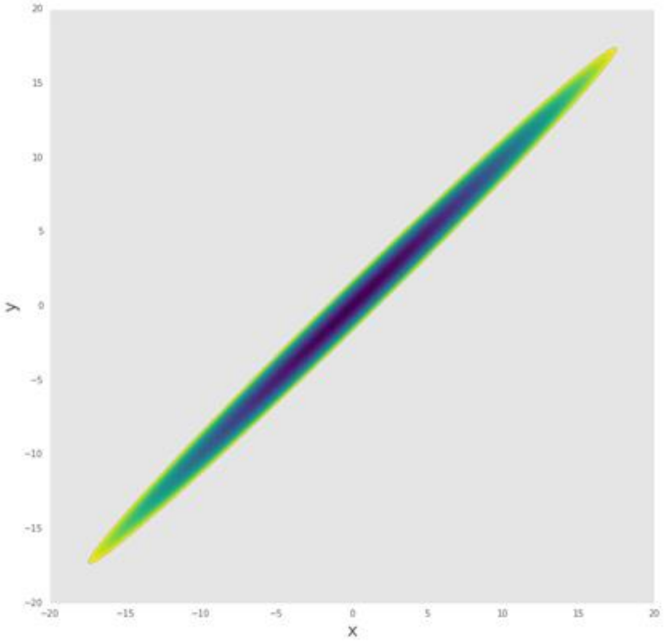
# Automatic Reparameterisation in Probabilistic Programming

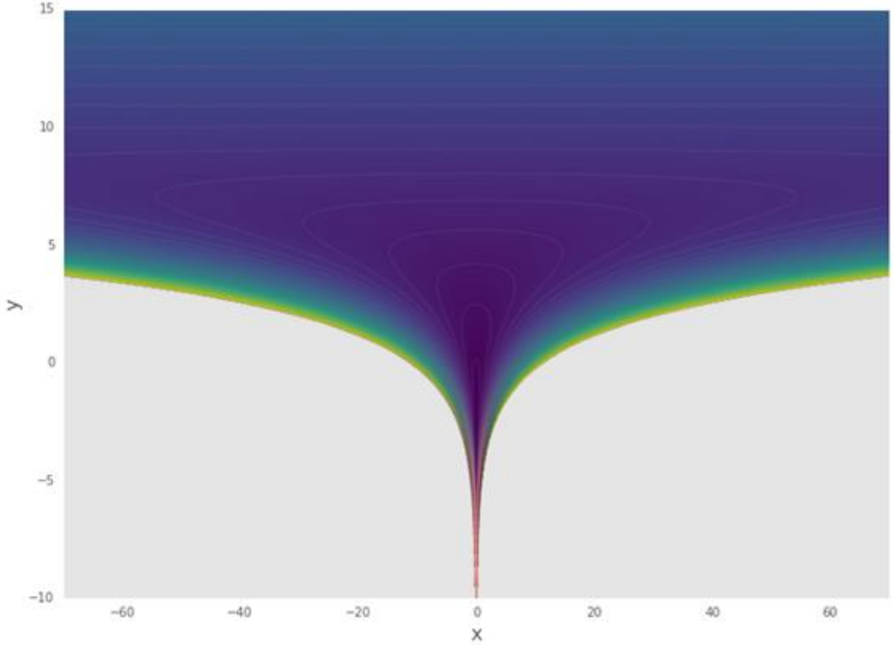
Maria I. Gorinova, Dave Moore, Matthew D. Hoffman

# The Problem

$$y \sim \mathcal{N}(0, 1)$$
$$x \sim \mathcal{N}(y, 0.1)$$

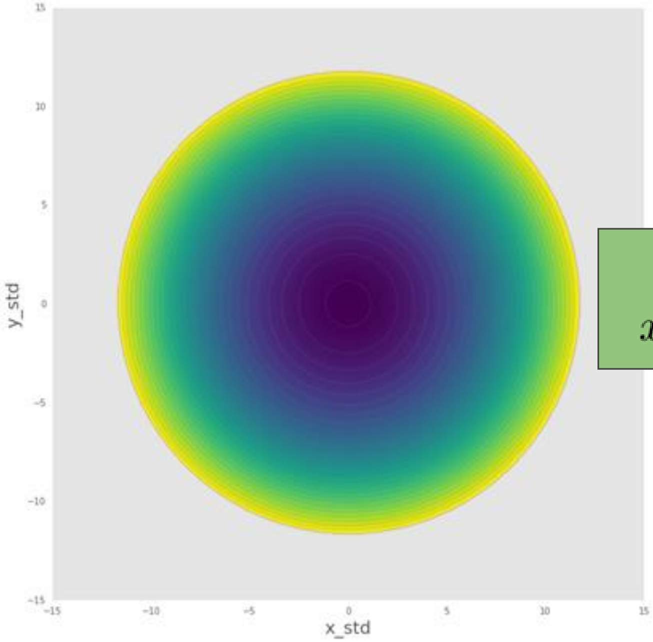



$$y \sim \mathcal{N}(0, 3)$$
$$x \sim \mathcal{N}(y, e^{y/2})$$



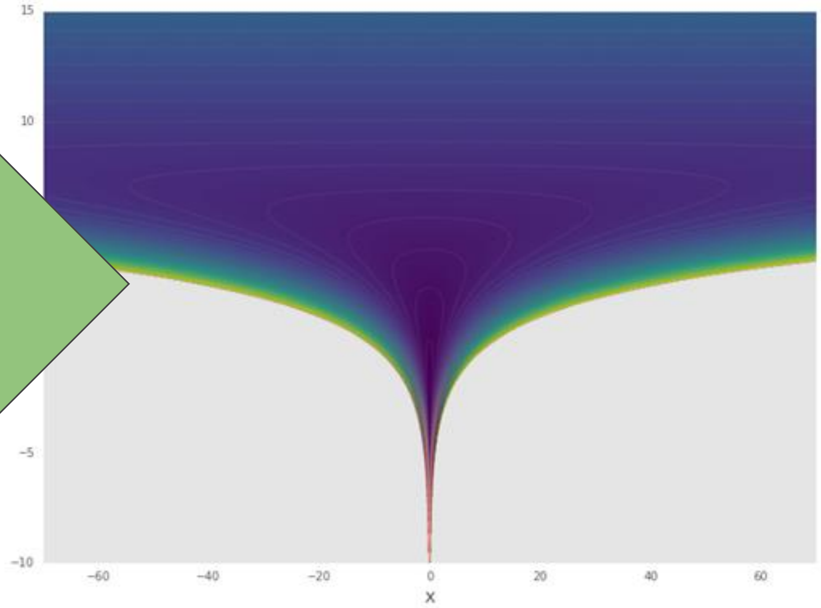
# What is Model Reparameterisation?

$$\tilde{y} \sim \mathcal{N}(0, 1)$$
$$\tilde{x} \sim \mathcal{N}(0, 1)$$




$$y = \tilde{y} \times 3$$
$$x = \tilde{x} \times e^{y/2}$$

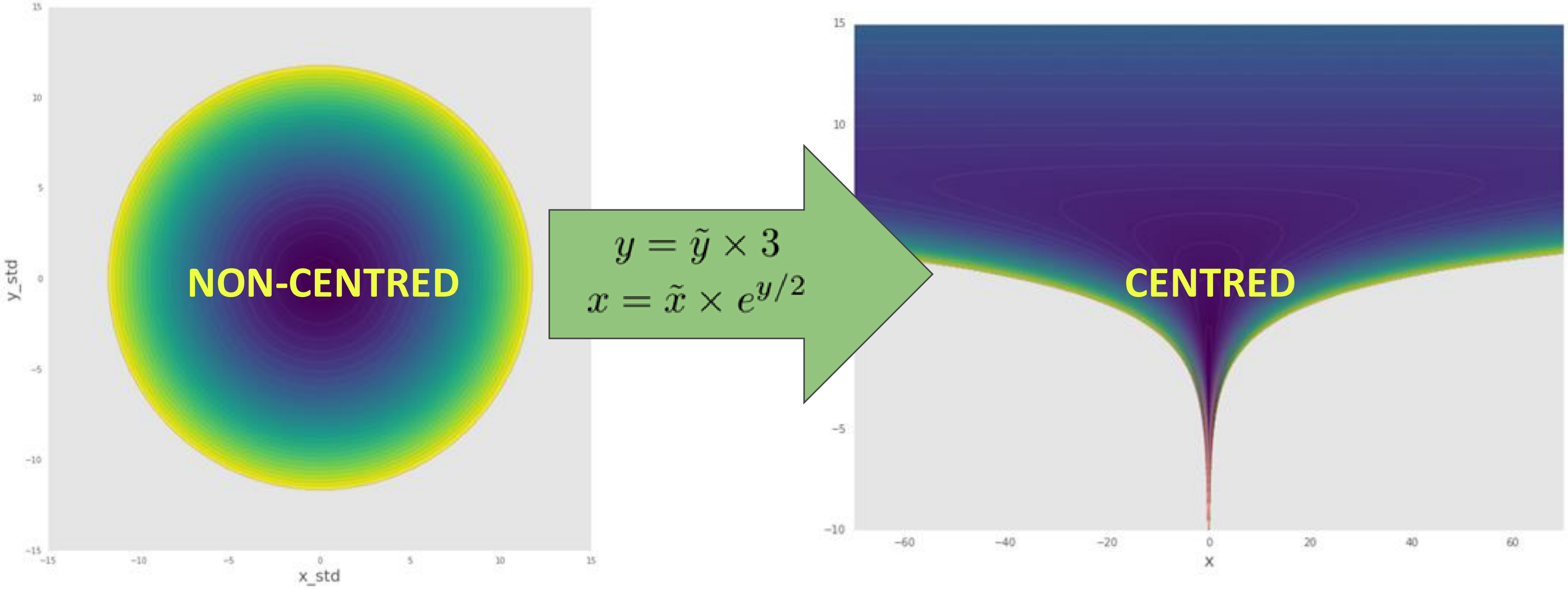
$$y \sim \mathcal{N}(0, 3)$$
$$x \sim \mathcal{N}(y, e^{y/2})$$



# What is Model Reparameterisation?

$$\tilde{y} \sim \mathcal{N}(0,1)$$
$$\tilde{x} \sim \mathcal{N}(0,1)$$

$$y \sim \mathcal{N}(0,3)$$
$$x \sim \mathcal{N}(y, e^{y/2})$$



# Understanding Reparameterisation Effects

Centred

$$\theta \sim \mathcal{N}(0,1) \qquad \mu \sim \mathcal{N}(\theta, \sigma_\mu)$$

$$y_n \sim \mathcal{N}(\mu, \sigma) \text{ for all } n \in 1 \ldots N$$

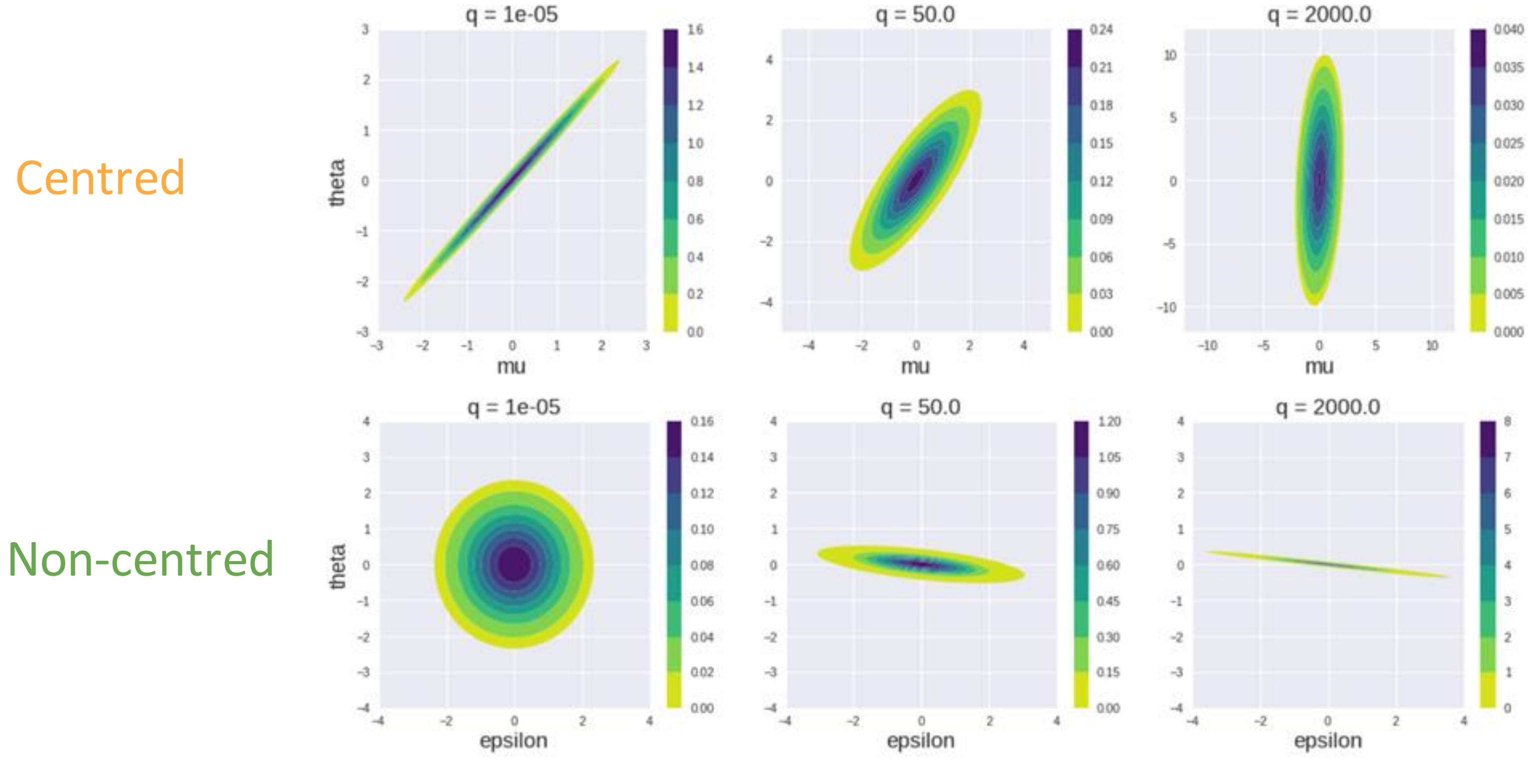
Non-centred

$$\theta \sim \mathcal{N}(0,1) \qquad \epsilon \sim \mathcal{N}(0,1) \qquad \mu = \theta + \sigma_\mu \epsilon$$

$$y_n \sim \mathcal{N}(\theta + \sigma_\mu \epsilon, \sigma) \text{ for all } n \in 1 \ldots N$$

# Understanding Reparameterisation Effects

$q = N/\sigma$



# Goal: Free modellers of the need to choose model parameterisation

# Reparameterisation in Edward2

## Centred

```python
def model(N, sigma, sigma_mu):

  theta = ed.Normal(0., 3.)


  mu = ed.Normal(theta, sigma_mu)


  y = ed.Normal(mu, sigma)
  return y
```

## Non-centred

```python
def model_ncp(N, sigma, sigma_mu):

  theta_std = ed.Normal(0., 1.)
  theta = 3. * theta_std

  mu_std = ed.Normal(0., 1.)
  mu = theta + mu_std * sigma_mu

  y = ed.Normal(mu, sigma)
  return y
```

# Reparameterisation in Edward2

*Interceptor*: A function that possibly changes how and if a RV is constructed.

```
def ncp(rv_constr, **kwargs):
  if is_location_scale(rv_constr):
   std = rv_constr(0., 1.)
   return kwargs["scale"] * std + kwargs["loc"]
```

```
with interception(ncp):
  theta = ed.Normal(0., 3.)
  mu = ed.Normal(theta, 1.)
```

=

```
theta_std = ed.Normal(0., 1.)
theta = 3. * theta_std
mu_std = ed.Normal(0., 1.)
mu = mu_std + theta
```

# Interleaved HMC (I-HMC)

*Input*:

- $p_{\text{CP}}(\boldsymbol{\theta})$: the CP target distribution
- $p_{\text{NCP}}(\boldsymbol{\psi})$: the NCP target distribution
- $f(\boldsymbol{\theta})$: an invertible function mapping CP variables to NCP variables

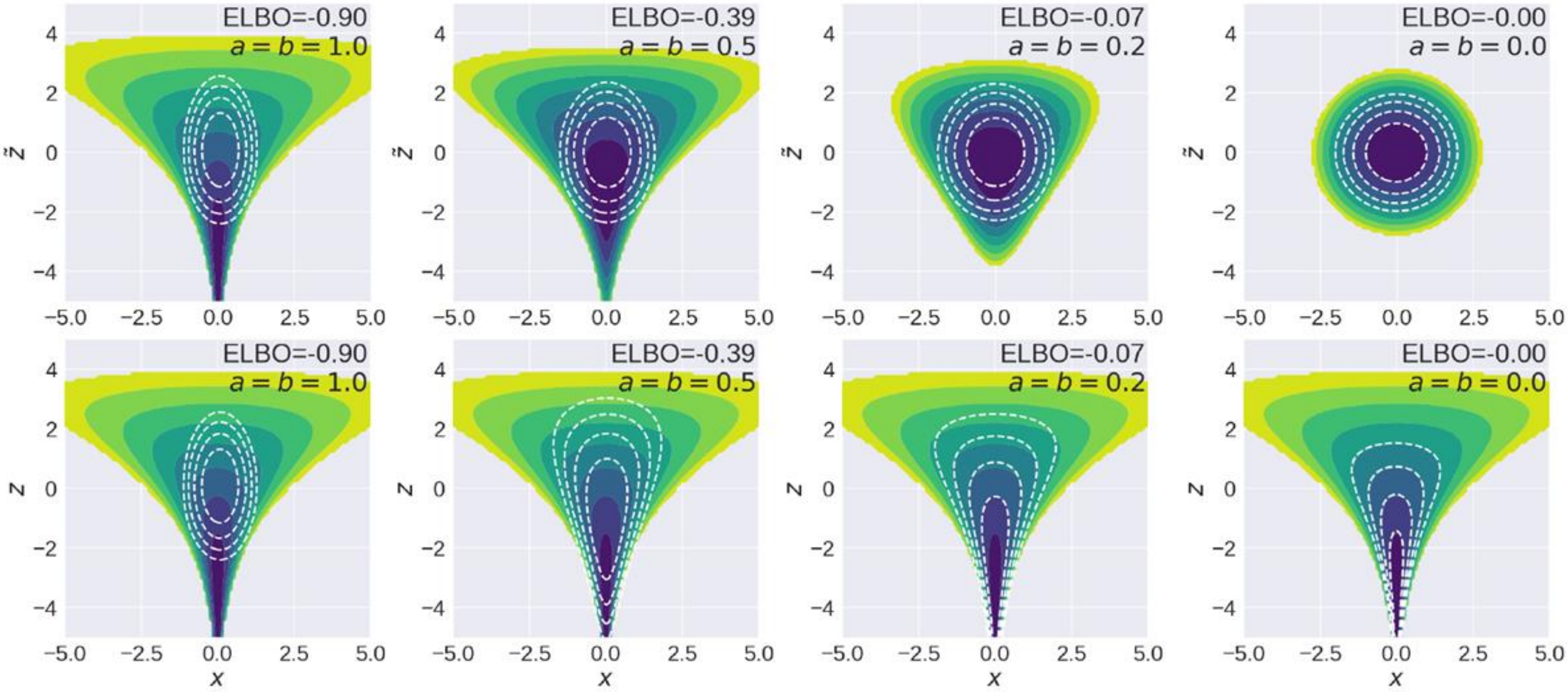
At the current sample $\boldsymbol{\theta}^{(n)}$:

1. Take one HMC step in *CP* space (using $p_{\text{CP}}(\boldsymbol{\theta}^{(n)})$) to obtain $\boldsymbol{\theta}^{(n+1/2)}$
2. Transform to NCP: $\boldsymbol{\psi}^{(n+1/2)} = f(\boldsymbol{\theta}^{(n+1/2)})$
3. Take on HMC step in *NCP* space (using $p_{\text{NCP}}(\boldsymbol{\psi}^{(n+1/2)})$) to obtain $\boldsymbol{\psi}^{(n+1)}$
4. Transform to CP: $\boldsymbol{\theta}^{(n+1)} = f^{-1}(\boldsymbol{\psi}^{(n+1)})$

# Variationally Inferred Parameterisation (VIP)

$$\tilde{z} \sim \mathcal{N}(a\mu, \sigma^b)$$

$$z = \mu + \sigma^{1-b}(\tilde{z} - a\mu)$$



ELBO=-0.90
$a = b = 1.0$
ELBO=-0.39
$a = b = 0.5$
ELBO=-0.07
$a = b = 0.2$
ELBO=-0.00
$a = b = 0.0$
$\tilde{z}$
$z$
$x$

# Preliminary Results

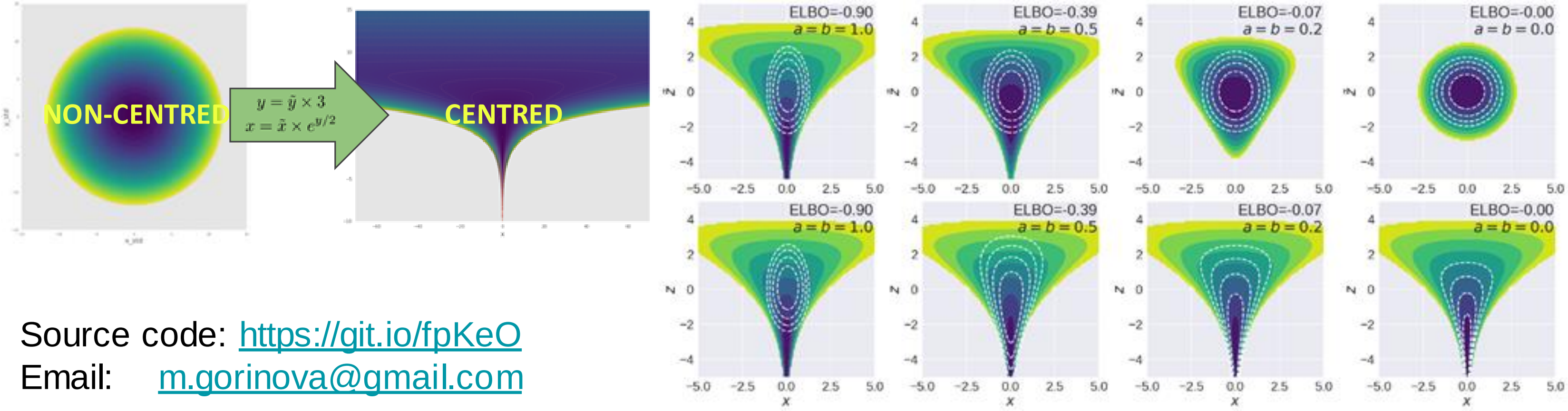
- Interleaved HMC is robust
- VIP-HMC finds a reasonable reparameterisation
- Mixed parameterisations might sometimes be superior to fully centred or fully non-centred

| | 8 Schools | Radon(MN) | Radon(NA) |
|---|---|---|---|
| HMC-CP | 92±4 | 798±276 | 2840±347 |
| HMC-NCP | 3475±849 | 340±35 | 187±36 |
| I-HMC | 3879±281 | 1495±129 | 2421±89 |
| HMC-VIP | 4986±660 | 1144±279 | 3273±145 |

Effective sample size to number of leapfrog steps

# Automatic Reparameterisation in Probabilistic Programming

Maria I. Gorinova, Dave Moore, Matthew D. Hoffman




Source code: https://git.io/fpKeO
Email: m.gorinova@gmail.com

# Thank you!