# Deep Variational Inference

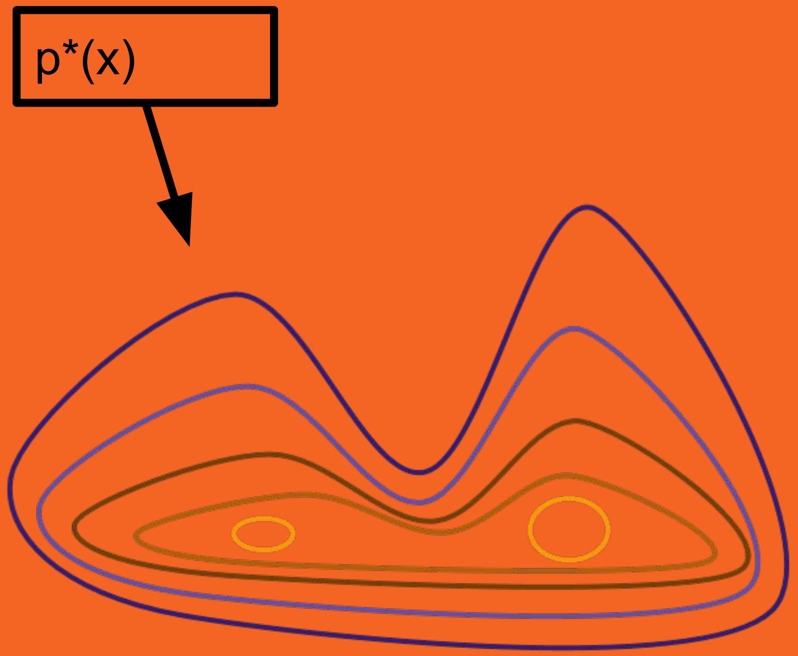
FLARE Reading Group Presentation
Wesley Tansey
9/28/2016

# What is Variational Inference?

- 

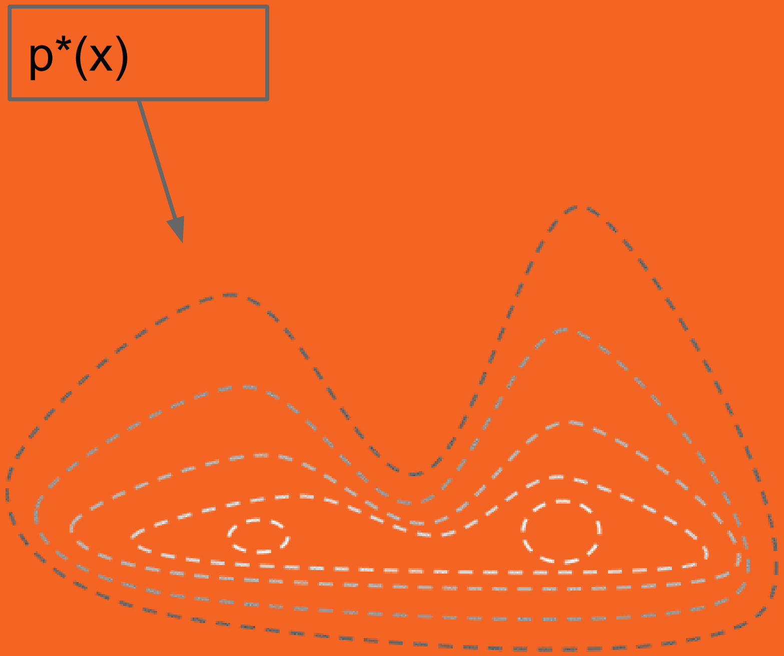
# What is Variational Inference?

- Want to estimate some distribution, $p^*(x)$



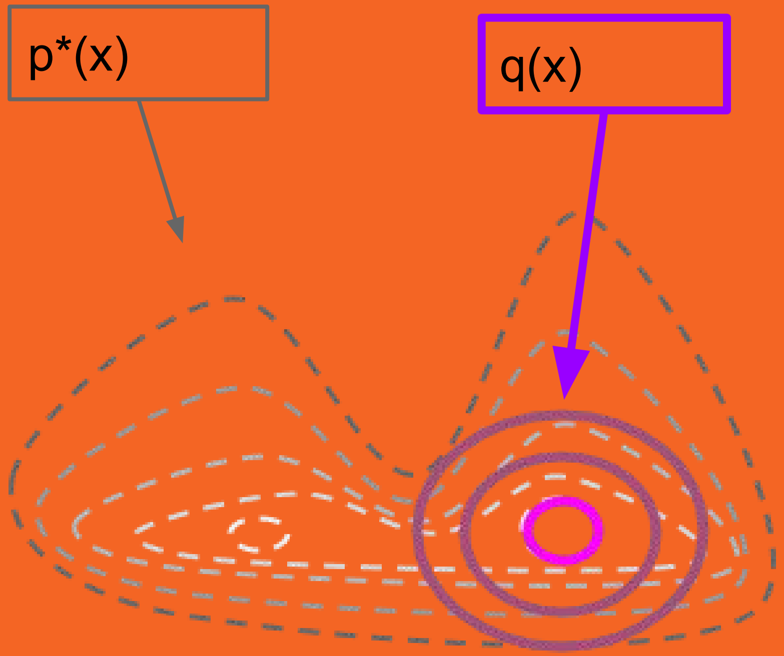
# What is Variational Inference?

- Want to estimate some distribution, p*(x)
- Too expensive to estimate



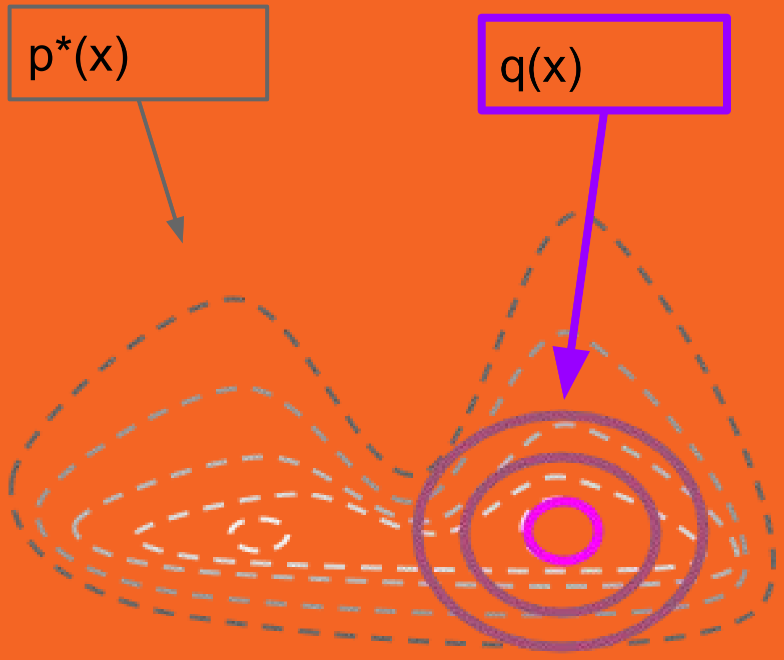
# What is Variational Inference?

- Want to estimate some distribution, p*(x)
- Too expensive to estimate
- Approximate it with a tractable distribution, q(x)



# What is Variational Inference?

- Fit q(x) inside of p*(x)
- Centered at a single mode
  - q(x) is unimodal here
  - VI is a MAP estimate



# What is Variational Inference?

- Mathematically:

$KL(q \,||\, p^*)$

$= \Sigma_x q(x) \log(q(x) / p^*(x))$

Still hard!

$p^*(x)$ usually has a tricky normalizing constant

# What is Variational Inference?

- Mathematically:

$KL(q \,||\, p^*)$

$= \Sigma_x q(x) \log(q(x) \,/\, p^*(x))$

- Use unnormalized p˜ instead

# What is Variational Inference?

- Mathematically:

$KL(q \,||\, p^*)$

$= \Sigma_x q(x) \log(q(x) / p^*(x))$

- Use unnormalized $\tilde{p}$ instead

$\log(q(x) / p^*(x))$

$= \log(q(x)) - \log(p^*(x))$

$= \log(q(x)) - \log(\tilde{p}(x) / Z)$

$= \log(q(x)) - \log(\tilde{p}(x)) - \log(Z)$

# What is Variational Inference?

- Mathematically:

$KL(q \,||\, p^*)$

$= \Sigma_x q(x) \log(q(x) / p^*(x))$

- Use unnormalized p˜ instead

$\log(q(x) / p^*(x))$

$= \log(q(x)) - \log(p^*(x))$

$= \log(q(x)) - \log(\tilde{p}(x) / Z)$

$= \log(q(x)) - \log(\tilde{p}(x)) - \log(Z)$

Constant
=> Can ignore in our optimization problem

# Mean Field VI

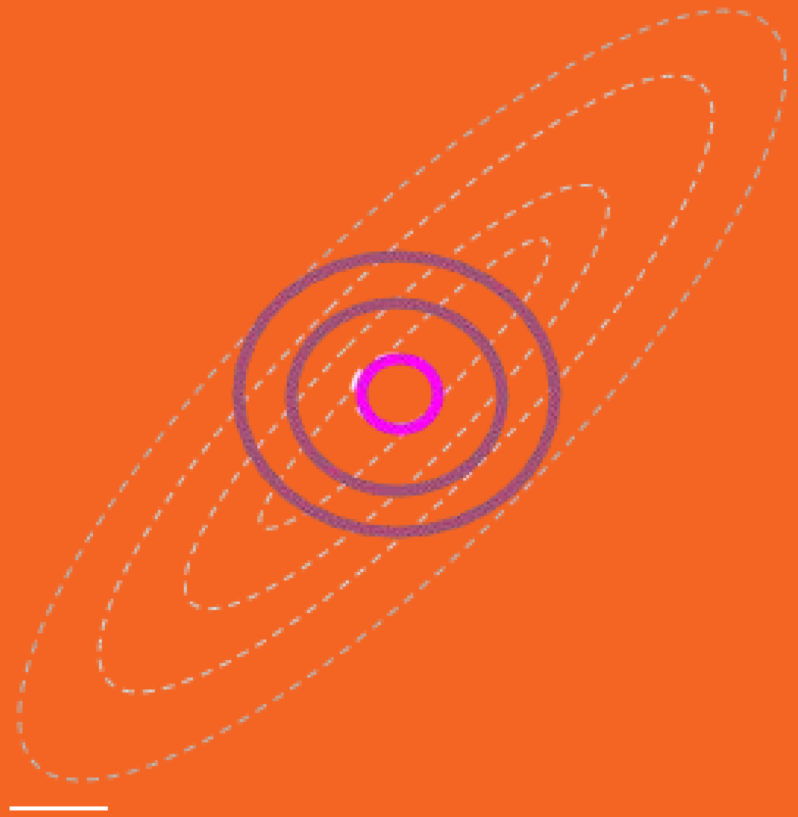
- Classical method
- Uses a factorized q:

$$q(x) = \prod_i q_i(x_i)$$

[1] Blei, Ng, Jordan, "*Latent Dirichlet Allocation*", JMLR, 2003.

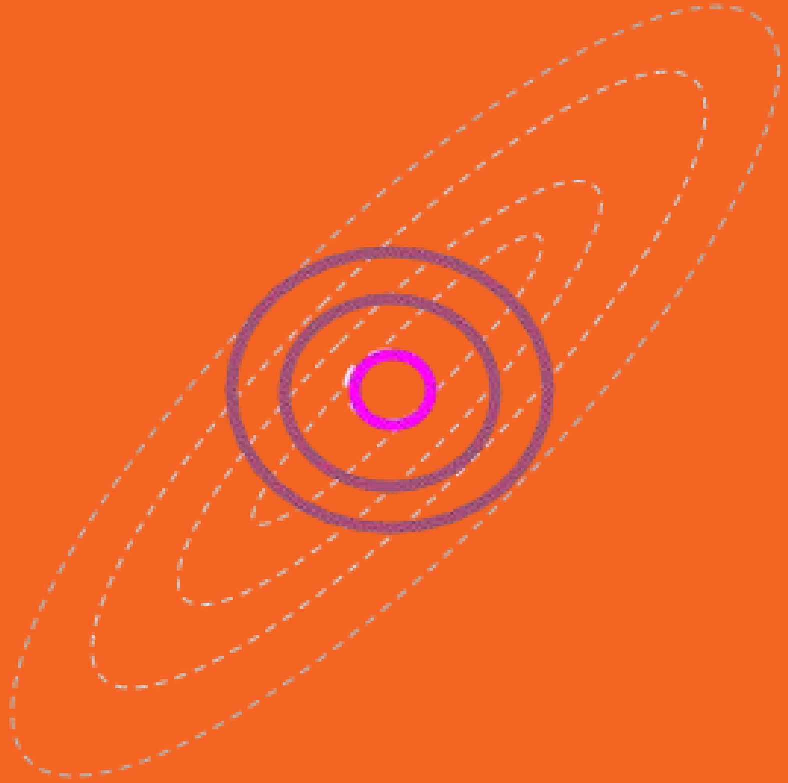
# Mean Field VI

- Example: Multivariate Gaussian
- Product of independent Gaussians for q
- Spherical covariance underestimates true covariance



# Variational Bayes

- Vanilla mean field VI assumes you know all the parameters, θ, of the true distribution, $p^*(x)$

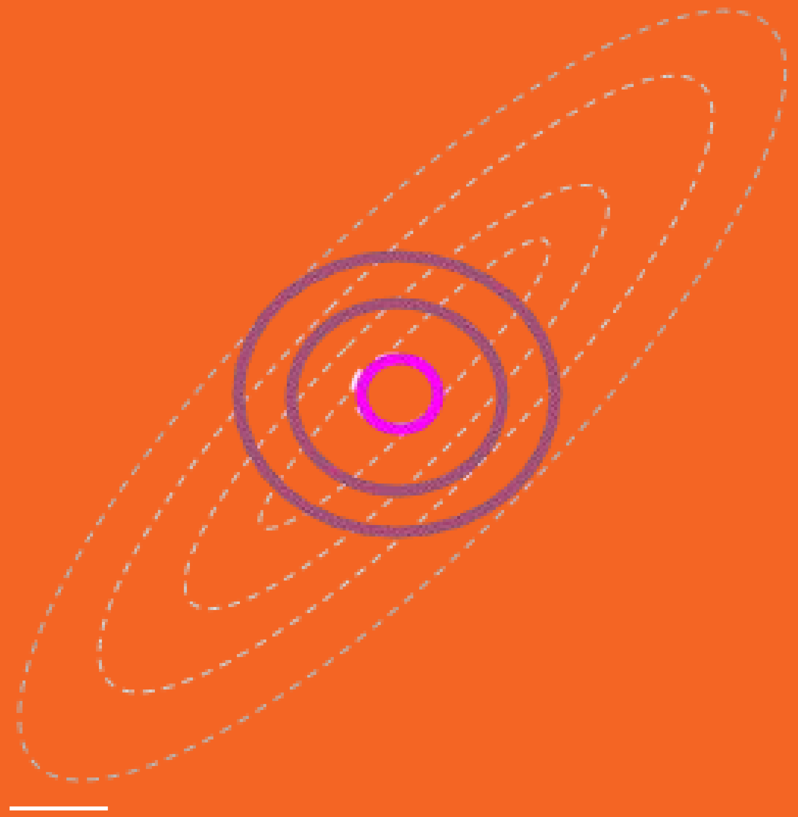


[1] Blei, Ng, Jordan, "*Latent Dirichlet Allocation*", JMLR, 2003.

# Variational Bayes

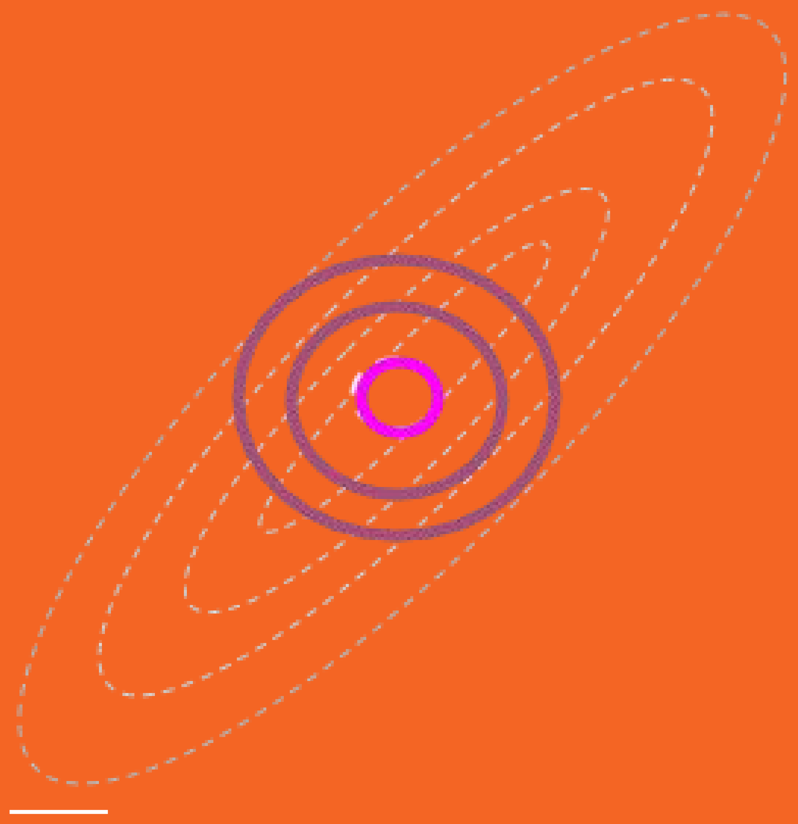
- Vanilla mean field VI assumes you know all the parameters, θ, of the true distribution, p*(x)
- Enter: Variational Bayes (VB)

[1] Blei, Ng, Jordan, "*Latent Dirichlet Allocation*", JMLR, 2003.



# Variational Bayes

- VB infers both the latent q(x) variables, z, *and* the p*(x) parameters, θ
- VB-EM was popularized for LDA[1]
  - E for z, M for θ

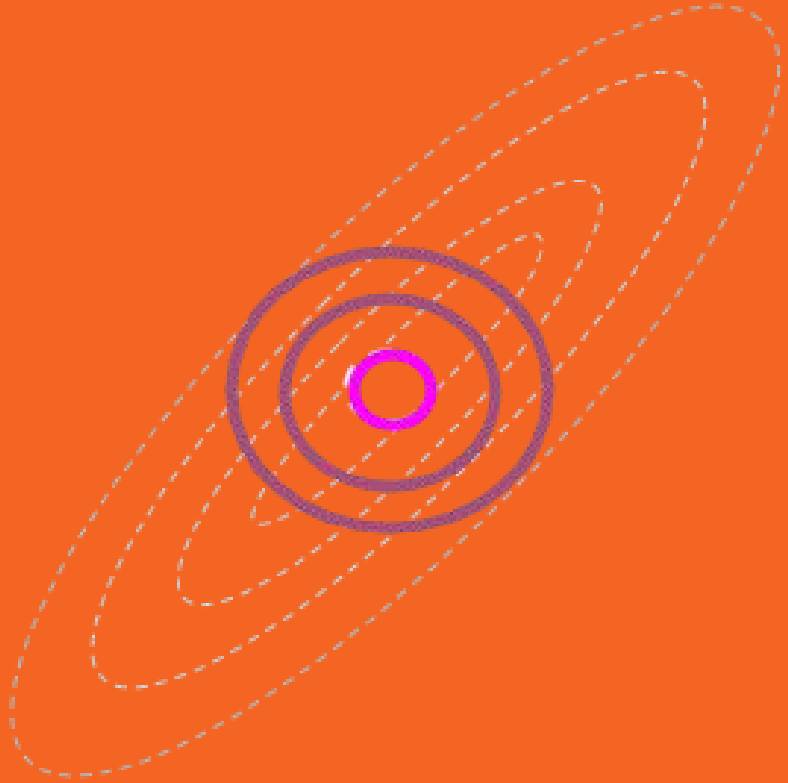


[1] Blei, Ng, Jordan, "*Latent Dirichlet Allocation*", JMLR, 2003.

# Variational Bayes

- VB usually uses a mean field approximation of the form:

$$q(x) = q(z_i \mid \theta)\prod_i q_i(x_i \mid z_i)$$



# Issues with Mean Field VB

- Requires analytical solutions of expectations w.r.t. $q_i$
  - Intractable in general
- Factored form limits the power of the approximation

# Issues with Mean Field VB

- Requires analytical solutions of expectations w.r.t. $q_i$
  - Intractable in general
- Factored form limits the power of the approximation

Solution:
Auto-Encoding Variational Bayes
(Kingma and Welling, 2013)

# Issues with Mean Field VB

- Requires analytical solutions of expectations w.r.t. $q_i$
  - Intractable in general
- Factored form limits the power of the approximation

Solution:
Auto-Encoding Variational Bayes
(Kingma and Welling, 2014)

Solution:
Variational Inference with Normalizing Flows
(Rezende and Mohamed, 2015)

# Auto-Encoding Variational Bayes[1]

High-level idea:

1) Optimizing the same lower bound that we get in VB

2) Data augmentation trick leads to lower-variance estimator

3) Lots of choices of $q(z|x)$ and $p(z)$ lead to partial closed-form

4) Use a neural network to parameterize $q_\phi(z \mid x)$ and $p_\theta(x \mid z)$

5) SGD to fit everything

[1] Kingma and Welling, "*Auto-Encoding Variational Bayes*", ICLR, 2014.

# 1) VB Lower Bound

- Given N iid data points, $(x^1, \ldots, x^n)$

- Maximize the marginal likelihood:

$$\log p_\theta(x^1,\ldots,x^n) = \Sigma_i \log p_\theta(x^{(i)})$$

# 1) VB Lower Bound

- Given N iid data points, $(x^1, \ldots, x^n)$

- Maximize the marginal likelihood:

$$\log p_\theta(x^1,\ldots,x^n) = \Sigma_i \log p_\theta(x^{(i)})$$

$$\log p_\theta(\mathbf{x}^{(i)})$$

$$= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{x}^{(i)}|\mathbf{z}))$$

$$+\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$$

# 1) VB Lower Bound

- Given N iid data points, $(x^1, \ldots, x^n)$
- Maximize the marginal likelihood:

$$\log p_\theta(x^1,\ldots,x^n) = \Sigma_i \log p_\theta(x^{(i)})$$

$$\log p_\theta(\mathbf{x}^{(i)})$$

$$= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}|\mathbf{x}^{(i)}))$$

$$+\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$$

Always positive

# 1) VB Lower Bound

- Given N iid data points, $(x^1, \ldots, x^n)$
- Maximize the marginal likelihood:

$$\log p_\theta(x^1,\ldots,x^n) = \Sigma_i \log p_\theta(x^{(i)})$$

$$\log p_\theta(\mathbf{x}^{(i)})$$

$$= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{x}^{(i)}|\mathbf{z}))$$

$$+\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$$

Lower bound

Always positive

# 1) VB Lower Bound

- Write lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

# 1) VB Lower Bound

- Write lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

Anyone want the derivation?

# 1) VB Lower Bound

- Write lower bound
- Rewrite lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

# 1) VB Lower Bound

- Write lower bound
- Rewrite lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

$$= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$

# 1) VB Lower Bound

- Write lower bound
- Rewrite lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

$$= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$

Derivation?

# 1) VB Lower Bound

- Write lower bound
- Rewrite lower bound
- Monte Carlo gradient estimator of expectation part

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

$$= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[f(\mathbf{z})\right] = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[f(\mathbf{z}) \nabla_{q_\phi} \log q_\phi(\mathbf{z}|\mathbf{x})\right]$$

$$\simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z}^{(l)}) \nabla_{q_\phi} \log q_\phi(\mathbf{z}^{(l)}|\mathbf{x})$$

# 1) VB Lower Bound

- Write lower bound
- Rewrite lower bound
- Monte Carlo gradient estimator of expectation part
  - Too high variance

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) =$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right]$$

$$= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[f(\mathbf{z})\right] = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[f(\mathbf{z})\nabla_{q_\phi} \log q_\phi(\mathbf{z}|\mathbf{x})\right]$$

$$\simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z}^{(l)})\nabla_{q_\phi} \log q_\phi(\mathbf{z}^{(l)}|\mathbf{x})$$

# 2) Reparameterization trick

- Rewrite $q_\phi(z^{(l)} \mid x)$

- Separate q into a deterministic function of **x** and an auxiliary noise variable **ϵ**
- Leads to lower variance estimator

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$$

$$\mathbf{z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$$

$$\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$$

# 2) Reparameterization trick

- Example: univariate Gaussian

- Can rewrite as sum of mean and a scaled noise variable

$$z \sim q_\phi(z|x) = \mathcal{N}(\mu, \sigma^2)$$

$$z = \mu + \sigma\epsilon$$

$$\epsilon \sim \mathcal{N}(0, 1)$$

# 2) Reparameterization trick

- Lots of distributions like this. Three classes given:
    - Tractable inverse CDF
    - Location-scale
    - Composition

Exponential, Cauchy, Logistic, Rayleigh, Pareto, Weibull, Reciprocal, Gompertz, Gumbel, Erlang

Laplace, Elliptical, Student's t, Logistic, Uniform, Triangular, Gaussian

Log-Normal (exponentiated normal)
Gamma (sum of exponentials)
Dirichlet (sum of Gammas)
Beta, Chi-Squared, F

# 2) Reparameterization trick

- Yields a new MC estimator

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[f(\mathbf{z})\right]$$

$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[f(g_\phi(\boldsymbol{\epsilon}, \mathbf{x}))\right]$$

$$\simeq \frac{1}{L}\sum_{l=1}^{L} f(g_\phi(\boldsymbol{\epsilon}^{(l)}, \mathbf{x}))$$

# 2) Reparameterization trick

- Plug estimator into the lower bound eq.
- KL term often can be integrated analytically
  - Careful choice of priors

$$\tilde{\mathcal{L}} = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) + \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{x}|\mathbf{z}^{(l)}))$$

# 2) Reparameterization trick

- Plug estimator into the lower bound eq.

- KL term often can be integrated analytically
  - Careful choice of priors

$$\tilde{\mathcal{L}} = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{x}|\mathbf{z}^{(l)}))$$

# 3) Partial closed form

- KL term often can be integrated analytically
  - Careful choice of priors
  - E.g. both Gaussian

$$\tilde{\mathcal{L}} = -D_{KL}\Big(q_\phi\big(\mathbf{z}\big|\mathbf{x}\big)\Big|\Big|p_\theta\big(\mathbf{z}\big)\Big) + \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\big(\mathbf{x}\big|\mathbf{z}^{(l)}\big)\big)$$

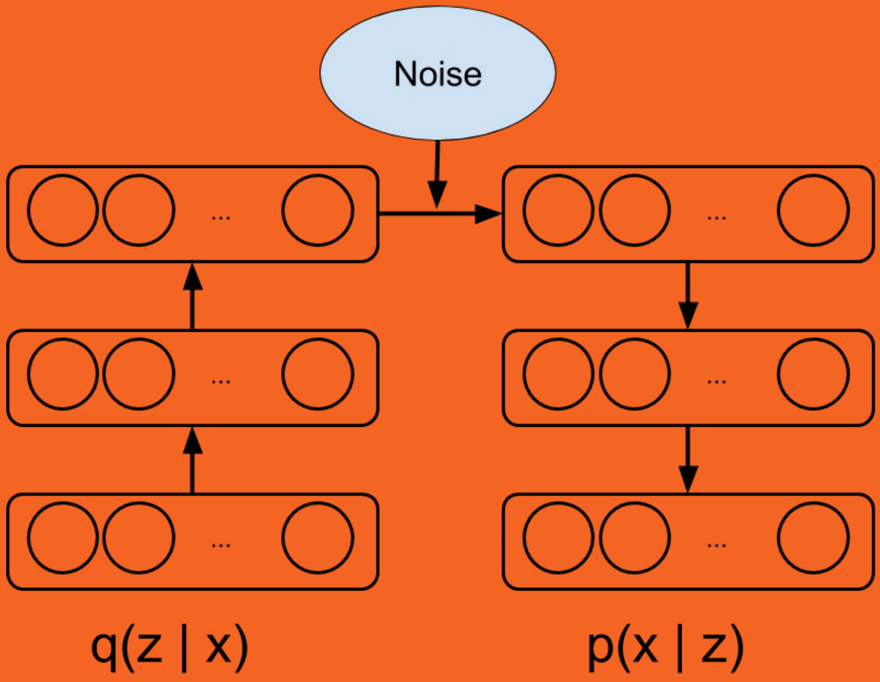
# 4) Auto-encoder connection

- Regularizer
- Reconstruction error
- Neural nets
  - Encode: q(z | x)
  - Decode: p(x | z)

$$\tilde{\mathcal{L}} = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{x}|\mathbf{z}^{(l)}))$$

# 4) Auto-encoder connection (alt.)

- q(z | x) encodes
- p(x | z) decodes
- “Information layer(s)” need to compress
  - Reals = infinite info
  - Reals + random noise = finite info




More info in Karol Gregor’s Deep Mind lecture: https://www.youtube.com/watch?v=P78QYjWh5sM

## Where are we with VI now? (2013'ish)

- Deep networks parameterize both $q(z \mid x)$ and $p(x \mid z)$
- Lower-variance estimator of expected log-likelihood
- Can choose from lots of families of $q(z \mid x)$ and $p(z)$

## Where are we with VI now? (2013'ish)

- Problem:
  - Most parametric families available are simple
  - E.g. product of independent univariate Gaussians
  - Most posteriors are complex

# Variational Inference with Normalizing Flows[1]

High-level idea:

1) VAEs are great, but our posterior $q(z|x)$ needs to be simple

2) Take simple $q(z \mid x)$ and apply series of k transformations to z to get $q_k(z \mid x)$. Metaphor: z “flows” through each transform.

3) Be clever in choice of transforms (computational issue)

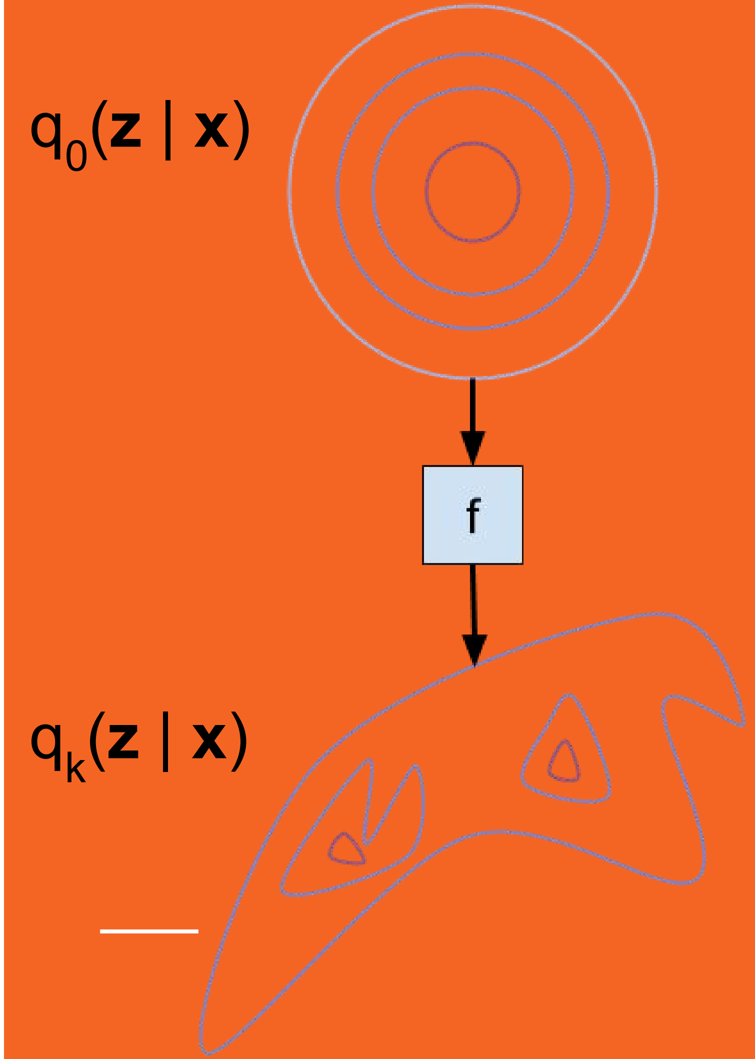
4) Variational posterior q now converges to true posterior p

5) Deep NN now parameterizes q and flow parameters

[1] Rezende, Danilo Jimenez, and Shakir Mohamed. "Variational inference with normalizing flows." *arXiv preprint arXiv:1505.05770* (2015)..

# What is a normalizing flow?

- Function that transforms a probability density through a sequence of invertible mappings



# Key equations (1)

- Chain rule lets us write $q_k$ as product of q0 and inverted determinants

$$q(\mathbf{z}') = q(\mathbf{z}) \left| \det \frac{\partial f^{-1}}{\partial \mathbf{z}'} \right|$$
$$= q(\mathbf{z}) \left| \det \frac{\partial f}{\partial \mathbf{z}} \right|^{-1}$$

# Key equations (2)

- Density $q_k(\mathbf{z}')$ obtained by successively composing k transforms

$$\mathbf{z}_K = f_K \circ \cdots \circ f_2 \circ f_1(\mathbf{z}_0)$$

# Key equations (3)

- Log likelihood of $q_k(\mathbf{z}')$ has a nice additive form

$$\log q_K(\mathbf{z}_K) = \log q_0(\mathbf{z}_0) - \sum_{k=1}^{K} \log \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right|$$

# Key equations (4)

- Expectation over $q_k$ can be written as an expectation under $q_0$
- Cute name: law of the unconscious statistician (LOTUS)

$$\mathbb{E}_{q_K}[h(\mathbf{z})] = \mathbb{E}_{q_0}[h(f_K \circ f_{K-1} \circ \cdots \circ f_1(\mathbf{z}_0))]$$

# Types of flows

1) Infinitesimal Flows:
   - Can show convergence in the limit
   - Skipping (theoretical; computationally expensive)
2) Invertible Linear-Time Flows:
   - log-det can be calculated efficiently

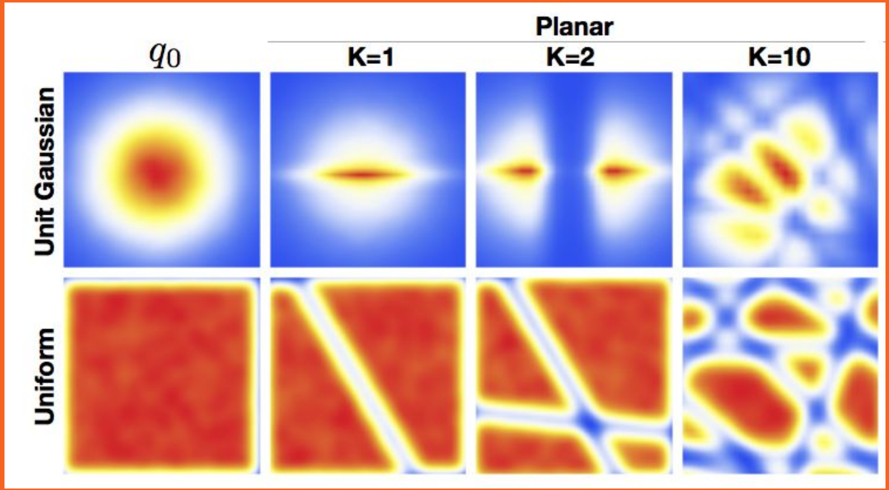
# Planar Flows

- Applies the transform:

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^T\mathbf{z} + b)$$

where:

$$\mathbf{w} \in \mathbb{R}^D, \mathbf{u} \in \mathbb{R}^D, b \in \mathbb{R}$$



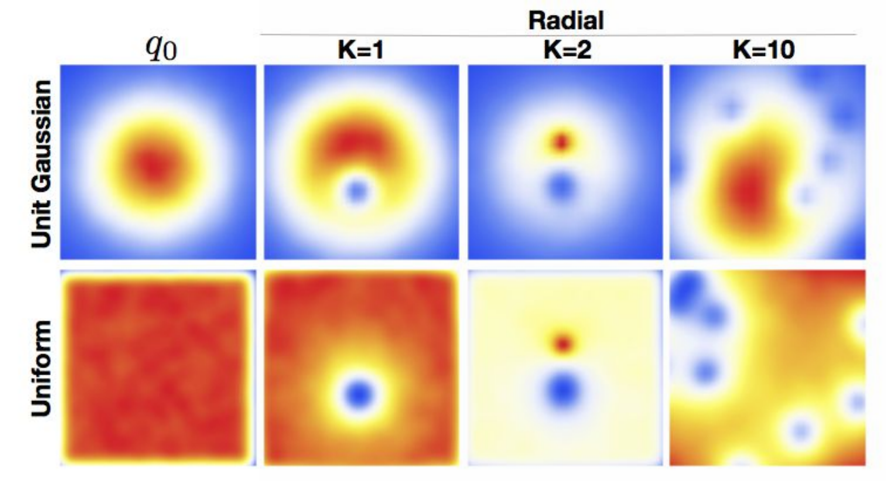
# Radial Flows

- Applies the transform:

$$f(\mathbf{z}) = \mathbf{z} + \beta h(\alpha, r)(\mathbf{z} - \mathbf{z}_0)$$

where:

$$r = |\mathbf{z} - \mathbf{z}_0|$$
$$h(\alpha, r) = 1/(\alpha + r)$$
$$\mathbf{z}_0 \in \mathbb{R}^D, \alpha \in \mathbb{R}^+, \beta \in \mathbb{R}$$



# Summary

- VI approx. p(x) via latent variable model
  - $p(x) = \Sigma_z\, p(z)p(x \mid z)$
- VAE introduces an auto-encoder approach
  - Reparameterization trick makes it feasible
  - Deep NNs parameterize $q(z \mid x)$ and $p(x \mid z)$
- NF takes $q(z|x)$ from simple to complex
  - Series of linear-time transforms
  - Convergence in the limit