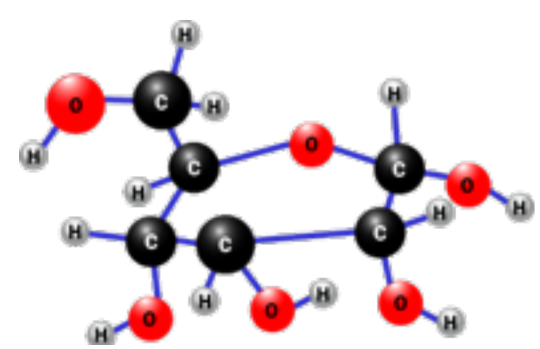
- Complex structures dominate machine learning applications:
  - Computer vision

  - Natural language processing




  - Computational biology



  - and more..

# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.

# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models

# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

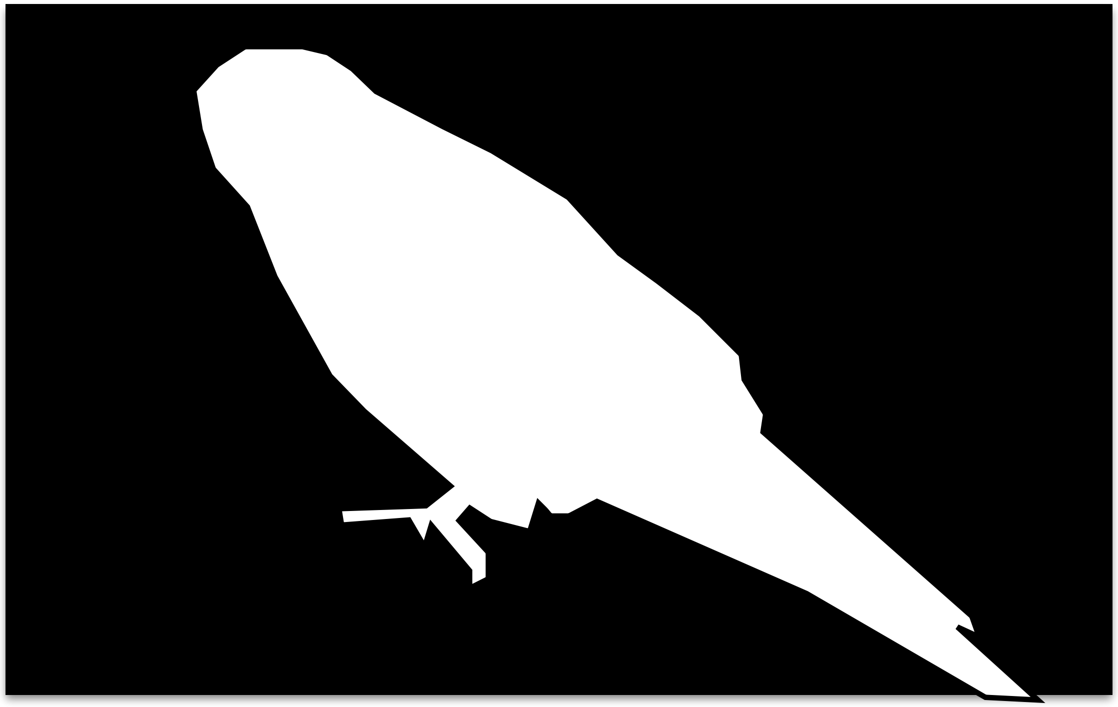
# Inference in machine learning

- machine learning applications are characterized by:
  - complex structures $y = (y_1, ..., y_n)$

# Inference in machine learning

- machine learning applications are characterized by:
  - complex structures $y = (y_1, ..., y_n)$

$$y \in \{0, 1\}^n$$



# Inference in machine learning

- machine learning applications are characterized by:
  - complex structures $y = (y_1, ..., y_n)$

# Inference in machine learning

- machine learning applications are characterized by:
  - complex structures $y = (y_1, ..., y_n)$
  - potential function that scores these structures

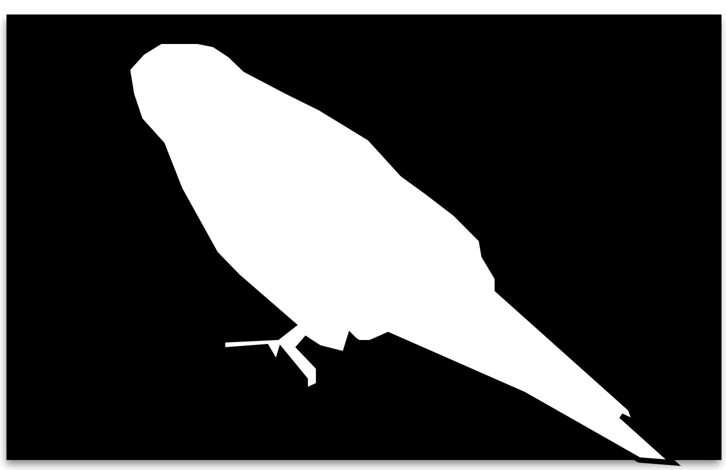
$$\theta(y_1, ..., y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$

# Inference in machine learning

- machine learning applications are characterized by:
  - complex structures $y = (y_1, ..., y_n)$
  - potential function that scores these structures

$$\theta(y_1, ..., y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$

high score



low score

# Gibbs distribution

$$p(y_1, ..., y_n) = \frac{1}{Z} \exp\left( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \right)$$

# Gibbs distribution

$$p(y_1, ..., y_n) = \frac{1}{Z} \exp\Big( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \Big)$$

- MCMC samplers:

# Gibbs distribution

$$p(y_1, ..., y_n) = \frac{1}{Z} \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang

# Gibbs distribution

$$p(y_1, ..., y_n) = \frac{1}{Z} \exp\left( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \right)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang
- Many efficient sampling algorithms for special cases:

# Gibbs distribution

$$p(y_1, \ldots, y_n) = \frac{1}{Z} \exp\left( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \right)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang
- Many efficient sampling algorithms for special cases:
  - Ising models (Jerrum 93)

# Gibbs distribution

$$p(y_1, \ldots, y_n) = \frac{1}{Z} \exp\Big( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \Big)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang
- Many efficient sampling algorithms for special cases:
  - Ising models (Jerrum 93)
  - Counting bi-partite matchings in planar graphs (Kasteleyn 61)

# Gibbs distribution

$$p(y_1, ..., y_n) = \frac{1}{Z} \exp\left(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\right)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang
- Many efficient sampling algorithms for special cases:
  - Ising models (Jerrum 93)
  - Counting bi-partite matchings in planar graphs (Kasteleyn 61)
  - Approximating the permanent (Jerrum 04)

# Gibbs distribution

$$p(y_1, \ldots, y_n) = \frac{1}{Z} \exp\Big( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \Big)$$

- MCMC samplers:
  - Gibbs sampling, Metropolis-Hastings, Swendsen-Wang
- Many efficient sampling algorithms for special cases:
  - Ising models (Jerrum 93)
  - Counting bi-partite matchings in planar graphs (Kasteleyn 61)
  - Approximating the permanent (Jerrum 04)
  - Many others…

# Gibbs distribution

- Gibbs distribution has a significant impact on statistics and computer science

# Gibbs distribution

- Gibbs distribution has a significant impact on statistics and computer science
  - Efficient sampling in Ising models (Jerrum 93)

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

# Gibbs distribution

- Gibbs distribution has a significant impact on statistics and computer science
  - Efficient sampling in Ising models (Jerrum 93)
  - Attractive pairwise potentials

$$\theta_{i,j}(y_i, y_j) = \begin{cases} w_{i,j} & \text{if } y_i = y_j \\ -w_{i,j} & \text{otherwise} \end{cases}$$

$$w_{i,j} \geq 0$$

  - No data terms

$$\theta_i(y_i) = 0$$

$$p(y) \propto \exp\left( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \right)$$

# Gibbs distribution

- Gibbs distribution has a significant impact on statistics and computer science
  - Efficient sampling in Ising models (Jerrum 93)
  - Attractive pairwise potentials

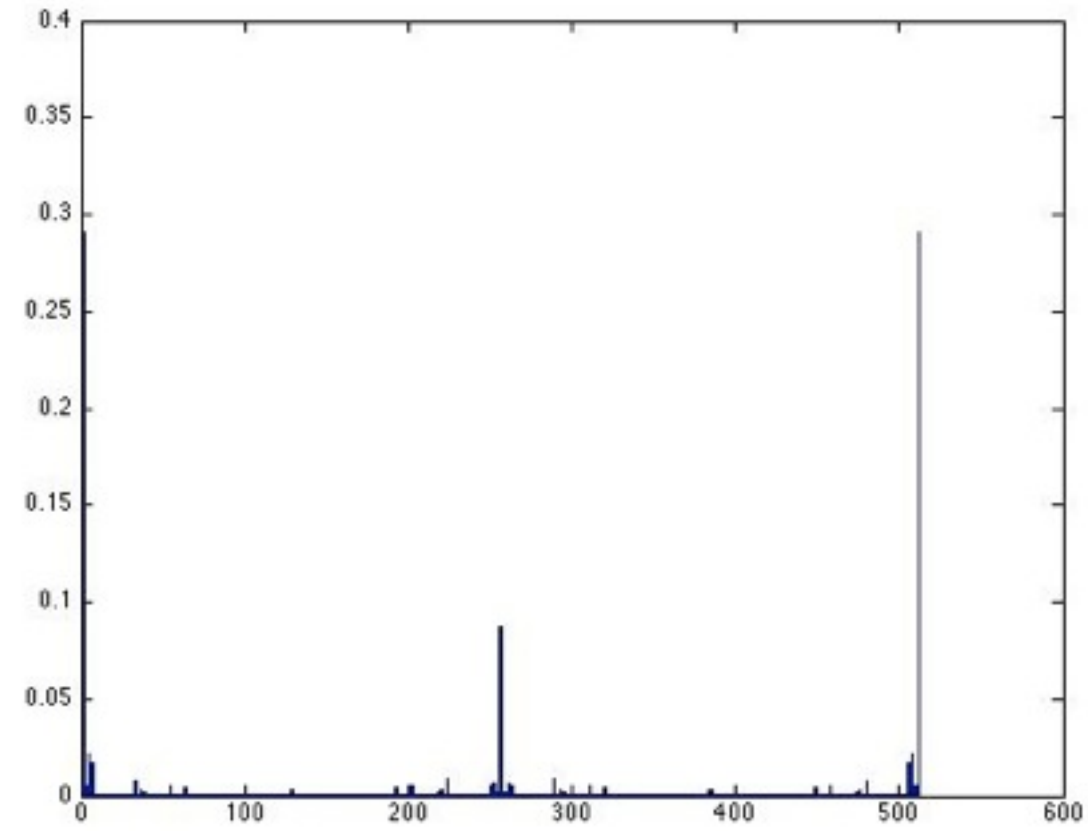
$$\theta_{i,j}(y_i, y_j) = \begin{cases} w_{i,j} & \text{if } y_i = y_j \\ -w_{i,j} & \text{otherwise} \end{cases}$$

$$w_{i,j} \geq 0$$

  - No data terms

$$\theta_i(y_i) = 0$$

$$p(y) \propto \exp\left( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \right)$$



# Gibbs distribution

- Gibbs distribution has a significant impact on statistics and computer science
  - Efficient sampling in Ising models (Jerrum 93)
  - Attractive pairwise potentials

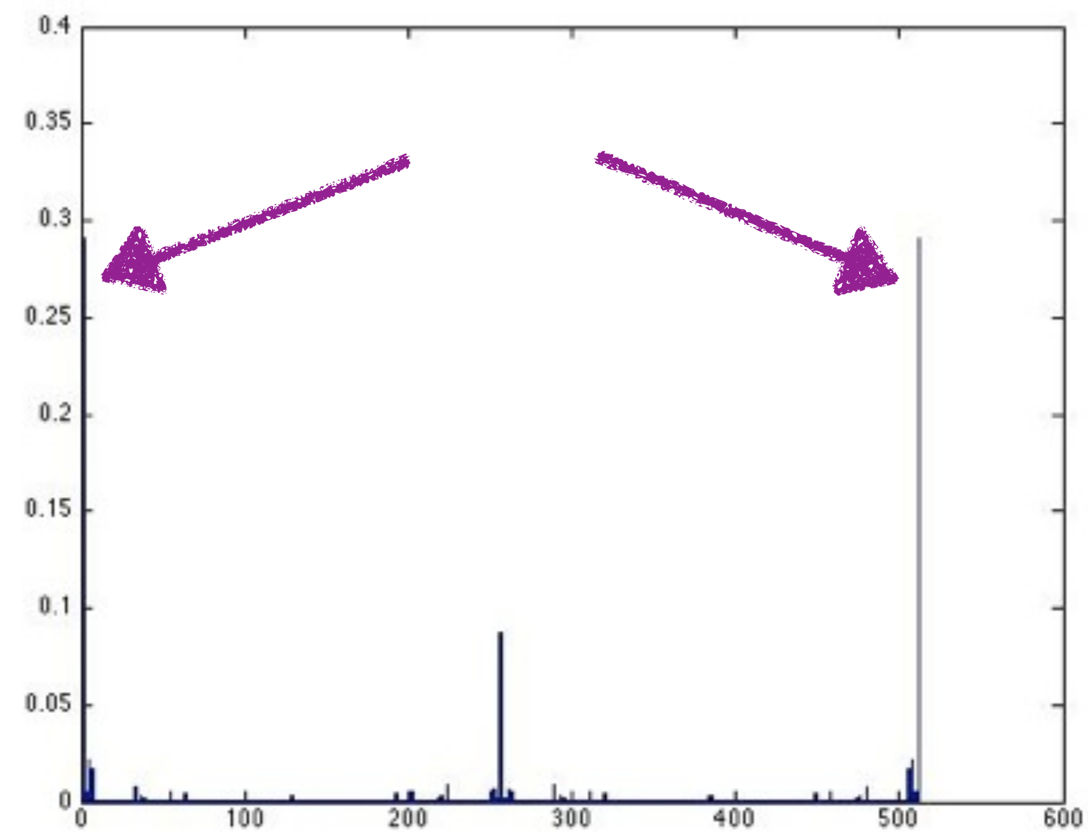
$$\theta_{i,j}(y_i, y_j) = \begin{cases} w_{i,j} & \text{if } y_i = y_j \\ -w_{i,j} & \text{otherwise} \end{cases}$$

$$w_{i,j} \geq 0$$

  - No data terms

$$\theta_i(y_i) = 0$$

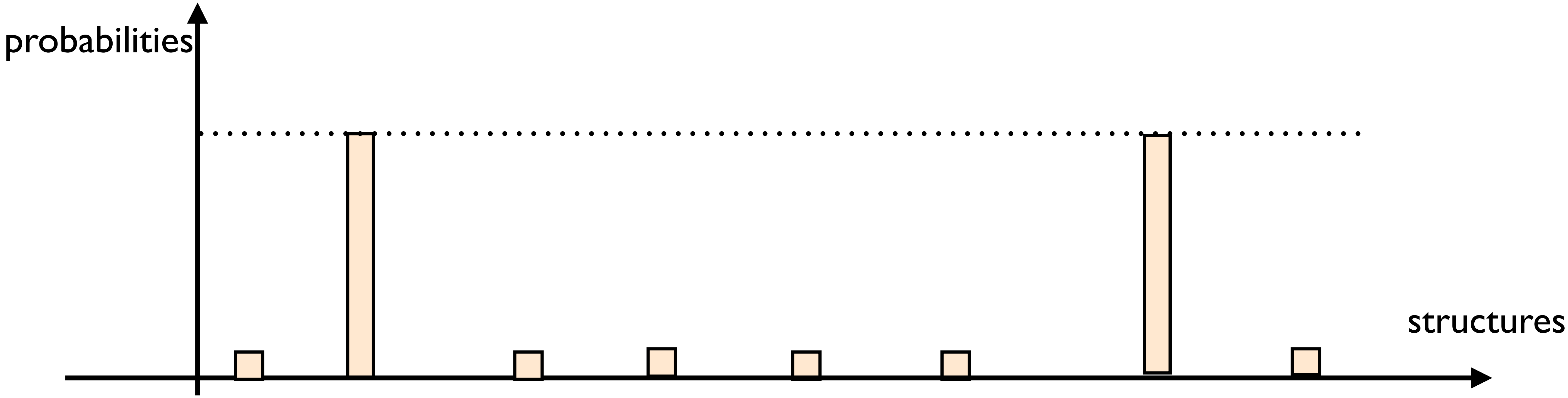
$$p(y) \propto \exp\left(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\right)$$




- Nicely behaved distribution that is centered around the (1,…,1) or (0,…,0)

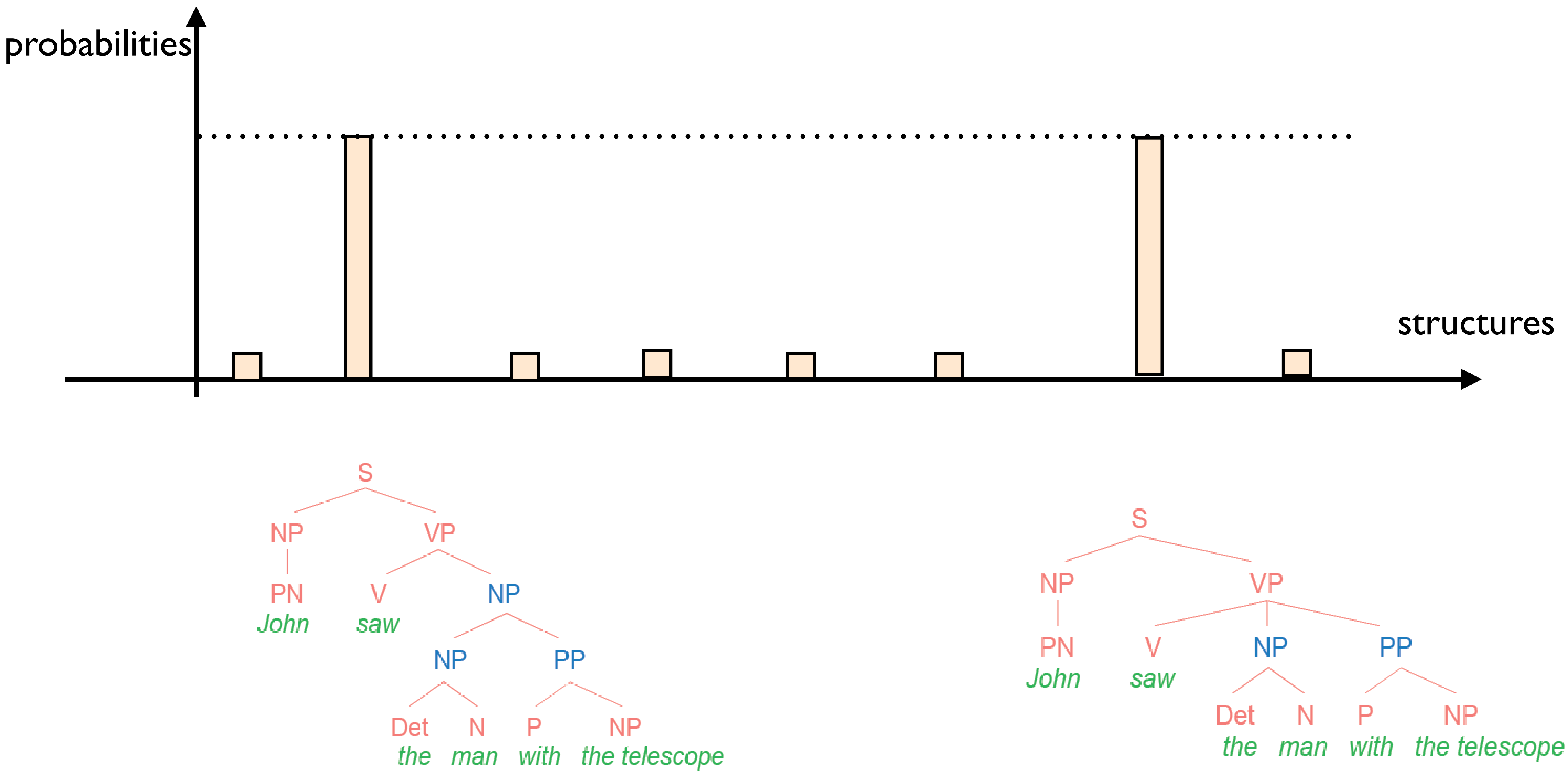
# Sampling likely structures

- Sampling likely structures may easily handle ambiguities



# Sampling likely structures

- Sampling likely structures may easily handle ambiguities

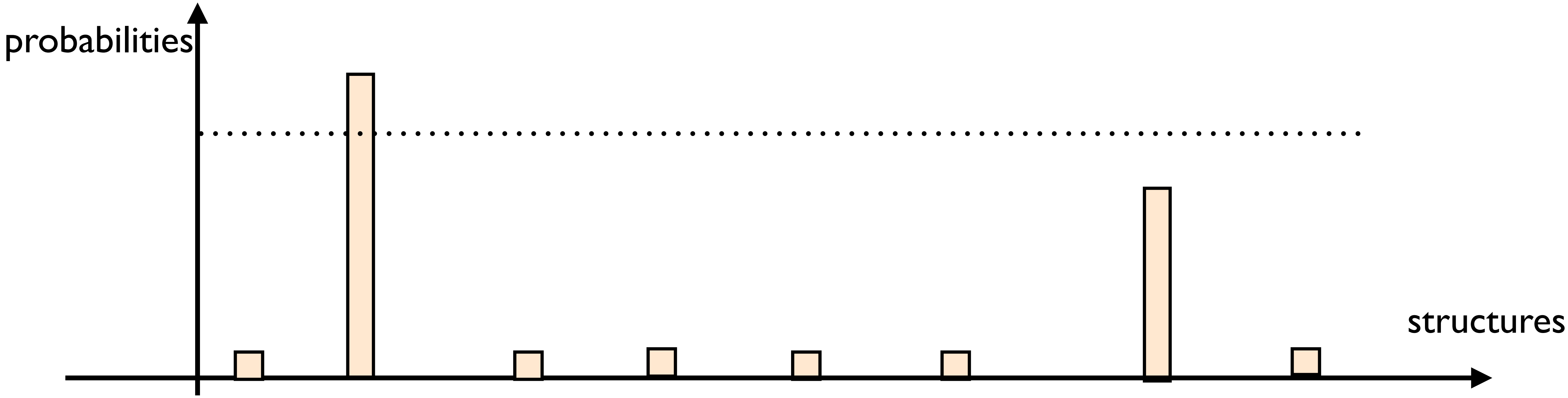


# Sampling likely structures

- Sampling likely structures may easily handle inaccurate modeling

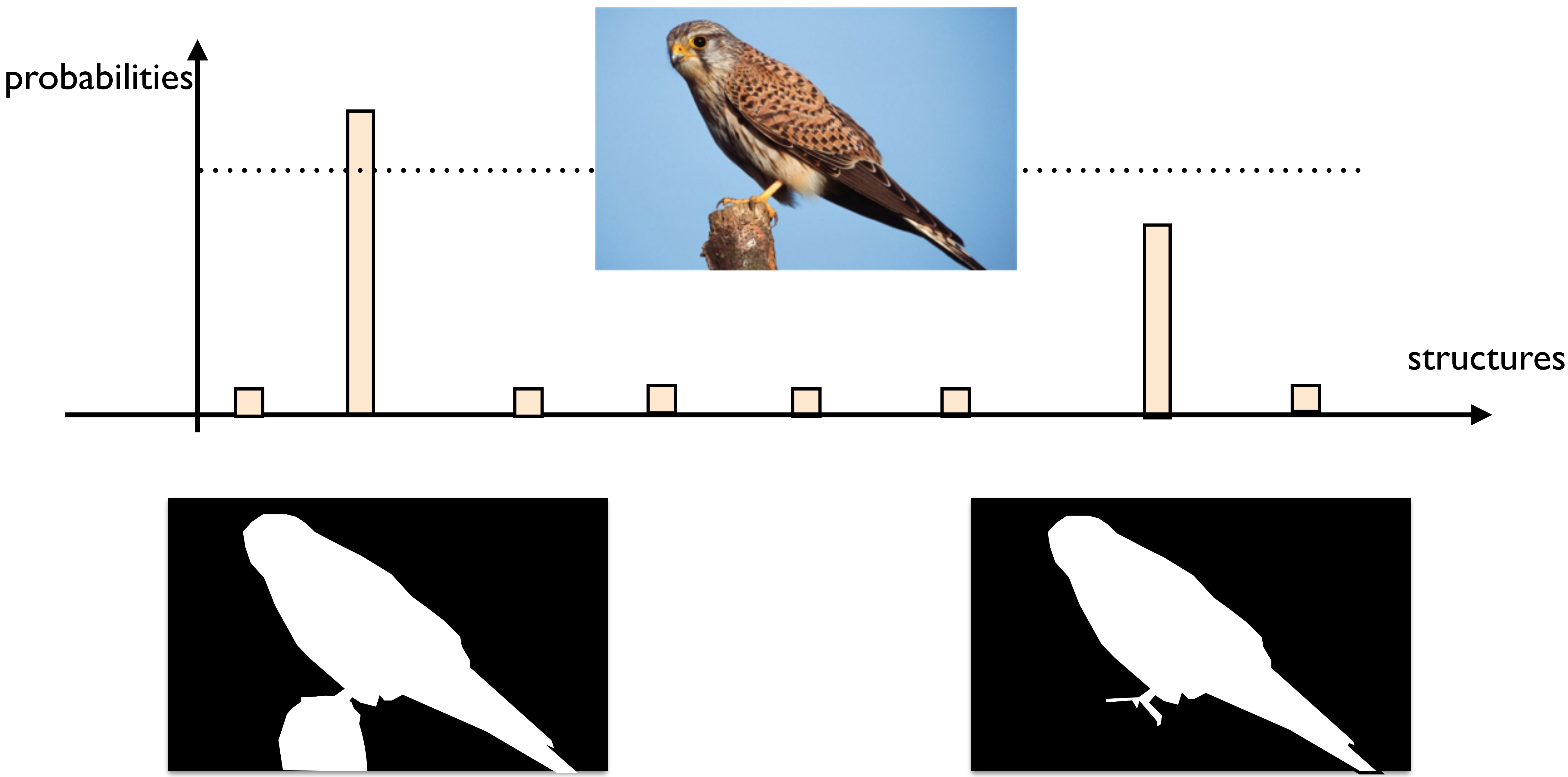
# Sampling likely structures

- Sampling likely structures may easily handle inaccurate modeling



# Sampling likely structures

- Sampling likely structures may easily handle inaccurate modeling



# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

- $x_i$ RGB color of pixel i

$$\theta_i(y_i) = \log q(y_i | x_i)$$

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

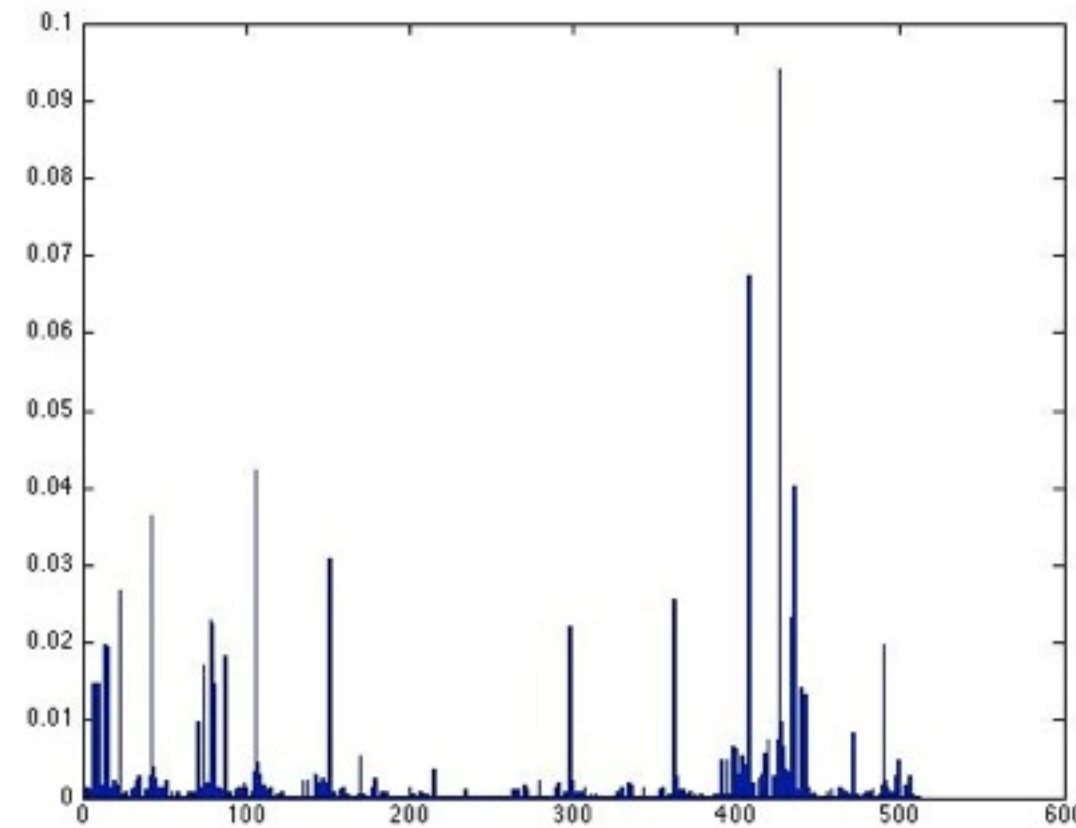
# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

- $x_i$ RGB color of pixel i

$$\theta_i(y_i) = \log q(y_i | x_i)$$

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

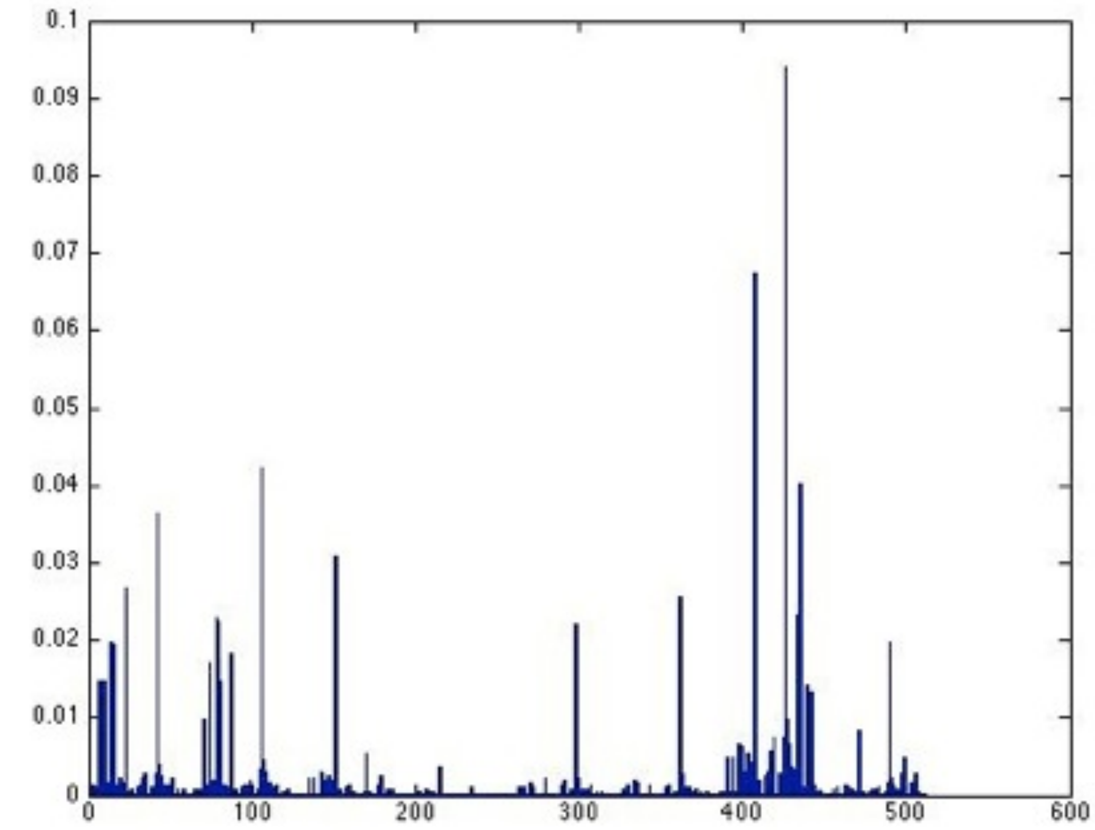


# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

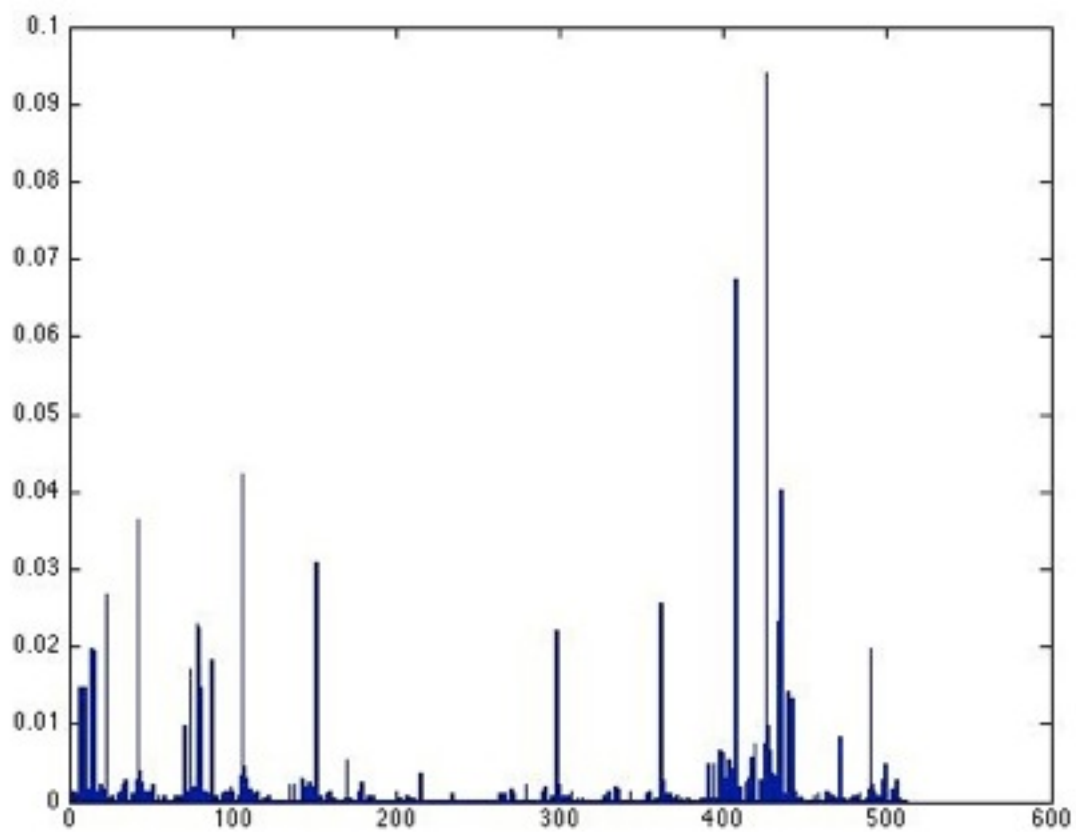
$$\theta_i(y_i) = \log q(y_i|x_i)$$

$$p(y) \propto \exp\left(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\right)$$



# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

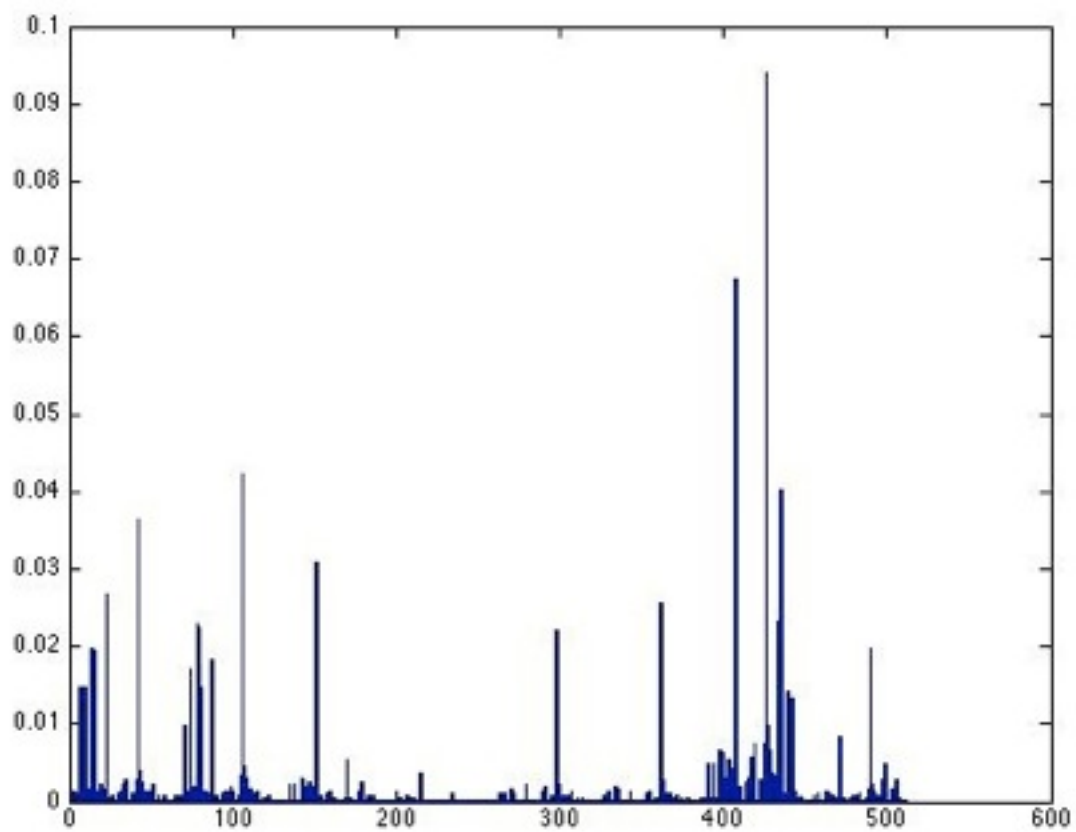



$$\theta_i(y_i) = \log q(y_i|x_i)$$

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$

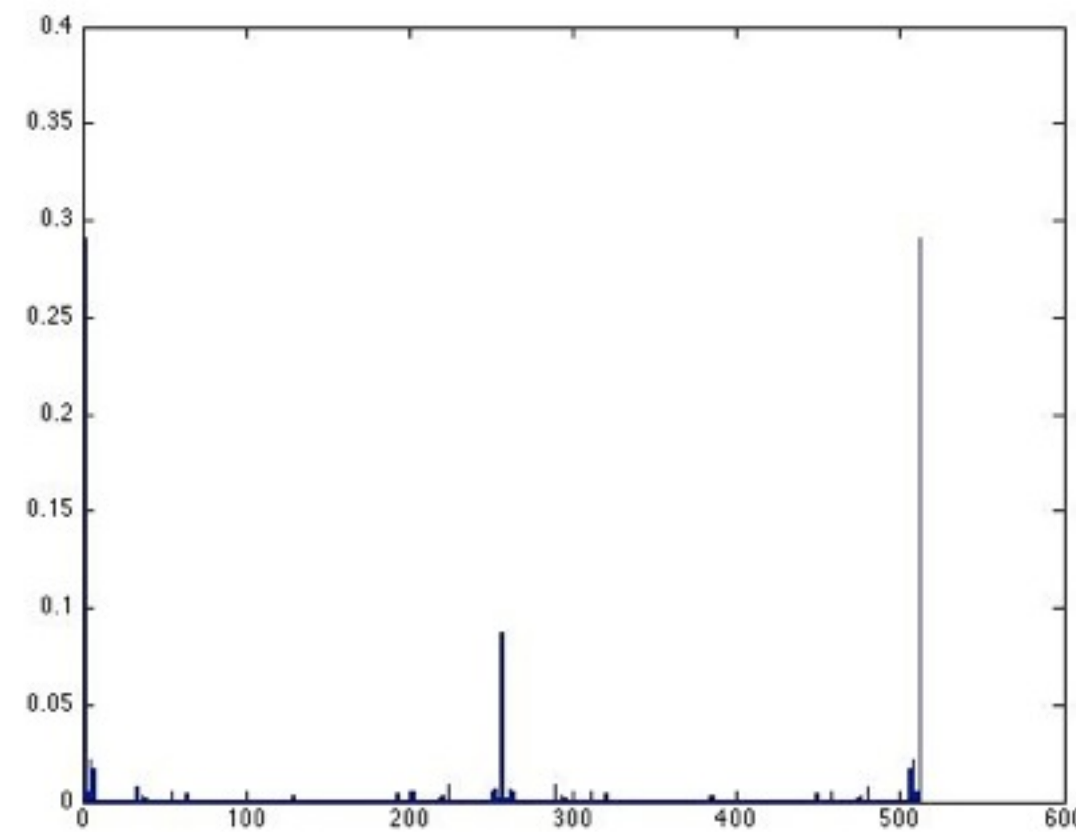
# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)




$$\theta_i(y_i) = \log q(y_i|x_i)$$

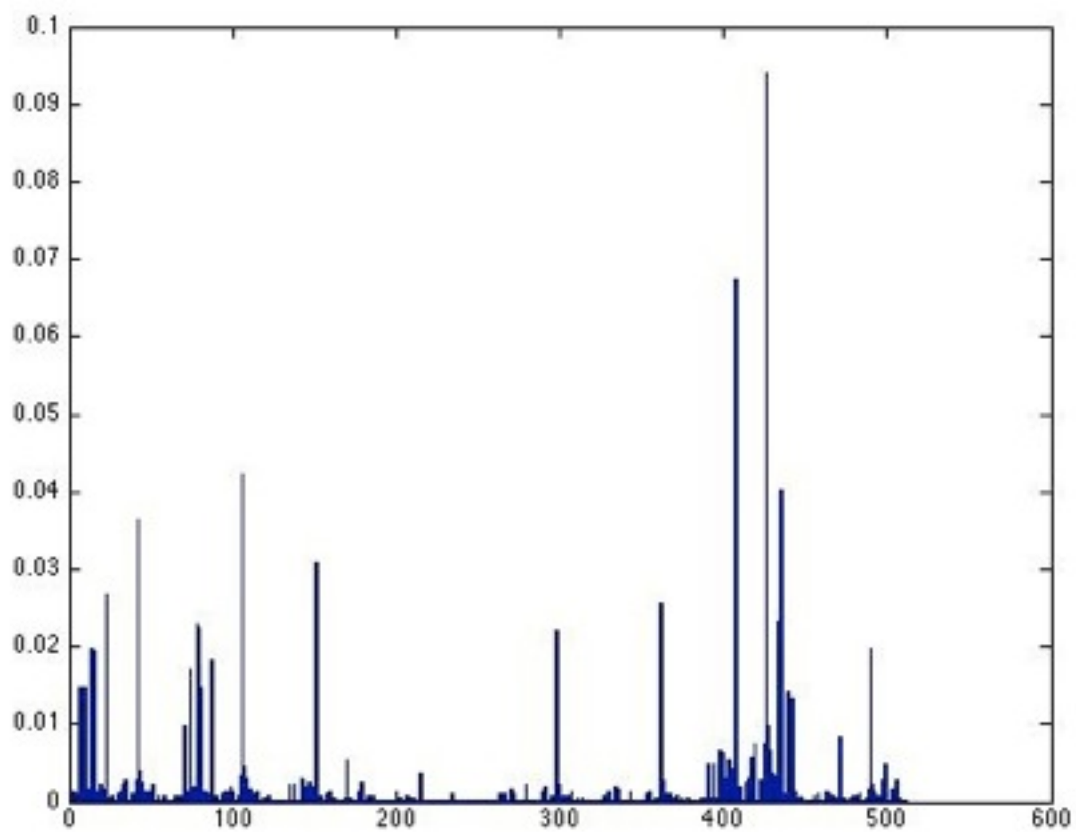
- Recall: sampling from the Gibbs distribution is easy in Ising models (Jerrum 93)




$$\theta_i(y_i) = 0$$

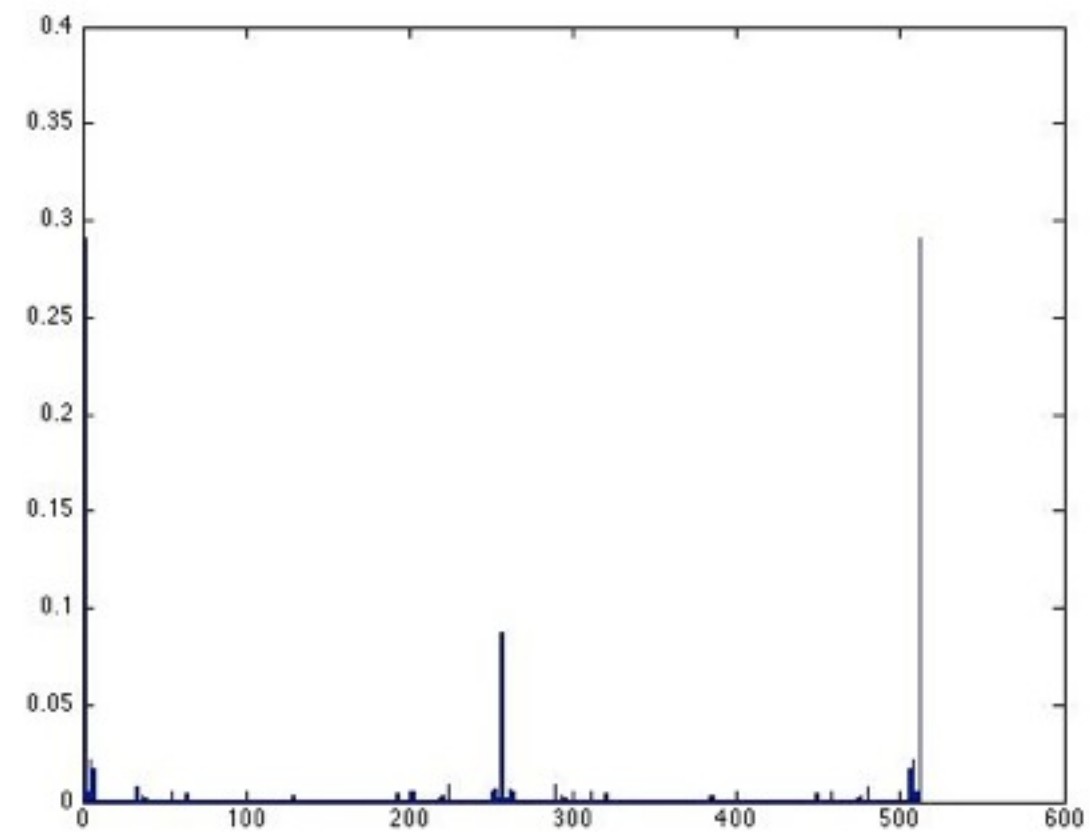
# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)




$\theta_i(y_i) = \log q(y_i|x_i)$

- Recall: sampling from the Gibbs distribution is easy in Ising models (Jerrum 93)

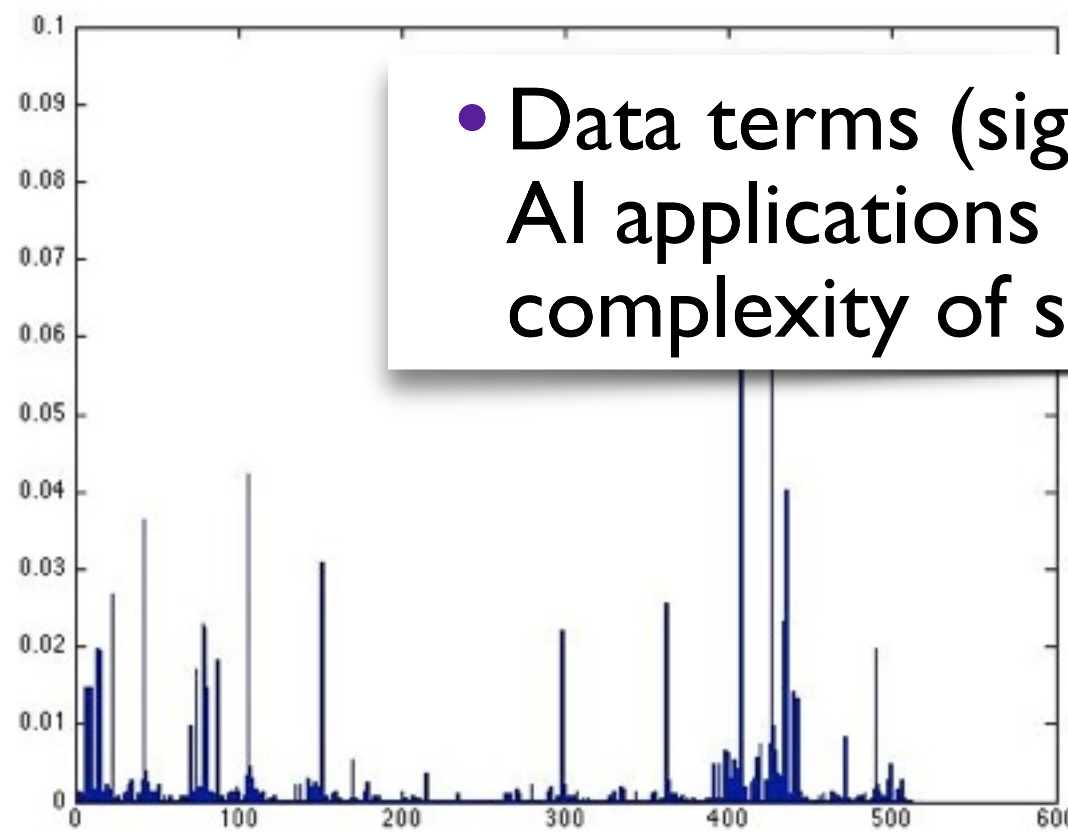


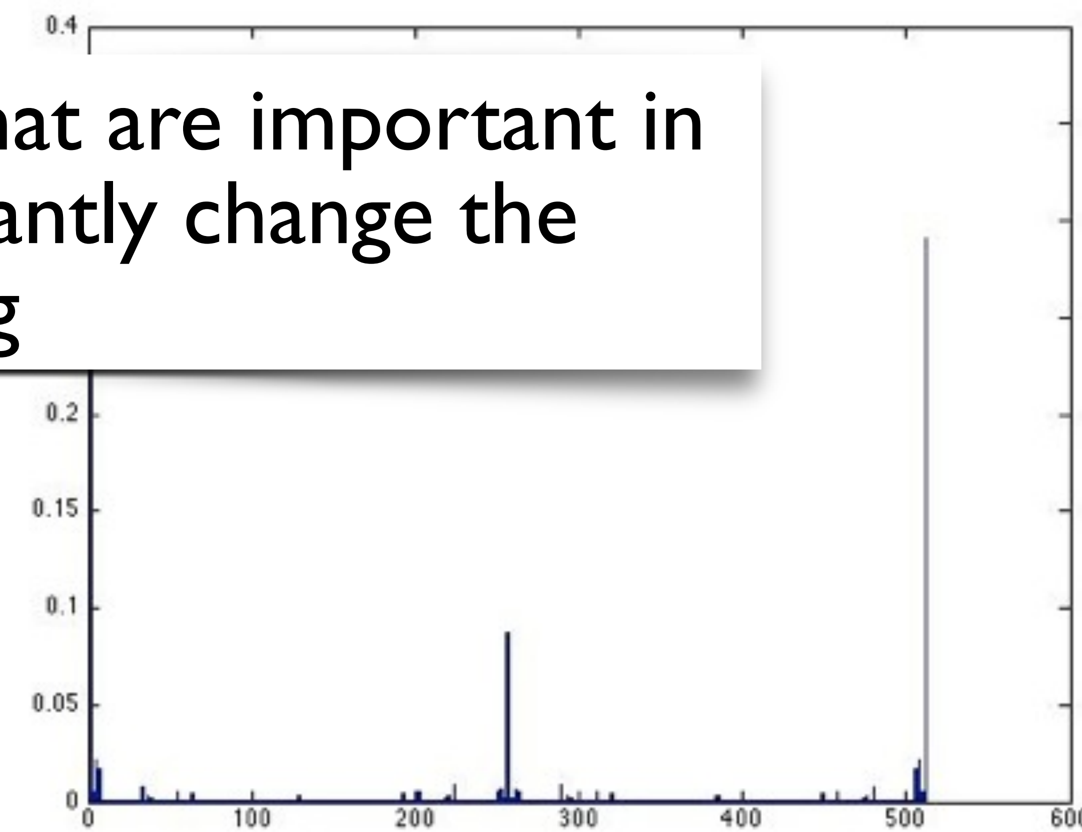

$\theta_i(y_i) = 0$

# Sampling likely structures

- Sampling from the Gibbs distribution is provably hard in AI applications (Goldberg 05, Jerrum 93)

- Recall: sampling from the Gibbs distribution is easy in Ising models (Jerrum 93)




- Data terms (signals) that are important in AI applications significantly change the complexity of sampling

$\theta_i(y_i) = \log q(y_i|x_i)$

$\theta_i(y_i) = 0$

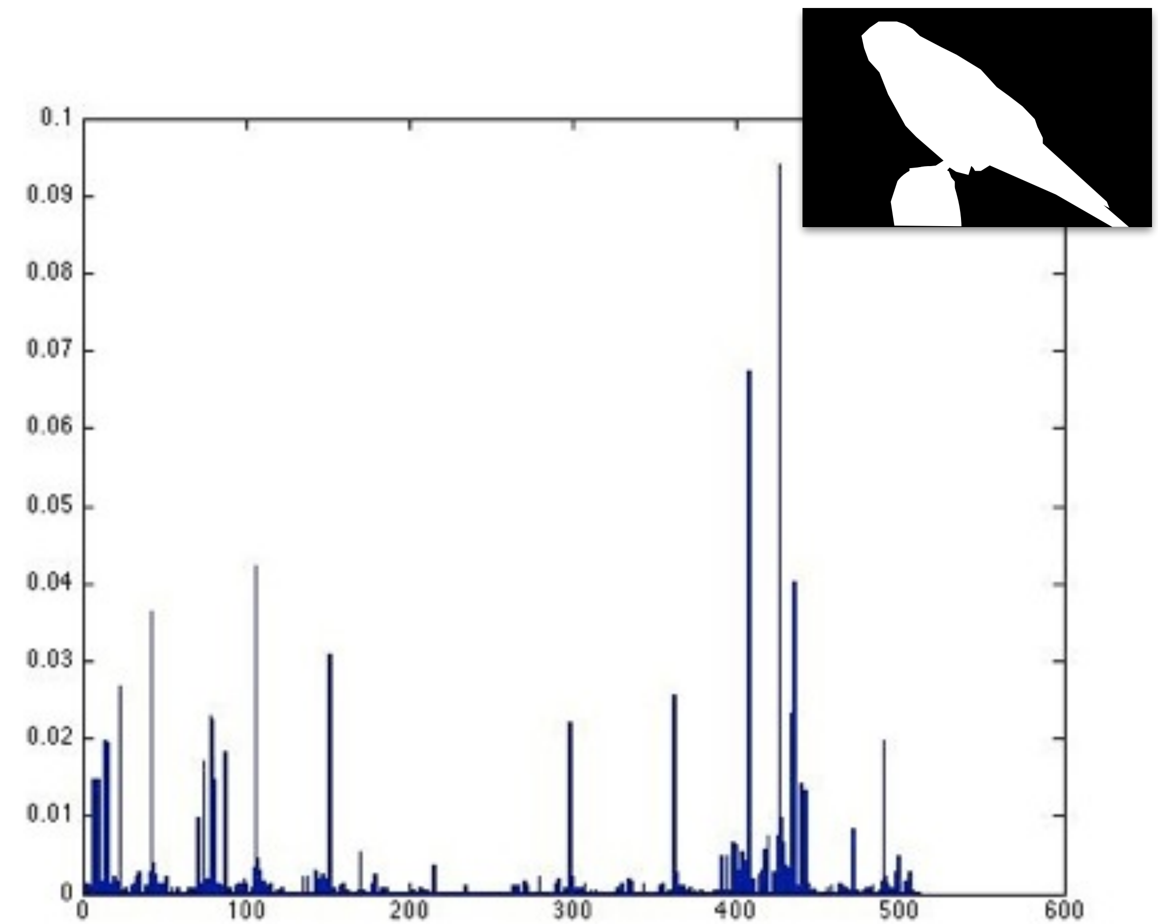
# Most likely structure

- Instead of sampling, it may be significantly faster to find the most likely structure

# Most likely structure

- Instead of sampling, it may be significantly faster to find the most likely structure

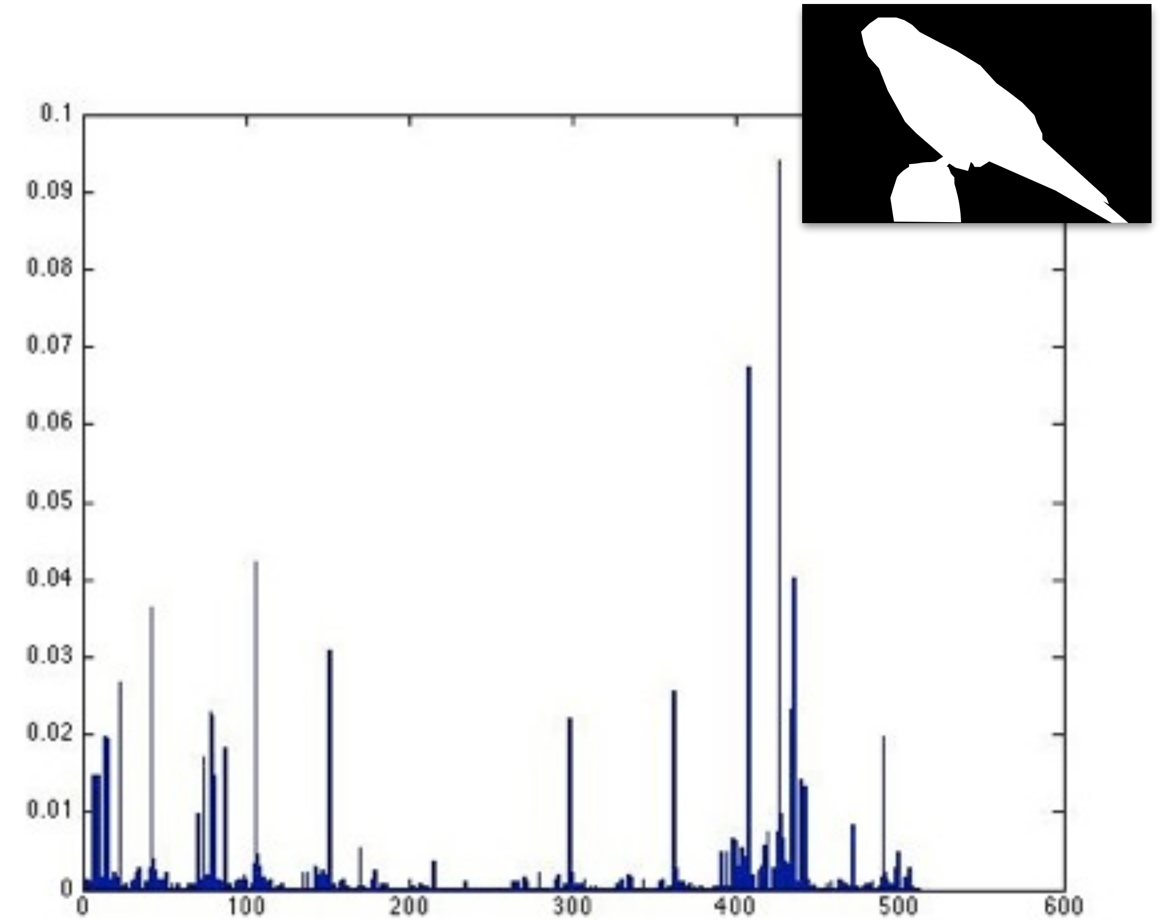
- The most likely structure



# Most likely structure

- Instead of sampling, it may be significantly faster to find the most likely structure
  - Graph-cuts

- The most likely structure



# Most likely structure

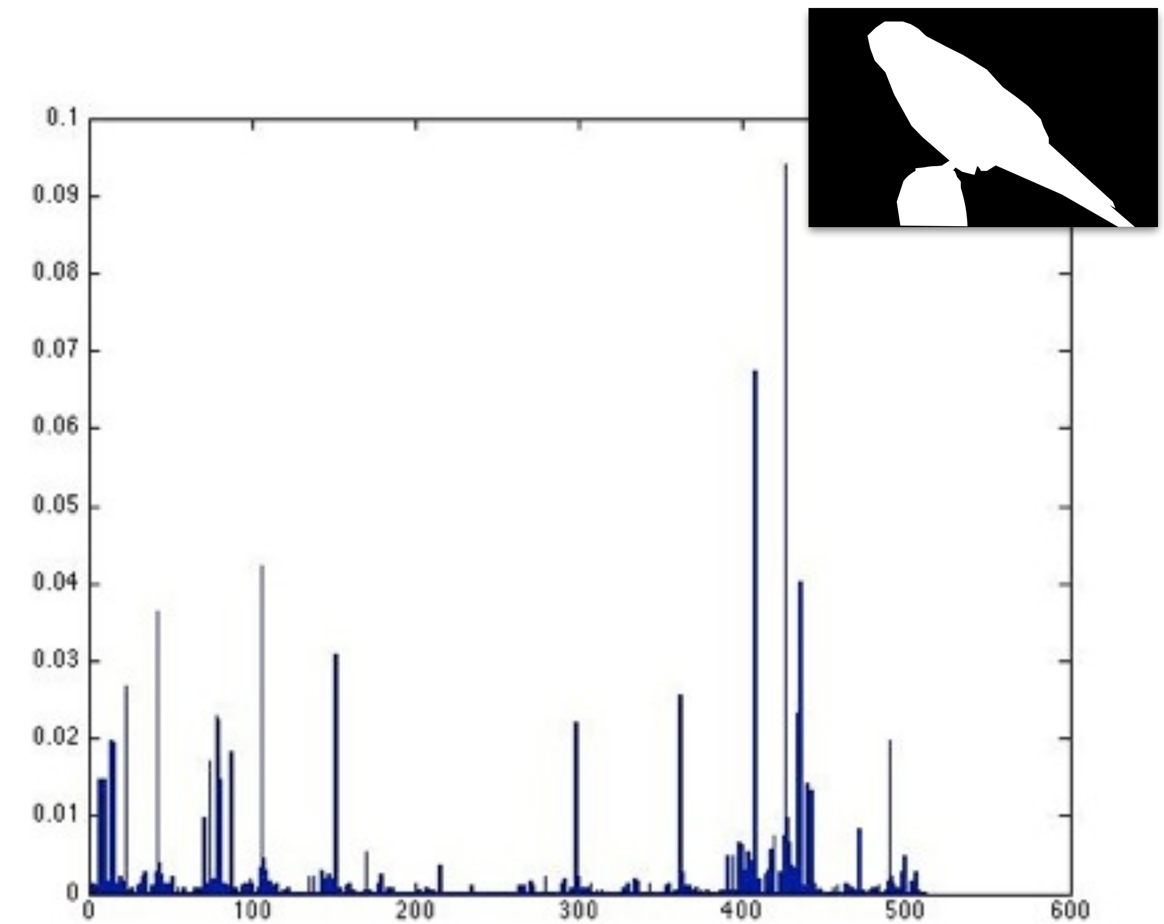
- Instead of sampling, it may be significantly faster to find the most likely structure
  - Graph-cuts

$$\theta_{i,j}(y_i, y_j) = \begin{cases} w_{i,j} & \text{if } y_i = y_j \\ -w_{i,j} & \text{otherwise} \end{cases}$$

$$w_{i,j} \geq 0$$

$$\theta_i(y_i) = \log q(y_i | x_i)$$

- The most likely structure



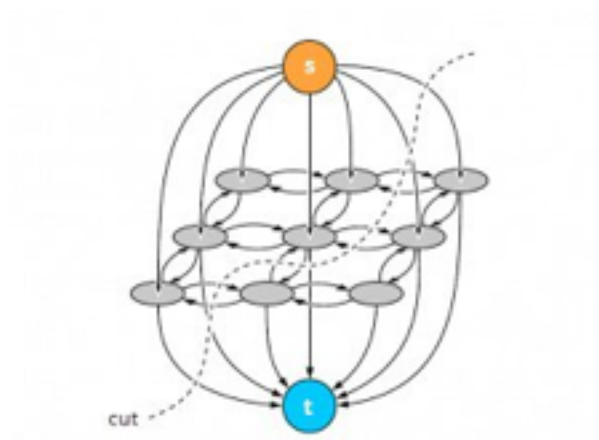
# Most likely structure

- Instead of sampling, it may be significantly faster to find the most likely structure
  - Graph-cuts

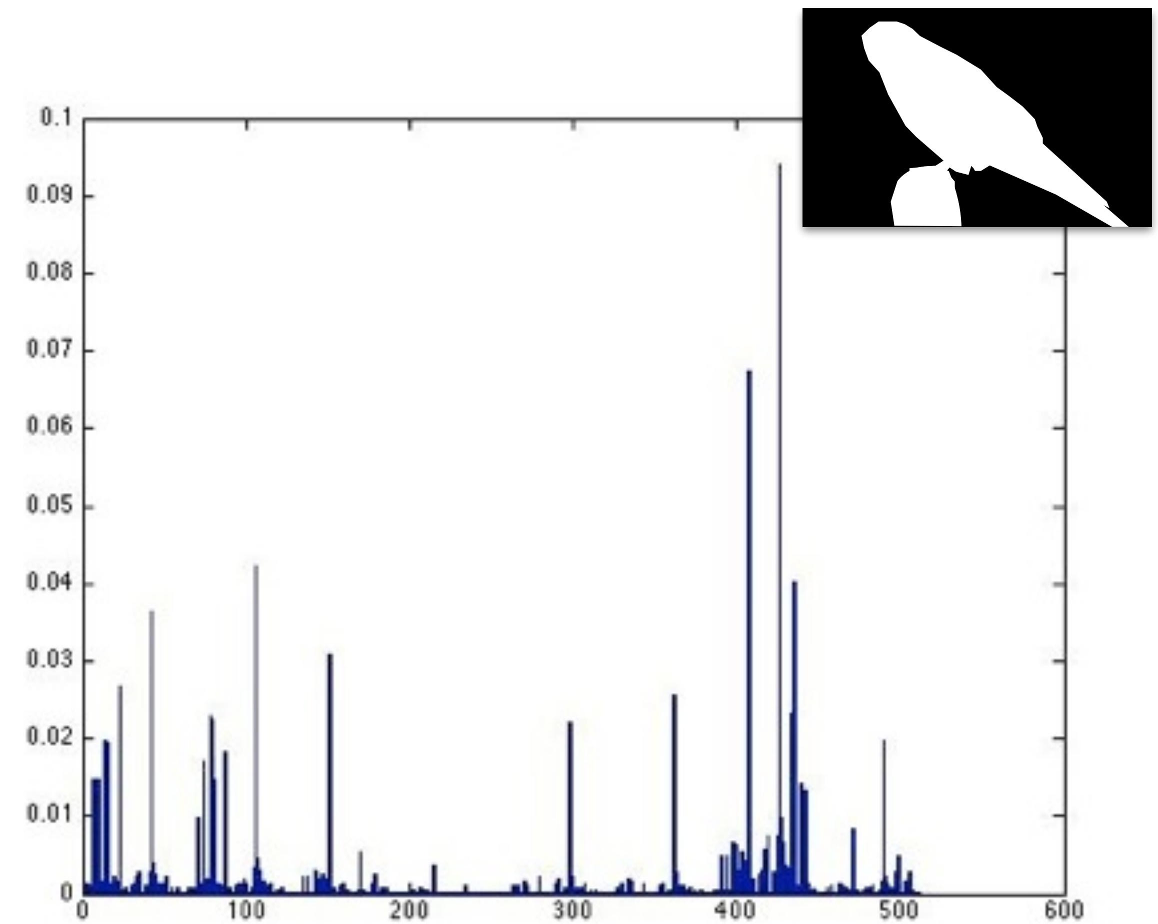
$$\theta_{i,j}(y_i, y_j) = \begin{cases} w_{i,j} & \text{if } y_i = y_j \\ -w_{i,j} & \text{otherwise} \end{cases}$$

$$w_{i,j} \geq 0$$

$$\theta_i(y_i) = \log q(y_i|x_i)$$




- The most likely structure



# Most likely structure

$$y^* = \arg\max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

# Most likely structure

$$y^* = \arg\max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.

# Most likely structure

$$y^* = \arg \max_{y_1, \ldots, y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:

# Most likely structure

$$y^* = \arg\max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),

# Most likely structure

$$y^* = \arg\max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),
  - Graph-cuts for image segmentation

# Most likely structure

$$y^* = \arg \max_{y_1, \ldots, y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),
  - Graph-cuts for image segmentation
  - branch and cut (Gurobi), local consistency (Larrosa 03, de Givry 05), mini-buckets (Dechter 97)

# Most likely structure

$$y^* = \arg \max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),
  - Graph-cuts for image segmentation
  - branch and cut (Gurobi), local consistency (Larrosa 03, de Givry 05), mini-buckets (Dechter 97)
  - Linear programming relaxations (Schlesinger 76, Wainwright 05, Kolmogorov 06, Komodakis 07, Werner 07, Sontag 08, Hazan 10, Kappes 13, Savchynskyy13, Tarlow 13)

# Most likely structure

$$y^* = \arg \max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),
  - Graph-cuts for image segmentation
  - branch and cut (Gurobi), local consistency (Larrosa 03, de Givry 05), mini-buckets (Dechter 97)
  - Linear programming relaxations (Schlesinger 76, Wainwright 05, Kolmogorov 06, Komodakis 07, Werner 07, Sontag 08, Hazan 10, Kappes 13, Savchynskyy13, Tarlow 13)
  - CKY for parsing

# Most likely structure

$$y^* = \arg \max_{y_1,\ldots,y_n} \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)$$

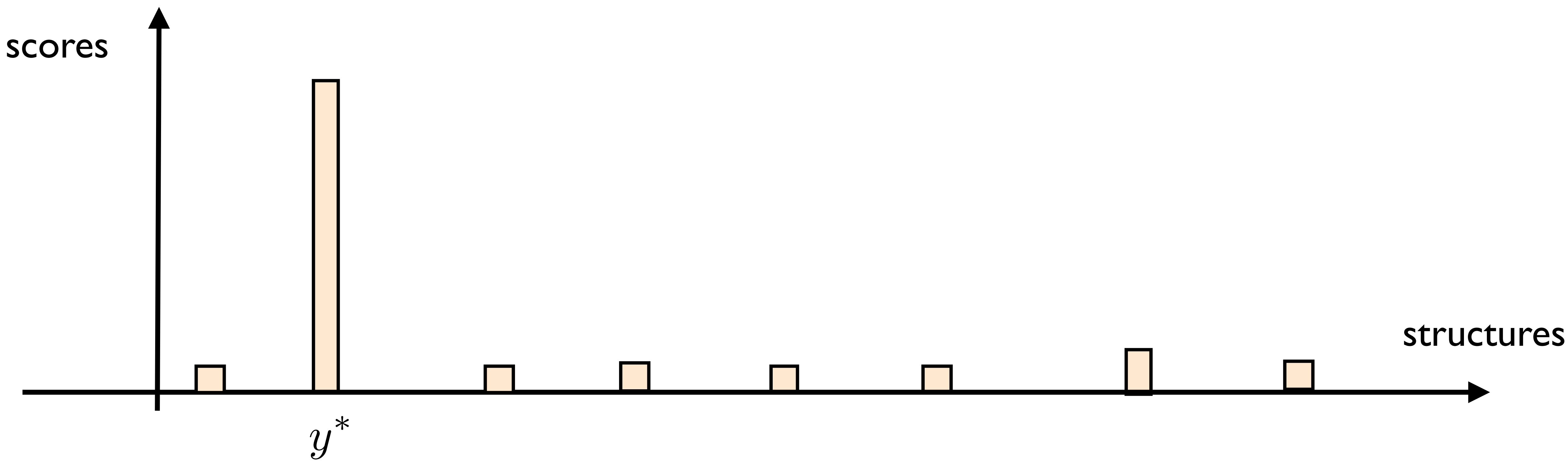
- Maximum a-posteriori (MAP) inference.
- Many efficient optimization algorithms for special cases:
  - Beliefs propagation: trees (Pearl 88), perfect graphs (Jebara 10),
  - Graph-cuts for image segmentation
  - branch and cut (Gurobi), local consistency (Larrosa 03, de Givry 05), mini-buckets (Dechter 97)
  - Linear programming relaxations (Schlesinger 76, Wainwright 05, Kolmogorov 06, Komodakis 07, Werner 07, Sontag 08, Hazan 10, Kappes 13, Savchynskyy13, Tarlow 13)
  - CKY for parsing
  - Many others…

# The challenge

Sampling from the likely high dimensional structures (with millions of variables, e.g., image segmentation with 12 million pixels) as efficient as optimizing

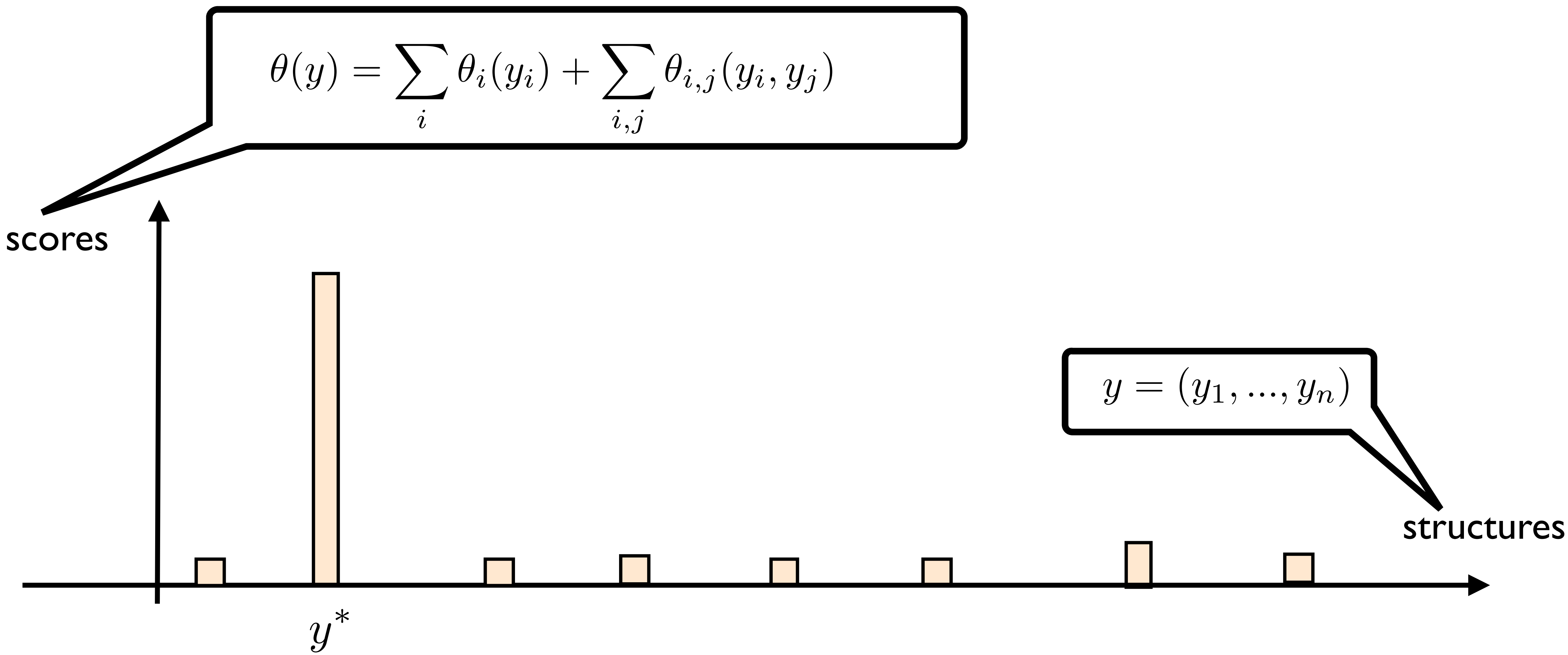
# Most likely structure

- Selecting the maximizing structure is appropriate when one structure (e.g., segmentation / parse) dominates others



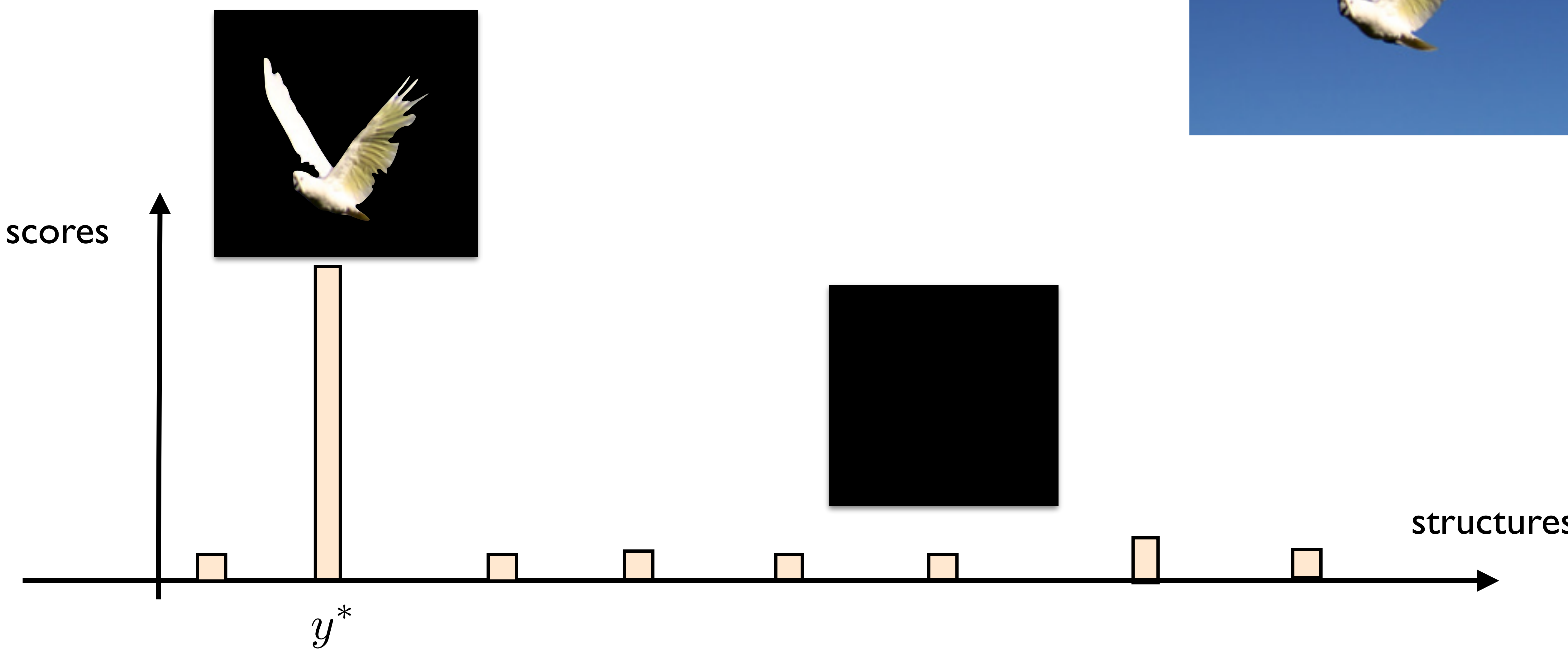
# Most likely structure

- Selecting the maximizing structure is appropriate when one structure (e.g., segmentation / parse) dominates others



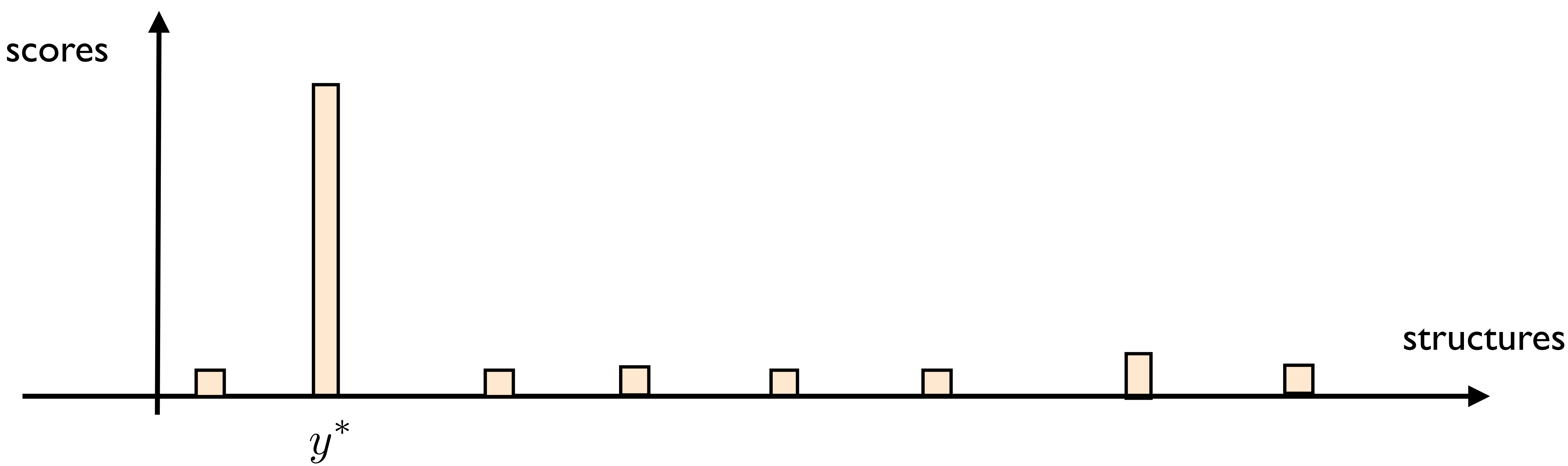
# Most likely structure

- Selecting the maximizing structure is appropriate when one structure (e.g., segmentation / parse) dominates others



# Most likely structure

- Selecting the maximizing structure is appropriate when one structure (e.g., segmentation / parse) dominates others

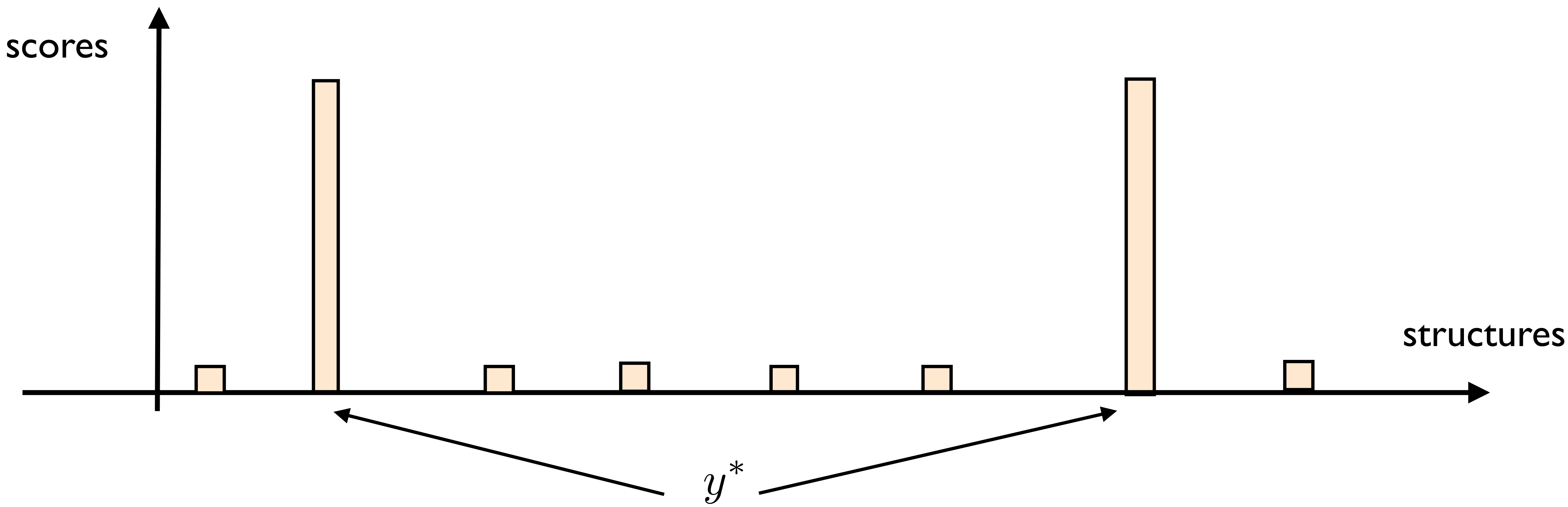


# Most likely structure

- The maximizing structure is not robust in case of multiple high scoring alternatives

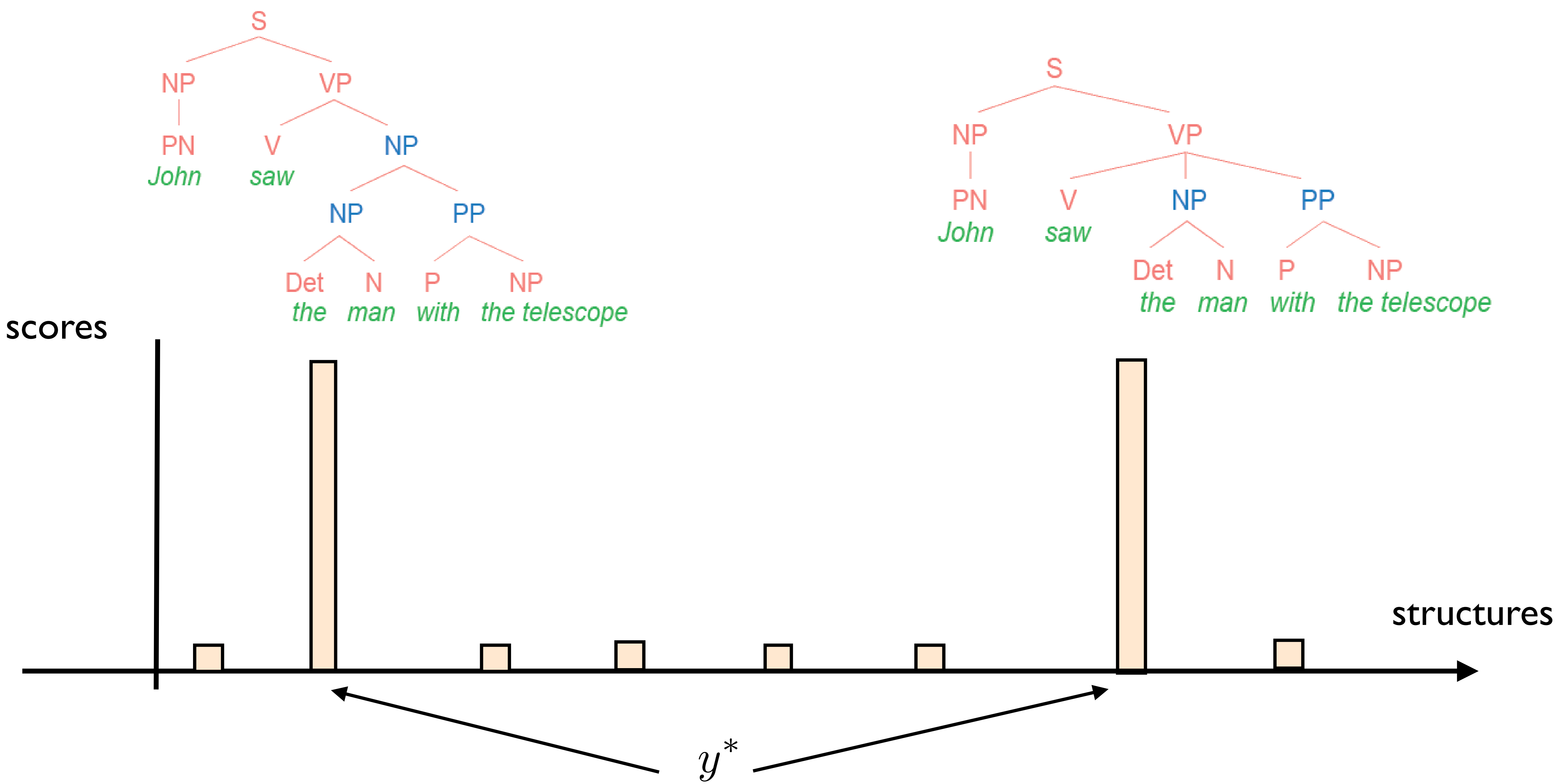
# Most likely structure

- The maximizing structure is not robust in case of multiple high scoring alternatives



# Most likely structure

- The maximizing structure is not robust in case of multiple high scoring alternatives

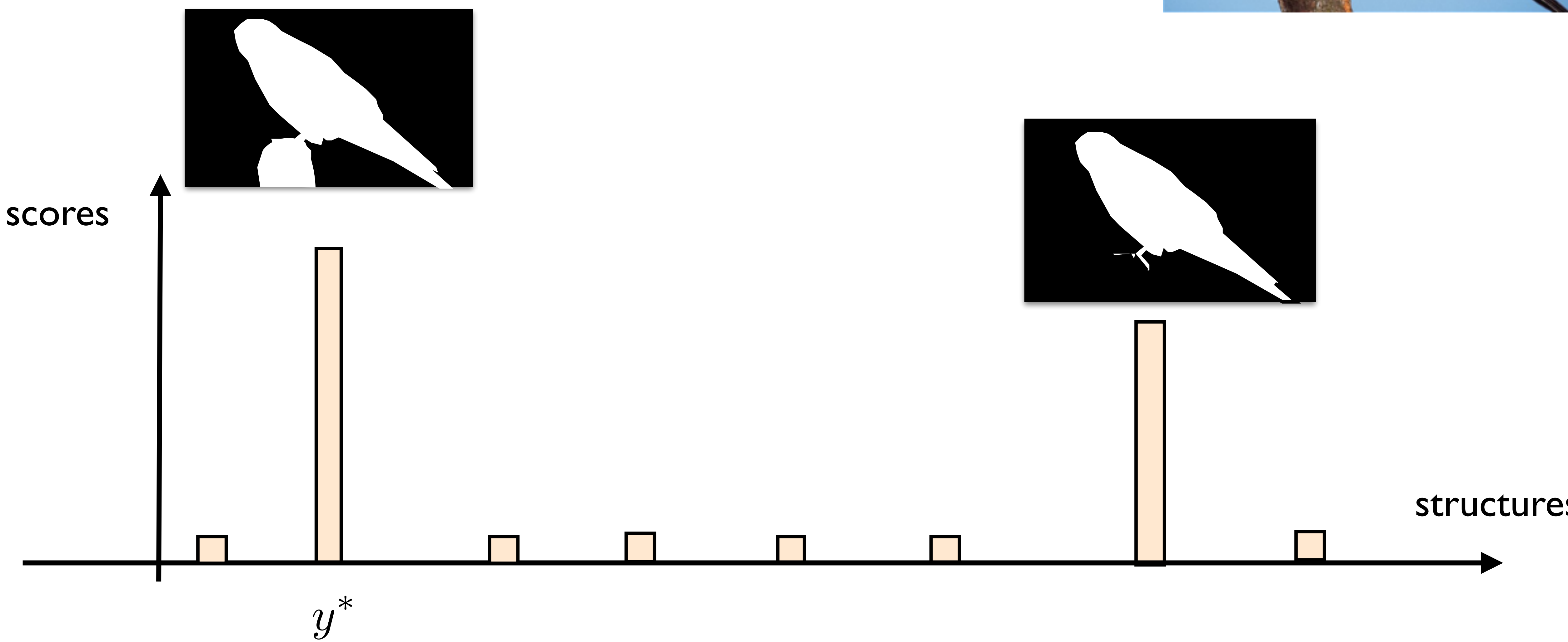


# Most likely structure

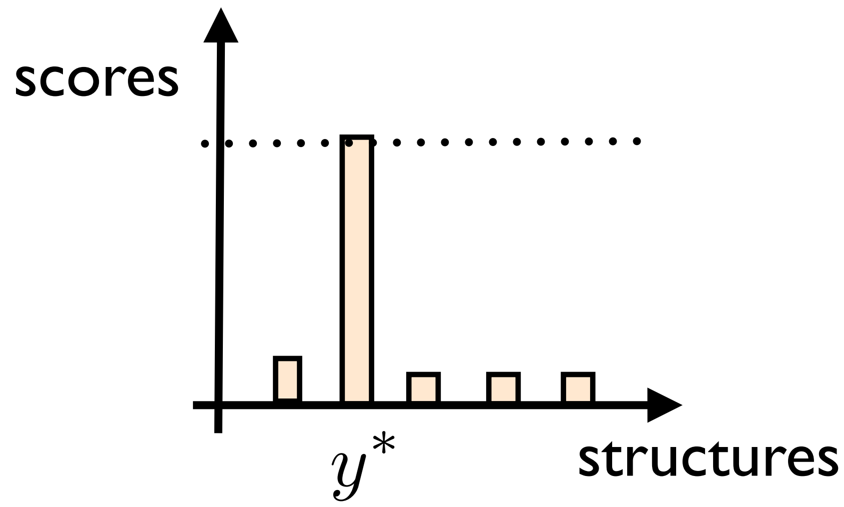
- The maximizing structure is not robust in case of multiple high scoring alternatives

# Most likely structure

- The maximizing structure is not robust in case of multiple high scoring alternatives

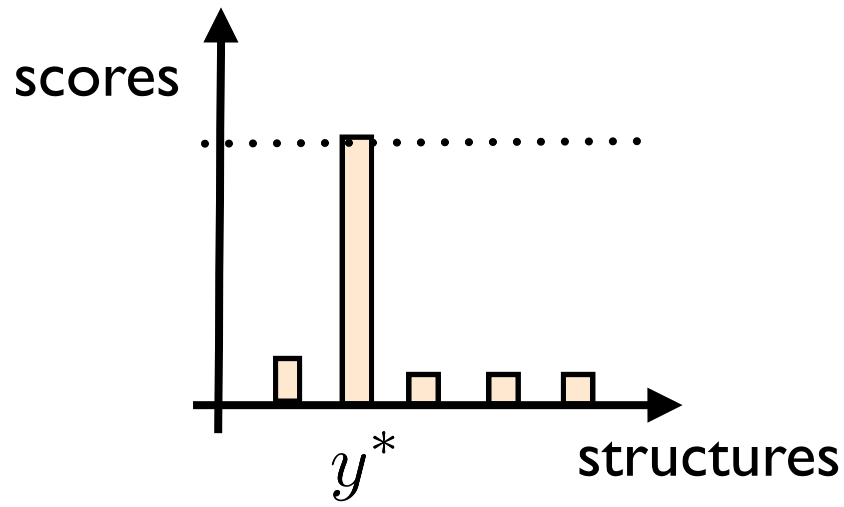


# Random perturbations



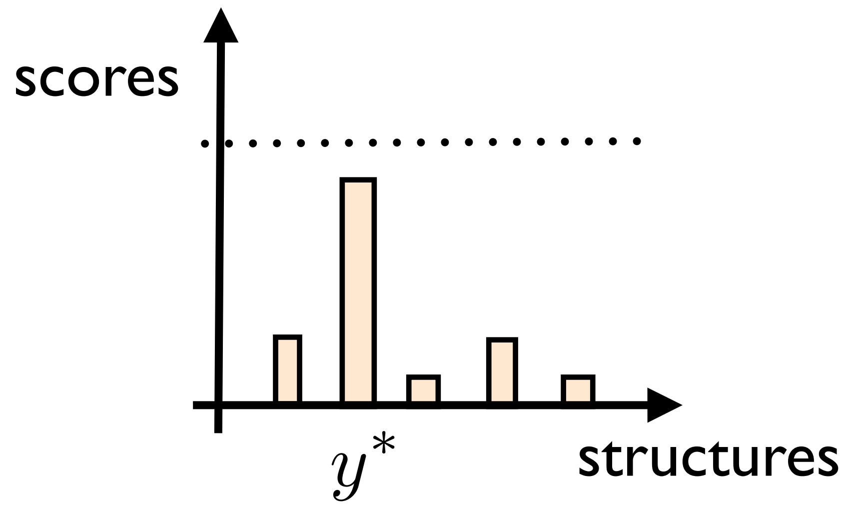

- Randomly perturbing the system reveals its complexity

# Random perturbations



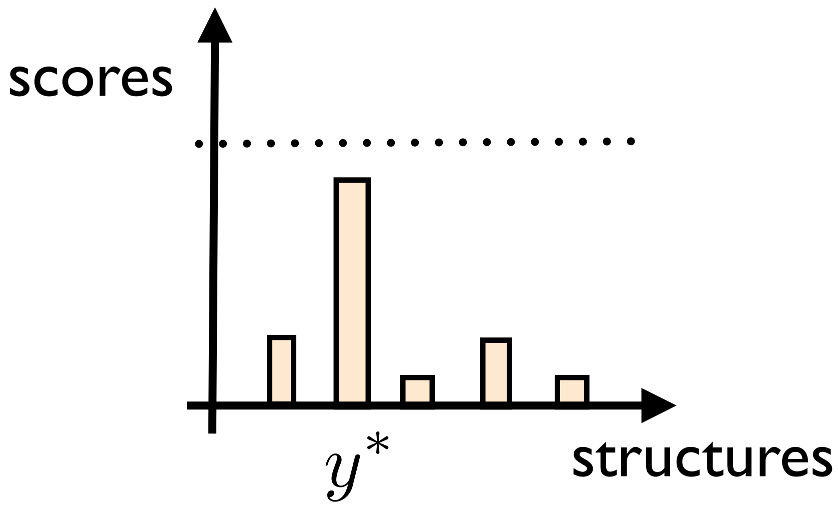

- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"

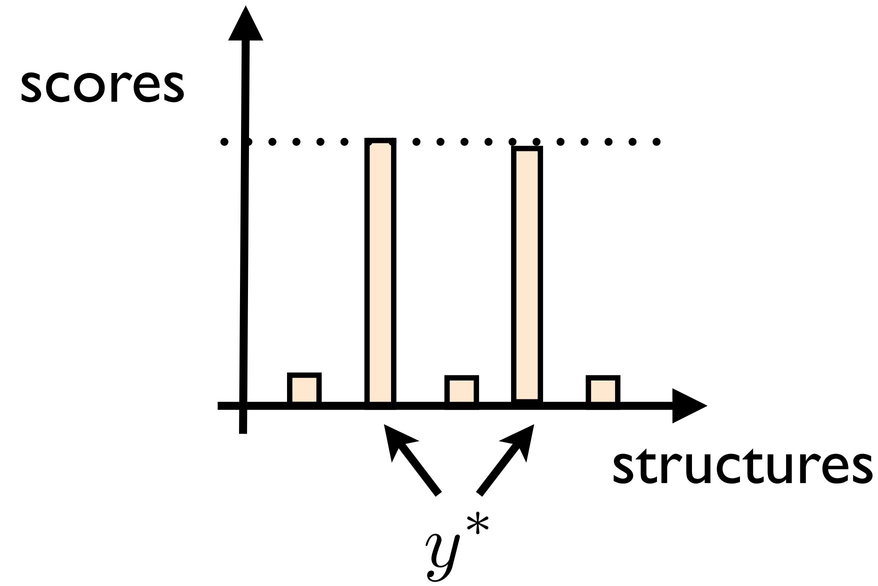
# Random perturbations




- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"

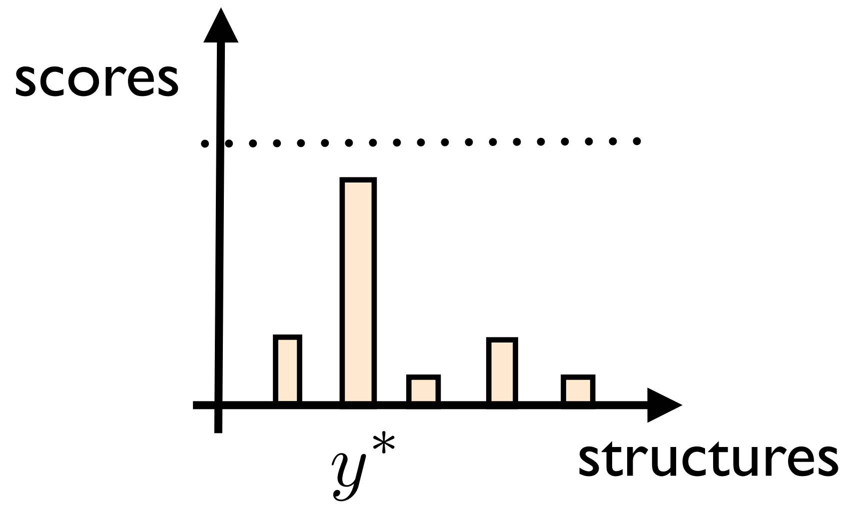
# Random perturbations

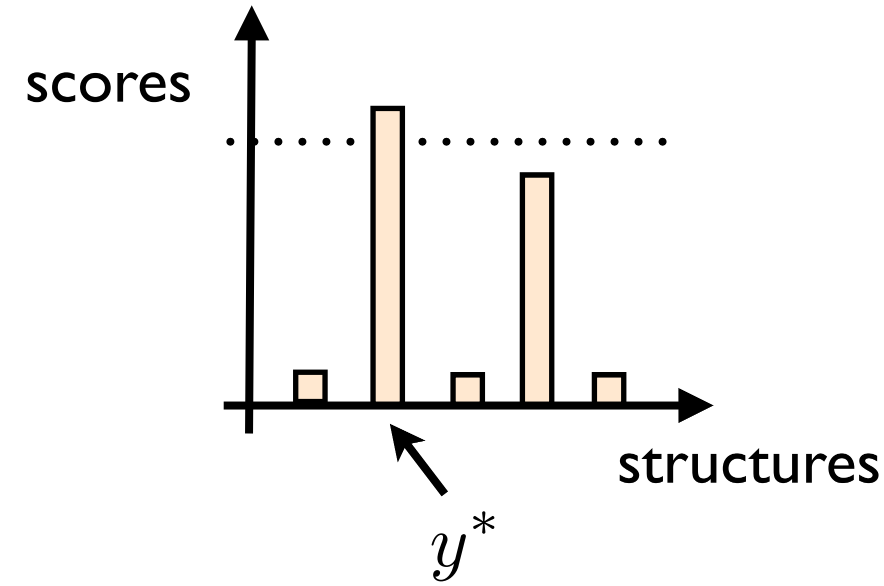



- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"
  - substantial effect when there are alternative high scoring structures

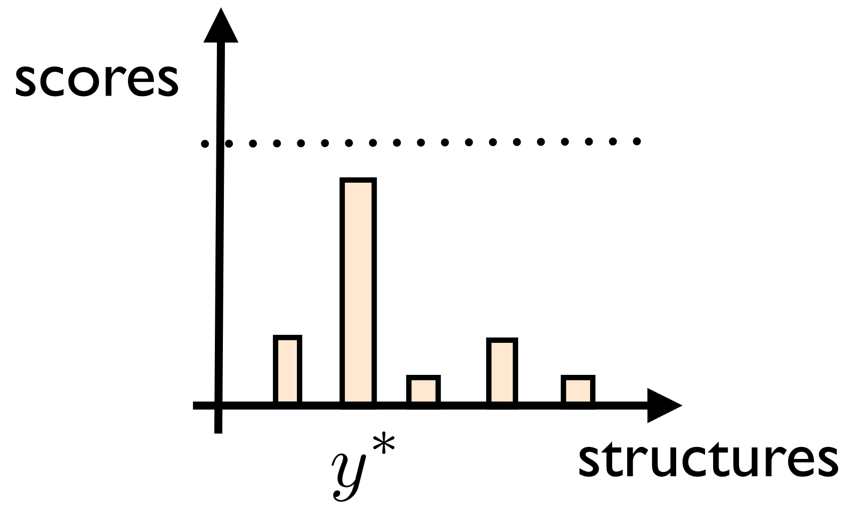
# Random perturbations

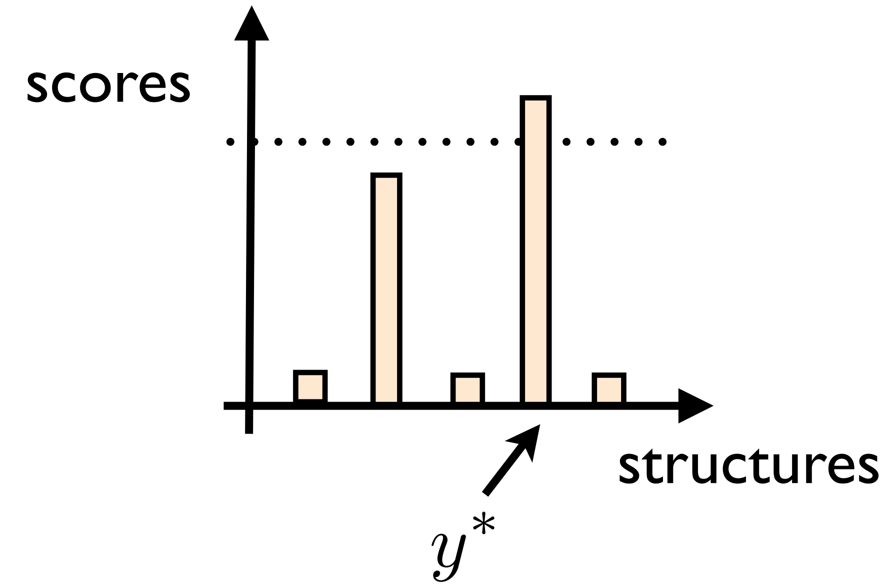



- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"
  - substantial effect when there are alternative high scoring structures

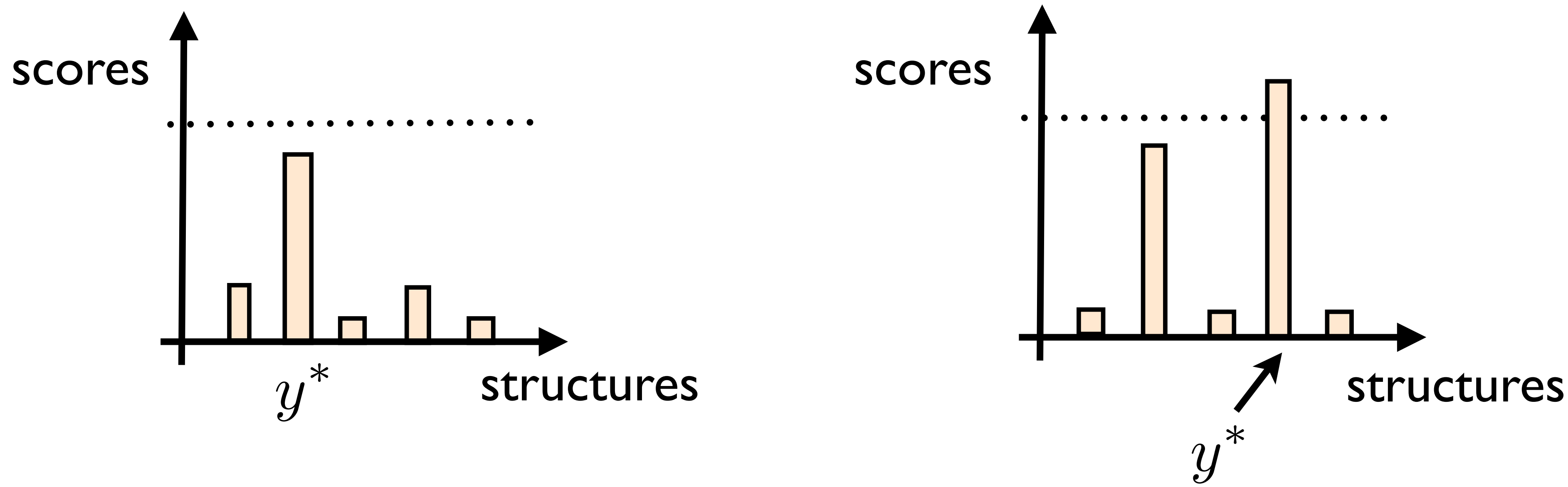
# Random perturbations




- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"
  - substantial effect when there are alternative high scoring structures

# Random perturbations



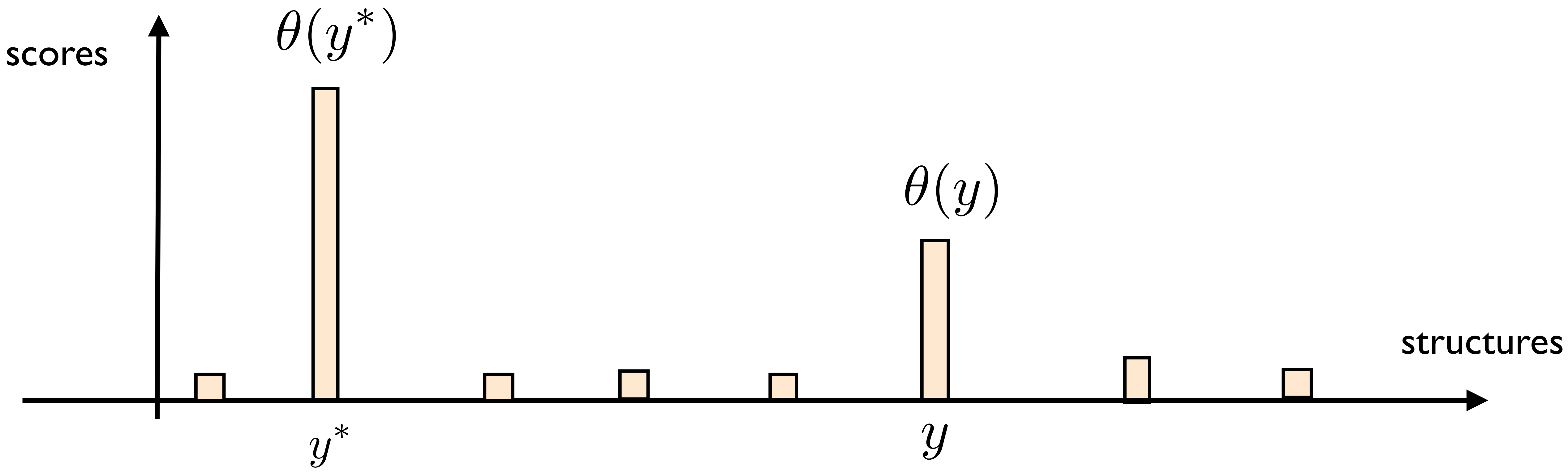

- Randomly perturbing the system reveals its complexity
  - little effect when the maximizing structure is "evident"
  - substantial effect when there are alternative high scoring structures

- Related work:
  - McFadden 74 (Discrete choice theory)
  - Talagrand 94 (Canonical processes)

# Random perturbations

- Notation:

scores (potential) $\theta(y)$

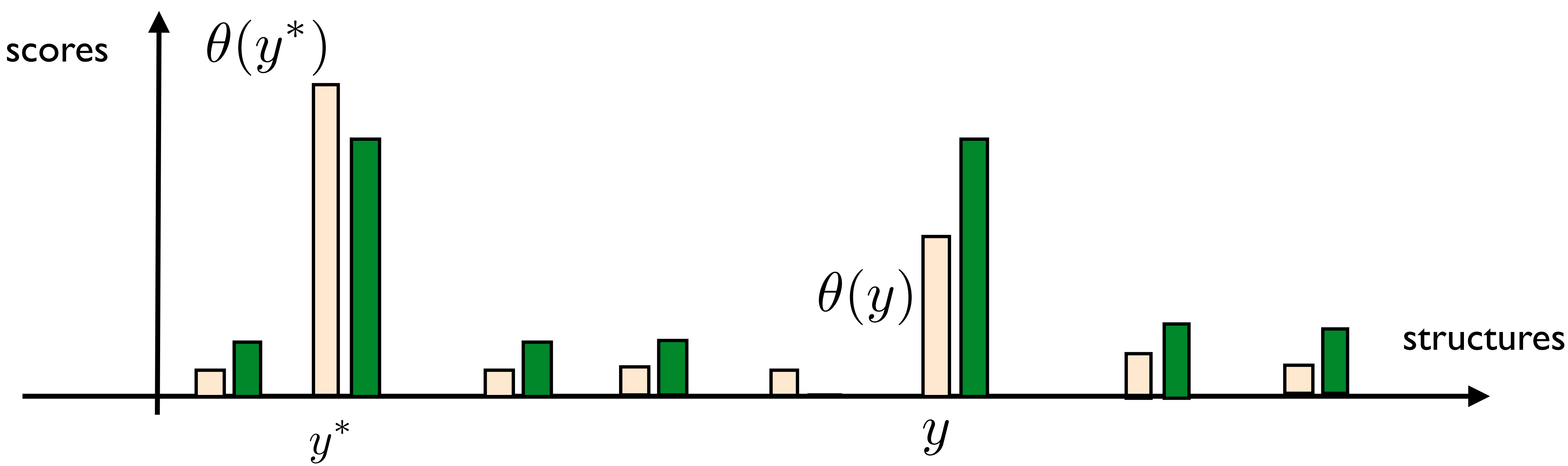


# Random perturbations

- Notation:

scores (potential) $\theta(y)$

perturbed score



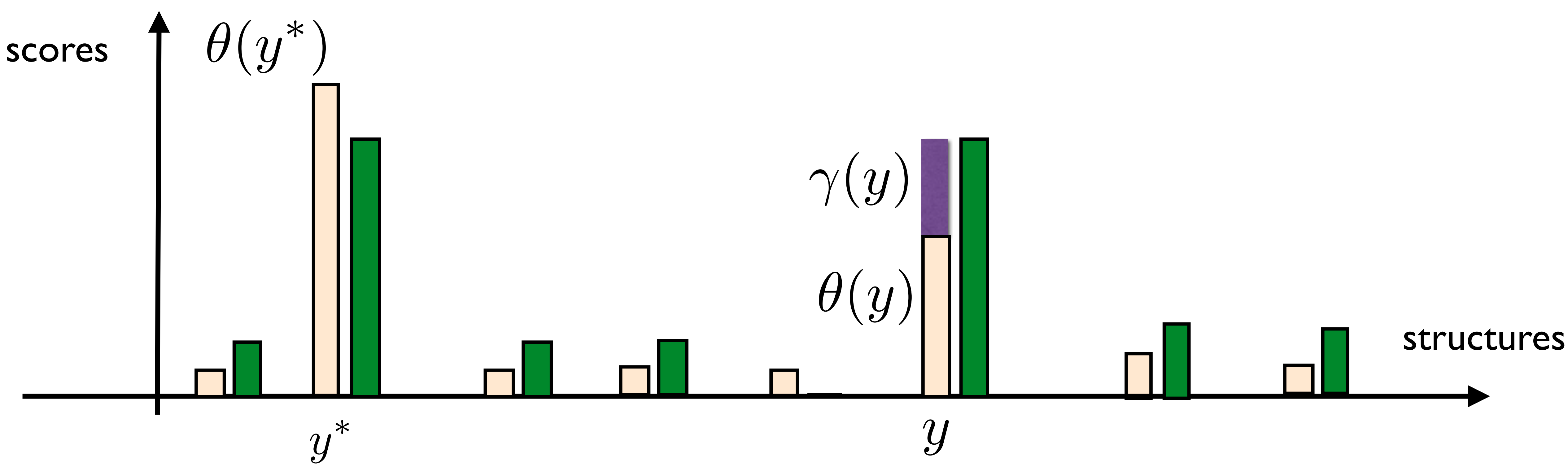
# Random perturbations

- Notation:

scores (potential) $\theta(y)$

perturbed score

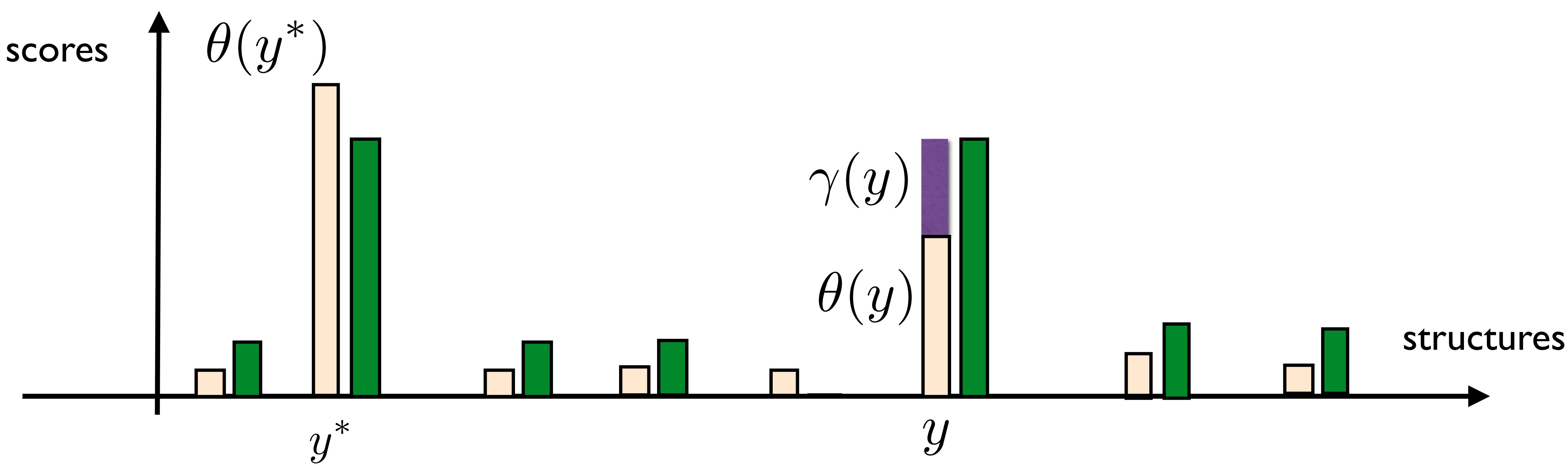
perturbations $\gamma(y)$



# Random perturbations

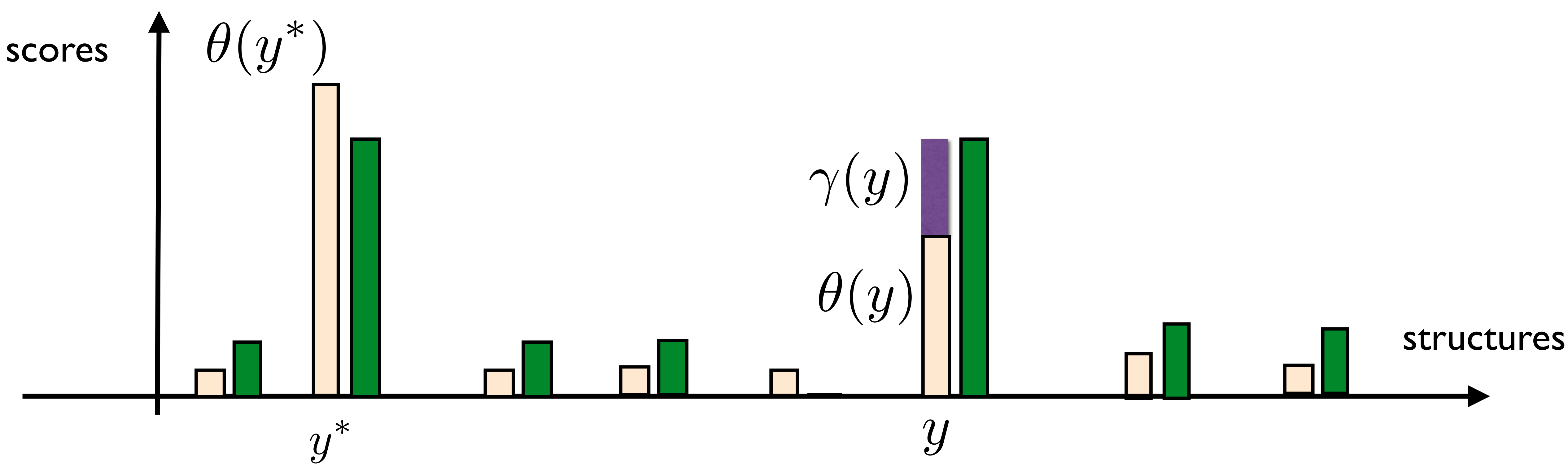
- Notation:

| | | |
|---|---|---|
| | scores (potential) | $\theta(y)$ |
| | perturbed score | $\theta(y) + \gamma(y)$ |
| | perturbations | $\gamma(y)$ |



# Random perturbations

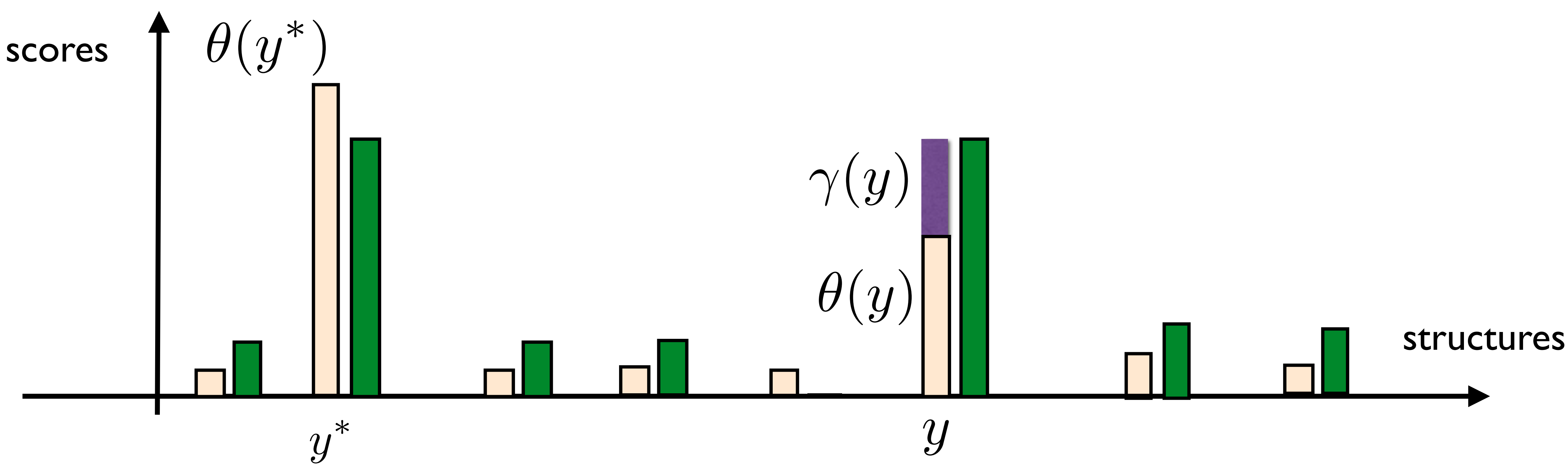
- For every structure y, the perturbation value $\gamma(y)$ is a random variable (y is an index, traditional notation is $\gamma_y$).



# Random perturbations

- For every structure y, the perturbation value $\gamma(y)$ is a random variable (y is an index, traditional notation is $\gamma_y$).

- Perturb-max models: how stable is the maximal structure to random changes in the potential function.



# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# Perturb-max models

- **Theorem**

Let $\gamma(y)$ be i.i.d. with Gumbel distribution with zero mean

$$F(t) \stackrel{def}{=} P[\gamma(y) \leq t] = \exp(-\exp(-t))$$

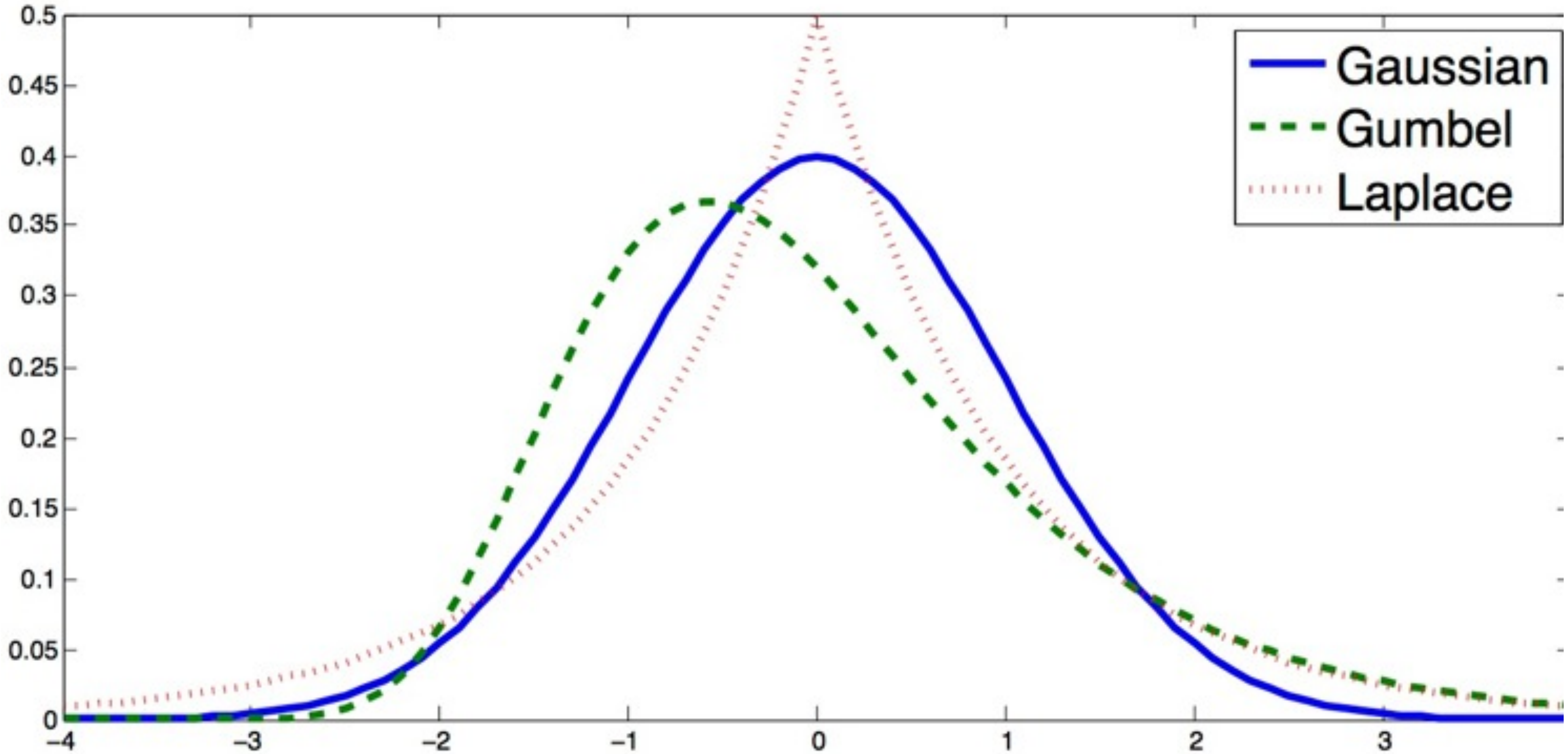
# Perturb-max models

- **Theorem**

Let $\gamma(y)$ be i.i.d. with Gumbel distribution with zero mean

$$F(t) \stackrel{def}{=} P[\gamma(y) \leq t] = \exp(-\exp(-t))$$

$$f(t) = F'(t) = \exp(-t)F(t)$$



# Perturb-max models

- **Theorem**

Let $\gamma(y)$ be i.i.d. with Gumbel distribution with zero mean

$$F(t) \stackrel{def}{=} P[\gamma(y) \leq t] = \exp(-\exp(-t))$$

# Perturb-max models

- **Theorem**

Let $\gamma(y)$ be i.i.d. with Gumbel distribution with zero mean

$$F(t) \stackrel{def}{=} P[\gamma(y) \leq t] = \exp(-\exp(-t))$$

then the perturb-max model is the Gibbs distribution

$$\frac{1}{Z}\exp(\theta(y)) = P_{\gamma \sim Gumbel}[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

Let $\gamma(y)$ be i.i.d Gumbel ( $P[\gamma(y) \leq t] = F(t)$ ). Then

$$\max_y \{\theta(y) + \gamma(y)\}$$

has Gumbel distribution whose mean is $\log Z$

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

Let $\gamma(y)$ be i.i.d Gumbel ( $P[\gamma(y) \le t] = F(t)$ ). Then

$$\max_y \{\theta(y) + \gamma(y)\}$$

$$\boxed{Z = \sum_y \exp(\theta(y))}$$

mean is $\log Z$

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

Let $\gamma(y)$ be i.i.d Gumbel ( $P[\gamma(y) \leq t] = F(t)$ ). Then

$$\max_y \{\theta(y) + \gamma(y)\}$$

has Gumbel distribution whose mean is $\log Z$

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

Let $\gamma(y)$ be i.i.d Gumbel ( $P[\gamma(y) \leq t] = F(t)$ ). Then

$$\max_y \{\theta(y) + \gamma(y)\}$$

has Gumbel distribution whose mean is $\log Z$

- **Proof:** $P_\gamma[\max_y \{\theta(y) + \gamma(y)\} \leq t] = \prod_y F(t - \theta(y))$

# Perturb-max models

- Why Gumbel distribution? $F(t) = \exp(-\exp(-t))$
- Since maximum of Gumbel variables is a Gumbel variable.

Let $\gamma(y)$ be i.i.d Gumbel ( $P[\gamma(y) \leq t] = F(t)$ ). Then

$$\max_y \{\theta(y) + \gamma(y)\}$$

has Gumbel distribution whose mean is $\log Z$

- **Proof:** $P_\gamma[\max_y \{\theta(y) + \gamma(y)\} \leq t] = \prod_y F(t - \theta(y))$

$$= \exp(-\sum_y \exp(-(t - \theta(y)))) = F(t - \log Z)$$

# Perturb-max models

- Max stability:

$$\log\Big(\sum_y \exp(\theta(y))\Big) = E_{\gamma \sim Gumbel}\Big[\max_y\{\theta(y) + \gamma(y)\}\Big]$$

- Implications (taking gradients):

# Perturb-max models

- Max stability:

$$\log\Big(\sum_y \exp(\theta(y))\Big) = E_{\gamma\sim Gumbel}\Big[\max_y\{\theta(y)+\gamma(y)\}\Big]$$

- Implications (taking gradients):

$$\frac{1}{Z}\exp(\theta(y)) = P_{\gamma\sim Gumbel}[y = \arg\max_{\hat{y}}\{\theta(\hat{y})+\gamma(\hat{y})\}]$$

# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

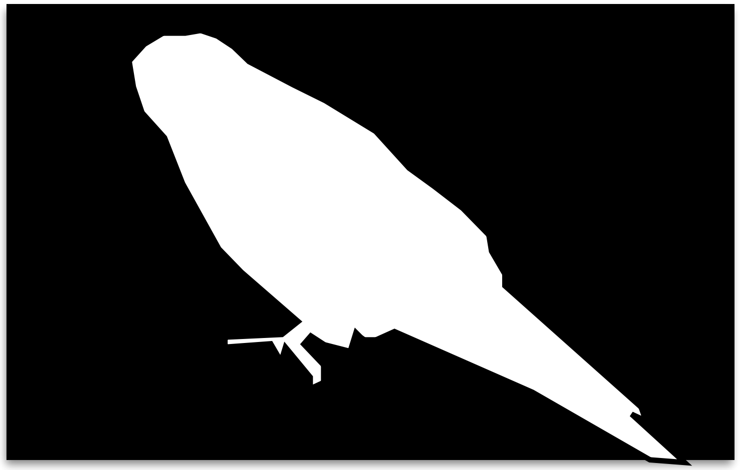
$$y = (y_1, ..., y_n)$$

# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

$$y = (y_1, ..., y_n)$$



# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

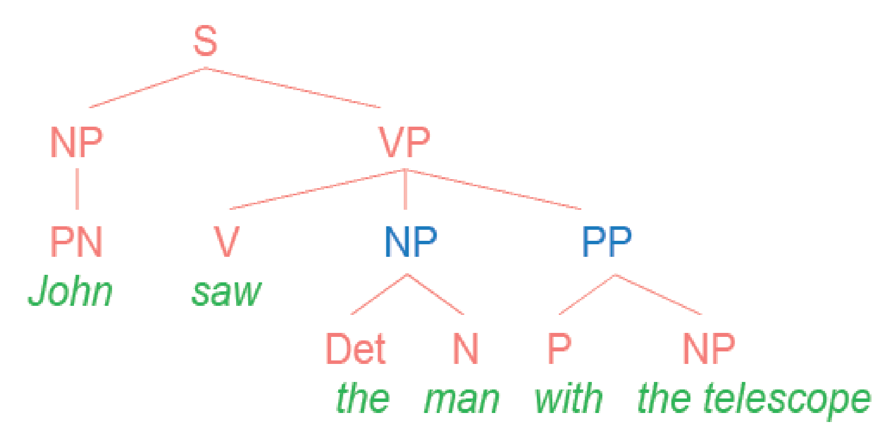
$$y = (y_1, ..., y_n)$$

# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

$$y = (y_1, ..., y_n)$$



# Perturb-max models

- Representing the Gibbs distribution using perturb-max models may require exponential number of perturbations

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \gamma(\hat{y})\}]$$

- Use low dimension perturbations (Papandreou & Yuille 11, Tarlow et al. 12, Hazan & Jaakkola 12)

$$P_\gamma[y = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \sum_{i=1}^{n} \gamma_i(\hat{y}_i)\}]$$

# Outline

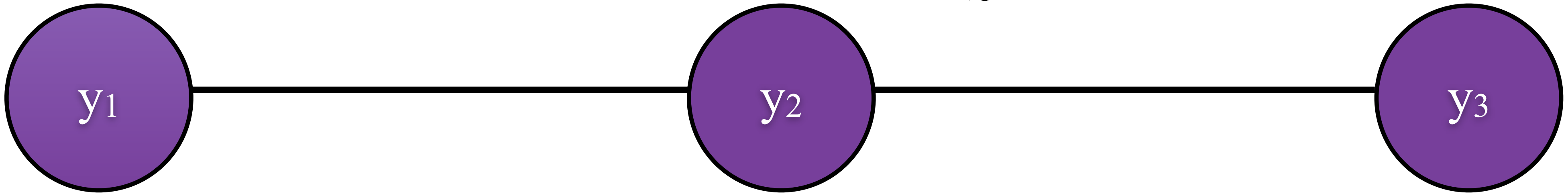
- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# The marginal polytope

$$\theta(y_1, ..., y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$

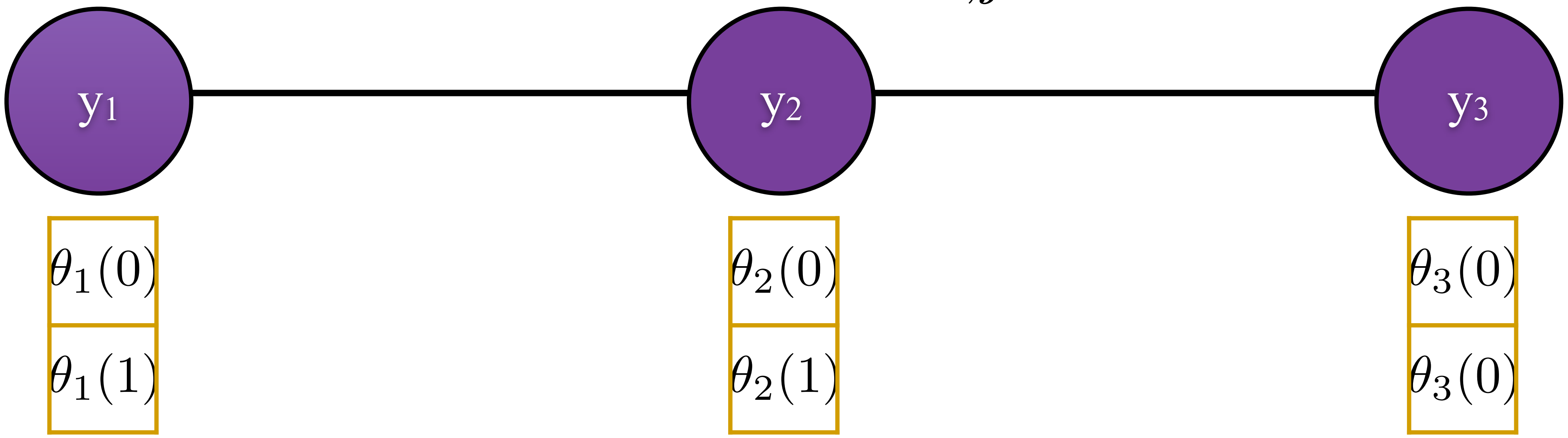
# The marginal polytope

$$\theta(y_1, \ldots, y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$



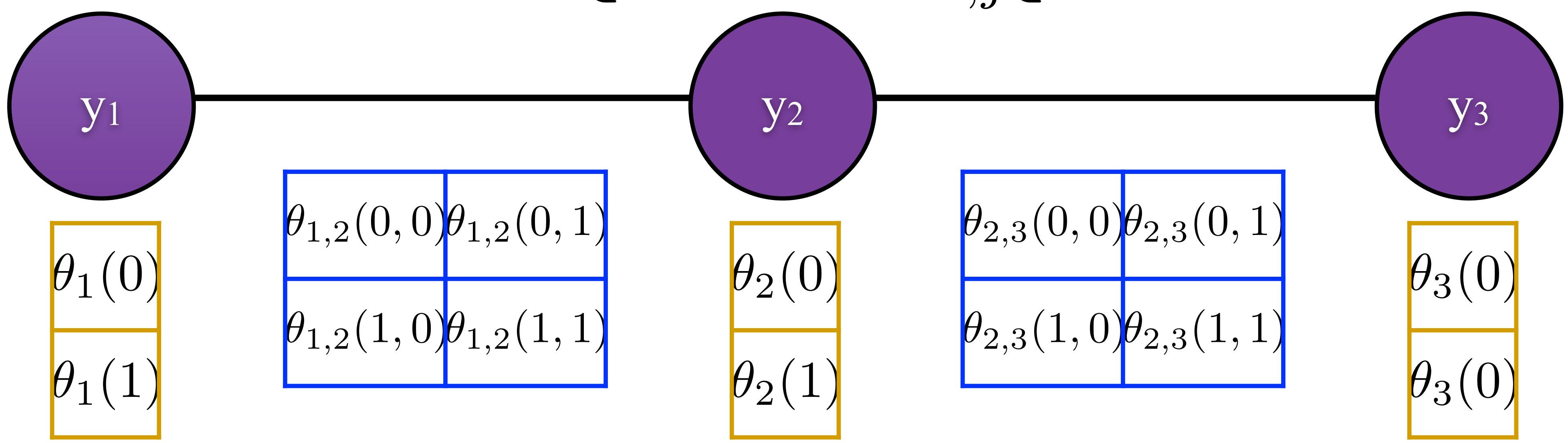
# The marginal polytope

$$\theta(y_1, ..., y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$



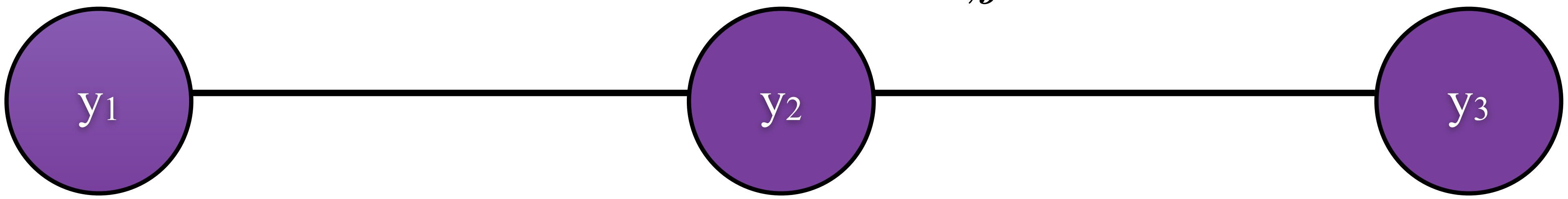
# The marginal polytope

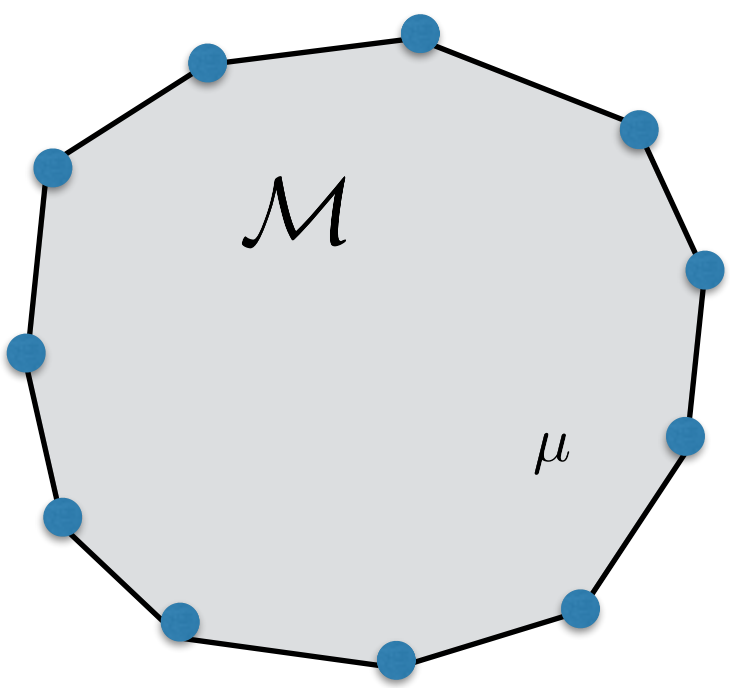
$$\theta(y_1, \dots, y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$



# The marginal polytope

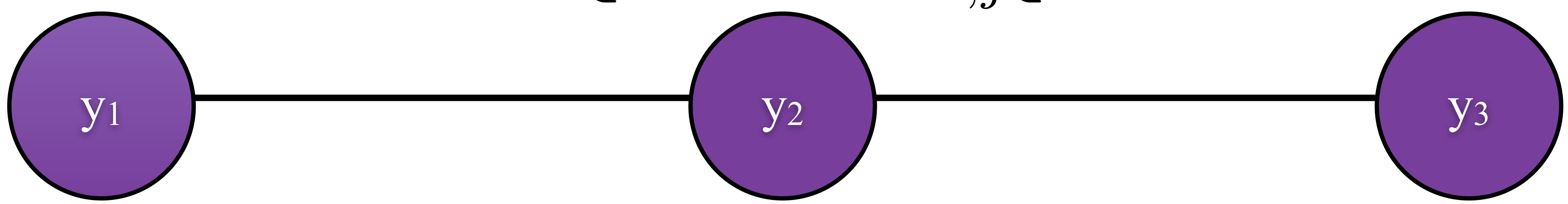
$$\theta(y_1, ..., y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$

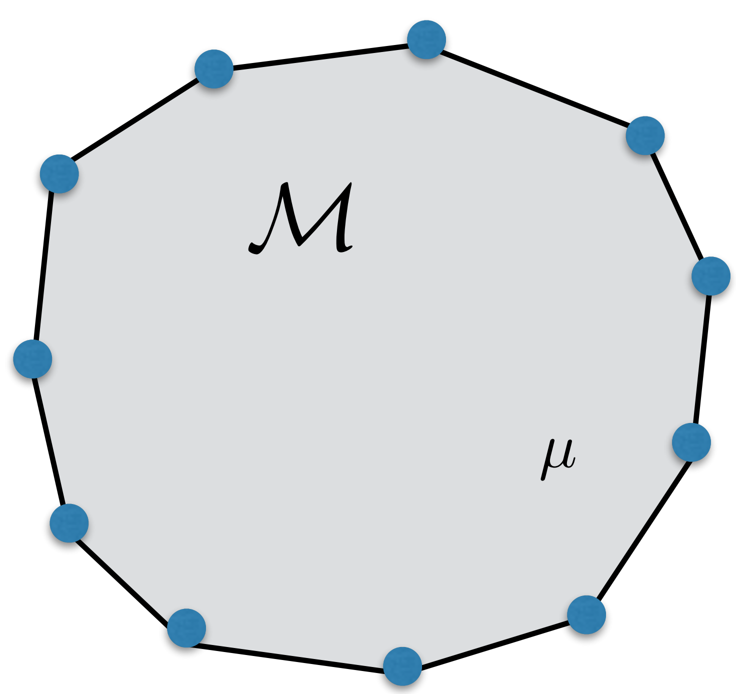


# The marginal polytope

$$\theta(y_1, \ldots, y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$

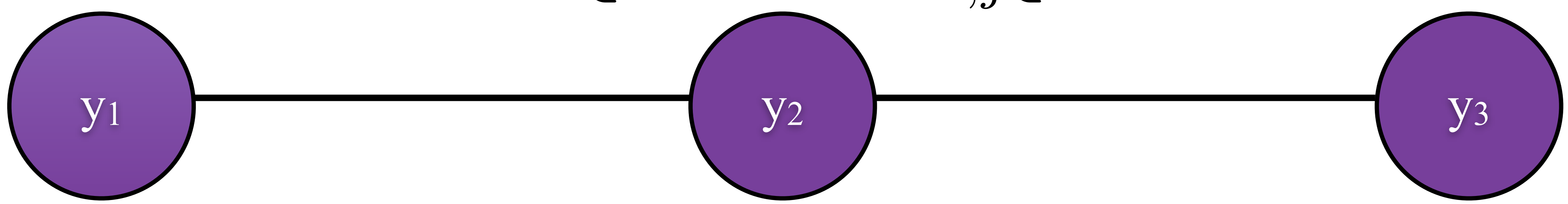


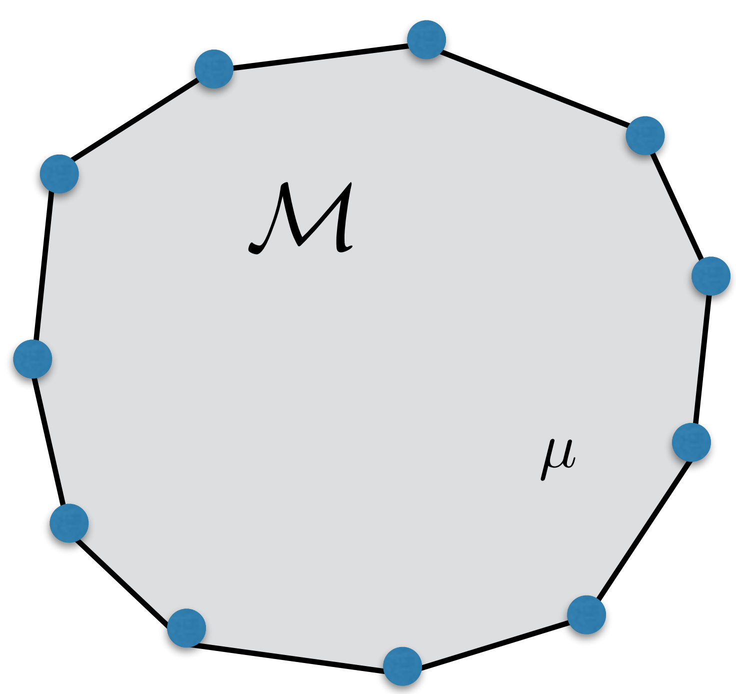




$$\mu = \begin{pmatrix} \mu_1(0), \mu_1(1), \mu_2(0), \mu_2(1), \mu_3(0), \mu_3(1), \\ \mu_{1,2}(0,0), \mu_{1,2}(0,1), \mu_{1,2}(1,0), \mu_{1,2}(1,1), \\ \mu_{2,3}(0,0), \mu_{2,3}(0,1), \mu_{2,3}(1,0), \mu_{2,3}(1,1)) \end{pmatrix}$$

# The marginal polytope

$$\theta(y_1, \ldots, y_n) = \sum_{i \in V} \theta_i(y_i) + \sum_{i,j \in E} \theta_{i,j}(y_i, y_j)$$



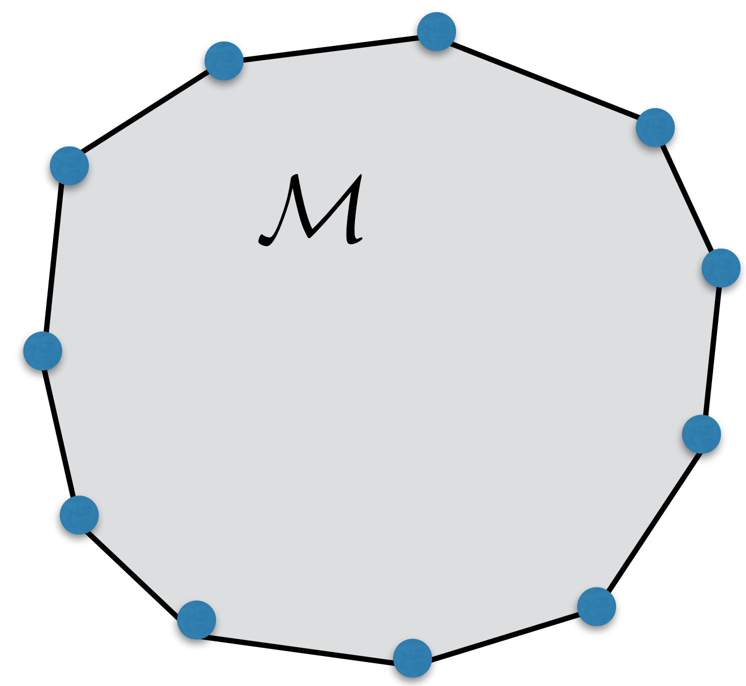




$$\mu = \begin{pmatrix} \mu_1(0), \mu_1(1), \mu_2(0), \mu_2(1), \mu_3(0), \mu_3(1), \\ \mu_{1,2}(0,0), \mu_{1,2}(0,1), \mu_{1,2}(1,0), \mu_{1,2}(1,1), \\ \mu_{2,3}(0,0), \mu_{2,3}(0,1), \mu_{2,3}(1,0), \mu_{2,3}(1,1)) \end{pmatrix}$$

$$\exists p(y_1, y_2, y_3) \text{ s.t. } \quad \mu_1(y_1) = \sum_{y_2, y_3} p(y_1, y_2, y_3), \ldots$$

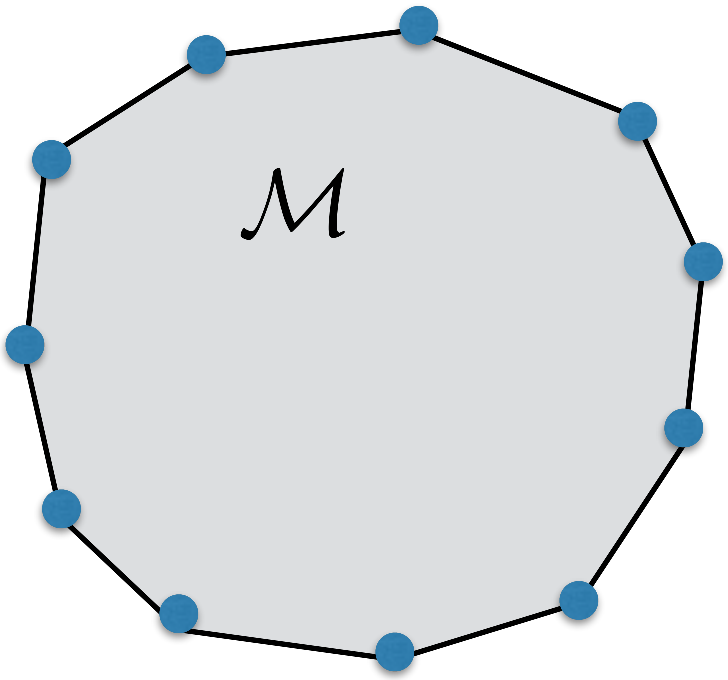
$$\mu_{1,2}(y_1, y_2) = \sum_{y_3} p(y_1, y_2, y_3), \ldots$$

# The marginal polytope



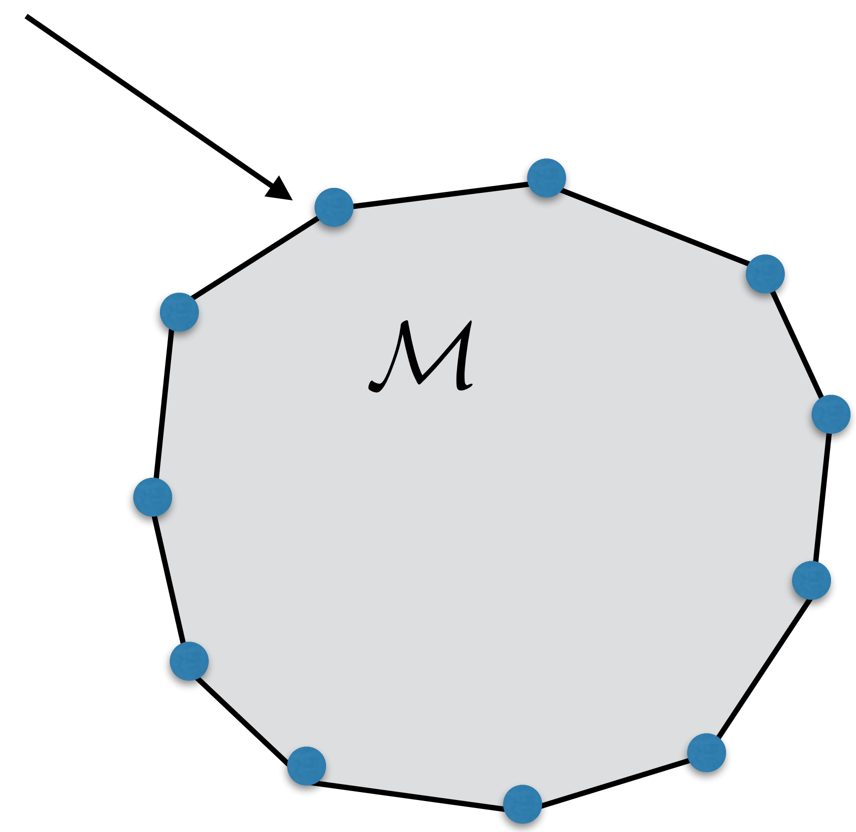
# The marginal polytope

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$



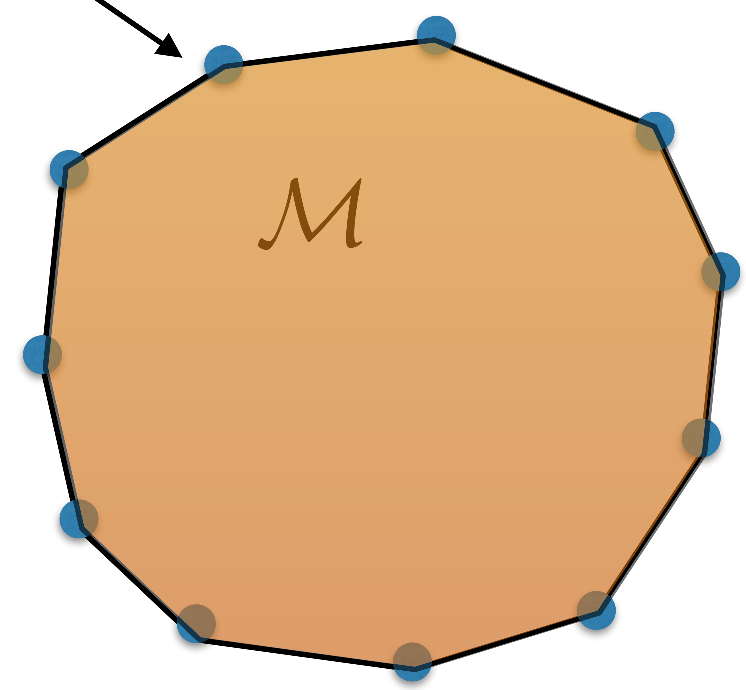
# The marginal polytope

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$



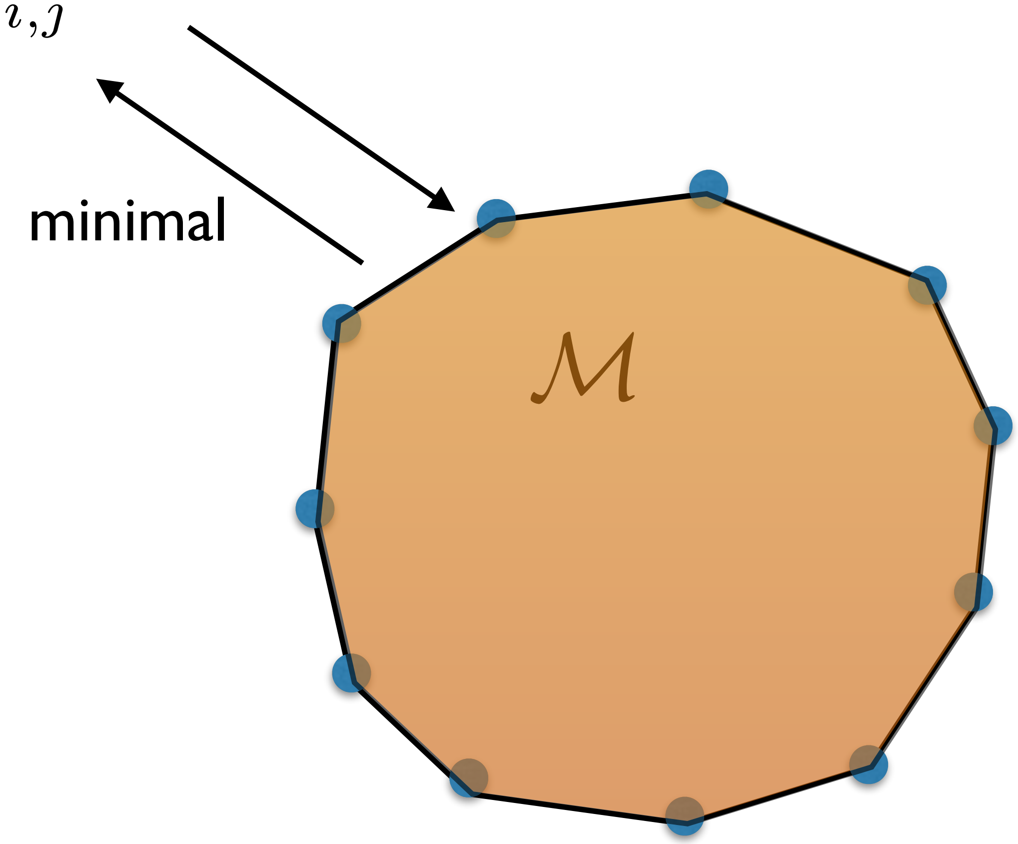
# The marginal polytope

$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$



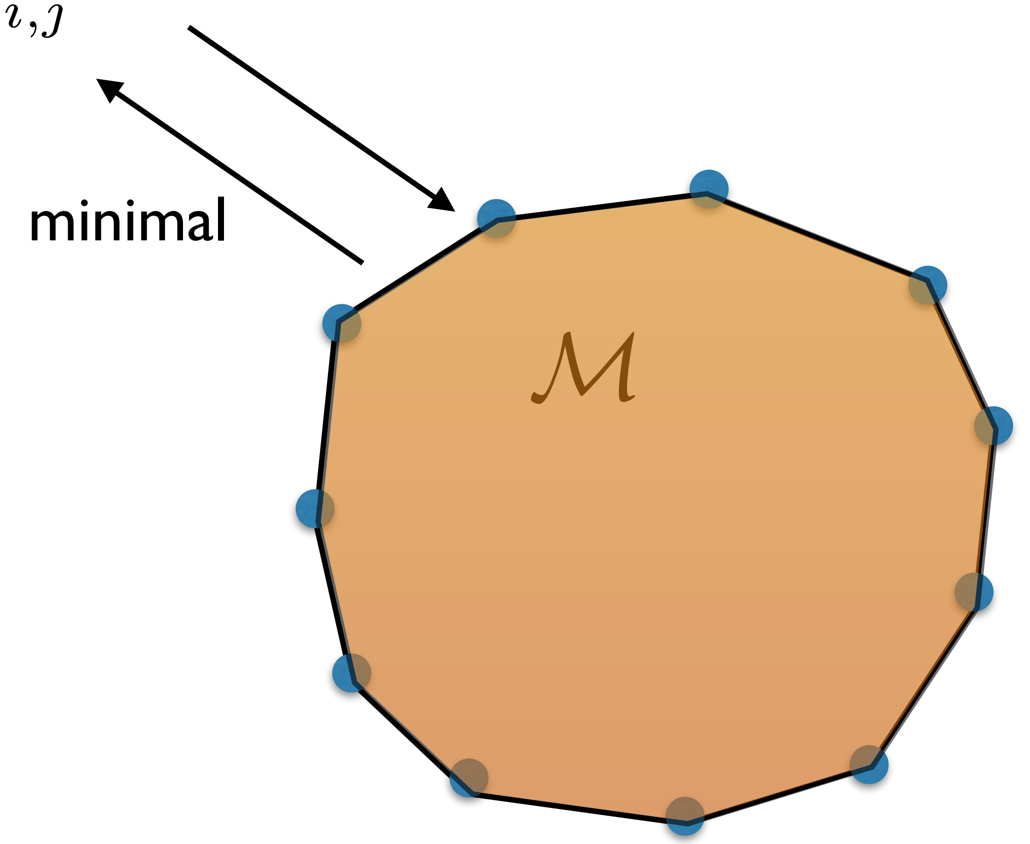
# The marginal polytope

$$p(y) \propto \exp\Big( \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) \Big)$$



# The marginal polytope

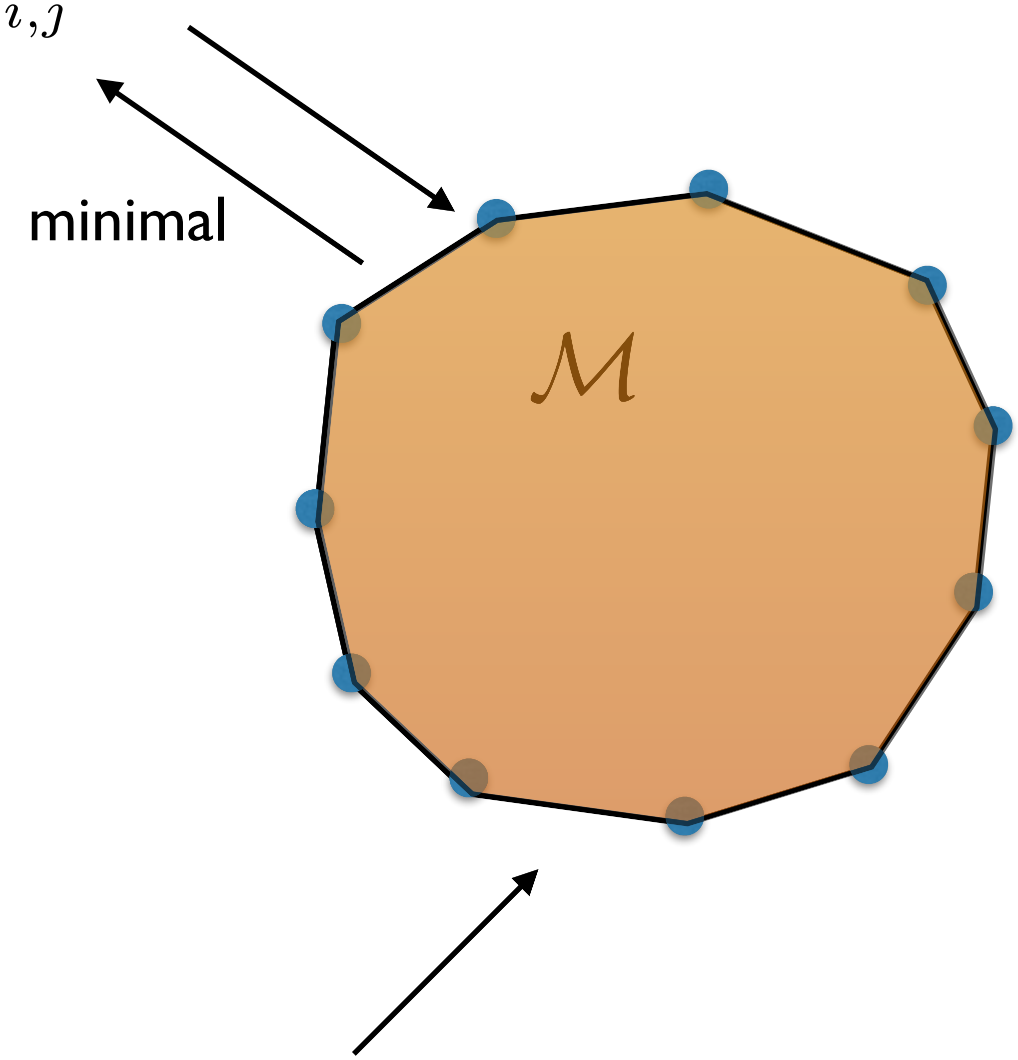
$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$




$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\}\Big]$$

# The marginal polytope

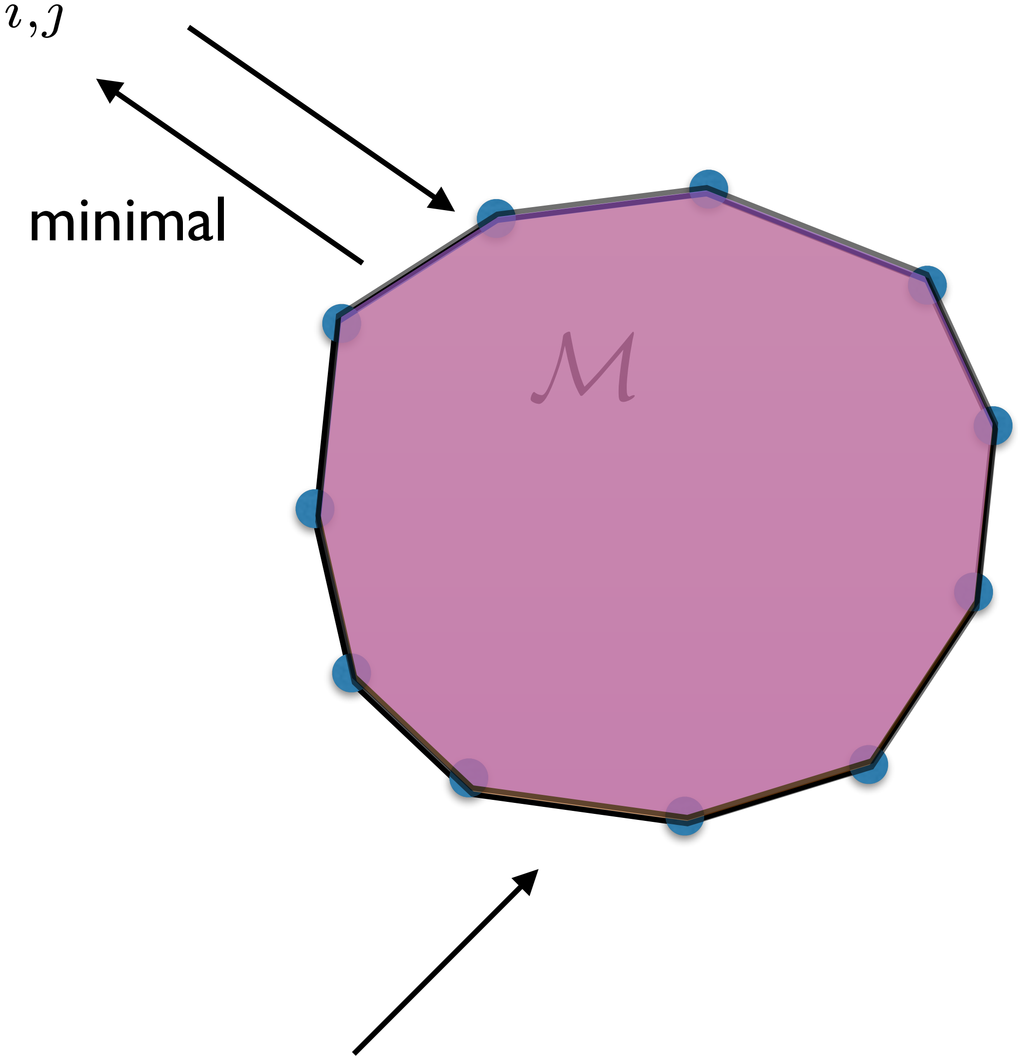
$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$




$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i)\Big\}\Big]$$

# The marginal polytope

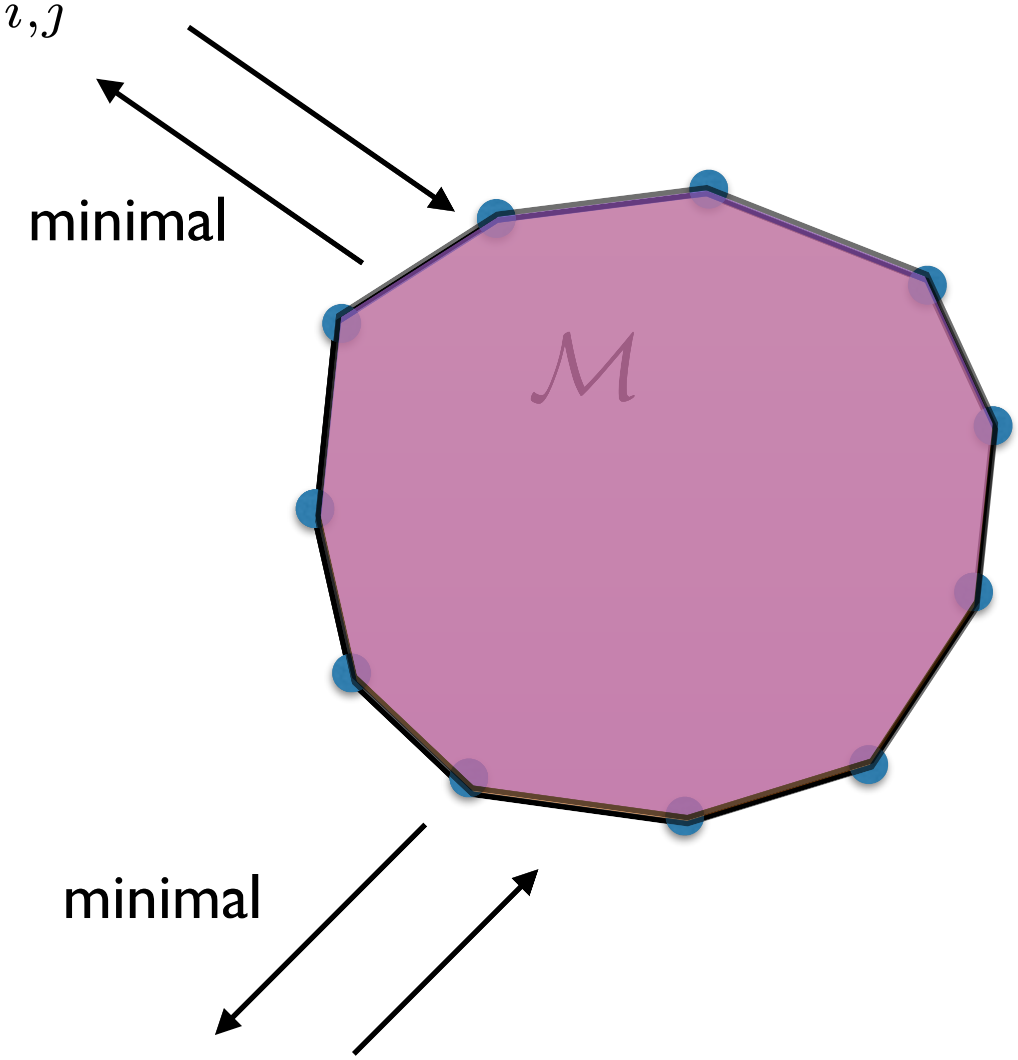
$$p(y) \propto \exp\Big(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\Big)$$




$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i)\Big\}\Big]$$

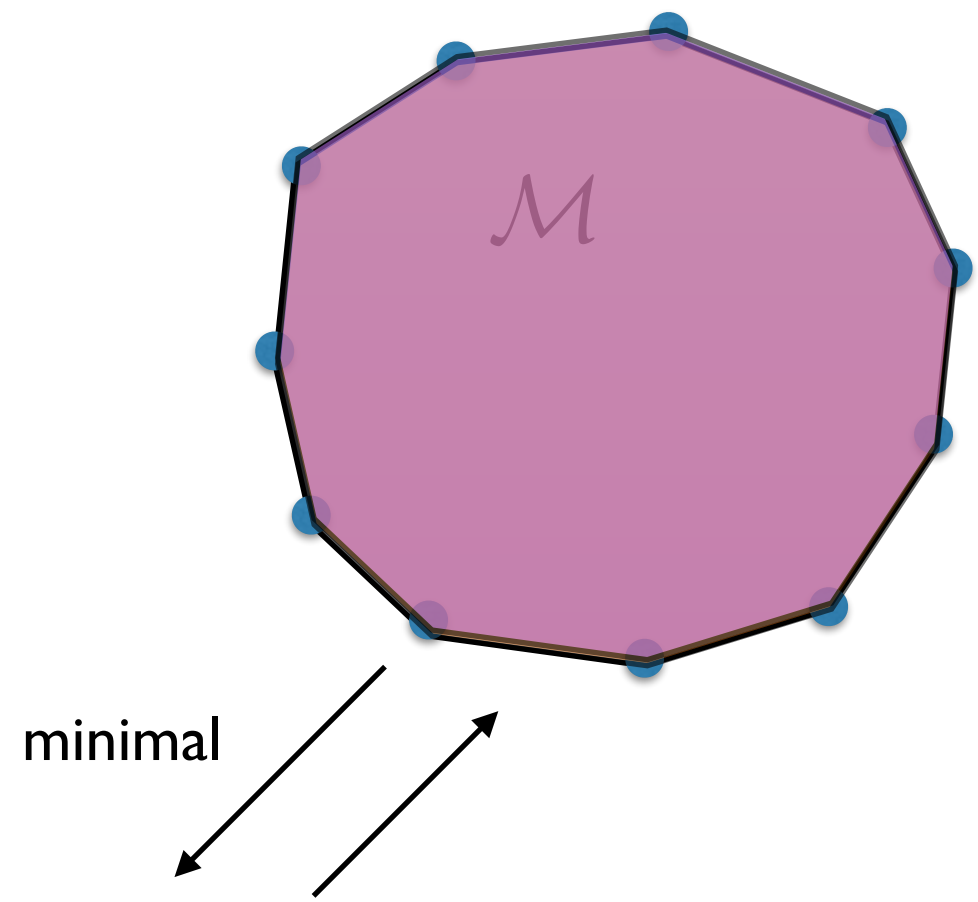
# The marginal polytope

$$p(y) \propto \exp\left(\sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j)\right)$$



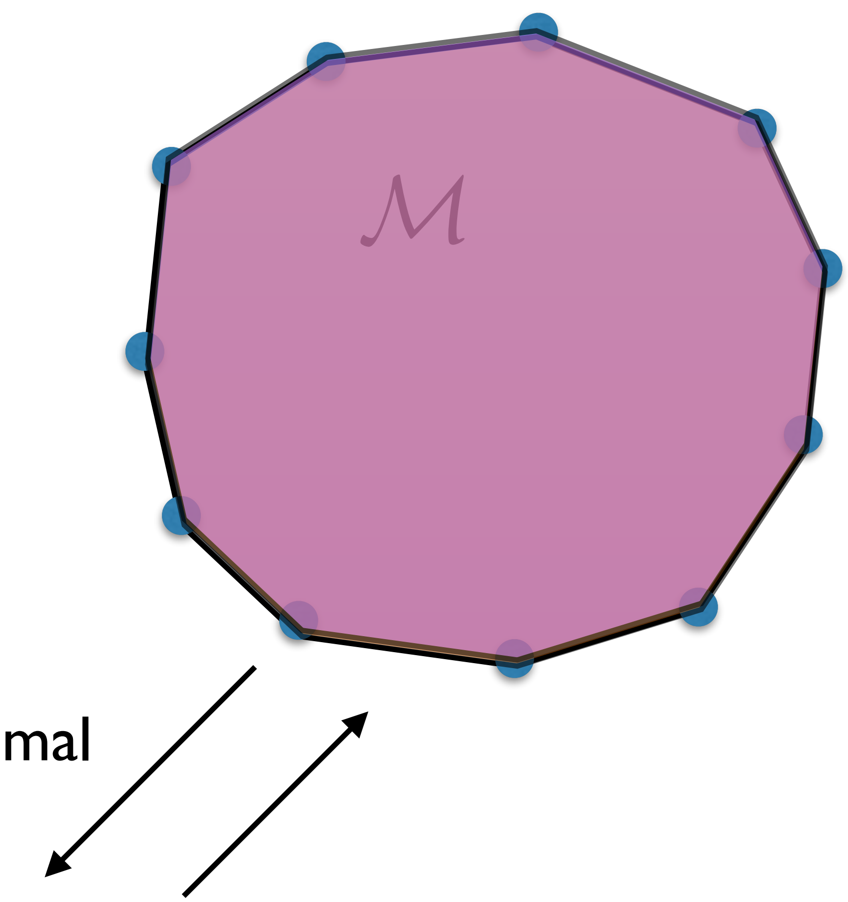

$$p(y) = P_\gamma\left[y = \arg\max_y \left\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \right\}\right]$$

# The marginal polytope




$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\}\Big]$$

# The marginal polytope

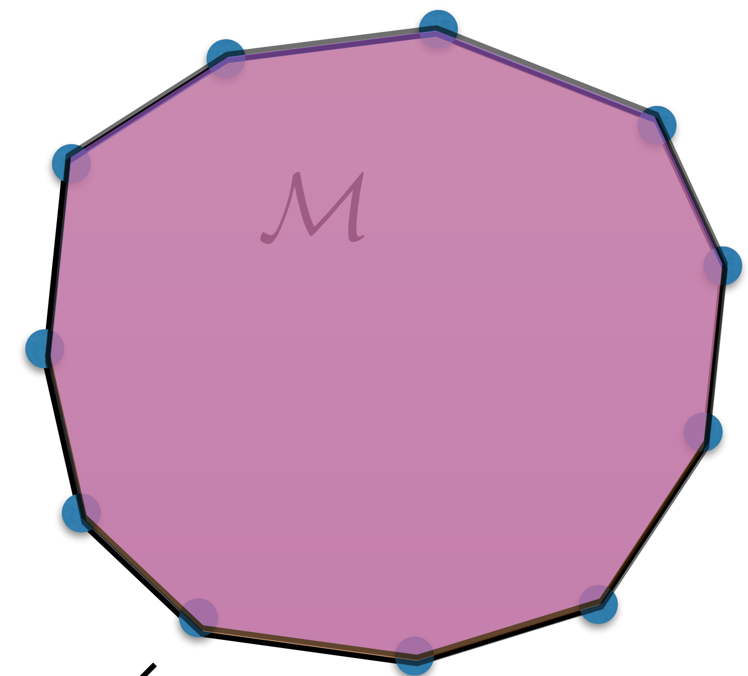



minimal

$$p(y) = P_\gamma \Big[ y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\} \Big]$$

# The marginal polytope

$$\mu = \begin{pmatrix} \mu_1(0), \mu_1(1), \mu_2(0), \mu_2(1), \mu_3(0), \mu_3(1), \\ \mu_{1,2}(0,0), \mu_{1,2}(0,1), \mu_{1,2}(1,0), \mu_{1,2}(1,1), \\ \mu_{2,3}(0,0), \mu_{2,3}(0,1), \mu_{2,3}(1,0), \mu_{2,3}(1,1)) \end{pmatrix}$$

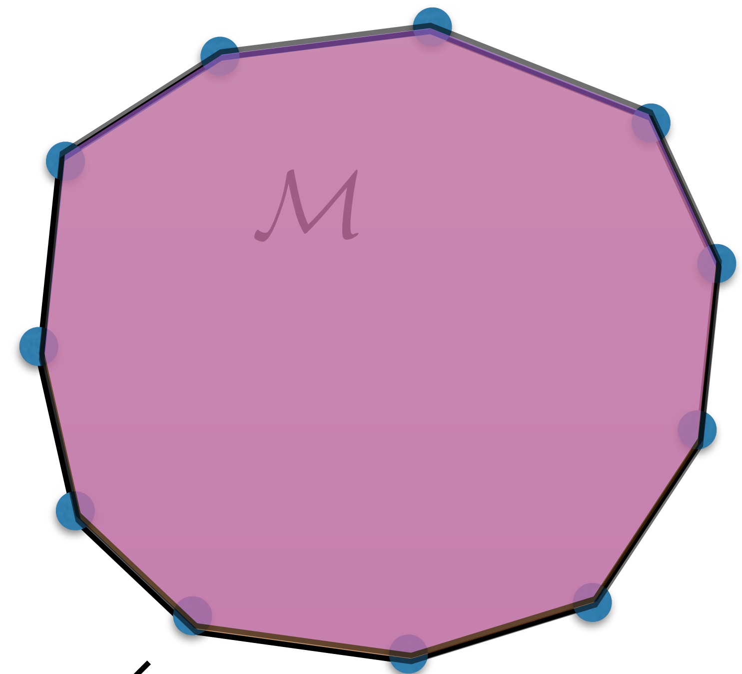



minimal

$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\}\Big]$$

# The marginal polytope

$$\mu = \begin{pmatrix} \mu_1(0), \mu_1(1), \mu_2(0), \mu_2(1), \mu_3(0), \mu_3(1), \\ \mu_{1,2}(0,0), \mu_{1,2}(0,1), \mu_{1,2}(1,0), \mu_{1,2}(1,1), \\ \mu_{2,3}(0,0), \mu_{2,3}(0,1), \mu_{2,3}(1,0), \mu_{2,3}(1,1)) \end{pmatrix}$$




minimal

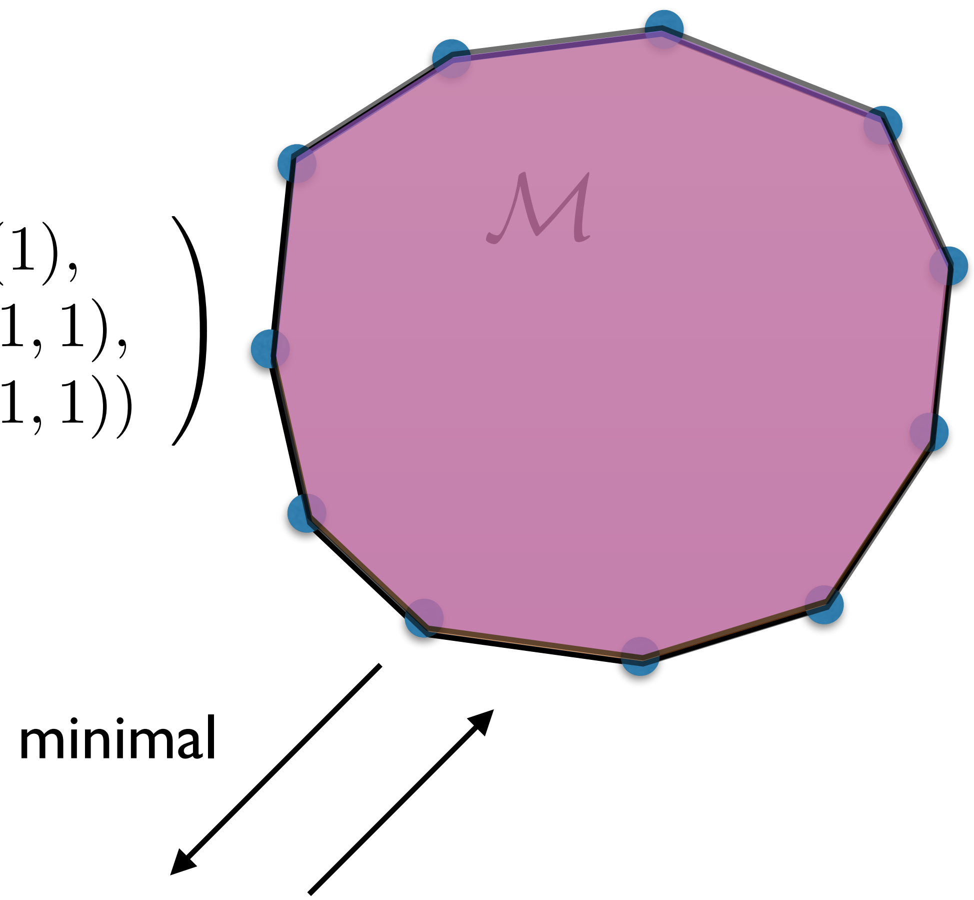
$$p(y) = P_\gamma \Big[ y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\} \Big]$$

- Proof idea:

$$\mu_i(y_i) = \frac{\partial E_\gamma \Big[ \max_y \big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \big\} \Big]}{\partial \theta_i(y_i)}$$

# The marginal polytope

$$\mu = \left( \begin{array}{l} \mu_1(0), \mu_1(1), \mu_2(0), \mu_2(1), \mu_3(0), \mu_3(1), \\ \mu_{1,2}(0,0), \mu_{1,2}(0,1), \mu_{1,2}(1,0), \mu_{1,2}(1,1), \\ \mu_{2,3}(0,0), \mu_{2,3}(0,1), \mu_{2,3}(1,0), \mu_{2,3}(1,1)) \end{array} \right)$$




$$p(y) = P_\gamma\Big[y = \arg\max_y \Big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \Big\}\Big]$$

- Proof idea:

$$\mu_i(y_i) = \frac{\partial E_\gamma\Big[\max_y \big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \big\}\Big]}{\partial \theta_i(y_i)}$$

$$\mu_{i,j}(y_i, y_j) = \frac{\partial E_\gamma\Big[\max_y \big\{ \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{i,j}(y_i, y_j) + \sum_i \gamma_i(y_i) \big\}\Big]}{\partial \theta_{i,j}(y_i, y_j)}$$

# Outline

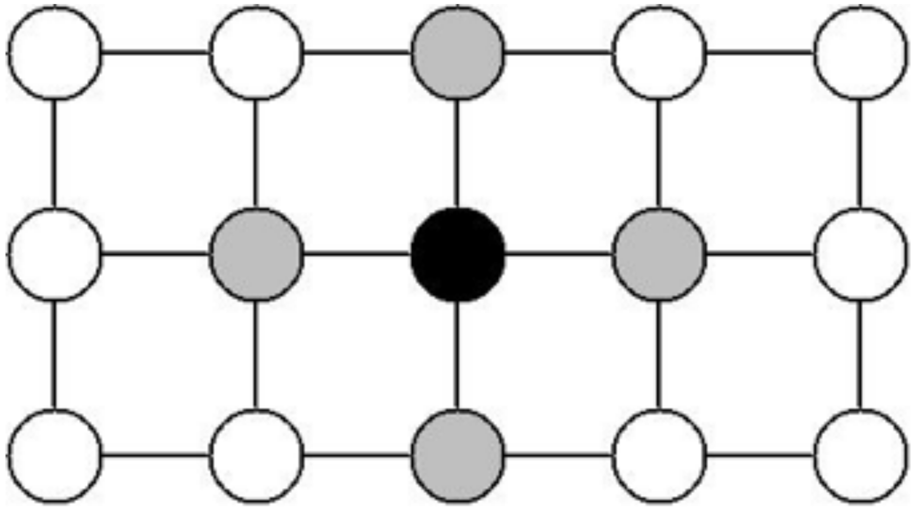
- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# Dependencies

- Gibbs distribution is defined by its Markov property:
  a variable is independent of the rest given its neighbors

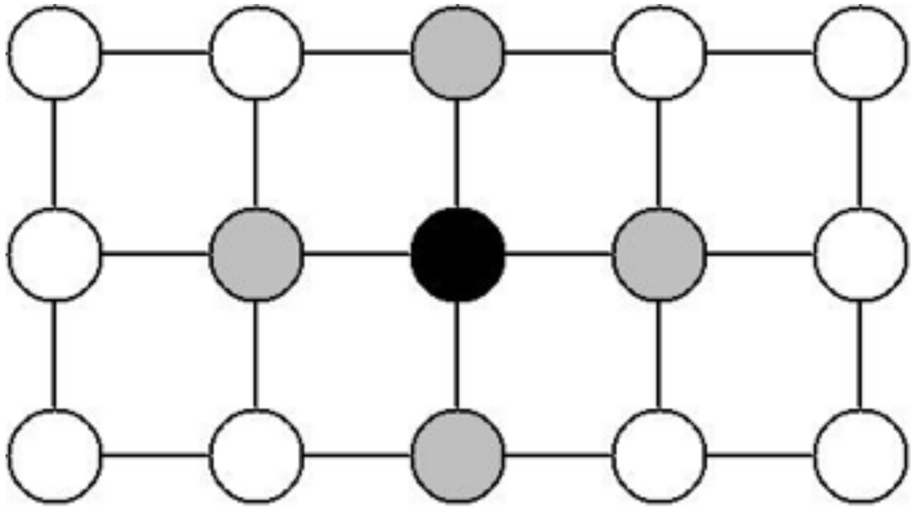
# Dependencies

- Gibbs distribution is defined by its Markov property: a variable is independent of the rest given its neighbors



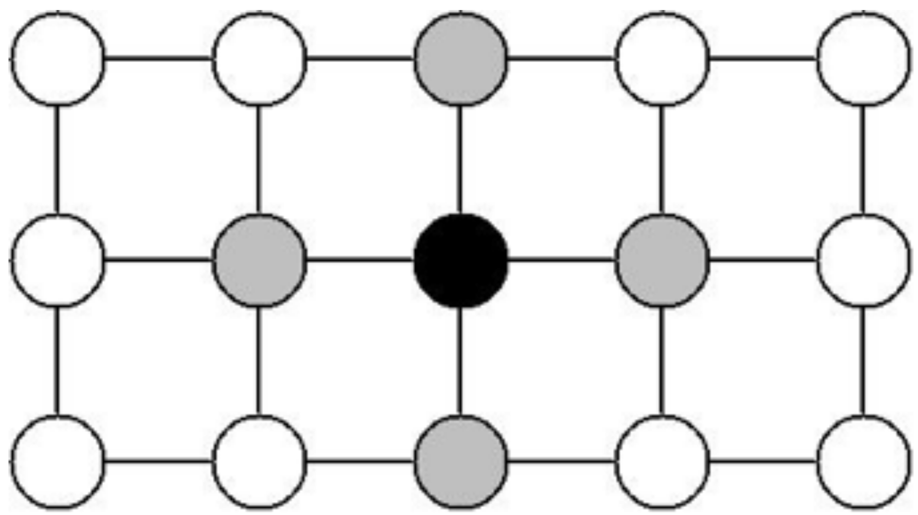
# Dependencies

- Gibbs distribution is defined by its Markov property: a variable is independent of the rest given its neighbors
  - Used in Gibbs sampling, Belief propagation etc.



# Dependencies

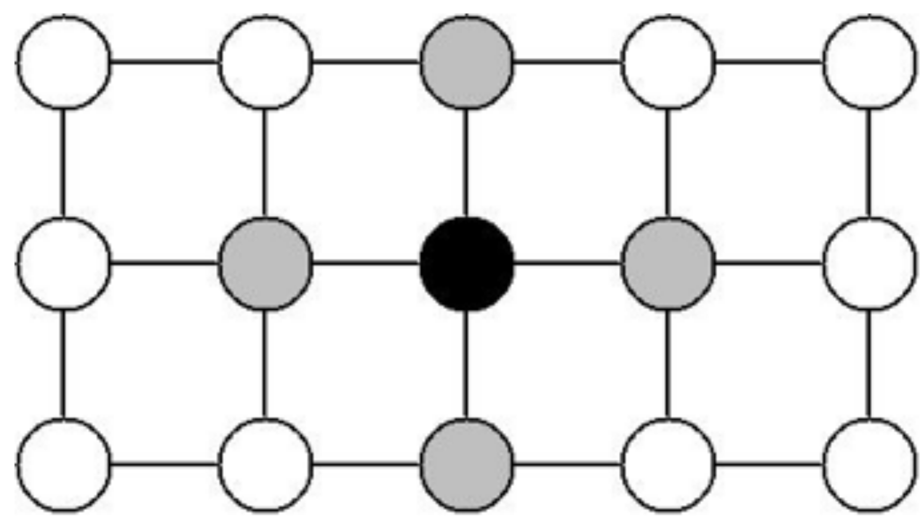
- Gibbs distribution is defined by its Markov property: a variable is independent of the rest given its neighbors
  - Used in Gibbs sampling, Belief propagation etc.



- What is the modeling power of perturb-max models ?

# Dependencies

- Gibbs distribution is defined by its Markov property: a variable is independent of the rest given its neighbors
  - Used in Gibbs sampling, Belief propagation etc.



- What is the modeling power of perturb-max models ?
- Can they model dependencies beyond graph neighborhoods?

# Long range dependencies

- Graphical model whose vertices are the joints.

# Long range dependencies

- Graphical model whose vertices are the joints.

- With Gibbs distribution the arm movements are independent of the legs given the body.

# Long range dependencies

- Graphical model whose vertices are the joints.

- With Gibbs distribution the arm movements are independent of the legs given the body.
- Perturb-max can model these long range dependencies (e.g., legs / arms dependencies).

# Gibbs distribution

$$p(y_1, y_2, y_3) = \frac{1}{Z} \exp\left(\theta_{1,2}(y_1, y_2) + \theta_{2,3}(y_2, y_3)\right)$$

# Gibbs distribution

$$p(y_1, y_2, y_3) = \frac{1}{Z} \exp\left(\theta_{1,2}(y_1, y_2) + \theta_{2,3}(y_2, y_3)\right)$$

- Conditional independence:

$$p(y_1, y_3 | y_2) = p(y_1 | y_2) p(y_3 | y_2)$$

# Gibbs distribution

$$p(y_1, y_2, y_3) = \frac{1}{Z} \exp\left(\theta_{1,2}(y_1, y_2) + \theta_{2,3}(y_2, y_3)\right)$$

- Conditional independence:

$$p(y_1, y_3 | y_2) = p(y_1 | y_2) p(y_3 | y_2)$$

- Probability of intersection = product of probabilities

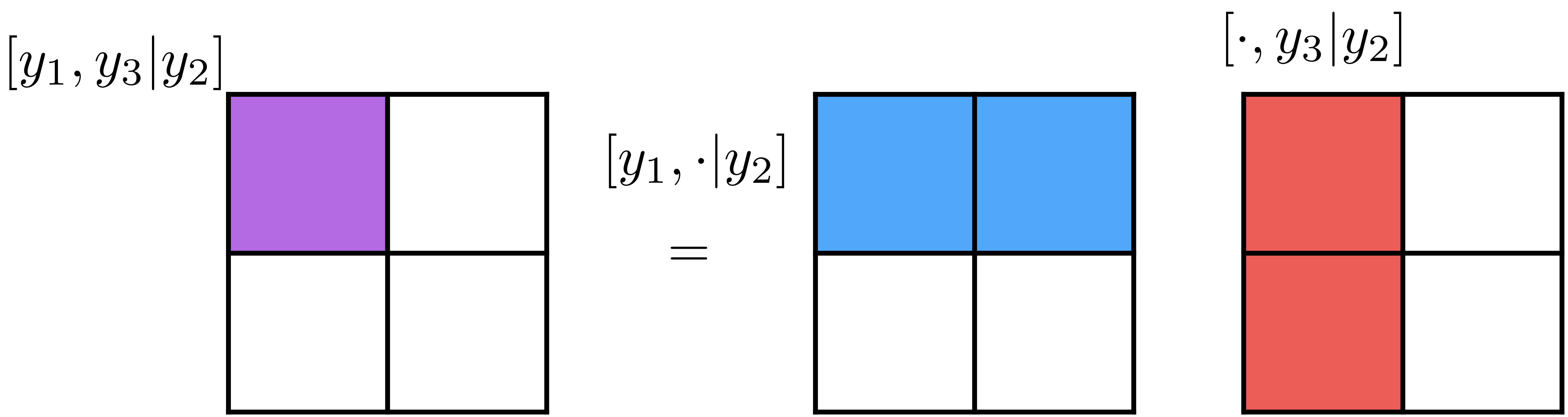
# Gibbs distribution

$$p(y_1, y_2, y_3) = \frac{1}{Z} \exp\left(\theta_{1,2}(y_1, y_2) + \theta_{2,3}(y_2, y_3)\right)$$

- Conditional independence:

$$p(y_1, y_3 | y_2) = p(y_1 | y_2) p(y_3 | y_2)$$

- Probability of intersection = product of probabilities



# Perturb-max models

$$P[y_1, y_2, y_3] = P_\gamma[y = \arg\max_{\hat{y}} \{\sum_{i,j} \theta_{i,j}(\hat{y}_i, \hat{y}_j) + \gamma_{i,j}(\hat{y}_i, \hat{y}_j)\}]$$

# Perturb-max models

$$P[y_1, y_2, y_3] = P_\gamma[y = \arg\max_{\hat{y}} \{ \sum_{i,j} \theta_{i,j}(\hat{y}_i, \hat{y}_j) + \gamma_{i,j}(\hat{y}_i, \hat{y}_j) \}]$$

- max-value flows in perturb-max models incorporate long-range interactions when using independent perturbations

# Perturb-max models

$$P[y_1, y_2, y_3] = P_\gamma[y = \arg\max_{\hat{y}}\{\sum_{i,j} \theta_{i,j}(\hat{y}_i, \hat{y}_j) + \gamma_{i,j}(\hat{y}_i, \hat{y}_j)\}]$$

- max-value flows in perturb-max models incorporate long-range interactions when using independent perturbations

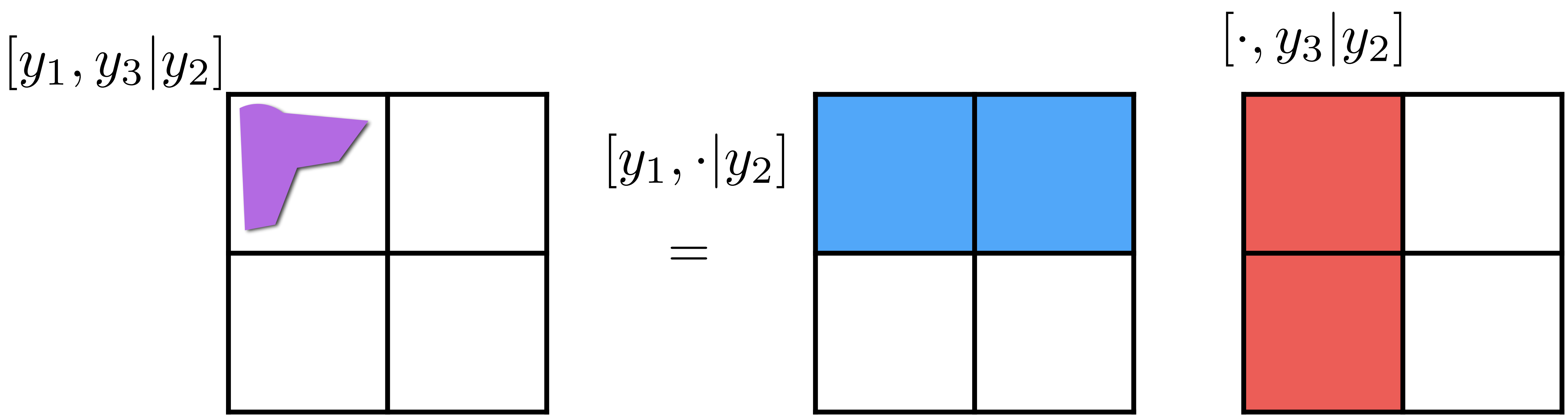
$$P[y_1, y_3 | y_2] \neq P[y_1 | y_2] P[y_3 | y_2]$$

# Perturb-max models

$$P[y_1, y_2, y_3] = P_\gamma[y = \arg\max_{\hat{y}}\{\sum_{i,j} \theta_{i,j}(\hat{y}_i, \hat{y}_j) + \gamma_{i,j}(\hat{y}_i, \hat{y}_j)\}]$$

- max-value flows in perturb-max models incorporate long-range interactions when using independent perturbations

$$P[y_1, y_3 | y_2] \neq P[y_1 | y_2] P[y_3 | y_2]$$



# Perturb-max models

- Proof idea: although fixing $y_2$ decomposes the max, information about the max-value flows.

# Perturb-max models

- Proof idea: although fixing $y_2$ decomposes the max, information about the max-value flows.

$$\max_{\hat{y}_1,\hat{y}_3}\{\theta_{1,2}(\hat{y}_1,y_2)+\gamma_{1,2}(\hat{y}_1,y_2)+\theta_{2,3}(y_2,\hat{y}_3)+\gamma_{2,3}(y_2,\hat{y}_3)\} =$$

# Perturb-max models

- Proof idea: although fixing $y_2$ decomposes the max, information about the max-value flows.

$$\max_{\hat{y}_1, \hat{y}_3}\{\theta_{1,2}(\hat{y}_1, y_2) + \gamma_{1,2}(\hat{y}_1, y_2) + \theta_{2,3}(y_2, \hat{y}_3) + \gamma_{2,3}(y_2, \hat{y}_3)\} =$$

$$\max_{\hat{y}_1}\{\theta_{1,2}(\hat{y}_1, y_2) + \gamma_{1,2}(\hat{y}_1, y_2)\} + \max_{\hat{y}_3}\{\theta_{2,3}(y_2, \hat{y}_3) + \gamma_{2,3}(y_2, \hat{y}_3)\}$$

# Perturb-max models

- Proof idea: although fixing $y_2$ decomposes the max, information about the max-value flows.

$$\max_{\hat{y}_1,\hat{y}_3}\{\theta_{1,2}(\hat{y}_1,y_2)+\gamma_{1,2}(\hat{y}_1,y_2)+\theta_{2,3}(y_2,\hat{y}_3)+\gamma_{2,3}(y_2,\hat{y}_3)\} =$$

$$\max_{\hat{y}_1}\{\theta_{1,2}(\hat{y}_1,y_2)+\gamma_{1,2}(\hat{y}_1,y_2)\}+\max_{\hat{y}_3}\{\theta_{2,3}(y_2,\hat{y}_3)+\gamma_{2,3}(y_2,\hat{y}_3)\}$$

- High $\gamma_{1,2}(y_1,y_2)$ will allow more values of $\gamma_{2,3}(y_2,y_3)$

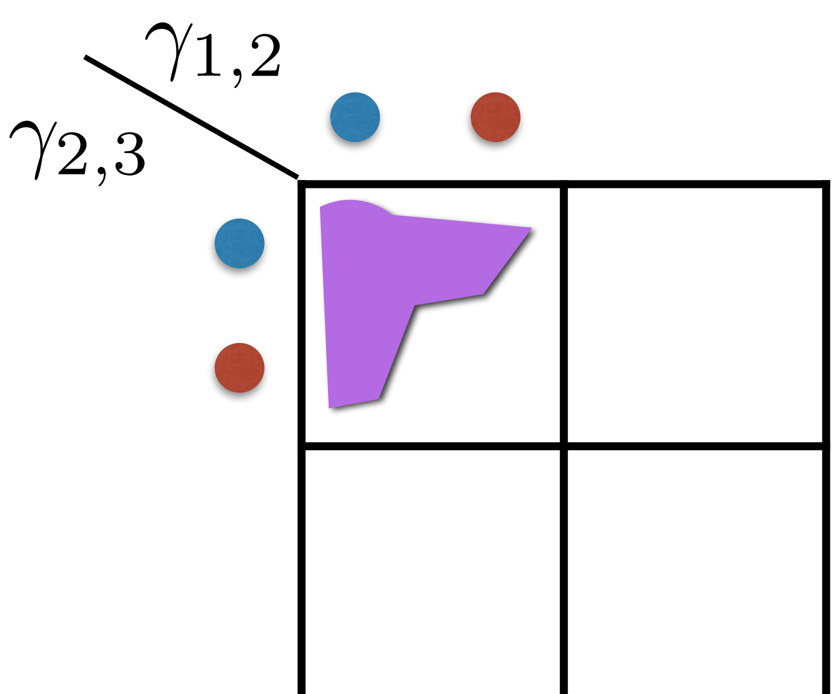
# Perturb-max models

- Proof idea: although fixing $y_2$ decomposes the max, information about the max-value flows.

$$\max_{\hat{y}_1,\hat{y}_3}\{\theta_{1,2}(\hat{y}_1,y_2)+\gamma_{1,2}(\hat{y}_1,y_2)+\theta_{2,3}(y_2,\hat{y}_3)+\gamma_{2,3}(y_2,\hat{y}_3)\} =$$

$$\max_{\hat{y}_1}\{\theta_{1,2}(\hat{y}_1,y_2)+\gamma_{1,2}(\hat{y}_1,y_2)\}+\max_{\hat{y}_3}\{\theta_{2,3}(y_2,\hat{y}_3)+\gamma_{2,3}(y_2,\hat{y}_3)\}$$

- High $\gamma_{1,2}(y_1,y_2)$ will allow more values of $\gamma_{2,3}(y_2,y_3)$




high

low

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \begin{bmatrix} -3 & 0 \\ 0 & 0 \end{bmatrix} \quad \theta_{i,j} = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$

# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# Log-likelihood learning

- Given training data $S$ of observed structures

# Log-likelihood learning

- Given training data $S$ of observed structures
  - e.g., foreground background segmentations $y \in S$

# Log-likelihood learning

- Given training data $S$ of observed structures
  - e.g., foreground background segmentations $y \in S$

- Learn the parameters $w$ that maximize the perturb-max likelihood

# Log-likelihood learning

- Given training data $S$ of observed structures
  - e.g., foreground background segmentations $y \in S$

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$p(y;w) = P_{\gamma \sim q_w}\left[y = \arg\max_{\hat{y}} \left\{ \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha;\gamma_\alpha) \right\}\right]$$

# Log-likelihood learning

- Examples
  - Learning segmentation potentials $y_i \in \{-1, 1\}$

# Log-likelihood learning

- Examples
  - Learning segmentation potentials $y_i \in \{-1, 1\}$

$$\theta_i(y_i; \gamma_i) = \gamma_i y_i$$

$$\theta_{i,j}(y_i, y_j; \gamma_{i,j}) = \begin{cases} \gamma_{i,j} & \text{if } y_i = y_j \\ -\gamma_{i,j} & \text{otherwise} \end{cases}$$

# Log-likelihood learning

- Examples
  - Learning segmentation potentials $y_i \in \{-1, 1\}$

$$\theta_i(y_i; \gamma_i) = \gamma_i y_i$$

$$\theta_{i,j}(y_i, y_j; \gamma_{i,j}) = \begin{cases} \gamma_{i,j} & \text{if } y_i = y_j \\ -\gamma_{i,j} & \text{otherwise} \end{cases}$$

- General notation:

$$\theta_\alpha(y_\alpha; \gamma_\alpha) = \gamma_\alpha \phi_\alpha(y_\alpha)$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

- Key fact: the perturb-max models are convolutions

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

- Key fact: the perturb-max models are convolutions

$$\begin{aligned} p(y; w) &= P_{\gamma \sim q_w}\left[ y = \arg\max_{\hat{y}} \Big\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\} \right] \\ &= \int q_w(\gamma) 1\big[y = \arg\max_{\hat{y}} \{\sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha)\}\big] d\gamma \end{aligned}$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

- Key fact: the perturb-max models are convolutions

$$p(y; w) = P_{\gamma \sim q_w}\left[ y = \arg\max_{\hat{y}} \Big\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\} \right]$$

$$= \int q_w(\gamma) 1 \big[ y = \arg\max_{\hat{y}} \underbrace{\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \} \big] d\gamma}_{\text{log-concave}}$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_{w} \sum_{y \in S} \log p(y; w)$$

- Key fact: the perturb-max models are convolutions

$$p(y;w) = P_{\gamma \sim q_w}\left[ y = \arg\max_{\hat{y}} \Big\{ \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\} \right]$$

$$= \int \underbrace{q_w(\gamma)}_{\text{strongly log-concave}} \underbrace{1\big[y = \arg\max_{\hat{y}} \{ \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \}\big]}_{\text{log-concave}} d\gamma$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

strongly concave ⇒ generalize well

- Key fact: the perturb-max models are convolutions

$$p(y;w) = P_{\gamma \sim q_w}\left[ y = \arg\max_{\hat{y}} \left\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \right\} \right]$$

$$= \int \underbrace{q_w(\gamma)}_{\text{strongly log-concave}} \underbrace{1\left[ y = \arg\max_{\hat{y}} \{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \} \right] d\gamma}_{\text{log-concave}}$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

strongly concave ⇒ generalize well

- Key fact: the perturb-max models are convolutions

$$p(y; w) = P_{\gamma \sim q_w}\left[y = \arg\max_{\hat{y}} \Big\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\}\right]$$

$$= \int q_w(\gamma) 1\big[y = \arg\max_{\hat{y}} \{\sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha)\}\big] d\gamma$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

strongly concave ⇒ generalize well

- Key fact: the perturb-max models are convolutions

$$p(y;w) = P_{\gamma \sim q_w}\Big[y = \arg\max_{\hat{y}} \Big\{ \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\}\Big]$$

$$= \int q_w(\gamma) 1\big[y = \arg\max_{\hat{y}} \{\sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha)\}\big] d\gamma$$

non-smooth

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

strongly concave ⇒ generalize well

- Key fact: the perturb-max models are convolutions

$$p(y; w) = P_{\gamma \sim q_w}\left[y = \arg\max_{\hat{y}} \Big\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big\}\right]$$

$$= \int q_w(\gamma) 1\Big[y = \arg\max_{\hat{y}} \{\sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha)\}\Big] d\gamma$$

smooth

non-smooth

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

strongly concave ⇒ generalize well

smooth (vanishing gradient ⇒ optimum)

- Key fact: the perturb-max models are convolutions

$$p(y; w) = P_{\gamma \sim q_w}\left[ y = \arg\max_{\hat{y}} \left\{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \right\} \right]$$

$$= \int \underbrace{q_w(\gamma)}_{\text{smooth}} \underbrace{1\left[y = \arg\max_{\hat{y}} \{ \sum_\alpha \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \}\right]}_{\text{non-smooth}} d\gamma$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

- Gradients match between prior and posterior predictions

$$\frac{\partial \sum_{y \in S} \log p(y; w)}{\partial w} = \sum_{y \in S} \left( E\left[ \gamma \;\middle|\; y = \arg\max_{\hat{y}} \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \right] - w \right)$$

# Log-likelihood learning

- Learn the parameters $w$ that maximize the perturb-max likelihood

$$\max_w \sum_{y \in S} \log p(y; w)$$

- Gradients match between prior and posterior predictions

$$\frac{\partial \sum_{y \in S} \log p(y; w)}{\partial w} = \sum_{y \in S} \Big( E\Big[ \gamma \,\Big|\, y = \arg\max_{\hat{y}} \sum_{\alpha} \theta_\alpha(\hat{y}_\alpha; \gamma_\alpha) \Big] - w \Big)$$

- When using log-concave perturbations the deviation of sampled average from the expectation decays exponentially (Orabona, Hazan, Sarwate, Jaakkola 14)

# Outline

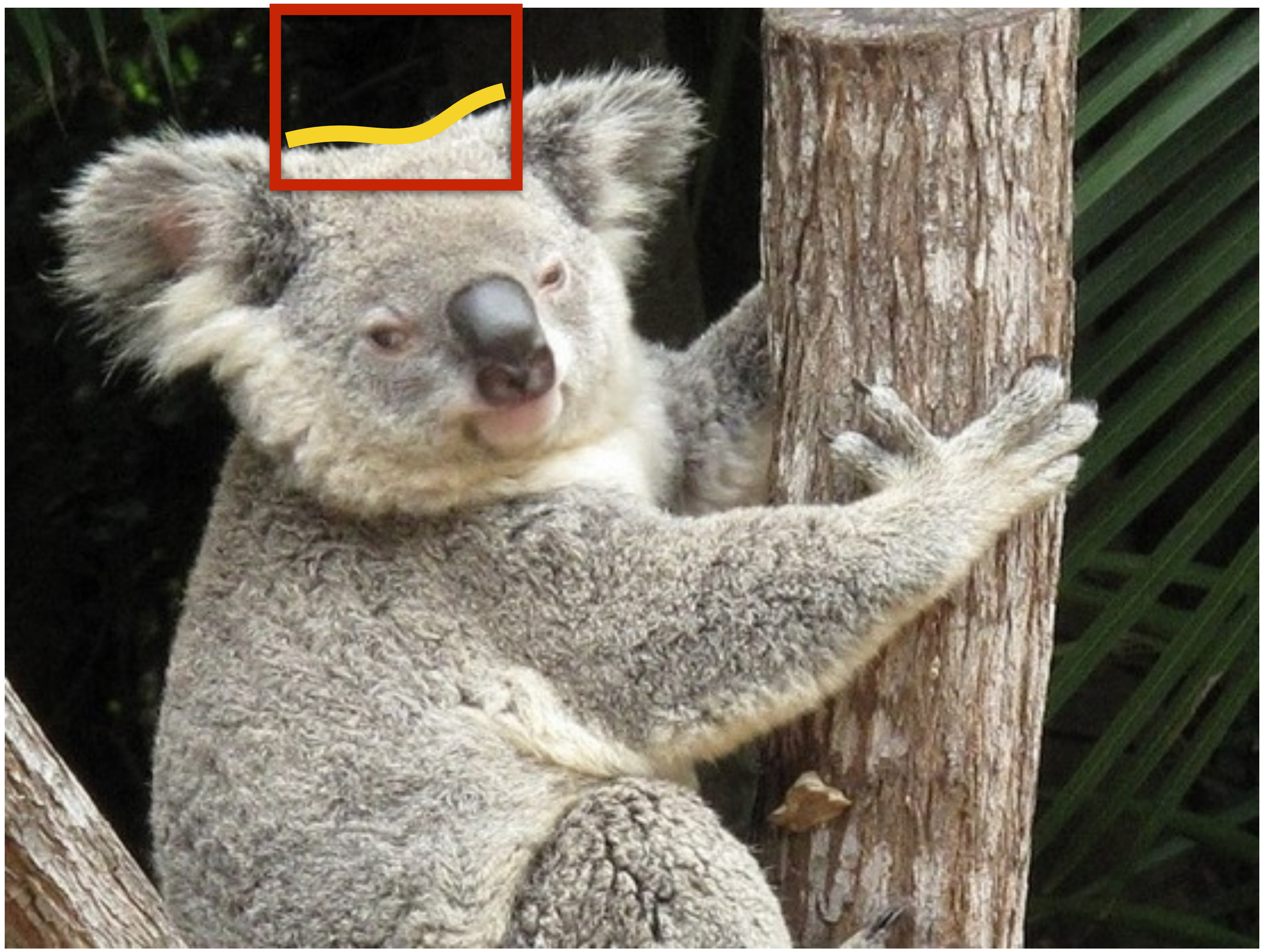
- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# Interactive learning

- Use perturb-max models to sample possible annotations and estimate user clicks [Maji, Hazan, Jaakkola 14]

# Interactive learning

- Use perturb-max models to sample possible annotations and estimate user clicks



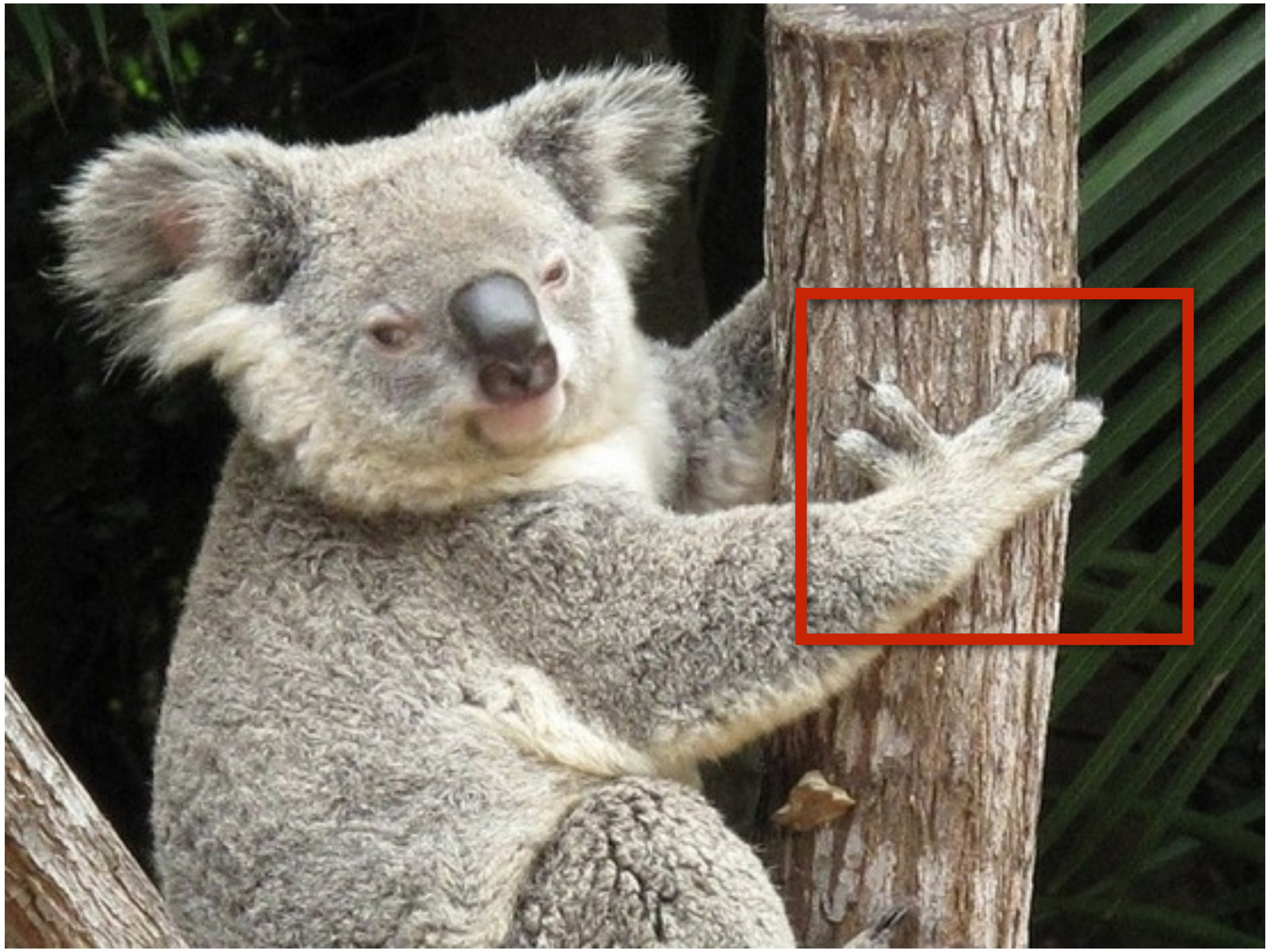
# Interactive learning

- Use perturb-max models to sample possible annotations and estimate user clicks

# Interactive learning

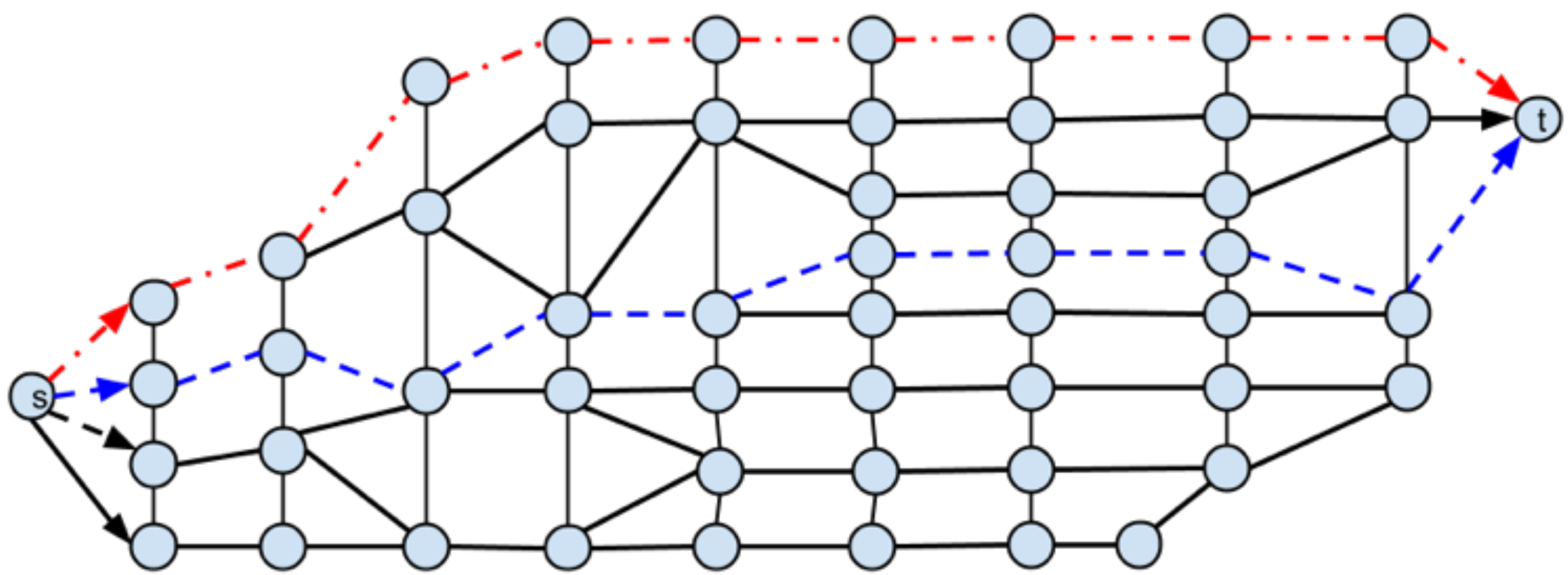
- Choose an area with the largest uncertainty reduction

$$U(p_\theta) = E_\gamma\Big[\sum_{i=1}^{n} \gamma_i(y_i^*)\Big] \qquad y^* = \arg\max_{\hat{y}}\{\theta(\hat{y}) + \sum_{i=1}^{n} \gamma_i(\hat{y}_i)\}$$



# Online Learning

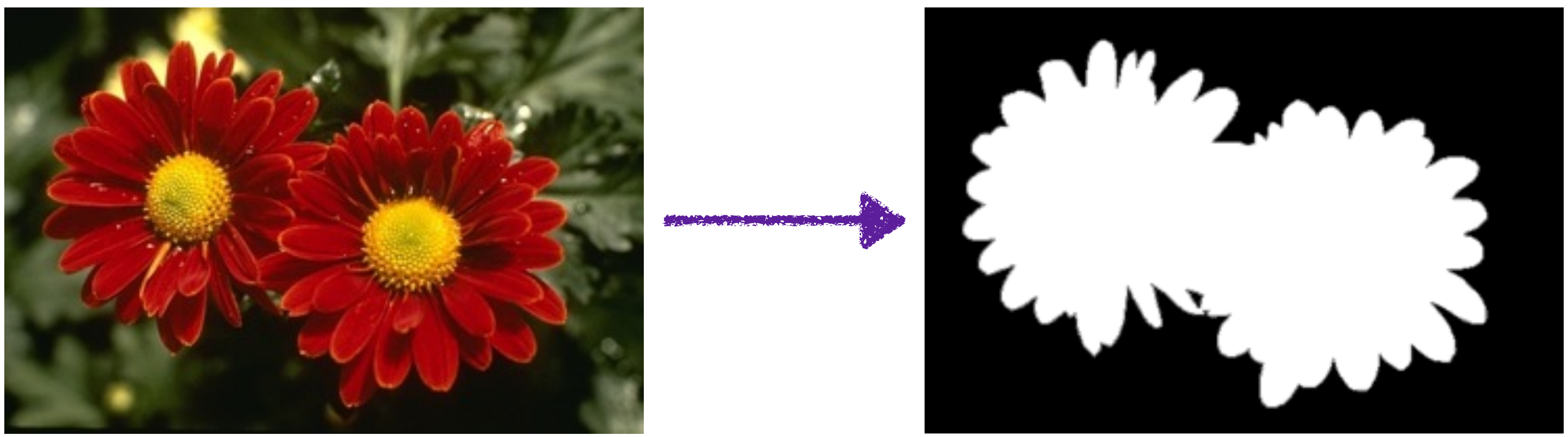
- How to choose your route to work?




- Choosing your route daily using perturb-max models is as good (on average) as choosing the best route [Kalai & Vempala 05, Cohen & Hazan 15]

# Loss minimization

- Can we learn from a finite sample set and generalize?



- Minimizing the average loss using perturb-max models generalize well [Hazan et al. 13]

# Outline

- Random perturbation - why and how?
  - Sampling likely structures as fast as finding the most likely one.
- Connections and Alternatives to Gibbs distribution:
  - the marginal polytope
  - the modeling power of perturb-max models
- Learning with perturb-max models
  - log-likelihood learning
  - interactive learning using new entropy bounds
  - online learning
  - loss minimization and PAC-Bayesian bounds

# Open problems

# Open problems

- Perturb-max models:

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?
  - How do the perturbations dimension affect the model properties?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?
  - How do the perturbations dimension affect the model properties?
  - In what ways higher dimension perturbations reveal complex structures in the model?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?
  - How do the perturbations dimension affect the model properties?
  - In what ways higher dimension perturbations reveal complex structures in the model?
  - How to apply perturbations in restricted spaces, e.g., super-modular potential functions?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?
  - How do the perturbations dimension affect the model properties?
  - In what ways higher dimension perturbations reveal complex structures in the model?
  - How to apply perturbations in restricted spaces, e.g., super-modular potential functions?
  - How to encourage diverse sampling?

# Open problems

- Perturb-max models:
  - When does fixing variables in the max-function amount to statistical conditioning?
  - When do perturb-max models preserve the most likely assignment?
  - How do the perturbations dimension affect the model properties?
  - In what ways higher dimension perturbations reveal complex structures in the model?
  - How to apply perturbations in restricted spaces, e.g., super-modular potential functions?
  - How to encourage diverse sampling?

# Thank you