Early Stopping is Nonparametric Variational Inference

Dougal Maclaurin, David Duvenaud, Ryan Adams

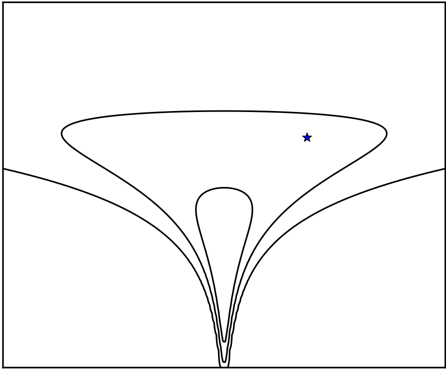
HARVARD
School of Engineering
and Applied Sciences

Good ideas always have Bayesian interpretations

Regularization	$=$	MAP inference
Limiting model capacity	$=$	Bayesian Occam's razor
Cross-validation	$=$	Estimating marginal likelihood
Dropout	$=$	Integrating out spike-and-slab
Ensembling	$=$	Bayes model averaging?
Early stopping	$=$	??

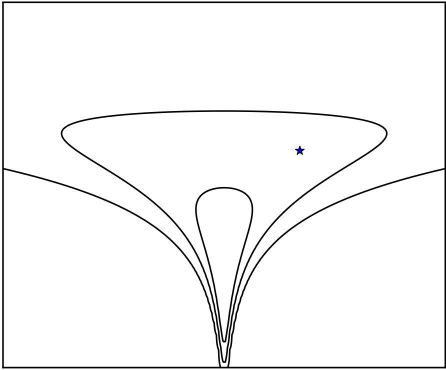
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



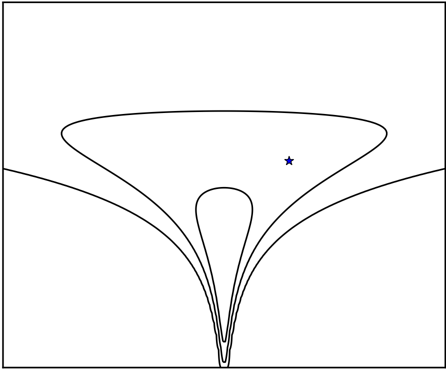
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



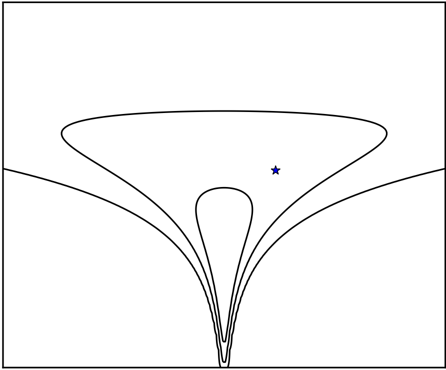
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



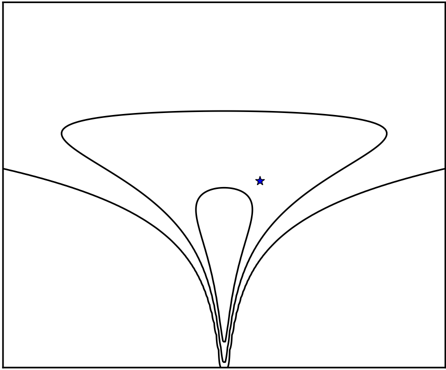
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



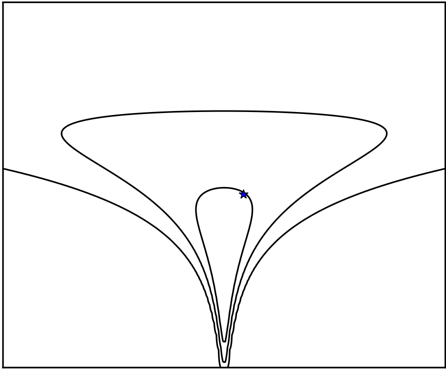
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



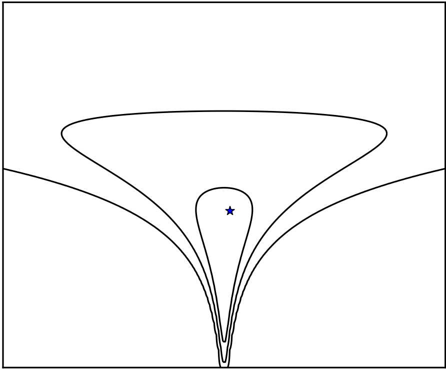
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



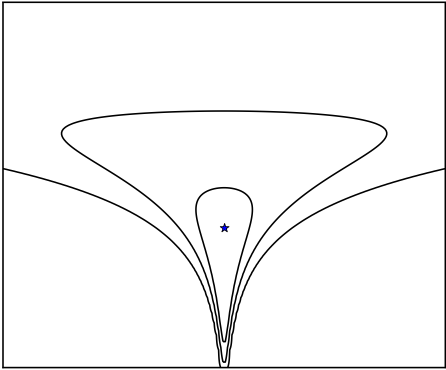
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



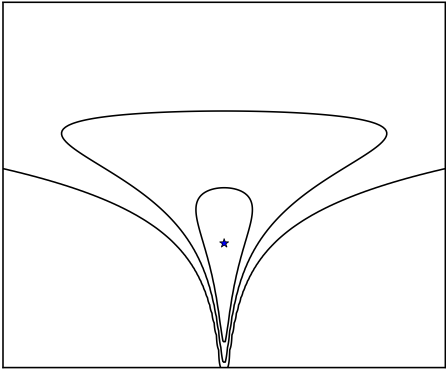
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



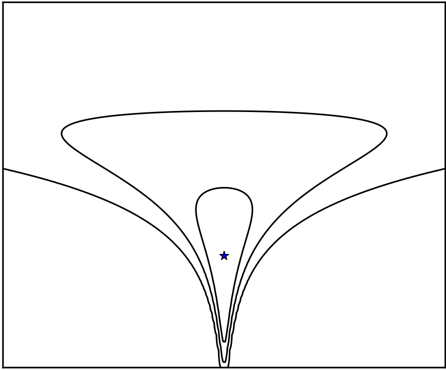
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



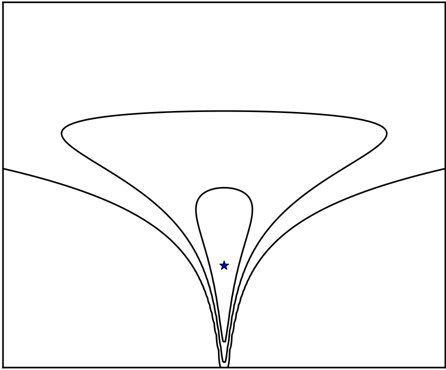
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



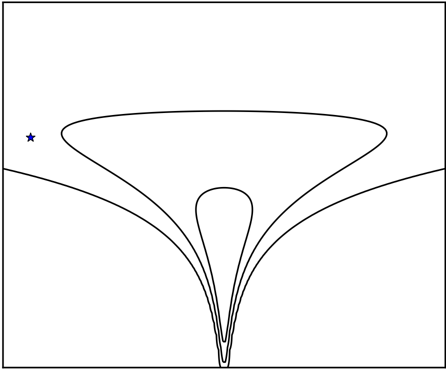
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



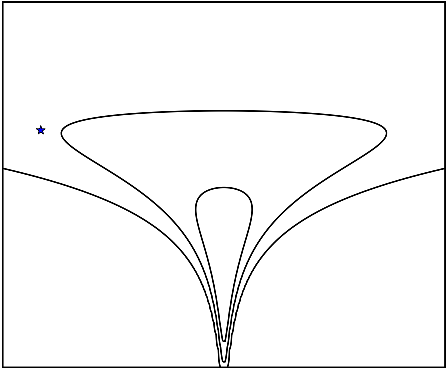
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



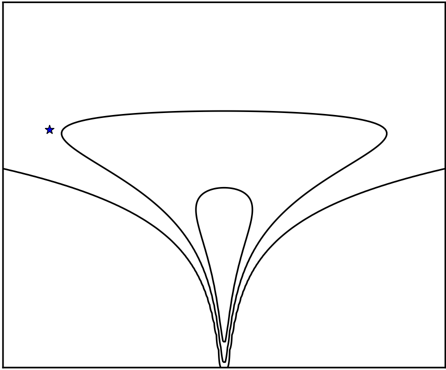
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



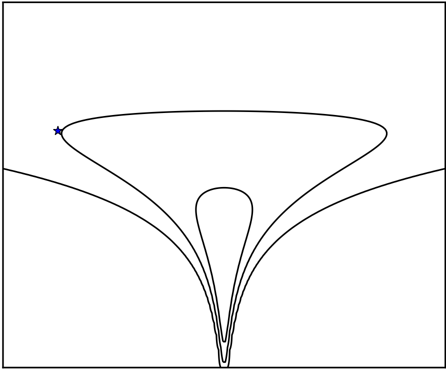
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



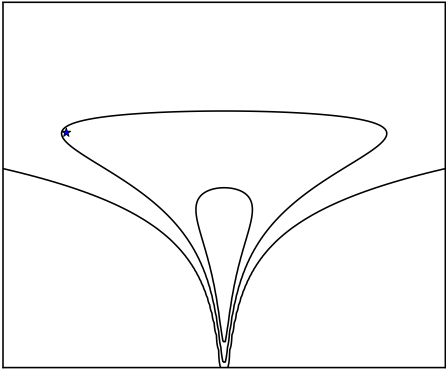
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...

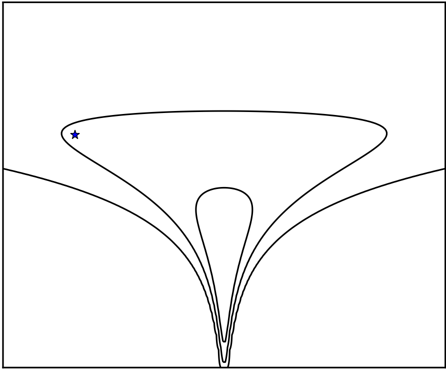


Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



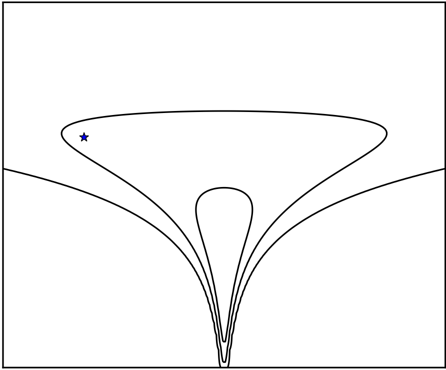
Gradient descent as a sampler



- Optimization paths start from random init, and move towards modes...

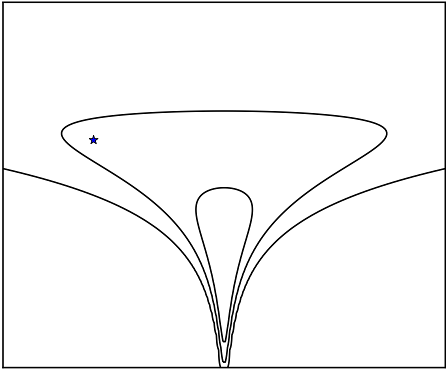
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



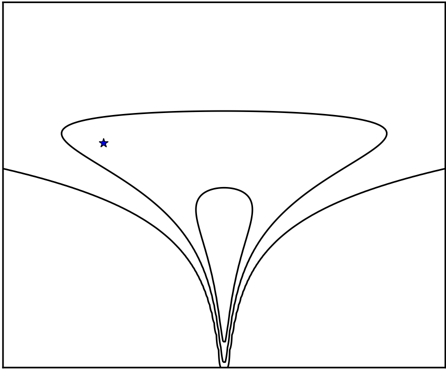
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



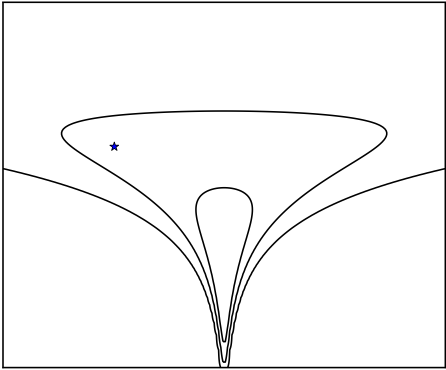
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



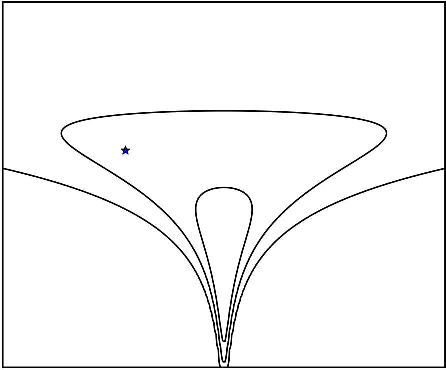
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



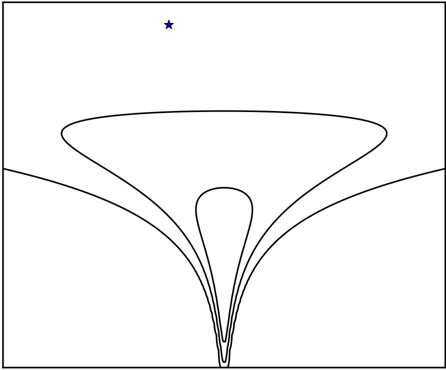
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



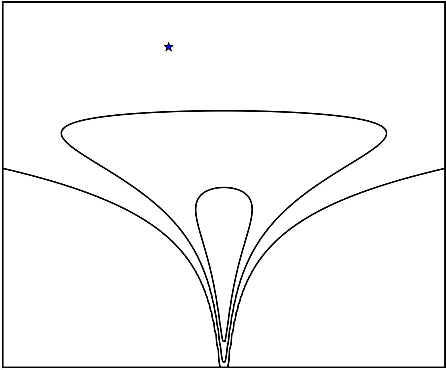
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



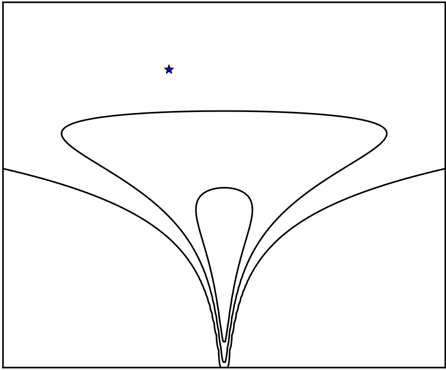
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



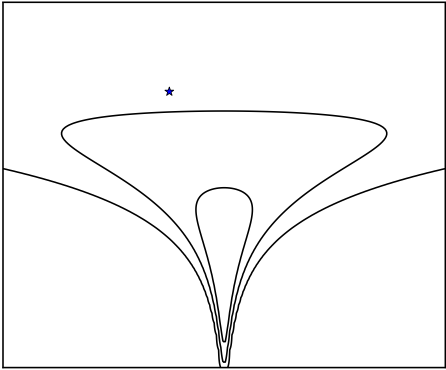
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



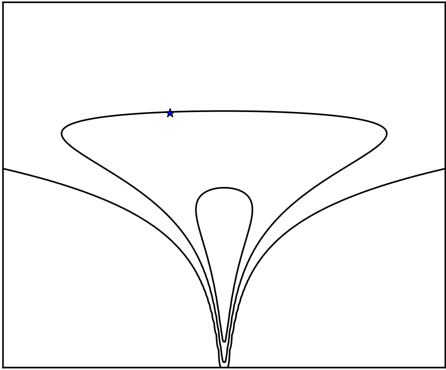
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



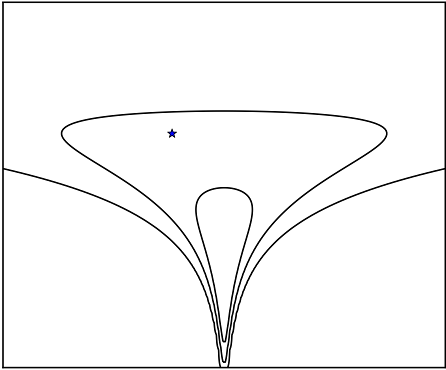
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



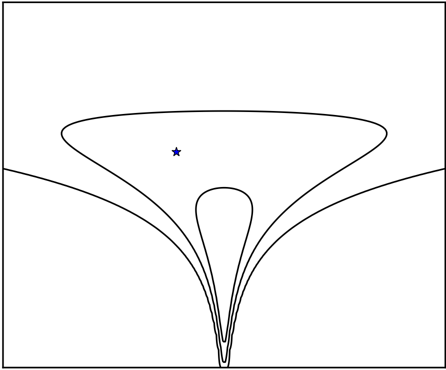
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



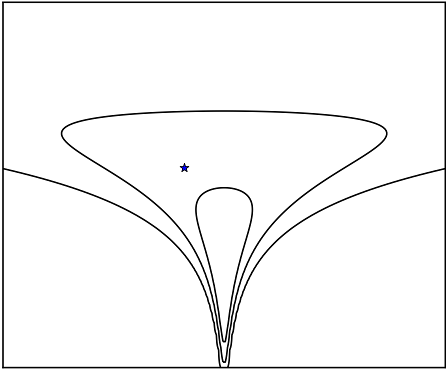
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



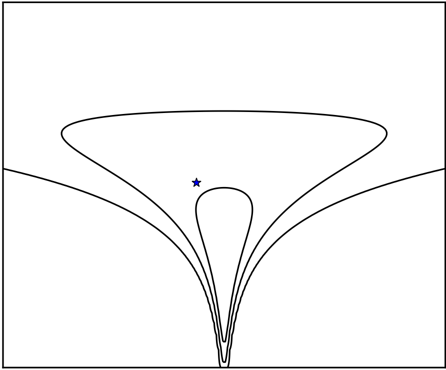
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



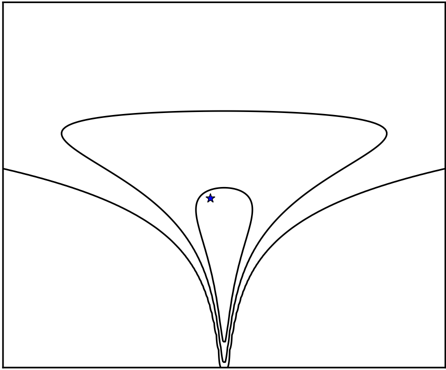
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



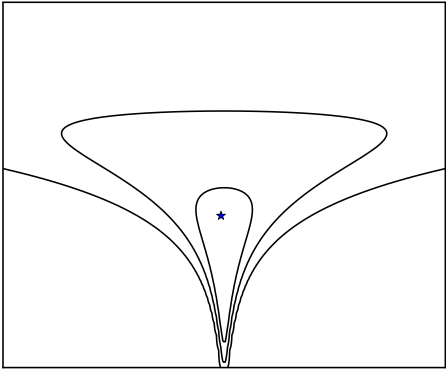
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



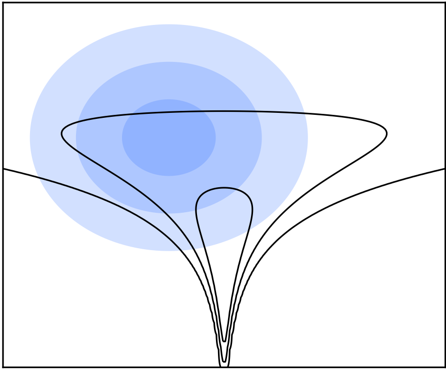
Gradient descent as a sampler

- Optimization paths start from random init, and move towards modes...



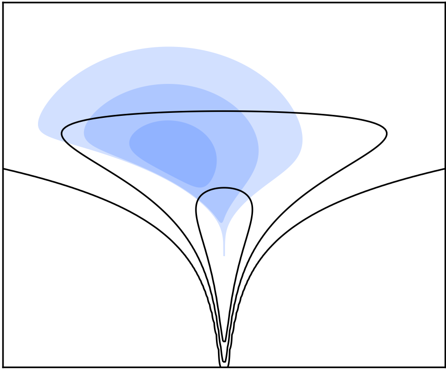
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



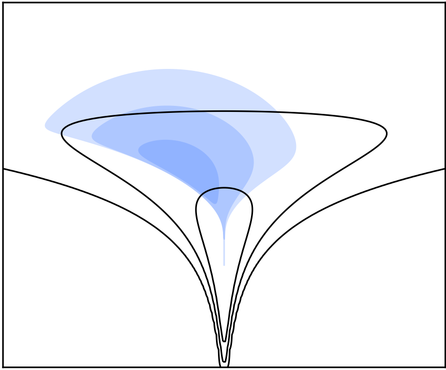
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



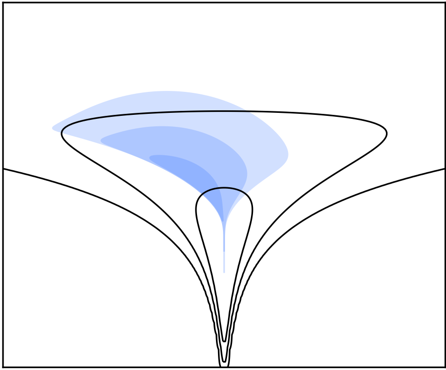
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



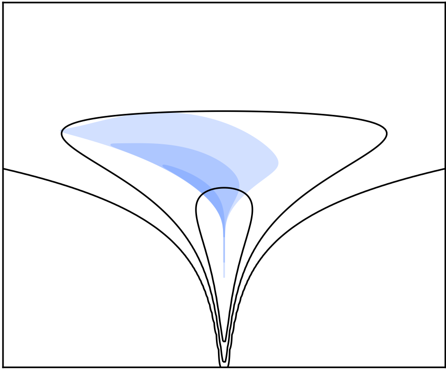
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



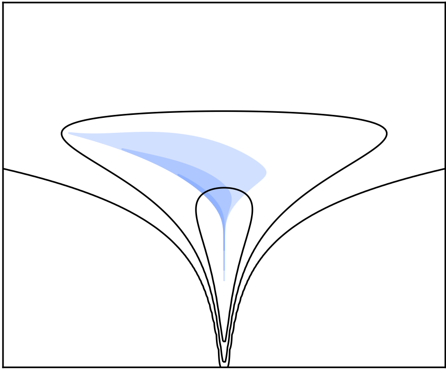
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



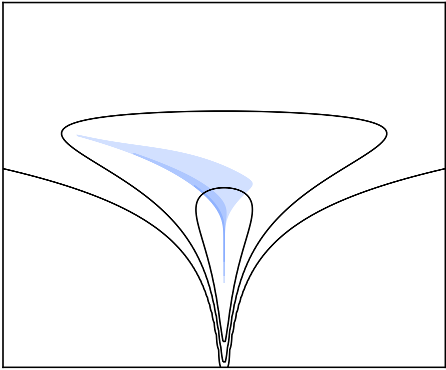
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



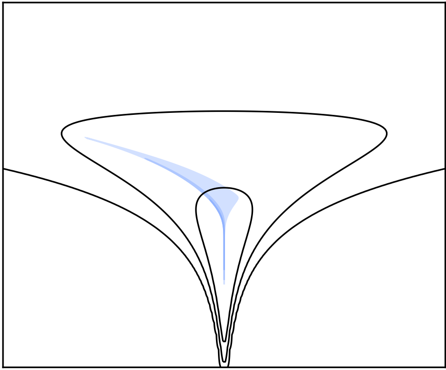
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



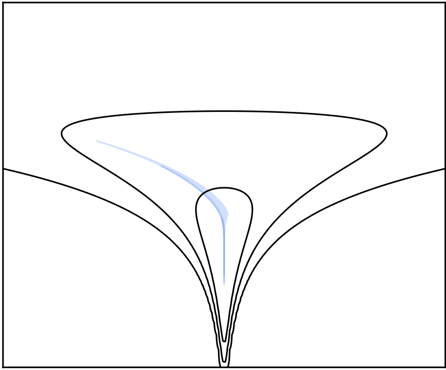
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



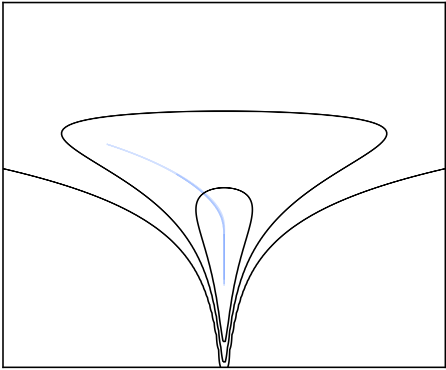
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



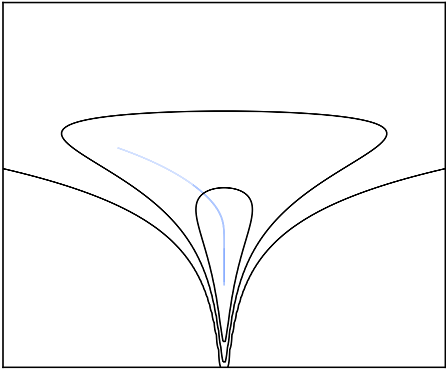
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



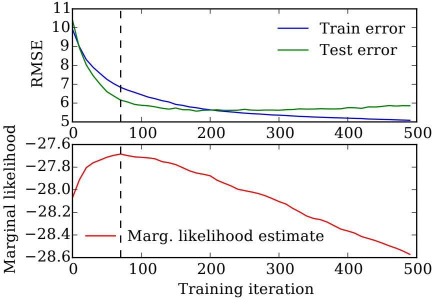
Implicit Distributions

- What about the implicit distribution of parameters after optimizing for t steps?
- Starts as a bad approximation (prior dist)
- Ends as a bad approximation (point mass)
- Ensembling $=$ taking multiple samples from dist
- Early stopping $=$ choosing best intermediate dist



Cross validation vs. marginal likelihood

- What if we could evaluate marginal likelihood of implicit distribution?
- Could choose all hypers to maximize marginal likelihood
- No need for cross-validation?



Variational Lower Bound

$$\log p(\mathbf{x}) \geq \underbrace{-\mathbb{E}_{q(\theta)}\left[-\log p(\theta, \mathbf{x})\right]}_{\text{Energy } E[q]} \quad \underbrace{-\mathbb{E}_{q(\theta)}\left[\log q(\theta)\right]}_{\text{Entropy } S[q]}$$

Energy estimated from optimized objective function (training loss is NLL):

$$\mathbb{E}_{q(\theta)}\left[-\log p(\theta, \mathbf{x})\right] \approx -\log p(\hat{\theta}_T, \mathbf{x})$$

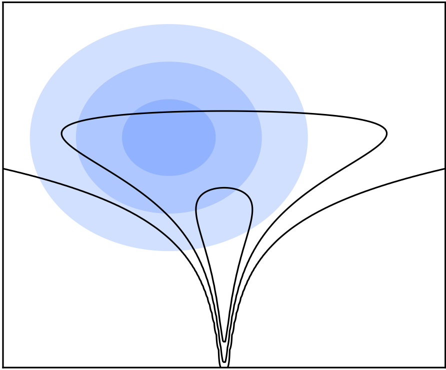
Entropy estimated by tracking change at each iteration:

$$-\mathbb{E}_{q(\theta)}\left[\log q(\theta)\right] \approx S[q_0] + \sum_{t=0}^{T-1} \log \left| J(\hat{\theta}_t) \right|$$

Using a single sample!

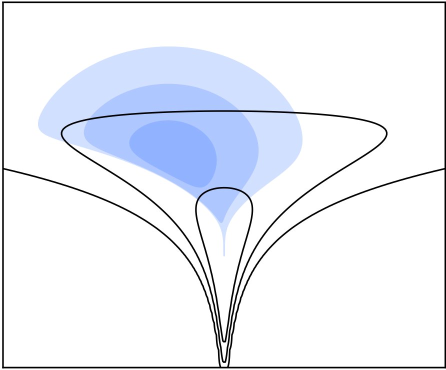
Estimating change in entropy

- Inuitively: High curvature makes entropy decrease quickly
- Can measure local curvature with Hessian
- Approximation good for small step-sizes



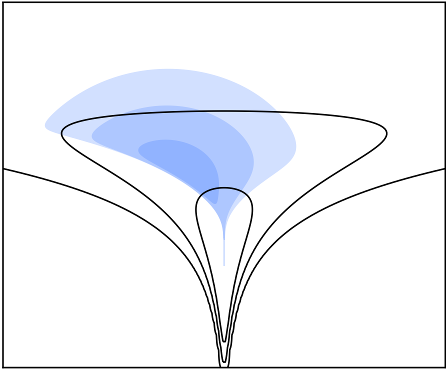
Estimating change in entropy

- Inuitively: High curvature makes entropy decrease quickly
- Can measure local curvature with Hessian
- Approximation good for small step-sizes



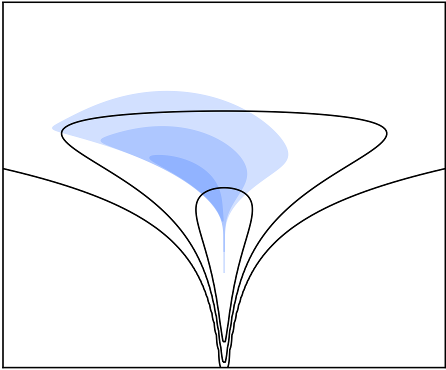
Estimating change in entropy

- Inuitively: High curvature makes entropy decrease quickly
- Can measure local curvature with Hessian
- Approximation good for small step-sizes



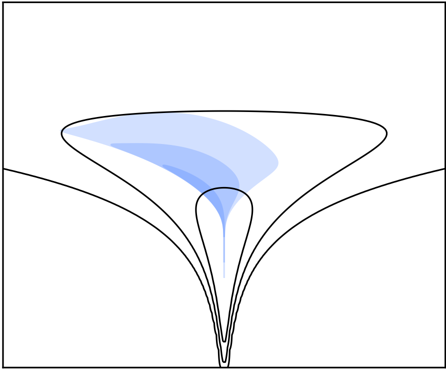
Estimating change in entropy

- Inuitively: High curvature makes entropy decrease quickly
- Can measure local curvature with Hessian
- Approximation good for small step-sizes



Estimating change in entropy

- Inuitively: High curvature makes entropy decrease quickly
- Can measure local curvature with Hessian
- Approximation good for small step-sizes



Estimating change in entropy

Volume change given by Jacobian of optimizer's operator:

$$S[q_{t+1}] - S[q_t] = \mathbb{E}_{q_t(\theta_t)}\left[\log\left|J(\theta_t)\right|\right]$$

Gradient descent update rule:

$$\theta_{t+1} = \theta_t - \alpha \nabla L(\theta),$$

Has Jacobian:

$$J(\theta_t) = I - \alpha \nabla\nabla L(\theta_t)$$

Entropy change estimated at a single sample:

$$S[q_{t+1}] - S[q_t] \approx \log\left|I - \alpha \nabla\nabla L(\theta_t)\right|$$

Final algorithm

Stochastic gradient descent

input: Weight init scale σ_0, step size α, negative log-likelihood $L(\theta, t)$
initialize $\theta_0 \sim \mathcal{N}(0, \sigma_0 \mathbf{I}_D)$
for $t = 1$ **to** T **do**
  $\theta_t = \theta_{t-1} - \alpha \nabla L(\theta_t, t)$
output sample θ_T,

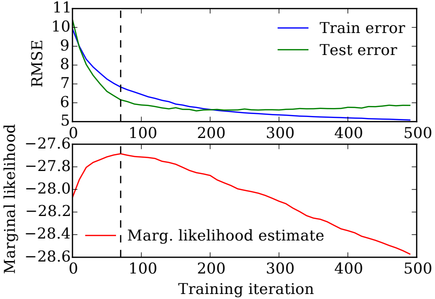
SGD with entropy estimate

input: Weight init scale σ_0, step size α, negative log-likelihood $L(\theta, t)$
initialize $\theta_0 \sim \mathcal{N}(0, \sigma_0 \mathbf{I}_D)$
initialize $S_0 = \frac{D}{2}(1 + \log 2\pi) + D \log \sigma_0$
for $t = 1$ **to** T **do**
  $S_t = S_{t-1} + \log |\mathbf{I} - \alpha \nabla \nabla L(\theta_t, t)|$
  $\theta_t = \theta_{t-1} - \alpha \nabla L(\theta_t, t)$
output sample θ_T, entropy estimate S_T

- Approximate bound: $\log p(\mathbf{x}) \gtrsim -L(\theta_T) + S_T$
- Determinant is $\mathcal{O}(D^3)$
- $\mathcal{O}(D)$ Taylor approximation using Hessian-vector products
- Scales linearly in parameters and dataset size

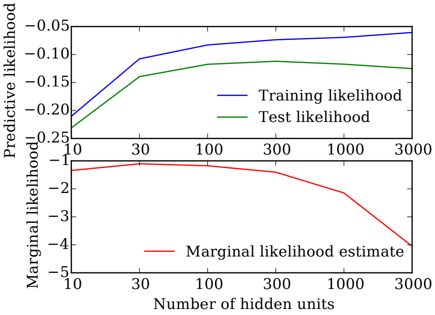
Choosing when to stop

- Neural network on the Boston housing dataset.
- SGD marginal likelihood estimate gives stopping criterion without a validation set



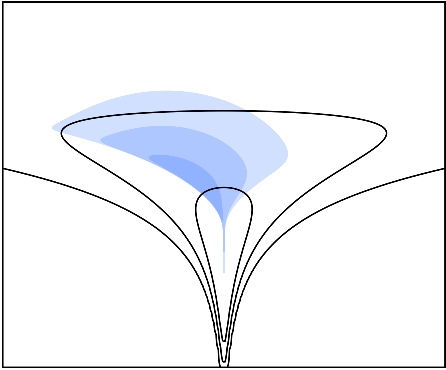
Choosing number of hidden units

- Neural net on 50000 MNIST examples
- Largest model has 2 million parameters
- Gives reasonable estimates, but cross-validation still better



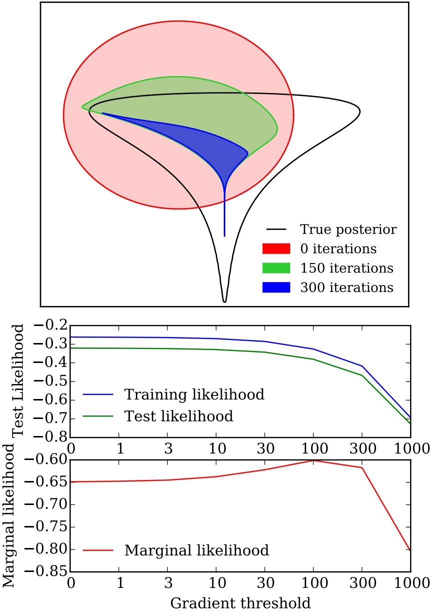
Limitations

- SGD not even trying to maximize lower bound – good approximation is by accident!
- Entropy term gets arbitrarily bad due to concentration, but true performance only gets as bad as maximum likelihood estimate



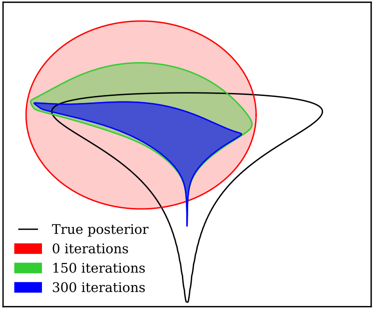
Entropy-friendly optimization

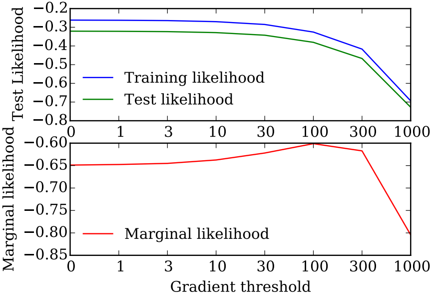
- Modified SGD to move slower near convergence, optimized new hyperparameter
- Hurts performance, but gives tighter bound
- ideally would match test likelihood



Entropy-friendly optimization

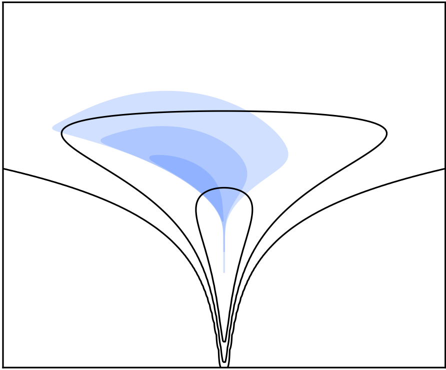
- Modified SGD to move slower near convergence, optimized new hyperparameter
- Hurts performance, but gives tighter bound
- ideally would match test likelihood





Limitations

- Irrelevant parameters can cause low entropy estimate
- No momentum - would need to estimate distribution (see Kingma & Welling, 2015)



Main Takeaways

- Optimization with random restarts implies nonparametric intermediate distributions
- Early stopping chooses among these distributions
- Ensembling samples from them
- Can scalably estimate variational lower bound on model evidence during optimization
- Another connection between practice and theory

Main Takeaways

- Optimization with random restarts implies nonparametric intermediate distributions
- Early stopping chooses among these distributions
- Ensembling samples from them
- Can scalably estimate variational lower bound on model evidence during optimization
- Another connection between practice and theory

Thanks!