UT DALLAS

CS 6347

Concave Entropy Approximations & Conditional Gradients

Maximum Entropy

$$\max_{q^1,\ldots,q^m} \sum_m H(q^m)$$

such that the moment matching condition is satisfied

$$\sum_m f(x^m, y^m) = \sum_m \sum_x q^m(x|y^m) f(x, y^m)$$

$q^1, \ldots, q^m$ are discrete probability distributions

and $f(x^m, y^m) = \sum_C f_C(x_C^m, y^m)$

Regularized MLE

- L_2 regularizer with a constant λ
  - λ is unknown and is chosen by cross-validation

Regularized log-likelihood:

$$\left\langle \theta, \sum_m \sum_C f_C(x_C^m, y^m) \right\rangle - \sum_m \log Z(\theta, y^m) \; - \frac{\lambda}{2} \|\theta\|_2^2$$

Regularized maximum entropy:

$$\max_{q^1,\ldots,q^m} \sum_m H(q^m) - \frac{1}{2\lambda} \left\| \sum_m f(x^m, y^m) - \sum_m \sum_x q^m(x|y^m) f(x, y^m) \right\|_2^2$$

Bethe Entropy

$$H_B(\tau) = -\sum_{i \in V} \sum_{x_i} \tau_i(x_i) \log \tau_i(x_i) - \sum_{C} \sum_{x_C} \tau_C(x_C) \log \frac{\tau_C(x_C)}{\prod_{k \in C} \tau_k(x_k)}$$

- τ are pseudomarginals in the marginal polytope
- Not concave in general
  - Real entropy is concave
  - Can make it concave by "reweighting" some of the pieces

Concave Entropy Approximations

$$H_\rho(\tau) = -\sum_{i \in V} \sum_{x_i} \tau_i(x_i) \log \tau_i(x_i) - \sum_C \rho_C \sum_{x_C} \tau_C(x_C) \log \frac{\tau_C(x_C)}{\prod_{k \in C} \tau_k(x_k)}$$

$$= -\sum_{i \in V} \sum_{x_i} \left(1 - \sum_{C \supset i} \rho_C\right) \tau_i(x_i) \log \tau_i(x_i) - \sum_C \rho_C \sum_{x_C} \tau_C(x_C) \log \tau_C(x_C)$$

- For each clique C, choose some real number $\rho_C \geq 0$
  - We can always choose the ρ such that the resulting approximation is concave
  - Use this as a surrogate for the true entropy

Reweighted Maximum Entropy

$$\max_{\tau^1,\ldots,\tau^M \in T} \sum_m H_\rho(\tau^m) - \frac{1}{2\lambda} \left\| \sum_m f(x^m, y^m) - \sum_m \sum_C \sum_{x_C} \tau_C^m(x_C|y^m) f_C(x_C, y^m) \right\|_2^2$$

- For appropriate choice of ρ this is a constrained concave optimization problem
- This approximate maximum entropy optimization problem is dual to an approximate MLE optimization problem where we approximate Z using the Bethe free energy with a concave entropy approximation
  - Note: strong duality holds when this problem is concave and you choose the same ρ for both max-entropy and MLE

Gradient Descent

- Suppose that we want to minimize a convex function $f(x)$
- Start with an initial point x^0

$$x^t = x^{t-1} - \gamma_t \nabla f(x^{t-1})$$

  - γ_t is a step size
- Idea: step along a decreasing direction

Franke-Wolfe

- Let's suppose that we want to minimize a convex function $f(x)$ over a convex set S
  - Could take one step of gradient descent
  - If we end up outside of S, just project back in (can be computationally expensive)
- An alternative: the Frank-Wolfe algorithm
  - To minimize a convex function over a convex set, it suffices to solve a series of linear optimization problems

Franke-Wolfe

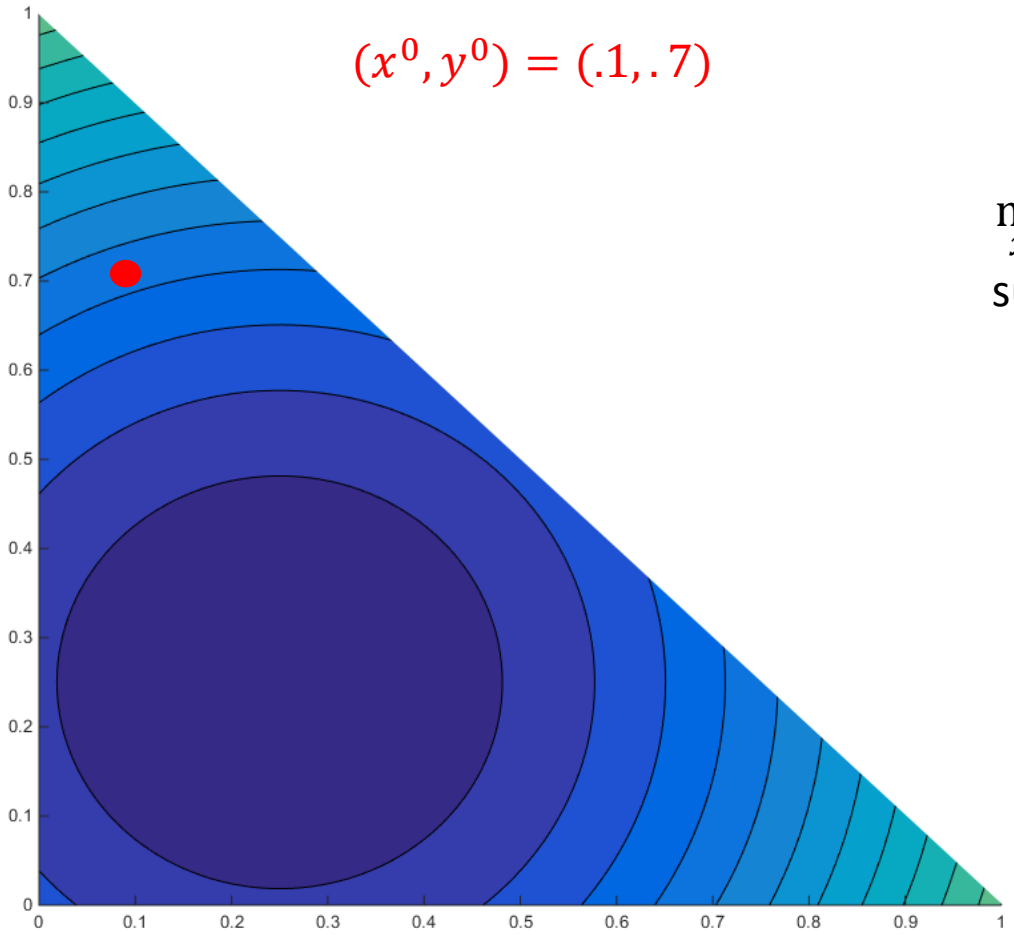
- Start with an initial point $x^0 \in S$

$$s^t = \arg\min_{x \in S} \langle x, \nabla f(x^{t-1}) \rangle$$

$$x^t = (1 - \gamma_t) x^{t-1} + \gamma_t s^t$$

- γ_t is the step size
  - The algorithm is guaranteed to converge if $\gamma_t = \frac{2}{2+t}$
  - Other choices are also possible

Franke-Wolfe



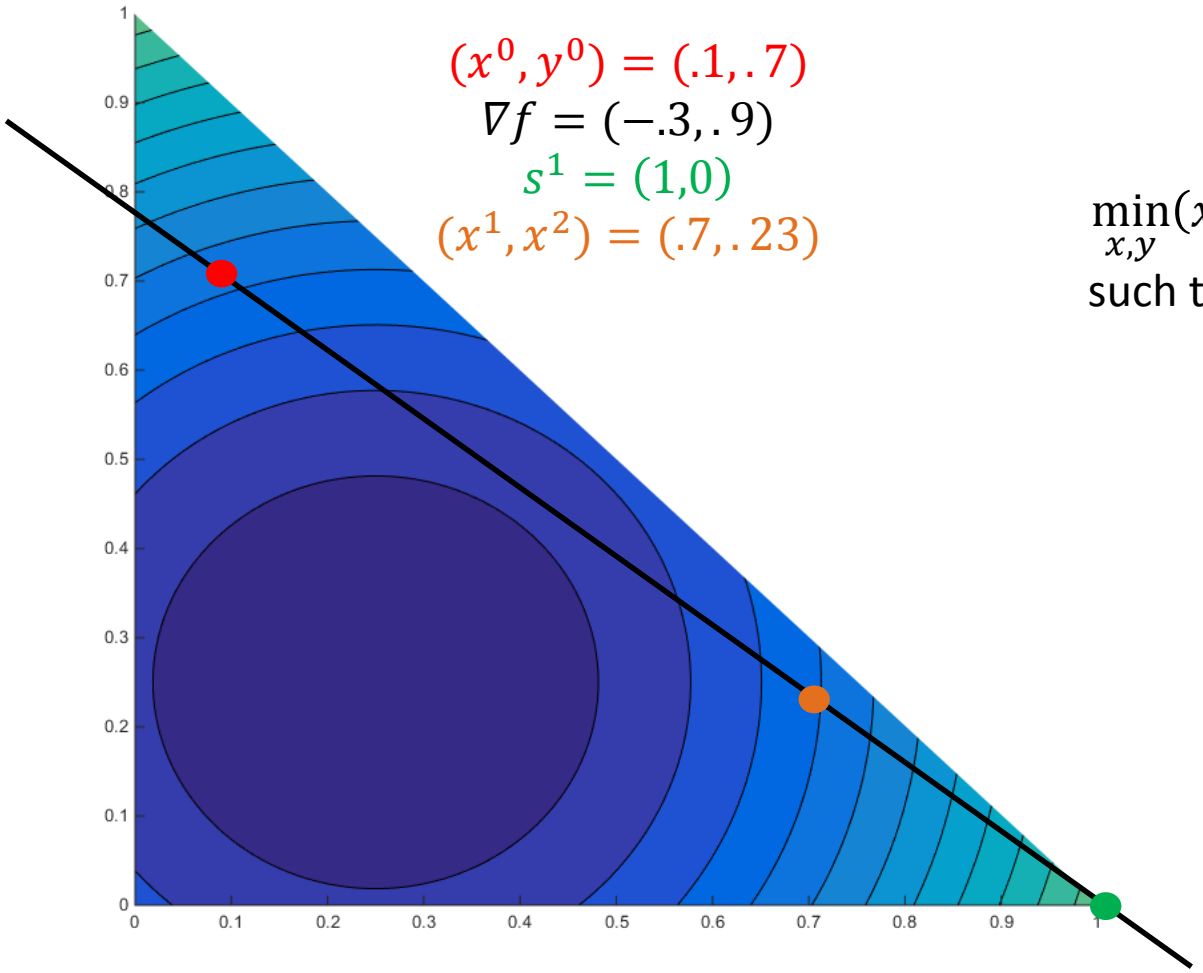
$(x^0, y^0) = (.1, .7)$

$$\min_{x,y} (x - .25)^2 + (y - .25)^2$$

such that

$$x + y \leq 1$$
$$0 \leq x, y \leq 1$$

Franke-Wolfe



$(x^0, y^0) = (.1, .7)$

$\nabla f = (-.3, .9)$

$s^1 = (1,0)$

$(x^1, x^2) = (.7, .23)$

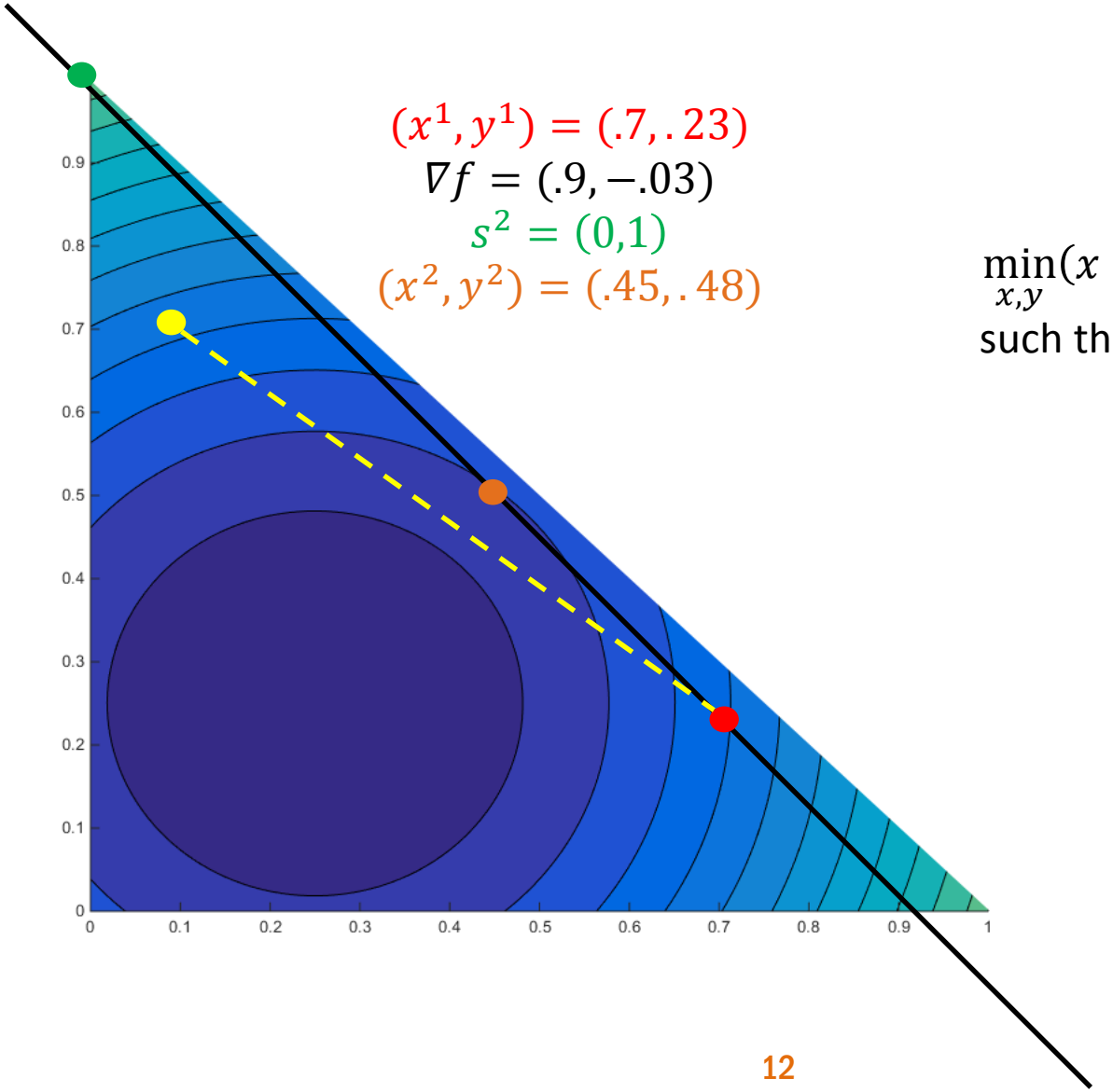
$$\min_{x,y} (x - .25)^2 + (y - .25)^2$$

such that

$$x + y \leq 1$$

$$0 \leq x, y \leq 1$$

Franke-Wolfe



$(x^1, y^1) = (.7, .23)$

$\nabla f = (.9, -.03)$

$s^2 = (0,1)$

$(x^2, y^2) = (.45, .48)$

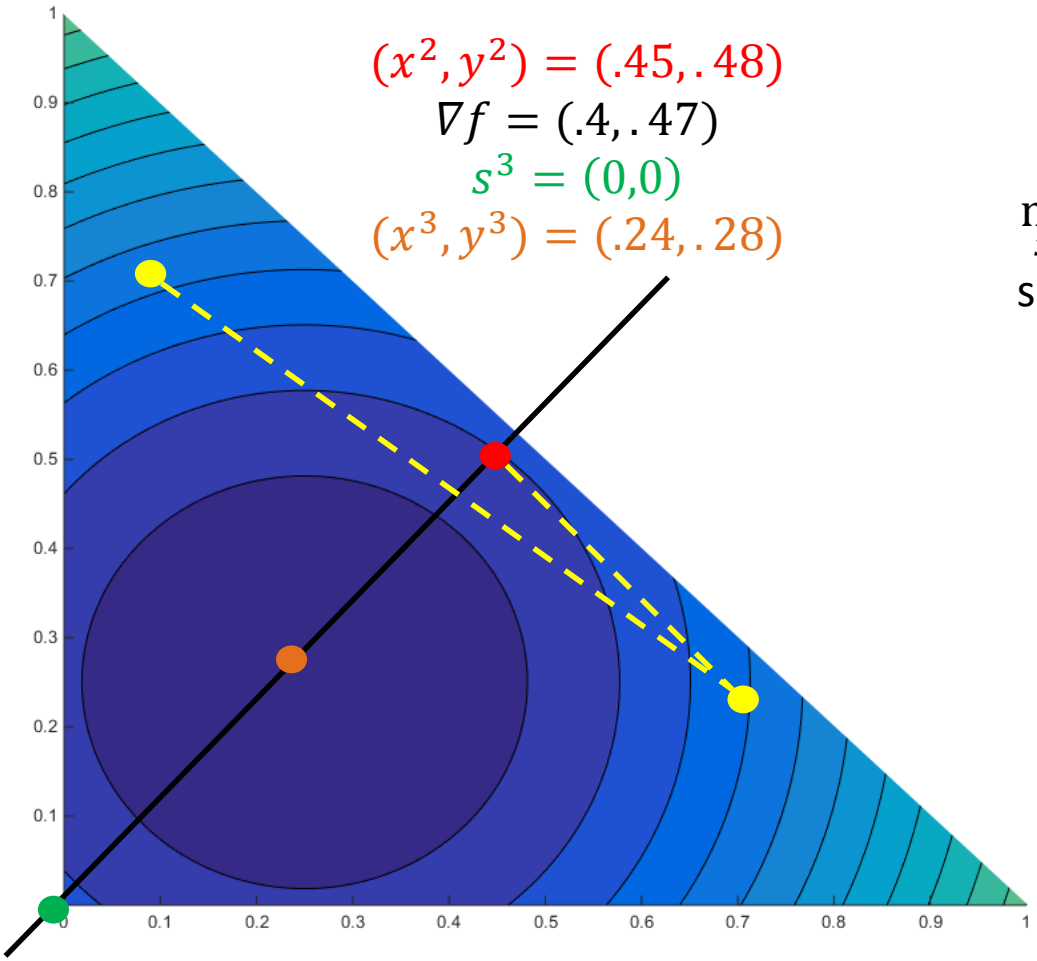
$$\min_{x,y} (x - .25)^2 + (y - .25)^2$$

such that

$$x + y \leq 1$$

$$0 \leq x, y \leq 1$$

Franke-Wolfe



$$\min_{x,y} (x - .25)^2 + (y - .25)^2$$

such that

$$x + y \leq 1$$
$$0 \leq x, y \leq 1$$

Reweighted Maximum Entropy

$$Ent(\tau^1, \dots, \tau^M) = \sum_m H_\rho(\tau^m) - \frac{1}{2\lambda} \left\| \sum_m f(x^m, y^m) - \sum_m \sum_C \sum_{x_C} \tau_C^m(x_C|y^m) f_C(x_C, y^m) \right\|_2^2$$

- To apply FW, need to compute the gradient with respect to $\tau^1, \dots, \tau^M$
- The optimization we need to solve is

$$\arg \max_{\mu^1, \dots, \mu^m \in T} \langle \mu, \nabla Ent(\tau^1, \dots, \tau^M) \rangle$$

- This is a linear programming problem over the local polytope
  - This means it corresponds to solving an approximate MAP problem!

MAP LP

$$\max_{\tau} \sum_{i \in V} \sum_{x_i} \tau_i(x_i) \log \phi_i(x_i) + \sum_{C} \sum_{x_C} \tau_C(x_C) \log \psi_C(x_C)$$

such that

$$\sum_{x_i} \tau_i(x_i) = 1 \qquad \text{For all } i \in V$$

$$\sum_{x_{C \setminus i}} \tau_C(x_C) = \tau_i(x_i) \qquad \text{For all } C, i \in C, x_i$$

$$\tau_i(x_i) \in [0,1] \qquad \text{For all } i \in V, x_i$$

$$\tau_C(x_C) \in [0,1] \qquad \text{For all } C, x_C$$

Reweighted Maximum Entropy

$$Ent(\tau^1, \ldots, \tau^M) = \sum_m H_\rho(\tau^m) - \frac{1}{2\lambda} \left\| \sum_m f(x^m, y^m) - \sum_m \sum_C \sum_{x_C} \tau_C^m(x_C|y^m) f_C(x_C, y^m) \right\|_2^2$$

- Can solve this optimization problem just by solving a series of approximate MAP (linear programming problems)
  - Many general purpose solvers exist for LPs
  - Could use belief propagation!

Reweighted Sum-Product

- We know that fixed points of loopy BP correspond to local optima of the Bethe free energy
- Is there an analog of sum-product for each choice of ρ?
  - Yes!

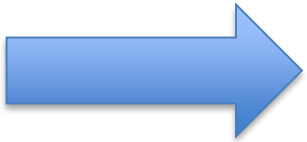
Reweighted Sum-Product

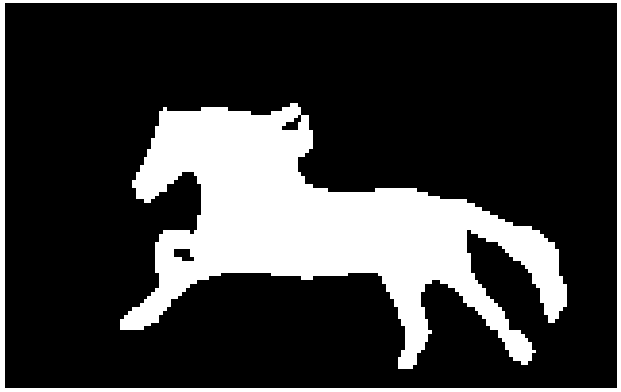
- $p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j)$

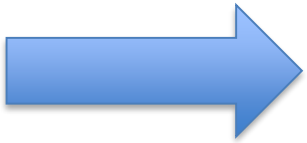
$$m_{i \to j}(x_j) = \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j)^{\frac{1}{\rho_{ij}}} \left[\frac{\prod_{k \in N(i)} m_{k \to \mathrm{i}}(x_i)^{\rho_{ki}}}{m_{j \to i(x_i)}} \right]$$

- $\rho = \vec{1}$ is equal to regular belief propagation

Image Segmentation







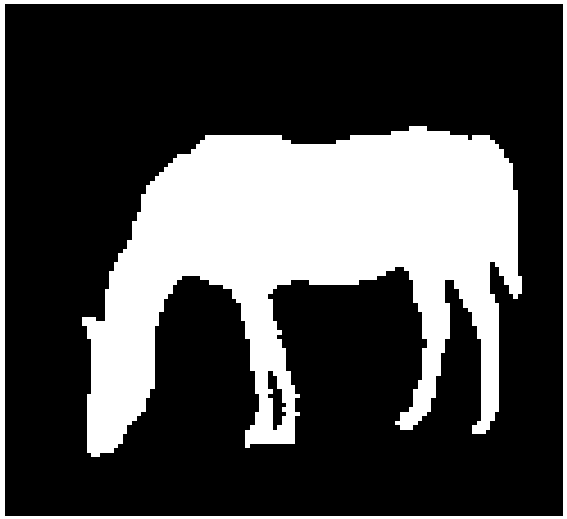
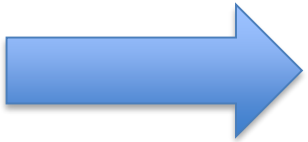
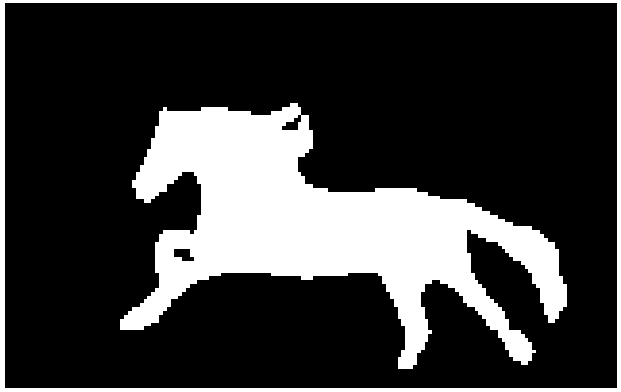


Image Segmentation

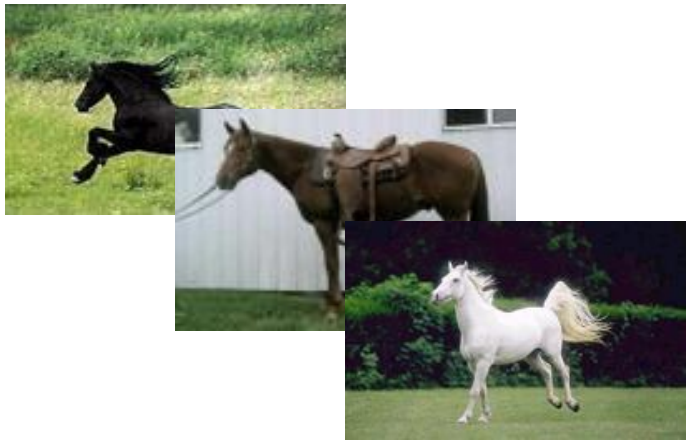


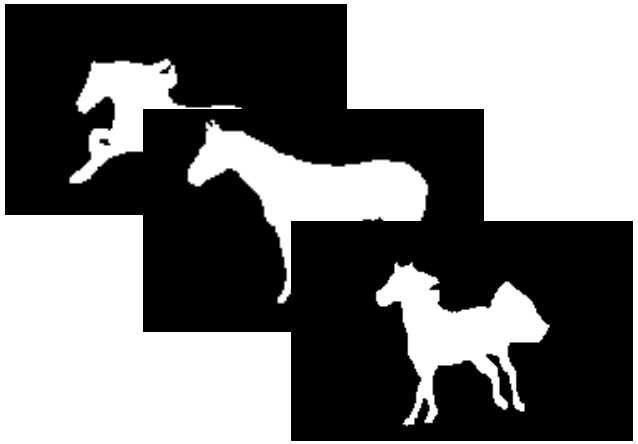


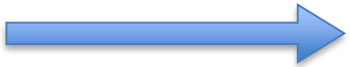
This image is 159x100 = 15,900 pixels

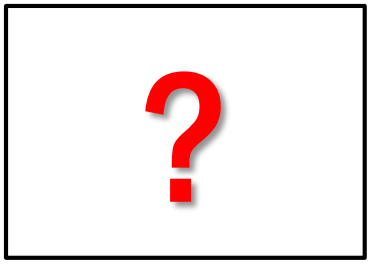
$2^{15,900}$ different possible segmentations!

Image Segmentation







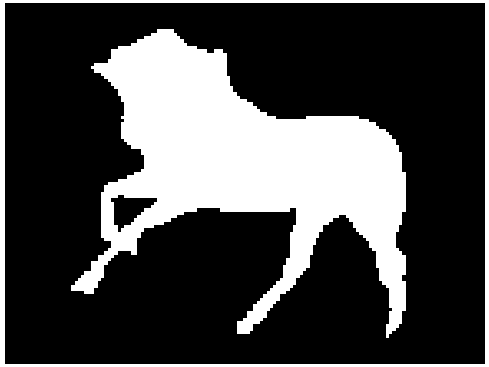


Given a set of labeled training examples, we want to learn the weights of an Ising model (with features) to correctly predict the segmentation of an unseen horse

Image Segmentation

Unseen Test Image

Ground Truth Segmentation



100 iterations
(9 mins)

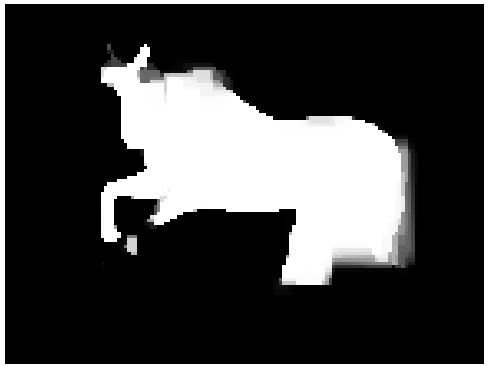
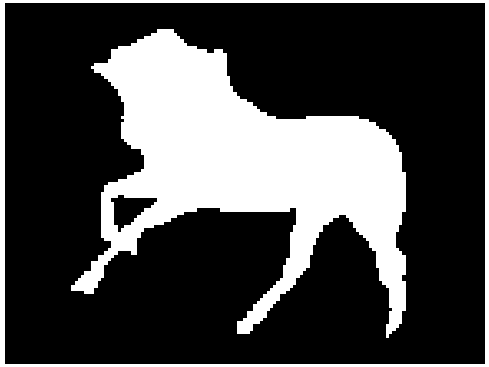


Image Segmentation

Unseen Test Image

Ground Truth Segmentation



250 iterations

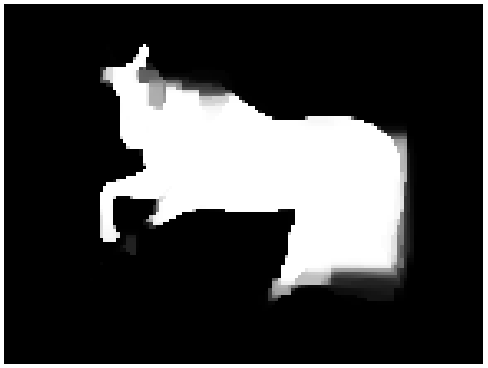
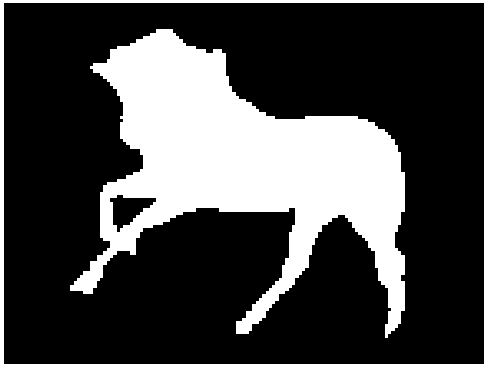


Image Segmentation

Unseen Test Image

Ground Truth Segmentation



2,000 iterations

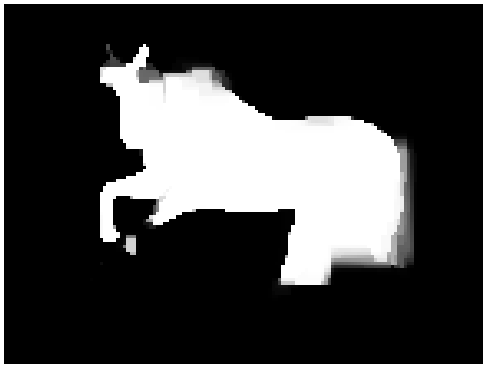
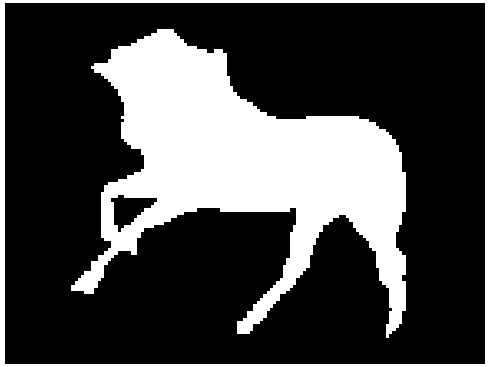


Image Segmentation

Unseen Test Image

Ground Truth Segmentation

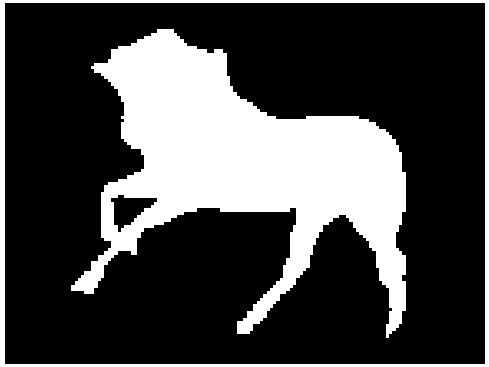


11,750 iterations

Image Segmentation

Unseen Test Image

Ground Truth Segmentation



100,000 iterations

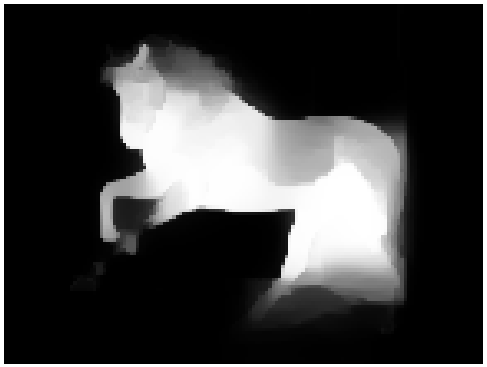
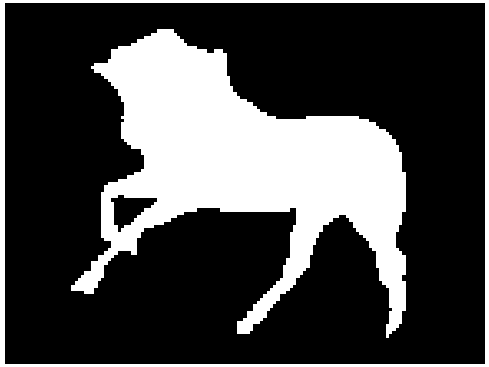


Image Segmentation

Unseen Test Image

Ground Truth Segmentation



250,000 iterations
(3.7 hours)

Test Error Over Time