# Automatic Differentiation Variational Inference

Philip Schulz and Wilker Aziz

https://github.com/philschulz/VITutorial

# What we know so far

- DGMs:

# What we know so far

- DGMs: probabilistic models parameterised by neural networks

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective:

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)
  - cannot be computed exactly

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)
  - cannot be computed exactly
    we resort to Monte Carlo estimation

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)
  - cannot be computed exactly
    we resort to Monte Carlo estimation
- But the MC estimator is not differentiable

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)
  - cannot be computed exactly
    we resort to Monte Carlo estimation
- But the MC estimator is not differentiable
  - Score function estimator: applicable to any model

# What we know so far

- DGMs: probabilistic models parameterised by neural networks
- Objective: lowerbound on log-likelihood (ELBO)
  - cannot be computed exactly
    we resort to Monte Carlo estimation
- But the MC estimator is not differentiable
  - Score function estimator: applicable to any model
  - Reparameterised gradients
    so far seems applicable only to Gaussian variables

Multivariate calculus recap

Reparameterised gradients revisited

ADVI

Example

# Multivariate calculus recap

Let $x \in \mathbb{R}^K$ and let $\mathcal{T} : \mathbb{R}^K \to \mathbb{R}^K$ be differentiable and invertible

- $y = \mathcal{T}(x)$
- $x = \mathcal{T}^{-1}(y)$

# Jacobian

The Jacobian matrix $J_{\mathcal{T}}(x)$ of $\mathcal{T}$
assessed at $x$ is the matrix of partial derivatives

$$J_{ij} = \frac{\partial y_i}{\partial x_j}$$

# Jacobian

The Jacobian matrix $J_{\mathcal{T}}(x)$ of $\mathcal{T}$
assessed at $x$ is the matrix of partial derivatives

$$J_{ij} = \frac{\partial y_i}{\partial x_j}$$

Inverse function theorem

$$J_{\mathcal{T}^{-1}}(y) = \left(J_{\mathcal{T}}(x)\right)^{-1}$$

# Differential (or inifinitesimal)

The **differential** $\mathrm{d}x$ of $x$
refers to an *infinitely small* change in $x$

# Differential (or inifinitesimal)

The **differential** $\mathrm{d}x$ of $x$
refers to an *infinitely small* change in $x$

We can relate the differential $\mathrm{d}y$ of $y = \mathcal{T}(x)$ to $\mathrm{d}x$

# Differential (or inifinitesimal)

The **differential** $\mathrm{d}x$ of $x$
refers to an *infinitely small* change in $x$

We can relate the differential $\mathrm{d}y$ of $y = \mathcal{T}(x)$ to $\mathrm{d}x$

- Scalar case

$$\mathrm{d}y = \mathcal{T}'(x)\mathrm{d}x = \frac{\mathrm{d}y}{\mathrm{d}x}\mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}x}T(x)\mathrm{d}x$$

where $\mathrm{d}y/\mathrm{d}x$ is the *derivative* of $y$ wrt $x$

# Differential (or inifinitesimal)

The **differential** $\mathrm{d}x$ of $x$
refers to an *infinitely small* change in $x$

We can relate the differential $\mathrm{d}y$ of $y = \mathcal{T}(x)$ to $\mathrm{d}x$

- Scalar case

$$\mathrm{d}y = \mathcal{T}'(x)\mathrm{d}x = \frac{\mathrm{d}y}{\mathrm{d}x}\mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}x}T(x)\mathrm{d}x$$

where $\mathrm{d}y/\mathrm{d}x$ is the *derivative* of $y$ wrt $x$

- Multivariate case

$$\mathrm{d}y = |\det J_{\mathcal{T}}(x)|\mathrm{d}x$$

the absolute value absorbs the orientation

# Integration by substitution

We can integrate a function $g(x)$ by substituting $x = \mathcal{T}^{-1}(y)$

$$\int g(x)\mathrm{d}x$$

# Integration by substitution

We can integrate a function $g(x)$ by substituting $x = \mathcal{T}^{-1}(y)$

$$\int g(x)\mathrm{d}x = \int g(\underbrace{\mathcal{T}^{-1}(y)}_{x})\,\underbrace{|\det J_{\mathcal{T}^{-1}}(y)|\mathrm{d}y}_{\mathrm{d}x}$$

# Integration by substitution

We can integrate a function $g(x)$
by substituting $x = \mathcal{T}^{-1}(y)$

$$\int g(x)\mathrm{d}x = \int g(\underbrace{\mathcal{T}^{-1}(y)}_{x}) \underbrace{|\det J_{\mathcal{T}^{-1}}(y)|\mathrm{d}y}_{\mathrm{d}x}$$

and similarly for a function $h(y)$

$$\int h(y)\mathrm{d}y$$

# Integration by substitution

We can integrate a function $g(x)$
by substituting $x = \mathcal{T}^{-1}(y)$

$$\int g(x)\mathrm{d}x = \int g(\underbrace{\mathcal{T}^{-1}(y)}_{x})\underbrace{|\det J_{\mathcal{T}^{-1}}(y)|\mathrm{d}y}_{\mathrm{d}x}$$

and similarly for a function $h(y)$

$$\int h(y)\mathrm{d}y = \int h(\mathcal{T}(x))|\det J_{\mathcal{T}}(x)|\mathrm{d}x$$

# Change of density

Let $X$ take on values in $\mathbb{R}^K$ with density $p_X(x)$

# Change of density

Let $X$ take on values in $\mathbb{R}^K$ with density $p_X(x)$ and recall that $y = \mathcal{T}(x)$ and $x = \mathcal{T}^{-1}(y)$

# Change of density

Let $X$ take on values in $\mathbb{R}^K$ with density $p_X(x)$ and recall that $y = \mathcal{T}(x)$ and $x = \mathcal{T}^{-1}(y)$

Then $\mathcal{T}$ induces a density $p_Y(y)$ expressed as

$$p_Y(y) = p_X(\mathcal{T}^{-1}(y))|\det J_{\mathcal{T}^{-1}}(y)|$$

# Change of density

Let $X$ take on values in $\mathbb{R}^K$ with density $p_X(x)$ and recall that $y = \mathcal{T}(x)$ and $x = \mathcal{T}^{-1}(y)$

Then $\mathcal{T}$ induces a density $p_Y(y)$ expressed as

$$p_Y(y) = p_X(\mathcal{T}^{-1}(y))|\det J_{\mathcal{T}^{-1}}(y)|$$

and then it follows that

$$p_X(x) = p_Y(\mathcal{T}(x))|\det J_{\mathcal{T}}(x)|$$

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*
a transformation $\mathcal{S}_\lambda : \mathbb{R}^K \to \mathbb{R}^K$ such that

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*
a transformation $\mathcal{S}_\lambda : \mathbb{R}^K \to \mathbb{R}^K$ such that

$$\mathcal{S}_\lambda(z) \sim \pi(\epsilon)$$

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*
a transformation $\mathcal{S}_\lambda : \mathbb{R}^K \to \mathbb{R}^K$ such that

$$\mathcal{S}_\lambda(z) \sim \pi(\epsilon)$$
$$\mathcal{S}_\lambda^{-1}(\epsilon) \sim q(z|\lambda)$$

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*
a transformation $\mathcal{S}_\lambda : \mathbb{R}^K \to \mathbb{R}^K$ such that

$$\mathcal{S}_\lambda(z) \sim \pi(\epsilon)$$
$$\mathcal{S}_\lambda^{-1}(\epsilon) \sim q(z|\lambda)$$

- $\pi(\epsilon)$ does not depend on parameters $\lambda$
  we call it a *base density*

# Revisiting reparameterised gradients

Let $Z$ take on values in $\mathbb{R}^K$ with pdf $q(z|\lambda)$

The idea is to count on a *reparameterisation*
a transformation $\mathcal{S}_\lambda : \mathbb{R}^K \to \mathbb{R}^K$ such that

$$\mathcal{S}_\lambda(z) \sim \pi(\epsilon)$$
$$\mathcal{S}_\lambda^{-1}(\epsilon) \sim q(z|\lambda)$$

- $\pi(\epsilon)$ does not depend on parameters $\lambda$
  we call it a *base density*
- $\mathcal{S}_\lambda(z)$ absorbs dependency on $\lambda$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}[g(z)]$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \int q(z|\lambda) g(z) \mathrm{d}z$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}[g(z)] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}[g(z)] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

$$= \int \pi(\epsilon)$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

$$= \int \pi(\epsilon) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|^{-1}}_{\text{inv func theorem}}$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}[g(z)] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

$$= \int \pi(\epsilon) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|^{-1}}_{\text{inv func theorem}} g(\underbrace{\mathcal{S}_\lambda^{-1}(\epsilon)}_{z})$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}[g(z)] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

$$= \int \pi(\epsilon) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|^{-1}}_{\text{inv func theorem}} g(\underbrace{\mathcal{S}_\lambda^{-1}(\epsilon)}_{z}) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right| \mathrm{d}\epsilon}_{\text{change of var}}$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \int q(z|\lambda) g(z) \mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z)) |\det J_{\mathcal{S}_\lambda}(z)|}_{\text{change of density}} g(z) \mathrm{d}z$$

$$= \int \pi(\epsilon) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|^{-1}}_{\text{inv func theorem}} g(\underbrace{\mathcal{S}_\lambda^{-1}(\epsilon)}_{z}) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|}_{\text{change of var}} \mathrm{d}\epsilon$$

$$= \int \pi(\epsilon) g(\mathcal{S}_\lambda^{-1}(\epsilon)) \mathrm{d}\epsilon$$

# Reparameterised expectations

If we are interested in

$$\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \int q(z|\lambda)g(z)\mathrm{d}z$$

$$= \int \underbrace{\pi(\mathcal{S}_\lambda(z))\left|\det J_{\mathcal{S}_\lambda}(z)\right|}_{\text{change of density}} g(z)\mathrm{d}z$$

$$= \int \pi(\epsilon) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|^{-1}}_{\text{inv func theorem}} g(\underbrace{\mathcal{S}_\lambda^{-1}(\epsilon)}_{z}) \underbrace{\left|\det J_{\mathcal{S}_\lambda^{-1}}(\epsilon)\right|\mathrm{d}\epsilon}_{\text{change of var}}$$

$$= \int \pi(\epsilon)g(\mathcal{S}_\lambda^{-1}(\epsilon))\mathrm{d}\epsilon = \mathbb{E}_{\pi(\epsilon)}\left[g(\mathcal{S}_\lambda^{-1}(\epsilon))\right]$$

# Reparameterised gradients

For optimisation, we need tractable gradients

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|\lambda)} [g(z)] = \frac{\partial}{\partial \lambda} \mathbb{E}_{\pi(\epsilon)} \left[ g(\mathcal{S}_{\lambda}^{-1}(\epsilon)) \right]$$

# Reparameterised gradients

For optimisation, we need tractable gradients

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \frac{\partial}{\partial \lambda}\mathbb{E}_{\pi(\epsilon)}\left[g(\mathcal{S}_{\lambda}^{-1}(\epsilon))\right]$$

since now the density does not depend on $\lambda$, we can obtain a gradient estimate

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \mathbb{E}_{\pi(\epsilon)}\left[\frac{\partial}{\partial \lambda}g(\mathcal{S}_{\lambda}^{-1}(\epsilon))\right]$$

# Reparameterised gradients

For optimisation, we need tractable gradients

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \frac{\partial}{\partial \lambda}\mathbb{E}_{\pi(\epsilon)}\left[g(\mathcal{S}_{\lambda}^{-1}(\epsilon))\right]$$

since now the density does not depend on $\lambda$, we can obtain a gradient estimate

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|\lambda)}\left[g(z)\right] = \mathbb{E}_{\pi(\epsilon)}\left[\frac{\partial}{\partial \lambda} g(\mathcal{S}_{\lambda}^{-1}(\epsilon))\right]$$

$$\overset{\text{MC}}{\approx} \frac{1}{M}\sum_{\substack{i=1 \\ \epsilon_i \sim \pi(\epsilon)}}^{M} \frac{\partial}{\partial \lambda} g(\mathcal{S}_{\lambda}^{-1}(\epsilon_i))$$

# Reparameterised gradients: Gaussian

We have seen one case, namely,
if $\epsilon \sim \mathcal{N}(0, I)$ and $Z \sim \mathcal{N}(\mu, \sigma^2)$

# Reparameterised gradients: Gaussian

We have seen one case, namely,
if $\epsilon \sim \mathcal{N}(0, I)$ and $Z \sim \mathcal{N}(\mu, \sigma^2)$
Then

$$Z \sim \mu + \sigma\epsilon$$

and

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{\mathcal{N}(z|\mu,\sigma^2)}\left[g(z)\right]$$

# Reparameterised gradients: Gaussian

We have seen one case, namely,
if $\epsilon \sim \mathcal{N}(0, I)$ and $Z \sim \mathcal{N}(\mu, \sigma^2)$
Then

$$Z \sim \mu + \sigma\epsilon$$

and

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{\mathcal{N}(z|\mu,\sigma^2)}[g(z)] \\ = \mathbb{E}_{\mathcal{N}(0,I)}\left[\frac{\partial}{\partial\lambda}g(z = \mu + \sigma\epsilon)\right]$$

# Reparameterised gradients: Gaussian

We have seen one case, namely,
if $\epsilon \sim \mathcal{N}(0, I)$ and $Z \sim \mathcal{N}(\mu, \sigma^2)$
Then

$$Z \sim \mu + \sigma\epsilon$$

and

$$\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{\mathcal{N}(z|\mu,\sigma^2)}\left[g(z)\right] \\
&= \mathbb{E}_{\mathcal{N}(0,I)}\left[\frac{\partial}{\partial\lambda}g(z = \mu + \sigma\epsilon)\right] \\
&= \mathbb{E}_{\mathcal{N}(0,I)}\left[\frac{\partial}{\partial z}g(z = \mu + \sigma\epsilon)\frac{\partial z}{\partial\lambda}\right]
\end{aligned}$$

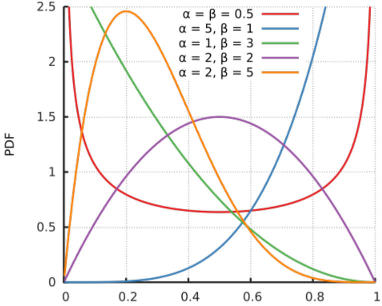
# Beyond

Many interesting densities cannot easily be reparameterised

# Beyond

Many interesting densities cannot easily be reparameterised

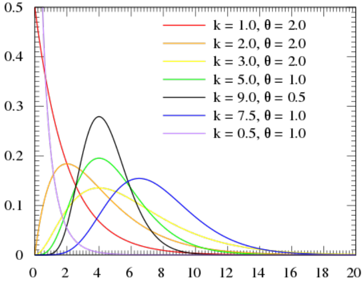
Beta



# Beyond

Many interesting densities cannot easily be reparameterised

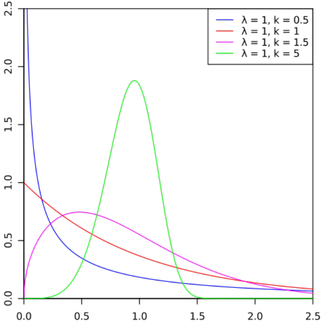
Gamma



# Beyond

Many interesting densities cannot easily be reparameterised

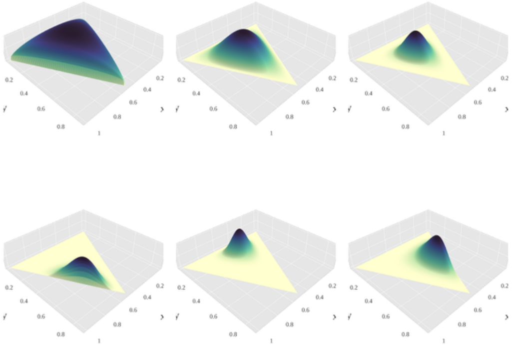
Weibull



# Beyond

Many interesting densities cannot easily be reparameterised

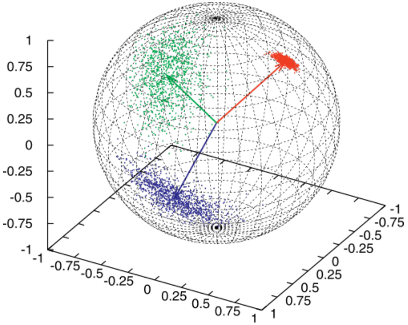
Dirichlet



# Beyond

Many interesting densities cannot easily be reparameterised

von Mises-Fisher



# Automatic Differentiation VI

Motivation

- many models have intractable posteriors
  their normalising constants (evidence) lack analytic solutions

# Automatic Differentiation VI

Motivation

- many models have intractable posteriors
  their normalising constants (evidence) lack analytic solutions
- but many models are differentiable
  that's the main constraint for using NNs

# Automatic Differentiation VI

Motivation

- many models have intractable posteriors
  their normalising constants (evidence) lack analytic solutions
- but many models are differentiable
  that's the main constraint for using NNs

Reparameterised gradients are a step towards automatising VI for differentiable models

# Automatic Differentiation VI

Motivation

- many models have intractable posteriors
  their normalising constants (evidence) lack analytic solutions
- but many models are differentiable
  that's the main constraint for using NNs

Reparameterised gradients are a step towards automatising VI for differentiable models

- but not every model of interest employs rvs for which a reparameterisation is known

# Example: Weibull-Poisson model

Suppose we have some ordinal data which we assume to be Poisson-distributed

$$X|z \sim \text{Poisson}(z) \qquad\qquad z \in \mathbb{R}_{>0}$$

# Example: Weibull-Poisson model

Suppose we have some ordinal data which we assume to be Poisson-distributed

$$X|z \sim \text{Poisson}(z) \qquad\qquad z \in \mathbb{R}_{>0}$$

and suppose we want to impose a Weibull prior on the Poisson rate

# Example: Weibull-Poisson model

Suppose we have some ordinal data which we assume to be Poisson-distributed

$$
\begin{aligned}
z|r, k &\sim \text{Weibull}(r, k) \quad & r \in \mathbb{R}_{>0}, k \in \mathbb{R}_{>0} \\
X|z &\sim \text{Poisson}(z) \quad & z \in \mathbb{R}_{>0}
\end{aligned}
$$

and suppose we want to impose a Weibull prior on the Poisson rate

# VI for Weibull-Poisson model

Generative model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$

# VI for Weibull-Poisson model

Generative model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$

Marginal

$$p(x|r, k) = \int_{\mathbb{R}_{>0}} p(z|r, k)p(x|\rho)\mathrm{d}z$$

# VI for Weibull-Poisson model

Generative model

$$p(x, z | r, k) = p(z | r, k) p(x | \rho)$$

Marginal

$$p(x | r, k) = \int_{\mathbb{R}_{>0}} p(z | r, k) p(x | \rho) \mathrm{d}z$$

ELBO

$$\mathbb{E}_{q(z|\lambda)} \left[ \log p(x, z | r, k) \right] + \mathbb{H}\left( q(z) \right)$$

# VI for Weibull-Poisson model

Generative model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$

Marginal

$$p(x|r, k) = \int_{\mathbb{R}_{>0}} p(z|r, k)p(x|\rho)\mathrm{d}z$$

ELBO

$$\mathbb{E}_{q(z|\lambda)}\left[\log p(x, z|r, k)\right] + \mathbb{H}\left(q(z)\right)$$

Can we make $q(z|\lambda)$ Gaussian?

# VI for Weibull-Poisson model

Generative model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$

Marginal

$$p(x|r, k) = \int_{\mathbb{R}_{>0}} p(z|r, k)p(x|\rho)\mathrm{d}z$$

ELBO

$$\mathbb{E}_{q(z|\lambda)}\left[\log p(x, z|r, k)\right] + \mathbb{H}\left(q(z)\right)$$

Can we make $q(z|\lambda)$ Gaussian?

No! $\mathrm{supp}(\mathcal{N}(z|\mu, \sigma^2)) = \mathbb{R}$

# Strategy

Build a change of variable into the model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$
$$= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z)$$

# Strategy

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \text{Weibull}(z \mid r, k)\, \text{Poisson}(x \mid z) \\
&= \text{Weibull}(\underbrace{\qquad}_{z} \mid r, k)\, \text{Poisson}(x \mid \underbrace{\qquad}_{z})
\end{aligned}
$$

# Strategy

Build a change of variable into the model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$
$$= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z)$$
$$= \text{Weibull}(\underbrace{\exp(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\exp(\zeta)}_{z})$$

# Strategy

Build a change of variable into the model

$$
\begin{aligned}
p(x, z|r, k) &= p(z|r, k)p(x|\rho) \\
&= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z) \\
&= \text{Weibull}(\underbrace{\exp(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\exp(\zeta)}_{z})|\det J_{\exp}(\zeta)|
\end{aligned}
$$

# Strategy

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \text{Weibull}(z \mid r, k)\, \text{Poisson}(x \mid z) \\
&= \text{Weibull}(\underbrace{\exp(\zeta)}_{z} \mid r, k)\, \text{Poisson}(x \mid \underbrace{\exp(\zeta)}_{z}) \left|\det J_{\exp}(\zeta)\right|
\end{aligned}
$$

# Strategy

Build a change of variable into the model

$$
\begin{aligned}
&p(x, z|r,k) = p(z|r,k)p(x|\rho) \\
&= \mathrm{Weibull}(z|r,k)\,\mathrm{Poisson}(x|z) \\
&= \mathrm{Weibull}(\underbrace{\exp(\zeta)}_{z}|r,k)\,\mathrm{Poisson}(x|\underbrace{\exp(\zeta)}_{z})|\det J_{\exp}(\zeta)|
\end{aligned}
$$

ELBO

$$
\mathbb{E}_{q(\zeta|\lambda)}[\ldots] + \mathbb{H}\,(q(\zeta))
$$

# Strategy

Build a change of variable into the model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$
$$= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z)$$
$$= \text{Weibull}(\underbrace{\exp(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\exp(\zeta)}_{z})|\det J_{\exp}(\zeta)|$$

ELBO

$$\mathbb{E}_{q(\zeta|\lambda)}[\ldots] + \mathbb{H}(q(\zeta))$$

Can we use a Gaussian approximate posterior?

# Strategy

Build a change of variable into the model

$$p(x, z|r, k) = p(z|r, k)p(x|\rho)$$
$$= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z)$$
$$= \text{Weibull}(\underbrace{\exp(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\exp(\zeta)}_{z})|\det J_{\exp}(\zeta)|$$

ELBO

$$\mathbb{E}_{q(\zeta|\lambda)}[\ldots] + \mathbb{H}(q(\zeta))$$

Can we use a Gaussian approximate posterior? Yes!

# Differentiable models

We focus on *differentiable probability models*

$$p(x, z) = p(x|z)p(z)$$

# Differentiable models

We focus on *differentiable probability models*

$$p(x, z) = p(x|z)p(z)$$

- members of this class have continuous latent variables $z$

# Differentiable models

We focus on *differentiable probability models*

$$p(x, z) = p(x|z)p(z)$$

- members of this class have continuous latent variables $z$
- and the gradient $\nabla_z \log p(x, z)$ is valid within the *support* of the prior $\mathrm{supp}(p(z)) = \{z \in \mathbb{R}^K : p(z) > 0\} \subseteq \mathbb{R}^K$

# Why do we need differentiable models?

Recall the gradient of the ELBO

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \frac{\partial}{\partial \lambda}\mathbb{H}\left(q(z;\lambda)\right)$$

# Why do we need differentiable models?

Recall the gradient of the ELBO

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \frac{\partial}{\partial \lambda}\mathbb{H}\left(q(z;\lambda)\right)$$

Reparameterisation requires $\frac{\partial}{\partial z}$

# Why do we need differentiable models?

Recall the gradient of the ELBO

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \frac{\partial}{\partial\lambda}\mathbb{H}\left(q(z;\lambda)\right)$$

Reparameterisation requires $\frac{\partial}{\partial z}$

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right]$$

# Why do we need differentiable models?

Recall the gradient of the ELBO

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \frac{\partial}{\partial\lambda}\mathbb{H}\left(q(z;\lambda)\right)$$

Reparameterisation requires $\frac{\partial}{\partial z}$

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] = \mathbb{E}_{\pi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p(x, z = \mathcal{S}_{\lambda}^{-1}(\epsilon))\right]$$

# Why do we need differentiable models?

Recall the gradient of the ELBO

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \frac{\partial}{\partial \lambda}\mathbb{H}\left(q(z;\lambda)\right)$$

Reparameterisation requires $\frac{\partial}{\partial z}$

$$\begin{aligned}\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] &= \mathbb{E}_{\pi(\epsilon)}\left[\frac{\partial}{\partial \lambda}\log p(x, z = \mathcal{S}_{\lambda}^{-1}(\epsilon))\right] \\ &= \mathbb{E}_{\pi(\epsilon)}\left[\frac{\partial}{\partial z}\log p(x,z)\frac{\partial}{\partial \lambda}\mathcal{S}_{\lambda}^{-1}(\epsilon)\right]\end{aligned}$$

# VI optimisation problem

Let's focus on the design and optimisation of the variational approximation

$$\arg\min_{q(z)} \mathrm{KL}\left(q(z) \;||\; p(z|x)\right)$$

# VI optimisation problem

Let's focus on the design and optimisation of the variational approximation

$$\arg\min_{q(z)} \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

To automatise the search for a variational approximation $q(z)$ we must ensure that

$$\mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))$$

# VI optimisation problem

Let's focus on the design and optimisation of the variational approximation

$$\arg\min_{q(z)} \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

To automatise the search for a variational approximation $q(z)$ we must ensure that

$$\mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))$$

- otherwise KL is not a real number $\mathrm{KL}\left(q \,||\, p\right) = \mathbb{E}_q\left[\log q\right] - \mathbb{E}_q\left[\log p\right] \stackrel{\text{def}}{=} \infty$

# Support matching constraint

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the posterior

$$\underset{q(z) \in \mathcal{Q}}{\arg\min} \, \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z) : \mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))\}$$

# Support matching constraint

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the posterior

$$\arg\min_{q(z)\in\mathcal{Q}} \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z) : \mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))\}$$

But what is the support of $p(z|x)$?

# Support matching constraint

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the posterior

$$\underset{q(z)\in\mathcal{Q}}{\arg\min}\,\mathrm{KL}\left(q(z)\;||\;p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z) : \mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))\}$$

But what is the support of $p(z|x)$?

- typically the same as the support of $p(z)$

# Support matching constraint

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the posterior

$$\arg\min_{q(z)\in\mathcal{Q}} \mathrm{KL}\left(q(z) \mid\mid p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z) : \mathrm{supp}(q(z)) \subseteq \mathrm{supp}(p(z|x))\}$$

But what is the support of $p(z|x)$?

- typically the same as the support of $p(z)$ as long as $p(x, z) > 0$ if $p(z) > 0$

# Parametric family

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the prior

$$\arg\min_{q(z)\in\mathcal{Q}} \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z;\lambda) : \lambda \in \Lambda, \mathrm{supp}(q(z;\lambda)) \subseteq \mathrm{supp}(p(z))\}$$

# Parametric family

So let's constrain $q(z)$ to a family $\mathcal{Q}$ whose support is included in the support of the prior

$$\arg\min_{q(z)\in\mathcal{Q}} \mathrm{KL}\left(q(z) \,||\, p(z|x)\right)$$

where

$$\mathcal{Q} = \{q(z;\lambda) : \lambda \in \Lambda, \mathrm{supp}(q(z;\lambda)) \subseteq \mathrm{supp}(p(z))\}$$

- a parameter vector $\lambda$ picks out a member of the family

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \Lambda} \mathbb{E}_{q(z;\lambda)} [\log p(x, z)] + \mathbb{H}(q(z; \lambda))$$

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda\in\Lambda} \mathbb{E}_{q(z;\lambda)}\left[\log p(x,z)\right] + \mathbb{H}\left(q(z;\lambda)\right)$$

subject to

$$\mathcal{Q} = \{q(z;\lambda) : \lambda \in \Lambda, \operatorname{supp}(q(z;\lambda)) \subseteq \operatorname{supp}(p(z))\}$$

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \Lambda} \mathbb{E}_{q(z;\lambda)}\left[\log p(x, z)\right] + \mathbb{H}\left(q(z;\lambda)\right)$$

subject to

$$\mathcal{Q} = \{q(z;\lambda) : \lambda \in \Lambda, \mathrm{supp}(q(z;\lambda)) \subseteq \mathrm{supp}(p(z))\}$$

Often there can be two constraints here

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \Lambda} \mathbb{E}_{q(z;\lambda)} [\log p(x, z)] + \mathbb{H}(q(z; \lambda))$$

subject to

$$\mathcal{Q} = \{q(z; \lambda) : \lambda \in \Lambda, \mathrm{supp}(q(z; \lambda)) \subseteq \mathrm{supp}(p(z))\}$$

Often there can be two constraints here

- support matching constraint

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \Lambda} \mathbb{E}_{q(z;\lambda)}\left[\log p(x, z)\right] + \mathbb{H}\left(q(z;\lambda)\right)$$

subject to

$$\mathcal{Q} = \{q(z;\lambda) : \lambda \in \Lambda, \mathrm{supp}(q(z;\lambda)) \subseteq \mathrm{supp}(p(z))\}$$

Often there can be two constraints here

- support matching constraint
- $\Lambda$ may be constrained to a subset of $\mathbb{R}^D$

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \Lambda} \mathbb{E}_{q(z;\lambda)} [\log p(x, z)] + \mathbb{H}(q(z; \lambda))$$

subject to

$$\mathcal{Q} = \{q(z; \lambda) : \lambda \in \Lambda, \operatorname{supp}(q(z; \lambda)) \subseteq \operatorname{supp}(p(z))\}$$

Often there can be two constraints here

- support matching constraint
- $\Lambda$ may be constrained to a subset of $\mathbb{R}^D$
  e.g. univariate Gaussian location lives in $\mathbb{R}$ but scale lives in $\mathbb{R}_{>0}$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$
how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$
from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$
how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$
from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$
how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$
from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$
- $\sigma = \exp(\lambda_\sigma)$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$
how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$
from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$
- $\sigma = \exp(\lambda_\sigma)$ or $\sigma = \text{softplus}(\lambda_\sigma)$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$

how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$

from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$
- $\sigma = \exp(\lambda_\sigma)$ or $\sigma = \text{softplus}(\lambda_\sigma)$

The vMF distribution is parameterised by a unit-norm vector $v$

how can we get $v$ from $\lambda_v \in \mathbb{R}^d$?

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$

how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$ from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$
- $\sigma = \exp(\lambda_\sigma)$ or $\sigma = \text{softplus}(\lambda_\sigma)$

The vMF distribution is parameterised by a unit-norm vector $v$

how can we get $v$ from $\lambda_v \in \mathbb{R}^d$?

- $v = \frac{\lambda_v}{\|\lambda_v\|_2}$

# Parameters in real coordinate space

Consider the Gaussian case: $Z \sim \mathcal{N}(\mu, \sigma)$

how can we obtain $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d_{>0}$ from $\lambda_\mu \in \mathbb{R}^d$ and $\lambda_\sigma \in \mathbb{R}^d$?

- $\mu = \lambda_\mu$
- $\sigma = \exp(\lambda_\sigma)$ or $\sigma = \text{softplus}(\lambda_\sigma)$

The vMF distribution is parameterised by a unit-norm vector $v$

how can we get $v$ from $\lambda_v \in \mathbb{R}^d$?

- $v = \frac{\lambda_v}{\|\lambda_v\|_2}$

It is typically possible to work with unconstrained parameters, it only takes an appropriate activation

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \mathbb{R}^D} \mathbb{E}_{q(z;\lambda)}\left[\log p(x, z)\right] + \mathbb{H}\left(q(z; \lambda)\right)$$

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \mathbb{R}^D} \mathbb{E}_{q(z;\lambda)} [\log p(x, z)] + \mathbb{H}\left(q(z; \lambda)\right)$$

subject to

$$\mathcal{Q} = \{q(z; \lambda) : \lambda \in \mathbb{R}^D, \mathrm{supp}(q(z; \lambda)) \subseteq \mathrm{supp}(p(z))\}$$

There is one constraint left

# Constrained optimisation for the ELBO

We maximise the ELBO

$$\arg\max_{\lambda \in \mathbb{R}^D} \mathbb{E}_{q(z;\lambda)} [\log p(x, z)] + \mathbb{H}(q(z; \lambda))$$

subject to

$$\mathcal{Q} = \{q(z; \lambda) : \lambda \in \mathbb{R}^D, \text{supp}(q(z; \lambda)) \subseteq \text{supp}(p(z))\}$$

There is one constraint left

- support of $q(z; \lambda)$ depends on the choice of prior and thus may be a subset of $\mathbb{R}^K$

# ADVI

A gradient-based black-box VI procedure

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
    - Appropriate transformations of unconstrained parameters!

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
   - Appropriate transformations of unconstrained parameters!
2. Custom $\text{supp}(p(z))$

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
   - Appropriate transformations of unconstrained parameters!
2. Custom $\mathrm{supp}(p(z))$
   - Express $z \in \mathrm{supp}(p(z)) \subseteq \mathbb{R}^K$ as a transformation of some unconstrained $\zeta \in \mathbb{R}^K$

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
    - Appropriate transformations of unconstrained parameters!
2. Custom $\text{supp}(p(z))$
    - Express $z \in \text{supp}(p(z)) \subseteq \mathbb{R}^K$ as a transformation of some unconstrained $\zeta \in \mathbb{R}^K$
    - Pick a variational family over the entire real coordinate space

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
   - Appropriate transformations of unconstrained parameters!
2. Custom $\text{supp}(p(z))$
   - Express $z \in \text{supp}(p(z)) \subseteq \mathbb{R}^K$ as a transformation of some unconstrained $\zeta \in \mathbb{R}^K$
   - Pick a variational family over the entire real coordinate space
   - basically, pick a Gaussian!

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
   - Appropriate transformations of unconstrained parameters!
2. Custom supp($p(z)$)
   - Express $z \in \text{supp}(p(z)) \subseteq \mathbb{R}^K$ as a transformation of some unconstrained $\zeta \in \mathbb{R}^K$
   - Pick a variational family over the entire real coordinate space
   - basically, pick a Gaussian!
3. Intractable expectations

# ADVI

A gradient-based black-box VI procedure

1. Custom parameter space
   - Appropriate transformations of unconstrained parameters!
2. Custom $\mathrm{supp}(p(z))$
   - Express $z \in \mathrm{supp}(p(z)) \subseteq \mathbb{R}^K$ as a transformation of some unconstrained $\zeta \in \mathbb{R}^K$
   - Pick a variational family over the entire real coordinate space
   - basically, pick a Gaussian!
3. Intractable expectations
   - Reparameterised Gradients!

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

Recall that we have a joint density $p(x, z)$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

Recall that we have a joint density $p(x, z)$ which we can use to construct $p(x, \zeta)$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

Recall that we have a joint density $p(x, z)$
which we can use to construct $p(x, \zeta)$

$$p(x, \zeta) = p(x, \underbrace{\qquad\qquad}_{z})|\det J \qquad (\quad)|$$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \mathrm{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

Recall that we have a joint density $p(x, z)$ which we can use to construct $p(x, \zeta)$

$$p(x, \zeta) = p(x, \underbrace{\mathcal{T}^{-1}(\zeta)}_{z})|\det J \quad ( \ )|$$

# Joint model in real coordinate space

Let's introduce an invertible and differentiable transformation

$$\mathcal{T} : \text{supp}(p(z)) \to \mathbb{R}^K$$

and define a transformed variable $\zeta \in \mathbb{R}^K$

$$\zeta = \mathcal{T}(z)$$

Recall that we have a joint density $p(x, z)$ which we can use to construct $p(x, \zeta)$

$$p(x, \zeta) = p(x, \underbrace{\mathcal{T}^{-1}(\zeta)}_{z})|\det J_{\mathcal{T}^{-1}}(\zeta)|$$

# VI in real coordinate space

We can design a posterior approximation whose support is $\mathbb{R}^K$

# VI in real coordinate space

We can design a posterior approximation whose support is $\mathbb{R}^K$

$$q(\zeta; \lambda)$$

# VI in real coordinate space

We can design a posterior approximation whose support is $\mathbb{R}^K$

$$q(\zeta;\lambda) = \underbrace{\prod_{k=1}^{K} q(\zeta_k;\lambda)}_{\text{mean field}}$$

# VI in real coordinate space

We can design a posterior approximation whose support is $\mathbb{R}^K$

$$q(\zeta;\lambda) = \underbrace{\prod_{k=1}^{K} q(\zeta_k;\lambda)}_{\text{mean field}} = \prod_{k=1}^{K} \mathcal{N}(\zeta_k \mid \mu_k, \sigma_k^2)$$

where

- $\mu_k = \lambda_{\mu_k}$ for $\lambda_{\mu_k} \in \mathbb{R}^K$
- $\sigma_k = \text{softplus}(\lambda_{\sigma_k})$ for $\lambda_{\sigma_k} \in \mathbb{R}^K$

# ELBO in real coordinate space

$\log p(x)$

# ELBO in real coordinate space

$$\log p(x) = \log \int p(x, z) \mathrm{d}z$$

# ELBO in real coordinate space

$$\log p(x) = \log \int p(x, z) \mathrm{d}z$$

$$= \log \int p(x, \mathcal{T}^{-1}(\zeta)) \left|\det J_{\mathcal{T}^{-1}}(\zeta)\right| \mathrm{d}\zeta$$

# ELBO in real coordinate space

$$\log p(x) = \log \int p(x, z) \mathrm{d}z$$

$$= \log \int p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)| \mathrm{d}\zeta$$

$$= \log \int q(\zeta) \frac{p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)|}{q(\zeta)} \mathrm{d}\zeta$$

# ELBO in real coordinate space

$$\log p(x) = \log \int p(x, z) \mathrm{d}z$$

$$= \log \int p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)| \mathrm{d}\zeta$$

$$= \log \int q(\zeta) \frac{p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)|}{q(\zeta)} \mathrm{d}\zeta$$

$$\overset{\text{JI}}{\geq} \int q(\zeta) \log \frac{p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)|}{q(\zeta)} \mathrm{d}\zeta$$

# ELBO in real coordinate space

$$
\begin{aligned}
\log p(x) &= \log \int p(x, z)\mathrm{d}z \\
&= \log \int p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)| \mathrm{d}\zeta \\
&= \log \int q(\zeta) \frac{p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)|}{q(\zeta)} \mathrm{d}\zeta \\
&\overset{\text{JI}}{\geq} \int q(\zeta) \log \frac{p(x, \mathcal{T}^{-1}(\zeta)) |\det J_{\mathcal{T}^{-1}}(\zeta)|}{q(\zeta)} \mathrm{d}\zeta \\
&= \mathbb{E}_{q(\zeta)} \left[ \log p(x, \mathcal{T}^{-1}(\zeta)) + \log |\det J_{\mathcal{T}^{-1}}(\zeta)| \right] + \mathbb{H}\left(q(\zeta)\right)
\end{aligned}
$$

# Reparameterised ELBO

Recall that for Gaussians we have a standardisation procedure $\mathcal{S}_\lambda(\zeta) \sim \mathcal{N}(\epsilon|0, I)$

$$\mathbb{E}_{q(\zeta;\lambda)}\left[\log p(x, \mathcal{T}^{-1}(\zeta)) + \log\left|\det J_{\mathcal{T}^{-1}}(\zeta)\right|\right] + \mathbb{H}\left(q(\zeta;\lambda)\right)$$

# Reparameterised ELBO

Recall that for Gaussians we have a standardisation procedure $\mathcal{S}_\lambda(\zeta) \sim \mathcal{N}(\epsilon|0, I)$

$$\mathbb{E}_{q(\zeta;\lambda)}\left[\log p(x, \mathcal{T}^{-1}(\zeta)) + \log\left|\det J_{\mathcal{T}^{-1}}(\zeta)\right|\right] + \mathbb{H}\left(q(\zeta;\lambda)\right)$$

$$= \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[\log p(x, \underbrace{\mathcal{T}^{-1}(\overbrace{\mathcal{S}_\lambda^{-1}(\epsilon)}^{\zeta})}_{z}) + \log\left|\det J_{\mathcal{T}^{-1}}(\mathcal{S}_\lambda^{-1}(\epsilon))\right|\right]$$

$$+ \mathbb{H}\left(q(\zeta;\lambda)\right)$$

# Gradient estimate

For $\epsilon_i \sim \mathcal{N}(0, I)$

$$\frac{\partial}{\partial \lambda} \mathrm{ELBO}(\lambda)$$

# Gradient estimate

For $\epsilon_i \sim \mathcal{N}(0, I)$

$$\frac{\partial}{\partial \lambda} \mathsf{ELBO}(\lambda) \overset{\mathsf{MC}}{\approx} \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial \lambda} \log \underbrace{p(x|\mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{likelihood}}$$

# Gradient estimate

For $\epsilon_i \sim \mathcal{N}(0, I)$

$$\frac{\partial}{\partial \lambda} \text{ELBO}(\lambda) \overset{\text{MC}}{\approx} \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial \lambda} \log \underbrace{p(x | \mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{likelihood}} + \frac{\partial}{\partial \lambda} \log \underbrace{p(\mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{prior}}$$

# Gradient estimate

For $\epsilon_i \sim \mathcal{N}(0, I)$

$$\frac{\partial}{\partial \lambda} \mathrm{ELBO}(\lambda) \overset{\mathrm{MC}}{\approx} \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial \lambda} \log \underbrace{p(x | \mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{likelihood}} + \frac{\partial}{\partial \lambda} \log \underbrace{p(\mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{prior}} + \frac{\partial}{\partial \lambda} \log \underbrace{\left| \det J_{\mathcal{T}^{-1}}(\mathcal{S}_\lambda^{-1}(\epsilon_i)) \right|}_{\text{change of volume}}$$

# Gradient estimate

For $\epsilon_i \sim \mathcal{N}(0, I)$

$$
\begin{aligned}
\frac{\partial}{\partial \lambda} \mathrm{ELBO}(\lambda) \overset{\mathrm{MC}}{\approx} \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial \lambda} \log \underbrace{p(x | \mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{likelihood}} \\
+ \frac{\partial}{\partial \lambda} \log \underbrace{p(\mathcal{T}^{-1}(\mathcal{S}_\lambda^{-1}(\epsilon_i)))}_{\text{prior}} \\
+ \frac{\partial}{\partial \lambda} \log \underbrace{\left| \det J_{\mathcal{T}^{-1}}(\mathcal{S}_\lambda^{-1}(\epsilon_i)) \right|}_{\text{change of volume}} \\
+ \frac{\partial}{\partial \lambda} \underbrace{\mathbb{H}\left(q(\zeta; \lambda)\right)}_{\text{analaytic}}
\end{aligned}
$$

# Practical tips

Many software packages know how to transform the support of various distributions

- Stan
- Tensorflow `tf.probability`
- Pytorch `torch.distributions`

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \mathrm{Weibull}(z \mid r, k)\, \mathrm{Poisson}(x \mid z)
\end{aligned}
$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \text{Weibull}(z \mid r, k)\, \text{Poisson}(x \mid z) \\
&= \text{Weibull}(\underbrace{\qquad\qquad}_{z} \mid r, k)\, \text{Poisson}(x \mid \underbrace{\qquad\qquad}_{z})
\end{aligned}
$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \mathrm{Weibull}(z \mid r, k)\, \mathrm{Poisson}(x \mid z) \\
&= \mathrm{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z} \mid r, k)\, \mathrm{Poisson}(x \mid \underbrace{\log^{-1}(\zeta)}_{z})
\end{aligned}
$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z|r, k) &= p(z|r, k)p(x|\rho) \\
&= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z) \\
&= \text{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\log^{-1}(\zeta)}_{z})\left|\det J_{\log^{-1}}(\zeta)\right|
\end{aligned}
$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z \mid r, k) &= p(z \mid r, k)\, p(x \mid \rho) \\
&= \text{Weibull}(z \mid r, k)\, \text{Poisson}(x \mid z) \\
&= \text{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z} \mid r, k)\, \text{Poisson}(x \mid \underbrace{\log^{-1}(\zeta)}_{z}) \left|\det J_{\log^{-1}}(\zeta)\right| \\
&= p(x, z = \log^{-1}(\zeta)) \left|\det J_{\log^{-1}}(\zeta)\right|
\end{aligned}
$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z|r, k) &= p(z|r, k)p(x|\rho) \\
&= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z) \\
&= \text{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\log^{-1}(\zeta)}_{z})\left|\det J_{\log^{-1}}(\zeta)\right| \\
&= p(x, z = \log^{-1}(\zeta))\left|\det J_{\log^{-1}}(\zeta)\right|
\end{aligned}
$$

ELBO

$$\mathbb{E}_{q(\zeta|\lambda)}\left[\ldots\right] + \mathbb{H}\left(q(\zeta)\right)$$

# Weibull-Poisson model

Build a change of variable into the model

$$
\begin{aligned}
p(x, z|r, k) &= p(z|r, k)p(x|\rho) \\
&= \text{Weibull}(z|r, k)\,\text{Poisson}(x|z) \\
&= \text{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z}|r, k)\,\text{Poisson}(x|\underbrace{\log^{-1}(\zeta)}_{z})\left|\det J_{\log^{-1}}(\zeta)\right| \\
&= p(x, z = \log^{-1}(\zeta))\left|\det J_{\log^{-1}}(\zeta)\right|
\end{aligned}
$$

ELBO

$$
\mathbb{E}_{q(\zeta|\lambda)}\left[\log p(x, z = \log^{-1}(\zeta))\left|\det J_{\log^{-1}}(\zeta)\right|\right] + \mathbb{H}\left(q(\zeta)\right)
$$

# Weibull-Poisson model

Build a change of variable into the model

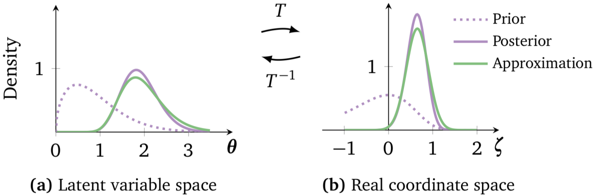
$$
\begin{aligned}
p(x, z | r, k) &= p(z | r, k) p(x | \rho) \\
&= \text{Weibull}(z | r, k) \, \text{Poisson}(x | z) \\
&= \text{Weibull}(\underbrace{\log^{-1}(\zeta)}_{z} | r, k) \, \text{Poisson}(x | \underbrace{\log^{-1}(\zeta)}_{z}) \left| \det J_{\log^{-1}}(\zeta) \right| \\
&= p(x, z = \log^{-1}(\zeta)) \left| \det J_{\log^{-1}}(\zeta) \right|
\end{aligned}
$$

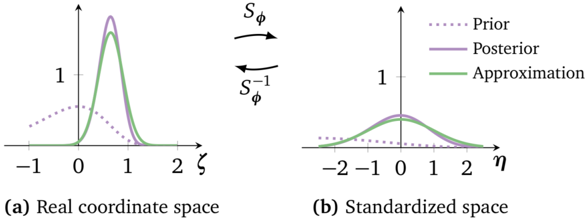
ELBO

$$
\mathbb{E}_{q(\zeta|\lambda)} \left[ \log p(x, z = \log^{-1}(\zeta)) \left| \det J_{\log^{-1}}(\zeta) \right| \right] + \mathbb{H}\left(q(\zeta)\right)
$$

$$
\mathbb{E}_{\phi(\epsilon)} \left[ \log p(x, z = \log^{-1}(\mathcal{S}^{-1}(\epsilon))) \left| \det J_{\log^{-1}}(\mathcal{S}^{-1}(\epsilon)) \right| \right] + \mathbb{H}\left(q(\zeta)\right)
$$

# Visualisation




Images from Kucukelbir et al. (2017)

# Wait... no deep learning?

Sure! Parameters may well be predicted by NNs

- approximate posterior location and scale
- Weibull rate and shape

Everything is now differentiable, reparameterisable, and the optimisation is unconstrained!

# Summary

ADVI is a big step towards blackbox VI

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space
- now we also know how to map latent variables to unconstrained real coordinate space

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space
- now we also know how to map latent variables to unconstrained real coordinate space
- it takes a change of variable built into the model

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space
- now we also know how to map latent variables to unconstrained real coordinate space
- it takes a change of variable built into the model

Think of ADVI as reparameterised gradients and autodiff expanded to many more models!

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space
- now we also know how to map latent variables to unconstrained real coordinate space
- it takes a change of variable built into the model

Think of ADVI as reparameterised gradients and autodiff expanded to many more models!

What's left?

# Summary

ADVI is a big step towards blackbox VI

- we knew how to map parameters to the unconstrained real coordinate space
- now we also know how to map latent variables to unconstrained real coordinate space
- it takes a change of variable built into the model

Think of ADVI as reparameterised gradients and autodiff expanded to many more models!

What's left? Our posteriors are still rather simple, aren't they?

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18(14):1–45, 2017. URL http://jmlr.org/papers/v18/16-107.html.