title of the big homework

Big homework in big course

Saurav Shekhar, (16-947-921)
shekhars@student.ethz.ch

November 6, 2018

Abstract

All notes regarding my masters thesis project

1 External links

- Thesis pre proposal
- Tasks document
- Gitlab repository might be removed
- Overleaf project not regularly updated

2 Literature notes

Variational inference.: [Blei et al., 2016] **INPROGRESS**

$$\overbrace{\mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right)}^{\downarrow \text{ Objective } \geq 0} := \mathbb{E}_{q(\mathbf{z})}\left[\log q(\mathbf{z})\right] - \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{z}|\mathbf{x})\right] \tag{1}$$

$$\mathrm{ELBO}(q) := \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{z},\mathbf{x})\right] - \mathbb{E}_{q(\mathbf{z})}\left[\log q(\mathbf{z})\right] \tag{2}$$

$$\mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right) + \underbrace{\mathrm{ELBO}(q)}_{\uparrow \text{Optimize}} = \underbrace{\log p(\mathbf{x})}_{\text{constant w. } q} \tag{3}$$

$$\mathrm{ELBO}(q) = \mathbb{E}\left[\log p(\mathbf{x}|\mathbf{z})\right] - \mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z})\right) \tag{4}$$

$$\mathrm{ELBO}(q) = \mathcal{Q}(\theta, \theta_t) - \mathcal{H}(\mathbf{z}|\mathbf{x}) \text{ //Entropy} \tag{5}$$

$$\tag{6}$$

Algorithm 1: Coordinate Ascent for VI

Input: A model $p(\mathbf{x},\mathbf{z})$, a data set $\mathbf{x}$

Output: A variational density $q(\mathbf{z}) = \prod_{j=1}^{m} q_j(z_j)$

```
Initialize: Variational factors q_j(z_j)
while the ELBO has not converged do
    for j ∈ {1, ..., m} do
        Set q_j(z_j) ∝ exp{E_{-j}[log p(z_j | z_{-j}, x)]}
    end
    Compute ELBO(q) = E[log p(z, x)] − E[log q(z)]
end
return q(z)
```

Exponential families conditional conjugacy. [Bauer, 2018] **TODO**define conditional conjugacy properly

Gradient Optimization for ELBO. We will try to solve the optimization problem from Gradient ascent perspective. This will open up opportunity for stochastic optimization [Robbins and Monro, 1951] [Robbins and Monro, 1985].

Moving from Gradient Opt to Stochastic VI

Algorithm 2: VI with conjugate family assumption

Input: A model p, variational family $q_{\phi(z)}, q_\lambda(z)$

while *ELBO is not converged* **do**
  for *each data point i* **do**
    Update $\varphi_i \leftarrow \mathbb{E}_\lambda\left[\eta_l(\beta, x_i)\right]$
  end
  Update $\lambda \leftarrow \mathbb{E}_\varphi\left[\eta_g(x, z)\right]$
end

1. subsample a data point t from full data
2. use current global param λ to update local param φ_t
3. update λ

Gradient optimization step $\lambda_{t+1} = \lambda_t + \delta \nabla_\lambda f(\lambda_t)$. An equivalent formulation (for small $d\lambda$) is

$$\arg\max_{d\lambda} f(\lambda + d\lambda) \text{ st. } ||d\lambda||^2 \leq \epsilon \tag{7}$$

Here we have euclidean distance metric, which is not the best choice for probability distributions. For ex - $q_\lambda \sim \mathcal{N}(0, 1000)$ is much closer distribution to $q_{\lambda''} \sim \mathcal{N}(10, 10000)$ than $q_{\lambda'} \sim \mathcal{N}(0, 0.001)$ is to $q_{\lambda'''}\mathcal{N}(0.1, 0.001)$ even though $||\lambda - \lambda''|| \geq ||\lambda' - \lambda'''||$

Natural gradient of ELBO: *natural gradient* accounts for geometric structure of probability parameters (λ). They wrap the parameter space in a sensible way such that moving in same direction in different directions amounts to equal change in symmetrized KL divergence.

$$\begin{gathered}\arg\max_{d\lambda} f(\lambda + d\lambda) \text{ st.} \\ D_{KL}^{sym}(q_\lambda, q_{\lambda+d\lambda}) \leq \epsilon \text{ where} \\ D_{KL}^{sym}(q, p) = KL(q||p) + KL(p||q)\end{gathered} \tag{8}$$

We need to find Riemannian metric [1] $G(\lambda)$ which transforms euclidean distance to symmetrized KL divergence:

$$d\lambda^\intercal d\lambda = D_{KL}^{sym}(q_\lambda(\beta), q_{\lambda+d\lambda}(\beta)) \tag{9}$$

Using information geometry [2], we can also rescale the gradients in the right space:

$$\hat{\nabla_\lambda} ELBO = G^{-1}(\lambda) \nabla_\lambda ELBO \text{ where} \tag{10}$$

$$G(\lambda) = \mathbb{E}\left[\Big(\nabla_\lambda \log q_\lambda(\beta)\Big)\Big(\nabla_\lambda \log q_\lambda(\beta)\Big)^\intercal\right] \tag{11}$$

$G(\lambda)$ is the Fisher information matrix. For our model class (conjugate exponential...) We've

$$\nabla_\lambda \log q_\lambda(\beta) = t(\beta) - \mathbb{E}\left[t(\beta)\right] \tag{12}$$

Combining 12 and 11

$$G(\lambda) = \nabla_\lambda^2 a(\lambda) = a''(\lambda) \tag{13}$$

From [Hoffman et al., 2013], equation of Euclidean gradient

$$\nabla_\lambda ELBO = a''(\lambda)\Big(\mathbb{E}\left[\eta(\mathbf{x}, \mathbf{z})\right] - \lambda\Big) \tag{14}$$

TODOrefresh 14 with value in [Blei et al., 2016]

Combining 10, 14 and 13

$$\begin{aligned} g(\lambda) = \widehat{\nabla_\lambda} ELBO &= \mathbb{E}\left[\eta(\mathbf{x}, \mathbf{z})\right] - \lambda \text{ and} \\ \lambda_t &= \lambda_{t-1} + \delta_t g(\lambda_{t-1}) \\ \Rightarrow \lambda_t &= (1 - \delta_t)\lambda_{t-1} + \delta_t \mathbb{E}\left[\eta(\mathbf{x}, \mathbf{z})\right] \end{aligned} \tag{15}$$

Algorithm 3: VI with conjugate family assumption

Input: A model p, variational family $q_{\phi(z)}, q_\lambda(z)$
1 **while** *ELBO is not converged* **do**
2   **for** *each data point i* **do**
3   Update $\varphi_i \leftarrow \mathbb{E}_\lambda \left[\eta_l(\beta, x_i)\right]$
4   **end**
5   Update $\lambda \leftarrow (1 - \delta_t)\lambda + \delta_t \mathbb{E}_{q(\varphi)} \left[\eta_g(x, z)\right]$
6 **end**

Stochastic Variational inference. in Algorithm 3 line 2-4, we have to iterate over all data to compute the new set of local variables φ. This does not scale well to large datasets. [Hoffman et al., 2013] So we have to use stochastic gradients. Noisy gradients H of f will converge to a local optimum as long as

- $\mathbb{E}\left[H\right] = \nabla f$
- Step size δ_t st: $\sum_1^\infty \delta_t = \infty$ and $\sum_1^\infty \delta_t^2 < \infty$

Now,

$$\mathbb{E}\left[\eta(\mathbf{x}, \mathbf{z})\right] = \left(\alpha_1 + \sum_1^n \mathbb{E}_q\left[t(z_i, x_i)\right], n + \alpha_2\right)$$

Noisy gradient by sampling

1. Sample $t \sim Uniform(1, \ldots, n)$
2. Rescale

$$\begin{aligned} g(\lambda) &= \left(\alpha_1 + n\mathbb{E}_q\left[t(z_t, x_t)\right], n + \alpha_2\right) - \lambda \\ &=: \hat{\lambda} - \lambda \end{aligned}$$

Algorithm 4: Stochastic VI

Input: A model $p(\mathbf{x}, \mathbf{z})$, data $\mathbf{x}$
Initialize:variational family $q_{\phi(z)}, q_\lambda(z)$ with params λ_0
Result: Global variational densities $q_\lambda(\beta)$
while *Stopping criteria not met* **do**
  Sample $t \sim Uniform(1, \ldots, n)$
  Update $\phi_t \leftarrow \mathbb{E}_\lambda \left[\eta_l(\beta, x_t)\right]$
  Compute global param estimate $\hat{\lambda} = \mathbb{E}_\varphi \left[\eta_g(z_t, x_t)\right]$
  Update $\lambda \leftarrow (1 - \delta_t)\lambda + \delta_t \hat{\lambda}$
end
return λ

Research on optimizing difficult variational objectives with Monte Carlo (MC) estimates. Write gradient of ELBO as expectation, compute MC estimates, use stochastic optimization with MC estimates. New approaches avoid any model-specific derivations, and are called 'Black-box' inference techniques. As examples, see - [Kingma and Welling, 2013] [Rezende et al., 2014] [Ranganath et al., 2014] [Ranganath et al., 2016] [Titsias and Lázaro- [Kucukelbir et al., 2017]

$$\text{ELBO} = \mathbb{E}_{q_\nu}\left[\log p_\theta(z, x)\right] - \mathbb{E}_q\left[\log q_\nu z\right]$$

ν params of variational family, θ params of model. We need unbiased estimates of $\nabla_{\nu,\theta} ELBO$ to maximize ELBO.

Black Box variational inference. INPROGRESS
From [Ranganath et al., 2014]

[1]seems to be some kind of transformation
[2]Hope so

We will form the derivative of the objec- tive as an expectation with respect to the variational approximation and then sample from the variational ap- proximation to get noisy but unbiased gradients, which we use to update our parameters. For each sample, our noisy gradient requires evaluating the joint distribution of the observed and sampled variables, the variational distribution, and the gradient of the log of the varia- tional distribution. This is a black box method in that the gradient of the log of the variational distribution and sampling method can be derived once for each type of variational distribution and reused for many models and applications.

We will form the $\nabla ELBO$ as an $\mathbb{E}_{q_\lambda}$ [...] and then sample S samples from the q_λ to get noisy but unbiased gradients (w.r.t λ), which we use to update λ. For each sample, our noisy gradient requires evaluating the $p(\mathbf{x}, \mathbf{z}_S), q(\mathbf{z}_S)$, and $\nabla \log q(\mathbf{z}_S)$. This is a black box method in that the $\nabla \log q(\mathbf{z}_S)$ and sampling method can be derived once for each type of variational distribution and reused for many models and applications.

Equation (2) of [Ranganath et al., 2014]

$$\nabla_\lambda \mathcal{L} = \mathbb{E}_q \Big[\nabla_\lambda \log q(z|\lambda) \Big(\log p(x, z) - \log q(z|\lambda) \Big) \Big] \text{ where} \tag{16}$$

$$\mathcal{L}(\lambda) \triangleq \mathbb{E}_{q_{\lambda_z}} [\log p(x, z) - \log q(z)] \text{ (ELBO)}$$

here it says that Equation 2/3 can be derived simply using the log trick but the authors use a complicated method in paper. Also derived in [Jalil Taghia and Schn, 2018] and [Bauer, 2018]
the gradient $\nabla_\lambda \log q(z|\lambda)$ of the log of a probability distribution is called the score function or REINFORCE
Basic algorithm

$$z_s \sim q(z|\lambda) \text{ for } s \in 1..S$$

$$\nabla_\lambda \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_\lambda \log q(z_s|\lambda) \Big(\log p(x, z_s) - \log q(z_s|\lambda) \Big) \tag{17}$$

Rao-Blackwellization and smart Control Variates to control variance
Variance still very high. Reparameterization and amortization come to rescue (See this tutorial from David Blei)

Good notes on Stochastic VI and Black Box VI from [Jalil Taghia and Schn, 2018]

Reparameterization trick.
TODO

Boosting Variational inference. **TODO**

Frank-Wolfe. **INPROGRESS**[Jaggi, 2013] [Pedregosa, 2018] [Pedregosa et al., 2018] [**?**] Here $x, \mathcal{D} \equiv$

Algorithm 5: Frank-Wolfe

Constrained Optimization: $\min_{x \in \mathcal{D}} f(\mathbf{x})$
for $t \in \{0, \ldots, T\}$ **do**
  $s^t \leftarrow \arg\min_{s \in \mathcal{D}} \langle \mathbf{s}, \nabla f(\mathbf{x}^t) \rangle$
  $\mathbf{x}^{t+1} \leftarrow$ **UpdateRule**$(\mathbf{x}^t, s^t, t, f)$
end

$x, \mathcal{D} \equiv q, \mathcal{A}$

UpdateRule can be

$$
\begin{aligned}
q^{t+1} &\leftarrow (1-\gamma)q^t + \gamma s^t = q^t + \gamma \overbrace{(s^t - q^t)}^{d_t} \text{ where} \\
\textbf{Variant0}: \gamma &\leftarrow \frac{2}{t+2} && (18) \\
\textbf{Variant1}: \gamma &\leftarrow \arg\min_{\gamma\in[0,1]} f((1-\gamma)q^t + \gamma s^t) && (19) \\
g_t &\leftarrow -\langle \nabla f(\mathbf{x}_t), d_t \rangle \\
\textbf{Exitcondition}: & g_t < \delta \\
\textbf{Variant2}: \gamma &\leftarrow \min\left(\frac{g_t}{L||d_t||^2}, 1\right) && (20) \\
\textbf{Variant3}: & \\
q^{t+1} &\leftarrow \arg\min_{q\in\bigcup_{i=1}^{t} s^t} f(q) \\
& && (21)
\end{aligned}
$$

20 has variants [Pedregosa, 2018] [Demyanov and Rubinov, 1970] **TODO**add more

$$\gamma \leftarrow \min\left\{\frac{g_t}{L\ \mathrm{diam}(\mathcal{D})^2}, 1\right\}$$

Boosting Black Box Variational inference. INPROGRESS
Boosting introduced in [Guo et al., 2016], connection with FW in [Locatello et al., 2017]. define a Linear Minimization Problem (LMO) as $\mathbf{LMO}_{\mathcal{A}}(y) := \arg\min_{s\in\mathcal{A}} \langle y, s\rangle$ In line 3 of 5, rewrite it as

$$s^t \leftarrow (\delta - \text{Approx-})\mathbf{LMO}_{\mathcal{A}}(\nabla f(q^t))$$

Algorithm for LMO in section 4 of [Locatello et al., 2018]. In Theorem 2, Curvature $\mathcal{C}_{f,\mathcal{A}}$ is bounded for D^{KL} if param. space of densities in $\mathcal{A}$ is bounded. In section 3, a bounded curvature for D^{KL} is obtained.

Black box LMO:
In this case $f(q^t) = \mathrm{KL}\left(q^t(\mathbf{z})||p(\mathbf{x},\mathbf{z})\right)$. Assuming θ are the parameters defining variational family $\mathcal{Q} \equiv \mathcal{A}$ We've to find $\nabla_\theta f(q^t)$, more specifically, we've to find

$$s^t \leftarrow (\delta - \text{ Approx.}) \arg\min_{s\in\mathcal{A}} \langle \nabla \mathrm{KL}\left(q^t(\mathbf{z})||p(\mathbf{x},\mathbf{z})\right), s\rangle$$

TODOadd how [Guo et al., 2016] [Locatello et al., 2017] deal with optimization of LMO. Also add the part about $\mathrm{conv}(\mathcal{A})$ being sufficient instead of $\mathcal{A}$.
Convergence of SGD not fully understood. To guarantee convergence of FW, solution of LMO should not be degenerate. This translates to a constraint on $||s||_\infty$ which is not practical. Every pdf with bounded $||\cdot||_\infty$ has bounded entropy and the converse holds true in most cases of interest. (Gaussian, Laplacian, ...). Assume $\mathcal{A}$ is such a family and $\bar{\mathcal{A}}$ is $\mathcal{A}$ w/o l_∞ norm constraint. **TODO**ask

$$\arg\min_{s\in\bar{\mathcal{A}}, \mathcal{H}(s)\geq -M} \langle \nabla \mathrm{KL}\left(q^t(\mathbf{z})||p(\mathbf{x},\mathbf{z})\right), s\rangle \overset{?}{\equiv} \arg\min_{s\in\bar{\mathcal{A}}, \mathcal{H}(s)\geq -M} \left\langle s, \log\frac{q^t}{p}\right\rangle$$

Using Lagrange multiplier λ

$$
\begin{aligned}
&\left\langle s, \log\left(\frac{s}{\sqrt[\lambda]{\frac{p}{q^t}}}\right)\right\rangle \\
&\equiv \arg\min_{s\in\bar{\mathcal{A}}} \mathrm{KL}\left(s || \sqrt[\lambda]{\frac{p}{q^t}} Z\right) \\
\mathrm{RELBO}(s,\lambda) &:= \mathbb{E}_s[\log p] - \mathbb{E}_s\left[\log q^t\right] - \lambda \mathbb{E}_s[\log s] && (22)
\end{aligned}
$$

For true LMO solution, will need to maximize for λ. Might end in saddle, fix or slowly decrease with time $\frac{1}{\sqrt{t+1}}$

3 Ideas

3.1 Line search

Line search in 19 is not working very well.

line search

$$\gamma' = \underset{\gamma \in [0,1]}{\mathrm{argmin}} \; KL\left(\underbrace{q^t + \gamma (s - q^t)} \,\middle\|\, p \right)$$

$S^* = \mathrm{argmin}\; KL\left(s \,\middle\|\, \sqrt{\frac{p}{q_t}}\, z \right)$

↓

RELBO

ELR

grad. $\quad q^t_\gamma$

$$\nabla_\gamma - \int_z q^t_\gamma \log\left(\frac{q^t_\gamma}{p}\right) dz$$

$$= - \int \underbrace{\nabla_\gamma q^t_\gamma}_{(s - q^t)} \log\left(\frac{q^t_\gamma}{p}\right) + \cancel{q^t_\gamma} \cdot \underbrace{\nabla_\gamma \log\left(\frac{q^t_\gamma}{p}\right)}_{\frac{1}{\cancel{q^t_\gamma}} \cdot \nabla_\gamma q^t_\gamma}$$

$$= - \int \nabla_\gamma q^t_\gamma \left(1 + \log q^t_\gamma - \log p \right)$$

$$= - \int (s - q^t) \left(1 + \underbrace{\log q^t_\gamma - \log p} \right)$$

code

$$= - \mathbb{E}_s \left[1 + \log q^t_\gamma - \log p \right] \leftarrow \text{sample-s}$$

res-s $\quad q_t$

$$+ \mathbb{E}_{q^t} \left[1 + \log q^t_\gamma - \log p \right] \leftarrow \text{sample-q}$$

res-q

$$= \begin{cases} \mu^{t+1} \leftarrow [\mu^t, \mu_s] \\ \sigma^{t+1} \leftarrow [\sigma^t, \sigma_s] \end{cases}$$

new mix $= (1-\gamma)\, q_t + \gamma s$

pbar

$$\gamma \leftarrow \frac{2}{k+2}$$

$$w = ((1-\gamma) w, \gamma]$$

q_{next}

$$\hat{\nabla}_\gamma \approx \frac{1}{N} \left[\text{res-q} - \text{res-s} \right]$$

$$\gamma_{new} = \gamma_{old} + 0.1 \cdot \frac{(\hat{\nabla}_\gamma)}{t+1}$$

once 0, will always report to 0 ??

project onto [0, 1]

```python
def line_search_dkl(weights, locs, diags, mu_s, cov_s, x, k):
  """Perform line search for the best step size gamma.

  Uses gradient ascent to find gamma that minimizes
  KL(q_t + gamma (s - q_t) || p)

  Args:
      weights: weights of mixture components of q_t
      locs: means of mixture components of q_t
      diags: deviations of mixture components of q_t
      mu_s: mean for LMO Solution s
      cov_s: cov matrix for LMO solution s
      x: target distribution p
      k: iteration number of Frank-Wolfe
  Returns:
      Computed gamma
  """
  def softmax(v):
      return np.log(1 + np.exp(v))
  # no. of samples to approximate $\nabla_\gamma$
  N_samples = 10
  # Create current iter $q_t$
  weights = [weights]
  qt_comps = [
      Normal(
          loc=tf.convert_to_tensor(locs[i]),
          scale=tf.convert_to_tensor(diags[i])) for i in range(len(locs))
  ]
  qt = Mixture(
      cat=Categorical(probs=tf.convert_to_tensor(weights)),
      components=qt_comps,
      sample_shape=N)
  qt = InfiniteMixtureScipy(stats.multivariate_normal)
  qt.weights = weights[0]
  qt.params = list(
      zip([[l] for l in locs], [[softmax(np.dot(d, d))] for d in diags]))
  # samples from $q_t$
  sample_q = qt.sample_n(N_samples)
  # create and sample from s
  s = stats.multivariate_normal([mu_s],
                                np.dot(np.array([cov_s]), np.array([cov_s])))
  sample_s = s.rvs(N_samples)
  # $q_{t+1}$ is mixture of $q_t$ and s with weights $(1-\gamma)$ and $\gamma$
  # Set its corresponding parameters and weights
  new_locs = copy.copy(locs)
  new_diags = copy.copy(diags)
  new_locs.append([mu_s])
  new_diags.append([cov_s])
  # initialize $\gamma$
  gamma = 2. / (k + 2.)
  # no. steps of gradient ascent
  n_steps = 10
  prog_bar = ed.util.Progbar(n_steps)
  for it in range(n_steps):
      print("line_search iter %d, %.5f" % (it, gamma))
      new_weights = copy.copy(weights)
      new_weights[0] = [(1. - gamma) * w for w in new_weights[0]]
      new_weights[0].append(gamma)
      # create $q^\gamma_{t+1}$
      q_next = InfiniteMixtureScipy(stats.multivariate_normal)
      q_next.weights = new_weights[0]
      q_next.params = list(
          zip([[l] for l in new_locs], [[np.dot(d, d)] for d in new_diags]))
      # Computes $\mathbb{E}[...] \propto \sum_v \log p - \log q^\gamma_{t+1}$
      def px_qx_ratio_log_prob(v):
          Lambda = 1.
```

```
            ret = x.log_prob([v]).eval()[0] - q_next.log_prob(v)
            ret /= Lambda
            return ret
        # Samples w.r.t s
        rez_s = [
            px_qx_ratio_log_prob(sample_s[ss]) for ss in range(len(sample_s))
        ]
        # Samples w.r.t q_{t+1}
        rez_q = [
            px_qx_ratio_log_prob(sample_q[ss]) for ss in range(len(sample_q))
        ]
        # TODO(sauravshekhar) measure how noisy gradients are
        # Gradient ascent step, step size decreasing as 1/(it+1)
        gamma = gamma + 0.1 * (sum(rez_s) - sum(rez_q)) / (N_samples *
                                                        (it + 1.))
        # Projecting it back to [0, 1], too small range?
        # FIXME(sauravshekhar) if projected to 0, all iterations will be same?
        if gamma >= 1 or gamma <= 0:
            gamma = max(min(gamma, 1.), 0.)
            break
    return gamma
```

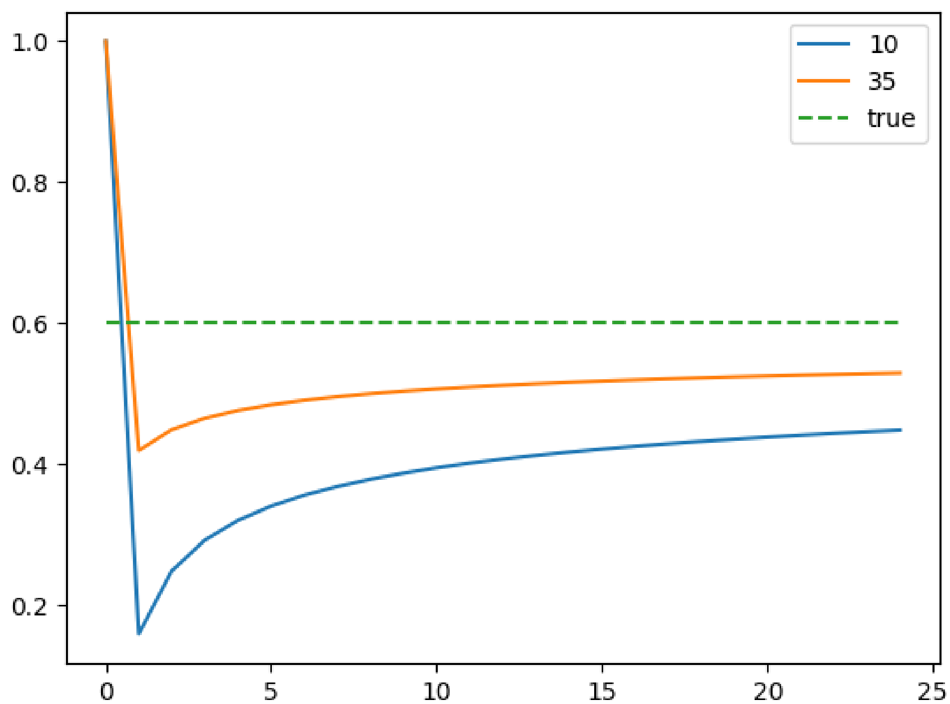

Figure 1: gamma with iterations for different n_samples

TODO make E_q plot with iterations starting from 1

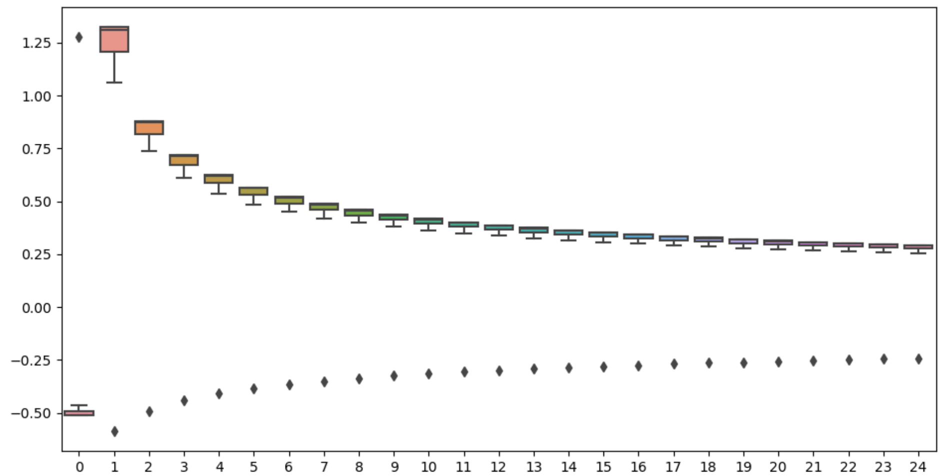

Figure 2: Boxplot for expectation w.r.t s for n_samples = 10

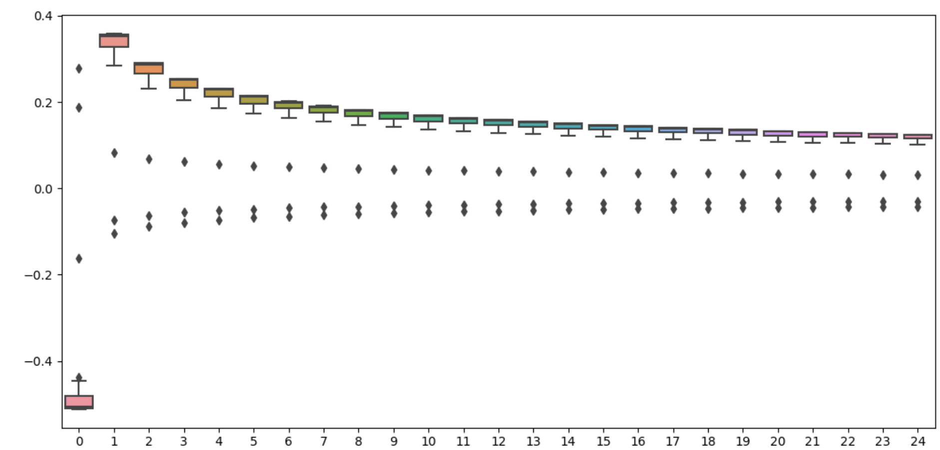

Figure 3: Boxplot for expectation w.r.t s for n_samples = 36

4 Math notes

Shannon Entropy.
Self information of event x = x is defined as $I(x) := -\log P(x)$

$$H(x) = \mathbb{E}_{x\sim P}\left[I(x)\right] = -\mathbb{E}_{x\sim P}\left[\log P(x)\right]$$

Cramer-Rao lower bound. [Balabdaoui and van de Geer, 2016] Suppose θ is an unknown deterministic parameter which is to be estimated from measurements x, distributed according to some pdf $f(x;\theta)$. The variance of any *unbiased estimator* $\hat{\theta}$ of θ is then bounded by reciprocal of Fischer Information $I(\theta)$:

$$\begin{aligned}
\mathrm{var}(\hat{\theta}) \geq \frac{1}{I(\theta)} &\text{ where} \\
I(\theta) &= \mathbb{E}\left[\left(\frac{\partial l(x;\theta)}{\partial\theta}\right)^2\right] \\
&= -\mathbb{E}\left[\frac{\partial^2 l(x;\theta)}{\partial\theta^2}\right]
\end{aligned}$$

Note: See Wikipedia for other more general versions

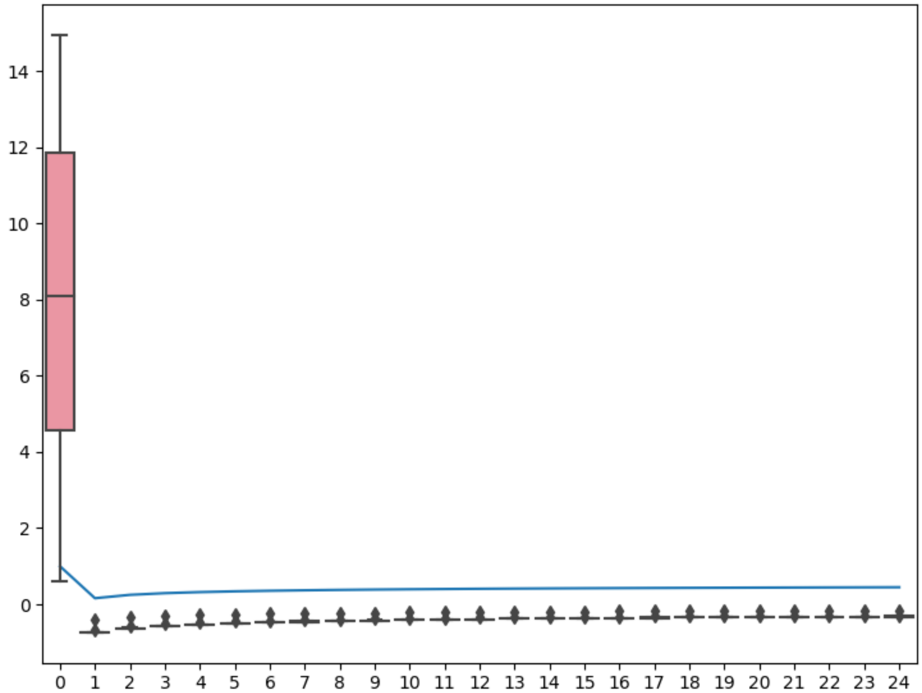

Figure 4: Boxplot for expectation w.r.t q_tĝamma for n_samples = 10

5 Code Notes

Normal distribution edward

```
from edward.models import Normal
from keras.layers import Dense

hidden = Dense(256, activation='relu')(x_ph)
qz = Normal(loc=Dense(10)(hidden),
scale=Dense(10, activation='softplus')(hidden))
```

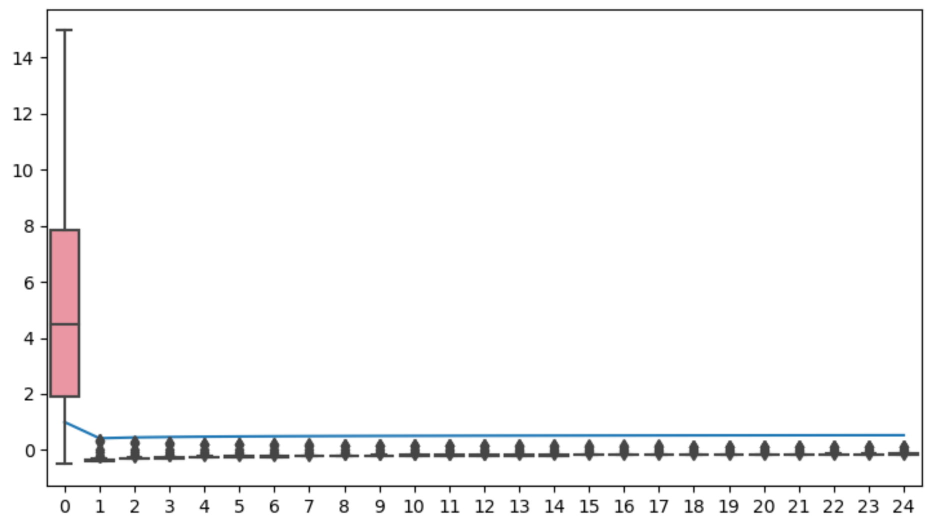

Figure 5: Boxplot for expectation w.r.t q_tĝamma for n_samples = 35

References

[Balabdaoui and van de Geer, 2016] Balabdaoui, F. and van de Geer, S. (2016). Fundamentals of mathematical statistics.

[Bauer, 2018] Bauer, S. (2018). Probabilistic graphical models for image analysis.

[Blei et al., 2016] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2016). Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*.

[Demyanov and Rubinov, 1970] Demyanov, V. F. and Rubinov, A. M. (1970). Approximate methods in optimization problems. *(Modern Analytic and Computational Methods in Science and Mathematics. IX.*

[Guo et al., 2016] Guo, F., Wang, X., Fan, K., Broderick, T., and Dunson, D. B. (2016). Boosting variational inference. *arXiv preprint arXiv:1611.05559*.

[Hoffman et al., 2013] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

[Jaggi, 2013] Jaggi, M. (2013). Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pages 427–435.

[Jalil Taghia and Schn, 2018] Jalil Taghia, L. M. and Schn, T. (2018). Probabilistic machine learning.

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[Kucukelbir et al., 2017] Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., and Blei, D. M. (2017). Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474.

[Locatello et al., 2018] Locatello, F., Dresdner, G., Khanna, R., Valera, I., and Rätsch, G. (2018). Boosting black box variational inference. *arXiv preprint arXiv:1806.02185*.

[Locatello et al., 2017] Locatello, F., Khanna, R., Ghosh, J., and Rätsch, G. (2017). Boosting variational inference: an optimization perspective. *arXiv preprint arXiv:1708.01733*.

[Pedregosa, 2018] Pedregosa, F. (2018). Notes on the frank-wolfe algorithm, part i. [Online; posted 21-March-2018].

[Pedregosa et al., 2018] Pedregosa, F., Askari, A., Negiar, G., and Jaggi, M. (2018). Step-size adaptivity in projection-free optimization. *arXiv preprint arXiv:1806.05123*.

[Ranganath et al., 2014] Ranganath, R., Gerrish, S., and Blei, D. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822.

[Ranganath et al., 2016] Ranganath, R., Tran, D., and Blei, D. (2016). Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333.

[Rezende et al., 2014] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286.

[Robbins and Monro, 1951] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

[Robbins and Monro, 1985] Robbins, H. and Monro, S. (1985). A stochastic approximation method. In *Herbert Robbins Selected Papers*, pages 102–109. Springer.

[Titsias and Lázaro-Gredilla, 2014] Titsias, M. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational bayes for non-conjugate inference. In *International Conference on Machine Learning*, pages 1971–1979.