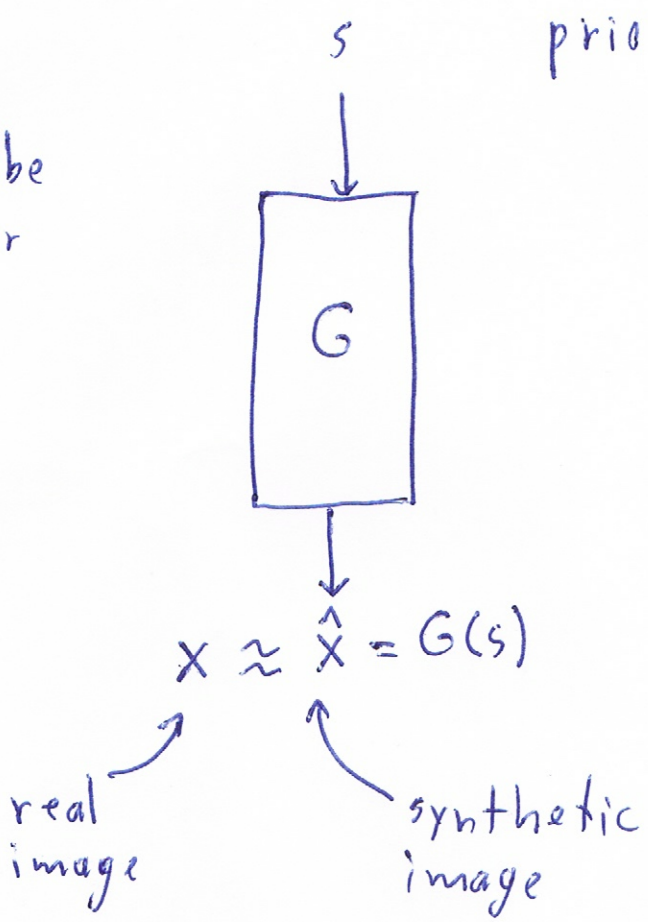
Generative Models



G may be given or learned.

$\dim(s) << \dim(x)$

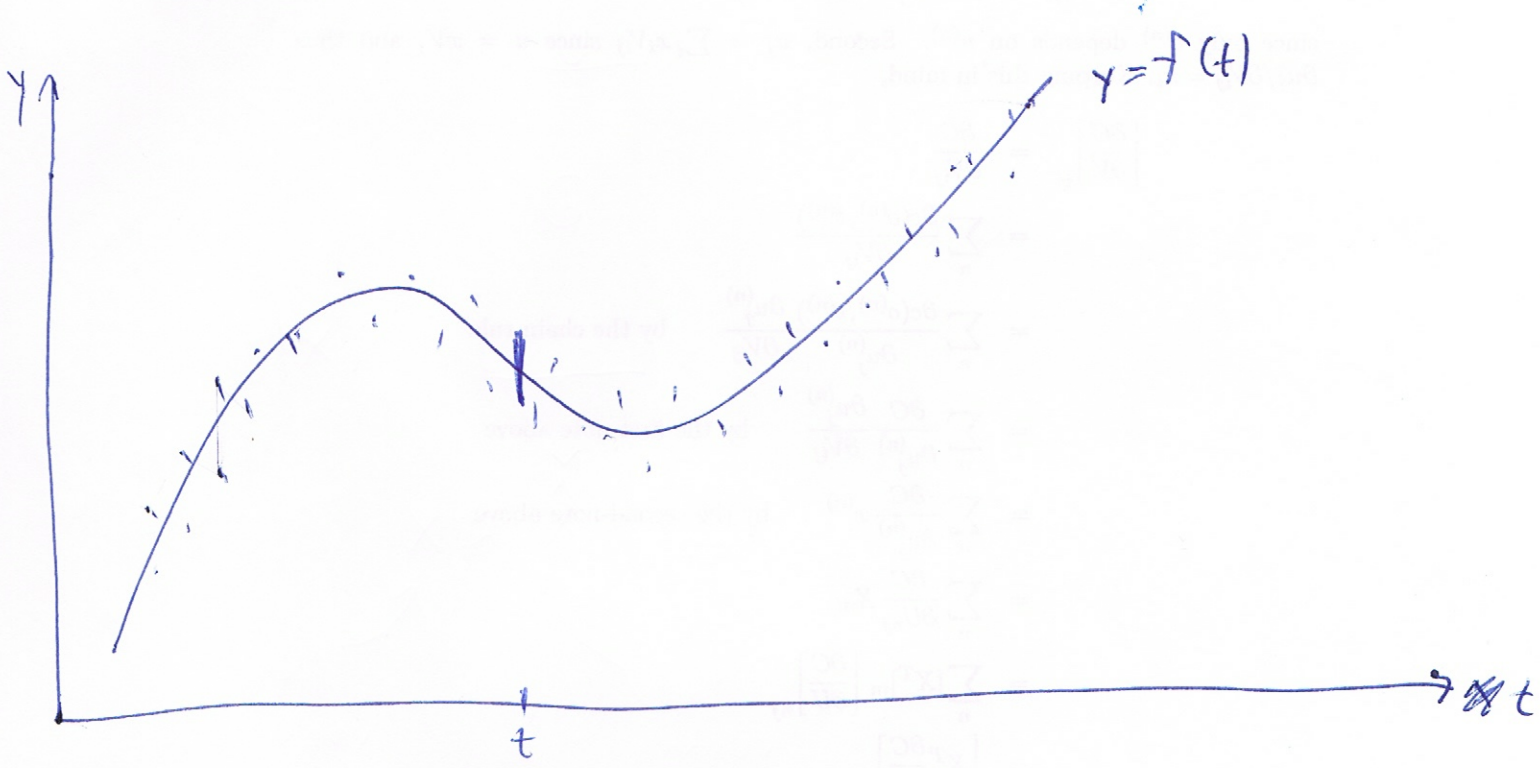
Pretend that we ~~can~~ randomly generate images "near" $\vec{x}$, with probability $P(x|\vec{x})$

eg. $P(x|\vec{x}) = \mathcal{N}(x|\vec{x}, \sigma) = \frac{e^{-\|x-\vec{x}\|/2\sigma^2}}{(\sqrt{2\pi}\sigma)^k}$

$k = \dim(x)$

so, images similar to $\vec{x}$ are most likely.

compare to regression

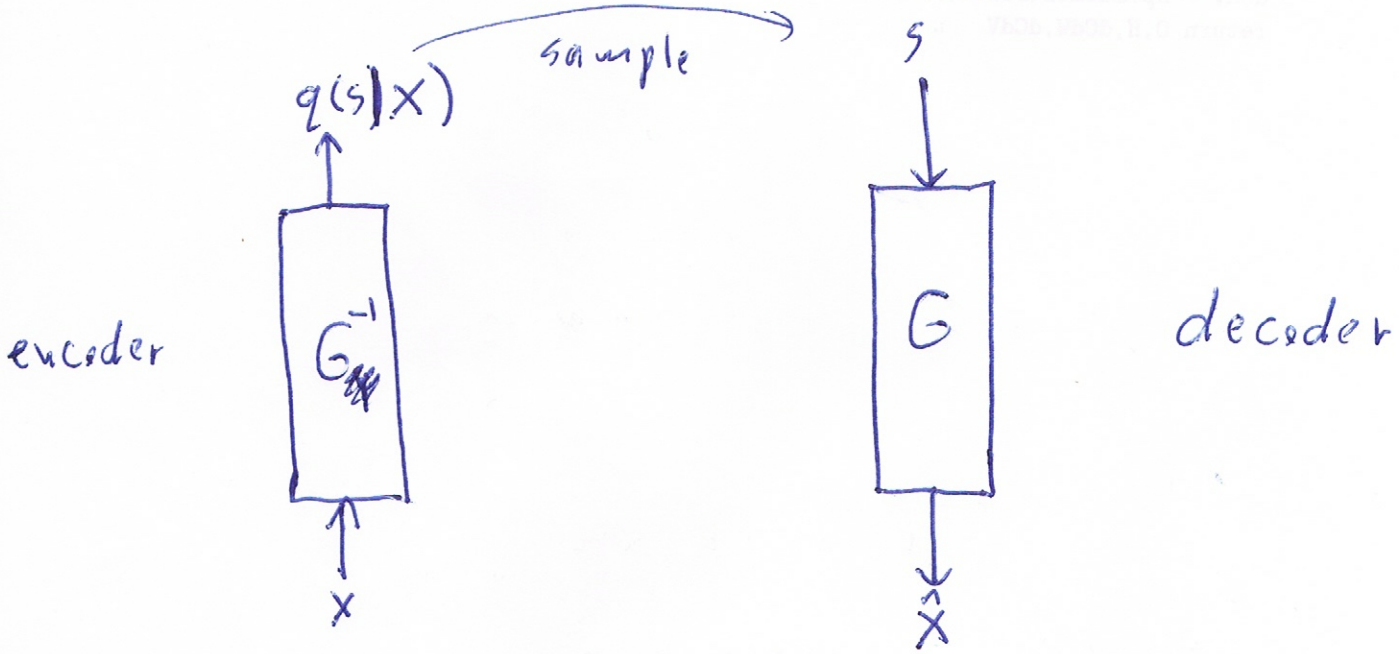


$$p(y|t) = \mathcal{N}(y|f(t), \sigma)$$

ie, given t, we predict a distribution for y. Typically, $f(t)$ is the mean of the distribution. In simple models, σ is fixed, independent of t.

Inverse Graphics

- compute (estimate) $p(s|x)$
- intractable in general.
- approximate $p(s|x)$ using a N.N. (encoder).



Train G^{-1} (+ maybe G)

so that $x \approx \hat{x}$ for all training data.

Backprop through G & G^{-1}. How?

Variational Autoencoders

Given training data $x^1 \dots x^N$,

train so that $P(x^1, \dots, x^N)$ is maximal.

$$P(x) = \sum_s P(x,s) = \sum_s P(x|s) \cdot P(s)$$

↑ huge sum

↑ ~~known~~ learnable

↑ known

Assume iid data, so $P(x^1 \dots x^N) = \prod_n P(x^n)$

$$\therefore \log P(x^1 \dots x^N)$$

$$= \sum_n \log P(x^n)$$

$$= \sum_n \log \sum_s P(x^n, s) \quad \left.\right\} \text{ sum inside log}$$

$$= \sum_n \log \left[\sum_s q(s|x^n) \cdot \frac{P(x^n, s)}{q(s|x^n)} \right]$$

$$\geq \sum_n \sum_s q(s|x^n) \log \frac{p(x^n, s)}{q(s|x^n)} \stackrel{\text{def}}{=} \mathcal{L}$$

- true for any distribution, q, because log is concave.

$\blacksquare$

exercise:

$$\mathcal{L} = \log p(x^1 \ldots x^N) - \sum_n KL[q(s|x^n), p(s|x^n)]$$

$\therefore$ maximizing $\mathcal{L}$ minimizes the KL, i.e., difference between $q(s|x^n)$ & $p(s|x^n)$

i.e., makes $q(s|x^n)$ a better approximation of $p(s|x^n)$.

Problem

The sum over S is huge & can't be evaluated. But we can approximate (to arbitrary accuracy):

$$\mathcal{L} = \sum_{n=1}^{N} \mathop{E}_{S\sim q} \log \frac{P(X^n, S)}{q(S|X^n)} \qquad *$$

$$\approx \sum_{n=1}^{N} \frac{1}{L} \sum_{\ell=1}^{L} \log \frac{P(X^n, S_\ell^n)}{q(S_\ell^n|X^n)}$$

where $S_\ell^n \sim q(S|X^n)$

If N is large, can use $L=1$.

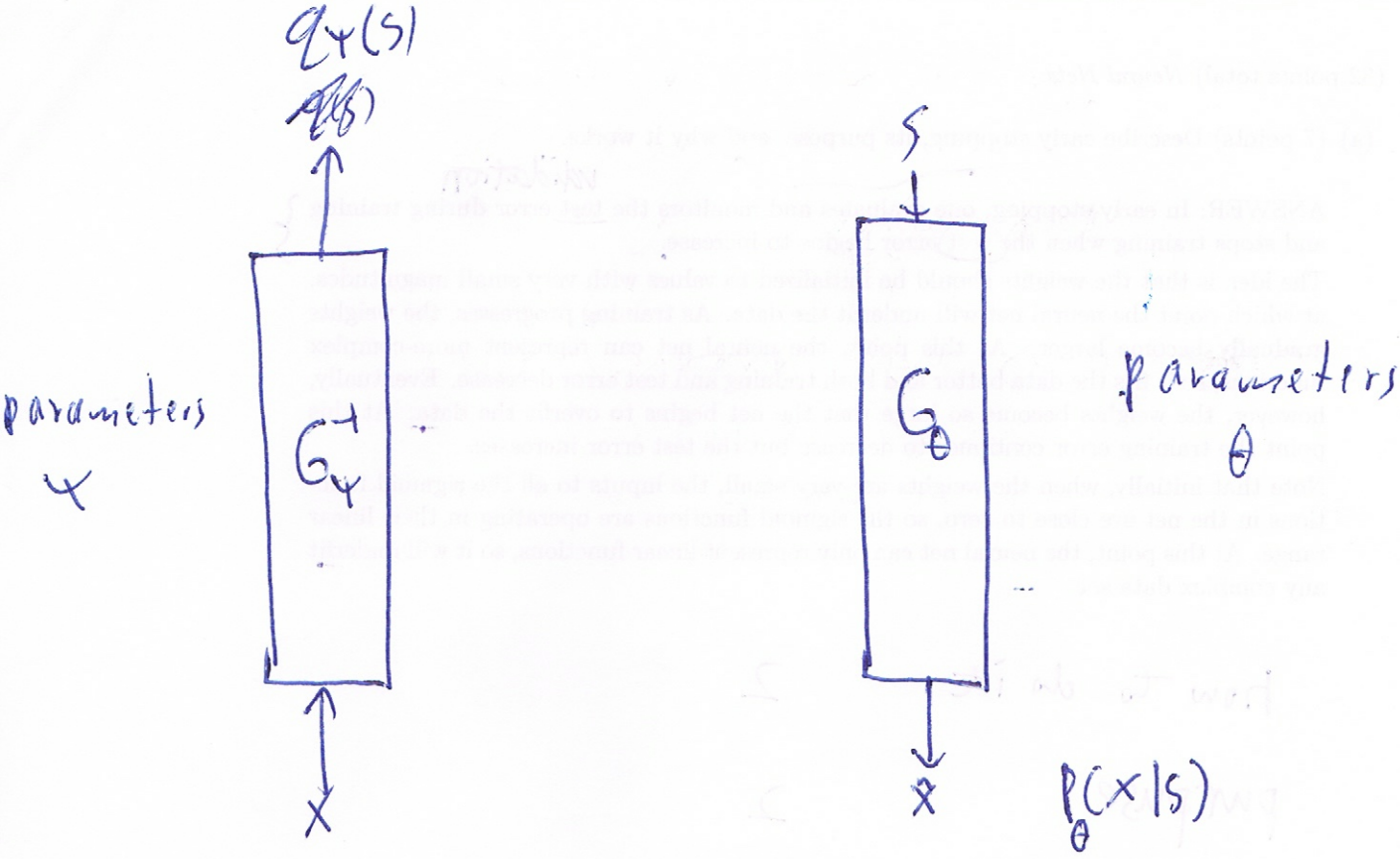
$$\therefore \mathcal{L} \approx \sum_n \log \frac{P(x^n, s^n)}{q(s^n|x^n)}$$

where $s^n \sim q(s|x^n)$

$$= \sum_n \log \frac{P(x^n|s^n) \cdot P(s^n)}{q(s^n|x^n)}$$

$$= \sum_n \left[\log P_\theta(x^n|s^n) + \log P(s^n) - \log q_\phi(s^n|x^n) \right]$$

Note: $\nabla_\theta \mathcal{L} = \sum_n \nabla_\theta \log P_\theta(x^n|s^n)$



$$p_\theta(x|s) = p(x|\hat{x} = G_\theta(s))$$

$$= \mathcal{N}(x \mid G_\theta(s), \alpha)$$

$$= \frac{e^{-(x - G_\theta(s))^2/2\alpha^2}}{(\sqrt{2\pi}\alpha)^k} \qquad k = \dim(x)$$

$$\log p_\theta(x|s) = -(x - G_\theta(s))^2/2\alpha^2 - k \log(\sqrt{2\pi}\alpha)$$

$$\nabla_\theta \mathcal{L} = \sum_n \nabla_\theta \log P_\theta(x^n | s^n)$$

$$= -\sum_n \nabla_\theta \left(x - G_\theta(s)\right)^2 / 2\sigma^2$$

$$= \sum_n \left(x - G(s)\right) \cdot \nabla_\theta G(s)$$

so, back-propagate through NN G

easy

note: G must be differentiable.

$\nabla_\psi \mathcal{L}$ is harder

$\mathcal{L}$ depends on the samples, s^n, which ~~in turn depend on~~ we drew from $q_\psi(s|x)$, which depends on ψ.

Go back to *.

$$\mathcal{L} = \sum_n \mathop{E}_{s \sim q_\psi} \log \frac{p_\theta(x^n, s)}{q_\psi(s|x^n)}$$

want $\nabla_\psi \mathcal{L}$

$$\mathcal{L} = \sum_n \mathop{E}_{s \sim q_\psi} \log p_\theta(x^n, s) \Big\} \text{ hard}$$

$$- \mathop{E}_{s \sim q_\psi} \log q_\psi(s|x^n) \Big\} \text{ easy}$$

In general

$$\nabla_\psi \underset{s\sim q}{E} f(s)$$

where q depends on ψ.

Two approaches, depending on whether s is continuous or discrete.

(1) Continuous s. (reparameterization trick).

example: Suppose $q = \mathcal{N}(\mu, \Sigma)$ where μ & Σ are the outputs of a NN (typically assume Σ is diagonal)

example: Suppose s is 1D & $q = \mathcal{N}(\mu, \sigma)$

(easily extended to any dimension)

Continuous case

(reparameterization trick)

Example. Suppose s is 1D

& $q = \mathcal{N}(\mu, \sigma)$

(easily extended to any dimension)

where μ & σ depend on ψ.

(eg. μ & σ are the output of a NN with weights ψ).

Then,

$$\mathop{E}_{s \sim q} f(s) = \mathop{E}_{\xi \sim \mathcal{N}(0,1)} f(\mu + \sigma \xi)$$

$$\therefore \nabla_\psi \underset{s\sim q}{E} f(s)$$

$$= \underset{\varepsilon\sim N(0,1)}{E} \nabla_\psi f(\mu + \sigma\varepsilon)$$

$$= \underset{\varepsilon\sim N(0,1)}{E} \frac{\partial f(s)}{\partial s}\bigg|_{s=\mu+\sigma\varepsilon}$$

$$= \underset{\varepsilon\sim N(0,1)}{E} \cdot f'(\mu+\sigma\varepsilon)\nabla_\psi(\mu+\sigma\varepsilon)$$

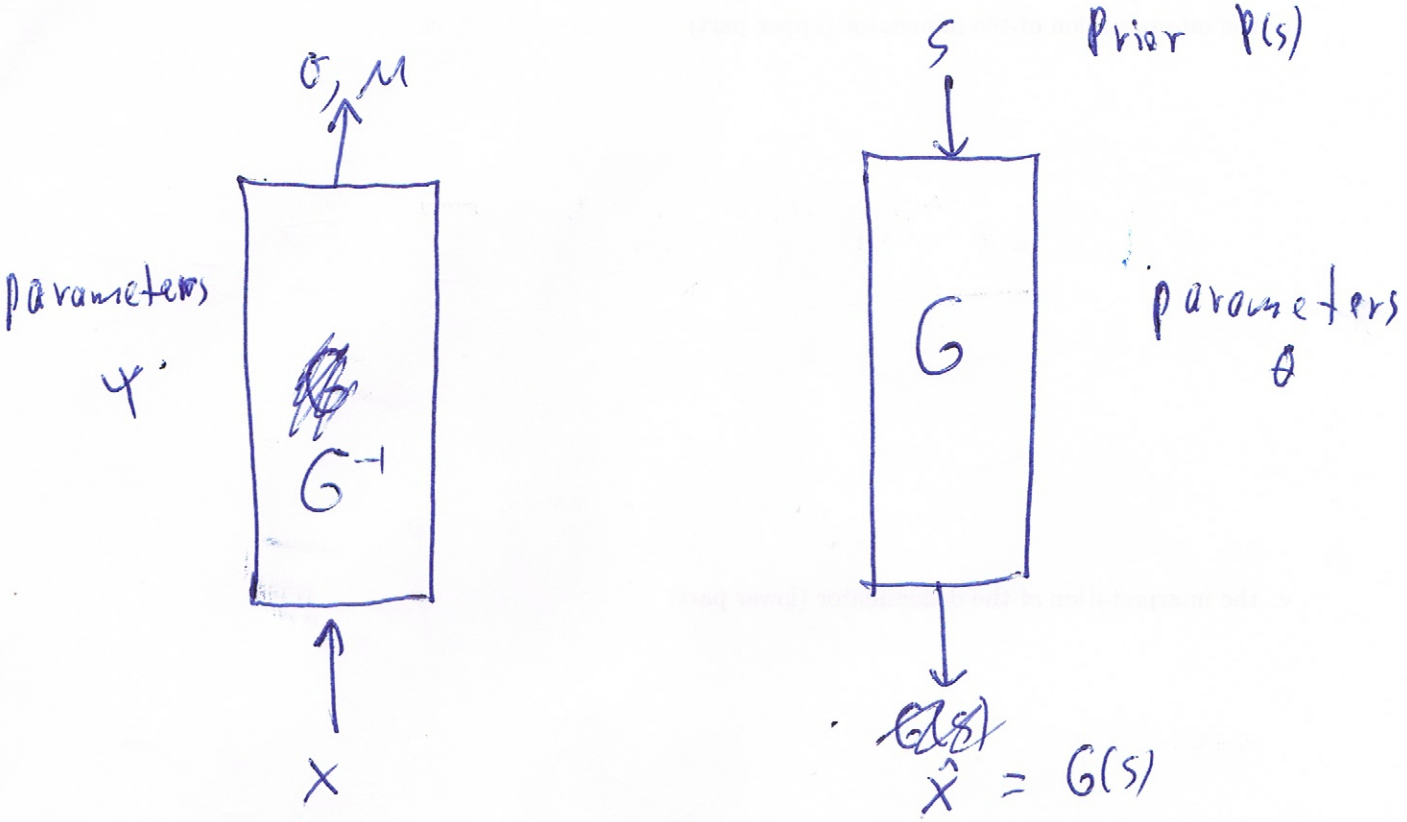
$$= \underset{\varepsilon\sim N(0,1)}{E} f'(\mu+\sigma\varepsilon)(\nabla_\psi \mu + \varepsilon \cdot \nabla_\psi \sigma)$$

$$\approx \frac{1}{L}\sum_{l=1}^{L} f'(\mu+\sigma\varepsilon_l)(\nabla_\psi \mu + \varepsilon_l \nabla_\psi \sigma)$$

where $\varepsilon_l \sim N(0,1)$

Note: f must be differentiable

Variational Auto encoders



$$P(x|\hat{x}) = \mathcal{N}(x|\hat{x}, \alpha)$$

typically $P(s) = \mathcal{N}(s|0,1)$

$$\log P(x|s) = \log P(x|\hat{x})$$

$$= -(x-\hat{x})^2/2\alpha^2 + C$$

$$= \propto (x - G(s))^2/2\alpha^2 + C$$

Recall

$$\mathcal{L} = \sum_n \mathop{E}_{s \sim q(s|x^n)} \log \frac{P_\theta(x^n, s)}{q_\phi(s|x^n)}$$

$$= \sum_n \mathop{E}_{s \sim q(s|x^n)} \log P_\theta(x^n, s)$$

$$- \sum_n \mathop{E}_{s \sim q(s|x^n)} \log q_\phi(s|x^n)$$

$$= \sum_n \mathop{E}_{s \sim q(s|x^n)} \log P_\theta(x^n|s)$$

$$+ \sum_n H(q(s|x^n)) \quad \leftarrow \text{Entropy of } q(s|x^n)$$

(usually a simple formula!)

eg. If $q(s|x) = \mathcal{N}(s|\mu, \sigma)$

then $H(s|x) = \log \sigma + c$

where $c = \frac{1}{2} + \frac{1}{2}\log(2\pi)$

$$\therefore \nabla_\psi \mathcal{L} \qquad f_n(s)$$

$$= \sum_n \nabla_\psi \underset{s\sim q(s|x^n)}{E} \overbrace{\log p_\theta(x^n, s)}$$

$$+ \sum_n \nabla_\psi H(q(s|x^n))$$

$$= \sum_n \nabla_\psi \underset{s\sim q(s|x^n)}{E} f_n(s)$$

$$+ \sum_n \nabla_\psi \log \sigma_n$$

where $s|x^n \sim \mathcal{N}(s|\mu_n, \sigma_n)$

$$= \sum_n \nabla_\psi \underset{\varepsilon\sim\mathcal{N}(0,1)}{E} f_n(\mu_n + \sigma_n \varepsilon) \Big\} \text{ hard.}$$

$$+ \sum_n \nabla_\psi \log \sigma_n \Big\} \underset{\text{(back prop through } G^{-1}\text{)}}{\text{easy}}$$

The hard part

$$\sum_n \nabla_\psi \mathop{E}_{\xi \sim N(0,1)} f_n(\mu_n + \sigma_n \xi)$$

$$= \sum_n \mathop{E}_{\xi \sim N(0,1)} \nabla_\psi f_n(\mu_n + \sigma_n \xi)$$

$$= \sum_n \mathop{E}_{\xi \sim N(0,1)} f_n'(\mu_n + \sigma_n \xi) \cdot \nabla_\psi (\mu_n + \sigma_n \xi)$$

$$\approx \sum_n \frac{1}{L} \sum_{\ell=1}^{L} f_n'(\mu_n + \sigma_n \xi_n^\ell) \nabla_\psi (\mu_n + \sigma_n \xi_n^\ell)$$

where $\xi_n^\ell \sim N(0,1)$

(can use $L=1$ when N is large)

$$\approx \sum_n f_n'(\mu_n + \sigma_n \xi_n)(\nabla_\psi \mu_n + \xi_n \cdot \nabla_\psi \sigma_n)$$

f_n': backprop through G

$\nabla_\psi \mu_n$, $\nabla_\psi \sigma_n$: backprop through G^{-1}

where $\xi_n \sim N(0,1)$

Discrete Case

(s is discrete)

Recall the general problem:

evaluate $\nabla_\psi \underset{s\sim q}{E} f(s)$

where (the parameters of) q depends on ψ.

Simple example:

$s \in \{0, 1\}$

$q(s=1) = \alpha$

$q(s=0) = 1-\alpha$

where α depends on ψ.

eg, α is the output of a neural net with weights ψ.

Note that even though s is discrete, the expectation over s is differentiable as long as α is differentiable in ψ.

eg. in the simple example above,

$$\begin{aligned} \underset{s\sim q}{E}\, f(s) &= q(1)f(1) + \sum_{s} q(s) f(s) \\ &= q(1)f(1) + q(0)f(0) \\ &= \alpha \cdot f(1) + (1-\alpha)\cdot f(0) \\ &= \alpha \cdot [f(1) - f(0)] + f(0) \end{aligned}$$

$$\therefore \nabla_\psi \underset{s\sim q}{E} f(s) = [f(1) - f(0)] \cdot \nabla_\psi \alpha$$

The problem is when s is a vector. Then the expectation (a sum) has a huge number of terms (eg, 2^{100}), far too many to enumerate.

But, because it is an expectation, we can try to approximate it using sampling.

~~Alas, for discrete distributions, this almost always returns 0 as the estimate of the gradient.~~

$$\nabla_\psi \mathop{E}_{s\sim q} f(s) \simeq \nabla_\psi \frac{1}{L} \sum_{\ell=1}^{L} f(s_\ell) \quad \text{where } s_\ell \sim q$$

Since q depends on ψ, how do we differentiate through the sampling process?

re Parameterization Trick doesn't work

example: $s \in \{0, 1\}$

$$\left.\begin{aligned} q(s=1) &= \alpha \\ q(s=0) &= 1-\alpha \end{aligned}\right\} \alpha \text{ depends on } \psi$$

To use reparam. trick, must make s a function of α & some other random variable, ε, ind. of ψ

eg. $s = [\varepsilon < \alpha]$ where $\varepsilon \sim U(0,1)$

$\therefore \Pr(\varepsilon < \alpha) = \alpha$

note $[\text{True}] = 1$

$[\text{False}] = 0$

$$\therefore \nabla_\psi \mathop{E}_{s\sim q} f(s)$$

$$= \nabla_\psi \mathop{E}_{\varepsilon \sim u(0,1)} f([\![\varepsilon < \alpha]\!])$$

$$= \mathop{E}_{\varepsilon \sim u(0,1)} \nabla_\psi f([\![\varepsilon < \alpha]\!])$$

$$= \mathop{E}_{\varepsilon \sim u(0,1)} \frac{\partial}{\partial \alpha} f([\![\varepsilon < \alpha]\!]) \cdot \frac{\partial \alpha}{\partial \psi}$$

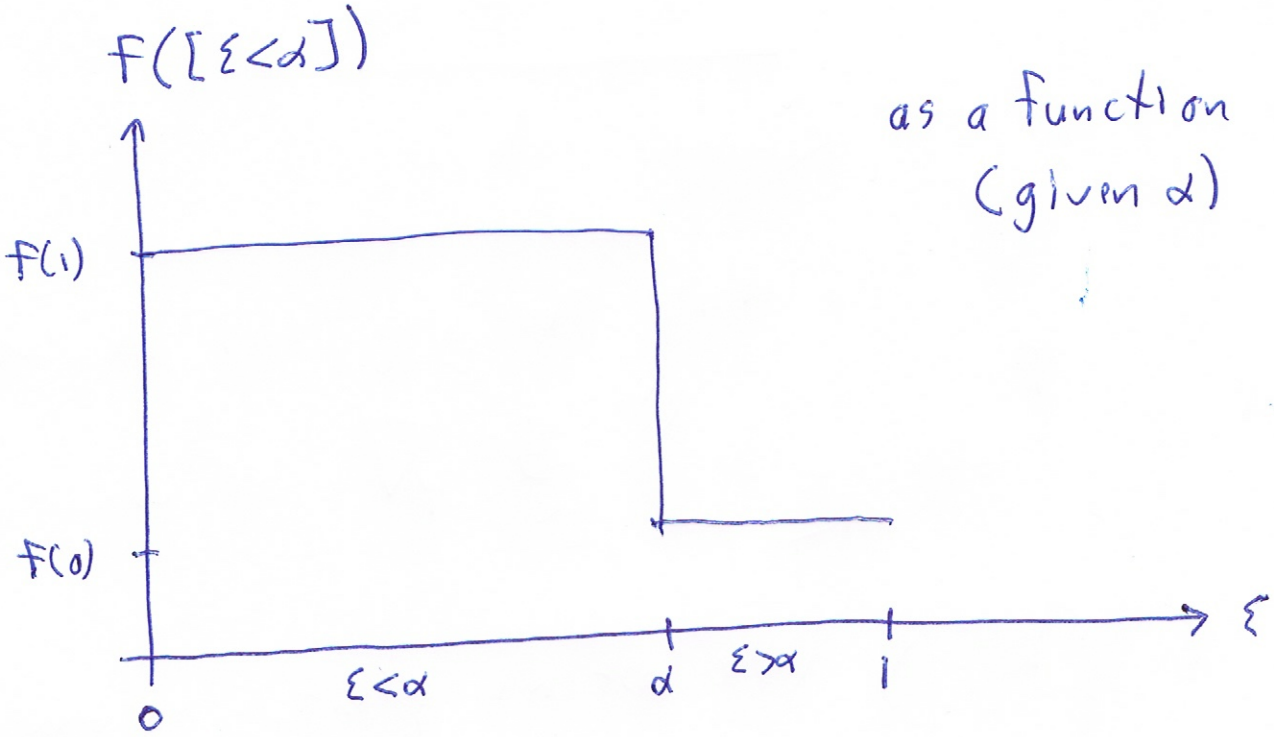
} attn to param. trick

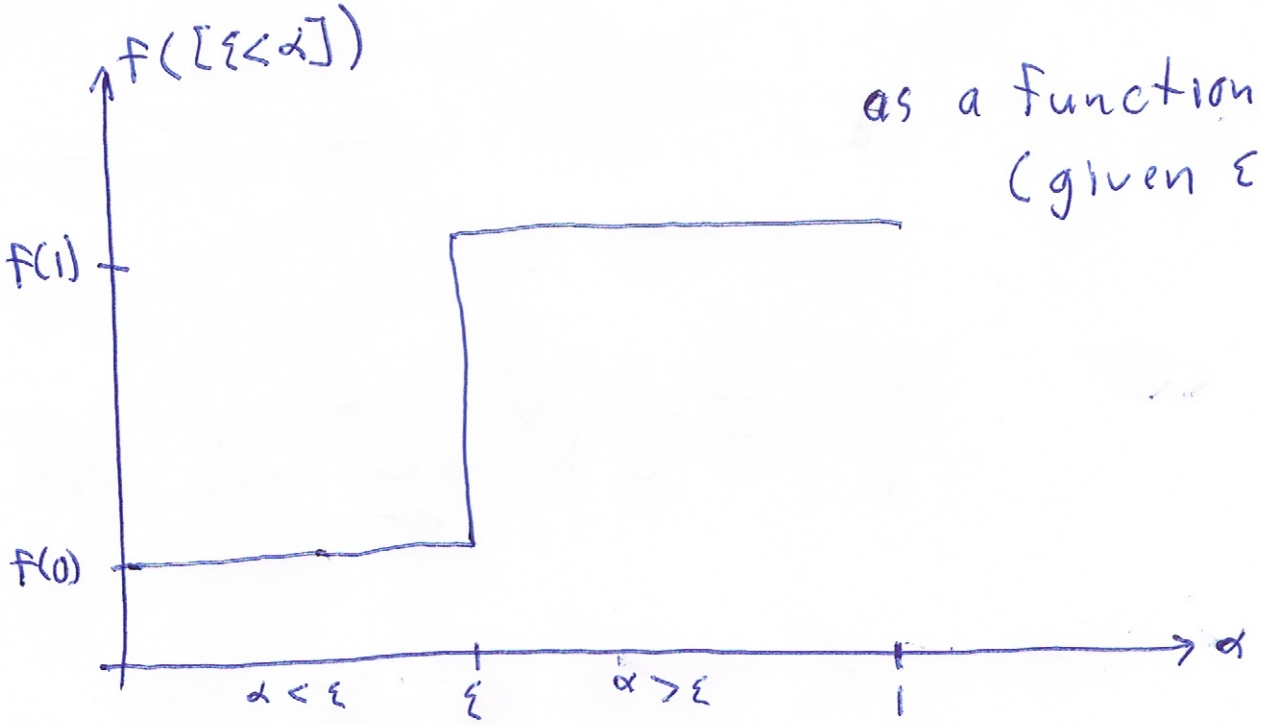
but $f([\![\varepsilon < \alpha]\!])$ is constant a.e.

$$f([\![\varepsilon < \alpha]\!]) = \begin{cases} f(1) & \text{if } \varepsilon < \alpha \\ f(0) & \text{o.w.} \end{cases}$$

$$\therefore \frac{\partial f([\![\varepsilon < \alpha]\!])}{\partial \alpha} = \begin{cases} 0 & \text{if } \varepsilon \neq \alpha \\ \infty & \text{a.e. if } \varepsilon = \alpha \end{cases}$$

In Pictures





$$\therefore \frac{\partial f([\xi<\alpha])}{\partial \alpha} = 0 \quad \text{almost everywhere}$$

$$\therefore \nabla_\psi \mathop{E}_{s\sim q} f(s)$$

$$= \mathop{E}_{\xi\sim U(0,1)} \frac{\partial f([\xi \le \alpha])}{\partial \alpha} \cdot \frac{\partial \alpha}{\partial \psi}$$

$$\simeq \frac{1}{L} \sum_{\ell=1}^{L} \frac{\partial f([\xi_\ell \le \alpha])}{\partial \alpha} \frac{\partial \alpha}{\partial \psi} \quad \text{where } \xi_\ell \sim U(0,1)$$

$$= \begin{cases} 0 & \text{if all } \xi_\ell \neq \alpha \\ \infty & \text{if some } \xi_\ell = \alpha \end{cases}$$

ie, a very high variance estimate.

what can we do?