# DD2434 Advanced Machine Learning
# Assignment 1AD

Reuben Gezang

November 2025

## D-Level

### Theory 1.D.1

#### Question 1.1.1

We start by stating the definition of the Kullback-Leibler divergence:

$$KL(q(Z)\|p(Z|X)) = \mathbb{E}_{q(\boldsymbol{Z})}[\log(\frac{q(Z)}{p(Z|X)})] = \int q(Z)\log(\frac{q(Z)}{p(Z|X)})dZ \tag{1}$$

Now we separete the fraction inside the logarithm:

$$KL(q(Z)\|p(Z|X)) = \int q(Z)\log(\frac{q(Z)p(X)}{p(X,Z)})dZ = \int q(Z)(\log(q(Z)) - \log(p(Z|X))dZ \tag{2}$$

Note that $p(Z|X) = \frac{p(X,Z)}{p(X)}$ and using this we can rewrite the equation as:

$$KL(q(Z)\|p(Z|X)) = \int q(Z)\log(q(Z))dZ - \int q(Z)\log(p(X,Z))dZ + \log(p(X))\int q(Z)dZ \tag{3}$$

Since $q(Z)$ is a probability distribution, we know that $\int q(Z)dZ = 1$. Now we can solve for $\log(p(X))$:

$$\log(p(X)) = KL(q(Z)\|p(Z|X)) - \int q(Z)\log(q(Z))dZ + \int q(Z)\log(p(X,Z))dZ \tag{4}$$

Combining the logarithm terms, we get:

$$\log(p(X)) = KL(q(Z)\|p(Z|X)) + \mathbb{E}_{q(\boldsymbol{Z})}[\frac{\log(p(X,Z))}{q(Z)}] \tag{5}$$

Now we can identify the Evidence lower bound (ELBO)

$$\mathcal{L}(q) = \mathbb{E}_{q(\boldsymbol{Z})}[\frac{\log(p(X,Z))}{q(Z)}] \tag{6}$$

and with this we have shown that:

$$\log(p(X)) = \mathcal{L}(q) + KL(q(Z)\|p(Z|X)) \tag{7}$$

concluding the proof.

### Question 1.1.2

KOLLA ÖVER IGEN In this question we are to describe (in one sentence) how the choice of variational family $q(Z)$ affects

- (i) The tightness of the ELBO
- (ii) The accuracy of the posterior approximation

(i) A more expressive variational family can lead to a tighter ELBO as it can better approximate the true posterior, reducing the KL divergence term.

(ii) The choice of variational family directly impacts the accuracy of the posterior approximation, as a limited family may not capture the true posterior's complexity, leading to a less accurate approximation.

## Question 1.D.2

### 1.1.3

For a mean field assumption and joint distribution

$$q(Z_1, Z_2, Z_3) = q_1(Z_1)q_2(Z_2)q_3(Z_3), \quad p(X, Z)$$

Let $q_1^*(Z_1)$ be the $q_1$ that maximizes the ELBO. We want to show that $q_1^*$ satisfies

$$\log q_1^*(Z_1) = \mathbb{E}_{-Z_1}[\log p(X, Z)]$$

We can start by inspecting the ELBO:

$$\mathcal{L}(q) = \mathcal{L}(q) = \mathbb{E}_{q(\boldsymbol{Z})}[\log(\frac{p(X, Z)}{q(Z)})] = \mathbb{E}_q[\log p(X, Z)] - \mathbb{E}_q[\log q(Z)]$$

and using the mean field assumption we can rewrite this as:

$$\mathcal{L}(q) = \mathbb{E}_{q(Z)}[\log p(X, Z)] - \mathbb{E}_{q(Z)}[\log(q_1(Z_1)q_2(Z_2)q_3(Z_3))]$$

and by separating the logarithm we get, and taking expectations over the relevant distribution ($z_i$ is independent of $z_j$ for $i \neq j$):

$$\mathbb{E}_{q(Z)}[\log(q_1(Z_1)q_2(Z_2)q_3(Z_3))] = \mathbb{E}_{q_1(Z_1)}[\log(q_1(Z_1))] + \mathbb{E}_{q_2(Z_2)}[\log(q_2(Z_2))] + \mathbb{E}_{q_3(Z_3)}[\log(q_3(Z_3))]$$

Since we are maximizing w.r.t $q_1(Z_1)$ we can ignore the terms that do not depend on it. Thus we can rewrite the ELBO as:

$$\mathcal{L}(q) = \mathbb{E}_{q(Z)}[\log p(X, Z)] - \mathbb{E}_{q_1(Z_1)}[\log(q_1(Z_1))] + C$$

where $C$ is a constant w.r.t $q_1(Z_1)$ (and can thus be ignored). Now we can rewrite the expectation over $q(Z)$ as:

$$\mathbb{E}_{q(Z)}[\log p(X, Z)] = \mathbb{E}_{q_1(Z_1)}[\mathbb{E}_{q_2(Z_2)q_3(Z_3)}[\log p(X, Z)]]$$

meaning that we can rewrite the ELBO as:

$$\mathcal{L}(q) = \mathbb{E}_{q_1(Z_1)}[\mathbb{E}_{q_2(Z_2)q_3(Z_3)}[\log p(X, Z)]] - \mathbb{E}_{q_1(Z_1)}[\log(q_1(Z_1))] + C$$

Using the fact that $\int_{Z_1} q(Z_1)dZ_1 = 1$ we will now optimize the ELBO w.r.t $q_1(Z_1)$ and with a lagrange multiplier $\lambda$.

$$\frac{\partial}{\partial q_1(Z_1)}\left(\mathbb{E}_{q_1(Z_1)}[\mathbb{E}_{q_2(Z_2)q_3(Z_3)}[\log p(X,Z)]] - \mathbb{E}_{q_1(Z_1)}[\log(q_1(Z_1))] + \lambda(\int_{Z_1} q(Z_1)dZ_1 - 1)\right) = 0 \tag{8}$$

giving that

$$\mathbb{E}_{q_2(Z_2)q_3(Z_3)}[\log p(X,Z)] - \log(q_1(Z_1)) - 1 + \lambda = 0 \rightarrow \log(q_1^*(Z_1)) = \mathbb{E}_{q_2(Z_2)q_3(Z_3)}[\log p(X,Z)] + \lambda - 1 \tag{9}$$

where $\lambda - 1$ is a additive constant that can be ignored when normalizing $q_1^*(Z_1)$. Thus we have shown that:

$$\log q_1^*(Z_1) = \mathbb{E}_{-Z_1}[\log p(X,Z)] \tag{10}$$

as required.

## Practice/Implementation - D level (I have chosen 1.D.3)

### 1.2.4

The log likelihood of the data (D) is:

$$\log(P(D|\mu,\tau)) = \frac{N}{2}\log(\tau) - \frac{N}{2}\log(2\pi) - \frac{\tau}{2}\sum_{i=1}^{N}(x_i - \mu)^2 \tag{11}$$

The log prior for $\mu$ and $\tau$ is: (Note that $\mu|\tau \sim \mathcal{N}(\mu_0, (\lambda_0\tau)^{-1})$ and $\tau \sim \text{Gamma}(a_0, b_0)$)

$$\log(P(\mu,\tau)) = \frac{1}{2}\log(\lambda_0\tau) - \frac{1}{2}\log(2\pi) - \frac{\lambda_0\tau}{2}(\mu-\mu_0)^2 + a_0\log(b_0) - \log(\Gamma(a_0)) + (a_0 - 1)\log(\tau) - b_0\tau \tag{12}$$

The log-variational distribution is (Where $q(\mu) \sim \mathcal{N}(\mu_N, \lambda_N^{-1})$ and $q(\tau) \sim \text{Gamma}(a_N, b_N)$):

$$\log(q(\mu,\tau)) = \frac{1}{2}\log(\lambda_N) - \frac{1}{2}\log(2\pi) - \frac{\lambda_N}{2}(\mu-\mu_N)^2 + a_N\log(b_N) - \log(\Gamma(a_N)) + (a_N - 1)\log(\tau) - b_N\tau \tag{13}$$

Finally we state the score functions of the variational distributions. Let

$$\omega = (\mu_N, \lambda_N, a_N, b_N)$$

be the variational parameters.

$$\nabla_\omega \log(q(\mu,\tau)) = \begin{bmatrix} \lambda_N(\mu - \mu_N) \\ \frac{1}{2\lambda_N} - \frac{1}{2}(\mu - \mu_N)^2 \\ \psi(a_N) - \log(b_N) + \log(\tau) \\ \frac{a_N}{b_N} - \tau \end{bmatrix} \tag{14}$$

where $\psi$ is the digamma function.

### 1.2.5

In this exercise we implement Algorithm 1 of the BBVI paper [Ranganath et al., 2014]. We reuse the data sampling script from 1.E.3 and prior parameters, meaning that

- $\mu_0 = 1.0$
- $\lambda_0 = 0.1$

- $a_0 = 1.0$
- $b_0 = 2.0$

I have chosen to use the Robbins-monro sequence to be $\rho_t = \frac{1}{1000+t}$ and the following plots show the ELBO over iterations and the expected value of $\mu$ and $\tau$ over iterations for dataset 2 with 100 samples.

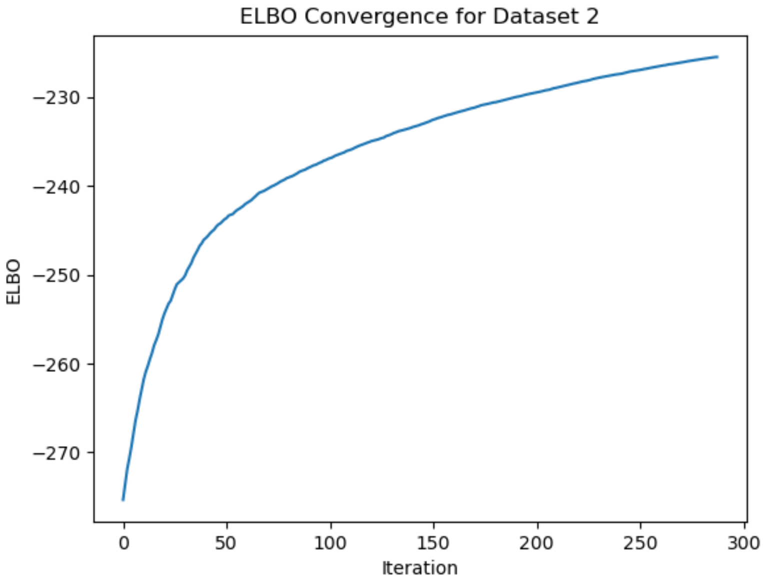

Figure 1: ELBO over iterations for dataset with 100 samples.

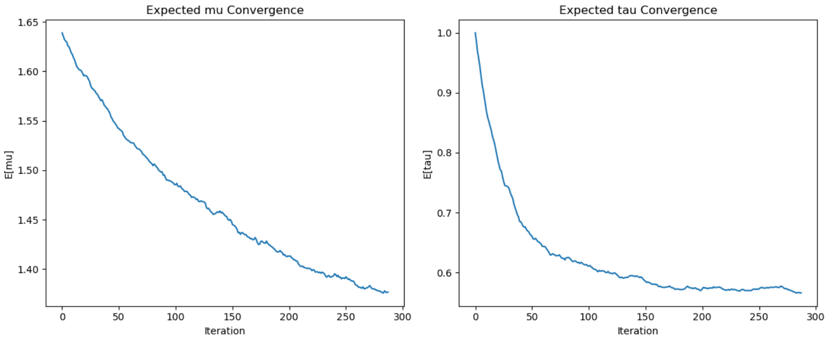

Figure 2: Expected value of $\mu$ and $\tau$ over iterations for dataset with 100 samples.

## C-level

### Theory 1.C.1

### Question 2.1.10

For this exercise we want to show that that the IWELBO (Importance-Weighted ELBO) is a valid lower bound on the log marginal likelihood $\log p(X)$. First of, we state the IWELBO:

$$\mathcal{L}_K(q) := \mathbb{E}_{Z_1,\ldots,Z_K}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}\frac{p(X,Z_k)}{q(Z_k|X)}\right)\right] \tag{15}$$

Now, note that we can rewrite the marginal likelihood as:

$$p(X) = \int p(X,Z)dZ = \int q(Z|X)\frac{p(X,Z)}{q(Z|X)}dZ = \mathbb{E}_{q(Z|X)}\left[\frac{p(X,Z)}{q(Z|X)}\right]$$

and we can extent this to $K$ samples:

$$p(X) = \mathbb{E}_{q(Z_1|X),\dots,q(Z_K|X)}\left[\frac{1}{K}\sum_{k=1}^{K}\frac{p(X,Z_k)}{q(Z_k|X)}\right]$$

Now, by applying Jensen's inequality that says that for a concave function $f$ and random variable $X$, $\mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$, (note that log is concave) we get:

$$\log p(X) = \log\left(\mathbb{E}_{q(Z_1|X),\dots,q(Z_K|X)}\left[\frac{1}{K}\sum_{k=1}^{K}\frac{p(X,Z_k)}{q(Z_k|X)}\right]\right) \geq \mathbb{E}_{q(Z_1|X),\dots,q(Z_K|X)}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}\frac{p(X,Z_k)}{q(Z_k|X)}\right)\right]$$

Thus, the IWELBO is a valid lower bound on the log marginal likelihood $\log p(X)$, as required.

### 2.1.11

Let $W_K = \frac{p(X,Z_K)}{q(Z_K|X)}$. The IWELBO can then be rewritten as:

$$\mathcal{L}_K(q) = \mathbb{E}_{Z_1,\dots,Z_K}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}W_k\right)\right]$$

We can also see that the standard ELBO can be rewritten as:

$$\mathcal{L}_1(q) = \mathbb{E}_{Z_1}\left[\log\left(W_1\right)\right] = \mathbb{E}_{Z_1,\dots,Z_K}\left[\frac{1}{K}\sum_{k=1}^{K}\log\left(W_k\right)\right]$$

This is because the $W_k$ are i.i.d. Now we can see that

$$\log\left(\frac{1}{K}\sum_{k=1}^{K}W_k\right) \geq \frac{1}{K}\sum_{k=1}^{K}\log\left(W_k\right)$$

This follows from the fact that the logarithm is strictly concave and according to Jensens inequality the logarithm of an average is greater than the average of the logaritms. Taking expectations of both sides, we get the final expression:

$$\mathcal{L}_K(q) = \mathbb{E}_{Z_1,\dots,Z_K}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}W_k\right)\right] \geq \mathbb{E}_{Z_1,\dots,Z_K}\left[\frac{1}{K}\sum_{k=1}^{K}\log\left(W_k\right)\right] = \mathcal{L}_1(q)$$

## Theory 1.C.2

### Question 2.1.12

In this exercise we derive the Rao-Blackwellized partial gradient of the ELBO w.r.t $\lambda_3$. We have a total of $n+4$ latent variables, $v, z, y_n, \omega_1, \omega_2$. The variational distribution is given by:

$$q(w_1, w_2, z, v, y) = q_{\lambda_1}(w_1)q_{\lambda_2}(w_2)q_{\lambda_3}(z)q_{\lambda_4}(v)\prod_n q_{\lambda_{5,n}}(y_n).$$

The formula for the gradient of the ELBO w.r.t $\lambda_3$ is:

$$\nabla_{\lambda_3}\mathcal{L}(\lambda) = \mathbb{E}_{q(Z|X)}[\nabla_{\lambda_3}\log q_{\lambda_3}(z)(\log p_3(x,z) - \log q(z|\lambda_3))]$$

In this formula, as defined in the paper by Ranganath we have that $p_3(x,z)$ is the terms in the joint that depend on those variables. Meaning that

$$p_3(x,z) = p(x|z,y_n)p(z|v)$$

This gives the final expression for the Rao-Blackwellized partial gradient of the ELBO w.r.t $\lambda_3$:

$$\nabla_{\lambda_3}\mathcal{L}(\lambda) = \mathbb{E}_{q(Z|X)}[\nabla_{\lambda_3}\log q_{\lambda_3}(z)(\log p(x|z,y_n) + \log p(z|v) - \log q_{\lambda_3}(z))]$$

## Practice/implementation - 1.C.3

### Question 2.2.13

To implement BBVI with Control variates, but without Rao-Blackwellization, we reuse the code from 1.D.5 and add control variates. We begin by calculating $f(z^{(s)}) = \log p(X, z^{(s)}) - \log q(z^{(s)}|\Theta)$ for each sample $s$. Next, we compute the score function for each sample $s$ and each variational parameter $\theta_j$. Then, we calculate the optimal baselines $(a^*)$ (control variates) for each parameter and set the gradient estimate using the baseline $(a^*)$. With these modifications, we run the BBVI algorithm and obtain the following plots for dataset 2 with 100 samples. The following results were obtained:

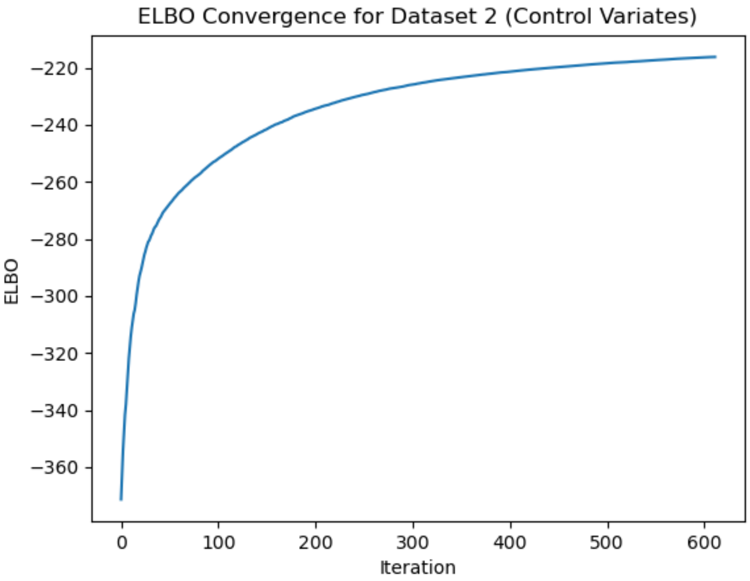

Figure 3: ELBO over iterations for dataset with 100 samples using BBVI with control variates.

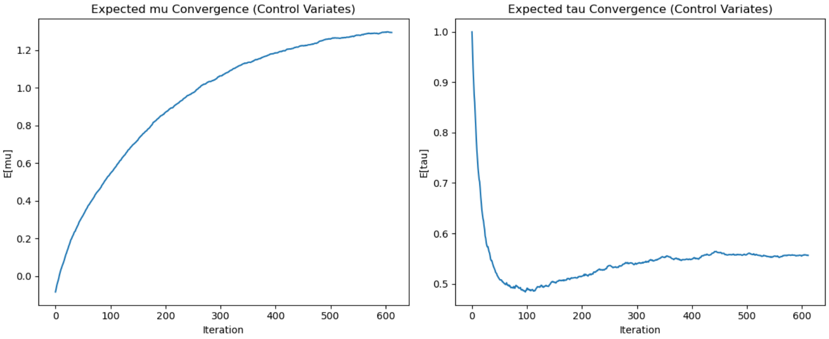

Figure 4: Expected value of $\mu$ and $\tau$ over iterations for dataset with 100 samples using BBVI with control variates.

## Practice/implementation - 1.C.4

### Question 2.2.14

The gamma distribution pdf is defined as

$$p(x|\alpha, \beta) = \frac{x^{\alpha-1}\beta^{\alpha}}{\Gamma(\alpha)}\exp(-\beta x) = \exp((\alpha - 1)log(x) - \beta x - log(\Gamma(\alpha)) + \alpha log(\beta))$$

In, canonical exponential form we can use the following identifications:

- $\boldsymbol{\eta}(\boldsymbol{\theta}) = [\alpha - 1, -\beta]$
- $h(x) = 1$
- $\boldsymbol{T}(x) = [\log x, x]$
- $A(\boldsymbol{\eta}) = \log \Gamma(\eta_1 + 1) - (\eta_1 + 1)\log(-\eta_2)$

With this we can identify

$$\nabla_{\boldsymbol{\eta}} A(\boldsymbol{\eta}) = \begin{bmatrix} \psi(\eta_1 + 1) - \log(-\eta_2) \\ -\frac{\eta_1+1}{\eta_2} \end{bmatrix}$$

**Question 2.2.15**

Let $J$ be the Jacobian matrix of $\eta$ w.r.t $\theta = (\alpha, \beta)$.

$$J = \begin{bmatrix} \frac{\partial \eta_1}{\partial \alpha} & \frac{\partial \eta_1}{\partial \beta} \\ \frac{\partial \eta_2}{\partial \alpha} & \frac{\partial \eta_2}{\partial \beta} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Furthermore, we have that $F(\eta)$ is the Fisher information matrix w.r.t the natural parameters $\eta$ and $F(\theta) = J^T F(\eta) J$. According to the Question description, we do not need to specify the explicit entries in $F$. Then we have that the natural gradient is given by:

$$\tilde{\nabla}_\theta \mathcal{L}(\theta) = F(\alpha, \beta)^{-1} \nabla_\theta \mathcal{L}(\theta) = J^T F(\eta)^{-1} J \nabla_\theta \mathcal{L}(\theta)$$

This follows from the fact that $J^{-1} = J$.

**Question 2.2.16**

Now we compute the fisher information matrix $F(\theta)$ explicitly. We have that $F(\eta) = \nabla_\eta \nabla_\eta^T A(\eta)$. Thus,

$$F(\eta) = \begin{bmatrix} \psi_1(\eta_1 + 1) & -\frac{1}{\eta_2} \\ -\frac{1}{\eta_2} & \frac{\eta_1+1}{\eta_2^2} \end{bmatrix}$$

where $\psi_1$ is the trigamma function. Now we can compute $F(\theta)$ as:

$$F(\theta) = J^T F(\eta) J = \begin{bmatrix} \psi_1(\alpha) & -\frac{1}{\beta} \\ -\frac{1}{\beta} & \frac{\alpha}{\beta^2} \end{bmatrix}$$

**Question 2.2.17**

The negative log likelihood is given by:

$$\mathcal{L}(\alpha, \beta) = -\log p(x_{1:N} | \alpha, \beta) = -\sum_{i=1}^{N} \log p(x_i | \alpha, \beta)$$

Since we have samples from a Gamma distribution we can rewrite this as

$$\mathcal{L}(\alpha, \beta) = -\frac{1}{N} \sum_{n=1}^{N} \left((\alpha - 1)\log x_n - \beta x_n - \log \Gamma(\alpha) + \alpha \log \beta\right)$$

The gradient of the negative log likelihood is then given by:

$$\nabla_{\alpha,\beta} \mathcal{L}(\alpha, \beta) = \begin{bmatrix} -\frac{1}{N} \sum_{n=1}^{N} (\log x_n - \psi(\alpha) + \log \beta) \\ -\frac{1}{N} \sum_{n=1}^{N} \left(-x_n + \frac{\alpha}{\beta}\right) \end{bmatrix}$$

### Question 2.2.18

This question is primarily solved in a python notebook. We sample 1000 data points from $Gamma(\alpha^* = 3.0, \beta^* = 2.0)$. Using this data, we implement both gradient descent and natural gradient descent to estimate $\alpha$ and $\beta$. We initilize both methods with a poor guess of $(\alpha, \beta) = (0.5, 8)$ and ensure that $\alpha, \beta$ stay positive during optimizaiton. The code can be found in ngd_vs_gd_1D_gamma_AD.ipynb notebook.

### Question 2.2.19

The results of running both gradient descent and natural gradient descent can be seen in the following plots.

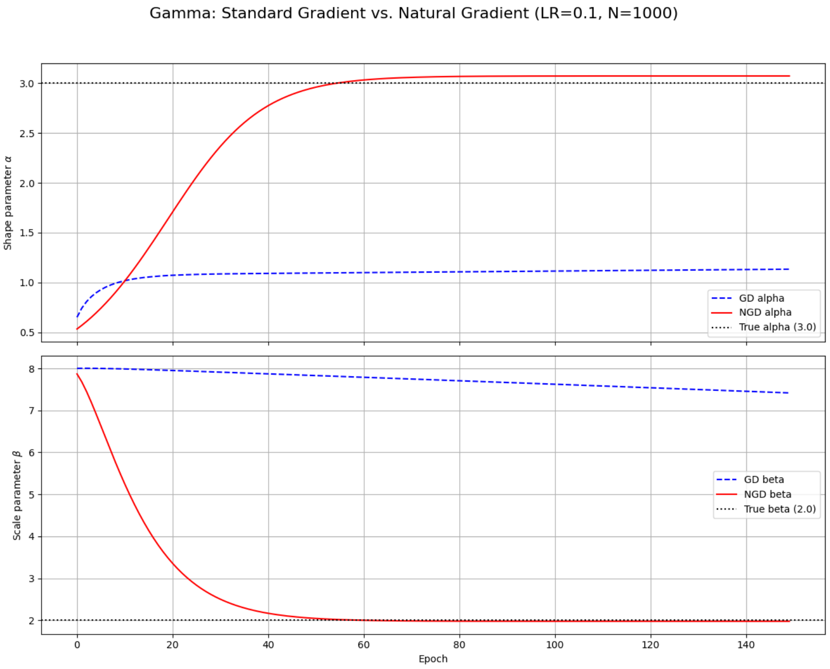

Figure 5: Gradient descent: Estimated $\alpha$ and $\beta$ over iterations.

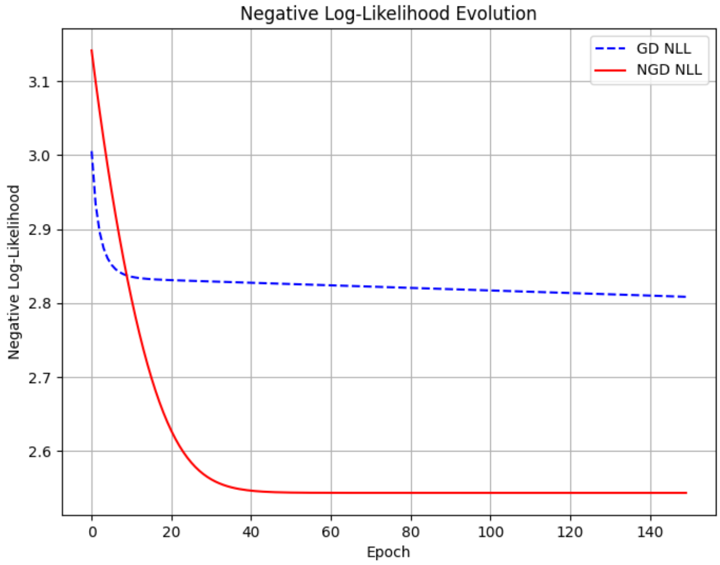

Figure 6: Natural gradient descent: Estimated $\alpha$ and $\beta$ over iterations.

Based on the plots, we can see that natural gradient descent converges faster to the true parameters $(\alpha^*, \beta^*) = (3.0, 2.0)$ compared to standard gradient descent. Furthermore, natural gradient descent does not reach a good approximation, even after 150 epochs. This is because natural gradient descent takes into account the geometry of the parameter space, leading to more efficient updates and avoids getting 'stuck'.

# B-level

## 1.B.1

### Question 3.1.20

To approximate and reparameterize the categorical distribution we can use the Gumbel-Softmax distribution as an approximation. Let $g_i \sim Gumbel(0, 1)$ for $i = 1, ..., K$, probabilities $\pi_1, ..., \pi_K$ and temperature $\tau$. A sample from the Gumbel-Softmax distribution is then given by:

$$y_i = \frac{\exp((\log(\pi_i) + g_i)/\tau)}{\sum_{j=1}^{K} \exp((\log(\pi_j) + g_j)/\tau)} \quad \text{for } i = 1, ..., K$$

This is the approximation of a one-hot encoded sample from a categorical distribution with class probabilities $\pi_1, ..., \pi_K$. As $\tau \to 0$, the Gumbel-Softmax distribution approaches the categorical distribution, and with $\tau \to \infty$, it approaches a uniform distribution. Note that the approximation is continuous and differentiable w.r.t the parameters $\pi_i$, allowing for gradient-based optimization.

For evaluation, we can use the argmax function to obtain a one-hot encoded sample from the Gumbel-Softmax distribution:

$$z = \text{one\_hot}(\text{argmax}_i(g_i + \log(\pi_i)))$$

The following code was implemented:

```python
# Hint: approximate the Categorical distribution with the Gumbel-Softmax distribution
def categorical_reparametrize(a, N, temp=0.1, eps=1e-20):
  # temp and eps are hyperparameters for Gumbel-Softmax

  dist = Gumbel(0,1)
  u = dist.sample((N, a.shape[0]))
  samples = F.softmax((torch.log(a + eps) + u) / temp, dim=1)

  return samples # make sure that your implementation allows the gradient to backpropagate

```

and the resulting output plot is shown below:

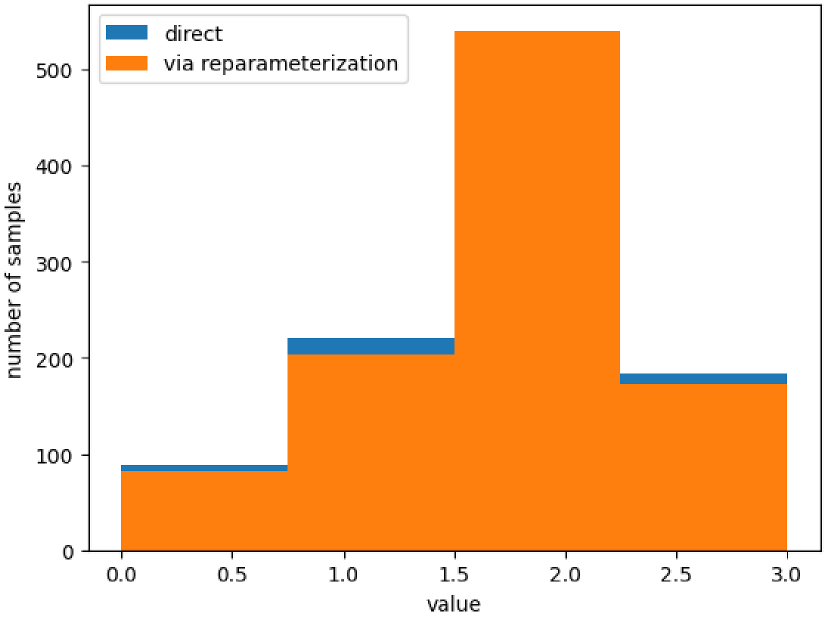

Figure 7: Output samples from the Gumbel-Softmax reparameterization.

## 1.B.2

### Question 3.1.21

Using the reparameterization trick we can express a sample $z$ from a parametric distribution $q(z)$ as a deterministic function of a random variable $\epsilon$, with some fixed sitribution and the parameters $\phi$ of $q_\phi(z)$, $(z = t(\epsilon, \phi))$. The paper gives the example of $q_\phi$ being a diagonal gaussian, and for $\epsilon \sim N(0, \mathbb{I})$, $z = \mu + \sigma\epsilon$ gives a sample from $q_\phi(z) = N(z|\mu, \sigma^2)$. Under such a parametrization of $z$ we can decompose the total derivative of the integrand of the estimator, w.r.t trainable parameters $\phi$ as:

$$\hat{\nabla}_{TD}(\epsilon, \phi) = \nabla_\phi \left[\log p(\boldsymbol{x}|\boldsymbol{z}) + \log p(\boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right] = \tag{16}$$

$$\nabla_{\boldsymbol{z}} \left[\log p(\boldsymbol{x}|\boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right] \nabla_\phi t(\epsilon, \phi) - \nabla_\phi \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \tag{17}$$

The reparameterized gradient estimator thus consists of two terms, the first is the path derivative and the second is the score function component.

### Question 3.1.22

$$\mathbb{E}_{q_\phi(z|x)}[\nabla_\phi \log q_\phi(z|x)] = \int q_\phi(z|x) \nabla_\phi \log q_\phi(z|x) dz = \int \nabla_\phi q_\phi(z|x) dz = \nabla_\phi \int q_\phi(z|x) dz = \nabla_\phi 1 = 0$$

The expectation of the score function is zero because the integral of the probability density function over its entire support is equal to 1, and the gradient of a constant (1) w.r.t any parameter is zero.

### Question 3.1.23

The authors propose that we can remove the score function term from the gradient estimate by setting $\phi'$ to the stop gradient.

### Question 3.1.24

The authors of the paper bring up the concept that if the score function is positively correlated with the remaining terms in the total derivative estimator, then the variance of the estimator can be reduced by subtracting the score function term. (The score function then acts as a control variate)

### Question 3.1.25

After extending the VAE implementation of 2E according to Algorithm 2, the following results were obtained for the ELBO over epochs on the MNIST dataset:

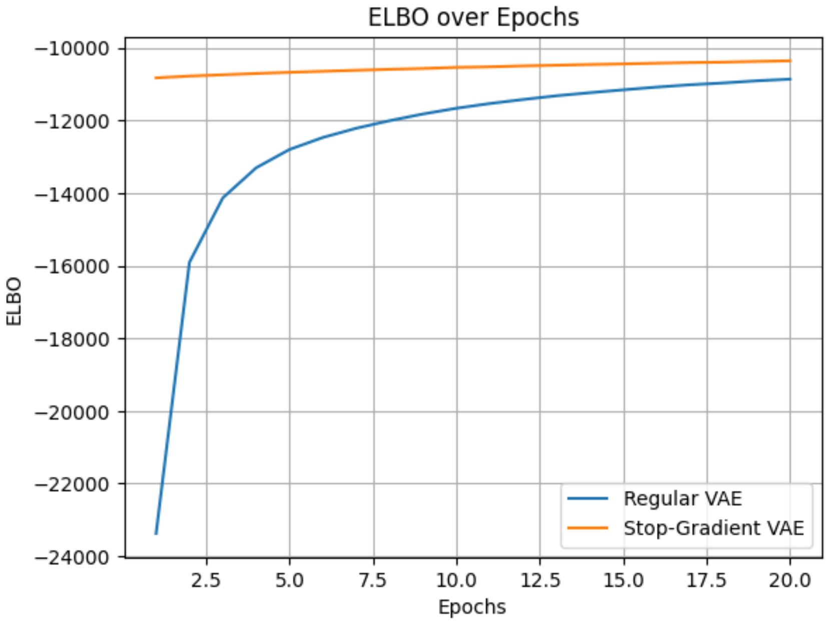

Figure 8: ELBO over epochs on MNIST using path derivative estimator.

We can clearly see that the Stop-Gradient VAE outperforms the standard VAE implementation from 2E. It is much more stable, converges faster and to a better ELBO value.

## A-level

### 1.A.1 - Theory

#### Question 4.1.26

### 1.A.2 - Practice/implementation

#### Question 4.2.27