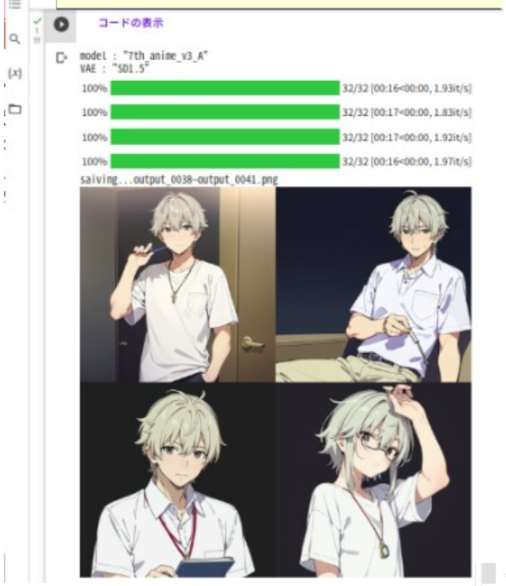
生成モデル

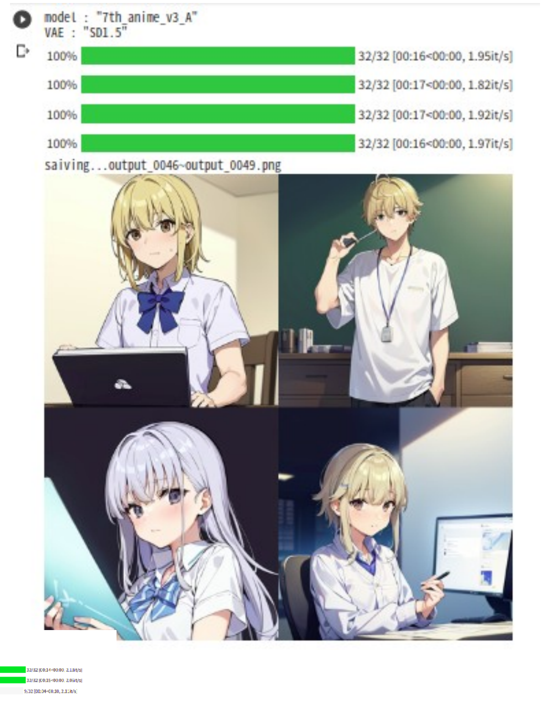
AI 画像

文章を入力してそれに対応した画像を出力する．
例えば「 uematsu 」と入れた結果の画像が以下．
(毎回ランダムなので実行する度に違う画像が出力され同じ画像は出力されない)
と，言うより出力できない．

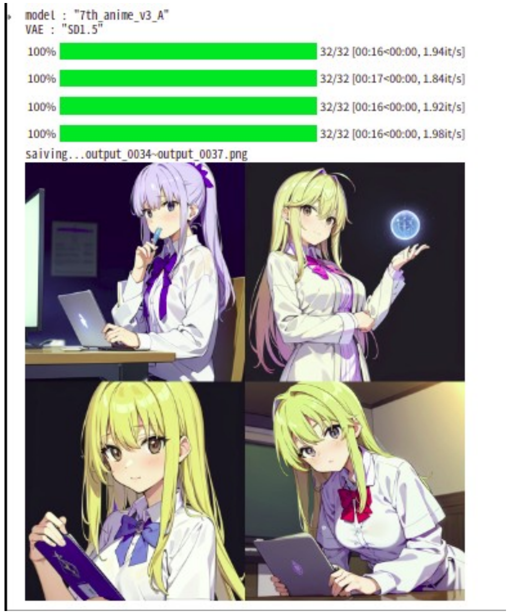
" ((masterpiece, best quality)),((A Certain Magical Index)),system engineer, ((mature male)), ((uematsu))

なんか美少年



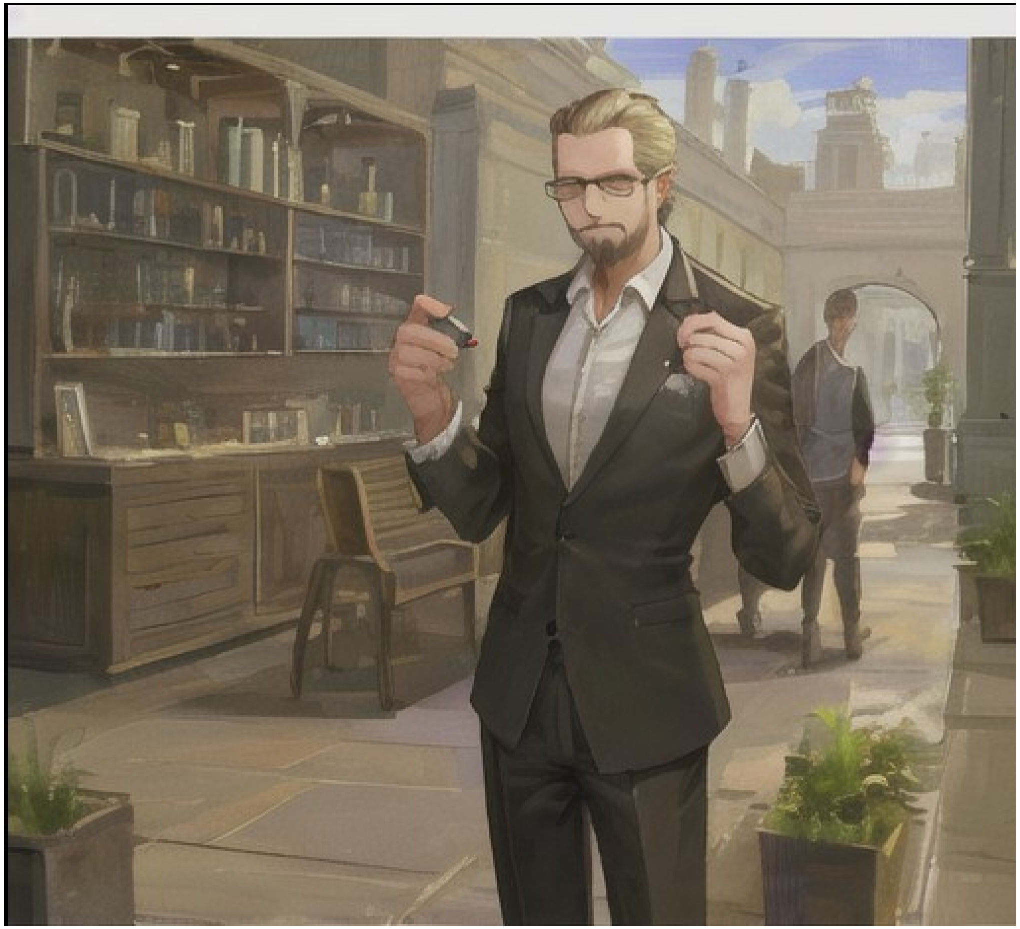


女性の”ウエマツ”も



追記：今は GPU の課金が必須なので Python 環境だけでは動かない．

生成の呪文



■ 私のプロフィール画像
私の性格や特徴，職業などのプロフィールを入力して生成した画像です．
さて，なんて入力したでしょうか？

欲しい画像を生成できるように，大喜利的な能力が今後は必要になる能力かもしれない．

欲しいモノを生成させるのに適切な呪文が唱えられる能力．
または，全く新しい価値を生成する呪文を唱えられる能力．

生成 AI

文章を入力するとそれに関連した画像を AI が生成してくれる

出力画像

欲しい画像

街中を爆走する
ホウキに乗った少女

AI が解析

※ 説明に都合の良い画像が生成できなかったので，既存の画像を流用しました．
実際は AI が世に出ていない画像を生成する (生み出す)

少し前までは無料で使えてたけど GPU への課金が
必須になってしまったので色々試せなくなって．

生成モデル

世の中のあらゆる画像は背後にデータを生成する装置があって確率的に画像を出力しているのではないか？

と，いうふうに考えて ...

背後にある確率分布を知る事が我々が AI で実現したい目的 .
→「街中で爆走するホウキに乗った少女」を出力する装置の確率分布が知りたい .
(欲しい画像のデータを発生させやすい確率をもった）

※ 赤と白の 2 色の画像が欲しければ , 1/2 の確率で赤と白が出力する装置など

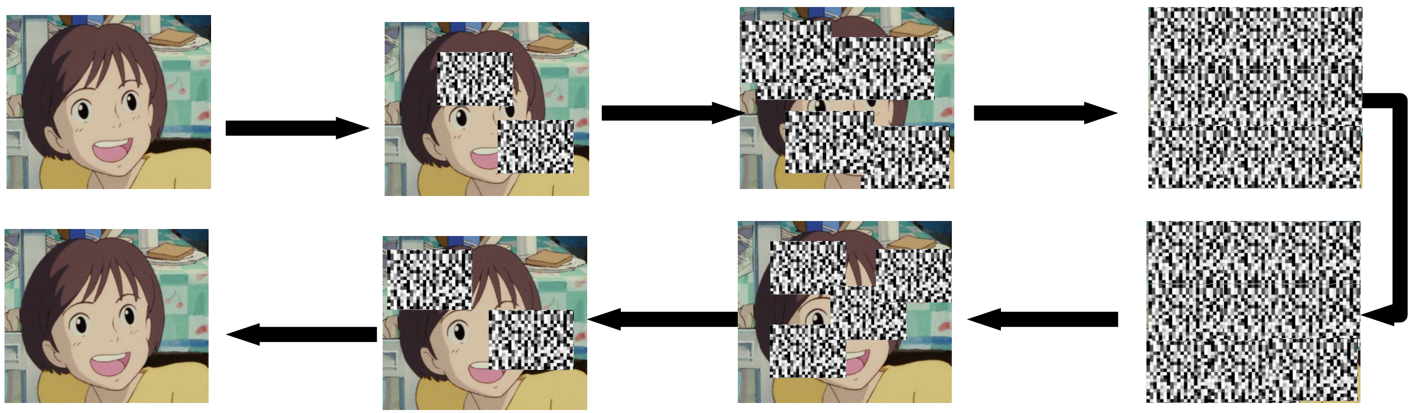
拡散モデル

■ 思想は単純

構造物を壊して全てノイズにする　→　ノイズだらけの無秩序な状態から秩序ある構造物の作り方を学習する

少しずつノイズをかけて徐々に画像を汚していく．

最終的にはノイズだけの画像になる．



生成は完全なノイズからスタート．逆拡散過程によって少しずつノイズを除去．
時効 t のノイズを推測させて少し綺麗な画像を作成，繰り返していくと最終的には綺麗な画像となる．

少女の絵を作るにはどうノイズを除去すれば良いか学習する．
汚した画像を工程を逆行しながら学習していく．

構造物 (秩序ある) 状態からノイズだらけ (無秩序) な状態を作る．次に無秩序な状態から秩序ある状態を作る方法を学習する．　少女の画像はこういう風にノイズを除去すれば良いよね．と言う事がわかる．
Stable Diffusion などで文章を入れたら，ノイズ状態だけの無秩序な状態から，文章に応じたノイズ除去をする事により新たな構造物を作る事ができる．その過程は確率的にランダムにノイズを除去するので同じ画像は生成されない．
(少女ならこういう感じに戻して，ホウキも文章にあったからホウキっぽく少し除去して ... など)

Stable Diffusion~ ステーブル・ディフュージョン

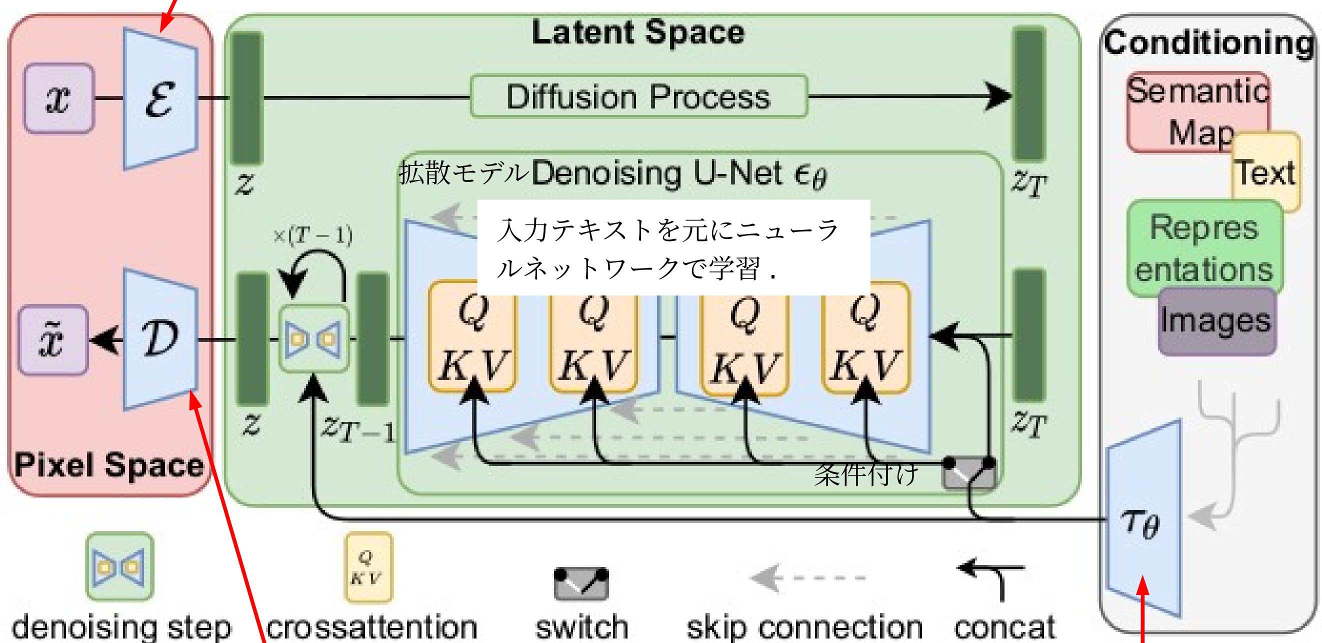
■ 構成する 3 つの要素

1. 拡散モデル (U-Net)
NN で実際に学習 & 生成する機能
※ 正確には画像ではなくてノイズを出力する

2. VAE
拡散モデルで高画質画像をそのまま計算すると物凄い計算量になるので情報を集約して学習して，生成した画像を元の解像度に戻す．
集約 : エンコーダ
戻し：デコーダ

3. Text Encoder & QKV
テキストベクトルに変換する．
学習済みの「Transformer」を使用して最終出力の層を QKV に渡す．
そこでは文章を条件付けしてノイズを除去していく．
※QKV はモデル内の情報と文章情報をリンクさせている．

■ エンコーダ
計算速度を確保するため，画像の情報を集約して小さいサイズにする．
実際の学習や生成は集約した小さいサイズで実施する．



■ デコーダ
集約された画像を元のサイズに戻す．
生成した情報を展開して画像にする．

入力されたテキスト

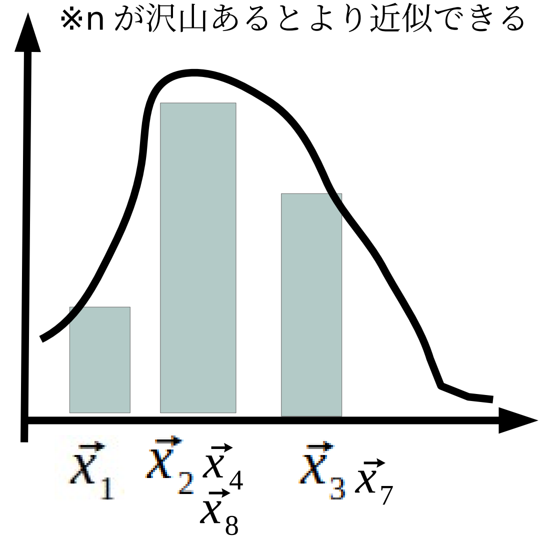
“High-Resolution Image Synthesis with Latent Diffusion Models"
Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B. (CVPR'22)

生成モデル

データを生み出す確率分布　$\vec{x} \sim P(\vec{x})$　を知ることが目的．

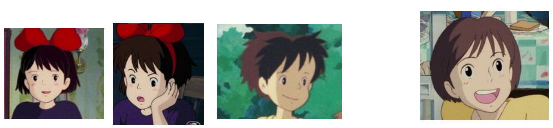
世の中の全ての画像は持っていないので，手元にある少ない画像から推測してやる必要がある．

ヒストグラムを書くと $P(\vec{x})$ を
近似しているのでは？
※n が沢山あるとより近似できる



持っているデータは n 個の少女の画像．
これら少女の画像を学習して新しい画像を生み出す．

$$\vec{x}_1, \vec{x}_2, \vec{x}_3, , , , , \vec{x}_n = \text{データ（ベクトル）}$$



$$P_0(\vec{x}) = \frac{1}{n} \sum_{i=1}^{n} \delta(\vec{x} - \vec{x}_1)$$

経験分布：　$P(\vec{x})$ は分からないが，おそらく　$P_0(\vec{x})$ は似ているはず．
　$P(\vec{x})$ は神様ではないので分からないがデータから推測する　$P_0(\vec{x})$ ならわかる．

ボルツマンマシン

$P(\vec{x})$ のモデルは分からないので知っている確率分布や関数で近似する．

$P(\vec{x}) \approx \underbrace{q(\vec{x})}_{\text{生成モデル}}$　データに関数を合わせられる (学習できる) ように θ をもうけてパラメータで可変可能にする．

ボルツマンマシン

$$q_\theta(\vec{x}) = \frac{1}{Z\theta} \exp\left(\underbrace{-E_\theta(\vec{x})}_{\text{エネルギー}}\right)$$

$E_\theta(\vec{x})$ は 2 次関数でもニューラルネットワークでも何でも良い．好きなモノを使う．

$$Z_\theta = \sum_{\vec{x}} \exp\left(-E_\theta(\vec{x})\right)$$

確率なので合計を 1 にするための規格化定数 (分配関数).
θ を色々動かしてデータに合う生成分布を作成する．

具体的には ...P という神様 (真) の分布と自身が作成したモデルの距離を近づけたい．

※ 距離はどの距離を採用してもよい　よく使用されるのは次ページで説明する KL 情報量

$$min\{D(P \| q_\theta)\} \approx min\{D(P_0 \| q_\theta)\}$$

できるだけ P に近くなるようデータから近似する．P は分からないのでデータ由来の $P_0(\vec{x})$ から近づける．

min は θ の関数であり，$P_0(\vec{x})$ へ近づけるのに動かせるのは θ のみ

距離の表現～ KL 情報量と KL は≧ 0 になる重要事項の証明

■KL 情報量 ～ Kullback-Leibler 情報量 (距離を比較する 1 つの方法)

$$D(P\|q)=\sum_{x} P(\vec{x})\log(\frac{P(\vec{x})}{q(\vec{x})})$$

期待値の差 ．どんな期待値かと言うと$q(\vec{x})$と $P(\vec{x})$ の比
(相対エントロピー).P と q が同じ時には 0 になるし 差異があれば正となるので距離に適任

$D(P\|q)\geq 0$ ※$=0$ は$P(\vec{x})=q(\vec{x})$の時 $D(P\|q)\neq D(q\|p)$ P と q を入れ替えると同じにならない . このため正確には距離とは違うが便利なので距離として扱う .

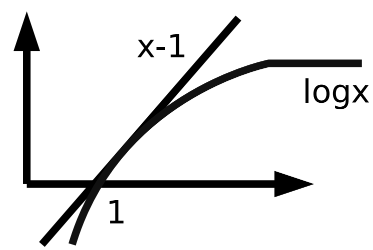
■Gibbs の不等式 $D(P\|q)\geq 0$ の証明

$$D(p\|q)=\sum_{\vec{x}} P(\vec{x})\log\underbrace{\frac{P(\vec{x})}{q(\vec{x})}}_{\frac{1}{x}}$$

右図の 1/x と見る

$$\geqq \sum_{\vec{x}}\left(1-\frac{q(\vec{x})}{p(\vec{x})}\right)=\underbrace{\sum_{\vec{x}} P(\vec{x})}_{=1}-\underbrace{\sum_{\vec{x}} q(\vec{x})}_{=1}=0$$

確率分布なので全部足すと 1． なので 1-1=0



$$\log x\leq x-1$$
$$[\log\frac{1}{x}\geq 1-x]$$

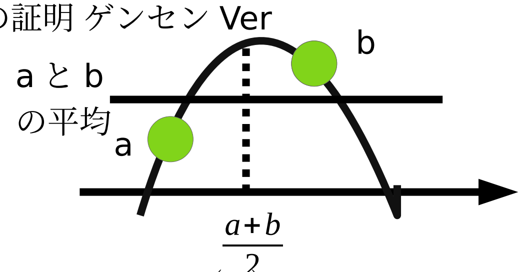
■Jensen の不等式 $D(P\|q)\geq 0$ の証明 ゲンセン Ver

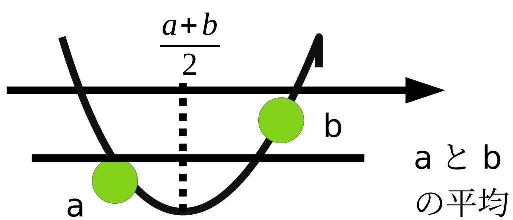
f(x) は上に凸　平均の f ≧ f の平均

$$f\left(\frac{a+b}{2}\right)\geq\frac{1}{2}[f(a)+f(b)]$$

f(x) は下に凸　平均の f ≦ f の平均

$$f\left(\frac{a+b}{2}\right)\leq\frac{1}{2}[f(a)+f(b)]$$





$$D(P\|q)=-\sum_{\vec{x}} P(\vec{x})\log\frac{q(\vec{x})}{p(\vec{x})}\geq -\log\sum_{\vec{x}} \cancel{P(\vec{x})}\frac{q(\vec{x})}{\cancel{p(\vec{x})}}=0$$

f の平均 . f は -log の事　※$\sum_{\vec{x}} q(\vec{x})=1\Rightarrow -\log 1=0$

log は上に凸だがマイナスして下に凸にする .
それに伴い $\frac{P(\vec{x})}{q(\vec{x})}\rightarrow\frac{q(\vec{x})}{p(\vec{x})}$

ベクトルの積分だが x_i の時しか有効にしないの意

ガウス分布を仮定してモデル作成

ガウス分布

$$q_\theta(x) \propto \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)$$

今回はガウスだが関数は好きな関数を使って良い．
選んだ関数でパラメータを使って$P_0(\theta)$ に合わせる．
ただし複雑だと使えないし簡単だと複雑なモノが作れないので良い塩梅を探す．

$$Z_\theta = \int dx \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) = \sqrt{2\pi\sigma^2}$$

分配関数 (規格化定数)

参考：ガウス積分 $\int_{-\infty}^{\infty} dx \exp\left(-\frac{1}{2}a x^2\right) = \sqrt{\frac{2\pi}{a}}$

$$q_\theta(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) \quad \theta = \{\mu, \sigma^2\}$$

ガウス分布の場合の θ はコレ．指定する分布によって θ が何かは変わる．この θ を動かして一番近い分布を見つけるのが目的．

KL 情報量

$-\frac{1}{2\sigma^2}(x-\mu)^2 - \frac{1}{2}\log 2\pi\sigma^2$ ※exp の log なので

$$D(P_0 \| q_\theta) = \int dx\, \underline{P_0(x)}\left(logP_o(x) - \underline{logq_\theta(x)}\right)$$

確率分布の対数をとった差の期待値．

経験分布　対数誤差の経験平均．経験平均：実際に取得したデータで平均を取る

θ のみ動かして最小化する

$$min\theta = \int dx\, P_0(x)\left(\frac{1}{2\sigma^2}(x-\mu)^2 + \frac{1}{2}log\sigma^2\right)$$

$logP_0(x)$や2πは定数なのでθに関係ない項目は無視．

$\frac{1}{n}\sum_{i=1}^{n}\delta(x-x_i)$ 積分するのに x とx_iが一致した分だけ (x-μ) の x に代入して全て足す

$$= \frac{1}{2\sigma^2}\underline{\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2} + \frac{1}{2}log\sigma^2$$

これを最小化する．(学習する)

最小二乗法

デルタ関数

$$\int d\vec{x}\,\delta(\vec{x}-\vec{x}_i) f(\vec{x}) = f(\vec{x}_i)$$

$$\int dx_1 \int dx_2 \int dx_{3......} \int dx_N$$

ベクトルの積分だが x_i の時しか有効にしないの意

学習 ~ 最適化 ~

$$\partial\mu : -\frac{1}{\sigma^2}\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2\right)=0 \therefore \mu=\frac{1}{n}\sum_{i=1}^{n}x_i$$

μ は Σ は関係ないので n 倍されて 1/n されるので μ
経験平均．実際に存在したデータで平均

$$\partial\sigma^2 : -\frac{1}{2(\sigma^2)^2}\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2\right)+\frac{1}{2\sigma^2}=0 \therefore \sigma^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2$$

σ ではなくて σ^2 での微分に注意．

上記で μ と σ^2 が分かるので生成シュミレータが作れる．

最尤法 (=KL 最小化)

$$\min_{\theta}\{D(P_0\|q_\theta)\}\Rightarrow \overbrace{\max_{\theta}}^{\text{“-” の min}}\{\sum_x P_0(\vec{x})\log q_\theta(\vec{x})\}$$ ※θに関係ない$P_0(\vec{x})\log P_0(\vec{x})$はいらない.

$P_0(\vec{x})$はデルタで経験分布なので x がx_i になった時だけ足し合わせる．

自分が作ったモデルに実際のデータを入れてたのが尤度関数．

モデルにデータを入れて確率が大きかったら良いモデル．小さければ悪いモデル．

$$=\max\{\frac{1}{n}\sum_{i=1}^{n}\log q_\theta(\vec{x}_i)\}$$

θ の対数尤度関数 (KL 最小化): モデルの良さの指標．
尤度と尤度が高いパラメータを提示できれば色々検討ができる．

KL 情報量がよく使わる理由 (性質の良さ)
指数分布族の関数を選択して使用する場合に対数を使うと exp が消せるから計算しやすい．
指数分布族で無い関数を使う場合は L1 ノルムとか KL 以外でも良い．
後は確率はかけ算するが対数がと足し算にできるので計算が楽．
なので相関があって独立ではなくかけ算できない場合は KL でなくても良い．
余談：かけ算と足し算のかきねをなくす．宇宙際タイミュラー理論などもある．
統一的に使うなら KL を使う必要はなし．

高次元に拡張 ~ またとりあえずガウス分布を仮定~

$$q_\theta(\vec{x}) \propto \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1}(\vec{x}-\vec{\mu})\right)$$

$$Z_\theta = \int d\vec{x} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1}(\vec{x}-\vec{\mu})\right)$$

$$\underbrace{(\vec{x}-\vec{\mu})^T P}_{\vec{y}^T} \underbrace{P^{-1}\Sigma^{-1}P}_{\Lambda} \underbrace{P^{-1}(\vec{x}-\vec{\mu})}_{\vec{y}}$$

補足：対角化 (対角化できるモノしか考えない)

$$P^{-1}\Sigma^{-1}P = \Lambda = \begin{pmatrix} \lambda_1 & .. & 0 \\ .. & .. & .. \\ 0 & .. & \lambda_n \end{pmatrix}$$

$$P^{-1}(\vec{x}-\vec{\mu}) = \vec{y}$$

x と平均の誤差

積分可能な形にできる．ヤコビアンで置換積分．

$y_1 \lambda_1 y_1$ のかけ算 , $y_2 \lambda_2 y_2$ のかけ算のように非対角要素がないので行列だけどスカラーのように計算ができる

参考：ガウス積分 $\int_{-\infty}^{\infty} dx \exp\left(-\frac{1}{2} a x^2\right) = \sqrt{\frac{2\pi}{a}}$

$$= \int d\vec{y} \exp\left(-\frac{1}{2}\vec{y}^T \Lambda \vec{y}\right) = \prod_{k=1}^{N} \int dy_k \exp\left(-\frac{1}{2}\lambda_k y_k^2\right)$$

全て積分 . 各成分で因数分解したのでそれで積分 .

$$= \prod_{K=1}^{N} \sqrt{\frac{2\pi}{\lambda_k}}$$

ちなみに $logdet(\Sigma) = logdet(\Lambda) = \sum_{k=1}^{N} \log \lambda_k$

$\prod_{k=1}^{N} \lambda_k = det(\Lambda) = det(\Sigma^{-1})$ の性質を使用する . ※対角行列なので $det(\Lambda)$. 対角化で det は変わらないので元の Σ^{-1} の $det(\Sigma^{-1})$ も同じ値.

また $det(\Sigma^{-1}) = \frac{1}{det(\Sigma)}$

$$\therefore q_\theta(\vec{x}) = \sqrt{\frac{1}{(2\pi)^N det(\Sigma)}} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1}(\vec{x}-\vec{\mu})\right)$$

これで My モデルができたので , μ と Σ^{-1} を動かしてデータに Fit させる事ができる .

学習

$$D(P_0\|q_\theta)=\int d\vec{x}\,P_0(\vec{x})(\log P_0(\vec{x})-\log q_\theta(\vec{x}))$$

2π は無視

$$=\frac{1}{2}\int d\vec{x}\,P_0(\vec{x})(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})+\frac{1}{2}\log\det(\Sigma)\quad=\frac{1}{2n}\sum_{i=1}^{n}(\vec{x}_i-\vec{\mu})^T\Sigma^{-1}(\vec{x}_i-\vec{\mu})+\frac{1}{2}\log\det(\Sigma)$$

$\frac{1}{n}\sum_{i=1}^{N}\delta(\vec{x}-\vec{x}_i)$ 1 次の時と同じ

誤差の 2 乗を分散共分散行列で重み付け

学習

$$\partial\vec{\mu}=\frac{1}{n}\sum_{i=1}^{n}\Sigma^{-1}(\vec{x}_i-\vec{\mu})=\vec{0}$$

※ 左から微分，右からも微分．Σ^{-1}は対称なので同じ係数であり 2 倍になる．

行列だけどただの係数と同様に扱う

$$\therefore\vec{\mu}=\frac{1}{n}\sum_{i=1}^{n}\vec{x}_i \text{ ※ } \mu\text{は}n\text{回足して}n\vec{\mu}\text{で}\frac{1}{n}\text{するので}$$

$$\partial\Sigma^{-1}=\frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i-\vec{\mu})(\vec{x}_i-\vec{\mu})^T$$

T の位置が変わっているのは計算するとこの形でないと合わないため．行列の計算箇所のみ抜き出し

$$\begin{pmatrix}\partial\Sigma^{-1}{}_{11} & \partial\Sigma^{-1}{}_{12} & \partial\Sigma^{-1}{}_{13} & ..\\ \partial\Sigma^{-1}{}_{21} & .. & .. & ..\\ .. & .. & .. & ..\end{pmatrix}=\begin{array}{c}\text{行}\qquad\text{列}\\ \begin{pmatrix}\frac{1}{2n}\sum_{i=1}^{n}(x_{i1}-\mu_1)(x_{i1}-\mu_1) & \frac{1}{2n}\sum_{i=1}^{n}(x_{i1}-\mu_1)(x_{i2}-\mu_2) & .. & ..\\ \frac{1}{2n}\sum_{i=1}^{n}(x_{i2}-\mu_2)(x_{i1}-\mu_1) & .. & .. & ..\\ .. & .. & .. & ..\end{pmatrix}\end{array}$$

Σ^{-1}の意味は各要素で微分すること．→ $(x_i-\mu$の1番目$, x_i-\mu$の1番目$),(x_i-\mu$の1番目$, x_i-\mu$の2番目$)$... など

$$\begin{pmatrix}a_1\\a_2\\.\\a_n\end{pmatrix}\begin{pmatrix}a_1 & a_2 & . & a_n\end{pmatrix}=\begin{pmatrix}a_1a_1 & a_1a_2 & a_1a_3 & ..\\ a_2a_1 & .. & .. & ..\\ .. & .. & .. & ..\end{pmatrix}$$

T の位置

det の微分

$$\frac{\partial}{\partial A}\log\det A=\frac{1}{A}(=A^{-1})$$

det の計算は固有値の積でできる

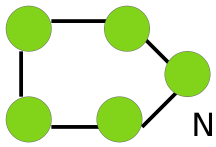
Σ^{-1}で微分するので log の−を出して分母にしておく

$$\partial\Sigma^{-1}:\frac{1}{2n}\sum_{i=1}^{n}(\vec{x}_i-\vec{\mu})(\vec{x}_i-\vec{\mu})^T-\frac{1}{2}\Sigma=0\ \therefore\ \Sigma=\frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i-\vec{\mu})(\vec{x}_i-\vec{\mu})^T$$

よくあるボルツマンマシン　※離散の場合

$\vec{x} \in \{-1,+1\}$　N 次元 (ノード数)

$q_\theta(\vec{x}) \propto \exp\left(\sum_{k \neq l} J_{kl} x_k x_l + \sum_{k=1}^{N} h_k x_k\right)$　Ising モデル



N=5.

画像の要素なら
256× タテ × ヨコ

n はデータ数

$Z_\theta = \sum_{\vec{x}} \exp(\ldots)$　離散になった場合，いっきに難しくなる．
スピングラスの Z_θ は計算が厳しいので近似計算が発展してきた．

$D(P_0 \| q_\theta) = \sum_{\vec{x}} P_0(\vec{x}) \left(\log P_0(\vec{x}) - \log q_\theta(\vec{x})\right)$

$$= \underline{-\sum_{\vec{x}} P_0(\vec{x}) \left(\sum_{k \neq l} J_{kl} x_k x_l + \sum_{k=1}^{N} h_k x_k\right)} + \underline{\log Z_\theta}$$

θ のみ

E: 内部エネルギー　$\frac{1}{n} \sum_{i=1}^{n} \delta(x - x_i)$

-βF: 自由エネルギー

つまり KL 情報量は「エネルギー」 − 「自由エネルギー」 ＝ エントロピー (マイナスエントロピー)

$$= -\frac{1}{n} \sum_{i=1}^{n} \left(\underline{\sum_{k \neq l} J_{kl} x_{ki} x_{li}} + \underline{\sum_{k=1}^{N}} h_k x_k\right) + logZ_\theta$$

関係

x のでやすさ .(画像の場合はその場所に白が出やすいか黒が出やすいか)
顔だったら目の近くはどうかなど .
そして隣のピクセルとのバランスを考えているのが $\underline{J_{kl}} x_{ki} x_{li}$

学習

$$\partial h_k = -\frac{1}{n}\sum_{i=1}^{n} x_{ki} + \frac{\partial}{\partial h_k} logZ_\theta = 0$$

$$\frac{\partial}{\partial h_k} logZ_\theta = \frac{1}{Z_\theta}\sum_{\vec{x}} \underbrace{x_k}_{\text{値}} \underbrace{\exp\left(\sum_{k\neq l} J_{kl} x_k x_l + \sum_{k=1}^{N} h_k x_k\right)}_{q_\theta(\vec{x})} \Rightarrow <x_k>_\theta$$ ※指数関数なのでまんま降りる

$h_k x_k$ h_k ごとに微分するのでx_kがそのまま降りる

(値　×　確率) それの和をとる　→　期待値

$$[\langle ...\rangle_\theta = \frac{1}{Z_\theta}\sum_{\vec{x}} \exp\left(\sum_{k\neq l} J_{kl} x_k x_l + \sum_{k=1}^{N} h_k x_k\right)\times .....]$$

なにかしらの物理量 (...) に exp の重みをつけて足し算する事を” <...>” で記載する．$\langle x_k\rangle_\theta = \frac{1}{n}\sum_{i=1}^{n} x_{ki}$ ⇔モデルの平均＝データの平均

データの平均と My モデルが一致するように θ を動かして合わせる .(デジタルツイーン)

計算は$q_\theta(\vec{x})$のシュミレータを作って$\langle x_k\rangle_\theta$を計算する.⇒マルコフ連鎖モンテカルロ法

$$\partial J_{kl} = -\frac{1}{n}\sum_{i=1}^{n} x_{ki} x_{li} + \frac{\partial}{\partial J_{kl}} logZ_\theta = 0$$

$x_k x_l$に関する期待値

$$\frac{\partial}{\partial J_{kl}} logZ_\theta = \frac{1}{Z_\theta}\sum_{\vec{x}} x_k x_l \underbrace{\exp\left(\sum_{k\neq l} J_{kl} x_k x_l + \sum_{k=1}^{N} h_k x_k\right)}_{q_\theta(\vec{x})} = \underbrace{\langle x_k x_l\rangle_\theta}_{\text{モデルの平均}} = \underbrace{\frac{1}{n}\sum_{i=1}^{n} x_{ki} x_{li}}_{\text{データの平均}}$$

θ によってきまるものがデータと一致する．
右辺は計算できる．x_k, x_lの異なる成分のクロスタームの平均．左辺は θ の関数．
できたらできる．

ただしこれで θ をきめるのは無理→勾配法でやる

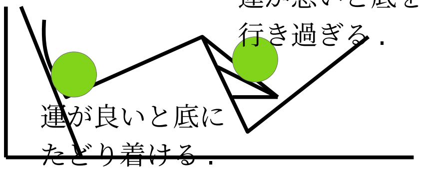
$$h_k = h_k - \eta_k\Big(\underbrace{\frac{1}{n}\sum_{i=1}^{n} x_{ki}}_{\text{微分}} - \underbrace{\langle x_k\rangle}_{\text{モデルの平均}}\Big)$$

$$J_{kl} = J_{kl} - \eta_{kl}\left(\frac{1}{n}\sum_{i=1}^{n} x_{ki} x_{li} - \langle x_k x_l\rangle\right)$$

η: ある程度の幅 (学習率) 小さい方が良い

J_{kl}はピクセルとの関係．同じ色になりやすいか．

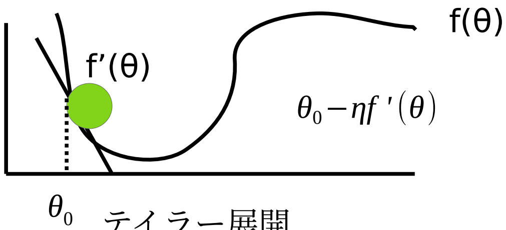
h_k は白になるかどうかの確率的な



η: が大きいと運が良ければ一気に底にいけるけど運が悪いと下がらず収束しない．

勾配法

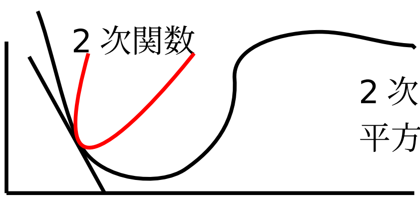
min{f(θ)} θ を徐々に動かして min を探す



傾きは負 (右下下がり) なのでプラスになる方向に ηf’(θ) 進む .
η は行きすぎないようにする係数 .

テイラー展開

$$f(\theta)=f(\theta_0)+f'(\theta)(\theta-\theta_0)+\frac{1}{2}f''(\theta_0)(\theta-\theta_0)^2+\ldots.$$



2 次関数ならできる .
平方完成

1 次関数だとちょっと行ったところしか分からないので , 2 次関数で接線の代わりに近似する .f(θ) より常に大きい 2 次関数を考える (赤線)
赤線は計算しやすく赤線の最小値を f(θ) が超えることはない .(上から蓋をする)

調整

$$f(\theta)\le f(\theta_0)+f'(\theta_0)(\theta-\theta_0)+\frac{1}{2}L(\theta-\theta_0)^2$$

← 代理関数 :L を大きくすれば f(θ) より大きくなる .
※3 次以降のテイラー展開は手間なので 2 次までで考える .

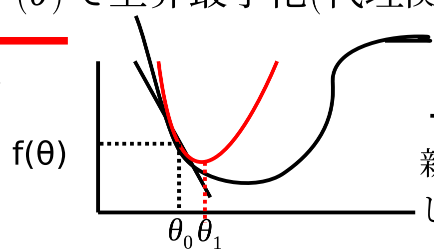
$$=f(\theta_0)+\frac{1}{2}L\left(\theta-\theta_0+\frac{1}{L}f'(\theta_0)\right)^2-\frac{(f'(\theta_0))^2}{2L}$$

θ - θ_0で平方完成 . 第 3 項は必ず負なので f(θ) から確実に下がる .
蓋が降りる .f(θ) は前より下がるので下がったところで新しい蓋を用意する .
逐次最小化を目指す . η=1/L(学習率)

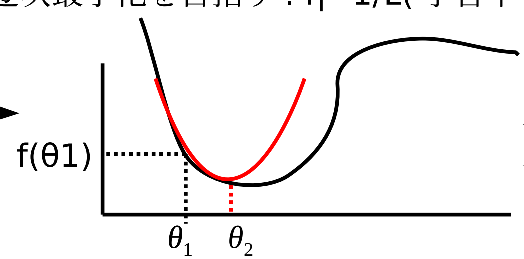
$$\theta=\theta_0-\frac{1}{L}f'(\theta)\text{で上界最小化(代理関数の)}$$

第 2 項をゼロ

勾配法



新しく蓋をしていく



η 小はゆっくり
η 大は上界が破綻する

繰り返せば , いずれ極小値にたどり着く .

代理関数

$f(\vec{\theta})$に対して　　ベクトルなので微分が各成分ごとにおこなわれているのでナムラ表記 (代理関数)

$$f_\eta(\vec{\theta},\vec{\theta}_0)=f(\vec{\theta}_0)+\vec{\nabla} f(\vec{\theta}_0)(\vec{\theta}-\vec{\theta}_0)+\frac{1}{2}(\vec{\theta}-\vec{\theta}_0)^T\frac{1}{\eta}(\vec{\theta}-\vec{\theta}_0)$$

L=1/η

$$\begin{pmatrix}\frac{\partial f}{\partial \theta_1}\\ \frac{\partial f}{\partial \theta_2}\\ \frac{\partial f}{\partial \theta_3}\\ ..\\ ..\end{pmatrix}\left\{\begin{pmatrix}\theta_1\\ \theta_2\\ \theta_3\\ ..\\ ..\end{pmatrix}-\begin{pmatrix}\theta_{10}\\ \theta_{20}\\ \theta_{30}\\ ..\\ ..\end{pmatrix}\right\}$$

内積なのでかける

$$d(\vec{\theta})=f(\vec{\theta})-f_\eta(\vec{\theta}-\vec{\theta}_0)\leq 0$$

f_η は代理関数でそれを引いているので負になるはず

目的は常に≦ 0 になる η を求める

$d(\vec{\theta})$の性質 .θ_0など知っている値から代入.

1.$d(\vec{\theta}_0)=0$

2.$\vec{\nabla} d(\vec{\theta})=\vec{\nabla} f(\vec{\theta})-\vec{\nabla} f(\vec{\theta}_0)-\frac{1}{\eta}(\vec{\theta}-\vec{\theta}_0)$　　2 次関数を微分したので 1 次関数

$\vec{\nabla} d(\vec{\theta}_0)=\vec{0}$

$\vec{\nabla}\vec{\nabla} d(\vec{\theta}_0)=\underbrace{\vec{\nabla}\vec{\nabla} f(\vec{\theta})}_{H}-\frac{1}{\eta}[H_{kl}=\frac{\partial^2 f(\theta)}{\partial\theta_k\partial\theta_l}]$

H: ヘシアン (ヘッセ行例)

これがどんな方向でも負になれば OK

$\vec{a}^T(H-\frac{1}{\eta})\vec{a}$

$$P^{-1}HP=\Lambda=\begin{pmatrix}\lambda_1 & .. & 0\\ .. & .. & ..\\ 0 & .. & \lambda_n\end{pmatrix}$$

2 次関数由来 . これが常に負になるようにする .

方向を a で表している .(θ - θ_0) に相当 .

d(θ) は f(θ)- f_η (θ, θ_0) でそれを 2 回微分したものに右から左からかける .

これが a がどんな方向に変化しても 2 次関数由来のところが負になる事を示せば良い .

$P^{-1}\vec{a}=\vec{b}$※あらゆる値$\vec{a},\vec{b}$とおく

$=\vec{b}^T(\Lambda-\frac{1}{\eta})\vec{b}=\sum_k(\lambda_k-\frac{1}{\eta})b_k^2\leq 0\ for\ all$

$\vec{a}=p\vec{b}\Rightarrow\vec{a}^T=\vec{b}^TP^T\quad H=P\Lambda P^{-1}$

$\vec{a}^T(H-\frac{1}{\eta})\vec{a}=\vec{b}^TP^TP(\Lambda-\frac{1}{\eta})P^{-1}P\vec{b}$

b_k^2 は常に正なのでココが常に負 .

$\eta\leq\frac{1}{max\,\lambda_k}$　1/λmax より小さければ OK. 実用的にはとりあえず 0.1 が多い .

k　最大固有値 . 一番大きい λ が負なら常に負になる .

f(θ) を 2 次関数化する .

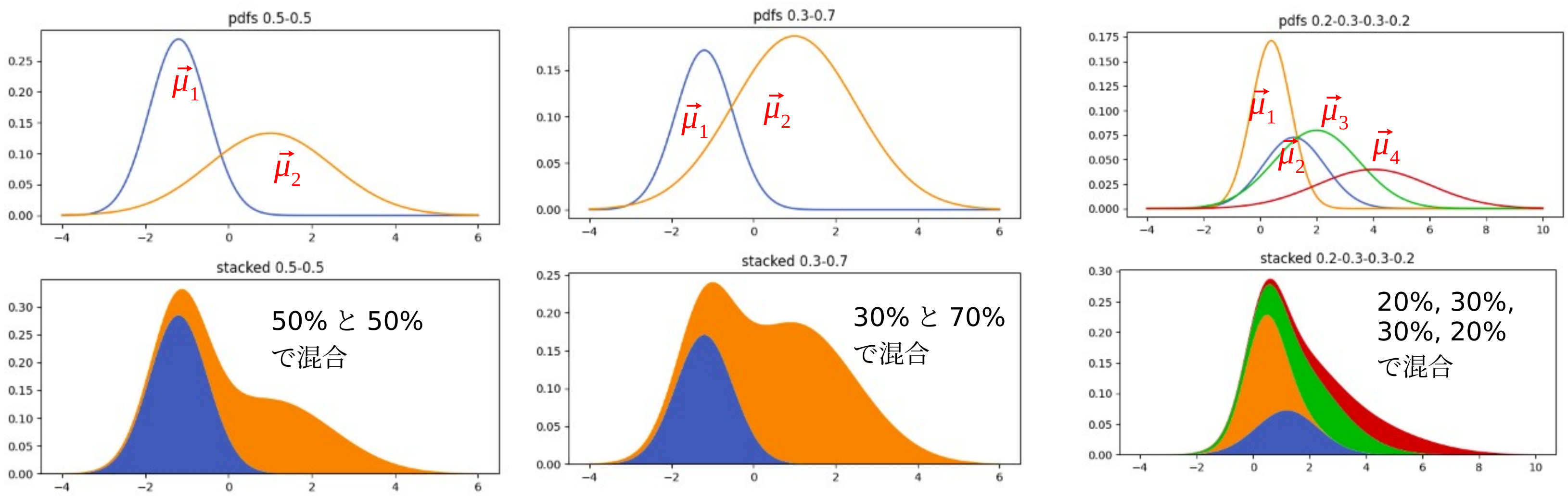
よりリッチなモデルに

和をとる

$q_\theta(\vec{x})=\sum_k C_k \exp(-\frac{1}{2}(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu}))/\sqrt{2\pi det(\Sigma)}$ 混合ガウス分布として k 個増やす .

ただし $\sum k C_k=1$ C_k は重み . 80% のガウス分布と 20% のガウス分布の和など

単峰ではない分布を作る



得られたデータ

$$\vec{x}_1,\vec{x_2}_{....}\vec{x}_n \Rightarrow P_0(\vec{x})=\frac{1}{n}\sum_{i=1}^{n}\delta(\vec{x}-\vec{x}_i)$$

データ

経験分布

最尤法

$$\max_{\theta}\left\{\frac{1}{n}\sum_{i=1}^{n}\log q_{\theta}(\vec{x}_i)\right\} = \max_{\theta}\left\{\frac{1}{n}\sum_{i=1}^{n}\log\sum_{k}C_k\exp\left(-\frac{1}{2}(\vec{x}_i-\vec{\mu}_k)\Sigma^{-1}(\vec{x}_i-\vec{\mu}_k)\right)/\sqrt{2\pi det(\Sigma_k)}\right\}+\lambda\left(\sum_k C_k-1\right)$$

$\left(\sum k\, C_k=1\right)$ ラグランジュの未定乗数法

$= \dfrac{1}{det(\Sigma_k^{-1})}$ det は逆行列にすると逆数になる事を使用して $\sum^{-1}$ を微分する

$$\partial\mu_k : \frac{1}{n}\sum_{i=1}^{n}\boxed{\frac{1}{\sum_k C_k\exp(\ldots)}}\times\boxed{C_k\exp(\ldots)}\,\Sigma_k^{-1}(\vec{x}_i-\vec{\mu}_k)$$

重み

log の微分　分母の微分

γ_{ik} : 負担率（ビショップ本で言う）

$$=\frac{1}{n}\sum_{i=1}^{n}\gamma_{ik}\Sigma_k^{-1}(\vec{x}_i-\vec{\mu}_k)=0 \therefore \vec{\mu}_k=\frac{\sum_{i=1}^{n}\gamma_{ik}\vec{x}_i}{\sum_{i=1}^{n}\gamma_{ik}}$$

データの重み付け平均

※ Σ_k^{-1} は両辺に逆行列をかければ消える．

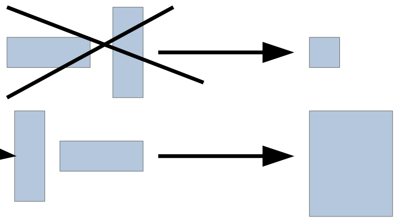
γ は k ごとに違う．あるデータにはコッチのガウス分布の重みが強いなど

上と同じ計算なので γ がでてくる

$$\partial\Sigma_k^{-1} : -\frac{1}{2n}\sum_{i=1}^{n}\gamma_{ik}(\vec{x}_i-\vec{\mu}_k)(\vec{x}_i-\vec{\mu}_k)^T+\frac{1}{2}\frac{1}{n}\sum_{i=1}^{n}\gamma_{ik}\Sigma_k=0$$

$(\Sigma_k^{-1})^{-1}$ なので

$$\therefore \Sigma_k=\frac{1}{\sum_{i=1}^{n}\gamma_{ik}}\sum_{i=1}^{n}\gamma_{ik}(\vec{x}_i-\vec{\mu}_k)(\vec{x}_i-\vec{\mu}_k)^T \ ■$$



スカラ．今回は結果を行列にしたいので転置をして下にする

行列で微分は各成分での微分なので結果も行列でないとダメ．なので転置．

$$\partial C_k : \frac{1}{n}\sum_{i=1}^{n}\frac{\exp(\ldots)}{\sum_k C_k\exp(\ldots)}+\lambda=0$$

$$\lambda=-\frac{1}{n}\sum_{i=1}^{n}\sum_{k}\gamma_{ik}$$

条件より C_k の和は 1 なので先に k について和をとり λ を求めてそれを代入する．

$$\times C_k \quad \frac{1}{n}\sum_{i=1}^{n}\gamma_{ik}+\lambda C_k=0 \therefore C_k=\frac{\sum_{i=1}^{n}\gamma_{ik}}{\sum_k\sum_{i=1}^{n}\gamma_{ik}} \ ■$$

分母に $\sum_k$ があるので全部足したら 1 になり規格化をみたしている．

x_i を入れる → フィットする γ_{ik} をみる → μ_k, Σ_k^{-1} を更新する → 繰り返し....(EMアルゴリズム)次ページ

（計算する）　γ_{ik} は尤度

EM アルゴリズム (Expection-Maximization アルゴリズム)

x(画像) など生成するのに複雑な関数が必要 . その関数は色々だすので乱数 (確率的) な振る舞いをするのでは？ つまり , x の裏には確率的な発生 (隠れ変数) がありこれを計算するのが EM アルゴリズム .
→ 普段見る絵や音楽を作るには複雑な関数が必要 .
　ただし複雑すぎても扱えないので知っている良い関数を選ぶ .

for 混合ガウス分布

ガウスの形

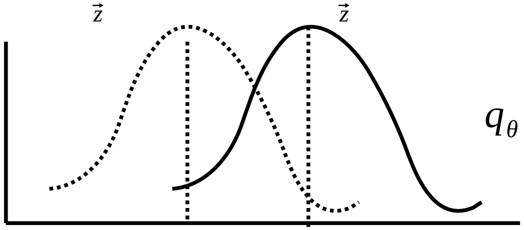
1. $\vec{\mu}_k, \Sigma_k^{-1}, C_k$ を計算.$(\gamma_{ik}$ ：fixed$)\gamma_{ik}$ は与えられるとして固定.

2. γ_{ik} を計算.$(\mu_k, \Sigma_{ik}^{-1}$ が変わるので更新$)$ データとの fit

2 つの動き

■ 隠れ変数

$$q_\theta(\vec{x}) = \sum_{\vec{z}} q_\theta(\vec{x}, \vec{z}) \left(= \sum_{\vec{z}} q_\theta(\vec{x}|\vec{z}) q_\theta(\vec{z}) \right)$$



Z_1 Z_2

$q_\theta(\vec{x}|\vec{Z}=z)$ に従う

$q_\theta(\vec{Z})$ に従う . xの前にまずZが選ばれる.

$q(\vec{x}, \vec{Z})$ 結合確率

$q(\vec{x}|\vec{Z}) = \dfrac{q(\vec{x}, \vec{Z})}{q(\vec{Z})}$ 条件付き確率

Z の不確定性を消す .
Z は確定しているので割り算 .
(各 Z に対しての)

例：混合ガウス分布

$q_\theta(Z)$ Zの比率 (どのガウス分布になりそうか)

$q_\theta(\vec{x}|Z)$ Zが与えられているので その中でのxの確率．

$$q_\theta(\vec{x},Z)=C_Z\exp(-\frac{1}{2}(\vec{x}-\vec{\mu}_Z)\Sigma_Z^{-1}(\vec{x}-\vec{\mu}_Z))/\sqrt{2\pi det(\Sigma_k)}$$

※xはガウス分布をいくつか足した分布だと仮定

CzでどれのガウスCz分布にするかZを決めて，Zを固定してxを生成する．

■ 勾配法：上界最小化(下界最大化)

$$log q_\theta(\vec{x})=\log\sum_{\vec{Z}} q_\theta(\vec{x},\vec{Z})\geq \text{下界}$$

まだデータxiはいれていない

$$\log q_\theta(\vec{x})=\log\frac{q_\theta(\vec{x},\vec{Z})}{\underline{q_\theta(\vec{Z}|\vec{x})}}$$

$\vec{x}$はgiven
$\vec{Z}$は不明

$\sum_{\vec{Z}} r(\vec{Z})$ 好きな分布で乱数を出力して和をとる.(ガウス...etc)

$$log q_\theta(\vec{x})=\underline{\sum_{\vec{Z}} r(\vec{Z})}\log\frac{q_\theta(\vec{x},\vec{Z})}{q_\theta(\vec{Z}|\vec{x})}$$

xとZがあるので右辺は計算できる．適当にZを作って平均．

xは与えられているので代入すれば計算可能

期待値計算

$$=\sum_{\vec{Z}} r(\vec{Z})\log\frac{q_\theta(\vec{x},\vec{Z})}{r(\vec{Z})}-\sum_{\vec{Z}} r(\vec{Z})\log\frac{q_\theta(\vec{Z}|\vec{x})}{r(\vec{Z})}$$

$$=\sum_{\vec{Z}} r(\vec{Z})\log\frac{q_\theta(\vec{x},\vec{Z})}{r(\vec{Z})}+\underline{D_{KL}(r(\vec{Z})\|q_\theta(\vec{Z}|\vec{x}))}\geqq 0$$

これは小さい方が真の関数と代理関数の差が小さいので良い．

- KLを作る

$$D_{KL}(p\|q)=\sum_{\vec{x}} p(\vec{x})\log\frac{P(\vec{x})}{q(\vec{x})}\geq 0$$

$$\underline{=\sum_{\vec{Z}} r(\vec{Z})\log\frac{q_\theta(\vec{x},\vec{Z})}{r(\vec{Z})}}$$

元々何がしたいかというとθに関してこの式を最大化したい．

変分下界→最大化する→代理関数！

EM アルゴリズムとの関係

前ページより

ここ (θ) だけが最大化の際の変数

$$\max_{\theta}\left\{\frac{1}{n}\sum_{i=1}^{n}\log q_\theta(\vec{x}_i)\right\}\geq\frac{1}{n}\sum_{i=1}^{n}\sum_{\vec{Z}} r(\vec{Z})\log\frac{q_\theta(\vec{x}_i,\vec{Z})}{r(\vec{Z})}$$

$r(\vec{Z})$は$q_\theta(\vec{Z}|\vec{x})$と距離が近い方が良い.

本当は$q_\theta(\vec{x}_i,\vec{Z})$と$q_\theta(\vec{Z}|\vec{x})$の$\theta$を同時に最大化したいが厳しいので2段階にする.

まず$q_\theta(\vec{Z}|\vec{x})$の$\theta$を最適化.

$q_\theta(\vec{Z}|\vec{x})$ [r をまず最適化] θと$r(\vec{Z})$の交互最適化をする.

$$Q(\theta,\theta_0)=\frac{1}{n}\sum_{i=1}^{n}\sum_{\vec{Z}} q_{\theta_0}(\vec{Z}|\vec{x}_i)\log q_\theta(\vec{x}_i,\vec{Z})$$

(Q 関数) この最適化で登場するのが代理関数から決まった Q 関数

この θ を動かして θ_0 に入れて交互にやっていく .

←Z で和を取る際に r(Z) は最適化された結果をいれる .

※ なので Q 関数には r の記載はない .r は最適化して固定されているので .

$$q_{\theta_0}(\vec{Z}|\vec{x}_i)=\frac{q_{\theta_0}(\vec{x}_i,\vec{Z})}{q_{\theta_0}(\vec{x}_i)}=\frac{q_{\theta_0}(\vec{x}_i,\vec{Z})}{\sum_{Z} q_{\theta_0}(\vec{x}_i,\vec{Z})}=\frac{C_z\exp(\dots)}{\sum_{Z} C_z\exp(\dots)}=\gamma_{iZ}$$

← Q 関数に代入する

Z で和をとることで不確実性は x だけになる .

最尤法の結果と同じ . 最尤法の対数尤度は難しいので相手にしない .
代わりに KL 情報量を使った不等式で解く .
代理関数は和がなくて log に直接かかるので簡単 .
代理関数を求める努力をした方が良い .

混合ガウス分布のq_θの箇所 (ガウス分布の式)

$$=\frac{1}{n}\sum_{i=1}^{n}\sum_{Z}\gamma_{iZ}\left[\left(-\frac{1}{2}(\vec{x}_i-\vec{\mu}_Z)^T\Sigma_Z^{-1}(\vec{x}_i-\vec{\mu}_Z)+\frac{1}{2}logdet\,\Sigma_Z^{-1}\right)+\log C_z\right]$$

ラグランジュの未定乗数

制約条件：$\sum_{Z} C_Z=1$の条件 $\lambda\left(\sum_{Z} C_Z-1\right)$を入れる.

$$\partial C_Z:\ \frac{1}{n}\sum_{i=1}^{n}\gamma_{iZ}\frac{1}{C_Z}+\lambda=0$$

$$\frac{1}{n}\sum_{i=1}^{n}\gamma_{iZ}+\lambda C_Z=0\Rightarrow\lambda=-\frac{1}{n}\sum_{i=1}^{n}\sum_{Z}\gamma_{iZ}\qquad C_z=\frac{\sum_{i=1}^{n}\gamma_{iZ}}{\sum_{Z}\sum_{i=1}^{n}\gamma_{iZ}}\ ■$$

$\sum Z$(和をとる)→ C_zに和をとると1になるという条件を入れているので.
最尤法の時と同じ

$$\partial\vec{\mu}_z:\ \frac{1}{n}\sum_{i=1}^{n}\gamma_{iZ}\left(+\Sigma_Z^{-1}(\vec{x}_i-\vec{\mu}_Z)\right)=\vec{0}\qquad\therefore\vec{\mu}_Z=\frac{\sum_{i=1}^{n}\gamma_{iZ}\vec{x}_i}{\sum_{i=1}^{n}\gamma_{iZ}}\ ■$$

Σ_Z^{-1}は左から逆行列をかければ消せる.

普通のガウス分布の時と同様 T の位置が右上に変わる

$$\partial\Sigma_Z^{-1}:\ \frac{1}{n}\sum_{i=1}^{n}\gamma_{iZ}\left(-\frac{1}{2}(\vec{x}_0-\vec{\mu}_z)(\vec{x}_i-\vec{\mu}_z)^T+\frac{1}{2}\Sigma_Z\right)=0$$

$logdet\,\Sigma_z^{-1}$は$\frac{1}{\Sigma_Z^{-1}}$なのでインバースのインバースでΣ_Z

$$\Sigma_Z=\frac{1}{\sum_{i=1}^{n}\gamma_{iZ}}\sum_{i=1}^{n}\gamma_{iZ}(\vec{x}_0-\vec{\mu}_z)(\vec{x}_i-\vec{\mu}_z)^T\ ■$$

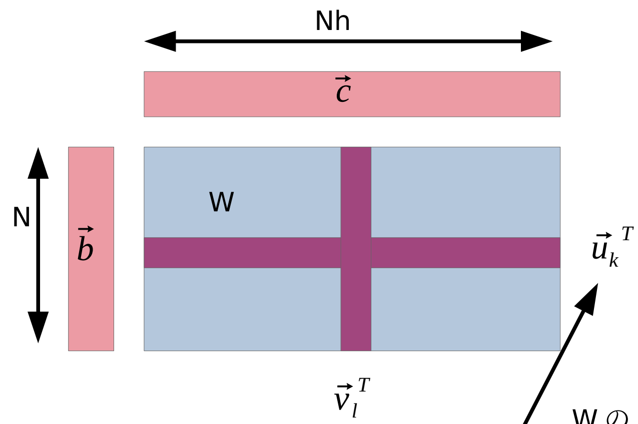
EM アルゴリズムは対数尤度を直接考えるのではなくて、変分下界を考える .
その変分下界の最大化を行っているアルゴリズム .

制限ボルツマンマシン

$$q_\theta(\vec{x},\vec{z})=\frac{1}{Z}\exp(\underbrace{\vec{x}^T W \vec{z}}_{\text{相互作用}}+\underbrace{\vec{b}^T\vec{x}+\vec{c}^T\vec{z}}_{\text{磁場}})$$

exp(...) で +1, -1 で和 .
exp のプラスと exp のマイナスの和→ cosh

$$\vec{x}\in\{-1,1\}^N \qquad \vec{z}\in\{-1,1\}^{Nh}$$



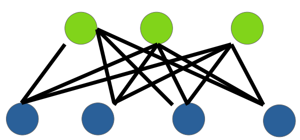
W の k 行目のベクトルが z に内積されたものが x_k の係数になっている . これがプラスとマイナスであるので足し算すると 2cosh ～ .
これが k=1 ～ n まで同じ式の形が続く .

W の k 行目のベクトルが内積にかかっている .

〇 得られているデータ

$\vec{x}_1,\vec{x}_2,\ldots,\vec{x}_n$ ：データ → $P_0(\vec{x})=\frac{1}{n}\sum_{i=1}^{n}\delta(\vec{x}-\vec{x}_i)$ 経験分布

〇 特徴



制限： $\cancel{\vec{x}^T A\vec{x}}, \cancel{\vec{z}^T B\vec{z}}$

何か

$q_\theta(\vec{z})$ は $q_\theta(\vec{x},\vec{z})$ にて $\vec{x}$ について和をとれば $q_\theta(\vec{z})$

$$q_\theta(\vec{x}|\vec{z})=\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z})} \quad (\text{ by } \quad q_\theta(\vec{z})=\sum_{\vec{x}} q_\theta(\vec{x},\vec{z})=\frac{1}{2}\prod_{k=1}^{N} 2\cosh(\underline{\vec{u}_k}^{T}\vec{z}+b_k)e^{\vec{c}^T\vec{z}}$$

$\vec{z}$ に何かを内積させる .

$$=\underbrace{\prod_{k=1}^{N}}_{\text{独立}}\frac{\exp(x_k\vec{u}_k^{\ T}\vec{z}+b_k x_k)}{2\cosh(\vec{u}_k^{\ T}\vec{z}+b_k)} \quad ■1$$

後々計算で使用するので条件付き確率を求めておく .

条件付き独立は凄い !
元々は全部の変数と関係付いてしまうので z はもちろん x_1 番目 , x_2 番目も気にしないといけなかった .
今まではモンテカルロ法にしろ x 全体を考えてサンプリングが必要であったが
$u_k z$, b_k で重みが決まり後は独立なので独立で処理ができる .

逆も同様にできる v_l はタテ

$$q_\theta(\vec{z}|\vec{x})=\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{x})} \quad (\text{ by } \quad q_\theta(\vec{x})=\sum_{\vec{z}} q_\theta(\vec{x},\vec{z})=\frac{1}{2}\prod_{l=1}^{Nh} 2\cosh(\vec{v}_l^{\ T}\vec{z}x+c_l)e^{\vec{b}^T\vec{x}} \quad =\prod_{l=1}^{Nh}\frac{\exp(z_l\vec{v}_l^{\ T}\vec{x}+c_l z_l)}{2\cosh(\vec{v}_l^{\ T}\vec{x}+c_l)} \quad ■2$$

制限ボルツマンマシン続き

これはサンプリングがしやすい．x が決まると z がでる．z が決まったら x がでる．
x が決まって z が決まったらスタックして z が決まったら z'. z' が決まったら z'' と奥に深い大量のパラメータを含んだ生成モデルが作れる．見えているのは x だけどその奥に z_1， z_2, z_3 .... と沢山の隠れ変数をもったモンスター生成モデルをつくる事が可能．
(ディープランニングのきっかけ .)
隠れ変数なので EM アルゴリズムを使って学習する．ただそれでもまだ難しいところがあるのでそこをいかに簡単にしていくかがキモでスライドの後半を参照．

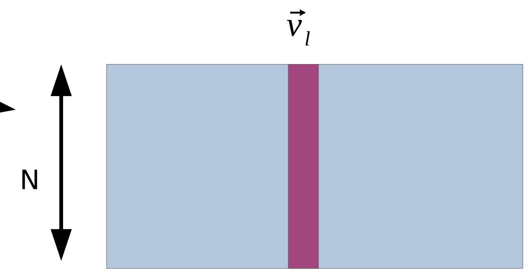
cosh の箇所の補足

指数関数の肩の和なので積にすることが可能．

$$\therefore \sum_{\vec{z}} q_\theta(\vec{x},\vec{z}) = \sum_{\vec{z}} \frac{1}{Z}\exp(\overbrace{\vec{x}^T W \vec{z}}^{\sum_l \vec{x}^T \vec{v}_l Z_l} + \vec{b}^T\vec{x} + \overbrace{\vec{c}^T\vec{z}}^{\sum_l c_l z_l})$$

$$e^{\sum_{l=1}^{Nh} a_l} = \prod_{l=1}^{Nh} e^{a_l}$$

$$= \sum_{\vec{z}} \frac{1}{Z}\exp(\vec{b}^T\vec{x}) \times \prod_{l=1}^{Nh} \exp(\vec{x}^T \vec{v}_l z_l + x_l z_l)$$



ベクトルと行列の積はベクトルになる

N 個の成分があり x との内積になる．
v_l は N 行ありそれが x との内積になる．
それが l 個 (列) ある．
(隠れ変数分の個数)
それぞれの値に z がかかっている．

$\vec{z}$ の全ての組み合わせで和→「$Z_l = -1, +1$」

$$\sum_{z_1=\pm1}\sum_{z_2=\pm1}\sum_{z_3=\pm1} \cdots \sum_{z_m=\pm1} \cdots$$

例：$\displaystyle\sum_{\substack{z_1=\pm1\\ z_2=\pm1}} e^{z_1}e^{z_2} = e^{+1}e^{+1} + e^{+1}e^{-1} + e^{-1}e^{+1} + e^{-1}e^{-1}$

$$\left(\sum_{z_1=\pm1} e^{Z_1}\right)\left(\sum_{z_2=\pm1} e^{Z_2}\right) = (e^{+1}+e^{-1})(e^{+1}+e^{-1})$$

Z については足し上げているので

$$= \frac{1}{Z}\exp(\vec{b}^T\vec{x}) \prod_{l=1}^{Nh} \sum_{z_l=\pm1} \exp(\vec{x}^T \vec{v}_l z_l + c_l z_l) \quad = \frac{1}{Z}\exp(\vec{b}^T\vec{x}) \prod_{l=1}^{Nh} 2\cosh(\vec{x}^T\vec{v}_l + c_l) \quad \blacksquare$$

独立と確率変数の作り方

独立だとなにが良いのか？
■1, ■2 は確率分布であるが，その分布に従った乱数を作りたくなる．
(EM アルゴリズムにてガウス分布に従った乱数を作るなどの状況)
その際に，複数の確率変数がある場合にそれらが関係していると，例えば 1 番の発生確率に応じて 2 番を修正したり ,1 番と 2 番の頻度によって 3 番が影響されたりなど複雑な処理が必要になる．※ 複雑な IF 分でロジックを実装する必要がある
しかし独立の場合はこれらを独立に考えられるので処理の実装が簡単．

例：

$$P(x)=\frac{e^{ax}}{2\cosh(a)}$$

で出力するとは？

$$P(+1)=\frac{e^{a}}{2\cosh(a)}=P$$

とおく．

$$P(-1)=\frac{e^{-a}}{2\cosh(a)}=1-P$$

乱数 r(0≦r<1) を作り
r<P → x = +1 にする．
otherwise → x = -1 にする．

「 P(x) に従う確率変数の作り方」
独立なら P だけ変えて一様乱数を Nh 個用意して，
乱数が P より大きいか小さいか並列的に処理して
TRUE なら 1, FALSE なら -1 にすればできる．

これらをやるのは複雑にはしたいけど生成プロトコルにおいて計算は簡単にしたい．このバランスを考えモデルを作れるのは数人でそれが価値となる．プログラミングやライブラリを使ってなんかの処理は誰かに任せれば良くて，そのコアを作るのが価値．
このスライドまでで，データを与えるまではできたので，以降はそのデータに合わせたパラメータを考えていく．
(最適化学習をしていく)

KL 最小化 (最尤法)

最尤法は計算が難しいので KL 情報量で計算．

$$D(P_0 \| q_\theta) = \sum_{\vec{x}} P_0(\vec{x})(\log P_0(\vec{x}) - \log q_\theta(\vec{x}))$$

$$= P_0(\vec{x}) = \frac{1}{n}\sum_{i=1}^{n} \delta(\vec{x} - \vec{x}_i)$$

$$= -\sum_{\vec{x}} P_0(\vec{x}) log q_\theta(\vec{x}) ※ P_0(\vec{x}), log P_0(\vec{x}) \text{は} \theta \text{と関係無いので無視}$$

θ のみ

パラメータを動かして最大化する事を学習．θ に関係ない箇所は放っておく．

$$= -\frac{1}{n}\sum_{i=1}^{n} \log \underline{q_\theta}(\vec{x}_i) \quad \leftarrow \text{最尤法} \quad ※ \frac{1}{n}\sum_{i=1}^{n} \log q_\theta(\vec{x}_i) \text{を尤度関数. この最大化と} KL \text{の最小化は同じこと.}$$

最適化

マイナスになるので KL は最小化を求める

この後の流れ～

x,$q_\theta(x)$ について考える．今考えているモデルは制限ボルツマンマシンの q_θ ．

x だけでなく，隠れ変数 z でモデルの複雑性を仕込んで複雑なモデルを作ろうとしている．

q(x) は z で和をとって足をつぶして $q_\theta(x)$ だけにしたもの．なので q(x) を求めるには和を取らなくてはいけない．

つまり $log q_\theta(x)$ というのは , log+ 和をとった q_θ でこれは計算できない．

なので最尤法での計算の代わりに変分下界を使用してそれを最大化する．

最尤法を直接計算する事も可能だが微分して 0 の計算は大変で計算も間違う．

変分下界はそれと同じ計算結果を出すので , EM アルゴリズムで計算した方が楽だし ,

変分下界は KL 情報量が正である (KL≧0) の不等式だけを作れば良いので簡単．

EM アルゴリズム (変分下界最大化)

$$\log q_\theta(\vec{x}) = \log\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z}|\vec{x})} \quad (\text{by } q_\theta(\vec{z}|\vec{x}) = \frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{x})}$$

EM アルゴリズムはしょぼいイメージがあるが，普遍的な凄いヤツ

$\cancel{\sum_x q_\theta(\vec{x},\vec{z})\times \ldots}$ z なんて分からない．分かるのは x(実データ) だけ，

なので x の条件付き確率

$\sum \vec{z}\, r(\vec{z})\times\ldots$ 確率分布の log の周りに新しく適当な確率分布用意．

z は分からないので r は適当にしてについて和をとる．

$$\log q_\theta(\vec{x}) = \sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z}|\vec{x})}$$

確率分布の log の周りに新しく適当な確率分布用意．

z は分からないので r は適当にしてについて和をとる．

左辺は z について何もしないので z は空回り．

$$D_{KL}(P\|q) = \sum_x P(\vec{x})\log\left(\frac{P(\vec{x})}{q(\vec{x})}\right) \geq 0$$

KL 情報量を使用．これは正なので．

$$\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z}|\vec{x})} = \sum_{\vec{z}} r(\vec{z})\log q_\theta(\vec{x}_i,\vec{z}) - \sum_{\vec{z}} r(\vec{z})\log q_\theta(\vec{z}|\vec{x})$$

KLの形にするため$\frac{1}{r(\vec{z})}$を$\log$に入れる

両方にいれているので展開すると$\log$のプラスとマイナスで$\frac{1}{r(\vec{z})}$は消える．

$$= \sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})} - \sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{z}|\vec{x})}{r(\vec{z})}$$

KL が使えるかたちに変形

分母分子逆なのでマイナス × マイナスでプラスになる．

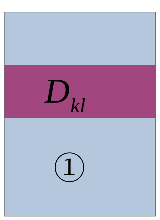
$$= \underline{\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}} + D_{KL}(r(\vec{z})\|q(\vec{z}|\vec{x}))$$

① 動かす θ はココだけ

$$\geq \underline{\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}}$$

$r(\vec{z}) = q_\theta(\vec{z}|\vec{x})$ でより下界◆

$\log q_\theta$ なので①の土台が上がれば $\log q_\theta$ も上がる．



変分下界→最大化！
θ の最大化をするためにこれを最大化する．
(代理関数)

勾配法も人気だがやっているのは変分下界

KL は元の関数とのギャップ．ギャップを小さくするのに KL は対数で最小二乗はヘシアンで実施．距離は KL でも最小二乗でも何でもいいが，ギャップの測り方が変わるだけでやっている事は同じ．元の関数とのギャップをどう埋めるか，その時の測り方によって変分下界の形は変わるが，変分下界を最大化してやれば元の関数を最大化する事を同じになる．

学習

前スライドの◆を代入

$$Q(\theta,\theta_0)=\frac{1}{n}\sum_{i=1}^{n}\sum_{\vec z} \underbrace{q_{\theta_0}}_{\text{fix}}(\vec z|\vec x_i)\log q_\theta(\vec x,\vec z)$$

x の箇所に x_i を次々いれる．

exp に直接 log
この θ は動く

i のデータを与えた時の期待値．

$$=\frac{1}{n}\sum_{i=1}^{n}(\vec x_i W\langle\vec z\rangle_i+\vec b^T\vec x_i+\vec c^T\langle\vec z\rangle_i)-logZ$$

$$Z=\sum_{\vec x}\sum_{\vec z}\exp(\vec x^T W\vec z+\vec b^T\vec x+\vec c^T\vec z)$$

動く θ

$$\sum_{\vec z} r(\vec z)\log\frac{\overline{q_\theta}(\vec x,\vec z)}{\underline{r(\vec z)}}$$

rが持っているθは$q_\theta(\vec z|\vec x)$で固定されている．
動くθは$q_\theta(\vec x,\vec z)$のθだけ

これを強調して書いたのが Q 関数

$$\langle\ldots\rangle_i=\sum\vec z\,q_{\theta_0}(\vec z|\vec x)\times\ldots$$

q_{θ_0}とパラメータ(θ_0)は固定して条件付きのxは具体的に代入する．条件独立なので計算しやすい．

zの中に$\vec x$がいるのでそれがでる．

$$\langle\ldots\rangle=\sum_{\vec x}\sum_{\vec z}q_{\theta_0}(\vec x,\vec z)\times\ldots$$

学習する

$\frac{1}{Z}$の Zの微分$\rightarrow q_\theta$そのもの

$$\partial\vec b:\ \frac{1}{n}\sum_{i=1}^{n}\vec x_i-\underline{\sum_{\vec x}\sum_{\vec z}\vec x\,q_\theta(\vec x,\vec z)}=0$$

期待値($\langle\vec x\rangle$で書ける)

$$\therefore\langle\vec x\rangle=\frac{1}{n}\sum_{i=1}^{n}\vec x_i \quad\blacksquare$$

$$\partial W:\ \frac{1}{n}\sum_{i=1}^{n}\vec x_i\langle\vec z^T\rangle_i-\sum_{\vec x}\sum_{\vec z}\vec x\,\vec z^T q_\theta(\vec x,\vec z)=0$$

$$\therefore\langle\vec x,\vec z\rangle=\frac{1}{n}\sum_{i=1}^{n}\vec x_i\langle\vec z^T\rangle_i \quad\blacksquare$$

$$\partial\vec c:\ \frac{1}{n}\sum_{i=1}^{n}\langle\vec z\rangle_i-\sum_{\vec x}\sum_{\vec z}\vec z\,q_\theta(\vec x,\vec z)=0$$

$$\therefore\langle\vec z\rangle=\frac{1}{n}\sum_{i=1}^{n}\langle\vec z\rangle_i \quad\blacksquare$$

本当は
b= ～
c= ～
W= ～
と方程式を解きたいが無理なので微分する．
結果だけ使って勾配法やるが$\langle\ldots\rangle$, $\langle\ldots\rangle_i$ の期待値の計算が大変．

その期待値の計算をマルコフ連鎖モンテカルロ法でやる

マルコフ連鎖モンテカルロ法 (MCMC)

in BM

$$\langle\ldots\rangle=\frac{1}{\underline{Z}}\sum_{\vec{x}}\underbrace{\exp(-E_\theta(\vec{x}))}_{P(\vec{x})}\times\ldots$$

確率分布→コレにしたがった x を作るのが目標

in RBM

$$\langle\ldots\rangle=\frac{1}{\underline{Z}}\sum_{\vec{x}}\sum_{\vec{z}}\underbrace{\exp(-E_\theta(\vec{x},\vec{z}))}_{P(\vec{x},\vec{z})}\times\ldots$$

期待値計算

N_{sam} 個で平均．

$$\langle f(\vec{x})\rangle=\underline{\sum_{\vec{x}}P(\vec{x})}\cdot f(\vec{x})\approx\underbrace{\frac{1}{N_{sam}}\sum_{k=1}^{N_{sam}}f(x_k)}_{サンプル}$$

確率分布 $P(\vec{x})$ に従って N_{sam} 個 $\vec{x}$ を作る．

基本的には f(x) に代入するだけ

ただ，その N_{sam} 個作る方法を

MCMC 法という

$q_\theta(\vec{x}),q_\theta(\vec{x},\vec{z})$ に従う確率変数をサンプリングする．

MCMC 法ができないとどんなに良いボルツマンマシンを作っても犬の画像など欲しい画像は生成できない．(データがだせないので)

・マスタ方程式 (離散時間) [目標分布 $P(\vec{x})$ を作る]

$$P_{t+1}(\vec{x})=\sum_{\vec{x}'}\underbrace{w(\vec{x}|\vec{x}')}_{遷移確率}\cdot P_t(\vec{x})$$

少しずつ $P_t(\vec{x})$ を $\underline{P(\vec{x})}$ にしたい．

例 $q_\theta(\vec{x})$ の場合はボルツマンマシン

$q_\theta(\vec{x},\vec{z})$ の場合は制限ボルツマンマシン

条件 =1 になるようにする．(確率なので)

(確率保存) $\overline{\sum_{\vec{x}}P_{t+1}(\vec{x})}=\boxed{\sum_{\vec{x}}}\sum_{\vec{x}'}\underbrace{\boxed{w(\vec{x}|\vec{x}')}}_{=1}\cdot P_t(\vec{x})$

考え方：$\vec{x}$ があると $w(\vec{x}|\vec{x}')P_t(\vec{x}')$ の和が直接こない．($\vec{x}$ は変わる)

先に $\sum w(\vec{x}|\vec{x}')$ で $\vec{x}$ 和をとるとココは必ず 1 とする．

すると $\vec{x}'$ $P_t(\vec{x}')$ に対して $\vec{x}'$ は直接かかるのでコレも 1 になる．

遷移確率 $w(\vec{x}|\vec{x}')$ を $\vec{x}$ で和をとれば 1 になるようにすれば $P_t(\vec{x}')$ の和が 1 にできて，次の時刻 $P_{t+1}(\vec{x}')$ の和を 1 にする事ができる．

① $\sum_{\vec{x}} w(\vec{x}|\vec{x}')=1$

この条件を満たせば

目標分布への収束 $P_{t+1}(\vec{x})=P_t(\vec{x})=P(\vec{x})$ 定常分布にする．

この大きさはモデルによる .t が大きいと収束して (変わらないで) 欲しい．

② $P(\vec{x})=\sum_{\vec{x}'}w(\vec{x}|\vec{x}')P(\vec{x}')$

つりあい条件 (Balanced Condition)→ これが一番重要！

MCMC 法ではコレを満たしていれば後はなんでも良い．

世の中のモノをモデル化すれば遷移確率は勝手に 1 になるが，My モデルなのでこの条件を自分で満たすように設定する．

My モデルは何を作っても自由だが確率の条件などは自分で作る．

詳細つりあい

①, ② を満たせば何でも良いが「詳細つりあい」を使うと w が作りやすい．

十分条件

詳細つりあい (=> つりあい条件) (Detail Balanced Condition) DBC

$w(\vec{x}|\vec{x}')P(\vec{x}')=w(\vec{x}'|\vec{x})P(\vec{x})$　ただし DBC は破っても良い．

左辺，右辺で x と x’ が反転　あくまで①と②を守っていれば良い．

$$\Big[\sum_{\vec{x}'} w(\vec{x}|\vec{x}')P(\vec{x}') \overset{\text{DBC}}{=} \sum_{\vec{x}'} \underline{w(\vec{x}'|\vec{x})P(\vec{x})} \overset{\text{BC}}{=} P(\vec{x})\Big]$$

$=①$

これだと w が指定できるけど DBC は遅いのでやらないにこした事はない．

補足： DBC を破るとは

期待値を計算するとブレが発生する .DBC をやらない方が MCMC 法のブレは小さくなる .(漸近分散が改善)

DBC を使うのは遅いし精度もソコソコ . MCMC 法を普通にやると 1 日～ 2 日要する .

ただし , 破っても自由すぎるので，どうアルゴリズムにするかがフロンティア .

$$\begin{array}{ccccc} P(\vec{x}) & & w & & P(\vec{x}') \\ \begin{pmatrix} P_{11} \\ P_{10} \\ P_{01} \\ P_{00} \end{pmatrix} & = & \boxed{\begin{array}{l}\text{4×4}\\ \text{行列をかけたら}\\ \text{次の確率になる}\end{array}} & & \begin{pmatrix} P_{11}'' \\ P_{10}' \\ P_{01}' \\ P_{00} \end{pmatrix} \end{array}$$

1,1 になる確率

1 ↑ 0　1 ↑ 0

スピンの上 (1)，下 (0) 向きとすると，それぞれの起こり得る確率の例 .

各確率に遷移行列をかけると次の行列になってこれがつりあう .

量子コンピュータだとモンテカルロ法は不要 .

量子ビットの確率振幅が持てるので，

それにシュレディンガー方程式やハミルトニアン，

ユニタリ変換をかけるのは w をかけるのと同じ .

今は 2 の 100 乗とかの状態を保持できないので MCMC 法 .

例 .　BM: ボルツマンマシン

$$P(\vec{x})=\frac{1}{Z}\exp(-E_\theta(\vec{x}))$$

分子 - 分母でエネルギーの変化で考える .

$\vec{x}$　から$\vec{x}'$　に変化させた . その時に遷移確率はどうあるべきか

→ エネルギーの差

目標は遷移確率 w を求める .w の式にする .

BM の w(i,j)　比にすると Z が消せる .

$$\frac{w(\vec{x}|\vec{x}')}{w(\vec{x}'|\vec{x})}=\frac{P(\vec{x})}{P(\vec{x}'')}=\exp(-\underline{(E_\theta(\vec{x})-E_\theta(\vec{x}'))})$$

後　前

BM の w(j,i)

$\Delta E(\vec{x}|\vec{x}')$ = 熱

$E_{\theta'}-E_\theta$ = 仕事と例えれる

比がきまれば足し算した時の大きさが決まれば全部決まる .

(タテに足せば 1，タテに足せば 1.... なので)← 確率保存

○ 遷移 ＝ 提案 × 受理

$$w(\vec{x}|\vec{x}')=\underline{c(\vec{x}|\vec{x}')}\,A(\vec{x}|\vec{x}')$$

$\vec{x}' \to \vec{x}$ を作る　それは OK/NG

$\vec{x}'$　から $\vec{x}$ に遷移するのに確率的に起こさないといけない . スピンが 1 個あって，今は上を向いている．次の状態 x にするにはプラスが良いかマイナスが良いかどちらかを提案する必要がある .

その提案について , OK か NG の許可を出すか出さないか決める機能があり , その根拠は遷移確率の通り , 詳細つりあい , つりあい，確率保存を満たしている

かチェックして満たしている場合は OK を出す .

メトロポリス法

例 .

$C(\vec{x}|\vec{x}')$

今のスピン状態

$\vec{x}'=\{+1, \underline{-1}, +1\}$　　これが提案 .

$\vec{x}=\{+1, \underline{+1}, +1\}$　　1 個を -1 → +1 に変更 .

※2spin でやっても良いが計算が大変

1spin flip

確率は 1/N　N 個のどれかが 1spin flip

どれか 1 個を動かすので ...

x' から x になるのは N 個のスピンの中から 1 個選んで

変えるということなので，これができたら ...

なぜ MCMC 法が必要なのか ?

ボルツマンマシンなどで学習させるのに EM アルゴリズムがある .

そこで期待値の計算がでてくる . 期待値は確率的な平均 .

それを実現するために，実際に確率分布に従った乱数を作る .

この手続きで期待値が計算できる . その時に MCMC 法が必要になる .

■ メトロポリス法 (for DBC) ※DBC を満たすものなのでこれが全てなわけではない .

$$A(\vec{x}|\vec{x}')=min\left\{1, \frac{P(\vec{x})}{P(\vec{x}')}\frac{c(\vec{x}'|\vec{x})}{c(\vec{x}|\vec{x}')}\right\}$$

(本にはあたかも絶対的なように書いてあるが)

詳細つりあいを満たす一つの方法

本当に DBC を満たすか 1 個だけ確認

分子　$w(\vec{x}|\vec{x}')=\cancel{c(\vec{x}|\vec{x}')}\boxed{\frac{P(\vec{x})}{P(\vec{x}')}\frac{\cancel{c(\vec{x}'|\vec{x})}}{\cancel{c(\vec{x}|\vec{x}')}}}$　<1　小さい

分母・分子の比をとれなので

分母　$w(\vec{x}'|\vec{x})=\cancel{c(\vec{x}'|\vec{x})}\times\underline{1}$

逆数なので 1 より大きいので min(1, ?) より 1

※ c も P も逆なので逆数 (x' と x で)

$$=\frac{P(\vec{x})}{P(\vec{x}')}\quad \left(=\exp\left(-\underbrace{\left(E_\theta(\vec{x})-E_\theta(\vec{x}')\right)}_{}\right)\right)$$

$\Delta E(\vec{x}|\vec{x}')$

DBC

$C(\vec{x}'|\vec{x})$の補足：本来はこの逆の提案する確率も考えなくてはいけない .(逆に提案する場合はどういう確率でおこるのか)

1spin flip がなぜ選ばれるかは (1/N) だとすぐに分かるから .

下から上に行くのも 1/N の確率 .

※ 真ん中のスピンが選ばれるのは 1/3 で選ばれると +1 が -1 になるのですぐに計算ができる .

逆の確率を考えるのがメチャ難しいので , 何も考えずに 1spin flip をみんなやっている .

実装

$$c(\vec{x}|\vec{x}')=\frac{1}{N}(1\,spin\,flip)$$ ←1 個のスピンをかえる

乱数を持ってくる

1 未満

乱数 r∈[0, 1) で

c は 1/N と 1/N で約分されている .(メトロポリス法に従った受理確率)

エネルギーの変化をみる

BM の場合

変化を受け入れる

$\Delta E(\vec{x}|\vec{x}')>0$ → $r\leq\boxed{\dfrac{P(\vec{x})}{p(\vec{x}')}}$ $\left(=\exp\left(-\left(E_\theta(\vec{x})-E_\theta(\vec{x}')\right)\right)\right)$ $\Delta E(\vec{x}|\vec{x}')$ なら OK

→ otherwise .... なら NG

$\Delta E(\vec{x}|\vec{x}')<0$ → 100% .... なら OK

※ メトロポリス法だとこの場合は 1 を超えるので min の方 (=1)

上記なので

$$=>A(\vec{x}|\vec{x}')=min\left\{1,\frac{P(\vec{x})}{P(\vec{x}')}\boxed{\frac{c(\vec{x}|\vec{x}')}{c(\vec{x}'|\vec{x})}}\right\}$$ の実現

多くの場合は約分して見えなくなっているが裏にはコレがある .(1/N の部分)

これを繰り返しすると$P(\vec{x})$ が求められる .

シュミレーテッド・アニーリング (SA) 1983

$$P(\vec{x}) = \frac{1}{Z(\beta)} \exp(-\beta E_\theta(\vec{x}))$$ 統計力学のギブスボルツマン分布

$$\beta = \frac{1}{T}$$ (逆温度)

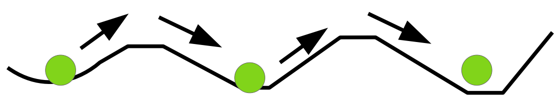
高温 T→∞ デタラメ
低温→ 0 エネルギー最小化

β は E_θ にかかるので β が小さいと E_θ が小さくなり 0 に近くなり全て受理される．

初期条件を変えて何度もくり返しをして最適解を見つける方法の代わりに考案．

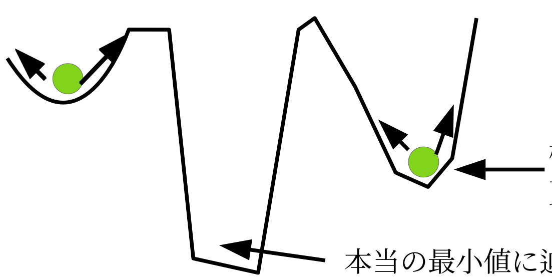
β=0 → β= 大 にして”最適化”問題をとく

(探索) (深堀)



β= 小 → 山を超えやすい

温度が高い時は勢いがあるのでエネルギーの障壁が低くなる．
(乗り越えられる)



β= 大 → 山を超えるのは大変

極小に捕まる．
更に低い場所がある可能性はあるが乗り超えられない．

本当の最小値に辿り着けない．

シュミレーテッド・テンパリング 1992

低温から温度を引き上げる (β を動かす).
低温のときに温度を戻して動きやすくして，また温度を下げていく .

SA は温度が低いと谷にはまり極小から抜け出せない .MCMC 法は連鎖で前の影響を受ける方法なので，壁が高いと超えられなくなる . かといってランダムだと意味がないのでこれらの解決を目指した方法の 1 つ .

β も確率変数にする

重み

$$P(\vec{x},\beta_k)=\frac{1}{Z_{all}}\exp(-\beta_k E_\theta(\vec{x})+g_k)$$

β_k も確率変数

分配関数 (x についての和)

$$Z_{all}=\sum_k \underline{\overline{\sum_{\vec{x}}\exp(-\beta_k E_\theta(\vec{x})+g_k)}}=\underline{\sum_k}(\beta_k)\cdot e^{g_k} \longrightarrow$$

いつものボルツマン分布

$$[P(\vec{x}|\beta_k)=\frac{P(\vec{x},\beta_k)}{\sum_{\vec{x}}P(\vec{x},\beta_k)}=\frac{1}{z(\beta_k)}\exp(-\beta_k E_\theta(\vec{x}))]$$

k も確率的なので和

k についての和を考える

$=z(\beta_k)$

β 固定→ x 動かす→ (止まったら)→β 動かす→ x を動かかす

分母 e^{g_k} が残る． 分子も e^{g_k}があるので約分 .

これが大事 .β の確率分布が何か調べる .(β が固定されれば MCMC 法を実施するだけ)

$$P(\beta_k)=\sum_{\vec{x}}P(\vec{x},\beta_k)=\frac{Z(\beta_k)}{Z_{all}}\underline{e^{g_k}}=\frac{1}{Z_{all}}$$

一様分布

β_k を固定して x について和をとると $z(\beta_k)$ が出てくる .

$\frac{1}{z(\beta_k)}$

e^{g_k} はパラメータで動かして学習させて $z(\beta_k)$ を約分できる値に状況によって変える .

β が動いて拾いあげる .

$$g_k=-\log Z(\beta_k)$$

g_k が分かれば自由エネルギー $(-\log Z(\beta_k))$ が分かる .

(log の ∑ なのでまともに計算するのは難しい .
というよりできない .)

計算の続き

g_k の計算　　β と β' の間隔はメッチャ小さく
|β − β'| << 1

※ 物理としてはエネルギーが低い状態を知りたい．
あるがまま秩序だった，低温の状態に比較的興味がある．
エネルギーが低いとは低温の時のこと．※ 個体→ (エネ +) → 液体→ (エネ +) → 気体
これを効率よく予言する方法が求められた．
エネルギーを損失関数に置き換えれば機械学習に応用できる．

テイラー

$$-\log Z(\beta')=-\log Z(\beta)+\frac{\partial}{\partial\beta}(-\log Z(\beta))|_{\beta=\beta}(\beta'-\beta)+\dots$$

前進差分と更新差分を両方計算

-)

$$-\log Z(\beta)=-\log Z(\beta')+\frac{\partial}{\partial\beta}(-\log Z(\beta))|_{\beta=\beta'}(\beta-\beta')+\dots$$

$$(\times\frac{1}{2})-\log Z(\beta')=-\log Z(\beta)+\frac{1}{2}[\langle E_\theta(\vec{x})\rangle_{\beta'}+\langle E_\theta(\vec{x})\rangle_{\beta}](\beta'-\beta)+\cancel{\dots}$$

打ち消し

移項する

なぜなら

$$\left(\frac{\partial}{\partial\beta}[-\log Z(\beta)]=\frac{1}{z(\beta)}\sum_{\vec{x}}\exp(-\beta E_\theta(\vec{x}))\,\underline{E_\theta(\vec{x})}=\underline{\langle E_\theta(\vec{x})\rangle_{\beta}}\right)$$

微分　　　　エネルギーの期待値

β という逆温度でサンプリングして MCMC 法をしているので既にサンプリングはしてある．
色々な β で MCMC 法をやっているので，それぞれの温度で計測したエネルギーをもってくれば計算できる．
β に関して期待値をとれなので .※β ごとにエネルギーの期待値

[ ] 平均をとる

$$\therefore g_{k+1}=g_k+\frac{1}{2}[\underline{\langle E_\theta(\vec{x})\rangle_{\beta_{k+1}}}+\underline{\langle E_\theta(\vec{x})\rangle_{\beta_k}}](\underline{\beta_{k+1}-\beta_k})$$

MCMC in β_{k+1}　　MCMC in β_k　　隣との温度差
※ 量子アニーリングでいうと横磁場

平均なので，エネルギーと逆温度の差をとって足しまくる．

$$w(\vec{x},\beta|\vec{x},\beta')=\underline{c(\vec{x},\beta|\vec{x},\beta')}\,\underline{A(\vec{x},\beta|\vec{x},\beta')}$$

$\vec{x}$ は固定

β^' から β を提案　　それは OK or NG

例：

$c(\vec{x},\beta|\vec{x},\beta')$　DBC を満たす $A(\vec{x},\beta|\vec{x},\beta')=\min\left\{1,\frac{P(\vec{x},\beta)\,\cancel{c(\vec{x},\beta'|\vec{x},\beta)}}{P(\vec{x},\beta')\,\cancel{c(\vec{x},\beta|\vec{x},\beta')}}\right\}$

$$\beta_{K-1}\xleftarrow{\frac{1}{2}}\beta_K\xrightarrow{\frac{1}{2}}\beta_{K+1}$$

ココは計算ずみ

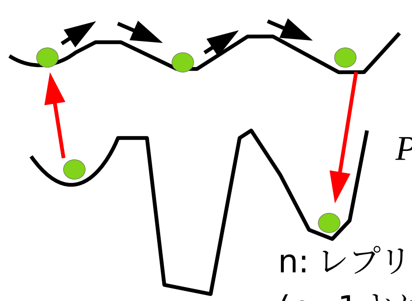
$$\frac{P(\vec{x},\beta_{k+1})}{P(\vec{x},\beta_k)}=\exp(-\beta_{k+1}E_\theta(\vec{x})+\beta_k E_\theta(\vec{x})+\overline{g_{k+1}-g_k})$$

現在　　平均

$$\fallingdotseq \exp(-(\beta_{k+1}-\beta_k))(\overline{E_\theta(\vec{x})}-\overline{\frac{1}{2}(\langle E_\theta(\vec{x})\rangle_{\beta_{k+1}}+\langle E_\theta(\vec{x})\rangle_{\beta_k})})$$

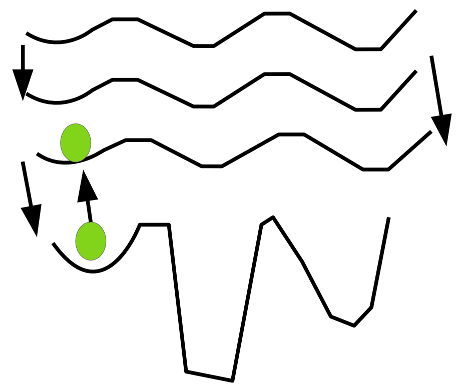
交換モンテカルロ法 (EMC) S.T の反省 1996 日本人

g_k の推定がしくるとSTは破たんするのでその反省とした方法



β を動かすのではなくて , x を異なる β にハメる .
低温だと乗り越えられないので高温にして乗り越えさせまた低温に戻す .
実際は異なる温度を沢山用意する (32 個くらい)
メッチャ低温とメッチャ高温

あまり多くてもよくない



n: レプリカ数
(n=1 とは限らない)

$P_{\beta_1}(x_1)\cdot P_{\beta_2}(x_2)\ldots.P_{\beta_n}(x_n)$

温度それぞれにスピンを用意 (n 個)

$$P(\vec{x}_1,\vec{x}_2,\ldots.,\vec{x}_n)=\frac{1}{Z_{all}}\exp(-\beta_1 E(\vec{x}_1)-\beta_2 E(\vec{x}_2)\ldots..\beta_n E(\vec{x}_n))$$

メトロポリスヘイスティングを採用した最大の理由は Z_{all} の分配関数が計算できないので , 比をとれば約分されて Z_{all} が消え入るから .

例： 1 番は低温 ,2 番は高温 , 1 番はぜんぜん動けない .
その時に温度交換して動いてないやつは高温に行って , 動いてたのは低温にいく .

実際にはどうやるかというと

$\beta_1,\beta_2,\ldots\beta_n$ を用意して

A ①↔② ③ ↔ ④ ⑤ と交互に入れ替える .

B ②↔③ ④↔⑤ 入れ替える .

1 と 2 を入れ替えるか提案して受理 or 却下 . 次のタイミングで 2 と 3, 4 と 5.

$\beta_{¿}$ を計算して x_1' になる

$\beta_{¿}$ を計算して x_2' になる

それぞれ MCMC 法

ある程度のところで温度を交換して X_2' を初期条件でβ_2で計算する .

今回は提案確率は必ず 1 番と 2 番を入れ替えるので 100%.
それが受理されるかはメトロポリスヘイスティングに任せる

$$w(\vec{x}_2,\vec{x}_1|\vec{x}_1,\vec{x}_2)=\underbrace{c(\vec{x}_2,\vec{x}_1|\vec{x}_1,\vec{x}_2)}_{=1(100\%)}\times A(\vec{x}_2,\vec{x}_1|\vec{x}_1,\vec{x}_2)$$

β_1 β_2 β_1 β_2 β の変更前と後

$$A(\vec{x}_2,\vec{x}_1|\vec{x}_1,\vec{x}_2)=min\left\{1,\frac{c(\vec{x}_1,\vec{x}_2|\vec{x}_2,\vec{x}_1)}{c(\vec{x}_2,\vec{x}_1|\vec{x}_1,\vec{x}_2)}\frac{P_{\beta_1}(\vec{x}_2)P_{\beta_2}(\vec{x}_1)}{P_{\beta_1}(\vec{x}_1)P_{\beta_2}(\vec{x}_2)}\right\}$$

before 提案が 1 なので
w はいれかえになる

今回は =1

ココが重要

=

$$\exp(-\beta_1 E(\vec{x}_2)-\beta_2 E(\vec{x}_1)+\beta_1 E(\vec{x}_1)+\beta_2 E(\vec{x}_2))$$

$$=\exp(-(\beta_1-\beta_2)(E(\vec{x}_2)-E(\vec{x}_1)))$$

Jarzynski 等式 (ジャルジンスキ)

仕事 (W)　　ΔF: 自由エネルギーの差 (ヘルムホルツの自由エネルギー)

$$\langle e^{-\beta w}\rangle_{0\to T}=e^{-\beta\Delta F}\left(=\frac{Z_t}{Z_0}\right)$$ 分配関数の比が計算できる

時刻 0→T に動かす

遷移確率 (仕事の w とは別)

$$\frac{1}{Z_0}\exp(-\beta E_0(\vec{x}))$$

$$\langle\ldots\rangle_{0\to T}=\sum_{\{\vec{x}_t\}} w_{T-1}(\vec{x}_T|\vec{x}_{T-1})w(\vec{x}_{T-1}|\vec{x}_{T-2})\ldots \underline{w_1}(\vec{x}_2|\vec{x}_1)\,\underline{w_0}(\vec{x}_1|\vec{x}_0)\,\underset{=}{P_0(\vec{x}_0)}$$

定常分布 (時刻 0)

t によってパラメータ (w) をかえる .　$P_1(\vec{x})$ を目標　$P_0(\vec{x})$ を目標

$$\left[\sum_{\vec{x}'} w_t(\vec{x}|\vec{x}')P_t(\vec{x}')=P_t(\vec{x})\right]$$ つりあい条件

DBC 不要 . エネルギーが t' に保存

$$P_t(\vec{x})=\frac{1}{Z_t}\exp(-\beta E_t(\vec{x}))$$

これがあると何がわかるかというと

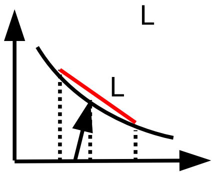
熱力学第 2 法則

$$\underline{\langle e^{-\beta w}\rangle_{0\to T}}\geq e^{-\beta\langle w\rangle_{0\to T}}$$ Jensen の不等式

L

$$\therefore\langle e^{-\beta\Delta F}\rangle\geq e^{-\beta\langle w\rangle_{0\to T}}$$

$$\langle w\rangle\geq\Delta F$$ log してマイナス除去



R

L と R だと R は必ず小さくなる .

★ 証明

$$\langle e^{-\beta w}\rangle_{0\to T}=\sum_{\{\vec{x}_t\}}\underline{e^{-\beta w}}\left(\prod_{t=1}^{T} w_{t-1}(\vec{x}_t|\vec{x}_{t-1})\right)P_0(\vec{x})$$

仕事はエネルギーの変化

$$\delta w_t=\underline{E_t}(\vec{x}_t)-\underline{E_{t-1}}(\vec{x}_t)$$

$$e^{-\beta w}=\prod_{t=1}^{T}\exp(-\beta\delta w_t)$$

パラメータ由来の変化 = 仕事

$$E_t(\vec{x}_t)-E_{t-1}(\vec{x}_t)$$

変数由来の変化 = 熱

同じエネルギーの変化だが熱と
仕事で明確に区別されている .

P に①と②の計算をしてやると次の時刻になる .

$$=\sum_{\{\vec{x}_t\}}\prod_{t=1}^{T}\left(e^{-\beta\delta w_T}w_{t-1}(\vec{x}_t|\vec{x}_{t-1})\right)P_0(\vec{x})$$

①
②

① $$\sum_{\vec{x}_0} w_0(\vec{x}_1|\vec{x}_0)P_0(\vec{x}_0)=P_0(\vec{x}_1)$$ つりあい条件

定義より　　δw の t=1 の時を考える .(いったん)

② $$e^{-\beta\langle E_1(\vec{x}_1)-E_0(\vec{x}_1)\rangle}\cdot \underline{P_0(\vec{x}_1)}=\frac{1}{Z_0}\exp(-\beta E_1(\vec{x}_1))$$

$$\frac{1}{Z_0}e^{-\beta E_0(\vec{x}_1)}$$

$$=\frac{Z_1}{Z_0}\cdot\frac{1}{Z_1}\exp(-\beta E_1(\vec{x}_1))=\underline{\frac{Z_1}{Z_0}P_1(\vec{x}_1)}$$

次の時刻の
定常分布

③ 繰り返すと

$$\sum_{\vec{x}_t}\frac{\cancel{Z_1}}{\underline{Z_0}}\frac{\cancel{Z_2}}{\cancel{Z_1}}\cdots\frac{\overline{Z_T}}{\cancel{Z_{T-1}}}\underline{P_t(\vec{x}_T)}=\frac{Z_T}{Z_0}\left(=e^{-\beta\Delta F}\right)$$

全部足すと
1 なので

Population Annealing(ポピュレーションアニーリング)

アニーリングはゆっくるやる必要があるが早くアニーリングをしたい .
Jarzynski は速さについては関係なので使うことができる .

情報 $E_\theta(\vec{x})$ が 0 になる

物理 $-\beta E_\theta(\vec{x})$ は β→0 になる

$P_0(\vec{x}_0)$ に対して $w_0(\vec{x}_1|\vec{x}_0)$ でないとつりあわない

$e^{-\beta\delta w_t}$ の意義

$\sum \vec{x}_1$ $\sum \vec{x}_0$ 定常分布

$$\langle\ldots\rangle_{0\to T}=\sum_{\{\vec{x}\}} w_{T-1}(\vec{x}_T|\vec{x}_{T-1})\ldots\times w_1(\vec{x}_2|\vec{x}_1)\times w_0(\vec{x}_1|\vec{x}_0)P_0(\vec{x})$$

つりあい $P_0(\vec{x}_1)$

なるのはコレが
同じ場合だけなので

$\neq\ P_1(\vec{x}_2)$ つりあいにならない .

$P_0(\vec{x}_0)$ は $w_1(\vec{x}_2|\vec{x}_1)w_0(\vec{x}_1|\vec{x}_0)$
のパラメータが違う .
私は今 P_0 なので上の遷移確率はつりあわない .
ゆっくり時間をかければ収束するが 1 回や 2 回では無理 .

$[P_{t-1}(\vec{x}_{t-1})$ とつりあい$]$

$$\sum w_{t-1}(\vec{x}_t|\vec{x}_{t-1})P_{t-2}(\vec{x}_{t-1})\neq P_{t-1}(\vec{x}_t)$$

$$e^{-\beta(E_{t-1}(\vec{x}_{t-1})-E_{t-2}(\vec{x}_{t-1}))}\overset{\text{足すと}}{\Rightarrow} P_{t-1}(\vec{x}_t t-1)\text{になる.}$$

エネルギーの差を入れて足すと定常分布にする事ができる .

$e^{-\beta\delta w_t}$ で定常分布の補正 (はやくても MCMC ができるように)

やってはダメな事 (今までの MCMC 法)

ギブスボルツマン分布

$$\vec{x}_0\rightarrow\vec{x}_1\rightarrow\vec{x}_2\rightarrow\ldots\rightarrow\vec{x}_t$$

MCMC

※ 早すぎると T の時に求めたい分布になっていない .

サンプルする (足し算)

$$\underset{P_1(\vec{x}_1)}{e^{-\beta\delta w_0}}\times\underset{P_2(\vec{x}_2)}{e^{-\beta\delta w_1}}\quad\underset{P_2(\vec{x}_t)}{e^{-\beta\delta w_{t-1}}}\longrightarrow\frac{Z_T}{Z_0}$$

ではどうするか ?
E を計算して指数関数
の肩にのせる .

ただしこのまま計算してはダメ .
メチャ小さい値や大きい値がある時に , そのかけ算なので 0 に潰れてしまったりする .
小 × 小 × 小 ... 小 = ほぼ 0

計算

このように計算する

それぞれ MCMC

$e^{-\beta(E_1(\vec{x}_1)-E_0(\vec{x}_1))}$ 大きい数字がかかる．具体的には前の時刻のエネルギーを使って実際に仕事を計算する．

$e^{-\beta\delta w_0}$　　$e^{-\beta\delta w_1}$

$$\{\vec{x}_0^{(1)}, \frac{1}{M}\} \to \{\vec{x}_1^{(1)}, \frac{1}{M}\} \to \{\vec{x}_1^{(1)}, 大\} \to \{\vec{x}_1^{(1)}, \frac{大}{2}\} \to \ldots \to \{\vec{x}_2^{(1)}, 中\} \to \ldots$$

リサンプリング

$$\{\vec{x}_1^{(1)}, \frac{大}{2}\} \to \ldots \to \{\vec{x}_2^{(1)}, 小\}$$

X

コピー

サンプルによってはxになる．
大，中，小様々な値になり，となりのスピンでは小さくなったりする．

$$\{\vec{x}_0^{(2)}, \frac{1}{M}\} \to \{\vec{x}_1^{(2)}, \frac{1}{M}\} \to \{\vec{x}_1^{(2)}, 小\}$$

X

小は基準を設けて無視する

具体的には M は 1000 〜 2000 個で並列処理

⋮

x をコピーして重要度を半分にした．
2 × 1/2 で同じなので元の大の影響は保たれている．

$$\{\vec{x}_0^{(M)}, \frac{1}{M}\} \to \{\vec{x}_1^{(M)}, \frac{1}{M}\} \to \{\vec{x}_1^{(M)}, 中\}$$

そのまま

$$\{\vec{x}_1^{(M)}, 中\} \longrightarrow \longrightarrow \{\vec{x}_2^{(M)}, 大\}$$

リサンプリング

平均すると $\langle e^{-\beta w}\rangle_{0\to 1}$ の計算

$e^{-\beta w}$ で重みづけされているので，$P_1(\vec{x}_1)$ になる．

平均すると $\langle e^{-\beta w}\rangle$ の計算 ← $P_2(\vec{x}_2)$

ただし刻み幅が小さくないとダメ
※ 飛びがあると重みの誤差が大きくなるので
ただ MCMC 法をやるよりも補正の効果が効くので短い時間でできる．

サンプル n 個について足し算をして 1/M をすると，分配関数の計算が各時刻できる．

短い時間で MCMC 法をやり，重みを補正するだけで，
各時刻の並行分布を求める事ができる．

■ メリット

低温から高温まで全てのデータを一気にとることができる．
(shift + Ent で 2 日ほどで全ての温度帯の MCMC 法が並行分布のサンプルリングができたデータが揃う．

Not MCMC 法

x というデータが散らばっている．これは何かの確率分布に従っているという仮定が今までの仮定．

それに非常に近い P_x があった.(データの分布). そのデータの分布に合った My モデルを作りたい．

パラメータ θ を動かしてデータに Fit させたいが，その際に自分で q_θ を作らないといけないので MCMC 法を利用してきた．

しかし MCMC 法は時間がかかるので使いたくない．

MCMC 法は時間をかけてコントラスト・ダイバージェンシ法や制限ボルツマンマシン法とか手をかえ品をかえ何とか BM が現実的に学習できるように工夫してきた．

MCMC 法を使った手順は BM をうまくデータに合わせてその途中途中は試し撃ちをして期待値を計算，データの期待値が合っていない場合はパラメータを直して MCMC 法をまたして〜を繰り返してきた．

これは手間なので MCMC 法をやらずに BM をやる方法が検討されてきた．

最小確率流法 (MCMC をやらない BM) 2011

google の社員？

定常分布←変化させようと思ったけど変化しない .BM では My モデルの行き着く先 .

$P(\vec{x})=\sum w(\vec{x}|\vec{x}')P(\vec{x})$

in BM= $q_\theta(\vec{x})$　　$q_\theta(\vec{x})$

$w_\theta(\vec{x}|\vec{x}')$ で動かないモノ .(定常分布)

$q_\theta(\vec{x})$ ではなくて本当に欲しいのは θ

確率分布が欲しいと錯覚している本当に欲しいのは θ.
パラメータがあればそのデータを再現できる .
(→ 何枚でも好きな画像を生成できる)

★ 目的

$$q_\theta(\vec{x})\approx P_0(\vec{x})=\frac{1}{n}\sum_{i=1}^{n}\delta(\vec{x}-\vec{x}_i)$$

確率流　※ θ がどこにあるかというと $w_\theta(\vec{x}|\vec{x}')$ にあるのでココだけ注目していれば良い .

定常分布は w_θ で動かない . w_θ と q_θ のパラメータ θ が一致していれば定常で変わらない .

$$P_1(\vec{x})=\sum_{\vec{x}'} w_\theta(\vec{x}|\vec{x}')P_0(\vec{x}')$$

$= q_\theta(\vec{x})$

動かすのは θ

※ $q_\theta(\vec{x})$を使うと MCMC 法が必要なので使わない .$q_\theta(\vec{x})\approx P_0(\vec{x})$ なので w_θの θ を少し動かして変わらなければ , $W_\theta P_0$ が定常になる . なので経験分布 P_0 を使う .

ただし $P_1(\vec{x})\approx P_0(\vec{x})$ の条件がある (キョリが 0 になれば良い).

$$D_{KL}(P_0(\vec{x})|P_1(\vec{x}))=\sum_{\vec{x}} P_0(\vec{x})\log\frac{P_0(\vec{x})}{P_1(\vec{x})}$$

ズレ

$$=-\sum_{\vec{x}} P_0(\vec{x})\log\frac{P_1(\vec{x})}{P_0(\vec{x})}$$

P_0 と P_1 は同じになって欲しいので比は1になって欲しい .

$$\frac{P_0(\vec{x})+P_1(\vec{x})-P_0(\vec{x})}{P_0(\vec{x})}$$

$$P_1(\vec{x})-P_0(\vec{x})=\sum_{\vec{x}'} w_\theta(\vec{x}|\vec{x}')P_0(\vec{x}')-\sum_{\vec{x}'}\delta(\vec{x}-\vec{x}')P_0(\vec{x}')$$

$$=\sum_{\vec{x}'}(w_\theta(\vec{x}|\vec{x}')-\delta(\vec{x}-\vec{x}'))P_0(\vec{x}')\equiv J_\theta(\vec{x})$$

経験分布　データからの確率流

x' については和をとったので変数は x のみ
データから確率分布をかえて漏れ出すようぃ漏れ出す箇所を計算 . それでもでなければ完璧で My 遷移確率はデータ分布に Fit している .

P_0 はxにx_iを入れた時だけ$\frac{1}{n}$を返す.★目的の式

$$=-\sum_{\vec{x}} P_0(\vec{x})\log\left(1+\frac{J_\theta(\vec{x})}{P_0(\vec{x})}\right)$$

KL の最小化は
コレの最小化

$$=-\frac{1}{n}\sum_{i=1}^{n}\log(1+nJ_\theta(\vec{x}_i))\approx-\sum_{i=1}^{n}J_\theta(\vec{x}_i)$$

データ　テイラー 1 次

実際に見た犬の
データのみ代入
して和をとれ .

続き

★ J_θの和　　遷移確率の和は 1　　x=x' で 1

$$\underbrace{\sum_{\vec{x}}}_{} J_\theta(\vec{x}) = \sum_{\vec{x}} \left(\sum_{\vec{x}'} \underbrace{w_\theta(\vec{x}|\vec{x}')}_{=1} - \underbrace{\delta(\vec{x}-\vec{x}')}_{=1} \right) P_0(\vec{x}') = 0$$

データ + ~~データ~~

データに無い部分

全ての犬猫画像やその他画像が全てのデータはない .

データ　　データでない (持っていない)

$$\therefore \sum_{i=1}^{n} J_\theta(\vec{x}_i) = -\sum_{i\neq\theta} J_\theta(\vec{x}_i)$$

符号を変えれば =

$$\text{よって} D_{kl}(P_0(\vec{x})|P_1(\vec{x})) = \sum_{j\notin D} J_\theta(\vec{x}_j)$$

コレを代わりに最小化

期待値計算なし !

データからデータ外への流れ

初期分布としてデータの経験分布を与えた .
そこから遷移確率でもれる確率分布について考える .
それをできるだけなくす .
それにより P_0 と P_1 が同じになる .

$$\text{元は} D_{KL}(P_0(\vec{x})|P_1(\vec{x})) = -\sum_{i=1}^{n} J_\theta(\vec{x}_i)$$

いざやると J_θ とは何 $P_0(\vec{x})$ は確率経験平均なので実際にあった x をブチ込む (代入) する .
遷移確率は何でも良い (メトロポリスでもその他) 受理確率 ,
提案確率が書かれたモノなら何でも自由に定義する .
そしたらそれにもとづいて計算する .
これは MCMC 法と比べて時間的に早い .2011 年で割と最近のイケてる手法 .

★MPF learning　遷移確率をどう決めるか

エネルギーの変化

$$w_\theta(\vec{x}_j|\vec{x}_i) = g_{ij} \exp\left(\frac{1}{2}\left(E_\theta(\vec{x}_i) - E_\theta(\vec{x}_j)\right)\right)$$

i→j の遷移を許す (+1), 許さない (-1)

$$\vec{x}_i = \{+1, -1, +1\}$$

許す　$g_{ij} = 1 (1\, spin\, flip)$

$$\vec{x}_j = \{+1, +1, +1\}$$

$$g_{ij} = 0$$

$$\vec{x}_k = \{-1, -1, +1\}$$

※1spin flip なら許すという
ルールなので 2spin の場合は 0

DBC 確認

$$\frac{w_\theta(\vec{x}_j|\vec{x}_i)}{w_\theta(\vec{x}_i|\vec{x}_j)} = \exp(-E_\theta(\vec{x}_j) + E_\theta(\vec{x}_i)) = \frac{q_\theta(\vec{x}_j)}{q_\theta(\vec{x}_i)}$$ OK

$$D_{KL}(P_0|P_1) = \sum_{j\notin D} \sum_{j\in D} [w_\theta(\vec{x}_j|\vec{x}_i) - \cancel{\delta(\vec{x}_j - \vec{x}_i)}]$$

対角は使わない .i はデータにある .j はデータにないので x_i と x_j が同じになる事はない .

$\vec{x}_i$ から $1\, spin\, flip$ して得られるものだけ足す

$$= \sum_{j\notin D} \sum_{j\in D} g_{ij} \exp\left(\frac{1}{2}\left(E_\theta(\vec{x}_i) - E_\theta(\vec{x}_j)\right)\right)$$

$$\frac{\partial D_{kl}}{\partial\theta} = \frac{1}{2} \sum_{j\notin D} \sum_{j\in D} g_{ij} \left(\frac{\partial E_\theta(\vec{x}_i)}{\partial\theta} - \frac{\partial E_\theta(\vec{x}_j)}{\partial\theta}\right) \exp(\ldots)$$

データとその周辺だけで勾配法 .
(MCMC はなし)

再び制限ボルツマンマシン

★ ギブスサンプリング

$$P_t+1(\vec{x})=\sum_{\vec{x}'} w(\vec{x}|\vec{x}')P_t(\vec{x})$$ DBC を満たす w を使う
メトロ～など

$w(\vec{x}|\vec{x}')$ 例： 1 spin flip → $x_{'i}$ は固定

i ではない i 以外固定

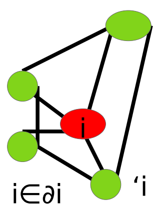
$\vec{x}'=\{+1,-1,+1\}$

i だけ flip 他は固定

$\vec{x}=\{+1,+1,+1\}$

例 :BM $P(\vec{x})=\frac{1}{Z}\exp(-E_\theta(\vec{x}))$

$$E_\theta(\vec{x})=\sum_{\langle ij\rangle} J_{ij}x_i x_j+\sum_{i=1}^{N} h_i x_i$$



i∈∂i

i の周りでつながっているの意

$$=x_i(\underbrace{\sum_{j\in\partial i} J_{ij}x_j+h_i}_{\lambda_i})+\sum_{k\neq i} h_k x_k+\sum_{\langle k,l\rangle} J_{kl}x_k x_l$$

i の周りの (∂) i 以外 i が全く関与していない外側

$J_{ij}\times x_i\times x_j$

全体の結合確率に対して， x_i 以外の確率を割り算してやる．
i 以外が固定化された条件確率が得られる．
条件付き確率は与えられるものの不確定性がないものになっているので，その確率で割れば得られる．
これをやるのが「ギブスサンプリング」(遷移確率の作り方の一つ)

$$w(\vec{x}|\vec{x}')=P(\vec{x}_i|\vec{x}_{'i})=\frac{P(\vec{x})}{P(\vec{x}_{'i})}$$

i 以外は固定した時の i が flip する確率が欲しい．
i 以外は条件として与えてその上での i が変わる確率 (つまり条件付き確率)

$$=\frac{P(\vec{x})}{\sum_{x_i} P(\vec{x})}$$

物理では熱浴法

$P(\vec{x})$に対してx_iだけで和をとればそれ以外のモノは残っている(分母)

$$P(x_i|\vec{x}_{'i})=\frac{\exp(x_i\lambda_i+\cancel{\sum_{k\neq i} h_k x_k+\sum J_{kl}x_k x_l})}{\sum_{x_i}\exp(x_i\lambda_i+\cancel{\sum_{k\neq i} h_k x_k+\sum J_{kl}x_k x_l})}=\frac{\exp(\lambda_i,x_i)}{2\cosh(\lambda_i)}$$

和を取るのは x_i だけなので約分出来る．

x_i が +1 の時は確率 (λ) で
-1 の時は確率 (-λ)
足せば 1(100%) になる．

+1 と -1 の足し算なので

$$P(x_i=+1|\vec{x}_{'i})=\frac{e^{\lambda_i}}{2\cosh(\lambda_i)}\equiv P_i$$ i の spin がプラスになる確率

$$P(x_i=+1|\vec{x}_{'i})=\frac{e^{-\lambda_i}}{2\cosh(\lambda_i)}\equiv 1-P_i$$

r∈(0, 1] で (一様乱数で)
$r<P_i$ なら $x_i=+1$
$otherwise\ x_i=-1$

詳細のつり合いを満たしているか確認

$$\frac{w(\vec{x}|\vec{x}')}{w(\vec{x}'|\vec{x})}=\frac{P(\vec{x})}{\cancel{\sum_{x_i} P(\vec{x})}}\frac{\cancel{\sum_{x_i} P(\vec{x}')}}{P(\vec{x}')}=\frac{P(\vec{x})}{P(\vec{x}')}$$

$\vec{x}_{'i}=\vec{x}'_{'i}$ DBC_OK

i 以外のスピンは変わらない．
なので動くスピンだけで和をとると約分される．

ブロック化サンプリングと RBM(制限 BM)

$$q_\theta(\vec{x},\vec{z})=\frac{1}{Z}\exp\left(\underline{\vec{x}^T w\vec{z}+\vec{b}^T\vec{x}+\vec{c}^T\vec{z}}\right)$$

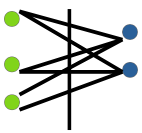
可視変数 (z は定数，固定)

$$E_\theta(\vec{x},\vec{z})=\vec{x}^T\left(\underbrace{w\vec{z}+\vec{b}}_{\vec{\lambda}}\right)+\vec{c}^T\vec{z}=\sum_{i=1}^{N}\lambda_i x_i+\vec{c}^T\vec{z}$$

or

$$=\vec{z}^T\left(\underbrace{w\vec{x}+\vec{c}}_{\vec{v}}\right)+\vec{b}^T\vec{x}=\sum_{j=1}^{Nh}v_j z_j+\vec{b}^T\vec{x}$$

隠れ変数
(x は定数，固定)

$\vec{x}$  $\vec{z}$

X と z を分ける .RBM ならできる .
x を固定して z をサンプリング .
z を固定して x をサンプリング .
独立なら可能 .

$$q_\theta(\vec{x}|\vec{z})=\frac{q_\theta(\vec{x},\vec{z})}{\underline{\sum_{\vec{x}}q_\theta(\vec{x},\vec{z})}}=\underline{\prod_{i=1}^{N}\frac{\exp(\lambda_i x_i)}{2\cosh(\lambda_i)}}$$

指数関数の和なので積

条件付きなので割る $q_\theta(\vec{z})$

ベクトル更新可能
(同じような式なら)
→ 並列化で更新ができる

$$q_\theta(\vec{z}|\vec{x})=\frac{q_\theta(\vec{x},\vec{z})}{\underline{\sum_{\vec{z}}q_\theta(\vec{x},\vec{z})}}=\underline{\prod_{j=1}^{Nh}\frac{\exp(v_i x_i)}{2\cosh(v_i)}}$$

$q_\theta(\vec{x})$

条件付き独立

★RBM の更新則

$$log q_\theta(\vec{x})=\log\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z}|\vec{x})}$$

勝手に乱数を作って足す

$\sum_{\vec{z}}r(\vec{z})\times...$ とりあえず仮で$\vec{z}$を作る

$$log q_\theta(\vec{x})=\sum_{\vec{z}}r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{q_\theta(\vec{z}|\vec{x})}$$

KL を使う

$$D_{KL}(P\|q)=\sum_x P(\vec{x})\log\left(\frac{P(\vec{x})}{q(\vec{x})}\right)\geq 0$$

$$=\underline{\sum_{\vec{z}}r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}}-\underline{\sum_{\vec{z}}r(\vec{z})\log\frac{q_\theta(\vec{z}|\vec{x})}{r(\vec{z})}}$$

z と x の分布なので使いにくい

こっちを KL で使う。
両方 z の分布

$$=\sum_{\vec{z}}r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}+D_{KL}(r(\vec{z})\|q_\theta(\vec{z}|\vec{x}))\geq 0$$

$$\geqq\underline{\sum_{\vec{z}}r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}}$$ 変分下界

$r(\vec{z})$とq_θを合わせるようなものを目指す.

$$Q(\theta,\theta_0)=\frac{1}{n}\sum_{i=1}^{n}\sum_{\vec{z}}\underline{q_{\theta_0}(\vec{z}|\vec{x}_i)}\log q_\theta(\vec{x}_i\vec{z})$$

KL は r と q がイコールなら距離 0
前の θ の値を使う .← rとq_θを合わせる意味でθ_0

$$=\frac{1}{n}\sum_{i=1}^{n}\left(\vec{x}^T w\underline{\langle\vec{z}\rangle_i}+\vec{b}^T\vec{x}_i+\vec{c}^T\langle\vec{z}\rangle_i\right)-logZ$$

x_iが与えられた時のz(x_iは実データ)

$$\frac{\partial Q}{\partial\vec{b}}=\frac{1}{n}\sum_{i=1}^{n}\vec{x}_i-\underline{\langle\vec{x}\rangle}$$

$$\frac{\partial Q}{\partial\vec{c}}=\frac{1}{n}\sum_{i=1}^{n}\langle\vec{z}\rangle_i-\underline{\langle\vec{z}\rangle}$$

$$\frac{\partial Q}{\partial\vec{w}}=\frac{1}{n}\sum_{i=1}^{n}\vec{x}_i\langle\vec{z}\rangle_i^T-\underline{\langle\vec{x}\vec{z}^T\rangle}$$

log が来る分 (MCMC で求める)

$\langle...\rangle=\sum_{\vec{x}\vec{z}}q_\theta(\vec{x},\vec{z})\times....$ が必要

$\langle\vec{x}\rangle=\frac{1}{n}\sum_{i=1}^{n}\vec{x}_i$ は解けないので勾配法
$w,\vec{b},\vec{c}$ について

この式に $w,\vec{b},\vec{c}$ はでて来ないのでパラメータ (w, b, c) を解けない .

RBM のブロックサンプリング =CD 法

$q_\theta(\vec{x},\vec{z})$ を定常分布でもちたい

$$P_{t+1}(\vec{x},\vec{z})=\sum_{\vec{x}\vec{z}} w(\vec{x},\vec{z}|\vec{x}',\vec{z}')P_t(\vec{x}',\vec{z}')$$

$$=q_\theta(\vec{x},\vec{z})q_\theta(\vec{z}|\vec{x}')$$

z はよく分からないのでデータ由来の x' を作って，
x' → z → z → x で
データ由来の x で考える

データ分布を初期分布として MCMC をやる .
x が与えられて , z を作って MCMC を 1step やれの意味 .

1step

$$\sum_{i=1}^{n}\langle\ldots\rangle_i=\frac{1}{Z}\sum_i\sum_z q_\theta(\vec{z}|\vec{x}_i)\times\ldots=\frac{1}{Z}\sum_{\vec{z}\vec{x}} q_\theta(\vec{z}|\vec{x})P_0(\vec{x})$$

初期分布

前ページの◆　z についての期待値

z から x を作る

$$\langle\ldots\rangle=\frac{1}{Z}\sum_{\vec{x}\vec{z}} q_\theta(\vec{x}|\vec{z})\times\ldots=\frac{1}{Z}\sum_{\vec{z}\vec{x}}\left(\sum_{\{\vec{x}\vec{z}\}} q_\theta(\vec{x}_T|\vec{z}_T)q_\theta(\vec{z}_T|x_{T-1})q_\theta(x_{T-1}|z_{T-1})\ldots\right)\ldots$$

1step

$P_0(\vec{x})$ から $1\,step$ と $Tstep$ 分のズレ ⇒ 勾配

ズレはよくない . なので T=1 でも (CD-1)→C- ディーワン
ズレは何でも良いので 1 回でもよい . 十分な事が多い .(多くても 10 回くらいやれば十分)

■ 条件つき確率でいいのでは？

今までは生成モデルを作るのに確率分布が欲しいと思っていた . しかし， 確率分布ではなくてそのパラメータ変化が分かればよい . 変化は条件付き確率 . 今までの確率分布を求めるのはちょっともったいない ....

$\vec{x}_1,\vec{x}_2,\ldots\ldots\vec{x}_n \sim P(\vec{x})$ ← これが知りたい (いや， これになりたい !)

猫の画像はこういう分布ですって言うより， 猫の画像が早く出せた方が良い . 分布がわからなくてもパラメータが分かれば生成できる .

$$P(\vec{x})\approx q_\theta(\vec{x})$$

$=\sum_{\vec{z}} q_\theta(\vec{x}\,\vec{z})$　z の変数を増して拡張していこう . 隠れ変数 (昔はこの路線) を入れて難しくしよう .

(条件付き確率を考えた方が良い)

$=\sum_{\vec{z}} q_\theta(\vec{x}\,\vec{z})q_\theta(\vec{z})$　q_θ を求めてもよくわかない . 複雑な確率分布が分かっても理解できない .(理解， 操作性)

かっこ良い犬の画像を作るのにパラメータなんか設定できない . 複雑で分からない .

学ぶべきはココ　$=\sum_{\vec{z}} q_\theta(\vec{x}\,\vec{z})q_0(\vec{z})$　ガウス分布など q_0 は簡単な分布で条件付き確率の法を考える .(パラメータ =0 の既知の分かりやすいガウスみたいな分布でやる)

既知 (扱い， 理解しやすいモデルで fix!)　どんな確率分布でも MCMC 法をやれば変わっていくので， カッコ良い犬が作れる .

モデル生成の仕組み

■ 発想は簡単

画像にノイズを乗せまくって汚す．

すると最後はノイズしかなくなる (乱数)

逆にノイズから構造物を作るにはどう変形すれば良いか学習させる．

ただの乱数から構造物に変形させる上手い変換方法を学んでいるので色々な乱数からいくらでも絵が作れる．

そのガウスノイズを入れていく過程を確率過程というが , 将来ちょっとずつズレていく (確率的にノイズが混入されて).

予想からズレていくような現象も微分方程式で書く事ができる (確率微分方程式).

その解析結果を利用して，コンピュータに，こういう風に分析するとノイズから構造物，ある種の無秩序な状態から秩序のある状態に結びつけるための関係式を計算する事ができて，コンピュータはノイズと意味のある構造物の関係を捕まえて，それを利用して乱数から絵をつくる．

その途中の過程で，どのタイプの変換をすればよいか文章の言葉，キーワードと関連してくる．

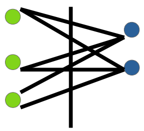
※ 猫と言ったらここをこういう風に変えるんだよなど

変分オートエンコーダー

$q_\Phi(\vec{z}|\vec{x})$ エンコード

（実データをよく分からない z に変換）

$\vec{x}$



$\vec{z}$

そのデータをつくるための出発点

よく分からない z から x に変換する時に q_θ

$q_\theta(\vec{x}|\vec{z})$ デコード

目的は$q_\theta(\vec{x})$がデータにFitするのを最大化したい.尤度最大化.

$q_\theta(\vec{x}|\vec{z})$ $q_0(\vec{z})$

分からない　コッチは分かる．

$$\log q_\theta(\vec{x})=\frac{q_\theta(\vec{x},\vec{z})}{q_\Phi(\vec{z}|\vec{x})}$$

条件付き確率で割り算

$\sum_{\vec{z}} r(\vec{z})\times\ldots$

$$\log q_\theta(\vec{x})=\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{q_\Phi(\vec{z}|\vec{x})}$$

やることはいつも一緒．(変分下界）適当に z を作って平均をとる．前と同じ方法．

なぜ KL かと言うと絶対に正 (≧0) なので下界を作れる .(r×log をみたら KL)

KL を使う

$$D_{KL}(P\|q)=\sum_{x} P(\vec{x})\log\left(\frac{P(\vec{x})}{q(\vec{x})}\right)\geq 0$$

$$=\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}-\sum_{\vec{z}} r(\vec{z})\log\frac{q_\Phi(\vec{z}|\vec{x})}{r(\vec{z})}$$

$$=q_\theta(\vec{x}|\vec{z})q_0(\vec{z})$$

$$=\sum_{\vec{z}} r(\vec{z})\log\frac{q_\theta(\vec{x},\vec{z})}{r(\vec{z})}+D_{KL}(r(\vec{z})\|q_\Phi(\vec{z}|\vec{x}))\geq 0\,(=0\text{ は }r=q_\Phi)$$

=0

$r=q_\Phi$ KLが最小になる時

$$=\sum_{\vec{z}} q_\Phi(\vec{z}|\vec{x})\log q_\theta(\vec{x}|\vec{z})+\sum_{\vec{z}} q_\Phi(\vec{z}|\vec{x})\log\frac{q_0(\vec{z})}{q_\Phi(\vec{z}|\vec{x})}$$

そしたらデコードできるか？

x から z を作る分布と適当に用意したガウス分布が一致するかどうかって事？

$$=\sum_{\vec{z}} q_\Phi(\vec{z}|\vec{x})\log q_\theta(\vec{x}|\vec{z})-D_{KL}(q_\Phi(\vec{z}|\vec{x})\|q_0(\vec{z}))$$

⇐ この式を最大化したい

ちゃんとデコードできる？（大） エンコードできる？（小） エンコードとデコードの良さを調べている．

思想のまとめ

あるガウス分布の空間があって，そこだったら偏差値的な数値で人の事をもしくは画像の良さを表現する事ができる．誰かが犬か猫のようなデータ x をはじき出した．それを z というモノに上手くエンコードできるか．そしてそれを再びデコードできるか．そのコスト関数を見ることによって θ と Φ が良いものを作ればどんな画像もガウス分布，またガウス分布からどんな画像も作れる．良い感じのデコーダが作れればガウス分布に乱数色々なサンプルが作れるので，色々な犬猫が無限に作れる .(乱数は無限個あるので). コレを上手く作れば絵も音も無限に作れる．例えばガウスなら平均的な所 (偏差値 50) なところから乱数を使えばよくある犬猫の画像だが，珍しい犬猫なら端の方の乱数を使えば良い．

気持ち．1 項目目 :x→z を作れますか？また z から x に戻せますか？

2 項目 :x が与えられたら z に変換できるか．どんなふうに変換．例えばガウス．

そしたらそれが近いかみる．近い方が良い．

続き

①$q_\Phi(\vec{z}|\vec{x})$して$q_\theta(\vec{x}|\vec{z})$で戻せるか？

②$q_\Phi(\vec{z}|\vec{x})$して$q_0(\vec{z})$にできるか？

② の計算

$q_0(\vec{z})=N(0,I)$ 理解しやすく操作性がある

$q_\Phi(\vec{z}|\vec{x})=N(\underline{\mu\Phi(\vec{x})},\sigma^2\underline{\Phi(\vec{x})})$ NN で作る． データ$\vec{x}$から$\vec{z}$を作る

(ニューラル〜)

損失関数に相当

z も μ もベクトルだがそれぞれの成分で計算ができるので 1 変数として進める．独立同分布で各成分が同じ事をやるので，一般化するなら分散共分散行列などにしてやればいいがめんどくさそうなので，暇な時にでもやってまずは実装しないと話にならない．

② $D_{KL}(q_\Phi(\vec{z}|\vec{x})\|q_0(\vec{z}))$

e の log なので log なし

$$=\int dz \frac{1}{\sqrt{(2\pi\sigma_\Phi^2)}}\exp\left(-\frac{1}{2\sigma_\Phi^2}(z-\mu_\Phi)^2\right)\left\{-\frac{1}{2\sigma_\Phi^2}(z-\mu_\Phi)^2-\frac{1}{2}\cancel{\log}\,2\pi\sigma_\Phi^2+\frac{1}{2}z^2+\frac{1}{2}\cancel{\log 2}\,\pi\right\}$$

$\langle\ldots\rangle_z$と書く

$Plog\frac{P}{q}$の部分　logP の部分　q_θのガウス分布

$\frac{1}{\sqrt{(2\pi\sigma_\Phi^2)}}$の$log$部分

ガウスの先頭箇所

確率分布をかけて積分しろ

→ 期待値をとれ

一見すると難しそうな式だが難しくはない

バラした

$$=\left\langle -\frac{1}{2\sigma_\Phi^2}(\underbrace{z^2}_{\sigma_\Phi^2}-2\mu_\Phi\underbrace{z}_{\mu_\Phi}+\mu_\Phi^2)+\frac{1}{2}\underbrace{z^2}_{\sigma_\Phi^2}-\frac{1}{2}\log\sigma_\Phi\right\rangle_z$$

$\langle\ldots\rangle_z$の計算(ガウスだからできる)

ガウス分布の分散は z^2 の期待値

ガウスの平均はμ_Φに集中しているのでzの期待値はμ_Φ

$$=\frac{1}{2}(\mu_\Phi^2+\sigma_\Phi^2-1-\log\sigma_\Phi^2)\quad\blacksquare$$

コレが損失関数なので最小化する

ガウス分布以外の分布だと難しそうなので使う気にならない．

(ガウスでもまぁまぁ面倒くさいが…)

μ_Φとσ_Φの学習ができる.(エンコーダ)

例

期待値μ_Φで分散$\sigma_\Phi{}^2$のガウス分布なので

$logq_\theta(\vec{x}|\vec{z})$をどうするか？$\Leftarrow\vec{z}$は$q_\Phi(\vec{z}|\vec{x})$でサンプリングできる．

このｚが入力された時に与えられたｘと整合するか？という事

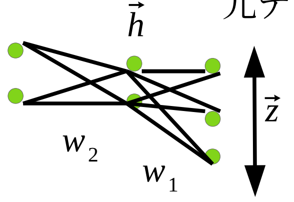
教師データ

$\vec{z}=\mu_\Phi(\vec{x})+\sigma_\Phi(\vec{x})\cdot\varepsilon$　　ε ~ N(0, 1)⇐ 簡単

元データはxなのでμ_Φ,σ_Φは決まる.(xがあるので)⇐これを元の犬猫に戻るようにNNを作れば良い

入力データ

$\vec{x}\approx\vec{y}$



これが$\vec{x}$と似ていれば良い

$\vec{h}=f_1(w_1\vec{z})$　　$\vec{x}\in\{0,1\}^N$ ※例でxは$0\vee1$の白黒画像として

$\vec{y}=sigmoid(w_2,\vec{h})\Leftarrow$もちろんもっと深くても良い

最後は 0→?%, 1→?% の確率が欲しいので

$logq_\theta(\vec{x}|\vec{z})=\sum_{k=1}^{N}[x_k\log y_k+(1-x_k)\log(1-y_k)]$　（交差エントロピー）

これで z から y が出て y と x の損失関数がでるので誤差逆伝播法をする事ができる．

$\vec{z}$が入力のNNに対して$backprop$(誤差逆〜)デコーダ

デコーダとエンコーダが分離できるので各々並列で学習ができる．

$\vec{Z}$は$\vec{x}$から作る人工データ

変分自己オートエンコーダでぐぐる．→　実装可能．

階層変分オートエンコーダ

前は NN を複雑にしたがこっちも複雑にできるのでは？

$$\vec{x} \overset{q_\Phi(\vec{z}|\vec{x})}{\rightleftarrows} \vec{z}$$

$$q_\theta(\vec{x},\vec{z})=q_\theta(\vec{x}|\vec{z})\underline{q_0(\vec{z})}$$

生成元

NN は複雑だがコッチはガウスなので簡単にできるのでは？
(NN を深くすると時間もかかり GPU 代もかかるので補助として)

1~T までである

前の時刻ででてきたヤツから次の時刻をつくる条件確率をかける

$$q_\Phi(\vec{z}_{1:T}|\vec{x})=\prod_{t=1}^{n}\underline{q_\Phi(\vec{z}_t|\vec{z}_{t-1})}(\vec{z}_0=\vec{x})$$

汚せば良いので適当で良い

$\mu_\Phi,\sigma_\Phi,\varepsilon$で汚せば良い

$$\vec{x} \rightleftarrows \vec{z}_1 \rightleftarrows \vec{z}_2 \cdots\cdots \rightleftarrows \vec{z}_T$$

ここに来る時はほぼノイズだらけ

ホワイトノイズから構造物を生成

$$q_\theta(\vec{x},\vec{z}_{1:T})=\prod_{t=1}^{T}\underline{q_\theta(\vec{z}_{t-1}|\vec{z}_t)}\,\underline{q_0(\vec{z}_t)}$$

生成元

逆戻しをするコレが大切 !(コッチを工夫)

$\mu_\Phi,\sigma_\Phi{}^2$が多くなる

NN は大変→ NN を並行きるか？不変性を求めるなら必要だが生成だけなら真面目に考えない .

■ 変分拡散モデル ［$\mu_\Phi,\sigma_\Phi{}^2$をシンプルに］　階層変分自己オートエンコーダの簡単 Ver(NN を使えば複雑なモノができるが NN を捨てる)

$$q_\Phi(\vec{z}_T|\vec{z}_{t-1})=N\left(\sqrt{dt}\,z_{t-1},(1-\alpha t)I\right)$$

※ 前のデータを期待値にして，その周りに分散させる .
α は後で考える．分散は (1-αt)I の単位行列 .

$$\therefore \vec{z}_t=\sqrt{dt}\,z_{t-1}+\sqrt{1-\alpha t}\,\vec{\varepsilon}_{t-1}$$

どんどん汚して$q_0(\vec{z}_T)=N(0,I)$に！

時間発展もシンプルに (最初から最後までの関係)

◆2　z_{t-1}の箇所に$t-2$で代入

$$\vec{z}_t=\sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\,\vec{z}_{t-2}+\sqrt{1-\alpha_{t-1}}\,\vec{\varepsilon}_{t-2}\right)+\sqrt{1-\alpha_t}\,\vec{\varepsilon}_{t-1}$$

$$=\sqrt{\alpha_t\alpha_{t-1}}\,\vec{z}_{t-2}+\underline{\sqrt{\alpha_t(1-\alpha_{t-1})}\,\vec{\varepsilon}_{t-2}+\sqrt{1-\alpha_t}\,\vec{\varepsilon}_{t-1}}$$

合成！ $\sqrt{a^2+b^2}$

$$=\sqrt{\alpha_t\alpha_{t-1}}\,\vec{z}_{t-2}+\sqrt{\alpha_t(1-\alpha_{t-1})+(1-\alpha_t)}\cdot\vec{\varepsilon}_{t-2}{}'$$

α_tをかければ良いことに気づく

$$=\sqrt{\alpha_t\alpha_{t-1}}\,\vec{z}_{t-2}+\sqrt{1-\alpha_t\alpha_{t-1}}\cdot\vec{\varepsilon}_{t-2}{}'$$

$$=\ldots=\sqrt{\prod_{k=1}^{t}\alpha_k}\,\underset{\vec{x}}{\underline{\vec{Z}_0}}+\sqrt{1-\prod_{k=1}^{t}\alpha_k}\cdot\vec{\varepsilon}_0{}'$$

$$\therefore q_\Phi(\vec{z}_t|\underset{\vec{x}}{\underline{\vec{z}_0}})=N\left(\sqrt{\prod_{k=1}^{t}\alpha_k}\,\underset{\vec{x}}{\underline{\vec{z}_0}},\left(1-\sqrt{\prod_{k=1}^{t}\alpha_k}\right)I\right)$$

補足

物理ではランジュバンダイナミクス

ビーカの中に「花粉」割とでかいコロイドを置いていたら水分子がランダムにぶつかっててた．その様子をみて，この世には決定的な力だけでなく確率的な力でモデリングする事象があると見つかった．その時の運動方程式がランダムフォースが通常位置エネルギー以外にかかる．ブラウン運動と同じ前の座標に応じて，$\sqrt{\alpha_t}\vec{z_{t-1}}$という力がかかりつつも更にガウスランダム変数でぶつけられるから同じ．それを方程式にしたのをランジュバン方程式．これを時刻 T までやれというプログラムは物理では既にある．

変分下界 (尤度最大)

NN はリーサルウェポンなのでどこで使うか .
GPU に金払えばいくらでもできるが金額を考えると使い時が重要 .
工夫できる箇所は工夫する .

$$log q_\theta(\vec{x}) = \log \frac{q_\theta(\vec{x}, \vec{z}_{1:T})}{q_\Phi(\vec{z}_{1:T}|\vec{x})}$$

$\times \sum_{\vec{z}_{1:T}} r(\vec{z}_{1:T})$ ※左辺は空振りでそのまま ($r(\vec{z})$ は任意の確率分布)

$$log q_\theta(\vec{x}) = \sum_{\vec{z}_{1:T}} r(\vec{z}_{1:T}) \log \frac{q_\theta(\vec{x}, \vec{z}_{1:T})}{q_\Phi(\vec{z}_{1:T}|\vec{x})}$$

KL を使う

$$D_{KL}(P\|q) = \sum_x P(\vec{x}) \log\left(\frac{P(\vec{x})}{q(\vec{x})}\right) \geq 0$$

$$= \sum_{\vec{z}_{1:T}} r(\vec{z}_{1:T}) \log \frac{q_\theta(\vec{x}, \vec{z}_{1:T})}{r(\vec{z}_{1:T})} - \sum_{\vec{z}_{1:T}} r(\vec{z}_{1:T}) \log \frac{q_\Phi(\vec{z}_{1:T}|\vec{x})}{r(\vec{z}_{1:T})}$$

$$= \sum_{\vec{z}_{1:T}} r(\vec{z}_{1:T}) \log \frac{q_\theta(\vec{x}, \vec{z}_{1:T})}{r(\vec{z}_{1:T})} + D_{KL}(r(\vec{z}_{1:T}) \| q_\Phi(\vec{z}_{1:T}|\vec{x})) \geq 0$$

$r = q_\Phi$

コレはさっき作った . 簡単 .

コレが工夫必要

$$= \sum_{\vec{z}_{1:T}} q_\Phi(\vec{z}_{1:T}|\vec{x}) \log \frac{q_\theta(\vec{x}, \vec{z}_{1:T})}{q_\Phi(\vec{z}_{1:T}|\vec{x})}$$

$$= \prod_{t=1}^{T} q_\theta(\vec{z}_{t-1}|z_t)\, q_0(\vec{z}_t)$$

戻す

簡単なガウス分布

$$= \prod_{t=1}^{T} q_\Phi(\vec{z}_t|z_{t-1})$$

$(\vec{z}_0 = \vec{x})$

$\langle \ldots \rangle_{\vec{z}_{1:T}|\vec{x}}$

これは簡単 .
ノイズを乗せてつくるだけ .

続き

logの中身が積なので足し算

$$=\left\langle \sum_{t=1}^{T}\log\frac{q_\theta(\vec{z_{t-1}}|\vec{z}_t)}{q_\Phi(\vec{z}_t|\vec{z_{t-1}})}+\log q_0(\vec{z}_T)\right\rangle_{\vec{z_{1:T}}|\vec{x}}$$

※ややこしいのはt-1→tの流れと，t→t-1の流れが混在している．
q_Φの方が簡単なのでq_Φを逆にする

比べると”t-1”↔”t”が混ざっている．※どちらかを逆の確率分布を持ってきて比較しやすく→ベイズの定理？

$\vec{x}$　$\vec{x}$　$|\vec{x}$　$|\vec{x}$　が可能か？

$$\underbrace{q_\Phi(\vec{z_{t-1}}|\vec{z}_t)}_{\text{逆}}=\frac{q_\Phi(\vec{z_{t-1}},\vec{z}_t)}{q_\Phi(\vec{z}_t)}=\frac{q_\Phi(\vec{z}_t|\vec{z_{t-1}})q_\Phi(\vec{z_{t-1}})}{q_\Phi(\vec{z}_t)}$$

$|\vec{x}$　$|\vec{x}$　が可能か？

$\cancel{q_\Phi(\vec{z}_t)は”分”}$

$q_\Phi(\vec{z}_t|\vec{x})$なら”分”　こっちは今までやってきたので
これで置き換えたい．

$q_\Phi(\vec{z}_t|\vec{z_{t-1}})=q_\Phi(\vec{z}_t|\vec{z_{t-1}},\vec{x})$(マルコフ性)

初期のxが何であれ前の$\vec{z_{t-1}}$が決まれば$\vec{z}_t$なのでxを入れようがいれまいが同じ．

t=2以降は反転した
確率で書いている．

数ページ前の◆2

$$=\left\langle \sum_{t=2}^{T}\log\frac{q_\theta(\vec{z_{t-1}}|\vec{z}_t)}{q_\Phi(\vec{z_{t-1}}|\vec{z}_t)}\cdot\boxed{\frac{q_\Phi(\vec{z_{t-1}}|\vec{z}_t)}{q_\Phi(\vec{z}_t|\vec{z}_t)}}+\overbrace{\log\frac{q_\theta(\vec{z}_0|\vec{z}_1)}{q_\Phi(\vec{z}_1|\vec{z}_0)}}^{\text{t=1の部分}}+log q_0(\vec{z}_T)\right\rangle$$

$=\vec{z}_0$

消せる

比の確認

$$=\frac{q_\Phi(\vec{z}_1|\vec{z}_0)}{\cancel{q_\Phi(\vec{z}_2|\vec{z}_0)}}\cdot\frac{\cancel{q_\Phi(\vec{z}_2|\vec{z}_0)}}{\cancel{q_\Phi(\vec{z}_3|\vec{z}_0)}}\cdots\frac{\cancel{q_\Phi(\vec{z_{T-1}}|\vec{z}_0)}}{q_\Phi(\vec{z}_T|\vec{z}_0)}$$

←logの和は積なので積の形にすると最初と最後だけ残る

$$=\left\langle \sum_{t=2}^{T}\log\frac{q_\theta(\vec{z_{t-1}}|\vec{z}_t)}{q_\Phi(\vec{z_{t-1}}|\vec{z}_t)}+log q_\theta(\vec{x}|\vec{z}_1)+\log\frac{q_0(\vec{z}_T)}{q_\theta(\vec{z}_T|\vec{x})}\right\rangle_{\vec{z_{1:T}}|\vec{x}}$$

③Path(Process)の距離　①デコード
きれいなデータを別の表現で書く　③エンコード(x→z)

計算

①：$\sum_{\vec{z}_{1:T}} \prod_{t=1}^{T} q_\Phi(\vec{z}_t|Z_{t-1}) \log q_\theta(\vec{x}|\vec{z}_1)$　　①を$\langle \dots \rangle_{\vec{z}_{1:T}|\vec{x}}$で期待値の式にした

時刻 0 と 1 に相当するものは抜き出す．

$$= \sum_{\vec{z}_1 \cdot \vec{z}_{2:T}} \Big(\underbrace{\prod_{t=2}^{T} q_\Phi(\vec{z}_t|Z_{t-1})}_{=1}\Big) q_\Phi(\vec{z}_1|\vec{x}) \log q_\theta(\vec{x}|\vec{z}_1)$$

$$\because \sum_{\vec{x}} q_\Phi(\vec{z}|\vec{z}')=1\Big[\frac{\overbrace{\sum_{\vec{z}} q_\Phi(\vec{z},\vec{z}')}^{=q_\Phi(\vec{z}')}}{q_\Phi(\vec{z}')}=1\Big]$$

logの中身はq_θのxとz_1だけなのでz_2以降は関係ないからz_2以降は抜き出す.

$$= \sum_{\vec{z}_1} q_\Phi(\vec{z}_1|\vec{x}) \log q_\theta(\vec{x}|\vec{z}_1)$$ ■　　前の式とまったく同じ．VAE① 再び

②：$\sum_{\vec{z}_{1:T}} \prod_{t=1}^{T} q_\Phi(\vec{z}_t|Z_{t-1}) \log \frac{q_0(\vec{z}_T)}{q_\Phi(\vec{z}_T|\vec{x})} = \sum_{\vec{z}_T} q_\Phi(\vec{z}_T|\vec{x}) \log \frac{q_0(\vec{z}_T)}{q_\Phi(\vec{z}_T|\vec{x})}$

$$= \sum_{\vec{z}_T} q_\Phi(\vec{z}_T|\vec{x})$$

$$= -D_{KL}(q_\Phi(\vec{z}_T|\vec{x}) \| q_0(\vec{z}_T))$$

t-1 までの確率については全て和をとっておく．
t-1 までの全ての要素を足して Z_T だけ残す．
$\vec{z}_T$ の形にするため．1:T-1 は全て和をとっておく．

③ の計算

③ だけ違う．事項 T-1 から T のプロセスが続きている．その間で Φ と θ という別のパラメータでパラメタライズして，Φ の方は自身が強さ α でバコバコノイズを乗せる事に決めた .θ はまだ残している .2 つのパラメータができるだけ離れないようになっていないと復元しにくい．
例：全然違う遠くにいる変化をしたやつが Zt の時間になったのでガウス分布になりましたので戻って来いとなっても戻れない．つかず離れずにしないとダメ．常に一緒くらいが良い .(各時刻で等しくなるレベル)

計算続き

③：$\sum_{\vec{z}_{1:T}} \prod_{t=1}^{T} q_\Phi(\vec{z}_k|\vec{z}_{t-1}) \sum_t \log \frac{q_\theta(\vec{z}_{t-1}|z_t)}{q_\Phi(\vec{z}_{t-1}|\vec{z}_t)}$

積になっているのはバラす．この log の中には t-1 と t の事しか書いてないのでそれ以外の時刻はから回りしているはず．
なので t-1 と t の事と 1～(1-t) までの事で分ける．

q_Φをバラす

$$= \sum_t \sum_{\vec{z}_{1:T}} \left(\underbrace{\prod_{k=t}^{T} q_\Phi(\vec{z}_k|\vec{z}_{k-1})}_{=1} q_\Phi(z_t|z_{t-1}) \underbrace{\left(\prod_{k=1}^{t-1} q_\Phi(\vec{z}_T|\vec{z}_{T-1})\right)}_{q_\Phi(\vec{z}_{t-1}|\vec{z}_0)} \cdot \log \frac{q_\theta(\vec{z}_{t-1}|z_t)}{q_\Phi(\vec{z}_{t-1}|z_t)} \right)$$

KL情報量っぽいけどZ_tとz_{t-1}が逆になっている(ベイズの定理)

t-1 と t の時刻の z についてはまだ
和をとっていない．

時刻0から1,2,3とやっていくが，その途中の時刻z_1, Z_2, z_3と足し算しろというのが左の$\sum$

(起点と最後の t-1 のところは残っているので足し算しない)

$(z_t \sim z_{t-1}$はまだ$)$和をとっていない

$\vec{z}_0$(データ)

$$= \sum_{t=2}^{T} \sum_{\vec{z}_t : \vec{z}_{t-1}} \boxed{q_\Phi(\vec{z}_t \cdot \vec{z}_{t-1}|\vec{x})} \log \frac{q_\theta(\vec{z}_{t-1}|\vec{z}_t)}{\boxed{q_\Phi(\vec{z}_{t-1}|\vec{z}_t)}}$$

KL をつかう．

元 $q_\Phi(\vec{z}_t|\vec{z}_{t-1}) q_\Phi(\vec{z}_{t-1}|\vec{x})$

書き換え
られる

新 $\underline{q_\Phi(\vec{z}_{t-1}|\vec{z}_t)} q_\Phi(\vec{z}_t|\vec{x})$

xが決まってz_tとZ_{t-1}がどちらも確率的に決まるので言っている事は一緒

これらの距離を初期条件 x でバーンと z_t をノイズ乗せて作っておいて，その時に KL 情報量を測るというのがコスト関数として各時刻につく．

$$= -\sum_{t=2}^{T} \sum_{\vec{z}_t} q_\Phi(\vec{z}_t|\vec{x}) - D_{KL}(q_\Phi(\vec{z}_{t-1}|\vec{z}_t) \| q_\theta(\vec{z}_{t-1}|\vec{z}t))$$

計算続き 2

残った項

①　②

$$★：q_\Phi(\vec{z_{t-1}}|\vec{z_t},\vec{z_0})=\frac{\underline{q_\Phi(\vec{z_{t-2}}|z_t)}\,\underline{q_\Phi(\vec{z_{t-1}}|\vec{z_0})}}{\underline{q_\Phi(\vec{z_t}|\vec{z_0})}}$$

③

※ $Z_{t-1}|z_t$ の分布は知らないので Z_0 で条件付けした分布で考える．

↑ これは知っている．

z_t-1 に注目して平方完成をする．

$$\propto \exp\left(\underbrace{-\frac{1}{2(1-\alpha_t)}(z_t-\sqrt{\alpha_t}\,\underline{z_{t-1}})^2}_{①}\underbrace{-\frac{1}{2(1-\bar{\alpha_{t-1}})}(\underline{z_{t-1}}-\sqrt{\bar{\alpha_{t-1}}}\vec{x})^2}_{②}+\underbrace{\frac{1}{2(1-\bar{\alpha}_t)}(\vec{z}_t-\sqrt{\bar{\alpha}_t}\vec{x})^2}_{③}\right)$$

①

平方完成

$$=\dots\cdot\exp\left(-\underbrace{\frac{1-\bar{\alpha}_t}{2(1-\alpha_t)(1-\bar{\alpha_{t-1}})}}_{=\frac{1}{\sigma_{q(t)}^{\;2}}}\left(\vec{z_{t-1}}-\underbrace{\frac{\sqrt{\bar{\alpha}_t}(1-\alpha_t)+\sqrt{\bar{\alpha_{t-1}}}(1-\alpha_t)\vec{x}}{1-\bar{\alpha}_t}}_{=\mu_{q(\vec{z}_t,\vec{x})}}\right)^2\right)$$

分散→

(時間によって)

←t の時の状態と初期状態 (x) で決まる期待値

これにもとづいて Z_{t-1} を生成すれば逆戻しができるということ！

$q_\Phi(\vec{z_{t-1}}|\vec{z})$ は $\underbrace{平均\,\mu_{q(\vec{z}_t,\vec{x})}}_{\text{x 由来}},\underbrace{分散\,\sigma_{q(t)}^{\;2}}_{\text{x 不要}}$

2 つのガウス KL(でてきた KL の計算)

復元するガウス過程を知っているq_Φと実際にノイズまみれの画像をデコードするq_θの距離をできるだけ短くする．

NNでも何でも使っていいのでq_θのデコードを作る．

$$D_{KL}(\underbrace{q_\Phi(\vec{z_{t-1}}|\vec{z_t})}_{N(\mu_q,\sigma_q{}^2)}\|\underbrace{q_\theta(\vec{z_{t-1}}|z_t)}_{N(\mu_{\theta(\vec{z}_t)},\sigma_q{}^2)})$$

平均は今の状態を見て作る．
ノイズは分散を揃える．(面倒くさいので)

$$=\int dz_{t-1}\underbrace{\frac{1}{\sqrt{2\pi\sigma_q}}\exp(-\frac{1}{2\sigma_q{}^2}(\vec{z_{t-1}}-\vec{\mu}_q)^2)}_{}\times[-\frac{1}{2\sigma_q{}^2}(\vec{z_{t-1}}-\vec{\mu}_q)^2+\frac{1}{2\sigma_q{}^2}(z_{t-1}-\mu_\theta)^2]$$

$\vec{z_{t-1}}$は平均μ_q, 分散$\sigma_q{}^2$だと言っている※定数になる

結果μ_qとμ_θの差だけみれば良い

平均μ_qとμ_θを近づけるモデルにする．

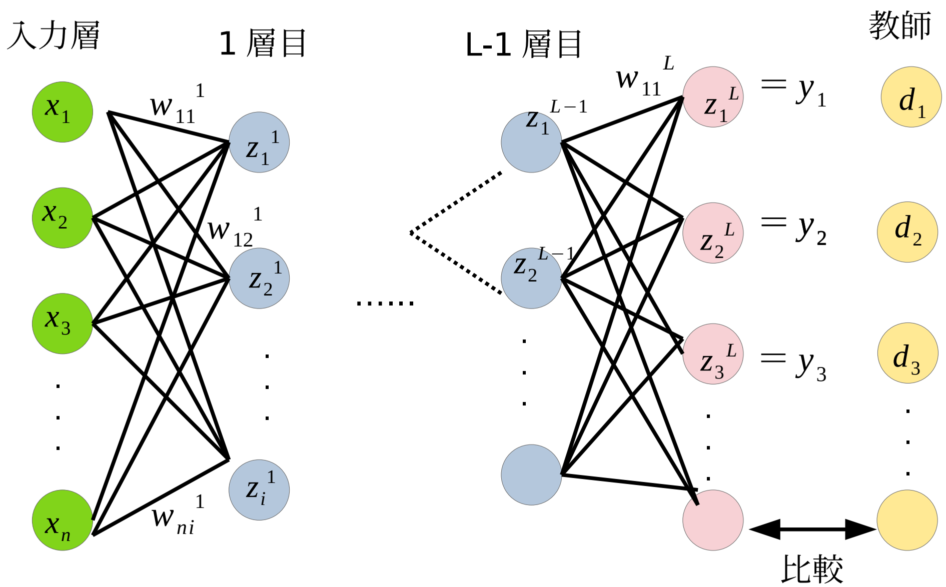
そうするとシンプルなモデルになる．もっと複雑な事もできるが ....
分散ズラすくらいなら対して変わらない事が経験上分かっている．

$\propto +(\vec{\mu}_q-\vec{\mu}_\theta)^2$ 結果として平均二乗誤差を最小化する．

なので平均二乗誤差を最小化する．
各段階を汚した，ちょっと汚した，もっと汚した , それぞれの二乗誤差を最小化するいつも NN をやれば良い．
それでできた , μ_θにもついた，z_t ガウスノイズから μ_θで徐々にキレイにしていくと，
元データに非常に近い確率分布が生成される．

arXiv.2208.11970　※2022 年
Understanding Diffusion Models: A Unified Perspective

補足：誤差逆伝播法



■ 誤差逆伝播法のこころ
教師データと合うように学習する際にパラメータを変更して調整する．
しかし，1 層目など最初の層のパラメータを変更すると，2 層，3 層 ...L 層など先の層全てに影響してしまうので複雑すぎて計算出来ない．
なのでゴール (最後の層) から逆に向かってパラメータを変更していく．
L-1 層の変更は L 層にしか影響しないので．

$$u_j^l = \underbrace{w_{0j}^l}_{\text{バイアス}} + \overbrace{w_{1j}^l z_1^{l-1} + w_{2j}^l Z_2^{l-1} \ldots}^{\text{線形和}}$$

※z に対する重み

1 つ前の重み付き線形和を u とおく

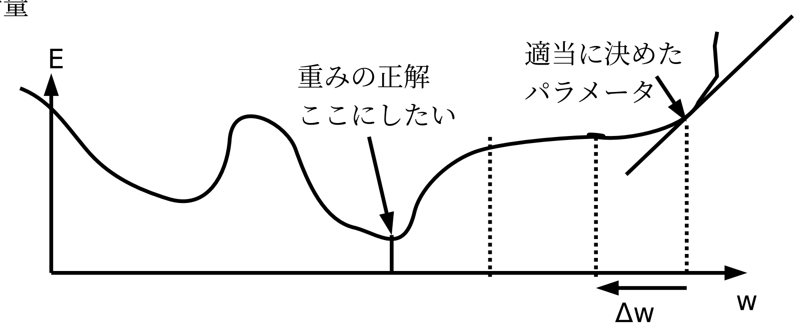
$z_j^l = f^l(u_j^l)$　活性化関数をかました値を次の層の値にする．

z と u の関係　u_j^l は z になる直前の値

※ 実際は同じ関数を使う事が多いが一般化した f^l にしておく

やりたいこと．誤差関数 (教師 -y) を最小化するように重みを最適化していく

更新量



重みの関数 (w) と誤差 (E)
ここでは重み 1 個の例

傾きを計算する．
正だったらもっと左に寄った方がいい．
Δw をどう決めるか？

$$\underline{\Delta w_{ij}^{\ l}} = -\eta \underline{\frac{\partial E}{\partial w_{ij}^{\ l}}}$$

傾き

コレが正ならマイナスに更新
負ならプラスに更新

η ：正の値でどの程度更新すれば良いのか制御する学習率

連鎖率

1 変数

y=f(u)
u=g(x)

y=f(g(x)) を分けて書いた形

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}$$

多変数

$$z(u_1, u_2, u_3, \ldots.)$$
$$u(x_1, x_2, x_3, \ldots.)$$

$$\frac{\partial z}{\partial x_k} = \frac{\partial z}{\partial u_1}\frac{\partial u_1}{\partial x_k} + \frac{\partial z}{\partial u_2}\frac{\partial u_2}{\partial x_k} + \ldots.. = \sum_i \frac{\partial z}{\partial u_i}\frac{\partial u_i}{\partial u_k}$$

2 変数

z=f(u, v)
u=g(x, y)
v=h(x, y)

z=f(g(x, y) , h(x, y))

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial u}\frac{\partial u}{\partial x} + \frac{\partial z}{\partial v}\frac{\partial v}{\partial x}$$

他の変数を固定して x を動かした時に , z はどれだけ変わるかなので x を動かすと u も v も変わる .

設定

$E=\frac{1}{2}\sum_i (y_1-d_1)^2$ 二乗誤差関数を使う．
1/2 は偏微分した際に消えるためにつける

動きが見やすいようにこの設定．
実際はこれでなくても良い．

$f^L(u)=u$ 恒等関数．最後の活性化関数だけ恒等関数を設定．

○ 出力層 ※ ゴールの側からスタートに向けて重みの更新を考える．

$$\frac{\partial E}{\partial w_{ij}^{L}}=\underbrace{\frac{\partial E}{\partial u_j^{L}}}_{=\delta_j^{L}}\underbrace{\frac{\partial u_j^{L}}{\partial w_{ij}^{L}}}_{=Z_i^{L-1}}$$

$$\overline{\partial u_j^{L}} = +\ldots+\underline{w_{ij}^{L} z_i^{L-1}}+w_{i+1\cdot j}^{L} z_{i+1}^{L-1}\ldots.$$

これがあるのはどこか一箇所だけ
w_{ij}^{L}で偏微分するから定数だけ落ちる．

これはw_{ij}^{L}ではない．

$$=\delta_i^{L} Z_i^{L-1}$$

w_{ij}^{L}

u_j^{L} | z_j^{L}

活性化関数で u→z

w_{ij}^{L}はu_j^{L}にかかっている．

w_{ij}^{L}を動かすと変化するのは
U_j^{L}だけなので，1変数みたいな書き方になっている．
他のuは動かない．

$$\delta_j^{L}=\frac{\partial E}{\partial u_j^{L}}=\frac{\overline{\partial E}}{\partial z_j^{L}}\frac{\partial z_j^{L}}{\underline{\partial u_j^{L}}}$$

$$=y_j \qquad =\frac{\partial u_j^{L}}{\partial u_j^{L}}=1$$

$$E=\frac{1}{2}\sum_i (y_i-d_i)^2=\frac{1}{2}\{\ldots+\underline{(y_j-d_j)^2}+\ldots\}\Rightarrow\frac{1}{2}\cdot 2\cdot(y_j-d_j)\cdot 1$$

y_jはここだけ

$=f^L(u_j^L)=u_j^L=z_j^L$ ※$f^L(u)$は恒等式なので

他の z は動かないので 1 変数の連鎖率みたいになっている

$=y_j-d_j$ ※これが伝播する

よって

$$\frac{\partial E}{\partial w_{ij}^{L}}=(y_j-d_j)\,z_i^{L-1}$$

○ 中間層

出力層の時と同じ

$$\frac{\partial E}{\partial w_{ij}^{l}}=\frac{\partial E}{\partial u_j^{l}}\frac{\partial u_j^{l}}{\partial w_{ij}^{l}}=\delta_j^l \underline{z_i^{l-1}}$$

これはニューラルネットワークに突っ込めば分かる値 (決まっている値) なのでδだけ考える .

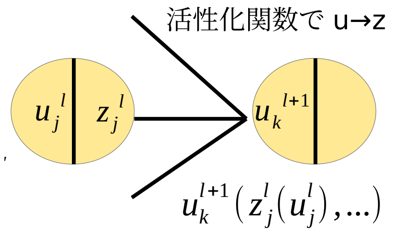
$$\delta_j^l=\frac{\partial E}{\partial u_j^l}=\sum_k \underbrace{\boxed{\frac{\partial E}{\partial u_k^{l+1}}}}_{=\delta_k^{l+1}}\frac{\partial u_k^{l+1}}{\partial u_j^l}$$

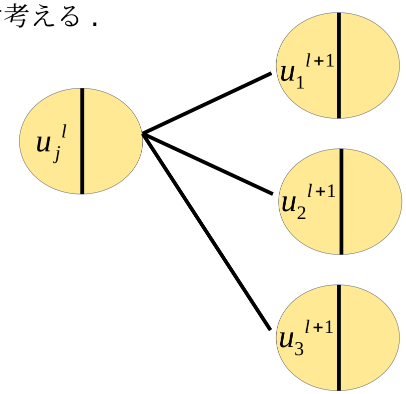
$$=\sum_k \delta_k^{l+1}\frac{\partial u_k^{l+1}}{\partial z_j^l}\frac{\partial z_j^l}{\partial u_j^l}$$

$=f^l(u_j^l)$

$=\{f^l(u_j^l)\}'$

$=..+...+w_{j\cdot k}^{l+1}z_j^l+....$





$E(u_1^{l+1}(u_j^l),u_2^{l+1}(u_j^l)....)$

u_j^l はu^{l+1}には$(u_1^{l+1},u_2^{l+2},...)$

全て入っているのでu_j^lを動かせば全て動くので，全ての変化量の和をとらなくてはいけないから∑ .

δ_j^lの値はδ_k^{l+1}の値から求められる.

(k=1, 2, 3...) l+1 の全ての (k)δ が分かれば δ_j^l が分かる .

$\delta^L \to \delta^{L-1} \to \delta^{L-2} \to ...\delta^1$ 出力層から追いかければ一層目のδ^1の値が分かる .

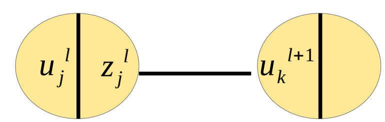
出力層で計算できている . $\delta^L=y_j-d_j$ なのでここから分かる .

誤差逆伝播法

$$=\sum_k \delta_k^{l+1}w_{jk}^{l+1}\{f^l(u_j^l)\}'$$

一つ前で使った重みを使えば計算できる .

u_j^l は 1 回データを投げて計算し終わったニューラルネットワークには必ず残っていいるので計算できる .



$u_k^{l+1}(z_j^l(u_j^l),...)$

u_j^lに活性化関数をかましたのがz_j^l

u_k^{l+1}はz_j^lを通してu_j^lの関数とみえる.

※$\frac{\partial u_k^{l+1}}{\partial u_j^l}$は∑なしで2個の偏微分におきかえができる.

u_j^lは1個しかないので

■ 誤差逆伝播法

1. 入力ベクトルx_nをネットワークに入れ，$u_j=\sum_i w_{ij} z_i$と$z_j=f(u_j)$を用いてネットワーク上を順伝播させすべての中間層と出力層の出力を求める．

2. $\delta_j=y_j-d_j$を用いて全ての出力層のδ_jを評価する．

3. $\delta_j=\sum_k \delta_k w_{jk}\{f(u_j)\}'$を用いて$\delta$を逆伝播させ，ネットワーク全ての中間層の$\delta_j$を得る．

4. $\frac{\partial E}{\partial w_{ij}}=\delta_j z_i$を用いて必要な微分を評価する．

参考

・ "High-Resolution Image Synthesis with Latent Diffusion Models", Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B., (CVPR'22)　 arXiv:2112.10752
Stable Diffusion のベースになった論文

・”Understanding Diffusion Models: A Unified Perspective”, Calvin Luo　arXiv:2208.11970
理論部分の元論文

・世界に衝撃を与えた画像生成 AI 「 Stable Diffusion 」を徹底解説！
とてもわかりやすく Stable Diffusion について解説してくれてます
https://qiita.com/omiita/items/ecf8d60466c50ae8295b

・情報科学としての数理情報学 2022 年　大関真之
理論部分は大関先生の講義で勉強した際に作成したノートをスライド化しました
https://youtube.com/playlist?list=PLsBJ3psEqyr-_9fWJJl_9bUaGEsS85vGx

・パターン認識と機械学習 (上) C.M. ビショップ著
→ 第 5 章ニューラルネットワーク (誤差逆伝播法) 箇所を参考にしています .

・絶対に理解させる誤差逆伝播法【深層学習】 - 予備校のノリで学ぶ「大学の数学・物理」
はじパタやビショップ本の誤差逆伝播法は最初はイメージが付きにくいので助かりました .
誤差逆伝播法の箇所はヨビノリさんの動画を参考にしています .
https://youtu.be/0itH0iDO8BE