CS492: Probabilistic Programming

Amortised Inference

Hongseok Yang
KAIST

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

```
(defquery biased-coin []
  (let [r (sample
             (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

```
(defquery biased-coin []
  (let [r (sample
             (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

Importance sampling with proposal q(r).

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

Importance sampling with proposal q(r).

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

$w_1 = 1$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1, r_1), \ldots, (w_n, r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)^*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = 1$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
             (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = 1 * p(.4)/q(.4)$

$r_1 = .4$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = .096 * p(.4)/q(.4)$
$r_1 = .4$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
             (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
    r))
```

$w_1 = .096 * p(.4)/q(.4)$
$r_1 = .4$

Importance sampling with proposal q(r).

1. Generate $(w_1, r_1), \ldots, (w_n, r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
              (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = .096 * p(.4)/q(.4)$
$r_1 = .4$

$w_2 = .144 * p(.6)/q(.6)$
$r_2 = .6$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = .096 * p(.4)/q(.4)$

$r_1 = .4$

$w_2 = .144 * p(.6)/q(.6)$

$r_2 = .6$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1, r_1), \ldots, (w_n, r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = .096 * p(.4)/q(.4)$
$r_1 = .4$

$w_2 = .144 * p(.6)/q(.6)$
$r_2 = .6$

Importance sampling with proposal $q(r)$.

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

How to find good q?

```
(defquery biased-coin []
  (let [r (sample
             (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

$w_1 = .096 * p(.4)/q(.4)$
$r_1 = .4$

$w_2 = .144 * p(.6)/q(.6)$
$r_2 = .6$

Importance sampling with proposal q(r).

1. Generate $(w_1,r_1), \ldots, (w_n,r_n)$ by running prog.

2. Estimate $\mathbb{E}_{p(r|a,b,c)}[f(r)] \approx \sum_i f(r_i)*(w_i/\sum_j w_j)$.

How to find good q? Use amortised inference!

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) false)]
   r))
```

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) false)]
    r))
```

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
          a (observe (flip r) true)
          b (observe (flip r) true)
          c (observe (flip r) true)]
    r))
```

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
    r))
```

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
   r))
```

Other examples: Finance model, captcha, brain, etc.

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
    r))
```

Other examples: Finance model, captcha, brain, etc.

Amortised inference.

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
   r))
```

Other examples: Finance model, captcha, brain, etc.

Amortised inference. 1) Learn a proposal q(x; y) parameterized by obs. y via preprocessing.

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
    r))
```

Other examples: Finance model, captcha, brain, etc.

Amortised inference. 1) Learn a proposal $q(x; y)$ parameterized by obs. y via preprocessing. 2) Use $q(x;y_0)$ for any actual observation y_0 later.

We often need to infer posterior of one model multiple times with different observations.

```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
   r))
```

Other examples: Finance model, captcha, brain, etc.

Amortised inference. 1) Learn a proposal $q(x; y)$ parameterized by obs. y via preprocessing. 2) Use $q(x;y_0)$ for any actual observation y_0 later.

neural nets

We often need to infer posterior of one model multiple times with different observations.

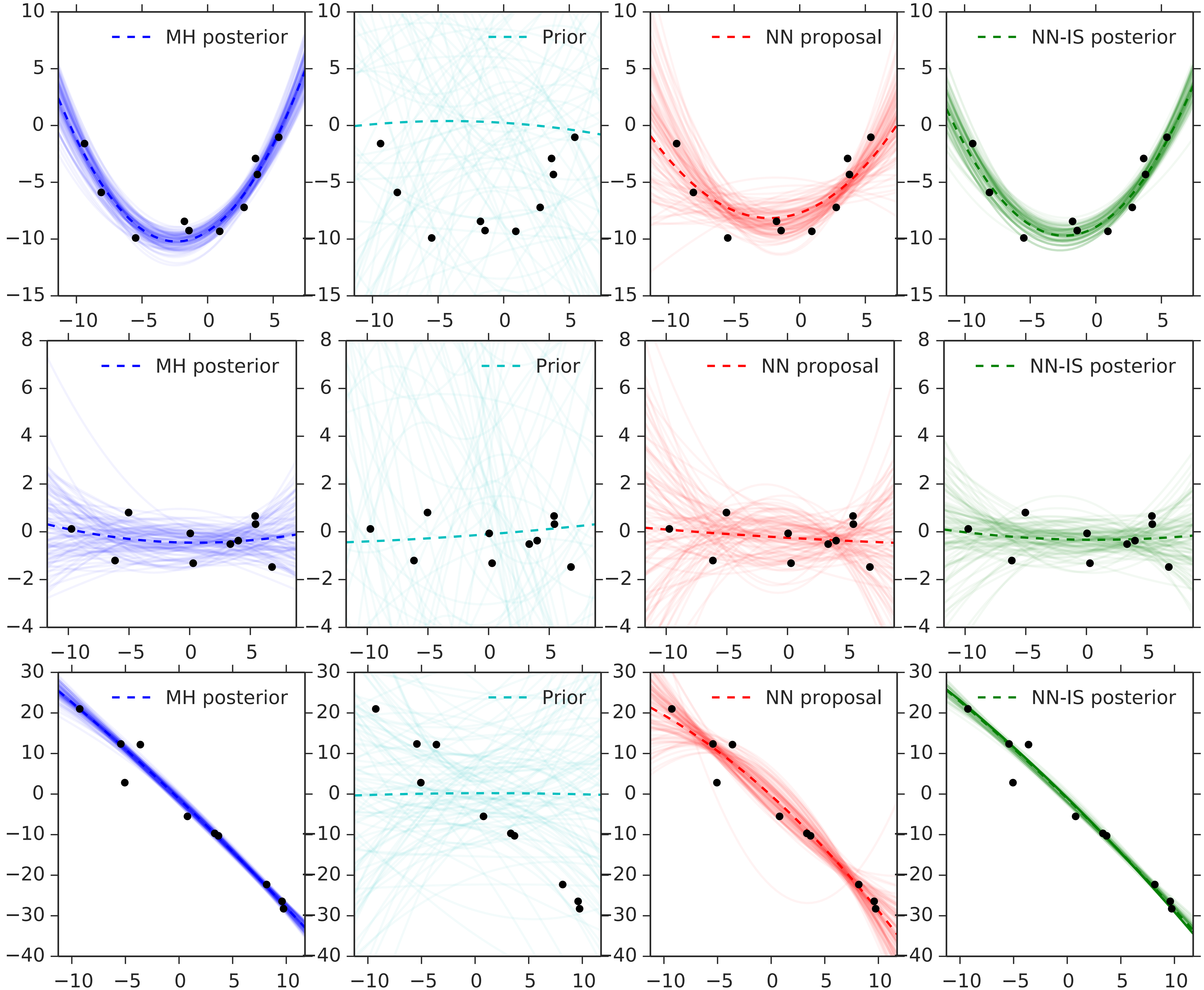
```
(defquery biased-coin []
  (let [r (sample (uniform-continuous 0 1))
        a (observe (flip r) false)
        b (observe (flip r) false)
        c (observe (flip r) true)]
```

sample/object duality and reverse KL

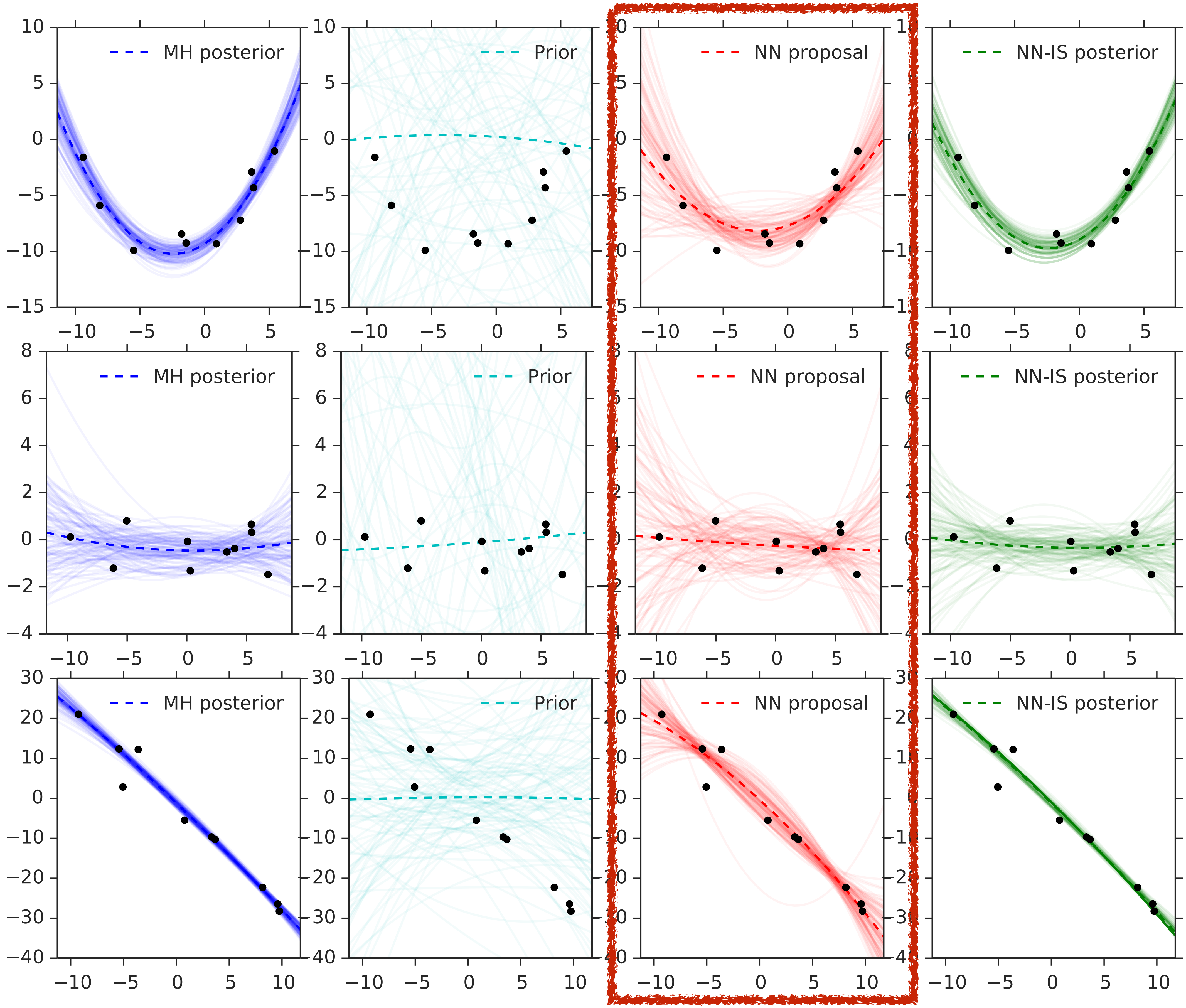
Other examples: Finance model, captcha, brain, etc.

Amortised inference. 1) Learn a proposal $q(x; y)$ parameterized by obs. y via preprocessing. 2) Use $q(x;y_0)$ for any actual observation y_0 later.

neural nets



Model for non-linear regression [Paige et al., ICML16]



Model for non-linear regression [Paige et al., ICML16]

Observed images

(W4kgvQ) (uV7FeWB) (MqhnpT)

more preprocessing

Samples

W4kgvQ	uV7EeWB	MqhnpT
WA4rjvQ	uV7FeWB	MypppT
Woxewd9	mTTEMMm	RIrpES
BKvu2Q	C9QDsoN	rS5FP2B

less preprocessing

Captcha solving [Le et al., AISTATS16]

Learning outcome

Can describe how amortised inference works for models written in math.

Can explain key ideas behind implementing amortised inference for probabilistic programs.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem

Specified by p(x) and p(y|x).
Interested in p(x|y).
But specific y not given yet.

Given:

1. joint dist. p(x,y) for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Differentiable wrt. θ for fixed x,y.
E.g. $q_\theta(x;y) = \text{normal}(x; f_\theta(y), g_\theta(y))$ for neural nets f,g.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is <u>good</u> for <u>most y</u>.

Proposal learning problem

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem tackled by amortised inf.

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Proposal learning problem tackled by amortised inf.

$\boxed{\text{y sampled from p(y)}}$

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

<u>Find θ</u> such that $q_\theta(x;y)$ is <u>good</u> for <u>most y</u>.

Proposal learning problem tackled by amortised inf.

y sampled from p(y)

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Small KL divergence from $p(x|y)$ to $q_\theta(x;y)$.

$\mathrm{KL}[\, p(x|y) \,||\, q_\theta(x;y) \,] = \mathbb{E}_{p(x|y)}[\, \log(p(x|y)/q_\theta(x;y)) \,]$.

Proposal learning problem

$\mathrm{argmin}_\theta\ \mathbb{E}_{p(y)}[\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ]]$.
Solve this by stochastic gradient descent.

inf.

y sampled from p(y)

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Small KL divergence from $p(x|y)$ to $q_\theta(x;y)$.
$\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ] = \mathbb{E}_{p(x|y)}[\ \log(p(x|y)/q_\theta(x;y))\ ]$.

Proposal learning problem

argmin$_\theta$ $\mathbb{E}_{p(y)}$[KL[p(x|y) || q_θ(x;y)]].
Solve this by stochastic gradient descent.

inf.

y sampled from p(y)

Given:

1. joint dist. p(x,y) for latent x and observed y,
2. proposal q_θ(x;y) parameterized by θ and y.

Find θ such that q_θ(x;y) is good for most y.

Small KL divergence from p(x|y) to q_θ(x;y).
KL[p(x|y) || q_θ(x;y)]= $\mathbb{E}_{p(x|y)}$[log(p(x|y)/q_θ(x;y))].

KL divergence from $p_1(x)$ to $p_2(x)$

Denoted by $KL[p_1(x) \,||\, p_2(x)]$.

$KL[p_1(x) \,||\, p_2(x)] := \mathbb{E}_{p_1(x)}[\log(p_1(x)/p_2(x))]$.

Average log ratio.

Measures how $p_1(x)$ is close to $p_2(x)$.

KL divergence from $p_1(x)$ to $p_2(x)$

Denoted by $KL[p_1(x) \,||\, p_2(x)]$.

$KL[p_1(x) \,||\, p_2(x)] := \mathbb{E}_{p_1(x)}[\log(p_1(x)/p_2(x))]$.

Average log ratio.

Measures how $p_1(x)$ is close to $p_2(x)$.

[Q] What is $KL[p(x) \,||\, p(x)]$?

KL divergence from $p_1(x)$ to $p_2(x)$

Denoted by $KL[p_1(x) \mid\mid p_2(x)]$.

$KL[p_1(x) \mid\mid p_2(x)] := \mathbb{E}_{p_1(x)}[\log(p_1(x)/p_2(x))]$.

Average log ratio.

Measures how $p_1(x)$ is close to $p_2(x)$.

[Q] What is $KL[[tt \mapsto .1; ff \mapsto .9] \mid\mid [tt \mapsto .5; ff \mapsto .5]]$?

Proposal learning problem inf.

$\mathrm{argmin}_\theta\ \mathbb{E}_{p(y)}[\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ]]$.
Solve this by stochastic gradient descent.

y sampled from p(y)

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Small KL divergence from $p(x|y)$ to $q_\theta(x;y)$.
$\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ] = \mathbb{E}_{p(x|y)}[\ \log(p(x|y)/q_\theta(x;y))\ ]$.

Proposal learning problem inf.

$\mathrm{argmin}_\theta\, \mathbb{E}_{p(y)}[\mathrm{KL}[\, p(x|y) \,||\, q_\theta(x;y) \,]]$.
Solve this by stochastic gradient descent.

y sampled from p(y)

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Small KL divergence from $p(x|y)$ to $q_\theta(x;y)$.
$\mathrm{KL}[\, p(x|y) \,||\, q_\theta(x;y) \,] = \mathbb{E}_{p(x|y)}[\, \log(p(x|y)/q_\theta(x;y)) \,]$.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[\mathrm{KL}[p(x|y)||q_\theta(x;y)]]$

Stochastic gradient descent for $\mathbb{E}_{p(y)}[\mathrm{KL}[p(x|y)||q_\theta(x;y)]]$

Initialise θ

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

…

(until θ doesn't change)

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Learning rate

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

…

(until θ doesn't change)

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

...

Can't compute, but can approximate.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

…

Can't compute, but can approximate.

Sample (x_1,y_1), …, (x_n,y_n) from $p(x,y)$.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

…

Can't compute, but can approximate.

Sample $(x_1,y_1), \ldots, (x_n,y_n)$ from $p(x,y)$.

$\nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]] \approx -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

…

Can't compute, but can approximate.

Sample $(x_1,y_1), \ldots, (x_n,y_n)$ from $p(x,y)$.

$\nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]] \approx -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

$\theta \leftarrow \theta - 0.01 * \nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

...

Can't compute, but can approximate

Sample (x_1,y_1), ..., (x_n,y_n) from $p(x,y)$.

$\nabla_\theta \mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]] \approx -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$.

exists since $q_\theta(x_i;y_i)$ is differentiable.

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

Repeat the following until θ doesn't change:

1. Sample $(x_1,y_1), \ldots, (x_n,y_n)$ from $p(x,y)$
2. $G \leftarrow -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$
3. $\theta \leftarrow \theta - 0.01 * G$

Proposal learning problem inf.

$\mathrm{argmin}_\theta\ \mathbb{E}_{p(y)}[\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ]]$.
Solve this by stochastic gradient descent.

y sampled from p(y)

Given:

1. joint dist. $p(x,y)$ for latent x and observed y,
2. proposal $q_\theta(x;y)$ parameterized by θ and y.

Find θ such that $q_\theta(x;y)$ is good for most y.

Small KL divergence from $p(x|y)$ to $q_\theta(x;y)$.
$\mathrm{KL}[\ p(x|y)\ ||\ q_\theta(x;y)\ ] = \mathbb{E}_{p(x|y)}[\ \log(p(x|y)/q_\theta(x;y))\ ]$.

What about probabilistic programs?

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

Repeat the following until θ doesn't change:

1. Sample $(x_1,y_1), \ldots, (x_n,y_n)$ from $p(x,y)$
2. $G \leftarrow -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$
3. $\theta \leftarrow \theta - 0.01 * G$

Stochastic gradient descent for $\mathbb{E}_{p(y)}[KL[p(x|y)||q_\theta(x;y)]]$

Initialise θ

Repeat the following until θ doesn't change:

1. Sample $(x_1,y_1), \ldots, (x_n,y_n)$ from $p(x,y)$
2. $G \leftarrow -1/n * \sum_{i=1..n} \nabla_\theta \log q_\theta(x_i;y_i)$
3. $\theta \leftarrow \theta - 0.01 * G$

How to sample y?

Sample/observe duality

To sample observations, just replace sample by observe.

```
(defquery biased-coin []
  (let [r (sample
            (uniform-continuous 0 1))
        a (observe (flip r) true)
        b (observe (flip r) true)
        c (observe (flip r) true)]
   r))
```

Sample/observe duality

To sample observations, just replace sample by observe.

```
(defquery biased-coin-joint []
  (let [r (sample
             (uniform-continuous 0 1))
        a (sample (flip r))
        b (sample (flip r))
        c (sample (flip r))]
    [r [a b c]]))
```

Sample/observe duality

To sample observations, just replace sample by observe.

```
(defquery biased-coin-joint []
  (let [r (sample
             (uniform-continuous 0 1))
        a (sample (flip r))
        b (sample (flip r))
        c (sample (flip r))]
    [r [a b c]]))
```

[Q] Define a sensible $q_\theta(r;a,b,c)$.

References

1. Inference networks for sequential Monte Carlo in graphical models. Paige et al. ICML'16.
2. Inference compilation and universal probabilistic programming. Let et al. AISTATS'17.